Enterprise Service Account API workflow

The purpose of this document is to illustrate the overall process and technical workflow for Adobe Enterprise partners and customers who want to integrate with the Adobe Stock API. The Stock API allows you to interact with Adobe Stock programmatically, rather than through a user interface.

Document contents

Introduction

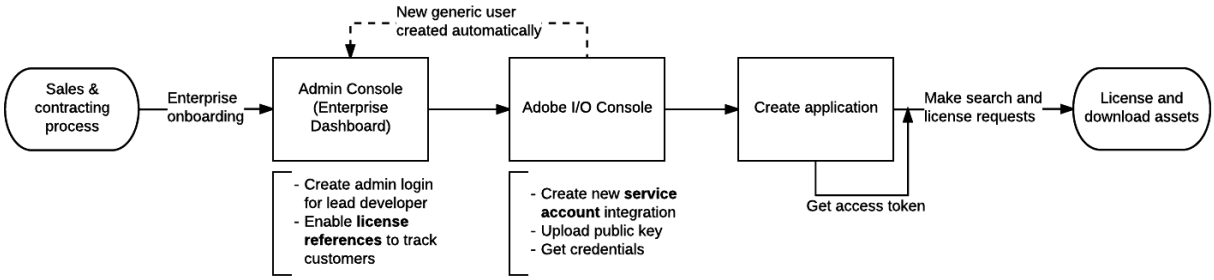
This paper outlines the workflow for any external integrator who wishes to license assets on behalf of its users. This could include:

- **Enterprise customers**. Integrator is an Enterprise Adobe customer who uses Stock for internal applications, such as for its branded websites. When it licenses assets, they are available for the entire organization.
- **Retailers (e.g., Print on Demand).** This partner produces commercial goods using Adobe Stock images, and sells them directly to consumers. Unlike the Enterprise use case, licensed assets are only available per user, and must be re-licensed to be used again.
- **Marketing platforms (e.g., social media aggregators and website builders).** There are multiple use cases, but the goal for the partner is to increase the value of their platform or application by offering the convenience of direct Stock access within their platform.

Some distinguishing factors of the Enterprise Service Account workflow:

- Partners must be on-boarded as Adobe Enterprise customers. As described below, applications must create a *service account* integration to allow asset licensing via the API, and only Enterprise customers may create service accounts.
    - This account allows the application to license assets on behalf of all the users in their organization, or on behalf of their customers. The end users would not need to interact directly with Adobe's API, or even need Adobe accounts.
- It is assumed that communication between the application and Adobe will be performed server-to-server, as opposed to client webpage-to-server. This ensures the highest level of security.

Workflow overview



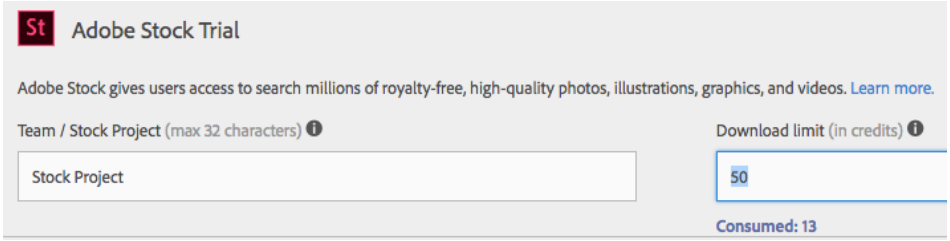
- The process begins with a contracting process with the Adobe sales team. While it is possible to test the search APIs without an Enterprise account, the licensing APIs require an account which has a quota of credits.
    - Sales and contracting is beyond the scope of this document. To get more information, please contact Grp-AdobeStockPartnerships@adobe.com.
- Once the account is provisioned and the partner goes through a brief onboarding process, a new Enterprise account is created which grants access to the Adobe Admin Console (also called the Enterprise Dashboard).
- The lead/primary developer would then sign into the separate Adobe I/O Console, which is used to create application integrations. The result of this process is an API key and a set of credentials which are used to authenticate requests.
- Now the integrator can start writing an application that uses the Stock API to make search queries, and programmatically generate an access token that allows the app to make authenticated license requests and fetch license history.
- When an end user interacts with the application and chooses to license an item, the application licenses an image on their behalf, and can download the asset for delivery.

