Logout

Flight Crew View

Client Settings

Client ID

f0cf9180d491f06e

Copy

Client Secret

****

Regenerate Client Secret

Redirect URI

Note: use a semicolon (;) to enter multiple Redirect URIs

App Name

FlightBridge

Note: The User Authorization page requires a valid Redirect URI before entering a Passkey (even the test Passkey).

Update URI and App Name

Table of Contents

OAuth 2.0 with FC View

FC View uses a slightly modified approach to OAuth 2.0 to enhance security and streamline the process for our users. Below is a brief description of the information exchange that takes place during the modified OAuth 2.0 process.

Flow Overview:

- **Step 1:** Your logbook app requests authorization from the user by sending them to the FC View authorization page.
- **Step 2:** The user enters the 8-character passkey (generated in the FC View app).
- **Step 3:** FC View validates the passkey and sends a short-lived Authorization Code to your Redirect URI.
- **Step 4:** Your server immediately exchanges the Authorization Code for a short-lived Access Token and a Refresh Token. These tokens are unique for each end-user and are used to access the API.
- **Step 5:** Your server will store the Refresh Token and Access Token and use the Access Token to access the API endpoints. When the Access Token expires, the Refresh Token is used to obtain a new Access Token and a new Refresh Token and this Step 5 is repeated.
- **Step 6:** If the user revokes access (revokes the Refresh and Access Tokens) or the Refresh Token expires, then the user must reauthorize the app (go to Step 1).

Understanding Tokens and Passkeys in FC View

In the FC View OAuth 2.0 flow, several types of tokens and passkeys are used to ensure secure and streamlined access to the API. Below is a detailed explanation of each token and passkey, their purposes, storage recommendations, and representations.

1. Passkey

Purpose: The Passkey is an 8-character code generated in the FC View app and used by the end-user to initiate the OAuth 2.0 authorization process.

Usage: The end-user enters the Passkey on the FC View authorization page to validate their identity and grant permission to the logbook app. FC View validates the Passkey and sends an Authorization Code to the Redirect URI that you set up here.

Storage: The Passkey should never be seen by the logbook client app nor the client servers. It is generated in the FC View app and entered by the user directly into the FC View Authorization page.

2. Authorization Code

Purpose: The authorization code is a short-lived code (5 minutes) sent to your Redirect URI after the user successfully enters their passkey. It is used to obtain an Access Token and a Refresh Token.

Usage: Your server exchanges the authorization code for an access token and a refresh token.

Storage: Authorization codes should not be stored. They are meant to be used immediately to obtain tokens and have a short expiration time.

3. Access Token

Purpose: The Access Token is a short-lived token that allows your server to access the API endpoints on behalf of the user and download their flights. The Access Token tells the Flight Crew View servers that you are an authorized logbook client and which user you are requesting access to.

Usage: Include the access token in the Authorization header of your API requests to authenticate and authorize access to the API.

Storage: Access tokens should be stored securely and temporarily, as they have a short lifespan (typically 1 hour). Ensure they are stored in a way that prevents unauthorized access.

4. Refresh Token

Purpose: The Refresh Token is a longer-lived token used to obtain new access tokens once the current Access Token expires.

Usage: When the Access Token expires, your server uses the Refresh Token to request a new Access Token AND a new Refresh Token. BOTH the Access Token and Refresh Token should be replaced.

The old Refresh Token will enter into a one-week grace period after usage (to allow for errors or if an updated Refresh Token is not received). The new Refresh Token will have a renewed expiration.

Storage: Refresh Tokens should be stored securely, ideally encrypted at rest. They have a lifespan of 3 months and should be protected from unauthorized access.

Token and Passkey Info & Best Practices

- **Token string length**: Refresh and Access Tokens are generally 64 characters, but may be longer or shorter. In no case will they be longer than 255 chars without appropriate notice.
- **Store tokens securely**: Use encryption and secure storage mechanisms to protect tokens from unauthorized access.
- **Limit token exposure**: Never expose tokens in client-side code or public repositories. Ensure tokens are only used server-side.
- **Implement token rotation**: Regularly rotate tokens to minimize the risk of misuse. Always use the latest tokens provided by the FC View API.
- **Monitor and log token usage**: Track token usage and monitor for any unusual activity that might indicate a security issue.

