Xpressdocs User Data Feed Integration Interface

August 2024

Xpressdocs Holdings, Inc. 1301 NE Loop 820, Fort Worth, TX 76131, USA
+1 817.321.7904 | https://www.xpressdocs.com/xd

Table of Contents

1 Introduction

This document describes the Xpressdocs User Data Feed Integration Interface. A partner/customer can implement this interface as an HTTPS API to allow Xpressdocs to retrieve current user data, which will then be ingested into the Xpressdocs platform. Please note that this is only an api specification, Xpressdocs does not provide this api but expects the partner/customer to build this api.

2 Security

Xpressdocs will send HTTPS requests to the customer's API for retrieval of user data. The customer's API endpoints may be secured using either HTTP Basic Auth or OAUTH2 authentication schemes.

2.1 Requirements

Xpressdocs supports two types of Authentication mechanisms to the customer's API: Basic Auth and OAUTH2. To ensure proper configuration for our system to authenticate to your API, it is necessary for you to supply one of the following security configuration settings to Xpressdocs:

2.1.1 Basic Auth

Setting	Description
username	Username for Basic Auth
password	Password for Basic Auth

The username and password will be Base64 URL encoded as a Basic Auth string in the `Authorization` header. The string format before Base64 encoding is:

```
Basic username:password
```

The `Authorization` header will be included in all requests.

Sample Request with Basic Auth:

```
GET /users?from_date=2023-11-01&to_date=2024-01-01&limit=10&offset=1 HTTP/1.1
Host: example.com/api
Authorization:    Basic c60d780549ff58154e01cdc0
```

2.1.2 OAUTH2

Setting	Description
client_id	Used for obtaining a token.
client_secret	Used for obtaining a token.
Authorization Payload Content-Type	Which payload `Content-Type` to use for authorization: • **application/json** • **application/x-www-form-urlencoded** Default `Content-Type` is **application/x-www-form-urlencoded** Used for obtaining a token.
Authorization Payload Data	Default payload data format: **client_id={{client_id}}&client_secret={{client_secret}}** Used for obtaining a token.
Authorization Endpoint	What endpoint to use for authorization. Example: If the full URL is **https://example.com/api/auth**, then the endpoint would be **/auth** Used for obtaining a token.

Sample Token Request Payload:

```
POST /auth HTTP/1.1
Host: example.com/api
Content-Type: application/x-www-form-urlencoded
Content-Length: 42

client_id=xpressdocs&client_secret=abc123
```

Sample Token Response:

This is an example of the data we expect to receive from the authentication call:

```
{
  "access_token": "dj0yJmk9N2pIazlsZk1zlNQSXJzZWNyZXQmeD00NA--",
  "expires": "2024-01-16T20:17:49Z"
}
```

The access token received from this call will then be used in the `Authorization` header for all non-authentication related calls. The string format is:

```
Bearer access_token
```

This `Authorization` header will be included in all subsequent requests to the API.

Sample Request with Bearer Token:

```
GET /users?from_date=2023-11-01&to_date=2024-01-01&limit=10&offset=1 HTTP/1.1
Host: example.com/api
Authorization:    Bearer dj0yJmk9N2pIazlsZk1zlNQSXQmeD00NA--
```

2.1.3 Error Handling

Requests made with valid Authorization should receive a response with an HTTP Status Code of **`200 OK`**. Requests made with invalid Authorization should receive a response with an HTTP Status Code of **`401 Unauthorized`**.

3 User Data Feed Integration

3.1 Overview

Xpressdocs expects the API to provide user data in the JSON format specified in the subsequent sections of this document. Xpressdocs current systems user account structure requires User and Office entities to be provided. If the customer has an extra layer of hierarchy that groups Offices into regions, then Region entities can also be provided by the API. Details on those entities data are provided in the sections below.
Xpressdocs will retrieve data at set intervals throughout the day. These intervals can be as short as every minute.

3.1.1 Associated Headshots

For best quality, associated headshot URLs should reference the image directly and not an image manipulation utility or other image proxy. If a proxy is required, the quality should be verified early in the integration process; high-resolution images are preferred. Please note that for printed images to be sharp without any blur or pixelation an image with at least 300 dpi is required.

