

Proline Installation and Management Guide

Release 2.3.0

Table of content

Table of content	1
A. Introduction	4
I. Architecture Overview	4
II. Requirements	5
B. Installing Proline : Automatic process	6
I. Installing Proline Server	6
Postgres & create a user proline	8
II. Setup Proline (using Admin GUI)	8
Launch Proline-Admin GUI install	9
Select configuration component	9
Proline Server Configuration	9
PostgreSQL	9
JMS Settings	10
Mount Points	11
Proline Module Configuration (sequence repository)	12
Currently the only configurable module is sequence repository.	12
PostgreSQL	12
JMS Settings	12
Parsing Rules	12
Summary	13
Installation Result	14
III. Installed scripts	14
C. Installing Proline : Manual process	14
I. Installing Proline Server	14
II. Configuring Proline manually	15
Proline Admin	15
Configuration files	15

Proline Server	16
Configure the Datastore	16
Configure the mount-points	16
Configure the Messaging connection	17
III. Setting up Proline Datastore with command line	17
the command line	17
on Windows systems	17
on Linux systems	17
D. Upgrading from a previous version	19
I. Upgrading Proline Server	19
II. Updating the Datastore	19
Backup Datastore	19
From the command line	20
on Windows systems	20
on Linux systems	20
From the graphical interface	20
III. Upgrading Sequence Repository	22
IV. Upgrading Proline Studio	22
E. Running Proline Server	23
I. Running the Server	23
Server Memory Setup	23
Running on Windows	23
Running on Linux	23
F. Sequence Repository	24
I. Sequence Repository Configuration	24
Server and Datastore description	24
Protein description parsing rule	25
II. Testing rules	26
III. Running Sequence Repository	27
Running as Service Daemon	27
Running as Service On Demand	28
G. Monitoring using Proline Admin GUI	29
Launch Proline-Admin GUI monitor	29
The graphical interface in details	29
Proline datastore properties summary	29
I. Proline user management	30
Create a new Proline user	30
Add to user/admin group	30
Reset Password	31
II. Proline project management	31
New Project	31
Activate/Disable Project	32
III. Proline databases management	32
Edit DB parameters	32

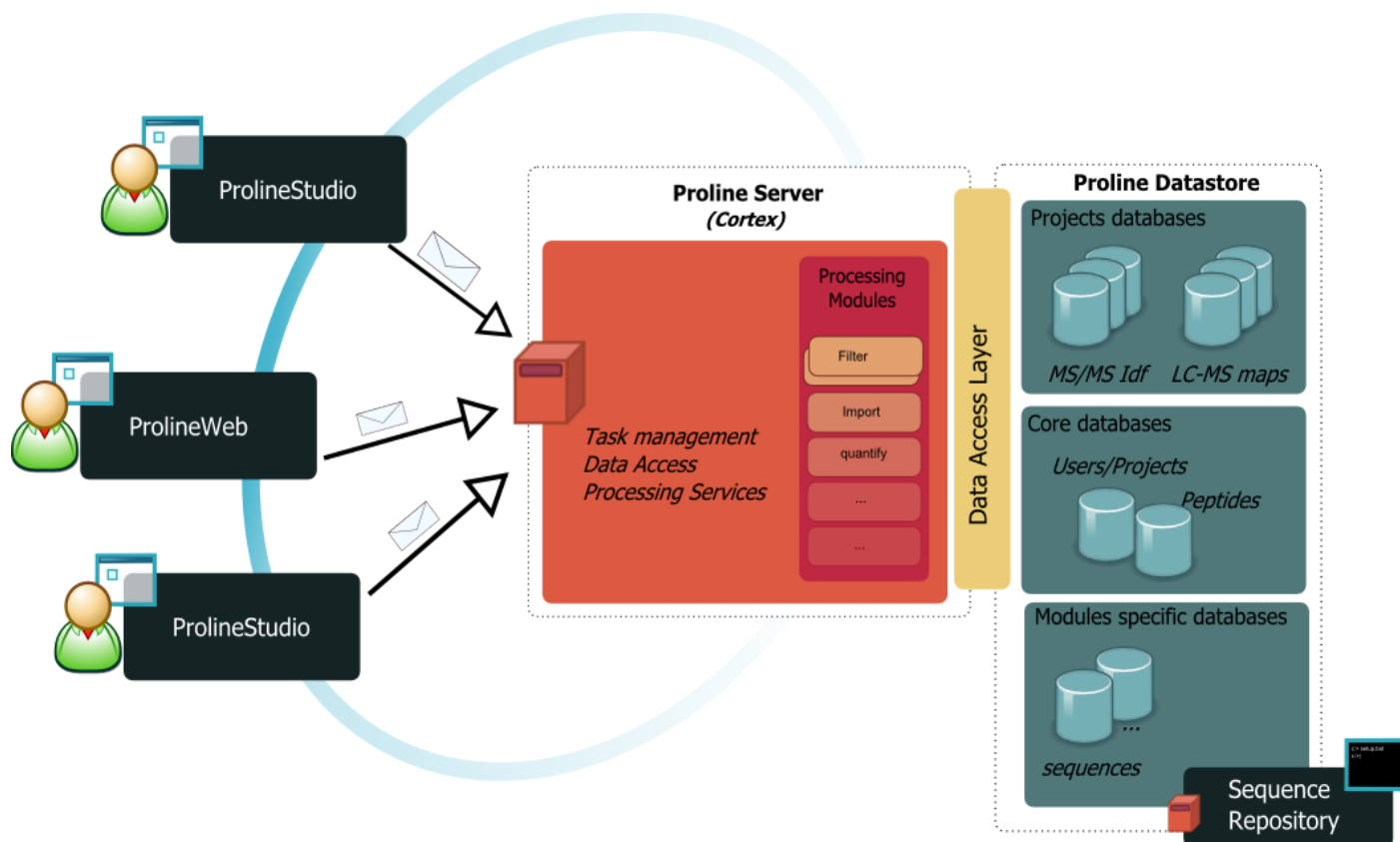
Check for updates	33
Upgrade all databases	33
H. Monitoring using Proline Admin CLI	34
Configure Proline-Admin	34
I. Proline User Management	34
Create a new Proline user	34
Change user group	34
Reset user password	35
II. Proline project management	35
Create a new Project	35
Activate/disable a project	35
III. Proline databases management	35
Check_for_updates	35
Upgrade_dbs	36
I. Annexe	36
Archiving Projects	36
PostgreSQL common configuration	37
Use pgAdmin to see database data	38

A. Introduction

I. Architecture Overview

The suite is based on different components (see figure below):

- **Proline Datastore** is based on a Relational Database Management System storing the data used by the software. Proline stores data in three different database schemas. The “core” database schemas created once at datastore initialization. It contains data related to users' projects (**UDSdb**). The two additional schemas are used to create a new database each time a new user project is created. Those databases store identification data, peptides sequences, post-translational modifications (**MSIdb**) and quantification data (**LCMSdb**) associated with users' projects.
- **Proline Server** is the core component handling user’s processing tasks. The server is associated with a messaging system responsible for sending and receiving messages between the graphical user interfaces and the server.
- **ProlineStudio** and **ProlineWeb** are two different graphical user interfaces, both allowing users to start tasks and visualize the resulting data. Either or both can be used:
 - **ProlineStudio** is a desktop application that must be installed on each individual user’s computer.
 - **ProlineWeb** is a web application requiring an additional web server installation, but can be accessed by a standard web browser on the client side.
- Proline **Sequence Repository** is an external module to retrieve protein sequences from Fasta files and store them in a sequence database.
- **Proline Admin** is a system administration application to setup and manage the Proline suite. This application is available as a command line application or with a graphical user interface. Some management may also be done using Proline Studio (see User Guide)



II. Requirements

The server-centric architecture of Proline imposes different requirements for the server computer and the client computers.

- Server-side Proline requirements:
 - o a **Java 17 JRE** must be installed on your server and should be the default version.
Run “java -version” to see the version of the installed Java

```
java version "17.0.3.1" 2022-04-22 LTS
Java(TM) SE Runtime Environment (build 17.0.3.1+2-LTS-6)
Java HotSpot(TM) 64-Bit Server VM (build 17.0.3.1+2-LTS-6, mixed mode, sharing)
```
 - o The PostgreSQL database server (versions above 11 recommended) must be installed and configured. For performance issues it is recommended to have a postgresql server on the same computer as Proline but it could be installed on a distinct one. By default, PostgreSQL settings are defined for modest hardware configurations but they can be modified to target more efficient hardware configurations (See [PostgreSQL optimization and authorizations](#)).
 - o Proline Server **must run in English “locale”**, on Linux OS, environment variable LANG=en_US.UTF-8 can be exported before starting Proline Server or the start_cortex script can be modified to add -Duser.language="en" -Duser.country="US" parameters.
- Client-side requirements for Proline Studio:
 - o On Windows 64-bits, Proline-Studio already includes a JRE, there is **no requirement**
 - o On other OS (Linux / MacOS), a Java 17 Runtime Environment (JRE) must be installed

Troubleshooting :

If you encounter a problem during the installation, do not hesitate to contact us and use the Forum (<http://www.profiroteomics.fr/forum/>)

B. Installing Proline : Automatic process

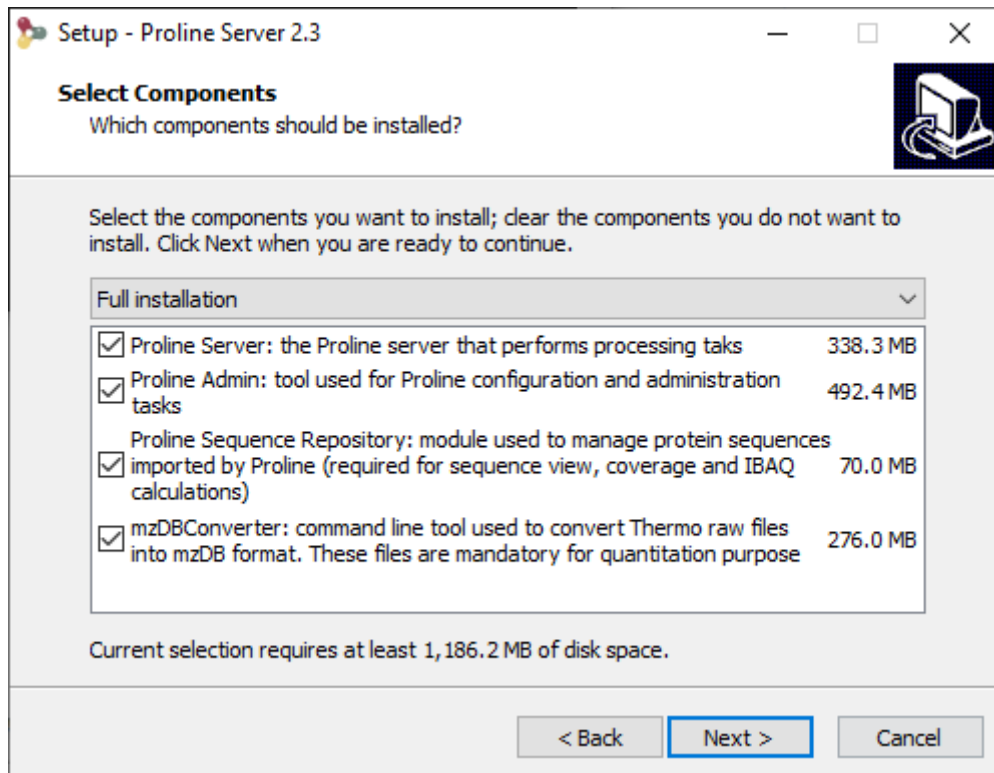
The section describes installing Proline Server using the Windows installer and configuring the components using Proline AdminGUI. This is only valid for Windows OS. For other OS, or if you prefer installing each component separately using manual installation, see [“manual section”](#). If you already have a version installed go to the [“upgrade section”](#).

