POLITECNICO DI TORINO

College of Engineering and Management

Master of Science in Engineering and Management

Master of Science Thesis

Development of an application in a "web page application" environment for controlling the accreditation process of the Calibration Laboratories Department, Accredia

Academic Supervisor:
Domenico Augusto Maisano

Company Tutor:
Paola Pedone

Candidate:
Mohsen Ebadpour

July 2021

Table of Contents

Table of Figures

1 Introduction

The upcoming thesis is authorized by Accredia calibration laboratories department to analyze and enhance the management of the accreditation procedures and processes performed by this department. In sections 1.1 to 1.4, the accreditation procedures and processes in the context and three of the most recognized global accreditation communities are briefly studied.

In section 1.5, the accreditation industry is analyzed in wider terms to accompany all of the entities who are involved in the accreditation process. The benefits arising from the accreditation procedures and processes, and in particular why accreditation is very important in facilitating global trade are discussed in sections 1.6 and 1.6.1.

Accredia is briefly presented in section 1.7, and section 1.7.1 briefly describes the thesis objectives and the relative working environment.

In section 1.8 how a part of the accreditation process is performed by Accredia, calibration laboratories department is narrated, somewhat detailed. By reading this section, one can start to imagine what kind of software would help to manage the process.

Proceeding to 1.9, here's the thesis's suggestion (web application) towards Accredia's approach for building a new ecosystem for the information flow of the accreditation process.

Section 1.10 refers to the project management methodologies used throughout developing this project.

Section 1.11 discusses prospective cloud computing integration benefits.

Section 1.12 describes how users can interact with the tools at their disposal.

Section 1.13 focuses on how the web app works with Nextcloud, the cloud service chosen by Accredia.

1.1 What is Accreditation?

Accreditation is a quality assurance process in which an authorized accreditation body formally assesses a Conformity Assessment Body (CAB) against recognized international standards and guides or other requirements under the accreditation system. Accreditation demonstrates the competence and impartiality of a CAB to carry out specific conformity assessment tasks such as testing, calibration, inspecting, and issuing certifications.

The accreditation bodies are approved and recognized by the International Laboratory Accreditation Cooperation (ILAC) and/or International Accreditation Forum (IAF).

1.2 The International Accreditation Forum (IAF)

The International Accreditation Forum (IAF) [1] is the global association of Accreditation Bodies, Certification Body Associations, and other organizations involved in conformity assessment activities in various fields, including management systems, products, services, and personnel.

IAF objectives are:

- Promote the worldwide acceptance of certificates of conformity issued by certification bodies accredited by an Accreditation Body Member.
- Maintain and develop a Multilateral Recognition Arrangement (MLA) between its Accreditation Body Members to ensure recognition of accredited certification between signatories.
- Bring together accreditation bodies and stakeholder groups to facilitate global trade.

1.3 International Laboratory Accreditation Cooperation (ILAC)

International Laboratory Accreditation Cooperation (ILAC) [2] is the international organization for accreditation bodies operating in accordance with ISO/IEC 17011 and involved in the accreditation of conformity assessment bodies including calibration laboratories (using ISO/IEC 17025), testing laboratories (using ISO/IEC 17025), medical testing laboratories (using ISO 15189), inspection bodies (using ISO/IEC 17020), proficiency testing providers (using ISO/IEC 17043) and reference material producers (using ISO 17034).

ILAC promotes the increased use and acceptance by industry as well as government of the results from accredited laboratories and inspection bodies, including results from accredited organizations in other countries.

The international arrangements in the fields of calibration, testing, medical testing, inspection, proficiency testing providers, and reference material producers accreditation are managed by ILAC, and IAF occupies in the fields of management systems, products, services, personnel, and other similar programs of conformity assessment.

Both organizations, ILAC and IAF, work together to enhance the accreditation and the conformity assessment worldwide.

1.4 European Accreditation (EA)

European Accreditation (EA) [3] is a non-profit European co-operation for Accreditation, it is formally appointed by the European Commission to develop and maintain a multilateral agreement of mutual recognition of the European Accreditation Multilateral Agreement (EA MLA), based on a harmonized accreditation infrastructure.

EA has 50 Members, the EA Members are National Accreditation Bodies (NAB) that are officially recognized by their national governments to assess and verify organizations that carry out conformity assessment activities such as certification, verification, inspection, testing, and calibration against international standards.

1.5 Stakeholders of the accreditation process

This covers a vast range of actors, first, there are global accreditation associations and cooperation such as ILAC, IAF, and EA along with all of the accreditation bodies recognized by ILAC and IAF.

Secondly, there are the parties who request to be accredited by the aforementioned recognized bodies, these parties are known as the Conformity Assessment Bodies (CABs). This includes but not limits to calibration laboratories, calibration centers, medical laboratories, and material producers, testing laboratories, and others. The accredited CABs experience a competitiveness improvement in the market, this affects the market structure as well as not accredited CABs.

Then there are the consumers, institutions, and businesses who trust the accredited goods or services such as using a conformity assessment service issued by an accredited CAB, and finally, the public administrations since there are many national accreditation bodies present in the world due to the necessity of obligations to put a good or service on the market to ensure the quality and reliability of the good or service.

Shortlist of stakeholders:

- Accreditation bodies and communities
- CABs: Accredited or not accredited
- Consumers, institutions, and businesses
- Public Administrations

1.6 Accreditation benefits

The stakeholders enjoy an increase of added value caused by the network effect present in the international conformity assessment communities. The economic advantage brought in by these standards and mutual recognitions can simply be the importance of these institutions in facilitating global trade and guaranteeing higher quality and safety level of goods and services.

Being granted an accreditation aligned with international standards helps CABs grow their reputation and performance, resulting in more competitiveness and international recognition for the accredited bodies.

1.6.1 Global trade

International trade is the exchange of capital, goods, and services across international borders or territories. We all enjoy and rely on a vast number and range of products and services supplied from overseas. It would be impossible to imagine a world in which our choice of goods and services was limited to those produced within the country in which we live.

Every year sees an increase in global trade figures running into many trillions of dollars. International trade represents a large share of the gross domestic product of most countries. National and international voluntary and mandatory technical regulations, standards, testing, inspection, and certification procedures support the continued movement of capital, goods, and services between countries are therefore of huge importance for individuals' health and well-being as well as to the economic health of entire nations around the globe. Products and services can cross borders to meet global demand without causing undue risk to the health and security of individuals or the environment.

Generally, these are introduced to meet the legitimate requirements of quality and safety that consumers, businesses, regulators, and other organizations demand in the case of goods and services, whatever their country of origin.

1.7 Accreditation in Italy

Accredia [4] is the national Italian accreditation body appointed by the Italian government. Accredia operates on a non-profit basis and complies with the application of the European Regulation 765/2008.

Accredia was founded in 2009 by combining the competencies of Sit, Sinal, and Sincert and it is organized in three departments in Milan, Rome, and Turin.

Up to the day of this research, over 2000 bodies are accredited by more than 500 assessors and experts at Accredia.

1.7.1 Accredia, calibration laboratories department

The upcoming work is related to the calibration laboratories department of Accredia, located in Turin. Its main objective is to develop an IT solution to control the accreditation process in terms of its steps, duration of the steps, and await decision-making points in some steps to make it more efficient and to reduce human-related error risk

The underdevelopment solution is inspired to foster the management of the process by providing a user-friendly interface and transparent record-keeping capability, and sending reminders and warnings in defined cases automatically.

Less than 10 people are working in the calibration laboratories department, in detail, there are two STDs (Departmental Technical Secretariat), one DDT (Director of the Department of Calibration Laboratories), and six FTs (Technical Officer), and one professional engineer (in accreditation and databases).

