

Media management application

This document shows how to install and use the multimedia management application

## Requirements

Here are the software and libraries used by the system

- Ffmpeg
- Mysql
- Python3.8+
  - Flask
  - Mysql-connector-python
  - Pillow
  - Werkzeug

## Installing

Before installing the packages copy the application package to /opt/mediamanager. Installing on other places is possible but the initialization scripts provided on the package should be changed accordingly

### Installing requirements

First we need to install the system packages for the FFmpeg, Python, Pip and MySQL (if database is on the same server

On Alma Linux it should be the following

```
dnf -y install mysql-server python38 python38-pip ffmpeg
```

On Ubuntu use this

```
apt update && apt install -y mysql-server python3.8 python3.8-venv python3-pip ffmpeg
```

This will install the required packages on most Debian and RedHad based Linux distros

## Python Libraries

The following is a list of Python packages/libraries needed by the application, use the following file with pip to install as shown on the next steps

```
Flask==3.0.3
Pillow==10.3.0
Werkzeug==3.0.3
mysql-connector-python==8.4.0
```

requirements.txt

Here is how to use the requirements.txt with pip, to help making our environment clean from noise of existing python install on the operating system we can use a virtual environment

First enter the application directory, once again, let's assume /opt/mediamanager on this document

```
cd /opt/mediamanager
```

Then we create the python virtual environment, initialize it and install the required packages

```
python3 -m venv myenv
source myenv/bin/activate
pip install -r requirements.txt
```

## Database

Once we have python and its libraries we should import the database schema to the MySQL server

To import the database schema and the basic system information you should use source command from within mysql client

```
mysql -u root -p
source database-schema.sql
```

## Adding permission

In order for the system to be able to query the database we need create and add permissions to the mediamanager user (or the username and password set on MySQL pool on app.py)

```
mysql -u root -p
CREATE USER 'mediamanager'@'localhost' IDENTIFIED BY 'm3d14m4n4g3r';
GRANT ALL PRIVILEGES ON mediamanager.* TO 'mediamanager'@'localhost';
FLUSH PRIVILEGES;
EXIT;
```

## Installing as service

To install the application as a service one can use either systemd or System V scripts

### Systemd

For systemd copy the mediamanager.service to the systemd units directory

```
cp mediamanager.service /etc/systemd/system/
```

Then enable and start the service

```
systemctl daemon-reload
systemctl enable mediamanager.service
systemctl start mediamanager.service
```

### SysV

On legacy SysV systems copy the mediamanager file to /etc/init.d/ or /etc/rc.d/ depending on your system

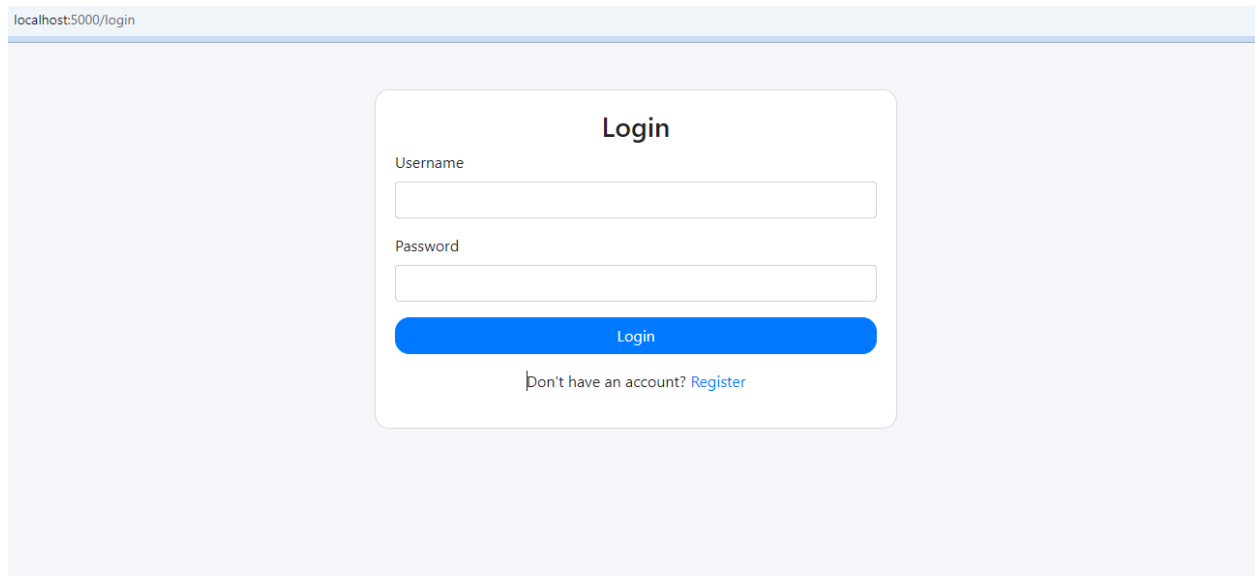
```
cp mediamanager /etc/init.d
```

Enable and start with the following

```
chmod +x /etc/init.d/mediamanager
chkconfig --add mediamanager
service mediamanager start
```

