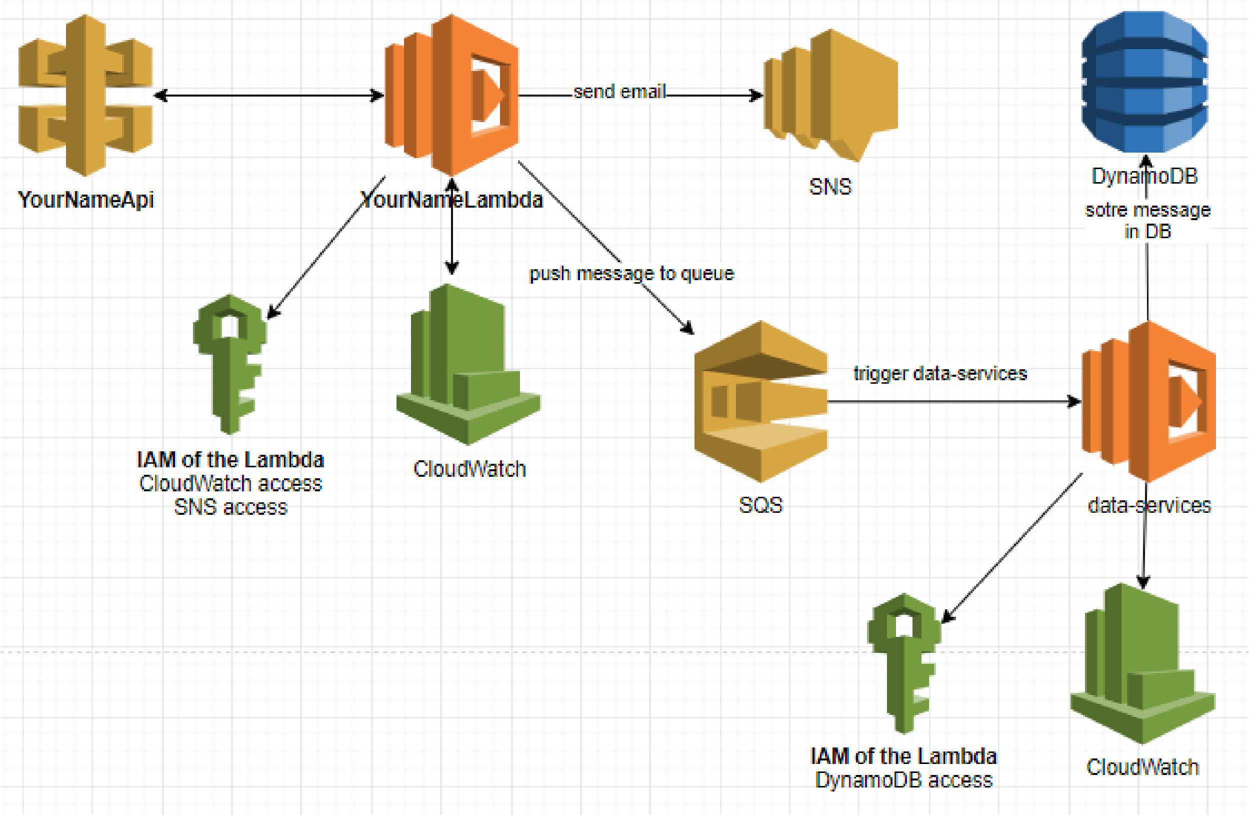
Assignment 13 – Securing an API with Cognito

No late and **no email** submission. Even if you have done it, if there no submission, it is 0.

Forums – If you have trouble understanding the concept, do some research. Still, confused? write how much progress you have made so far then ask a question in the Forums section. **Plus 2** scores on the exam if you asked a good question or answered correctly.

Bonus task with the highest score

- When an API gateway sends a “GET course by id” request from your client to the lambda, extract query parameters, path parameter and publish it to the SNS topic that emails those extracted data.
- Write a lambda that sends a message to the SQS and SNS. Write another lambda that listens to the queue and stores the message. Below is the architecture diagram but you don’t need to deplay the first lambda behind API gateway.



Hands-on lab

1. Create a user pool for the Course API in AWS Cognito.
    a. Go to Cognito -> click on **Manage User Pools** -> Top right corner, click on **Create a user pool**.
    b. In **Name** section, **CourseUserPool** as Pool name. Click on **Step through settings.**
    c. In **Attributes section,** Select **Email address or phone number.** In **Which standard attributes do you want to require?,** check **email** and **name.** Click on **Next step.**

