



DEVELOPER DOCUMENTATION • RELEASE 4.0.0

WeSign API

Integration & Reference Guide

The complete REST reference for WeSign v3 – templates, document collections, recipients, mass distribution, self signing, reporting and administration.

- | | | | |
|----|--|----|---|
| 01 | Introduction • what the API does | 08 | Field metadata • fields defined in XML |
| 02 | Environments • base URL and hosts | 09 | Enumerations • every coded value |
| 03 | Authentication • tokens and OTP | 10 | Endpoint reference • 118 operations |
| 04 | Conventions • formats and errors | 11 | Data models • every object |
| 05 | Core concepts • the object model | 12 | Troubleshooting • and result codes |
| 06 | Quick start • zero to signed PDF | 13 | Endpoint index • A-Z with pages |
| 07 | Callbacks • and redirects | 14 | Support • and resources |

API VERSION	PRODUCT RELEASE	ENVIRONMENT	DOCUMENT
v3	WeSign 4.0.0	Production	4.0.0 / rev. A

BASE URL

<https://wse.comsigntrust.com/api>

INTERACTIVE REFERENCE – SWAGGER UI

<https://wse.comsigntrust.com/api/swagger/index.html>

Contents

1	Introduction	2
2	Environments and base URL	4
3	Authentication	6
4	API conventions	8
5	Core concepts	11
6	Quick start: from zero to a signed PDF	15
7	Callbacks and redirects	19
8	Defining fields inside the document	21
9	Enumerations	24
10	Endpoint reference	28
10.1	Service configuration	29
10.2	Authentication & the current user	30
10.3	Company administration	41
10.4	Contacts & contact groups	47
10.5	Templates	57
10.6	Document collections	67
10.7	Signers	87
10.8	Distribution (mass sending)	90
10.9	Self sign & certificate signing	96
10.10	Signing links & video sessions	104
10.11	Reports	107
10.12	Dashboard	110
10.13	Licence & quota	111
11	Data models	112
12	Troubleshooting	149
13	Endpoint index	152
14	Support and resources	156

SECTION 1

Introduction

What the WeSign API does, who it is for, and how this document is organised.

WeSign is the electronic signature platform of ComSign / Comda. It takes a document, puts it in front of the right people, collects legally binding signatures and returns a sealed PDF together with a full audit trail.

The **WeSign API** exposes that same engine over HTTPS as a JSON REST interface. Everything an operator can do in the WeSign web application can be driven from your own system: a CRM that sends its contracts automatically, a back office that mass-distributes a form to thousands of customers, a portal that lets a customer sign inside your own screens.

What you can build

Send for signature

Upload a document, place signature and data fields on it, and send it to one or many recipients by e-mail, SMS, WhatsApp or an in-branch tablet.

[POST /v3/DocumentCollections](#)

Embed signing in your product

Receive a per-recipient signing link and open it inside your own application, then get a callback the moment the document is signed.

[signerLinks\[\]](#) • [callbackUrl](#)

Mass distribution

Push one template to a whole recipient list from an Excel file, each copy pre-filled with that recipient's own data.

[POST /v3/Distribution](#)

Self signing

Sign a document in your own name with a graphic signature, a smart card, a server certificate or an eIDAS qualified signature.

[POST /v3/SelfSign](#)

Track & retrieve

Poll the status of every envelope, download the signed PDF, the audit trail, the collected field values as XML, CSV or JSON.

[GET /v3/DocumentCollections/{id}](#)

Administer

Create users and groups, manage the address book, read usage reports and the remaining quota of the subscription.

[/v3/Admins](#) • [/v3/Reports](#)

Who this document is for

This guide is written for developers and integration engineers. It assumes you are comfortable with HTTP, JSON and bearer-token authentication. No WeSign-specific SDK is required – any HTTP client will do.

How to read this document

- **Sections 2–4** are the ground rules: where the service lives, how you authenticate, and the conventions every endpoint follows.
- **Section 5** explains the object model. Read it once and the endpoint list stops looking like a list.
- **Section 6** walks a complete integration end to end, from an empty account to a downloaded signed PDF.

- **Sections 9–12** are the reference: every endpoint, every enumeration and every data model, generated directly from the live OpenAPI definition of release 4.0.0.

Always-current source

The reference chapters of this document are generated from the OpenAPI 3.0 definition published by the service itself.

The interactive version of the same definition — where you can authenticate and fire real requests — is at

<https://wse.comsigntrust.com/api/swagger/index.html>.

SECTION 2

Environments and base URL

Which host to call, how the path is built, and how to move an integration from test to production.

PRODUCTION – BASE URL

`https://wse.comsigntrust.com/api`

Every path in this document is relative to this address. A call to `/v3/Templates` is therefore `https://wse.comsigntrust.com/api/v3/Templates`.

Address map

RESOURCE	ADDRESS
API base URL	<code>https://wse.comsigntrust.com/api</code>
Interactive reference (Swagger UI)	<code>https://wse.comsigntrust.com/api/swagger/index.html</code>
OpenAPI 3.0 definition	<code>https://wse.comsigntrust.com/api/swagger/v3/swagger.json</code>
Web application	<code>https://wse.comsigntrust.com</code>
Test & integration environment	<code>https://devtest.comda.co.il/api</code> • Swagger UI

How a request URL is built

`https://wse.comsigntrust.com/api` `/v3` `/DocumentCollections/{id}/audit/{offset}`

environment host + /api

API version

resource path

keep this one value in configuration always v3 in release 4.0.0 exactly as printed in this document

Test first, then production

Build and certify your integration against the test environment. It runs the same code as production and the same API version, but it is not bound by production quotas and its data can be reset. When the integration is ready, the only thing that changes is the host name – keep the base URL in configuration, never hard-coded.

Accounts are not shared between environments

Credentials, templates, contacts and documents exist separately in each environment. Ask your ComSign contact for a dedicated test account; do not test with production credentials.

On-premise installations

WeSign can also be deployed inside your own data centre. The API is identical; only the host changes. In that case the base URL is `https://<your-wesign-host>/api` and the Swagger UI is served from `https://<your-wesign-host>/api/swagger/index.html`.

Transport requirements

REQUIREMENT	DETAIL
Protocol	HTTPS. A request to <code>http://</code> is answered with a <code>301</code> redirect to the same address on HTTPS – call HTTPS directly rather than relying on the redirect.
TLS	TLS 1.2 or newer.
Character set	UTF-8 throughout, request and response. Hebrew, Arabic and Latin content are all supported.
Response media type	JSON endpoints answer with <code>application/json; charset=utf-8</code> . Download endpoints answer with the media type of the file they return.

SECTION 3

Authentication

WeSign uses JWT bearer tokens. This section covers the sign-in flows, the OTP second factor, token lifetime and refresh.

Every protected endpoint expects a JSON Web Token in the standard `Authorization` header:

```
Authorization: Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9...
```

A small number of endpoints are deliberately open and need no token – they are marked **NO TOKEN** in the reference. Everything else is marked **BEARER TOKEN**, and endpoints reserved for company administrators carry **COMPANYADMIN** as well.

The standard sign-in flow

1 Sign in with the account credentials

```
POST https://wse.comsigntrust.com/api/v3/Users/login
Content-Type: application/json

{
  "email": "integration@yourcompany.com",
  "password": "S3cure!Passw0rd"
}
```

The response is a `UserTokensResponseDTO`:

```
{
  "token": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9...", // access token - send as Bearer
  "refreshToken": "8f2c1e...", // used to obtain a new access token
  "authToken": null // carried back on refresh; pass it through unchanged
}
```

2 If the company enforces a one-time password, complete the second factor

When the company is configured to force OTP, the login response carries an OTP token instead of a usable access token, and a code is sent to the user. Exchange them:

```
POST https://wse.comsigntrust.com/api/v3/Users/validate0tpflow

{ "otpToken": "<token from the login response>", "code": "481920" }
```

The code can be re-sent with `POST /v3/Users/resend0tp`.

3 Call the API

Attach the access token to every subsequent request. Nothing else – no API key, no query parameter, no cookie – is required.

4 Refresh before the token expires

POST `https://wse.comsigntrust.com/api/v3/Users/refresh`

```
{ "jwtToken": "<current access token>",
  "refreshToken": "<refresh token>",
  "authToken": null }
```

The response contains a fresh `token` . Refreshing does not invalidate work in progress.

Other ways in

SITUATION	ENDPOINT	NOTES
Active Directory / SAML single sign-on	<code>POST /v3/Users/externalLogin</code>	Post the token issued by the identity provider; WeSign returns its own token pair.
Password has expired	<code>POST /v3/Users/validateExpiredPasswordFlow</code>	Sign-in is refused until a new password is set through this endpoint.
Forgotten password	<code>POST /v3/Users/password</code>	Sends a reset mail. The user completes the reset with <code>PUT /v3/Users/password</code> .
Working inside another group	<code>POST /v3/Users/SwitchGroup/{groupId}</code>	Returns a new token pair scoped to that group. All later calls act inside it.
Ending the session	<code>GET /v3/Users/Logout</code>	Invalidates the session and returns the SSO logout URL when one is configured.

Permission levels

USER TYPE	VALUE	CAN DO
Basic	1	Send documents, sign, manage own contacts and documents.
Editor	2	Everything a Basic user can do, plus creating and editing templates.
CompanyAdmin	3	Everything above, plus the <code>/v3/Admins</code> endpoints: users, groups and company-wide reporting.

Recommended practice for server-to-server integrations

- Create a **dedicated integration user**; do not reuse a person's account.
- Cache the access token and reuse it until it is close to expiry, rather than signing in on every request.
- Keep credentials in a secret store, never in source control, and never send them from a browser.
- Handle `401` as "token no longer valid" – refresh once, then sign in again. It arrives with an empty body.

Never expose a token to a signer

A signing link returned in `signerLinks[]` already carries its own short-lived, single-purpose credential. Send the signer that link – never your API token.

SECTION 4

API conventions

The rules that apply to every endpoint: content types, identifiers, dates, paging, file encoding and the error format.

Requests

ASPECT	RULE
Method semantics	GET reads, POST creates, PUT updates, DELETE removes. A few batch operations use PUT because they carry a body of identifiers – for example PUT /v3/DocumentCollections/deletebatch .
Content type	Send Content-Type: application/json . The service also accepts application/json-patch+json and text/json ; they behave identically.
Accept	application/json for data, or omit it. Download endpoints return the binary payload with the matching content type (application/pdf , application/zip , text/csv).
Identifiers	Every object is identified by a GUID (UUID v4), for example 3f2504e0-4f89-11d3-9a0c-0305e82c3301 .
Dates	ISO 8601 in UTC on the wire – 2026-09-08T09:30:00Z . Date <i>values</i> typed into document fields use the display format dd/MM/yyyy and times use HH:mm .
Booleans	JSON true / false . Query-string booleans are spelled the same way.
Enumerations	Always sent and returned as integers . The names are listed in section 8.
Unknown fields	Ignored on input. Do not fail on new fields appearing in a response – the API is extended additively.

Paging and sorting

The list endpoints share the same four query parameters:

PARAMETER	TYPE	MEANING
offset	integer	Index of the first record to return.
limit	integer	How many records to return.
sortBy	string	Field to sort on, e.g. name , creationTime , status .
sortOrder	string	asc or desc .

Watch the starting index

The base index is not the same everywhere, and each endpoint states its own rule in the reference. Document collections, distributions and template pages count from **1**; usage reports and the "pages range" endpoints count from **0**. When in doubt, request the first two pages and compare.

A list response is an object with a single named array inside it – documentCollections , templates , contacts , users and so on – and no total count:

```
{ "documentCollections": [ { ... }, { ... } ] }
```

Page until a request returns fewer items than the `limit` you asked for; that page is the last one.

Free-text search

Most list endpoints accept a `key` parameter. It is a "contains" match, case insensitive, applied to the natural text of the object — a contact's name, e-mail or phone, a template's name, a document collection's name or its recipients.

Where a `searchParameter` is also offered, it narrows the match to one column.

Sending files

Files are never multipart. They travel inside the JSON body as a **base64 data URI**, including the MIME prefix:

```
"base64File": "data:application/pdf;base64,JVBERi0xLjcKJc..."
```

FILE	PREFIX TO USE
PDF	<code>data:application/pdf;base64,</code>
Word (.docx)	<code>data:application/vnd.openxmlformats-officedocument.wordprocessingml.document;base64,</code>
Excel (.xlsx)	<code>data:application/vnd.openxmlformats-officedocument.spreadsheetml.sheet;base64,</code>
Excel (.xls)	<code>data:application/vnd.ms-excel;base64,</code>
Image (PNG / JPEG)	<code>data:image/png;base64,</code> • <code>data:image/jpeg;base64,</code>

The prefix is mandatory

Sending the bare base64 payload without the `data:<mime>;base64,` prefix is the single most common cause of a rejected upload. The service uses the prefix to determine the file type — there is no separate type field.

Receiving files

Download endpoints stream the file directly. The shape of the result depends on how much you asked for: a single document comes back as a **PDF**, several documents come back as a **ZIP** whose entries are named after the documents or the signers. Where you would rather stay in JSON, the parallel `/json` endpoints return the same content base64 encoded inside a [DownloadDTO](#).

Errors

Failures use conventional HTTP status codes. When there is a body, it is a [GeneralError](#):

```
{
  "errors": { "error": [ "Invalid credential" ] },
  "title": "Invalid operation error occurred",
  "status": 2,
  "traceId": ""
}
```

FIELD	WHAT IT HOLDS
<code>errors</code>	A map of field name to the messages for that field. General failures use the key <code>error</code> . This is the part worth logging and showing to an operator.
<code>title</code>	A short category for the failure.
<code>status</code>	An internal service code , not the HTTP status. Do not branch on it – branch on the HTTP status code of the response.
<code>traceId</code>	The correlation id of the request in the service logs. It can be empty; when it is present, quote it in a support request.

Status codes you will meet

STATUS	MEANING AND WHAT TO DO
200	Success. Endpoints that have nothing to return answer 200 with an empty body.
400	The request was rejected: a missing or malformed field, a value out of range, an object that does not belong to your company, or a quota that has run out. Read <code>errors</code> – it names the problem. Do not retry unchanged.
401	No token, an expired token or an invalid one. The body is empty. Refresh the token once; if the refresh is refused too, sign in again.
404	No such route. Almost always a typo in the path or a missing <code>/v3</code> segment.
405	Right path, wrong HTTP method – for example <code>GET</code> on an endpoint that is defined as <code>POST</code> .
500	The service failed while handling a valid request. Retry once with backoff; if it persists, send the request and the time of the call to support.

Branch on the HTTP status, log the body

Treat the HTTP status code as the decision, and `errors` plus `traceId` as the diagnosis. The individual endpoint pages in section 9 list 200, 400 and 500 because those are the responses the service declares; 401, 404 and 405 are produced by the web layer in front of it and apply to every endpoint.

Idempotency and retries

Creating a document collection is **not** idempotent: repeating the call sends the document again and consumes quota again. If a create call times out, do not blindly retry – list the recent collections (`GET /v3/DocumentCollections` with a narrow `from / to` window and your `key`) and check whether the first attempt succeeded. Read operations and downloads are safe to retry freely.

Rate of calls

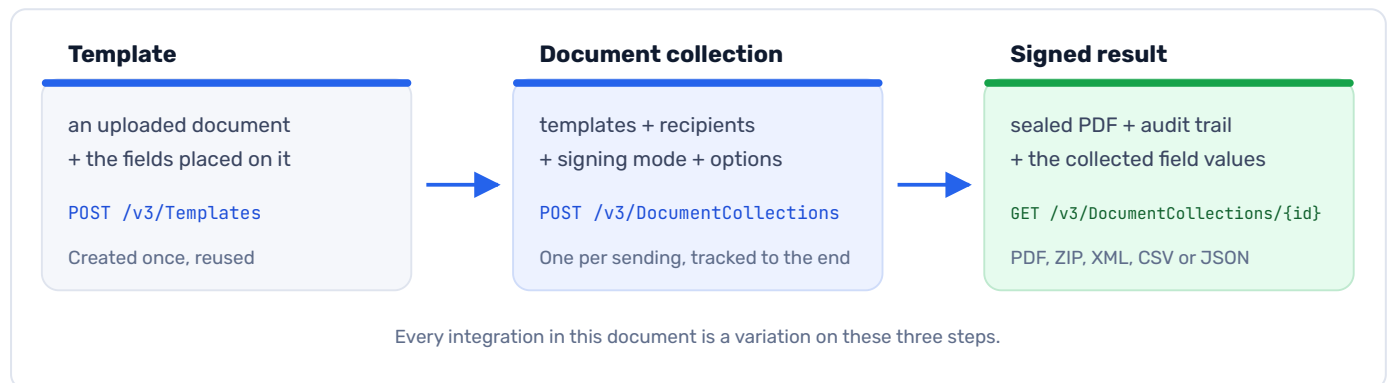
There is no published per-second rate limit, but the subscription does cap the number of documents, SMS messages, WhatsApp messages and identifications per month. Poll status endpoints at a sensible interval – or better, use a callback URL and stop polling altogether (section 7). Query the remaining quota at any time with `GET /v3/License?readUsage=true`.

SECTION 5

Core concepts

The object model behind the endpoints: templates and fields, document collections, recipients, signing modes and identification.

The object model at a glance



Templates

A **template** is an uploaded document together with the layout of the fields placed on it. It is created once and reused for every sending. A template that is only meant for a single sending can be created with `isOneTimeUseTemplate: true` so it does not clutter the template library.

Creating a template is a two-step operation in practice:

- `POST /v3/Templates` uploads the file and returns the template id.
- `PUT /v3/Templates/{id}` writes the field layout onto it.

Alternatively, if the source PDF already carries the field definitions as XML metadata, pass that XML (base64 encoded) in `metaData` on creation and the fields are imported automatically. The supported metadata field types are `Graphic_Signature`, `SmartCard_Signature`, `Server_Signature`, `Text`, `Email`, `Phone`, `Number`, `Date`, `Time`, `Checkbox`, `ChoiceGroup` and `RadioGroup`.

Field geometry

Field positions are **normalised to the page**, not measured in points, so a layout survives a change of page size. `x` and `y` are the fraction of the page width and height of the field's origin; `width` and `height` are the fraction of the page the field occupies. All four are strictly between 0 and 1, and the service enforces sensible bounds:

FIELD	WIDTH (FRACTION OF PAGE)	HEIGHT (FRACTION OF PAGE)
Text, choice and signature fields	> 0.08 and < 0.51	> 0.01 and < 0.09
Radio buttons and check boxes	> 0.01 and < 0.65	> 0.007 and < 0.045

`page` is 1-based. `recipientRole` ties the field to one recipient of the envelope – that is how you build a contract where the customer signs one box and the manager signs another.

Field types

Text fields
Free text, date, number, phone, e-mail, time, or a custom pattern validated by `customerRegex`.
`TextFieldType` 1-7

Signature fields
Graphic (drawn or typed), smart card, or server certificate.
`SignatureFieldType` 1-3

Choice fields
Drop-down lists, radio groups and check boxes, each with its own options and selected value.
`choiceFields` · `radioGroupFields` · `checkboxFields`

Any field can be marked `mandatory` – the recipient cannot finish until it is filled – and a text field can be pre-filled and hidden (`isHidden`) when it only exists to carry data back to your system.

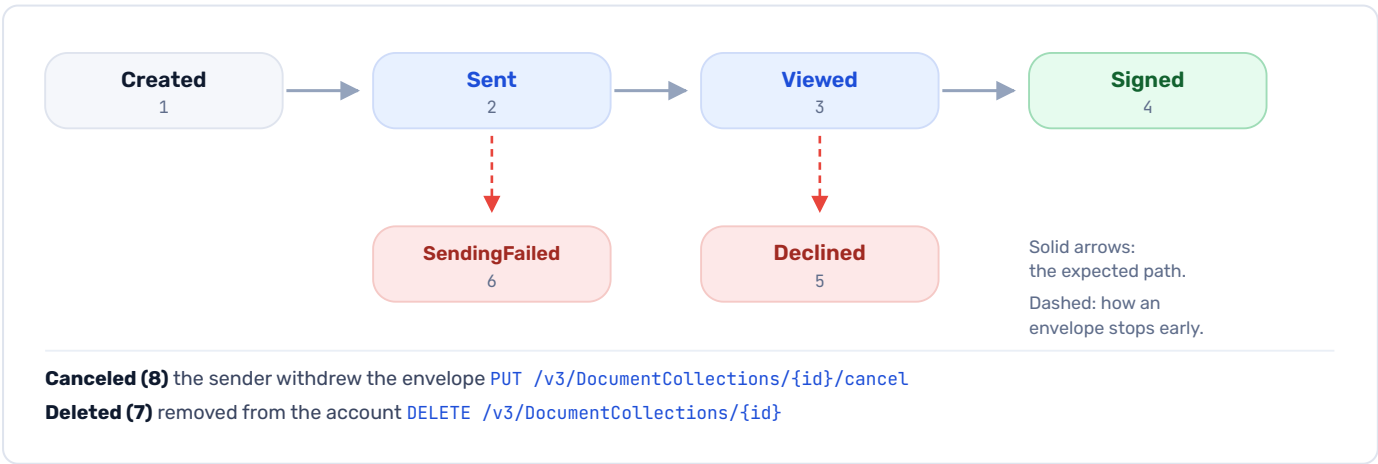
Document collections

A **document collection** is the envelope. It joins one or more templates to one or more recipients, in a chosen signing mode, and it is the object whose status you follow from *Created* to *Signed*. Everything a recipient sees – the documents, the attachments, the notes, the identification requirements – is defined on it.

Signing modes

MODE	VALUE	BEHAVIOUR
OrderedGroupSign	1	Sequential. Recipient 2 is invited only after recipient 1 has signed. The create call returns the link of the <i>first</i> recipient only.
GroupSign	2	Parallel. Everyone is invited at once and can sign in any order.
Online	3	Live session. The sender and the recipient are in the document at the same time – at a counter, on a tablet, or in a video call. The sender's own link comes from <code>GET /v3/DocumentCollections/{id}/senderLink/{signerId}</code> .
SelfSign	100	The caller signs their own document. Created through the <code>/v3/SelfSign</code> endpoints.

Status lifecycle



Each recipient additionally carries their own `SignerStatus` (*Sent*, *Viewed*, *Signed*, *Rejected*) with the exact timestamp of each transition, so a partially signed envelope tells you precisely who is holding it up.

Recipients

A recipient – a [SignerDTO](#) – can be addressed in two ways, and you must supply one of them:

From the address book

Pass `contactId` together with `sendingMethod`. The address and the name come from the stored contact.

Ad hoc

Pass `contactMeans` (the e-mail address or phone number) together with `contactName`, and the matching `sendingMethod`.

Useful per-recipient options:

FIELD	PURPOSE
<code>signerFields</code>	Pre-fill named fields for this recipient before the document opens.
<code>signerAttachments</code>	Ask the recipient to upload files (an ID copy, a form). Each entry can be mandatory.
<code>senderAppendices</code>	Read-only documents attached for this recipient to review but not sign.
<code>senderNote</code>	A note shown inside the signing session after the link is opened.
<code>emailNote</code>	A note rendered in the invitation e-mail itself, up to 250 characters. Reaches only recipients whose channel is e-mail.
<code>linkExpirationInHours</code>	How long the personal signing link stays valid.
<code>phoneExtension</code>	International dialling prefix for the phone number, for example <code>972</code> for Israel.

Delivery channels

[SendingMethod](#) decides how the invitation reaches the recipient: **SMS (1)**, **Email (2)**, **Tablet (3)** for in-branch signing on a registered device, and **WhatsApp (4)** for mass distribution. E-mail requires a valid address; SMS and WhatsApp require a valid phone number.

