DOCUMENT REFERENCE :

|  |  |  |  |
| --- | --- | --- | --- |
| **APPROVERS LIST** | **NAME** | **SIGN OFF** | **DATE** |
| **LEAD ARCHITECT** | A.VEREPT & J.CADEL |  | 11/01/2024 |
| **PEER REVIEWER** |  |  |  |
| **DIGITAL I&O** |  |  |  |
| **DIGITAL SECURITY** |  |  |  |
| **QUALITY (GXP)** |  |  |  |

| **Version** | **Date** | **Author** | **Subject of the modification** |
| --- | --- | --- | --- |
| 1 | 11/01/2023 | A.VEREPT & J.CADEL | Draft |
| 2 | 05/02/2024 | O.BLONDEZ | Update Middleware part |
| 3 | 16/02/2024 | A.VEREPT | Update Dataplatform part |

|  |
| --- |
| **REFERENCE DOCUMENTS** |
| Document source : None |

|  |
| --- |
| **ADDRESSEES** |
| For application :  For information : |

SOMMAIRE

[1. OBJECT 3](#_Toc158975397)

[2. INTRODUCTION 3](#_Toc158975398)

[3. SCOPE OVERVIEW 3](#_Toc158975399)

[3.1. In Scope of the document 3](#_Toc158975400)

[3.2. Out of scope of the document 4](#_Toc158975401)

[4. SOLUTION OVERVIEW 4](#_Toc158975402)

[4.1. Assumptions 4](#_Toc158975403)

[4.2. Constraints 4](#_Toc158975404)

[4.3. solution overview 4](#_Toc158975405)

[5. MIDDLEWARE 6](#_Toc158975406)

[5.1.1. Description (No data transformation) 6](#_Toc158975407)

[5.1.2. Data sizing 9](#_Toc158975408)

[5.1.3. Scheduling / Trigger 9](#_Toc158975409)

[5.2.1. Description (No data transformation) 9](#_Toc158975410)

[5.2.2. Data sizing 11](#_Toc158975411)

[5.2.3. Scheduling / Trigger 11](#_Toc158975412)

[6. .NET 11](#_Toc158975413)

[7. dataPlatform 11](#_Toc158975414)

[8. Landscape 14](#_Toc158975415)

[9. Extent of Usage 14](#_Toc158975416)

[10. SECURITY REQUIREMENTS 15](#_Toc158975417)

[10.1. Authorizing access to Azure Storage 15](#_Toc158975418)

[10.2. encryption of the Data 15](#_Toc158975419)

[10.2.1. Encryption of the data stored on Data Lake 15](#_Toc158975420)

[11. TECHNICAL DESIGN 15](#_Toc158975421)

[11.1. Networks 15](#_Toc158975422)

[11.2. Operations Requirements 15](#_Toc158975423)

[11.2.1. Monitoring requirements 15](#_Toc158975424)

[11.2.2. Backup requirements 15](#_Toc158975425)

# OBJECT

Roquette strategy is to increase its expertise in the conception of new products and processes to better serve their customers.

To deliver these advanced functionalities, Roquette has decided to invest in a new capability to predict and modelize pharmaceutics processes. This capability is the Roquette Scientific Data Platform.

This capability will deliver data ingestion, processing and visualization features. It will also be a foundation for future modeling capabilities which will need data to operate.

# INTRODUCTION



# SCOPE OVERVIEW

This solution is under the perimeter of the Data & Advanced Analytics direction at the Group level and will be made available to other functions inside Roquette on a case-by-case basis.

In a near future, external systems may interact with the solution.

## In Scope of the document

* + General application design of the tools to:
    - Publish analysis data from MediPharma workstation towards snowflake for data visualization by Qlik
    - Publish analysis data from ELN towards snowflake for data visualization by Qlik
    - Network design for .net & TALAND

## Out of scope of the document

* + Functional design of the solution
  + Technical components design that are already deployed (ex: Data Lake, TALEND, Snowflake, Sharepoint)
  + AZURE Network design within the data plateform

