Oxford University Clinical Research Unit

Serum bank application : Instructions manual

Project Manager:
Hannah CLAPHAM

Author:
Tristan MAUNIER

Version 1.0 : August 2018

Link to the project's GitHub repository :
https://github.com/tmaunier/sboucru

oucru

Contents

1 Installation **3**
- 1.1 Requirements . . . 3
  - 1.1.1 Python and MySQL . . . 3
  - 1.1.2 Virtual environment . . . 3
- 1.2 Installation . . . 4
  - 1.2.1 Set the project . . . 4
    - 1.2.1.1 Virtual environment . . . 4
    - 1.2.1.2 Git clone . . . 4
    - 1.2.1.3 Requirements . . . 4
  - 1.2.2 Database . . . 5

2 Start the application **6**
- 2.1 Command lines . . . 6
- 2.2 Script bash and Alias . . . 6

3 Import data **8**
- 3.1 Import data . . . 8
  - 3.1.1 File templates . . . 9
  - 3.1.2 Errors report . . . 9
  - 3.1.3 Import test results . . . 9
- 3.2 Undo import . . . 9

4 Export data **10**
- 4.1 Initial dataset . . . 10
- 4.2 Apply filters . . . 11
  - 4.2.1 Serum filters . . . 11
  - 4.2.2 Freezer filters . . . 11
  - 4.2.3 Tests filters . . . 11
- 4.3 Dataset validation . . . 12
  - 4.3.1 Validate or Modify your dataset . . . 12
  - 4.3.2 Check sera's status . . . 12
  - 4.3.3 Count results . . . 12
- 4.4 Data selection to export . . . 13
  - 4.4.1 Columns selection form . . . 13
  - 4.4.2 Output file type . . . 13

5 Other features **14**
- 5.1 Data visualization . . . 14
- 5.2 Latest import screen . . . 14
- 5.3 Modify serum's location . . . 14
- 5.4 Switch serum's status . . . 14

5.5 Tables tab 15

6 Administration site **16**
6.1 Users and groups of users management 17
6.1.1 Create an user 17
6.1.2 Edit user's information 17
6.2 Database and data management 18

7 Scenarios : How to ... **19**
7.1 Get sera's location 19
7.2 Add sera in the database 20
7.3 Check test results and status information 20

Chapter 1

Installation

The following installation instructions are adapted to Unix system only i.e. Mac OS and Linux distributions, if you are working on Windows please refer to django's documentation.
However the software's requirements are the same from one system to another, the project is able to running on any system once all the requirements are installed.

1.1 Requirements

1.1.1 Python and MySQL

- This project is written in python 3, version 3.6, it should be able to work with every python version newer than this one. Most of the time python is already install, you can check it opening a terminal and typing this following command line :

  `python -V  or python3 -V`

  If python is well installed you should get this kind of answer :

  `Python 3.6.2`

  Otherwise please refer to the official python website and install python on your machine. *https://www.python.org/downloads/*.
  Pip is the package installer for python3, it's necessary to set up this project so you can check if it's installed just as python with this command line :

  `pip -V`

- MySQL is the selected query language in this project it has to be installed on your computer to make the project work. Installing MySQL takes time and the installation varies according to your computer, there are a lot of good documentation online so it won't be explained further here. Once you are able to create a new user and a database in your terminal through the mysql interface (In the Shell, you don't need to install MAMP or LAMP for this project) then you can go to the next step.

1.1.2 Virtual environment

This project requires specific versions of different packages, if you update one of these tools the project may be broken. The solution is to create a virtual environment in which you will install the requirements and set the project in, so you can update or install any of those packages anywhere else on your computer.

1.2 Installation

1.2.1 Set the project

First you have to create a folder, I advice to name it "Sites" so every similar project could be store in this folder as well.

1.2.1.1 Virtual environment

In this folder you'll install the virtual environment necessary to save the right requirements for the application. Type the following command lines in your shell :

```
pip3 install virtualenv
```

Once it's well installed, you have to create one environment :

```
virtualenv -p python3 ENVpy3
```

"ENVpy3" is the name of the folder in which you'll have the files corresponding to this environment. Usually it corresponds to the path where you want to set the environment but to make it easier you can write this command line from the right folder directly. I choose this name because it's explicit so you can remember that it corresponds to the virtual environment set in python3 but you are free to choose the one which suits you best.

1.2.1.2 Git clone

Once you have created the virtual environment you should have a folder "Sites" (or whatever the name you gave) and in this folder another folder named ENVpy3. At this point it is the moment to clone the git repository of this project :

```
git init
git clone https://github.com/tmaunier/sboucru
```

Check that now you have two folders in "Sites, a new one called "sboucru" appeared.

1.2.1.3 Requirements

If you open this new folder "sboucru", you'll find a file called requirements.txt in which are stored every packages and versions needed for this project. You don't have to install those packages one by one, pip has a command for this situation :

```
pip3 install -r requirements.txt
```

If some errors occur during this step please refer to the documentation of the corresponding package.

1.2.2 Database

The last step of the installation is to connect the project to the database.