Adobe Admin Console overview

Enterprise customers interact with their Adobe account via the Admin Console, where they can add new users, see their active entitlements, and manage their products. This is where any or all the following activities might occur:

- Create a system administrator account for the integrator. This account will be used to create credentials for the application, however this account will *not* be used to make API requests. Instead, the Adobe I/O Console will generate a technical account which will be used to authenticate API calls (see *Adobe I/O Console workflow*, below).
    - Note that it is not necessary to add additional users via the Console unless they need the ability to license assets directly from the Stock website, or re-download assets for fulfillment. It is expected that your application will interact with the Stock API in place of user interaction.
    - Admin Console documentation: https://helpx.adobe.com/enterprise/help/aedash.html
- Enable reference fields that get assigned as extra metadata for each transaction, simplifying tracking and reporting. For example, in a retail or marketing tools use case, the customer name and reference can be specified as part of the license request, and it will appear when the license history is requested.[1] This could potentially be added to an invoice for that customer.
    - References are configured on the **Permissions** tab of the product configuration. If these options do not appear in the Console, contact the Adobe team to enable this feature.
- Reassign quota to multiple internal brands. By default, all Stock credits are allocated to a single Product License Configuration (PLC), usually called "Default Adobe Stock configuration." It is possible to create additional PLCs, and change the distribution of those credits. For example, if there are two distinct branded websites sharing the quota and you wish to keep them separate, you can assign each 50% of the quota (or whatever ratio you want). Each PLC will have a separate license history for tracking purposes.
    - Note that the application configuration is also tied to the PLC. If you have two separate brands, each with its own PLC and quota, you would need to create two service account integrations (see *Adobe I/O Console workflow*, below). The application and logic could remain the same, but you would typically have two configuration settings for the different sites.
    - If you are testing Adobe Stock using a trial account, you may need to set up the quota for the first time to allow your application the right to license images. Follow instructions found at https://helpx.adobe.com/enterprise/help/manage-products-and-configurations.html for creating a PLC. Once the configuration is created, you will add quota (image credits) as shown in the screenshot below. Here, 50 credits were added to the PLC, and 13 credits have been used.

[1] License history can either be retrieved via the API, or viewed and downloaded as CSV from the Adobe Stock website.



Adobe I/O Console workflow

From this point forward, it is assumed that the systems integration team will take over the remaining tasks. The lead developer will use the I/O Console to create one or more service account integrations,[1] each of which will generate a unique API key and credentials used to generate authentication tokens.

Before you begin

To complete the steps below, you will need to generate or purchase a public key certificate which will be used to sign your application. For testing purposes, this is easily accomplished from the command line using a tool like OpenSSL.[2] For production, typically this would either come from your website administrator, or be purchased from a third-party certificate authority. You will also need a corresponding private key.

To create a self-signed certificate for development, this single OpenSSL command will generate a private key and public certificate needed for the following steps.

```
$ openssl req -x509 -sha256 -nodes -days 365 -newkey rsa:2048
-keyout myPrivate.key -out myPublic.crt
```

See https://www.adobe.io/apis/cloudplatform/console/authentication/createcert.html for more information.

Other tools

To simplify testing, it's recommended that you also consider installing these additional tools.

- **Postman**. Postman is a free application available in a standalone version and as a Google Chrome app. It allows you to test API requests, attaching required headers and parameters, and save your requests for future testing. Also, it can export requests in different formats, including cURL, PHP and JavaScript code.

[1] Multiple configurations are necessary if you have more than one Product License Configuration (PLC) configured. Another common reason for creating separate configurations is for development and production. Although both will ultimately point to the same production instance, a new PLC with very little quota (e.g., ten or fewer credits) could be created for test purposes.
[2] https://www.openssl.org/

  - https://www.getpostman.com

- **Secure web server.** As you begin to test and integrate the API into your main application, you will need an environment able to send and receive secure (HTTPS) requests. For local testing, you can use a self-signed certificate (see instructions under *Authentication workflow*, below) that you trust on your machine. You can either run a full-fledged local server using Apache, or it is easy to create a simple one using Node.js or Python.
  - Apache (use alone, or with PHP or Apache Tomcat): https://www.apache.org/
  - Node JS (use the Express module): https://nodejs.org/en/
  - Python (SimpleHTTPServer module): https://www.python.org/
  - Installing a self-signed certificate.
    - Windows: https://blogs.technet.microsoft.com/sbs/2008/05/08/installing-a-self-signed-certificate-as-a-trusted-root-ca-in-windows-vista/
    - Mac: http://tosbourn.com/getting-os-x-to-trust-self-signed-ssl-certificates/

