# Improving the digital collaborative practices of activist structures by enhancing the open source sharing and collaboration software Nextcloud

Romain Lebrun Thauront

From **September6** to**2021 February 252022**

**Company tutor:** Pierre-Yves Gosset
**Technical tutor:** Tristan Roussillon
**Humanities tutor:** Virginie Muller


Locaux Motiv'
10bis rue Jangot
Lyon69007
(+33 | 0)6 60207080
contact@framasoft.org


Bâtiment Claude Chappe
20 avenue Albert Einstein
Villeurbanne69621 CEDEX
(+33 | 0)4 72436060
tc-s@insa-lyon.fr

# Acknowledgements

I would like to thank the whole Framasoft association and all the residents of the Locaux Motiv' at bis10 rue Jangot for their welcome, and especially the Framasoft team, Angie, Anne-Marie, Aurore, Chocobozz, Josephk, Luc, Pouhiou, Pyg, Spf and Tcit. For your human approach to the work world, your openness, your support, your good mood and everything that makes the work atmosphere at Framasoft! A special thanks to Thomas Citharel for his quick and precise answers to my convoluted questions and to my internship tutor,
Pierre-Yves Gosset, for having proposed to me to take part in this adventure and for the ortho-typographical rereading of this report.

# Table of contents

# Table of figures

# 1 Framasoft, a popular education association for digital issues

Framasoft is a non-profit organization created in 2004, whose purpose is "popular education on digital issues and cultural commons" [1]. Although it is mostly known for its alternative web services and its campaign "Let's get rid of the Internet", its actions are very diverse and range from software development to the training of associative actors to healthy digital practices.

## 1.1 Activities of the association

### 1.1.1 Campaigns to raise awareness of digital issues

Launched at the end of the 2014,campaign to raise awareness of surveillance capitalism and its alternatives called "Let's unglue the Internet" has led, over three years, to the implementation of 37 web services to show that alternatives to GAFAM were possible.

However, despite a constant influx of visitors, maintaining this diversity of services was too much work for the association. Faced with this success, instead of recruiting and growing, Framasoft chose to decrease its size by gradually closing several services with the "Deframasoftisons internet" campaign. This political choice contrary to the entrepreneurial logic was motivated by several reasons: Framasoft wishes to remain a small association with at most employees10; Framasoft wishes to see several alternatives being created rather than becoming **the** big alternative; Framasoft wishes to concentrate on other objectives than the proposal of alternative services.

Other major projects have succeeded Degooglisons: initiated in late 2017, the "Contributopia" campaign aims to bring a wider public to contribute to cultural commons and especially to free software, but also to learn how to integrate these contributions when conducting an open source project. At the end Framasoft2021, launches 2 projects aiming at the digital empowerment of some targeted actors with responsible and ethical digital content. The first one, "Émancip'asso", aims to allow associations that wish to stop using Google, Amazon, Facebook, Apple and Microsoft (GAFAM) tools to take the step by training service providers to accompany such approaches. The second, "Fra- macloud", aims to use the Nextcloud tool, a collaboration and file sharing tool, as a pretext and as a means to make actors of social justice and social progress collaborate.

### 1.1.2 Software development

One of the actions of Framasoft is software development. Some volunteers and employees participate in the community development of free software used by Framasoft services, such as the mailing list software SYMPA or

the collaborative writing software Etherpad. But Framasoft also develops its own software, including PeerTube, an online video sharing software, Mobilizon, an alternative to Facebook events, and Yakforms, a forms and surveys software. Supporting these software consists in providing a significant amount of development work, animating the community around the software and reviewing and integrating the community's contributions to the software.

### 1.1.3 Enhancement of free culture

Framasoft aims at the cultural commons as a whole. Thus, Framasoft seeks to participate in the non-software commons. This is done by using free licenses for the publication of documents produced within Framasoft, for the books published by the Framabook publishing house, and for the graphic productions of the association. These last ones are obtained through services from artists of the free culture movement such as David Revoy who realizes the majority of the illustrations used in the communication and the software of Framasoft.

### 1.1.4 Networked actions

The decreasing position of Framasoft, in order to allow it to explore new projects, pushes the association to regularly separate from some projects and to pass them on to other actors.

This part of Framasoft's action is led on several fronts, one of the most important being the CHATONS collective. The Collectif des Hébergeurs Alterna- tifs, Transparents, Ouverts, Neutres et Solidaires (Collective of Alternative, Transparent, Open, Neutral and Solidarity Hosting Companies) is a group of associations, individuals or companies with various statuses offering alternative online services. The closing of Framasoft's services also aims to give more space to CHATONS on the alternative web services scene. This collective was founded and is still coordinated by Framasoft.

Framasoft has also been working for a long time in collaboration with free software and ethical digital actors such as APRIL or the Quadrature du Net in order to make known and defend the common values with these partners. Nevertheless, Framasoft is looking to form more and more partnerships with structures that fight for different issues than free software. The goal is to reaffirm the political intentions (even though they are not partisan) of Framasoft, to work towards a convergence of struggles, and to learn by observing structures different from its own.

## 1.2 Economic and social positioning

### 1.2.1 Financing

In the2020, association Framasoft is financed at 93.5% by donations [2]. The association does not have "customers" as such but beneficiaries who show their support by monetary donation. Some software developments are also funded by

donations.

through grants from private or public organizations. For example, several features of PeerTube and Mobilizon have been funded by the NLnet Foundation, such as searching for videos across all PeerTube instances.

However, the association is not against development services provided that these developments are related to Mobilizon, PeerTube or Yakforms software and that they do not conflict with the design and values of these software.

This financing by donations is double-edged: it allows a great freedom of action to Framasoft which is free to choose how to use its funds but it creates a dependence on donations which is a potentially unstable income.

### 1.2.2 Competition and market economy

Framasoft is an association under the French law of 1901, whose resources are almost exclusively based on donations (about 500 000€ per year), so it can be considered as an "off-market" structure. The structure is not in competition with other companies, because its goal is not to occupy or dominate a market or an economic sector. It cannot therefore be placed on a competitive footing with the GAFAMs, not only because of its voluntarily constrained size, but also because of its intrinsic objectives (respect of privacy, in particular).

Moreover, Framasoft does not really have an equivalent in France, in Europe, or even in the world. Indeed, the association is at the crossroads of different fields: popular education, web services hosting (SaaS), and software development. This gives it a unique position, especially in the free software ecosystem.

However, the services offered by Framasoft can compete with other alternative services. The association is aware of this situation and seeks to limit its impact in order to disseminate the offer of alternatives. This is done through mechanisms of limiting the number of accounts or storage space on various Framasoft services, and by highlighting competing alternatives.

### 1.2.3 The strengths and weaknesses of the association

Framasoft is an association with projects as varied as its members, and is in the midst of pivoting its flagship activity from "alternative service proposals" to "To lead the digital empowerment of activist structures". The association is well known for its online services, but its associative nature and its political positions are not well known. There is therefore a risk that the public will not understand this repositioning, which is already being felt with the closure of services. The association is therefore counting on its communication skills, its transparency and its collective decision-making to carry out this project successfully.

These different points of concern are summarized in the following SWOT matrix (Figure 1).

| | Positif | Négatif |
|---|---|---|
| Interne | **Forces**<br>Prise de décision démocratique et participative.<br>Compétences techniques et communicationnelles fortes. | **Faiblesses**<br>Coopération difficile entre les bénévoles et les salariés (pas les mêmes disponibilités ni les même attentes). |
| Externe | **Opportunités**<br>Reconnaissance du public, bonne image médiatique.<br>Réseau important au sein des millieux militants et des millieux de l'économie sociale et solidaire. | **Menaces**<br>Méconnaissance du public sur les objectifs et la taille réelle de Framasoft. Notamment visible sur la campagne de fermeture de services.<br>Dépendance aux dons.<br>Réactions négatives de certains acteurs ou publics face à des positions politiques plus assumées. |

Figure1 - SWOT matrix of the Framasoft association in 2021

## 1.3 Organization of work in a structure that combines salaried and volunteer work

Framasoft is an association "law1901" which is composed of members37, volunteers and employees. There is, in the statutes, no diférences between salaried or voluntary member. It is managed by a co-direction consisting of Pierre-Yves Gosset and Pou- hiou Noénaute. The role of the co-directors is to coordinate all the actions of the association. The coordination of certain areas is delegated to specific committees, such as Human Resources (see Appendix A). The committees are executive and not part of the decision making process.

When decisions impacting the association are to be made, they are made in a way that is intended to be democratic. The whole association is informed of the ins and outs, and if the subject is debated, a majority vote is taken. This vote can be held in a general assembly (GA) for the most important subjects (See Appendix B).

While there are no differences between volunteer and salaried members in terms of decision-making, the two groups have different expectations, needs, and available time and these differences must be reconciled. Thus, there is both a desire to involve the whole association as much as possible and a desire to leave some autonomy to the employees. For example, although the majority of instant messaging channels and e-mail lists are open to all, there are channels reserved for employees for daily work life and certain organizational aspects that are the sole responsibility of employees. The autonomy of employees also includes the autonomy of all members, who are free to make and assume decisions that they do not consider impactful for the association.

# 2 Framacloud, creating collaboration between activist structures

My internship at Framasoft is part of the launch of a larger project mentioned in section 1.1.1, the "Framacloud" project. This project aims at using the Nextcloud tool as a lever for collaboration between actors of social justice and social progress [1].

Nextcloud is a client-server software for online file sharing and collaboration and can be seen as a free alternative to Google Workspace. It is a fork of OwnCloud with a growing user community and commercial support from Nextcloud GmbH, which leads its development.

The Framacloud project is based on the observation that mastering digital tools, and in particular the ability to collaborate online, is a determining factor in the impact of an organization on society. Associations, and even more so associations working for progress and social justice, are particularly vulnerable to this problem. If the use of information technology is well established in the work of associations, they often suffer from a lack of skills and a lack of ethical alternatives [3] (this last point can be crucial for militant structures).

