Compliance Intelligence

REST API Developer Guide

February 2026

IMPORTANT NOTE - TO BE READ BEFORE CONSULTING THE DOCUMENT:

This API Developers Guide is the exclusive property of Enhesa NV, located at Louizalaan 287, 1050 Brussels, Belgium and registered under Company number 0474.389.386 (RPR Brussels), VAT BE0474.389.386 and is protected under the applicable intellectual property rights and trade secrets legislation. Enhesa NV retains all proprietary rights, including without limitation any intellectual property rights and trade secrets rights to the content.

No reproduction, modification, distribution, public display, public communication or other exploitation of this Platform Integration Guide in any form or by any means, electronic or mechanical, is allowed without prior written consent of Enhesa NV. This Platform Integration Guide is strictly confidential and shall be used by recipient solely for internal purposes and on an as-needed basis. If you are not the intended recipient of this Platform Integration Guide, you are explicitly requested to destroy this copy.

Disclaimer: Enhesa NV can never be held liable for direct or indirect damage arising from the use of this Platform Integration Guide. By use of this API Developers Guide, you waive any claim and any recourse against Enhesa NV concerning the use of information made available herein. This Platform Integration Guide is not provided with any guaranty, warranty, or representation as to quality or suitability for any particular purpose and does not constitute legal counsel or advice.

Contents

Overview

This document provides Enhesa business partners and customers with documentation on how to use the Compliance Intelligence REST API for integration with external platforms or systems.

The Enhesa REST API replaces the legacy SOAP XML feed that many platform partners currently use. Please note that the SOAP XML Service will be sunset in June 2026.

The Compliance Intelligence API integration enables partners to sync facility-specific, applicable compliance content into their own platform. In the Compliance Intelligence Dashboard, the Enhesa tenant links its contracted content to each facility and configures applicability (which regulations and requirements apply for that facility). The partner integration then authenticates via OAuth (client credentials) and calls the CI API using the tenant and facility identifiers to retrieve only the applicable content per facility. Partners can then run a daily incremental sync to pull a delta of what has changed since the last run, including both content updates (major and minor revisions) and changes driven by updated client applicability or compliance answers. The resulting content is stored in the partner system and used to power downstream workflows such as audits and self-assessments, legal register creation, and task or corrective action management (CAPAs). OAuth replaces the legacy basic authentication approach, providing stronger security and better control over access via scoped tokens

How is this difference from the current XML delivery

- ***Scope of data delivered***
    - XML (today): Delivers broad Enhesa data sets by jurisdiction, including objects like applicability, requirements, and regulations, regardless of whether a given facility actually needs all of it.
    - API (new): Delivers only applicable requirements by facility, based on the Enhesa tenant's configured applicability in the Compliance Intelligence Dashboard.

- ***File size and performance***
    - XML (today): Large files that can take minutes to download and parse.
    - API (new): Smaller, targeted responses designed for faster retrieval and easier processing (pull what you need, per facility).

- ***Business logic ownership***

- XML (today): Partners must re-create Enhesa business logic in their own platform to interpret and apply the data (notably around applicability and stringency in the US), which increases complexity and long-term maintenance burden.

- API (new): Removes much of that burden. Applicability is handled end-to-end by Enhesa within the Dashboard, and the partner receives the already-filtered applicable results.

- ***Configuration model***
    - XML (today): Jurisdiction-based delivery often forces partners to infer how content should map to facilities and operations.
    - API (new): Facility-based configuration is explicit: Enhesa clients link contracted content to facilities and set applicability per facility in the Dashboard.

- ***Integration simplicity and maintainability***
    - XML (today): More complex integrations due to heavy payloads and duplicated logic, making builds slower and updates harder to maintain.
    - API (new): Cleaner interface: partners focus on retrieving and storing applicable content and using it in workflows (audits/self-assessments, legal registers, CAPAs), rather than rebuilding compliance decisioning logic.