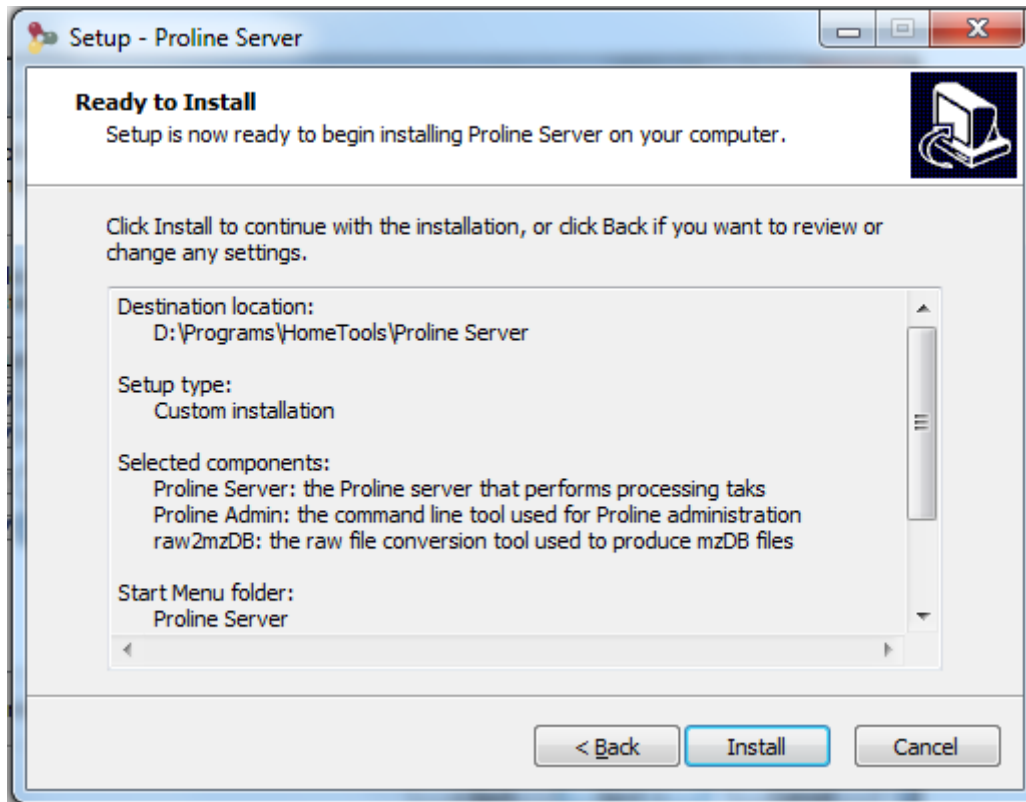
I. Installing Proline Server

Download the automated installer from the Proline website (<https://www.profiroteomics.fr/proline/proline-downloads/>).

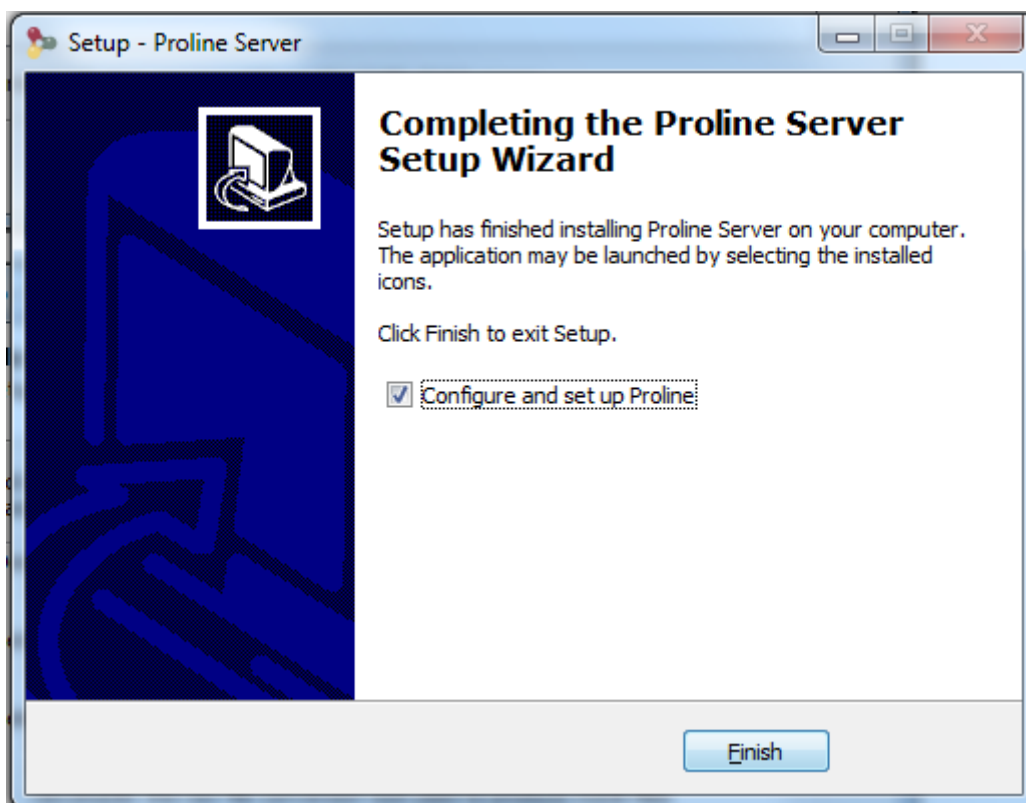
The wizard will guide you through the installation process. By default, the installer will unpack all components on the computer. However, it is possible to install only some components, to install them on distinct computers if it fits better your hardware architecture.

It is recommended to put Proline Server and Proline Admin on the same computer. Sequence Repository is recommended to be installed on the computer where Fasta files are accessible.





When components are installed, the installer proposes to open the configuration tool ([Proline Admin GUI](#)). After a first install, Proline Server requires configuration and Proline Datastore needs to be set up. See warning below before launching Proline Admin ...



Postgres & create a user proline

Warning: Before running setup you should **create a specific Postgresql user** for Proline with “create database” rights.

You can keep your installer windows open, create the Postgresql user using createuser.exe in a command Window(<pg_install_path>\PostgreSQL\11\bin\createuser.exe) .

```
createuser -U postgres -P -e -l -d <username>
```

This command line will ask you the password of the new user, later ask you the password of postgres, that is 'postgres' in case it is always default values...

```
#if you need delete the created user  
dropuser -U postgres username
```

If you can't create a user in the command line, you can create the 'user' by any db admin tools such as pgAdmin. Use pgAdmin-create login/group The password is to define in the 'Definition' tab.

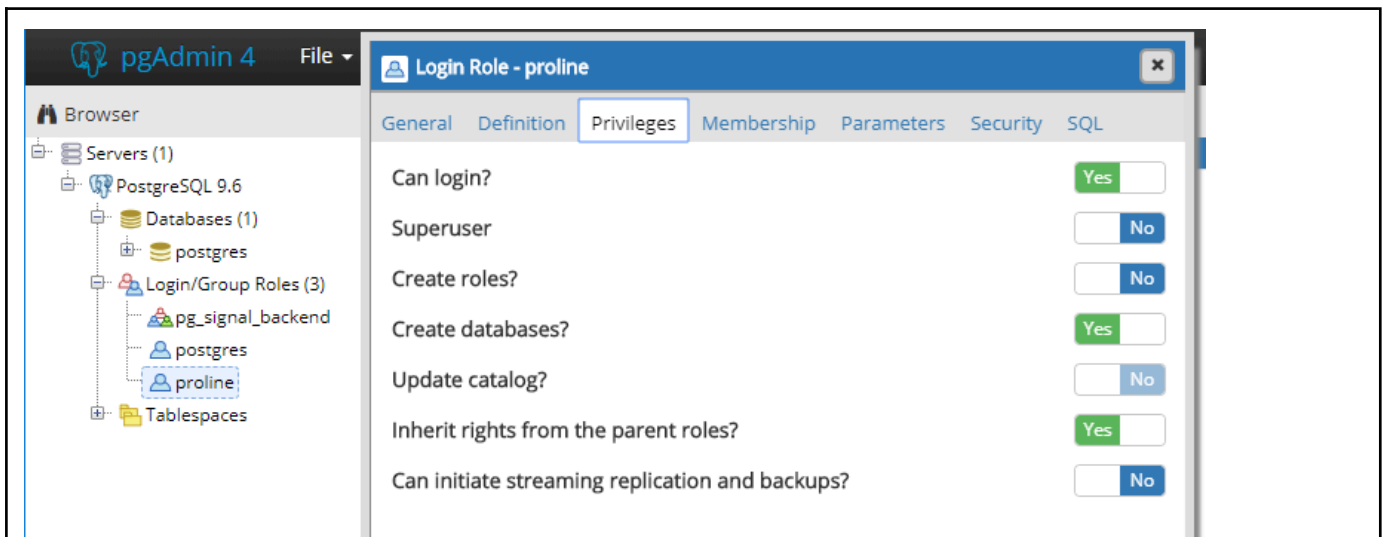


figure: add a proline user
rights of the proline user: can login, create database

Once you're sure the proline user is created, you can finish the Proline install and launch Proline Admin. When clicking on Finish, The Proline-Admin GUI install will be automatically started, in order to help you to configure your installation.

II. Setup Proline (using Admin GUI)

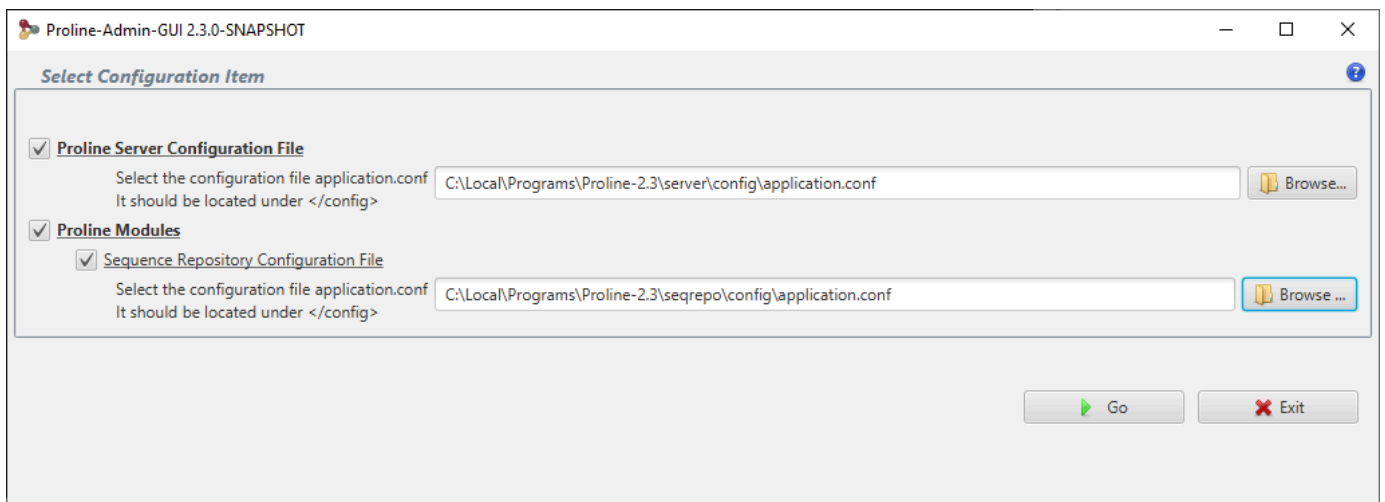
Proline-Admin GUI is a graphical tool that allows Proline administrators to configure each component. It includes the configuration files editing and the connection configuration to Postgresql database.

Launch Proline-Admin GUI install

Proline-Admin-GUI install should be automatically executed after installing Proline. If you have to use it afterwards, execute the script `install.sh` (if you are using Linux) or `install.bat` (if you are using Windows) located in the `<proline_install>/admin` folder.

Select configuration component

In this first step you need to select which component of Proline you want to configure. For a standard installation of Proline on a single server, you should configure all of them but if installation has been done on multiple servers, you can configure each component one after the other.