The objectives of Framacloud are therefore to put forward a collaborative platform for activist structures, to adapt it to the needs of associations and to be able to provide access to all the small activist structures that need it[4].

In order to achieve these goals, Framasoft has bet on Nextcloud. In spite of these various defects, Nextcloud seems to be the most successful and promising alternative solution. The project will involve Nextcloud developments, but also the increase of the current Nextcloud offer of Framasoft from account5000 to account.50000

The launch of Framacloud was planned for the end of 2020, but the pan-demic context of the last few years has delayed the project. So the beginning of my internship marks the launch of the project. I had to realize, in addition to a development work, an exploratory work to refine our understanding of the possible perspectives via this software. I was also quite autonomous in my development work because I was the only developer on this subject. Nevertheless, I had a follow-up of the project, notably via my internship tutor Pierre-Yves Gosse who also works on the Framacloud project. I was also able to benefit from the technical advice of my colleagues and in particular Thomas Citharel, who is one of the main contributors of the Nextcloud calendar management *plugin*.

1. By the term "actors", we wish to encompass a wide range of organizations: associations, trade unions, informal groups, companies, ...

# 3 *Sorts* : Give Nextcloud users a view on their files

The objective of my internship is to improve the use of the Nextcloud open source software by an activist and associative public. It took place in three parts, as represented on the Gantt chart of my internship in appendix C: a first part of training and skills development, a second part of survey and needs analysis, and a third part of design and realization of a technical solution answering one of the identified needs.

This work of identification and response to a need led to the development of "Sorts", a Nextcloud *plugin* allowing users to better visualize their files thanks to a tree-like file explorer and conditional filters.

## 3.1 Getting to grips with Nextcloud development and related collaboration processes

During this first part of the internship my goal was to acquire enough skills and knowledge about Nextcloud software development to be able to choose the software development I would work on during the third part.

This increase in skills was a bit long but it was necessary to discern the feasibility and the time required by the improvement proposals of the Nextcloud users that we surveyed during the second part of the internship. Moreover, the end of this skills development was intertwined with the realization of a survey and my participation in the support of Yakforms (which we will talk about hereafter) which induced a delay on the end of this part. (See Gantt chart in appendix C)

### 3.1.1 Development environment

Nextcloud being a fork of OwnCloud, its development starts with the development of OwnCloud in Thus2010. its original software *stack* follows the classic model of LAMP web applications: A backend in PHP, a frontend in JavaScript with jQuery, distributed by an Apache server and supported by a MySQL database.

However, the software has been modernized: other databases and web servers are supported, the current developments are in recent versions of PHP (PHP 7 and 8) and the frontend is gradually rewritten in VueJS [5]. So I developed with these two technologies, and set up the following development server to test and visualize my developments locally:

— Apache server (localhost)
— SQLite database (localhost)

— PHP8 with Xdebug
— Nextcloud (Git-release, then Nextcloud 23)

During the last month of development, we also set up an external test server linked to a GitLab continuous integration system (see part 3.3)

If I had already had first experiences of web development during my studies at INSA and during my internship at equensWorldline,4 I had never worked with PHP or VueJS. So I started by learning the basics of this language and framework. Then, I realized some first dummy *plugins* and contributed to the development of Nextcloud by solving some *good first issues*.

### 3.1.2 General operation of a Nextcloud *plugin*

My skills development was done through the exploration of the developer documentation, which allowed me not only to learn the PHP language but also to understand the functioning of the Nextcloud *plugin* system.

A Nextcloud *plugin consists of* a folder containing all of the application's backend and frontend code, along with its dependencies and credentials. Figure 2 details the main folders of a Nextcloud *plugin.* The PHP code and credentials of the *plugin* are discovered by Nextcloud via reflexivity mechanisms. In particular, the URLs defined in the plugin are registered in the Nextcloud router to load the plugin when requests are made on these URLs.

A Nextcloud *plugin* also has access to various APIs provided by Nextcloud that allow the *plugin* to integrate with the software, and access Nextcloud data structures and features such as files, users, access rights or shares.

For the frontend, the API takes the form of default JavaScript functions, pre-packaged dependencies for Nextcloud (e.g., a pre-packaged ajax REST client with the Nextcloud instance URL), and VueJS *components* to build a consistent user interface between the *plugins* and the Next-cloud *core.*

For the backend, the API takes the form of PHP namespace: \OCA is dedicated to the code of the *plugins*, \OCP is dedicated to the public APIs that the *plugins* can import and call in order to access the functionalities of Nextcloud, and \OC is reserved for the private APIs (used by \OCP)

In order to understand how the above works and how the main Nextcloud APIs work, I developed two dummy *plugins.* The first one is the Nextcloud tutorial *plugin*, allowing the management of simple notes, stored in database; the second one is an original *plugin*, allowing to manage simple *todo lists*, stored in text files.

These two experiments allowed me to understand the fundamentals of

```
plugin_name/
| COPYING
| Readme.md
| appinfo/
    ||--                        info.xml//Identification ofthe plugin
|`--                            route.php//URLfor which to load the plugin
                                  |(andPHP functions to call)
|-- lib///PHP code              , automatically loaded when
                                  |a' query
    ||--AppInfo
    ||`--  Application.php '//Entry point of the \textit{plugin} during
its
||automatic                          loading
    ||--Controller//Point     ' of    entry of http requests
                                  |definedin routes.php
|`--...                          // rest of the PHP code for the
                                   |completion of queries
                               node_module///Frontend dependencies
                               src////Frontend code
|-- Test Units to test the
|Backend                           features
`--  vendor///                   PHP backend dependencies
```

Figure2 - Main folders and files of a Nextcloud *plugin*

backend of Nextcloud *plugins* and the functioning of the most used APIs, but also to learn how to use the Nextcloud documentation.

### 3.1.3 Collaborative process in open source development

This period of skill building also allowed me to get used to the work processes in which my internship took place. These are the processes of Framasoft where I was integrated into the team of all 10 employees but also the processes of collaboration in the development of an Open Source software, rich in interactions with the Nextcloud community and the developers of Nextcloud GmbH.

Regarding the participation in the development of Nextcloud, the developments carried out by Nextcloud GmbH follow a classic open-source development scheme: the versioning is done via Git and the code is hosted on the GitHub platform which serves as a project tracker and *bug-tracker* and is a gateway for communication with the community. However, all interactions on GitHub are focused on code and technical aspects. The Nextcloud help forum is

therefore there for the more general communication between Nextcloud GmbH and the community.

Not knowing yet at the beginning of my internship if my technical subject would consist in a *plugin* or developments in the *core of* Nextcloud, this rise in skills also had for goal to realize a first contribution to the development of Nextcloud. So I contributed to different official Nextcloud *plugins* by fixing small bugs or developing small features. For example, I added a button to empty the "trash" of Calendar and Todo applications (see figure 3 and public merge request on GitHub https://github.com/Nextcloud/tasks/pull/1802).

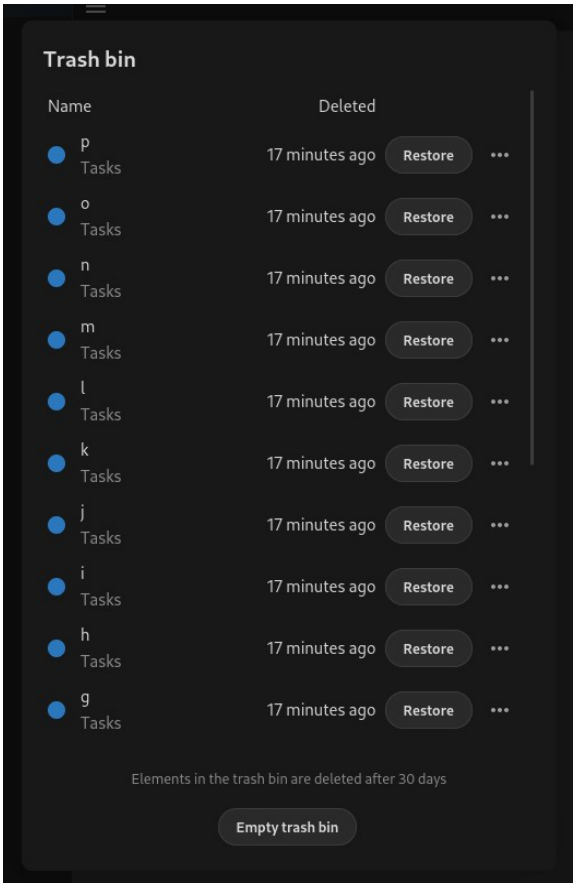



Figure 3 - Todo application trash garbage can interface, with button to delete all items (Empty trash bin)

This first contribution allowed me to better understand how a VueJS frontend works, especially the *component* layout, but more importantly to understand the process of contributing to an Open-Source project:

1. A bug or a missing feature is identified

2. Creation of an *issue* on the corresponding github *repository*
3. A developer wishes to resolve this *issue* and states his desire in the comments of the issue
4. Fork of the source code in question
5. Opening a new Git branch
6. Development
7. Merge request citing settled outcome
8. Return of one of the *maintainers* of the *repository* in question and improve the merge request until it passes the automatic tests (*pipeline)* and is approved
9. The merge request is integrated into the upstream code and the *issue* is marked as resolved.

When we opened the development of the "Sorts" *plugin* to internal contributions at Framasoft, I could also see this process from the *maintainer* side, by reviewing and accepting merge requests proposed by my colleagues (see part 3.3).

## 3.2 Designing a *plugin*: from the analysis of the need to the technical so- lution

At the heart of the Framacloud project is the desire to democratize collaborative digital technology and to meet the needs of activist groups for progress and social justice.