What you can do via API vs what requires the Dashboard

What you can do via the API (partner integration)

- Retrieve applicable content by facility: Pull the facility-specific set of requirements (and related data) that are applicable based on the Enhesa tenant's configuration.
- Support daily updates (delta sync): Retrieve incremental changes since the last sync, including:
    - Content updates (major/minor revisions)
    - Changes caused by updated applicability or compliance answers

- Store and use the content in your platform: Enable workflows such as audits and self-assessments, legal register creation, and task/CAPA management using the data retrieved via the API.

What requires the Compliance Intelligence Dashboard (Enhesa tenant configuration and governance)

- Version history: Viewing full requirement/regulation version history is available in the Dashboard.
- Change notes: Viewing detailed change notes or editorial commentary is available in the Dashboard.
- Applicability configuration: Linking contracted content to facilities and configuring applicability is performed in the Dashboard.
- Compliance answers and applicability inputs: Changes to compliance answers (where applicable) are managed in the Dashboard and can drive updates in what the API returns.

Important limitation

- The API returns only applicable requirements. The API is not designed to return the full universe of Enhesa content or all applicability logic inputs. Partners should treat the Dashboard as the system of record for configuration (contracted content and applicability), and the API as the delivery mechanism for the resulting applicable content set per facility.

Getting started

Key Concepts

Partner / Integration Client (OAuth): the calling application identity used in the OAuth client-credentials flow.

Enhesa Client (clientId): the Enhesa business customer for which you are authorized to request data.

Facility (facilityId): a client site; requirements are returned per facility and language.

Requirement: an applicable compliance obligation; each requirement may reference one or more legal foundations.

Legal Foundation: source content such as a regulation or citation supporting a requirement

Pagination - a technique in APIs to manage and retrieve large data sets efficiently.

Environments and Base URLs

The following is the main production base URL:

https://api.enhesa.com/compliance-intelligence/v1

NOTE: You will be given a Sandbox client ID to use for integration testing.

API versioning and breaking changes

Our REST APIs will evolve as the product grows. We create a **new API version (for example, moving from `/v1` to `/v2`) only when a change is breaking**. A change is considered **breaking** if it could cause an existing integration to fail or behave incorrectly without any code changes.

Breaking changes (new API version required)

- Removing or renaming endpoints (operations)
- Removing or renaming request parameters or response fields
- Adding a **new required** parameter, or making an optional parameter required
- Changing a field's data type or format in a way that impacts parsing/validation
- Removing enum values

- Adding new validation rules that reject requests that previously succeeded
- Changing authentication or authorization requirements (for example, scopes, permissions, token requirements)

Non-breaking changes (no new API version)

- Adding new endpoints (operations)
- Adding new **optional** parameters or headers
- Adding new response fields or headers
- Adding enum values

Important: Adding new properties to a response is explicitly **non-breaking**. Partners are expected to safely ignore any fields they do not use. This keeps integrations stable while allowing the API to evolve without unnecessary version proliferation.

Pagination and Filtering

Several endpoints return paginated results. When pagination is supported, the response includes the fields below to help you understand the full result set and where you are within it:

- **totalItemsNumber**: Total number of items that match the request (across all pages).
- **itemsPerPage**: Number of items returned per page (page size).
- **totalPages**: Total number of pages available, calculated as totalItemsNumber / itemsPerPage (rounded up).
- **page**: The current page number returned in this response.

To retrieve a different page, include the pagination request parameters for that endpoint (for example, `page=2` and optionally `itemsPerPage=50`). Default and maximum values can vary by endpoint. See the OpenAPI specification for the exact parameter names, defaults, and limits for each endpoint.

1.1 Errors

Errors are returned as JSON with an errorCode and errorDescription. Validation errors may include an errors array with individual field issues.

- **401 Unauthorized:** missing/invalid token or malformed request.
- **403 Forbidden:** authenticated but not authorized for the resource.

- **500 Server Error:** unexpected condition or unhandled exception.

Authentication

Getting a Bearer Token

All API requests require an OAuth 2.0 Bearer token for authentication. To obtain a token, follow the steps below.