Identifying the recipient

Two independent controls harden a signature, and they can be combined:

CONTROL	VALUES	EFFECT
<code>otpMode</code>	None 0, CodeRequired 1, PasswordRequired 2, CodeAndPasswordRequired 3	A one-time code sent to the recipient, a shared password, or both, before the document opens. When a code or password is required, supply <code>otpIdentification</code> on the recipient.
<code>authenticationMode</code>	None 0, IDP 1, VisualIDP 2	Identity verification through an identification provider – document based, or visual (selfie / video) identification – before signing is allowed.

Pair the mode with the identification value

Setting `otpMode` to anything other than `0` without also supplying `otpIdentification` is rejected. Visual identifications are also metered separately from documents – check `remainingVisualIdentifications` on the licence.

Distribution

Distribution is mass sending: one template, one call, an unlimited recipient list, and a private copy for every recipient pre-filled with their own values. Feed it an Excel file – column 1 first name, column 2 last name, column 3

phone or e-mail, and any further column a field name — or build the recipient array yourself. A distribution is then tracked, resent and downloaded as one unit.

Self signing

When the caller is the signer, use `/v3/SelfSign`. The document is created, edited and then closed — `DocumentOperation.Close (3)` is what actually signs it. Beyond the graphic signature, WeSign can seal the file with a **server certificate** (Signer1), a **smart card** through the desktop client, or an **eIDAS qualified signature** after an identity check.

Contacts

The address book keeps a name, an e-mail, a phone number, a default channel, a free-text search tag and any stored signature images (seals). Contacts can be grouped, imported in bulk from an XLSX file, and searched by any part of their details. Addressing a recipient by `contactId` keeps your payloads small and your address data in one place.

Licence and quota

A subscription meters documents per month, SMS and WhatsApp messages, visual identifications, video conferences, templates and users, and it switches whole features on or off. Read it with `GET /v3/License?readUsage=true`, or from `GET /v3/Users` which returns the same counters on the user's `program` object. Checking the remaining quota before a large distribution is cheaper than discovering it half way through.

SECTION 6

Quick start: from zero to a signed PDF

A complete integration in six calls, with copy-ready requests.

The walk-through below sends a one-page agreement to a single recipient by e-mail, with a one-time code, and retrieves the signed file. Replace the host, the credentials and the ids as you go. `$TOKEN` is the access token from step 1.

1 Sign in

```
curl -X POST "https://wse.comsigntrust.com/api/v3/Users/login" \
-H "Content-Type: application/json" \
-d '{"email":"integration@yourcompany.com","password":"S3cure!Passw0rd"}'
```

```
{ "token": "eyJhbGciOi...", "refreshToken": "8f2c1e...", "authToken": null }
```

When the company enforces a one-time password, the login call returns an OTP token rather than a usable access token – exchange it as described in section 3.

Keep `token` for every call below. If your company enforces OTP, complete `POST /v3/Users/validate0tpflow` first (section 3).

2 Upload the document as a template

```
curl -X POST "https://wse.comsigntrust.com/api/v3/Templates" \
-H "Authorization: Bearer $TOKEN" -H "Content-Type: application/json" \
-d '{
  "name": "Service agreement 2026",
  "base64File": "data:application/pdf;base64,JVBERi0xLjcKJc...",
  "isOneTimeUseTemplate": false
}'
```

```
{ "templateId": "3f2504e0-4f89-11d3-9a0c-0305e82c3301",
  "templateName": "Service agreement 2026",
  "pagesCount": 1 }
```

Remember the `data:application/pdf;base64,` prefix – without it the upload is rejected.

3 Place the fields on the template

Coordinates are fractions of the page (section 5). Here we put one text field near the top and one graphic signature field at the bottom of page 1.

```
curl -X PUT "https://wse.comsigntrust.com/api/v3/Templates/3f2504e0-4f89-11d3-9a0c-0305e82c3301" \
-H "Authorization: Bearer $TOKEN" -H "Content-Type: application/json" \
-d '{
  "name": "Service agreement 2026",
  "fields": {
    "textFields": [{
      "fieldName": "CustomerName", "textFieldType": 1, "page": 1,
      "x": 0.12, "y": 0.22, "width": 0.35, "height": 0.03,
      "mandatory": true, "recipientRole": "Customer"
    }],
    "signatureFields": [{
      "fieldName": "CustomerSignature", "signingType": 1, "page": 1,
      "x": 0.12, "y": 0.78, "width": 0.30, "height": 0.05,
      "mandatory": true, "recipientRole": "Customer"
    }]
  }
}'
```

If the source PDF already carried its field definitions as XML metadata, this step is not needed – pass the XML in `metaData` at step 2 instead. To read back what is on the page, call `GET /v3/Templates/{id}/pages/{page}`.

4 Send it for signature

```
curl -X POST "https://wse.comsigntrust.com/api/v3/DocumentCollections" \
-H "Authorization: Bearer $TOKEN" -H "Content-Type: application/json" \
-d '{
  "documentName": "Service agreement - ACME Ltd.",
  "documentMode": 2,
  "templates": ["3f2504e0-4f89-11d3-9a0c-0305e82c3301"],
  "senderNote": "Please review and sign by Friday.",
  "signers": [{
    "contactName": "Dana Levi",
    "contactMeans": "dana@acme.example.com",
    "sendingMethod": 2,
    "otpMode": 1,
    "otpIdentification": "0501234567",
    "linkExpirationInHours": 168,
    "emailNote": "Your service agreement is ready.",
    "signerFields": [{
      "templateId": "3f2504e0-4f89-11d3-9a0c-0305e82c3301",
      "fieldName": "CustomerName", "fieldValue": "ACME Ltd."
    }]
  }],
  "notifications": { "shouldSend": true, "shouldSendSignedDocument": true },
  "callbackUrl": "https://your-app.example.com/hooks/wesign",
  "redirectUrl": "https://your-app.example.com/thanks?doc=[docId]"
}'
```

```
{
  "documentCollectionId": "9b1deb4d-3b7d-4bad-9bdd-2b0d7b3dcb6d",
  "signerLinks": [
    { "signerId": "c56a4180-65aa-42ec-a945-5fd21dec0538",
      "link": "https://.../sign/8f21c0e4b7..." }
  ]
}
```

Store `documentCollectionId` against your own record. The link is the recipient's personal signing URL – WeSign has already sent it by e-mail, but you can also present it inside your own application.

5 Follow the progress

```
curl "https://wse.comsigntrust.com/api/v3/DocumentCollections/info/9b1deb4d-3b7d-4bad-9bdd-2b0d7b3dcb6d" \
-H "Authorization: Bearer $TOKEN"
```

```
{ "documentCollectionId": "9b1deb4d-...", "name": "Service agreement - ACME Ltd.",
  "documentStatus": 4, "mode": 2,
  "signers": [ { "name": "Dana Levi", "status": 3, "timeSigned": "2026-09-08T11:04:12Z" } ] }
```

`documentStatus: 4` is *Signed*. Better still, let WeSign tell you: with `callbackUrl` set, your endpoint is called as the status changes and no polling is needed (section 7).

6

Download the result

```
# the sealed PDF (a ZIP when the envelope holds several documents)
curl -OJ "https://wse.comsigntrust.com/api/v3/DocumentCollections/9b1deb4d-3b7d-4bad-9bdd-2b0d7b3dcb6d" \
  -H "Authorization: Bearer $TOKEN"

# the audit trail, timestamps shifted to Israel time (UTC-3 offset argument)
curl -OJ "https://wse.comsigntrust.com/api/v3/DocumentCollections/9b1deb4d-.../audit/-3" \
  -H "Authorization: Bearer $TOKEN"

# the values the recipient typed, as JSON
curl "https://wse.comsigntrust.com/api/v3/DocumentCollections/9b1deb4d-.../fields/json" \
  -H "Authorization: Bearer $TOKEN"
```

That is the whole loop. Everything else in this document is a variation on it: more recipients, an ordered flow, a mass distribution, stronger identification, or a different way to sign.

Where to go next

Send to many at once

One template, a whole recipient list, private copies.

[Section 9.8](#) • [Distribution](#)

Sign in your own name

Graphic, smart card, server certificate or eIDAS.

[Section 9.9](#) • [SelfSign](#)

Publish a public link

One template, one URL, anyone who opens it gets their own copy.

[Section 9.10](#) • [Links](#)

SECTION 7

Callbacks and redirects

How to be told that something happened instead of asking for it.

Two optional URLs on a document collection remove almost all polling from an integration. Both are set when the collection is created and both are plain absolute HTTPS URLs.

callBackUrl – server to server

WeSign calls this URL as the document collection progresses, so your system learns about a signature without asking. Point it at an endpoint that is reachable from the internet, answers quickly, and treats the notification as a trigger rather than as the truth:

1. WeSign calls your `callBackUrl`
2. Your handler returns `200` immediately and queues the work
3. Your worker calls `GET /v3/DocumentCollections/info/{id}` to read the authoritative status
4. If the status is `Signed(4)`, download the PDF

Why re-read the status

Verifying through the API keeps the integration correct even if a notification is delayed, duplicated, or arrives out of order – and it means an unauthenticated call to your endpoint can never make your system believe a document was signed.

redirectUrl – where the recipient lands

After the recipient finishes signing in the browser, WeSign redirects them to this address. It is how you bring a customer back into your own flow instead of leaving them on a WeSign confirmation page.

Note the spelling

The field is spelled `redirectUrl` in the API – without the second *c*. It is kept as it is for backward compatibility. `redirectTo` is silently ignored.

The placeholder `[docId]` anywhere in the URL is replaced with the id of the document collection before the redirect, which lets you match the returning browser to your own record:

```
"redirectUrl": "https://your-app.example.com/contracts/done?doc=[docId]"
```

the recipient is sent to

```
https://your-app.example.com/contracts/done?doc=9b1deb4d-3b7d-4bad-9bdd-2b0d7b3dcb6d
```

The same placeholder works on `POST /v3/DocumentCollections/simple`.

Built-in notifications

Independently of your own hooks, WeSign sends its own messages. Control them per envelope with the `notifications` object, and per user through the account configuration:

SETTING	EFFECT
<code>notifications.shouldSend</code>	Send the invitation to the recipients at all. Set it to <code>false</code> when you deliver the signing link yourself.
<code>notifications.shouldSendSignedDocument</code>	Mail the finished, sealed PDF to the participants once everyone has signed.
<code>shouldNotifyWhileSignerSigned</code> / <code>...Viewed</code>	Notify the sender as each recipient views or signs. Part of the user configuration.
<code>shouldNotifySignReminder</code> and <code>signReminderFrequencyInDays</code>	Automatic reminders to recipients who have not signed yet. The minimum interval is one day.

If you cannot expose a callback endpoint

Poll, but poll politely. Ask for the list rather than for each envelope, and narrow it by date and status:

```
GET https://wse.comsigntrust.com/api/v3/DocumentCollections?from=2026-09-08&to=2026-09-09&signed=true&limit=100&offset=1
```

A few minutes between polls is plenty for a signing flow that involves a human.

SECTION 8

Defining fields inside the document

Ship a document that already carries its own field layout, and let WeSign import it in a single call.

Section 6 placed the fields on a template by calling `PUT /v3/Templates/{id}` with explicit coordinates. There is a second route, and it suits teams who own the source document: mark the field positions **inside the document itself** as text placeholders, describe them in a small XML file, and pass that XML – base64 encoded – in `metaData` when the template is created. WeSign finds the placeholders, turns them into real fields, and the template is ready in one call.

When to prefer this route

The document is authored in Word or InDesign and changes often – a designer moves a paragraph and the fields have to move with it. Placeholders travel with the text, so the layout never has to be measured again.

Step 1 – put placeholders in the document

Type a placeholder where each field belongs, wrapped in the delimiters you declare (braces by convention), and pad it with spaces to the width the field should have:

```
{Signer1_sig      }      {Signer2_sig      }
{Signer1_text     }      {Signer2_text     }
{Signer1_email    }      {Signer2_email    }
{Signer1_date     }      {Signer2_date     }
{Signer1_phone    }      {Signer2_phone    }
{Signer1_number   }      {Signer2_number   }
{Signer1_list     }      {Signer2_list     }
{Signer1_checkbox}      {Signer2_checkbox}
{radio_8} {radio_9}      {radio_11} {radio_12}
```

Two details decide how the result looks:

- The **width of the placeholder** – the braces plus the padding spaces – becomes the width of the field. Pad it to the size you want.
- The **colour of the placeholder text** is what a reader would otherwise see. Set it to white so it is invisible in the source document, and declare that colour in the XML.

Naming the placeholders after the recipient – `Signer1_`, `Signer2_` – is a convention rather than a requirement, but it keeps a two-party contract readable.

Step 2 – describe the fields in XML

The XML says what each placeholder becomes. The example below covers every supported type for one recipient; a second recipient is the same block with the other set of names.

```
<?xml version="1.0"?>
<PDFMetaData>
  <PlaceholderColor>black</PlaceholderColor>
  <Parenthesis>{}</Parenthesis>
  <SkipValidation>true</SkipValidation>

  <Fields>
    <Field Fieldname="Signer1_sig"      Type="Graphic_Signature" />
    <Field Fieldname="Signer1_text"     Type="Text"      Description="Signer1_text" />
    <Field Fieldname="Signer1_email"    Type="Email"     Description="Signer1_email" />
    <Field Fieldname="Signer1_phone"    Type="Phone"     IsMandatory="false"
      Description="Signer1_phone" />
    <Field Fieldname="Signer1_number"   Type="Number"    IsMandatory="false"
      Description="Signer1_number" />
    <Field Fieldname="Signer1_date"     Type="Date"      IsMandatory="false"
      Description="Signer1_date" />
    <Field Fieldname="Signer1_checkbox" Type="Checkbox" IsChecked="false" />

    <GroupField Fieldname="Signer1_list" Type="ChoiceGroup" IsMandatory="false"
      Description="Signer1_list">
      <Field Type="Choice" IsSelected="true" Option="1" />
      <Field Type="Choice" Option="2" />
      <Field Type="Choice" Option="3" />
    </GroupField>

    <GroupField Type="RadioGroup" GroupName="Signer1_grp1">
      <Field Fieldname="radio_8" Type="RadioField" Value="44" IsSelected="true" />
      <Field Fieldname="radio_9" Type="RadioField" Value="55" />
    </GroupField>
  </Fields>
</PDFMetaData>
```

Document settings

ELEMENT	MEANING
PlaceholderColor	The colour the placeholders are written in, so WeSign knows what to look for and what to erase.
Parenthesis	The delimiters that wrap every placeholder. {} by convention.
SkipValidation	When true, the import does not insist that the document and the XML agree on the number of fields.

Field types

TYPE	ATTRIBUTES	BECOMES
Graphic_Signature	Fieldname, IsMandatory	A drawn or typed signature field.
SmartCard_Signature	Fieldname, IsMandatory	A smart-card signature field.
Server_Signature	Fieldname, IsMandatory	A server-certificate signature field.
Text	Fieldname, IsMandatory, Description	A free-text field.
Email • Phone • Number • Date • Time	as Text	A text field carrying the matching validation.
Checkbox	Fieldname, IsMandatory, IsChecked	A single check box.

TYPE	ATTRIBUTES	BECOMES
ChoiceGroup	Fieldname , IsMandatory , Description , plus nested Field Type="Choice" entries with Option and IsSelected	A drop-down list.
RadioGroup	GroupName , plus nested Field Type="RadioField" entries with Fieldname , Value and IsSelected	A set of mutually exclusive radio buttons.

Step 3 – create the template with the XML attached

POST <https://wse.comsigntrust.com/api/v3/Templates>

```
{
  "name": "Service agreement 2026",
  "base64File": "data:application/pdf;base64,JVBERi0xLjcKJc...",
  "metaData": "PD94bWwgdmVyc2lvbj0iMS4wIj8+DQo8UERGTWV0YURhdGE+...",
  "isOneTimeUseTemplate": false
}
```

`metaData` is the XML file itself, UTF-8 encoded and then base64 encoded. The response is the usual template id, and the fields are already in place – read them back with `GET /v3/Templates/{id}/pages/{page}` to confirm.

If the import is rejected

Two causes account for almost every failure: the document and the XML disagree on how many fields there are (result code 66 – set `SkipValidation` or fix the mismatch), and the XML itself cannot be parsed (result code 65). Both arrive as a 400.

SECTION 9

Enumerations

Every enumerated value the API accepts or returns. All of them travel as integers.

WeSign sends and receives enumerations as **numbers**, never as names. The tables below give the meaning of each number. Where the service reserves values for internal states, that is noted explicitly.

Primary enumerations

DocumentMode

Signing flow of a document collection.

VALUE	NAME
1	OrderedGroupSign
2	GroupSign
3	Online
100	SelfSign

Value 4 is reserved for documents created through the Distribution mechanism.

DocumentStatus

Lifecycle state of a document collection (or of a single document inside it).

VALUE	NAME
1	Created
2	Sent
3	Viewed
4	Signed
5	Declined
6	SendingFailed
7	Deleted
8	Canceled

Values 0 and 9 are reserved for internal service states and are not returned by the public endpoints.

SignerStatus

Per-recipient progress inside a document collection.

VALUE	NAME
1	Sent
2	Viewed
3	Signed
4	Rejected

SendingMethod

Channel used to deliver the signing invitation to a recipient.

VALUE	NAME
1	SMS
2	Email
3	Tablet
4	WhatsApp

OtpMode

Second factor a recipient must pass before the document opens.

VALUE	NAME
0	None
1	CodeRequired
2	PasswordRequired
3	CodeAndPasswordRequired

AuthMode

Identity verification performed on the recipient before signing.

VALUE	NAME
0	None
1	IDP
2	VisualIDP

Value 3 is reserved for a combined identification mode.

UserType

Permission level of a WeSign account.

VALUE	NAME
1	Basic
2	Editor
3	CompanyAdmin

Values 4-7 are reserved for internal and management roles.

TextFieldType

Validation applied to a data-entry field on a template.

VALUE	NAME
1	Text
2	Date
3	Number
4	Phone
5	Email
6	Custom
7	Time

Values 8-9 are reserved for future field types.

SignatureFieldType

Kind of signature produced in a signature field.

VALUE	NAME
1	Graphic
2	SmartCard
3	Server

DocumentOperation

Action applied when updating a self-sign document.

VALUE	NAME
1	Save
2	Decline
3	Close

Language

Interface / notification language.

VALUE	NAME
1	en
2	he

Additional enumerations

These types appear in configuration, reporting and internal objects. The service publishes the accepted numeric values; the interactive reference at <https://wse.comsigntrust.com/api/swagger/index.html> always shows the current set.

TYPE	MEANING	VALUES
CompanyStatus	Company state.	0, 1
CreationSource	Interface the object was created from.	0, 1, 2
DateFormat	Date presentation format used for the account and for date fields.	1, 2, 3, 4
GroupStatus	Group state.	1, 2
ManagementReportFrequency	Schedule of an internal management report.	0, 1, 2
ManagementReportType	Internal management report identifier.	0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15
MessageType	Internal notification template identifier.	0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20
ProgramResetType	How the subscription quota is refreshed.	0, 1, 2, 3
ProviderType	Third-party provider (SMS gateway, identification provider, and similar).	1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 20
ReportFrequency	Schedule of an automatically generated report.	0, 1, 2, 3
ReportType	Type of scheduled report.	0
SearchParameter	Which column a free-text search key is matched against.	0, 1, 2
SignatureFieldKind	Distinguishes a regular signature field from an initials field.	0, 1
SigningFileType	File type produced or consumed by a signing operation.	0, 1, 2, 3, 4
SingleLinkAdditionalResourceType	Extra resource attached to a template single link.	1
UserStatus	Account state.	1, 2, 3, 4

SECTION 10

Endpoint reference

All 118 operations of WeSign API v3, grouped by area and generated from the live OpenAPI definition.

AREA	ENDPOINTS	PURPOSE
Service configuration	2	Public settings the client needs before a user signs in, plus the list of signing tablets registered for the company.
Authentication & the current user	22	Sign-in, one-time-password verification, token refresh and everything that belongs to the account of the caller.
Company administration	9	Company administrators manage the users and groups of their own company. Every endpoint in this group requires a CompanyAdmin token.
Contacts & contact groups	15	The address book used to address signing requests, together with contact groups and the signature images stored for a recipient.
Templates	13	A template is an uploaded document plus the field layout placed on it. Templates are the building blocks of every signing process.
Document collections	28	A document collection is the envelope that is actually sent for signature: one or more templates, one or more recipients, a signing mode and a status you can track to completion.
Signers	1	Recipient level operations on an existing document collection.
Distribution (mass sending)	8	Send the same template to hundreds of recipients in one call, each one receiving a private copy pre-filled with their own data.
Self sign & certificate signing	8	Sign a document yourself - with a graphic signature, a smart card, a server certificate (Signer1) or an eIDAS qualified signature.
Signing links & video sessions	4	Retrieve the links the caller still has to sign, publish a public single link for a template, and open a video-identification signing session.
Reports	6	Usage reporting for the company, plus scheduled reports delivered automatically.
Dashboard	1	Aggregated counters for the home screen.
Licence & quota	1	The limits of the subscription and how much of it is used.

How to read an endpoint card

The header carries the HTTP method, the path (relative to `https://wse.comsigntrust.com/api`) and badges:

BEARER TOKEN means a token is required, **NO TOKEN** means the endpoint is open, **COMPANYADMIN** means it is restricted to company administrators, and **FILE DOWNLOAD** means the response is a binary file rather than JSON. Path and query parameters, the request body and the possible responses follow.

SECTION 10.1 • CONFIGURATION

Service configuration

Public settings the client needs before a user signs in, plus the list of signing tablets registered for the company.

2 ENDPOINTS IN THIS SECTION

GET

 /v3/Configuration

GET

 /v3/Configuration/tablets

GET

 /v3/Configuration

NO TOKEN

Read init configuration

RESPONSES

STATUS	BODY	MEANING
200	InitConfigurationDTO	OK
400	GeneralError	Bad Request
500	GeneralError	Internal Server Error

GET

 /v3/Configuration/tablets

BEARER TOKEN

Read tablets list

QUERY PARAMETERS

NAME	TYPE	DESCRIPTION
key	string	optional —

RESPONSES

STATUS	BODY	MEANING
200	TabletsConfigurationDTO	OK
400	GeneralError	Bad Request
500	GeneralError	Internal Server Error

SECTION 10.2 • USERS

Authentication & the current user

Sign-in, one-time-password verification, token refresh and everything that belongs to the account of the caller.