1.8 Use cases and personas analysis

The list below has gathered the actors who are involved in the accreditation process performed by the Calibration Laboratories Department. The accreditation process in this section is not studied in full scale, instead, it is a demonstration of the type of company routines that can change for good by using new software

- STD: Departmental Technical Secretariat
- FT: Technical Officer
- DDT: Director of the Department of Calibration Laboratories
- CAB: Conformity Assessment Body

Other acronyms that are used and refer to the steps of the accreditation process:

- ED: Esame Documentale aka documental assessment
- RT: Technical report
- ILC: Inter-Laboratory Comparison
- DA: Application for Accreditation

The hereafter process refers to the process of a CAB applying to be accredited by Accredia. STD responds to CAB's DA. If DA is accepted, STD assigns an FT to the process who is in charge of managing the process. It is needed to say that in case of negligible errors in compiling DA, acceptance with reservation can be granted to DA. In such a case, the applicant is obliged to provide the missing or miss compiled documents within a definite timeframe.

After acceptance of the DA (or acceptance with reserve), FT proceeds with preparing the initial and internal price quotation (scheda preventivo), which is an internal document and is validated by DDT after its creation by FT.

The price quotation which is deliverable to CAB (preventivo) is created by STD and is based upon the internal price quotation that was created and validated respectively by FT and DDT.

CAB signs the preventivo, STD confirms the legitimacy of the document, and if all is well, the ED phase begins. STD goes on with creating and delivering assignments to inspectors/assessors, they report their evaluation to FT by filling MD-08-01-DT form (technical procedures for ED).

Furthermore, some of the inspectors might be appointed for issuing technical reports (RT) which is a document that contains the evaluation of the participation and results of the ILC to which the CAB has participated. The latter is a part of the documental assessment (ED).

Based on the assessors' feedback, FT concludes an outcome and foresees a series of actions to be taken, depending on the outcome of ED.

In case of a negative outcome for the evaluation of ILC, FT requests CAB for corrective actions. Corrective actions are also requested if the outcome reported in the MD-08-01-DT is negative. For this latter, three loops of evaluation are allowed to reach satisfactory results

1.9 The feasibility study and Web page application

After a few brainstorming sessions with engineer Pedone, who is an FT and also the company tutor supervising this project, on the characteristics of the conceptual software, choosing the web page app was an easy decision. The alternative options could be programming a plug-in app for a cloud service or programming an application for an email service.

A cloud plug-in is too dependent on the cloud itself, for example, an update of the cloud service can block the process. It's worth mentioning that this solution and the web app are compatible and can be integrated to work in one ecosystem.

Writing an app for an email service would lack a user interface (not user-friendly) and it is costly to develop. It's hard or even impossible to connect such an app with cloud services.

On the other hand, a web app is the easiest form of application to access. It is widely spread. Thanks to its many reliable open-source development tools and communities, there are much fewer additional costs. In particular, two of the main assets used throughout the development are Laravel and Nextcloud which are both open-source tools and are better explained in the next chapters.

1.10 Agile project management and Scrum

This project started with frequent meetings among engineer Paola Pedone and me to transfer the knowledge of the accreditation process and compile the characteristics of useful software in the context. Soon web page app appeared as an appealing approach, then the idea was presented to the IT staff of Accredia, and a green light was granted, in particular by Nello Palomba who is in charge of the network administration and data security of Accredia.

A crucial contribution, in particular, for the design and development of the web app came from engineer Amir Ghaffari, whom I know for a long time. He joined the team from the early stages. Diego Orgiazzi, the aforementioned professional engineer, joined the team and started to participate in the group meetings a couple of months after the project's initialization. His presence was very decisive in aligning the project with Accredia's demand.

Scrum product development methodology helps to correctly implement the desired features and minimize the misalignment in developing complex software. [5]

Time boxing is one of the main principles of Scrum framework and agile management in general, which was used to run a few sprints in developing this project. The needs and requirements of the new software were often studied in group meetings, moreover, the ideas for the new solution were discussed in these group meetings. After reaching a mutual understanding, the design and development team consisting of Amir Ghaffari and me, would start a new sprint and commit to delivering as much as possible within a pre-defined time frame, this is called time boxing. [6]

The achievements of each sprint were reported back to Accredia's referents (engineers Pedone and Orgiazzi) to seek feedback and perform sprint retrospective meetings.

Furthermore, occasional meetings with the IT staff of Accredia were conducted. These meetings often were informative, guidance seeking, and alignment sessions.

1.11 Data storage and cloud computing

Accredia had already noticed the criticalities that are better explained in the next chapter, and was considering to use a cloud service named Nextcloud as a tool to store and share the accreditation files online even before start of this project.

Storing the accreditation files on Nextcloud enables the users to benefit from cloud computing advantages, for instance, easy accessibility and the opportunity to collaborate on a document in real-time. Focusing more on the online collaboration feature, it is very useful for the on-site assessment (VsC) phase of the accreditation process, which is queued after ED. It is worth mentioning here that the VsC phase of the accreditation process is not covered in this thesis.

Every assessor who is appointed to perform VsC delivers the part he/she is responsible for, by filling the same piece of document with other assessors.

This project is developed to work with Nextcloud, moreover, the development team seized the opportunity to ensure the confidentiality of the files by storing the files nowhere other than on predefined super admin accounts on Nextcloud, and then share the files with specific users whenever it is needed.

1.12 Communications with the users in the prospective software

A web app provides a user-friendly interface to get information regarding decisions taken during the process. This is the view component of the MVC (Model, View, and Controller) software development architecture pattern which communicates with users and updates them on the process's progress. FT, STD, and DDT are the direct users of this view component.

Even though CABs and Inspectors are not direct users of the web app, they are still considered in the software development as reminder emails are going to be sent to CABs and inspectors where necessary. Moreover, CABs and assessors/inspectors are responsible for procuring some documents throughout the process, hence they will be granted access to the Nextcloud drive of Accredia to make modifications. Events on Nextcloud are monitored by a freshly developed file-watcher software, and then the data are fed back to the back-end of the web app where these data are parsed to figure out if any user fulfilled her/his task by making an expected modification on Nextcloud in order to advance the process to the next step.

1.13 Object-oriented programming paradigm

The web app developed in this project will track the accreditation files and their attributes. With Nextcloud acting as the host, there's the requirement to be able to sense the events happening on Nextcloud, to this end, in this project the File-watcher application which is a plugin for Nextcloud and is an API (application programming interface) to adapt Nextcloud's events to the web app is developed from scratch.

It is worth mentioning that by using the Webdav API of Nextcloud [7] users of the web app were already able to access Nextcloud using the web app, and upload or download files, however, this was only one-way communication with Nextcloud.

The aforementioned File-watcher program enables the underdevelopment software to keep track of the process even when a modification is done directly from Nextcloud, without using the web app. This is the anticipated use case scenario for inspectors and CABs. Moreover, the direct users of the web app can upload a file using the web app or Nextcloud, and the results will be the same.

2 The Criticalities in the current setting of information flow

This chapter starts with clarifying the intention of the applications developed and their scope according to the full-scale accreditation cycle performed by Accredia, calibration laboratories department. (Sections 2.1 to 2.3)

Sections 2.4 to 2.6 reason why new software (Nextcloud, Web app) would be beneficial to the management of the process. In section 2.7 the accreditation files within the project’s scope are categorized on Nextcloud. Section 2.8 reports the categorization of the prospective users.

2.1 Management of the accreditation process

The process had been mainly managed by FT and it is going to stay the same way after the possible adoption of this work by Accredia as this project does not intend to change the existing formalities in the accreditation processes and cycles performed by the calibration laboratories department.