Obtain OAuth client credentials

Request client credentials (client_ID, client_secret, EnhesaClientID) and scope from the **Enhesa Product Support Team.** Client credentials will be sent via secure email.

Auth Flow Terminology:

client_id – The OAuth client ID used to authenticate your request in the Client Credentials flow.
NOTE: The client_id does not represent the Enhesa client but rather the third party making the API calls (which can be either a Platform Partner or direct Client). Every entity making an API call has a single client_id. If you are a Platform Partner, you can use your client_id to manage multiple "enhesaClientIDs"

client_secret – The secret key used by OAuth to authenticate your request. **Do not share.**

grant_type – A hardcoded value of "*client_credentials"* to specify the authentication flow.

audience – Specifies the intended recipient of the token. You should use the value: https://api.enhesa.com/compliance-intelligence.

scope – The permissions (scopes) the token should have. Each scope corresponds to an API or a set of API permissions. This will be provided by Enhesa Product Support.

enhesaclientid – The business client identifier for which you are requesting data on behalf of. There can only be one of these IDs in a token request, as the token is issued for a single Enhesa Client. These IDs will be provided by Enhesa Product Support for each Enhesa client you are pulling data for.

Request a Bearer Token

Make a POST request to the **https://identity.enhesa.com/connect/token** endpoint using your **client credentials**.

```
POST https://identity.enhesa.com/connect/token
Content-Type: application/x-www-form-urlencoded

client_id=YOUR_CLIENT_ID
&client_secret=YOUR_CLIENT_SECRET
&grant_type=client_credentials
```

```
&audience=https://api.enhesa.com/compliance-intelligence
&scope=YOUR_SCOPE_1 YOUR_SCOPE_2
&enhesaclientid=YOUR_ENHESA_CLIENT_ID
```

Example Response (If valid)

```
{
  "access_token": "eyJhbGciOiJIUzI1...",
  "token_type": "Bearer",
  "expires_in": 3600,
  "scope": "your-scope-1 your-scope-2"
}
```