## Login Page

The first page the users should see is the login page, here we put username and password the enter the main system.



localhost:5000/login

### Login

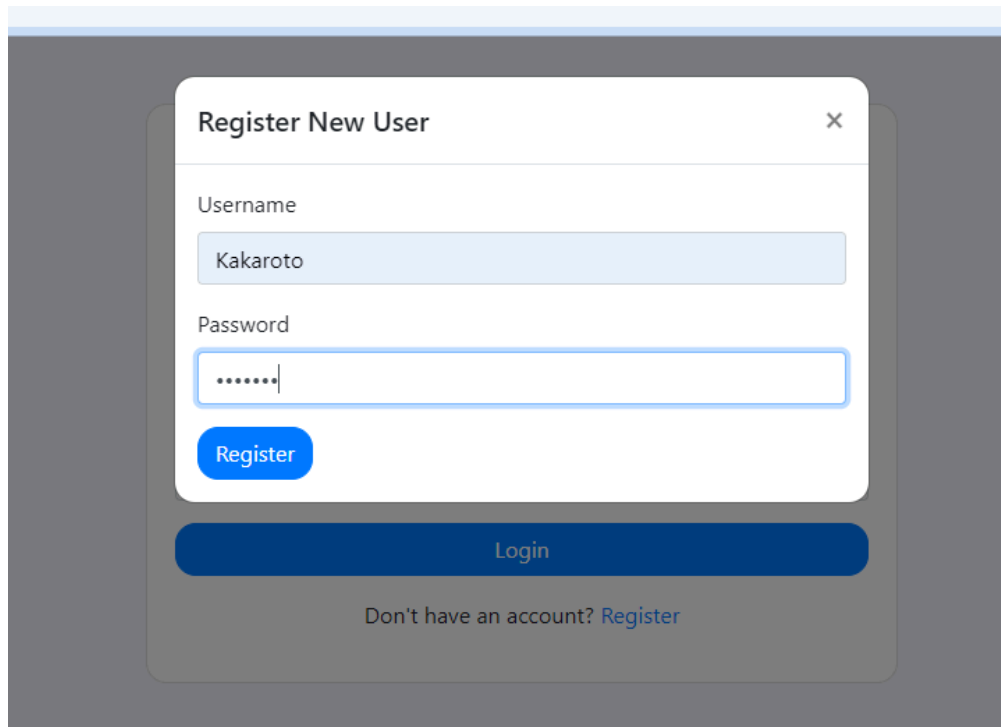
Username

Password

Login

Don't have an account? [Register](#)

By default the system will allow the visitor to create a new account, just click register and fill the simple form



A modal dialog box titled "Register New User" with a close button (X) in the top right corner. It contains two input fields: "Username" with the value "Kakaroto" and "Password" with masked characters ".....". Below the password field is a blue "Register" button. At the bottom of the dialog is a dark blue "Login" button. Below the dialog, centered on the page, is the text "Don't have an account? [Register](#)".