We therefore had a strong desire to identify the needs of targeted users before embarking on a technical implementation. This was concretely translated into a work of exploration of Nextcloud and the writing of a short report of astonishment which makes place of the frictions which I could have had with the use of the software, but especially with the realization of the survey "Nextcloud and the structures of the social progress and the social justice" whose results will be published during the year.

### 3.2.1 Studies of the limits of Nextcloud in an associative use

In order to help us with the creation and coherence of the survey form, and thus limit the number of biases we could have had in designing and distributing it ourselves, we called on two external service providers: the designer Marie-Cécile Godwin and Stéphanie Lucien-Brun from La Fabrique à Liens, a specialist in "digital empowerment".

The goal of this survey was to target the friction points that social progress and social justice organizations already using Nextcloud had encountered, as well as their expectations and uses of the software. Having now a good vision of Nextcloud functionalities, I proposed first prototypes of survey

which fed our reflections on the questions to be addressed to deal with the different use cases of the software and on which Marie-Cécile Godwin was able to rely to build the final survey form.

This survey was distributed at the end of October through Framasoft's communication channels. I was then in charge of making a synthesis of the first 180 results collected during ten days of survey with the aim of highlighting the most important expectations of the targeted public.

Out of 174 results, 143 results matched the intended audience and were retained in my analysis.

In order to extract trends from these results, I manually added keywords to each response, characterizing the main concerns of the response. Then, using a python script, I sorted the keywords by number of oc- currences and compiled in a document, for each keyword, all the answers referring to it. Finally, from this document, I wrote a summary of the respondents' frictions and expectations by keyword.

The most important concerns identified were:

1. Collaborative editing of documents (text, spreadsheet, ...) is not sufficiently per- formant
2. Nextcloud is not powerful enough (slow software)
3. The integrated video conferencing application is not powerful enough compared to the alternatives
4. Synchronization of files with devices is too erratic
5. Users have trouble finding their way around their organization's file tree.
6. The sharing functionality is neither intuitive nor ergonomic

Of these problems, the first two are outside our scope. The first one concerns external software (OnlyOffice or Collabora) and participating in their improvement would have required a new training period of several months. The second one concerns the general optimization of the whole software, on which we can't do much during a limited training period.

Topics 3 and 4 are topics on which we would have had more capacity to act, but for which the software developments might not fit into the remaining internship time.

So we decided to go with the fourth topic and help users better understand the hierarchy of their files and folders shared by their organization.

This survey was crucial for the choice of my technical topic and to guide its realization, but it is also an important resource for the rest of the Framacloud project. Nevertheless, it should be kept in mind that it has some biases,

In particular, the fact that among these 143 responses, 50% of the respondents claim to work in the digital sector and are therefore likely to have a digital culture that is not representative of the target user group.

Moreover, as far as my synthesis is concerned, my manual categorization and my effort to synthesize necessarily induces a bias of my own, even though I have tried to remain as faithful as possible to the original answers.

### 3.2.2 Spell Prototyping

Thus, we looked for ways to improve the navigation of users through their files, while being limited by several constraints.

The first was that there were three months of internship time left to devote to development, which included end-of-year vacations, writing the internship report, and the internship defense.

The second is that the official "Files" application is very *legacy*. The majority of its code dates from the fork with OwnCloud and has not undergone any major development. Nextcloud GmbH plans to rewrite it soon, but this is not on the roadmap for the next two versions. Proposing modifications to the current "Files" application or participating in its rewriting therefore did not seem feasible to us.

So we decided to develop a *plugin* rather than to propose upstream developments. The development of a *plugin* allowed us to have full control over the features we wanted to develop and the certainty that my developments would be published at the end of my internship.

However, from the point of view of Nextcloud software, having many *plugins* with few supports is a weakness. Moreover, an unofficial Nextcloud *plugin* may be less used by users and especially associations. They don't always have the management of their instance and no or little control on the applications installed on it, and some hosting companies may refuse to install unofficial applications.

So, we are not aiming so much, with this *plugin, at* the direct impact as at the indi- rect impact: by proposing new concepts around the interactions with the files, the *plugin* that I have developed acts as a *Proof Of Concept* which, we hope, will influence the rewriting of "Files".

In order to have a "roadmap" for the development of this new *plugin* (called "Sorts"), I defined with my tutor, Pierre-Yves Gosset, all the functionalities we wanted to implement in order to facilitate the access to :

1. Suggest a new file explorer

(a) Putting forward more metadatas, including tags that users can put.
(b) Allowing to "scroll" the file tree for a better understanding of the classification of files and folders.

2. Propose a multiple filter system
   (a) Allowing the user to combine search conditions on all available metadata (size, weight, tags, ...)
   (b) Allowing quick access to the most used filters (through default filters, history or favorites)
3. Stop at viewing, and do not allow modification of files
   (a) Allows to reduce the "danger" of the *plugin in* case something goes wrong.
   (b) Allows for the development of a finished product within the allotted time (implementing modifications would mean redeveloping the "Files" application)
   (c) Modifications can be made in the official "Files" application, providing a mechanism to open files and folders found in "Spells" in "Files

From this scope I created an interactive prototype of the user interface of
This is the first time that the "Sorts" program has been used with the "Penpot" web application.

Figure 4 shows the "file explorer" view of the prototype: the central panel consists of a list of files with the "Project" folder unrolled; the *external* symbol after the name of each file allows it to be opened in Files; unrolling a folder is done by a single click and entering a folder is done via a double click. Exit a folder (go back up the tree) is done via the *breadcrumb* at the top of the interface (symbol *Home* > Documents); filters are accessible via the search bar next to the *breadcrumb* and the side navigation panel allows quick access to favorite searches.

The figure 5 shows the filter selection view and its results: the upper part of the interface shows all combinable filter criteria and the lower part is similar to the file explorer, but the files shown do not belong to the same folder, the path of each file is specified next to its name. The folders are also scrollable to allow the user to quickly inspect the contents of a folder corresponding to the query sent.

This prototyping stage allowed us to quickly realize the elements necessary for a good user experience and to quickly gather feedback on how we envision the *plugin*.

We used it to publish a note of intent on the official Nextcloud forum, which led to feedback from the Nextcloud community and also, albeit a little late, from developers at Nextcloud GmbH.