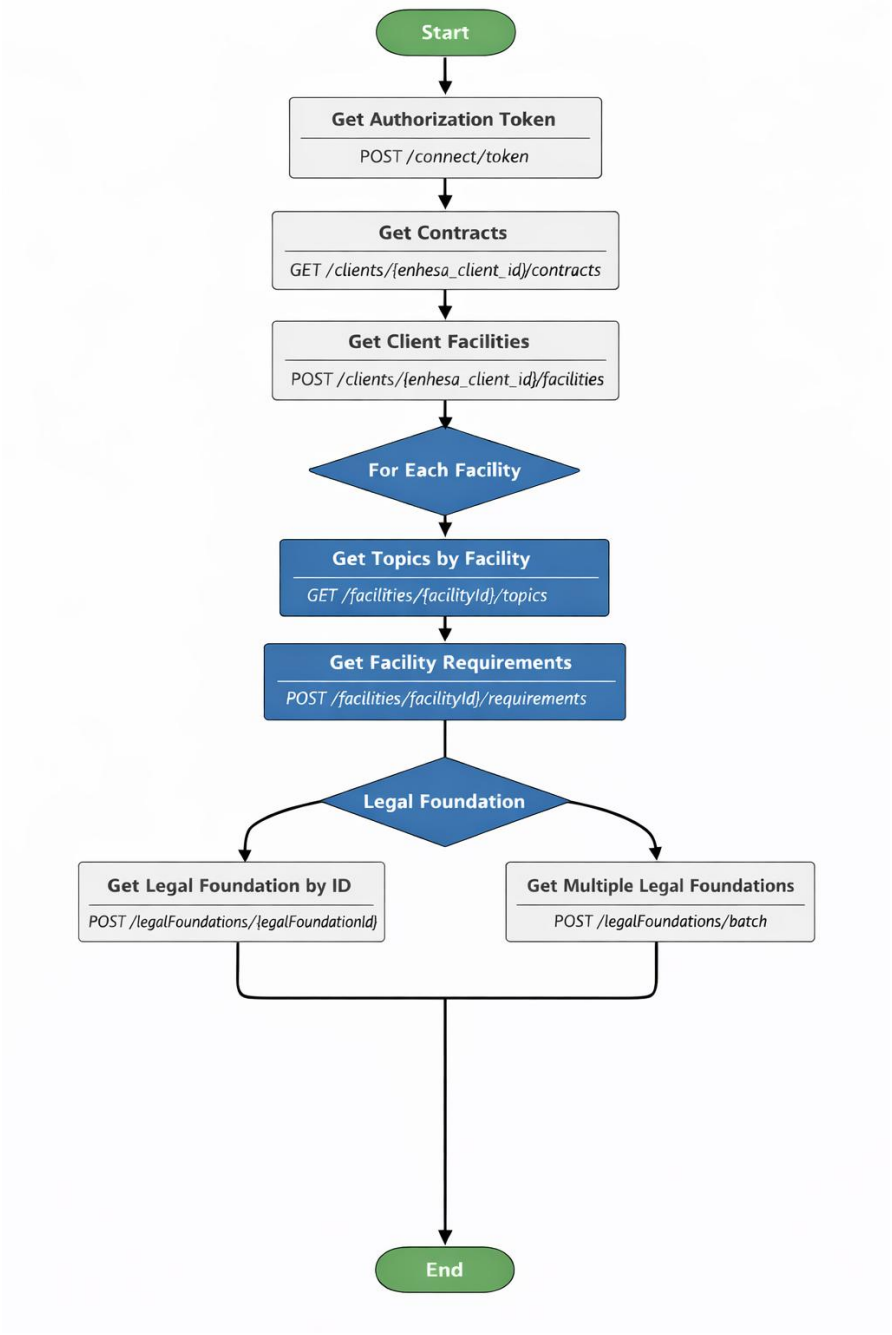
Core Workflows

The CI REST API is designed around a client → facilities → requirements model. Requirements returned by the API are applicable requirements based on applicability configured in the Enhesa Compliance Intelligence application. A typical workflow looks like:

- Get **contracts** to discover subscribed jurisdictions, languages, and categorie;
- List **facilities** for a client and note supported languages per facility;
- For each facility:
    - Retrieve **topics**
    - Retrieve **requirements** for each language (optionally using fromDate/toDate for incremental change sync).
        - Retrieve **legal foundation** details for legalFoundationId values referenced by requirements (single or batch).

Note: Applicability must be performed in the Compliance Intelligence application as the API only sends applicable requirements.

See a visual representation of the workflow below.


Start
Get Authorization Token
POST /connect/token
Get Contracts
GET /clients/{enhesa_client_id}/contracts
Get Client Facilities
POST /clients/{enhesa_client_id}/facilities
For Each Facility
Get Topics by Facility
GET /facilities/{facilityId}/topics
Get Facility Requirements
POST /facilities/facilityId}/requirements
Legal Foundation
Get Legal Foundation by ID
POST /legalFoundations/{legalFoundationId}
Get Multiple Legal Foundations
POST /legalFoundations/batch
End