22 ENDPOINTS IN THIS SECTION

POST /v3/Users **PUT** /v3/Users **GET** /v3/Users **GET** /v3/Users/groups
POST /v3/Users/SwitchGroup/{groupid} **POST** /v3/Users/resendOtp **POST** /v3/Users/validateOtpflow
POST /v3/Users/validateExpiredPasswordFlow **POST** /v3/Users/login **GET** /v3/Users/Logout
PUT /v3/Users/activation **POST** /v3/Users/activation **POST** /v3/Users/externalLogin **POST** /v3/Users/password
PUT /v3/Users/password **POST** /v3/Users/refresh **POST** /v3/Users/change **POST** /v3/Users/unsubscribeuser
POST /v3/Users/changepaymentrule **POST** /v3/Users/UpdatePhone **POST** /v3/Users/UpdatePhoneValidateOtp
DELETE /v3/Users/DeletePhone

POST /v3/Users

NO TOKEN

User Sign up

UserLanguage: en = 1, he = 2

REQUEST BODY CreateUserDT0

FIELD	TYPE	DESCRIPTION
name	string	—
language	<u>Language</u>	en = 1, he = 2
dateFormat	<u>DateFormat</u>	1, 2, 3, 4
email	string	—
password	string	—
reCAPCHA	string	—
sendActivationLink	boolean	—
username	string	—

RESPONSES

STATUS	BODY	MEANING
200	<u>LinkResponse</u>	OK
400	<u>GeneralError</u>	Bad Request
500	<u>GeneralError</u>	Internal Server Error

PUT

/v3/Users

BEARER TOKEN

Update User

UserLanguage: en = 1, he = 2 UserType: Basic = 1, Editor = 2, CompanyAdmin = 3

REQUEST BODY UpdateUserDT0

FIELD	TYPE	DESCRIPTION
name	string	—
email	string	—
userConfiguration	<u>UserConfigurationDT0</u>	—
└ signatureColor	string	—
└ shouldSendSignedDocument	boolean	—
└ shouldNotifyWhileSignerSigned	boolean	—
└ shouldNotifyWhileSignerViewed	boolean	—
└ shouldDisplayNameInSignature	boolean	—
└ shouldNotifySignReminder	boolean	—
└ signReminderFrequencyInDays	integer	—
└ language	<u>Language</u>	en = 1, he = 2
└ dateFormat	<u>DateFormat</u>	1, 2, 3, 4
username	string	—

RESPONSES

STATUS	BODY	MEANING
200	no body	OK
400	<u>GeneralError</u>	Bad Request
500	<u>GeneralError</u>	Internal Server Error

GET

/v3/Users

BEARER TOKEN

Get your own user details

Language: en = 1, he = 2

UserType: Basic = 1, Editor = 2, CompanyAdmin = 3

RESPONSES

STATUS	BODY	MEANING
200	ExtendedUserResponseDTO	OK
400	GeneralError	Bad Request
500	GeneralError	Internal Server Error

GET

/v3/Users/groups

BEARER TOKEN

Get all groups that the current user belongs to

RESPONSES

STATUS	BODY	MEANING
200	UserGroupsDTO	OK
400	GeneralError	Bad Request
500	GeneralError	Internal Server Error

POST

/v3/Users/SwitchGroup/{groupId}

BEARER TOKEN

Switch user's active group context

After switching groups, new tokens are returned for the selected group context.

PATH PARAMETERS

NAME	TYPE		DESCRIPTION
groupId	string (uuid)	required	The ID of the group to switch to

RESPONSES

STATUS	BODY	MEANING
200	UserTokensResponseDTO	OK
400	GeneralError	Bad Request
500	GeneralError	Internal Server Error

POST /v3/Users/resendOtp**NO TOKEN****Resend OTP (One-Time Password) code to user**

Resends the OTP code to the user's registered contact method.

REQUEST BODY OtpResendDTO

FIELD	TYPE	DESCRIPTION
otpToken	string	—

RESPONSES

STATUS	BODY	MEANING
200	<u>UserTokensResponseDTO</u>	OK
400	<u>GeneralError</u>	Bad Request
500	<u>GeneralError</u>	Internal Server Error

POST /v3/Users/validateOtpFlow**NO TOKEN****Validate OTP code and complete login flow**

Validates the OTP code entered by the user and completes the login process.

REQUEST BODY OtpDTO

FIELD	TYPE	DESCRIPTION
otpToken	string	—
code	string	—

EXAMPLE REQUEST BODY — every optional field is shown; send only the ones you need

```
{
  "otpToken": "b7d1f0a4c2e94...",
  "code": "481920"
}
```

RESPONSES

STATUS	BODY	MEANING
200	<u>UserTokensResponseDTO</u>	OK
400	<u>GeneralError</u>	Bad Request
500	<u>GeneralError</u>	Internal Server Error

POST

/v3/Users/validateExpiredPasswordFlow

NO TOKEN

Update password for expired password flow

Allows user to update their password when it has expired.

REQUEST BODY RenewPasswordDTO

FIELD	TYPE	DESCRIPTION
renewPasswordToken	string	—
newPassword	string	—
oldPassword	string	—

RESPONSES

STATUS	BODY	MEANING
200	no body	OK
400	<u>GeneralError</u>	Bad Request
500	<u>GeneralError</u>	Internal Server Error

POST

/v3/Users/login

NO TOKEN

User login case the company force OTP will return token for OTP

REQUEST BODY LoginRequestDTO

FIELD	TYPE	DESCRIPTION
email	string	—
password	string	—

EXAMPLE REQUEST BODY — every optional field is shown; send only the ones you need

```
{
  "email": "dana@acme.example.com",
  "password": "S3cure!Passw0rd"
}
```

RESPONSES

STATUS	BODY	MEANING
200	<u>UserTokensResponseDTO</u>	OK
400	<u>GeneralError</u>	Bad Request
500	<u>GeneralError</u>	Internal Server Error

GET /v3/Users/Logout

BEARER TOKEN

Logout current user

Invalidates the current session and returns logout URL if configured.

RESPONSES

STATUS	BODY	MEANING
200	LogoutResponseDTO	OK
400	GeneralError	Bad Request
500	GeneralError	Internal Server Error

PUT /v3/Users/activation

BEARER TOKEN

REQUEST BODY [ActivationDTO](#)

FIELD	TYPE	DESCRIPTION
token	string	—

RESPONSES

STATUS	BODY	MEANING
200	UserTokensResponseDTO	OK
400	GeneralError	Bad Request
500	GeneralError	Internal Server Error

POST /v3/Users/activation

NO TOKEN

Resend activation link**REQUEST BODY** [BaseUserDTO](#)

FIELD	TYPE	DESCRIPTION
email	string	—

RESPONSES

STATUS	BODY	MEANING
200	no body	OK
400	GeneralError	Bad Request
500	GeneralError	Internal Server Error

POST

/v3/Users/externalLogin

NO TOKEN

External login for login using AD or SAML login

REQUEST BODY ExternalLoginDTO

FIELD	TYPE	DESCRIPTION
token	string	—

RESPONSES

STATUS	BODY	MEANING
200	<u>UserTokensResponseDTO</u>	OK
400	<u>GeneralError</u>	Bad Request
500	<u>GeneralError</u>	Internal Server Error

POST

/v3/Users/password

NO TOKEN

Reset password, send reset password mail to user

REQUEST BODY BaseUserDTO

FIELD	TYPE	DESCRIPTION
email	string	—

RESPONSES

STATUS	BODY	MEANING
200	no body	OK
400	<u>GeneralError</u>	Bad Request
500	<u>GeneralError</u>	Internal Server Error

PUT

/v3/Users/password

NO TOKEN

Update password

REQUEST BODY RenewPasswordDTO

FIELD	TYPE	DESCRIPTION
renewPasswordToken	string	—
newPassword	string	—
oldPassword	string	—

RESPONSES

STATUS	BODY	MEANING
200	<u>UserTokensResponseDTO</u>	OK
400	<u>GeneralError</u>	Bad Request
500	<u>GeneralError</u>	Internal Server Error

POST

/v3/Users/refresh

NO TOKEN

Refresh token

REQUEST BODY TokensDTO

FIELD	TYPE	DESCRIPTION
jwtToken	string	—
refreshToken	string	—
authToken	string	—

EXAMPLE REQUEST BODY — every optional field is shown; send only the ones you need

```
{
  "jwtToken": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9...",
  "refreshToken": "8f2c1e7d94b64f0a9d3c...",
  "authToken": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9..."
}
```

RESPONSES

STATUS	BODY	MEANING
200	<u>RefreshResponseDTO</u>	OK
400	<u>GeneralError</u>	Bad Request
500	<u>GeneralError</u>	Internal Server Error

POST /v3/Users/change**BEARER TOKEN****Change password****REQUEST BODY** ChangePasswordDTO

FIELD	TYPE	DESCRIPTION
oldPassword	string	—
newPassword	string	—

RESPONSES

STATUS	BODY	MEANING
200	<u>UserTokensResponseDTO</u>	OK
400	<u>GeneralError</u>	Bad Request
500	<u>GeneralError</u>	Internal Server Error

POST /v3/Users/unsubscribeuser**BEARER TOKEN****Unsubscribe current user from notifications****RESPONSES**

STATUS	BODY	MEANING
200	no body	OK
400	<u>GeneralError</u>	Bad Request
500	<u>GeneralError</u>	Internal Server Error

POST /v3/Users/changepaymentrule**BEARER TOKEN****Get URL for changing payment rule**

Returns a URL where the user can change their payment settings.

RESPONSES

STATUS	BODY	MEANING
200	<u>LinkResponse</u>	OK
400	<u>GeneralError</u>	Bad Request
500	<u>GeneralError</u>	Internal Server Error

POST

/v3/Users/UpdatePhone

BEARER TOKEN

Initiate phone number update process

Sends an OTP code to the new phone number for verification.

REQUEST BODY UpdateUserPhoneDTO

FIELD	TYPE	DESCRIPTION
phoneNumber	string	—
code	string	—

RESPONSES

STATUS	BODY	MEANING
200	no body	OK
400	<u>GeneralError</u>	Bad Request
500	<u>GeneralError</u>	Internal Server Error

POST

/v3/Users/UpdatePhoneValidateOtp

BEARER TOKEN

Validate OTP code and complete phone number update

Validates the OTP code sent to the new phone number and updates the user's phone number.

REQUEST BODY UpdateUserPhoneDTO

FIELD	TYPE	DESCRIPTION
phoneNumber	string	—
code	string	—

RESPONSES

STATUS	BODY	MEANING
200	no body	OK
400	<u>GeneralError</u>	Bad Request
500	<u>GeneralError</u>	Internal Server Error

DELETE

/v3/Users/DeletePhone

BEARER TOKEN

Clears the caller's own phone number. No OTP - ownership only needs proving when a number is claimed, not when it is given up.

RESPONSES

STATUS	BODY	MEANING
200	no body	OK
400	GeneralError	Bad Request
500	GeneralError	Internal Server Error

SECTION 10.3 • ADMINS

Company administration

Company administrators manage the users and groups of their own company. Every endpoint in this group requires a **CompanyAdmin** token.

9 ENDPOINTS IN THIS SECTION

POST

 /v3/Admins/groups

GET

 /v3/Admins/groups

PUT

 /v3/Admins/groups/{id}

DELETE

 /v3/Admins/groups/{id}

POST

 /v3/Admins/users

GET

 /v3/Admins/users

PUT

 /v3/Admins/users/{id}

DELETE

 /v3/Admins/users/{id}

PUT

 /v3/Admins/dev/password

POST

 /v3/Admins/groups

BEARER TOKEN

COMPANYADMIN

Create Group

REQUEST BODY AdminCreateGroupDTO

FIELD	TYPE	DESCRIPTION
name	string	—

RESPONSES

STATUS	BODY	MEANING
200	CreateGroupResponseAdminDTO	OK
400	GeneralError	Bad Request
500	GeneralError	Internal Server Error

GET

 /v3/Admins/groups

BEARER TOKEN

COMPANYADMIN

Get all groups in my company

RESPONSES

STATUS	BODY	MEANING
200	AdminAllGroupsResponseDTO	OK
400	GeneralError	Bad Request
500	GeneralError	Internal Server Error

PUT

/v3/Admins/groups/{id}

BEARER TOKEN

COMPANYADMIN

Update group

PATH PARAMETERS

NAME	TYPE		DESCRIPTION
id	string (uuid)	required	—

REQUEST BODY

AdminCreateGroupDT0

FIELD	TYPE		DESCRIPTION
name	string		—

RESPONSES

STATUS	BODY	MEANING
200	no body	OK
400	GeneralError	Bad Request
500	GeneralError	Internal Server Error

DELETE

/v3/Admins/groups/{id}

BEARER TOKEN

COMPANYADMIN

Delete group

PATH PARAMETERS

NAME	TYPE		DESCRIPTION
id	string (uuid)	required	—

RESPONSES

STATUS	BODY	MEANING
200	AdminAllGroupsResponseDT0	OK
400	GeneralError	Bad Request
500	GeneralError	Internal Server Error

POST

/v3/Admins/users

BEARER TOKEN

COMPANYADMIN

Create user

UserType: Basic = 1, Editor = 2, CompanyAdmin = 3

REQUEST BODY AdminCreateUserDT0

FIELD	TYPE	DESCRIPTION
name	string	—
email	string	—
username	string	—
type	UserType	Basic = 1, Editor = 2, CompanyAdmin = 3
groupId	string (uuid)	—
additionalGroupsIds	string (uuid)[]	—
language	Language	en = 1, he = 2

EXAMPLE REQUEST BODY — every optional field is shown; send only the ones you need

```
{
  "name": "Service agreement 2026",
  "email": "dana@acme.example.com",
  "username": "integration",
  "type": 1,
  "groupId": "7e3b9a04-52c6-4d19-8b7e-1f0c5d2a6934",
  "additionalGroupsIds": [
    "3f2504e0-4f89-11d3-9a0c-0305e82c3301"
  ],
  "language": 1
}
```

RESPONSES

STATUS	BODY	MEANING
200	CreateUserResponseAdminDT0	OK
400	GeneralError	Bad Request
500	GeneralError	Internal Server Error

GET

/v3/Admins/users

BEARER TOKEN

COMPANYADMIN

Get all users in my company

UserType: Basic = 1, Editor = 2, CompanyAdmin = 3

QUERY PARAMETERS

NAME	TYPE		DESCRIPTION
key	string	optional	—
offset	integer	optional	—
limit	integer	optional	—
sortBy	string	optional	—
sortOrder	string	optional	—

RESPONSES

STATUS	BODY	MEANING
200	<u>AdminAllUsersResponseDT0</u>	OK
400	<u>GeneralError</u>	Bad Request
500	<u>GeneralError</u>	Internal Server Error

PUT

/v3/Admins/users/{id}

BEARER TOKEN

COMPANYADMIN

Update user

PATH PARAMETERS

NAME	TYPE		DESCRIPTION
id	string (uuid)	required	—

REQUEST BODY

AdminCreateUserDT0

FIELD	TYPE	DESCRIPTION
name	string	—
email	string	—
username	string	—
type	UserType	Basic = 1, Editor = 2, CompanyAdmin = 3
groupId	string (uuid)	—
additionalGroupsIds	string (uuid)[]	—
language	Language	en = 1, he = 2

RESPONSES

STATUS	BODY	MEANING
200	no body	OK
400	GeneralError	Bad Request
500	GeneralError	Internal Server Error

DELETE

/v3/Admins/users/{id}

BEARER TOKEN

COMPANYADMIN

Delete user

PATH PARAMETERS

NAME	TYPE		DESCRIPTION
id	string (uuid)	required	—

RESPONSES

STATUS	BODY	MEANING
200	no body	OK
400	GeneralError	Bad Request
500	GeneralError	Internal Server Error

PUT

/v3/Admins/dev/password

BEARER TOKEN

Update password by dev admin user

REQUEST BODY

RenewPasswordByDevAdminUserDTO

FIELD	TYPE	DESCRIPTION
userId	string (uuid)	—
email	string	—
newPassword	string	—

RESPONSES

STATUS	BODY	MEANING
200	UserTokensResponseDTO	OK
400	GeneralError	Bad Request
500	GeneralError	Internal Server Error

SECTION 10.4 • CONTACTS

Contacts & contact groups

The address book used to address signing requests, together with contact groups and the signature images stored for a recipient.

15 ENDPOINTS IN THIS SECTION

POST

/v3/Contacts

GET

/v3/Contacts

POST

/v3/Contacts/bulk

PUT

/v3/Contacts/deletebatch

GET

/v3/Contacts/{id}

PUT

/v3/Contacts/{id}

DELETE

/v3/Contacts/{id}

GET

/v3/Contacts/signatures/{docCollectionId}

DELETE

/v3/Contacts/signatures/{docCollectionId}

PUT

/v3/Contacts/signatures

GET

/v3/Contacts/Groups

GET

/v3/Contacts/Group/{id}

DELETE

/v3/Contacts/Group/{id}

PUT

/v3/Contacts/Group/{id}

POST

/v3/Contacts/Group

POST

/v3/Contacts

BEARER TOKEN

Create new contact

SendingMethod: SMS = 1, Email = 2, Tablet = 3

REQUEST BODY ContactDTO

FIELD	TYPE	DESCRIPTION
name	string	—
email	string	—
phone	string	—
phoneExtension	string	—
defaultSendingMethod	<u>SendingMethod</u>	SMS = 1, Email = 2, Tablet = 3, WhatsApp = 4
seals	<u>SealDTO[]</u>	—
└ name	string	—
└ base64Image	string	—
searchTag	string	—

EXAMPLE REQUEST BODY — every optional field is shown; send only the ones you need

```
{
  "name": "Service agreement 2026",
  "email": "dana@acme.example.com",
  "phone": "0501234567",
  "phoneExtension": "972",
  "defaultSendingMethod": 2,
  "seals": [
    {
      "name": "Service agreement 2026",
      "base64Image": "data:image/png;base64,iVBORw0KGgoAAAANSUh..."
    }
  ],
  "searchTag": "vip"
}
```

RESPONSES

STATUS	BODY	MEANING
200	<u>CreateContactResponseDTO</u>	OK
400	<u>GeneralError</u>	Bad Request
500	<u>GeneralError</u>	Internal Server Error

GET /v3/Contacts**BEARER TOKEN****Search contacts**

key: key to search if contains in contact details - name\email\phone

QUERY PARAMETERS

NAME	TYPE		DESCRIPTION
key	string	optional	—
offset	integer	optional	—
limit	integer	optional	—
popular	boolean	optional	—
recent	boolean	optional	—
includeTabletMode	boolean	optional	—
sortBy	string	optional	—
sortOrder	string	optional	—

RESPONSES

STATUS	BODY	MEANING
200	AllContactResponseDTO	OK
400	GeneralError	Bad Request
500	GeneralError	Internal Server Error

POST /v3/Contacts/bulk**BEARER TOKEN****Create bulk of contacts from XLSX file****REQUEST BODY** [ContactsDTO](#)

FIELD	TYPE	DESCRIPTION
base64File	string	—

RESPONSES

STATUS	BODY	MEANING
200	CreateContactsResponseDTO	OK
400	GeneralError	Bad Request
500	GeneralError	Internal Server Error

PUT

/v3/Contacts/deletebatch

BEARER TOKEN

Delete a batch of contacts by their IDs

REQUEST BODY BatchRequestDTO

FIELD	TYPE	DESCRIPTION
ids	string[]	—

RESPONSES

STATUS	BODY	MEANING
200	string[]	OK
400	<u>GeneralError</u>	Bad Request
500	<u>GeneralError</u>	Internal Server Error

GET

/v3/Contacts/{id}

BEARER TOKEN

Get contact by ID

PATH PARAMETERS

NAME	TYPE	DESCRIPTION
id	string (uuid)	required —

RESPONSES

STATUS	BODY	MEANING
200	<u>ContactResponseDTO</u>	OK
400	<u>GeneralError</u>	Bad Request
500	<u>GeneralError</u>	Internal Server Error

PUT

/v3/Contacts/{id}

BEARER TOKEN

Update contact

PATH PARAMETERS

NAME	TYPE		DESCRIPTION
id	string (uuid)	required	—

REQUEST BODY

ContactDTO

FIELD	TYPE	DESCRIPTION
name	string	—
email	string	—
phone	string	—
phoneExtension	string	—
defaultSendingMethod	SendingMethod	SMS = 1, Email = 2, Tablet = 3, WhatsApp = 4
seals	SealDTO[]	—
└ name	string	—
└ base64Image	string	—
searchTag	string	—

RESPONSES

STATUS	BODY	MEANING
200	no body	OK
400	GeneralError	Bad Request
500	GeneralError	Internal Server Error

DELETE

/v3/Contacts/{id}

BEARER TOKEN

Delete contact

PATH PARAMETERS

NAME	TYPE		DESCRIPTION
id	string (uuid)	required	—

RESPONSES

STATUS	BODY	MEANING
200	no body	OK
400	GeneralError	Bad Request
500	GeneralError	Internal Server Error

GET /v3/Contacts/signatures/{docCollectionId}**BEARER TOKEN****Get saved signature images for a specific document collection****PATH PARAMETERS**

NAME	TYPE		DESCRIPTION
docCollectionId	string (uuid)	required	The document collection ID

RESPONSES

STATUS	BODY	MEANING
200	SignaturesImagesDTO	OK
400	GeneralError	Bad Request
500	GeneralError	Internal Server Error

DELETE /v3/Contacts/signatures/{docCollectionId}**BEARER TOKEN****Delete a saved signature image for a specific document collection****PATH PARAMETERS**

NAME	TYPE		DESCRIPTION
docCollectionId	string (uuid)	required	The document collection ID

REQUEST BODY [DeleteSignatureImageDTO](#)

FIELD	TYPE	DESCRIPTION
signatureImage	string	—

RESPONSES

STATUS	BODY	MEANING
200	no body	OK
400	GeneralError	Bad Request
500	GeneralError	Internal Server Error

PUT /v3/Contacts/signatures**BEARER TOKEN****Update saved signature images for a specific document collection****REQUEST BODY** SignaturesImagesDT0

FIELD	TYPE	DESCRIPTION
documentCollectionId	string (uuid)	—
signaturesImages	string[]	—

RESPONSES

STATUS	BODY	MEANING
200	no body	OK
400	<u>GeneralError</u>	Bad Request
500	<u>GeneralError</u>	Internal Server Error

GET /v3/Contacts/Groups**BEARER TOKEN****Get all contact groups****QUERY PARAMETERS**

NAME	TYPE		DESCRIPTION
key	string	optional	Search key for group name or member
offset	integer	optional	Offset for pagination
limit	integer	optional	Limit for pagination
sortBy	string	optional	Sort key: name or membersCount (default: name)
sortOrder	string	optional	asc or desc (default: asc)

RESPONSES

STATUS	BODY	MEANING
200	<u>AllContactsGroupsResponseDT0</u>	OK
400	<u>GeneralError</u>	Bad Request
500	<u>GeneralError</u>	Internal Server Error

GET

/v3/Contacts/Group/{id}

BEARER TOKEN

Get a specific contact group by ID

PATH PARAMETERS

NAME	TYPE		DESCRIPTION
id	string (uuid)	required	Contact group ID

RESPONSES

STATUS	BODY	MEANING
200	ContactsGroupResponseDT0	OK
400	GeneralError	Bad Request
500	GeneralError	Internal Server Error

DELETE

/v3/Contacts/Group/{id}

BEARER TOKEN

Delete a contact group by ID

PATH PARAMETERS

NAME	TYPE		DESCRIPTION
id	string (uuid)	required	Contact group ID

RESPONSES

STATUS	BODY	MEANING
200	no body	OK
400	GeneralError	Bad Request
500	GeneralError	Internal Server Error

PUT

/v3/Contacts/Group/{id}

BEARER TOKEN

Update a contact group by ID

PATH PARAMETERS

NAME	TYPE		DESCRIPTION
id	string (uuid)	required	Contact group ID

REQUEST BODY

ContactsGroupDTO

FIELD	TYPE	DESCRIPTION
name	string	—
contactsGroupMembers	ContactGroupMemberDTO[]	—
└─ contactId	string (uuid)	—
└─ order	integer	—

RESPONSES

STATUS	BODY	MEANING
200	no body	OK
400	GeneralError	Bad Request
500	GeneralError	Internal Server Error

POST

/v3/Contacts/Group

BEARER TOKEN

Create a new contact group

REQUEST BODY ContactsGroupDTO

FIELD	TYPE	DESCRIPTION
name	string	—
contactsGroupMembers	<u>ContactGroupMemberDTO[]</u>	—
└─ contactId	string (uuid)	—
└─ order	integer	—

EXAMPLE REQUEST BODY — every optional field is shown; send only the ones you need

```
{
  "name": "Service agreement 2026",
  "contactsGroupMembers": [
    {
      "contactId": "a4f9d2c1-77b3-4e58-9c10-2d6e8f0b3a41",
      "order": 1
    }
  ]
}
```

RESPONSES

STATUS	BODY	MEANING
200	<u>CreateContactGroupResponseDTO</u>	OK
400	<u>GeneralError</u>	Bad Request
500	<u>GeneralError</u>	Internal Server Error

SECTION 10.5 • TEMPLATES

Templates

A template is an uploaded document plus the field layout placed on it. Templates are the building blocks of every signing process.