Creating the integration in Adobe I/O

1. Access the Adobe I/O Console here: https://console.adobe.io.
   - Sign in using a system administrator login created in the Admin Console, above.
2. Click the **New Integration** button.
3. Select the following items, clicking **Continue** each time:
   - **Access an API** > **Adobe Stock** > **Service Account integration**[1] > **New integration**
4. This opens a screen where you will enter your integration details.
   - **Name**: Your application's name. This will not be sent in your API requests; however, a good practice might be to give it the same web-friendly name you will be using later when sending the required X-Product header (see *Application flow details*, below).
   - **Description**: E.g., "Integration of Stock API with MyWebsite.com."
   - **Public key**: Here you will upload/drag and drop the public certificate you generated or purchased earlier.
     - Note that you cannot reuse the same certificate for multiple integrations. This is why it may be convenient to create a self-signed certificate for testing purposes, and later replace it with a production version.

[1] There are two types of application integrations with Adobe Stock: OAuth and Service Account. OAuth requires a different workflow that is not described here. For Enterprise use cases, you will need to choose Service Account integration.

  - **License**: This is optional, and only applies if you created multiple PLCs as described earlier. Choose the PLC associated with this application.

- Once saved, the I/O Console will generate several pieces of information you will need later.
  - Copy everything in the **Client Credentials** section (including the Client Secret, which you must safeguard like your private key), and store in a secure location.
  - Optionally, select the **JWT** tab and copy the JWT Payload. This can be used for troubleshooting.

Authentication workflow

While the Search API only requires an API key, accessing licensing functions requires additional authentication. Adobe Stock uses JSON Web Tokens[1] (JWTs) to exchange client credentials and grant access to account information.

At a high level, your application will generate a JWT that includes your credentials and makes claims to use APIs, and if accepted, the Stock API will respond back with an access token (another JWT). For more information on how JWTs work at Adobe, code examples and sample libraries, see https://www.adobe.io/apis/cloudplatform/console/authentication/jwt_workflow.html.

Your JWT must contain the following claims:

Claim	Description
`exp`	Required. The expiration time, an absolute number of seconds since 1/1/1970 GMT. You must ensure that the expiration time is later than the time of issue. After this time, the JWT is no longer valid. An expiration period is typically one day.
`iss`	Required. The issuer, your organization ID in the format ***org_ident*@AdobeOrg**. Identifies your organization that has been configured for access to the Adobe Stock API.
`sub`	Required. The subject, your API client account ID in the format: ***id*@techacct.adobe.com.**
`aud`	Required. The audience for the token, in the format: **https://ims-na1.adobelogin.com/c/*api_key***.
configured claims	Required. The API-access claim configured for your organization: **https://ims-na1.adobelogin.com/s/ent_stocksearch_sdk**.
`jti`	Optional. A unique identifier for the token, if configured for your organization. If required, you must use a decimal number greater than any valued used before, in order to prevent replay attacks. Otherwise, the request fails. To ensure an acceptable value, you can use the current Unix time (seconds since 1970).

[1] https://en.wikipedia.org/wiki/JSON_Web_Token

JWT files have a limited life span. It is recommended that you generate an access token at least once per session. On the application side, you will use the client credentials (the "claims" or "payload") and your private key to generate your JWT requesting access. There are several open-source libraries that can create the JWT for Java, Node.js, PHP, Python, etc. Once your request JWT is created, it will be securely posted to Adobe's IMS server (Identity Management Service) for validation.

```
curl -X POST \
  https://ims-na1.adobelogin.com/ims/exchange/v1/jwt \
  -H 'content-type: application/x-www-form-urlencoded' \
  -d 'jwt_token={encoded-jwt}&client_id={client-id}
&client_secret={client-secret}'
```

The IMS server will respond with a JSON message that includes your access token.

```
{
  "token_type": "bearer",
  "access_token": "{encoded-token}",
  "expires_in": 86399981
}
```

This access token will be used for every licensing operation. While it is optional for search requests, if supplied it will return whether the asset has already been licensed.

