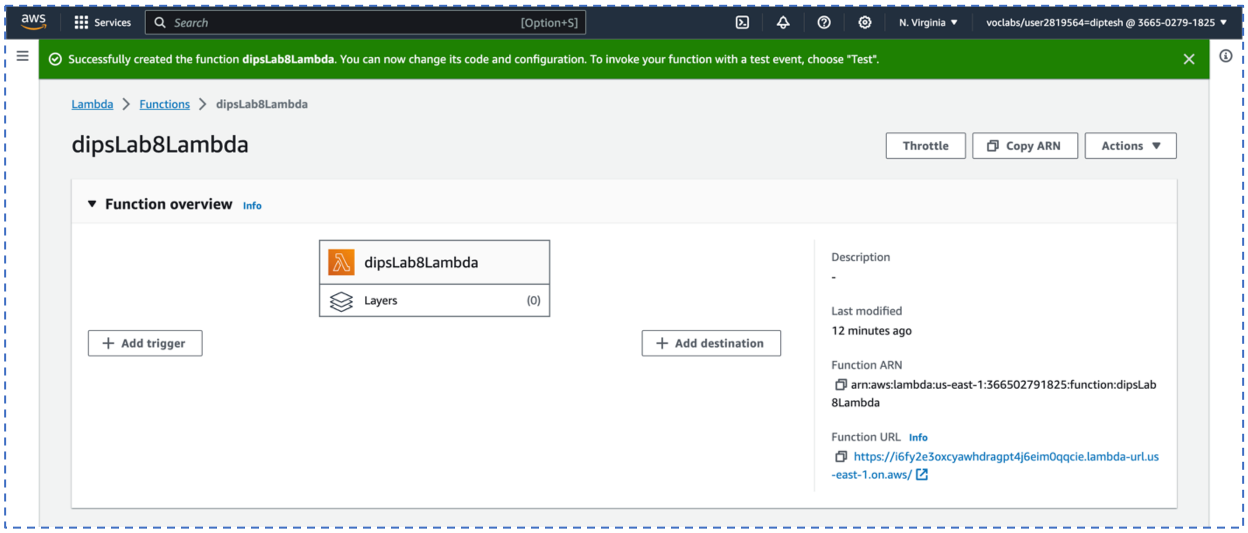
MAHARISHI INTERNATIONAL UNIVERSITY
Formerly Maharishi University of Management

DynamoDB – CRUD operation using Lambda function URL.

- Lambda Function URL created.


Successfully created the function **dipsLab8Lambda**. You can now change its code and configuration. To invoke your function with a test event, choose "Test".

Lambda > Functions > dipsLab8Lambda

dipsLab8Lambda

Function overview

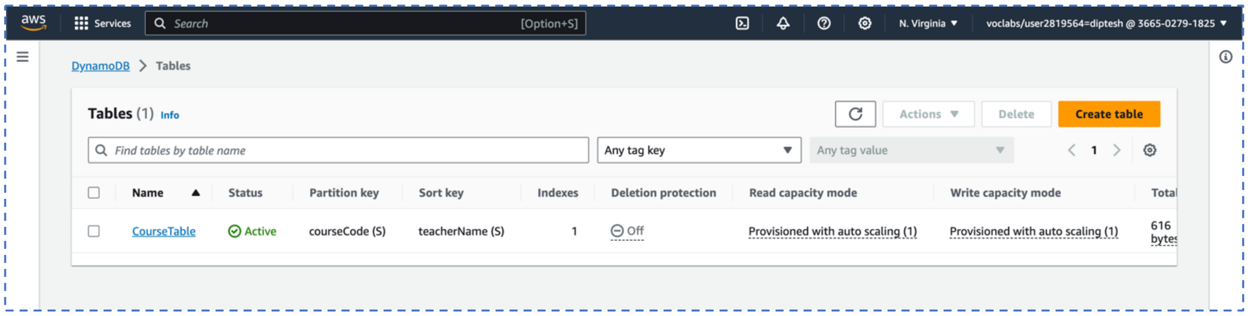
Description
-

Last modified
12 minutes ago

Function ARN
arn:aws:lambda:us-east-1:366502791825:function:dipsLab8Lambda

Function URL
https://i6fy2e3oxcyawhdragpt4j6eim0qqcie.lambda-url.us-east-1.on.aws/

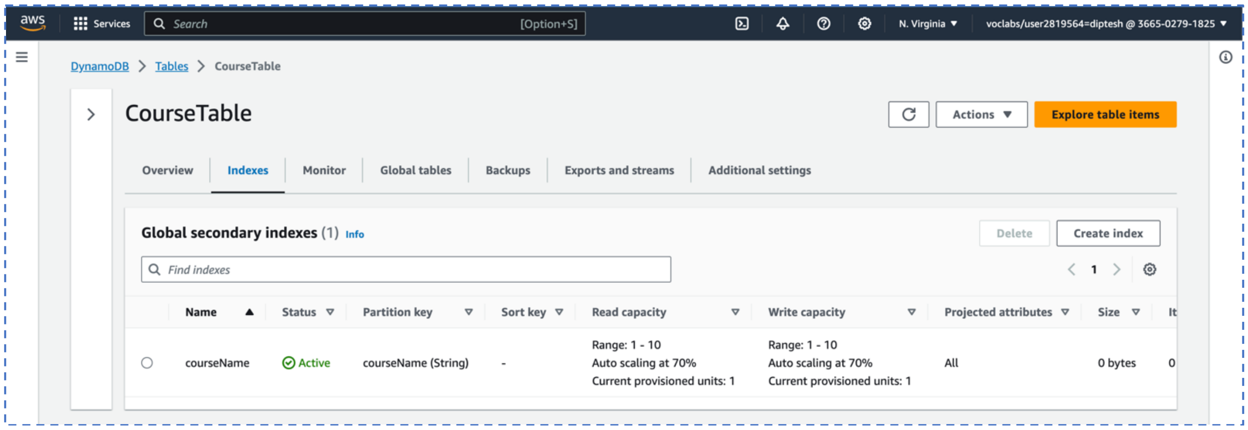
- DynamoDB table created.


DynamoDB > Tables

Tables (1)

Name	Status	Partition key	Sort key	Indexes	Deletion protection	Read capacity mode	Write capacity mode	Total
CourseTable	Active	courseCode (S)	teacherName (S)	1	Off	Provisioned with auto scaling (1)	Provisioned with auto scaling (1)	616 bytes

- Global Secondary Index created.


DynamoDB > Tables > CourseTable

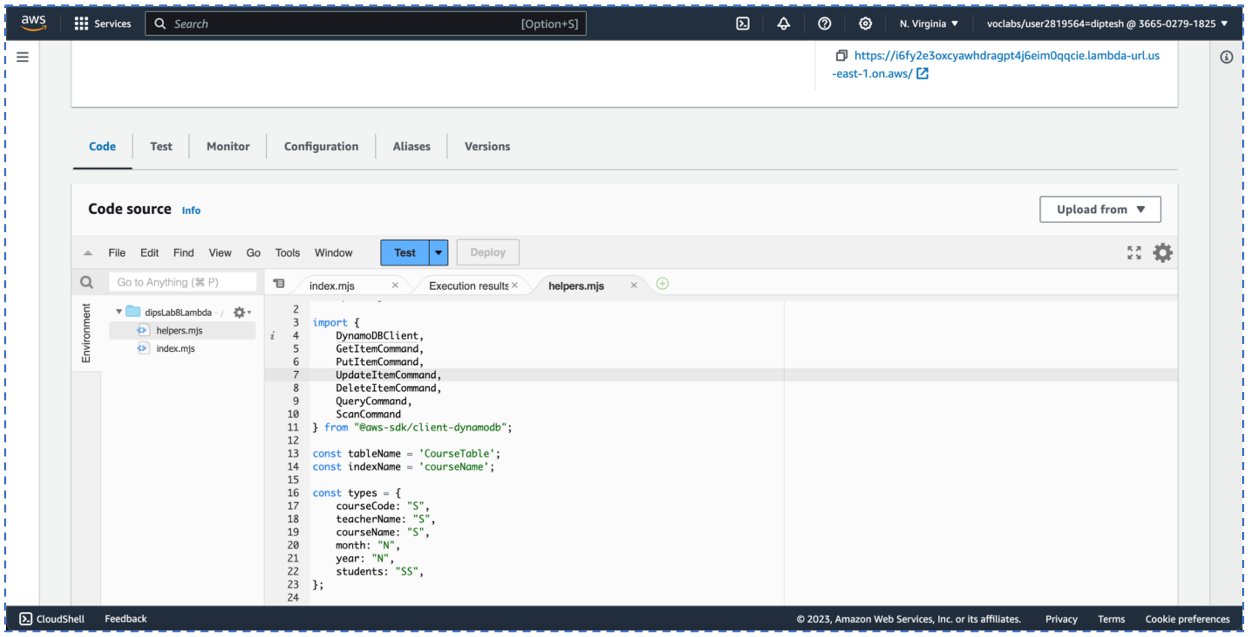
CourseTable

Overview | Indexes | Monitor | Global tables | Backups | Exports and streams | Additional settings

Global secondary indexes (1)