13 ENDPOINTS IN THIS SECTION

POST

/v3/Templates

GET

/v3/Templates

PUT

/v3/Templates/{id}

DELETE

/v3/Templates/{id}

POST

/v3/Templates/{id}

GET

/v3/Templates/{id}/pages

GET

/v3/Templates/{id}/pages/{page}

GET

/v3/Templates/{id}/pages/range

POST

/v3/Templates/{id}/favorite

DELETE

/v3/Templates/{id}/favorite

GET

/v3/Templates/{id}/download

PUT

/v3/Templates/deletebatch

POST

/v3/Templates/merge

POST

/v3/Templates

BEARER TOKEN

Create new template

XML Base64(optional-value) allow you to embed fields from pdf-meta-data Encoding UTF-8
Type="Graphic_Signature" properties Fieldname , IsMandatory Type="SmartCard_Signature" properties Fieldname , IsMandatory Type="Server_Signature" properties Fieldname , IsMandatory Type="Server_Signature" properties Fieldname , IsMandatory Type="Checkbox" properties Fieldname , IsMandatory Type="Text" properties Fieldname , IsMandatory Type="Email" properties Fieldname , IsMandatory Type="Phone" properties Fieldname , IsMandatory Type="Time" properties Fieldname , IsMandatory Type="Number" properties Fieldname , IsMandatory Type="Date" properties Fieldname , IsMandatory Type="ChoiceGroup" properties Fieldname , IsMandatory Type="ChoiceGroup" Contains Choice field , Option and IsSelected Type="RadioGroup" Contains RadioField field , Option , IsSelected and Value

REQUEST BODY CreateTemplateDTO

FIELD	TYPE	DESCRIPTION
base64File	string	—
name	string	—
metaData	string	—
isOneTimeUseTemplate	boolean	—

EXAMPLE REQUEST BODY — every optional field is shown; send only the ones you need

```
{
  "base64File": "data:application/pdf;base64,JVBERi0xLjcKJc...",
  "name": "Service agreement 2026",
  "metaData": null,
  "isOneTimeUseTemplate": false
}
```

RESPONSES

STATUS	BODY	MEANING
200	<u>TemplateCountResponseDTO</u>	OK
400	<u>GeneralError</u>	Bad Request
500	<u>GeneralError</u>	Internal Server Error

GET

/v3/Templates

BEARER TOKEN

Get all templates by search criteria

QUERY PARAMETERS

NAME	TYPE		DESCRIPTION
key	string	optional	—
from	string	optional	—
to	string	optional	—
offset	integer	optional	—
limit	integer	optional	—
popular	boolean	optional	—
recent	boolean	optional	—
onlyMine	boolean	optional	When true, return only templates created by the calling user.
onlyFavorites	boolean	optional	When true, return only templates the calling user marked as favorite.
sortBy	string	optional	—
sortOrder	string	optional	—

RESPONSES

STATUS	BODY	MEANING
200	AllTemplatesResponseDTO	OK
400	GeneralError	Bad Request
500	GeneralError	Internal Server Error

PUT

/v3/Templates/{id}

BEARER TOKEN

Update template fields

Date format: dd/MM/yyyy , for example 02/08/2019 Time format: HH:MM , for example 15:59 X , Y should be value between 0 to 1 (not include) Width of textField, choiceField and signatureField should be value between 0.08 to 0.51 (not include) Height of textField, choiceField and signatureField should be value between 0.01 to 0.09 (not include) Width of radio and checkBoxField should be value between 0.01 to 0.65 (not include) Height of radio and checkBoxField should be value between 0.007 to 0.045 (not include)

TextFieldType: Text = 1, Date = 2, Number = 3, Phone = 4, Email = 5, Custom = 6, Time = 7

SignaturFieldType: Graphic = 1, SmartCard = 2, Server = 3

PATH PARAMETERS

NAME	TYPE		DESCRIPTION
id	string (uuid)	required	—

REQUEST BODY UpdateTemplateDTO

FIELD	TYPE	DESCRIPTION
name	string	—
fields	<u>PDFFields</u>	—
└ textFields	<u>TextField[]</u>	—
└ ┌ fieldName	string	—
└ ┌ description	string	—
└ ┌ x	number	—
└ ┌ y	number	—
└ ┌ width	number	—
└ ┌ height	number	—
└ ┌ mandatory	boolean	—
└ ┌ page	integer	—
└ ┌ recipientRole	string	—
└ ┌ textFieldType	<u>TextFieldType</u>	Text = 1, Date = 2, Number = 3, Phone = 4, Email = 5, Custom = 6, Time = 7
└ ┌ customerRegex	string	—
└ ┌ value	string	—
└ ┌ isHidden	boolean	—
└ signatureFields	<u>SignatureField[]</u>	—
└ ┌ fieldName	string	—
└ ┌ description	string	—
└ ┌ x	number	—
└ ┌ y	number	—
└ ┌ width	number	—

FIELD	TYPE	DESCRIPTION
↳ height	number	—
↳ mandatory	boolean	—
↳ page	integer	—
↳ recipientRole	string	—
↳ image	string	—
↳ signingType	<u>SignatureFieldType</u>	Graphic = 1, SmartCard = 2, Server = 3
↳ signatureKind	<u>SignatureFieldKind</u>	0, 1
↳ radioGroupFields	<u>RadioGroupField[]</u>	—
↳ fieldName	string	—
↳ radioFields	<u>RadioField[]</u>	—
↳ selectedRadioName	string	—
↳ checkBoxFields	<u>CheckBoxField[]</u>	—
↳ fieldName	string	—
↳ description	string	—
↳ x	number	—
↳ y	number	—
↳ width	number	—
↳ height	number	—
↳ mandatory	boolean	—
↳ page	integer	—
↳ recipientRole	string	—
↳ isChecked	boolean	—
↳ choiceFields	<u>ChoiceField[]</u>	—
↳ fieldName	string	—
↳ description	string	—
↳ x	number	—
↳ y	number	—
↳ width	number	—
↳ height	number	—
↳ mandatory	boolean	—
↳ page	integer	—
↳ recipientRole	string	—
↳ options	string[]	—
↳ selectedOption	string	—

RESPONSES

STATUS	BODY	MEANING
200	no body	OK

STATUS	BODY	MEANING
400	GeneralError	Bad Request
500	GeneralError	Internal Server Error

DELETE

/v3/Templates/{id}

BEARER TOKEN

Delete template

PATH PARAMETERS

NAME	TYPE		DESCRIPTION
id	string (uuid)	required	—

RESPONSES

STATUS	BODY	MEANING
200	no body	OK
400	GeneralError	Bad Request
500	GeneralError	Internal Server Error

POST

/v3/Templates/{id}

BEARER TOKEN

Duplicate existing template

PATH PARAMETERS

NAME	TYPE		DESCRIPTION
id	string (uuid)	required	—

REQUEST BODY

[DuplicateTemplateDTO](#)

FIELD	TYPE	DESCRIPTION
isOneTimeUseTemplate	boolean	—

RESPONSES

STATUS	BODY	MEANING
200	DuplicateTemplateResponseDTO	OK
400	GeneralError	Bad Request
500	GeneralError	Internal Server Error

GET

/v3/Templates/{id}/pages

BEARER TOKEN

Get pages count by template id

PATH PARAMETERS

NAME	TYPE		DESCRIPTION
id	string (uuid)	required	—

RESPONSES

STATUS	BODY	MEANING
200	TemplateCountResponseDT0	OK
400	GeneralError	Bad Request
500	GeneralError	Internal Server Error

GET

/v3/Templates/{id}/pages/{page}

BEARER TOKEN

Get page details by template id, first page is '1'.

page number start from 1 Response

TextFieldType: Text = 1, Date = 2, Number = 3, Phone = 4, Email = 5, Custom = 6, Time = 7

SignaturFieldType: Graphic = 1, SmartCard = 2, Server = 3

PATH PARAMETERS

NAME	TYPE		DESCRIPTION
id	string (uuid)	required	—
page	integer	required	—

RESPONSES

STATUS	BODY	MEANING
200	TemplatePageResponseDT0	OK
400	GeneralError	Bad Request
500	GeneralError	Internal Server Error

GET /v3/Templates/{id}/pages/range**BEARER TOKEN****Get page details by template id, first page is '1'.**

offset number start from 1 limit number start from 0 Response

TextFieldType: Text = 1, Date = 2, Number = 3, Phone = 4, Email = 5, Custom = 6, Time = 7

SignaturFieldType: Graphic = 1, SmartCard = 2, Server = 3

PATH PARAMETERS

NAME	TYPE		DESCRIPTION
id	string (uuid)	required	—

QUERY PARAMETERS

NAME	TYPE		DESCRIPTION
offset	integer	optional	—
limit	integer	optional	—
withImages	boolean	optional	—

RESPONSES

STATUS	BODY	MEANING
200	TemplatePagesRangeResponseDTO	OK
400	GeneralError	Bad Request
500	GeneralError	Internal Server Error

POST /v3/Templates/{id}/favorite**BEARER TOKEN****Mark a template as favorite for the calling user**

Favorites are personal - marking a template does not change it for anyone else. Calling this repeatedly for the same template is a no-op.

PATH PARAMETERS

NAME	TYPE		DESCRIPTION
id	string (uuid)	required	Template id

RESPONSES

STATUS	BODY	MEANING
200	no body	OK
400	GeneralError	Bad Request
500	GeneralError	Internal Server Error

DELETE

/v3/Templates/{id}/favorite

BEARER TOKEN

Remove a template from the calling user's favorites

Favorites are personal - unmarking a template does not change it for anyone else. Calling this for a template that is not a favorite is a no-op.

PATH PARAMETERS

NAME	TYPE		DESCRIPTION
id	string (uuid)	required	Template id

RESPONSES

STATUS	BODY	MEANING
200	no body	OK
400	GeneralError	Bad Request
500	GeneralError	Internal Server Error

GET

/v3/Templates/{id}/download

BEARER TOKEN

FILE DOWNLOAD

Download template

PATH PARAMETERS

NAME	TYPE		DESCRIPTION
id	string (uuid)	required	—

RESPONSES

STATUS	BODY	MEANING
200	binary file	OK
400	GeneralError	Bad Request
500	GeneralError	Internal Server Error

PUT

/v3/Templates/deletebatch

BEARER TOKEN

Delete multiple templates by batch

REQUEST BODY BatchRequestDTO

FIELD	TYPE	DESCRIPTION
ids	string[]	—

RESPONSES

STATUS	BODY	MEANING
200	no body	OK
400	<u>GeneralError</u>	Bad Request
500	<u>GeneralError</u>	Internal Server Error

POST

/v3/Templates/merge

BEARER TOKEN

Merge multiple templates into a single template

Combines multiple templates (by ID or base64 file) into one template. Pages are merged in order.

REQUEST BODY MergeTemplatesDTO

FIELD	TYPE	DESCRIPTION
templates	string[]	—
name	string	—
isOneTimeUseTemplate	boolean	—

EXAMPLE REQUEST BODY — every optional field is shown; send only the ones you need

```
{
  "templates": [
    "string"
  ],
  "name": "Service agreement 2026",
  "isOneTimeUseTemplate": false
}
```

RESPONSES

STATUS	BODY	MEANING
200	<u>TemplateCountResponseDTO</u>	OK
400	<u>GeneralError</u>	Bad Request
500	<u>GeneralError</u>	Internal Server Error

SECTION 10.6 • DOCUMENTCOLLECTIONS

Document collections

A document collection is the envelope that is actually sent for signature: one or more templates, one or more recipients, a signing mode and a status you can track to completion.

28 ENDPOINTS IN THIS SECTION

GET /v3/DocumentCollections **POST** /v3/DocumentCollections **GET** /v3/DocumentCollections/{id}
DELETE /v3/DocumentCollections/{id} **POST** /v3/DocumentCollections/downloadbatch
GET /v3/DocumentCollections/{id}/ExtraInfo/json **GET** /v3/DocumentCollections/{id}/json
GET /v3/DocumentCollections/{id}/signer/{signerId} **GET** /v3/DocumentCollections/info/{id}
GET /v3/DocumentCollections/{id}/senderLink/{signerId} **GET** /v3/DocumentCollections/{id}/audit/{offset}
POST /v3/DocumentCollections/simple **GET** /v3/DocumentCollections/{id}/documents/{documentId}/pages
GET /v3/DocumentCollections/{id}/documents/{documentId}/pages/{page}
GET /v3/DocumentCollections/{id}/documents/{documentId} **PUT** /v3/DocumentCollections/deletebatch
PUT /v3/DocumentCollections/{id}/cancel
GET /v3/DocumentCollections/{id}/signers/{signerId}/method/{sendingMethod}
GET /v3/DocumentCollections/{collectionId}/reactivate
GET /v3/DocumentCollections/{id}/DocumentCollectionLinks **POST** /v3/DocumentCollections/share
GET /v3/DocumentCollections/export **GET** /v3/DocumentCollections/exportDistribution
GET /v3/DocumentCollections/{id}/fields **GET** /v3/DocumentCollections/{id}/fields/json
GET /v3/DocumentCollections/{id}/fields/CsvXml **PUT** /v3/DocumentCollections/{id}/signer/{signerId}/replace
PUT /v3/DocumentCollections/{id}/serversign

GET /v3/DocumentCollections

BEARER TOKEN

Get all document collections by search criteria

Request offset number start from 1 limit number start from 1 Response

SendingMethod: SMS = 1, Email = 2, Tablet = 3

DocumentStatus: Created = 1, Sent = 2, Viewed = 3, Signed = 4, Declined = 5, SendingFailed = 6, Deleted = 7, Canceled = 8

Mode: OrderedGroupSign = 1, GroupSign = 2, Online = 3, SelfSign = 100

SignerStatus: Sent = 1, Viewed = 2, Signed = 3, Rejected = 4

QUERY PARAMETERS

NAME	TYPE		DESCRIPTION
key	string	optional	—
sent	boolean	optional	—
viewed	boolean	optional	—
signed	boolean	optional	—
declined	boolean	optional	—
sendingFailed	boolean	optional	—
canceled	boolean	optional	—
userId	string	optional	—
from	string	optional	—
to	string	optional	—
offset	integer	optional	—
limit	integer	optional	—
searchParameter	<u>SearchParameter</u>	optional	0, 1, 2
sortBy	string	optional	—
sortOrder	string	optional	—

RESPONSES

STATUS	BODY	MEANING
200	<u>AllDocumentCollectionsResponseDTO</u>	OK
400	<u>GeneralError</u>	Bad Request
500	<u>GeneralError</u>	Internal Server Error

POST /v3/DocumentCollections

BEARER TOKEN

Create document collection from template/s

EmailNote: optional per signer note rendered in the invitation email, up to 250 characters If RediretUrl contains [docId] placeholder, this placeholder will replace by original document collection id For each signer you must attach contactId with sendingMethod or contactMeans with contactName Response In case of documentCollection mode is OrderedGroupSign we will return link for first signer only

Mode: OrderedGroupSign = 1, GroupSign = 2, Online = 3

SendingMethod: SMS = 1, Email = 2, Tablet = 3

OtpMode: None = 0, CodeRequired = 1, PasswordRequired = 2, CodeAndPasswordRequired = 3

AuthenticationMode: None = 0, IDP = 1, VisualIDP = 2

REQUEST BODY CreateDocumentCollectionDTO

FIELD	TYPE	DESCRIPTION
documentMode	<u>DocumentMode</u>	OrderedGroupSign = 1, GroupSign = 2, Online = 3, SelfSign = 100
documentName	string	—
templates	string (uuid)[]	—
senderNote	string	—
signers	<u>SignerDTO[]</u>	—
└ contactId	string (uuid)	—
└ sendingMethod	<u>SendingMethod</u>	SMS = 1, Email = 2, Tablet = 3, WhatsApp = 4
└ contactMeans	string	—
└ phoneExtension	string	—
└ contactName	string	—
└ signerAttachments	<u>SignerAttachmentDTO[]</u>	—
└ name	string	—
└ isMandatory	boolean	—
└ signerFields	<u>SignerFieldDTO[]</u>	—
└ templateId	string (uuid)	—
└ fieldName	string	—
└ fieldValue	string	—
└ linkExpirationInHours	integer	—
└ senderNote	string	—
└ emailNote	string	Per-recipient note rendered in a dedicated section of the invitation email, up to 250 characters. Reaches only recipients whose SendingMethod is Email. Distinct from SenderNote, which is shown inside the signing session after the recipient opens the link.

FIELD	TYPE	DESCRIPTION
└ senderAppendices	<u>Appendix[]</u>	—
└ name	string	—
└ base64File	string	—
└ fileExtention	string	—
└ fileContent	string (base64)	—
└ otpIdentification	string	—
└ otpMode	<u>OtpMode</u>	None = 0, CodeRequired = 1, PasswordRequired = 2, CodeAndPasswordRequired = 3
└ authenticationMode	<u>AuthMode</u>	None = 0, IDP = 1, VisualIDP = 2
readOnlyFields	<u>SignerFieldDTO[]</u>	—
└ templateId	string (uuid)	—
└ fieldName	string	—
└ fieldValue	string	—
redirectUrl	string	—
callBackUrl	string	—
senderAppendices	<u>Appendix[]</u>	—
└ name	string	—
└ base64File	string	—
└ fileExtention	string	—
└ fileContent	string (base64)	—
sharedAppendices	<u>SharedAppendixDTO[]</u>	—
└ signerIndexes	integer[]	—
└ appendix	<u>Appendix</u>	—
└ name	string	—
└ base64File	string	—
└ fileExtention	string	—
└ fileContent	string (base64)	—
notifications	<u>DocumentNotificationsDTO</u>	—
└ shouldSend	boolean	—
└ shouldSendSignedDocument	boolean	—
shouldSignUsingSigner1AfterDocumentSigningFlow	boolean	—
shouldEnableMeaningOfSignature	boolean	—
dateFormat	<u>DateFormat</u>	1, 2, 3, 4

EXAMPLE REQUEST BODY — every optional field is shown; send only the ones you need

```

{
  "documentMode": 2,
  "documentName": "Service agreement - ACME Ltd.",
  "templates": [
    "3f2504e0-4f89-11d3-9a0c-0305e82c3301"
  ],
  "senderNote": "Please review and sign by Friday.",
  "signers": [
    {
      "contactId": "a4f9d2c1-77b3-4e58-9c10-2d6e8f0b3a41",
      "sendingMethod": 2,
      "contactMeans": "dana@acme.example.com",
      "phoneExtension": "972",
      "contactName": "Dana Levi",
      "signerAttachments": [
        {
          "name": "Service agreement 2026",
          "isMandatory": true
        }
      ],
      "signerFields": [
        {
          "templateId": "3f2504e0-4f89-11d3-9a0c-0305e82c3301",
          "fieldName": "CustomerName",
          "fieldValue": "ACME Ltd."
        }
      ],
      "linkExpirationInHours": 168,
      "senderNote": "Please review and sign by Friday.",
      "emailNote": "Your service agreement is ready.",
      "senderAppendices": [
        {
          "name": "Service agreement 2026",
          "base64File": "data:application/pdf;base64,JVBERi0xLjcKJc...",
          "fileExtention": "pdf",
          "fileContent": null
        }
      ],
      "otpIdentification": "0501234567",
      "otpMode": 1,
      "authenticationMode": 0
    }
  ],
  "readOnlyFields": [
    {
      "templateId": "3f2504e0-4f89-11d3-9a0c-0305e82c3301",
      "fieldName": "CustomerName",
      "fieldValue": "ACME Ltd."
    }
  ],
  "redirectUrl": "https://your-app.example.com/thanks?doc=[docId]",
  "callbackUrl": "https://your-app.example.com/hooks/wesign",
  "senderAppendices": [
    {
      "name": "Service agreement 2026",
      "base64File": "data:application/pdf;base64,JVBERi0xLjcKJc...",
      "fileExtention": "pdf",
      "fileContent": null
    }
  ],
  "sharedAppendices": [
    {

```

```
    "signerIndexes": [
      0
    ],
    "appendix": {
      "name": "Service agreement 2026",
      "base64File": "data:application/pdf;base64,JVBERi0xLjcKJc...",
      "fileExtention": "pdf",
      "fileContent": null
    }
  },
  "notifications": {
    "shouldSend": true,
    "shouldSendSignedDocument": true
  },
  "shouldSignUsingSigner1AfterDocumentSigningFlow": false,
  "shouldEnableMeaningOfSignature": true,
  "dateFormat": 1
}
```

RESPONSES

STATUS	BODY	MEANING
200	CreateDocumentCollectionResposneDTO	OK
400	GeneralError	Bad Request
500	GeneralError	Internal Server Error

GET

/v3/DocumentCollections/{id}

BEARER TOKEN

FILE DOWNLOAD

Download document collection

If documentCollection contain 1 document, it will download as PDF file. If documentCollection contain more than 1 document, it will download as ZIP file.

PATH PARAMETERS

NAME	TYPE		DESCRIPTION
id	string (uuid)	required	Document Collection ID

RESPONSES

STATUS	BODY	MEANING
200	binary file	OK
400	GeneralError	Bad Request
500	GeneralError	Internal Server Error

DELETE

/v3/DocumentCollections/{id}

BEARER TOKEN

Delete document collection by id

PATH PARAMETERS

NAME	TYPE		DESCRIPTION
id	string (uuid)	required	—

RESPONSES

STATUS	BODY	MEANING
200	no body	OK
400	GeneralError	Bad Request
500	GeneralError	Internal Server Error

POST

/v3/DocumentCollections/downloadbatch

BEARER TOKEN

FILE DOWNLOAD

Download multiple document collections as batch

If batch contains 1 document, it will download as PDF file. If batch contains more than 1 document, it will download as ZIP file.

REQUEST BODY

[DownloadBatchRequestDTO](#)

FIELD	TYPE	DESCRIPTION
ids	string[]	—

RESPONSES

STATUS	BODY	MEANING
200	binary file	OK
400	GeneralError	Bad Request
500	GeneralError	Internal Server Error

GET

/v3/DocumentCollections/{id}/ExtraInfo/json

BEARER TOKEN

Download document collection with extra information as JSON

Returns document files as base64 encoded strings with template IDs and names.

PATH PARAMETERS

NAME	TYPE		DESCRIPTION
id	string (uuid)	required	Document collection ID

RESPONSES

STATUS	BODY	MEANING
200	DownloadDTO	OK
400	GeneralError	Bad Request
500	GeneralError	Internal Server Error

GET

/v3/DocumentCollections/{id}/json

BEARER TOKEN

Download document collection as JSON

Returns document files as base64 encoded strings.