By following these guidelines, you can ensure secure and efficient use of tokens and passkeys within the FC View OAuth 2.0 framework.

How to Access the API

🔒 HTTPS Required

All API calls and OAuth redirects MUST use HTTPS. Non-HTTPS requests will be rejected.

🖥️ Server-Side Only

All API calls should be made from your secure backend server. The client app should never make any calls to the API.

Any requests from the client app should be routed through your servers.

1. **Client ID**: Use the provided Client ID to identify your application.

   Example:
   `client_id=f0cf9180d491f06e`

2. **Client Secret**: Click the "Regenerate Client Secret" button above if you need a new client Secret. IMPORTANT: generating a new one will invalidate the old Client Secret and you will not be able to refresh any Access Tokens with the old Client Secret.

⚠️ Security Warning

Never store your Client Secret in client-side code or public repositories and never send it to your client app. Always keep it secure and transmit it only over HTTPS directly from your server to the /token/ endpoint to regenerate new Access and Refresh Tokens.

---

3. **Redirect URI and App Name**: Set your Redirect URI to the URL where you want to receive Authorization Codes. This field is required and must match the value you send the user to the FC View Authorization page. The App Name is used to identify your application to the user during the authorization process.

   NOTE: The App Name is not required, but the user will be instructed to not authorize the application if they do not recognize it, so make sure it's a recognizable name.

   Error Condition: If the Redirect URI is invalid or does not match the one registered here, you will receive a 400 Bad Request response. Ensure the Redirect URI is correctly configured in your settings.

---

4. **Generate API Passkey**: The end-user can generate an 8-character passkey in the FC View app:

   The end-user must open their Flight Crew View app and navigate to the app's menu > Settings, and expand the "Logbook Passkey" section.

   Select "Generate Passkey" and copy the 8-character Passkey.

   Testing only: See the next step for using a test Passkey.

---

5. **User Authorization Page**: Forward the user to our authorization page with the Client ID and Redirect URI, and a unique state parameter to maintain user context.

   Example URL:
   `https://www.flightcrewview2.com/logbook/logbookuserauth/?client_id=f0cf9180d491f06e&redirect_uri=https%3A%2F%2Fexample.com&state=YOUR_UNIQUE_STATE`

   Note: The state parameter is used to maintain state between the authorization request and the callback. It is set by you and it should be unique for every authorization request. In most cases, it is used to identify the user making the request, but it should not include any sensitive information. It will be returned along with the Authorization Code in the Redirect URI callback, and should be validated to ensure it matches the value sent in the request.

   Note: The Redirect URI must match the one you set in your client settings.

   Note: Do NOT include your Client Secret here, the request will be validated against the Client ID and the Redirect URI.

   Testing only: The Passkey "TEST1234" will always authenticate successfully for a test user's Authorization Code. The flights under the test user's account are static and will never change.

   Testing only: Authorization Codes are only valid for 5 minutes. They should be used immediately and discarded. However, Authorization Codes generated using the test Passkey will have a lifetime of 1 hour to facilitate testing.

Error Condition: If the Client ID or Redirect URI is invalid or mismatched, you will receive a 400 Bad Request response. Ensure all parameters are correctly configured.

---

6. **Authorization Code**: Once the user enters the passkey, our page will send an Authorization Code back to the specified Redirect URI along with the state parameter.

   Example Redirect URI:
   ```
   https://example.com?code=AUTHORIZATION_CODE&state=YOUR_UNIQUE_STATE
   ```

   Error Condition: If the Authorization Code is invalid or expired, you will receive a 401 Unauthorized response. Ensure the Authorization Code is used promptly and correctly.

---

7. **Exchange Authorization Code**: Exchange this Authorization Code for an Access Token and a Refresh Token. These tokens are unique for each user and verify the user's identity and your application's authorization to access the API. They should be stored securely on your server. See the Quick Reference for the /token/ API Endpoint below.