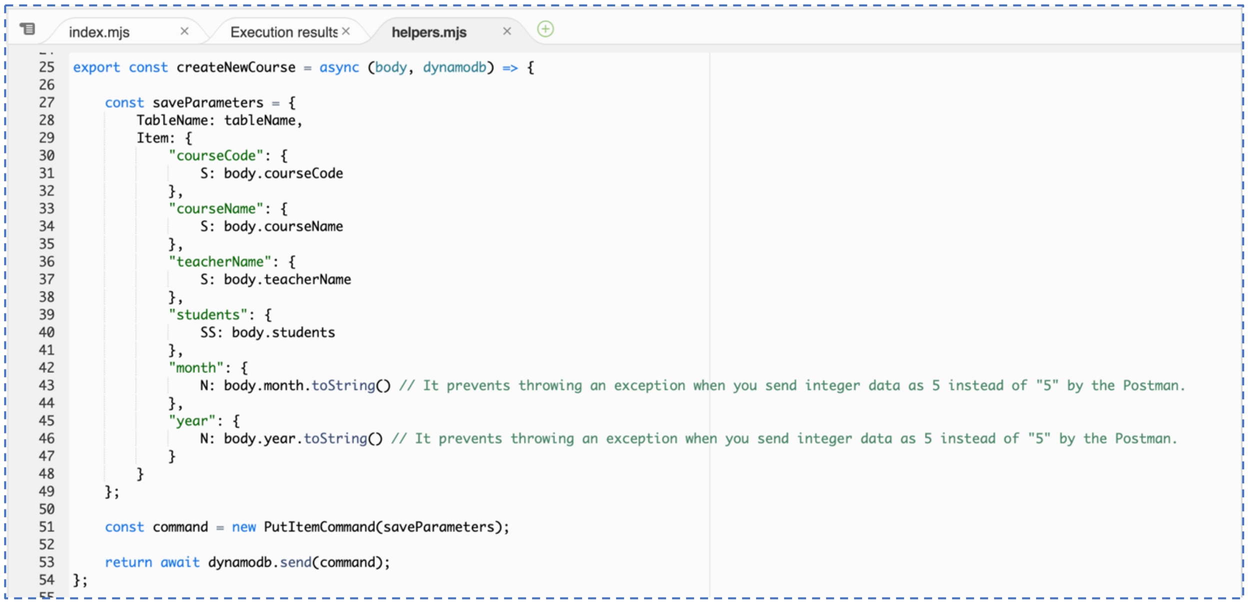
Name	Status	Partition key	Sort key	Read capacity	Write capacity	Projected attributes	Size	It
courseName	Active	courseName (String)	-	Range: 1 - 10 Auto scaling at 70% Current provisioned units: 1	Range: 1 - 10 Auto scaling at 70% Current provisioned units: 1	All	0 bytes	0

- File "helpers.mjs" (Source Code lines 1 – 24).



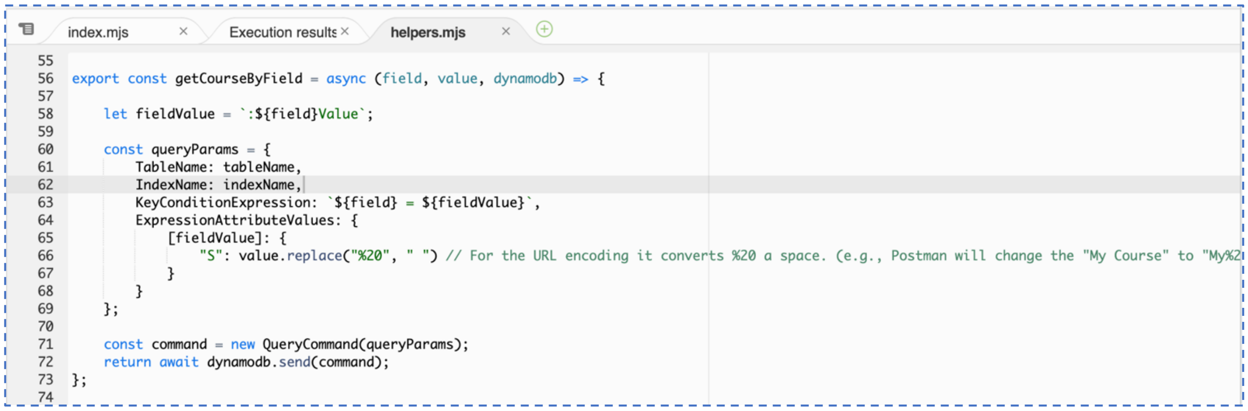
```javascript

import {
    DynamoDBClient,
    GetItemCommand,
    PutItemCommand,
    UpdateItemCommand,
    DeleteItemCommand,
    QueryCommand,
    ScanCommand
} from "@aws-sdk/client-dynamodb";

const tableName = 'CourseTable';
const indexName = 'courseName';

const types = {
    courseCode: "S",
    teacherName: "S",
    courseName: "S",
    month: "N",
    year: "N",
    students: "SS",
};

```

- File 'helpers.mjs' (Source Code lines 25 – 54) – 'createNewCourse' Method.



```javascript
export const createNewCourse = async (body, dynamodb) => {

    const saveParameters = {
        TableName: tableName,
        Item: {
            "courseCode": {
                S: body.courseCode
            },
            "courseName": {
                S: body.courseName
            },
            "teacherName": {
                S: body.teacherName
            },
            "students": {
                SS: body.students
            },
            "month": {
                N: body.month.toString() // It prevents throwing an exception when you send integer data as 5 instead of "5" by the Postman.
            },
            "year": {
                N: body.year.toString() // It prevents throwing an exception when you send integer data as 5 instead of "5" by the Postman.
            }
        }
    };

    const command = new PutItemCommand(saveParameters);

    return await dynamodb.send(command);
};
```

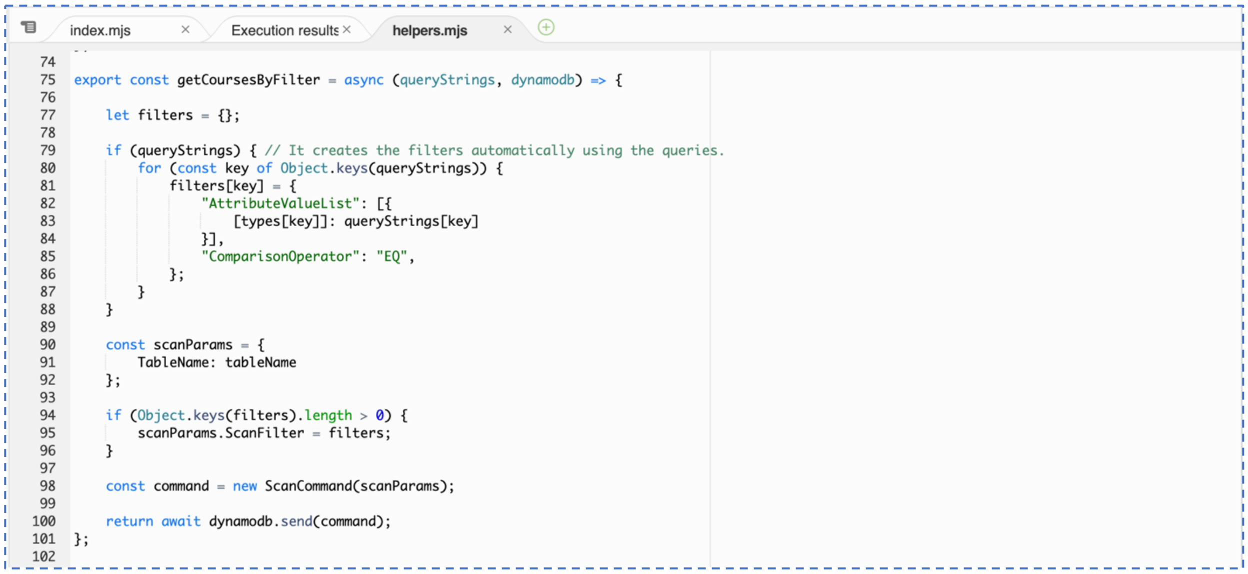
- File 'helpers.mjs' (Source Code lines 55 – 74) – 'getCourseByField' Method.



index.mjs | Execution results | helpers.mjs

```
55
56  export const getCourseByField = async (field, value, dynamodb) => {
57
58      let fieldValue = `:${field}Value`;
59
60      const queryParams = {
61          TableName: tableName,
62          IndexName: indexName,
63          KeyConditionExpression: `${field} = ${fieldValue}`,
64          ExpressionAttributeValues: {
65              [fieldValue]: {
66                  "S": value.replace("%20", " ") // For the URL encoding it converts %20 a space. (e.g., Postman will change the "My Course" to "My%2
67              }
68          }
69      };
70
71      const command = new QueryCommand(queryParams);
72      return await dynamodb.send(command);
73  };
74
```

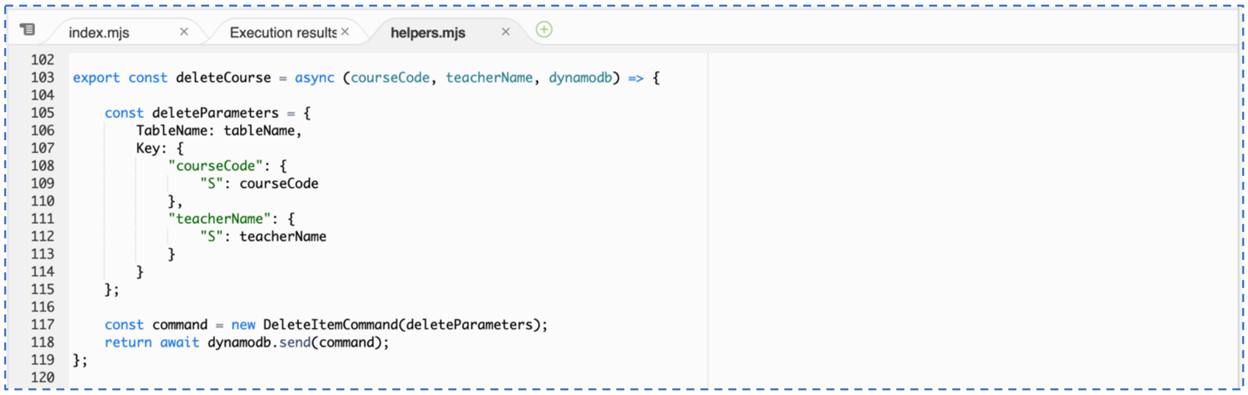
- File 'helpers.mjs' (Source Code lines 75 – 101) – 'getCourseByFilter' Method.



index.mjs | Execution results | helpers.mjs

```
74
75  export const getCoursesByFilter = async (queryStrings, dynamodb) => {
76
77      let filters = {};
78
79      if (queryStrings) { // It creates the filters automatically using the queries.
80          for (const key of Object.keys(queryStrings)) {
81              filters[key] = {
82                  "AttributeValueList": [{
83                      [types[key]]: queryStrings[key]
84                  }],
85                  "ComparisonOperator": "EQ",
86              };
87          }
88      }
89
90      const scanParams = {
91          TableName: tableName
92      };
93
94      if (Object.keys(filters).length > 0) {
95          scanParams.ScanFilter = filters;
96      }
97
98      const command = new ScanCommand(scanParams);
99
100     return await dynamodb.send(command);
101 };
102
```