Register New User

Username

Kakaroto

Password

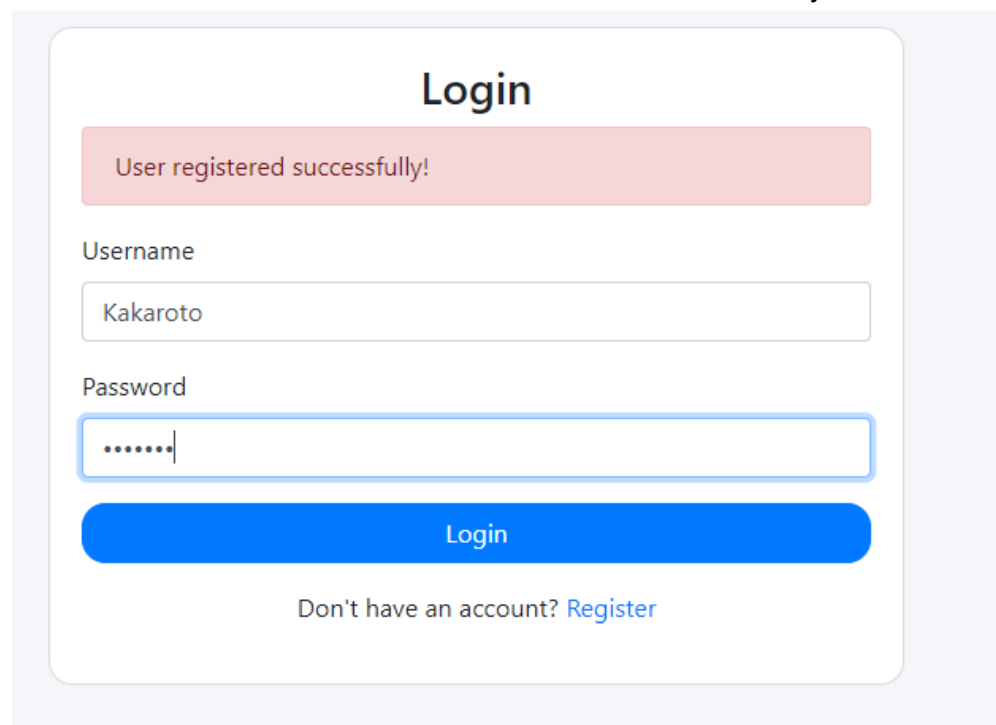
.....

Register

Login

Don't have an account? [Register](#)

The system will then confirm the user creation, now use it to enter the system



A login form titled "Login". At the top, a red message box says "User registered successfully!". Below this are two input fields: "Username" with the value "Kakaroto" and "Password" with masked characters ".....". Below the password field is a blue "Login" button. At the bottom of the form is the text "Don't have an account? [Register](#)".

Login

User registered successfully!

Username

Kakaroto

Password

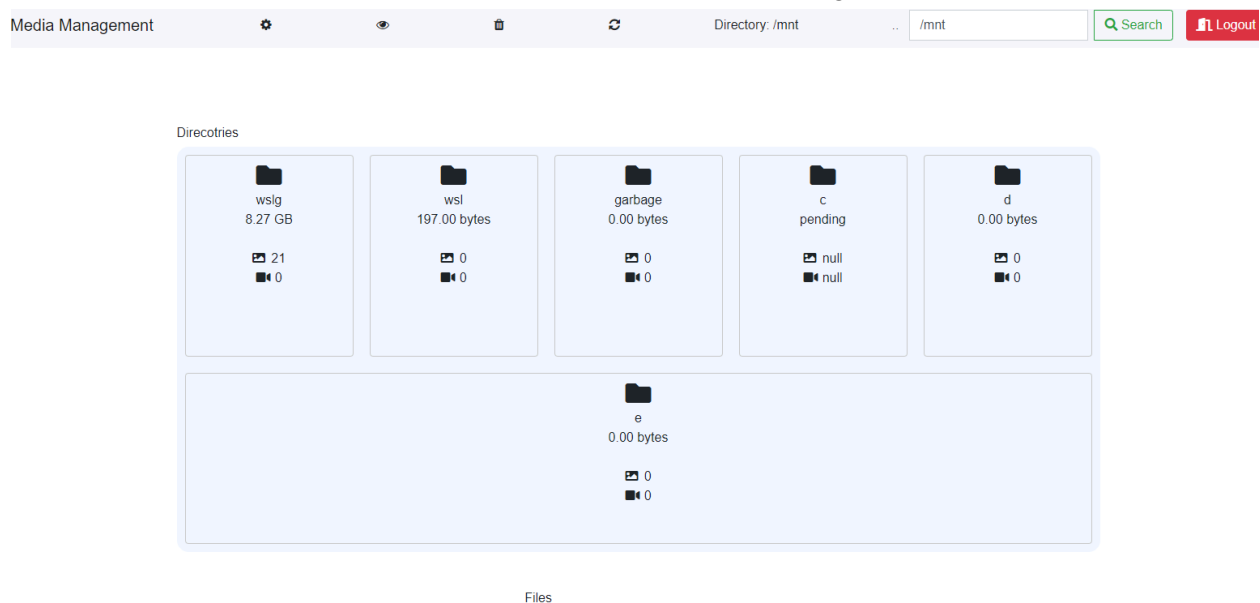
.....

