expert

Back

Enterprise API (1.0.1)

Download OpenAPI specification: Download

Property Finder Enterprise API Gateway

Introduction

Property Finder provides a collection of APIs that enable you to integrate with our various products and services. Our APIs use JSON for request and response bodies and employ standard HTTP response codes. You can consume the APIs directly using any HTTP or REST library.
For any additional support or questions related to the integration, please contact us at integration.support@propertyfinder.ae.

Usage Restrictions

This API is intended **strictly for server-to-server communication**.

- **Do not** use this API directly from frontend applications (e.g., browsers, mobile apps).
- **Do not** use this API as a Backend-for-Frontend (BFF) for public or semi-public client apps.
- Requests should originate from **secure, server-side environments** only.

The following sections cover below mentioned topics:

- Authentication
- Rate Limiting

Authentication

When working with the API, you need to pass an access token in the Authorization header of all your requests. This will be in the form of a Bearer token, for example: `Bearer <ACCESS_TOKEN>`

Obtaining the token

To obtain the token, you will need OAuth 2.0 credentials. Your OAuth 2.0 credentials consist an API Key and an API Secret (corresponding to OAuth 2.0 client ID and client secret).

Getting OAuth 2.0 credentials

To get your OAuth 2.0 credentials, open the PF Expert application, and log in. Once you are logged in, navigate to the API Credentials section under the Developer Resources tab in the left sidebar. From here, you can generate a new API key and API secret. Be sure to use the type as `API Integration`.
You can exchange these for an access token by calling the Issue JWT Token Endpoint. You can pass the API key and API secret in the request body.
`API Key` : Your unique client identifier on PF Expert.
`API Secret` : Your client secret, used for authentication when requesting an access token.

Example request

```
curl --location 'https://atlas.propertyfinder.com/v1/auth/token' \
--header 'Content-Type: application/json' \
--header 'Accept: application/json' \
--data
'
  {
    "apiKey":"<API_KEY>",
    "apiSecret":"<API_SECRET>"
  }
'
```

The token server will return a Bearer accessToken in JSON Web Token (JWT) format which you should use in the Authorization header of your API requests.

Example response

```
{
  "accessToken": "<ACCESS_TOKEN>",
  "expiresIn": 1800,
  "tokenType": "Bearer"
}
```

Example request

```
curl --location 'https://atlas.propertyfinder.com/v1/users?page=1&perPage=15' \
--header 'Accept: application/json' \
--header 'Authorization: Bearer <ACCESS_TOKEN>'
```

Your access token will be valid for the duration (in seconds) indicated by the `expiresIn` field in the response. When it expires, you'll need to request a new one. Currently, an issued access token will expire in **30 minutes**. Once a token expires, you must request a new one. **No refresh token flow is supported in this integration.**

Unauthorized Requests

If you attempt to access a protected resource without valid authorization you will receive a **401 Unauthorized** or **403 Forbidden** HTTP error.

```
HTTP/1.1 401 Unauthorized
Content-Type: application/json
{
   "detail": "access denied (decision id: 5593a6df-2c67-4891-bd0a-83f3edfe9b14)
   "title": "Unauthorized",
   "type": "AUTHENTICATION"
}
```

Rate Limiting

To ensure fair use and maintain high availability of our APIs, we enforce rate limiting on all requests made to our endpoints.

Default Rate Limits

Unless otherwise specified, all clients are subject to the following default limits:

Request Type	Limit	Example Operations
Issue JWT Token endpoint	**60 requests per minute**	*Issue JWT token*
All other endpoints	**650 requests per minute**	*Create user*, *Search Listings*, etc.

> **Note:** The rate limits are applied per IP address and per client. Other factors may also be considered.

If you exceed your allotted rate limit: You will receive a **429 Too Many Requests** HTTP response. The request can then be safely retried after some time. Example:

```
HTTP/1.1 429 Too Many Requests
```

Best Practices

- To avoid frequently hitting the rate limit, retry requests using an incremental backoff with jitter.
- Cache responses for frequently accessed data to reduce duplicate API calls.

Publishing a Listing

General Considerations

- Every listing **must be associated with a Public Profile** representing the identity of a PF Expert user responsible for the listing, as displayed publicly on the Property Finder website. Public profiles are associated with PF Expert users, and can be retrieved using the [GET] /v1/users endpoint.
- Every listing **must be associated with a valid Location** from Property Finder's location tree.
- The list of **accepted amenities depends on both the Property Type and Category** of the listing.
  - See country-specific rules below for allowed combinations.
- The listing creation process is split into two steps:
  - Creating the listing in draft mode ([POST] /v1/listings)
  - Publishing the listing ([POST] /v1/listings/{id}/publish)

High-Level Flow

1. **Obtain Public Profile Id**
   Call [GET] /v1/users to retrieve available users and their public profiles. select the appropriate `publicProfile.id`.


```
{
  "publicProfile": {
    "id": 216582,
    ...
  }
  ...
}
```


2. **Obtain Location ID**
   Call [GET] /v1/locations?search=Marina to retrieve the valid location and obtain the correct `id`.


```
  {
    "data": [
      {
        "coordinates": {
```

```
            "lat": 25.078367,
            "lng": 55.14041
        },
        "id": 50,
        "name": "Dubai Marina",
        ...
    }
  ]
}
```


3. **Create Listing (Draft Mode)**
   Call ([POST] /v1/listings) to create a new listing in draft state, providing the required data including `publicProfileId` and `locationId`. The response will return the newly created `listingId`.
4. (*Optional*) **Get Publishing Price**
   Call [GET] /v1/listings/{id}/publish/prices to retrieve the price to publish the listing.
5. **Publish Listing**
   Call [POST] /v1/listings/{id}/publish to publish the listing and make it publicly available on the Property Finder website.

Region-Specific Flows

Dubai (UAE)

To ensure compliance with the **Dubai Land Department's (DLD)** regulations, particularly under the DLD Strict Adherence initiative, it's imperative that all property listings on Property Finder accurately reflect the details registered with the DLD. This includes critical information such as price, property type, and location.
To facilitate this, our Enterprise API provides a structured workflow that mandates the retrieval of official permit details before creating or updating any listing.

Special Business Rules

- Dubai Land Department (DLD) compliance is mandatory for listings in Dubai.
- Permit ID and License Number must be obtained before creating or updating listings.
- Listing verification is automatically triggered after publishing.
- Listings located within DIFC and JAFZA are exempt from DLD compliance and permit validation.

High-Level UAE Flow (Dubai-specific)

1. Retrieve Public Profile ID ([GET] /v1/users)
2. Retrieve Location ID ([GET] /v1/locations)
3. Retrieve Permit Details ([GET] /v1/compliances/{permitNumber}/{licenseNumber})
4. Create Listing ([POST] /v1/listings with permit and license details

5. Publish Listing ([POST] /v1/listings/{id}/publish)
6. Listing Verification is triggered automatically

Step 3 — Retrieve Official Permit Details

- **Endpoint:** [GET] /v1/compliances/{permitNumber}/{licenseNumber}
- **Required Parameters:**
  - `permitType`
  - `permitNumber`
  - `licenseNumber` (company license number)
- **Purpose:** Retrieve latest official DLD permit data to ensure that the listing aligns with official records.
- **Why:** Prevent discrepancies in critical fields like price, location, and property type. Any discrepancies between your listing and the DLD data can lead to listing rejection or legal complications.

DLD Listing Type Enforcement

From the compliance response, extract data[].property.listingType.
This value defines the official property classification and must be mapped to a valid **type** when creating a listing.

Mapping: listingType → allowed listing type values

DLD Listing Type	Allowed Listing Type Values
Unit	Apartment, Duplex, Full Floor, Half Floor, Penthouse, Hotel Apartment
Villa	Villa, Townhouse, Bungalow
Building	Compound, Whole Building, Bulk Units, Labor Camp, Office space, Retail, Shop, Showroom, Warehouse, Factory, Business Center, Coworking spaces, Staff Accommodation
Land	Land, Farm, Villa, Townhouse, Bungalow, Whole Building

Example:

If DLD listingType = "Unit", valid **listing type** values include: "Apartment", "Duplex", "Penthouse", etc.

Step 4 — Create Listing

- Use retrieved permit data to populate all regulated fields.
- Ensure that the **listing type** matches the value allowed by the DLD listingType

Best Practices

- Always use the most recent data obtained from the compliance endpoint to populate your listing.

- Avoid manual alterations to critical fields like price or property type without first updating them with the DLD and retrieving the new permit details.
- Ensure that all mandatory fields are accurately filled to prevent listing rejections.

Ongoing Maintenance and Updates

- Price or Detail Changes: Before making any changes to listing details via the API, ensure the information is first updated with the DLD. Once updated, use the `GET /v1/compliances/` endpoint to retrieve the latest permit details and then update your listing accordingly.
- Regular Verification: Periodically verify that your listings remain consistent with DLD records, especially if there are changes in regulations or property details.

Region-Specific Rules

UAE – Allowed Categories, Property Types & Amenities

Property Types and Amenities by Category

Category	Allowed Property Types	Allowed Amenities
commercial	farm, land	*No amenities allowed*
commercial	bulk-rent-unit, bulk-sale-unit, business-center, co-working-space, factory, full-floor, half-floor, labor-camp, office-space, retail, shop, show-room, staff-accommodation, villa, warehouse, whole-building	shared-gym, covered-parking, networked, shared-pool, dining-in-building, conference-room, lobby-in-building, vastu-compliant
residential	land	*No amenities allowed*
residential	apartment, bulk-rent-unit, bulk-sale-unit, bungalow, compound, duplex, full-floor, half-floor, hotel-apartment, penthouse, townhouse, villa, whole-building	central-ac, built-in-wardrobes, kitchen-appliances, security, concierge, maid-service, balcony, private-gym, shared-gym, private-jacuzzi, shared-spa, covered-parking, maids-room, study, childrens-play-area, pets-allowed, barbecue-area, shared-pool, childrens-pool, private-garden, private-pool, view-of-water, view-of-landmark, walk-in-closet, lobby-in-building, vastu-compliant

Egypt – Allowed Categories, Property Types & Amenities

Property Types and Amenities by Category

Category	Allowed Property Types	Allowed Amenities
commercial	farm, land	*No amenities allowed*
commercial	bulk-rent-unit, bulk-sale-unit, cafeteria, clinic, co-working-space, factory, full-floor, half-floor, hotel-apartment, ivilla, medical-facility, office-space, restaurant, retail, shop, show-room, staff-accommodation, villa, warehouse, whole-building	shared-gym, covered-parking, networked, dining-in-building, conference-room, lobby-in-building
residential	land	*No amenities allowed*
residential	apartment, bulk-rent-unit, bulk-sale-unit, bungalow, cabin, chalet, duplex, full-floor, half-floor, hotel-apartment, ivilla, palace, penthouse, roof, townhouse, twin-house, villa, whole-building	central-ac, built-in-wardrobes, kitchen-appliances, security, balcony, shared-gym, shared-spa, covered-parking, maids-room, study, shared-pool, childrens-pool, private-garden, private-pool, view-of-water, view-of-landmark, walk-in-closet, lobby-in-building

Bahrain – Allowed Categories, Property Types & Amenities

Property Types and Amenities by Category

Category	Allowed Property Types	Allowed Amenities
commercial	land	*No amenities allowed*
commercial	bulk-rent-unit, bulk-sale-unit, hotel-apartment, labor-camp, medical-facility, office-space, retail, shop, show-room, staff-accommodation, warehouse, whole-building	central-ac, security, balcony, shared-gym, covered-parking, networked, shared-pool, private-garden, private-pool, view-of-water, dining-in-building, conference-room, lobby-in-building