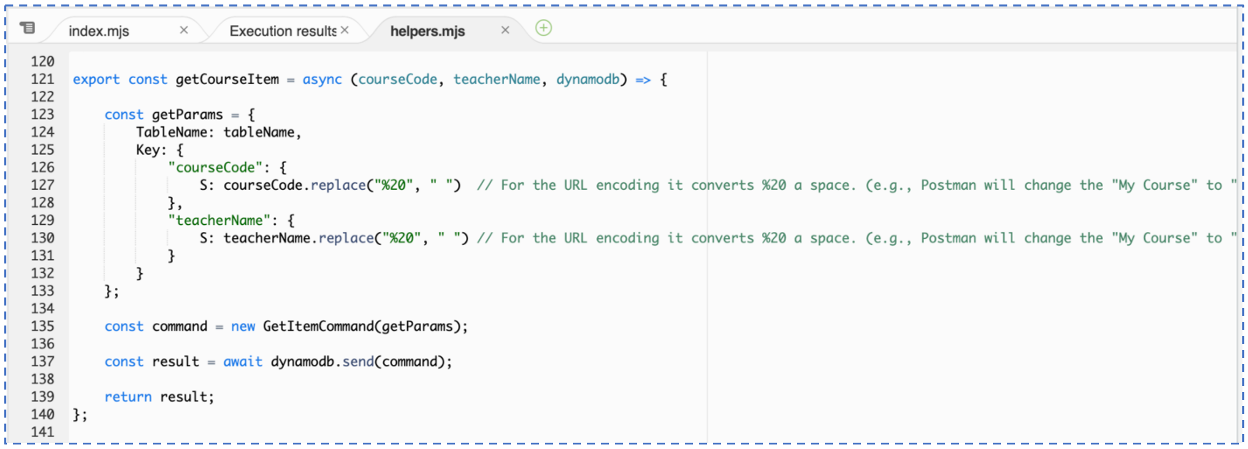
- File 'helpers.mjs' (Source Code lines 102 – 119) – 'deleteCourse' Method.



index.mjs | Execution results | helpers.mjs

```
102
103 export const deleteCourse = async (courseCode, teacherName, dynamodb) => {
104
105     const deleteParameters = {
106         TableName: tableName,
107         Key: {
108             "courseCode": {
109                 "S": courseCode
110             },
111             "teacherName": {
112                 "S": teacherName
113             }
114         }
115     };
116
117     const command = new DeleteItemCommand(deleteParameters);
118     return await dynamodb.send(command);
119 };
120
```

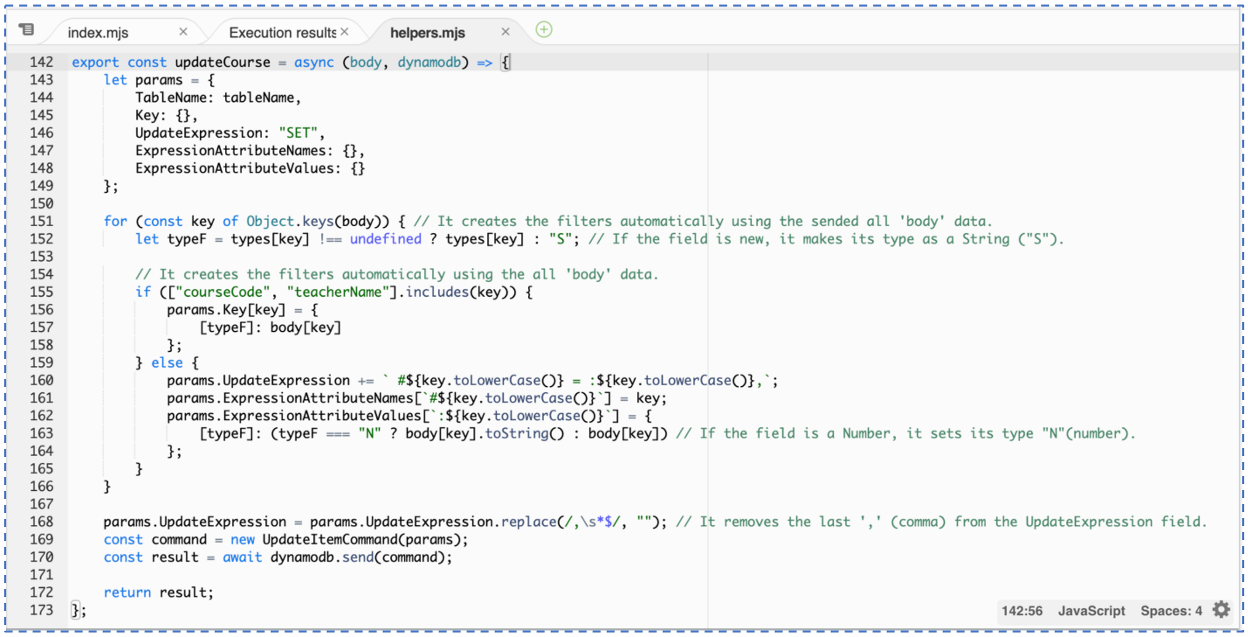
- File ‘helpers.mjs’ (Source Code lines 120 – 141) – ‘getCourseItem’ Method.



index.mjs × | Execution results × | helpers.mjs ×

```
120
121 export const getCourseItem = async (courseCode, teacherName, dynamodb) => {
122
123     const getParams = {
124         TableName: tableName,
125         Key: {
126             "courseCode": {
127                 S: courseCode.replace("%20", " ")  // For the URL encoding it converts %20 a space. (e.g., Postman will change the "My Course" to "
128             },
129             "teacherName": {
130                 S: teacherName.replace("%20", " ") // For the URL encoding it converts %20 a space. (e.g., Postman will change the "My Course" to "
131             }
132         }
133     };
134
135     const command = new GetItemCommand(getParams);
136
137     const result = await dynamodb.send(command);
138
139     return result;
140 };
141
```

- File ‘helpers.mjs’ (Source Code lines 142 – 172) – ‘updateCourse’ Method.

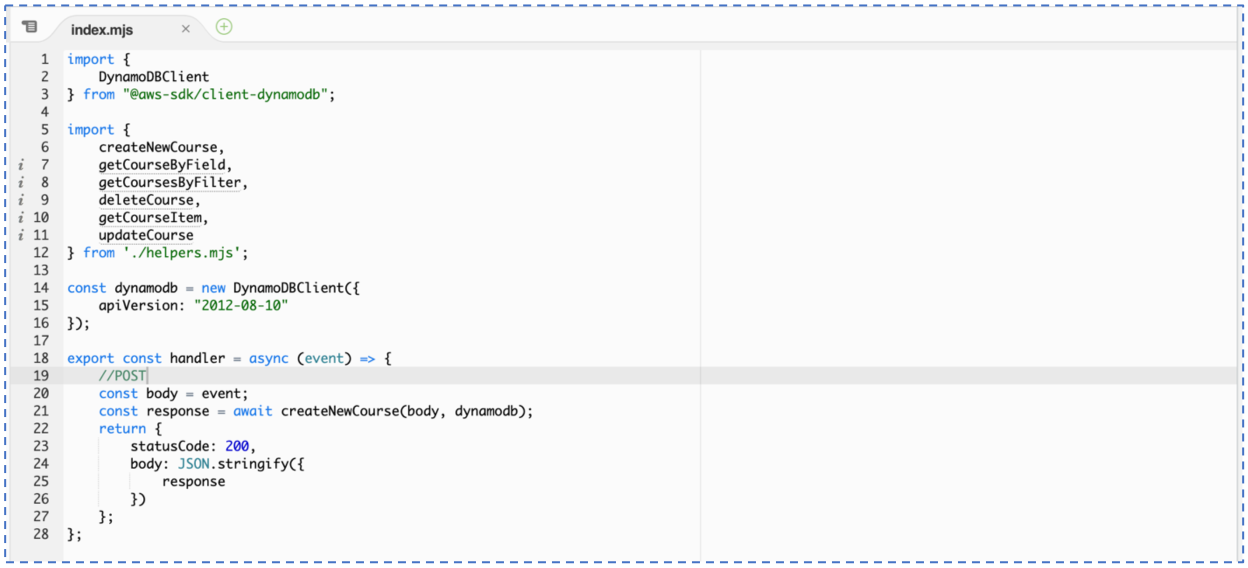


index.mjs × | Execution results × | helpers.mjs ×

```
142 export const updateCourse = async (body, dynamodb) => {
143     let params = {
144         TableName: tableName,
145         Key: {},
146         UpdateExpression: "SET",
147         ExpressionAttributeNames: {},
148         ExpressionAttributeValues: {}
149     };
150
151     for (const key of Object.keys(body)) { // It creates the filters automatically using the sended all 'body' data.
152         let typeF = types[key] !== undefined ? types[key] : "S"; // If the field is new, it makes its type as a String ("S").
153
154         // It creates the filters automatically using the all 'body' data.
155         if (["courseCode", "teacherName"].includes(key)) {
156             params.Key[key] = {
157                 [typeF]: body[key]
158             };
159         } else {
160             params.UpdateExpression += ` #${key.toLowerCase()} = :${key.toLowerCase()},`;
161             params.ExpressionAttributeNames[`#${key.toLowerCase()}`] = key;
162             params.ExpressionAttributeValues[`:${key.toLowerCase()}`] = {
163                 [typeF]: (typeF === "N" ? body[key].toString() : body[key]) // If the field is a Number, it sets its type "N"(number).
164             };
165         }
166     }
167
168     params.UpdateExpression = params.UpdateExpression.replace(/,\s*$/, ""); // It removes the last ',' (comma) from the UpdateExpression field.
169     const command = new UpdateItemCommand(params);
170     const result = await dynamodb.send(command);
171
172     return result;
173 };
```

