Financeit API Integration Guide: Implementing Authorization Code Flow with PKCE

Version: 1.0.1

Date: Nov 12, 2024

Overview

Which Flow Should You Use?

There are two authorization flows available for applications integrating with the Financeit API: the **authorization code flow** described in this document, and the **client credentials flow**. The authorization code flow allows the application to access resources on behalf of a user, while the client credentials flow is used for server-to-server interactions where no user is involved.

The appropriate flow to implement depends on whether your application is serving a single merchant or a broker with multiple merchants, and the API endpoints you require access to. See the below table for a breakdown of available actions in different scenarios:

Authorization Flow	**Client type**	**Scopes**		
		calculator, direct invites	***loans, partners***	***single_access_ links***
Authorization code	Any	Yes	Yes	Yes
Client credentials	Individual Merchant	Yes	Yes	No
Client credentials	Broker or Multi-location business	Yes	No	No

For more information on the scopes listed in the above table, please refer to the *Available Scopes* section. Please contact the Financeit team if you prefer to proceed with the client credentials flow.

Pre-requisites

This guide assumes that you have met with the Financeit team to review your API integration use case and have received our sandbox environment testing credentials. The testing credentials you should have received include:

- app_id: The test application's client ID
- app_secret: The test application's client secret
- login: Login for test partner account
- password: Password for test partner account

Additionally, you should have received the sandbox environment domain to be used in testing (eg. https://training.financeit.ca).

OAuth Authorization Code Flow Summary

When working with the Financeit API, your application will access Financeit on behalf of your users. With the authorization code flow, you will have each user sign into Financeit to grant your application permission to access their data.

Financeit uses the OAuth 2.0 authorization code flow to authorize users for your application which requires the following steps:

1. **Request User Authorization:** Users are redirected to Financeit to provide authorization for your application to access their Financeit data.
2. **Accept the Authorization Response***:* After the user approves the requested authorization, they will be redirected back to your application with an authorization code.
3. **Request an Access Token***:* Your application exchanges the authorization code for an access token that can be used to make secure calls to the Financeit API.

The implementation of the above steps are outlined in the remainder of this document.

Request User Authorization

Example request:

```
GET
https://www.financeit.ca/<locale>/partner/authorize-client?client_id=
CLIENT_ID
&response_type=code
&redirect_uri=https://client.app/callback
&scope=calculator
&state=STATE
&code_challenge_method=S256
&code_challenge=CODE_CHALLENGE
```

To request authorization from the user, a GET request must be made to the /authorize-client endpoint. This request should include the below parameters:

Query Parameter	**Required?**	**Description**
client_id	Required	The app ID you received from Financeit when registering.
response_type	Required	Set to “code”
redirect_uri	Required	The URL in your application where users will be sent after authorization. This must match the redirect URL submitted during registration.
scope	Required	A space separated list of scopes. See details below in the *Available Scopes* section.
state	Required	A unique string value of your choice that is hard to guess. Used to prevent CSRF. For example, *state=DCEeFWf45A53sdfKef424*
code_challenge	Required	A Base64-encoded SHA256 hash of the code_verifier. See details below in the *PKCE Extension* section.
code_challenge_method	Required	Set to “S256”

Available Scopes (scope field)

If no scopes are specified within the authorization request, authorization will be granted only for the calculator scope. If you find your application requires additional permissions, a new authorization request for a user may be initiated at any time. The available scopes are listed below:

Name	**Description**
api:calculator	Grants access to the calculator API endpoint which provides loan payment and rate estimates, and a link to a Financeit loan application. It also grants access to the promo_programs API endpoint which can assist in populating the information needed for the calculator.
api:direct_invites	Send a Financeit loan application link to a customer.
api:loans	View information for existing loan applications.
api:partners	View partner details.
api:single_access_links	Generate a link that provides access to create or review a loan on the Financeit platform.

PKCE Extension (code_challenge field)

PKCE is an extension to the Authorization Code flow to prevent CSRF and authorization code injection attacks. The PKCE extension requires the creation of a code challenge from a code verifier.

Code Verifier

According to the PKCE standard, a code verifier is a high-entropy cryptographic random string with a length between 43 and 128 characters (the longer the better). It can contain letters, digits, underscores, periods, hyphens, or tildes.

Code Challenge

Once the code verifier has been generated, you must transform (hash) it using the SHA256 algorithm and encode the hash using Base64 URL-safe encoding without padding to produce the code challenge. This is the code_challenge value that will be included as a parameter in the user authorization request.