The applications developed in this project are tools to facilitate the management of the process by substituting the means and manners of how the accreditation process has been followed prior. This covers a very wide range of activities, these activities were limited to the project’s scope.

2.2 Project scope

The description of the accreditation procedure in the use case and personas section in the first chapter doesn’t present the full-scale accreditation process and yet it is getting clear that implementing a software capable of managing the whole process demands well over the effort put into one thesis. This work suggests utilizing a web page application to address the need for new software. Furthermore, an initial version of the software which covers preliminary activities that have to be carried out for ED phase to begin has been developed. The CAB’s request for accreditation has been skipped therefore the starting point of the part of the process that is implemented is the creation of a new accreditation process by STD to respond to CAB’s request for accreditation. It is the intention of this project to consider implementing all of the functionalities essential for implementing the full-scale accreditation cycle.

2.3 The full-scale accreditation cycle

The full-scale accreditation process lasts four years. The AOA (activities on arrows) Figure 1 Full-scale accreditation cycle summarizes the full-scale accreditation cycle performed by Accredia calibration laboratories department. The main activities in the first set (arrow) of activities in Figure 1 (Accreditation/renewal) are presented in more detail in Figure 2.

In the case of a CAB applying to be accredited for the first time, the Accreditation process takes point A to B Figure 1 Full-scale accreditation cycleif the accreditation is granted at point B, two surveillance phases will take place. The first surveillance takes place within 12 months after when accreditation is granted, and it follows the same steps as the VsC phase. 18 months after the first surveillance, the second one is carried out. CAB can apply to renew the accreditation within a definite time before the expiration date of the accreditation (18 months after the second surveillance).

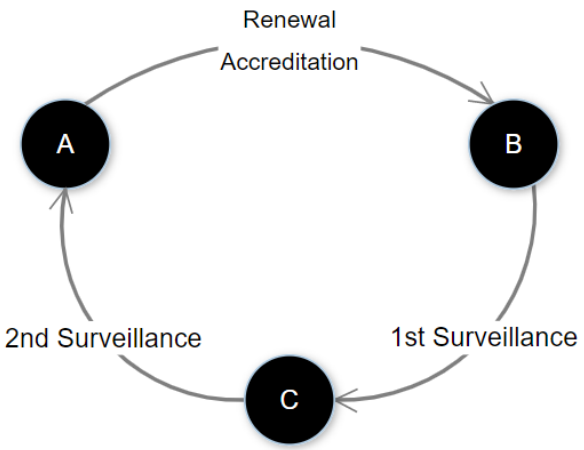


Figure 1 Full-scale accreditation cycle

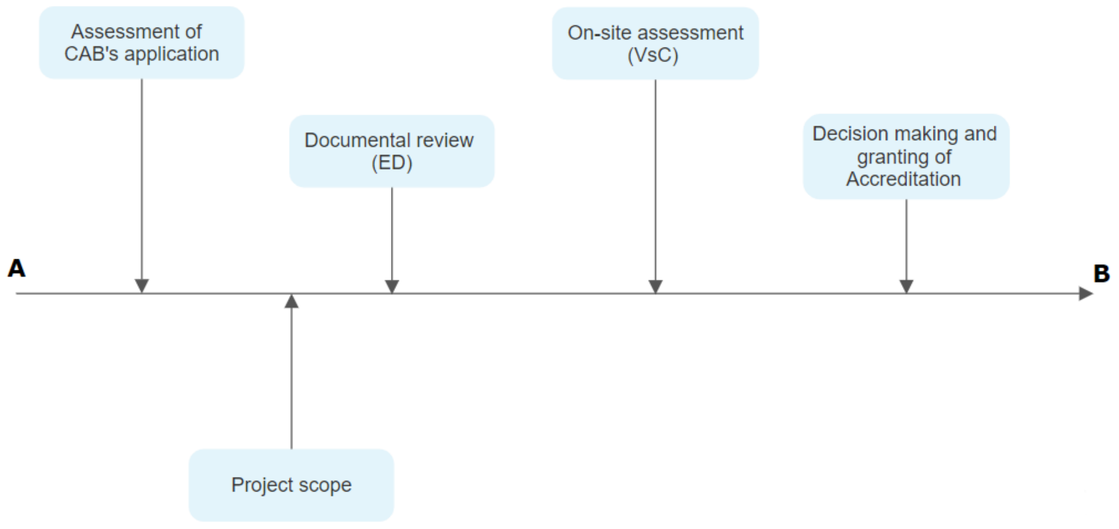

Assessment of CAB's application
Documental review (ED)
On-site assessment (VsC)
Decision making and granting of Accreditation
A
B
Project scope

Figure 2 Accreditation, Renewal

The activities presented in Figure 2 are the same set of activities that are carried out during accreditation and the renewal of accreditation processes. It's worth mentioning that three loops of documental review (ED) are allowed to reach satisfactory results in the accreditation process, however, there's a single possible trial to pass ED in the renewal process.

As it is indicated by the project scope box in Figure 2, the first set of activities (Assessment of CAB's application) implemented in functional conditions.

2.4 The need for a new set of software

The accreditation cycle studied so far has been lacking a unified and tailored software to digitize the paperwork and information flow. Unified in the sense that the conceptual software would embed all phases of the accreditation process all in once or be compatible with the other software and tools used or developed by Accredia. The underdevelopment software is aimed to be tailored to facilitate the process's management, possessing characteristics like an automated reminding system and real-time record-keeping of process progress.

2.5 The added value by the new software in the management of the process

Accredia is moving toward cloud computing and the project on hand is a complementary asset to Nextcloud which was chosen to be used by Accredia. One of the main objectives of this thesis is to build software capable of sending notifications, to-do messages, and reminders based on the process status. For instance, after FT uploads the Scheda Preventivo that is to be approved by

DDT, a message should be reached to DDT to approve the document and after DDT's approval, another notification should depart towards STD to prepare the Preventivo.

Another good example, which is not implemented in this thesis, is when FT expects the document evaluation from assessors/inspectors or corrective actions from CABs in the ED phase. It's enough to realize that one unseen email can block the process since there's no reminder email planned.

There's a sequential connection between users, meaning that succeeding in one task takes the process forward and any blockage will bring the process to a halt.

The underdevelopment software interacts with users and helps to avoid unwanted blockage that might occur in the current settings of the information flow in the accreditation process.

Accredia uses protocols to index incoming and outgoing document communication with external parties such as CABs. The management system developed in this project indexes all of the documents upon the moment of their creation, as a result, the need for using protocols can be questioned to become gradually less important if the software proves to be robust, however, for the scope of this thesis, STD is notified to take care of protocolling the incoming documents when CABs upload a file on the Nextcloud.

2.6 Nextcloud

Another irritating criticality that is eventually aimed to be solved by Accredia's choice of cloud computing, in particular, Nextcloud is to avoid using email services for communicating files among FT and CABs and inspectors.

FT used to transfer the files attached to the incoming emails to the internal server of Accredia, while Nextcloud is the solution to overcome avoiding this repetitive task. The high quantity of these communications makes the management of the accreditation process very unnecessarily hectic for FT. If FT forgets to respond to a mail or misses an email there will be no reminding system, the risk of committing an error and missing documents is high.

CABs and assessors will be granted credentials to log in to Nextcloud installed on the subdomain of Accredia to download and upload documents. Moreover, CABs and inspectors are informed by emails when a file is ready for them on Nextcloud. They are also notified by automatic emails when the programmed process requires a certain piece of document to be procured by CABs and inspectors.

This is a good example of how Accredia's new approach will save the workforce's time and decrease human-related error commitment risk.

