

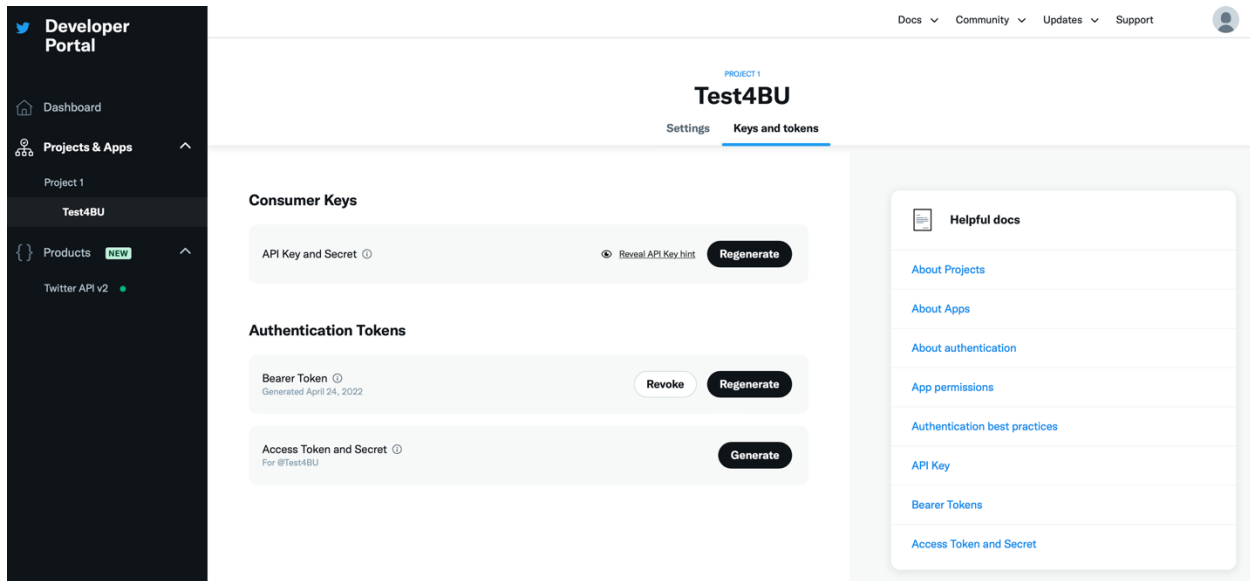
## API Lab

### Part I

Step 1: Create a twitter account

Step 2: Sign up for developer platform <https://developer.twitter.com/>

Step 3: Create an App



Step 4: Set up User Authentication Settings with the following parameters (next page)

The screenshot displays the Twitter Developer Portal interface. On the left is a dark sidebar with the 'Developer Portal' header and navigation links: 'Dashboard', 'Projects & Apps' (selected), 'Project 1', 'Test4BU' (selected), 'Products' (with a 'NEW' badge), and 'Twitter API v2'. The main content area has a top navigation bar with 'PROJECT 1', 'Test4BU', and tabs for 'Settings' (active) and 'Keys and tokens'. The 'App details' section includes an 'Edit' button and fields for 'NAME' (Test4BU), 'APP ID' (24080981), and 'APP ICON' (Twitter logo). A 'DESCRIPTION' field contains the text: 'This information will be visible to people who've authorized your App. This app was created to use the Twitter API.' Below this is the 'User authentication settings' section, which states 'Authentication not set up' and provides a 'Set up' button. To the right, an 'Authentication docs' panel lists 'Authentication methods' and 'v2 endpoints available with OAuth 2.0'.

Developer Portal

Dashboard

Projects & Apps

Project 1

Test4BU

Products **NEW**

Twitter API v2

PROJECT 1

Test4BU

Settings Keys and tokens

**App details** [Edit](#)

NAME  
Test4BU

APP ID  
24080981

APP ICON

DESCRIPTION  
This information will be visible to people who've authorized your App  
This app was created to use the Twitter API.

**User authentication settings**

Authentication not set up

OAuth 2.0 and OAuth 1.0a are authentication methods that allow users to sign in to your App with Twitter. They also allow your App to make specific requests on behalf of authenticated users. You can turn on one, or both methods.

[Set up](#)

**Authentication docs**

[Authentication methods](#)

[v2 endpoints available with OAuth 2.0](#)

# User authentication settings

The authentication method that's right for your App depends on the types of scopes (permissions) it will use. [Learn more by reading our authentication mapping doc.](#)

## OAuth 2.0 NEW



- Can be used with the Twitter API v2 only
- Allows you to pick specific scopes (also known as, permissions)

## OAuth 1.0a



- Can be used with Twitter API v1.1 and v2
- Uses broad authorization with coarse scopes

### OAUTH 2.0 SETTINGS

Type of App Automated App or bot

This type of App uses **confidential clients**, which securely authenticate with the authorization server. They keep your client secret safe.

