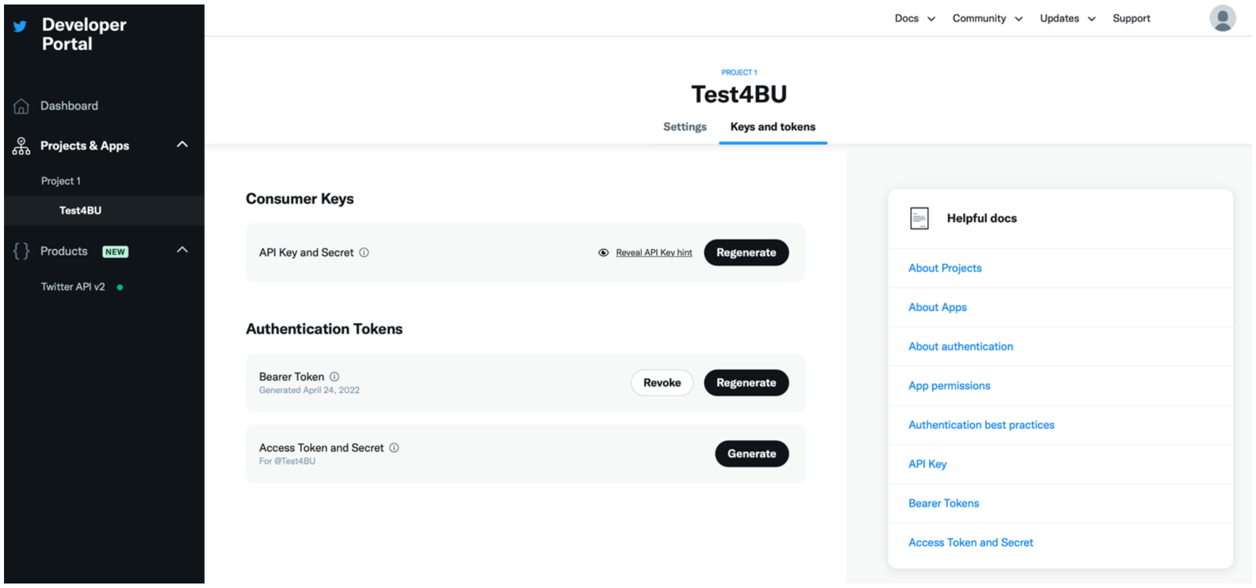
API Lab

Part I

Step 1: Create a twitter account

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

Step 3:Create an App



Consumer Keys

API Key and Secret — Reveal API Key hint — Regenerate

Authentication Tokens

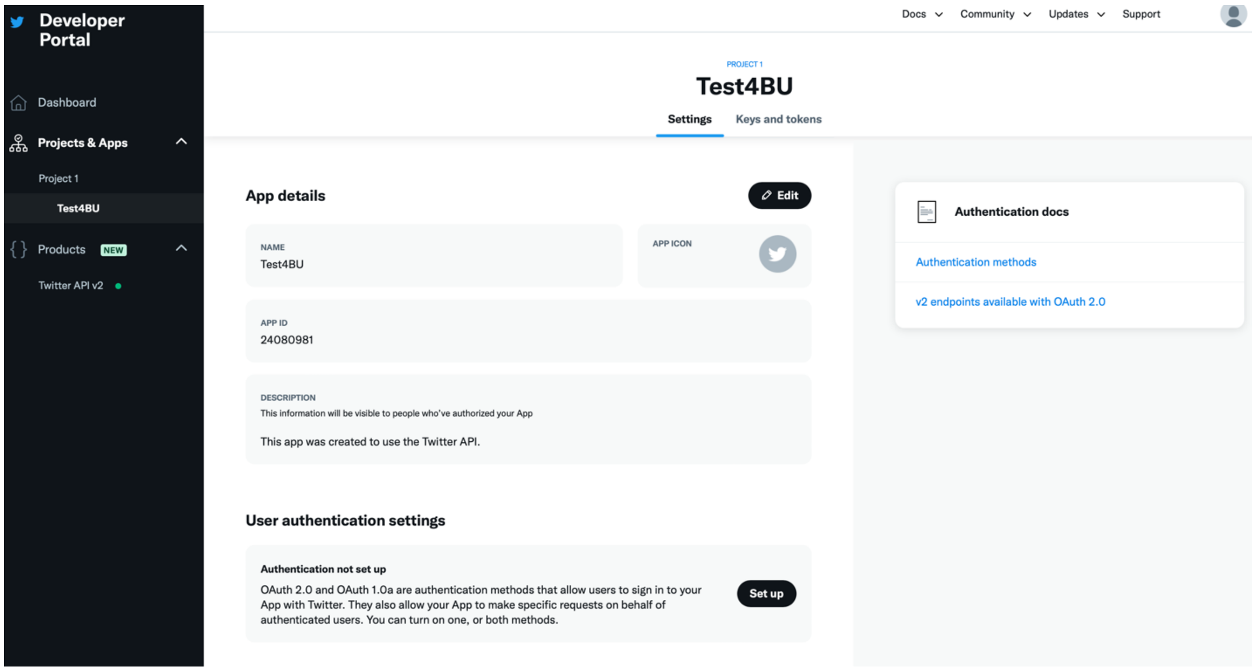
Bearer Token
Generated April 24, 2022 — Revoke — Regenerate

