Prism - 1.5.0
Setup & 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.

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

Application server

Available hard disk space : 4 Go

Minimum available RAM : 1.8 Go

Recommended available RAM : 2.8 Go

Software :

Microsoft SQL Server 2022 16.0+ or Microsoft SQL Server 2019 15.0+ or Microsoft SQL Server 2017 14.0+

IIS Version 10.0+

ASP.NET Core Runtime 6.0+ (Hosting Bundle) : download link

End-user

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

Minimum resolution : 1280 x 800

Setup

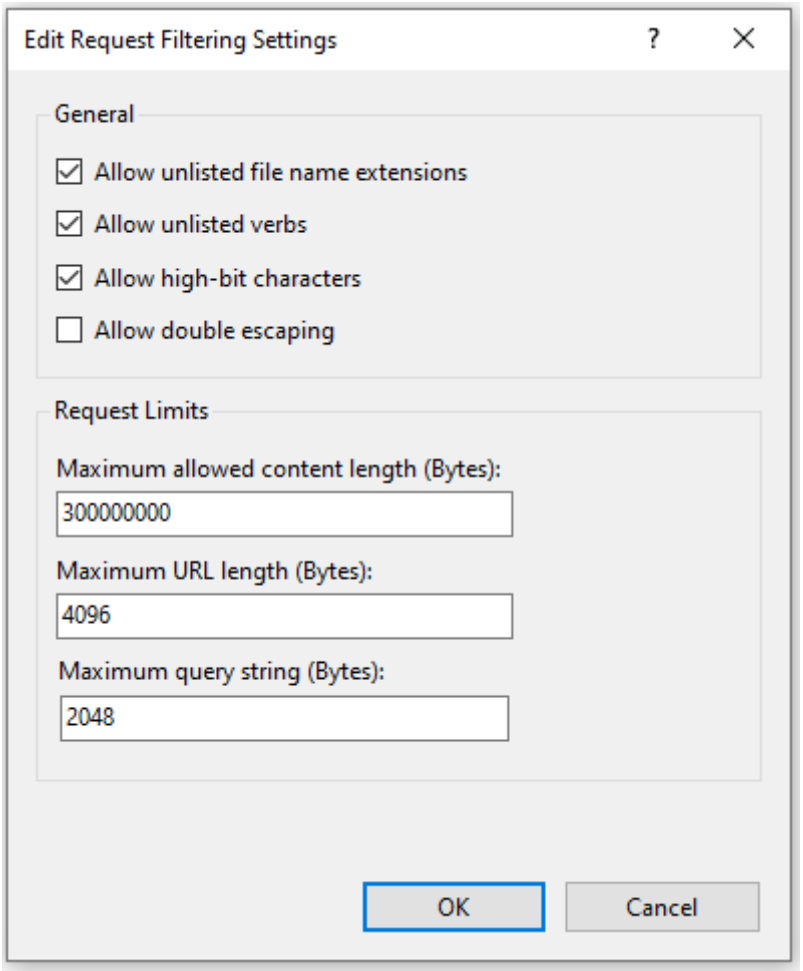
Extract all content from the **Prism_1_5_0.**zip archive into a new folder.

On IIS :

Create an Application Pool. Use the value **No Managed Code** for the **.NET CLR version** field.

Create a new website, using the physical path of the folder into which the archive content was extracted, and using the Application Pool created.

- The application may need to receive request bodies up to 300 Mo. In **Request Filtering**, click **Edit Feature Settings…** and enter a **Maximum allowed content length** at least this value.



- The application needs to write into some files in its internal folders, espcially in subfolders **./Resources** and for the file ***./Arelle/cache/arelle/cachedUrlCheckTimes.json***. Make sure the application has the corresponding rights.

- For security purposes, it is generally recommended to remove http headers that indicate the IIS version used. To remove them, follow the procedure described in the followin link under sections Server Header, “Using URL Rewrite”: https://techcommunity.microsoft.com/t5/iis-support-blog/remove-unwanted-http-response-headers/ba-p/369710

Configuration

(Optional) Using a database to log uses of the application

In the **appsettings.json** file in the root folder:

Assign the « **yes** » value to the property **UseDB** in the **Custom** section**.**

Assign to the **ConnectionString** field in the **Custom** section a connection string to a database that will be used by the application**.**

- Delimiter characters : backslash \ and quotes ‘ " in the connection string must be escaped by adding a backslash before each of them.
- The application creates the necessary tables in the database, but it does not create the database. The connection string must point to an existing database, either empty or previously used by the application (in its current version of a previous version).

If you do not wish to use a database, keep the « **no** » value in the « **UseDB** » field.

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 **OIDC** section of the **appsettings.json** must be filled with the following parameters:

- **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
- **CallbackPath :** the (relative) 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 « /signin-oidc ».

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

If you do not wish to use authentication, remove or rename the « **OIDC** » section.