PATH PARAMETERS

NAME	TYPE		DESCRIPTION
id	string (uuid)	required	Document collection ID

RESPONSES

STATUS	BODY	MEANING
200	DownloadDTO	OK
400	GeneralError	Bad Request
500	GeneralError	Internal Server Error

GET

/v3/DocumentCollections/{id}/signer/{signerId}

BEARER TOKEN

FILE DOWNLOAD

Download signer attachments of specified document collection id

PATH PARAMETERS

NAME	TYPE		DESCRIPTION
id	string (uuid)	required	—
signerId	string (uuid)	required	—

RESPONSES

STATUS	BODY	MEANING
200	binary file	OK
400	GeneralError	Bad Request
500	GeneralError	Internal Server Error

GET

/v3/DocumentCollections/info/{id}

BEARER TOKEN

Get info for document collection

PATH PARAMETERS

NAME	TYPE		DESCRIPTION
id	string (uuid)	required	—

RESPONSES

STATUS	BODY	MEANING
200	DocumentCollectionResposneDTO	OK
400	GeneralError	Bad Request
500	GeneralError	Internal Server Error

GET /v3/DocumentCollections/{id}/senderLink/{signerId}

BEARER TOKEN

Get sender signing link for live mode document collection

PATH PARAMETERS

NAME	TYPE		DESCRIPTION
id	string (uuid)	required	—
signerId	string (uuid)	required	—

RESPONSES

STATUS	BODY	MEANING
200	SenderLiveSigningLinkDTO	OK
400	GeneralError	Bad Request
500	GeneralError	Internal Server Error

GET /v3/DocumentCollections/{id}/audit/{offset}

BEARER TOKEN

FILE DOWNLOAD

Download document collection trace as separate file

offset of client in hours from UTC time for example in order to get Israel time you should pass '-3'

PATH PARAMETERS

NAME	TYPE		DESCRIPTION
id	string (uuid)	required	—
offset	integer	required	—

RESPONSES

STATUS	BODY	MEANING
200	binary file	OK
400	GeneralError	Bad Request
500	GeneralError	Internal Server Error

POST

/v3/DocumentCollections/simple

BEARER TOKEN

Document for "Lite" production version

If RediretUrl contains [docId] placeholder, this placeholder will replace by original document collection id

REQUEST BODY CreateSimpleDocumentDTO

FIELD	TYPE	DESCRIPTION
templateId	string (uuid)	—
documentName	string	—
signerName	string	—
signerMeans	string	—
signerExtention	string	—
rediretUrl	string	—
callBackUrl	string	—
signerOTPMean	string	—
shouldSend	boolean	—
shouldSendSignedDocument	boolean	—

EXAMPLE REQUEST BODY — every optional field is shown; send only the ones you need

```
{
  "templateId": "3f2504e0-4f89-11d3-9a0c-0305e82c3301",
  "documentName": "Service agreement - ACME Ltd.",
  "signerName": "Dana Levi",
  "signerMeans": "dana@acme.example.com",
  "signerExtention": "pdf",
  "rediretUrl": "https://your-app.example.com/thanks?doc=[docId]",
  "callBackUrl": "https://your-app.example.com/hooks/wesign",
  "signerOTPMean": "dana@acme.example.com",
  "shouldSend": true,
  "shouldSendSignedDocument": true
}
```

RESPONSES

STATUS	BODY	MEANING
200	<u>CreateDocumentCollectionResposneDTO</u>	OK
400	<u>GeneralError</u>	Bad Request
500	<u>GeneralError</u>	Internal Server Error

GET

/v3/DocumentCollections/{id}/documents/{documentId}/pages

BEARER TOKEN

Get pages count by document id

PATH PARAMETERS

NAME	TYPE		DESCRIPTION
id	string (uuid)	required	—
documentId	string (uuid)	required	—

RESPONSES

STATUS	BODY	MEANING
200	DocumentCountResponseDTO	OK
400	GeneralError	Bad Request
500	GeneralError	Internal Server Error

GET

/v3/DocumentCollections/{id}/documents/{documentId}/pages/{page}

BEARER TOKEN

Get information about specific page in document

Page number start from 1

PATH PARAMETERS

NAME	TYPE		DESCRIPTION
id	string (uuid)	required	—
documentId	string (uuid)	required	—
page	integer	required	—

RESPONSES

STATUS	BODY	MEANING
200	DocumentPageResponseDTO	OK
400	GeneralError	Bad Request
500	GeneralError	Internal Server Error

GET

/v3/DocumentCollections/{id}/documents/{documentId}

BEARER TOKEN

Get information about pages range in document

offset number start from 1 limit number start from 0

PATH PARAMETERS

NAME	TYPE		DESCRIPTION
id	string (uuid)	required	—
documentId	string (uuid)	required	—

QUERY PARAMETERS

NAME	TYPE		DESCRIPTION
offset	integer	optional	—
limit	integer	optional	—
inViewMode	boolean	optional	—

RESPONSES

STATUS	BODY	MEANING
200	DocumentPagesRangeResponseDTO	OK
400	GeneralError	Bad Request
500	GeneralError	Internal Server Error

PUT

/v3/DocumentCollections/deletebatch

BEARER TOKEN

Delete multiple document collections by batch

REQUEST BODY [DeleteBatchRequestDTO](#)

FIELD	TYPE	DESCRIPTION
ids	string[]	—

RESPONSES

STATUS	BODY	MEANING
200	no body	OK
400	GeneralError	Bad Request
500	GeneralError	Internal Server Error

PUT /v3/DocumentCollections/{id}/cancel**BEARER TOKEN****Cancel document collection by id****PATH PARAMETERS**

NAME	TYPE		DESCRIPTION
id	string (uuid)	required	—

RESPONSES

STATUS	BODY	MEANING
200	no body	OK
400	<u>GeneralError</u>	Bad Request
500	<u>GeneralError</u>	Internal Server Error

GET /v3/DocumentCollections/{id}/signers/{signerId}/method/{sendingMethod}**BEARER TOKEN****Resend document collection link to signer**

SendingMethod: SMS = 1, Email = 2, Tablet = 3

PATH PARAMETERS

NAME	TYPE		DESCRIPTION
id	string (uuid)	required	—
signerId	string (uuid)	required	—
sendingMethod	<u>SendingMethod</u>	required	SMS = 1, Email = 2, Tablet = 3, WhatsApp = 4

QUERY PARAMETERS

NAME	TYPE		DESCRIPTION
shouldSend	boolean	optional	—

RESPONSES

STATUS	BODY	MEANING
200	<u>SignerLink</u>	OK
400	<u>GeneralError</u>	Bad Request
500	<u>GeneralError</u>	Internal Server Error

GET

/v3/DocumentCollections/{collectionId}/reactivate

BEARER TOKEN

Reactivate document collection to signer

PATH PARAMETERS

NAME	TYPE		DESCRIPTION
collectionId	string (uuid)	required	—

QUERY PARAMETERS

NAME	TYPE		DESCRIPTION
shouldSend	boolean	optional	—

RESPONSES

STATUS	BODY	MEANING
200	SignerLink[]	OK
400	GeneralError	Bad Request
500	GeneralError	Internal Server Error

GET

/v3/DocumentCollections/{id}/DocumentCollectionLinks

BEARER TOKEN

Get signing links for document

PATH PARAMETERS

NAME	TYPE		DESCRIPTION
id	string (uuid)	required	—

RESPONSES

STATUS	BODY	MEANING
200	SignerLink	OK
400	GeneralError	Bad Request
500	GeneralError	Internal Server Error

POST

/v3/DocumentCollections/share

BEARER TOKEN

Share document collection to contact

REQUEST BODY ShareDTO

FIELD	TYPE	DESCRIPTION
documentCollectionId	string (uuid)	—
signerName	string	—
signerMeans	string	—

EXAMPLE REQUEST BODY — every optional field is shown; send only the ones you need

```
{  "documentCollectionId": "9b1deb4d-3b7d-4bad-9bdd-2b0d7b3dcb6d",  "signerName": "Dana Levi",  "signerMeans": "dana@acme.example.com"}
```

RESPONSES

STATUS	BODY	MEANING
200	no body	OK
400	<u>GeneralError</u>	Bad Request
500	<u>GeneralError</u>	Internal Server Error

GET /v3/DocumentCollections/export

BEARER TOKEN

FILE DOWNLOAD

Export documents collections details to CSV file**QUERY PARAMETERS**

NAME	TYPE		DESCRIPTION
sent	boolean	optional	—
viewed	boolean	optional	—
signed	boolean	optional	—
declined	boolean	optional	—
sendingFailed	boolean	optional	—
canceled	boolean	optional	—
language	Language	optional	en = 1, he = 2

RESPONSES

STATUS	BODY	MEANING
200	binary file	OK
400	GeneralError	Bad Request
500	GeneralError	Internal Server Error

GET /v3/DocumentCollections/exportDistribution

BEARER TOKEN

FILE DOWNLOAD

Export distribution documents to CSV file

Language: en = 1, he = 2

QUERY PARAMETERS

NAME	TYPE		DESCRIPTION
language	Language	optional	Language for export (default: English) en = 1, he = 2

RESPONSES

STATUS	BODY	MEANING
200	binary file	OK
400	GeneralError	Bad Request
500	GeneralError	Internal Server Error

GET

/v3/DocumentCollections/{id}/fields

BEARER TOKEN

FILE DOWNLOAD

Export document collection fields to XML file

PATH PARAMETERS

NAME	TYPE		DESCRIPTION
id	string (uuid)	required	—

RESPONSES

STATUS	BODY	MEANING
200	binary file	OK
400	GeneralError	Bad Request
500	GeneralError	Internal Server Error

GET

/v3/DocumentCollections/{id}/fields/json

BEARER TOKEN

Export document collection fields data as JSON

Returns PDF form fields data including text fields, checkboxes, choice fields, etc.

PATH PARAMETERS

NAME	TYPE		DESCRIPTION
id	string (uuid)	required	Document collection ID

QUERY PARAMETERS

NAME	TYPE		DESCRIPTION
includeSigantures	boolean	optional	Whether to include signature fields in the export

RESPONSES

STATUS	BODY	MEANING
200	PDFFields	OK
400	GeneralError	Bad Request
500	GeneralError	Internal Server Error

GET /v3/DocumentCollections/{id}/fields/CsvXml

BEARER TOKEN

FILE DOWNLOAD

Export document collection fields to XML file and to csv file in a zip file

PATH PARAMETERS

NAME	TYPE		DESCRIPTION
id	string (uuid)	required	—

RESPONSES

STATUS	BODY	MEANING
200	binary file	OK
400	GeneralError	Bad Request
500	GeneralError	Internal Server Error

PUT /v3/DocumentCollections/{id}/signer/{signerId}/replace

BEARER TOKEN

Replace old signer with new signer

PATH PARAMETERS

NAME	TYPE		DESCRIPTION
id	string (uuid)	required	—
signerId	string (uuid)	required	—

REQUEST BODY [ReplaceSignerDTO](#)

FIELD	TYPE	DESCRIPTION
newSignerName	string	—
newSignerMeans	string	—

RESPONSES

STATUS	BODY	MEANING
200	no body	OK
400	GeneralError	Bad Request
500	GeneralError	Internal Server Error

PUT

/v3/DocumentCollections/{id}/serversign

BEARER TOKEN

Extra server document collection signing by id

PATH PARAMETERS

NAME	TYPE		DESCRIPTION
id	string (uuid)	required	—

RESPONSES

STATUS	BODY	MEANING
200	no body	OK
400	<u>GeneralError</u>	Bad Request
500	<u>GeneralError</u>	Internal Server Error

SECTION 10.7 • SIGNERS

Signers

Recipient level operations on an existing document collection.

1 ENDPOINTS IN THIS SECTION

PUT

/v3/Signers/{id}/signer/{signerId}/replace

PUT

/v3/Signers/{id}/signer/{signerId}/replace

BEARER TOKEN

Replace old signer with new signer in document collection

Replaces an existing signer with a new signer, updating contact information, notes, and authentication settings.

PATH PARAMETERS

NAME	TYPE		DESCRIPTION
id	string (uuid)	required	Document collection ID
signerId	string (uuid)	required	Signer ID to replace

REQUEST BODY

ReplaceSignerWithDetailsDTO

FIELD	TYPE	DESCRIPTION
newSignerName	string	—
newSignerMeans	string	—
newNotes	Notes	—
└ id	string (uuid)	—
└ signerNote	string	—
└ userNote	string	—
└ emailNote	string	—
newOtpMode	OtpMode	None = 0, CodeRequired = 1, PasswordRequired = 2, CodeAndPasswordRequired = 3
newOtpIdentification	string	—
newAuthenticationMode	AuthMode	None = 0, IDP = 1, VisualIDP = 2

EXAMPLE REQUEST BODY

— every optional field is shown; send only the ones you need

```
{
  "newSignerName": "Noa Cohen",
  "newSignerMeans": "noa@acme.example.com",
  "newNotes": {
    "id": "3f2504e0-4f89-11d3-9a0c-0305e82c3301",
    "signerNote": "Please review and sign by Friday.",
    "userNote": "Sent by the billing system.",
    "emailNote": "Your service agreement is ready."
  },
  "newOtpMode": 1,
  "newOtpIdentification": "0501234567",
  "newAuthenticationMode": 0
}
```

RESPONSES

STATUS	BODY	MEANING
200	no body	OK
400	GeneralError	Bad Request

STATUS	BODY	MEANING
500	<u>GeneralError</u>	Internal Server Error

SECTION 10.8 • DISTRIBUTION

Distribution (mass sending)

Send the same template to hundreds of recipients in one call, each one receiving a private copy pre-filled with their own data.

8 ENDPOINTS IN THIS SECTION

POST

 /v3/Distribution/signers

POST

 /v3/Distribution

GET

 /v3/Distribution

GET

 /v3/Distribution/{id}

DELETE

 /v3/Distribution/{id}

GET

 /v3/Distribution/resend/{id}

GET

 /v3/Distribution/resend/{id}/status/{status}

GET

 /v3/Distribution/download/{id}

POST

 /v3/Distribution/signers

BEARER TOKEN

Extract signers from excel file

Excel file columns representation : **Mandatory columns:** column 1 - FirstName, column 2 - LastName, column 3 - Phone/Email **Optional columns:** column 4 --to-- column N - FieldName All worksheets (tabs) in the workbook are read; hidden or empty sheets are skipped. Expected files type: xls ----- data:application/vnd.ms-excel;base64,.... xlsx ----- data:application/vnd.openxmlformats-officedocument.spreadsheetml.sheet;base64,....

REQUEST BODY SignersForDistributionMechanismDTO

FIELD	TYPE	DESCRIPTION
base64File	string	—

RESPONSES

STATUS	BODY	MEANING
200	<u>ReadSignersFromFileResponseDTO</u>	OK
400	<u>GeneralError</u>	Bad Request
500	<u>GeneralError</u>	Internal Server Error

POST

/v3/Distribution

BEARER TOKEN

Send new document collection for each signer and set all fields in document collection to signer

Each signer may carry an optional **sendingMethod** - SMS = 1, Email = 2, WhatsApp = 4 (Tablet is not a distribution channel). Email requires a valid email among the signer means; SMS and WhatsApp require a valid phone. When omitted, the channel is inferred from the means: a valid email becomes Email, otherwise SMS.

REQUEST BODY CreateDistributionDocumentsDT0

FIELD	TYPE	DESCRIPTION
name	string	—
templateId	string (uuid)	—
signers	BaseSigner[]	—
└ fullName	string	—
└ sendingMethod	SendingMethod	SMS = 1, Email = 2, Tablet = 3, WhatsApp = 4
└ signerMeans	string	—
└ signerSecondaryMeans	string	—
└ phoneExtension	string	—
└ shouldSendOTP	boolean	—
└ fields	FieldNameToValuePair[]	—
└ └ fieldName	string	—
└ └ fieldValue	string	—
signDocumentWithServerSigning	boolean	—
shouldEnableMeaningOfSignature	boolean	—

EXAMPLE REQUEST BODY — every optional field is shown; send only the ones you need

```
{
  "name": "Service agreement 2026",
  "templateId": "3f2504e0-4f89-11d3-9a0c-0305e82c3301",
  "signers": [
    {
      "fullName": "Dana Levi",
      "sendingMethod": 2,
      "signerMeans": "dana@acme.example.com",
      "signerSecondaryMeans": "0501234567",
      "phoneExtension": "972",
      "shouldSendOTP": true,
      "fields": [
        {
          "fieldName": "CustomerName",
          "fieldValue": "ACME Ltd."
        }
      ]
    }
  ],
  "signDocumentWithServerSigning": false,
  "shouldEnableMeaningOfSignature": true
}
```

RESPONSES

STATUS	BODY	MEANING
200	no body	OK
400	GeneralError	Bad Request
500	GeneralError	Internal Server Error

GET /v3/Distribution

BEARER TOKEN

Get all distribution document collections by search criteria

Request offset number start from 1 limit number start from 1 Response

QUERY PARAMETERS

NAME	TYPE		DESCRIPTION
key	string	optional	—
from	string	optional	—
to	string	optional	—
offset	integer	optional	—
limit	integer	optional	—
searchParameter	SearchParameter	optional	0, 1, 2
sortBy	string	optional	—
sortOrder	string	optional	—

RESPONSES

STATUS	BODY	MEANING
200	AllDistributionDocumentsResponseDTO	OK
400	GeneralError	Bad Request
500	GeneralError	Internal Server Error

GET

/v3/Distribution/{id}

BEARER TOKEN

Get all documents of distribution id

Request offset number start from 1 limit number start from 1 Response

DocumentStatus: Created = 1, Sent = 2, Viewed = 3, Signed = 4, Declined = 5, SendingFailed = 6, Deleted = 7, Canceled = 8

SignerStatus: Sent = 1, Viewed = 2, Signed = 3, Rejected = 4

SendingMethod: SMS = 1, Email = 2, Tablet = 3, WhatsApp = 4

PATH PARAMETERS

NAME	TYPE		DESCRIPTION
id	string (uuid)	required	—

QUERY PARAMETERS

NAME	TYPE		DESCRIPTION
key	string	optional	—
from	string	optional	—
to	string	optional	—
offset	integer	optional	—
limit	integer	optional	—

RESPONSES

STATUS	BODY	MEANING
200	<u>AllDistributionDocumentsExpandedResposneDTO</u>	OK
400	<u>GeneralError</u>	Bad Request
500	<u>GeneralError</u>	Internal Server Error

DELETE

/v3/Distribution/{id}

BEARER TOKEN

Delete all documents of distribution id

PATH PARAMETERS

NAME	TYPE		DESCRIPTION
id	string (uuid)	required	—

RESPONSES

STATUS	BODY	MEANING
200	no body	OK
400	<u>GeneralError</u>	Bad Request
500	<u>GeneralError</u>	Internal Server Error

GET

/v3/Distribution/resend/{id}

BEARER TOKEN

Resend all unsigned documents of distribution id to there signers

PATH PARAMETERS

NAME	TYPE		DESCRIPTION
id	string (uuid)	required	—

RESPONSES

STATUS	BODY	MEANING
200	no body	OK
400	<u>GeneralError</u>	Bad Request
500	<u>GeneralError</u>	Internal Server Error

GET

/v3/Distribution/resend/{id}/status/{status}

BEARER TOKEN

Resend documents in specific status for distribution id

DocumentStatus: Created = 1, Sent = 2, Viewed = 3, Signed = 4, Declined = 5, SendingFailed = 6, Deleted = 7, Canceled = 8

PATH PARAMETERS

NAME	TYPE		DESCRIPTION
id	string (uuid)	required	Distribution ID
status	DocumentStatus	required	Document status to resend Created = 1, Sent = 2, Viewed = 3, Signed = 4, Declined = 5, SendingFailed = 6, Deleted = 7, Canceled = 8

RESPONSES

STATUS	BODY	MEANING
200	no body	OK
400	GeneralError	Bad Request
500	GeneralError	Internal Server Error

GET

/v3/Distribution/download/{id}

BEARER TOKEN

FILE DOWNLOAD

Download all signed documents of distribution id

As zip file, every document name inside zip file will be the signer name

PATH PARAMETERS

NAME	TYPE		DESCRIPTION
id	string (uuid)	required	–

RESPONSES

STATUS	BODY	MEANING
200	binary file	OK
400	GeneralError	Bad Request
500	GeneralError	Internal Server Error

SECTION 10.9 • SELF SIGN

Self sign & certificate signing

Sign a document yourself - with a graphic signature, a smart card, a server certificate (Signer1) or an eIDAS qualified signature.

8 ENDPOINTS IN THIS SECTION

POST /v3/SelfSign **PUT** /v3/SelfSign **DELETE** /v3/SelfSign/{id} **GET** /v3/SelfSign/download/smartcard

POST /v3/SelfSign/sign **POST** /v3/SelfSign/CreateSmartCardSigningFlow **POST** /v3/SelfSign/sign/verify

POST /v3/SelfSign/CheckidentityFlowEIDASSign

POST /v3/SelfSign

BEARER TOKEN

Create SelfSign document

REQUEST BODY CreateSelfSignDocumentDTO

FIELD	TYPE	DESCRIPTION
base64File	string	—
name	string	—
sourceTemplateId	string	—

EXAMPLE REQUEST BODY — every optional field is shown; send only the ones you need

```
{
  "base64File": "data:application/pdf;base64,JVBERi0xLjcKJc...",
  "name": "Service agreement 2026",
  "sourceTemplateId": "string"
}
```

RESPONSES

STATUS	BODY	MEANING
200	<u>SelfSignCountResponseDTO</u>	OK
400	<u>GeneralError</u>	Bad Request
500	<u>GeneralError</u>	Internal Server Error

PUT

/v3/SelfSign

BEARER TOKEN

Update SelfSign document

DocumentOperation : Save = 1, Decline = 2, Close = 3. Close will sign document

REQUEST BODY UpdateSelfSignDocumentDTO

FIELD	TYPE	DESCRIPTION
documentCollectionId	string (uuid)	—
documentId	string (uuid)	—
name	string	—
fields	<u>PDFFields</u>	—
└ textFields	<u>TextField[]</u>	—
└ ┌ fieldName	string	—
└ ┌ description	string	—
└ ┌ x	number	—
└ ┌ y	number	—
└ ┌ width	number	—
└ ┌ height	number	—
└ ┌ mandatory	boolean	—
└ ┌ page	integer	—
└ ┌ recipientRole	string	—
└ ┌ textFieldType	<u>TextFieldType</u>	Text = 1, Date = 2, Number = 3, Phone = 4, Email = 5, Custom = 6, Time = 7
└ ┌ customerRegex	string	—
└ ┌ value	string	—
└ ┌ isHidden	boolean	—
└ signatureFields	<u>SignatureField[]</u>	—
└ ┌ fieldName	string	—
└ ┌ description	string	—
└ ┌ x	number	—
└ ┌ y	number	—
└ ┌ width	number	—
└ ┌ height	number	—
└ ┌ mandatory	boolean	—
└ ┌ page	integer	—
└ ┌ recipientRole	string	—
└ ┌ image	string	—
└ ┌ signingType	<u>SignatureFieldType</u>	Graphic = 1, SmartCard = 2, Server = 3
└ ┌ signatureKind	<u>SignatureFieldKind</u>	0, 1

FIELD	TYPE	DESCRIPTION
└ radioGroupFields	RadioGroupField[]	—
└ fieldName	string	—
└ radioFields	RadioField[]	—
└ selectedRadioName	string	—
└ checkBoxFields	CheckBoxField[]	—
└ fieldName	string	—
└ description	string	—
└ x	number	—
└ y	number	—
└ width	number	—
└ height	number	—
└ mandatory	boolean	—
└ page	integer	—
└ recipientRole	string	—
└ isChecked	boolean	—
└ choiceFields	ChoiceField[]	—
└ fieldName	string	—
└ description	string	—
└ x	number	—
└ y	number	—
└ width	number	—
└ height	number	—
└ mandatory	boolean	—
└ page	integer	—
└ recipientRole	string	—
└ options	string[]	—
└ selectedOption	string	—
operation	DocumentOperation	Save = 1, Decline = 2, Close = 3
signerAuthentication	SignerAuthentication	—
└ signer1Credential	Signer1Credential	—
└ password	string	—
└ certificateId	string	—
└ shouldUseAddDetails	boolean	—
└ signerToken	string	—
└ otpDetails	OtpDetails	—
└ identification	string	—
└ code	string	—

FIELD	TYPE	DESCRIPTION
└ attempts	integer	—
└ expirationTime	string (date-time)	—
└ mode	<u>OtpMode</u>	None = 0, CodeRequired = 1, PasswordRequired = 2, CodeAndPasswordRequired = 3
└ signer1Configuration	<u>Signer1Configuration</u>	—
└ endpoint	string	—
└ user	string	—
└ password	string	—
└ authenticationMode	<u>AuthMode</u>	None = 0, IDP = 1, VisualIDP = 2
useForAllFields	boolean	—

RESPONSES

STATUS	BODY	MEANING
200	<u>SelfSignUpdateDocumentResult</u>	OK
400	<u>GeneralError</u>	Bad Request
500	<u>GeneralError</u>	Internal Server Error

DELETE

/v3/SelfSign/{id}

BEARER TOKEN

Delete SelfSign document

PATH PARAMETERS

NAME	TYPE		DESCRIPTION
id	string (uuid)	required	—

RESPONSES

STATUS	BODY	MEANING
200	no body	OK
400	<u>GeneralError</u>	Bad Request
500	<u>GeneralError</u>	Internal Server Error

GET /v3/SelfSign/download/smartcard

BEARER TOKEN

FILE DOWNLOAD

Download smartCard desktop client installer**RESPONSES**

STATUS	BODY	MEANING
200	binary file	OK
400	GeneralError	Bad Request
500	GeneralError	Internal Server Error

POST /v3/SelfSign/sign

BEARER TOKEN

FILE DOWNLOAD

Sign file using Signer1 service

Signs a file using external Signer1 signing service and returns the signed file.