Access Token and Secret
For @Test4BU — Generate

Helpful docs

- About Projects
- About Apps
- About authentication
- App permissions
- Authentication best practices
- API Key
- Bearer Tokens
- Access Token and Secret

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



Test4BU

App details

NAME
Test4BU

APP ICON

APP ID
24080981

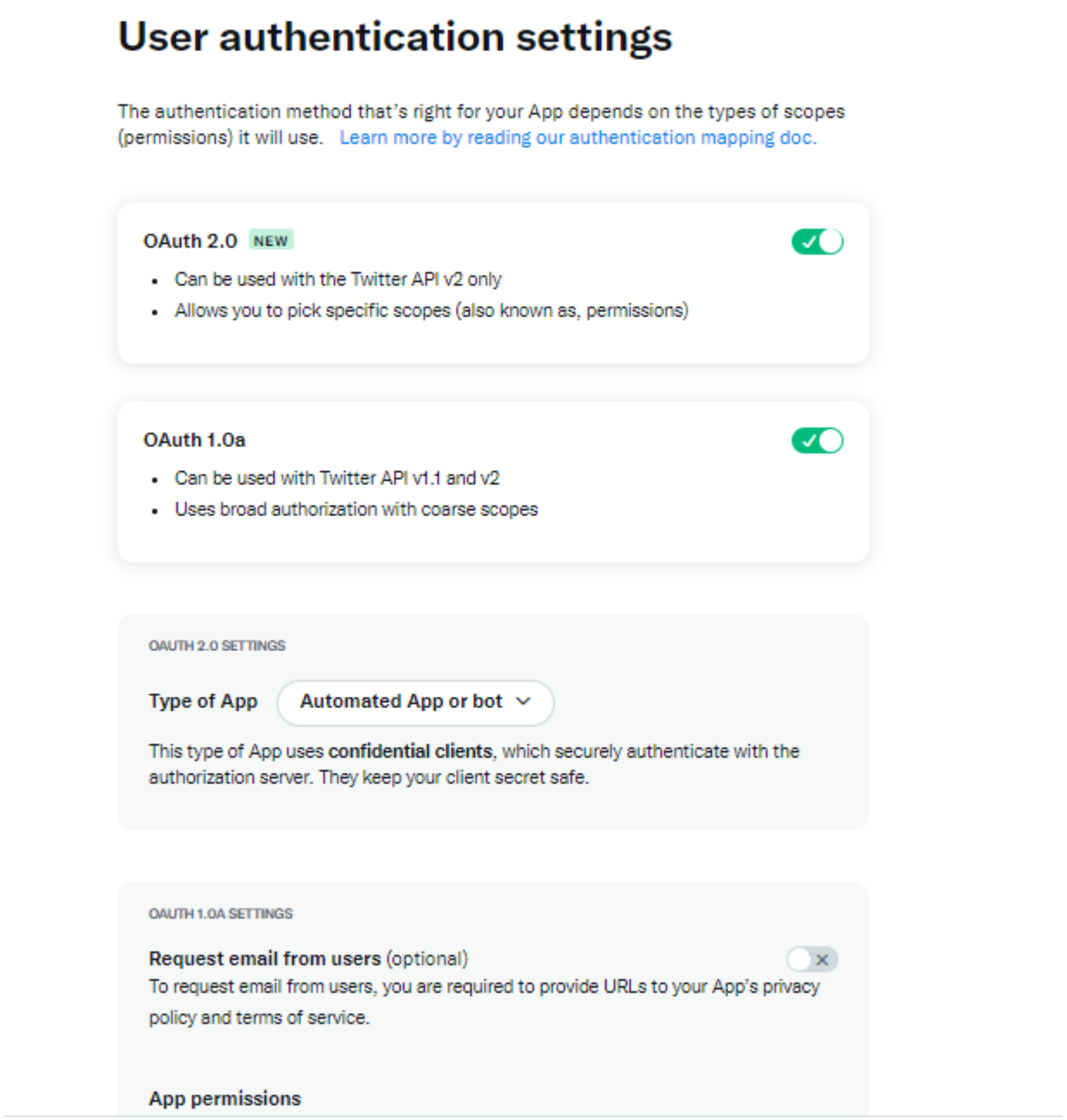
DESCRIPTION
This information will be visible to people who've authorized your App