- Open a terminal, start mysql.
- As root create a new database
- Create a new user and grant him all privileges on this database.
- Admins will provide you a backup sql file, as the new user make a copy of this backup to your new database
- Once your database is set on mysql, open this file on your text editor : *sboucru* > *sboucru* > *settings.py*
- you'll find these following lines

```
DATABASES = {
    'default': {
        'ENGINE': 'django.db.backends.mysql',
        'NAME': 'database_name',
        'USER':'username',
        'PASSWORD':'*****',
        'PORT':'', #example : 8080
        'HOST': 'localhost',
    }
}
```

- Replace "database_name" by the name of your database, "username" by the user name you just created, "*****" by the corresponding mysql password and fill the port field with an available localhost port ("8080" should work most of the time).

Congratulations the installation is done.

Chapter 2

Start the application

In this section you'll see how to start the application with command lines and then how to create a file that allow you to start it with a simple alias on your terminal.

2.1 Command lines

In your terminal, go to the "Sites" folder containing the project and the virtual environment

```
cd ENVpy3
source bin/activate
cd ../sboucru
python manage.py runserver 8080
```

Once the project is started, you can open your browser and type the following adress :
localhost:8080
on your navbar. You should be able to see the login page of the application appearing on your screen.

2.2 Script bash and Alias

If you are not confident with command lines you can also create a file with those commands inside and create an alias which permits you to start the project only thanks to a keyword. In your Home directory (symbolized by $\sim$ in the shell), create a new file named *sbo.sh* for example and copy these following commands inside:

```
cd Documents/Sites/ENVpy3
source bin/activate
cd ../sboucru
python manage.py runserver 8080
```

When you'll ask to your machine to run this script it will automatically run every lines included into the file.
The second part of the job here is to create an alias which is a keyword that you can use to execute a command in the shell. Here the aim is to use the keyword 'serum_bank' to run the previous script, thus every time you want to start the application you just have to type 'serum_bank' in the shell.

Still in your Home directory :

- In Linux, you have to open the file named *.bashrc* in a text editor as root.
- In Mac OS, you have to create a file named *.bash_profile* (if it's not created yet).

Be careful with this file it's probably one of the most important file in your system, if you follow exactly what is saying here you won't have any problem.
Just insert this following line in the file :

```
alias serum_bank='. ./sbo.sh'
```

Save and close the file. Then you have to source the file to make the alias work. In a new shell, type :

```
source ~/.bashrc
or
source ~/.bash_profile
```

It should work now, type 'serum_bank' in a new terminal and check that the development server is well started. You can then open your browser and go to *localhost:8080*.

Chapter 3

Import data

Import data is one of the main feature of this application, you are able to import different type of data, check latest import or undo an import.

3.1 Import data

In this version of the application, you are able to import sera, sera's location, Elisa test results and Protein microarray test results. In each of these case you'll have to upload a file, the program will read the information inside and distribute the data into the database. This functionality is based on the fact that the program is able to recognize the file headers as well as the format of the data contained in the file.
I draw your attention here to the fact that it is extremely important to provide readable data to the program in order to successfully import data. By readable data I mean that the program must be able to recognize the headers of the document, for which a template file system is available and also that the file does not contain inconsistent data (for example a word in the age column) .
Every *Import pages* look like the following figure :

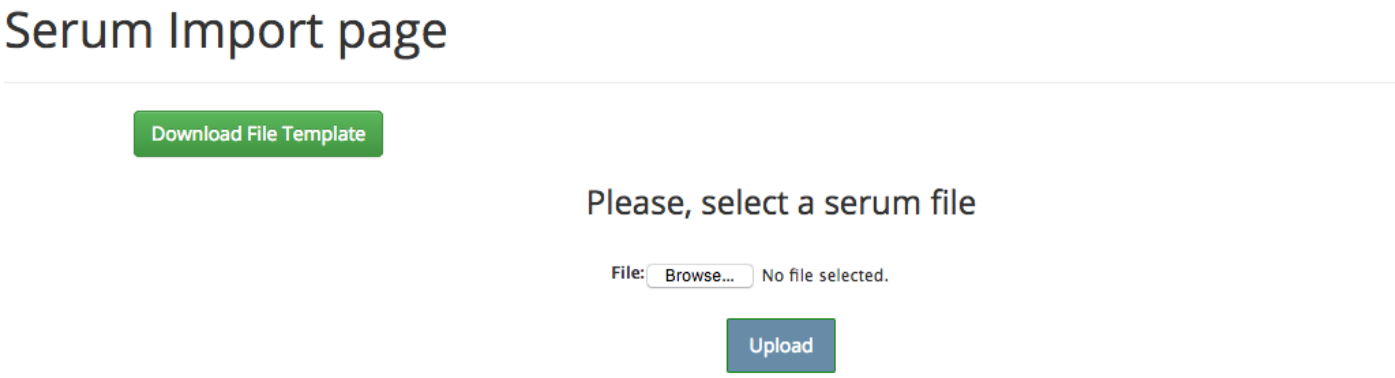

Figure 1: Import page template, serum import page as example

3.1.1 File templates

As you can see in the previous figure (Figure 1) there is a direct link (a green button) to download a file template in every import page. When you click on this button, you will be redirected to a new page where you can select any type of file which interest you and download it in different file types. This feature allows you to use a file with the right headers to import your data.