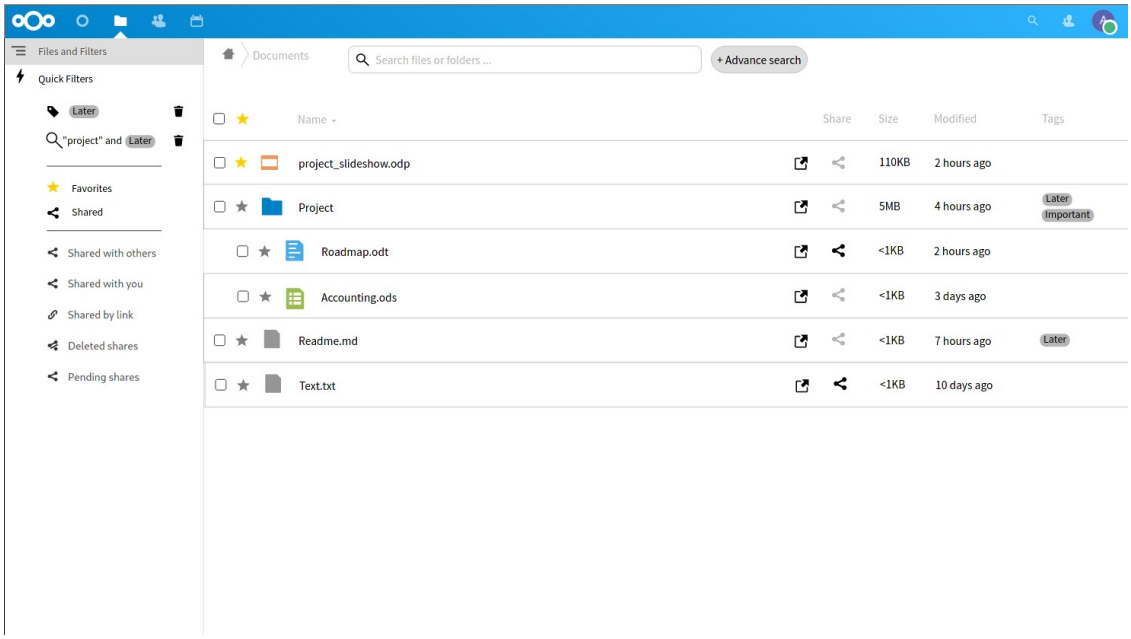



Figure4 - Interactive prototype of the "Sorts" *plugin*, file explorer

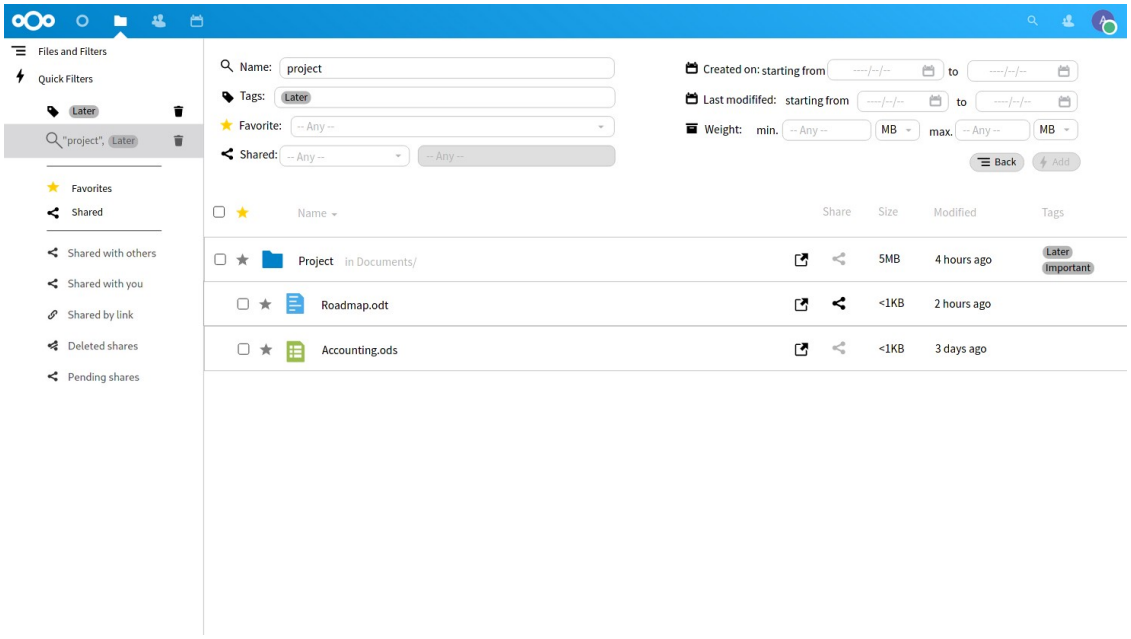



Figure5 - Interactive prototype of the "Spells" *plugin*, combined filters

This feedback led to a meeting with Nextcloud GmbH about "Sorts". This meeting allowed us to establish a first contact with Nextcloud GmbH and to consider how my developments could have the best added value for Nextcloud. Indeed, the multiplication of little followed applications is not ideal for the software ecosystem, but, as mentioned above, the deadlines of my internship did not allow me to pivot my technical subject at one and a half month of my end of internship. So we kept as a priority the development of
"Spells".

The prototype presented here was built quickly with this feedback whereas redeveloping features would have been much more time consuming.

## 3.3 Development process of the Sorts plugin

The development of "Sorts" started as soon as the prototyping was finished. However, the beginning of the development was more a phase of setting up the project. It was a matter of re-modeling the code base of the "tutorial" application of Next Cloud in order to start with a blank application model, to open a *repository* on Framagit, the GitLab instance of Framasoft, and to set up a development schedule.

So I made a first decomposition of the *plugin* in big functionalities, to which I assigned a development time:

— Backend of the file explorer
— Frontend (UI) of the file explorer
— Backend of simple filters (weight, size, file names, ...)
— Frontend (UI) for filter selection
— Backend of more complex filters (tags and sharing information)

But these forecasts came up against the reality of software development. In particular, I had trouble keeping a real division of tasks in the first few weeks of development.

Starting from a blank sheet of paper, the development of the first elements of the ba- ckend and the frontend were very interdependent. Once a first version of the file explorer was done, it was much easier to follow sub-developments of tasks defined with the classic model of one Git branch per *feature*. This improvement is clearly visible in the Git history of the plugin which shows regular *commits* from the beginning of January 2022.

This improvement allowed me to set up a second schedule more accurate of the development times, if we don't count the refactorings of the source code, necessary step with my understanding of the technologies used which was growing during the development.

The development time being relatively short, I concentrated on

the functional aspects of the plugin, and left aside the aesthetic considerations of the user interface for which I have less knowledge and less interest.

These aspects were nevertheless addressed at the end of January and beginning of February with the participation of JosephK and Thomas Citharel in the development of "Sorts". This first opening of the gSorts code internally allowed us to set up and test the collaboration processes around the *plugin*. I was able to read, comment and accept merge requests from my colleagues on aesthetic aspects but also on bug fixes of the *plugin*. This collaboration required me to write the project roadmap and prioritize the tasks to be done for a Beta1 release, to be published on the Nextcloud *plugin* store.

This implementation of a multi-party development environment also allowed the implementation of a continuous integration system and GitLab *pipeline* similar to what is done in professional software development.

Continuous integration allows the deployment of the latest changes to the software on a development server, which allows us to visualize and test the latest integrated developments.

The pipeline system allowed to test the correct functioning of new modifications before their integration thanks to a chain of tests of the source code, but also to standardize the coding conventions of the project (*lint* ).

## 3.4 Details of how Spells work

This part deals with the main technical choices made during the development of the "Sorts" plugin. "Sorts" being a free and open-source software, its source code is available online at the following URL: https://framagit.org/framasoft/nextcloud/sorts, and the reader can refer to it, if he wishes, for more details on the plugin's functioning.

The general functioning of a Nextcloud *plugin* having already been explained briefly in the 3.1.2, section, I would like to specify below the main lines of the specific functioning of "Sorts".

### 3.4.1 General concepts of backend and frontend

The backend of "Sorts" implements a REST API whose purpose is to provide the frontend of "Sorts" with a list of files and their respective metadata, via a JSON object. It does not implement an API for external clients (*cross-origin*).

Metadata are all information about a file, which are not its content, for example its name, its size or the tags associated to it. The metadata that are displayed in "Sorts" come from different public APIs of Nextcloud [6] :

— \OCP\Files
— Name

    - Size
    - Date of modification
    - Favorites
- \OCP\SystemTags
    - Tags
- \OCP\Share
    - Shared with
    - Shared by
    - Shared by link

Managing the different APIs is one of the challenges of "Sorts", both because retrieving a list of files results in at best three database queries (one per API), and secondly because the basis of Nextcloud's conditional search system is part of "Sorts" and the integration of parameters relating to tags and sharing information requires trade-offs that are discussed in the section 3.4.3
.

The frontend of "Sorts" is a VueJS application. It is an assembly of several VueJS components, the details of which can be seen in figure 9, detailed in the 3.4.4.

Using VueJS allows you to take advantage of its *responsive* display system: when a JavaScript variable linked to an element of the HTML page of the user interface is updated, the elements of the user interface linked to this variable are automatically refreshed (this is notably the case of the list of files which is stored in an object array).

### 3.4.2 Design of a tree-like file explorer

Between the prototype (figure 4), and the *plugin* (figure 6), there are some differences. We have chosen to leave the filter selection interface always visible, because being the major *feature of* the *plugin* we wanted it always accessible. We also removed the possibility to search by creation date, because, if the attribute exists in the PHP objects provided by Nextcloud to describe a file, it always has a null value.

To retrieve the contents of a folder, an HTTP request is made to the "Sorts" backend, which makes a first call to the "Files" API to retrieve the list of contained files with their metadatas, then this list is passed to two other APIs to retrieve the tags and the information related to the sharing of the document. At the end of the chain, this process results in three requests to the database.

The question of the number of database queries is a major concern because it is often the most limiting factor on large instances with several thousand files, especially since a database explorer can be used to search the database.

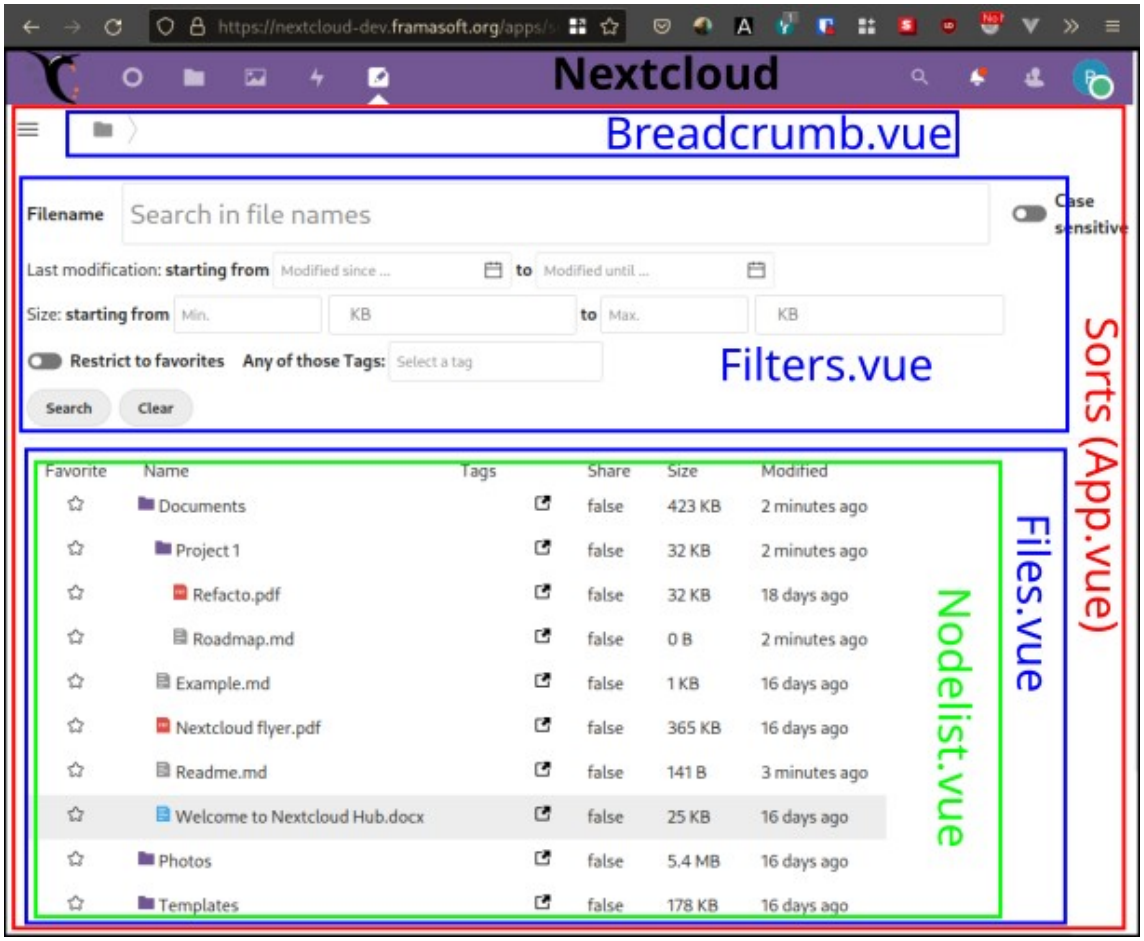



Figure6 - "File Explorer" view of "Spells

The consequence could be to make the service unusable by overwhelming the server with database queries, or to make the user interface unusable by loading too many items at once. The consequence could be to make the service unusable by overwhelming the server with database queries, or to make the user interface unusable by loading too many items at once. Two solutions seemed possible to limit this problem: paginate the displayed files and avoid cases where several folders could be reloaded at once.