Moreover, hosting accreditation files on Nextcloud brings on document collaboration possibilities among users which will be appreciated in the future phases of the accreditation process.

The web app developed in this project is integrated with Nextcloud and is going to provide a transparent view interface to update users about process progress and send automated to-do messages, notifications, and reminders.

2.7 The folder structure

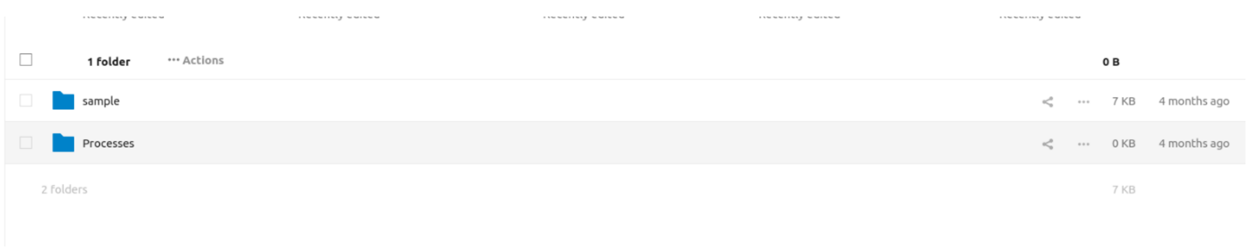
This section doesn't illustrate the structure of how the accreditation files are stored in Accredia, calibration laboratories department. It is a symbolic set of folders to work with the software developed in this project and is limited to the part of the process developed.

Each process's files are going to be saved in one folder named after the title STD enters upon the creation of the process. The folder structure of the folders containing the accreditation files of the different processes is in the same format.

The template of the folder structure with empty folders is saved on the super admin's Nextcloud account. Names of the folder set are reported below:

- 00acceptance_letter
- 01scheda_preventivo
- 02preventivo
- 03assignments_inspectors

Additionally, there's a configuration.xlsx file among the mentioned folders above. There's a folder called "sample" containing the four empty folders mentioned above along with the mentioned configuration file. (See Figure 3 Admin's Nextcloud account)



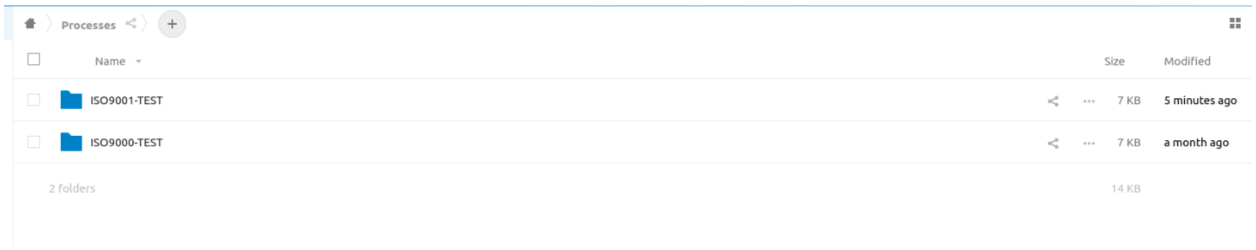
```
1 folder    ··· Actions                                   0 B
sample                                    7 KB   4 months ago
Processes                                 0 KB   4 months ago
2 folders                                 7 KB
```

Figure 3 Admin's Nextcloud account

By starting a new process, the web app copies the aforementioned "sample" folder to another folder on the super admin's Nextcloud account (Processes folder in Figure 3 Admin's Nextcloud account) and renames it according to the process's title. The web app indexes this newly created and renamed folder to the newly created process. Then the web app shares this folder with justified users. The details of these actions are present in chapter 4.

If a user uploads a document for a process using the web app, the web app is programmed to store the document in the related folder (Acceptance letter, Preventivo, etc.) of the folder structure of the related process.

The folder that the sample folder is copied to and renamed inside is called Processes (Figure 4 Processes folder on Admin's Nextcloud account) and contains all of the folders that are indexed to the processes.



```
Processes  +
Name                                      Size   Modified
ISO9001-TEST                              7 KB   5 minutes ago
ISO9000-TEST                              7 KB   a month ago
2 folders                                 14 KB
```

Figure 4 Processes folder on Admin's Nextcloud account

The accreditation process is confidential, thus sharing accreditation files on Nextcloud is a delicate job that the web app is responsible for. The files are categorized based on their users.

DDT and STD are considered to have access to all of the files in all of the processes, while FT can have access to all of the files of the processes that they are assigned to. Not all of the files of the process are shared with inspectors or CABs related to the process, for the scope of the thesis, the web app will share the following files in each process with the relative CAB:

- The acceptance letter uploaded by STD
- The preventivo uploaded by STD

Please note that the web app does not share the Scheda preventivo which is created by FT and later approved by DDT in between the two files that are going to be shared with CAB.

The folder 03_assignments_inspectors is shared with inspectors and the internal staff of Accredia (DDT, STD, and assigned FT).

2.8 Users: direct vs indirect

After numerous brainstorming sessions on the feasibility of the web app and Nextcloud integration with senior engineers Pedone and Orgiazzi it was decided that laboratories and assessors are not considered to be the direct users of the web app. In other words, DDT, STD, FT have access to both Nextcloud and the web app interface whereas CABs and assessors can access only Nextcloud.

In the meanwhile, the underdevelopment software monitors all of the users' actions on Nextcloud to reach the thesis objectives and be able to send emails to laboratories and assessors and messages to internal staff of Accredia through the web app based on the process's progress, whether if it was updated using the web app or Nextcloud.

3 Tools and methodologies used throughout the project

This chapter contains a summary of the theoretical methodologies and the tools used throughout the project. In sections 3.1, 3.1.1, and 3.1.2 PHP the programming language that used in developing this project is studied, pointing out its popularity.

Section 3.2 refers to the advantages of PHP's popularity, introducing Laravel framework. A software development methodology is introduced in section 3.3. Sections 3.4, 3.4.1, 3.4.2, 3.4.3, and 3.4.4 Laravel framework is studied aligned with the methodology that was mentioned in section 3.3.

Section 3.5 introduces Docker, a tool that is used to deploy and scale the web application.

Section 3.6 introduces Redis, a tool that is used in the integration of Nextcloud and the web app.

3.1 PHP

PHP is a popular object-oriented and general-purpose server-side scripting language that is specially tailored for web development. PHP was first created in 1995 by Rasmus Lerdorf and 8 versions have been introduced so far, and the latest stable version, which was released in March 2021 is used to develop this project.

3.1.1 PHP's popularity according to Web Technology Surveys

According to a study done by Web Technology Surveys [8] regarding technology usage statistics of the top 10 million websites established before 2013, PHP is the most popular server-side programming language with a staggering 79.2% dominance by April 2021. Please note that a website may use more than one server-side programming language. The languages not represented in Figure 5 all combined, account for less than one percent of the websites.