Login

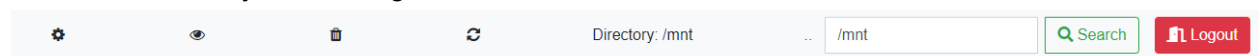
Don't have an account? [Register](#)

## Home Screen

The main screen contains some buttons and the actual file listing as follows



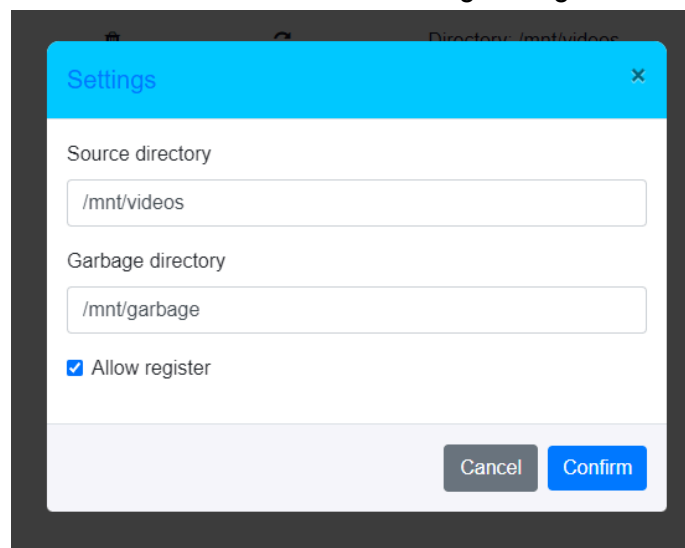
Here i will cover the function of each control on the navbar, starting on the settings where we should finish the system configuration



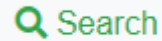
## Settings



The gear icon will open the settings dialog, you can set here the source directory, the directory for deleted files and should disable the user registering featured when not in use



## Search

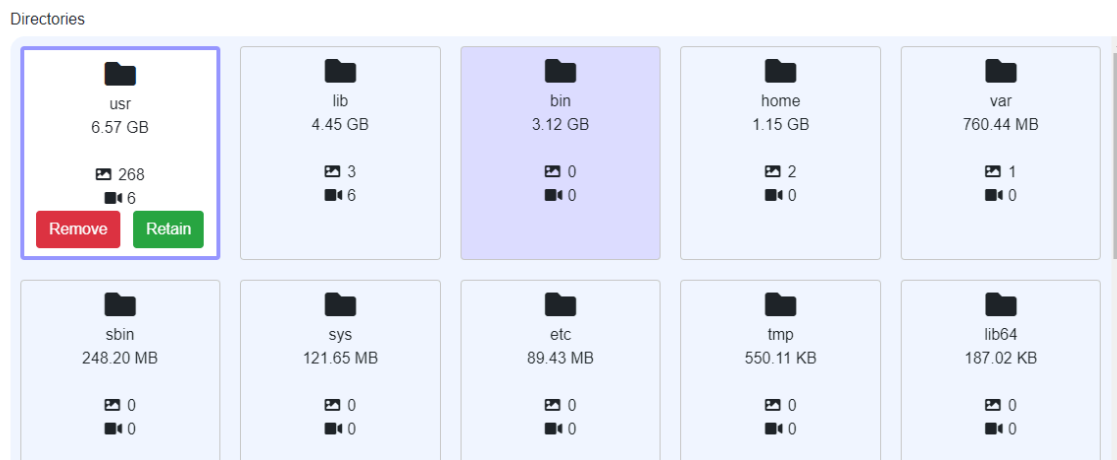


Now the system is ready to manage files, click on the search button if the system is not already showing the desired files, the search button also should be used to refresh the content if needed

## Navigation

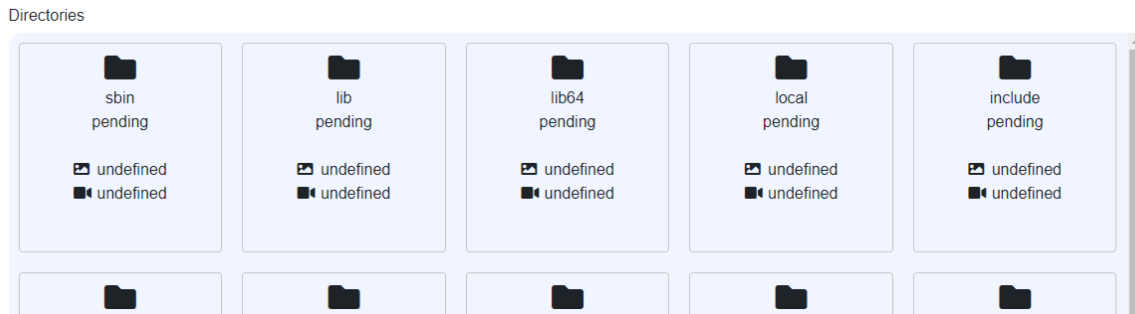
The image below shows the selection of a directory with a click, now the user can send the directory to garbage (deleted files) or retain it by clicking the buttons below the file description.

