Integration Guide

Implement the front-end code

Get started by connecting your front-end login button to the server endpoint. The following is an example in HTML:

```
<a href='{SERVER_ENDPOINT_OAUTH}'>Continue with TikTok</a>
```

Implement the server code to handle authorization grant flow

The server code must be responsible for the following:

- Ensuring that the client secret and refresh token are stored securely.
- Ensuring that the security for each user is protected by preventing request forgery attacks.
- Handling the refresh flow before access token expiry.
- Managing the access token request flow for each user.

Redirect request to TikTok's authorization server

Create an anti-forgery state token

You must prevent request forgery attacks to protect the security of your users. The first step before making the redirect request to TikTok's authorization server is to create a unique session token to maintain the state between the request and callback.

You will later match this unique session token with the authentication response to verify that the user is making the request and not a malicious attacker.

One of the simple approaches to a state token is a randomly generated alphanumeric string constructed using a random-number generator. For example:

```
let array = new Uint8Array(30);
const csrfState = window.crypto.getRandomValues(array);
```

Initial redirect to TikTok's authorization page

To make the initial redirect request to TikTok's authorization server, the following query parameters below must be added to the Authorization Page URL using the `application/x-www-form-urlencoded` format.

For example, you can use an online URL encoder to encode parameters. Select **UTF-8** as the destination character set.

Parameter	Type	Description
`client_key`	String	The unique identification key provisioned to the partner.
`scope`	String	A comma (,) separated string of authorization scope(s). These scope(s) are assigned to your application on the TikTok for Developers website. They handle what content your application can and cannot access. If a scope is toggleable, the user can deny access to one scope while granting access to others.
`redirect_uri`	String	The redirect URI that you requested for your application. It must match one of the redirect URIs you registered for the app.

`state`	String	The state is used to maintain the state of your request and callback. This value will be included when redirecting the user back to the client. Check if the state returned in the callback matches what you sent earlier to prevent cross-site request forgery. The state can also include customized parameters that you want TikTok service to return.
`response_type`	String	This value should always be set to `code`.
`disable_auto_auth`	int	Controls whether the authorization page is automatically presented to users. When set to 0, skips the authorization page for valid sessions. When set to 1, always displays the authorization page.

Redirect your users to the authorization page URL and supply the necessary query parameters. Note that the page can only be accessed through HTTPS.

Type	**Description**
URL	`https://www.tiktok.com/v2/auth/authorize/`
Query parameters	`client_key=<client_key>&response_type=code&scope=<scope>&redirect_uri=<redirect_uri>&state=<state>`

Note: If you are an existing client and use `https://www.tiktok.com/auth/authorize/` as the authorization page URL, please register a redirect URI for your app and migrate to the new URL mentioned above.

The following is an example using Node, Express, and JavaScript:

```
const express = require('express');
const app = express();
const fetch = require('node-fetch');
const cookieParser = require('cookie-parser');
const cors = require('cors');

app.use(cookieParser());
app.use(cors());
app.listen(process.env.PORT || 5000).

const CLIENT_KEY = 'your_client_key' // this value can be found in
app's developer portal

app.get('/oauth', (req, res) => {
    const csrfState = Math.random().toString(36).substring(2);
    res.cookie('csrfState', csrfState, { maxAge: 60000 });

    let url = 'https://www.tiktok.com/v2/auth/authorize/';

    // the following params need to be in
`application/x-www-form-urlencoded` format.
    url += '?client_key={CLIENT_KEY}';
    url += '&scope=user.info.basic';
    url += '&response_type=code';
    url += '&redirect_uri={SERVER_ENDPOINT_REDIRECT}';
    url += '&state=' + csrfState;

    res.redirect(url);
})
```

TikTok prompts a users to log in or sign up

The authorization page takes the user to the TikTok website if the user is not logged in. They are then prompted to log in or sign up for TikTok.

TikTok prompts a user for consent

After logging in or signing up, an authorization page asks the user for consent to allow your application to access your requested permissions.

Manage authorization response

If the user authorizes access, they will be redirected to `redirect_uri` with the following query parameters appended using `application/x-www-form-urlencoded` format:

Parameter	Type	Description
`code`	String	Authorization code that is used in getting an access token.
`scopes`	String	A comma-separated (,) string of authorization scope(s), which the user has granted.
`state`	String	A unique, non-guessable string when making the initial authorization request. This value allows you to prevent CSRF attacks by confirming that the value coming from the response matches the one you sent.

`error`	String	If this field is set, it means that the current user is not eligible for using third-party login or authorization. The partner is responsible for handling the error gracefully.
`error_description`	String	If this field is set, it will be a human-readable description about the error.

Manage access token

Using the `code` appended to your `redirect_uri`, you can obtain `access_token` for the user, which completes the flow for logging in with TikTok.