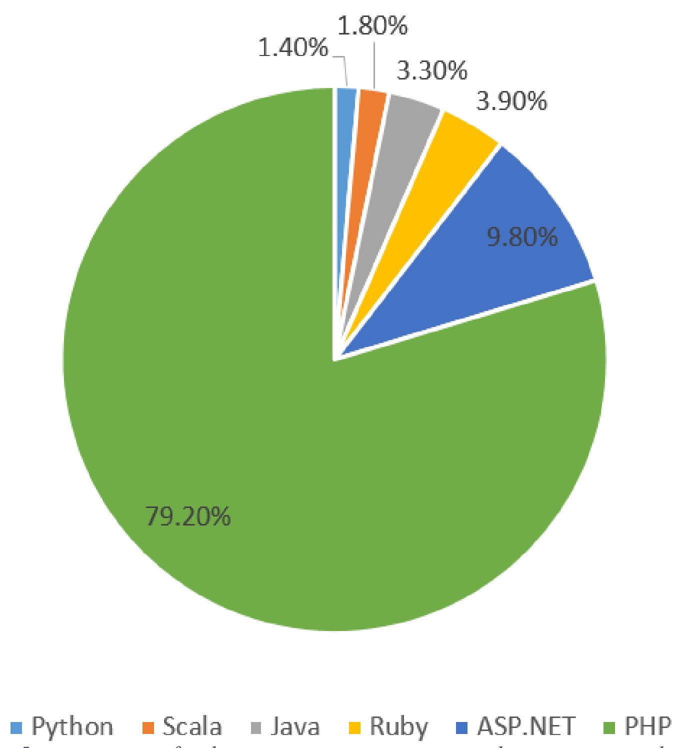


Figure 5 percentages of websites using various server-side programming language

In this study:

- The time range was limited to 2013 in order to limit the impact of domain spammers.
- Top 10 million websites either listed in Alexa (https://www.alexa.com/) or Tranco (https://tranco-list.eu/) are included in the sample.
- The sites that have no useful content, e.g. sites that only show the default web server page have been excluded from the study.
- Different subdomains are not considered separate websites. For example, this means that all the subdomains of wordpress.com, blogger.com, and other similar sites are counted as one website only.
- Redirected domains are not included. For instance, Sun.com redirects to Oracle.com and is therefore not counted
- Top 10 million websites have no statistical significance since Alexa's and Tranco's definition of "website" have slight differences

Although measuring websites traffic by their popularity isn't considered accurate, it decently represents a sample of established sites and PHP dominance.

3.1.2 PHP's popularity according to Wappalyzer

Wappalyzer [9] is a technology profiler that identifies technologies on websites. The top Programming languages technologies according to wappalyzer.com based on market share in 2020 are reported in Figure 6 Market share of the top programming languages technologies on websites. The languages not mentioned in figure 6, have less than one percent market share, all combined.

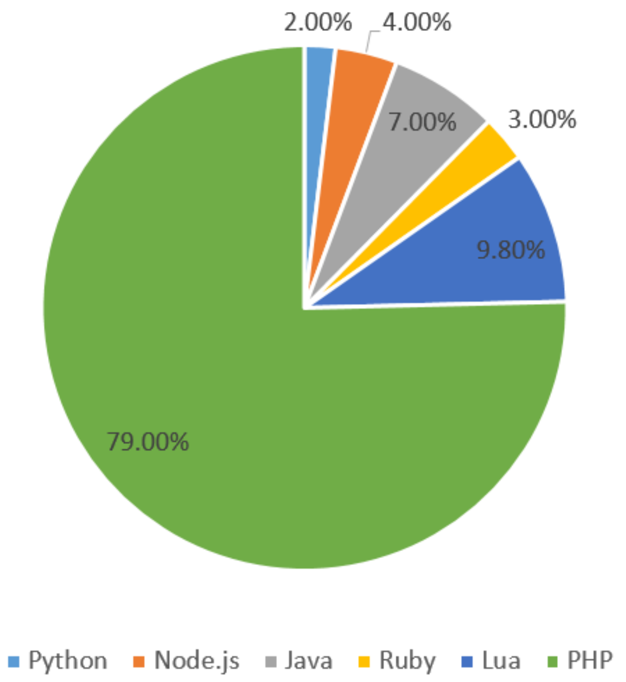


Figure 6 Market share of the top programming languages technologies on websites

Both Figures 5 and 6 confirm PHP's massive popularity in the website business.

3.2 Frameworks in PHP

PHP's popularity brings in a huge amount of development tools and frameworks which mostly contain out of the box functions, classes, and libraries aiming to provide the elements for software design pattern realization [10], [11]. A number of these frameworks are blessed with proactive and supportive development communities. Laravel that is used in this project and explained later in this chapter, is considered to be among one of the best PHP frameworks. Features like easy data migration and template engine viewing make Laravel an ideal choice for developing complex applications.

Using a well-tested framework allows a significant reduction in the amount of the written code that can otherwise cause errors and escalate to a breakdown.

3.3 The MVC (model view controller) architecture

MVC is a widely spread architectural pattern used to develop and design software. Software is separated into three components: Model, View, and Controller. The first letters of each component make up the MVC acronym. In the development of complex applications like this project, using MVC architecture gives some structure to the code, making it easier to work with, and dramatically enhances maintainability.

The Model defines the data structure and stores the values in any number of data storage systems, moreover, the model responds to queries sent by the controller and can notify participants about any information changes in event-driven systems.

As it is seen in (Figure 7) the View component renders the data from the model into a suitable form to update users. Users are involved in the process's cycles and the users' actions are transmitted to the Controller component of MVC architecture, where the controller calls the model's objects to carry out necessary actions, the model component that has been manipulated by the controller updates the user interface or the view component.

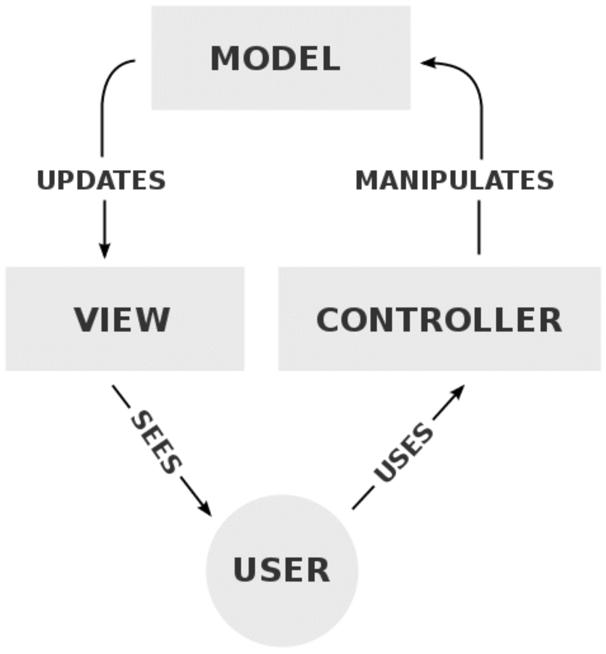


Figure 7MVC software development architecture pattern [12]

3.4 Laravel

Laravel is a free, open-source PHP-based web framework that is licensed under the MIT license (Massachusetts Institute of Technology) and is largely based on the MVC architecture pattern. Laravel was first released in June 2011 and 8 versions have been released by the date of the research. The most updated stable version of Laravel at the time has been used to develop this project (version 8.16.1). Until this date, about 2500 contributors have contributed well over 30 thousand contributions to build this well-known tool, as almost 130 million installs have been reported so far. [13], [14]

3.4.1 Advantages of using Laravel

Laravel provides a developer-friendly experience and makes the development process easier and faster. It demands little resources to run the applications thus lowers hosting costs. Laravel has a huge ecosystem and its numerous libraries foster the development rate. Moreover, it is easily scalable via Docker.

There is an enormous synergy between Laravel's open-source architecture and its massive popularity and its performance. Laravel is updated frequently to fix the most recent vulnerabilities, making it more secure and robust almost every day.

Automation of testing environment and integration with 3rd-party services are the other benefits that arise from using Laravel. [15], [16]

3.4.2 Model in Laravel

The Illuminate Database (library used) component is the database toolkit that serves as the database layers of Laravel and provides query builder, object-relational mapper (ORM), and schema builder.

Eloquent is the ORM in Laravel, when using Eloquent, each database table has a corresponding "Model" that is used to interact with that table. In addition to retrieving records from the database table, eloquent models allow you to insert, update, and delete records from the table as well.