# SOLUTION OVERVIEW

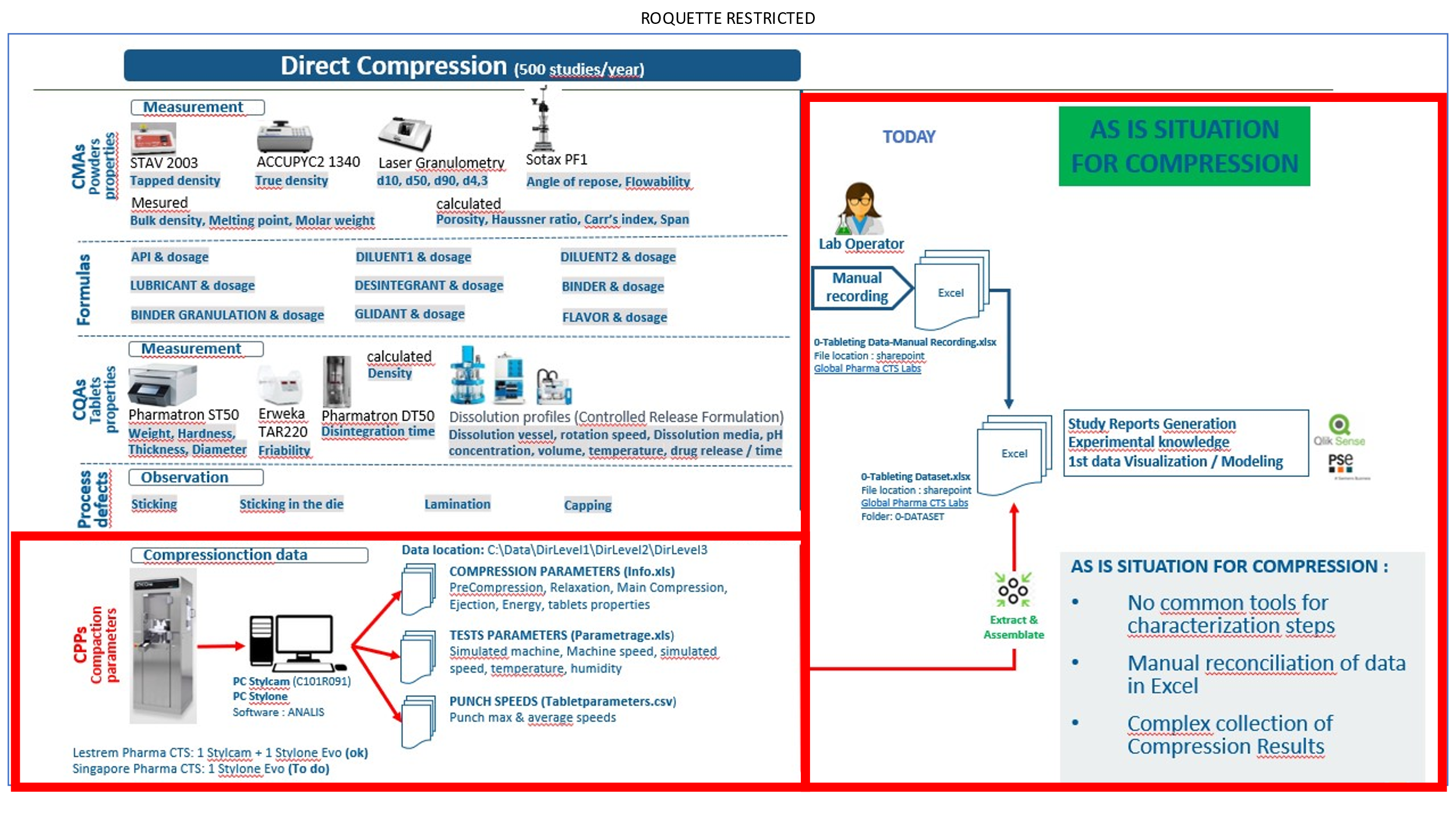
## Assumptions

The provider selected by the project team is Microsoft Azure cloud platform, TALEND & .net development frameworks