Category	Allowed Property Types	Allowed Amenities
residential	land	*No amenities allowed*
residential	apartment, bulk-rent-unit, bulk-sale-unit, bungalow, chalet, compound, duplex, hotel-apartment, penthouse, townhouse, villa, whole-building	central-ac, built-in-wardrobes, kitchen-appliances, security, concierge, maid-service, balcony, private-gym, shared-gym, private-jacuzzi, shared-spa, covered-parking, maids-room, study, childrens-play-area, pets-allowed, barbecue-area, shared-pool, childrens-pool, private-garden, private-pool, view-of-water, view-of-landmark, walk-in-closet, lobby-in-building

Saudi Arabia – Allowed Categories, Property Types & Amenities

Property Types and Amenities by Category

Category	Allowed Property Types	Allowed Amenities
commercial	farm, land	electricity, waters, sanitation, no-services, fixed-phone, fibre-optics, flood-drainage
commercial	factory, office-space, shop, show-room, warehouse	central-ac, security, balcony, shared-gym, covered-parking, networked, view-of-water, view-of-landmark, dining-in-building, conference-room, lobby-in-building, electricity, waters, sanitation, no-services, fixed-phone, fibre-optics, flood-drainage
residential	farm, land	electricity, waters, sanitation, no-services, fixed-phone, fibre-optics, flood-drainage
residential	apartment, chalet, compound, full-floor, rest-house, villa, whole-building	central-ac, built-in-wardrobes, kitchen-appliances, security, concierge, private-gym, shared-gym, private-jacuzzi, shared-spa, covered-parking, maids-room, barbecue-area, shared-pool, childrens-pool, private-garden, private-pool, view-of-water, walk-in-closet, lobby-in-building, electricity, waters, sanitation, no-services, fixed-phone, fibre-optics, flood-drainage

Qatar – Allowed Categories, Property Types & Amenities

Property Types and Amenities by Category

Category	Allowed Property Types	Allowed Amenities
commercial	farm, land	electricity, waters, sanitation, no-services, fixed-phone, fibre-optics, flood-drainage
commercial	factory, office-space, shop, show-room, warehouse	central-ac, security, balcony, shared-gym, covered-parking, networked, view-of-water, view-of-landmark, dining-in-building, conference-room, lobby-in-building, electricity, waters, sanitation, no-services, fixed-phone, fibre-optics, flood-drainage
residential	farm, land	electricity, waters, sanitation, no-services, fixed-phone, fibre-optics, flood-drainage
residential	apartment, chalet, compound, full-floor, rest-house, villa, whole-building	central-ac, built-in-wardrobes, kitchen-appliances, security, concierge, private-gym, shared-gym, private-jacuzzi, shared-spa, covered-parking, maids-room, barbecue-area, shared-pool, childrens-pool, private-garden, private-pool, view-of-water, walk-in-closet, lobby-in-building, electricity, waters, sanitation, no-services, fixed-phone, fibre-optics, flood-drainage

Changelog

2025-08-12

`GET` **/v1/leads**

- `!WARNING!` for the 'query' request parameter 'perPage', the max was decreased from '100.00' to '50.00'

- added the required property 'data/items/listing/reference' to the response with the '200' status

2025-08-07

`GET` /v1/users/

- added the new optional 'query' request parameter 'email'
- added the new optional 'query' request parameter 'id'

2025-08-05

`GET` /v1/credits/transactions

- endpoint added

2025-07-30

`GET` /v1/credits/balance

- endpoint added

2025-07-29

WHPayloadLead

- added `reference` field to `listing` object

2025-07-25

`GET` **/v1/leads**

- added the new optional 'query' request parameter 'projectId'

`POST` **/v1/webhooks**

- added the new 'listing.published' enum value to the request property 'eventId'
- added the new 'listing.unpublished' enum value to the request property 'eventId'

Webhooks

- added the new 'listing.published' event
- added the new 'listing.unpublished' event

2025-07-23

`POST` **/v1/listings**

- the request property 'media/images/items/large' was deprecated
- the request property 'media/images/items/medium' was deprecated
- the request property 'media/images/items/thumbnail' was deprecated
- the request property 'media/images/items/watermarked' was deprecated
- `!WARNING!` the request property 'media/images/items/original' became required
- `!WARNING!` the request property 'media/images/items/original/url' became required

`PUT` **/v1/listings/{id}**

- the request property 'media/images/items/large' was deprecated
- the request property 'media/images/items/medium' was deprecated
- the request property 'media/images/items/thumbnail' was deprecated
- the request property 'media/images/items/watermarked' was deprecated
- `!WARNING!` the request property 'media/images/items/original' became required
- `!WARNING!` the request property 'media/images/items/original/url' became required

`POST` **/v1/webhooks**

- added the new 'publicProfile.verification.approved' enum value to the request property 'eventId'

- added the new 'publicProfile.verification.rejected' enum value to the request property 'eventId'

`PATCH` **/v1/users/{id}**

- the request property 'password' was deprecated

2025-07-21

`POST` **/v1/listings/{id}/unpublish**

- endpoint added

2025-07-18

`GET` **/v1/leads**

- added the optional property 'data/items/responseLink' to the response with the '200' status

2025-07-16

`GET` **/v1/listings/{id}/publish/prices**

- endpoint added
- This new endpoint allows to retrieve the publishing price for listings.

2025-07-09

`POST` **/v1/public-profiles/{id}/submit-verification**

- endpoint added

`GET` **/v1/stats/public-profiles**

- endpoint added

2025-06-25

`GET` **/v1/listings**

- Added: `filter[advertisementNumber]` in `query` - This allows filtering of listings by advertisement number (DTCM/RERA)

Auth

Authorization API endpoints

Issue JWT token

Issue JWT token based on API key and secret

REQUEST BODY SCHEMA: application/json

`apiKey` required	string	= 40 characters
`apiSecret` required	string	= 32 characters

Responses

> 200 Auth Token

> 400 Bad Schema Request

> 401 Unauthorized

> 422 Business Validation Error

> 429 Rate Limited

> 500 Internal Server Error

> 502 Bad Gateway

`POST /v1/auth/token`

Request samples

Payload

Content type
application/json

Copy

```
{
    "apiKey": "abcde.abcdefghijklmnopqrstuvwxyz01234567",
    "apiSecret": "abcdefghijklmnopqrstuvwxyz012345"
}
```

Response samples

200 | 400 | 401 | 422 | 429 | 500 | 502

Content type
application/json

Copy

```
{
    "accessToken": "eyJhbGciOiJIUz.....Qssw5c",
    "tokenType": "Bearer",
    "expiresIn": 1800
}
```

Users

User API endpoints

Create user.

AUTHORIZATIONS: > *jwt*

REQUEST BODY SCHEMA: application/json
required

`email` required	string <email> Email address of the user. Used mainly for authentication.
`firstName` required	string >= 2 characters
`lastName` required	string >= 2 characters
`mobile` required	string Mobile phone number of the user. Used mainly for authentication(e.g. 2FA).
`roleId` required	integer Access control role id (e.g. 6 for Decision maker). Refer to Fetch Roles API
`publicProfile` > required	object

Responses

> 201 User Created

> 400 Bad Schema Request

> 401 Unauthorized

> 422 Business Validation Error

> 429 Rate Limited

> 500 Internal Server Error

> 502 Bad Gateway

POST /v1/users/

Request samples

Payload

Content type
application/json

Copy Expand all Collapse all


```
{
    "email": "user@example.com",
    "firstName": "John",
    "lastName": "Doe",
    "mobile": "string",
    "roleId": 0,
  − "publicProfile": {
        "name": "string",
        "email": "user@example.com",
        "phone": "string",
        "phoneSecondary": "string",
        "whatsappPhone": "string",
        "imageUrl": "http://example.com",
      + "bio": { ... },
      + "position": { ... },
        "linkedinAddress": "http://example.com",
        "experienceSince": 2015,
        "nationality": "AD",
      + "spokenLanguages": [ ... ],
      + "compliances": [ ... ]
    }
}
```


Response samples

201 400 401 422 429 500 502

Content type
application/json

Copy Expand all Collapse all


```
{
    "id": 1234,
    "firstName": "John",
```

```
    "lastName": "Doe",
    "email": "abcd@example.com",
    "mobile": "+9710123456789",
    "status": "active",
    "roleId": 3,
  − "publicProfile": {
        "id": 1234,
        "name": "John Doe",
        "email": "john@propertyfinder.ae",
        "phone": "+9715823456789",
        "phoneSecondary": "+9715823456789",
        "whatsappPhone": "+9715823456789",
      + "bio": { ... },
      + "position": { ... },
        "linkedinAddress": "https://www.linkedin.com/in/johndoe",
      + "imageVariants": { ... },
      + "verification": { ... },
      + "compliances": [ ... ],
        "isSuperAgent": true
    }
}
```


Search Users.

Retrieves a list of users associated with a specific client or organisation.

AUTHORIZATIONS: > *jwt*

QUERY PARAMETERS

`page`
integer <int32> `>= 1`
Default: `1`
Example: `page=1`
Page number, starts with 1.

`perPage`
integer <int32> `[ 1 .. 100 ]`
Default: `50`
Example: `perPage=10`
Number of items per page (max 100 allowed)

`search`
string
Example: `search=entity_name`
Search term to filter entity

`status`	string Enum: `"active"` `"inactive"` Example: `status=active` User's status
`roleId`	string Example: `roleId=3` The user's role ID
`publicProfileId`	string Comma separated list of public profile IDs to filter the response.
`id`	string Comma separated list of user IDs to filter the response.
`email`	string Comma separated list of user emails to filter the response.

Responses

- 200 List client's users
- 400 Bad Request Schema
- 401 Unauthorized
- 403 Forbidden
- 429 Rate Limited
- 500 Internal Server Error
- 502 Bad Gateway

```
GET  /v1/users/
```

Response samples

200 | 400 | 401 | 403 | 429 | 500 | 502

Content type
application/json

Copy Expand all Collapse all


```
{
  – "data": [
      + { ... }
    ],
  – "pagination": {
        "total": 100,
        "page": 1,
        "perPage": 10,
        "totalPages": 10,
        "nextPage": 2,
        "prevPage": 1
    }
}
```


Update Private Profile.

Update a user's private profile details. Request body should contain only the fields that need to be updated.

AUTHORIZATIONS: > *jwt*

PATH PARAMETERS

id required	string

REQUEST BODY SCHEMA: application/json
required

firstName	string
lastName	string
email	string <email>
~~password~~	string Deprecated This field is deprecated and will be removed in future versions. This filed DOES NOT update the password of the user.

~~phone~~	string Deprecated This field is deprecated and will be removed in future versions. Please use mobile instead.
mobile	string
roleId	string
isActive	boolean

Responses

— 204 User profile updated successfully, no content to return

› 400 Bad Schema Request

› 401 Unauthorized

› 403 Forbidden

› 404 Not Found

› 422 Business Validation Error

› 429 Rate Limited

› 500 Internal Server Error

PATCH /v1/users/{id}

Request samples

Payload

Content type
application/json

Copy

```
{
    "firstName": "string",
```

```
    "lastName": "string",
    "email": "user@example.com",
    "password": "string",
    "phone": "string",
    "mobile": "string",
    "roleId": "string",
    "isActive": true
}
```


Response samples

400 | 401 | 403 | 404 | 422 | 429 | 500

Content type
application/problem+json

Copy Expand all Collapse all


```
{
    "detail": "",
  – "errors": [
      + { ... }
    ],
    "title": "Bad Request",
    "type": "SCHEMA_VALIDATION"
}
```


Update public profile.