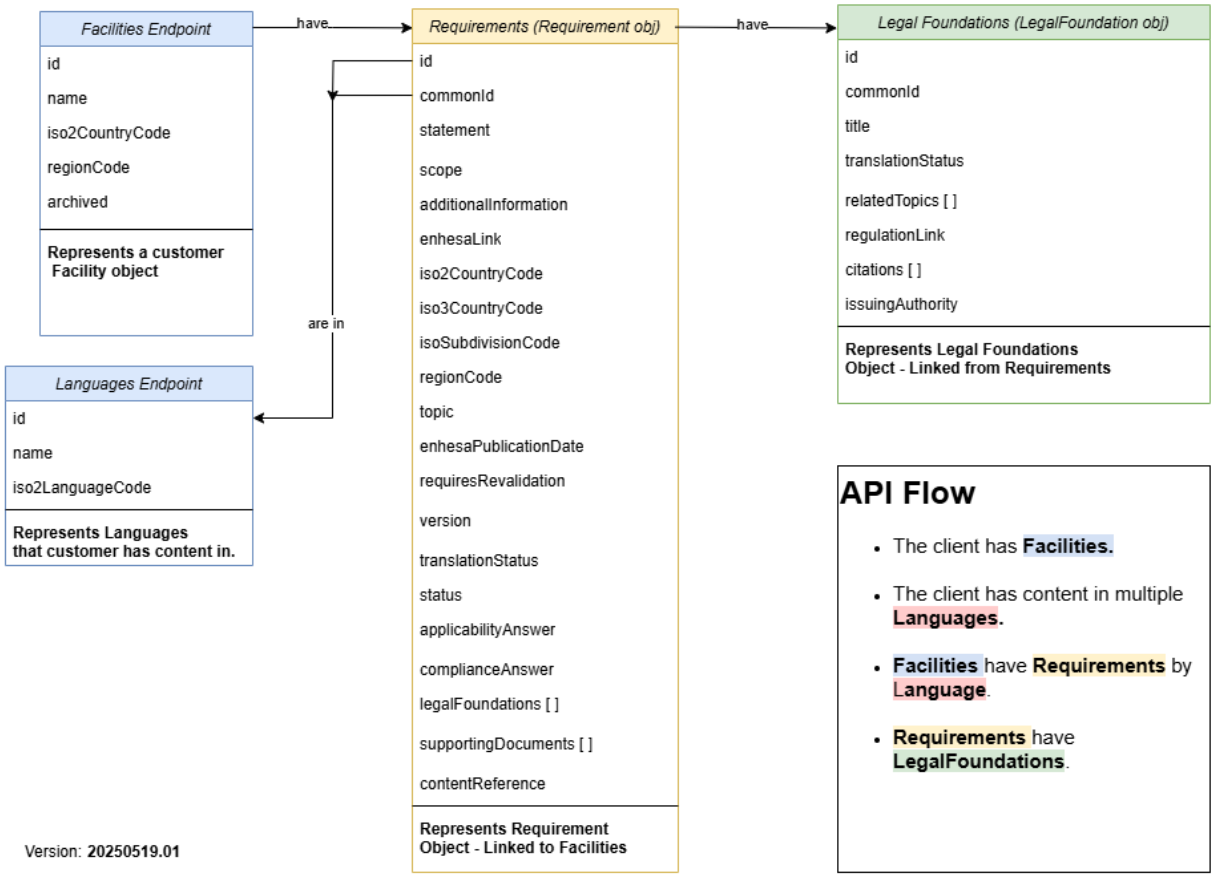
Entity Relationships

To better understand the core workflow in accessing data, follow the below steps and diagram.

The **client** has Facilities
The Facility endpoint is used to acquire a list of client facilities.
It also contains the subscribed languages for the facilities
The Facilities have Requirements, by Language.
The Requirements endpoint will return list of requirements per facility, in the requested language.
Requirements have Legal Foundations (e.g. a Regulation).
Legal Foundation endpoints can be used to get detailed Legal Foundation data.

This information relates to **basic** relationships of content and usage of the APIs to request a client's information. It is **not** a comprehensive list of all the objects and models available. For that information, please see the OpenAPI spec.

Enhesa CI Public API Entity Relationship



API Flow

- The client has **Facilities.**
- The client has content in multiple **Languages.**
- **Facilities** have **Requirements** by **Language**.
- **Requirements** have **LegalFoundations**.

Change Management & Incremental Syncs

Change Management and Incremental Sync

For incremental synchronizations, the **Requirements endpoint** supports requesting changes since a date using fromDate (and optionally toDate) provided in the request body. Dates use YYYY-MM-DD format.

```
{
  "fromDate": "2026-01-01",
  "toDate": "2026-01-31"
}
```

How It Works

When the API receives a request with fromDate, the system:

- Compares fromDate with the last tracked date stored for that integration.
- Retrieves major, minor and applicability changes modified on or after this date.Returns the latest version of each applicable requirement.