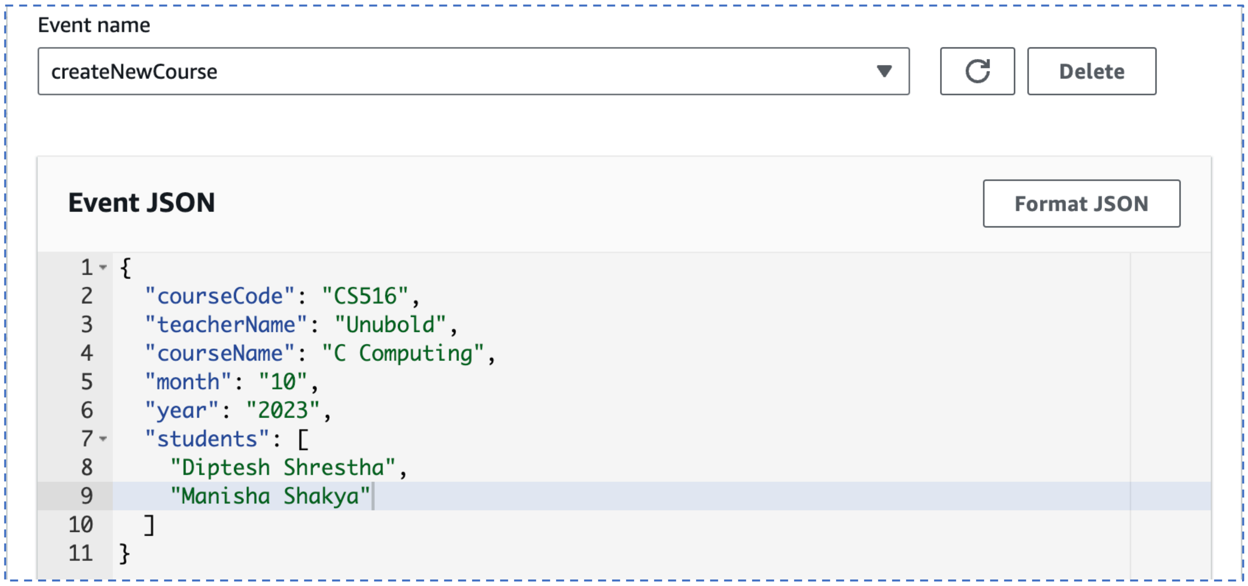
142:56 JavaScript Spaces: 4

- File 'index.mjs' updated for the POST operation.



```
import {
    DynamoDBClient
} from "@aws-sdk/client-dynamodb";

import {
    createNewCourse,
    getCourseByField,
    getCoursesByFilter,
    deleteCourse,
    getCourseItem,
    updateCourse
} from './helpers.mjs';

const dynamodb = new DynamoDBClient({
    apiVersion: "2012-08-10"
});

export const handler = async (event) => {
    //POST
    const body = event;
    const response = await createNewCourse(body, dynamodb);
    return {
        statusCode: 200,
        body: JSON.stringify({
            response
        })
    };
};
```

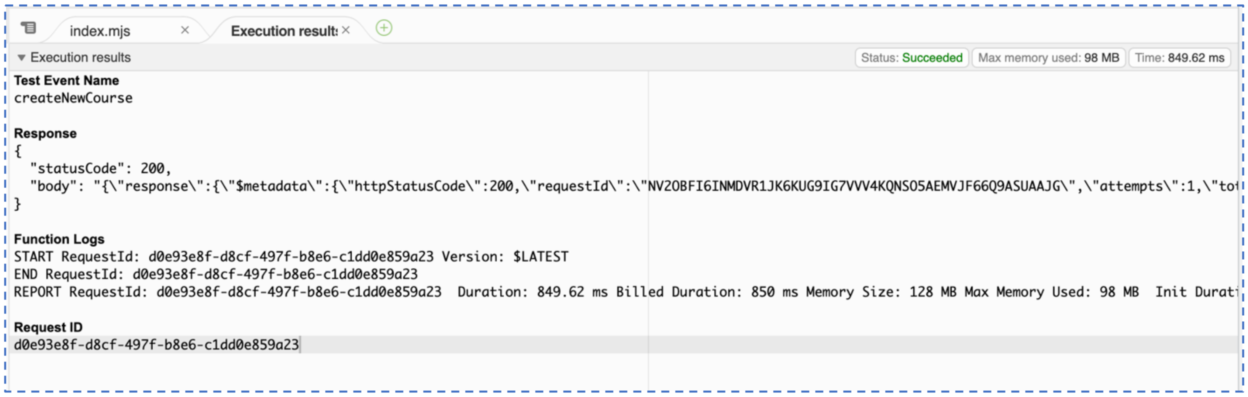
- New Event 'createNewCourse' to TEST the POST operation.



Event name: createNewCourse

```
{
  "courseCode": "CS516",
  "teacherName": "Unubold",
  "courseName": "C Computing",
  "month": "10",
  "year": "2023",
  "students": [
    "Diptesh Shrestha",
    "Manisha Shakya"
  ]
}
```

- TEST execution result for POST operation.



index.mjs | Execution results

Execution results — Status: Succeeded | Max memory used: 98 MB | Time: 849.62 ms

Test Event Name

```
createNewCourse
```

Response

```
{
  "statusCode": 200,
  "body": "{\"response\":{\"$metadata\":{\"httpStatusCode\":200,\"requestId\":\"NV2OBFI6INMDVR1JK6KUG9IG7VVV4KQNSO5AEMVJF66Q9ASUAAJG\",\"attempts\":1,\"to
}
```

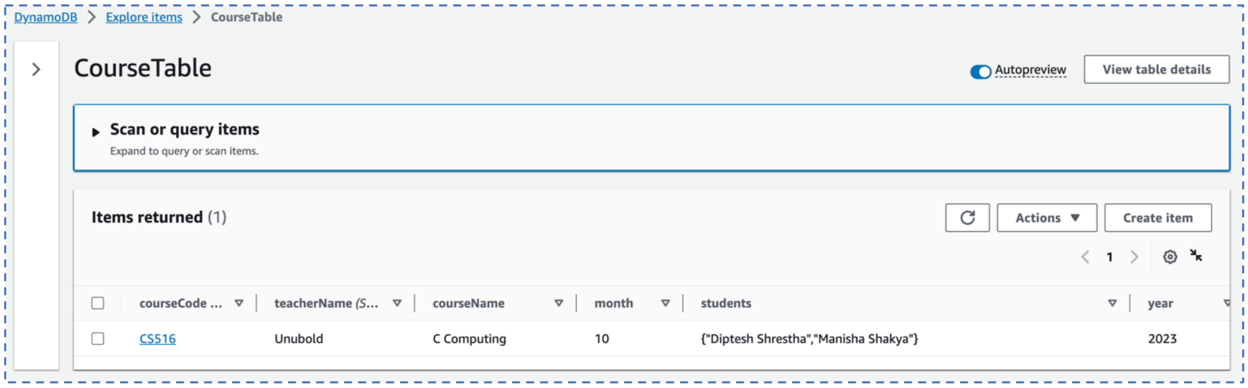
Function Logs

```
START RequestId: d0e93e8f-d8cf-497f-b8e6-c1dd0e859a23 Version: $LATEST
END RequestId: d0e93e8f-d8cf-497f-b8e6-c1dd0e859a23
REPORT RequestId: d0e93e8f-d8cf-497f-b8e6-c1dd0e859a23  Duration: 849.62 ms Billed Duration: 850 ms Memory Size: 128 MB Max Memory Used: 98 MB  Init Durat
```

Request ID

```
d0e93e8f-d8cf-497f-b8e6-c1dd0e859a23
```

- New record inserted in the DynamoDB table named 'CourseTable'.



DynamoDB > Explore items > CourseTable

CourseTable

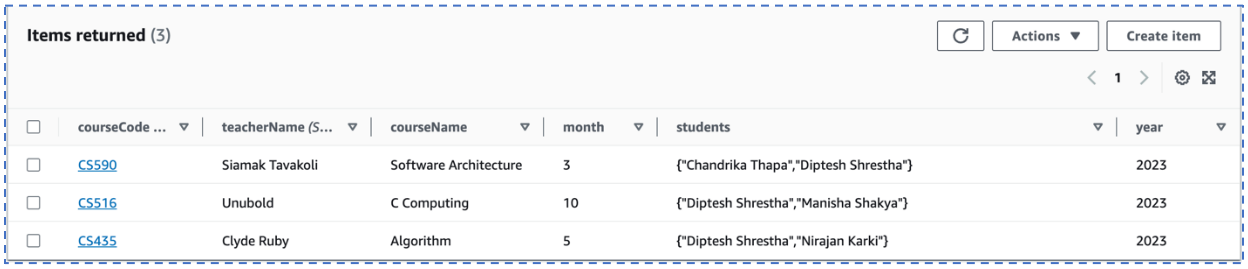
Autopreview | View table details

Scan or query items
Expand to query or scan items.

Items returned (1)

courseCode ...	teacherName (S...	courseName	month	students	year
CS516	Unubold	C Computing	10	{"Diptesh Shrestha","Manisha Shakya"}	2023

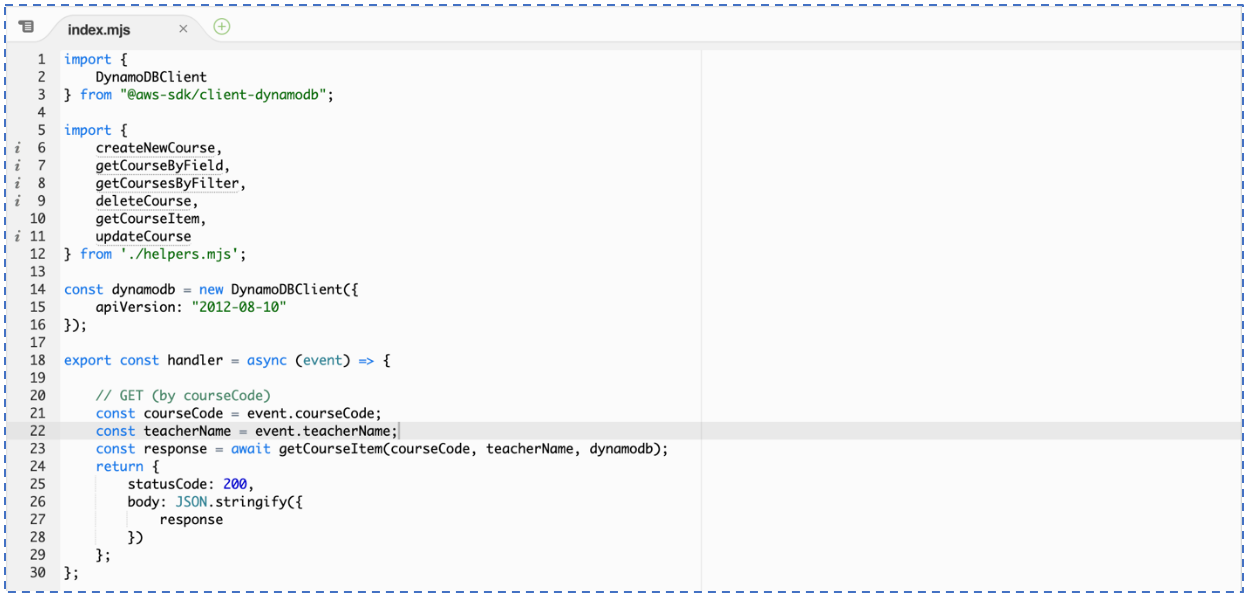
- Added few more records to the 'CourseTable'.



