Assignment 9 – API Gateway and Cognito

PART I – Lambda and DynamoDB [is already done in the previous assignments]

In the previous lab, we prepared a CRUD app in Lambda backed with DynamoDB. In this lab, we will make that an API. So, it is publicly accessible. We will also secure it with JWT tokens from Cognito.

PART II - API gateway and Cognito

- Create a CRUD API for the sample course app in API Gateway.
    - Create a REST API. The process is straightforward. Only give it a name. Create a resource called "courses".
    - Create a "method" in it.
        - Select a method such as POST, GET, PATCH.
        - You must toggle **Lambda proxy integration**
        - How you write your function is up to you. Normally, I create ONE function for all CRUD operations. So select the Lambda that has the implementation.
- Create the following 4 endpoints:

/courses POST – It will save a course record and return 201 status.

/courses GET – It will return all items in the table if there are no query parameters. If there are query parameters, return records that match the query string criteria. Under the hood, it will use the "scan" function with/without "filter-expression".

/courses/{courseCode} GET – It will return the items that match the "courseCode" provided in the path param. Here, you will practice path parameters in API Gateway and the "query" function in DynamoDB.

/courses PATCH – Updates an existing item.

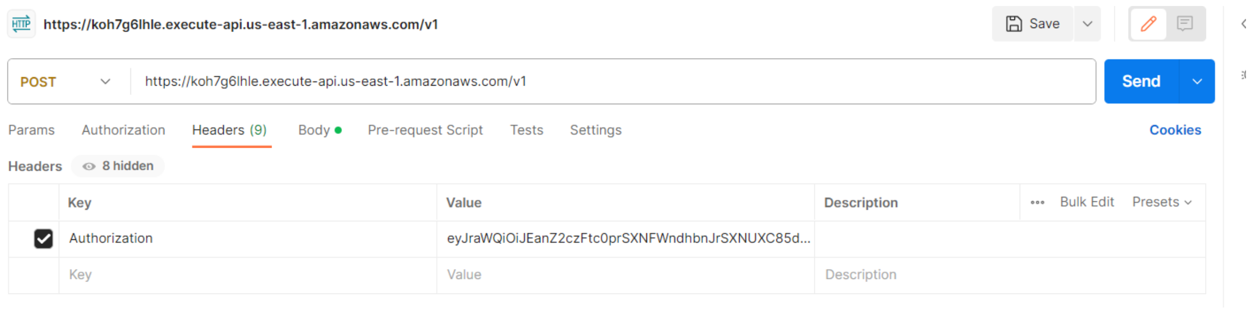
You can call different CRUD operations based on the path and method like this:

```
const routeKey = `${event.httpMethod} ${event.resource}`;

switch (routeKey) {
  case 'GET /courses': {
    ...
  }
  case 'POST /courses': {

    ...
  }
  case 'GET /courses/{userId}': {

    ...
  }
```

- Create a **Cognito User pool** for the app. The main configs are, enabling the hosted UI to get tokens and selecting IMPLICIT GRANT. Step by step:
  - Select only "email" in Step 1
  - Select No MFA in Step 2
  - Nothing to do in Step 3
  - Select Send email with Cognito in Step 4
  - In step 5 you will configure multiple things.
    - Give user pool name
    - select Use the Cognito Hosted UI
    - Give any domain prefix and give the app client name
    - Give http://localhost:3000 in callback URLs
    - Expand Advanced app client settings and scroll down and select **Implicit Grant** and remove Authorization code grant in OAuth 2.0 grant types
- Secure the API using tokens from the Cognito User pool.
  - Go to Authorizers in the API
  - Give it a name. Select Cognito as Authorizer type. Select to user pool. The token source is "Authorization".
  - Go to the resources. Click on the method and click on the "Method request". Click "Edit" in the Method request settings. In the Authorization select the authorizer. **Deploy** the API.

Test the API

1. Go to the Cognito pool and click on the "App integration". Scroll all way down and click on the app client. Click on the "`View Hosted UI`" in the hosted UI section. Sign up and enter the code in your email. It will redirect you the callback URL you entered and in the URL you will see **id_token** as a hash parameter.
2. Copy the **ID Token**. Provide it in the header as **Authorization**.


https://koh7g6lhle.execute-api.us-east-1.amazonaws.com/v1

POST https://koh7g6lhle.execute-api.us-east-1.amazonaws.com/v1

	Key	Value	Description
✓	Authorization	eyJraWQiOiJEanZ2czFtc0prSXNFWndhbnJrSXNUXC85d...	
	Key	Value	Description

Common errors

- 403 Forbidden - You are hitting an endpoint that is not defined in the resource. For example, you only have a GET endpoint defined in the API Gateway but trying to access the POST endpoint.
- 502 Bad Gateway – Your lambda should return a proper JSON that API Gateway understands.