Xpressdocs Listing Feed Integration Interface

Apr 2026

Xpressdocs Partners, Ltd. 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 Listing Feed Integration Interface. A customer can implement this interface as an HTTPS API to allow Xpressdocs to retrieve current listing data, which will then be ingested into the Xpressdocs platform. Xpressdocs will make this data available to the end users of the customer. The data may be utilized throughout the Xpressdocs platform for a variety of products and services, including Automated Property Marketing Program, My Listings, and other services.

2 Security

Xpressdocs will send HTTPS requests to the customer's API for retrieval of listing 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 /listings?from_date=2023-11-01&to_date=2024-01-01&limit=10&offset=0 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 /api/auth HTTP/1.1
Host: example.com
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": "688ef044e-5a24f3dba-2e05c3ebe09f28",
  "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
```

Sample Request with Bearer Token:

```
GET /api/listings?from_date=2023-11-01&to_date=2024-01-01&limit=10&offset=0
HTTP/1.1
Host: example.com
Authorization:    Bearer 688ef044e-5a24f3dba-2e05c3ebe09f28
```

This `Authorization` header will be included in all subsequent requests to the API, until the expiry time indicated by the `expires` field received with the token. The authorization token request will be repeated before expiration or upon receiving a 401 Unauthorized response.

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 Listing Feed Integration

3.1 Overview

Xpressdocs expects the API to provide listing data in the JSON format specified in the subsequent sections of this document. This data must include all pertinent property data, any image URLs associated with the listing, Agent data, and any applicable dates, such as Date Listed, Open House data, and Sold Date. Xpressdocs will retrieve data at set intervals throughout the day. These intervals can be as short as every minute.

3.1.1 Associated Photos

For best quality, associated photo 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.

When Xpressdocs imports associated photos, they will be imported in the same order provided in the data feed.

3.1.2 API Configuration

The following configuration settings can be provided to Xpressdocs to better adapt to the customer's API:

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

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
from_date	ISODate format timestamp: YYYY-MM-DD	Yes	Used to limit listings returned to only those last modified after the `from_date`	
to_date	ISODate format timestamp: YYYY-MM-DD	No	Used to limit listings to only those modified before the `to_date`	This field will be used when preloading old data. A future value may be used in delta pulls.
limit	Number	Yes	The maximum number of records to return.	Used for pagination. Xpressdocs would typically use 100
offset	Number	No	The number of records to skip.	Used for pagination. We will increment this by the limit until there are no results. Must be zero-indexed.
listingId	String	No	Customer's unique id reference for listing. When this filter is used, we expect only one listing to be returned if it exists.	Support for this parameter is not strictly necessary for the listing feed to function, but it will help Xpressdocs troubleshoot issues.

4.2 Data Feed Structure

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

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

4.3 Data Feed Schemas

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

```
{
  "location": {
    // Object described under section 4.3.1
  },
  "agent": {
    // Object described under section 4.3.2
  },
  "buyerAgent": {
    // Object described under section 4.3.3
  },
  "openHouse": [
    {
      // Object described under section 4.3.4
    }
  ],
  "pictures": [
    {
      // Object described under section 4.3.5
    }
  ],
  "CompanyListingAttributes": [
    {
      // Note the first letter is capitalized due to legacy reasons
      // Object described under section 4.3.6
    }
  ],
  // Additional properties at the top level of the object are
  // described under section 4.3.7
}
```

Field	Data Type	Required	Description
location	Object	Yes	Location of the property
agent	Object	No	Seller representative Agent information
buyerAgent	Object	No	Buyer representative Agent information
openHouse	Array of Objects	No	List of Open House events
pictures	Array of Objects	No	List of Image data objects
CompanyListingAttributes	Array of Objects	No	Exclusive Content information

4.3.1 Location Schema Data Fields

```
"location": {
  "streetAddress": "",
  "city": "",
  "state": "",
  "zip": "",
  "unitNumber": "",
  "lat": "",
  "long": ""
}
```

Field	Data Type	Required	Description
streetAddress	String	Yes	Property Street Address
unitNumber	String	No	Property Unit Number
city	String	Yes	Property City
state	String	Yes	Property State
zip	String	Yes	Property Postal Code
lat	String	No	Property Latitude
long	String	No	Property Longitude

4.3.2 Agent Schema Data Fields

See the subsections below for details about Dynamic Quantities and Banked Dollars.

The following fields should be included in the `agent` object.