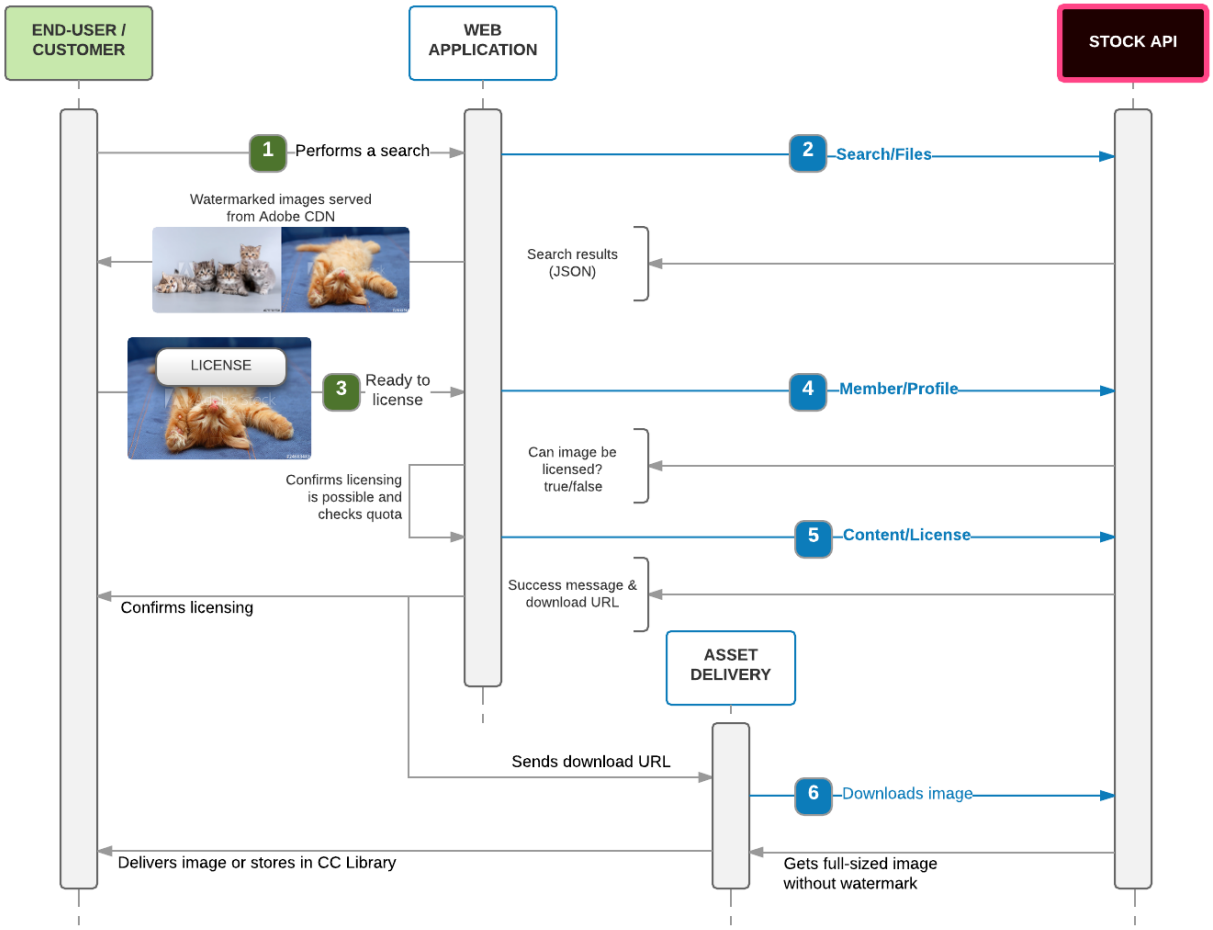
Application flow details

The application will make and receive RESTful calls between the end-user and Adobe Stock, including licensing and download of images. Since the Stock API is unaware of the end-users in this flow, image licenses will be granted to the application (technically, to the parent organization), and credits will be deducted from the PLC. For a discussion of quotas and PLCs, see *Adobe Admin Console overview*, above.

The diagram below illustrates the overall process, with requests to the Stock API colored in blue. In each case, there are some mandatory parameters or headers that must be supplied, such as the access token that is sent with the License API requests and download URL for the full-sized image.