d. In **Policies** section, nothing to change. Click on **Next step.**

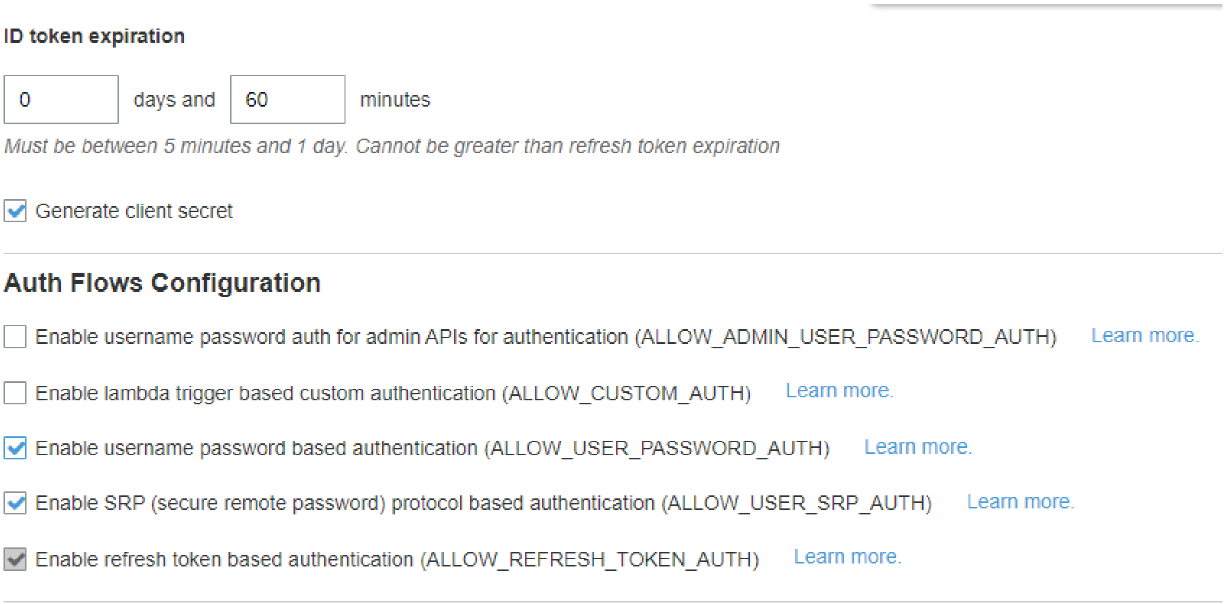
e. In **MFA and verifications** section, nothing to change. Click on **Next step.**

f. In **Message customization** section, nothing to change. Click on **Next step.**

g. In **Tags** section, nothing to change. Click on **Next step.**

h. In **Devices** section, nothing to change. Click on **Next step**

i. In **App Clients** section, click on **Add an app client. CourseApiClient** as **App client name.** Uncheck **Generate client secret.** Uncheck **Enable lambda trigger based custom authentication.** Check **Enable username password based authentication.** Then click on **Create app client.**



ID token expiration

0 days and 60 minutes

Must be between 5 minutes and 1 day. Cannot be greater than refresh token expiration

☑ Generate client secret

Auth Flows Configuration

□ Enable username password auth for admin APIs for authentication (ALLOW_ADMIN_USER_PASSWORD_AUTH) Learn more.

□ Enable lambda trigger based custom authentication (ALLOW_CUSTOM_AUTH) Learn more.

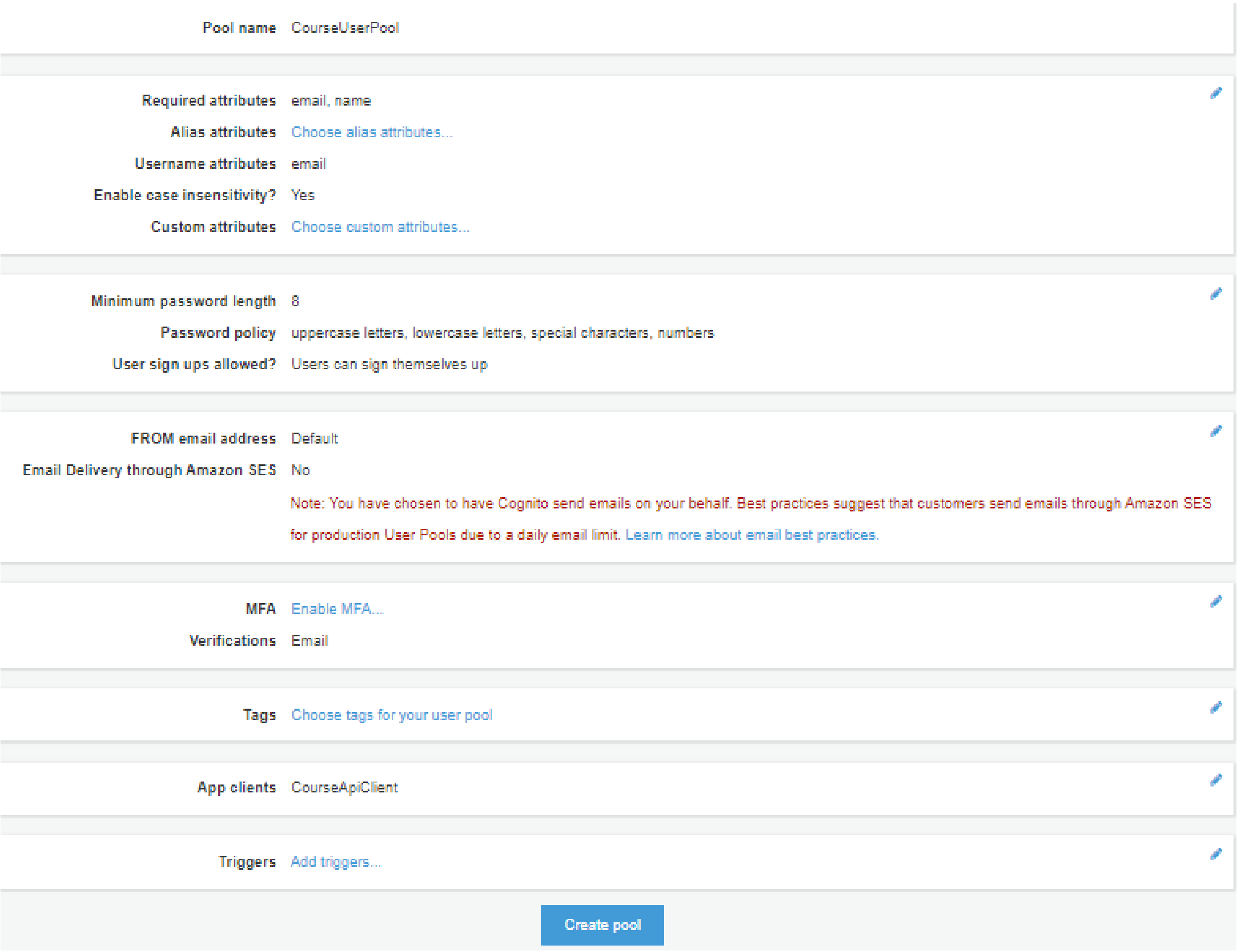
☑ Enable username password based authentication (ALLOW_USER_PASSWORD_AUTH) Learn more.