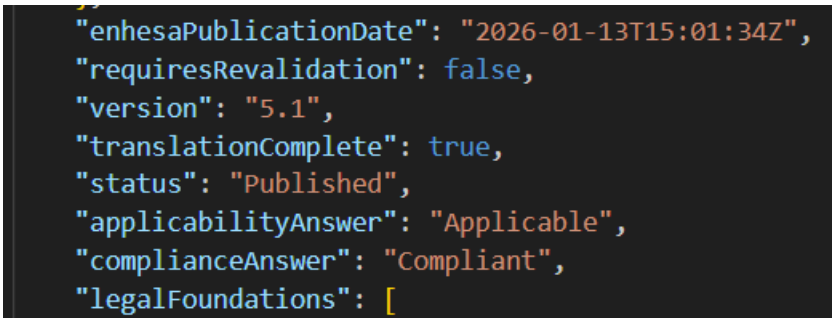
This approach supports incremental updates and reduces payload size.

How the API Handles Requirement Updates

When a requirement gets updated, typically you want to flag that requirement for revalidation by a user. Below is the suggested workflow:

- The version attribute is formatted as a string with two numeric segments separated by a dot
- The major.minor version format shown on requirements (for example, 3.1). The first number is the major version and indicates a meaningful change. If a requirement moves from 3.1 to 4.1, that is a major update and should be treated as requiring client review (revalidation). If it moves from 3.1 to 3.2, that is a minor update. The attribute “enhesaPublicationDate” that shows the date this change was published.

Example: "enhesaPublicationDate": "2026-01-13T15:01:34Z"



Important Notes:

.

- The API does not expose full version history
    - If users want to see the complete version history or review the details of each change, they must use the Enhesa Dashboard through the provided Enhesa links.
- The API does not include detailed change descriptions.
    - Users must visit the Enhesa Dashboard to view the full change notes associated with each update.

Recommended approach

- Store the timestamp of your **last successful sync** (per tenant and facility).
- For each incremental run, set `fromDate` to that stored timestamp.
- To reduce the risk of missing updates due to processing delays or clock skew, consider overlapping by a small window (for example, set `fromDate` to **last successful sync minus 24 hours**).

Understanding id vs commonId

In the **Requirements** endpoint, two identifiers are used to distinguish between the master content identity and individual localized or version specific instances: `commonId` and `id`.

commonId – Master Content Identifier

The `commonId` represents the underlying requirement, regardless of language or version. All language variants and regional representations share the same `commonId`. It acts as the stable, universal key for grouping the same requirement across systems.

Purpose:
Identifies the same requirement regardless of:

- language (EN, local)
- region-specific variations
- minor updates

Example Scenario:
If a requirement exists in:

- English
- Spanish
- French

All three versions will share the same commonId. Think of commonId as the "master record ID" for the requirement.

id – Localized or Version-Specific Identifier

The `id` field represents a specific instance of the requirement, typically a unique version tied to a particular language or extraction. English and local language versions of the same requirement will have different `id` values but share the same `commonId`.

Purpose:

- Distinguish English vs Local langauge versions
- Track updates over time
- Allow systems to store separate rows for each localized requirement

Think of id as the “specific version” of the requirement.

IMPORTANT NOTE: If you are upgrading your integration from the legacy XML interface, you will need to do a mapping of QNCodes (former Requirements ID) to the new ids. Enhesa Product Support will provide you with a full mapping file. Please see Appendix 1 for an example of the mapping file.

Understanding the "requiresRevalidation" Field in Requirements Endpoint

The "requiresRevalidation" field is a boolean flag returned by the **Requirements** endpoint. It indicates whether a change to a requirement requires user revalidation through the Enhesa Dashboard. A value of "true" means the user must review and validate the updated requirement; "false" means no validation action is required.