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.



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

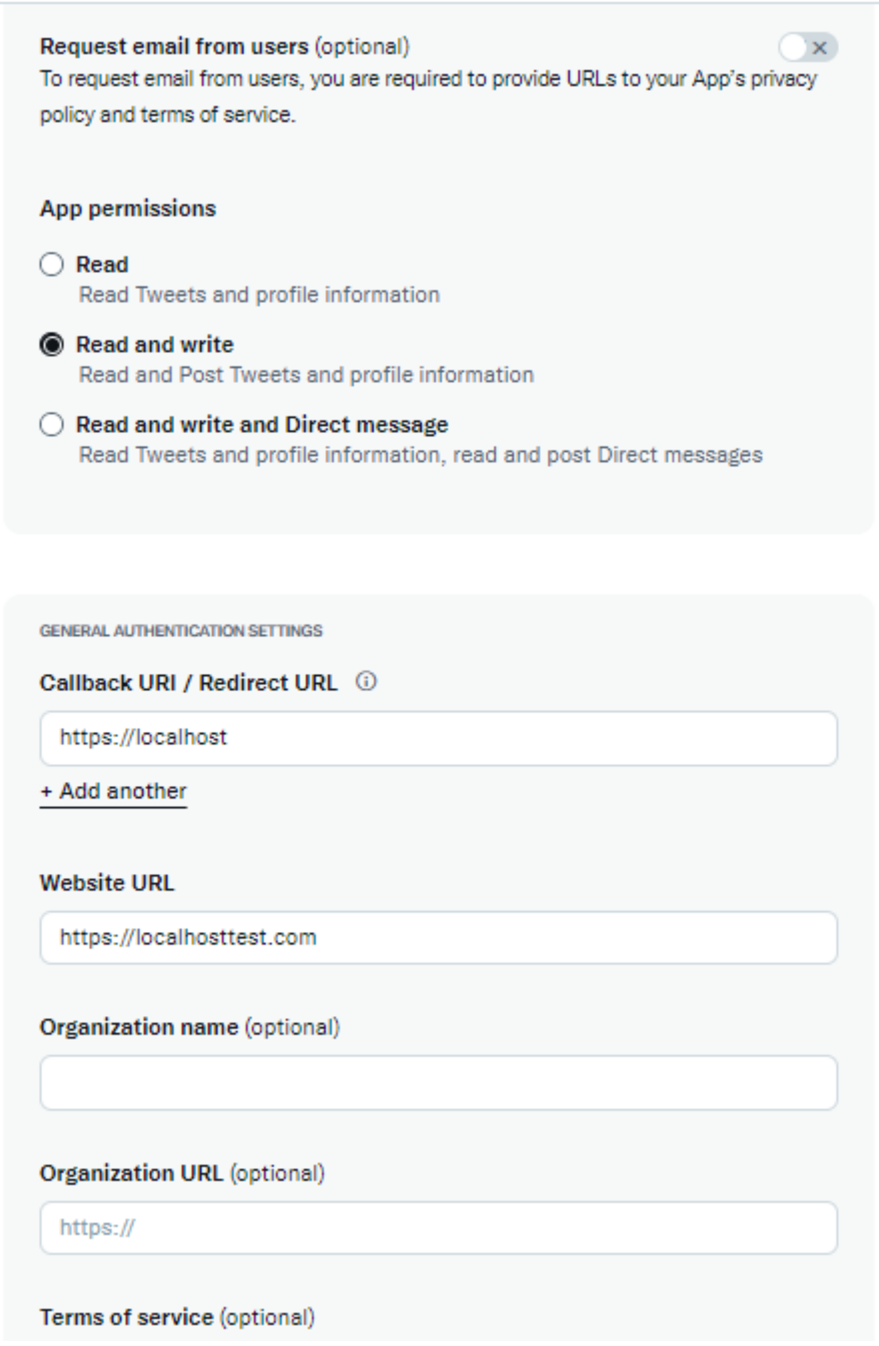
This type of App uses **confidential clients**, which securely authenticate with the authorization server. They keep your client secret safe.