☑ Enable SRP (secure remote password) protocol based authentication (ALLOW_USER_SRP_AUTH) Learn more.

☑ Enable refresh token based authentication (ALLOW_REFRESH_TOKEN_AUTH) Learn more.

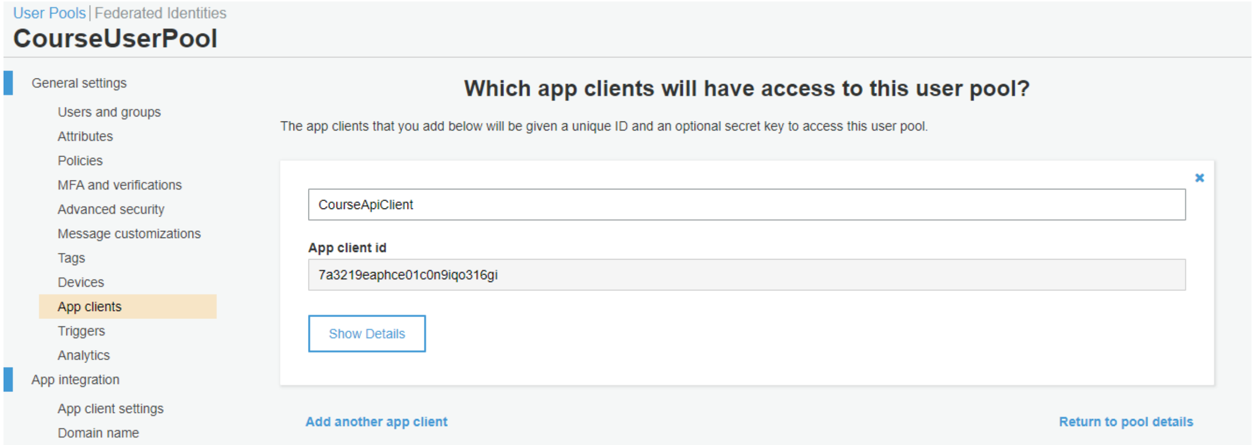
j. Click on **return to pool details.**

k. Hit **Create pool**.



Pool name	CourseUserPool
Required attributes	email, name
Alias attributes	Choose alias attributes...
Username attributes	email
Enable case insensitivity?	Yes
Custom attributes	Choose custom attributes...
Minimum password length	8
Password policy	uppercase letters, lowercase letters, special characters, numbers
User sign ups allowed?	Users can sign themselves up
FROM email address	Default
Email Delivery through Amazon SES	No
	Note: You have chosen to have Cognito send emails on your behalf. Best practices suggest that customers send emails through Amazon SES for production User Pools due to a daily email limit. Learn more about email best practices.
MFA	Enable MFA...
Verifications	Email
Tags	Choose tags for your user pool
App clients	CourseApiClient
Triggers	Add triggers...

2. Create a user in your user pool via AWS CLI.
    a. Grab your tokens from AWS Academy. There is **AWS CLI,** hit **Show**.
    b. Copy and paste the token into ~/.aws/credentials.
    c. Go to your user pool and click on **App clients** in the left sidebar. Copy the **App client id**.



Which app clients will have access to this user pool?

The app clients that you add below will be given a unique ID and an optional secret key to access this user pool.

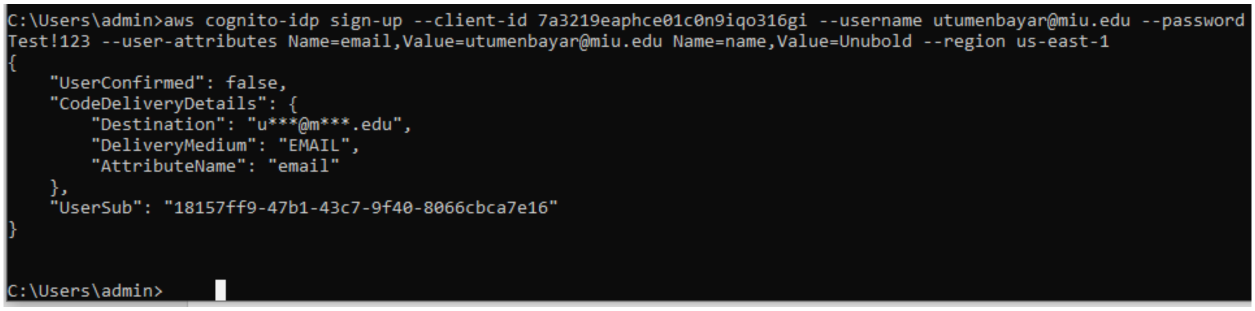
CourseApiClient

App client id

7a3219eaphce01c0n9iqo316gi

    d. Run the command below to create a user in your pool. Replace app_client_id, your_first_name and your_email accordingly.