Another possibility is to click again on it (not on the buttons) to enter the directory



As we click on the directory, the system will start loading its contents and this operation may take a long time depending on how many files their sizes/types and subdirectories and their complexity.

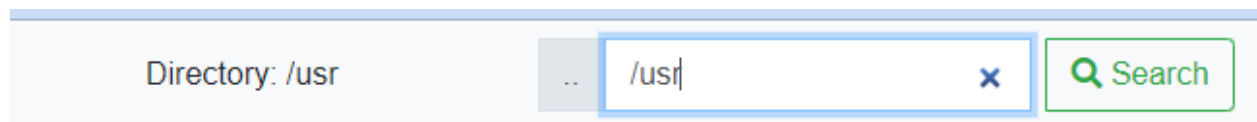
The detailed information on how this happens is show later on another section of this document, also have in mind that the long wait should only happen once per directory and only the big ones with many videos and subdirectories will have a really long loading time



Notice the directories with the pending information below their name, this indicates that the system is still working to get the actual size and the count of images and videos under this directory. Once the information is retrieved for a given file the system will sort it by size, the biggest ones will always display at the top



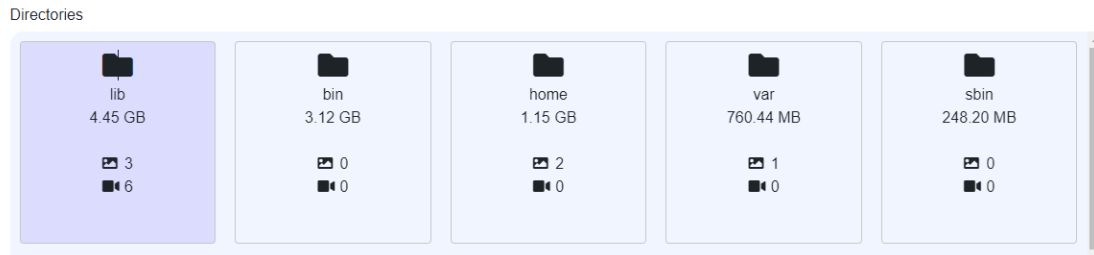
Now notice the button beside the address/location input, we can click on it to move up one directory ( and return to / on the example )






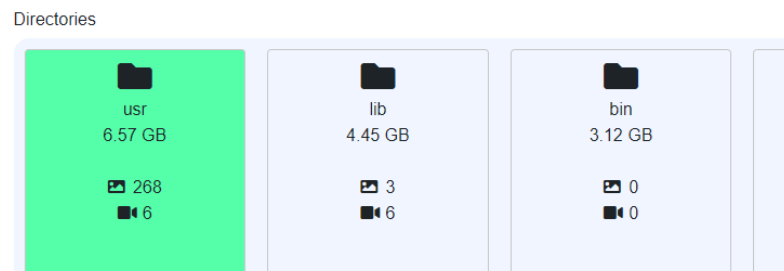
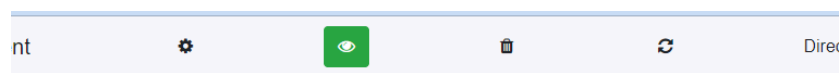
## Retention

Back on the root directory / , we can now click on the usr directory again and click retain, it will make it disappear from the screen and the /usr directory will get marked as a file to retain within the database



Notice the image above does not shows the icon for the /usr directory anymore, to show it we