Pagination seemed to us to be adapted to the display of files and folders responding to a filter (display of *n* results, with a button to request more), but unsuited to the exploration of the tree structure. Indeed, the whole advantage of a tree file explorer is to be able to see the content of a folder in its "context", the context being the content of its parent folders.

So I decided not to page the contents of a folder, but to avoid reloading folders recursively. The consequence is that when a user goes back to the file explorer the parent folder is reloaded but not the child folders which were already open and therefore may not be up to date anymore. Secondly, the information of the already opened folders is lost when the page is reloaded. Thus, reloading a page will only reopen the folder at the root of the viewed tree and not all open subfolders (the folder currently at the root of the viewed tree, as well as the metadata and the order in which the files are displayed, are stored in the URL as shown in figures 8 and therefore they 9,persist when reloading).

Implementing a passive refresh mechanism for the files viewed when they are modified by an external source is one of the prospects for improving the *plugin*.

The tree file explorer also raised the question of how to display a tree of variable depth (the files, with different folders open) with VueJS. The VueJS directives for controlling HTML contain only conditions and *foreach* loops (*v-if* and *v-for* ) but the depth of the tree being variable, displaying it requires a recursive algorithm or *while* loops.

To solve this problem, I decided to flatten the tree into an array by calling a recursive function `func` (`nodesList` being the flattened array and `nodesTree` being the file tree) :

```
computed: {
    /**
     * Return the plane array of nodes, with depth properties on each nodes
     *
     * @return {object|null}
     */
    nodesList() {
        if (this.nodesTree === null) {
            return null
```

```
        }
        const result = []
        const func = function(nodes, depth) {
            nodes.forEach((node) => {
                node.depth = depth
                result.push(node)
                if (Array.isArray(node.nodes)) {
                    func(node.nodes, depth +1 )
                }
            })
        }

        func(this.nodesTree,0 )
        return result
    },
}
```

The resulting array contains the "depth" information of a file, which is directly used for indenting the files contained in a subfolder. The nodes of an open folder are also processed and inserted into the flattened array directly after their parent, which keeps them under their respective parent.

This nodesList array is declared as a function because it is a *computed property*. This is a VueJS mechanism: the object resulting from a *computed property* is computed at its first use and kept in cache; as long as the data with which the *computed property* is used does not change, the cache is returned at an access rather than recomputing the function.

Moreover, VueJS recalculates the *computed property* and refreshes its display automatically when the data on which it is based is modified (in our case, as soon as the nodesTree file tree is modified).

On the figure 6, we can see that the display of files (and therefore the code presented above) is managed by the Nodelist.view component. The reason for managing it in a sub-component rather than in Files.vue is that the display of files is common to the file explorer and the display of filter results. The Nodelist.vue component is therefore reused in 2 different contexts. The separation into *frontend* components is discussed in more detail in the section 3.4.4.

### 3.4.3 Conditional Filter Design

There are two search modules available through the Nextcloud PHP framework. The first one is the SearchProvider which is used in the global search of Nextcloud. This global search allows you to search for the same string among all the resources present in the Nextcloud instance (be it files, appointments, emails or conversations). Each app-

cation wishing to extend this functionality implements the SearchProvider interface in order to return results, and the SearchProvider linked to files only searches in the names of the files or in the content of the files. This API, which is the most known search function of Nextcloud, does not correspond to conditional filters, whose added value lies in the ability to filter other characteristics such as weight, modification date or tags.

So I used in the realization of "Spells" the second API of research: \OCP\Files\Search, which already implements a filter engine on the metadatas of \(see section 3.4.1).

This filter engine defines an ISearchOperator interface, which contains the desired filtering conditions, a SearchOrder class, which contains the order and the name of the metadata by which to sort the results, and a SearchQuery class which defines the whole query: an ISearchOperator, a SearchOrder and paging parameters (the maximum number of results to return, and the offset of its results)

Two classes implement ISearchOperator:

1. SearchComparison which defines a filter on a metadata, composed of :
   - a *Field*, the metadata to compare
   - a *Type*, the comparison to be applied :
     - equal (strict)
     - strict superior
     - greater than or equal to
     - strictly inferior
     - less than or equal to
     - similar (if the provided string is contained in the search field)
     - similar, but with the same box.
   - a *Value* : the value to compare (a number, a string or a boolean)
2. SearchBinaryOperator which defines a logical operation
   - AND which is performed on ISearchOperator2
   - OR which is performed on ISearchOperator2
   - NOT which is performed on a single SearchComparison

A set of conditions is created by combining SearchComparisons into SearchBinaryOperators, the stacking of which results in a SearchBinaryOperator containing the set of conditions for performing the desired file search.

Thus, to perform a search on a user's files, the FilterManager class in the "Sorts" backend retrieves a JSON description of the filters specified by the user, and then the corresponding SearchComparison are instantiated and combined by SearchBinaryOperator AND. A SearchQuery is instantiated with the SearchBinaryOperator and a SearchOrder, and passed to the Folder::search() function at the root of the user's files.

user.

This function will solve the query by creating a corresponding SQL query using a PHP framework for creating SQL queries.

The difficulty in implementing these filters on the Files API metadatas was to find and understand the API \OCPilesearch, which is little used and little documented.

Its use allows us to have confidence in the accuracy of the SQL queries generated, to have a single SQL query with joins regardless of the number of criteria chosen and has the advantage of offering *out-of-the-box* pagination. However, its problem is that this system does not integrate the metadatas linked to tags and sharing information. It is therefore necessary to use other APIs to filter these metadatas and find a way to combine them with existing results.

Using the SystemTags API, we can retrieve the list of files tagged with a certain list of tags, and using the Share API, we can retrieve the list of all files corresponding to a sharing information (shared by me, shared with me by so-and-so or shared via a public link). But these are independent requests. So, you need 3 queries to the database to filter on all the criteria, and they will each return as many or more results than the equivalent combined query. You then have to cross-reference these results somehow, and since you need the entire responses from all three queries to do this, you lose the advantage you had in paging.

On a large Nextcloud instance with several thousand files, such queries can be very heavy and impact the service. The use of "Sorts" in its current version is therefore not necessarily recommended for such ins- tances. This is one of the most critical problems of the *plugin* and there are three possible solutions to solve it:

1. Rewrite an equivalent system to include the ability to add SQL tables containing tag and share information to joins.
2. Modify the API to include tag and share searches.
3. Do not make combined requests for these two metadatas but one request per API.

The first solution requires too much time for the internship and the second one is not possible because such an API modification will probably be refused by Nextcloud GmbH. Indeed, modifications to the *core of* Nextcloud are not accepted without a strong need and a long period of testing to be sure not to introduce bugs in the applications that depend on it.

So "Sorts" performs three requests to retrieve three arrays of files that each *match* the criteria related to an API, then returns the intersection to the frontend

of these three tables.

This choice makes the use of filters by tags and sharing information unviable for large Nextcloud instances and therefore we do not recommend the use of "Sorts" in its current version. In the future of the application, we will have to mitigate this problem because the filters by tags and sharing information are very much expected by the users.

A first possible improvement is to proceed, in the context of a filter on the metadatas of the Files API and of at least one other API, in the following way:

1. Recover all files that *match* the filters on the Files data
2. Retrieve tag and sharing information for these files (as when you want to display them)
3. Browse these files with a loop, and filter by comparing their tag or share metadata with an `if`

If the request on the `Files` API already strongly limits the number of possible files, then steps 2 and 3 can be much less cumbersome than with the current system because the amount of data to be retrieved and processed is less.

A second improvement would be to still paginate filters that combine sharing information or tags with metadatas from files. For example, to return the first 20 results of files modified in the last month and having the tag "important", one could proceed as follows:

1. Retrieve the first20 files modified in the last month via `\OCP\Files`
2. Among these 20 results, isolate those with the "important" tag, either by the result intersection method or by the method using a result path.
3. If the number of isolated results is less than 20, request the next 20 results from the API, and return to step 2
4. Otherwise, we cut the isolated results to 20 elements, and return them with the API pagination index we stopped at.
5. Subsequent requests pass the id of the last item received and, as an off-set, the paging index of \OCP\Files rather than the response "batch" number they would normally provide.

As for the frontend, an *HTML form is* used to retrieve the user search criteria (see figure 7). This *form* makes use of several components from the Nextcloud VueJS component library: date selection fields, multiple choice fields, *switches*, and tag selection. The reuse of these elements allows a more unified look with the rest of Nextcloud, but also

not to "reinvent the wheel" and to be consistent with users' language preferences.