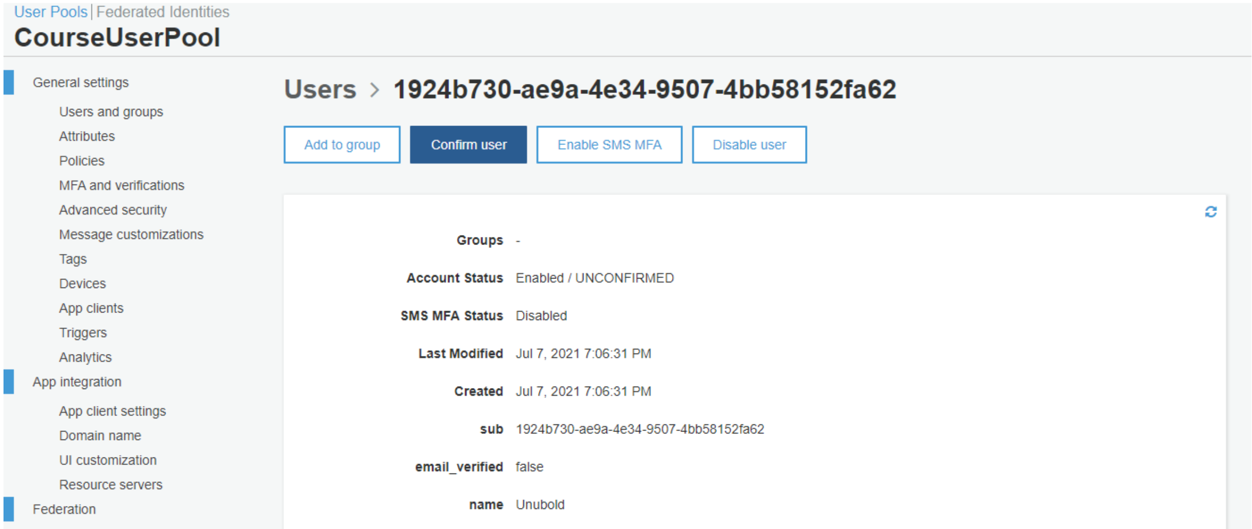
```
aws cognito-idp sign-up --client-id <<app_client_id>> --username <<your_email>> --
password Test123# --user-attributes Name=email,Value=<<your_email>>
Name=name,Value=<<your_first_name>> --region us-east-1
```



```
C:\Users\admin>aws cognito-idp sign-up --client-id 7a3219eaphce01c0n9iqo316gi --username utumenbayar@miu.edu --password
Test!123 --user-attributes Name=email,Value=utumenbayar@miu.edu Name=name,Value=Unubold --region us-east-1
{
    "UserConfirmed": false,
    "CodeDeliveryDetails": {
        "Destination": "u***@m***.edu",
        "DeliveryMedium": "EMAIL",
        "AttributeName": "email"
    },
    "UserSub": "18157ff9-47b1-43c7-9f40-8066cbca7e16"
}


C:\Users\admin>
```

e. Go to your user pool and click on **Users and groups** in the left sidebar. Hit refresh icon on top right corner. That will pull the newly-created user. Click on the username which is UUID hyperlink. Click on **Confirm user** button.



Groups -

Account Status Enabled / UNCONFIRMED

SMS MFA Status Disabled

Last Modified Jul 7, 2021 7:06:31 PM

Created Jul 7, 2021 7:06:31 PM

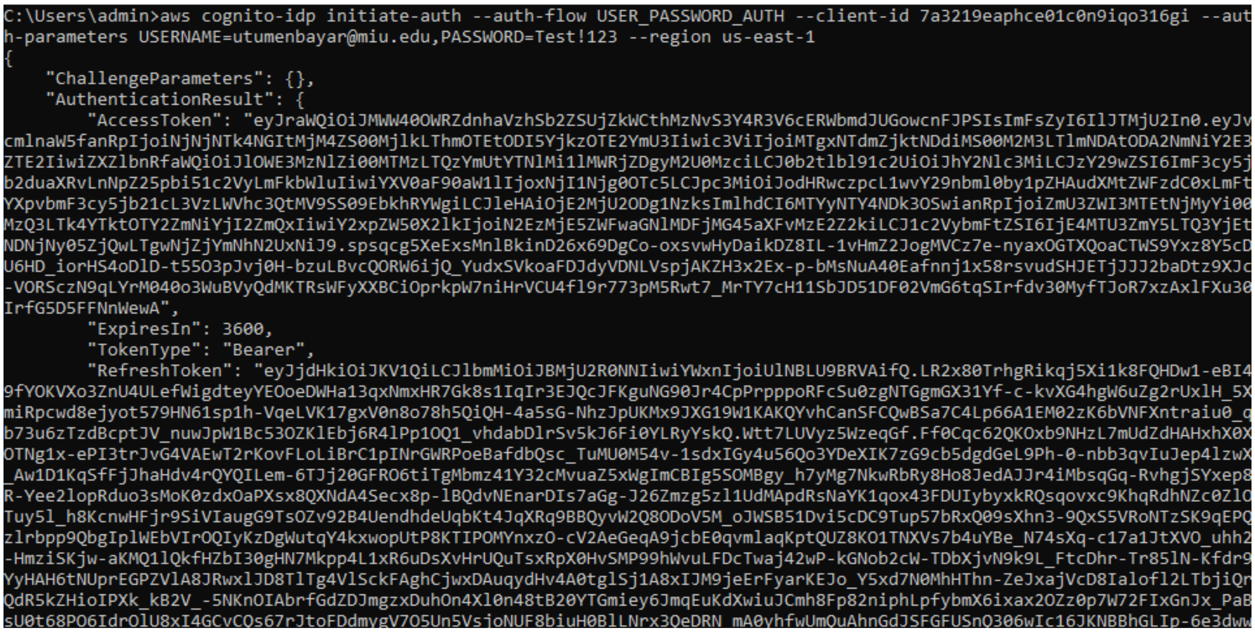
sub 1924b730-ae9a-4e34-9507-4bb58152fa62

email_verified false

name Unubold

f. Execute the command below that returns token associated with the user. That you need to provide after securing the API to store and retrieve data from the back-end or lambda. You may need to re-execute this command to get the new tokens in case it expired.

```
aws cognito-idp initiate-auth --auth-flow USER_PASSWORD_AUTH --client-id
<<app_client_id>> --auth-parameters USERNAME=<<your_email>>,PASSWORD=Test123# --
region us-east-1
```