Update a user's public profile details. Request body should contain only the fields that need to be updated.

AUTHORIZATIONS: > *jwt*

PATH PARAMETERS

`id` required	string

REQUEST BODY SCHEMA: application/json required

`name`	string Full name of the user, that will be exposed on the PF Website.

`email`

string <email>
Email of the user, that will be exposed on the PF Website.

`phone`

string
Phone number of the user, that will be exposed on the PF Website.

`phoneSecondary`

string
Secondary phone number of the user, that will be used when the phone number is not reachable.

`whatsappPhone`

string
WhatsApp phone number of the user, that will be exposed on the PF Website.

`imageUrl`

string <uri>
URL of the profile image.

`bio` ›

object
Localized brief biography of the user, displayed on the PF Website. Can include professional background, expertise, or achievements.

`position` ›

object
Localized current job title or position of the user within the company.

`linkedinAddress`

string <uri>
LinkedIn URL of the user.

`experienceSince`

integer
Year when the user started their professional career. Represented as a four-digit year (e.g., 2015).

`nationality`

string

Enum	Description
AF	Afghanistan
AX	Aland Islands
AL	Albania
DZ	Algeria
AS	American Samoa
AD	Andorra
AO	Angola
AI	Anguilla
AQ	Antarctica

Enum	Description
AG	Antigua and Barbuda
AR	Argentina
AM	Armenia
AW	Aruba
AU	Australia
AT	Austria
AZ	Azerbaijan
BS	Bahamas
BH	Bahrain
BD	Bangladesh
BB	Barbados
BY	Belarus
BE	Belgium
BZ	Belize
BJ	Benin
BM	Bermuda
BT	Bhutan
BO	Bolivia
BA	Bosnia and Herzegovina
BW	Botswana
BV	Bouvet Island
BR	Brazil
IO	British Indian Ocean Territory
VG	British Virgin Islands
BN	Brunei Darussalam
BG	Bulgaria
BF	Burkina Faso
BI	Burundi
KH	Cambodia
CM	Cameroon

Enum	Description
CA	Canada
CV	Cape Verde
KY	Cayman Islands
CF	Central African Republic
TD	Chad
CL	Chile
CN	China
CX	Christmas Island
CC	Cocos (Keeling) Islands
CO	Colombia
KM	Comoros
CG	Congo (Brazzaville)
CD	Congo, Democratic Republic of the
CK	Cook Islands
CR	Costa Rica
HR	Croatia
CU	Cuba
CY	Cyprus
CZ	Czech Republic
CI	Côte d'Ivoire
DK	Denmark
DJ	Djibouti
DM	Dominica
DO	Dominican Republic
EC	Ecuador
EG	Egypt
SV	El Salvador
GQ	Equatorial Guinea
ER	Eritrea
EE	Estonia

Enum	Description
ET	Ethiopia
FK	Falkland Islands (Malvinas)
FO	Faroe Islands
FJ	Fiji
FI	Finland
FR	France
GF	French Guiana
PF	French Polynesia
TF	French Southern Territories
GA	Gabon
GM	Gambia
GE	Georgia
DE	Germany
GH	Ghana
GI	Gibraltar
GR	Greece
GL	Greenland
GD	Grenada
GP	Guadeloupe
GU	Guam
GT	Guatemala
GG	Guernsey
GN	Guinea
GW	Guinea-Bissau
GY	Guyana
HT	Haiti
HM	Heard Island and Mcdonald Islands
VA	Holy See (Vatican City State)
HN	Honduras
HK	Hong Kong, Special Administrative Region of China

Enum	Description
HU	Hungary
IS	Iceland
IN	India
ID	Indonesia
IR	Iran
IQ	Iraq
IE	Ireland
IM	Isle of Man
IL	Israel
IT	Italy
JM	Jamaica
JP	Japan
JE	Jersey
JO	Jordan
KZ	Kazakhstan
KE	Kenya
KI	Kiribati
KP	Korea, Democratic People's Republic of
KR	South Korea
XK	Kosovo
KW	Kuwait
KG	Kyrgyzstan
LA	Lao PDR
LV	Latvia
LB	Lebanon
LS	Lesotho
LR	Liberia
LY	Libya
LI	Liechtenstein
LT	Lithuania

Enum	Description
LU	Luxembourg
MO	Macao, Special Administrative Region of China
MK	Macedonia, Republic of
MG	Madagascar
MW	Malawi
MY	Malaysia
MV	Maldives
ML	Mali
MT	Malta
MH	Marshall Islands
MQ	Martinique
MR	Mauritania
MU	Mauritius
YT	Mayotte
MX	Mexico
FM	Micronesia, Federated States of
MD	Moldova
MC	Monaco
MN	Mongolia
ME	Montenegro
MS	Montserrat
MA	Morocco
MZ	Mozambique
MM	Myanmar
NA	Namibia
NR	Nauru
NP	Nepal
NL	Netherlands
AN	Netherlands Antilles
NC	New Caledonia

Enum	Description
NZ	New Zealand
NI	Nicaragua
NE	Niger
NG	Nigeria
NU	Niue
NF	Norfolk Island
MP	Northern Mariana Islands
NO	Norway
OM	Oman
PK	Pakistan
PW	Palau
PS	Palestine
PA	Panama
PG	Papua New Guinea
PY	Paraguay
PE	Peru
PH	Philippines
PN	Pitcairn
PL	Poland
PT	Portugal
PR	Puerto Rico
QA	Qatar
RO	Romania
RU	Russian Federation
RW	Rwanda
RE	Réunion
SH	Saint Helena
KN	Saint Kitts and Nevis
LC	Saint Lucia
PM	Saint Pierre and Miquelon

Enum	Description
VC	Saint Vincent and Grenadines
BL	Saint-Barthélemy
MF	Saint-Martin (French part)
WS	Samoa
SM	San Marino
ST	Sao Tome and Principe
SA	Saudi Arabia
SN	Senegal
RS	Serbia
SC	Seychelles
SL	Sierra Leone
SG	Singapore
SK	Slovakia
SI	Slovenia
SB	Solomon Islands
SO	Somalia
ZA	South Africa
GS	South Georgia and the South Sandwich Islands
SS	South Sudan
ES	Spain
LK	Sri Lanka
SD	Sudan
SR	Suriname *
SJ	Svalbard and Jan Mayen Islands
SZ	Swaziland
SE	Sweden
CH	Switzerland
SY	Syria
TW	Taiwan, Republic of China
TJ	Tajikistan

Enum	Description
TZ	Tanzania
TH	Thailand
TL	Timor-Leste
TG	Togo
TK	Tokelau
TO	Tonga
TT	Trinidad and Tobago
TN	Tunisia
TR	Turkey
TM	Turkmenistan
TC	Turks and Caicos Islands
TV	Tuvalu
UG	Uganda
UA	Ukraine
AE	United Arab Emirates
GB	United Kingdom
UM	United States Minor Outlying Islands
US	United States of America
UY	Uruguay
UZ	Uzbekistan
VU	Vanuatu
VE	Venezuela
VN	Vietnam
VI	Virgin Islands, US
WF	Wallis and Futuna Islands
EH	Western Sahara
YE	Yemen
ZM	Zambia
ZW	Zimbabwe

Nationality of the user (2 letter country code).

spokenLanguages Array of integers or null unique

Enum	Description
1	English
2	Arabic
3	French
4	Polish
5	German
6	Russian
7	Hindi
8	Urdu
9	Croatian
10	Spanish
11	Persian/Farsi
12	Greek
13	Tagalog
14	Bengali
15	Tamil
16	Malayalam
17	Other
18	Kyrgyz
19	Uzbek
20	Kazakh
21	Mandarin
22	Italian
23	Portuguese
24	Dutch
25	Hungarian
26	Azerbaijani
27	Turkish
28	Memon

Enum	Description
29	Gujarati
30	Ukrainian
32	Bulgarian
33	Swedish
34	Romanian
35	Afrikaans
36	Punjabi
37	Danish
38	Serbian
39	Norwegian
40	Cantonese
41	Bahasa Melayu
42	Shona
43	Pashto
44	Albanian
45	Amharic
46	Baluchi
47	Belarusian
48	Berber
49	Catalan
50	Czech
51	Finnish
52	Japanese
53	Javanese
54	Kannada
55	Korean
56	Kurdi
57	Latvian
58	Malay
59	Sinhalese

Enum	Description
60	Slovak
61	Slovene
62	Somali
63	Sudanese
64	Swahili
65	Telugu
66	Thai
67	Macedonian
68	Lithuanian
69	Armenian
70	Sindhi

Array of the spoken languages. Can be empty array.

`compliances` ›

Array of objects or null

An array of compliances. Can be an empty array.

Responses

— **204** Public profile updated successfully, no content to return

› **400** Bad Schema Request

› **401** Unauthorized

› **403** Forbidden

› **404** Not Found

› **422** Business Validation Error

› **429** Rate Limited

› **500** Internal Server Error

`PATCH /v1/public-profiles/{id}`

Request samples

Payload

Content type
application/json

Copy Expand all Collapse all


```
{
    "name": "string",
    "email": "user@example.com",
    "phone": "string",
    "phoneSecondary": "string",
    "whatsappPhone": "string",
    "imageUrl": "http://example.com",
  – "bio": {
        "primary": "string",
        "secondary": "string"
    },
  – "position": {
        "primary": "string",
        "secondary": "string"
    },
    "linkedinAddress": "http://example.com",
    "experienceSince": 2015,
    "nationality": "AD",
  – "spokenLanguages": [
        1
    ],
  – "compliances": [
      + { ... }
    ]
}
```


Response samples

400 401 403 404 422 429 500

Content type
application/problem+json

Copy Expand all Collapse all

```
{
    "detail": "",
  – "errors": [
      + { ... }
    ],
    "title": "Bad Request",
    "type": "SCHEMA_VALIDATION"
}
```


Submit verification request for a public profile.

Public Profile Verification Post request.

AUTHORIZATIONS: > *jwt*

PATH PARAMETERS

`id` required	string

REQUEST BODY SCHEMA: application/json

`phone` required	string Private Profile Phone number of the public profile to be used for verification
`documentUrl` required	string <uri> URL to the verification document - license

Responses

> 200

> 400 Bad Schema Request

> 401 Unauthorized

> 403 Forbidden

> 404 Not Found

> 409 Conflict

> 422 Business Validation Error

> 429 Rate Limited

> 500 Internal Server Error

```
POST   /v1/public-profiles/{id}/submit-verification
```

Request samples

Payload

Content type
application/json

Copy

```
{
    "phone": "string",
    "documentUrl": "http://example.com"
}
```

Response samples

200 | 400 | 401 | 403 | 404 | 409 | 422 | 429 | 500

Content type
application/json

Copy

```
{
    "status": "unverified"
}
```

Roles

Roles API endpoints

Fetch Roles

Returns the list of user roles available in the account. This includes both base roles (such as agent, admin, decision maker, etc), as well as any custom roles that have been created in the account.