3.1.2 API Configuration

The following configuration settings are required (unless noted as optional) for Xpressdocs to integerate with your API:

Setting	Options	Description
Host	URL	The root URL for all API calls. Example: If the full URL is **`https://example.com/api/user`**, then the host would be **`https://example.com/api`**
User Endpoint	String	What endpoint to use for retrieving user data. Example: If the full URL is **`https://example.com/api/users`**, then the endpoint would be **`/users`**
Office Endpoint	String	What endpoint to use for retrieving office data. Example: If the full URL is **`https://example.com/api/offices`**, then the endpoint would be **`/offices`**

Region Endpoint	String	Optional endpoint to use for retrieving region data. Example: If the full URL is **`https://example.com/api/regions`**, then the endpoint would be **`/regions`**

See Details below for description of Users, Offices, and Regions

4 Data Feed Fields and Structure

This section describes the structure of the JSON data feed in detail. All the fields to be included, whether they are required, and a brief description of each field are outlined below.

4.1 API Parameters

Xpressdocs requires that the API support delta pulls. The data returned must be zero indexed. The following parameters should be supported by the API URL as Query Parameters:

Parameter	Data Type	Required	Description	Notes
fromDate	ISODate format timestamp	Yes	Used to limit entities returned to only those last modified after the `fromDate`	
toDate	ISODate format timestamp	No	Used to limit entities to only those modified before the `toDate`	This field will not normally be used unless preloading old data.
limit	Number	Yes	Used for paginating results.	Xpressdocs would limit this to 100
offset	Number	Yes	Used for paginating results.	We will cycle through this until there are no results. Must be zero-indexed.
entityId	String	No	Customer's unique id reference for region, office, or user. When this filter is used, we expect only one entity to be returned if it exists.	

4.2 Data Feed Structure

The user data feed structure consists of three entities: regions, offices, and users.

1. Users: These are the base-level entities in our system.
2. Offices: These entities are location-based groupings of users and include address information.
3. Regions: These optional entities group offices into larger, typically geographical, areas.

Typically, a user is associated with one office, but may be assigned to multiple offices using the officeIdList array. Only office admins would have access to multiple offices.
Only one region may be associated with an office, but multiple regions (using regionIdList array) may be associated with region admin users giving them access to all offices under those regions.

Each entity requires a distinct endpoint. The endpoints will be called in the following order: regions, then offices, and finally users.

4.2.1 Users

The customer's API response should be a JSON object with a single root field name called **`users`** at the top level. Its value will be an array of `user` objects:

```
{
  "users": [
    {
        // The user object is described in detail in the following sections
    }
  ]
}
```

4.2.2 User Schema

Each user will be its own object. The object will include the fields described below.

```
{
  "userId": "",
  "officeId": "",
  "active": true,
  "firstName": "",
  "middleName": "",
  "lastName": "",
  "directPhone": "",
  "directPhone2": "",
  "email": "",
  "loginLevel": 5,
  "headshotUrl": "",
  "license": "",
  "url": "",
```

```
  "agentDisplay1": "",
  "agentDisplay2": "",
  "agentDisplay3": "",
  "agentDisplay4": "",
  "agentDisplay5": "",
  "agentDisplay6": "",
  "agentDisplay7": "",
  "agentDisplay8": "",
  "officeIdList": [],
  "regionIdList": []
}
```

Field	Type	Req'd	Description
userId	String	Yes	Unique Identifier for the user Should match SSO Feed
officeId	String	Yes	ID for office to which the user belongs
active	Boolean	No	Indicates if the user is active (default: true)
firstName	String	Yes	First name of the user
middleName	String	No	Middle name of the user
lastName	String	Yes	Last name of the user
directPhone	String	No	Direct phone number of the user
directPhone2	String	No	Secondary phone number of the user
email	String	Yes	Email address of the user
loginLevel	Integer	No	Login level of the user (3 - Company Admin, 4 - Region/Office Admin, 5 - User. default: 5)
headshotUrl	String	No	URL of the user's headshot
license	String	No	License information of the user
url	String	No	URL associated with the user
agentDisplay1	String	No	Custom display field for the user (default: 'firstname' 'lastname')
agentDisplay2	String	No	Custom display field for the user
agentDisplay3	String	No	Custom display field for the user
agentDisplay4	String	No	Custom display field for the user (default: 'directPhone')
agentDisplay5	String	No	Custom display field for the user (default: 'directPhone2')
agentDisplay6	String	No	Custom display field for the user (default: 'license')
agentDisplay7	String	No	Custom display field for the user (default: 'email')
agentDisplay8	String	No	Custom display field for the user (default: 'url')
officeIdList	Array	No	List of office IDs associated with the Office Admin
regionIdList	Array	No	List of region IDs associated with the Region Admin