3.1.2 Errors report

After uploading a file on an import page, it may happen that some errors occur on some lines. If it's the case a report will appear on the page showing you the errors. The sera concerned by this report won't be imported but the others will.
In case you find that some of your data has not been imported and didn't appear in the report, you should look closer your data in the file, the data may be written in a wrong way or some characters may be present in wrong cells (be careful of blanks, dashes and dots).
After correcting the errors on your file you can upload it again, only the data that doesn't exist yet in the database will be added.

3.1.3 Import test results

You can import test results in exactly the same way than the other types of data but it exists some points that you have to be aware of. In this version of the application you are able to import Elisa and Protein Microarray tests results, but Elisa and PMA results files are quite different. Results for every pathogens tested with a PMA test are stored in the same file in contrast to Elisa tests results where every pathogen tested has its own file with specific headers. If you want to import Elisa test results you have to specify first which pathogen data you want to import.

3.2 Undo import

It seems that making an import mistake would be very punishing if you couldn't be able to undo it. So this feature exist in the application, but to preserve security and the integrity of the database, users have to respect some rules. Indeed an user is able to erase data only if he imported it previously, he also have to specify the date of the import and an approximate time. With this feature you can only erase data if you imported it in the last 3 days, otherwise you can make a special request to an administrator who is able to erase everything.
If you spent your day importing data you'd be happy not to erase everything you import on the same day that you made the mistake. So in order to save a part of your time of work, you have to specify a time range of 2 hours in which you commit your mistake and only data imported in this range of time will be erased from the database.

Chapter 4

Export data

The export of data from a database consists in collecting data of interest by means of queries, in this project the user will be able to interact with the database through forms allowing him to apply filters accurate and thus to obtain a dataset corresponding to his needs.

4.1 Initial dataset

When you want to export some data from a serum bank, the thing is about to know which sera are concerned about your request. There are two scenarios possible :

- First, you don't know how many and which sera are concerned about your request. Then the thing to do is to skip this step 1, it means that you consider as your initial dataset every sera existing in the database. From this dataset you are able to apply filters to get the data you need to work on.
  Note : In the case you want the dataset containing every single serum stored in the database, you can skip every steps of this page as well and submit your dataset in the last tab.

- The second scenario is if you know exactly on which sera you want to work on, then this step 1 is made for this case. You can upload a list of sera from here to make it your initial dataset instead of considering the whole database as it.
  Notes :
  - You can still apply filters on your dataset after uploaded it during the step 1.
  - Be careful with the sera you upload at this step, only sera that already exist in the database will be taken into account.

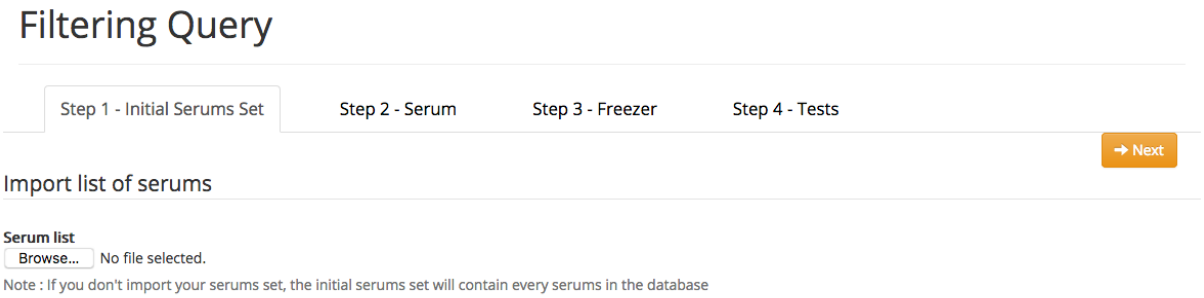

Figure 2: Define the initial dataset

4.2 Apply filters

Once you have defined your initial dataset, you can apply different kind of filters on it in order to selecting the sera you are interested on.

4.2.1 Serum filters

This first filtering form allow the user to restrict his dataset according to the different parameters linked to any serum. Indeed all sera contained in this database have related information of the same type, which makes sorting these data possible.

4.2.2 Freezer filters

The freezer step could be useful in case you want to have information on sera localized in a specific place in freezers. A scenario could be that you dropped a box and every samples are not usable anymore, here filtering with the id of this box you can have an output list containing the sample ids of the sera which were stored in that box. And then thanks to this list you can switch their status to unavailable (ref. *5.4 Switch serum's status*).

4.2.3 Tests filters

Sometimes you will need to know which test has been done on which sera or which test hasn't been done on which sera. This following form (*Step 4 - Tests*) allows you to apply filters on your dataset depending on which test has been performed or not.