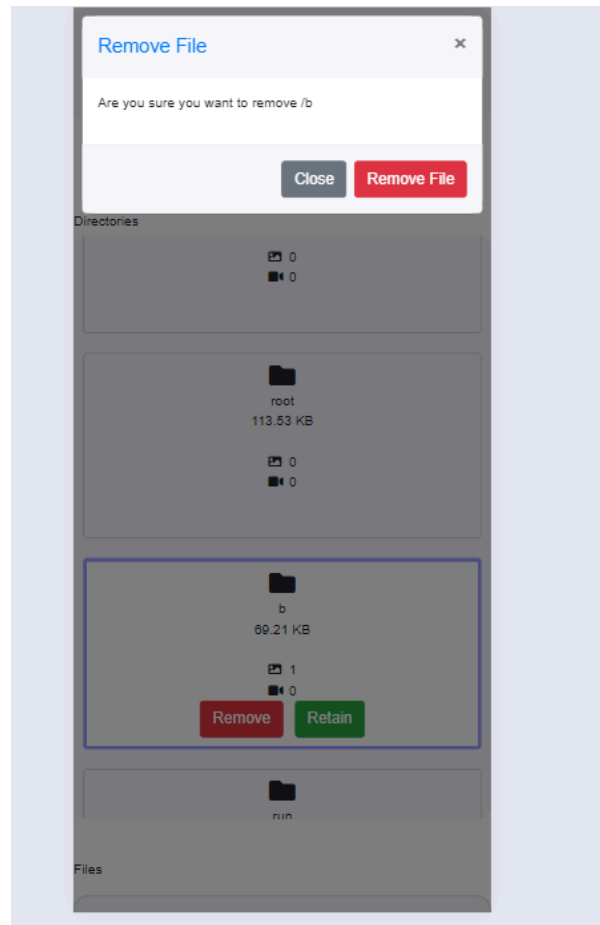
can click on the eye  icon, it will turn to green indicating the display of retained files is enabled as shown on the image below.



To remove the file/directory **from the retain state** just select the file and click the retain button again, the highlight will be removed indicating it is not retained anymore


## Removal

To send a file to garbage just click the Remove button below the selected file's name and confirm on the dialog as the next image




For this example we are using /usr anymore for obvious reasons. Also notice we are using the mobile layout to show how the system would adapt small screens

## Garbage view

By clicking the trash  icon, the system will load the contents under the garbage directory


The next image shows the view on the garbage directory, it works the same way as an ordinary directory, it will display files and directories, their properties and thumbnails, or even allow display/download. The only difference is that one can not retain or remove files on this view



Notice also the trash icon now as become a home  icon, by clicking on it the user returns to the normal view

## Invalidate

If you want the system to remove existing information on the database regarding the current directory and its contents to force it to recreate the entries and its media cache you can use the

invalidate  icon. Notice that it will not remove entries marked with status retain

## Internals

### How the listing works,

Internally the system will use the os library to query for files, then for each file it will query the database for existing entries, check its modification time, if file is fresh we skip to the next, if not the system will proceed checking the file type, if it is a regular file the system will query it's size, save the entry on the database and return the information to the user, if the file is a directory the system will not query the operating system for the file size, it will mark this information as pending on the database and will be addressed

### How cache works

Once a file entry is on the database one of the system's threads will check for its data status, if there are pending information, like size, one thread will be created to query these from the system. If the pending information is the thumbnail the system will use either the Pillow library or FFMpeg depending if it is image or video respectively, to create the thumbnail under /static/thumbnails

### File freshness and visualization updates

While listing files and directories the system will let some of the information as pending for performance improvement as previously mentioned, for this reason the frontend needs to periodically check the system for any existing updates on the file entries on the database.

The update check is done has an adaptive interval, from the moment a user queries for a directory the system will set the query interval to 5 seconds, then each time the query returns no new information the system will add 5 more seconds to the timeout to the maximum of 30 seconds of timeout, if a query returns an updated file the timeout returns to 5.

This algorithm allows enough time for long operations like guessing the size of a big directory size while not generating a long delay on the actual display of the files, enabling the possibility to operate on other files on the same directory

## Routes

Once the user lands on the home screen, the system will use ajax to query the system's API for the actual data, here are the routes provided by the system

/

The main route will either send the user to the home screen on the login area if is not already authenticated

/login, /logout

login receives a user and a password to check against a valid entry on the database and send user to home screen with a valid session, logout will delete the user session

/start

This is the main route to list files and navigate through the system, it receives a directory argument and returns a list of files/directories

/query, /querydir

These routes are queried periodically to get information about files and directories with pending information

## Actions

These are the file operations

`/retain`

This toggles and queries the retain status of a file

`/move`

Moves a file from source to destination, it receives 2 parameters, source file and destination

`/remove`

Moves a file to the garbage directory set on the database

`/invalidate`

Will remove entries from the database matching a given path, it **will not** remove entries with retain status

`/serve`

Transfers a file from the server, the method for this route is get and the parameter is filepath