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

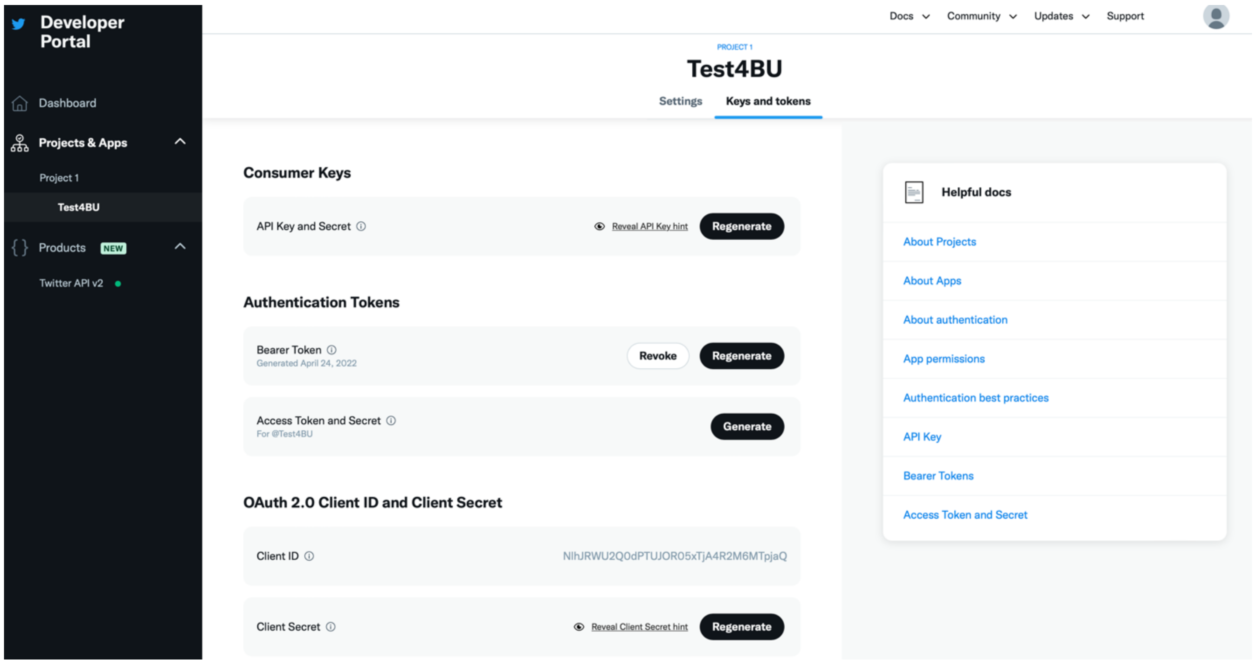
Organization name (optional)

Organization URL (optional)

https://

Terms of service (optional)