```
"agent": {
  "firstName": "",
  "lastName": "",
  "userId": "",
  "emailAddress": "",
  "autoListedCount": "",
  "autoPriceChangeCount": "",
  "autoOpenHouseCount": "",
  "autoSoldCount": "",
  "listedBankedCount": "",
  "listedBankedExp": "",
  "soldBankedCount": "",
  "soldBankedExp": "",
  "listingBankedCount": "",
  "listingBankedExp": ""
}
```

Field	Data Type	Required	Description
firstName	String	Yes	First Name of Associated Employee/Agent
lastName	String	Yes	Last Name of Associated Employee/Agent
userId	String	Yes	Unique Identifier of Associated Employee/Agent. Either this should match agent SSO ID, or the email Address field below must match the Agent data in Xpressdocs. Without this match, listings won't be associated to Agent accounts.
emailAddress	String	Yes	Email Address of Associated Employee/Agent
autoListedCount	Number	No	Dynamic Quantity for Just Listed
autoPriceChangeCount	Number	No	Dynamic Quantity for Price Change
autoOpenHouseCount	Number	No	Dynamic Quantity for Open House
autoSoldCount	Number	No	Dynamic Quantity for Just Sold
listedBankedCount	Number	No	Banked Dollars count for Just Listed

listedBankedExp	ISODate format timestamp: YYYY-MM-DD	No	Banked Dollars expiration date for Just Listed
soldBankedCount	Number	No	Banked Dollars count for Just Sold
soldBankedExp	ISODate format timestamp: YYYY-MM-DD	No	Banked Dollars expiration date for Just Sold
listingBankedCount	Number	No	Banked Dollars count for the first event of any type
listingBankedExp	ISODate format timestamp: YYYY-MM-DD	No	Banked Dollars expiration date for the first event of any type

Dynamic Quantities

Dynamic Quantity fields override the default order quantity for auto-generated orders. Each field corresponds to a specific listing event type (“Just Listed”, “Open House”, “Price Change”, or “Just Sold”).

Each field accepts an integer. If the value is 0, empty, or omitted, the system uses the configured program or user default quantity instead.

Banked Dollars

Banked Dollar fields generate a marketing voucher when an event is created. The field value represents a number of pieces; the voucher balance is calculated from the per-piece cost of the default template (defined in the program) at that quantity. Banked Dollar fields are available for the "Just Listed” and “Just Sold” event types.

The listingBankedCount field applies to the listing as a whole rather than a specific event type. When set, only the first event generated for the listing record will produce a voucher, regardless of which event type it is.

Each count field has a corresponding expiration field that specifies the date the voucher will expire if unused. If the field is omitted or left empty, then the voucher will never expire.

4.3.3 Buyer Agent Schema Data Fields

See the subsections below for details about Dynamic Quantities and Banked Dollars.

The following fields should be included in the buyerAgent object.

```
"buyerAgent": {
  "firstName": "",
  "lastName": "",
  "userId": "",
  "emailAddress": "",
  "autoNewNeighborCount": "",
  "newNeighborBankedCount": "",
  "newNeighborBankedExp": "",
  "listingBankedCount": "",
  "listingBankedExp": ""
}
```

Field	Data Type	Required	Description
firstName	String	Yes	First Name of Associated Employee/Agent
lastName	String	Yes	Last Name of Associated Employee/Agent
userId	String	Yes	Unique Identifier of Associated Employee/Agent. Either this should match agent SSO ID, or the email Address field below must match the Agent data in Xpressdocs. Without this match, listings won't be associated to Agent accounts.
emailAddress	String	Yes	Email Address of Associated Employee/Agent
autoNewNeighborCount	Number	No	Dynamic Quantity for New Neighbor
newNeighborBankedCount	Number	No	Banked Dollars count for New Neighbor
newNeighborBankedExp	ISODate format timestamp:	No	Banked Dollars expiration date for New Neighbor

	YYYY-MM-DD		
listingBankedCount	Number	No	Banked Dollars count for the first event of any type
listingBankedExp	ISODate format timestamp: YYYY-MM-DD	No	Banked Dollars expiration date for the first event of any type

Dynamic Quantities

Dynamic Quantity fields override the default order quantity for auto-generated orders. For the buyer agent, dynamic quantity applies only to the "New Neighbor" event type.

The field accepts an integer. If the value is 0, empty, or omitted, the system uses the configured program or user default quantity instead.

Banked Dollars

Banked Dollar fields for the buyer agent work the same way as for the seller agent: the field value represents a number of pieces, and the voucher balance is calculated from the per-piece cost of the default template (defined in the program) at that quantity. For the buyer agent, a Banked Dollar field is available only for the "New Neighbor" event type.

The listingBankedCount field works the same as on the seller agent but since for the buyer agent the only possible event currently is "New Neighbor", this field will only apply when a "New Neighbor" event is triggered.

If the same person appears as both the seller agent and buyer agent on the same listing, and both include a listingBankedCount, that agent would receive two separate vouchers if the property is sold.

Each count field has a corresponding expiration field that specifies the date the voucher will expire if unused. If the field is omitted or left empty, then the voucher will never expire.

4.3.4 Open House Details Schema Data Fields

The Open House field will be an array of objects. The following data fields should be included in each `openHouse` object.

```
"openHouse": [
  {
    "date": "2024-01-16",
    "startTime": "10:00:00",
    "endTime": "17:00:00"
  }
]
```

Field	Data Type	Required	Description
date	String (YYYY-MM-DD)	Yes	Date of the open house
startTime	String (HH:MM:SS)	Yes	Start time of the open house
endTime	String (HH:MM:SS)	Yes	End time of the open house

4.3.5 Pictures Schema Data Fields

The Pictures field will be an array of objects. The following data fields should be included in each Pictures object.