“Proline Server Configuration File”: The full path to the Proline server configuration file. Its default location is `<proline_install>/server/config/application.conf`

“Proline Modules”:

- “Sequence Repository Configuration File”: the full path to the Sequence Repository Configuration file its default location is `<proline_install>/seqrepos/config/application.conf`

Proline Server Configuration

This page will be displayed if Proline server configuration file path has been selected and defined in the first step

PostgreSQL

The PostgreSQL tab lets you edit the parameters to connect to the Proline databases:

- “Host”: the host name of the server running PostgreSQL. Using localhost or 127.0.0.1 is strongly discouraged, Proline databases will be available for this computer only.
- “Port”: the port number 5432 is the default value.
- “User”: the **proline** PostgreSQL user name to connect to the database (created at the end of the installation).
- “Password”: the **proline** user password to connect to the database.

Proline-Admin-GUI 2.3.0-SNAPSHOT

Proline Server Configuration

PostgreSQL JMS Server Mount Points

Database Server

Host:

⚠ WARNING: Using localhost or 127.0.0.1 is not advised, as it will make Proline available from the network. Enter host name value

Port:

User:

Password: Show password

You can verify the configuration using the “Test connection” button. For more details (see PostgreSQL [Connections and Authentication](#)).

JMS Settings

Warning: This part should not be modified, unless you use your own HornetQ messaging server!

Proline-Admin-GUI 2.3.0-SNAPSHOT

Proline Server Configuration

PostgreSQL JMS Server Mount Points

Use embedded JMS server Use specific JMS server

JMS Server

Host:

Port:

Proline Queue Name:

The communication between your local machine and the JMS Server is configured here. It is advised to use the embedded JMS server with its default values but if a JMS server has already been set up on another machine you can specify its address here.

- “Host”: the host name, its default value is “localhost”.
- “Port”: the port number, its default value is 5445.
- “Proline Queue Name”: the name of the queue that receives messages to a consumer, it's default value is “ProlineServiceRequestQueue”.

Mount Points

Proline-Admin-GUI 2.3.0-SNAPSHOT

Proline Server Configuration

PostgreSQL JMS Server **Mount Points**

File Locations

Raw files path: + Add

Alias = Full path [Browse] [Remove]

mzDB files path: + Add

mzdb_files = ./data/mzdb [Browse] [Remove]

mzdb_files2 = A Another Path [Browse] [Remove]

Result files path: + Add

mascot_data = ./data/mascot [Browse] [Remove]

XTandem_result = Full path [Browse] [Remove]

← Previous Next → [Apply] [Exit]

Proline Studio and Proline Web will be able to access the files from the locations provided here. You can add as many mount points as you want. To do so, just select a path and a unique name for the mount point. Only the name will be visible by the Proline users so it should be explicit.

Three types of entries are available: Raw files, mzDB files and result files. The identification result files from Mascot, OMSSA, Xltandem and others must be defined in the *Result file path*. **Proline users will only be able to import data from the “Result files” mount point.** mzdb files path is the location used during quantitation to find converted raw files.

Note: When providing mount points paths, you can use slash (“/”) or antislash (“\”) separator. Both formats are valid in Proline-Admin GUI install.

Proline Module Configuration (sequence repository)

Currently the only configurable module is sequence repository.

PostgreSQL

The PostgreSQL tab lets you edit the parameters to connect to the Proline databases server. This tab should contain the same information as the Proline Server configuration one. The same parameters should be used. See chapter [PostgreSQL](#) for more details.

JMS Settings

The communication between the computer hosting the module and the JMS Server is configured here. Same parameters as those specified in Proline Server configuration should be used. See chapter [JMS Settings](#) for more details.

Parsing Rules

The Parsing Rules tab lets you edit the parsing rules used by sequence repository to capture the protein accession numbers in your Fasta files. These rules should match the rules that are used in search engines such as Mascot: the search engine will extract the accession number from a Fasta file to write it in its result file, and Proline will search for this accession number in the same Fasta file using specified rules.

The screenshot shows the 'Proline-Admin-GUI 2.3.0-SNAPSHOT' window with the 'Sequence Repository Configuration' tab selected. The 'Parsing Rules' sub-tab is active. The configuration includes:

- Default Protein Accession:** A text field containing '>(\S+)' and a 'Default' button.
- Local Fasta Directories:** A '+ Add' button and a text field containing 'D:\Fasta', with 'Browse' and 'Remove' buttons.
- Parsing Rules:** A '+ Add' button and a list of three rules, each with a 'Remove' button:
 - Rule 1:** Id: label1, Fasta File Version: _(?:D(?:Decoy))_(.*)\.fasta, Fasta Pattern: ISA_, Accession Parse Rule: >\w{2}\{[^\}]+\}
 - Rule 2:** Id: label2, Fasta File Version: _(?:D(?:Decoy))_(.*)\.fasta, Fasta Pattern: UP_S_cerevisiae_MyDB, Accession Parse Rule: >\w{2}\{[^\}*\}(\S+)
 - Rule 3:** Id: UPS, Fasta File Version: _(?:D(?:Decoy))_(.*)\.fasta, Fasta Pattern: UPS1UPS2_ups1_ups2, Accession Parse Rule: >[^\}*\}(\S+)

At the bottom of the window are navigation buttons: 'Previous', 'Next', 'Apply', and 'Exit'.

- “Default Protein Accession”:
 - The default parsing rule to capture the protein accession number from a Fasta file. This parsing rule will be used on all Fasta files unless a specific parsing rule is defined below

- The default value is « >(\S+) »: It means that the text between the “greater than” sign and the first space character will be considered as the accession number. For instance, “sp|P10276|RARA_HUMAN” will be the accession number in the following Fasta entry:


```
>sp|P10276|RARA_HUMAN Retinoic acid receptor alpha OS=Homo sapiens OX=9606 GN=RARA PE=1 SV=2
```
- The “Default” button will reset the Default Protein Accession to its default value.
- “Local Fasta Directories”:
 - add one or more directories in which Fasta files are contained. The Sequence Repository will search for Fasta files in subdirectories as well.
- “Parsing Rules”:
 - Parsing rules can be added for specific Fasta files on which the default parsing rule would not work. It can be necessary if you have for instance different naming conventions for Uniprot and NCBI Fasta files, or for users who have different ways to generate their files. A parsing rule requires the following information:
 - “Id “: a unique identifier for the parsing rule.
 - “Fasta pattern”: this rule will be used on all the Fasta files matching this pattern. The match is done on the fasta file name. You can use regular expressions but the pattern is case insensitive, so make sure that the pattern will not match unwanted Fasta files.
 - “Fasta File Version”: a regular expression with capturing group to extract a release version from the Fasta file name (case insensitive). The rule may be used even if the regular expression does not match any version.
 - “Accession parsing rule”: a regular expression with capturing group for the protein accession number.

For more details and testing see [Testing rules of Sequence Repository](#) or dedicated [forum topic](#)

Summary

This section provides a summary of everything you have configured so far. It lets you check the new settings before updating the configuration files:

- Proline Server Configurations
 - “PostgreSQL: OK” if the connection to the database successfully established otherwise “PostgreSQL: NOK”
 - “JMS Server”: The type of JMS server (Embedded or Spécific)
 - “Mount Points”: the number of mount points by type that has been defined
 - “Set up or upgrade all Proline databases”: **Warning:** If there is a significant gap in version numbers, this action can take a while. This is mandatory for new installation or, in most cases, when upgrading proline version.
- Proline Sequence repository
 - “PostgreSQL: OK” if the connection to the database successfully established otherwise “PostgreSQL: NOK”
 - “JMS Server”: The type of JMS server. (Embedded or Spécific)
 - “Parsing rules”: number of defined parsing rules

Warning:

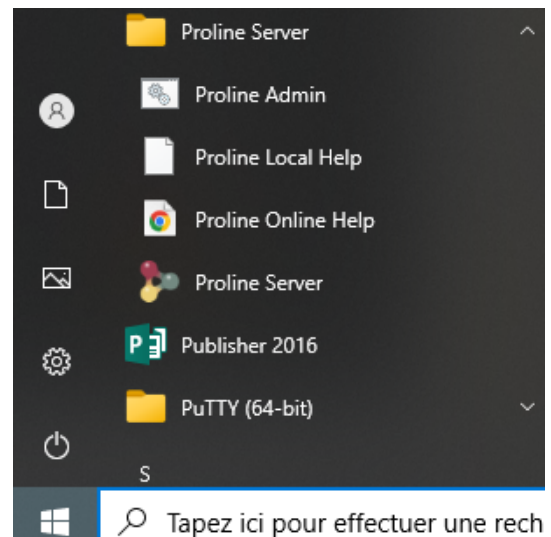
- Click the “Apply” button to save all specified settings in corresponding configuration files. All database **setup and/or upgrades will also be executed.**
- Click the “Exit” button to close the application. If “Apply” has not been done before, all changes will be lost

Installation Result

Once installation is done, the following folders are created under 'proline_server' root :

▼ Proline Server	
> admin	- admin, seqrepo and server folders contain respectively the Proline Admin, Sequence Repository and Proline Server modules.
data	
> mzDBConverter	- mzDBConverter contains the converter zip file. Unzip it to be able to convert raw files into mzDB format
resources	
> seqrepo	- resources contains, among others, installation documentation and user guide
> server	- Uninstaller is also available at the root : uninst000.exe

If chosen during installation, shortcuts may have been added to Desktop and to Start up menu



III. Installed scripts

Once all components have been installed and configured you should find the following scripts and shortcuts

- At startup, the Proline Server script is automatically started. This will launch the 2 needed modules to start the Proline server (the JMS (hornetQ) server and Proline Cortex). Be sure your postgresql server will be started before this script. Otherwise it won't work, you will have to run it manually, see desktop shortcut below
- If you choose to "create desktop" : 2 shortcuts have been added to your desktop, "Proline Server " and "ProlineAdmin Monitor". The first one is the same as the startup script. The second one will launch Proline Admin Monitor to help you manage Proline installation (see [Monitoring chapter](#)).
- On the Windows Menu, a Proline Server folder has been added with the following entries
 - ProlineAdmin Monitor: to help you manage Proline installation (see [Monitoring chapter](#)).
 - ProlineAdmin Install & Configure: to configure the installed modules as done during installation (see [Setup Proline using AdminGUI](#)). This should be done once. after, ProlineAdmin Monitor should be used.
 - Proline Server : Same as startup script.
 - Proline Local Help: Open user guide pdf file
 - Proline Online Help: Open user guide on profiproteomics website

C. Installing Proline : Manual process

This section describes how to install and setup Proline manually using archives files and editing configuration files. Nevertheless, it is also possible to install Proline modules manually and use ProlineAdmin for configuration purpose (see previous section : [Setup Using ProlineAdmin](#))

I. Installing Proline Server

First check that all [requirements](#) are installed on the computer, then download the zip archive containing Proline components from the website (<https://www.profiroteomics.fr/proline/proline-downloads/>)

The Proline Server archive file contains three other archives corresponding to the different components (Proline Server/ Sequence Repository/ Proline Admin). The user and installation guides are also provided in this archive. Unzip those components on the appropriate computer (Proline Server and Proline Admin are recommended to be on the same computer. Sequence Repository is recommended to be installed on the computer where the Fasta files are located).

After components are unzipped, you should configure them, this **can be done using Proline AdminGui in an automatic way** ([see previous chapter](#)) or **configure it manually** (see [next chapter](#)).

Warning: Before running setup you should **create a Postgresql user** for Proline with “create database” rights. See [This section](#) for details on how to create a user using pgAdmin or using createuser.exe.

II. Configuring Proline manually

Proline Admin

Edit the configuration file `config/application.conf` located in the Proline Admin folder. Then perform the datastore setup by running the dedicated script ([see below](#)).

The Proline Admin program files are located in the “.\admin” subfolder of the Proline installation directory if you used the Windows installer. Otherwise they should be in the folder obtained after Proline Admin GUI archive file extraction.