Step 5: Save the following keys



PROJECT 1

Test4BU

Settings | Keys and tokens

Consumer Keys

API Key and Secret — Reveal API Key hint — Regenerate

Authentication Tokens

Bearer Token — Generated April 24, 2022 — Revoke — Regenerate

Access Token and Secret — For @Test4BU — Generate

OAuth 2.0 Client ID and Client Secret

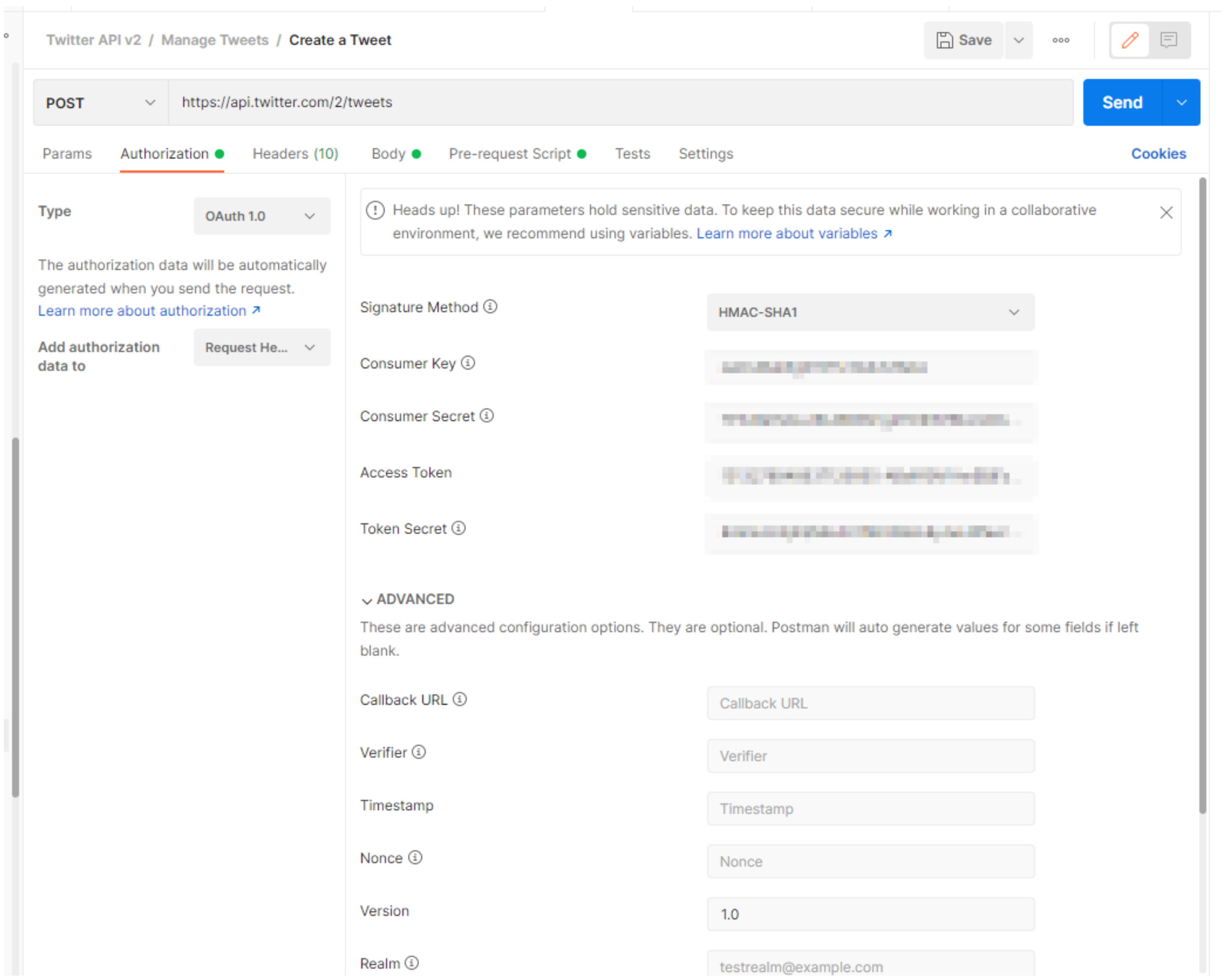
Client ID — NlhJRWU2Q0dPTUJOR05xTjA4R2M6MTpjaQ

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**

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

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...