### OAUTH 1.0A SETTINGS

**Request email from users** (optional)



To request email from users, you are required to provide URLs to your App's privacy policy and terms of service.

**App permissions**

**Request email from users** (optional)



To request email from users, you are required to provide URLs to your App's privacy policy and terms of service.

**App permissions**

- ☐ **Read**  
Read Tweets and profile information
- ☒ **Read and write**  
Read and Post Tweets and profile information
- ☐ **Read and write and Direct message**  
Read Tweets and profile information, read and post Direct messages

GENERAL AUTHENTICATION SETTINGS

**Callback URI / Redirect URL** ⓘ

https://localhost

+ Add another

**Website URL**

https://localhosttest.com

**Organization name** (optional)

**Organization URL** (optional)

https://

**Terms of service** (optional)

## Step 5: Save the following keys

The screenshot shows the Twitter Developer Portal interface. On the left is a dark sidebar with the 'Developer Portal' logo and navigation links: 'Dashboard', 'Projects & Apps' (selected), 'Project 1', 'Test4BU' (selected), 'Products' (with a 'NEW' badge), and 'Twitter API v2'. The main content area is titled 'Test4BU' with a 'PROJECT 1' label. Below the title are tabs for 'Settings' and 'Keys and tokens' (which is active). The 'Keys and tokens' section contains three main categories: 'Consumer Keys', 'Authentication Tokens', and 'OAuth 2.0 Client ID and Client Secret'. Under 'Consumer Keys', there is a field for 'API Key and Secret' with a 'Reveal API Key hint' icon and a 'Regenerate' button. Under 'Authentication Tokens', there is a 'Bearer Token' section with a 'Revoke' button and a 'Regenerate' button, and an 'Access Token and Secret' section with a 'Generate' button. Under 'OAuth 2.0 Client ID and Client Secret', there is a 'Client ID' field showing the value 'NihjRWU2Q0dPTUJjOR05xTjA4R2M6MTpjajQ' and a 'Client Secret' field with a 'Reveal Client Secret hint' icon and a 'Regenerate' button. On the right side of the page, there is a 'Helpful docs' sidebar with links to 'About Projects', 'About Apps', 'About authentication', 'App permissions', 'Authentication best practices', 'API Key', 'Bearer Tokens', and 'Access Token and Secret'.

**Developer Portal**

Dashboard

Projects & Apps

Project 1

Test4BU

Products **NEW**

Twitter API v2

**Test4BU** PROJECT 1

Settings **Keys and tokens**

**Consumer Keys**

API Key and Secret ⓘ [Reveal API Key hint](#) **Regenerate**

**Authentication Tokens**

Bearer Token ⓘ **Revoke** **Regenerate**  
Generated April 24, 2022

Access Token and Secret ⓘ **Generate**  
For @Test4BU

**OAuth 2.0 Client ID and Client Secret**

Client ID ⓘ NihjRWU2Q0dPTUJjOR05xTjA4R2M6MTpjajQ

Client Secret ⓘ [Reveal Client Secret hint](#) **Regenerate**

**Helpful docs**

- [About Projects](#)
- [About Apps](#)
- [About authentication](#)
- [App permissions](#)
- [Authentication best practices](#)
- [API Key](#)
- [Bearer Tokens](#)
- [Access Token and Secret](#)

Step 6: Make a POSTMAN request to test the key. Pay attention to all the details on this page! You need all of these settings as they appear.

Twitter API v2 / Manage Tweets / Create a Tweet

Save

POST

https://api.twitter.com/2/tweets

Send

Params

Authorization

Headers (10)

Body

Pre-request Script

Tests

Settings

Cookies

Type

OAuth 1.0

The authorization data will be automatically generated when you send the request.  
[Learn more about authorization](#)

Add authorization data to

Request He...

Heads up! These parameters hold sensitive data. To keep this data secure while working in a collaborative environment, we recommend using variables. [Learn more about variables](#)

Signature Method ⓘ

HMAC-SHA1

Consumer Key ⓘ

Consumer Secret ⓘ

Access Token

Token Secret ⓘ

ADVANCED

These are advanced configuration options. They are optional. Postman will auto generate values for some fields if left blank.