Configuration files

To allow Proline Admin to access Proline Datastore, you should modify the `application.conf`

```
server-config-file = ""
pwx-config-file = ""
seq-repo-config-file = ""

proline-config {
  driver-type = "postgresql" // valid values are: h2, postgresql or sqlite
  data-directory = "/Path/to/Proline/Data" //Not used actually...
}

auth-config {
  user="<user-proline>" //// SET TO Database Proline user login
  password="<proline_user_password>" //// SET TO Database Proline user password
}
```

```

host-config {
  host="your_postgresql_server_host" //!! Do NOT put "localhost", but the real IP address or
  fully qualified name
  port="5432" //or other port used to access your DBMS
}

uds-db { }

...
}

```

Note:

- first paths (server-config-file, pwx-config-file and seq-repo-config-file) **may only be used when using Admin GUI configuration editor**. Reference modules configuration files are automatically updated by the GUI.
- proline-config no change should be done, postgresql is actually the sql server used in production.
- auth-config: specify the **PostgreSQL Proline user** (and its password). This user should have been created when installing/setting up PostgreSQL. see warning above.
- host-config: IP address or name of the server hosting PostgreSQL. Don't use localhost but fully qualified name as it will be used by user GUI from other computers.
- default naming scheme of databases created by Proline can be modified by editing uds-db, msi_db and lcms-db entries, but **it should not be done without precise knowledge** and if not necessary


```

msi-db {
  connection-properties {
    dbName = "mysi_db_project"
  }
}

```
- Other parameters should not be modified unless specific usage.

Proline Server

When using command line configuration, you should manually edit and configure Proline Server file config/application.conf. The server folder is located in the ".\server" subfolder if the Windows installer has been used or in the folder in which the "Proline Server" archive has been unzipped.

application.conf

```

proline-config {
  driver-type = "postgresql" // valid values are: h2, postgresql or sqlite
  data-directory = "/Path/to/Proline/Data" //Not used actually...
}

auth-config {
  user="<user-proline>" //!! SET TO Database Proline user login
  password="<proline_user_password>" //!! SET TO Database Proline user password
}

host-config {
  host="your_postgresql_server_host" //!! Do NOT put "localhost", but the real IP address or
  fully qualified name
}

```



```

port="5432" //or other port used to access your DBMS
}

uds-db { }

...

mount_points {
  result_files {
    mascot_data = "D:/" //under window environment
    omssa_data = "/local/omssa/data" //under linux environment
    xtandem_data = "\\my_server\data"
    ...
  }

  raw_files {
  }

  mzdb_files {
    mzdb_files = "/local/data/mzdb"
    ...
  }
}

...

```

Configure the Datastore

Edit the `application.conf` file in the same way you did it for Proline Admin (see [above](#)). If your configuration is valid, the Proline Server will be able to use the datastore you've created using Proline Admin.

Configure the mount-points

Also in `application.conf`

Result identification files (Mascot, OMSSA or X!Tandem) as well as mzDB files (for the XIC Quantitation process) are only browsed from the Proline Server side.

Warning: on a Windows server installation, please use UNC style to access remote disks instead of mounted disk on letters. Otherwise Proline won't be fully functional. ex: use `\\my_server\disk\folder` instead of a mounted disk on Z: (Z: mounted as `\\my_server\disk\folder`)

Mascot or OMSSA path should be configured in `result_files` sub-entry, administrator can add one or more mappings as `label = "<absolute/directory/path>"`. mzDB files path should be set under `mzdb_files` sub-entry.

Label can be any valid string chosen by administrators to help users identify `mount_point`. If multiple repositories are defined, labels must be different.

Configure the Messaging connection

Unless you use your own HornetQ messaging server, this file should not be modified !

If needed, edit the `jms-node.conf` file to specify : the server hosting the messaging server (`jms_server_host`), the name of the Proline Queue (`proline_service_request_queue_name`; **warning:** should also be modified on client side), the number of thread Proline server could use (`service_thread_pool_size`; `-1` let the server define the appropriated value)

```

node_config {
  jms_server_host = "localhost"
  jms_server_port = 5445
}

```

```
proline_service_request_queue_name = "ProlineServiceRequestQueue"  
service_thread_pool_size = -1  
xic_files_pool_size = 2  
enable_imports = true  
}
```

III. Setting up Proline Datastore with command line

Once Proline configuration files have been edited, you must initialise Proline datastore using the Proline-Admin software provided with the Proline Suite. It is available as a command-line tool and is packaged with the Graphical Interface Proline Admin GUI.

Warning: Configuration files have to be set up before using command line scripts.

the command line

Set up the datastore by executing the following command:

on Windows systems

under the directory of <ProlineServer>\admin

```
> run_cmd.bat setup  
or  
> setup_proline.bat
```

on Linux systems

```
> chmod +x run_cmd.sh  
> run_cmd.sh setup  
or  
> chmod +x setup_proline.sh  
> setup_proline.sh
```

Once a datastore has been created, it is necessary to run an upgrade for additional configurations. See [next paragraph, “Updating the datastore”](#) (no backup is needed in this use case !)

D. Upgrading from a previous version

I. Upgrading Proline Server

If using the installer, there is no “upgrader”, so you must first uninstall the previous version of the server. The uninstallation will not remove previous data stored in Proline Datastore nor will it remove logs files.

Warning : Before doing so, to keep previous configuration in order to copy it in new version, copy files <Proline_install>/server/config/*,<Proline_install>/admin/config/* and <Proline_install>/seqrepo/config/* in a safe folder. You can also keep the previous scripts in order to use the same settings in new ones (do not replace new ones with previous scripts ! compare and merge them).

Then you have to do a fresh install, using the same folder as the previous installation. (See [Installing Proline Server using Windows installer](#) or [Installing Proline server using archive files](#)) . Once installation is done you can skip the setup step and merge the saved configuration files with the one in the fresh installation folders.

II. Updating the Datastore

To update the different Proline Databases, you need to get the appropriate version of Proline-Admin. You can use the command line or the provided graphical tool, Proline-Admin Gui monitor.

Backup Datastore

Once Proline databases have been upgraded, **you can not downgrade to the previous version**. This is why it’s **highly recommended to make a copy of your databases** before the upgrade process. In case of error during the update process , you can use the copy to restore your Proline databases.

In case of problems, **you can contact us for help or any question. Support will not be possible without a copy of all your databases.**

You can use the command **line to backup all your databases**. For example, to backup all databases under the file called all_dbs.backup:

```
C:\Program Files\PostgreSQL\9.6\bin>pg_dumpall.exe -U postgres -f
"D:\databases_back_up\all_dbs.backup"
```

Please wait until the backup process finishes with success. The saved file can be reloaded with psql similar to [pg_dumpall](#).

One effective measure to prevent facing those repeated password prompts is a [~/pgpass](#) file.

Here is the syntax the ~/pgpass file requires to work :

```
hostname:port:database:username:password
```

With a `~/pgpass` file present in your development environment, containing the necessary credentials for the postgres role, you can omit the `-W` (also `-w`) option and run `pg_dumpall` without manually authenticating with the password.

From the command line

Update the datastore by executing the following command:

on Windows systems

```
> run_cmd.bat upgrade_dbs
```

on Linux systems

```
> chmod +x run_cmd.sh  
> run_cmd.sh upgrade_dbs
```

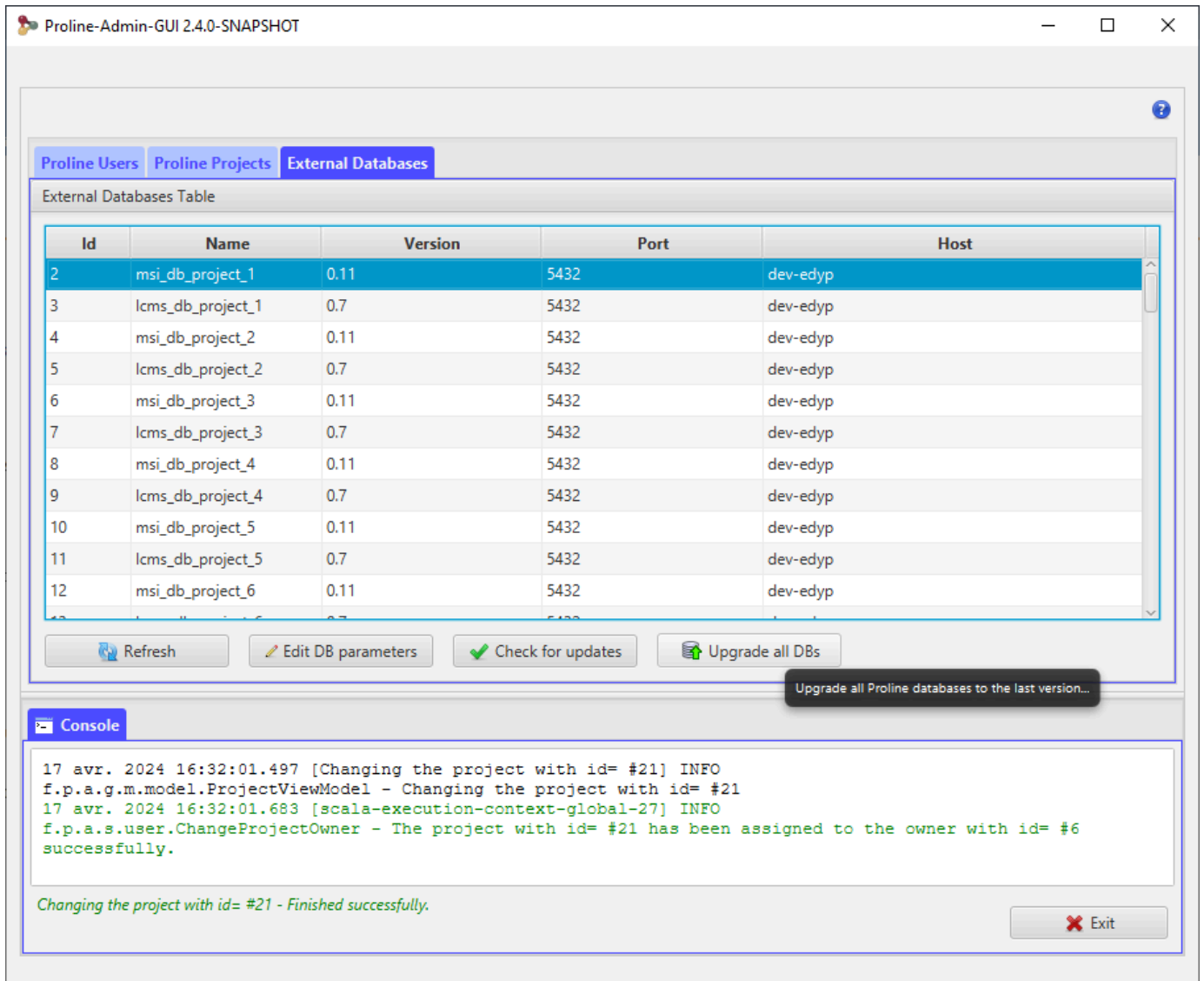
Verbose mode is used in order to display as much information as needed. You can view all displayed information in `proline_admin_log.txt`, and especially search for any “ERROR” message.

From the graphical interface

Open the graphical interface by running `monitor.bat` (on Windows) or `monitor.sh` (on linux system).

Note: If you don't have a valid postgres configuration, you can always edit postgresql settings in the Proline Admin GUI monitor.

In “External Databases” tab you can *Check for updates* and finally *Upgrade all DBs*.



Warning :

- This action may take a while if you have lots of projects or If there is a significant gap in version numbers.
Do NOT interrupt the operation
- To update the datastore you should specify the required amount of memory (it depends on your database size) to avoid the Java heap size error messages.

To control the memory (RAM) used by the application:

- You can open the `run_cmd.bat` file (on Windows system) or the `run_cmd.sh` file (on Linux system) located under the directory of `<ProlineAdmin_Install>` (or `<ProlineServer\admin is installer was used for installation>` Proline_Admin< if you are using the command line.
- You can open the `monitor.bat` file (on Windows system) or `monitor.sh` file (on a Linux system) located under the directory of `<ProlineAdmin_Install>` (or `<ProlineServer\admin is installer was used for installation>`), if you are using the graphical interface .

Hence, you can use :

- Xmx to specify the maximum heap size.
- Xms to specify the initial java heap size.