## Constraints

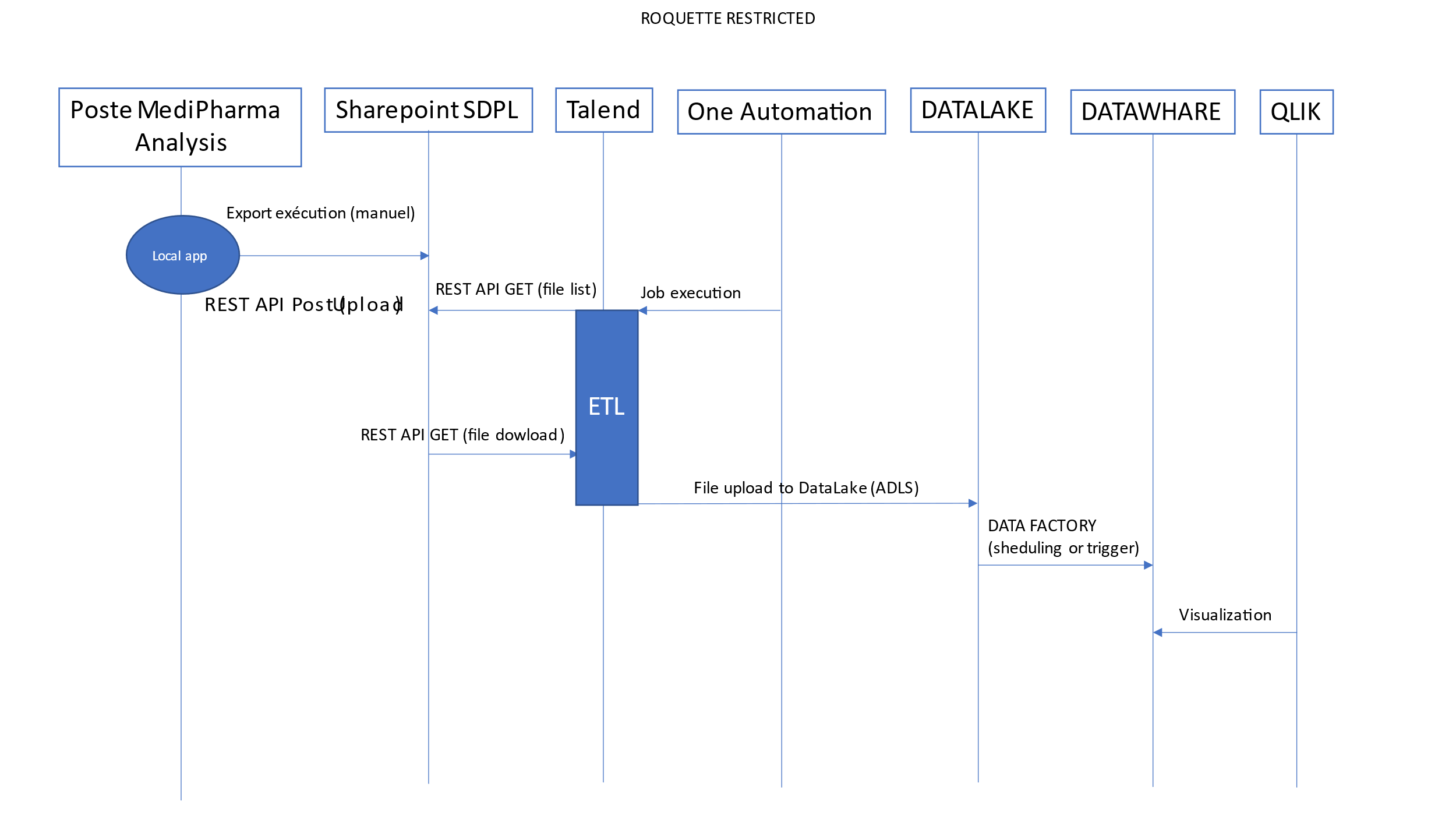
To enhance the security level, teams will be implementing security standards and best practices enabling authentication and encrypted transfer protocols