Items returned (3)

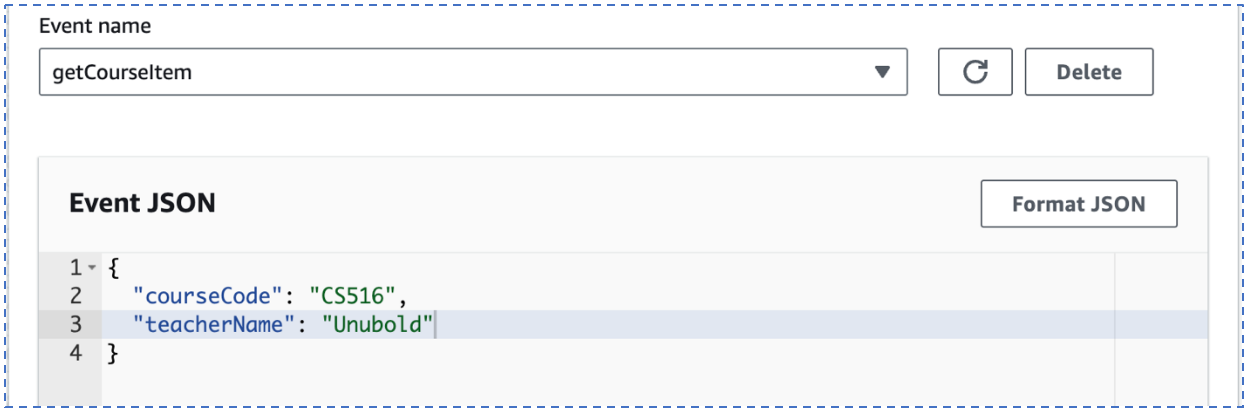
courseCode ...	teacherName (S...	courseName	month	students	year
CS590	Siamak Tavakoli	Software Architecture	3	{"Chandrika Thapa","Diptesh Shrestha"}	2023
CS516	Unubold	C Computing	10	{"Diptesh Shrestha","Manisha Shakya"}	2023
CS435	Clyde Ruby	Algorithm	5	{"Diptesh Shrestha","Nirajan Karki"}	2023

- File ‘index.mjs’ updated for the GET operation (get course by courseCode & teacherName).



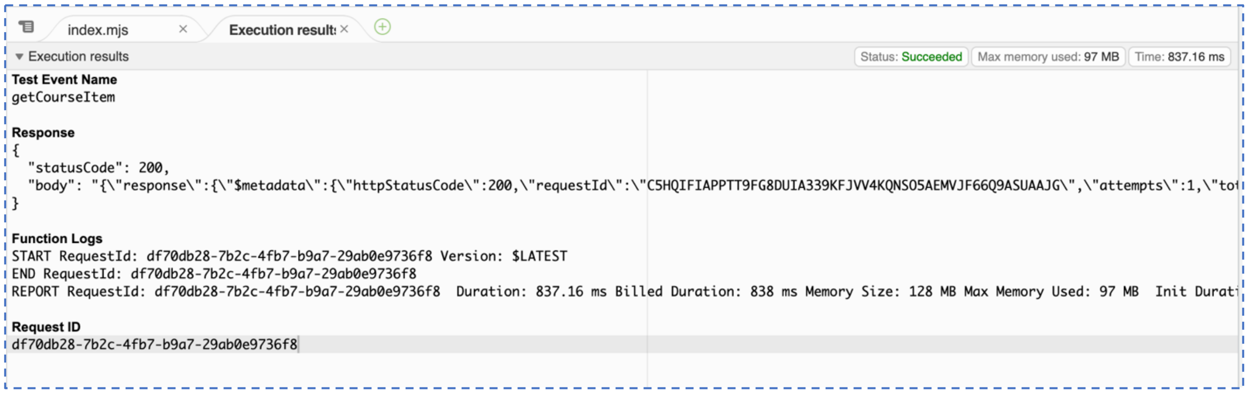
```javascript
import {
    DynamoDBClient
} from "@aws-sdk/client-dynamodb";

import {
    createNewCourse,
    getCourseByField,
    getCoursesByFilter,
    deleteCourse,
    getCourseItem,
    updateCourse
} from './helpers.mjs';

const dynamodb = new DynamoDBClient({
    apiVersion: "2012-08-10"
});

export const handler = async (event) => {

    // GET (by courseCode)
    const courseCode = event.courseCode;
    const teacherName = event.teacherName;
    const response = await getCourseItem(courseCode, teacherName, dynamodb);
    return {
        statusCode: 200,
        body: JSON.stringify({
            response
        })
    };
};
```

- Test data updated for getCourseItem.



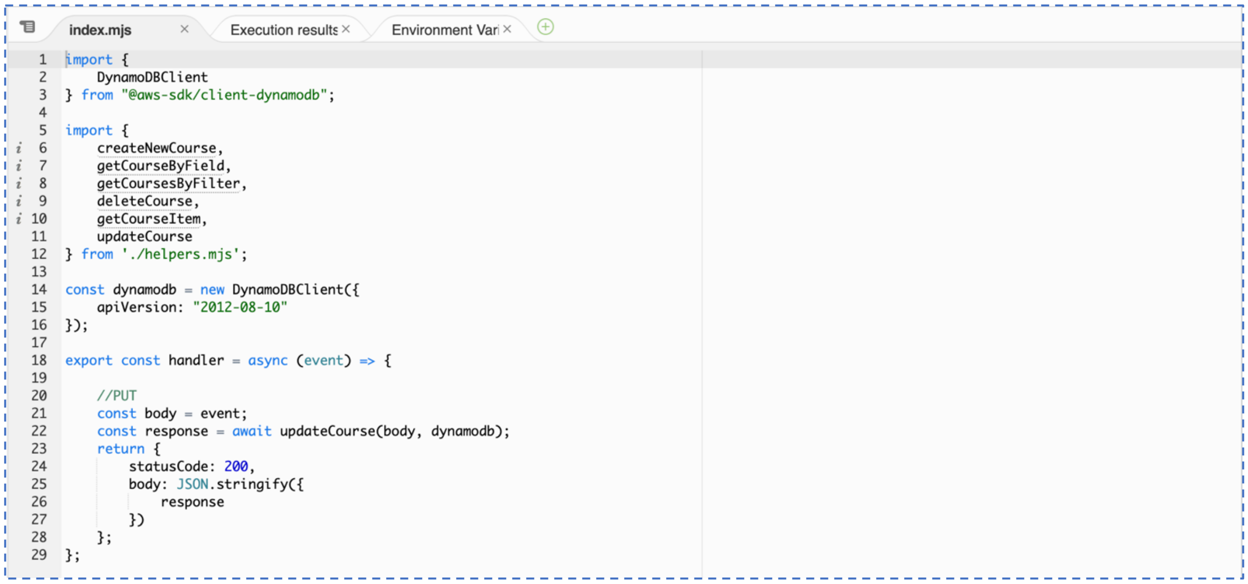
```json
{
  "courseCode": "CS516",
  "teacherName": "Unubold"
}
```

- Result of the getCourseItem.



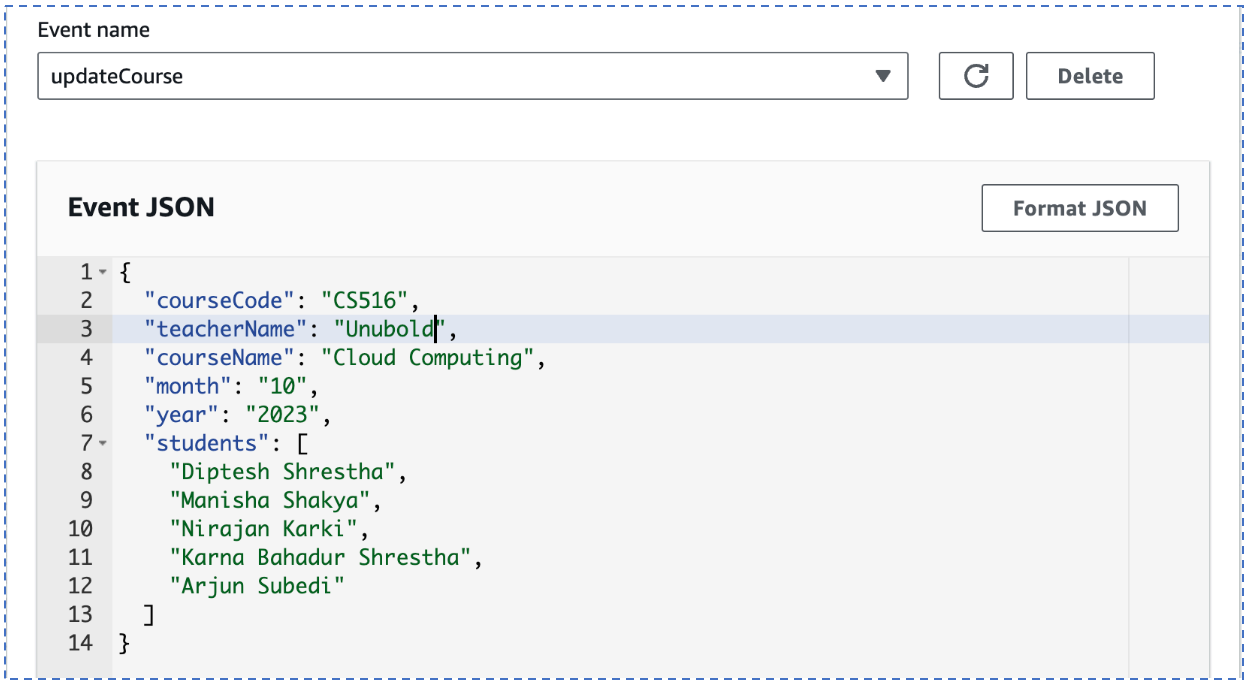
```
Test Event Name
getCourseItem

Response
{
  "statusCode": 200,
  "body": "{\"response\":{\"$metadata\":{\"httpStatusCode\":200,\"requestId\":\"C5HQIFIAPPTT9FG8DUIA339KFJVV4KQNSO5AEMVJF66Q9ASUAAJG\",\"attempts\":1,\"to
}

Function Logs
START RequestId: df70db28-7b2c-4fb7-b9a7-29ab0e9736f8 Version: $LATEST
END RequestId: df70db28-7b2c-4fb7-b9a7-29ab0e9736f8
REPORT RequestId: df70db28-7b2c-4fb7-b9a7-29ab0e9736f8  Duration: 837.16 ms Billed Duration: 838 ms Memory Size: 128 MB Max Memory Used: 97 MB  Init Durat

Request ID
df70db28-7b2c-4fb7-b9a7-29ab0e9736f8
```