-XX:+UseG1GC to use the Garbage First (G1) Collector.

For example, to specify the maximum heap size with value 12 Gbytes and to use the Garbage First collector using the GUI monitor, edit monitor.bat:

```
.\java\jdk\bin\java [modules option, keep existing] -Xmx12G -XX:+UseG1GC -cp "[keep existing class path]" -Dlogback.configurationFile=config/logback.xml fr.proline.admin.gui.Monitor %*
```

III. Upgrading Sequence Repository

There is no upgrade procedure for this module. You should do a fresh install (see [installing Sequence Repository](#)) but you **can copy the previous configuration files in the fresh installation** ! Specifically parsing rules conf file which should not change from one version to another, except if explicitly indicated in release note.

You can always edit the Sequence Repository configuration file using Proline-Admin GUI graphical tool :

- Run install.bat
- Select "Proline Modules" > "Sequence Repository Configuration File"
- Select Sequence Repository Configuration. for more details see ([Sequence Repository](#)).

IV. Upgrading Proline Studio

Proline Studio application distribution is a zip file that must be unzipped on each client PC. You should unzip it under a different folder than the previous version. The executable file is in the install directory, start_studio.bat

E. Running Proline Server

Note: All the scripts listed in this section are in the directory <Proline_Server>/server/ path or where the Proline Server archive file has been unzipped.

If you have used the Windows-installer, see the following scripts description : [Installed scripts](#). However, the memory setup may be useful !

I. Running the Server

Server Memory Setup

The default amount of memory used by the server can be changed in the `start_cortex.bat` (`start_cortex.sh` on Linux) file. If the computer is configured with a large amount of memory, it is recommended to increase this value. Change the value of the `-Xmx` java option to `-Xmx100g` to increase the available memory to 100Gb.

Running on Windows

On Windows, the server is started from the command : `start_server.bat`.

(This single command will start HornetQ then Cortex by calling successively `start_HornetQ.bat` and `start_cortex.bat`).

Note:

1. If java 17 is not the default java, you may need to modify the script to use the correct java. Open the `start_cortex.bat` file and modify the line
“`java -Xmx20G -XX:+UseG1GC ...`” with “`<path_to_java17>\bin\java -Xmx100G -XX:+UseG1GC ...`”
2. HornetQ should be run with java 1.8. This required version is embedded with hornetQ and is explicitly referenced by `run.bat`

Running on Linux

On Linux, the server is started by running `start_HornetQ.sh` first then `start_cortex.sh`.

Note:

1. If java 17 is not the default java, you may need to modify the script to use the correct java. Open the `start_cortex.sh` file and modify the line
“`java -Xmx20G -XX:+UseG1GC ...`” with “`<path_to_java17>/bin/java -Xmx100G -XX:+UseG1GC ...`”
2. It may be necessary to run ‘dos2unix’ on script files , if message such as `./start_hornetQ.sh: /bin/sh^M: bad interpreter: No such file or directory` occurs. In this case run :
“`dos2unix start_hornetQ.sh`”
3. HornetQ should be run with java 1.8. This required version is embedded with hornetQ and is explicitly referenced by `run.sh`

F. Sequence Repository

Although this module is optional, it is recommended to install it to have access to the protein sequences in the user interfaces. The module can be installed from the Windows installer or from the archive file by simply extracting the content of the archive.

This module can be installed on the same computer running Proline Server. However, as it will parse Fasta files to extract sequences and descriptions from them, the Sequence Repository will be more efficient if it is installed on the computer where the Fasta files are stored. In any case, you must also be able to access the PostgreSQL server from this computer.

I. Sequence Repository Configuration

Configuration files are located in the folder `<seqrepo_folder>/config`.

Configuration could be done through Proline Admin GUI interface : [see specific chapter](#). Nevertheless it is possible to configure this module manually :

Server and Datastore description

`application.conf` file defines the datastore and server description to access to the UDS database. Properties specified here should be the same as the one you specify while [configuring the Proline Server](#).

```
proline-config {
  driver-type = "postgresql" // valid values are: h2, postgresql
  max-pool-connection=3 //Beta properties : specify maximum number of pools connected to DB
  Server. default to 3
}

//User and Password to connect to the database server.
auth-config {
  user="<user-proline>"
  password="<password-proline>"
}

//Databases server Host
host-config {
  host="<host>"
  port="5432"
}

uds-db {
  connection-properties {
    dbName = "uds_db" }
}

h2-config {
  script-directory = "/h2"
  connection-properties {
    connectionMode = "FILE"
    driver = "org.h2.Driver"
  }
}

postgresql-config {
  script-directory = "/postgresql"
  connection-properties {
    connectionMode = "HOST"
    driver = "org.postgresql.Driver"
  }
}
```



```
}  
}
```

note:

- `auth-config`: Specify the PostgreSQL Proline user (and its password). They are the same as specified in `application.conf` for Proline Server
- `host-config`: IP address or host name, and port, of the PostgreSQL server.

Do not change anything else.

Protein description parsing rule

As this module is used to extract Protein sequence and description from a Fasta file for a specific protein accession, it is necessary to configure the rule used to parse the protein ACC, from Fasta description line. This is similar to the rules specified in Mascot Server, but these specified rules don't use the same semantics !

To do this, `parsing-rules.conf` file should be edited. In this file it is necessary to escape (this means prefix with `\`) some characters: `\`, `:` and `'`

```
//Specify path to fasta files for SeqRepository daemon. Multiple path separated by ','  
between []  
//On linux system : local-fasta-directories =["/local/mascot/sequence"]  
local-fasta-directories =["D:\\mascot\\sequence"]  
  
// Rules used for parsing fasta entries. Multiple rules could be specified.  
// name : identifying rule definition  
// fasta-name : FASTA file name must match specified Java Regex CASE_INSENSITIVE. multiple  
Regex separated by ',' between [] could be specified  
// fasta-version : Java Regex with capturing group for fasta release version extraction  
(CASE_INSENSITIVE)  
// protein-accession : Java Regex with capturing group for protein accession extraction  
  
parsing-rules = [{  
  name="label1",  
  fasta-name=["uniprot"],  
  fasta-version=".*_([^_]*).fasta",  
  protein-accession = ">\\w{2}\\\\|([^\|]+)\\\\|"  
},  
{  
  name="label2",  
  fasta-name=["myDB"],  
  fasta-version=".*_([^_]*).fasta",  
  protein-accession = ">\\w{2}\\\\|^[^\|]*\\\\| (\\S+)"  
}  
]  
  
//Default Java Regex with capturing group for protein accession if fasta file name doesn't  
match parsing_rules RegEx  
// >(\\S+) : String after '>' and before first space  
default-protein-accession = ">(\\S+)"
```

For example, label1 rule will capture P07259 from line sp|P07259|PYR1_YEAST ... ⇒ a '>' then 2 char then '|' then (capture until '|') and '|'... This rule will be applied for all Fasta files prefixed with uniprot.

You can see [dedicated forum](#) for more information

II. Testing rules

In order to verify the specified configuration, once previous files have been configured and saved, run the following tool under sequence repository installation directory run-TestConfiguration.bat (Windows) or run-TestConfiguration.sh (Linux).

An output will be displayed with all Fasta files found and for each which rule will be applied. The first 3 entries of each will also be displayed with the extracted protein accession.

Output should look like:

```
Scanning [D:\temp\fasta]
[D:\temp\fasta] scan terminated
Number of traversed dirs: 1
Found FASTA file names: 7
---- Scanning Fasta local path ----
Using default rule ">(\S+)" for fasta "Nouvelle_base_données_sara.fasta"
Accession "tr|F8WIX8|H2A.1_MOUSE" will be used for entry ">tr|F8WIX8|H2A.1_MOUSE
Original_Name=F8WIX8_MOUSE Histone H2A OS=Mus musculus GN=Hist1h2a1 PE=3 SV=1".
Accession "tr|Q5M8Q2|H2A.L.1.3_MOUSE" will be used for entry ">tr|Q5M8Q2|H2A.L.1.3_MOUSE
Original_Name= Q5M8Q2_MOUSE Histone H2A OS=Mus musculus GN=OTTMUSG00000016789 PE=2 SV=1".
Accession "tr|J3QP08|H2A.L.1.4_MOUSE" will be used for entry ">tr|J3QP08|H2A.L.1.4_MOUSE
Original_Name= J3QP08_MOUSE Histone H2A OS=Mus musculus GN=Gm14501 PE=3 SV=1".

[UPS1UPS2_D_20121108.fasta] matches Fasta Name Regex "UPS1UPS2_"
Using rule ">[^\|]*\|(\S+)" for "UPS1UPS2_D_20121108.fasta"
Release (using rule "_(?:D|(?Decoy))_(.*)\.fasta") = "20121108"
Accession "ALBU_HUMAN_UPS" will be used for entry ">P02768ups|ALBU_HUMAN_UPS Serum albumin
(Chain 26-609) - Homo sapiens (Human)".
Accession "NEDD8_HUMAN_UPS" will be used for entry ">Q15843ups|NEDD8_HUMAN_UPS NEDD8 (Chain
1-81) - Homo sapiens (Human)".
Accession "RASH_HUMAN_UPS" will be used for entry ">P01112ups|RASH_HUMAN_UPS GTPase HRas
(Chain 1-189) - Homo sapiens (Human)".

Using default rule ">(\S+)" for fasta
"uniprot-taxonomy%3A-Mus+musculus+%28Mouse%29+%5B10090%5D-.fasta"
Accession "sp|Q9CQV8|1433B_MOUSE" will be used for entry ">sp|Q9CQV8|1433B_MOUSE 14-3-3
protein beta/alpha OS=Mus musculus GN=Ywhab PE=1 SV=3".
Accession "sp|P62259|1433E_MOUSE" will be used for entry ">sp|P62259|1433E_MOUSE 14-3-3
protein epsilon OS=Mus musculus GN=Ywhae PE=1 SV=1".
Accession "sp|P68510|1433F_MOUSE" will be used for entry ">sp|P68510|1433F_MOUSE 14-3-3
protein eta OS=Mus musculus GN=Ywhah PE=1 SV=2".

[iSa_D_20130403.fasta] matches Fasta Name Regex "ISA_"
Using rule ">\w{2}\|([^\|]+)\|" for "iSa_D_20130403.fasta"
Release (using rule "_(?:D|(?Decoy))_(.*)\.fasta") = "20130403"
Accession "Q99M51tag" will be used for entry ">sp|Q99M51tag|NCK1_strep-tag Cytoplasmic
protein NCK1 OS=Mus musculus GN=Nck1 PE=1 SV=1".
Accession "Q9ES52tag" will be used for entry ">sp|Q9ES52tag|SHIP1_strep-tag
Phosphatidylinositol 3,4,5-trisphosphate 5-phosphatase 1 OS=Mus musculus GN=Inpp5d PE=1
SV=2".
Accession "###REV###Q99M51tag" will be used for entry ">sp|###REV###Q99M51tag|NCK1_strep-tag
Reverse sequence, was Cytoplasmic protein NCK1 OS=Mus musculus GN=Nck1 PE=1 SV=1".
```

```
[UP_MouseEDyP_D_20150629.fasta] matches Fasta Name Regex "UP_"
Using rule ">\w{2}\|([^\|]*)\|\S+" for "UP_MouseEDyP_D_20150629.fasta"
Release (using rule "_([^\_])*\.fasta") = "9"
Accession "Q9CQV8" will be used for entry ">sp|Q9CQV8|1433B_MOUSE 14-3-3 protein beta/alpha
OS=Mus musculus GN=Ywhab PE=1 SV=3".
Accession "Q9CQV8-2" will be used for entry ">sp|Q9CQV8-2|1433B_MOUSE Isoform Short of
14-3-3 protein beta/alpha OS=Mus musculus GN=Ywhab".
Accession "P62259" will be used for entry ">sp|P62259|1433E_MOUSE 14-3-3 protein epsilon
OS=Mus musculus GN=Ywhae PE=1 SV=1".

[S_cerevisiae_Decoy_20121108.fasta] matches Fasta Name Regex "S_cerevisiae_"
Using rule ">\w{2}\|([^\|]*)\|\S+" for "S_cerevisiae_Decoy_20121108.fasta"
Release (using rule "_([^\_])*\.fasta") = "8"
Accession "P38903" will be used for entry ">sp|P38903|2A5D_YEAST Serine/threonine-protein
phosphatase 2A 56 kDa regulatory subunit delta isoform OS=Saccharomyces cerevisiae (strain
ATCC 204508 / S288c) GN=RTS1 PE=1 SV=2".
Accession "P31383" will be used for entry ">sp|P31383|2AAA_YEAST Protein phosphatase PP2A
regulatory subunit A OS=Saccharomyces cerevisiae (strain ATCC 204508 / S288c) GN=TPD3 PE=1
SV=3".
Accession "Q00362" will be used for entry ">sp|Q00362|2ABA_YEAST Protein phosphatase PP2A
regulatory subunit B OS=Saccharomyces cerevisiae (strain ATCC 204508 / S288c) GN=CDC55 PE=1
SV=2".

[UPS1UPS2_Decoy_20121108.fasta] matches Fasta Name Regex "UPS1UPS2_"
Using rule ">[^\|]*\|(\S+)" for "UPS1UPS2_Decoy_20121108.fasta"
Release (using rule "_(?:D|(?:Decoy))_(.*)\.fasta") = "20121108"
Accession "ALBU_HUMAN_UPS" will be used for entry ">P02768ups|ALBU_HUMAN_UPS Serum albumin
(Chain 26-609) - Homo sapiens (Human)".
Accession "NEDD8_HUMAN_UPS" will be used for entry ">Q15843ups|NEDD8_HUMAN_UPS NEDD8 (Chain
1-81) - Homo sapiens (Human)".
Accession "RASH_HUMAN_UPS" will be used for entry ">P01112ups|RASH_HUMAN_UPS GTPase HRas
(Chain 1-189) - Homo sapiens (Human)".
```