AUTHORIZATIONS: > *jwt*

Responses

> 200 List of roles

> 400 Bad Request Schema

> 401 Unauthorized

> 403 Forbidden

> 429 Rate Limited

> 500 Internal Server Error

> 502 Bad Gateway

```
GET /v1/roles
```

Response samples

200 | 400 | 401 | 403 | 429 | 500 | 502

Content type
application/json

Copy Expand all Collapse all


```
{
  – "data": [
      + { … }
    ]
```

```
}
```

Listings

Listing API endpoints

Creates a new listing

Creates a new listing. The request body should contain all the required fields for the listing creation. The response will include the details of the created listing.

Country specific required fields

For each country, there are specific required fields that need to be included in the request body. In order for created listing to get published afterwards, Below is the Required fields and their conditions.

AE-Specific Listing — Required Fields

Field	Required if
`compliance`	When `uaeEmirate` = `dubai` or `uaeEmirate` = `abu_dhabi`
`compliance.listingAdvertismentNumber`	When `uaeEmirate` = `dubai` or `uaeEmirate` = `abu_dhabi`
`compliance.type`	When `uaeEmirate` = `dubai` or `uaeEmirate` = `abu_dhabi`
`category`	Always
`type`	Always
`furnishingType`	Always
`media.images.original`	Always
`price`	Always
`price.type`	Always
`price.amounts`	Include the amount matching `price.type` (e.g., `amounts.daily` for `daily`, `amounts.sale` for `sale`). For rental types (`yearly`,

Field	Required if
	`monthly`, `weekly`, `daily`), other `amounts.*` values are optional.
`downPayment`	When `price.type` = `sale`
`location`	Always
`uaeEmirate`	Always
`reference`	Always (must be unique)
`bedrooms`	Required if listing type is not `Land` or `Farm`
`bathrooms`	Required if listing type is not `Land` or `Farm`
`title.en`	Always
`description.en`	Always
`size`	Always
`hasParkingSpace`	When listing type = `co-working-space`

AUTHORIZATIONS: > *jwt*

REQUEST BODY SCHEMA: application/json

required

age

integer

Number of years since the property was handed over

amenities

Array of strings

Items Enum: `"central-ac"` `"built-in-wardrobes"` `"kitchen-appliances"` `"security"` `"concierge"` `"private-gym"` `"shared-gym"` `"private-jacuzzi"` `"shared-spa"` `"covered-parking"` `"maids-room"` `"barbecue-area"` `"shared-pool"` `"childrens-pool"` `"private-garden"` `"private-pool"` `"view-of-water"` `"walk-in-closet"` `"lobby-in-building"` `"electricity"` `"waters"` `"sanitation"` `"no-services"` `"fixed-phone"` `"fibre-optics"` `"flood-drainage"` `"balcony"` `"networked"` `"view-of-landmark"` `"dining-in-building"` `"conference-room"` `"study"` `"maid-service"` `"childrens-play-area"` `"pets-allowed"` `"vastu-compliant"`

assignedTo >

object

availableFrom

string <date>

`bathrooms`	string Enum: `"none"` `"1"` `"2"` `"3"` `"4"` `"5"` `"6"` `"7"` `"8"` `"9"` `"10"` `"11"` `"12"` `"13"` `"14"` `"15"` `"16"` `"17"` `"18"` `"19"` `"20"`
`bedrooms`	string Enum: `"studio"` `"1"` `"2"` `"3"` `"4"` `"5"` `"6"` `"7"` `"8"` `"9"` `"10"` `"11"` `"12"` `"13"` `"14"` `"15"` `"16"` `"17"` `"18"` `"19"` `"20"`
`category`	string Enum: `"residential"` `"commercial"`
`compliance` ›	object
`createdBy` ›	object This is the public profile id of the user. Refer to Search Users API response for more details - look at the id field, under publicProfile
`description` ›	object If provided, one of `en` or `ar` should be set.
`developer`	string
`finishingType`	string Enum: `"fully-finished"` `"semi-finished"` `"unfinished"`
`floorNumber`	string
`furnishingType`	string Enum: `"unfurnished"` `"semi-furnished"` `"furnished"`
`hasGarden`	boolean
`hasKitchen`	boolean
`hasParkingOnSite`	boolean
`landNumber`	string
`location` ›	object
`media` ›	object
`mojDeedLocationDescription`	string

numberOfFloors	integer
ownerName	string
parkingSlots	integer
plotNumber	string
plotSize	number
price >	object
projectStatus	string Enum: `"completed"` `"off_plan"` `"completed_primary"` `"off_plan_primary"`
reference	string
size	number
street >	object
title >	object If provided, one of `en` or `ar` should be set.
type	string Enum: `"compound"` `"whole-building"` `"factory"` `"land"` `"rest-house"` `"apartment"` `"shop"` `"warehouse"` `"chalet"` `"office-space"` `"full-floor"` `"villa"` `"farm"` `"show-room"` `"bulk-sale-unit"` `"restaurant"` `"roof"` `"half-floor"` `"twin-house"` `"cabin"` `"bulk-rent-unit"` `"ivilla"` `"co-working-space"` `"hotel-apartment"` `"retail"` `"duplex"` `"townhouse"` `"staff-accommodation"` `"medical-facility"` `"palace"` `"penthouse"` `"clinic"` `"cafeteria"` `"bungalow"` `"labor-camp"` `"business-center"`
uaeEmirate	string Enum: `"dubai"` `"abu_dhabi"` `"northern_emirates"`
unitNumber	string

Responses

> 200 Listing response

> 400 Bad Schema Request

> 401 Unauthorized

> 403 Forbidden

> 409 Conflict

> 422 Business Validation Error

> 429 Rate Limited

> 500 Internal Server Error

POST /v1/listings

Request samples

Payload

Content type
application/json

Copy Expand all Collapse all


```
{
    "age": 0,
  – "amenities": [
        "central-ac"
    ],
  – "assignedTo": {
        "id": 0
    },
    "availableFrom": "2019-08-24",
    "bathrooms": "none",
    "bedrooms": "studio",
    "category": "residential",
```

```
– "compliance": {
        "advertisementLicenseIssuanceDate": "2019-08-24T14:15:22Z",
        "listingAdvertisementNumber": "string",
        "type": "rera",
        "userConfirmedDataIsCorrect": true
  },
– "createdBy": {
        "id": 0
  },
– "description": {
        "ar": "string",
        "en": "string"
  },
  "developer": "string",
  "finishingType": "fully-finished",
  "floorNumber": "string",
  "furnishingType": "unfurnished",
  "hasGarden": true,
  "hasKitchen": true,
  "hasParkingOnSite": true,
  "landNumber": "string",
– "location": {
        "id": 0
  },
– "media": {
    + "images": [ … ],
    + "videos": { … }
  },
  "mojDeedLocationDescription": "string",
  "numberOfFloors": 0,
  "ownerName": "string",
  "parkingSlots": 0,
  "plotNumber": "string",
  "plotSize": 0,
```

```
  – "price": {
      + "amounts": { ... },
        "downpayment": 0,
        "minimalRentalPeriod": 0,
      + "mortgage": { ... },
        "numberOfCheques": 0,
        "numberOfMortgageYears": 0,
      + "obligation": { ... },
        "onRequest": true,
      + "paymentMethods": [ ... ],
        "type": "yearly",
        "utilitiesInclusive": true,
      + "valueAffected": { ... }
    },
    "projectStatus": "completed",
    "reference": "string",
    "size": 0,
  – "street": {
        "direction": "North",
        "width": 0
    },
  – "title": {
        "ar": "string",
        "en": "string"
    },
    "type": "compound",
    "uaeEmirate": "dubai",
    "unitNumber": "string"
}
```


Response samples

200 | 400 | 401 | 403 | 409 | 422 | 429 | 500

Content type
application/json

Copy Expand all Collapse all


```
{
    "age": 0,
    "aiContentUsed": true,
  – "amenities": [
        "central-ac"
    ],
```

```
– "assignedTo": {
        "id": 0,
        "name": "string",
    + "photos": { ... }
  },
  "availableFrom": "2019-08-24",
  "bathrooms": "none",
  "bedrooms": "studio",
  "category": "residential",
– "compliance": {
        "advertisementLicenseIssuanceDate": "2019-08-24T14:15:22Z",
        "listingAdvertisementNumber": "string",
    + "regaResponse": { ... },
        "type": "rera",
        "userConfirmedDataIsCorrect": true
  },
  "createdAt": "2019-08-24T14:15:22Z",
– "createdBy": {
        "id": 0,
        "name": "string",
    + "photos": { ... }
  },
  "ctsPriority": 0,
– "description": {
        "ar": "string",
        "en": "string"
  },
  "developer": "string",
  "finishingType": "fully-finished",
  "floorNumber": "string",
  "furnishingType": "unfurnished",
  "hasGarden": true,
  "hasKitchen": true,
  "hasParkingOnSite": true,
  "id": "string",
  "landNumber": "string",
– "location": {
        "id": 0
  },
– "media": {
    + "images": [ ... ],
    + "videos": { ... }
  },
  "mojDeedLocationDescription": "string",
  "numberOfFloors": 0,
  "ownerName": "string",
  "parkingSlots": 0,
```

```
"pfCategoryId": 0,
"pfTypeId": 0,
"plotNumber": "string",
"plotSize": 0,
– "portals": {
  + "propertyfinder": { ... }
},
– "price": {
  + "amounts": { ... },
    "downpayment": 0,
    "minimalRentalPeriod": 0,
  + "mortgage": { ... },
    "numberOfCheques": 0,
    "numberOfMortgageYears": 0,
  + "obligation": { ... },
    "onRequest": true,
  + "paymentMethods": [ ... ],
    "type": "sale",
    "utilitiesInclusive": true,
  + "valueAffected": { ... }
},
– "products": {
  + "featured": { ... },
  + "premium": { ... },
  + "standard": { ... }
},
"projectStatus": "completed",
– "qualityScore": {
    "color": "red",
  + "details": { ... },
    "value": 0
},
"reference": "string",
"size": 0,
– "state": {
  + "reasons": [ ... ],
    "stage": "draft",
    "type": "draft"
},
– "street": {
    "direction": "North",
    "width": 0
},
– "title": {
    "ar": "string",
    "en": "string"
},
```

```
    "type": "bulk-sale-unit",
    "uaeEmirate": "dubai",
    "unitNumber": "string",
    "updatedAt": "2019-08-24T14:15:22Z",
  − "updatedBy": {
        "id": 0,
        "name": "string",
      + "photos": { ... }
    },
    "verificationStatus": "string"
}
```


Search Listings

AUTHORIZATIONS: > *jwt*

QUERY PARAMETERS

`draft`	boolean Example: `draft=true` Get published or non-published listings (default will return published listings)
`archived`	boolean Example: `archived=true` Get archived listings
`isCtsEligible`	boolean Default: `true` Example: `isCtsEligible=false` Get listings that are eligible for CTS
`filter[state]`	string Enum: `"draft"` `"live"` `"takendown"` `"archived"` `"unpublished"` `"pending_approval"` `"rejected"` `"approved"` `"failed"` Filter by listing state. States are different based on the draft parameter
`filter[ids]`	string Example: `filter[ids]=XXXXX,YYYYY,ZZZZZ` Comma separated list of listing ids

`filter[reference]`	string
`filter[listingLevel]`	string Enum: `"featured"` `"premium"` `"standard"` Filter by listing level
`filter[completionStatus]`	string Enum: `"completed"` `"completed_primary"` `"off_plan"` `"off_plan_primary"` Filter by completion status
`filter[furnishingType]`	string Enum: `"furnished"` `"unfurnished"` `"semi-furnished"` Filter by furnished type
`filter[locationId]`	string Example: `filter[locationId]=123,456,789` Comma separated list of location ids
`filter[assignedToId]`	string Example: `filter[assignedToId]=123,456,789` Comma separated list of public profile ids, assigned to the listing
`filter[type]`	string Filter listing by type (apartment, villa, etc)
`filter[category]`	string Enum: `"commercial"` `"residential"` Filter by listing category
`filter[offeringType]`	string Enum: `"rent"` `"sale"` Filter by offering type
`filter[bedrooms]`	string Example: `filter[bedrooms]=studio,2,3` Filter listings by number of bedrooms.
`filter[bathrooms]`	string Example: `filter[bathrooms]=1,2,3` Filter listings by number of bathrooms.
`filter[unitNumber]`	string Example: `filter[unitNumber]=123,B-923` Filter listings by unit number.