   Example using Basic Access Authentication:
   ```
   POST https://www.flightcrewview2.com/logbook/api/token/
   Authorization: Basic ZjBjZjkxODBkNDkxZjA2ZTpZT1VSX0NMSUVOVF9TRUNSRVQ=
   Content-Type: application/x-www-form-urlencoded

   grant_type=authorization_code&code=AUTH_CODE
   ```

   Alternative using POST parameters:
   ```
   POST https://www.flightcrewview2.com/logbook/api/token/
   Content-Type: application/x-www-form-urlencoded

   grant_type=authorization_code&code=AUTH_CODE&client_id=f0cf9180d491f06e
   &client_secret=YOUR_CLIENT_SECRET
   ```

   Note: The grant type is "authorization_code" for this step.

   Note: The Authorization Code expires in 5 minutes, so this step needs to happen immediately after receiving the Authorization Code.

   Note: The Access Token is used for the api endpoints (/flights/) and expires in 1 hour (3600 seconds), so save it and use it until it expires. Once it expires, use the Refresh Token at the /token/ endpoint to generate a new Access Token and a new Refresh Token.

   The Refresh Token is used to generate new Access Tokens and expires in 3 months but will be replaced with a new one each time it is used to obtain a new Access Token.

   Error Condition: If the Authorization Code is invalid or expired, you will receive a 401 Unauthorized response. Ensure the Authorization Code is used promptly.

> ⚠ **Security Warning**

All requests to the /token/ endpoint with your Client Secret should originate directly from your trusted server. The request should never originate from the client app or be routed through the client app.

---

8. **Include Access Token in API Requests**: The Access Token should be included in the Authorization header of your API requests. See the Quick Reference for the /flights/ API Endpoint below

Example API Request with Bearer Access Token:

```
GET https://www.flightcrewview2.com/logbook/api/flights/?
start_datetime_local=2024-05-01+12%3A34%3A56&end_datetime_local=2024-09-
01+12%3A34%3A56
Authorization: Bearer YOUR_ACCESS_TOKEN
```

Note: See the Quick Reference for the /flights/ API Endpoint below for more information about the date/time inputs.

Note: See the Automated Polling warning in the Rate Limits section below.

Error Condition: If the Access Token is missing or invalid, you will receive a 401 Unauthorized response. Ensure the Access Token is included and valid.

---

9. **Refresh an expired Access Token**: Use the Access Token to access the API. When the Access Token expires, use the Refresh Token to obtain a new Access Token and a new Refresh Token. See the Quick Reference for the /token/ API Endpoint below.

Example using Basic Access Authentication:

```
POST https://www.flightcrewview2.com/logbook/api/token/
Authorization: Basic ZjBjZjkxODBkNDkxZjA2ZTpZT1VSX0NMSUVOVF9TRUNSRVQ=
Content-Type: application/x-www-form-urlencoded

grant_type=refresh_token&refresh_token=REFRESH_TOKEN
```

Alternative using POST parameters:

```
POST https://www.flightcrewview2.com/logbook/api/token/
Content-Type: application/x-www-form-urlencoded

grant_type=refresh_token&refresh_token=REFRESH_TOKEN&client_id=f0cf9180d491f06e
&client_secret=YOUR_CLIENT_SECRET
```

Note: The grant type is "refresh_token" for this step.

Note: If the Refresh Token is used within its expiry period (3 months), a new refresh token will be issued along with a new Access Token. The old Refresh Token will have a short grace period to account for any errors, but the old Refresh Token should be discarded and the new Refresh Token used on the next request.

Error Condition: If the Refresh Token is invalid or expired, you will receive a 401 Unauthorized response. Ensure the Refresh Token is used correctly and within its validity period.

⚠ Security Warning

All requests to the /token/ endpoint with your Client Secret should originate directly from your trusted server. The request should never originate from the client app or be routed through the client app.

Testing

Testing only: The Passkey "TEST1234" will always authenticate successfully for a test user's Authorization Code. The flights under the test user's account are static and will never change.

Testing only: Authorization Codes are only valid for 5 minutes. They should be used immediately and discarded. However, Authorization Codes generated using the test Passkey will have a lifetime of 1 hour to facilitate testing.