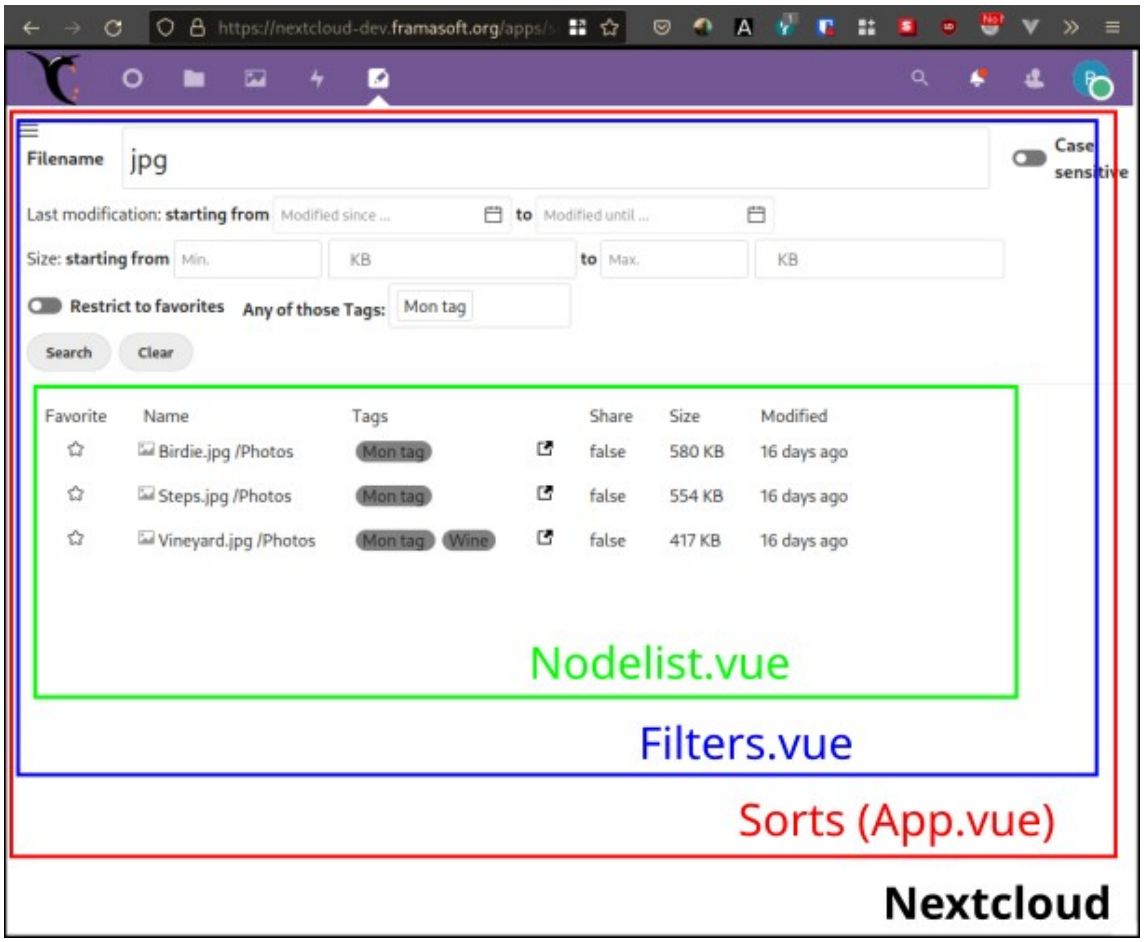



Figure7 - "Filters" view of "Spells

The details of the division into components of the frontend will be discussed in the next section (3.4.4)

### 3.4.4 Example of refactoring

One of the major interests of VueJS is the separation of the frontend into components. Any component can instantiate others via a simple HTML tag with the name of the component to instantiate.

Parent components pass information to child components via `v-bind:childvariable="expressionParent"` directives: the `child variable` is accessible to the child component and reflects at any time `expressionParent` which is computed in the parent component, and which can be a variable or a call to a method.

The child components can in turn send information to their parent via events: the parent registers at the creation of the component a method to be called when a certain event is emitted using a directive v-on:myEvent="methodParent()". The child can then emit events via the emit() function, which takes as argument the name of the event to be emitted and an optional variable containing data to be passed as parameter to the Parent() method.

By mid-January, after a month of development, the components of the *plugin* were arranged as shown in the figure A 8.first separation into components had already been thought out during development: the App component, which is the main component and the entry point of the application, loads the three components corresponding to the three parts of the user interface: the Breadcrumb, the filter form and the file list.

The problem with this structure is that it concentrates all the logic, data and state management on Nodelist. This is the component that retrieves the user's files from a tree (nodesTree) and displays them, whether they are files from the file explorer or files resulting from a filter.

So Nodelist has all the methods to make requests to the backend (to ask for the contents of a folder or to ask for the results of a search), and the Filters component is just a form for choosing filters, which passes the user's selection to Nodelist in an indirect way (Filters sends an event to App, which *binds* the variable storing this event to the Nodelist component)

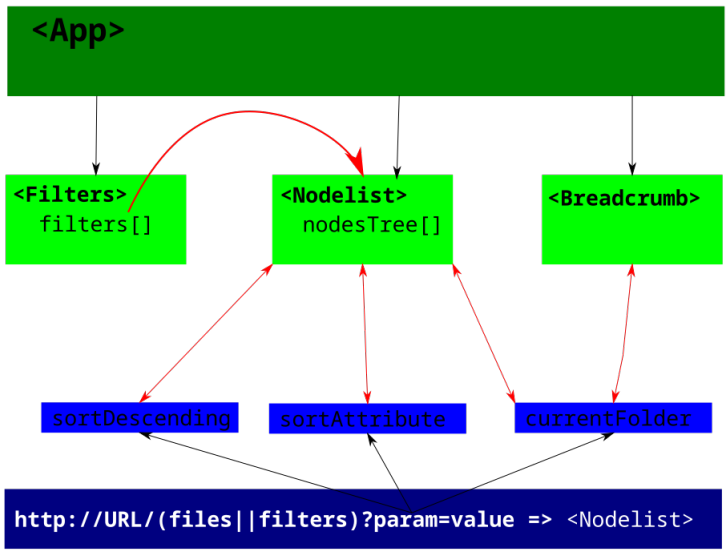



Figure8 - Schematic of the Sorts composites before refactoring

This structure is unconventional (two components from the same parent should not have to exchange information), and very unmaintainable. A refactoring was necessary to start again on good bases and finish the development more easily. So I modified the components to reach the or- ganization shown in figure 9.

Nodelist is here stripped of all logic, it is only a presentation element to which we v-bind a nodesTree file tree, which it takes care of flattening and displaying (see section 3.4.2), and which emits events when a folder is clicked or double-clicked (unfolded or entered into the folder). The retrieval of the nodesTree file tree and its modification is left to the components instantiating Nodelist.

The Filters component now contains methods for querying the filtered files in addition to presenting a user interface for selecting these filters. A new FileExplorer component has been introduced to retrieve from the backend and modify the file tree in a file exploration context.

The App component also determines the state of the application depending on whether the URL begins with "files" or "filters". FileExplorer and Breadcrumb are loaded by App if and only if the URL starts with "files". App also passes the information to Filters via a *v-bind* directive so that it can display its Nodelist file list or not. Thus, the URL results in two different views: a "file explorer" view, figure and6, a "filters" view, figure 7.

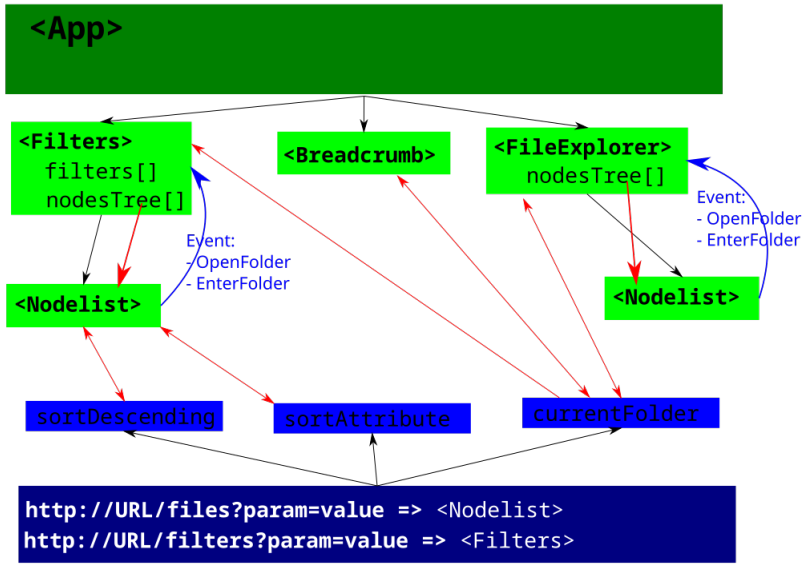



Figure9 - Schematic of the "Spells" components after refactoring

## 3.5 Other tasks within Framasoft

During my internship, I was in charge of other tasks than the main subject mentioned above.

I worked on the "Yakforms" software, a software for creating forms created by Framasoft and whose development we wish to be taken over by the community. A lot of work has been done in this direction by Théophile Le- marié, former student of the Telecommunication department of INSA of Lyon, during his end of studies internship in 2006. I2020. had the task to test the documentation written by Théophile by installing the software and by reporting errors and difficulties not covered by the documentation of the installation process.

I also participated regularly from October to November in the technical support of the Framasoft service based on this software. This experience allowed me to see another facet of the work of Framasoft employees and to get familiar with the infrastructures hosting Framasoft services and their functioning.

It was also planned that I would try to develop a prototype of a JavaScript "tutorial" library to be integrated into the Nextcloud front-end. The idea behind this task was to judge the technical feasibility of creating a "guided tour" of the Nextcloud web interface, which was one of the needs identified during the survey of activist structures.

This topic was planned as a side development in case the development of "Sorts" was faster than we estimated. As the development time of "Sorts" was consequent, I finally did not tackle this task during my internship.

# 4 Digital literacy and the impact of alternative organizations

The challenge of the "Framacloud" project, in which my internship is taking place, is to digitally equip the actors of social progress and social justice [1] to participate in their empowerment and increase their impact on society.

This objective stems from several intuitions. Firstly, that the mastery of digital tools is an important factor for the impact of a group on society and secondly, that the structures of social progress and social justice are disadvantaged in their use of and access to the digital. Furthermore, is our choice to respond to this issue with actions around digital collaboration justified? Are these intuitions well-founded and what place does digital collaboration have in the digital empowerment of the targeted actors?