Behavior of “requiresRevalidation”

Applicable vs Not Applicable Requirements

Currently, the API only returns applicable requirements. Therefore, the "requiresRevalidation" flag applies exclusively to applicable requirements. “Not applicable” requirements are not included in the API response, so users do not receive revalidation information for them.

Upcoming Email Notification Feature (Q2 2026)

A new email notification feature planned for Q2 2026 will notify users when any requirement requires revalidation. This direct notification will inform users independently of API responses. As a result, the "requiresRevalidation" field becomes less critical, because notifications will cover both applicable and non-applicable requirements.

Note: Once the user performs revalidation on the dashboard, the status of “requiresRevalidation” will be set to False.

Endpoint Reference

Please find a complete listing of endpoints, schema and property descriptions in the Open API spec: on our Support Page

Get Contracts

GET https://api.enhesa.com/compliance-intelligence/v1/clients/{{enhesa_client_id}}/contracts

- Provides a list of jurisdictions, languages and categories

Get Client Facilities

POST https:// api.enhesa.com /compliance-intelligence/v1/clients/{{enhesa_client_id}}/facilities

- Provides a list of all facilities and jurisdictions

Get Topics by Facility

GET https:// api.enhesa.com /compliance-intelligence/v1/clients/{{enhesa_client_id}}/facilities/{{facilityId}}/topics?languageCode={{languageCode}}

- Provides a list of topics per facility

Get Client Facility Requirements

POST https:// api.enhesa.com /compliance-intelligence/v1/clients/{{enhesa_client_id}}/facilities/{{facilityId}}/requirements?languageCode={{languageCode}}

- Provides the Applicable requirements and associated Legal Foundation ID's per facility.

Get Legal Foundation by ID

POST https:// api.enhesa.com /compliance-intelligence/v1/clients/{{enhesa_client_id}}/legalFoundations/{{legalFoundationId}}

- Provides the Legal Foundations on each Foundation ID provided

Get Multiple Legal Foundations (Batch)

POST https:// api.enhesa.com /compliance-intelligence/v1/clients/{{enhesa_client_id}}/legalFoundations/batch

- Provides the Legal Foundation on based on a list of Legal Foundation IDs

Postman Setup and Usage

We provide a Postman collection with two environments (for example: **Test/Sandbox** and **Production**) to help you authenticate and run the Compliance Intelligence API calls quickly. After importing the collection and selecting the appropriate environment, configure the variables below so requests can be executed without manual edits.

For access to the Postman Environments and Collections you can follow the link below, or reach out to services@enhesa.com.

https://support.enhesa.com/hc/en-us/articles/36455830825748-Enhesa-Integration-REST-API-Documentation

Environment variables to set

- **baseUrl**: The base URL for the CI API environment you are using (for example, the sandbox or production API host).
- **tokenUrl**: The OAuth token endpoint URL for the environment.
- **client_id**: Your OAuth client ID (issued by Enhesa).
- **client_secret**: Your OAuth client secret (issued by Enhesa).
- **enhesaClientId**: The Enhesa tenant identifier (the customer you are pulling data for).
- **facilityId**: The facility identifier you want to retrieve applicable content for.
- **languageCode**: The language to return content in (for example, en, if supported by the endpoint).

Once these variables are set, you can run the collection in the sequence below to validate connectivity and perform an initial sync.

Recommended call order (Quickstart flow)

1. **Get Access Token (OAuth Client Credentials)**
    - Generates a bearer token using `client_id, client_secret`, and required scopes.
2. **Get Contracts**
    - Confirms the tenant has contracted content available for delivery.
3. **List Facilities**
    - Retrieves facilities available under the tenant and helps you confirm or select the correct `facilityId`.

4. **Get Topics (by Facility)**
    - Returns available topic filters for the facility (if used in your integration).
5. **Get Applicable Requirements (by Facility)**
    - Pulls the facility-specific set of applicable requirements (optionally filtered by topic, language, date range, etc. depending on the endpoint).
6. **Get Legal Foundations (optional)**
    - Retrieve legal foundations as needed (for example, to support legal register workflows).
7. **Daily Delta Sync (ongoing operations)**
    - Run the “changes since last sync” requests daily to pull only updates (content changes and applicability/compliance-answer-driven changes) since your last successful run.