## solution overview

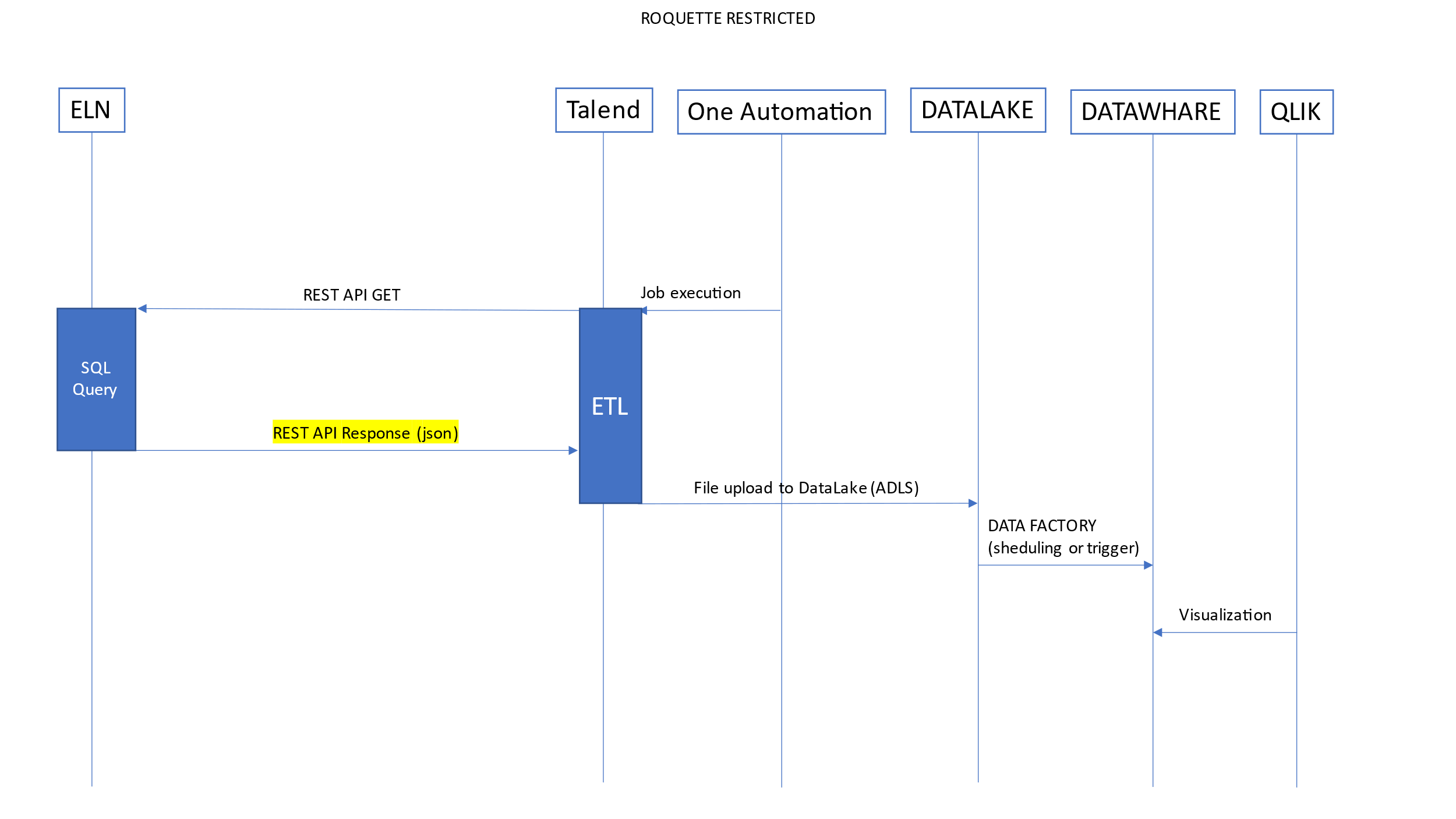


Description des 2 flux de données :

* Flow 1 : MediPharma workstation



* Flow 2 : ELN



# MIDDLEWARE

* 1. **Flow 1 : Sharepoint – (MediPharma workstation)**

### Description (No data transformation)

Une image contenant texte, capture d’écran, diagramme, Police

Description générée automatiquement

**Step 1 : Job execution (FEX)**

* **Frequency : Every 2 hours**

Error management in job execution:

Perform 3 retries in case of an error in the job:

- If execution 1 fails = 1 minute to wait before retry

- If execution 2 fails = 5 minute to wait before retry

- If execution 3 fails = job failed and notification sent to TALEND group by email **talend\_support\_l3@roquette.com**

The SharePoint is divided into directories for each of the 3 sites (Lestrem, Singapore, Philadelphia)

Create a single flow to retrieve the files present on SharePoint in the 3 directories corresponding to the 3 sites (Lestrem, Singapour, Philadelphie).

**Step 2 : REST API GET (File list)**

Using the getSharepointItemsList webservice to retrieve the list of files in the json response.

**Connection information:**

* Authentication method: Token (A new account will be created for this flow)
  + KLIF: to product in build
  + PROD : to product in UAT
* Endpoint : [https://roquettegroup.sharepoint.com/sites/SDOL-KLIF/\_api/Web](https://roquettegroup.sharepoint.com/sites/SDOL-KLIF/_api/Web?xsdata=%3D%3D&sdata=NVdFb1d2NSt6aXZWUWxnK2w3WTJrMlZ5NEJoNzk4ZjBFNE5RNzFpdStmbz0%3D&ovuser=b9fec68c-c92d-461e-9a97-3d03a0f18b82%2Colivier.blondez%40cgi.com)
* Sharepoint location :
  + DEV : [SDPL [DEV] - Home (sharepoint.com)](https://roquettegroup.sharepoint.com/sites/dev-SDPL)
  + PROD : [SDPL - Home (sharepoint.com)](https://roquettegroup.sharepoint.com/sites/one-collab-SDPL)

|  |
| --- |
| Name / ClientId / Secret value    APP\_POWERSHELL-SDPL\_DEV / 7f22ebae-7538-4b06-854d-6dfa6de16a1f / F-g8Q~Zg7G97Mc4z6OF1qUQy8AuY6ssSy7QcHc4\_    APP\_POWERSHELL-SDPL\_PROD / 80336d21-77b3-40d2-8997-4eb9851691c8 / rk48Q~ay1notojutJogNVoUhbEeWh8UvBTGTlatU    APP\_TALEND-SDPL\_DEV / 717d5700-8663-453c-a69d-a7e65c66a939 / o7a8Q~L2UNIs34o1IdZ-axplx8CHksVrx2ZuVcf2    APP\_TALEND-SDPL\_PROD / 93f6c62a-7cdf-4f39-884c-a1887e57ccd8 / 4mI8Q~K3Zi.YN4zMSZQIdaN2mgk1EwI-nWNptde2        curl --location 'https://login.microsoftonline.com/1c81ebf9-6e52-4cf8-b2c4-a3b65e90edf9/oauth2/v2.0/token' \  --form 'grant\_type="client\_credentials"' \  --form 'client\_id=""' \  --form 'client\_secret=""' \  --form 'scope="https://graph.microsoft.com/.default"' |