These configurations are not easy. You should test every new parsing rule to be sure they are correct !

Don't hesitate to ask help, using our mail or the forum (<http://www.profiroteomics.fr/forum/>) !

III. Running Sequence Repository

Sequence Repository can be run in two modes: as a daemon or as service on demand. In the first case, it will automatically scan for new protein sequences in every project, with a given frequency (default is every two hours). This can be costly from a performance point of view..

In the last case, new sequences will only be retrieved when the user requests it for a specified identification summary. This feature is available in Proline Studio with the "Retrieve protein sequences" command.

Running as Service Daemon

To run Sequence Repository as a Daemon, you must modify and execute `run-RetrieveService.bat` (Windows) or `run-RetrieveService.sh` (Linux).

By default, RetrieveService will be run with option "-t 2" : the scan will be executed every 2 hours. Edit the `run-RetrieveService` script to change this value.

Other available options are:

```
Usage: <main class> [options]
```

```
Options:
```

```
-f, --forceUpdate
```

```
force update of MSIdb result summaries and biosequences (even if already updated)
```

```
Default: false
```

```
-p, --project
```

```
the ID of the single project to process
```

```
Default: 0
```

```
-t, --time
```

```
the daemon periodicity (in hours)
```

```
Default: -1
```

```
-debug
```

```
set logger level to DEBUG (verbose mode)
```

```
Default: false
```

Running as Service On Demand

To run Sequence Repository on demand, you must execute `run-RetrieveOnDemand.bat` (Windows) or `run-RetrieveOnDemand.sh` (Linux)

Note: the scripts could be modified to increase the amount of memory used.

G. Monitoring using Proline Admin GUI

Proline administrators can manage Proline users, projects and external databases via a graphical interface tool called Proline-Admin GUI monitor (recommended) or from the command line.

Some action can also be done with Proline Studio (see User Guide- Proline Studio - Administration)

This graphical tool helps administrators to realise several operations for example: Add a Proline user, disable a Proline project, create a new project, ...

Launch Proline-Admin GUI monitor

The graphical interface can be started by double clicking on the starting scripts:

- `monitor.bat` (on Windows).
- `monitor.sh` (on Linux).

Note:

All these scripts are in the directory `<Server_Installation_Path>/Proline-Admin-GUI/`

The scripts will launch the application and you will find your logs under the folder logs.

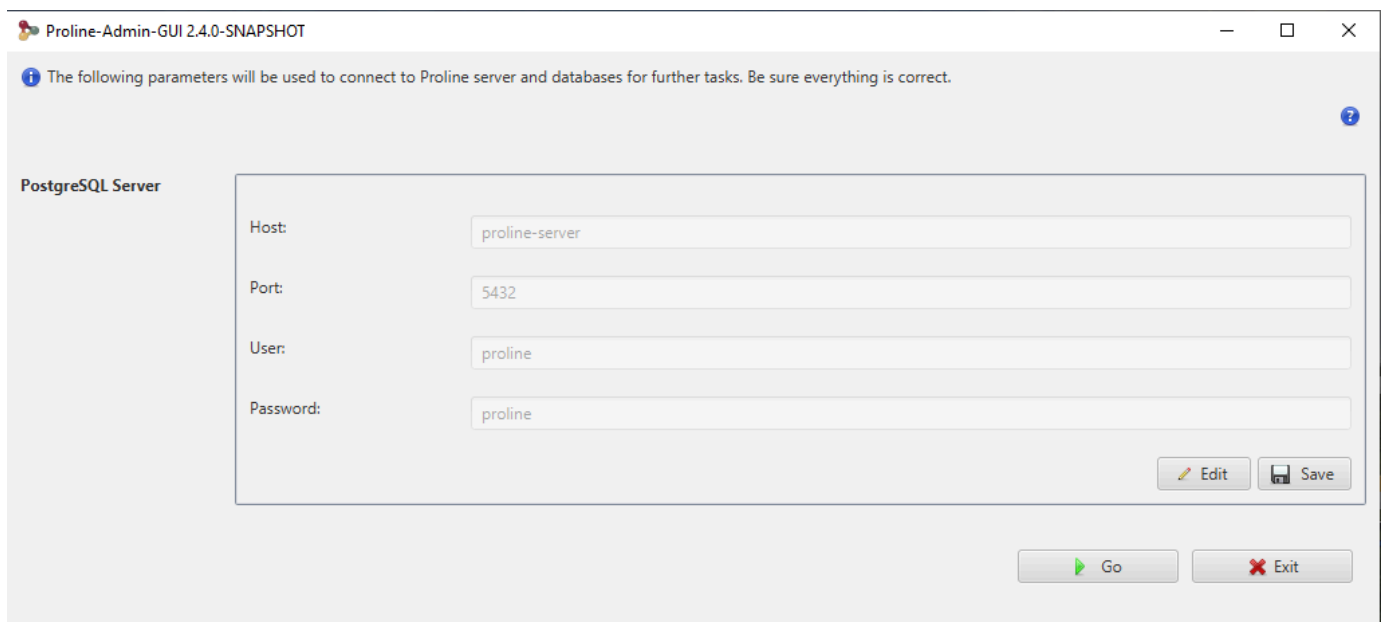
The graphical interface in details

Warning: Proline-Admin GUI monitor will not work if you are not connected to the database server.

Before using the application make sure that you have already setup Proline configuration files and the datastore.

Proline datastore properties summary

- PostgreSQL Server: the properties used in Proline-Admin to connect to the databases.



The screenshot shows a window titled "Proline-Admin-GUI 2.4.0-SNAPSHOT". At the top, there is a message: "The following parameters will be used to connect to Proline server and databases for further tasks. Be sure everything is correct." Below this, the "PostgreSQL Server" configuration form is displayed. It contains four input fields: "Host" with the value "proline-server", "Port" with "5432", "User" with "proline", and "Password" with "proline". To the right of these fields are "Edit" and "Save" buttons. At the bottom of the window, there are "Go" and "Exit" buttons.

If Proline has been installed properly and you are connected to the database server. Click on the go button to start using Proline-Admin GUI monitor. You will have a general view with Proline users, Proline projects and Proline external databases tabs.

If not well configured, click on 'Edit' to specify database host, port, user and password as in configuration steps. (see [above](#))

On all tabs, a “Refresh” button allow user to synchronise the table view with the existing data from the database

I. Proline user management

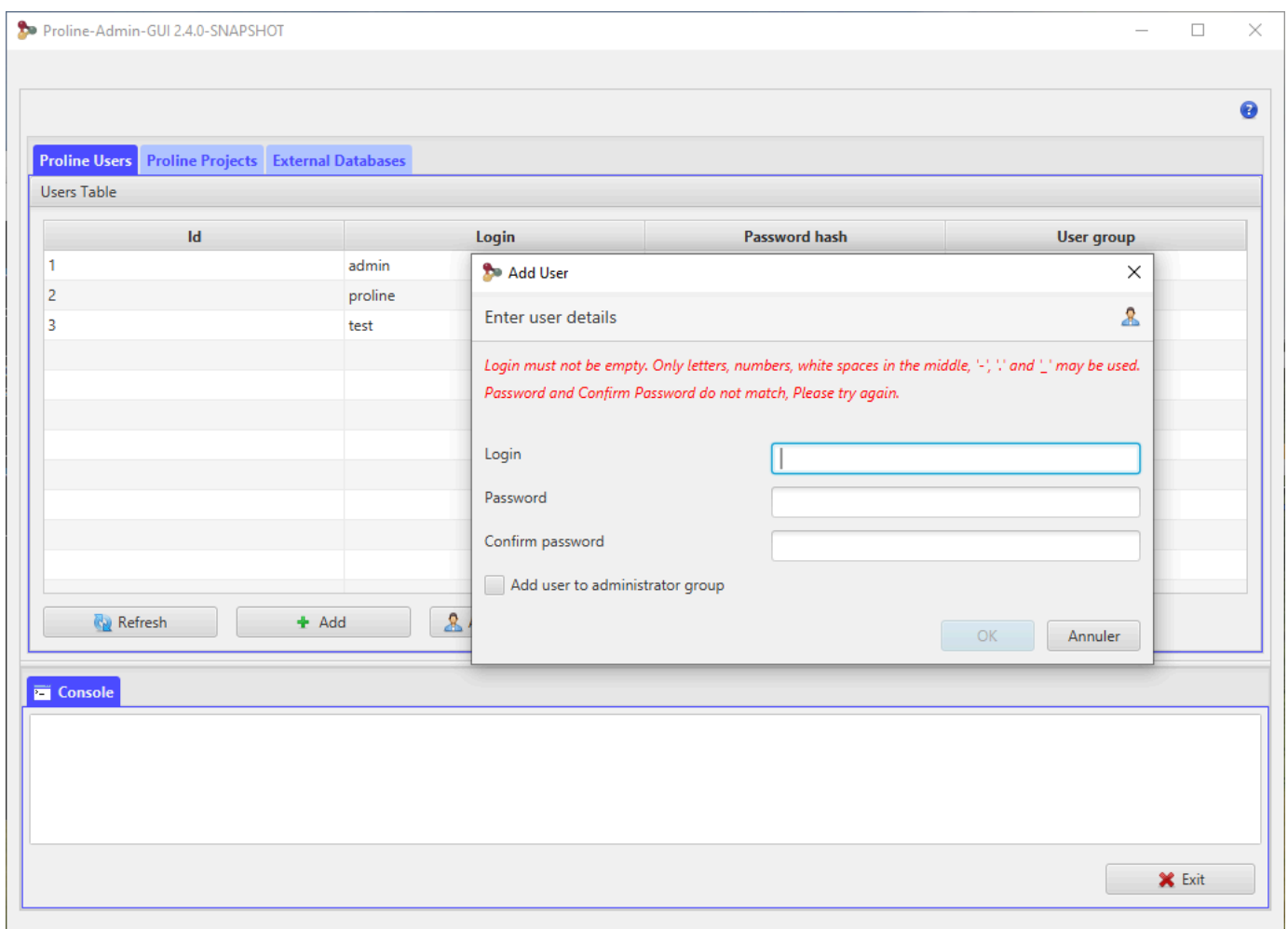
Note Creation and modification of Proline users can be done using **Proline Studio**

Create a new Proline user

Click on the ‘Add’ button, you must fill the required fields.