REQUEST BODY [Signer1FileSigingDTO](#)

FIELD	TYPE	DESCRIPTION
base64File	string	—
sigingFileType	SigningFileType	0, 1, 2, 3, 4
signer1Credential	Signer1Credential	—
└ password	string	—
└ certificateId	string	—
└ shouldUseADDetails	boolean	—
└ signerToken	string	—
fileName	string	—

RESPONSES

STATUS	BODY	MEANING
200	binary file	OK
400	GeneralError	Bad Request
500	GeneralError	Internal Server Error

POST

/v3/SelfSign/CreateSmartCardSigningFlow

BEARER TOKEN

Create smart card signing flow session

Creates a signing session for smart card-based signing with field mapping.

REQUEST BODY SmartCardSigningFlowDTO

FIELD	TYPE	DESCRIPTION
token	string	—
fields	SmartCardSignFlowFieldsDTO[]	—
└ documentId	string (uuid)	—
└ fieldName	string	—
└ image	string	—

RESPONSES

STATUS	BODY	MEANING
200	no body	OK
400	GeneralError	Bad Request
500	GeneralError	Internal Server Error

POST /v3/SelfSign/sign/verify**BEARER TOKEN****Verify Signer1 credentials**

Validates Signer1 service credentials before signing.

REQUEST BODY SignerAuthentication

FIELD	TYPE	DESCRIPTION
signer1Credential	<u>Signer1Credential</u>	—
└ password	string	—
└ certificateId	string	—
└ shouldUseADDetails	boolean	—
└ signerToken	string	—
otpDetails	<u>OtpDetails</u>	—
└ identification	string	—
└ code	string	—
└ attempts	integer	—
└ expirationTime	string (date-time)	—
└ mode	<u>OtpMode</u>	None = 0, CodeRequired = 1, PasswordRequired = 2, CodeAndPasswordRequired = 3
signer1Configuration	<u>Signer1Configuration</u>	—
└ endpoint	string	—
└ user	string	—
└ password	string	—
authenticationMode	<u>AuthMode</u>	None = 0, IDP = 1, VisualIDP = 2

RESPONSES

STATUS	BODY	MEANING
200	no body	OK
400	<u>GeneralError</u>	Bad Request
500	<u>GeneralError</u>	Internal Server Error

POST

/v3/SelfSign/CheckidentityFlowEIDASSign

BEARER TOKEN

Check identity flow for EIDAS signing

Processes identity verification flow for EIDAS-compliant signing.

REQUEST BODY IdentityFlowDTO

FIELD	TYPE	DESCRIPTION
signerToken	string	—
code	string	—

RESPONSES

STATUS	BODY	MEANING
200	<u>IdentityCheckFlowResult</u>	OK
400	<u>GeneralError</u>	Bad Request
500	<u>GeneralError</u>	Internal Server Error

SECTION 10.10 • LINKS

Signing links & video sessions

Retrieve the links the caller still has to sign, publish a public single link for a template, and open a video-identification signing session.

4 ENDPOINTS IN THIS SECTION

GET

 /v3/Links

POST

 /v3/Links/videoConference

GET

 /v3/Links/Template/{id}

POST

 /v3/Links/template/{id}

GET

 /v3/Links

BEARER TOKEN

Get current user links to sign on all documents that not sign already

QUERY PARAMETERS

NAME	TYPE	DESCRIPTION
key	string	optional —
offset	integer	optional —
limit	integer	optional —
from	string	optional —
to	string	optional —
sortBy	string	optional —
sortOrder	string	optional —

RESPONSES

STATUS	BODY	MEANING
200	UserSigningLinksResponseDTO	OK
400	GeneralError	Bad Request
500	GeneralError	Internal Server Error

POST

/v3/Links/videoConference

BEARER TOKEN

Create video conference for document signing

Creates a video conference session for collaborative document signing.

REQUEST BODY CreateVideoConferencesDTO

FIELD	TYPE	DESCRIPTION
documentCollectionName	string	—
videoConferenceUsers	<u>VideoConferenceUser[]</u>	—
└ means	string	—
└ fullName	string	—
└ sendingMethod	<u>SendingMethod</u>	SMS = 1, Email = 2, Tablet = 3, WhatsApp = 4
└ phoneExtension	string	—

EXAMPLE REQUEST BODY — every optional field is shown; send only the ones you need

```
{
  "documentCollectionName": "Service agreement 2026",
  "videoConferenceUsers": [
    {
      "means": "dana@acme.example.com",
      "fullName": "Dana Levi",
      "sendingMethod": 2,
      "phoneExtension": "972"
    }
  ]
}
```

RESPONSES

STATUS	BODY	MEANING
200	<u>VideoConferenceResult</u>	OK
400	<u>GeneralError</u>	Bad Request
500	<u>GeneralError</u>	Internal Server Error

GET

/v3/Links/Template/{id}

BEARER TOKEN

Get single link information for template

Returns the single signing link configuration for a template.

PATH PARAMETERS

NAME	TYPE		DESCRIPTION
id	string (uuid)	required	Template ID

RESPONSES

STATUS	BODY	MEANING
200	TemplateSingleLinkDTO	OK
400	GeneralError	Bad Request
500	GeneralError	Internal Server Error

POST

/v3/Links/template/{id}

BEARER TOKEN

Create or update single link information for template

Creates or updates the single signing link configuration for a template.

PATH PARAMETERS

NAME	TYPE		DESCRIPTION
id	string (uuid)	required	Template ID

REQUEST BODY

TemplateSingleLinkDTO

FIELD	TYPE	DESCRIPTION
templateId	string (uuid)	—
singleLinkAdditionalResources	<u>SingleLinkAdditionalResource[]</u>	—
└ templateId	string (uuid)	—
└ type	<u>SingleLinkAdditionalResourceType</u>	1
└ data	string	—
└ isMandatory	boolean	—

RESPONSES

STATUS	BODY	MEANING
200	no body	OK
400	GeneralError	Bad Request
500	GeneralError	Internal Server Error

SECTION 10.11 • REPORTS

Reports

Usage reporting for the company, plus scheduled reports delivered automatically.

6 ENDPOINTS IN THIS SECTION

GET /v3/Reports/UsageData **GET** /v3/Reports/Settings **POST** /v3/Reports/FrequencyReports

GET /v3/Reports/FrequencyReports **DELETE** /v3/Reports/FrequencyReports

GET /v3/Reports/FrequencyReports/Download

GET /v3/Reports/UsageData

BEARER TOKEN

Get user usage data reports

offset number start from 0 limit number start from 1 If isCSV is true, returns CSV file instead of JSON response.

DocumentStatus: Created = 1, Sent = 2, Viewed = 3, Signed = 4, Declined = 5, SendingFailed = 6, Deleted = 7, Canceled = 8

QUERY PARAMETERS

NAME	TYPE		DESCRIPTION
from	string (date-time)	optional	Start date for report period
to	string (date-time)	optional	End date for report period
docStatuses	<u>DocumentStatus[]</u>	optional	Filter by document statuses Created = 1, Sent = 2, Viewed = 3, Signed = 4, Declined = 5, SendingFailed = 6, Deleted = 7, Canceled = 8
groupIds	string (uuid)[]	optional	Filter by group IDs
includeDistributionDocs	boolean	optional	Whether to include distribution documents
offset	integer	optional	Pagination offset
limit	integer	optional	Pagination limit
isCSV	boolean	optional	Return as CSV file if true, otherwise JSON

RESPONSES

STATUS	BODY	MEANING
200	<u>AllUsageDataReportsDTO</u>	OK
400	<u>GeneralError</u>	Bad Request
500	<u>GeneralError</u>	Internal Server Error

GET

/v3/Reports/Settings

BEARER TOKEN

Report screen settings

An authorized API call. Returns whether historical (archive) reporting is available, and the maximum date range in days supported without it.

RESPONSES

STATUS	BODY	MEANING
200	ReportSettings	OK
400	GeneralError	Bad Request
500	GeneralError	Internal Server Error

POST

/v3/Reports/FrequencyReports

BEARER TOKEN

Create scheduled frequency reports

Sets up periodic reports that will be automatically generated and sent.

QUERY PARAMETERS

NAME	TYPE		DESCRIPTION
frequency	ReportFrequency	optional	Report frequency (Daily, Weekly, Monthly, etc.) 0, 1, 2, 3
reportTypeStr	string	optional	Type of report to generate

RESPONSES

STATUS	BODY	MEANING
400	GeneralError	Bad Request
500	GeneralError	Internal Server Error

GET

/v3/Reports/FrequencyReports

BEARER TOKEN

Get all scheduled frequency reports

Returns list of all configured periodic reports for the user.

RESPONSES

STATUS	BODY	MEANING
200	AllUserPeriodicReports	OK
400	GeneralError	Bad Request
500	GeneralError	Internal Server Error

DELETE

/v3/Reports/FrequencyReports

BEARER TOKEN

Delete all scheduled frequency reports

Removes all configured periodic reports for the current user.

RESPONSES

STATUS	BODY	MEANING
400	GeneralError	Bad Request
500	GeneralError	Internal Server Error

GET

/v3/Reports/FrequencyReports/Download

NO TOKEN

Download generated frequency report file

Downloads a previously generated frequency report file by its ID.

QUERY PARAMETERS

NAME	TYPE		DESCRIPTION
id	string (uuid)	optional	Report file ID

RESPONSES

STATUS	BODY	MEANING
400	GeneralError	Bad Request
500	GeneralError	Internal Server Error

SECTION 10.12 • DASHBOARD

Dashboard

Aggregated counters for the home screen.

1 ENDPOINTS IN THIS SECTION

GET

/v3/Dashboard/view

GET

/v3/Dashboard/view

BEARER TOKEN

Get dashboard view data

Returns dashboard statistics and overview data for the current user.

RESPONSES

STATUS	BODY	MEANING
200	no body	OK
400	<u>GeneralError</u>	Bad Request
500	<u>GeneralError</u>	Internal Server Error

SECTION 10.13 • LICENSE

Licence & quota

The limits of the subscription and how much of it is used.

1 ENDPOINTS IN THIS SECTION

GET

/v3/License

BEARER TOKEN

QUERY PARAMETERS

NAME	TYPE		DESCRIPTION
readUsage	boolean	optional	—

RESPONSES

STATUS	BODY	MEANING
200	LicenseInfoResponse	OK
400	GeneralError	Bad Request
500	GeneralError	Internal Server Error

SECTION 11

Data models

Every object the API exchanges, in alphabetical order, with its fields and types.

Objects are listed alphabetically, each with its own properties. A type printed as a link is itself a model in this chapter, so a nested object is one jump away; enumerations link to section 8. The request bodies in section 9 show the same objects already expanded, so that each endpoint can be read on its own.

ActivationDTO request body

FIELD	TYPE	DESCRIPTION
token	string	—

AdditionalGroupMapper nested object

FIELD	TYPE	DESCRIPTION
id	string (uuid)	—
companyId	string (uuid)	—
userId	string (uuid)	—
groupId	string (uuid)	—
company	Company	—
user	User	—
group	Group	—

AdminAllGroupsResponseDTO response body

FIELD	TYPE	DESCRIPTION
groups	GroupResponseAdminDTO[]	—

AdminAllUsersResponseDTO response body

FIELD	TYPE	DESCRIPTION
users	UserResponseAdminDTO[]	—

AdminCreateGroupDTO request body

FIELD	TYPE	DESCRIPTION
name	string	—

AdminCreateUserDTO request body

FIELD	TYPE	DESCRIPTION
name	string	—
email	string	—
username	string	—
type	<u>UserType</u>	Basic = 1, Editor = 2, CompanyAdmin = 3
groupId	string (uuid)	—
additionalGroupsIds	string (uuid)[]	—
language	<u>Language</u>	en = 1, he = 2

AllContactResponseDTO response body

FIELD	TYPE	DESCRIPTION
contacts	<u>ContactResponseDTO[]</u>	—

AllContactsGroupsResponseDTO response body

FIELD	TYPE	DESCRIPTION
contactGroups	<u>ContactsGroupResponseDTO[]</u>	—

AllDistributionDocumentsExpandedResposneDTO response body

FIELD	TYPE	DESCRIPTION
totalPending	integer	—
totalSigned	integer	—
totalServerSigned	integer	—
totalDeclined	integer	—
totalFailed	integer	—
documentCollections	<u>DistributionDocumentExpandedResposneDTO[]</u>	—
totalCreatedButNotSent	integer	—
totalViewed	integer	—
shouldSignUsingSigner1AfterDocumentSigningFlow	boolean	—

AllDistributionDocumentsResposneDTO response body

FIELD	TYPE	DESCRIPTION
documentCollections	<u>DistributionDocumentResposneDTO[]</u>	—

AllDocumentCollectionsResposneDTO response body

FIELD	TYPE	DESCRIPTION
documentCollections	<u>DocumentCollectionResposneDTO[]</u>	—

AllTemplatesResponseDTO response body

FIELD	TYPE	DESCRIPTION
templates	TemplateResponseDTO[]	—

AllUsageDataReportsDTO response body

FIELD	TYPE	DESCRIPTION
usageDataReports	UsageDataReportDTO[]	—

AllUserPeriodicReports response body

FIELD	TYPE	DESCRIPTION
userPeriodicReports	UserPeriodicReport[]	—

Appendix nested object

FIELD	TYPE	DESCRIPTION
name	string	—
base64File	string	—
fileExtention	string	—
fileContent	string (base64)	—

BaseSigner nested object

FIELD	TYPE	DESCRIPTION
fullName	string	—
sendingMethod	SendingMethod	SMS = 1, Email = 2, Tablet = 3, WhatsApp = 4
signerMeans	string	—
signerSecondaryMeans	string	—
phoneExtension	string	—
shouldSendOTP	boolean	—
fields	FieldNameToValuePair[]	—

BaseUserDTO request body

FIELD	TYPE	DESCRIPTION
email	string	—

BatchRequestDTO request body

FIELD	TYPE	DESCRIPTION
ids	string[]	—

ChangePasswordDTO request body

FIELD	TYPE	DESCRIPTION
oldPassword	string	—
newPassword	string	—

CheckBoxField nested object

FIELD	TYPE	DESCRIPTION
fieldName	string	—
description	string	—
x	number	—
y	number	—
width	number	—
height	number	—
mandatory	boolean	—
page	integer	—
recipientRole	string	—
isChecked	boolean	—

ChoiceField nested object

FIELD	TYPE	DESCRIPTION
fieldName	string	—
description	string	—
x	number	—
y	number	—
width	number	—
height	number	—
mandatory	boolean	—
page	integer	—
recipientRole	string	—
options	string[]	—
selectedOption	string	—

Company nested object

FIELD	TYPE	DESCRIPTION
id	string (uuid)	—
name	string	—
status	<u>CompanyStatus</u>	0, 1
programId	string (uuid)	—
programUtilization	<u>ProgramUtilization</u>	—
companyConfiguration	<u>CompanyConfiguration</u>	—
companySigner1Details	<u>CompanySigner1Details</u>	—
transactionId	string	—

CompanyConfiguration nested object

FIELD	TYPE	DESCRIPTION
companyId	string (uuid)	—
signatureColor	string	—
shouldSendSignedDocument	boolean	—
shouldNotifyWhileSignerSigned	boolean	—
shouldEnableSignReminders	boolean	—
signReminderFrequencyInDays	integer	—
canUserControlReminderSettings	boolean	—
signerLinkExpirationInHours	integer	—
language	Language	en = 1, he = 2
base64Logo	string	—
emailTemplates	EmailHtmlBodyTemplates	—
documentDeletionConfiguration	DocumentDeletionConfiguration	—
companyMessages	CompanyMessage[]	—
messageProviders	MessageProvider[]	—
shouldSendWithOTPByDefault	boolean	—
enableVisualIdentityFlow	boolean	—
enableDisplaySignerNameInSignature	boolean	—
defaultSigningType	SignatureFieldType	Graphic = 1, SmartCard = 2, Server = 3
shouldSendDocumentNotifications	boolean	—
documentNotificationsEndpoint	string	—
shouldForceOTPInLogin	boolean	—
shouldEnableMeaningOfSignatureOption	boolean	—
shouldEnableVideoConference	boolean	—
whatsAppEnabled	boolean	—
shouldAddAppendicesAttachmentsToSendMail	boolean	—
isPersonzliedPFX	boolean	—
recentPasswordsAmount	integer	—
passwordExpirationInDays	integer	—
minimumPasswordLength	integer	—
enableTabletsSupport	boolean	—
ssoLoginUrl	string	—
scimBearerToken	string	—
scimDefaultGroupId	string (uuid)	—

CompanyMessage nested object

FIELD	TYPE	DESCRIPTION
id	string (uuid)	—
companyId	string (uuid)	—
sendingMethod	SendingMethod	SMS = 1, Email = 2, Tablet = 3, WhatsApp = 4
messageType	MessageType	0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20
content	string	—
language	Language	en = 1, he = 2

CompanySigner1Details nested object

FIELD	TYPE	DESCRIPTION
certId	string	—
certPassword	string	—
shouldSignAsDefaultValue	boolean	—
shouldShowInUI	boolean	—
signer1Configuration	Signer1Configuration	—

CompanySigner1DetailsResponseDTO nested object

FIELD	TYPE	DESCRIPTION
certId	string	The company's certificate id. Not a secret - see the remarks on this type.
shouldSignAsDefaultValue	boolean	Whether the server-signing option should start out selected.
shouldShowInUI	boolean	Whether the company has server signing switched on in the management panel. Clients gate the visibility of the server-signing control on this; the server enforces it independently when a document collection is created.

ContactDTO request body

FIELD	TYPE	DESCRIPTION
name	string	—
email	string	—
phone	string	—
phoneExtension	string	—
defaultSendingMethod	SendingMethod	SMS = 1, Email = 2, Tablet = 3, WhatsApp = 4
seals	SealDTO[]	—
searchTag	string	—

ContactGroupMemberDTO nested object

FIELD	TYPE	DESCRIPTION
contactId	string (uuid)	—
order	integer	—

ContactGroupMemberResponseDTO nested object

FIELD	TYPE	DESCRIPTION
id	string (uuid)	—
name	string	—
contactId	string (uuid)	—
order	integer	—
contact	<u>ContactResponseDTO</u>	—

ContactResponseDTO response body

FIELD	TYPE	DESCRIPTION
id	string (uuid)	—
name	string	—
email	string	—
phone	string	—
phoneExtension	string	—
defaultSendingMethod	<u>SendingMethod</u>	SMS = 1, Email = 2, Tablet = 3, WhatsApp = 4
seals	<u>Seal[]</u>	—
searchTag	string	—

ContactsDTO request body

FIELD	TYPE	DESCRIPTION
base64File	string	—

ContactsGroupDTO request body

FIELD	TYPE	DESCRIPTION
name	string	—
contactsGroupMembers	<u>ContactGroupMemberDTO[]</u>	—

ContactsGroupResponseDTO response body

FIELD	TYPE	DESCRIPTION
id	string (uuid)	—
name	string	—
contactsGroupMembers	<u>ContactGroupMemberResponseDTO[]</u>	—

CreateContactGroupResponseDTO response body

FIELD	TYPE	DESCRIPTION
id	string (uuid)	—

CreateContactResponseDTO response body

FIELD	TYPE	DESCRIPTION
contactId	string (uuid)	—

CreateContactsResponseDTO response body

FIELD	TYPE	DESCRIPTION
contactsId	string (uuid)[]	—

CreateDistributionDocumentsDTO request body

FIELD	TYPE	DESCRIPTION
name	string	—
templateId	string (uuid)	—
signers	BaseSigner[]	—
signDocumentWithServerSigning	boolean	—
shouldEnableMeaningOfSignature	boolean	—

CreateDocumentCollectionDTO request body

FIELD	TYPE	DESCRIPTION
documentMode	DocumentMode	OrderedGroupSign = 1, GroupSign = 2, Online = 3, SelfSign = 100
documentName	string	—
templates	string (uuid)[]	—
senderNote	string	—
signers	SignerDTO[]	—
readOnlyFields	SignerFieldDTO[]	—
redirectUrl	string	—
callbackUrl	string	—
senderAppendices	Appendix[]	—
sharedAppendices	SharedAppendixDTO[]	—
notifications	DocumentNotificationsDTO	—
shouldSignUsingSigner1AfterDocumentSigningFlow	boolean	—
shouldEnableMeaningOfSignature	boolean	—
dateFormat	DateFormat	1, 2, 3, 4

CreateDocumentCollectionResponseDTO response body

FIELD	TYPE	DESCRIPTION
documentCollectionId	string (uuid)	—
signerLinks	SignerLink[]	—

CreateGroupResponseAdminDTO response body

FIELD	TYPE	DESCRIPTION
groupId	string (uuid)	—

CreateSelfSignDocumentDTO request body

FIELD	TYPE	DESCRIPTION
base64File	string	—
name	string	—
sourceTemplateId	string	—

CreateSimpleDocumentDTO request body

FIELD	TYPE	DESCRIPTION
templateId	string (uuid)	—
documentName	string	—
signerName	string	—
signerMeans	string	—
signerExtention	string	—
redirectUrl	string	—
callbackUrl	string	—
signerOTPMMeans	string	—
shouldSend	boolean	—
shouldSendSignedDocument	boolean	—

CreateTemplateDTO request body

FIELD	TYPE	DESCRIPTION
base64File	string	—
name	string	—
metaData	string	—
isOneTimeUseTemplate	boolean	—

CreateUserDTO request body

FIELD	TYPE	DESCRIPTION
name	string	—
language	Language	en = 1, he = 2
dateFormat	DateFormat	1, 2, 3, 4
email	string	—
password	string	—
reCAPCHA	string	—
sendActivationLink	boolean	—
username	string	—

CreateUserResponseAdminDTO response body

FIELD	TYPE	DESCRIPTION
userId	string (uuid)	—

CreateVideoConferencesDTO request body

FIELD	TYPE	DESCRIPTION
documentCollectionName	string	—
videoConferenceUsers	VideoConferenceUser[]	—

