Assignments 7 and 8 – Lambda with CRUD operations

Serverless is awesome and costs $0 even if you didn't delete them. Do this and the next assignments in your personal account.

Upload screenshots of:

- Source code for all CRUD operations. Otherwise, you will get 1 out of 2 points.
- Records in the DynamoDB table
- Responses after testing for each CRUD operation. Test it in the AWS console with test events.

There are so many resources you can check out such as the "Be a better dev" channel if you need help.

Task 0. Practice version, alias, and weighted (canary) deployment

Task 1. Create a DynamoDB table, Lambda, and implement the Save functionality

- Create an IAM policy that allows CRUD operations on the table. Do this and the next assignments in your personal account. They don't cost. AWS Academy setup is different from regular accounts which may result in insufficient permission errors. Therefore, please practice these assignments in a standard environment, your own AWS account.
- Create a DynamoDB table, "CourseTable".
  a. courseCode (string) -> Partition key
  b. teacherName (string) -> Sort key
- Implement the PutItem (Create) operation in the Lambda. You can add any attribute at any time in the payload to see DynamoDB's flexibility and schemless behavior.
  a. courseName (string)
  b. month (number) – Which month of the year the course was taught
  c. year (number)
  d. students (string set)

Task 2 – Implement the rest of the CRUD operations

1. Update the Course Lambda to do the rest of the **CRUD** operations, read, update, and delete. Scan items with filters. Include all CRUD operations **in one function**.
   a. **GetItem** and **DeleteItem** – Relatively simple. You just need to provide the composite key (courseCode and teacherName)
   b. **Scan** - Get items with some filter for example, month, and year. You can get these values as a query parameter or path parameter.
   c. **UpdateItem** - update non-key attributes such as month and year. You need to provide the composite key to update specific attributes of the item.
2. Test each CRUD operation by creating an "**apigateway-aws-proxy**" test event. The "**path**" and "**httpMethod**" attributes in the event can be used to choose the specific operation.

This task takes the longest time. That is why you have more time compared to other assignments. Don't wait until the last minute. Start in advance. You can do your own research on the DynamoDB CRUD APIs. There are resources all over the internet. Refer:

- Official documentation
- SDK
- Internet blogs

Sample code for creating an item. Don't copy and paste. If you copy and paste, in most cases it doesn't work or you learn nothing. Just refer to it and write the code yourself!

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

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

export const handler = async(event) => {
    const saveParameters = {
        TableName: 'prep',
        Item: {
            "courseCode": {
                S: 'CS516'
            },
            "courseName": {
                S: 'CC'
            },
            "teacherName": {
                S: 'Uno'
            }
        }
    };

    const command = new PutItemCommand(saveParameters);
    await dynamodb.send(command);

    const response = {
        statusCode: 200,
        body: 'success'
    };

    return response;
};
```