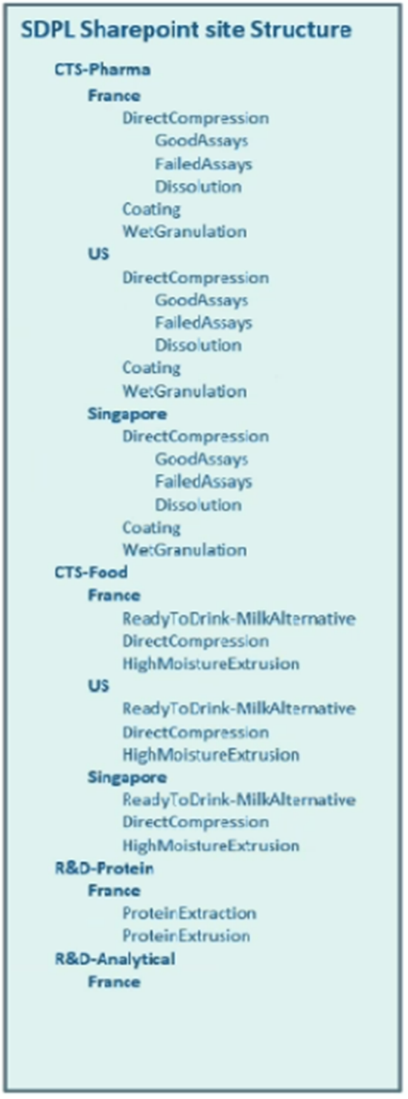
It will be necessary to make 3 successive calls to the getSharepointItemsList webservice (step 2) to retrieve the files in the 3 directories corresponding to the 3 sites (Lestrem, Singapore, Philadelphia).

It will also be necessary to make 3 successive calls to the getFileById webservice (step 3) to download the files in the 3 directories corresponding to the 3 sites.

CTS-Pharma is the only folder structure to browse initially, it is planned to add additional CTS later.

The project request is fixed on a single CTS (Le CTS-Pharma) as well as on the 3 sites corresponding to the 3 countries (LES: Lestrem, PHI: Philadelphia and SIN: Simgapore).

Plan a scalable structure to plan to add CTS if necessary as well as countries if necessary.



**Step 3 : REST API GET (file download)**

Using the getFileById webservice to download the files retrieved in the json response at the step 2.

**Connection information :**

* Authentication method : Token
  + KLIF : to product in build
  + PROD : to product in UAT
* Endpoint : [https://roquettegroup.sharepoint.com/sites/SDOL-KLIF/\_api/Web](https://roquettegroup.sharepoint.com/sites/SDOL-KLIF/_api/Web?xsdata=%3D%3D&sdata=NVdFb1d2NSt6aXZWUWxnK2w3WTJrMlZ5NEJoNzk4ZjBFNE5RNzFpdStmbz0%3D&ovuser=b9fec68c-c92d-461e-9a97-3d03a0f18b82%2Colivier.blondez%40cgi.com)

After recovering the files, they will be renamed in order to add as a prefix the CTS name and the trigram corresponding to the site of origin. Here are the trigrams for each of the sites:

* LES: Lestrem
* PHI: Philadelphia
* SIN: Simgapore

The format file name will be: [CTS Name]-[origin’s site trigram]-[File name].csv

For exemple : CTS-Pharma-LES-Dissolution-2024-02-12-17.10.15.677.csv

**Step 4 (This step is common to flows 1 and 2) :**

Files coming from the MiddleWare will be dropped in the landing zone of Azure Blob storage Datalake by the "File upload to DataLake (ADLS)" Talend ETL stream which is used by both flow 1 (MediPharma workstation) and flow 2 (ELN).

In this stream, the following information will be used to send file into the DATALAKE :

**Connection information :**

* Property type : Built-in
* Authentication method : Shared Key
  + KLIF : to product in build
  + PROD : to product in UAT
* Account Name : stroqlandzonedev (Test to be planned for PROD)
* Endpoint Suffix : dfs.core.windows.net

**Execution information :**

* ADLS FileSystem : sdpl-talend
* Blob Path : fileName (\*.JSON / \*.CSV) (Test to be planned)
* Local : filePath
  + E:\MIDL\Files\SDPL\ELN\OUT\\*.json pour flow 2 (ELN)
  + E:\MIDL\Files\SDPL\SHAREPOINT\OUT\\*.json pour le flow 1 (Sharepoint)
* UseFileList : fileMask (\*.JSON - \*.CSV)

Noticed :

* Timestamp to include in file names
* Potentially need an AD group so that the developer can access the Data lake interface

**Structure in the landing zone of Azure Blob storage Datalake to deposit files:**

**Container information :**

* + - CSV files from ELN and JSON files from Sharepoint :

storages accounts : stroqlandzone{dev/klif/prod}

container : sdpl-talend

* + - Manifest file :

storage account : stroqmanifest{dev/klif/prod}

container : manifest

folder : LANDING\_ZONE\_TO\_STAGING/

### Data sizing

### Scheduling / Trigger

* 1. **Flow 2 : ELN**

### Description (No data transformation)

Une image contenant texte, capture d’écran, diagramme, Police

Description générée automatiquement

**Step 1 : Job execution**

Error management in job execution :

Perform 3 retries in case of an error in the job :

- If execution 1 fails = 1 minute to wait before retry

- If execution 2 fails = 5 minute to wait before retry

- If execution 3 fails = job failed and notification sent to TALEND group by email **talend\_support\_l3@roquette.com**

Create a package designed to be able to easily add 1 request if needed.