In summary, the typical flow is:

- **Search**. Call the Search API and get back results. This call can either be made with authentication or not—i.e., using the bearer token generated in the *Authentication workflow* section, above—with the difference that authenticated search requests can also return whether the asset is already licensed. This could be used to display a "licensed" flag or badge on the asset.
- **Get profile**. Gets the current remaining quota and confirms that the asset can be licensed.
- **License**. Gets a license for the asset and deducts credit.
- **Download**. Delivers the asset itself to the user or for a fulfillment process.



Search

1. The workflow begins when the end-user (or customer) performs a search on the partner's site for Adobe Stock images.
2. The web application submits a **Search/Files** request to the Stock API. This API can search based on text keywords, or execute a *similarity search*, where the Stock API uses machine learning to compare the data of the image you upload (or an image URL) to its repository of 90M of images.

- Endpoint
  - `https://stock.adobe.io/Rest/Media/1/Search/Files`
- Required headers
  - `x-Product` : arbitrary string identifying your app
  - `x-api-key` : generated when you created the integration on I/O Console

- Parameters [1]
  - The API can accept multiple search parameters and result filters. At minimum, search requires at least one `search_parameters[]` command.
  - Search results will only return a maximum of 64 assets. For details on paginating results, see https://www.adobe.io/apis/creativecloud/stock/docs/api/search.html#paginate.

Search/Files request

```
GET /Rest/Media/1/Search/Files?locale=en-US
&search_parameters[words]=kittens HTTP/1.1
Host: stock.adobe.io
X-Product: CFSTest/0.1
x-api-key: ...0e3f
```

- Response: Stock API responds with a JSON object containing the number of results, and an array of metadata for each file.
  - The API limits the number of files returned in the search, therefore to get more results, you must use the pagination mechanism when requesting subsequent pages.
  - Some of the most important metadata returned is the title of each image, the Stock content/media ID, and URLs to watermarked previews and thumbnails.

Search/Files response[1] (truncated to show one file result)


```
{
  "nb_results": 247038,
  "files": [
    {
      "id": 75950374,
      "title": "five kittens",
      "width": 2500,
      "height": 1667,
      "creator_name": "adyafoto",
      "creator_id": 205216144,
      "thumbnail_url":
"https://as2.ftcdn.net/jpg/00/75/95/03/500_F_75950374_yNANaKsbx7oLzG
JUFSzXW7j5cGoiKDT9.jpg",
      "thumbnail_html_tag": "<img
src=\"https://as2.ftcdn.net/jpg/00/75/95/03/500_F_75950374_yNANaKsbx
7oLzGJUFSzXW7j5cGoiKDT9.jpg\" alt=\"five kittens\" title=\"Photo:
five kittens\" zoom_ratio=\"1.25\" zoom_depth_max=\"2\" />",
      "thumbnail_width": 500,
      "thumbnail_height": 334,
      "media_type_id": 1,
      "vector_type": null,
```


[1] For more information on search parameters and filtering results, see https://www.adobe.io/apis/creativecloud/stock/docs/api/search.html.

```
      "content_type": "image/jpeg",
      "category": {
        "id": 44,
        "name": "Cats"
      },
      "premium_level_id": 0
    }, { ... },
]}
```

3. The web application parses the results and shows thumbnails for each asset and any other data (such as keywords) that may help the user refine their search. Some filters are necessary from a business perspective. For example, if photos are only being offered, there is a parameter to restrict search only to images and not return vector illustrations or videos. Similarly, if the application cannot sell Premium content for contractual reasons, there is a filter to restrict the search to Standard items only.

Get profile

4. When the end-user is ready to get an unwatermarked version of the asset (whether it will be used directly by the user, or on behalf of the user on a printable good), the asset must be licensed from Adobe Stock. The application submits a **Member/Profile** request to the Stock API with the content ID of the image. The purpose is to confirm it is possible to license this image, and perhaps get the total number of credits available. Note that the "profile" request does not get the profile of the end-user, but of the organization. The quota returned corresponds to the PLC set up in the Admin Console.