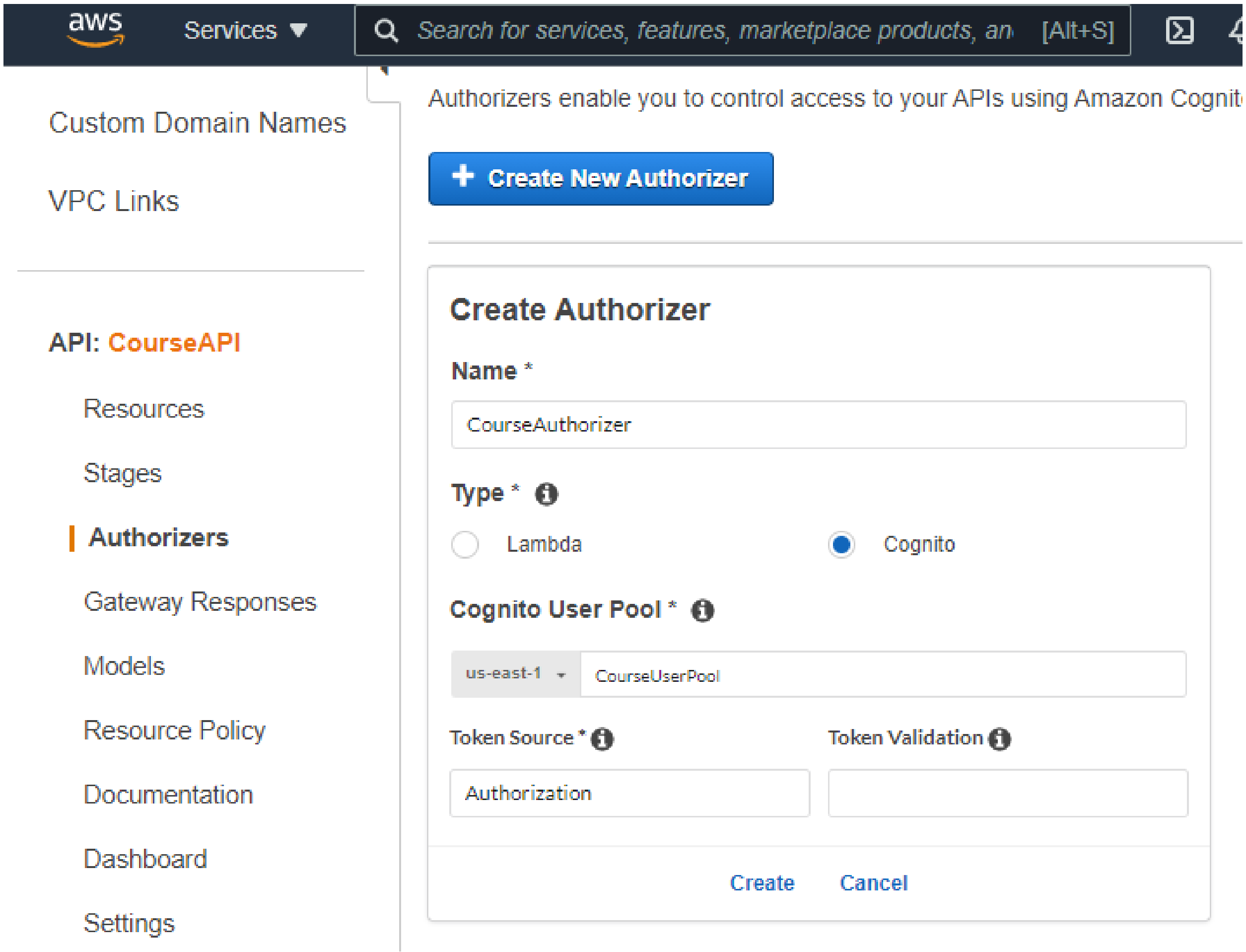


```
C:\Users\admin>aws cognito-idp initiate-auth --auth-flow USER_PASSWORD_AUTH --client-id 7a3219eaphce01c0n9iqo316gi --aut
h-parameters USERNAME=utumenbayar@miu.edu,PASSWORD=Test!123 --region us-east-1
{
    "ChallengeParameters": {},
    "AuthenticationResult": {
        "AccessToken": "eyJraWQiOiJMWW40OWRZdnhaVzhSb2ZSUjZkWCthMzNvS3Y4R3V6cERWbmdJUGowcnFJPSIsImFsZyI6IlJTMjU2In0.eyJv
cmlnaW5fanRpIjoiNjNjNTk4NGItMjM4ZS00MjlkLThmOTEtODI5YjkzOTE2YmU3Iiwic3ViIjoiMTgxNTdmZjktNDdiMS00M2M3LTlmNDAtODA2NmNiY2E3
ZTE2IiwiZXZlbnRfaWQiOiJlOWE3MzNlZi00MTMzLTQzYmUtYTNlMi1lMWRjZDgyM2U0MzciLCJ0b2tlbl91c2UiOiJhY2Nlc3MiLCJzY29wZSI6ImF3cy5j
b2duaXRvLnNpZ25pbi51c2VyLmFkbWluIiwiYXV0aF90aW1lIjoxNjI1Njg0OTc5LCJpc3MiOiJodHRwczpcL1wvY29nbml0by1pZHAudXMtZWFzdC0xLmFt
YXpvbmF3cy5jb21cL3VzLWVhc3QtMV9SS09EbkhRYWgiLCJleHAiOjE2MjU2ODg1NzksImlhdCI6MTYyNTY4NDk3OSwianRpIjoiZmU3ZWI3MTEtNjMyYi00
MzQ3LTk4YTktOTY2ZmNiYjI2ZmQxIiwiY2xpZW50X2lkIjoiN2EzMjE5ZWFwaGNlMDFjMG45aXFvMzE2Z2kiLCJ1c2VybmFtZSI6IjE4MTU3ZmY5LTQ3YjEt
NDNjNy05ZjQwLTgwNjZjYmNhN2UxNiJ9.spsqcg5XeExsMnlBkinD26x69DgCo-oxsvwHyDaikDZ8IL-1vHmZ2JogMVCz7e-nyaxOGTXQoaCTWS9Yxz8Y5cD
U6HD_iorHS4oDlD-t55O3pJvj0H-bzuLBvcQORW6ijQ_YudxSVkoaFDJdyVDNLVspjAKZH3x2Ex-p-bMsNuA40Eafnnj1x58rsvudSHJETjJJJ2baDtz9XJc
-VORSczN9qLYrM040o3WuBVyQdMKTRsWFyXXBCiOprkpW7niHrVCU4fl9r773pM5Rwt7_MrTY7cH11SbJD51DF02VmG6tqSIrfdv30MyfTJoR7xzAxlFXu30
IrfG5D5FFNnWewA",
        "ExpiresIn": 3600,
        "TokenType": "Bearer",