- File 'index.mjs' updated for the PUT operation.



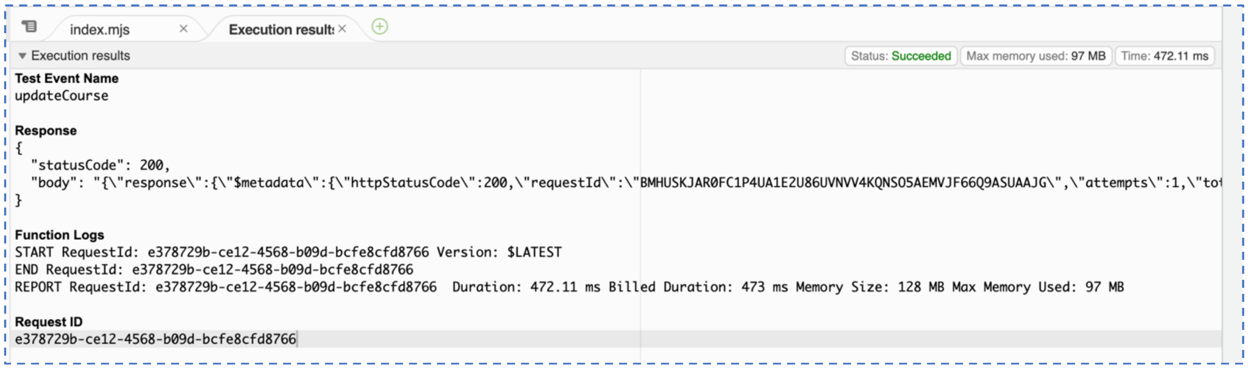
```
import {
    DynamoDBClient
} from "@aws-sdk/client-dynamodb";

import {
    createNewCourse,
    getCourseByField,
    getCoursesByFilter,
    deleteCourse,
    getCourseItem,
    updateCourse
} from './helpers.mjs';

const dynamodb = new DynamoDBClient({
    apiVersion: "2012-08-10"
});

export const handler = async (event) => {

    //PUT
    const body = event;
    const response = await updateCourse(body, dynamodb);
    return {
        statusCode: 200,
        body: JSON.stringify({
            response
        })
    };
};
```

- Test data updated for updateCourse.



```
{
  "courseCode": "CS516",
  "teacherName": "Unubold",
  "courseName": "Cloud Computing",
  "month": "10",
  "year": "2023",
  "students": [
    "Diptesh Shrestha",
    "Manisha Shakya",
    "Nirajan Karki",
    "Karna Bahadur Shrestha",
    "Arjun Subedi"
  ]
}
```

- Result of the updateCourse.



index.mjs | Execution results

Execution results — Status: Succeeded | Max memory used: 97 MB | Time: 472.11 ms

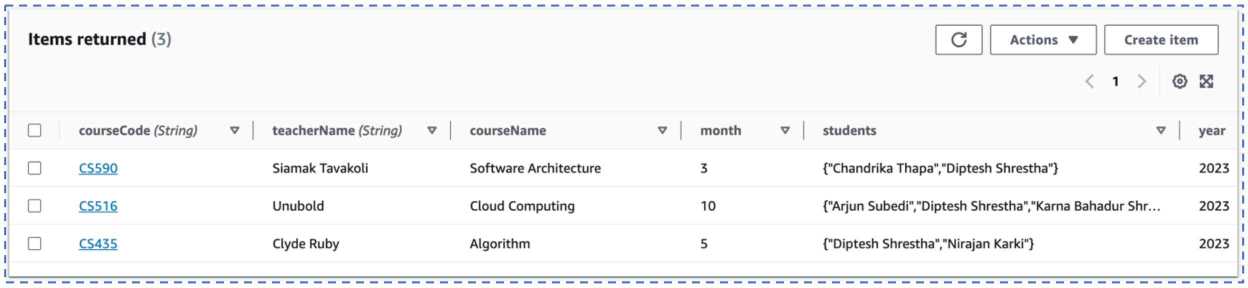
Test Event Name
updateCourse

Response
```
{
  "statusCode": 200,
  "body": "{\"response\":{\"$metadata\":{\"httpStatusCode\":200,\"requestId\":\"BMHUSKJAR0FC1P4UA1E2U86UVNVV4KQNSO5AEMVJF66Q9ASUAAJG\",\"attempts\":1,\"tot
}
```

Function Logs
```
START RequestId: e378729b-ce12-4568-b09d-bcfe8cfd8766 Version: $LATEST
END RequestId: e378729b-ce12-4568-b09d-bcfe8cfd8766
REPORT RequestId: e378729b-ce12-4568-b09d-bcfe8cfd8766  Duration: 472.11 ms Billed Duration: 473 ms Memory Size: 128 MB Max Memory Used: 97 MB
```

Request ID
e378729b-ce12-4568-b09d-bcfe8cfd8766

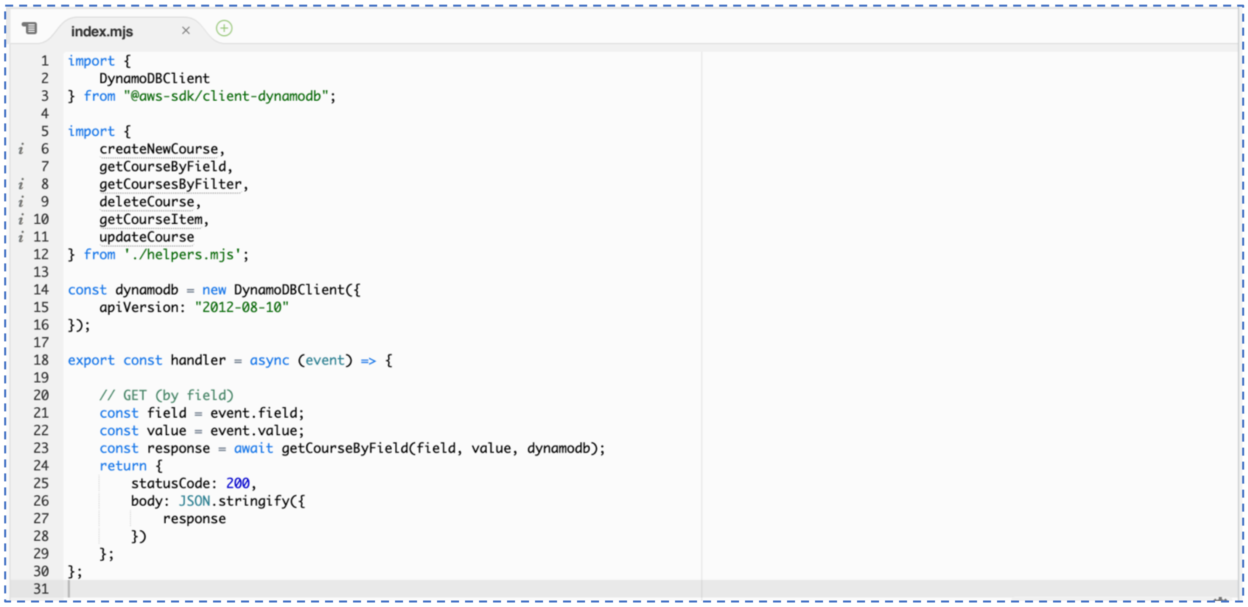
- Record updated in the ‘CourseTable’ – courseName and student’s data were updated.