`filter[advertisementNumber]`	string Example: `filter[advertisementNumber]=1234,5678` Filter by listing advertisement number (filter by RERA and DTCM number)
`filter[size][from]`	number Example: `filter[size][from]=1000` Filter listings by size starting from
`filter[size][to]`	number Example: `filter[size][to]=2000` Filter listings by size ending at
`filter[price][from]`	number Example: `filter[price][from]=1000` Filter listings by price starting from
`filter[price][to]`	number Example: `filter[price][to]=2000` Filter listings by price ending at
`filter[listingLevelExpiresAt][from]`	string <date-time> Example: `filter[listingLevelExpiresAt][from]=2025-04-30T09:54:18+04:00` Filter by listing level expiration date (from). From and to should be used together.
`filter[listingLevelExpiresAt][to]`	string <date-time> Example: `filter[listingLevelExpiresAt][to]=2025-04-30T09:54:18+04:00` Filter by listing level expiration date (to). From and to should be used together.
`filter[projectStatus]`	string Enum: `"completed"` `"completed_primary"` `"off_plan"` `"off_plan_primary"` Filter by projectStatus
`filter[spotlightStatus]`	string Enum: `"eligible"` `"outcompeted"` `"spotlight"` `"winning_offer"` `"expired"` `"not_eligible"` Filter by listing spotlight status
`filter[ctsPriority]`	integer Enum: `100` `0` Filter by listing cts priority

`filter[verificationStatus]`

string
Enum: `"pending"` `"approved"` `"rejected"` `"expired"` `"deleted"`
Filter by listing verification status

`perPage`

integer <int32> `[ 1 .. 100 ]`
Default: `50`
Example: `perPage=10`
Number of items per page (max 100 allowed)

`page`

integer <int32> `>= 1`
Default: `1`
Example: `page=1`
Page number, starts with 1.

`orderBy`

string
Enum: `"createdAt"` `"price"` `"publishedAt"`
Example: `orderBy=price`
Field to order by

`sort[createdAt]`

string
Enum: `"asc"` `"desc"`
Sort by createdAt

`sort[price]`

string
Enum: `"asc"` `"desc"`
Sort by price

`sort[publishedAt]`

string
Enum: `"asc"` `"desc"`
Sort by publishedAt

Responses

> 200 Listings search response

> 400 Bad Schema Request

> 401 Unauthorized

> 403 Forbidden

> 422 Business Validation Error

> 429 Rate Limited

› 500 Internal Server Error

GET /v1/listings

Response samples

200 | 400 | 401 | 403 | 422 | 429 | 500

Content type
application/json

Copy Expand all Collapse all


```
{
  – "results": [
      + { ... }
    ],
  – "pagination": {
        "total": 100,
        "page": 1,
        "perPage": 10,
        "totalPages": 10,
        "nextPage": 2,
        "prevPage": 1
    },
  – "activeCts": {
      + "locations": [ ... ],
      + "categories": [ ... ],
      + "offeringTypes": [ ... ]
    }
}
```


Updates an existing listing

AUTHORIZATIONS: › *jwt*

PATH PARAMETERS

id required	string

REQUEST BODY SCHEMA: application/json

required

age

integer

Number of years since the property was handed over

amenities

Array of strings

Items Enum: `"central-ac"` `"built-in-wardrobes"` `"kitchen-appliances"` `"security"` `"concierge"` `"private-gym"` `"shared-gym"` `"private-jacuzzi"` `"shared-spa"` `"covered-parking"` `"maids-room"` `"barbecue-area"` `"shared-pool"` `"childrens-pool"` `"private-garden"` `"private-pool"` `"view-of-water"` `"walk-in-closet"` `"lobby-in-building"` `"electricity"` `"waters"` `"sanitation"` `"no-services"` `"fixed-phone"` `"fibre-optics"` `"flood-drainage"` `"balcony"` `"networked"` `"view-of-landmark"` `"dining-in-building"` `"conference-room"` `"study"` `"maid-service"` `"childrens-play-area"` `"pets-allowed"` `"vastu-compliant"`

assignedTo ›

object

availableFrom

string <date>

bathrooms

string

Enum: `"none"` `"1"` `"2"` `"3"` `"4"` `"5"` `"6"` `"7"` `"8"` `"9"` `"10"` `"11"` `"12"` `"13"` `"14"` `"15"` `"16"` `"17"` `"18"` `"19"` `"20"`

bedrooms

string

Enum: `"studio"` `"1"` `"2"` `"3"` `"4"` `"5"` `"6"` `"7"` `"8"` `"9"` `"10"` `"11"` `"12"` `"13"` `"14"` `"15"` `"16"` `"17"` `"18"` `"19"` `"20"`

category

string

Enum: `"residential"` `"commercial"`

compliance ›

object

createdBy ›

object

This is the public profile id of the user. Refer to Search Users API response for more details - look at the id field, under publicProfile

description ›

object

If provided, one of `en` or `ar` should be set.

developer	string
finishingType	string Enum: `"fully-finished"` `"semi-finished"` `"unfinished"`
floorNumber	string
furnishingType	string Enum: `"unfurnished"` `"semi-furnished"` `"furnished"`
hasGarden	boolean
hasKitchen	boolean
hasParkingOnSite	boolean
landNumber	string
location ›	object
media ›	object
mojDeedLocationDescription	string
numberOfFloors	integer
ownerName	string
parkingSlots	integer
plotNumber	string
plotSize	number
price ›	object
projectStatus	string Enum: `"completed"` `"off_plan"` `"completed_primary"` `"off_plan_primary"`
reference	string
size	number
street ›	object
title ›	object

If provided, one of `en` or `ar` should be set.

`type`	string Enum: `"compound"` `"whole-building"` `"factory"` `"land"` `"rest-house"` `"apartment"` `"shop"` `"warehouse"` `"chalet"` `"office-space"` `"full-floor"` `"villa"` `"farm"` `"show-room"` `"bulk-sale-unit"` `"restaurant"` `"roof"` `"half-floor"` `"twin-house"` `"cabin"` `"bulk-rent-unit"` `"ivilla"` `"co-working-space"` `"hotel-apartment"` `"retail"` `"duplex"` `"townhouse"` `"staff-accommodation"` `"medical-facility"` `"palace"` `"penthouse"` `"clinic"` `"cafeteria"` `"bungalow"` `"labor-camp"` `"business-center"`
`uaeEmirate`	string Enum: `"dubai"` `"abu_dhabi"` `"northern_emirates"`
`unitNumber`	string

Responses

- 200 Listing response
- 400 Bad Schema Request
- 401 Unauthorized
- 403 Forbidden
- 404 Not Found
- 422 Business Validation Error
- 429 Rate Limited
- 500 Internal Server Error

`PUT /v1/listings/{id}`

Request samples

Payload

Content type
application/json

Copy Expand all Collapse all


```
{
  "age": 0,
  – "amenities": [
      "central-ac"
    ],
  – "assignedTo": {
      "id": 0
    },
    "availableFrom": "2019-08-24",
    "bathrooms": "none",
    "bedrooms": "studio",
    "category": "residential",
  – "compliance": {
      "advertisementLicenseIssuanceDate": "2019-08-24T14:15:22Z",
      "listingAdvertisementNumber": "string",
      "type": "rera",
      "userConfirmedDataIsCorrect": true
    },
  – "createdBy": {
      "id": 0
    },
  – "description": {
      "ar": "string",
      "en": "string"
    },
    "developer": "string",
    "finishingType": "fully-finished",
    "floorNumber": "string",
    "furnishingType": "unfurnished",
    "hasGarden": true,
    "hasKitchen": true,
    "hasParkingOnSite": true,
    "landNumber": "string",
  – "location": {
      "id": 0
    },
  – "media": {
    + "images": [ ... ],
    + "videos": { ... }
    },
    "mojDeedLocationDescription": "string",
```

```
    "numberOfFloors": 0,
    "ownerName": "string",
    "parkingSlots": 0,
    "plotNumber": "string",
    "plotSize": 0,
  – "price": {
    + "amounts": { ... },
      "downpayment": 0,
      "minimalRentalPeriod": 0,
    + "mortgage": { ... },
      "numberOfCheques": 0,
      "numberOfMortgageYears": 0,
    + "obligation": { ... },
      "onRequest": true,
    + "paymentMethods": [ ... ],
      "type": "yearly",
      "utilitiesInclusive": true,
    + "valueAffected": { ... }
    },
    "projectStatus": "completed",
    "reference": "string",
    "size": 0,
  – "street": {
      "direction": "North",
      "width": 0
    },
  – "title": {
      "ar": "string",
      "en": "string"
    },
    "type": "compound",
    "uaeEmirate": "dubai",
    "unitNumber": "string"
}
```


Response samples

200 | 400 | 401 | 403 | 404 | 422 | 429 | 500

Content type
application/json

Copy Expand all Collapse all


```
{
    "age": 0,
    "aiContentUsed": true,
```

```
– "amenities": [
        "central-ac"
  ],
– "assignedTo": {
        "id": 0,
        "name": "string",
    + "photos": { ... }
  },
  "availableFrom": "2019-08-24",
  "bathrooms": "none",
  "bedrooms": "studio",
  "category": "residential",
– "compliance": {
        "advertisementLicenseIssuanceDate": "2019-08-24T14:15:22Z",
        "listingAdvertisementNumber": "string",
    + "regaResponse": { ... },
        "type": "rera",
        "userConfirmedDataIsCorrect": true
  },
  "createdAt": "2019-08-24T14:15:22Z",
– "createdBy": {
        "id": 0,
        "name": "string",
    + "photos": { ... }
  },
  "ctsPriority": 0,
– "description": {
        "ar": "string",
        "en": "string"
  },
  "developer": "string",
  "finishingType": "fully-finished",
  "floorNumber": "string",
  "furnishingType": "unfurnished",
  "hasGarden": true,
  "hasKitchen": true,
  "hasParkingOnSite": true,
  "id": "string",
  "landNumber": "string",
– "location": {
        "id": 0
  },
– "media": {
    + "images": [ ... ],
    + "videos": { ... }
  },
  "mojDeedLocationDescription": "string",
```

```
"numberOfFloors": 0,
"ownerName": "string",
"parkingSlots": 0,
"pfCategoryId": 0,
"pfTypeId": 0,
"plotNumber": "string",
"plotSize": 0,
– "portals": {
  + "propertyfinder": { ... }
},
– "price": {
  + "amounts": { ... },
    "downpayment": 0,
    "minimalRentalPeriod": 0,
  + "mortgage": { ... },
    "numberOfCheques": 0,
    "numberOfMortgageYears": 0,
  + "obligation": { ... },
    "onRequest": true,
  + "paymentMethods": [ ... ],
    "type": "sale",
    "utilitiesInclusive": true,
  + "valueAffected": { ... }
},
– "products": {
  + "featured": { ... },
  + "premium": { ... },
  + "standard": { ... }
},
"projectStatus": "completed",
– "qualityScore": {
    "color": "red",
  + "details": { ... },
    "value": 0
},
"reference": "string",
"size": 0,
– "state": {
  + "reasons": [ ... ],
    "stage": "draft",
    "type": "draft"
},
– "street": {
    "direction": "North",
    "width": 0
},
```

```
  – "title": {
        "ar": "string",
        "en": "string"
    },
    "type": "bulk-sale-unit",
    "uaeEmirate": "dubai",
    "unitNumber": "string",
    "updatedAt": "2019-08-24T14:15:22Z",
  – "updatedBy": {
        "id": 0,
        "name": "string",
      + "photos": { ... }
    },
    "verificationStatus": "string"
}
```


