# Software Requirements Specifications

# HUB

## Purpose

The purpose of this document is to give a detailed description of the requirements for the “HUB” application.

## Scope

HUB is platform for automatically transferring data required by a project from the data-lake to the project’s space, based upon subscriptions made by that project. The subscription is made by an administrative user.

## Overall description

This section will give an overview of the whole system. The system will be explained in its context to show how the system interacts with other systems and introduce the basic functionality of it.

The development of the platform includes the development of the following components:

- **HTML/CSS, JS:** Develop an UI that allows user to choose the data he wants to subscribe to

- **Python:** Check for user right to access to the data (LDAP/Ranger query), access to data on the HDFS

- **Hive/Spark :** Table creation to allow columns selection and file-formatting

- **Oozie workflow:** Schedule the data delivery

The connection between components is showed in figure 1.



Figure 1.

## Specific requirements

### External interface Requirements

This section provides a detailed description of all inputs into and outputs from the system.

### User Interfaces

**Interface 1**

An administrative user of the platform should see a submit form when he opens the application. The submit form allows the user to submit his **identity** and the **project directory** he is currently working on, as shown in figure 2.

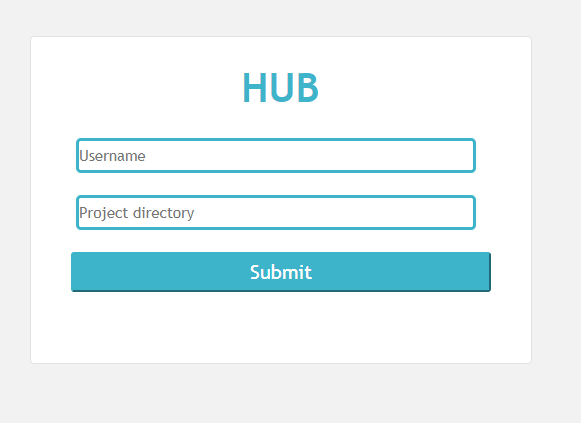


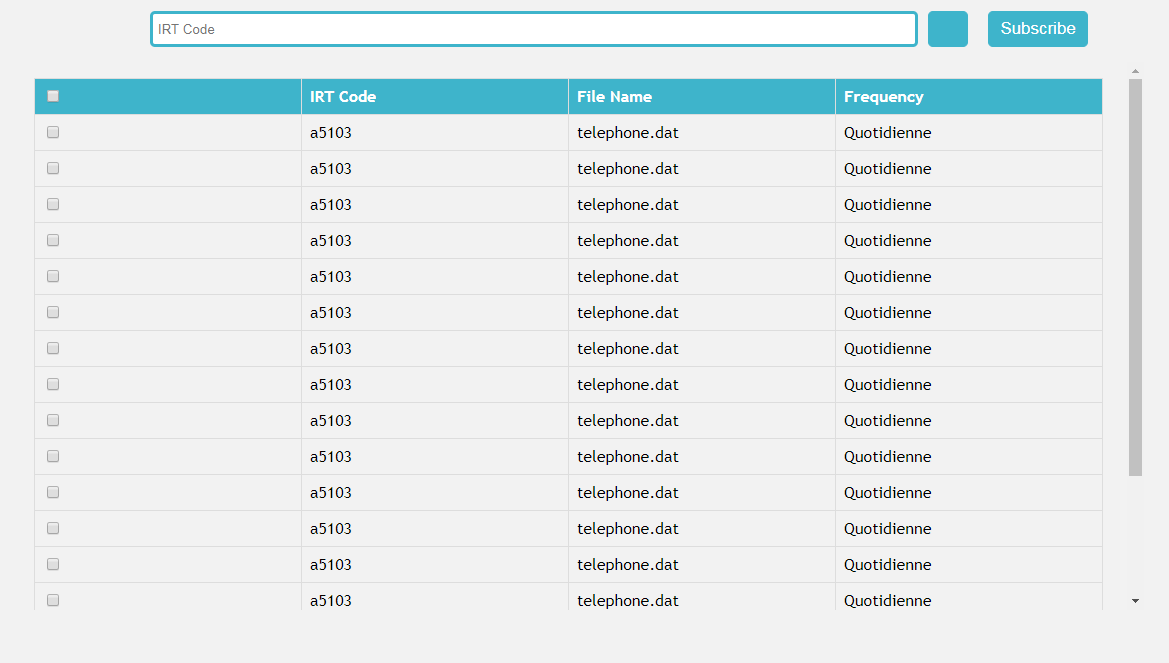
Figure 2

**Interface 2**

Once the user submits these information, she should be able to see the search page. Given the application id (cd\_IRT), the search page allows the user to look up for the files contained in the directory of this application in the HDFS. The search page also allows the user to select the files resulting from his search and to review the table contained in this file, regardless of the file format. Finally, the search page allows the user to select and subscribe to desired tables and columns, as well as the format he wants to export this selection to, as shown in figure 3.

When a subscription is made, the user will receive the specific tables with specific columns in the format that he subscribed for in his project space .The user can cancel the subscription by select the same file and click ‘unsubscribe’ at any time.

The user can change his project. In this case, he should be redirected back to “Interface 1”.



**Figure 3**

### Software interfaces

The HUB framework communicates two external interfaces. First, it connects with LDAP and external server (to be defined)\* in order to check for the user’s right to access to the requested directory. Second, it communicates with the HDFS in order to list all the files contained in an HDFS directory that was requested by the user.

\*The ranger policies will be queried and uploaded to this external server under the authentication of an authorized admin (i.e. Wandrille Domin).

### Functional requirements

**ID: FR1 - TITLE: Authentication**

* Given the application code (code IRT), the platform should be able to look up for the corresponding directory in the HDFS of this application using the data catalog, which will be referred to as “the application directory”.
* The administrative user will interact with the platform using a generic administrative ID. The interest of using the administrative ID is that it gives the user who doesn’t have the right to write to a directory the right to do so. Given two inputs: **administrative ID** and **application diretory**, the platform should first connect to LDAP to check in which group the user belongs to. It should then look up in the ranger policies to see if the group has the right to access to the requested directory in the HDFS.Any subsequent process should fail if the authentication fails.

**ID: FR2 - TITLE: Search**

* If the user is authorized to access to the requested application directory, he should be able to see all of the files contained in this directory, which will be referred to as “the search results”.
* Search results can be viewed in a list. Each element in the list represents a specific file in the directory of the application in the HDFS.
* There should be a maximum of 14 results displayed. More results can be view using a scrolling bar.

**ID: FR3 - TITLE: Create and register internal table**

* Internal table should be created for each file in the search results. The input file formats supported are txt, csv, json, xml and ccb
* The user should be able to view the internal table when clicking on the file name in the search results. He should also be able to select the columns that he wants to register.

**ID: FR4 - TITLE: Write the table to the project directory**

* The user should be able to choose the output file format that he wants to write to his project space. The supported file formats are txt, csv, json and xml
* After the user selected the desired tables, columns and the output file format, he can click the “subscribe” button. The “subscribe” button immediately add the selected files into the user’s project space. It will also initialize the process schedule with oozie.

**ID: FR5 - TITLE: Schedule the process with oozie**

* The application directory registered by a user should be checked according to the frequency listed in the data catalog. When new data arrives at the registered directory, its subset(s) will be transferred directly to the user’s project space, according to the subscription detailed made by that user. One user can subscribe to several data subsets.