Authentication logs

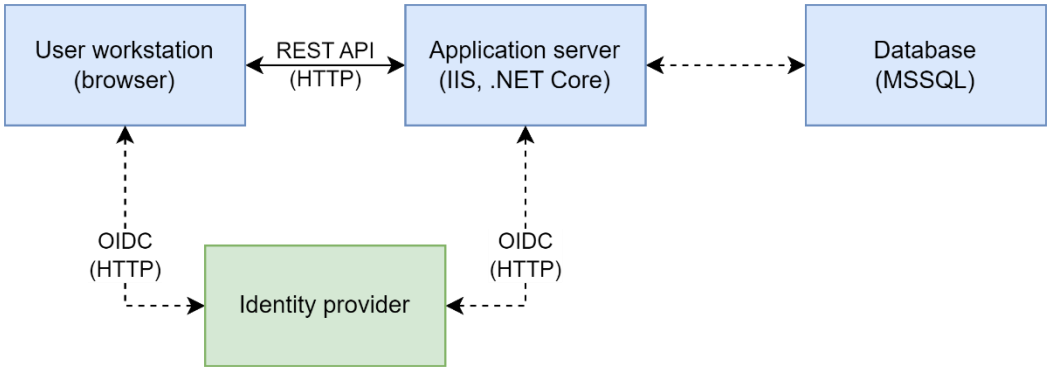
For debugging purposes, you may activate specific logging of authentication events by assigning the « **yes** » value to the **UseAuthLogs** field in the **Custom** section in the **appsettings.json** file.

When activated, these logs will be written in **debuglogs.txt** file at the root of the application folder.

You may then start the website.

Architecture

The solution is a three-tier application, to be deployed on a internal network. The use of the database and front-end are however technically optional, and it is possible to only rely on the application server's API.
The application may also use authentication, through an OIDC Single-Sign-On, in which case an identity provider is also necessary. No other data inflow or outflow is required. In particular, the application does not require any access to the Internet.



Database

Using the database is optional. When activated, the database is only used to store information into a single table named **Record**, where a single line will be added whenever the application is used to output the Excel file from an annual financial report.

```
SELECT TOP (1000) [Id]
      ,[ProcessingTime]
      ,[Filename]
      ,[Filehash]
      ,[LEI]
      ,[EntityName]
      ,[ReportingPeriodEnd]
  FROM [ESEFCHECKER106].[dbo].[Record]
```

ésultats | Messages

Id	ProcessingTime	Filename	Filehash	LEI	EntityName	ReportingPeriodEnd
1	2022-01-31 00:00...	scor-2020-12-31-FR.zip	673856F8 23E671AF D0...	96950056ULJ4JI7V3752		2020-12-31 00:00:...
2	2022-01-31 00:00...	TC1_valid.zip	51D4B5A2 A2C0F610 6E...	254900ARU0VC1WY6GJ71	ABC PLC	2019-12-31 00:00:...
5	2022-01-31 00:00...	TC1_valid.zip	67733427 03FBA902 8E...	254900ARU0VC1WY6GJ71	ABC PLC	2019-12-31 00:00:...
6	2022-01-31 00:00...	TC2_invalid.zip	3750B251 C5F5D814 95...	254900ARU0VC1WY6GJ71	ABC PLC	2019-12-31 00:00:...
7	2022-01-31 00:00...	abionyxpharma-2020-12-31AR.zip	BE59A96F 23872C13 83...	969500785J7VIC5YPC96	ABIONYX PHARMA	2020-12-31 00:00:...
8	2022-01-31 00:00...	abionyxpharma-2020-12-31AR.zip	BE59A96F 23872C13 83...	969500785J7VIC5YPC96	ABIONYX PHARMA	2020-12-31 00:00:...

In particular, the table stores :

- 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, if it is marked up in the report
- The end of the reporting period as described in the electronic data in the report.

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 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), if that option is used.

In case of an interruption (e.g. server shutdown) while a file is being processed, the files stored are cleaned at startup and every few minutes while the application runs.

Third party tools

The application relies on the following third party tools :

- **Arelle** version 2.37.0 : Conformant validation tool for XBRL documents and Inline XBRL viewer plugin, Apache 2.0 license (https://github.com/Arelle/Arelle/blob/master/License.txt)
- **iTextSharp** version 5.5.13.3 : PDF reading library
- **Selenium WebDriver** version 4.17.0 : Web navigation automation tool
- **ChromeDriver** version 121.0.6167.85 : WebDriver to interface Selenium with Chrome
- **Chrome** version 121.0.61617.85 : Web browser

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.

Authentication entrypoints (for OIDC)

Route : **/api/login/hasLogin**
Method : GET
Parameters : none
Responses :

- 200 (OK) : **true** if the application has been configured to restrict access to users authentified through an OIDC identity provider, **false** otherwise.

Route : **/api/login/goLogin**
Method : GET
Parameters : none
Responses :

- 200 (OK) : the url the browser must redirect to to perform the authentication