        "RefreshToken": "eyJjdHkiOiJKV1QiLCJlbmMiOiJBMjU2R0NNIiwiYWxnIjoiUlNBLU9BRVAifQ.LR2x80TrhgRikqj5Xi1k8FQHDw1-eBI4
9fYOKVXo3ZnU4ULefWigdteyYEOoeDWHa13qxNmxHR7Gk8s1IqIr3EJQcJFKguNG90Jr4CpPrppoRFcSu0zgNTGgmGX31Yf-c-kvXG4hgW6uZg2rUxlH_5X
miRpcwd8ejyot579HN61sp1h-VqeLVK17gxV0n8o78h5QiQH-4a5sG-NhzJpUKMx9JXG19W1KAKQYvhCanSFCQwBSa7C4Lp66A1EM02zK6bVNFXntraiu0_q
b73u6zTzdBcptJV_nuwJpW1Bc53OZKlEbj6R4lPp1OQ1_vhdabDlrSv5kJ6Fi0YLRyYskQ.Wtt7LUVyz5WzeqGf.Ff0Cqc62QKOxb9NHzL7mUdZdHAHxhX0X
OTNg1x-ePI3trJvG4VAEwT2rKovFLoLiBrC1pINrGWRPoeBafdbQsc_TuMU0M54v-1sdxIGy4u56Qo3YDeXIK7zG9cb5dgdGeL9Ph-0-nbb3qvIuJep4lzwX
_Aw1D1KqSfFjJhaHdv4rQYQILem-6TJj20GFRO6tiTgMbmz41Y32cMvuaZ5xWgImCBIg5SOMBgy_h7yMg7NkwRbRy8Ho8JedAJJr4iMbsqGq-RvhgjSYxep8
R-Yee2lopRduo3sMoK0zdxOaPXsx8QXNdA4Secx8p-lBQdvNEnarDIs7aGg-J26Zmzg5zl1UdMApdRsNaYK1qox43FDUIybyxkRQsqovxc9KhqRdhNZc0ZlO
Tuy5l_h8KcnwHFjr9SiVIaugG9TsOZv92B4UendhdeUqbKt4JqXRq9BBQyvW2Q8ODoV5M_oJWSB51Dvi5cDC9Tup57bRxQ09sXhn3-9QxS5VRoNTzSK9qEPQ
zlrbpp9QbgIplWEbVIrOQIyKzDgWutqY4kxwopUtP8KTIPOMYnxzO-cV2AeGeqA9jcbE0qvmlaqKptQUZ8KO1TNXVs7b4uYBe_N74sXq-c17a1JtXVO_uhh2
-HmziSKjw-aKMQ1lQkfHZbI30gHN7Mkpp4L1xR6uDsXvHrUQuTsxRpX0HvSMP99hWvuLFDcTwaj42wP-kGNob2cW-TDbXjvN9k9L_FtcDhr-Tr85lN-Kfdr9
YyHAH6tNUprEGPZVlA8JRwxlJD8TlTg4VlSckFAghCjwxDAuqydHv4A0tglSj1A8xIJM9jeErFyarKEJo_Y5xd7N0MhHThn-ZeJxajVcD8Ialofl2LTbjiQr
QdR5kZHioIPXk_kB2V_-5NKnOIAbrfGdZDJmgzxDuhOn4Xl0n48tB20YTGmiey6JmqEuKdXwiuJCmh8Fp82niphLpfybmX6ixax2OZz0p7W72FIxGnJx_PaB
sU0t68PO6IdrOlU8xI4GCvCQs67rJtoFDdmygV7O5Un5VsjoNUF8biuH0BlLNrx3QeDRN mA0yhfwUmQuAhnGdJSFGFUSnQ306wIc16JKNBBhGLIp-6e3dww
```