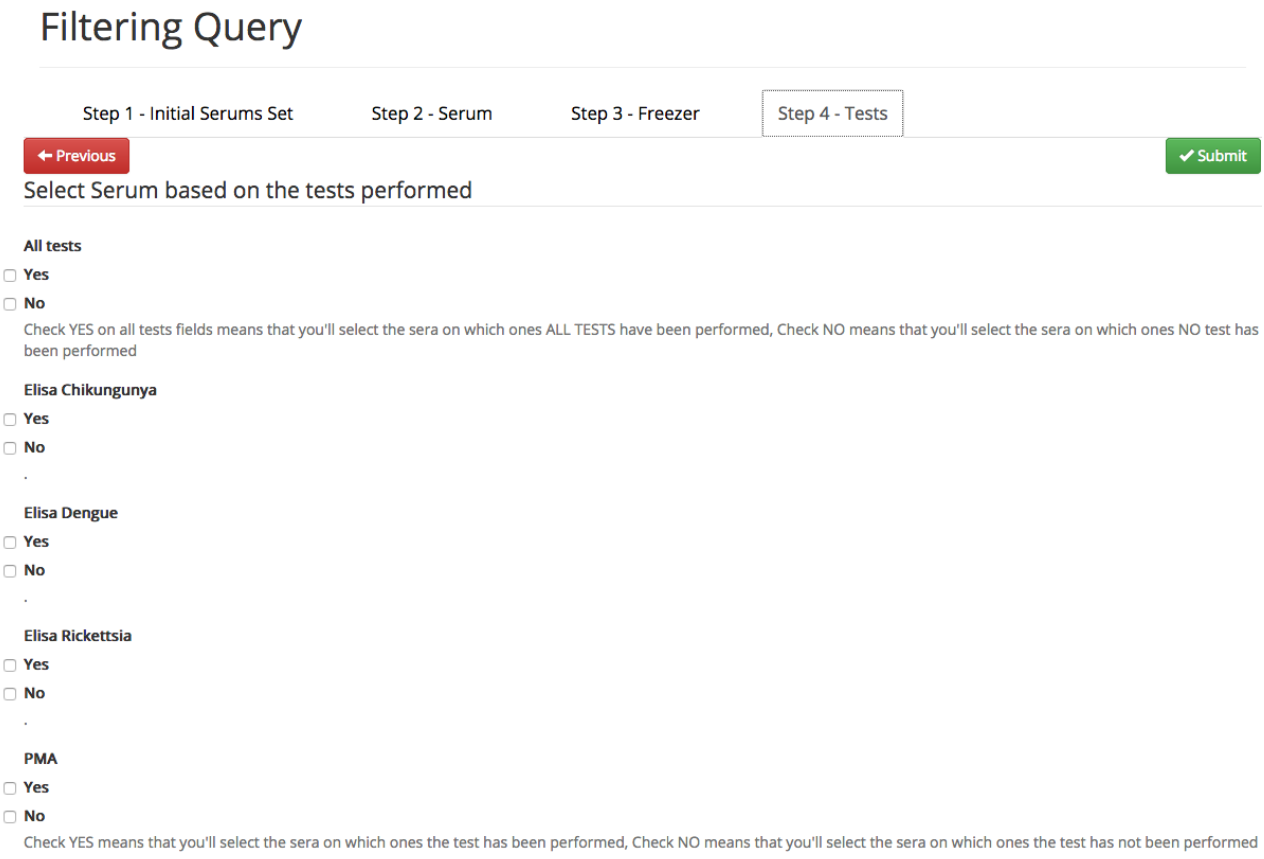

Figure 3: Apply filters depending on the tests performed

4.3 Dataset validation

Once you've submitted the dataset form, a graph will show you how many sera are included in your final dataset.

4.3.1 Validate or Modify your dataset

At this step you can either modify your dataset coming back to the filtering forms if you decide that is not exactly what you needed or you can validate it and keep going to the export page.

4.3.2 Check sera's status

You can also display a table showing all sera from your dataset and their status. It's possible for you to download this table as well in order to keep a track of this information.

4.3.3 Count results

In the same idea as the previous feature, you can display a table showing all sera from your dataset and a count of the tests which have been performed on each of these sera. Different counts are displayed according to the nature of the test and the total count is also calculated. Downloading this table is possible too.

4.4 Data selection to export

After your dataset is validated, you can choose which data connected to those sera will be written in your output file.

4.4.1 Columns selection form

You can choose column by column which information you want to include in your output file. By default all information contained in the database linked to the sera in the dataset are included but you can uncheck data you don't need or select more precisely the one you really need.