For testing purposes, you can use the test passkey to generate an Authorization Code (enter it in step 5 above). You will then be able to proceed with the rest of the process of generating tokens and downloading flights.

The flights under the Test User's account are static and will always be the same.

Example

You can find an example, written in PHP, here: FCV Logbook API Github Example

---

Rate Limits

To ensure fair use and stability of our API, the following rate limits are in place:

- **Authorization Requests:** Limited to 5 requests per minute per ip address (end user IP).
- **/token/ requests:** Limited to 5 requests per minute based on your level of access (this can be increased upon request).
- **/flights/ api requests:** Limited to 300 per minute. Individual end users are limited to 10 requests per minute.

If you exceed these limits, you will receive a 429 Too Many Requests status code. Please contact our support team if you need higher limits for your application.

⚠ Automated Polling

-Automated polling/syncing with the API is only allowed at a rate of once/day/end-user.

-It should be performed overnight (North America) and limited to a quarter of your rate limit.

-The start_datetime_local should be limited to no further back than 2 months ago during automated polling.

-These polling suggestions are not strictly enforced, but excessive automated polling will be investigated and access potentially revoked.

Authorization Header

Authorization headers are critical in API authentication. There are two primary ways to use them: with Bearer Access Tokens and Basic Access Authentication.

Bearer Access Tokens

Bearer Access Tokens are used to access API endpoints for user data. Include the Access Token in the Authorization header of your API requests along with the word "Bearer":

Example API Request with Access Token:

```
GET https://www.flightcrewview2.com/logbook/api/flights/?start_datetime_local=2024-05-01+12%3A34%3A56&end_datetime_local=2024-09-01+12%3A34%3A56
Authorization: Bearer YOUR_ACCESS_TOKEN
```

Basic Access Authentication

Basic Access Authentication involves base-64 encoding your client credentials (base-64 encode this string: f0cf9180d491f06e:YOUR_CLIENT_SECRET) and including the result in the Authorization header along with the word "Basic". This method is primarily used for the /token/ endpoint:

Example using Basic Access Authentication:

```
POST https://www.flightcrewview2.com/logbook/api/token/
Authorization: Basic ZjBjZjkxODBkNDkxZjA2ZTpZT1VSX0NMSUVOVF9TRUNSRVQ=
Content-Type: application/x-www-form-urlencoded

grant_type=authorization_code&code=AUTH_CODE
```

Note: The base-64 encoded string in this example is just an example. You'll need to use your real Client ID and Client Secret to generate that string.

⚠ Security Warning

All requests to the /token/ endpoint with your Client Secret should originate directly from your trusted server. The request should never originate from the client app or be routed through the client app.

Revoking Access

The Flight Crew View app maintains a list of logbook connections that they have authorized and allows them to revoke access. This invalidates the Refresh Token and the app should be able to

gracefully respond to this situation by requesting the end-user to re-authorize (if wanted).

Your logbook client app should also give them the access to revoke this connection. In that case, you should use the /revokeToken/ endpoint (see below) to revoke access so that the connection will no longer show up in their Flight Crew View app.

A call to the /revokeToken/ endpoint will be treated as if the Refresh Token is compromised. All Refresh Tokens (if there are multiple) will be revoked and the authorization grant from the end-user will be ended.

If the end-user wishes to re-authorize your logbook, then you will need to guide them through the beginning of the Authorization Process again.

Testing Note: If you are using the Test Passkey (TEST1234) to test under multiple logbook test accounts, revoking a single Refresh Token will revoke all tokens generated from that Passkey. It effectively severs any authorizations between your logbook and the Flight Crew View user (which in this case, are all the same Flight Crew View test user).

API Error Codes

Code	Meaning
400	Bad Request - Check your request parameters
401	Unauthorized - Invalid or missing authentication
403	Forbidden - You don't have permission for this request
429	Too Many Requests - You've hit a rate limit

Quick Reference for the /flights/ API Endpoint

The `/flights/` API endpoint allows you to retrieve flight data for a user. You must include a valid Access Token in the Authorization header of your request.