g. Store the tokens in file instead of priting it out in terminal. So you can copy and paste the tokens later on. The file will be stored in the directory that you are on right now. In this case, the file is stored on C:\Users\admin.

```
aws cognito-idp initiate-auth --auth-flow USER_PASSWORD_AUTH --client-id
<<app_client_id>> --auth-parameters USERNAME=<<your_email>>,PASSWORD=Test123# --
region us-east-1 > tokens.txt
```

3. Secure the POST endpoint.
    a. Go to API Gateway. Go to your API. Click on **Authorizers** in the left sidebar.Click on **Create New Authorizer.**
    b. Name as **CourserAuthorizer.** Type is **Cognito.** Select the user pool you created. **Token Source** is **Authorization**.



Authorizers enable you to control access to your APIs using Amazon Cognit

Create Authorizer

Name *: CourseAuthorizer

Type *: ○ Lambda ● Cognito

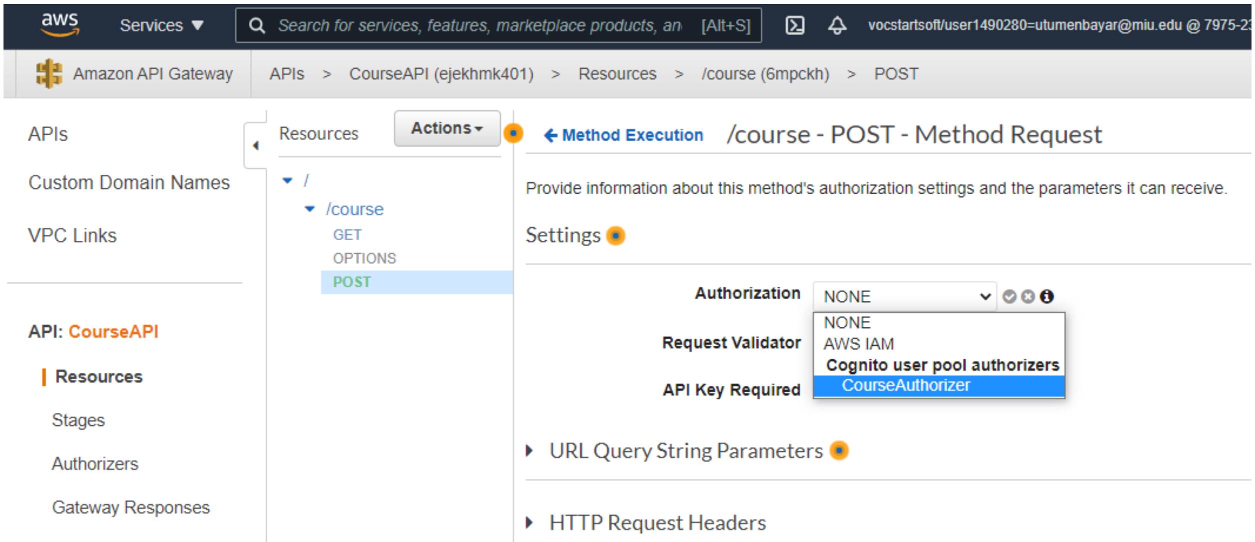
Cognito User Pool *: us-east-1 CourseUserPool

Token Source *: Authorization

Token Validation:

c. Go to **Resources.** Select the **POST** method under course resource.

d. Refresh the whole page. Click on **Method Request. Authorization** is the authorizer you just created. Click on OK icon.


aws
Services ▼
Search for services, features, marketplace products, an [Alt+S]
vocstartsoft/user1490280=utumenbayar@miu.edu @ 7975-2
Amazon API Gateway
APIs > CourseAPI (ejekhmk401) > Resources > /course (6mpckh) > POST
APIs
Custom Domain Names
VPC Links
API: CourseAPI
Resources
Stages
Authorizers
Gateway Responses
Resources
Actions ▾
/
/course
GET
OPTIONS
POST
← Method Execution

/course - POST - Method Request

Provide information about this method's authorization settings and the parameters it can receive.

Settings

Authorization: NONE
- NONE
- AWS IAM
- **Cognito user pool authorizers**
- CourseAuthorizer