**Step 2 : REST API GET (Query execution)**

Biovia (3DS) will provide us with a web service (in order to retrieve CISPRO and ELN data) which we will call every 2 hours.

This web service will return a json file to us with the data that has been modified.

/!\ Plan for the scenario where you had to add a third request along the way

The project request is fixed on 2 requests.

Provide scalable code to allow queries to be added if necessary.

**Connection information :**

* Authentication method : To be completed
  + KLIF : to product in build
  + PROD : to product in UAT
* Endpoint : To be completed

Ajouter les éléments de connexion (Compte & identifiants), Volume de donnée, noms de fichiers

**Step 3 : REST API GET - Second request (Query execution)**

Biovia (3DS) will provide us with a web service (in order to retrieve CISPRO and ELN data) which we will call every 2 hours.

This web service will return a json file to us with the data that has been modified.

/!\ Plan for the scenario where you had to add a third request along the way

The project request is fixed on 2 requests.

Provide scalable code to allow queries to be added if necessary.

**Connection information :**

* Authentication method : To be completed
  + KLIF : to product in build
  + PROD : to product in UAT
* Endpoint : To be completed

Ajouter les éléments de connexion (Compte & identifiants), Volume de donnée, noms de fichiers

**Step 4 (This step is common to flows 1 and 2) :**

Files coming from the MiddleWare will be dropped in the landing zone of Azure Blob storage Datalake by the "File upload to DataLake (ADLS)" Talend ETL stream which is used by both flow 1 (MediPharma workstation) and flow 2 (ELN).

In this stream, the following information will be used to send file into the DATALAKE :

**Connection information :**

* Property type : Built-in
* Authentication method : Shared Key
  + KLIF : to product in build
  + PROD : to product in UAT
* Account Name : stroqlandzonedev (Test to be planned for PROD)
* Endpoint Suffix : dfs.core.windows.net

**Execution information :**

* ADLS FileSystem : sdpl-talend
* Blob Path : fileName (\*.JSON / \*.CSV) (Test to be planned)
* Local : filePath
  + E:\MIDL\Files\SDPL\ELN\OUT\\*.json pour flow 2 (ELN)
  + E:\MIDL\Files\SDPL\SHAREPOINT\OUT\\*.json pour le flow 1 (Sharepoint)
* UseFileList : fileMask (\*.JSON - \*.CSV)

Noticed :

* Timestamp to include in file names
* Potentially need an AD group so that the developer can access the Data lake interface

**Structure in the landing zone of Azure Blob storage Datalake to deposit files:**

**Container information :**

* + - CSV files from ELN and JSON files from Sharepoint :

storages accounts : stroqlandzone{dev/klif/prod}

container : sdpl-talend

* + - Manifest file :

storage account : stroqmanifest{dev/klif/prod}

container : manifest

folder : LANDING\_ZONE\_TO\_STAGING/

### Data sizing

### Scheduling / Trigger

# .NET

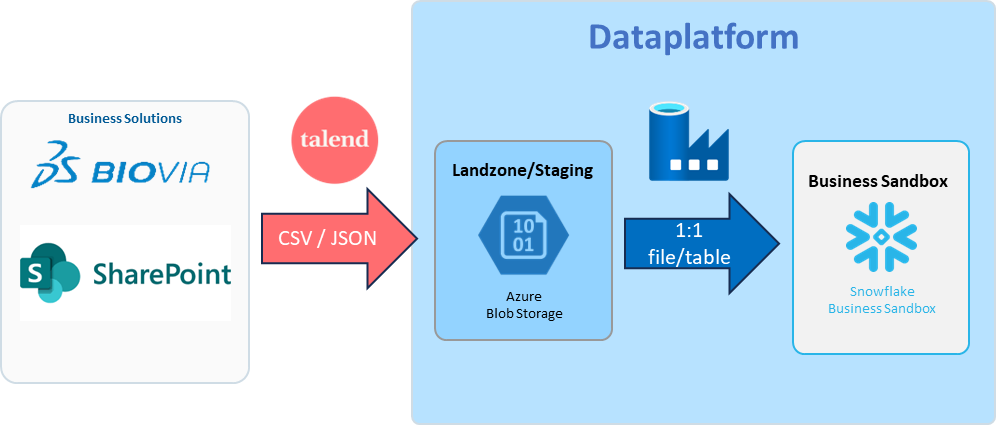
TODO

# dataPlatform

* 1. **Common to all flows**
     1. Description

The data structure of the different documents is defined at the beginning of the project. This structure is not to be changed frequently as the semi-structured format of data in Snowflake is not as flexible as unstructured data. A process needs to be defined between business, middleware and data team to fix the rules regarding a change of structure (add/change/remove/rename columns for example).