- Login: the identifier for the new account. You must choose a login that does not exist.
- Password: the user password.
- Confirm password: both passwords must be the same.
- Add user to the Proline administrator group: it adds this user to the Proline administrator group.

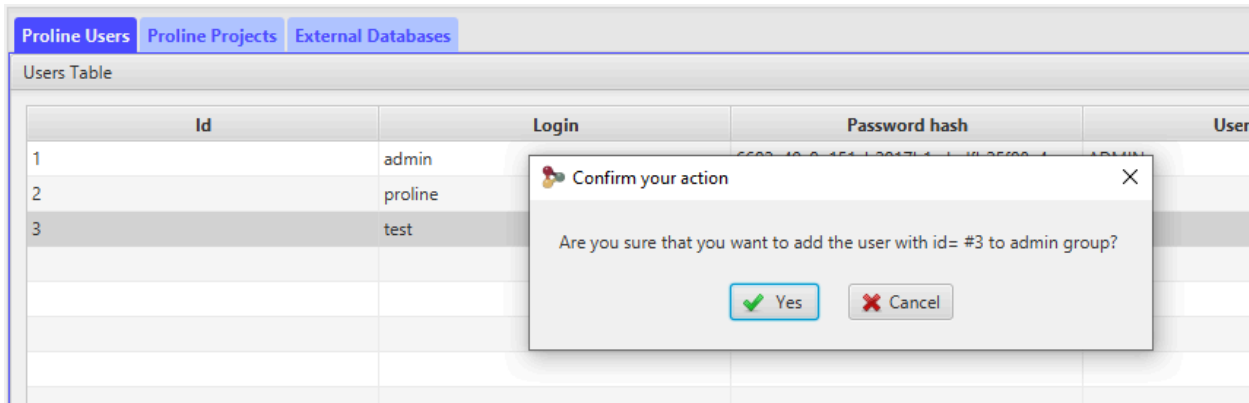
Note: standard users should not be in the administrator group.



Add to user/admin group

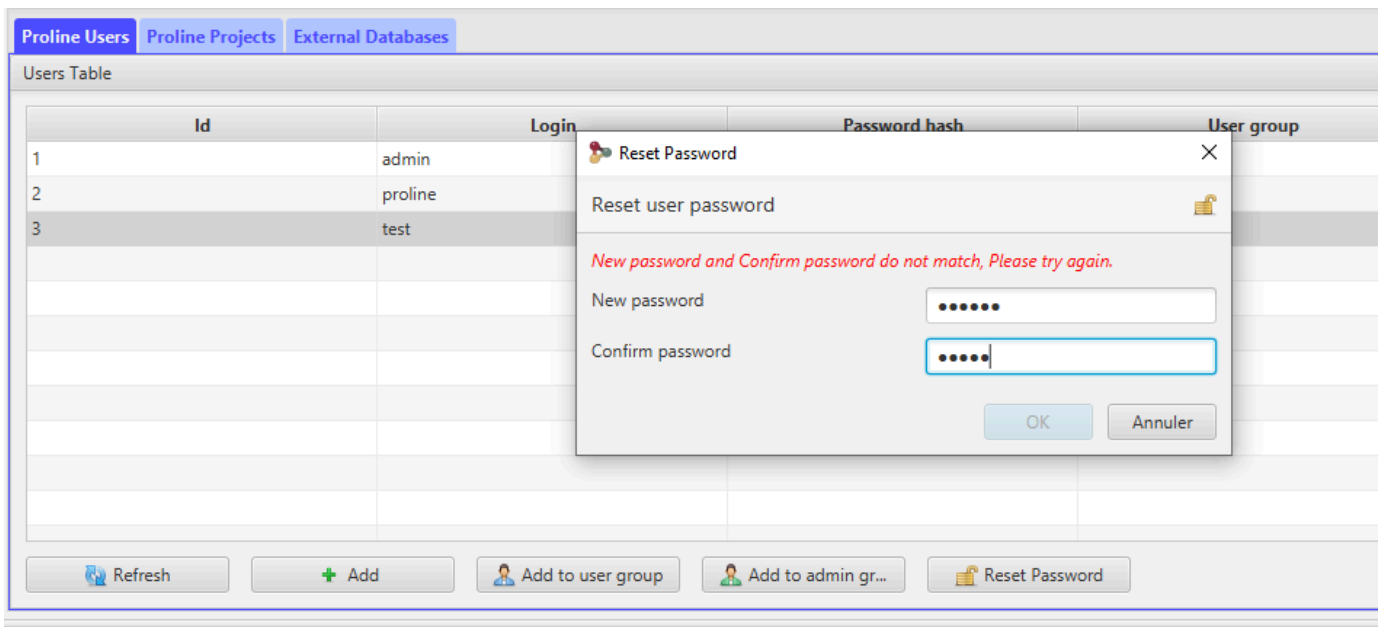
This action will add the selected user to the admin group or to the user group.

Note: standard Proline users should not be in the Admin group.



Reset Password

This action will change the password for the selected Proline user. The entered passwords must match.



Note:

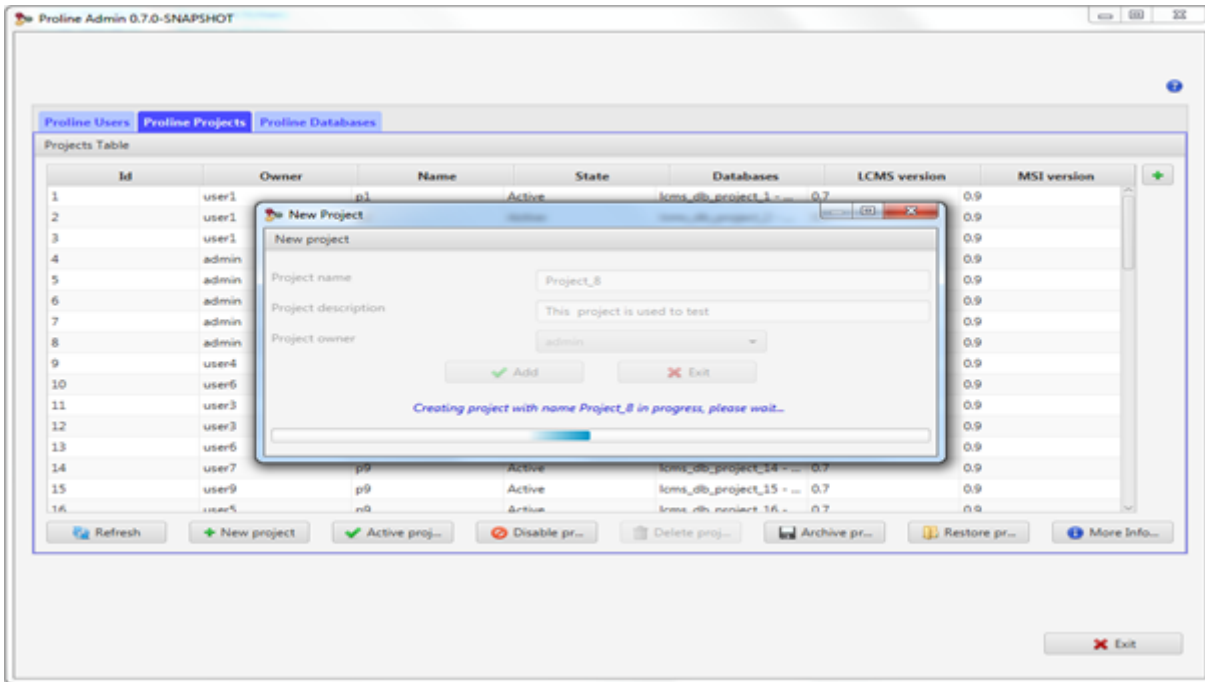
- You can sort the table on any column by clicking on its header. Shift-click on another to activate multi-sorting.

II. Proline project management

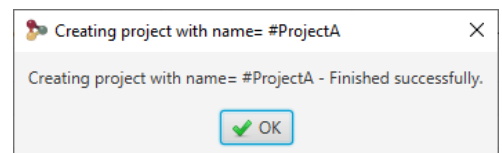
New Project

This action will create a new Proline project. You must fill the required fields.

- Project name: the project name must not be empty.
- Project description: The project description is optional.
- Project owner: The project owner must not be empty. You can select the owner from the list of users.

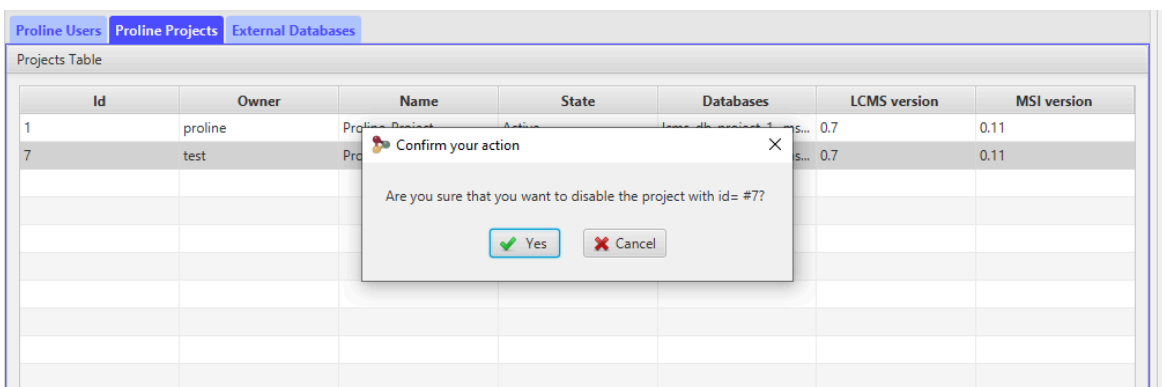


Warning: the creation process may take a few minutes as it requires databases creation. Wait until the success (or Fail) popup appears
Note : project creation can be done using **Proline Studio**



Activate/Disable Project

This action will activate/disable the selected Proline project. The project will not be deleted and can be re-activated later.

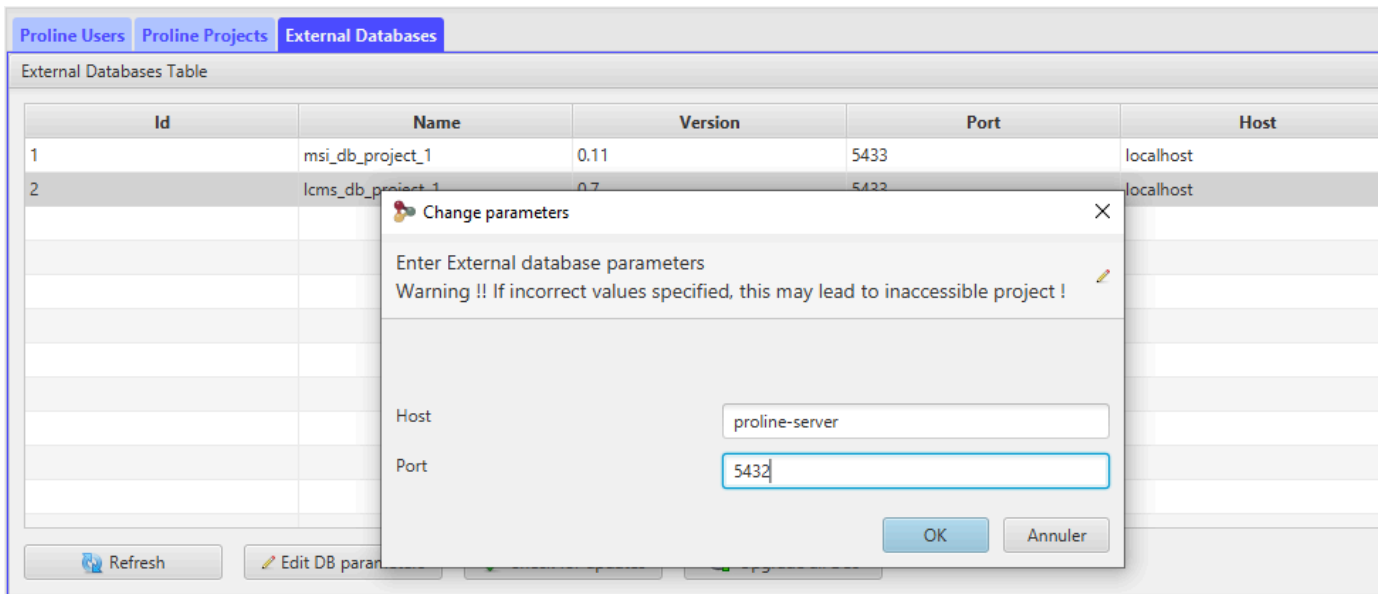


III. Proline databases management

Edit DB parameters

This action allows the Proline administrator to change the connection properties of the selected database(s). Only host name and port number can be updated.

