Prism - 1.7.0
SaaS Technical Documentation

About

Prism provides analysis, comparison, and review capabilities for annual financial reports prepared according to the technical specifications of the « single electronic format » in use in the European Union and the United Kingdom.

Users upload financial reports to the application by sending requests to a REST API. The application processes the report and replies to the query with a file downloaded to the user's workstation. **Reports are uploaded, processed in-memory and the result are sent to the user. Reports and analyses are never stored in the application server.**

Logs kept to measure usage are restricted to simple information (timestamp, client id) and never includes the report's contents.

Key features

- Creation of an Excel file laying out the information contained in electronic format into several sheets, each providing a different viewpoint into the analysed file.
- Validation of the file through a third-party certified conformant validation engine.
- Creation of a « viewer » interactive HTML file for manual review of the report.
- Comparison of versions of reports in an Excel sheet, highlighting the key changes between the two files.

Technical prerequisites

End-user

Browser : Chrome version 85+ or Edge version 85+ or Firefox version 80+

Minimum resolution : 1280 x 800

Internet access : a connection allowing the upload and download of files up to 100MB.

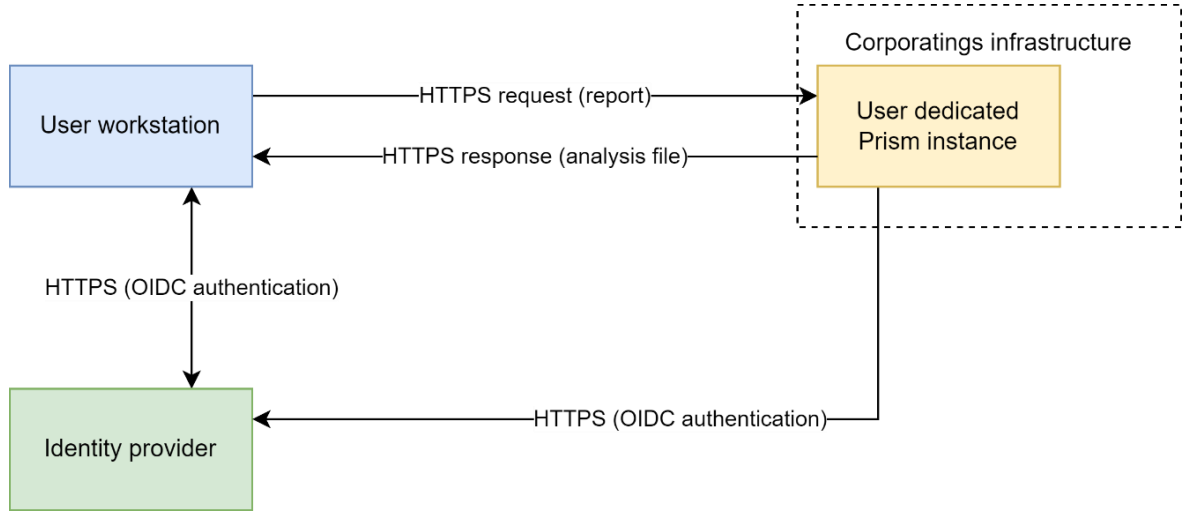
Using an identity provider (OIDC)

To restrict the access to the API to authenticated users, and to activate the authentication in the front-end, the following parameters must be provided to Corporatings:

- **Provider** : the base URL for the entry points of the identity provider. In particular, it is expected that a request to à **{url} /.well-known/openid-configuration** yields the list of OIDC and Oauth 2.0 entry points.
- **ClientId** / **ClientSecret** : the credentials for the Prism application at the identity provider, to be used when requesting or recvoating tokens
- **Domain :** the address at which the application will be served. This should be consistent with the callback address, at which the application will be called back and receive the authorization code. By default, you should be able to keep the commonly used value "{domain}/signin-oidc".

This authentication follows the **Authorization code flow**.

Architecture



The solution is web application. The application exposes a simple-front end that enables users to submit files to the different entry points of the application.