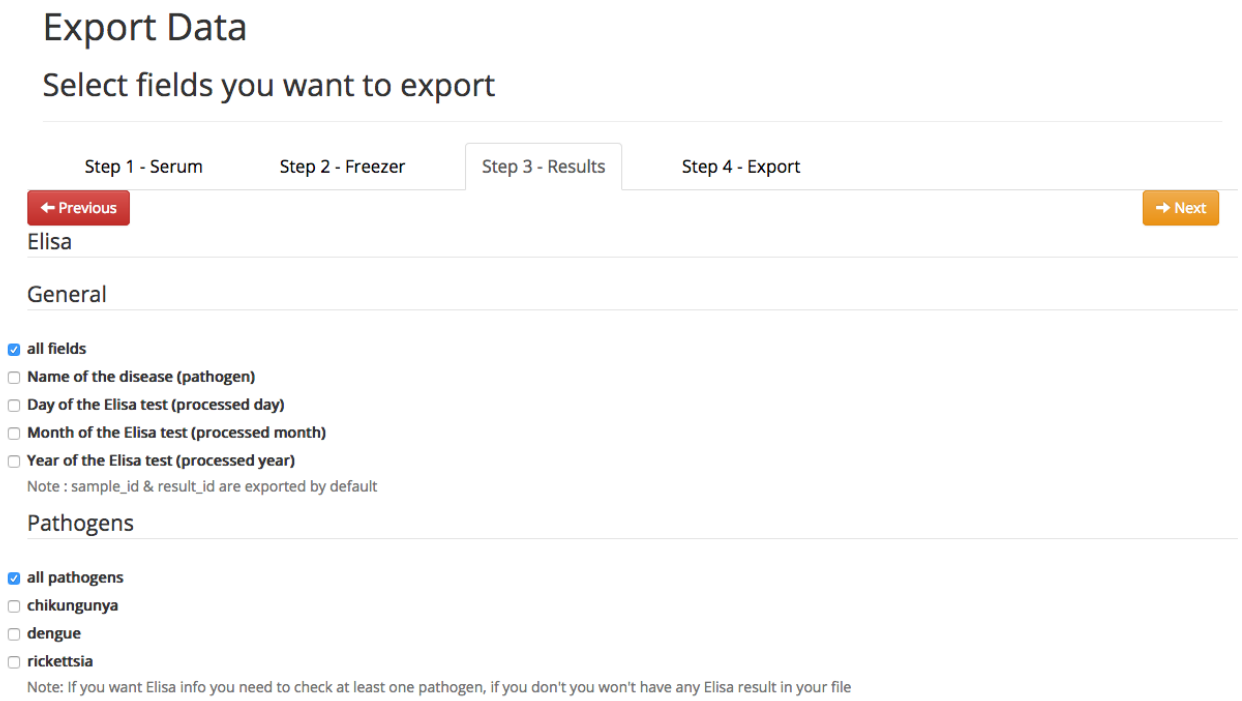

Figure 4: Selecting columns to the output file

4.4.2 Output file type

This application supports multiple file formats whether input or output. Among these types of files are csv, xsl, xlsx and ods.
The output file is a multi-sheets file, it means that the data are separate in different sheets in the file to make it readable. Sheets will be present or not in the file depending on the boxes you checked before to click on the export button. **Note :** Unfortunately in this version of the application at this step the csv format is obsolete, tags are included into the output file that make it unusable except using a script that can extract the data avoiding the tags.

Chapter 5

Other features

In addition to the main features such as data import and export, this application has additional features that enhance the user experience and others that seem essential for the long-term use of this tool.

5.1 Data visualization

Indeed when you logged in your session, you will see a histogram on the screen. This graph corresponds to the number of serums stored in the database, classified by years and by provenance site. This histogram is interactive, you have access to specific information hovering your mouse cursor on a data bar and every time you refresh the page, this graph is updated. This can be useful for tracking the evolution of the contents of the database.

5.2 Latest import screen

On the same page as the previous graph you have an overview of the last imports made in the database, sorted by data type. You can find out the amount of data involved, the date of import, and the user who made it.

5.3 Modify serum's location

This feature is essential for the work of monitoring serum stocks in freezers. It works exactly the same way than importing serum's location, except that this time instead of checking that the sera concerned don't exist in the database, this time the program will check that the sera concerned exist in the location table and will update its location with the new data imported here.

5.4 Switch serum's status

Like the previous one, switching serum's status is an essential feature for the work of monitoring serum stocks. In case that a serum is not available anymore physically in the freezers, users have to be able to know that he can't make more tests with this sample. Uploading a list of sera with this function allow the user to switch the status of those samples. Note that switching status means that you can make a serum available again if you upload its id through this function.

5.5 Tables tab

In this page, an user can look at the whole site and ward tables. Moreover 20 random sera are displayed in a table, it gives you an overview of how the serum and freezer tables look like in the database.