Tip: Store the access token returned by the token request in the environment (collection scripts can do this automatically). Subsequent requests should use `Authorization: Bearer {{access_token}}` so you can run the full sequence without copying tokens between calls.

Support

- **Enhesa Product Support**: support@enhesa.com
- If you lose client credentials, Product Support can regenerate and share via a secure mechanism (e.g., 1Password).
- Enhesa provides Postman environments (sandbox with static data and production) via the Enhesa support portal.

APPENDIX 1: QNCode to NextGen Codes Example

If you are upgrading your integration from the Legacy XML interface to the REST API, you will need handle a mapping of the Requirements QNCodes to the NextGen IDs.

Enhesa Product Support will provide you with full mapping file during your integration upgrade. Below is an example of what the mappings will look like:

LegacyQnCode	NextGenRequirementId	NextGenRequirementCommonId
CNQ12255	25f3812e-bb3c-6d57-e063-f392abd58337	25f38129-9b66-6d57-e063-f392abd58337
KYQ00080	25f3812c-cd0c-6d57-e063-f392abd58337	1d5aec40-17d5-d293-e063-f392abd51fcf
TRQ02248	25f3812e-a91a-6d57-e063-f392abd58337	25f38129-7132-6d57-e063-f392abd58337
DZQ00949	25f3812f-9358-6d57-e063-f392abd58337	1d5ae1f9-3218-d349-e063-f392abd57958
SIQ01458	3bc6dc61-9973-7bc1-e063-0e63a8c0ad62	3bc6dc61-9972-7bc1-e063-0e63a8c0ad62
DEQ04858	2f20f816-eaad-6a61-e063-0e63a8c0dabd	2f20f816-eaac-6a61-e063-0e63a8c0dabd
JPQ00076	25f38130-13bb-6d57-e063-f392abd58337	25f38126-9040-6d57-e063-f392abd58337
UKQ00881	25f3812b-944c-6d57-e063-f392abd58337	25f38127-08b5-6d57-e063-f392abd58337
CYQ00702	25f3812f-9a51-6d57-e063-f392abd58337	1d5af150-c4c8-d291-e063-f392abd5ac29

Belgium
Louise Center Building
Avenue Louise 287
1050 Brussels
t. +32 2 775 97 97

United States
1911 North Fort Myer
Drive Suite 1150
Arlington, VA 22209
t: +1 202 552 1090

Japan
1-5-15, Hirakawacho
Chiyoda-ku
Tokyo 102-0093
t. +81 (0)3 6261 2138

China
Room 1723, 17F, Building 1
No. 1198 Century Avenue
Shanghai China
t. +86 21 5072 1956

Canada
130 Spadina Avenue
Suite #402
Toronto

www.enhesa.com
info@enhesa.com

APPENDIX 2: Modification History

Author	Title	Version	Release Date (dd.mm.yyyy)	Description of Change
Bob Underbrink	EHS Product Manager	1.1	06.02.2026	Adding new endpoints
Bob Underbrink	EHS Product Manager	1.0	08.10.2025	Official Release
Eric Hepler	Director of Applications Engineering Americas	0.2	19.05.2025	Clarifications to object model including language codes, citations, change notes
Eric `Hepler	Director of Applications Engineering Americas	0.1		Initial Version

Belgium
Louise Center Building
Avenue Louise 287
1050 Brussels
t. +32 2 775 97 97

United States
1911 North Fort Myer
Drive Suite 1150
Arlington, VA 22209
t: +1 202 552 1090

Japan
1-5-15, Hirakawacho
Chiyoda-ku
Tokyo 102-0093
t. +81 (0)3 6261 2138

China
Room 1723, 17F, Building 1
No. 1198 Century Avenue
Shanghai China
t. +86 21 5072 1956

Canada
130 Spadina Avenue
Suite #402
Toronto

www.enhesa.com
info@enhesa.com