- Endpoint
  - `https://stock.adobe.io/Rest/Libraries/1/Member/Profile`
- Required headers[1]
  - Same headers as Search requests.
  - `Authentication` : **`Bearer`** access token generated by Adobe IMS. This is a long base-64 encoded string representation. See *Authentication workflow*, above.
- Parameters
  - `content_id` : Asset's unique identifier, obtained from a search response's `id` attribute.
  - `license` : Stock licensing state for the asset (defaults to **`Standard`**.)
  - `locale` : Default is **`en_US`**.

[1] For more information on headers and parameters for license requests, see https://www.adobe.io/apis/creativecloud/stock/docs/licensing.html.

Member/Profile request

```
GET /Rest/Libraries/1/Member/Profile?content_id=75428610
&license=Standard&locale=en_US HTTP/1.1
Host: stock.adobe.io
X-Product: CFS_Test_1.0
x-api-key: ...0e3f
Authorization: Bearer eyJ4NX...ztcSQ
```

- Response: The Stock API returns JSON indicating the remaining credits, and whether it is possible to license the image.
  - It should always be *possible* to license an image, if it is allowed by the PLC and there are available credits. As mentioned earlier, if the contract/PLC does not permit Premium content but the application did not filter out these images, then it may *not* be possible to license the image. The `"state" : "possible"` field indicates that this image *can* be licensed.

Member/Profile response


```
{
    "available_entitlement": {
        "quota": 100,
        "license_type_id": 16,
        "has_credit_model": true,
        "has_agency_model": false,
        "is_cce": true,
        "full_entitlement_quota": {
            "credits_quota": 100
        }
    },
    "member": {
        "stock_id": 41003529
    },
    "purchase_options": {
        "state": "possible",
        "requires_checkout": false,
        "message": "This will use 1 of your 100 credits."
    }
}
```


License

5. The application now sends a **Content/License** request, which will license the image and deduct credits. Calling this URL again on the same asset will not trigger a license action but will deliver the URL, unless the application fits into a specific retail use case (see below).

- Endpoint
  - `https://stock.adobe.io/Rest/Libraries/1/Content/License`

- Required headers: Same as **Member/Profile**.
- Parameters: Same as **Member/Profile**, unless the contract or licensing stipulates that the file must be re-licensed for multiple uses. This is common in the Print on Demand use case, where the retailer may need to license the image again for each additional customer or additional print application. In this case, an optional parameter is added:
  - `license_again` : **`true`** | **`false`**[1]

Content/License request

```
GET /Rest/Libraries/1/Content/License?content_id=75950374
&license=Standard HTTP/1.1
Host: stock.adobe.io
X-Product: CFS_Test_1.0
x-api-key: ...0e3f
Authorization: Bearer eyJ4NX...ztcSQ
```

- Response: Stock API responds with a JSON object indicating that licensing was successful and the updated quota. Among the information returned is the **URL to download the full asset**, which the web application will store in its database and/or forward to its fulfillment/manufacturing division.

Content/License response

```
{
    "member": {
        "stock_id": "yNANaKsbx7oLzGJUFSzXW7j5cGoiKDT9,1097854"
    },
    "available_entitlement": {
        "quota": 79,
        "license_type_id": 15,
        "has_credit_model": true,
        "has_agency_model": false,
        "is_cce": true,
        "full_entitlement_quota": {
            "image_quota": 80,
            "credits_quota": 34
        }
    },
    "contents": {
        "75950374": {
            "content_id": "75950374",
            "size": "Comp",
            "purchase_details": {
                "state": "just_purchased",
                "license": "Standard",
```

[1] In most cases, setting the value to `false` is the same as not setting it at all.

```
                "date": "2017-09-05 23:47:14",
                "url":
"https://primary.staging.adobestock.com/Rest/Libraries/Download/7595
0374/1",
                "content_type": "image/jpeg",
                "width": 2500,
                "height": 1667
            },
            "comp": {
                "url":
"https://primary.staging.adobestock.com/Rest/Libraries/Watermarked/D
ownload/75950374/1",
                "content_type": "image/jpeg",
                "width": 1000,
                "height": 668
            },
            "thumbnail": {
                "url":
"https://as2.ftcdn.net/jpg/00/75/95/03/500_F_75950374_yNANaKsbx7oLzG
JUFSzXW7j5cGoiKDT9.jpg",
                "content_type": "image/jpeg",
                "width": 500,
                "height": 334
            }
        }
    }
}
```


Download

6. To finish the workflow, the application or fulfillment team will download the image without a watermark. In the previous response returned from the Content/License request (above), the download URL is highlighted.