Items returned (3)

courseCode (String)	teacherName (String)	courseName	month	students	year
CS590	Siamak Tavakoli	Software Architecture	3	{"Chandrika Thapa","Diptesh Shrestha"}	2023
CS516	Unubold	Cloud Computing	10	{"Arjun Subedi","Diptesh Shrestha","Karna Bahadur Shr…	2023
CS435	Clyde Ruby	Algorithm	5	{"Diptesh Shrestha","Nirajan Karki"}	2023

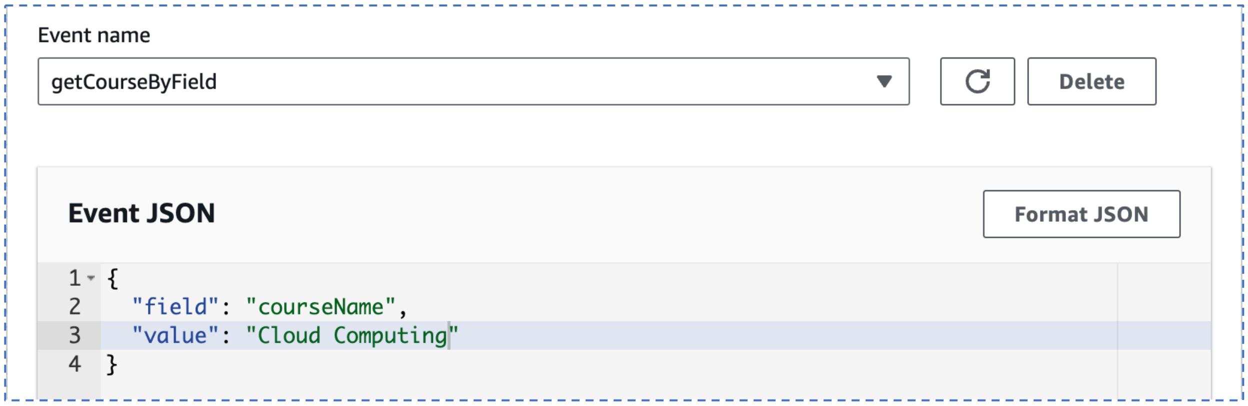
- File ‘index.mjs’ updated for the GET operation (Query on index field).



index.mjs

```javascript
import {
    DynamoDBClient
} from "@aws-sdk/client-dynamodb";

import {
    createNewCourse,
    getCourseByField,
    getCoursesByFilter,
    deleteCourse,
    getCourseItem,
    updateCourse
} from './helpers.mjs';

const dynamodb = new DynamoDBClient({
    apiVersion: "2012-08-10"
});

export const handler = async (event) => {

    // GET (by field)
    const field = event.field;
    const value = event.value;
    const response = await getCourseByField(field, value, dynamodb);
    return {
        statusCode: 200,
        body: JSON.stringify({
            response
        })
    };
};

```

- Test data updated for getCourseByField.



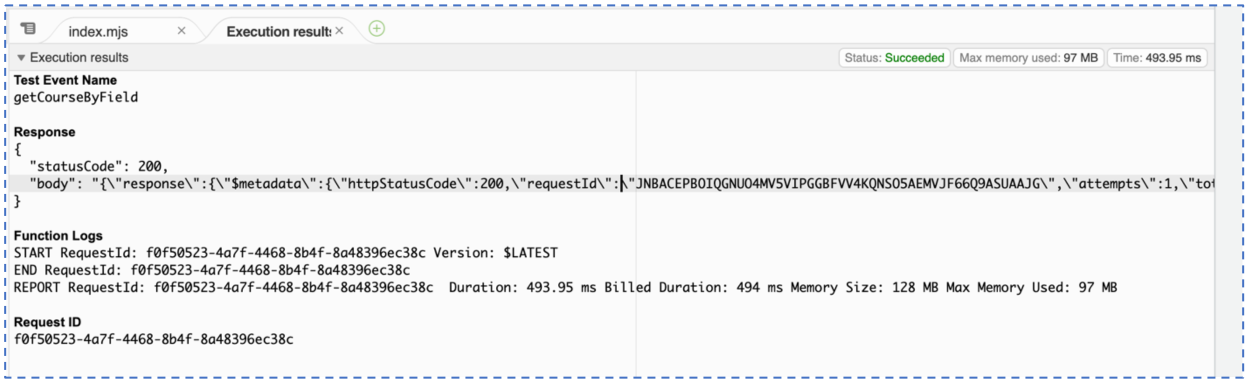
Event name

getCourseByField

Event JSON

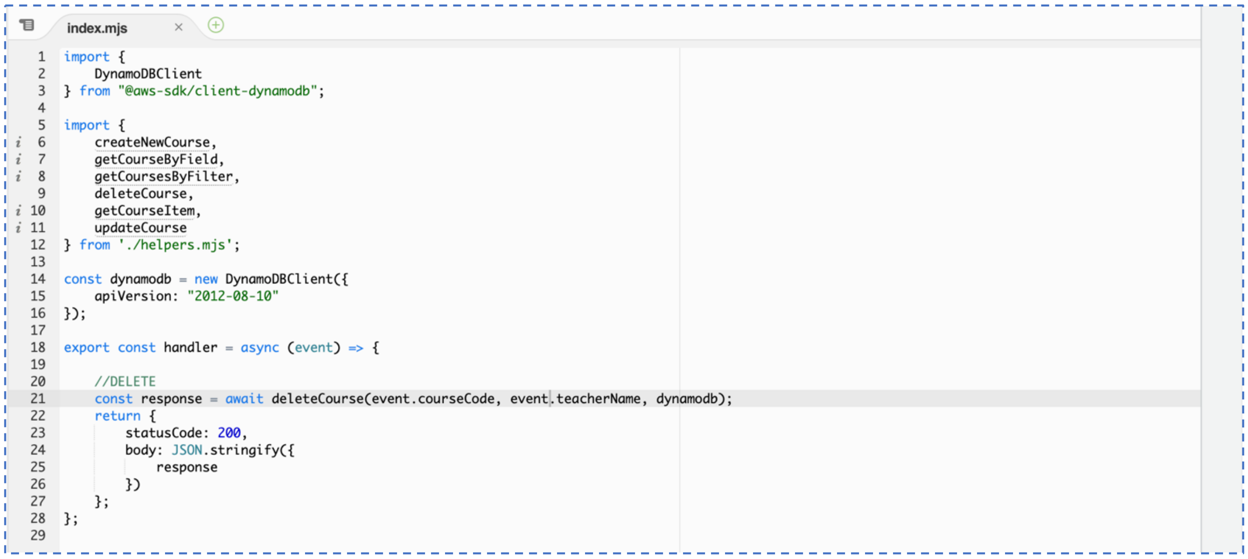
```
{
  "field": "courseName",
  "value": "Cloud Computing"
}
```

- Result of the getCourseByField.



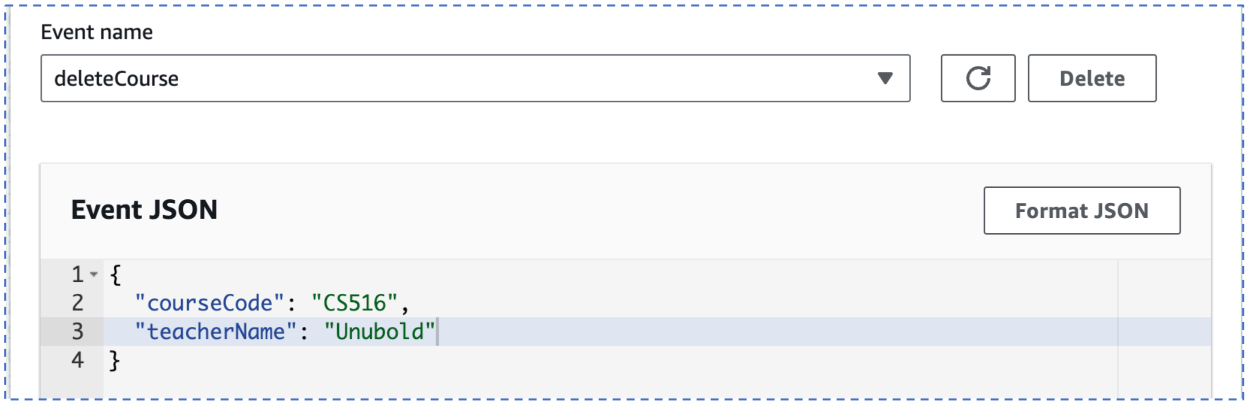
```
Test Event Name
getCourseByField

Response
{
  "statusCode": 200,
  "body": "{\"response\":{\"$metadata\":{\"httpStatusCode\":200,\"requestId\":\"JNBACEPBOIQGNUO4MV5VIPGGBFVV4KQNSO5AEMVJF66Q9ASUAAJG\",\"attempts\":1,\"tot
}

Function Logs
START RequestId: f0f50523-4a7f-4468-8b4f-8a48396ec38c Version: $LATEST
END RequestId: f0f50523-4a7f-4468-8b4f-8a48396ec38c
REPORT RequestId: f0f50523-4a7f-4468-8b4f-8a48396ec38c  Duration: 493.95 ms Billed Duration: 494 ms Memory Size: 128 MB Max Memory Used: 97 MB

Request ID
f0f50523-4a7f-4468-8b4f-8a48396ec38c
```

- File 'index.mjs' updated for the DELETE operation.



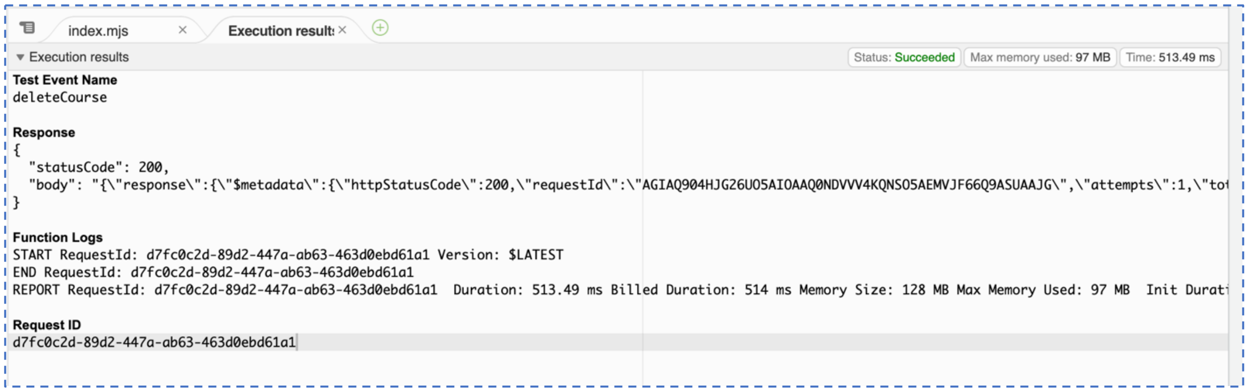
```
import {
    DynamoDBClient
} from "@aws-sdk/client-dynamodb";

import {
    createNewCourse,
    getCourseByField,
    getCoursesByFilter,
    deleteCourse,
    getCourseItem,
    updateCourse
} from './helpers.mjs';

const dynamodb = new DynamoDBClient({
    apiVersion: "2012-08-10"
});

export const handler = async (event) => {

    //DELETE
    const response = await deleteCourse(event.courseCode, event.teacherName, dynamodb);
    return {
        statusCode: 200,
        body: JSON.stringify({
            response
        })
    };
};

```

- Test data updated for deleteCourse.