Example Request

```
GET https://www.flightcrewview2.com/logbook/api/flights/?start_datetime_local=2024-05-01+12%3A34%3A56&end_datetime_local=2024-09-01+12%3A34%3A56
Authorization: Bearer YOUR_ACCESS_TOKEN
```

Note: The start_datetime_local and end_datetime_local field is a datetime field in this format: 'YYYY-MM-DD HH:MM:SS' and url encoded.

Note: The start_datetime_local is optional and if not provided, the API will return the user's entire history up to end_datetime_local.

Note: The end_datetime_local is optional and if not provided, the API will return flights up to two months from into the future from the current date.

Note: The start/end range will only look at the departure time in the departure airport timezone to determine if it is in the requested range.

Note: Users usually think of dates and times in local. These local datetimes are the start and end thresholds for the flight's departure time only.

Note: If you prefer to work in UTC, you can exchange start_datetime_local and end_datetime_local with start_datetime_utc and end_datetime_utc. Local times will be used if no inputs are given or if both utc and local times are given.

Example API Response

-Any of these fields may also be null or missing.
-fcv_flight_id is a unique id for that flight and can be referenced later to update that same flight in order to avoid duplicates.
-The dates are in YYYY-MM-DD HH:MM:SS format.
-If is_deadhead is 1, then the flight is a deadhead.
-tail_info is downloaded from the user's schedule and fcv_tail_number is pulled from FlightAware. fcv_tail_number should be used by default.


```
{
    "flights": [
        {
            "fcv_flight_id": "FCV_FLT_ID_8572488_TEST",
            "flight_number": "2748",
            "trip_number": "O7B46 : 03FEB",
            "is_deadhead": 0,
            "dep_airport": "BOS",
            "dep_airport_icao": "KBOS",
            "arr_airport": "PHL",
            "arr_airport_icao": "KPHL",
            "block": "0135",
            "tail_info": "1234\/",
            "crew_list": [
                {
                    "position": "CA",
                    "name": "John Doe",
                    "id": "987654"
                },
                {
                    "position": "FO",
                    "name": "Jane Doe",
                    "id": "876543"
                }
            ],
            "fcv_tail_number": "N123AB",
            "fcv_aircraft_type": "E75L",
            "scheduled_out_local": "2024-07-01 08:35:00",
            "scheduled_out_utc": "2024-07-01 12:35:00",
            "scheduled_in_local": "2024-07-01 10:13:00",
            "scheduled_in_utc": "2024-07-01 14:13:00",
            "actual_out_local": "2024-07-01 08:33:00",
            "actual_out_utc": "2024-07-01 12:33:00",
            "actual_off_local": "2024-07-01 08:53:50",
            "actual_off_utc": "2024-07-01 12:53:50",
            "actual_on_local": "2024-07-01 10:01:52",
            "actual_on_utc": "2024-07-01 14:01:52",
            "actual_in_local": "2024-07-01 10:08:00",
            "actual_in_utc": "2024-07-01 14:08:00",
            "dep_runway": "09",
            "arr_runway": "27R"
        },
        {
            "fcv_flight_id": "FCV_FLT_ID_8572489_TEST",
            "flight_number": "3921",
            "trip_number": "O7B46 : 03FEB",
            "is_deadhead": 1,
            "dep_airport": "PHL",
            "dep_airport_icao": "KPHL",
            "arr_airport": "BOS",
```

```
            "arr_airport_icao": "KBOS",
            "block": "0132",
            "tail_info": "1234\/",
            "crew_list": [
                {
                    "position": "CA",
                    "name": "John Doe",
                    "id": "987654"
                },
                {
                    "position": "FO",
                    "name": "Jane Doe",
                    "id": "876543"
                }
            ],
            "fcv_tail_number": "N456CD",
            "fcv_aircraft_type": "E75L",
            "scheduled_out_local": "2024-07-01 10:56:00",
            "scheduled_out_utc": "2024-07-01 14:56:00",
            "scheduled_in_local": "2024-07-01 12:22:00",
            "scheduled_in_utc": "2024-07-01 16:22:00",
            "actual_out_local": "2024-07-01 10:54:00",
            "actual_out_utc": "2024-07-01 14:54:00",
            "actual_off_local": "2024-07-01 11:14:50",
            "actual_off_utc": "2024-07-01 15:14:50",
            "actual_on_local": "2024-07-01 12:22:52",
            "actual_on_utc": "2024-07-01 16:22:52",
            "actual_in_local": "2024-07-01 12:29:00",
            "actual_in_utc": "2024-07-01 16:29:00",
            "dep_runway": "27R",
            "arr_runway": "09"
        }
    ]
}
```