## 4.1 What impact on society can we have by mastering digital technology?

In the introduction to Digital Culture, Dominique CARDON compares the profound rupture brought about by the advent of digital technology with the invention of the im- primer in the 15th century and the cultural revolution that followed [7]. It goes without saying that today digital technology has taken a central place in society and daily life: teleworking, digital entertainment, online procedures, online purchases,
...

What political impact can digital literacy have on an increasingly digitalized society?

A first angle of observation is that of the liberation of popular expression via the Internet which, according to Dominique CARDON [8] and Dominique BOULLIER [9], has allowed the appearance of a new form of democracy.

This new "internet democracy" is the result of the media weight of social networks, which allow everyone to produce and comment on opinions, news or demands on an equal footing. These networks, even though they are used by traditional newspapers and political figures, sometimes give greater weight to individuals without any prior "public notoriety capital" (whether this capital is militant, political or media).

This new form of democracy is taking shape via large-scale popular social movements such as the Arab Spring or the Yellow Vests. The latter was built via social networks and especially the Twitter platform

1. By the term "actors", we wish to encompass a wide range of organizations: associations, trade unions, informal groups, companies, ...

and is quite characteristic of these new movements: refusal of hierarchy (and in particular the refusal to designate spokespersons) and great diversity of opinions within its members [10].

These new movements, for which the mastery of social networks has been a key element, have had important political impacts and are an example of the political impact of digital mastery.

However, a second observation mitigates these positive effects of digital technology for social change. According to the book "The revolution that wasn't" by the political scientist Jen Schradi, quoted by Le Monde in the article "Is the Internet right-wing?" [11], right-wing and extreme right-wing militant groups have a greater presence on the Internet and are also the most hierarchical, which allows them to have greater visibility and to disseminate their ideas more quickly and strongly.

The use of digital technology is therefore an important lever for activism and can lead to large-scale social movements, but the internet is a scene dominated by conservative themes. Are the structures of social progress and social justice disadvantaged in their access to the digital?

## 4.2 Do alternative structures have access to digital collaboration?

On what aspects could the structures of social justice and social progress have difficulties in accessing digital technology?

Guillaume GARCZYNSKI, in his article "Fracture numérique, Fracture sociale", highlights the fact that many categories of disadvantaged people are kept away from the digital world. These barriers to inclusion can be financial (people receiving minimum social benefits, undocumented migrants, etc.), or linked to digital illiteracy (elderly people, rural people, young people with only recreational use of the digital world, etc.). This digital divide is the consequence of a social divide and causes a social divide. It mainly affects audiences targeted by social change struc- tures or likely to be active in these structures.[12]

The social movements mentioned in the section above also undermine the collective dimen- sion of the classic structures of social struggles (unions, associations, parties, etc.) in favor of an individualization of positions. This individua- lization can harm social struggles by fragmenting voices and claims [13] [10].

Wouldn't it be missing a mastery of digital collaboration to allow social struggles to recreate the collective and the convergence in a context of digitalization of society?

In any case, this is the niche that the Framacloud project wishes to address by using the Nextcloud software as a pretext to create collaboration between the structures of social progress and social justice.

In addition to individual digital challenges, the problem with collaborative digital is that it requires more infrastructure and organization than just access to social networks. It is necessary to choose the services through which to collaborate, for activists to learn how to use them, for their use to become habitual and for collaboration processes to be thought out and put in place according to the needs of each structure. These prerequisites explain why, among French associations, only 39% use digital technology to better collaborate, the majority of which are associations that declare themselves comfortable with digital technology [3].

Moreover, the most easily accessible collaboration tools today are the large, centralized solutions of corporate surveillance capitalism, such as the "Google Suite. These tools raise many concerns about their use by an activist audience, which describes the concern that personal data is not sufficiently protected as the main obstacle to associative use of digital collaboration [12].

There is thus an urgent need, if we take the point of view of progressive rather than conservative forces, to accompany activist structures towards a greater mastery of digital collaboration processes.

## 4.3 The limits of the digitalization of struggles

In the race for digital activism, there is an urgency to equip activist structures for social change. But how important is this digital scene compared to the non-digital world?

Activists for social progress and social justice would be more focused on concrete field actions than on communication, especially digital communication [11].

For example, support and assistance to disadvantaged populations cannot be replaced by digital interactions. Digital technology cannot, in this context, replace physical action and is limited to raising public awareness of these issues or coordinating activists [13].

Grassroots activism is also a catalyst for engagement. It is often at the origin of the long-term commitment of many individuals in the world of activism [13]. But also, active online activists would be those activists who are already aware of and involved in face-to-face activist structures [7].

Digital technology has changed activism, for which it is a new door

or a new tool, but it does not replace physical activism.

In light of these elements, it seems possible to establish a strong correlation between the mastery of digital uses (especially collaborative) on the one hand, and the social and political impact of structures on the other. However, it also appears that the mastery of these tools and techniques would be a necessary but not sufficient condition for the structures working for progress and social justice to see their ideas and actions prevail over those of more conservative movements.

# 5 Job : *Full-Stack* web development on an Open-Source application

Framasoft being a small associative structure of ten employees, forming a single team where everyone works on different projects, the positions and team management processes are a bit different from what is done in larger entrepreneurial structures.

For example, the job of my internship tutor is that of co-director of Frama Soft. It's a very transversal job, mixing team coordination, definition of development and communication strategies for Framasoft and representation of the association to the media and partners.

Nevertheless, one can draw some similarities between the role I played during this internship and more "classic" positions in the working world.

The technical part of my internship could be close to the work of a *Full-Stack* web developer, with the addition of a "project management" aspect that was left to me. This brings my job closer to the technical project management positions, modulo the parts related to team management.

Nevertheless, the field in which there are the most similarities to be drawn is that of Open-Source. Developing software with collaborators from different backgrounds, amateurs or professionals, from different companies and nationalities, working almost exclusively remotely, is one of the strong characteristics of this movement.

During this internship, I was able to evolve in such a context by participating in the development of Nextcloud. I also created and managed an open-source project with the *plugin* "Sorts", which led me to exchange with developers of the German company Nextcloud GmbH on several occasions, but also to learn how to manage software contributions.

In addition to my librist beliefs, I really liked the working environment of open source development and I want to continue my career in this environment. I am currently considering applying for backend development positions, especially at Nextcloud GmbH. This type of position would be a good match for my interest in backend development, technical issues, and the *processes* involved in Open-Source development.

# 6 Feedback: an internship at Framasoft

Technically I deepened the web development skills I had acquired during my training with the handling of PHP and VueJS which are among the major technologies of web development. I also learned a lot about reading and understanding the source code of a "real" professional software. By this I mean that the magnitude of Nextcloud's source code (about millions6 of lines of code) is much larger than anything we cover during our training and understanding how a software spreads over several hundreds of files requires understanding the mechanisms of "articulation" of the different parts of the source code.

The development of "Sorts" was also an opportunity to do some project management. In particular, I had to go over my planning several times, the breakdown of my development into functionalities and I had to go through several refactoring stages. This trial and error is both due to the fact that I was learning the technologies I was using as I went along, but also because I made several estimation errors that I would know how to avoid today.

This internship also allowed me to review my approach to telecommuting and I got a progression on my ability to compartmentalize my attention and stay focused on my work from home. First, I was working on Open-Source projects with developers external to Framasoft or telecommuting colleagues (more than half of the Framasoft team is not in Lyon). Secondly, I worked on Mondays and Fridays from home and the rest of the time from Framasoft's offices at Locaux Motiv', which is a shared workspace for social economy structures.

This hybrid arrangement has proved to be a real advantage for my motivation and efficiency. My presence at Locaux Motiv' has allowed me to enjoy a work atmosphere rich in human interaction as well as a well-separated work/life environment, and my remote work days have allowed me greater flexibility in my hours and to save on my commuting time. I would like to be able to find this way of working in my future jobs.

Finally, this internship was also a first experience of collaboration with various actors in an international context, whether through interactions on GitHub when I contributed to the development of Nextcloud or during the video conference meeting, in English, with German engineers of Nextcloud GmbH.

To conclude, I feel today able to enter the world of salaried work, to invest myself in long term projects and to approach software development at a professional level.

# 7 Conclusion

As my internship took place at the beginning of the "Framacloud" project, there is still a lot of work, exploration and projects to implement to reach the main objective: to make Nextcloud a lever of collaboration between structures militating for progress and social justice.

As for the "Sorts" *plugin, the* objective of the next weeks will be to publish a first version. It will be a question of correcting its last errors, of making it as accessible as possible by "smoothing" the rather austere user interface and of making it public via a press release on the Framasoft website and a publication of the *plugin* on the official Nextcloud *plugins* website.

Thereafter, the Framasoft salaried team being voluntarily limited to 10 people and already working on several dozens of projects in parallel, it is not planned to add salaried development time to "Sorts". A follow-up of the *project* will nevertheless be carried out, with two objectives: to keep the current *plugin* functional and to collect feedback in order to refine our understanding of the needs of the users targeted by the "Framacloud" project. Future developments will be left to the community or to possible future internship subjects. I also wish to remain a volunteer member of the association and *maintain* the *plugin* in my free time. I will be able to follow the patches and potential developments.

As for the rest of the Framacloud project, "The road is long, but the way is clear [1]. There are many ways to reach the project's objective: it will be a matter of continuing to improve the technical solution through software developments, of communicating about the project, of accompanying the targeted structures and of proposing a Nextcloud offer to any militant association that expresses the need for it [4].

In order to carry out these missions, we will need to understand the needs and expectations of our target users. This is a work that has been started with my internship and the future of the project will be based on my amazement report and the results of the survey, but also on new resources such as the collection of user feedback on the deployment of the "Sorts" *plugin* and the increase of the Nextcloud offer. It will also be about collaborating with Nextcloud GmbH to understand their vision of the future of the software and where we can improve its associative use.

The Framacloud project, with its current scope, should last for years3 [4] and reach completion in 2024/2025. The project is still in its infancy and its objectives can therefore be further refined as progress is made and issues are discovered in the months to come.