Deletes a listing

AUTHORIZATIONS: > *jwt*

PATH PARAMETERS

`id` required	string

Responses

> 200 Pending deletion

— 204 Deleted

> 401 Unauthorized

> 403 Forbidden

> 404 Not Found

> 429 Rate Limited

› 500 Internal Server Error

DELETE /v1/listings/{id}

Response samples

200 | 401 | 403 | 404 | 429 | 500

Content type
application/json

Copy Expand all Collapse all


```
{
    "id": "string",
  – "state": {
        "type": "live_pending_deletion",
        "stage": "live",
      + "reasons": [ ... ]
    }
}
```


Calculates the price for a listing to be published as standard

AUTHORIZATIONS: › *jwt*

PATH PARAMETERS

`id` required	string The listing ID for which to calculate the publish price.

Responses

› 200 Listing Publish Prices response

› 400 Bad Schema Request

› 401 Unauthorized

› 403 Forbidden

› 404 Not Found

› 422 Business Validation Error

› 429 Rate Limited

› 500 Internal Server Error

GET /v1/listings/{id}/publish/prices

Response samples

200 | 400 | 401 | 403 | 404 | 422 | 429 | 500

Content type
application/json

Copy Expand all Collapse all


```
[
 − {
      "feature": "publish",
    + "purchasableProducts": [ … ]
   }
]
```


Publishes Listing To PropertyFinder website

AUTHORIZATIONS: › *jwt*

PATH PARAMETERS

id required	string

Responses

> 200 Listing state response

> 400 Bad Schema Request

> 401 Unauthorized

> 403 Forbidden

> 404 Not Found

> 422 Business Validation Error

> 429 Rate Limited

> 500 Internal Server Error

POST /v1/listings/{id}/publish

Response samples

200 | 400 | 401 | 403 | 404 | 422 | 429 | 500

Content type
application/json

Copy Expand all Collapse all


```
{
  – "state": {
        "type": "draft",
        "stage": "draft",
      + "reasons": [ ... ]
    },
    "id": "string"
}
```


Unpublishes Listing from PropertyFinder website

AUTHORIZATIONS: > *jwt*

PATH PARAMETERS

id required	string

Responses

> 200 Listing state response

> 400 Bad Schema Request

> 401 Unauthorized

> 403 Forbidden

> 404 Not Found

> 422 Business Validation Error

> 429 Rate Limited

> 500 Internal Server Error

POST /v1/listings/{id}/unpublish

Response samples

200 | 400 | 401 | 403 | 404 | 422 | 429 | 500

Content type
application/json

Copy Expand all Collapse all


```
{
  – "state": {
        "type": "draft",
        "stage": "draft",
      + "reasons": [ ... ]
    },
    "id": "string"
}
```

Upgrade listing

Upgrade a listing to premium or featured

AUTHORIZATIONS: > *jwt*

PATH PARAMETERS

`id` required	string

REQUEST BODY SCHEMA: application/json
required

Listing upgrades request

`featureId` required	string Enum: `"featured"` `"premium"` The feature to apply to the listing
`productId` required	string Product ID to purchase. Obtained from GET /upgrades response under 'purchasableProducts[].id'. Use this to buy a new boost.
`renewalEnabled`	boolean Default: `true` Whether to enable automatic renewal when the upgrade expires

Responses

> 200 Listing state response

> 400 Bad Schema Request

> 401 Unauthorized

> 403 Forbidden

> 404 Not Found

> 422 Business Validation Error

> 429 Rate Limited

> 500 Internal Server Error

`POST /v1/listings/{id}/upgrades`

Request samples

Payload

Content type
application/json

Copy


```
{
    "featureId": "featured",
    "productId": "string",
    "renewalEnabled": true
}
```


Response samples

200 | 400 | 401 | 403 | 404 | 422 | 429 | 500

Content type
application/json

Copy Expand all Collapse all


```
{
  – "state": {
        "type": "draft",
        "stage": "draft",
      + "reasons": [ ... ]
    },
    "id": "string"
}
```


Available upgrades

Retrieve a list of available upgrade options for a specific listing

AUTHORIZATIONS: > *jwt*

PATH PARAMETERS

id required	string

Responses

> 200 Listing Upgrades response

> 400 Bad Schema Request

> 401 Unauthorized

> 403 Forbidden

> 422 Business Validation Error

> 429 Rate Limited

> 500 Internal Server Error

```
GET  /v1/listings/{id}/upgrades
```

Response samples

200 | 400 | 401 | 403 | 422 | 429 | 500

Content type
application/json

Copy Expand all Collapse all


```
{
  – "options": [
      + { … }
    ]
}
```

Compliances

Compliance API endpoints

Get permit details by permit type, number and license

Returns either the property permit data or if project permit exists, the corresponding project details.

AUTHORIZATIONS: › *jwt*

PATH PARAMETERS

`permitNumber`
required
string
The DLD listing (permit) number to validate

`licenseNumber`
required
string
The company license number

QUERY PARAMETERS

`permitType`
required
string
The listing (permit) type to validate.

HEADER PARAMETERS

`Accept-Language`
string
Default: `en`
Enum: `"en"` `"ar"`
Example: `en`
Language code

Responses

› **200** Successful response with either property or project data

› **400** Bad Request Schema

› **401** Unauthorized

> 403 Forbidden

> 422 Business Validation Error

> 429 Rate Limited

> 500 Internal Server Error

> 502 Bad Gateway

```
GET  /v1/compliances/{permitNumber}/{licenseNumber}
```

Response samples

200 | 400 | 401 | 403 | 422 | 429 | 500 | 502

Content type
application/json

Copy Expand all Collapse all


```
{
    "status": "success",
  − "data": [
      + { ... }
    ]
}
```


Listing verifications

List listing verification submissions

This endpoint allows clients to retrieve a comprehensive and paginated list of listing verification submissions within the system. It supports advanced filtering through various query parameters such as submission ID, listing ID, listing reference, agent broker ID, status, and time ranges

(createdAt, updatedAt, expiresAt). Clients can use these filters to narrow down the results to only the relevant submissions they are interested in.
To provide more flexibility and context in the response, the API supports include parameters for embedding related resources such as submitted documents and historical status transitions. This avoids the need for additional round trips to fetch related data.
In addition, the endpoint supports customizable sorting (e.g., by creation date, status, expiration) and pagination controls to efficiently handle large data sets.

AUTHORIZATIONS: > *jwt*

QUERY PARAMETERS

`filter[id]`
string
Comma-separated list of submission IDs.

`filter[listingId]`
string
Comma-separated list of listing IDs.

`filter[listingReference]`
string
Comma-separated list of listing references.

`filter[publicProfileId]`
string
Comma-separated list of publicProfile IDs.

`filter[status]`
string
Example: `filter[status]=pending,approved,rejected,expired`
Comma-separated list of statuses.

`filter[createdAt][from]`
string <date-time>
Example: `filter[createdAt][from]=2025-04-30T09:54:18+04:00`
Date and time in RFC 3339 format: `YYYY-MM-DDTHH:MM:SSZ`

`filter[createdAt][to]`
string <date-time>
Example: `filter[createdAt][to]=2025-04-30T09:54:18+04:00`
Date and time in RFC 3339 format: `YYYY-MM-DDTHH:MM:SSZ`

`filter[updatedAt][from]`
string <date-time>
Example: `filter[updatedAt][from]=2025-04-30T09:54:18+04:00`
Date and time in RFC 3339 format: `YYYY-MM-DDTHH:MM:SSZ`

`filter[updatedAt][to]`
string <date-time>
Example: `filter[updatedAt][to]=2025-04-30T09:54:18+04:00`
Date and time in RFC 3339 format: `YYYY-MM-DDTHH:MM:SSZ`

`filter[expiresAt][from]`	string <date-time> Example: `filter[expiresAt][from]=2025-04-30T09:54:18+04:00` Date and time in RFC 3339 format: `YYYY-MM-DDTHH:MM:SSZ`
`filter[expiresAt][to]`	string <date-time> Example: `filter[expiresAt][to]=2025-04-30T09:54:18+04:00` Date and time in RFC 3339 format: `YYYY-MM-DDTHH:MM:SSZ`
`include`	string Example: `include=document,history` Comma-separated list of related entities to include in the response.
`page`	integer <int32> `>= 1` Default: `1` Example: `page=1` Page number, starts with 1.
`perPage`	integer <int32> `[ 1 .. 100 ]` Default: `50` Example: `perPage=10` Number of items per page (max 100 allowed)
`orderBy`	string Example: `orderBy=name` Field to order by

Responses

> 200 Successful response containing submission data.

> 400 Bad Schema Request

> 401 Unauthorized

`GET /v1/listing-verifications`

Response samples

200 400 401

Content type
application/json

Copy Expand all Collapse all


```
{
  – "submissions": [
      + { ... }
    ],
  – "pageMetadata": {
        "total": 100,
        "page": 1,
        "perPage": 10,
        "totalPages": 10,
        "nextPage": 2,
        "prevPage": 1
    }
}
```


Create a new listing verification submission

This endpoint allows clients to initiate a new listing verification submission by providing the listing details, agent/broker identity, and supporting documents grouped by category.
The request must include the listingId and agentBrokerId fields, which associate the submission with a specific listing and the responsible agent or broker. Optionally, clients can attach categorized documents to support the verification process. Document categories include: authorization, ownership, identification, representationPao, representationId, and others.

REQUEST BODY SCHEMA: application/json

`listingId` required	string
`publicProfileId` required	integer
`documents` ›	object

Responses

› 201 Successful response containing submission data.