DeleteBatchRequestDTO request body

FIELD	TYPE	DESCRIPTION
ids	string[]	—

DeleteSignatureImageDTO request body

FIELD	TYPE	DESCRIPTION
signatureImage	string	—

DistributionDocumentExpandedResponseDTO nested object

FIELD	TYPE	DESCRIPTION
documentCollectionId	string (uuid)	—
distributionId	string (uuid)	—
name	string	—
creationTime	string (date-time)	—
isWillDeletedIn24Hours	boolean	—
user	string	—
documentStatus	<u>DocumentStatus</u>	Created = 1, Sent = 2, Viewed = 3, Signed = 4, Declined = 5, SendingFailed = 6, Deleted = 7, Canceled = 8
documentsIds	string (uuid)[]	—
signer	<u>SignerResponseDTO</u>	—

DistributionDocumentResponseDTO nested object

FIELD	TYPE	DESCRIPTION
documentCollectionId	string (uuid)	—
distributionId	string (uuid)	—
name	string	—
creationTime	string (date-time)	—
isWillDeletedIn24Hours	boolean	—
user	string	—

DocumentCollectionResponseDTO response body

FIELD	TYPE	DESCRIPTION
documentCollectionId	string (uuid)	—
distributionId	string (uuid)	—
name	string	—
documentStatus	<u>DocumentStatus</u>	Created = 1, Sent = 2, Viewed = 3, Signed = 4, Declined = 5, SendingFailed = 6, Deleted = 7, Canceled = 8
mode	<u>DocumentMode</u>	OrderedGroupSign = 1, GroupSign = 2, Online = 3, SelfSign = 100
documentsIds	string (uuid)[]	—
signers	<u>SignerResponseDTO</u> []	—
creationTime	string (date-time)	—
signedTime	string (date-time)	—
isWillDeletedIn24Hours	boolean	—
user	<u>UserResponseDTO</u>	—
shouldSignUsingSigner1AfterDocumentSigningFlow	boolean	—

DocumentCountResponseDTO response body

FIELD	TYPE	DESCRIPTION
documentId	string (uuid)	—
templateId	string (uuid)	—
documentName	string	—
pagesCount	integer	—

DocumentDeletionConfiguration nested object

FIELD	TYPE	DESCRIPTION
deleteSignedDocumentAfterXDays	integer	—
deleteUnsignedDocumentAfterXDays	integer	—

DocumentNotificationsDTO nested object

FIELD	TYPE	DESCRIPTION
shouldSend	boolean	—
shouldSendSignedDocument	boolean	—

DocumentPageResponseDTO response body

FIELD	TYPE	DESCRIPTION
documentId	string (uuid)	—
name	string	—
pdfFields	<u>PDFFields</u>	—
pagesCount	integer	—
pageNumber	integer	—
pageImage	string	—
pageWidth	number	—
pageHeight	number	—

DocumentPagesRangeResponseDTO response body

FIELD	TYPE	DESCRIPTION
documentPages	<u>DocumentPageResponseDTO[]</u>	—

DownloadBatchRequestDTO request body

FIELD	TYPE	DESCRIPTION
ids	string[]	—

DownloadDTO response body

FIELD	TYPE	DESCRIPTION
files	<u>FilesDTO[]</u>	—

DuplicateTemplateDT0 request body

FIELD	TYPE	DESCRIPTION
isOneTimeUseTemplate	boolean	—

DuplicateTemplateResponseDT0 response body

FIELD	TYPE	DESCRIPTION
name	string	—
newTemplateId	string (uuid)	—

EmailHtmlBodyTemplates nested object

FIELD	TYPE	DESCRIPTION
beforeSigningBase64String	string	—
afterSigningBase64String	string	—

ExtendedUserResponseDTO response body

FIELD	TYPE	DESCRIPTION
id	string (uuid)	—
companyId	string (uuid)	—
groupId	string (uuid)	—
name	string	—
companyName	string	—
groupName	string	—
email	string	—
phone	string	—
type	<u>UserType</u>	Basic = 1, Editor = 2, CompanyAdmin = 3
creationTime	string (date-time)	—
userConfiguration	<u>UserConfiguration</u>	—
programUtilizationId	string (uuid)	—
program	<u>ProgramResponseDTO</u>	—
companyLogo	string	—
companySigner1Details	<u>CompanySigner1DetailsResponseDTO</u>	—
shouldSendWithOTPByDefault	boolean	—
enableVisualIdentityFlow	boolean	—
enableVideoConferenceFlow	boolean	—
enableWhatsApp	boolean	—
defaultSigningType	<u>SignatureFieldType</u>	Graphic = 1, SmartCard = 2, Server = 3
transactionId	string	—
lastSeen	string (date-time)	—
username	string	—
enableSignReminderSettings	boolean	—
enableMeaningOfSignature	boolean	—
enableDisplaySignerNameInSignature	boolean	—
additionalGroupsIds	string (uuid)[]	—
shouldSignEidasSignatureFlow	boolean	—
enableTabletsSupport	boolean	—

ExternalLoginDTO request body

FIELD	TYPE	DESCRIPTION
token	string	—

FieldNameToValuePair nested object

FIELD	TYPE	DESCRIPTION
fieldName	string	—
fieldValue	string	—

FilesDTO nested object

FIELD	TYPE	DESCRIPTION
ext	string	—
data	string	—
name	string	—

GeneralError nested object

FIELD	TYPE	DESCRIPTION
errors	object	—
title	string	—
status	integer	—
traceId	string	—

Group nested object

FIELD	TYPE	DESCRIPTION
id	string (uuid)	—
name	string	—
companyId	string (uuid)	—
groupStatus	<u>GroupStatus</u>	1, 2
users	<u>User</u> []	—

GroupResponseAdminDTO nested object

FIELD	TYPE	DESCRIPTION
groupId	string (uuid)	—
name	string	—

IdentityCheckFlowResult response body

FIELD	TYPE	DESCRIPTION
token	string (uuid)	—

IdentityFlowDTO request body

FIELD	TYPE	DESCRIPTION
signerToken	string	—
code	string	—

InitConfigurationDTO response body

FIELD	TYPE	DESCRIPTION
<code>enableFreeTrailUsers</code>	boolean	—
<code>enableTabletsSupport</code>	boolean	—
<code>enableSigner1ExtraSigningTypes</code>	boolean	—
<code>shouldUseReCaptchaInRegistration</code>	boolean	—
<code>shouldUseSignerAuth</code>	boolean	—
<code>shouldUseSignerAuthDefault</code>	boolean	—
<code>enableShowSSOOnlyInUserUI</code>	boolean	—
<code>maxTemplatesPerProcess</code>	integer	Max templates that can be selected into a single signing process. Comes from GeneralSettings.MaxTemplatesPerProcess (appsettings.json), so the clients don't hard code it. Default 10.

LicenseCounters nested object

FIELD	TYPE	DESCRIPTION
<code>templates</code>	integer	—
<code>users</code>	integer	—
<code>documentsPerMonth</code>	integer	—
<code>smsPerMonth</code>	integer	—
<code>visualIdentificationsPerMonth</code>	integer	—
<code>useActiveDirectory</code>	boolean	—

LicenseInfoResponse response body

FIELD	TYPE	DESCRIPTION
<code>licenseLimits</code>	WeSignLicense	—
<code>licenseUsage</code>	LicenseCounters	—

LinkResponse response body

FIELD	TYPE	DESCRIPTION
<code>link</code>	string	—

LoginRequestDTO request body

FIELD	TYPE	DESCRIPTION
<code>email</code>	string	—
<code>password</code>	string	—

LogoutResponseDTO response body

FIELD	TYPE	DESCRIPTION
logoutURL	string	—

ManagementPeriodicReport nested object

FIELD	TYPE	DESCRIPTION
id	string (uuid)	—
userId	string (uuid)	—
reportType	<u>ManagementReportType</u>	0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15
lastTimeSent	string (date-time)	—
reportFrequency	<u>ManagementReportFrequency</u>	0, 1, 2
reportParameters	string	—
emails	<u>ManagementPeriodicReportEmail[]</u>	—
user	<u>User</u>	—

ManagementPeriodicReportEmail nested object

FIELD	TYPE	DESCRIPTION
id	string (uuid)	—
email	string	—
periodicReportId	string (uuid)	—
report	<u>ManagementPeriodicReport</u>	—

MergeTemplatesDTO request body

FIELD	TYPE	DESCRIPTION
templates	string[]	—
name	string	—
isOneTimeUseTemplate	boolean	—

MessageProvider nested object

FIELD	TYPE	DESCRIPTION
id	string (uuid)	—
companyId	string (uuid)	—
sendingMethod	<u>SendingMethod</u>	SMS = 1, Email = 2, Tablet = 3, WhatsApp = 4
providerType	<u>ProviderType</u>	1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 20
from	string	—
server	string	—
port	integer	—
user	string	—
password	string	—
enableSsl	boolean	—

Notes nested object

FIELD	TYPE	DESCRIPTION
id	string (uuid)	—
signerNote	string	—
userNote	string	—
emailNote	string	—

OtpDTO request body

FIELD	TYPE	DESCRIPTION
otpToken	string	—
code	string	—

OtpDetails nested object

FIELD	TYPE	DESCRIPTION
identification	string	—
code	string	—
attempts	integer	—
expirationTime	string (date-time)	—
mode	<u>OtpMode</u>	None = 0, CodeRequired = 1, PasswordRequired = 2, CodeAndPasswordRequired = 3

OtpResendDTO request body

FIELD	TYPE	DESCRIPTION
otpToken	string	—

PDFFields response body

FIELD	TYPE	DESCRIPTION
textFields	TextField[]	—
signatureFields	SignatureField[]	—
radioGroupFields	RadioGroupField[]	—
checkboxFields	CheckBoxField[]	—
choiceFields	ChoiceField[]	—

ProfileProgram nested object

FIELD	TYPE	DESCRIPTION
name	string	—
note	string	—
expired	string (date-time)	—
lastResetDate	string (date-time)	—
nextRenewalDate	string (date-time)	—
documentsLimits	integer	—
remainingDocuments	integer	—
smsLimit	integer	—
remainingSMS	integer	—
visualIdentificationsLimit	integer	—
videoConferenceLimit	integer	—
whatsAppLimit	integer	—
remainingVisualIdentifications	integer	—
remainingVideoConference	integer	—
remainingWhatsApp	integer	—
templatesLimit	integer	—
remainingTemplates	integer	—
usersLimit	integer	—
remainingUsers	integer	—
serverSignature	boolean	—
smartCard	boolean	—
isEcommerceProgram	boolean	—
viewLicense	UIViewLicense	—
programResetType	ProgramResetType	0, 1, 2, 3
isSmsProviderSupportGloballySend	boolean	—

ProgramResponseDTO nested object

FIELD	TYPE	DESCRIPTION
name	string	—
note	string	—
expiredTime	string (date-time)	—
lastResetDate	string (date-time)	—
nextRenewalDate	string	Calendar date the next quota refresh takes effect, as "yyyy-MM-dd". Null when the plan does not renew (DocumentsLimitOnly, TimeAndDocumentsLimit, free trial). The renewal job runs at 23:55 the previous evening, so the quota is already fresh from 00:00 on this date. A plain date string rather than a DateTime, deliberately. Responses are serialised by Newtonsoft with the UTC converters disabled and EF returns these values with DateTimeKind.Unspecified, so a DateTime would go out as "2026-05-15T00:00:00" - no Z, no offset. If anyone ever enables those converters, the same value would render as 14/05 for a client west of UTC: exactly the off-by-one this ticket removes, reintroduced at the serialisation layer. A date string cannot be shifted by a time zone.
documentsForMonth	integer	—
remainingDocumentsForMonth	integer	—
smsLimit	integer	—
remainingSMS	integer	—
visualIdentificationsLimit	integer	—
videoConferenceLimit	integer	—
whatsAppLimit	integer	—
remainingVisualIdentifications	integer	—
remainingVideoConference	integer	—
remainingWhatsApp	integer	—
remainingTemplates	integer	—
templatesLimit	integer	—
remainingUsers	integer	—
usersLimit	integer	—
shouldShowSelfSign	boolean	—
shouldShowGroupSign	boolean	—
shouldShowLiveMode	boolean	—
shouldShowContacts	boolean	—
serverSignature	boolean	—
smartCard	boolean	—
isEcommerceProgram	boolean	—

FIELD	TYPE	DESCRIPTION
<code>uiViewLicense</code>	UIViewLicense	—
<code>programResetType</code>	ProgramResetType	0, 1, 2, 3
<code>isSmsProviderSupportGloballySend</code>	boolean	—

ProgramUtilization nested object

FIELD	TYPE	DESCRIPTION
<code>id</code>	string (uuid)	—
<code>startDate</code>	string (date-time)	—
<code>lastResetDate</code>	string (date-time)	—
<code>users</code>	integer	—
<code>templates</code>	integer	—
<code>documentsUsage</code>	integer	—
<code>documentsLimit</code>	integer	—
<code>documentsSentNotifyCount</code>	integer	—
<code>programResetType</code>	ProgramResetType	0, 1, 2, 3
<code>sms</code>	integer	—
<code>smsSentNotifyCount</code>	integer	—
<code>visualIdentifications</code>	integer	—
<code>visualIdentificationUsedNotifyCount</code>	integer	—
<code>videoConference</code>	integer	—
<code>videoConferenceUsedNotifyCount</code>	integer	—
<code>whatsApp</code>	integer	—
<code>whatsAppSentNotifyCount</code>	integer	—
<code>expired</code>	string (date-time)	—

RadioField nested object

FIELD	TYPE	DESCRIPTION
<code>fieldName</code>	string	—
<code>description</code>	string	—
<code>x</code>	number	—
<code>y</code>	number	—
<code>width</code>	number	—
<code>height</code>	number	—
<code>mandatory</code>	boolean	—
<code>page</code>	integer	—
<code>recipientRole</code>	string	—
<code>value</code>	string	—

RadioGroupField nested object

FIELD	TYPE	DESCRIPTION
fieldName	string	—
radioFields	RadioField[]	—
selectedRadioName	string	—

ReadSignersFromFileResponseDTO response body

FIELD	TYPE	DESCRIPTION
signers	BaseSigner[]	—

RefreshResponseDTO response body

FIELD	TYPE	DESCRIPTION
token	string	—

RenewPasswordByDevAdminUserDTO request body

FIELD	TYPE	DESCRIPTION
userId	string (uuid)	—
email	string	—
newPassword	string	—

RenewPasswordDTO request body

FIELD	TYPE	DESCRIPTION
renewPasswordToken	string	—
newPassword	string	—
oldPassword	string	—

ReplaceSignerDTO request body

FIELD	TYPE	DESCRIPTION
newSignerName	string	—
newSignerMeans	string	—

ReplaceSignerWithDetailsDTO request body

FIELD	TYPE	DESCRIPTION
newSignerName	string	—
newSignerMeans	string	—
newNotes	<u>Notes</u>	—
newOtpMode	<u>OtpMode</u>	None = 0, CodeRequired = 1, PasswordRequired = 2, CodeAndPasswordRequired = 3
newOtpIdentification	string	—
newAuthenticationMode	<u>AuthMode</u>	None = 0, IDP = 1, VisualIDP = 2

ReportSettings response body

FIELD	TYPE	DESCRIPTION
historicalReportsEnabled	boolean	—
maxNonHistoricalRangeDays	integer	—

Seal nested object

FIELD	TYPE	DESCRIPTION
id	string (uuid)	—
name	string	—
base64Image	string	—

SealDTO nested object

FIELD	TYPE	DESCRIPTION
name	string	—
base64Image	string	—

SelfSignCountResponseDTO response body

FIELD	TYPE	DESCRIPTION
documentCollectionId	string (uuid)	—
documentId	string (uuid)	—
name	string	—
pagesCount	integer	—

SelfSignUpdateDocumentResult response body

FIELD	TYPE	DESCRIPTION
token	string (uuid)	—
redirectUrl	string	—

SenderLiveSigningLinkDTO response body

FIELD	TYPE	DESCRIPTION
link	string	—

ShareDTO request body

FIELD	TYPE	DESCRIPTION
documentCollectionId	string (uuid)	—
signerName	string	—
signerMeans	string	—

SharedAppendixDTO nested object

FIELD	TYPE	DESCRIPTION
signerIndexes	integer[]	—
appendix	Appendix	—

SignatureField nested object

FIELD	TYPE	DESCRIPTION
fieldName	string	—
description	string	—
x	number	—
y	number	—
width	number	—
height	number	—
mandatory	boolean	—
page	integer	—
recipientRole	string	—
image	string	—
signingType	SignatureFieldType	Graphic = 1, SmartCard = 2, Server = 3
signatureKind	SignatureFieldKind	0, 1

SignaturesImagesDTO request & response

FIELD	TYPE	DESCRIPTION
documentCollectionId	string (uuid)	—
signaturesImages	string[]	—

Signer1Configuration nested object

FIELD	TYPE	DESCRIPTION
endpoint	string	—
user	string	—
password	string	—

Signer1Credential nested object

FIELD	TYPE	DESCRIPTION
password	string	—
certificateId	string	—
shouldUseADDetails	boolean	—
signerToken	string	—

Signer1FileSingingDTO request body

FIELD	TYPE	DESCRIPTION
base64File	string	—
singingFileType	SigningFileType	0, 1, 2, 3, 4
signer1Credential	Signer1Credential	—
fileName	string	—

SignerAttachmentDTO nested object

FIELD	TYPE	DESCRIPTION
name	string	—
isMandatory	boolean	—

SignerAttachmentResponseDTO nested object

FIELD	TYPE	DESCRIPTION
name	string	—
base64File	string	—

SignerAuthentication request body

FIELD	TYPE	DESCRIPTION
signer1Credential	Signer1Credential	—
otpDetails	OtpDetails	—
signer1Configuration	Signer1Configuration	—
authenticationMode	AuthMode	None = 0, IDP = 1, VisualIDP = 2

SignerAuthenticationResponseDTO nested object

FIELD	TYPE	DESCRIPTION
otpDetails	OtpDetails	—
authenticationMode	AuthMode	None = 0, IDP = 1, VisualIDP = 2

SignerDTO nested object

FIELD	TYPE	DESCRIPTION
contactId	string (uuid)	—
sendingMethod	SendingMethod	SMS = 1, Email = 2, Tablet = 3, WhatsApp = 4
contactMeans	string	—
phoneExtension	string	—
contactName	string	—
signerAttachments	SignerAttachmentDTO[]	—
signerFields	SignerFieldDTO[]	—
linkExpirationInHours	integer	—
senderNote	string	—
emailNote	string	Per-recipient note rendered in a dedicated section of the invitation email, up to 250 characters. Reaches only recipients whose SendingMethod is Email. Distinct from SenderNote, which is shown inside the signing session after the recipient opens the link.
senderAppendices	Appendix[]	—
otpIdentification	string	—
otpMode	OtpMode	None = 0, CodeRequired = 1, PasswordRequired = 2, CodeAndPasswordRequired = 3
authenticationMode	AuthMode	None = 0, IDP = 1, VisualIDP = 2

SignerFieldDTO nested object

FIELD	TYPE	DESCRIPTION
templateId	string (uuid)	—
fieldName	string	—
fieldValue	string	—

SignerLink response body

FIELD	TYPE	DESCRIPTION
signerId	string (uuid)	—
link	string	—

SignerResponseDTO nested object

FIELD	TYPE	DESCRIPTION
id	string (uuid)	—
name	string	—
status	SignerStatus	Sent = 1, Viewed = 2, Signed = 3, Rejected = 4
timeSent	string (date-time)	—
timeViewed	string (date-time)	—
timeSigned	string (date-time)	—
timeRejected	string (date-time)	—
sendingMethod	SendingMethod	SMS = 1, Email = 2, Tablet = 3, WhatsApp = 4
signerNote	string	—
userNote	string	—
emailNote	string	—
attachments	SignerAttachmentResponseDTO[]	—
contact	ContactResponseDTO	—
signerAuthentication	SignerAuthenticationResponseDTO	—

SignersForDistributionMechanismDTO request body

FIELD	TYPE	DESCRIPTION
base64File	string	—

SingleLinkAdditionalResource nested object

FIELD	TYPE	DESCRIPTION
templateId	string (uuid)	—
type	SingleLinkAdditionalResourceType	1
data	string	—
isMandatory	boolean	—

SmartCardSignFlowFieldsDTO nested object

FIELD	TYPE	DESCRIPTION
documentId	string (uuid)	—
fieldName	string	—
image	string	—

SmartCardSigningFlowDTO request body

FIELD	TYPE	DESCRIPTION
token	string	—
fields	SmartCardSignFlowFieldsDTO[]	—

Tablet nested object

FIELD	TYPE	DESCRIPTION
id	string (uuid)	—
name	string	—
companyId	string (uuid)	—

TabletsConfigurationDTO response body

FIELD	TYPE	DESCRIPTION
tablets	Tablet[]	—

TemplateCountResponseDTO response body

FIELD	TYPE	DESCRIPTION
templateId	string (uuid)	—
templateName	string	—
pagesCount	integer	—

TemplatePageResponseDTO response body

FIELD	TYPE	DESCRIPTION
templateId	string (uuid)	—
name	string	—
pdfFields	PDFFields	—
pageImage	string	—
ocrString	string	—
pageWidth	number	—
pageHeight	number	—
pageNumber	integer	—

TemplatePagesRangeResponseDTO response body

FIELD	TYPE	DESCRIPTION
templatePages	TemplatePageResponseDTO[]	—

TemplateResponseDTO nested object

FIELD	TYPE	DESCRIPTION
templateId	string (uuid)	—
name	string	—
userId	string (uuid)	—
userName	string	—
pagesCount	integer	—
isFavorite	boolean	—
timeCreated	string (date-time)	—
singleLinkUrl	string	—

TemplateSingleLinkDTO request & response

FIELD	TYPE	DESCRIPTION
templateId	string (uuid)	—
singleLinkAdditionalResources	SingleLinkAdditionalResource[]	—

TextField nested object

FIELD	TYPE	DESCRIPTION
fieldName	string	—
description	string	—
x	number	—
y	number	—
width	number	—
height	number	—
mandatory	boolean	—
page	integer	—
recipientRole	string	—
textFieldType	TextFieldType	Text = 1, Date = 2, Number = 3, Phone = 4, Email = 5, Custom = 6, Time = 7
customerRegex	string	—
value	string	—
isHidden	boolean	—

TokensDTO request body

FIELD	TYPE	DESCRIPTION
jwtToken	string	—
refreshToken	string	—
authToken	string	—

UIViewLicense nested object

FIELD	TYPE	DESCRIPTION
shouldShowSelfSign	boolean	—
shouldShowGroupSign	boolean	—
shouldShowLiveMode	boolean	—
shouldShowContacts	boolean	—
shouldShowTemplates	boolean	—
shouldShowDocuments	boolean	—
shouldShowProfile	boolean	—
shouldShowUploadAndsign	boolean	—
shouldShowAddNewTemplate	boolean	—
shouldShowEditTextField	boolean	—
shouldShowEditSignatureField	boolean	—
shouldShowEditEmailField	boolean	—
shouldShowEditPhoneField	boolean	—
shouldShowEditDateField	boolean	—
shouldShowEditNumberField	boolean	—
shouldShowEditListField	boolean	—
shouldShowEditCheckboxField	boolean	—
shouldShowEditRadioField	boolean	—
shouldShowDistribution	boolean	—
shouldShowMultilineText	boolean	—

UpdateSelfSignDocumentDTO request body

FIELD	TYPE	DESCRIPTION
documentCollectionId	string (uuid)	—
documentId	string (uuid)	—
name	string	—
fields	PDFFields	—
operation	DocumentOperation	Save = 1, Decline = 2, Close = 3
signerAuthentication	SignerAuthentication	—
useForAllFields	boolean	—