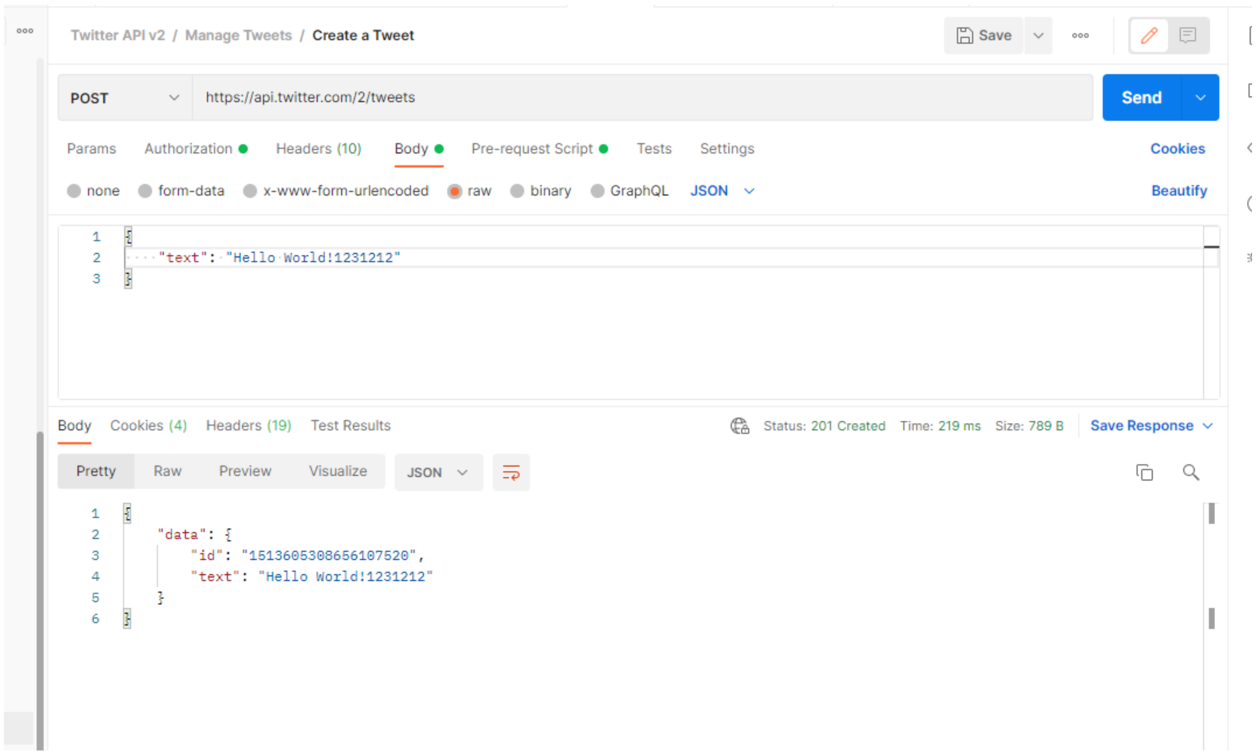
Heads up! These parameters hold sensitive data. To keep this data secure while working in a collaborative environment, we recommend using variables. Learn more about variables ↗

Signature Method	HMAC-SHA1
Consumer Key	[redacted]
Consumer Secret	[redacted]
Access Token	[redacted]
Token Secret	[redacted]

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



Twitter API v2 / Manage Tweets / Create a Tweet

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

```
{
    "text": "Hello World!1231212"
}
```

Status: 201 Created Time: 219 ms Size: 789 B

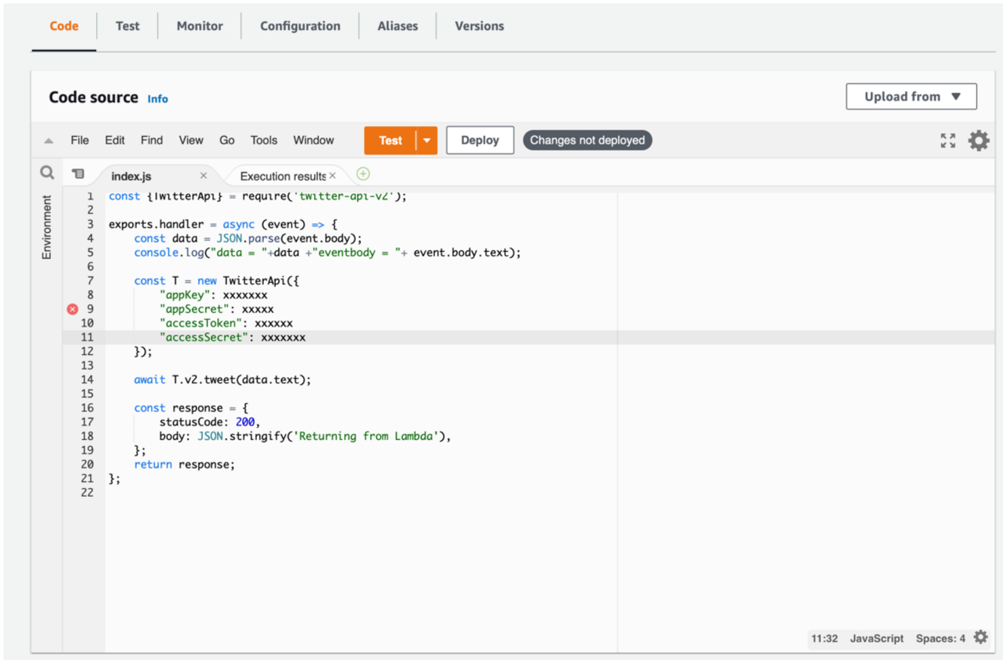
```
{
    "data": {
        "id": "1513605308656107520",
        "text": "Hello World!1231212"
    }
}
```

