**DOCUMENTATION FOR FEATURE : Import cheque file used in CS Module (Cheque Santé Module) IN BACKEND**

**Overview**:To make an import file from a file field in a web page using form data, we need to upload the file first, serialized it if needed, and stored into the storage destination field designing in database. here are the follow steps to import a file :

1. **PERFORMING THE UPLOAD FILE** (USING DJANGO REST FRAMEWORK)
2. Define in **views.py** file a function that could allow the upload of file. In our case it is called **upload\_cheque\_file (*request*)** that takes request in parameter to upload the file .
   1. The function calls a serializer class (in serialize.py file) named **UploadSerializer(*data=request.data*)**that takes the requested data in parameter and set it into the serialize object: **serializer**
   2. Verify if the serializer object is valid : **serializer.is\_valid(*raise\_exception=True*)**
   3. Use the try ... Except to check whether an error raised up about the format of file or anything else, when calling the function **upload\_cheque\_to\_db(*request.user, file*)** to store the uploaded file, and insert the current user into corresponding field of the table
3. In the urls.py file, set the route that could be used when making post to upload the file.
4. **STORING THE FILE , CREATION OF ROW INTO CHEQUEIMPORT TABLE AND INSERTION OF DATA FROM FILE WITHIN CHEQUEIMPORTLINE TABLE**
5. Storing the uploaded file , creation of row into **ChequeImport** table

From **models.py** file,define a function that could insert data into **ChequeImport** table and **ChequeImportLine** table in same time. In our case, this function is: **upload\_cheque\_to\_db(*request.user, file*);** it is called in **upload\_cheque\_file (*request*)** in views.py file. It takes in parameter the user and the file.

1. use the try ... Except to check whether the file to be inserted is in good format; if yes,create a row in **ChequeImport** table with : **chequeImportCreated = ChequeImport.objects.create(*user=user, stored\_file=file***)
2. then create into **ChequeImportLine** table row, the data from the file

by doing: **tableChequeToImport = insert\_data\_to\_cheque\_line(*chequeImportCreated.stored\_file, chequeImportCreated*)** which takes the stored file and the ChequeImport id created. This function will allows the insertions into **ChequeImportLine** table.

1. Insertion of data from stored file within **ChequeImportLine** Table

From models.py file, define a function which will insert the data from stored file into **ChequeImportLine** Table.In our case, the function is **insert\_data\_to\_cheque\_line(*csv\_file, chequeImport***). it is called in insert\_data\_to\_cheque\_line function to insert the data. This function take file, and the id associated to the stored file in **ChequeImport** table.  
 a)calls first a function named **parse\_csv\_file(*csv\_file***) which takes the file in parameter and parses it into the correct format taht could be inserted.

b- then calls the loop and iterates over each row of parsed file.In every row, check if the row contains the correct status in the **CheckStatus** column, if not, an exception will be raised as : **logger.exception("*Import Cheque Statut anormal :*").** if the row contains the status listed, then check whether the cheque number in the file already exists in the **chequeImportLineCode** attribut of **ChequeImportLine** table with: **if ChequeImportLine.objects.filter(*chequeImportLineCode=row['NumCheque']*).exists():** if yes, update the status. If not, create or insert the new cheque number,the status and the **idImport** sent before, into the **ChequeImportLine**  table.

You can get more information about the process by following this link: <https://github.com/mngoe/openimis-be-cs_py/tree/develop/cs>

1. **HOW TO INSTALL THE OPENIMIS-BE\_PY APPLICATION (MAIN MODULE)**

* clone this repo (creates the **openimis-be\_py** directory)
* For example, By running on terminal or the git bash terminal, the command:
* **git clone https://github.com/mngoe/openimis-be\_py.git**
* install python 3, recommended in a [virtualenv](https://virtualenv.pypa.io) or by using your IDE (**Pycharm or Visual Studio Code**)
* install [pip](https://pip.pypa.io)
* within **openimis-be\_py** directory
  + install openIMIS (external) dependencies: **pip install -r requirements.txt**. For development workstations, one can use **pip install -r dev-requirements.txt** instead for more modules.
  + generate the openIMIS modules dependencies file (**from openimis.json config**): **python modules-requirements.py openimis.json > modules-requirements.txt**
  + if you previously installed openIMIS on another version, it seems safe to uninstall all previous modules-requirement to be sure it match current version **pip uninstall -r modules-requirements.txt**
  + install openIMIS current modules: **pip install -r modules-requirements.txt**
  + Configure the database connection After installing your database management system (sql server for example) and creating your database (**openimis\_db** for example) you have to configure to get access to your application.  
    create a **openimis-be\_py/.env** file to provide your database connection info: DB\_HOST=mssql-host-server (localhost or 127.0.0.1 in our case)  
    DB\_PORT=mssql-port (1433)  
    DB\_NAME=database-name (Your database name)  
    DB\_USER=database-user (openimis in our case)  
    DB\_PASSWORD=database-password (your database password)
  + start openIMIS from within **openimis-be\_py/openIMIS**: **python manage.py runserver**

At this stage, you may (depends on the database you connect to) need to:

* apply django migrations, from **openimis-be\_py/openIMIS: python manage.py migrate**
* create a superuser for django admin console, from **openimis-be\_py/openIMIS: python manage.py createsuperuser** (will not prompt for a password) and then **python manage.py changepassword <username>**

1. **HOW TO INSTALL CS MODULE WITHIN THE MAIN MODULE OPENIMIS-BE\_PY**

If not yet added within openimis-be\_py module (main module), you may add it after cloning it from the repository.

· from **/openimis-be\_py/openIMIS**: register your module in the pip requirements of openIMIS, referencing your 'local' codebase: **pip install -e ../../openimis-be-cs\_py/**

· register your module to openIMIS django site in **/openimis-be\_py/openimis.json**