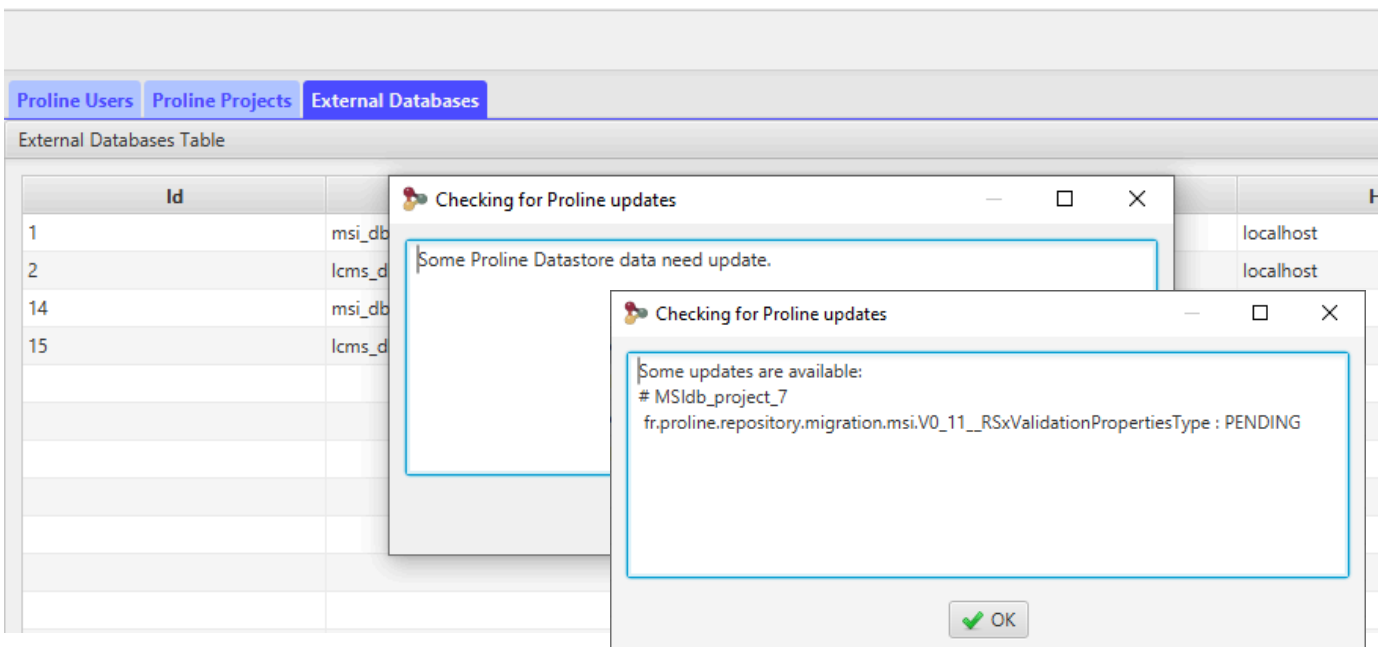
Warning: the database connection properties must not be empty. if incorrect values are entered, the project will not be accessible from the user any more.



Check for updates

This action will check for available updates. It will check the state of all projects. It summarises the number of migrations to apply to the databases.

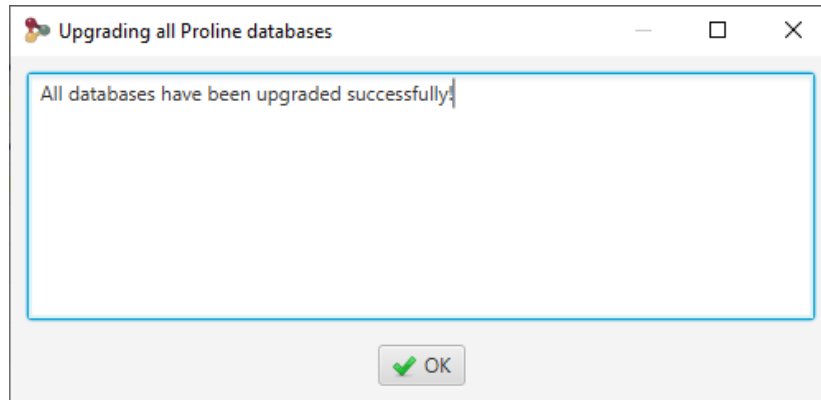
Note: This action may take several minutes. Wait until the 'finish' popup appears.



Upgrade all databases

This action will upgrade all databases to the last version.

Note: If there are lots of projects or if there is a significant gap in version number, this action may take a while. You **must wait** until the migrations are applied to all databases.



H. Monitoring using Proline Admin CLI

If you can't or don't want to use the GUI, you still can manage the Proline server using the command line interface.

Configure Proline-Admin

To access the Proline datastore in order to manage users and projects, the postgresql configuration should have been correctly done ! see [Configure Proline Admin](#) chapter (specifically postgresql part)

I. Proline User Management

Create a new Proline user

You can create a Proline user with the Proline Admin "RunCommand" script. Open a command line window and type the following command:

- Windows: `run_cmd.bat create_user -l <user_login> -p <user_password> -a`
- Linux: `sh run_cmd.sh create_user -l <user_login> -p <user_password> -a`

Note: The password is optional. If not provided, the default password will be "proline".
If you specify -a the user will be in the administrator group otherwise in the user group.

Change user group

- Windows: `run_cmd.bat change_user_group -uid <user_id> -a`
- Linux: `sh run_cmd.sh change_user_group -uid <user_id> -a`

Note: If you specify -a (optional) the user will be in the administrator group otherwise in the user group.

Reset user password

- Windows: `run_cmd.bat change_password -uid <user_id> -np <new_password>`
- Linux: `sh run_cmd.sh change_password -uid <user_id> -np <new_password>`

II. Proline project management

Create a new Project

Run the following command line from the Proline-Admin directory. Open a command line window and run :

- Windows: `run_cmd.bat create_project -oid <owner_id> -n <project_name> -desc <project_description>`
- Linux: `sh run_cmd.sh create_project -oid <owner_id> -n <project_name> -desc <project_description>`

Note:

- The project description is optional.
- Project creation can be done using Proline Studio

Activate/disable a project

Run the following command line from the Proline-Admin directory. Open a command line window and run :

- Windows: `run_cmd.bat change_project_state -pid <project_id> -d`
- Linux: `sh run_cmd.sh change_project_state -pid <project_id> -d <`

Note: Use -d to disable the project, do not use -d to activate it.

III. Proline databases management

Check for updates

Run the following command line from the Proline-Admin directory. Open a command line window and run:

- Windows: `run_cmd.bat check_for_updates`
- Linux: `sh run_cmd.sh check_for_updates`

Note :

- This action will check the state of all databases and check for available updates without installing any.
- This action can take several minutes.

Upgrade_dbs

Run the following command line from the Proline Admin directory. Open a command line window and run:

- Windows: `run_cmd.bat upgrade_dbs`
- Linux: `sh run_cmd.sh upgrade_dbs`

Note:

- This action will upgrade all databases to the last version.
- This action can take a while.

I. Annexe

Archiving Projects

There is currently no way to automatically archive Proline Projects. By the way, here is a manual protocol to do it in order to free disk space on server side and to be able to restore these projects if needed. The restore project operation is not trivial so it should not be used to switch between few projects but to archive projects that may not be used anymore. The restore operation should be a last resort.

NOTE. When archiving a project it is necessary to archive the generic database at the same time. For this reason, it is preferable to archive a set of projects.

You could see the list of active projects associated with databases size on disk using administration Projects and Databases view. See Proline User Guide (HowTo-ProlineStudio> Administration > Projects and Databases section).

Steps to archive projects!

1. **Backup** from PgAdmin (or any postgresql server admin tool)
 - a. For each project, backup both associated databases (`msi_db_project_<ID>`, `lcms_db_project_<ID>`)
 - b. Backup `uds_db` and `seq_db` databases. `seq_db` exists only if the Sequence Repository module has been installed.
2. **Tag project** as none active and optionally archived. This can be done
 - a. from PgAdmin (or any postgresql server admin tool). Run following SQL query on `uds_db` by replacing `<ID1>`, `<ID2>`... by the effective project Ids and using correct date:

```
update project set serialized_properties = '{"is_active":false,
"archive_date":"2019-06-20 14:00:00.000"}' where id IN (<ID1>, <ID2>);uds_db
```
 - b. from Proline Admin, use the [Activate](#) option. In this case, no archive information will be stored, only `'{"is_active":false}'` will be registered. This has no effect on Proline behaviour.
3. **Delete database**, from PgAdmin (or any postgresql server admin tool)
Select each projects associated databases (`msi_db_project_ID`, `lcms_db_project_ID`) and select 'Delete/Drop'

PostgreSQL common configuration

The PostgreSQL configuration files are located in the “data” folder. By default, this folder is <PostgreSQL_install>/11/data.

First you must ensure that PostgreSQL accepts remote connections. The `postgresql.conf` file must be modified to allow remote computer connections. Edit the `listen_addresses` line to allow IP of remote computers. See [PostgreSQL Documentation](#).

The `pg_hba.conf` file lists the IP addresses that are allowed to connect to your databases. In order to allow connections to the PostgreSQL server, make sure to add the following line (or more specific mask if needed): `host all all 0.0.0.0/0 md5`. The configuration of `pg_hba.conf` is documented in [PostgreSQL documentation](#).

Create a PostgreSQL user for Proline with “create database” rights.

PostgreSQL can affect the Proline Server performances; it is generally a good idea to optimise PostgreSQL parameters.

Most useful tunable parameters in postgresql.conf

Following recommended memory sizes are given for a server with **16 GiB** of physical memory and about **8 GiB** dedicated for the PostgreSQL instance.

<code>max_connections</code>	Number of concurrent SQL sessions (each Proline Server task can use 1 to 5 SQL sessions, each Proline-Studio instance can use some SQL sessions). Default value 100 → increase if many users
<code>tcp_keepalives_idle</code>	Number of seconds before sending a keepalive packet on an otherwise idle TCP connection. Help with broken router / firewall and checking for dead peers. Default value 0 (2 hours) → 300 (5 minutes)
<code>shared_buffers</code>	Use about 1/4 of physical memory dedicated to the PostgreSQL instance. Default value 32 MB → 2048 MB
<code>checkpoint_segments</code>	Use (<code>shared_buffers</code> / 16) ; max. 64 or 256 for write-heavy bulk loading. Default value 3 → 128
<code>checkpoint_completion_target</code>	0.9 for high value of <code>checkpoint_segments</code> Default value 0.5 → 0.9
<code>temp_buffers</code>	Per session Used for temporary tables creation. Default value 8 MB → 512 MB
<code>work_mem</code>	Several per session Used for hashing, sorting and <code>IN</code> operator when processing queries. Default value 1 MB → 4 MB to 64 MB
<code>maintenance_work_mem</code>	Used for initial index creation and <code>VACUUM</code> operations. Default value 16 MB → 1024 MB

effective_cache_size	Assumption about the effective size of the disk cache to optimize index use (Monitor physical memory allocated by system to disk cache operations). Default value 128 MB → 4096 MB
----------------------	--

Use pgAdmin to see database data

id	name	description	creation
1	Project1	test	2018-10-18 14:47:09.87