Chapter 6

Administration site

This website is accessible only to administrators of the application. This tool is automatically generated by django when creating a new project. From this interface you can have access to the whole database including the data, the tables and users information. If you plan to use this interface you must be extremely cautious when using it. Indeed as administrator you have all the powers on the data stored in the database, it means that a difficultly reversible accident can quickly happen. The interface is as follows :

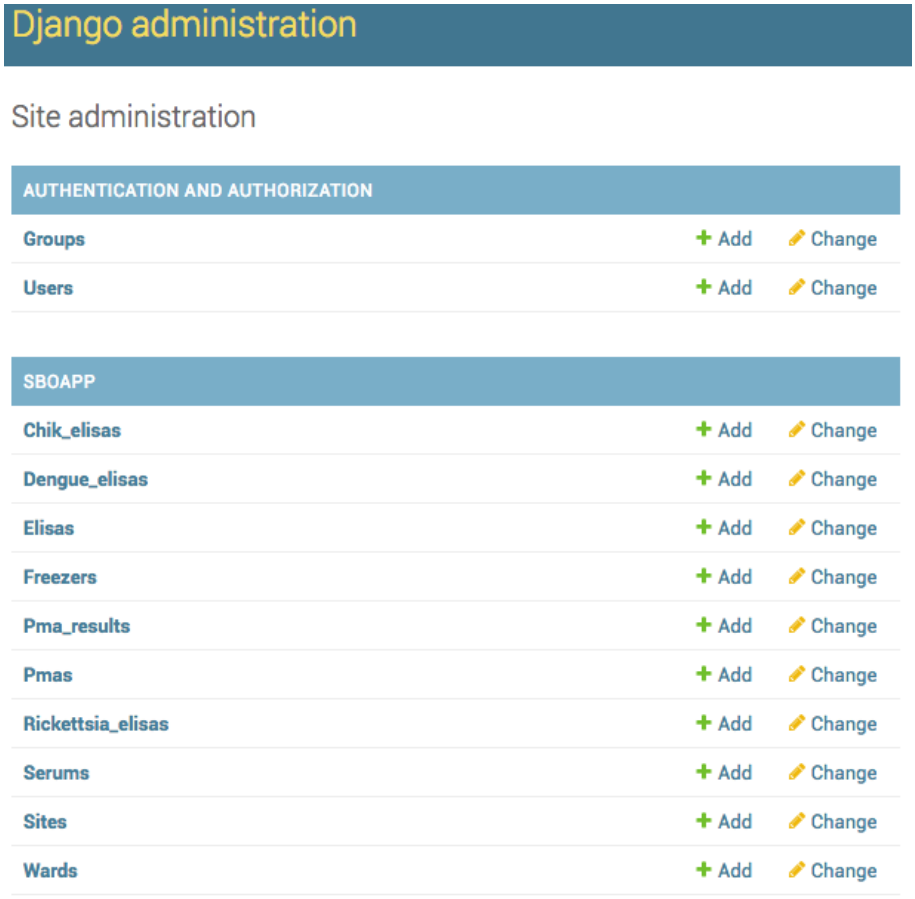

Figure 5: Site administration interface

6.1 Users and groups of users management

In the first section named *Authentication and Authorization* you can manage users and group of users. Creating a group of user allow you to define permissions for several users at the same time. At this moment only two groups exist, **OUCRU_staff** which allow users to log in the application and use the different features available and **Admins** which give access to the administration site in addition to the same permissions than the other group.
In this section you are also able to create, edit and delete users (not recommended).

6.1.1 Create an user

When you are in the Users's page of the administration site, you'll have access to the list of the existing users, a filter panel and a *Add user* button located in the top right corner of the page. Clicking on this button, a classic form asking an username and a password

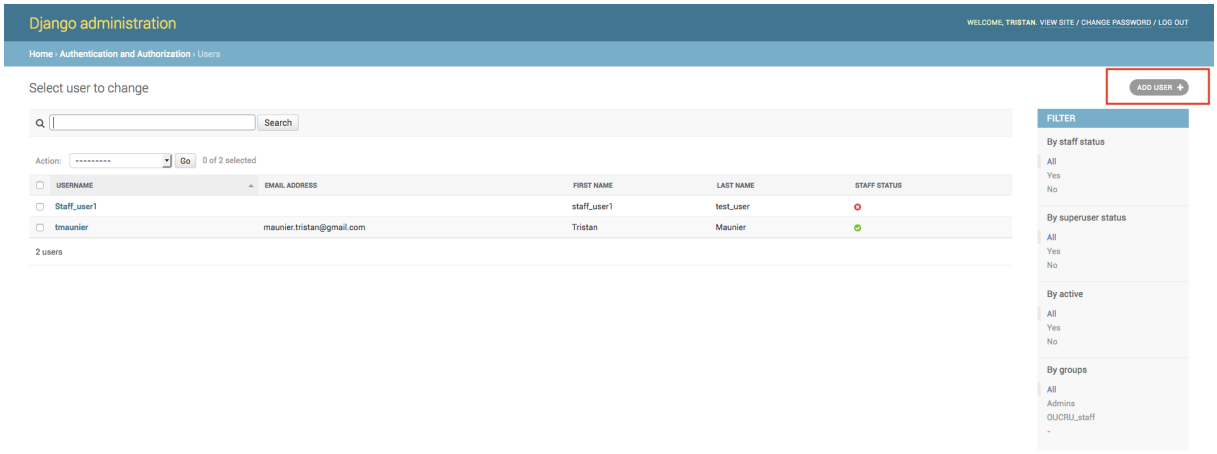

Figure 6: Users page

for the new user will be displayed. After creating a new user you can set his permissions clicking on his name on the users list.