These tools are used to store all of the information apart from storing the accreditation files themselves, which are saved only on Nextcloud, however, the files on Nextcloud are indexed using the mentioned tools to be able to call these objects whenever it's needed.

Moreover, the process structure class, which is like the backbone of the software and handles tasks like sending to-do messages is programmed within the model component. [17], [18]

3.4.3 View in Laravel

The web app user interface is powered by a templating engine called Blade that is included with Laravel. All Blade templates are compiled into plain PHP code and cached until modified, meaning Blade adds zero overhead to the application. [19]

3.4.4 Controller in Laravel

The controller receives information from the view component as the user updates the view component. The controller manipulates the model by calling the model's objects and performing appropriate actions. Last but not least, the controller updates the view or the user interface.

Livewire is a full-stack framework for Laravel that is a link between view and controller. Livewire makes Laravel more back-end developer-friendly by processing PHP codes and transforming them to JavaScript. The render method of an instance of Livewire class renders the view and the other methods interact with the view as controller. [20]

3.5 Docker: Horizontal and vertical scaling of web applications

In theory, there are several solutions proposed to increase the flexibility of the application by controlling the horizontal and vertical elasticity of container-based applications to better cope with varying workloads, however, it's a common belief that vertical scaling outruns horizontal scaling. [21] But what is the vertical vs horizontal scaling in the context?

Adding more units to the pool of resources is called horizontal scaling or scaling out, whereas fostering the power of the existing unit is called vertical scaling or scaling up.

An example of horizontal scaling can be adding more VMs (virtual machine) to the system but like this, there's an inevitable need to install the operating system and essential drivers on the VMs before utilizing them. This is a fixed cost to bear while integrating VMs to the pool of resources which consists of a fraction of the processing power and memory of the VMs. This of course favors vertical scaling.

Scaling the web application via Docker is an example of both vertical and horizontal scaling, which has been used to scale the underdevelopment application. Docker is written in Golang and it is a native software of Linux and Windows operating systems. However, Docker for macOS, is architecturally different than Docker on Linux. Docker for macOS instead has to spin up its own Linux VM. [22]

Moreover, Docker is conceived to be a more stable and cost-effective scaling approach rather than adding more VMs or physical servers.

3.6 Redis

Redis is an open-source NoSQL data structure licensed under the BSD (Berkeley Software Distribution) license, Redis is used as a database, cache, and message broker, and stores the data as key values in memory [23]. Redis is a lightweight software with very high performance used in this project to adapt Nextcloud in order to interact with the web app, specifically to send the information of what has happened on Nextcloud to the web app, events like uploading or renaming a file. This information communication is essential for the development of software that is going to have two input portals, Nextcloud or the web app.

4 The web page application

Although this chapter is called the web page application, more than one application is discussed in this chapter. This chapter starts with screenshots of the end-product of this project. Section 4.1 neglects the back-end-related topics and accompanies the reader to the starting point of a new process. Section 4.2 provides the guideline used in the next sections, section 4.2 is focused on the back-end of the web app and defining process structure.

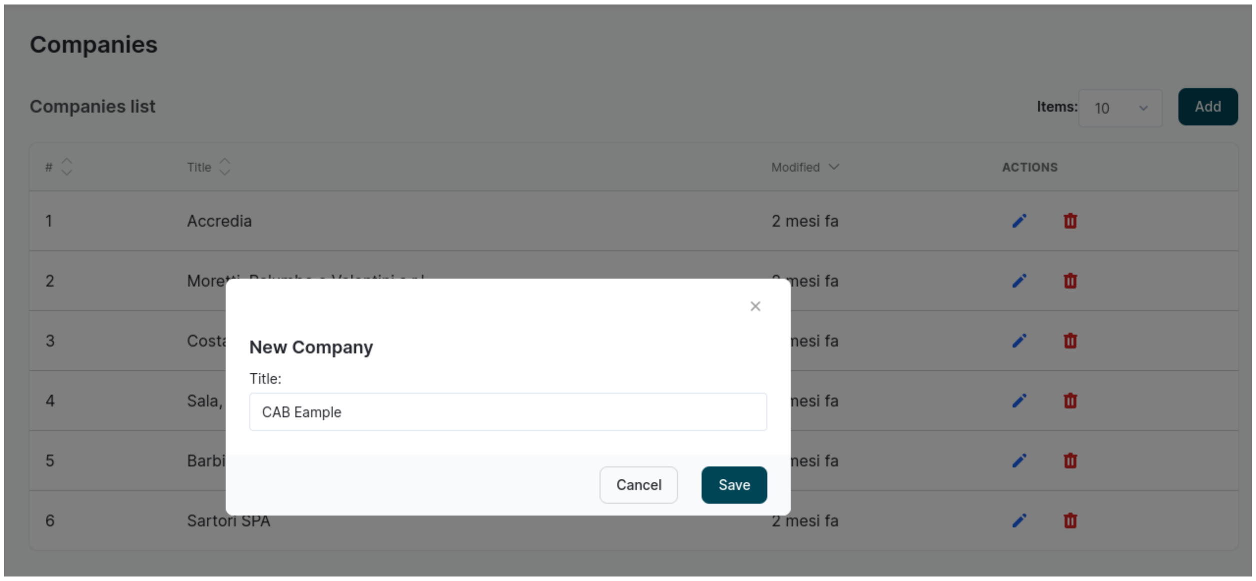
Sections 4.3, 4.4, 4.5, 4.6, 4.7, and 4.8 follow the guidelines provided in section 4.2. Section 4.3 discusses what happens after creating/adding a new process. Section 4.4 discusses the technical aspects of uploading a file using the web app. In section 4.5 technical aspects of decision-making are discussed. Section 4.6 refers to uploading of files using Nextcloud. Subsection 4.6.1 briefly explains the other application developed in this project that works on Nextcloud. Subsections 4.6.2 and 4.6.3 refer to the technical aspects related to the web app in adapting the information received from Nextcloud.

Sections 4.6 and 4.7 rapidly demonstrate the replicability of the web app's functionalities.

Section 4.9 analyzes the suggested approach and points out a critical point of the project.

4.1 User management

The part of the process that is sought to be implemented starts with STD creating a new process to respond to a DA, however, the applicant's contact information has to be transmitted to the web app before starting a new process. After STD logs into the web app, before creating and starting a new process, STD is ought to add a new company using the Companies panel of the web app (Figure 8 Companies panel, add modal).


Companies

Companies list

Items: 10 Add

#	Title	Modified	ACTIONS
1	Accredia	2 mesi fa	✏ 🗑
2	Moret	mesi fa	✏ 🗑
3	Costa	mesi fa	✏ 🗑
4	Sala,	mesi fa	✏ 🗑
5	Barbi	mesi fa	✏ 🗑
6	Sartori SPA	2 mesi fa	✏ 🗑