```
"pictures": [
  {
    "caption": "",
    "pictureUrl": "",
    "thumbUrl": ""
  }
]
```

Field	Data Type	Required	Description
caption	String	No	Caption for the Image
pictureUrl	String	Yes	Associated Photo URL
thumbUrl	String	Yes	Associated Photo thumbnail URL

4.3.6 CompanyListingAttributes Schema Data Fields

This field controls Exclusive Content for the listing. Exclusive Content is a template-access feature that unlocks a curated set of exclusive templates for users whose listing data meets configured eligibility criteria.

These values are customized per client; if you intend to use this feature, please contact customer service on setting this feature up for you.

The CompanyListingAttributes field will be an array of objects. The following data fields should be included in each CompanyListingAttributes object.

*Please note that these field names, unlike the others, need to be capitalized.

```
"CompanyListingAttributes": [
  {
    "AttributeId": "",
    "AttributeName": ""
  }
]
```

Field	Data Type	Required	Description
AttributeId	String	Yes	Attribute ID for exclusive content
AttributeName	String	No	Name of the attribute

4.3.7 Listing Details Fields

The following fields should be placed at the top level inside the `listing` object, alongside the fields that were previously discussed in 4.3.1 – 4.3.6:

```
"status": "",
"listingId": "",
"listedPrice": 1,
"soldPrice": 1,
"listingUrl": "",
"mlsId": "",
"mlsName": "",
"dateListed": "",
"dateSold": "",
"title": "",
"propertyType": "",
"description": "",
"bedrooms": 1,
"fullBathrooms": 1,
"halfBathrooms": 1,
"bathrooms": 1,
"livingArea": "",
"lotSize": "",
"yearBuilt": "",
"displayAddress": "",
"taxAnnualAmount": 1,
"virtualTourUrl": "",
"elementarySchool": "",
"highSchool": "",
"juniorHighSchool": "",
"schoolDistrict": "",
"poolYN": true,
"singleStory": true,
"lotSizeSquareFeet": "",
"neighborhood": "",
"fireplace": true,
"roof": "",
"heatingType": "",
"cooling": "",
"garageCount": "",
"imageCount": 1,
"publicRemarks": "",
"roomCount": 1,
"secondaryEmailAddresses": "",
"foundation": "",
"stories": "",
"fence": "",
"pool": "",
"construction": "",
"heat": "",
"air": "",
"block": "",
"levels": ""
```

Field	Data Type	Required	Description
status	String	Yes	Property Status (pending, closed, available, sold)
listingId	String	No	Unique ID for the listing
listedPrice	String | Number	No	Listed Price
soldPrice	String | Number	No	Sold Price
listingUrl	String	No	Listing URL
mlsId	String	Yes	Property Listing Unique Identifier
mlsName	String	Yes	Name of Listing Service
dateListed	String (ISO DateTime Format: YYYY-MM-DD HH:mm:ss)	Yes	Property Listing Date
dateSold	String (ISO DateTime Format: YYYY-MM-DD HH:mm:ss)	No*	Property Sold Date * Required if **`propertyStatus`** is **`sold`**
title	String	No	Title of Listing
propertyType	String	No	Type of Property (Single Family, Residential, Condo, Lots & Land)
description	String	No	Description of Property
bedrooms	String | Number	No	Number of Bedrooms
fullBathrooms	String | Number	No	Number of Full Bathrooms
halfBathrooms	String | Number	No	Number of Half Bathrooms
bathrooms	String | Number	No	Number of Bathrooms
livingArea	String	No	Number of Living Areas or Dimensions of Living Area
lotSize	String	No	Property Lot Size in Acres
yearBuilt	String	No	Year Property was Built
displayAddress	String	No	True/False indication of whether address should be displayed
taxAnnualAmount	String | Number	No	Amount of Annual Tax
virtualTourUrl	String	No	Virtual Tour URL

Field	Data Type	Required	Description
elementarySchool	String	No	Name of elementary school
highSchool	String	No	Name of high school
juniorHighSchool	String	No	Name of junior high school
schoolDistrict	String	No	Name of school district
poolYN	Boolean	No	Pool Y or N
singleStory	Boolean	No	Single story
lotSizeSquareFeet	String	No	Square feet of lot size
neighborhood	String	No	Name of subdivision or neighborhood
fireplace	String	No	Fireplace Description
roof	String	No	Type of roof
heatingType	String	No	Type of heating system
cooling	String	No	Description of cooling system
garageCount	String	No	Dimensions or description of Garage
publicRemarks	String	No	Public remarks
roomCount	String	No	Number of rooms
secondaryEmailAddress	String	No	Secondary Email Address
foundation	String	No	Type of foundation
stories	String	No	Number of stories
fence	String	No	Type of fence
pool	String	No	Pool description
construction	String	No	Type of home construction

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 listing 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.