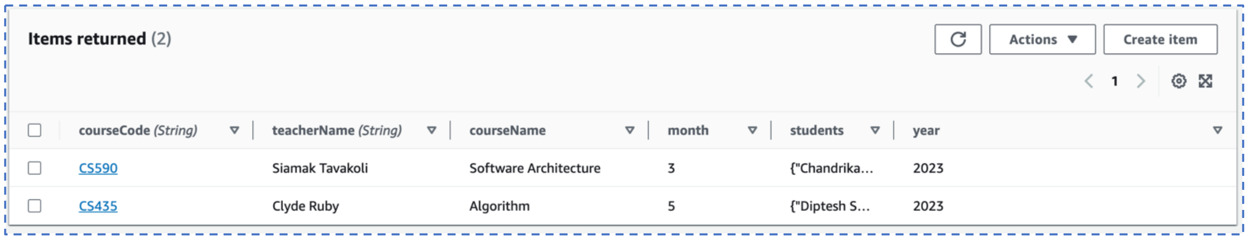
```
{
  "courseCode": "CS516",
  "teacherName": "Unubold"
}
```

- Result of the deleteCourse.



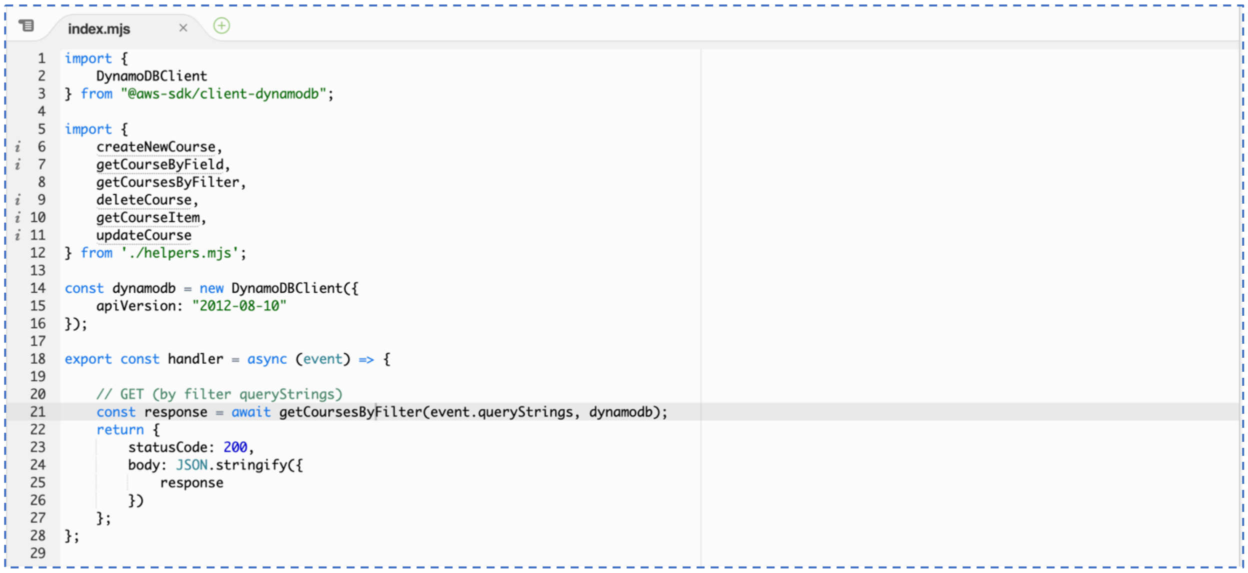
```
Test Event Name
deleteCourse

Response
{
  "statusCode": 200,
  "body": "{\"response\":{\"$metadata\":{\"httpStatusCode\":200,\"requestId\":\"AGIAQ904HJG26UO5AIOAAQ0NDVVV4KQNSO5AEMVJF66Q9ASUAAJG\",\"attempts\":1,\"to
}

Function Logs
START RequestId: d7fc0c2d-89d2-447a-ab63-463d0ebd61a1 Version: $LATEST
END RequestId: d7fc0c2d-89d2-447a-ab63-463d0ebd61a1
REPORT RequestId: d7fc0c2d-89d2-447a-ab63-463d0ebd61a1  Duration: 513.49 ms Billed Duration: 514 ms Memory Size: 128 MB Max Memory Used: 97 MB  Init Durat

Request ID
d7fc0c2d-89d2-447a-ab63-463d0ebd61a1
```

- Record deleted from the 'CourseTable' (No more record with courseCode : CS516).



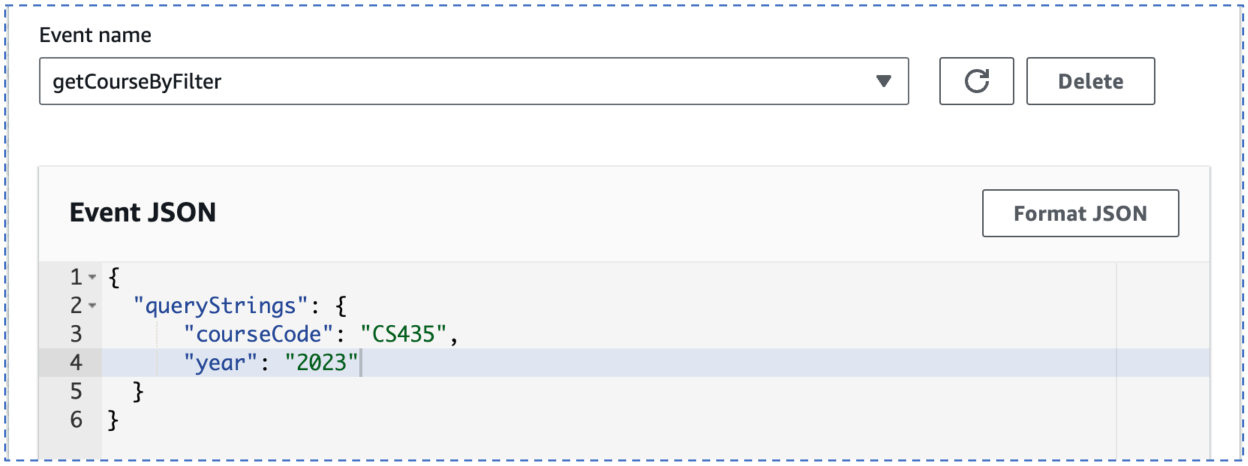
	courseCode *(String)*	teacherName *(String)*	courseName	month	students	year
☐	CS590	Siamak Tavakoli	Software Architecture	3	{"Chandrika…	2023
☐	CS435	Clyde Ruby	Algorithm	5	{"Diptesh S…	2023

- File 'index.mjs' updated for the GET operation (Scan query – by filter).



```
import {
    DynamoDBClient
} from "@aws-sdk/client-dynamodb";

import {
    createNewCourse,
    getCourseByField,
    getCoursesByFilter,
    deleteCourse,
    getCourseItem,
    updateCourse
} from './helpers.mjs';

const dynamodb = new DynamoDBClient({
    apiVersion: "2012-08-10"
});

export const handler = async (event) => {

    // GET (by filter queryStrings)
    const response = await getCoursesByFilter(event.queryStrings, dynamodb);
    return {
        statusCode: 200,
        body: JSON.stringify({
            response
        })
    };
};
```

- Test data updated for getCourseByFilter.

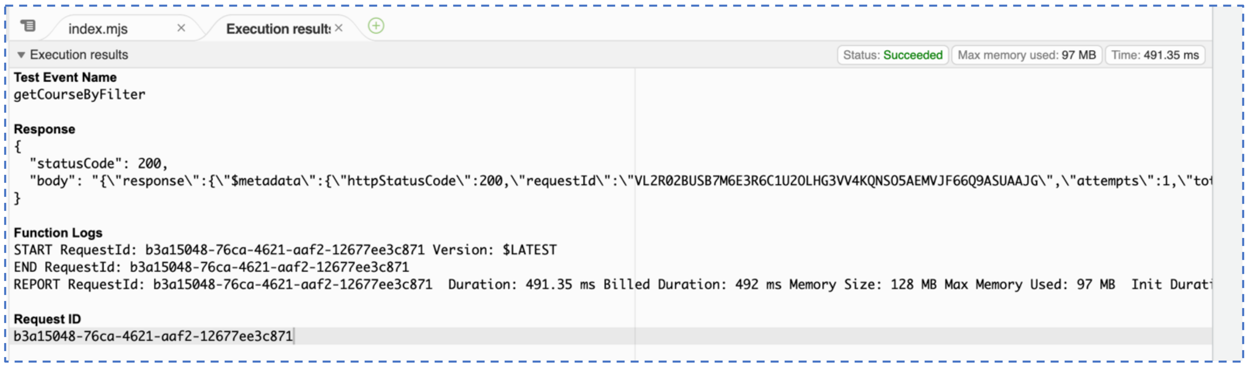


Event name: getCourseByFilter

Event JSON

```
{
    "queryStrings": {
        "courseCode": "CS435",
        "year": "2023"
    }
}
```

- Result of the getCourseByFilter.



```
Execution results                Status: Succeeded | Max memory used: 97 MB | Time: 491.35 ms
Test Event Name
getCourseByFilter

Response
{
  "statusCode": 200,
  "body": "{\"response\":{\"$metadata\":{\"httpStatusCode\":200,\"requestId\":\"VL2R02BUSB7M6E3R6C1U2OLHG3VV4KQNSO5AEMVJF66Q9ASUAAJG\",\"attempts\":1,\"to
}

Function Logs
START RequestId: b3a15048-76ca-4621-aaf2-12677ee3c871 Version: $LATEST
END RequestId: b3a15048-76ca-4621-aaf2-12677ee3c871
REPORT RequestId: b3a15048-76ca-4621-aaf2-12677ee3c871  Duration: 491.35 ms Billed Duration: 492 ms Memory Size: 128 MB Max Memory Used: 97 MB  Init Durat

Request ID
b3a15048-76ca-4621-aaf2-12677ee3c871
```