6.1.2 Edit user's information

As an administrator you can change every information linked to a specific user, but also set his permissions and assign him a group. You also have access to the date and time of this user's creation and last login. There are three interesting parameters in the *permissions* section of this page :

- Active, it allows the user to log in the application and use the different features. You should uncheck this parameter instead of delete user.
- Staff status, it gives access to the administration site.
- Superuser status, a superuser has access to everything even if permissions are not allowing him to do something. You should define one only superuser per project so if you want to restrict admins on specific actions, they can't ignore permissions.

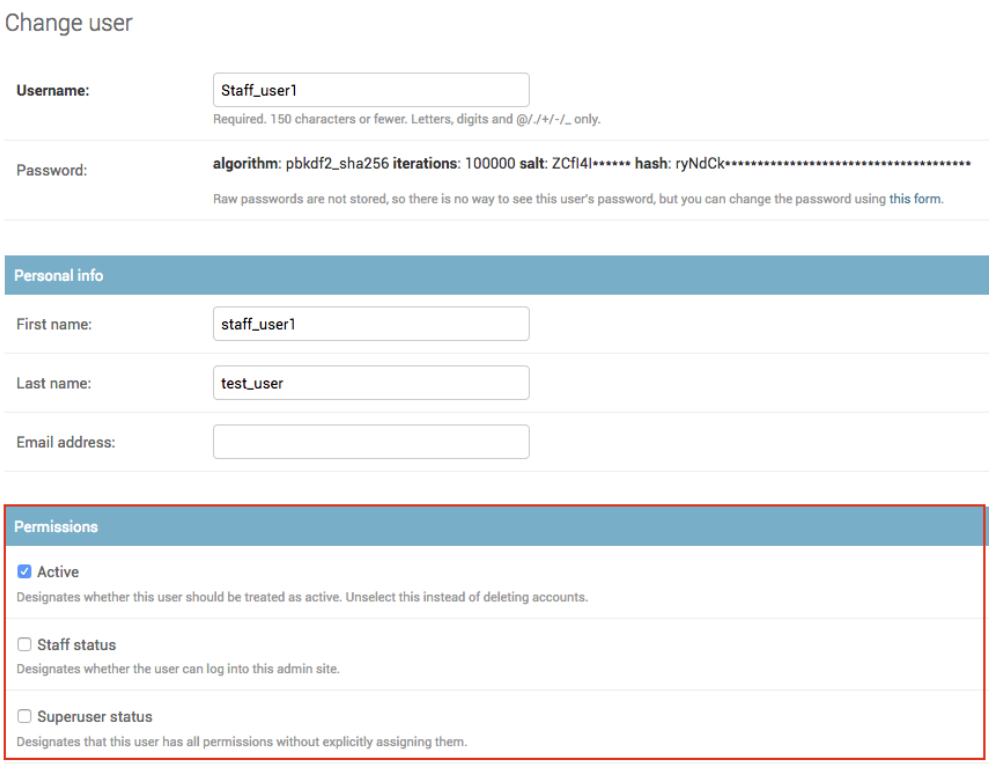

Figure 7: User's information

In this project if you create a staff user check only the *Active box* and if you create an admin user you decide if you give him a superuser privilege or not but at least don't forget to allow him the right to have access to the administration site.

6.2 Database and data management

In the second section of the administration site called *SBOAPP (Serum Bank OUCRU application)* you can browse all the database's tables. In every table you can add, edit or delete an object, I highly advice not to execute any of these actions here except if you really need to edit or delete manually some data. I insist on the fact that you must be careful when you are using this interface, a mistake can cause terrible consequences.

Chapter 7

Scenarios : How to ...

These scenarios are only general cases that you may encounter when working with this application, you can adapt these instructions to suit your situation.

7.1 Get sera's location

Scenario : You are working in the lab and you have a list of sera on which you want to make a test. You have to take them from the freezers so you want their location.

1. Log in the application
2. Prepare your list of sera in a spreadsheet. The file template is just composed of one column *Sample_id* with the list of the ids
3. Go to *Data Manager* > *Sort & Export Data*
4. Upload your list of sera in the *Step 1 - Set Serums set* tab
5. Ignore the other tabs and submit
6. On the *Queryset Validation* page, verify that the graph correspond to your dataset, if it's the case click on the green button *Validate Queryset*. Otherwise check that your file does not contain any errors and try again.
7. In this case you are interested about the location of your sera, so uncheck all the boxes in the (*Serum and Results*) tabs and go to the *Freezer* tab. Here select the information you need thanks to the form and when you are ready go to the *Export* tab.
8. Choose your favorite file type and export your data.

Note : You can also set your initial dataset with filters directly through the application without uploading a list of sera.

7.2 Add sera in the database

Scenario : You just received a new set of sera from an hospital and you want to import it in the database.