New Company

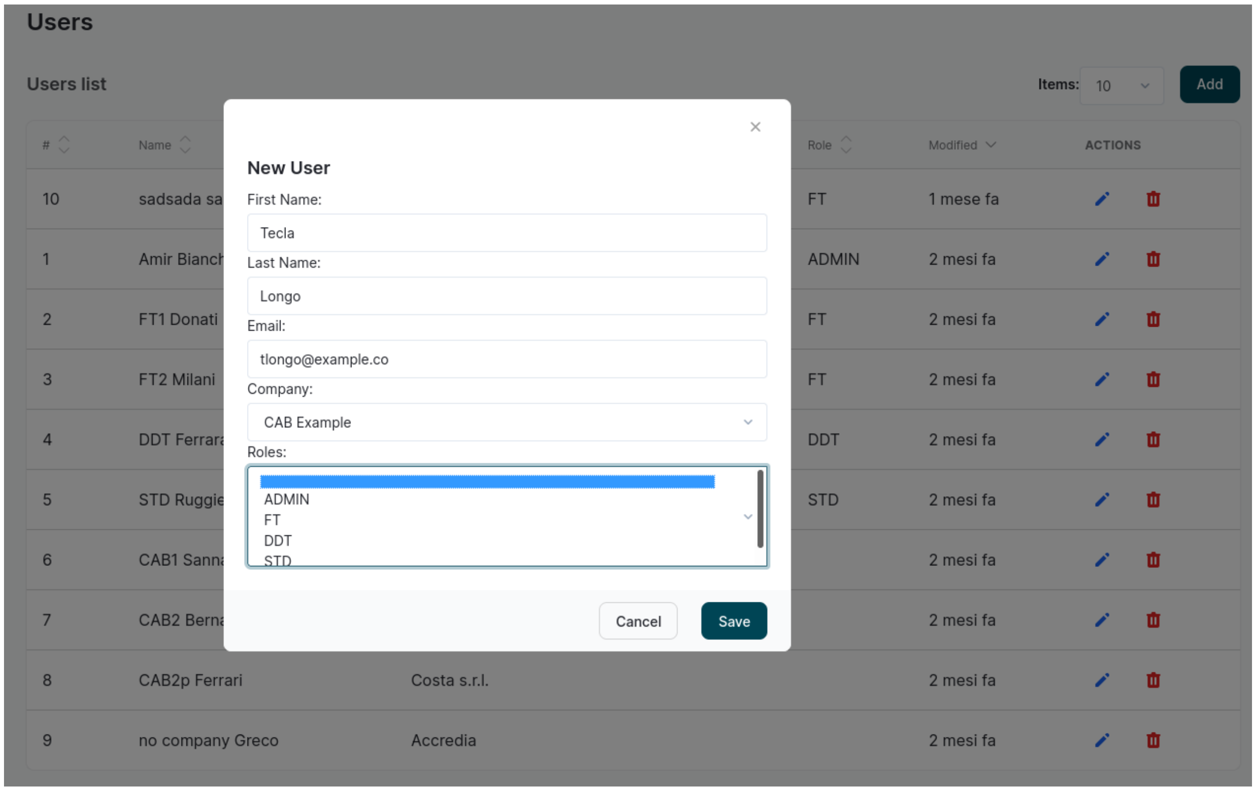
Title: CAB Eample

Cancel Save

Figure 8 Companies panel, add modal

A new company that represents CAB in this project can be added using the Companies panel of the web app. The list of the existing companies is also present in the Companies panel.

And then STD can add a new user using the Users panel of the web app (Figure 9 Users panel). During which, STD assigns the user to the companies that were added prior.



Users

Users list

Items: 10

Add

#	Name		Role	Modified	ACTIONS
10	sadsada sa		FT	1 mese fa	
1	Amir Bianch		ADMIN	2 mesi fa	
2	FT1 Donati		FT	2 mesi fa	
3	FT2 Milani		FT	2 mesi fa	
4	DDT Ferrar		DDT	2 mesi fa	
5	STD Ruggie		STD	2 mesi fa	
6	CAB1 Sann			2 mesi fa	
7	CAB2 Berna			2 mesi fa	
8	CAB2p Ferrari	Costa s.r.l.		2 mesi fa	
9	no company Greco	Accredia		2 mesi fa	

New User

First Name: Tecla

Last Name: Longo

Email: tlongo@example.co

Company: CAB Example

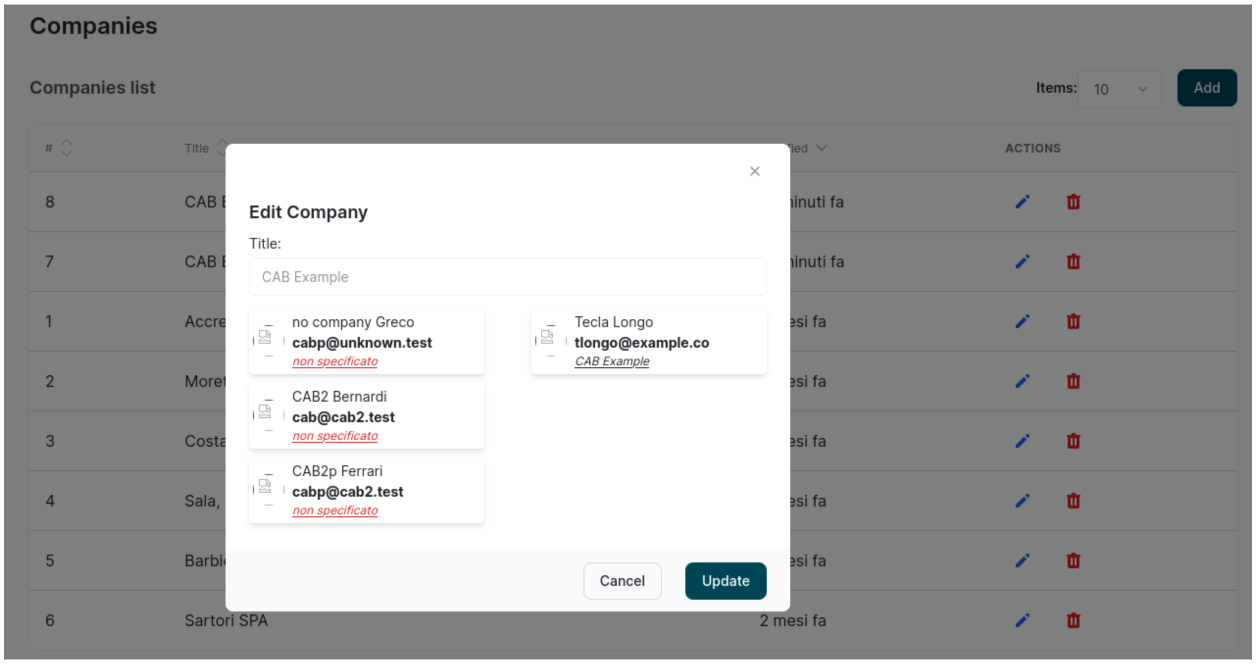
Roles:
ADMIN
FT
DDT
STD

Cancel Save

Figure 9 Users panel

Please note that adding a new user to assign as a company representative in Figure 9 Users panel, the blank option is to be selected in the roles selection section if the company is considered CAB. More on this panel, STD can also add a new user with an FT role and select the company Accredia. Like this, STD can assign the newly added user as the FT to manage the process upon the modal of adding a new process.

A user can be unassigned from a company by first clicking on the edit button of the desired company (the blue pen icon in Figure 10 Companies panel, Edit modal) and then clicking on the assigned user will render him/her unassigned. The unassigned user can be assigned back to the company by clicking on him/her once he/she is unassigned.