4.2.3 Offices

The customer's API response should be a JSON object with a single root field name called **`offices`** at the top level. Its value will be an array of `office` objects:

```
{
  "offices": [
    {
        // The user object is described in detail below
    }
  ]
}
```

4.2.4 Office Schema

Each office will be its own object. The object will include the fields described below.

```
{
    "officeId": "",
    "active": true,
    "regionId": "",
    "officeName": "",
    "officeLegalName": "",
    "officeAddress1": "",
    "officeAddress2": "",
    "officeCity": "",
    "officeState": "",
    "officeZip": "",
    "officeCountry": "US",
    "officePhone": "",
    "officeFax": "",
    "officeEmail": "",
    "officeDisclaimer": "",
    "officeDisplay1": "",
    "officeDisplay2": "",
    "officeDisplay3": "",
    "officeDisplay4": "",
    "officeDisplay5": "",
    "officeDisplay6": ""
}
```

Field	Type	Req'd	Description
officeId	String	Yes	Unique identifier for office/location Should match SSO Feed
active	Boolean	No	Indicates if the office is active (default: true)
regionId	String	No	Optional ID for region to which the office belongs
officeName	String	Yes	Name identifying the office/location
officeLegalName	String	No	Legal name of the office/location
officeAddress1	String	No	Address line 1 of the office/location
officeAddress2	String	No	Address line 2 of the office/location
officeCity	String	No	City where the office/location is situated
officeState	String	No	Two-digit state code of the office/location
officeZip	String	No	Five-digit zip code of the office/location
officeCountry	String	No	Two-digit country code (default: 'US')
officePhone	String	No	Phone number of the office/location
officeFax	String	No	Fax number of the office/location
officeEmail	String	No	Default email of the office/location
officeDisclaimer	String	No	Disclaimer for the office/location
officeDisplay1	String	No	Custom display field for the office (default: 'officeLegalName' or 'officeName')
officeDisplay2	String	No	Custom display field for the office (default: 'officeAddress1' 'officeAddress2')
officeDisplay3	String	No	Custom display field for the office (default: 'officeCity', 'officeState' 'officeZip')
officeDisplay4	String	No	Custom display field for the office (default: 'officePhone')
officeDisplay5	String	No	Custom display field for the office (default: 'officeFax')
officeDisplay6	String	No	Custom display field for the office

4.2.5 Regions

The customer's API response should be a JSON object with a single root field name called **`regions`** at the top level. Its value will be an array of `region` objects:

```
{
  "regions": [
    {
        // The user object is described in detail in the following sections
    }
  ]
}
```

4.2.6 Region Schema

Each region will be its own object. The object will include the fields described below.

```
{
    "regionId": "",
    "active": true,
    "regionCountry": "US",
    "name": ""
}
```

Field	Type	Req'd	Description
regionId	String	Yes	Unique identifier for region
active	Boolean	No	Indicates if the region is active (default: true)
regionCountry	String	No	Two-digit country code (default: 'US')
name	String	Yes	Name identifying the region

5 Testing

5.1 Environments

Xpressdocs has the following environments:

Name	Description
Stage/Dev	Staging environment for validating your endpoint
Production	Live production environment

6 Tasks

The following tasks are required to complete a data feed integration:

1. Provide an API endpoint and configuration information where Xpressdocs can fetch user data in the above format.
2. Xpressdocs will validate the data feed response contains all required elements and is well-formed JSON.
3. Xpressdocs will notify the client that the data feed import can begin.