1. Log in the application
2. Go to *Import Data* > *Import serum*
3. Open the file and check that the headers correspond to the right template. You have access to this template through the *Download file template* page (if you have any issue with this step, please refer to the *File templates* part of this document).
4. Once the file has the right headers, you can upload it.
5. The sera has been imported in the database, you can return to the dashboard Welcome tab and look in the latest import section if the information has been updated.

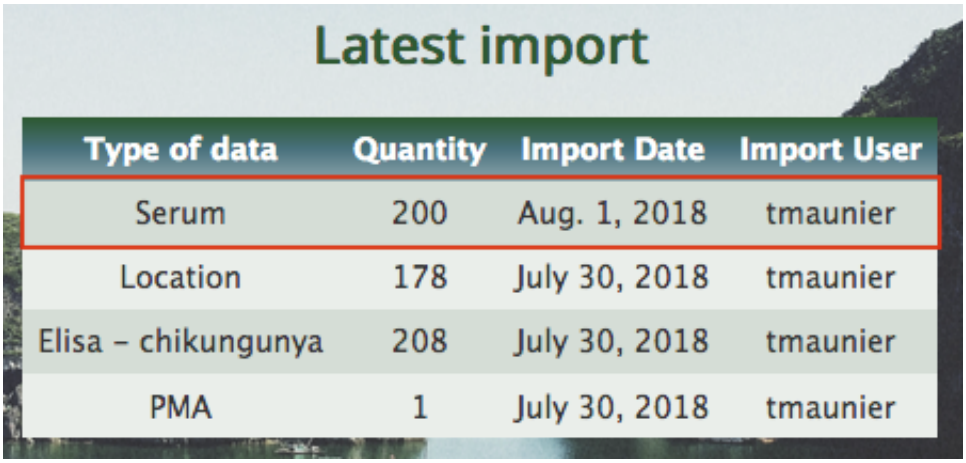

Figure 8: Latest import table

7.3 Check test results and status information

Scenario : You want to work on a sera dataset based on the tests performed and their status. In this example you want to know on which sera an Elisa - Chikungunya test has been done and if they are available or not.

1. Log in the application
2. Go to *Data Manager* > *Sort & Export Data*
3. It's time to set your dataset, you have 3 possibilities:
   (a) Select all of them → ignore Step 1, 2 , 3 and 4 and submit
   (b) Upload a list of sera → Step 1
   (c) Sort from the complete dataset stored in the database to your sera of interest → Step 2, 3 and 4

4. In this case you want to select only the sera on which an Elisa -Chikungunya test has been performed, so you have to follow the third choice (c).

5. Then go to the *Step 4 - Tests* tab and check the box 'Yes' in the 'Elisa Chikungunya' section. Checking this box means that you want only sera that have been tested with an Elisa - Chikungunya test.

6. On the *Queryset Validation* page, you want to know if those sera are still available or not in the freezers, so you can click on the *Check serums status* button. On this page you can either look at the information you need directly or either download a file in which the same information will be written. Once you're ready go to the previous page, validate your queryset and go to the *Export Data* page

7. On the *Export Data* page you have to select the column you want in your output file. The first thing to do is to uncheck the boxes corresponding to the data you don't want. Obviously here it is very important not to forget to select the results corresponding to the Elisa test and to the desired pathogen (Chikungunya in this example) → *Step 3 - Results* tab.

8. You can finally choose your favorite file type and export your data.

Note : If you wanted to work only on the sera which are available you could also apply the status filter (→*Filtering query page, Step 2 - Serum tab*) on your dataset.

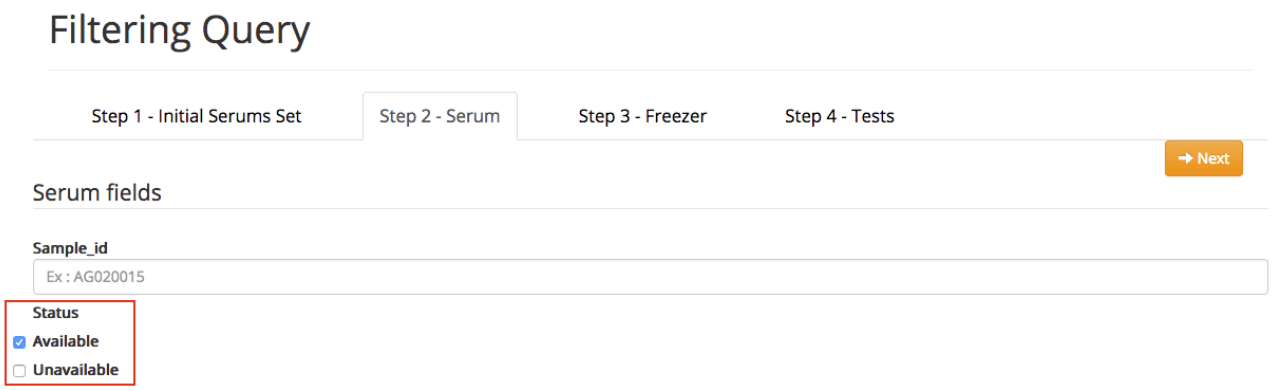

Figure 9: Available sera selection