> 400 Bad Schema Request

> 401 Unauthorized

> 422 Business Validation Error

> 429 Rate Limited

> 500 Internal Server Error

> 502 Bad Gateway

POST /v1/listing-verifications

Request samples

Payload

Content type
application/json

Copy Expand all Collapse all


```
{
    "listingId": "string",
    "publicProfileId": 0,
  – "documents": {
      + "authorization": [ ... ],
      + "ownership": [ ... ],
      + "identification": [ ... ],
      + "representationPao": [ ... ],
      + "representationId": [ ... ],
      + "others": [ ... ]
    }
}
```


Response samples

201 400 401 422 429 500 502

Content type
application/json

Copy


```
{
    "submissionId": 0
}
```


Resubmit a previously rejected submission

This endpoint allows clients to resubmit a previously rejected verification submission, but it is strictly limited to auto submissions — a type of submission that is automatically generated by the system without requiring supporting documents from the user.
Auto submissions are typically created when a listing is eligible for automated verification, and they are often rejected due to issues that can be resolved without requiring the agent or broker to manually upload new documentation.
When using this endpoint, clients must provide the listingId of the listing whose auto submission needs to be resubmitted. The system will locate the most recent rejected auto submission for that listing and, if found and eligible, transition its status back to pending. This re-initiates the verification process without requiring the user to create a brand new submission or re-upload any documents.
This endpoint is useful in scenarios such as: • The rejection reason was temporary (e.g., backend data mismatch that has since been resolved) • The listing has been updated and now satisfies automated checks • A manual override or external fix has made the listing verifiable again

AUTHORIZATIONS: › *jwt*

PATH PARAMETERS

`submissionId` required	integer <int64> The ID of the listing verification submission to be resubmitted.

Responses

— **200** Submission status changed to pending.

› **400** Bad Schema Request

› **401** Unauthorized

› **404** Not Found

POST /v1/listing-verifications/{submissionId}/resubmit

Response samples

400 | 401 | 404

Content type
application/problem+json

Copy Expand all Collapse all


```
{
    "detail": "",
  – "errors": [
      + { ... }
    ],
    "title": "Bad Request",
    "type": "SCHEMA_VALIDATION"
}
```


Check listing verification eligibility

This endpoint allows clients to check whether a specific listing qualifies to begin the verification process. Before a verification submission is created—especially for auto submissions that don't require documents—this check determines whether the listing meets all necessary criteria.
The evaluation includes both technical and business rules such as: • Listing location (e.g., certain cities may be excluded) • Listing quality metrics (like score thresholds) • Agent or broker eligibility • Duplicate submission prevention
The response includes two key flags: • eligible: Indicates whether the listing qualifies for any kind of verification. • autoSubmit: Indicates whether the listing is eligible for auto submission (which requires no documents or manual input).
If eligible is false, the response may also contain a helpDetails object to help the client understand why the listing is not eligible and how to resolve the issue.
This check should be performed before attempting to create or resubmit a verification

AUTHORIZATIONS: > *jwt*

REQUEST BODY SCHEMA: application/json

`listingId` required	string

Responses

› 200 Successful response containing eligibility check data.

— 400 Indicates the request is invalid due to missing or incorrect parameters.

— 401 Returned when the client is not authorized to access this endpoint.

— 403 Client or agent permissions do not meet the eligibility criteria.

— 500 Indicates an issue occurred on the server while processing the request.

```
POST   /v1/listing-verifications/eligibility-check
```

Request samples

Payload

Content type
application/json

Copy

```
{
    "listingId": "string"
}
```

Response samples

200

Content type
application/json

Copy Expand all Collapse all

```
{
    "eligible": true,
    "autoSubmit": true,
    "helpDetails": { }
}
```

Locations

Location API endpoints

Locations list

Retrieve a list of locations

AUTHORIZATIONS: > *jwt*

QUERY PARAMETERS

`page`
integer <int32> >= 1
Default: 1
Example: page=1
Page number, starts with 1.

`perPage`
integer <int32> [1 .. 100]
Default: 50
Example: perPage=10
Number of items per page (max 100 allowed)

`search`
required
string >= 2 characters
Example: search=entity_name
Search term to filter entity

`filter[id]`
string
Example: filter[id]=1234
Location ID to filter the response.

`filter[type]`
string
Enum: "REGION" "GOVERNORATE" "CITY" "TOWN" "VILLAGE" "DISTRICT" "STREET" "COMMUNITY" "SUBCOMMUNITY" "PROJECT" "TOWER" "COMPOUND" "AREA" "PROVINCE" "SUBDISTRICT"
Example: filter[type]=AREA
Location type to filter the response.

HEADER PARAMETERS

`Accept-Language`
string
Default: en
Enum: "en" "ar"
Example: en

Language code

Responses

- 200 List of locations
- 400 Bad Request Schema
- 401 Unauthorized
- 403 Forbidden
- 429 Rate Limited
- 500 Internal Server Error
- 502 Bad Gateway

```
GET  /v1/locations
```

Response samples

200 | 400 | 401 | 403 | 429 | 500 | 502

Content type
application/json

Copy Expand all Collapse all

```
{
  – "data": [
      + { ... }
    ],
  – "pagination": {
        "total": 100,
        "page": 1,
        "perPage": 10,
        "totalPages": 10,
        "nextPage": 2,
        "prevPage": 1
    }
}
```


Leads

Lead API endpoints

Fetch leads

Fetch leads, filtering on type, status, date, assigned user/agent etc.

AUTHORIZATIONS: > *jwt*

QUERY PARAMETERS

`page` — integer <int32> `>= 1`
Default: `1`
Example: `page=1`
Page number, starts with 1.

`perPage` — number `[ 1 .. 50 ]`
Default: `50`
Example: `perPage=20`
Number of items per page (max 50 allowed)

`orderBy` — string
Example: `orderBy=name`
Field to order by

`orderDirection` — string

Enum: `"asc"` `"desc"`
Example: `orderDirection=asc`
Order direction

`search`

string
Example: `search=entity_name`
Search term to filter entity

`id`

Array of strings

`status`

Array of strings
Items Enum: `"sent"` `"delivered"` `"read"` `"replied"`

`channel`

Array of strings
Items Enum: `"whatsapp"` `"email"` `"call"`

`entityType`

Array of strings
Items Enum: `"listing"` `"project"` `"developer"` `"agent"` `"company"`

`publicProfileId`

Array of integers

`listingId`

Array of strings

`listingReference`

Array of strings

`listingCategory`

Array of strings
Items Enum: `"commercial"` `"residential"`

`listingOffering`

Array of strings
Items Enum: `"sale"` `"rent"`

`developerId`

Array of strings

`projectId`

Array of strings

`senderName`

Array of strings

`senderPhone`

Array of strings

`senderEmail`

Array of strings

`tag`

Array of strings

`excludeTag`

Array of strings

`createdAtFrom`

string <date-time>
Example: `createdAtFrom=2025-04-30T09:54:18+04:00`
Date and time in RFC 3339 format: YYYY-MM-DDTHH:MM:SSZ

Optional filter to specify the earliest creation date. **The date must not be older than 3 months from the current date.**

`createdAtTo`

string <date-time>
Example: `createdAtTo=2025-04-30T09:54:18+04:00`
Date and time in RFC 3339 format: `YYYY-MM-DDTHH:MM:SSZ`

Responses

- **200** List of leads
- **400** Bad Request Schema
- **401** Unauthorized
- **403** Forbidden
- **429** Rate Limited
- **500** Internal Server Error
- **502** Bad Gateway

`GET /v1/leads`

Response samples

200 | 400 | 401 | 403 | 429 | 500 | 502

Content type
application/json

Copy Expand all Collapse all


```
{
  – "data": [
      + { … }
    ],
```

```
  – "pagination": {
        "total": 100,
        "page": 1,
        "perPage": 10,
        "totalPages": 10,
        "nextPage": 2,
        "prevPage": 1
    }
}
```


Projects

Projects API endpoints

Get project details

Retrieve detailed information about a specific project

AUTHORIZATIONS: › *jwt*

PATH PARAMETERS

`id` required	string

HEADER PARAMETERS

`Accept-Language`	string Default: `en` Enum: `"en"` `"ar"` Example: `en` Language code

Responses

› **200** Project details retrieved successfully

› **401** Unauthorized

> 403 Forbidden

> 404 Not Found

> 429 Rate Limited

> 500 Internal Server Error

> 502 Bad Gateway

```
GET  /v1/projects/{id}
```

Response samples

200 | 401 | 403 | 404 | 429 | 500 | 502

Content type
application/json

Copy Expand all Collapse all


```
{
    "id": "string",
  – "title": {
        "en": "string",
        "ar": "string"
    },
  – "developer": {
        "id": "string",
      + "name": { ... }
    },
  – "location": {
        "id": "string",
      + "name": { ... }
    },
    "dldId": "string",
    "startingPrice": "string"
}
```


Statistics

Statistics API endpoints

Get Public Profile Statistics data (only available for countries where SuperAgent is launched (e.g. AE, QA, BH etc.)

Get Public Profile Statistics data

AUTHORIZATIONS: > *jwt*

QUERY PARAMETERS

`search`
string
Example: `search=entity_name`
Search term to filter entity

`page`
integer <int32> `>= 1`
Default: `1`
Example: `page=1`
Page number, starts with 1.

`perPage`
integer <int32> `[ 1 .. 100 ]`
Default: `50`
Example: `perPage=10`
Number of items per page (max 100 allowed)

`orderBy`
string
Example: `orderBy=name`
Field to order by

`orderDirection`
string
Enum: `"asc"` `"desc"`
Example: `orderDirection=asc`
Order direction

Responses

> 200 Get agent statistics response

> 400 Bad Schema Request

> 401 Unauthorized

> 403 Forbidden

> 404 Not Found

> 422 Business Validation Error

> 429 Rate Limited

> 500 Internal Server Error

```
GET  /v1/stats/public-profiles
```

Response samples

200 | 400 | 401 | 403 | 404 | 422 | 429 | 500

Content type
application/json

Copy Expand all Collapse all


```
{
  – "data": [
      + { ... }
    ],
  – "pagination": {
        "total": 100,
        "page": 1,
        "perPage": 10,
        "totalPages": 10,
        "nextPage": 2,
        "prevPage": 1
    },
  – "sort": {
        "by": "string",
        "direction": "asc"
    }
}
```


Credits

Get credit balance

Retrieves the credit balance information including the including total, remaining, and used credits, as well as the billing cycle dates.

AUTHORIZATIONS: > *jwt*

QUERY PARAMETERS

`publicProfileId` integer

The public profile ID of the user to retrieve the balance for. If left empty, the credit balance information for the company is returned.

Responses

> 200 Credit balance response

> 400 Bad Request Schema

> 401 Unauthorized

> 403 Forbidden

> 422 Business Validation Error

> 429 Rate Limited

> 500 Internal Server Error

> 502 Bad Gateway

```
GET  /v1/credits/balance
```

Response samples

200 | 400 | 401 | 403 | 422 | 429 | 500 | 502

Content type
application/json

Copy Expand all Collapse all


```
{
    "total": 138361,
    "remaining": 111278,
    "used": 27083,
  – "cycle": {
        "startDate": "2019-08-24T14:15:22Z",
        "endDate": "2019-08-24T14:15:22Z"
    }
}
```


API

Overview

Webhooks are HTTP callbacks that are triggered by specific events in the system. They allow you to receive real-time notifications about changes or actions that occur within the system, such as lead creation, updates, or other relevant events.
Webhooks are typically used to integrate with external systems or services, enabling you to automate workflows and respond to events as they happen.
For example, when a new lead is created, a webhook can be triggered to send the lead data to an external CRM system or notify a third-party service.
Webhooks are configured to send HTTP POST requests to a specified URL, containing relevant data about the event that occurred. This allows you to process the data in real-time and take appropriate actions based on the received information.

Subscription Multiplicity

It is possible to create multiple subscriptions for the same event type. Each subscription operates independently and will receive a separate delivery when the corresponding event is emitted.

Security

HMAC Signature

If a **secret** was defined during the event subscription setup, each event delivery will include an HMAC-SHA256 signature.