Each client of Prism has their own dedicated instance of Prism (dedicated subdomain and dedicated pod in our Kubernetes infrastructure).

No other data inflow or outflow than the ones illustrated above are required. In particular, the application does not require any access to the Internet to perform the analysis of the file.

Logs

The following information about the usage of the application is kept :

- The date at which the report was processed
- The file name
- The SHA256 hash of the file
- The entity identifiers, or LEI(s), used in the file (for a valid file, only one, but may technically be zero or several)
- The name of the entity
- The end of the reporting period.

Handling of the data uploaded

The files sent to the application are ZIP archives. They are usually sizeable enough (2 to 100 Mo) that the data throughput may a major factor in the overall time it takes to analyze a file.

The uploaded files are kept in-memory in the application server only for the duration necessary to perform the analysis. There is not trace left of them on the server outside of the logs kept in the database (name, hash, reporting period and identity of the entity).

Third party tools

The application relies on the following third party tools :

- **Arelle** version 2.41 : Conformant validation tool for XBRL documents and Inline XBRL viewer plugin, Apache 2.0 license
  (https://github.com/Arelle/Arelle/blob/master/License.txt)

REST API

Route : **/api/audit/ready**
Method : GET
Parameters : none
Responses :

- 200 (OK) : a string, « ready » or « not ready », indicating that the database is ready to store information. Not useful if the database is not used to store logs.

Route : **/api/audit/viewerscript**
Method : GET
Parameters : none
Responses :

- 200 (OK) : the body is a static file, « ixbrlviewer.js ». This file should be in the same folder as the interactive viewers in order for them to properly work.

Route : **/api/audit/blocks**
Method : POST
Parameters :

- **Request body** (binary file) : the ESEF/UKSEF file to be analysed. The file should be a ZIP archive following the base structure of an XBRL “unconstrained Report Package”.
- **lang :** “en” or “fr” : determines the language of the HTML template used to present the data

Responses :

- 200 (OK) : the body is an HTML file, where the non-numerical information of the report is gathered and rendered.

Route : **/api/audit/blocksComparison**
Method : POST
Parameters :

- **Request body** (binary files) : the ESEF/UKSEF file to be analysed first, and the file to be used for comparison second. The files should be ZIP archives following the base structure of an XBRL “unconstrained Report Package”.

Responses :

- 200 (OK) : the body is an HTML file, where the non-numerical information of the two reports is gathered and rendered.

Route : **/api/audit/arelle**
Method : POST
Parameters :

- **Request body** (binary file) : the ESEF/UKSEF file to be analysed. The file should be a ZIP archive following the base structure of an XBRL “unconstrained Report Package”.
- **lang :** “en” or “fr” : determines the language of the HTML template used to present the data

Responses :

- 200 (OK) : the body is the HTML « viewer » file created to the plugin of Arelle.

Route : **/api/audit/list/{simplify}?lang={lang}**
Method : POST
Parameters :

- **Request body** (single or array of binary files) : the first file is the ESEF/UKSEF file to be analysed. The file should be a ZIP archive following the base structure of an XBRL "unconstrained Report Package".
  A second file can optionally be used, in which case it will be used for a comparison with the first file.
- **simplify** (1 or 0) : this parameter changes how some validation messages are written in the Excel output.
  Value **0** generally creates more compact messages and uses the technical names of the elements.
  Value **1** generally separates issues into different lines and uses the human-readable labels to refer to elements. It also introduces a few suggestions in those messages.
- **lang** (**en** or **fr**): the language for the messages and the interface in the output Excel files.

Responses :

- 200 (OK) : the body is an Excel file that includes several sheets analyzing the report data. If a second file was provided, the comparison results will be available in an extra sheet within the same Excel file.

Route : **/api/audit/bank/{simplify}?lang={lang}**
Function exactly the same as the above **/api/audit/list/{simplify}?lang={lang}**, but the "Benchmark" sheet in the output Excel file performs a benchmark especially appropriate for financial institutions.