Quick Reference for the /token/ API Endpoint

The `/token/` API endpoint allows you to exchange an authorization code for access and refresh tokens or to refresh an expired access token using a refresh token. This endpoint is critical for managing authentication and authorization in the FC View API.

Exchange Authorization Code for Tokens

Use this request to exchange an authorization code for access and refresh tokens:

Example using Basic Access Authentication:

```
POST https://www.flightcrewview2.com/logbook/api/token/
Authorization: Basic BASE64_ENCODED_CLIENT_ID_AND_SECRET
Content-Type: application/x-www-form-urlencoded

grant_type=authorization_code&code=AUTHORIZATION_CODE
```

Replace `BASE64_ENCODED_CLIENT_ID_AND_SECRET` with the Base64 encoded string of your client ID and client secret.

Replace `AUTHORIZATION_CODE` with the authorization code received from the user authorization flow.

Alternative using POST parameters:

```
POST https://www.flightcrewview2.com/logbook/api/token/
Content-Type: application/x-www-form-urlencoded

grant_type=authorization_code&code=AUTH_CODE&client_id=f0cf9180d491f06e
&client_secret=YOUR_CLIENT_SECRET
```

Example Response

```
{
    "access_token": "YOUR_ACCESS_TOKEN",
    "token_type": "Bearer",
    "expires_in": 3600,
    "refresh_token": "YOUR_REFRESH_TOKEN"
}
```

Refresh Access Token

Use this request to refresh an expired access token using a refresh token:

```
POST https://www.flightcrewview2.com/logbook/api/token/
Authorization: Basic BASE64_ENCODED_CLIENT_ID_AND_SECRET
Content-Type: application/x-www-form-urlencoded

grant_type=refresh_token&refresh_token=YOUR_REFRESH_TOKEN
```

Replace `YOUR_REFRESH_TOKEN` with the refresh token you received during the initial token exchange.

Example Response

```
{
    "access_token": "YOUR_NEW_ACCESS_TOKEN",
    "token_type": "Bearer",
    "expires_in": 3600,
    "refresh_token": "YOUR_NEW_REFRESH_TOKEN"
}
```

Quick Reference for the /revokeToken/ API Endpoint

The `/revokeToken/` API endpoint allows you to revoke a specific refresh token, which severs the authorization completely by revoking all associated tokens (both Refresh and Access Tokens) for the user.

Revoke a Token

Use this request to revoke a specific refresh token:

```
POST https://www.flightcrewview2.com/logbook/api/revokeToken/
Authorization: Basic BASE64_ENCODED_CLIENT_ID_AND_SECRET
Content-Type: application/x-www-form-urlencoded
```

```
refreshToken=YOUR_REFRESH_TOKEN
```

Replace `BASE64_ENCODED_CLIENT_ID_AND_SECRET` with the Base64 encoded string of your client ID and client secret.

Replace `YOUR_REFRESH_TOKEN` with the refresh token you want to revoke.

Alternative using POST parameters:

```
POST https://www.flightcrewview2.com/logbook/api/revokeToken/
Content-Type: application/x-www-form-urlencoded

client_id=f0cf9180d491f06e
&client_secret=YOUR_CLIENT_SECRET&refreshToken=YOUR_REFRESH_TOKEN
```

Replace `YOUR_REFRESH_TOKEN` with the refresh token you want to revoke.

Example Response

```
{
    "success": "token_revoked"
}
```

Additional Information

If you encounter any issues or need further assistance, please contact our support team at support@flightcrewview.com.