The resulting code_challenge should match the output of the following Ruby code:

```
Base64.urlsafe_encode64(Digest::SHA256.digest(code_verifier), padding: false)
```

Example

Your implementation should output the code_challenge below when using the following code_verifier:

code_verifier: *“T51LC12HKKFZggjDt3vrdcwEaNLFElg3H_KkuDtMQYQ”*
Matching code_challenge: *“8GR4pmPbe066cVRmWSG2m_n4lBzRfz-M38Kpi_dnR0o”*

Accept the Authorization Response

User Approves the Request

By providing valid Financeit credentials and clicking to confirm, the user approves your application's request to access Financeit data on their behalf. The OAuth service then redirects the user back to the URL specified in the redirect_uri field. This redirection contains two query parameters within the URL:

Query Parameter	**Value**
code	An authorization code that can be exchanged for an access token.
state	The value of the state parameter supplied in the request.

The authorization code should be stored to be used during a later request summarized in the *Request an Access Token* section.

Failed Requests

If the user does not accept your request or if an error has occurred, the response query string contains the following parameters:

Query Parameter	**Value**
error	The type of failure
error_description	The reason the authorization failed.
state	The value of the state parameter supplied in the request (if provided in the initial request).

Request an Access Token

Request

Example request:

```
POST https://www.financeit.ca/en/api/v3/oauth/token
```

After the user has accepted the authorization request from the previous step, you can exchange the authorization code for an access token. You must send a POST request to the /token endpoint with the following parameters:

Body Parameters	**Required?**	**Value**
grant_type	Required	Set to “authorization_code”.
code	Required	The authorization code returned from the previous request.
redirect_uri	Required	This parameter is used for validation only (there is no actual redirection). The value of this parameter must exactly match the value of redirect_uri supplied when requesting the authorization code.
client_id	Required	The app ID you received from Financeit when you registered.
client_secret	Required	The app secret you received from Financeit when you registered.
code_verifier	Required	The value of this parameter must match the value of the code_verifier that your app generated in the previous step.

Successful Response

On success, the response will have a 200 OK status and the following JSON data in the response body:

Key	Type	Description
access_token	String	The access token, as issued by the authorization server.
token_type	String	How the access token may be used: always "Bearer".
expires_in	Int	The number of seconds remaining until the token expires.
refresh_token	String	The refresh token which your application can use to obtain another access token. This token must be kept secure. See the following section on refreshing tokens.
scope	String	The scope of access granted by the access token. This is a space-delimited list of the permissions associated with the token.
created_at	Time	The Unix timestamp, indicating when the access token was created.

Failed Response

On failure, the response will have a 400 Bad Request or 401 Unauthorized status and the following JSON data in the response body:

Key	Type	Description
error	String	A string identifying the error type.
error_description	String	A more detailed explanation of the error.

Refreshing Tokens

A refresh token is received in every successful token exchange with the /en/api/v3/oauth/token endpoint. It can be used to obtain a new access token from the same endpoint.

Request

Example request:

```
POST https://www.financeit.ca/en/api/v3/oauth/token
```

Requesting an access token using a refresh token uses the same endpoint as the access token exchange using the authorization code. To refresh an access token you will need to use a POST request to the /token endpoint with the following parameters:

Body Parameters	**Required?**	**Value**
grant_type	Required	Set to “refresh_token”.
refresh_token	Required	The refresh token returned during the previous token request.
client_id	Required	The app ID you received from Financeit when you registered.
client_secret	Required	The app secret you received from Financeit when you registered.

Successful Response

On success, the response will have a 200 OK status and the following JSON data in the response body:

Key	Type	Description
access_token	String	The access token, as issued by the authorization server.
token_type	String	How the access token may be used: always "Bearer".
expires_in	Int	The number of seconds remaining until the token expires.
refresh_token	String	A new refresh_token is issued to be used in the next request to refresh the user's access token. The refresh_token used in this request is now expired.
scope	String	The scope of access granted by the access token. This is a space-delimited list of the permissions associated with the token.
created_at	Time	The Unix timestamp, indicating when the access token was created.

Failed Response

On failure, the response will have a 400 Bad Request or 401 Unauthorized status and the following JSON data in the response body:

Key	Type	Description
error	String	A string identifying the error type.
error_description	String	A more detailed explanation of the error.

Next Steps

Now that you're able to obtain and refresh access tokens for your users, feel free to review the Financeit API documentation at https://www.financeit.ca/api/v3/index.html for a breakdown of the API endpoints available for use.

Once your application is ready for integration with the Financeit production API, please contact the Financeit team again to receive production credentials such as your app_id and app_secret.

There will be a few steps before production credentials are issued including:

- A demo of the merchant's API integration flow
- Verification the API was used successfully in the test environment.
- Confirmation of the nature of how you will be using the API. For example, embedding it in a mobile app, or using it on a web site.
- Confirmation that the application isn't likely to generate unusually high traffic, e.g. not a major retailer