Request Validator

API Key Required

URL Query String Parameters

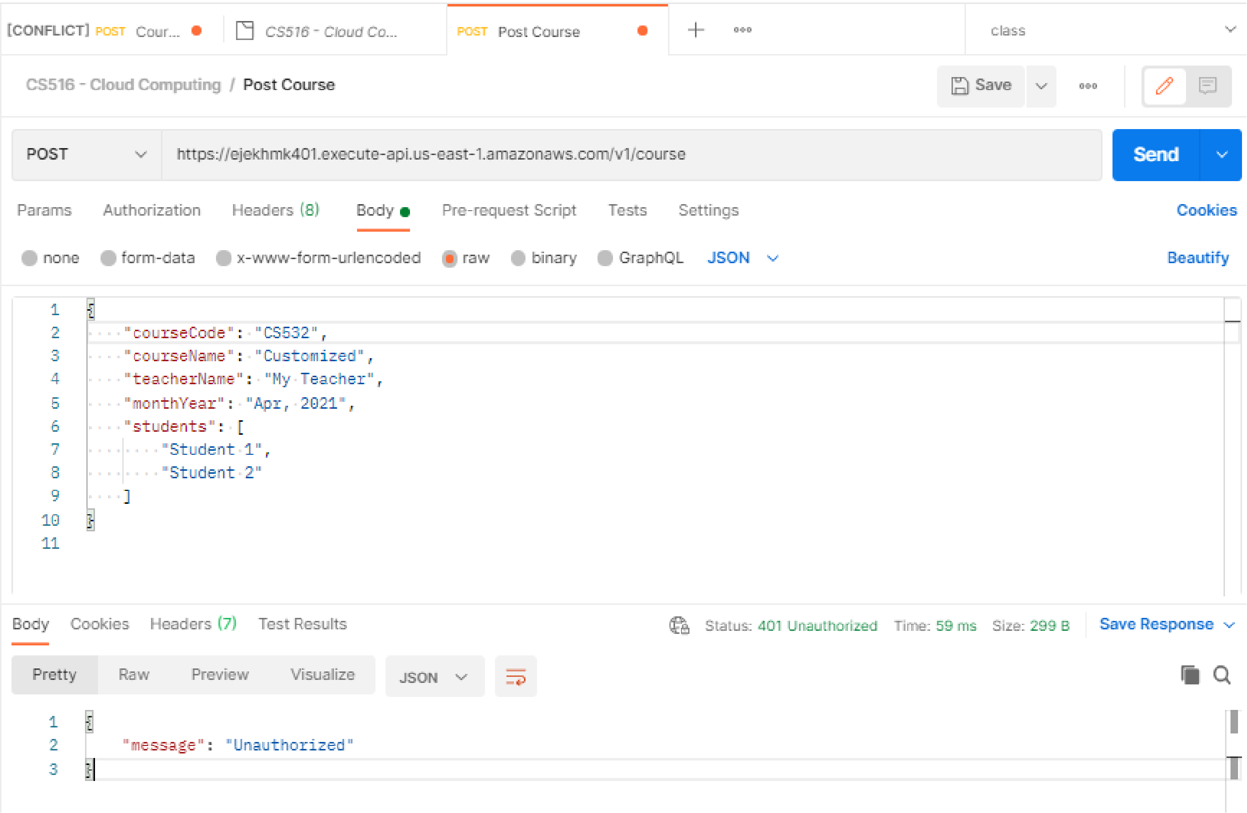
HTTP Request Headers

e. Secure the GET endpoint as well by using the authorizer you created earlier. Do the step c and d on the GET.

f. Actions -> Deploy API -> Go with the existing stage.

4. Test.

a. As see you below. Your endpoint is secured. You must provide the tokens that we generated in previous steps in Authorization header.



POST https://ejekhmk401.execute-api.us-east-1.amazonaws.com/v1/course

Send

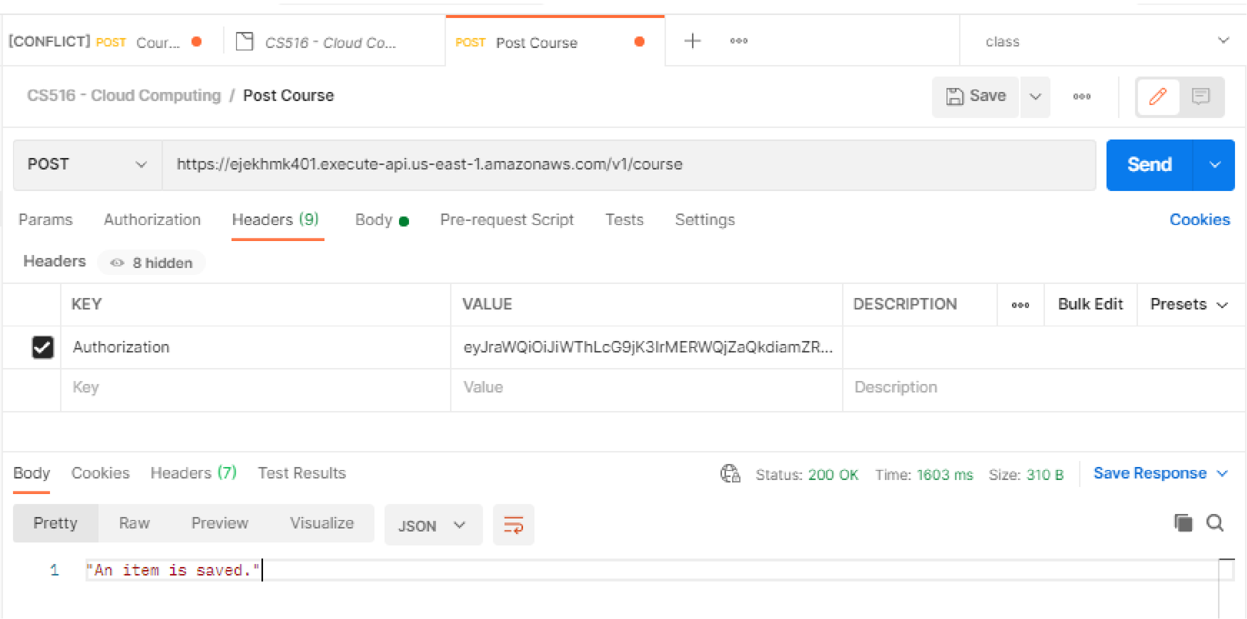
```
{
    "courseCode": "CS532",
    "courseName": "Customized",
    "teacherName": "My Teacher",
    "monthYear": "Apr, 2021",
    "students": [
        "Student 1",
        "Student 2"
    ]
}

```

Status: 401 Unauthorized Time: 59 ms Size: 299 B

```
{
    "message": "Unauthorized"
}
```

b. Copy the **ID Token**. Provide it in the header as **Authorization.**



CS516 - Cloud Computing / Post Course

POST https://ejekhmk401.execute-api.us-east-1.amazonaws.com/v1/course

	KEY	VALUE	DESCRIPTION
☑	Authorization	eyJraWQiOiJiWThLcG9jK3lrMERWQjZaQkdiamZR...	
	Key	Value	Description

Status: 200 OK Time: 1603 ms Size: 310 B

```
"An item is saved."
```

5. Submit screen shots.
    a. A request without token. Your API must return unauthorized.
    b. A request with token. You should be able to get all courses and save a course.