- The signature is computed using the full **event payload** as input and the provided secret as the key.
- The resulting signature is sent in the `X-Signature` HTTP header as a hexadecimal string.
- This allows subscribers to verify the authenticity and integrity of the event payload.

If no secret is defined, the signature header will be omitted.

List events

List subscribed events

AUTHORIZATIONS: > *jwt*

QUERY PARAMETERS

`eventType`

string
Example: `eventType=agent.created`
Event type to filter webhook events

Responses

> **200** List client's webhooks

> **401** Unauthorized

> **403** Forbidden

> **429** Rate Limited

> **500** Internal Server Error

> **502** Bad Gateway

GET /v1/webhooks

Response samples

200 | 401 | 403 | 429 | 500 | 502

Content type
application/json

Copy Expand all Collapse all


```
{
  – "data": [
      + { … }
    ]
}
```


Subscribe to event

Creates a new event subscription

AUTHORIZATIONS: > *jwt*

REQUEST BODY SCHEMA: application/json

eventId (required)
string
Enum: `"user.created"` `"user.updated"` `"user.deleted"` `"user.activated"` `"user.deactivated"` `"lead.created"` `"lead.updated"` `"lead.assigned"` `"publicProfile.verification.approved"` `"publicProfile.verification.rejected"` `"listing.published"` `"listing.unpublished"`
The event type that will trigger the webhook. This can be one of the predefined events.

callbackUrl (required)
string <uri>

secret
string <= 32 characters
A secret key used to sign the webhook payload(HMAC). This is optional and can be used for additional security.

Responses

> 201 Created Webhook

> 401 Unauthorized

> 403 Forbidden

> 429 Rate Limited

> 500 Internal Server Error

> 502 Bad Gateway

POST /v1/webhooks

Request samples

Payload

Content type
application/json

Copy

```
{
    "eventId": "user.created",
    "callbackUrl": "https://example.com/webhook",
    "secret": "pF3Z9km2L7xQhR8dBnAeTfYu"
}
```

Response samples

201 | 401 | 403 | 429 | 500 | 502

Content type
application/json

Copy

```
{
    "eventId": "users.created",
    "url": "https://example.com",
    "createdAt": "2021-01-01T00:00:00Z"
```

}

Delete event subscription

AUTHORIZATIONS: > *jwt*

PATH PARAMETERS

`eventId` required	string Example: `agent.created` Event ID

Responses

— 204 Event subscription deleted successfully

> 401 Unauthorized

> 403 Forbidden

> 429 Rate Limited

> 500 Internal Server Error

> 502 Bad Gateway

DELETE /v1/webhooks/{eventId}

Response samples

401 | 403 | 429 | 500 | 502

Content type

application/problem+json

Copy

```
{
    "detail": "unauthorized",
    "title": "Unauthorized",
    "type": "AUTHENTICATION"
}
```


Events

user.created Webhook

Triggered when a new user is created

AUTHORIZATIONS: > *jwt*

REQUEST BODY SCHEMA: application/json
required

`id` required	string <uuid>
`type` required	string
`timestamp` required	string <date-time>
`entity` > required	object
`payload` > required	object

Request samples

Payload

Content type
application/json

Copy Expand all Collapse all


```
{
    "id": "554f3eaf-814a-4068-80b8-7beaaedb7194",
    "type": "string",
    "timestamp": "2019-08-24T14:15:22Z",
  – "entity": {
        "id": "string",
        "type": "user"
    },
  – "payload": {
        "email": "example@propertyfinder.ae"
    }
}
```


user.updated Webhook

Triggered when a user is updated

AUTHORIZATIONS: > *jwt*

REQUEST BODY SCHEMA: application/json
required

`id` required	string <uuid>
`type` required	string
`timestamp` required	string <date-time>
`entity` > required	object
`payload` > required	object Additional user update details

Request samples

Payload

Content type
application/json

Copy Expand all Collapse all


```
{
    "id": "554f3eaf-814a-4068-80b8-7beaaedb7194",
    "type": "string",
    "timestamp": "2019-08-24T14:15:22Z",
  − "entity": {
        "id": "string",
        "type": "user"
    },
  − "payload": {
        "email": "user@example.com",
      + "changes": [ ... ]
    }
}
```


user.deleted Webhook

Triggered when a user is deleted

AUTHORIZATIONS: > *jwt*

REQUEST BODY SCHEMA: application/json
required

`id` required	string <uuid>
`type` required	string
`timestamp` required	string <date-time>
`entity` > required	object
`payload` > required	object

Request samples

Payload

Content type
application/json

Copy Expand all Collapse all


```
{
    "id": "554f3eaf-814a-4068-80b8-7beaaedb7194",
    "type": "string",
    "timestamp": "2019-08-24T14:15:22Z",
  – "entity": {
        "id": "string",
        "type": "user"
    },
  – "payload": {
        "email": "example@propertyfinder.ae"
    }
}
```


user.activated Webhook

Triggered when a user is activated

AUTHORIZATIONS: > *jwt*

REQUEST BODY SCHEMA: application/json
required

`id` required	string <uuid>
`type` required	string
`timestamp` required	string <date-time>
`entity` > required	object
`payload` > required	object

Request samples

Payload

Content type
application/json

Copy Expand all Collapse all


```
{
    "id": "554f3eaf-814a-4068-80b8-7beaaedb7194",
    "type": "string",
    "timestamp": "2019-08-24T14:15:22Z",
  – "entity": {
        "id": "string",
        "type": "user"
    },
  – "payload": {
        "email": "example@propertyfinder.ae"
    }
}
```


user.deactivated Webhook

Triggered when a user is deactivated

AUTHORIZATIONS: > *jwt*

REQUEST BODY SCHEMA: application/json
required

`id` required	string <uuid>
`type` required	string
`timestamp` required	string <date-time>
`entity` > required	object

payload > required	object

Request samples

Payload

Content type
application/json

Copy Expand all Collapse all


```
{
    "id": "554f3eaf-814a-4068-80b8-7beaaedb7194",
    "type": "string",
    "timestamp": "2019-08-24T14:15:22Z",
  – "entity": {
        "id": "string",
        "type": "user"
    },
  – "payload": {
        "email": "example@propertyfinder.ae"
    }
}
```


lead.created Webhook

Triggered when a new lead is created

AUTHORIZATIONS: > *jwt*

REQUEST BODY SCHEMA: application/json
required

id required	string
type required	string
timestamp required	string <date-time>

`entity` > required	object
`payload` > required	object

Request samples

Payload

Content type
application/json

Copy Expand all Collapse all


```
{
    "id": "lead-created-12345678",
    "type": "string",
    "timestamp": "2019-08-24T14:15:22Z",
  – "entity": {
        "id": "string",
        "type": "lead"
    },
  – "payload": {
        "channel": "whatsapp",
        "status": "sent",
        "entityType": "listing",
      + "publicProfile": { ... },
      + "listing": { ... },
      + "project": { ... },
      + "developer": { ... },
        "responseLink": "https://example.com",
      + "sender": { ... }
    }
}
```


lead.updated Webhook

Triggered when a new lead is updated

AUTHORIZATIONS: > *jwt*

REQUEST BODY SCHEMA: application/json

required

id required	string
type required	string
timestamp required	string <date-time>
entity › required	object
payload › required	object

Request samples

Payload

Content type

application/json

Copy Expand all Collapse all

```
{
    "id": "lead-updated-12345678",
    "type": "string",
    "timestamp": "2019-08-24T14:15:22Z",
  − "entity": {
        "id": "string",
        "type": "lead"
    },
  − "payload": {
        "channel": "whatsapp",
        "status": "sent",
        "entityType": "listing",
      + "publicProfile": { ... },
      + "listing": { ... },
      + "project": { ... },
      + "developer": { ... },
        "responseLink": "https://example.com",
      + "sender": { ... }
    }
}
```


lead.assigned Webhook

Triggered when a new lead is assigned

AUTHORIZATIONS: > *jwt*

REQUEST BODY SCHEMA: application/json
required

`id` required	string
`type` required	string
`timestamp` required	string <date-time>
`entity` > required	object
`payload` > required	object

Request samples

Payload

Content type
application/json

Copy Expand all Collapse all


```
{
    "id": "lead-assigned-12345678",
    "type": "string",
    "timestamp": "2019-08-24T14:15:22Z",
  − "entity": {
        "id": "string",
        "type": "lead"
    },
  − "payload": {
        "channel": "whatsapp",
        "status": "sent",
        "entityType": "listing",
      + "publicProfile": { ... },
      + "listing": { ... },
      + "project": { ... },
      + "developer": { ... },
        "responseLink": "https://example.com",
      + "sender": { ... }
    }
}
```


publicProfile.verification.approved Webhook

Triggered when a public profile verification state is changed to 'approved'

AUTHORIZATIONS: > *jwt*

REQUEST BODY SCHEMA: application/json
required

`id` required	string <uuid>

type required	string
timestamp required	string <date-time>
entity > required	object
payload > required	object

Request samples

Payload

Content type
application/json

Copy Expand all Collapse all


```
{
    "id": "554f3eaf-814a-4068-80b8-7beaaedb7194",
    "type": "string",
    "timestamp": "2019-08-24T14:15:22Z",
  – "entity": {
        "id": "string",
        "type": "publicProfile"
    },
  – "payload": {
        "email": "example@propertyfinder.ae"
    }
}
```


publicProfile.verification.rejected Webhook

Triggered when a public profile verification state is changed to 'rejected'

AUTHORIZATIONS: > *jwt*

REQUEST BODY SCHEMA: application/json
required

id required	string <uuid>
type required	string
timestamp required	string <date-time>
entity > required	object
payload > required	object

Request samples

Payload

Content type
application/json

Copy Expand all Collapse all


```
{
    "id": "554f3eaf-814a-4068-80b8-7beaaedb7194",
    "type": "string",
    "timestamp": "2019-08-24T14:15:22Z",
  – "entity": {
        "id": "string",
        "type": "publicProfile"
    },
  – "payload": {
        "email": "example@propertyfinder.ae",
        "reason": "Invalid BRN"
    }
}
```


listing.published Webhook

Triggered when a listing is published

AUTHORIZATIONS: > *jwt*

REQUEST BODY SCHEMA: application/json
required

id required	string <uuid>
type required	string
timestamp required	string <date-time>
entity › required	object
payload required	object Empty payload for events that do not require additional data

Request samples

Payload

Content type
application/json

Copy Expand all Collapse all


```
{
    "id": "8d245d222fdce687434ac1afad4effa4",
    "type": "string",
    "timestamp": "2019-08-24T14:15:22Z",
  – "entity": {
        "id": "01K0YB4HEKM08V901DVJ5ATVYF",
        "type": "listing"
    },
    "payload": { }
}
```


listing.unpublished Webhook

Triggered when a listing is unpublished

AUTHORIZATIONS: › *jwt*

REQUEST BODY SCHEMA: application/json
required

id required	string <uuid>
type required	string
timestamp required	string <date-time>
entity > required	object
payload required	object Empty payload for events that do not require additional data

Request samples

Payload

Content type
application/json

Copy Expand all Collapse all


```
{
    "id": "8d245d222fdce687434ac1afad4effa4",
    "type": "string",
    "timestamp": "2019-08-24T14:15:22Z",
  – "entity": {
        "id": "01K0YB4HEKM08V901DVJ5ATVYF",
        "type": "listing"
    },
    "payload": { }
}
```