The data coming from the Middleware will be dropped in a landing zone (Azure Blob storage) and handle by Data factory pipelines.



This is the following process followed by the data:

**In case of an exploration phase (No data transformation):** Landing zone Datalake -> Staging Datalake -> Business Sandbox DWH

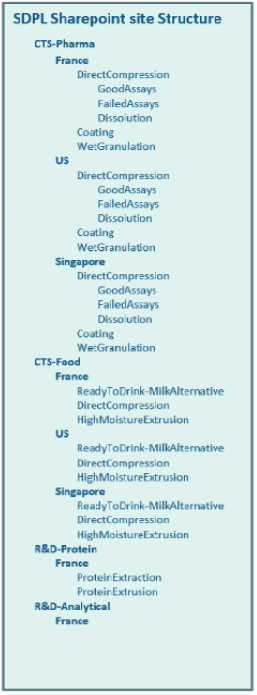
**In case of an industrialization:** Landing zone Datalake -> Staging Datalake -> Bronze DWH -> Silver DWH -> Gold DWH

* + 1. Scheduling / Trigger / Orchestration

Azure Datalake is able to create a trigger or a schedule for each file input coming from the Middleware. This trigger a flow of Azure Datafactory pipelines following the different transports steps

* 1. **Flow 1 : MediPharma workstation**
     1. Description

The files dropped are **CSV** or **JSONs** files. There is 3 sites -> 3 requests



More CTS and/or locations may be added in the future.

Need to be defined and fixed once and for all: csv structure (separator, delimiter, position of the header, escape character, line terminator and quote character)

Encoding is UTF-8

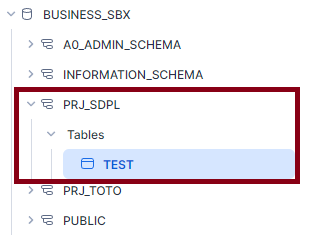
* 1. **Flow 2 : ELN**
     1. Description

The files dropped are **JSONs** files.

Encoding is UTF-8

* 1. **Snowflake Sandbox**
     1. Description

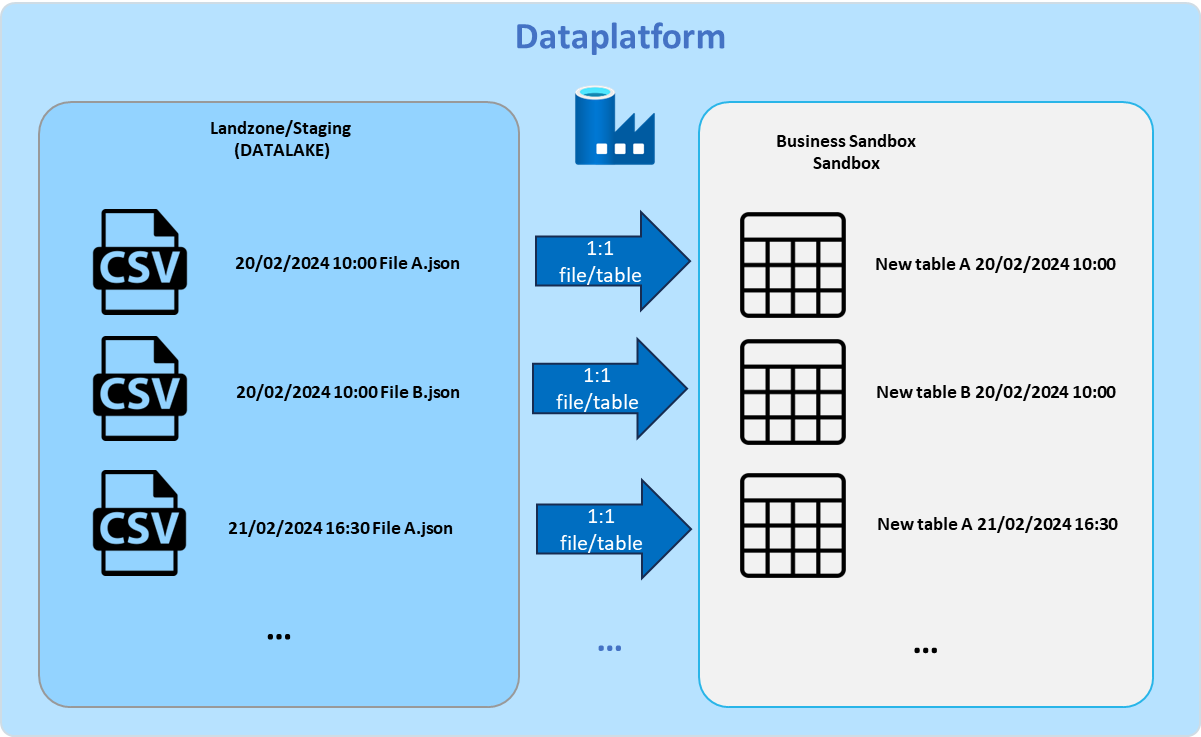
The Business Sandbox is created in Snowflake as a Database ‘BUSINESS\_SBX’ and an SDPL Schema is named ‘PRJ\_SDPL’. User will be able to read/write tables in this schema according to there granted permissions.