Callback URL ⓘ

Callback URL

Verifier ⓘ

Verifier

Timestamp

Timestamp

Nonce ⓘ

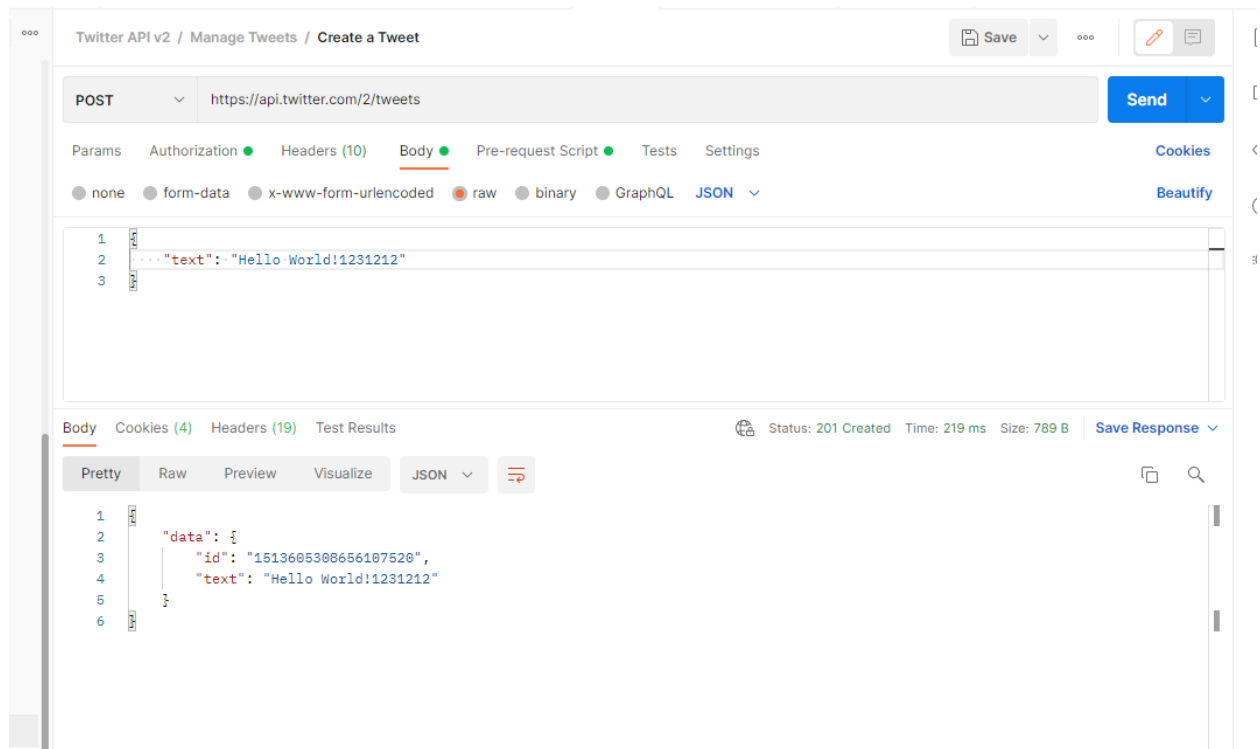
Nonce

Version

1.0

Realm ⓘ

testrealm@example.com

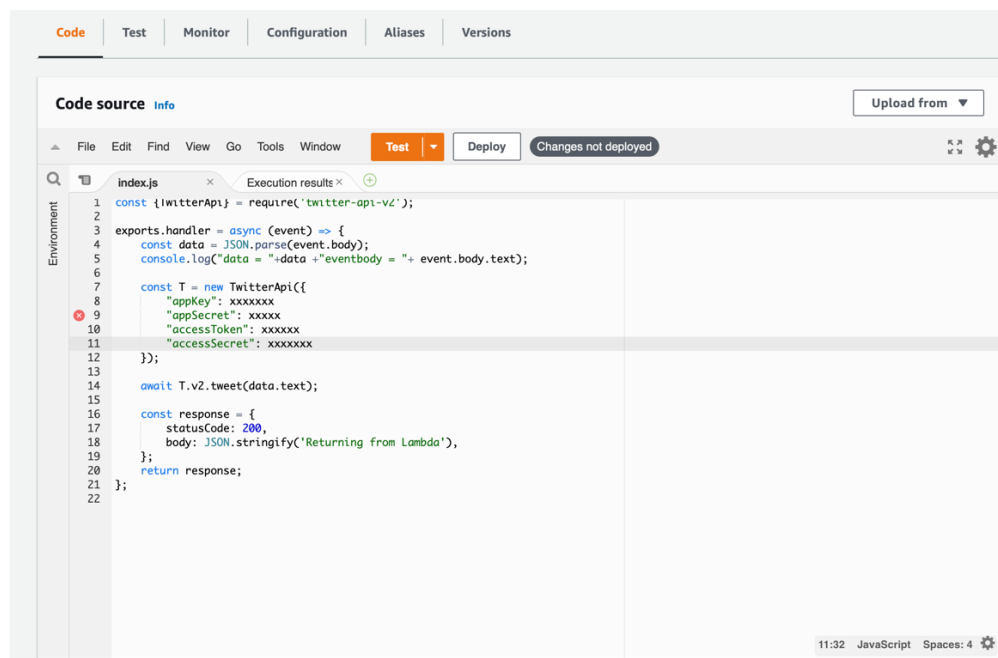


Step 7: Congratulations! Take a screenshot of your tweet.

## Part II

Step 1: On your AWS account, create a Lambda function from scratch. You can use the default settings when you create the Lambda function. When you get into the lambda function, use the Upload from button to upload the tweet.zip file that is attached to this assignment on Blackboard Learn

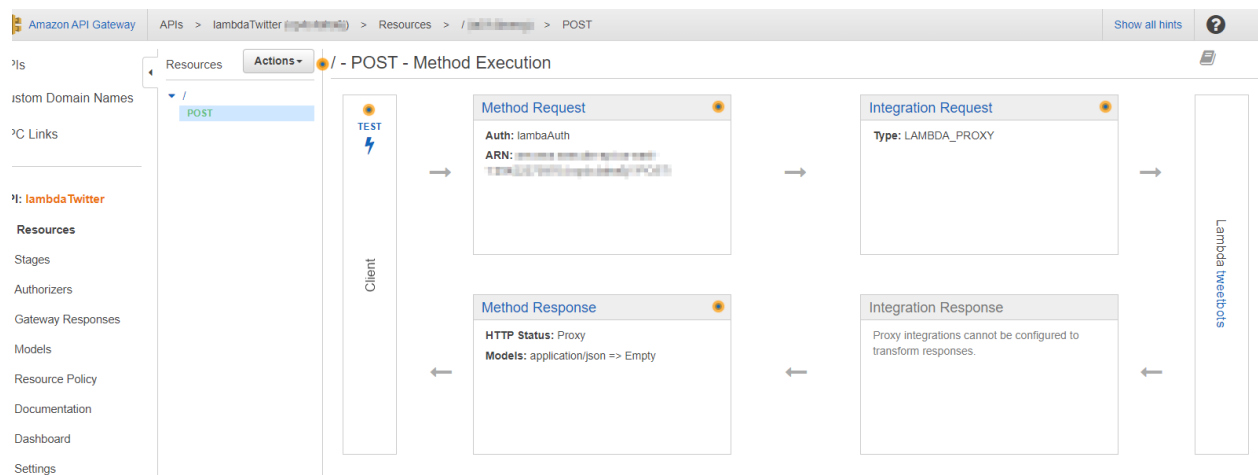
Step 2: Update the required fields <YOUR\_APP\_USER\_TOKEN> with the values you created when you signed up for a tweeter developer account.



The screenshot shows the AWS Lambda console's 'Code source' tab. The code is a JavaScript file named 'index.js' that uses the 'twitter-api-v2' library. It defines an asynchronous handler that takes an event, parses the body, and logs it. It then creates a new 'TwitterApi' instance with placeholder values for 'appKey', 'appSecret', 'accessToken', and 'accessSecret'. The handler calls 'T.v2.tweet(data.text)' and returns a response with status code 200 and a body stringifying 'Returning from Lambda'.

```
1 const {TwitterApi} = require('twitter-api-v2');
2
3 exports.handler = async (event) => {
4   const data = JSON.parse(event.body);
5   console.log("data = " + data + "eventbody = " + event.body.text);
6
7   const T = new TwitterApi({
8     "appKey": "xxxxxxx",
9     "appSecret": "xxxxxx",
10    "accessToken": "xxxxxxx",
11    "accessSecret": "xxxxxxx"
12  });
13
14  await T.v2.tweet(data.text);
15
16  const response = {
17    statusCode: 200,
18    body: JSON.stringify('Returning from Lambda'),
19  };
20  return response;
21 };
22
```

Step 3: Create an API Gateway for the above Lambda function by adding a trigger





Step 4: Click on Test and fill in the following string:

Request Body

```
1 {"text": "Hello from Lambda!"}
```

Hit the Test button, this should invoke your lambda function, and put a tweet out on your twitter feed, this time through lambda!

Please discuss the difference between all of the keys you were provided with, namely the API Key, API Key Secret, Bearer Token, Client ID, Client Secret, Access token, and Access token secret.