1. Framasoft's unofficial motto

# References


[1] Framasoft. *Statutes of the Framasoft association*. url2021. : https://soutenir. framasoft.org/sites/default/files/statuts-Framasoft-2021.pdf.

[2] Framasoft. *Activity report 2020*. url2021. : https : / / support . framasoft.org/sites/default/files/Framasoft_rapport_activite_ 2020-1.0.pdf.

[3] Solidatech and ResearchSolidarités. *The place of digital in the associative project in 2019*. 2019. url: https://recherches-solidarites. org/wp-content/uploads/2019/12/Num%C3%A9rique-Report-2019.pdf.

[4] Framasoft. *Framacloud pre-project: Software to the people!* 2021.

[5] various. *Nextcloud developer documentation*. Nextcloud23. GmbH version. url2022.: https://docs.nextcloud.com/server/23/developer_manual.

[6] various. *Nextcloud PHP API*. latest version. Nextcloud GmbH. url2022. : https://nextcloud-server.netlify.app/namespaces/ocp.html.

[7] Dominique CARDON. *Culture Numérique*. Paris: Presses de Sciences Po, 2016.

[8] Dominique CARDON. "The digital public space". In: *Culture Numé- rique*. Paris : presses de Sciences Po, 2016.

[9] Dominique BOULLIER. "Political Sociology of the Digital". In: *Sociologie du numérique*. Armand Colin, isbn2016.: 978-2-200-29165-5.

[10] Natacha Souillard et al. "Les Gilets jaunes, étude d'un mouvement social au prisme de ses arènes médiatiques." In *Terminal, Information Technology, Culture and Society (*1272020). doi: 10.4000/terminal.5631.

[11] InternetActu. *Is the Internet right-wing?* 2019. url: https://www.lemonde. en/blog/internetactu/2019/05/08/internet-is-it-right-wing/.

[12] Guillaume Garczynski. "Digital divide, social divide." In: *Project Review* No. 371, Is the Internet Reinventing Activism (2019). doi: 10.3917/ pro.371.0033.

[13] Anaïs Theviot, "Le militantisme, cinquante ans après Mai 68". In: *Project Review* No. 371, Does the Internet Reinvent Activism (2019). doi: 10.3917/ pro.371.0037.


# Glossary

**fork** Source code base created by a developer from the official source code (upstream) in order to develop a feature (to be proposed to the upstream for integration) OR New software created on the basis of the source code of a pre-existing software. 8,9,13, 15

**merge request** Request to integrate a new piece of code to the source code of a software. 12,13, 19

**Mobilizon** Free alternative to Facebook events. 5, 6

**Nextcloud GmbH** German limited liability company (GmbH) responsible for the development of the Nextcloud software. 8, 11, 12, 15, 16, 18, 25, 34, 35

**PeerTube** Free client-server and peer-to-peer video distribution software. 5, 6

**upstream** The official source code of the software OR a software from which a second software is derived (fork). 13, 15

# Acronyms

**CHATONS** Collectif des Hébergeurs Alternatifs, Transparents, Ouverts, Neutres et Solidaires. 5

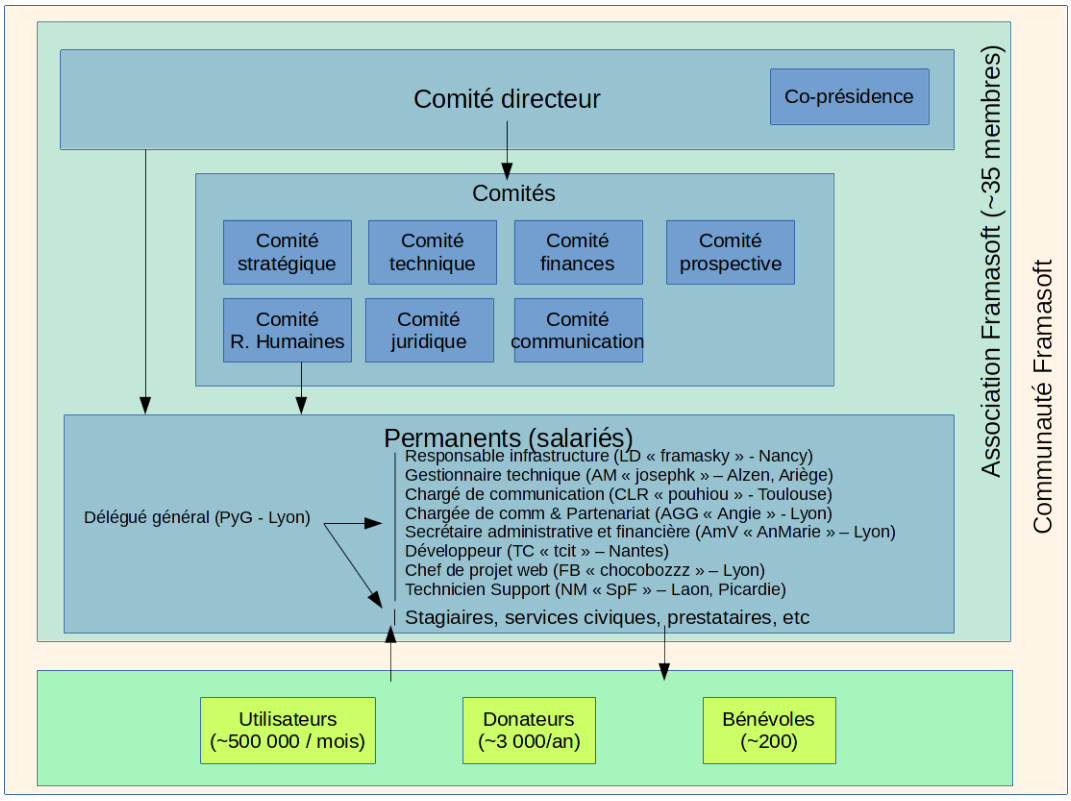
**GAFAM** Google, Amazon, Facebook, Apple and Microsoft. 4

**LAMP** Linux, Apache, MySql, PHP (classic web server software stack). 9

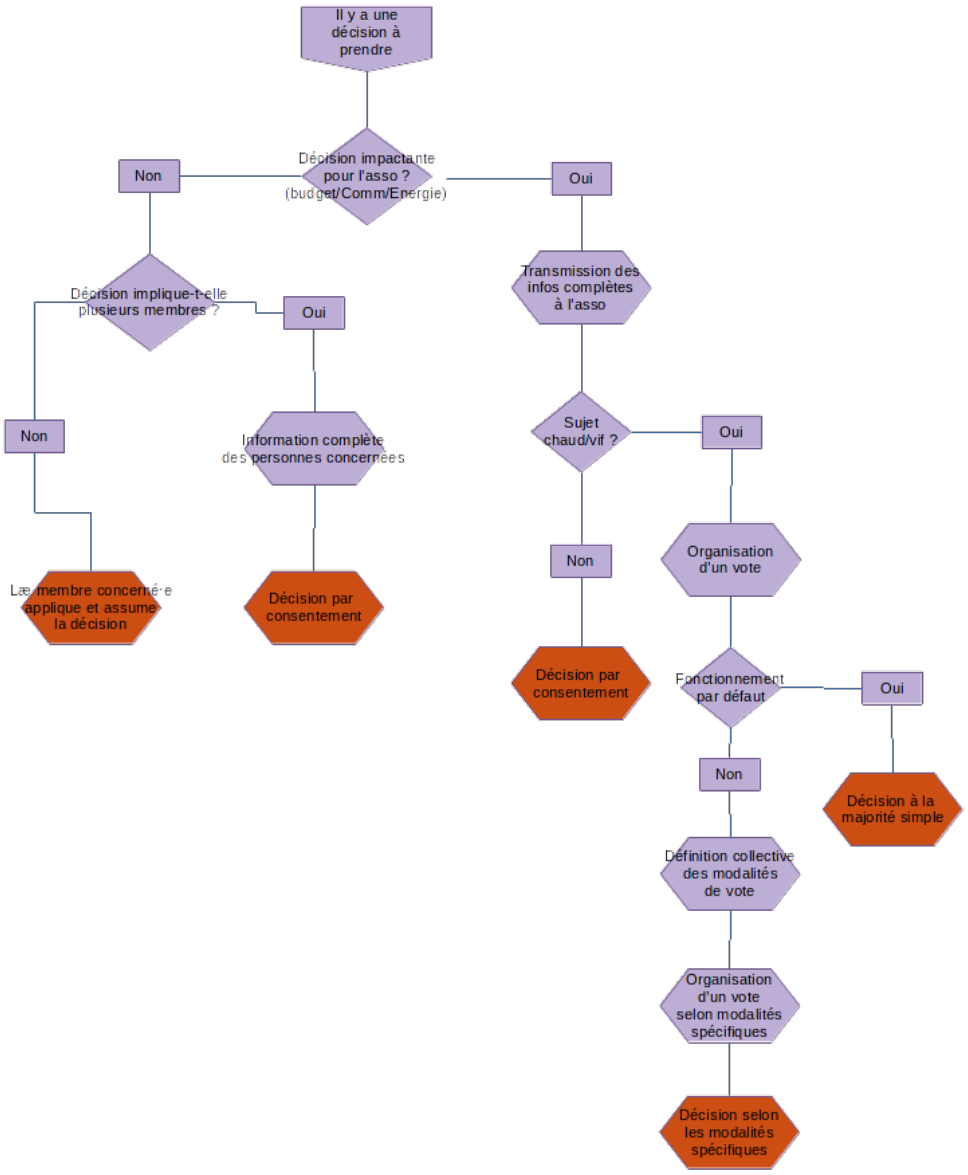
**SWOT** Strengths, Weaknesses, Opportunities, Threats. 3, 7

# Annexes

## A Organization chart of the association



# B Decision-making procedures at Framasoft



# C Gantt of the course