Data will be loaded as follow: **One file coming in the datalake generate one table in snowflake**. The naming convention will be ‘{filename}-{timestamp of loading}’

/!\ Even if two files have the same structure and names, they will induce the creation of a new unique table

Example:



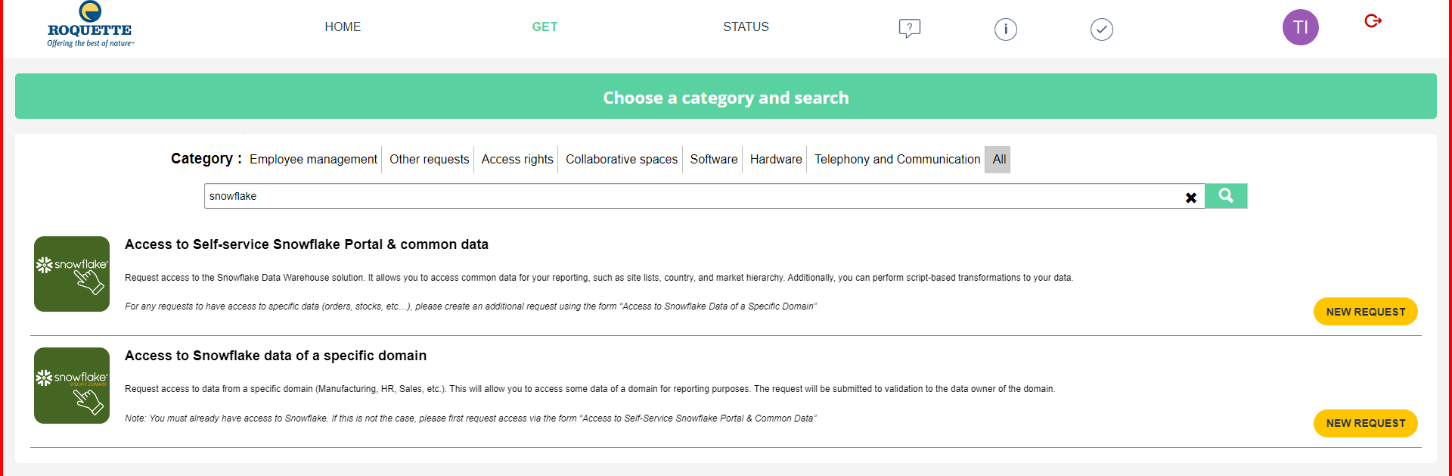
Data team is responsible to create the new tables for each file. However, users are responsible for their data once it has been correctly loaded.

* + 1. User Permissions

There are 2 levels of permissions for the users in this project:

* SDPL project writer *(RG\_USR\_CORP\_SNF\_FR\_PRJ\_SDPL\_WRITER\_DEV)*
  + Permissions to read/create/delete/update tables in PRJ\_SDPL
* SDPL project reader *(RG\_USR\_CORP\_SNF\_FR\_PRJ\_SDPL\_READER\_DEV)*
  + Permissions to read tables in PRJ\_SDPL

Members of each group will be defined at the start of the project and can be added later either by filling an Easy Vista request or by the project admin themself.



# Landscape

T

# Extent of Usage

# SECURITY REQUIREMENTS

## Authorizing access to Azure Storage

Azure supports using Azure Active Directory (AD) to authorize requests to the different resources. With Azure AD, role-based access control (RBAC) is used to grant permissions to a security principal, which may be a user, group, or application service principal.

Users will be Assigned reader permissions with AAD Groups and Talend will connect to the datalake landing zone using an Access Key

## encryption of the Data

### Encryption of the data stored on Data Lake

**See Data Lake HLD for more details**

# TECHNICAL DESIGN

* + TODO

## Networks

|  |  |  |  |
| --- | --- | --- | --- |
| **Source** | **Destination** | **Port** | **Connectivity** |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

## Operations Requirements

### Monitoring requirements

### Backup requirements