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

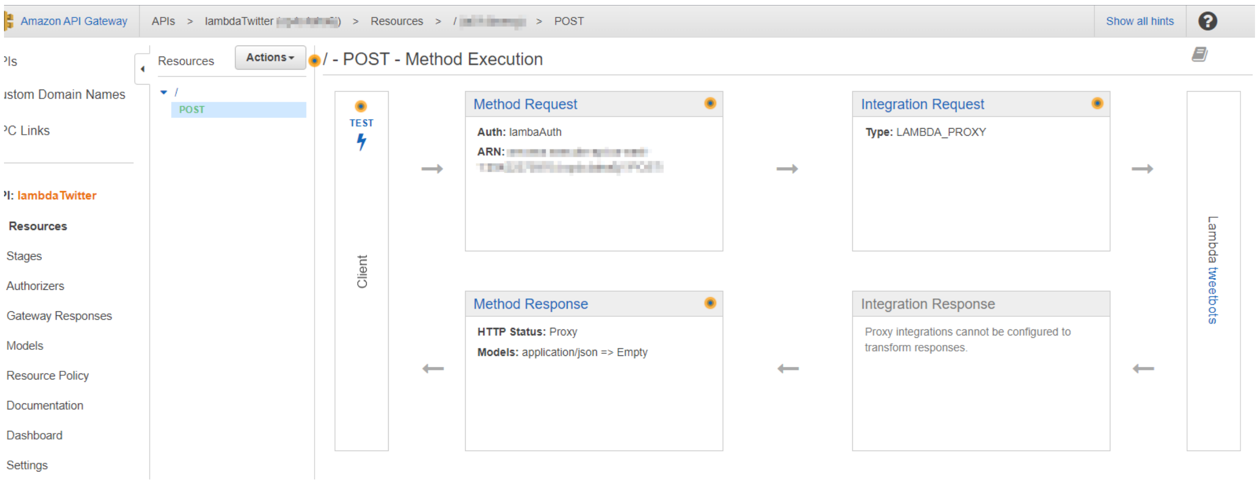
Step 2: Update the required fields `<YOUR_APP_USER_TOKEN>` with the values you created when you signed up for a tweeter developer account.



```
const {TwitterApi} = require('twitter-api-v2');

exports.handler = async (event) => {
    const data = JSON.parse(event.body);
    console.log("data = "+data +"eventbody = "+ event.body.text);

    const T = new TwitterApi({
        "appKey": xxxxxxx
        "appSecret": xxxxx
        "accessToken": xxxxxx
        "accessSecret": xxxxxxx
    });

    await T.v2.tweet(data.text);

    const response = {
        statusCode: 200,
        body: JSON.stringify('Returning from Lambda'),
    };
    return response;
};
```

11:32 JavaScript Spaces: 4

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



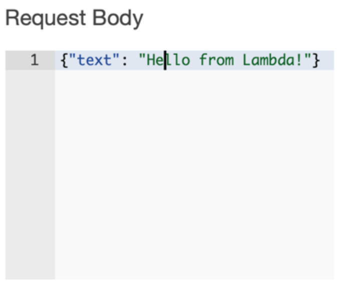
Auth: lambaAuth
ARN: [redacted]

Type: LAMBDA_PROXY

HTTP Status: Proxy
Models: application/json => Empty

Proxy integrations cannot be configured to transform responses.

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


Request Body

```
{"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.