- Endpoint (note the domain is different from previous requests)
  - `https://stock.adobe.com/Rest/Libraries/Download/{content_id}/1`
- Required headers: None.
- Parameters:
  - `token`: Access token string (*without* `Authorization: Bearer` prefix.)
- Response: Stock server will respond with a 200 and/or 302 response, depending on the type of asset. Files will either be delivered directly from the URL, or the server will issue a redirect to a download destination, which the application or browser must follow to receive the file.

Download request

```
GET /Rest/Libraries/Download/88295836/1?token=eyJ4NX...ztcSQ
HTTP/1.1
```

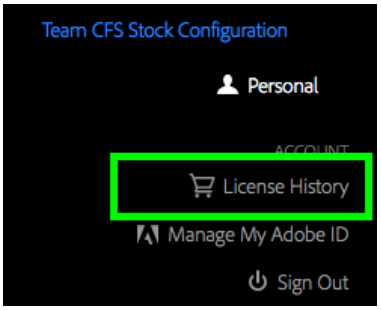
```
Host: stock.adobe.com
```

Note that the access token does not need to the same one generated during the purchase flow; it simply needs to contain the same credentials. Also, the licensing and download do not need to happen simultaneously, and once licensed, the file may be re-downloaded without penalty.

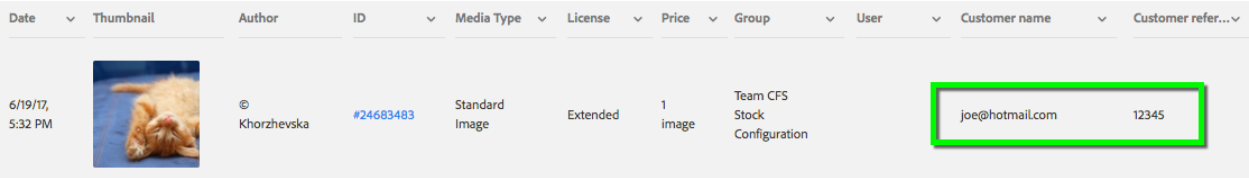
Additional workflows

In addition to the primary flow described above of Search > Get Profile > License, there are other common scenarios that may come up.

- Get license history (manual method). For tracking and reporting, the organization may need to periodically get its license history. If license references were supplied with each order (e.g., customer email and ID), this is where this information would be found. This task can be done on demand (manually) using the Adobe Stock website, or the application can perform this task programmatically using the API.
    - Sign into Adobe Stock (https://stock.adobe.com/), click on your name or name of your PLC in the top nav toolbar, open the flyout menu, and choose **License History.**



        - A table will open showing a detailed history of every licensing transaction. Click the gear icon to show additional columns, such as *Customer Name* and *Customer Reference* (ID).



- Get license history (API method).
    - Endpoint
        - `https://stock.adobe.io/Rest/Libraries/1/Member/LicenseHistory`
    - Required headers: Same as **Member/Profile**.
    - Parameters: None

Member/LicenseHistory request

```
GET /Rest/Libraries/1/Member/LicenseHistory HTTP/1.1
Host: stock.adobe.io
X-Product: CFS_Test_1.0
x-api-key: ...0e3f
Authorization: Bearer eyJ4NX...ztcSQ
```

- Response: The returned JSON response will resemble a Search API response, where you will receive an array of file metadata.

Member/LicenseHistory response


```
{
  "nb_results": 16,
  "files": [
    {
      "license": "Standard",
      "license_date": "6/19/17, 5:32 PM",
      "download_url":
"https://stock.adobe.com/Download/DownloadFileDirectly/mTFwPEzGtGYQN
Gpm9SnEzAtqlF4hHKTu",
      "id": 24683483,
      "title": "kitten sleeps on the back",
      "creator_name": "Khorzhevska",
      "creator_id": 200468973,
      "content_url": "https://stock.adobe.com/stock-photo/kitten-
sleeps-on-the-back/24683483",
      "media_type_id": 1,
      "vector_type": null,
      "content_type": "image/jpeg",
      "height": 1805,
      "width": 2715
    },
    { ... }
  ]
}
```


Adobe Systems Incorporated
345 Park Avenue
San Jose, CA 95110-2704
USA
www.adobe.com

5/15