UpdateTemplateDTO request body

FIELD	TYPE	DESCRIPTION
name	string	—
fields	PDFFields	—

UpdateUserDTO request body

FIELD	TYPE	DESCRIPTION
name	string	—
email	string	—
userConfiguration	UserConfigurationDTO	—
username	string	—

UpdateUserPhoneDTO request body

FIELD	TYPE	DESCRIPTION
phoneNumber	string	—
code	string	—

UsageDataReportDTO nested object

FIELD	TYPE	DESCRIPTION
groupName	string	—
pendingDocumentsCount	integer	—
signedDocumentsCount	integer	—
declinedDocumentsCount	integer	—
canceledDocumentsCount	integer	—
distributionDocumentsCount	integer	—

User nested object

FIELD	TYPE	DESCRIPTION
id	string (uuid)	—
companyId	string (uuid)	—
name	string	—
companyName	string	—
groupName	string	—
email	string	—
phone	string	—
password	string	—
type	UserType	Basic = 1, Editor = 2, CompanyAdmin = 3
status	UserStatus	1, 2, 3, 4
creationSource	CreationSource	0, 1, 2
creationTime	string (date-time)	—
groupId	string (uuid)	—
programUtilizationId	string (uuid)	—
programUtilization	ProgramUtilization	—
userConfiguration	UserConfiguration	—
userTokens	UserTokens	—
profileProgram	ProfileProgram	—
companyLogo	string	—
lastSeen	string (date-time)	—
username	string	—
passwordSetupTime	string (date-time)	—
isSsoUser	boolean	—
additionalGroupsMapper	AdditionalGroupMapper[]	—
userPeriodicReports	UserPeriodicReport[]	—
managementPeriodicReports	ManagementPeriodicReport[]	—

UserConfiguration nested object

FIELD	TYPE	DESCRIPTION
signatureColor	string	—
shouldSendSignedDocument	boolean	—
shouldNotifyWhileSignerSigned	boolean	—
shouldNotifyWhileSignerViewed	boolean	—
shouldDisplayNameInSignature	boolean	—
shouldNotifySignReminder	boolean	—
signReminderFrequencyInDays	integer	—
language	Language	en = 1, he = 2
dateFormat	DateFormat	1, 2, 3, 4

UserConfigurationDTO nested object

FIELD	TYPE	DESCRIPTION
signatureColor	string	—
shouldSendSignedDocument	boolean	—
shouldNotifyWhileSignerSigned	boolean	—
shouldNotifyWhileSignerViewed	boolean	—
shouldDisplayNameInSignature	boolean	—
shouldNotifySignReminder	boolean	—
signReminderFrequencyInDays	integer	—
language	Language	en = 1, he = 2
dateFormat	DateFormat	1, 2, 3, 4

UserGroupsDTO response body

FIELD	TYPE	DESCRIPTION
groups	GroupResponseAdminDTO[]	—

UserPeriodicReport nested object

FIELD	TYPE	DESCRIPTION
id	string (uuid)	—
userId	string (uuid)	—
reportType	ReportType	0
lastTimeSent	string (date-time)	—
reportFrequency	ReportFrequency	0, 1, 2, 3
user	User	—

UserResponseAdminDTO nested object

FIELD	TYPE	DESCRIPTION
id	string (uuid)	—
name	string	—
email	string	—
username	string	—
type	UserType	Basic = 1, Editor = 2, CompanyAdmin = 3
groupId	string (uuid)	—
additionalGroupsIds	string (uuid)[]	—

UserResponseDTO nested object

FIELD	TYPE	DESCRIPTION
id	string (uuid)	—
companyId	string (uuid)	—
groupId	string (uuid)	—
name	string	—
email	string	—
type	UserType	Basic = 1, Editor = 2, CompanyAdmin = 3
creationTime	string (date-time)	—
userConfiguration	UserConfiguration	—
programUtilizationId	string (uuid)	—
program	ProgramResponseDTO	—
companyLogo	string	—
companySigner1Details	CompanySigner1DetailsResponseDTO	—
lastSeen	string (date-time)	—
username	string	—

UserSigningLinkResponseDTO nested object

FIELD	TYPE	DESCRIPTION
documentCollectionId	string (uuid)	—
name	string	—
mode	DocumentMode	OrderedGroupSign = 1, GroupSign = 2, OnLine = 3, SelfSign = 100
creationTime	string (date-time)	—
signingLink	string	—
documentsIds	string (uuid)[]	—
status	DocumentStatus	Created = 1, Sent = 2, Viewed = 3, Signed = 4, Declined = 5, SendingFailed = 6, Deleted = 7, Canceled = 8
sender	string	—

UserSigningLinksResponseDTO response body

FIELD	TYPE	DESCRIPTION
documentCollections	<u>UserSigningLinkResponseDTO[]</u>	—

UserTokens nested object

FIELD	TYPE	DESCRIPTION
userId	string (uuid)	—
jwtToken	string	—
refreshToken	string	—
lastLoginGroupId	string (uuid)	—
refreshTokenExpiredTime	string (date-time)	—
resetPasswordToken	string	—
authToken	string	—
otpAttempts	integer	—

UserTokensResponseDTO response body

FIELD	TYPE	DESCRIPTION
token	string	—
refreshToken	string	—
authToken	string	—
requiresEulaAcceptance	boolean	Indicates if EULA acceptance is required (UI requests only)
eulaVersion	string	Current EULA version that needs to be accepted (UI requests only)

VideoConferenceResult response body

FIELD	TYPE	DESCRIPTION
conferenceHostUrl	string	—

VideoConferenceUser nested object

FIELD	TYPE	DESCRIPTION
means	string	—
fullName	string	—
sendingMethod	<u>SendingMethod</u>	SMS = 1, Email = 2, Tablet = 3, WhatsApp = 4
phoneExtension	string	—

WeSignLicense nested object

FIELD	TYPE	DESCRIPTION
expirationTime	string (date-time)	—
licenseCounters	LicenseCounters	—
uiViewLicense	UIViewLicense	—

SECTION 12

Troubleshooting

The failures integrators actually hit, and what each one means.

WeSign answers a rejected request with `400` and a `GeneralError` body. The list below maps the situations that come up most often during an integration to their cause.

SYMPTOM	CAUSE AND REMEDY
Upload rejected, or the template has no pages	The base64 payload was sent without its <code>data:<mime>;base64,</code> prefix, or the prefix does not match the real file type. Rebuild the string exactly as shown in section 4.
A large scanned PDF fails to upload	Some scanners write pages with an oversized media box. Re-generate the PDF at a normal page size (A4 at 144 dpi is a safe target) and upload again.
Field rejected as out of range	<code>x</code> , <code>y</code> , <code>width</code> and <code>height</code> are fractions of the page between 0 and 1, not points, and each field type has its own bounds — see the table in section 5. A width of <code>120</code> is always wrong.
"Signer must have contact details"	Every recipient needs either <code>contactId</code> + <code>sendingMethod</code> , or <code>contactMeans</code> + <code>contactName</code> . One of the two, never neither.
OTP settings refused	<code>otpMode</code> is not <code>0</code> but <code>otpIdentification</code> is missing. The two always travel together.
The e-mail or SMS never arrives	Check <code>notifications.shouldSend</code> — if it is <code>false</code> WeSign deliberately sends nothing and expects you to deliver the link. Then check the recipient status: <code>SendingFailed</code> (6) means the address or number was refused by the carrier. Re-send with <code>GET /v3/DocumentCollections/{id}/signers/{signerId}/method/{sendingMethod}</code> .
Only one signing link came back	Expected in <code>OrderedGroupSign</code> (1): the next recipient is invited only after the previous one signs. Use <code>GroupSign</code> (2) if everyone should sign at once.
The recipient's link no longer opens	It passed <code>linkExpirationInHours</code> , or the envelope was cancelled. Reactivate with <code>GET /v3/DocumentCollections/{collectionId}/reactivate</code> , or issue a fresh link with the re-send endpoint.
Every call suddenly returns 401 with an empty body	The access token expired or was rejected. Refresh it with <code>POST /v3/Users/refresh</code> ; if the refresh is refused as well, sign in again. A <code>401</code> never carries an error body — do not try to parse one.
An object "does not exist" although the id is right	It belongs to another company, another group, or another environment. Confirm which group the token is scoped to with <code>GET /v3/Users/groups</code> and switch with <code>POST /v3/Users/SwitchGroup/{groupId}</code> .
Sending stops part way through a batch	The monthly quota ran out. Read <code>remainingDocumentsForMonth</code> , <code>remainingSMS</code> , <code>remainingWhatsApp</code> and <code>remainingVisualIdentifications</code> from <code>GET /v3/License?readUsage=true</code> before a large run.
The redirect after signing does nothing	The field is <code>rediretUrl</code> , not <code>redirectUrl</code> . An unknown field is ignored silently.
Excel recipients are not detected	<code>POST /v3/Distribution/signers</code> expects first name, last name and phone/e-mail in columns 1–3, and a field name in every further column header. All worksheets are read; empty and hidden sheets are skipped.

SYMPTOM	CAUSE AND REMEDY
Paging returns the same rows twice	The base index differs per endpoint – some count from 1, some from 0. Check the endpoint's own note in the reference.
Hebrew text comes back garbled	Something in the chain is not UTF-8. Set the charset explicitly on both the request and the response handling.

Result codes

The `status` field of an error body is a WeSign result code, not the HTTP status. The codes below are the set published for WeSign v3. New codes are added as the service grows, so treat a code you do not recognise as a generic failure and rely on the `errors` messages and the HTTP status.

CODE	MESSAGE	CODE	MESSAGE
0	Something went wrong. Please try again later.	38	Invalid signer id
1	Invalid document id	39	Cannot share unsigned document
2	Invalid credential	40	Invalid file type
3	Invalid token	41	Not all fields exist in documents
4	Invalid email address	42	Not all fields belong to signer
5	Email already exists	43	Not all mandatory fields filled in
6	User not exist, please register	44	Program already exist
7	Activation is required	45	Program not exist
8	User is not group creator	46	Company already exists
9	User belong to another group	47	Company not exist
10	Program utilization get to max limit	48	There are related companies to this program
11	Cannot delete group creator	49	Invalid user type
12	Invalid template id	50	Invalid expired time
13	Invalid page number	51	Email belong to other company
14	Invalid positive offset	52	There are users in group
15	Invalid limit	53	Failed init license request
16	Invalid Date Time format	55	Invalid license
17	User is not editor / companyAdmin	56	Documents exceed license limit
18	Template is in use, there is documents related to it	57	Sms exceed license limit
19	Template not belong to user group	58	Users exceed license limit
20	User program expired	59	Templates exceed license limit
21	Contact not created by user	60	Invalid base64string format
22	Field name not exist	61	Forbidden to create free trial user
23	Invalid contact id	62	Cannot parse *.csv file to contacts
24	Contact with same means already exists	63	Sign operation failed
25	Duplicate seals ids	64	ActiveDirectory Config Not Exist
26	Template with same name already exist	65	Cant parse xml file to pdf meta data

CODE	MESSAGE	CODE	MESSAGE
27	Invalid DocumentCollection id	66	Amount of fields in a pdf is not the same as the amount of fields in a meta data
28	Contact not belong to user group	67	Document not belong to the user group
29	Document not belong to document collection	68	Cant parse interface to xml
30	Signer method not fit to contact means	69	Cannot create template from another user
31	Group with same name already exist in company	70	Cannot download attachments from unsigned document
32	Invalid GroupId	71	Signer not uploaded attachments or not required to
33	Cannot delete your own user	72	Can't resend in the wrong order
34	Operation not allow by free trial user	73	Invalid Phone
35	Invalid refresh token	74	Invalid ReCaptcha token
36	Cannot download unsigned document	75	Cannot create signing link for signer that already signed or declined
37	Cannot resend signed document		

Code 54 is not published in this range.

Before you contact support

Have these four things ready and most cases are solved in one exchange:

- The `traceId` from the error body, when it is not empty, and the `errors` object.
- The exact request – method, URL and body – with credentials removed.
- The **environment** (Production: <https://wse.comsigntrust.com/api>) and the time of the call, with the time zone.
- The **document collection id** or template id involved, when there is one.

SECTION 13

Endpoint index

Every endpoint of WeSign API v3 in alphabetical order, with the page it is documented on.

Use this list to go from a path you already know to its reference page. The paths are relative to <https://wse.comsigntrust.com/api> ; each page number links straight to the full entry in section 10.

METHOD	PATH	OPERATION	PAGE
PUT	/v3/Admins/dev/password	Update password by dev admin user	46
GET	/v3/Admins/groups	Get all groups in my company	41
POST	/v3/Admins/groups	Create Group	41
DELETE	/v3/Admins/groups/{id}	Delete group	42
PUT	/v3/Admins/groups/{id}	Update group	42
GET	/v3/Admins/users	Get all users in my company	44
POST	/v3/Admins/users	Create user	43
DELETE	/v3/Admins/users/{id}	Delete user	45
PUT	/v3/Admins/users/{id}	Update user	45
GET	/v3/Configuration	Read init configuration	29
GET	/v3/Configuration/tablets	Read tablets list	29
GET	/v3/Contacts	Search contacts	49
POST	/v3/Contacts	Create new contact	48
POST	/v3/Contacts/bulk	Create bulk of contacts from XLSX file	49
PUT	/v3/Contacts/deletebatch	Delete a batch of contacts by their IDs	50
POST	/v3/Contacts/Group	Create a new contact group	56
DELETE	/v3/Contacts/Group/{id}	Delete a contact group by ID	54
GET	/v3/Contacts/Group/{id}	Get a specific contact group by ID	54
PUT	/v3/Contacts/Group/{id}	Update a contact group by ID	55
GET	/v3/Contacts/Groups	Get all contact groups	53
PUT	/v3/Contacts/signatures	Update saved signature images for a specific document collection	53
DELETE	/v3/Contacts/signatures/{docCollectionId}	Delete a saved signature image for a specific document collection	52
GET	/v3/Contacts/signatures/{docCollectionId}	Get saved signature images for a specific document collection	52
DELETE	/v3/Contacts/{id}	Delete contact	51
GET	/v3/Contacts/{id}	Get contact by ID	50
PUT	/v3/Contacts/{id}	Update contact	51
GET	/v3/Dashboard/view	Get dashboard view data	110
GET	/v3/Distribution	Get all distribution document collections by search criteria	92
POST	/v3/Distribution	Send new document collection for each signer and set all fields in document...	91

METHOD	PATH	OPERATION	PAGE
GET	/v3/Distribution/download/{id}	Download all signed documents of distribution id	95
GET	/v3/Distribution/resend/{id}	Resend all unsigned documents of distribution id to there signers	94
GET	/v3/Distribution/resend/{id}/status/{status}	Resend documents in specific status for distribution id	95
POST	/v3/Distribution/signers	Extract signers from excel file	98
DELETE	/v3/Distribution/{id}	Delete all documents of distribution id	94
GET	/v3/Distribution/{id}	Get all documents of distribution id	93
GET	/v3/DocumentCollections	Get all document collections by search criteria	68
POST	/v3/DocumentCollections	Create document collection from template/s	69
PUT	/v3/DocumentCollections/deletebatch	Delete multiple document collections by batch	79
POST	/v3/DocumentCollections/downloadbatch	Download multiple document collections as batch	73
GET	/v3/DocumentCollections/export	Export documents collections details to CSV file	83
GET	/v3/DocumentCollections/exportDistribution	Export distribution documents to CSV file	83
GET	/v3/DocumentCollections/info/{id}	Get info for document collection	75
POST	/v3/DocumentCollections/share	Share document collection to contact	82
POST	/v3/DocumentCollections/simple	Document for "Lite" production version	77
GET	/v3/DocumentCollections/{collectionId}/reactivate	Reactivate document collection to signer	81
DELETE	/v3/DocumentCollections/{id}	Delete document collection by id	73
GET	/v3/DocumentCollections/{id}	Download document collection	72
GET	/v3/DocumentCollections/{id}/audit/{offset}	Download document collection trace as separate file	76
PUT	/v3/DocumentCollections/{id}/cancel	Cancel document collection by id	80
GET	/v3/DocumentCollections/{id}/DocumentCollectionLinks	Get signing links for document	81
GET	/v3/DocumentCollections/{id}/documents/{documentId}	Get information about pages range in document	79
GET	/v3/DocumentCollections/{id}/documents/{documentId}/pages	Get pages count by document id	78
GET	/v3/DocumentCollections/{id}/documents/{documentId}/pages/{page}	Get information about specific page in document	78
GET	/v3/DocumentCollections/{id}/ExtraInfo/json	Download document collection with extra information as JSON	74
GET	/v3/DocumentCollections/{id}/fields	Export document collection fields to XML file	84
GET	/v3/DocumentCollections/{id}/fields/CsvXml	Export document collection fields to XML file and to csv file in a zip file	85
GET	/v3/DocumentCollections/{id}/fields/json	Export document collection fields data as JSON	84
GET	/v3/DocumentCollections/{id}/json	Download document collection as JSON	74
GET	/v3/DocumentCollections/{id}/senderLink/{signerId}	Get sender signing link for live mode document collection	76
PUT	/v3/DocumentCollections/{id}/serversign	Extra server document collection signing by id	86
GET	/v3/DocumentCollections/{id}/signer/{signerId}	Download signer attachments of specified document collection id	75

METHOD	PATH	OPERATION	PAGE
PUT	/v3/DocumentCollections/{id}/signer/{signerId}/replace	Replace old signer with new signer	85
GET	/v3/DocumentCollections/{id}/signers/{signerId}/method/{sendingMethod}	Resend document collection link to signer	80
GET	/v3/License		111
GET	/v3/Links	Get current user links to sign on all documents that not sign already	104
GET	/v3/Links/Template/{id}	Get single link information for template	106
POST	/v3/Links/template/{id}	Create or update single link information for template	106
POST	/v3/Links/videoConference	Create video conference for document signing	105
DELETE	/v3/Reports/FrequencyReports	Delete all scheduled frequency reports	109
GET	/v3/Reports/FrequencyReports	Get all scheduled frequency reports	108
POST	/v3/Reports/FrequencyReports	Create scheduled frequency reports	108
GET	/v3/Reports/FrequencyReports/Download	Download generated frequency report file	109
GET	/v3/Reports/Settings	Report screen settings	108
GET	/v3/Reports/UsageData	Get user usage data reports	107
POST	/v3/SelfSign	Create SelfSign document	96
PUT	/v3/SelfSign	Update SelfSign document	97
POST	/v3/SelfSign/CheckidentityFlowEIDASSign	Check identity flow for EIDAS signing	103
POST	/v3/SelfSign/CreateSmartCardSigningFlow	Create smart card signing flow session	101
GET	/v3/SelfSign/download/smartcard	Download smartCard desktop client installer	100
POST	/v3/SelfSign/sign	Sign file using Signer1 service	100
POST	/v3/SelfSign/sign/verify	Verify Signer1 credentials	102
DELETE	/v3/SelfSign/{id}	Delete SelfSign document	99
PUT	/v3/Signers/{id}/signer/{signerId}/replace	Replace old signer with new signer in document collection	88
GET	/v3/Templates	Get all templates by search criteria	59
POST	/v3/Templates	Create new template	58
PUT	/v3/Templates/deletebatch	Delete multiple templates by batch	66
POST	/v3/Templates/merge	Merge multiple templates into a single template	66
DELETE	/v3/Templates/{id}	Delete template	62
POST	/v3/Templates/{id}	Duplicate existing template	62
PUT	/v3/Templates/{id}	Update template fields	60
GET	/v3/Templates/{id}/download	Download template	65
DELETE	/v3/Templates/{id}/favorite	Remove a template from the calling user's favorites	65
POST	/v3/Templates/{id}/favorite	Mark a template as favorite for the calling user	64
GET	/v3/Templates/{id}/pages	Get pages count by template id	63
GET	/v3/Templates/{id}/pages/range	Get page details by template id, first page is '1'.	64
GET	/v3/Templates/{id}/pages/{page}	Get page details by template id, first page is '1'.	63
GET	/v3/Users	Get your own user details	32
POST	/v3/Users	User Sign up	30
PUT	/v3/Users	Update User	31
POST	/v3/Users/activation	Resend activation link	35

METHOD	PATH	OPERATION	PAGE
PUT	/v3/Users/activation		35
POST	/v3/Users/change	Change password	38
POST	/v3/Users/changepaymentrule	Get URL for changing payment rule	38
DELETE	/v3/Users/DeletePhone	Clears the caller's own phone number. No OTP – ownership only needs...	40
POST	/v3/Users/externalLogin	External login for login using AD or SAML login	36
GET	/v3/Users/groups	Get all groups that the current user belongs to	32
POST	/v3/Users/login	User login case the company force OTP will return token for OTP	34
GET	/v3/Users/Logout	Logout current user	35
POST	/v3/Users/password	Reset password, send reset password mail to user	36
PUT	/v3/Users/password	Update password	37
POST	/v3/Users/refresh	Refresh token	37
POST	/v3/Users/resend0tp	Resend OTP (One-Time Password) code to user	33
POST	/v3/Users/SwitchGroup/{groupId}	Switch user's active group context	32
POST	/v3/Users/unsubscribeuser	Unsubscribe current user from notifications	38
POST	/v3/Users/UpdatePhone	Initiate phone number update process	39
POST	/v3/Users/UpdatePhoneValidate0tp	Validate OTP code and complete phone number update	39
POST	/v3/Users/validateExpiredPasswordFlow	Update password for expired password flow	34
POST	/v3/Users/validate0tpflow	Validate OTP code and complete login flow	33

SECTION 14

Support and resources

Where to find the live reference, the product documentation and a human.

Interactive API reference

The same definition this document is generated from, with an *Authorize* button and a *Try it out* button on every endpoint.

<https://wse.comsigntrust.com/api/swagger/index.html>

OpenAPI 3.0 definition

Machine readable. Feed it to your code generator to produce a typed client in any language.

<https://wse.comsigntrust.com/api/swagger/v3/swagger.json>

WeSign web application

Useful during development: build a template by hand in the UI, then read it back through the API to see exactly what the payload should look like.

<https://wse.comsigntrust.com>

Technical support

Integration questions, credentials for the test environment, and anything this document does not answer.

pro-support@comda.co.il

Practical advice for a first integration

- **Start in the user interface.** Create one template and send one document by hand. Then reproduce it through the API. Everything you do in the UI is visible through the API, which makes the UI the best debugging tool you have.
- **Keep the base URL in configuration.** Moving from test to production should be a setting, not a deployment.
- **Store the ids.** Keep `documentCollectionId` and the template ids next to your own records; almost every follow-up call needs them.
- **Prefer callbacks to polling,** and re-read the status through the API when a callback arrives.
- **Log the `traceId`** of every failed call.
- **Check the quota** before a mass distribution.

Version of this document

Product release	WeSign 4.0.0
API version	v3
Specification	OpenAPI 3.0.1 – WeSign API , 118 operations across 93 paths and 177 data models
Environment described	Production – https://wse.comsigntrust.com/api
OpenAPI definition	https://wse.comsigntrust.com/api/swagger/v3/swagger.json
Document revision	4.0.0 / rev. A, 8 September 2026

Keeping up with changes

The API is extended additively: new fields and new endpoints appear, existing ones keep their meaning. Build clients that ignore unknown response fields, and re-read the OpenAPI definition after a WeSign release to see what is new.

© 2026 ComSign / Comda Ltd. All rights reserved. WeSign is a trademark of ComSign / Comda Ltd. This document is provided for the use of WeSign customers and their integration partners. The interface described here is the one published by release 4.0.0; the live OpenAPI definition remains the authoritative reference.

END OF DOCUMENT