Companies

Companies list

Items: 10 Add

#	Title	…ied	ACTIONS
8	CAB E	…inuti fa	
7	CAB E	…inuti fa	
1	Accre	…esi fa	
2	Moret	…esi fa	
3	Costa	…esi fa	
4	Sala,	…esi fa	
5	Barbi	…esi fa	
6	Sartori SPA	2 mesi fa	

Edit Company

Title:

CAB Example

no company Greco
cabp@unknown.test
non specificato

Tecla Longo
tlongo@example.co
CAB Example

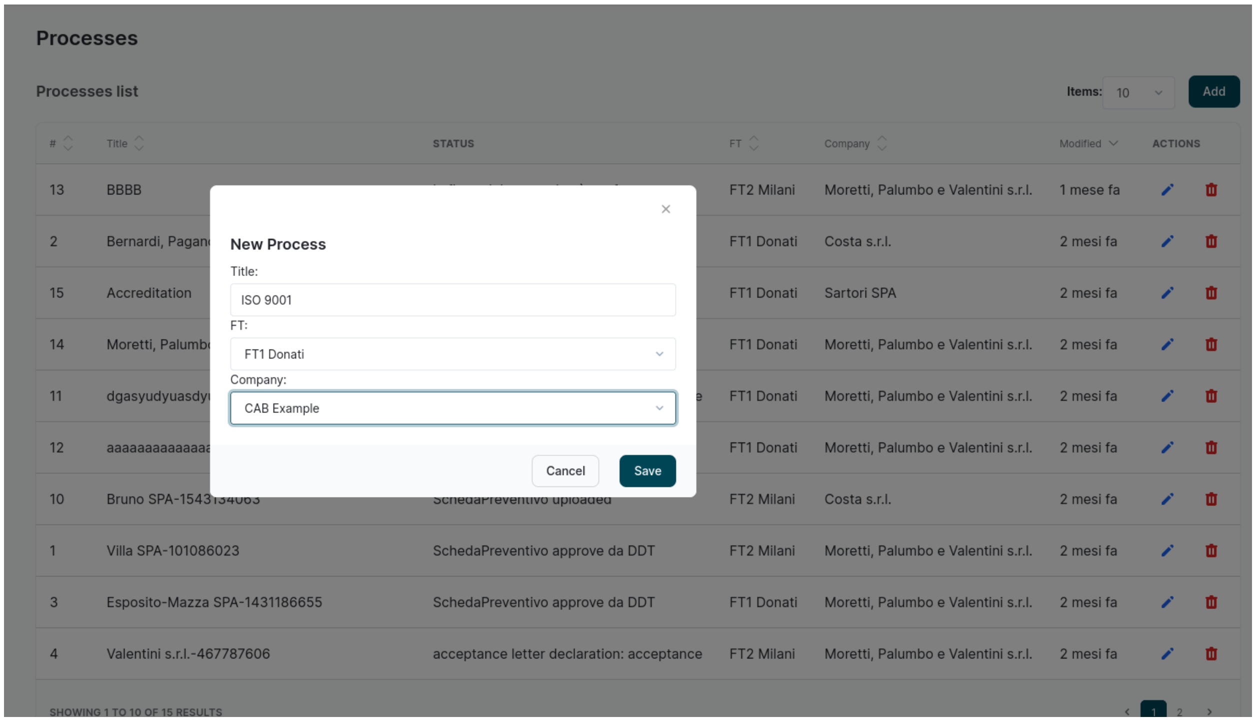
CAB2 Bernardi
cab@cab2.test
non specificato

CAB2p Ferrari
cabp@cab2.test
non specificato

Cancel Update

Figure 10 Companies panel, Edit modal

After these steps, STD can start a new process by using the modal for adding a new process in the process panel (Figure 11 Processes panel, add modal).



Processes

Processes list

Items: 10 Add

#	Title	STATUS	FT	Company	Modified	ACTIONS
13	BBBB		FT2 Milani	Moretti, Palumbo e Valentini s.r.l.	1 mese fa	
2	Bernardi, Pagan		FT1 Donati	Costa s.r.l.	2 mesi fa	
15	Accreditation		FT1 Donati	Sartori SPA	2 mesi fa	
14	Moretti, Palumb		FT1 Donati	Moretti, Palumbo e Valentini s.r.l.	2 mesi fa	
11	dgasyudyuasdy		FT1 Donati	Moretti, Palumbo e Valentini s.r.l.	2 mesi fa	
12	aaaaaaaaaaaaa		FT1 Donati	Moretti, Palumbo e Valentini s.r.l.	2 mesi fa	
10	Bruno SPA-1543134063	SchedaPreventivo uploaded	FT2 Milani	Costa s.r.l.	2 mesi fa	
1	Villa SPA-101086023	SchedaPreventivo approve da DDT	FT2 Milani	Moretti, Palumbo e Valentini s.r.l.	2 mesi fa	
3	Esposito-Mazza SPA-1431186655	SchedaPreventivo approve da DDT	FT1 Donati	Moretti, Palumbo e Valentini s.r.l.	2 mesi fa	
4	Valentini s.r.l.-467787606	acceptance letter declaration: acceptance	FT2 Milani	Moretti, Palumbo e Valentini s.r.l.	2 mesi fa	

SHOWING 1 TO 10 OF 15 RESULTS

New Process

Title: ISO 9001

FT: FT1 Donati

Company: CAB Example

Cancel Save

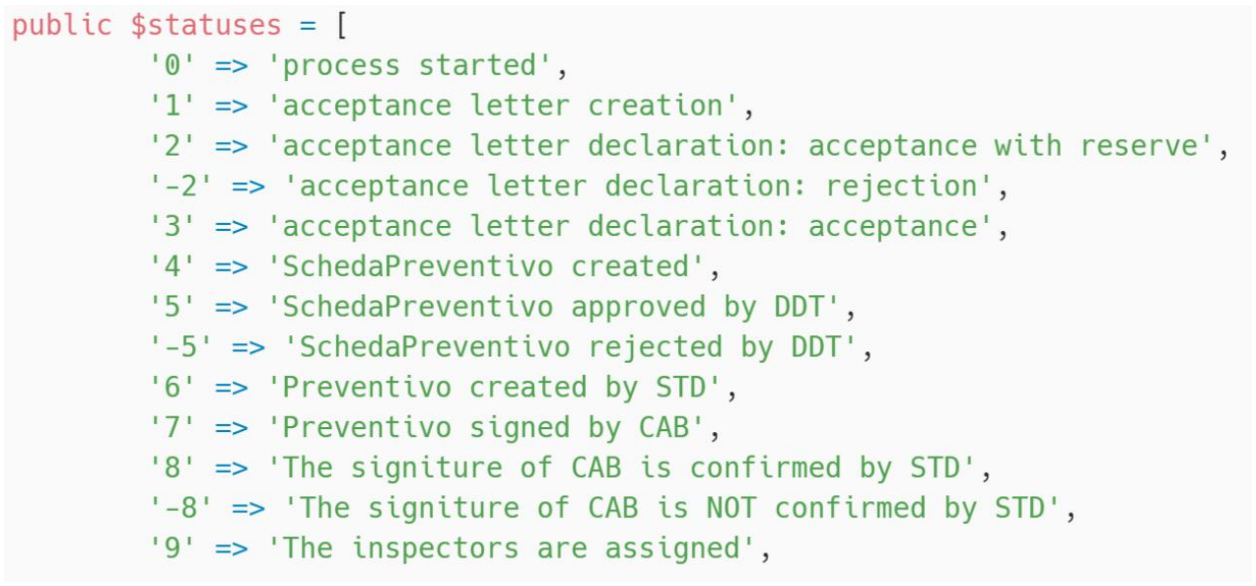
Figure 11 Processes panel, add modal

4.2 Defining checkpoints/steps

Before going on with creating a new process, let's discuss the process class of the model component of the application which is like a roadmap with well-defined checkpoints containing information on the process attributes, statuses, deadlines, warning dates, to-do messages, and the roles to deliver the to-do messages in each checkpoint.

Please note that this class handles to-do messages on the web app and communicates with the direct users of the web app. CABs are informed by emails which are better explained later in this chapter (Event-Listener logic).

As an example, Figure 12 the statuses function, Figure 13 to-do messages function, and Figure 14 to-do roles function of the processes class are reported below:

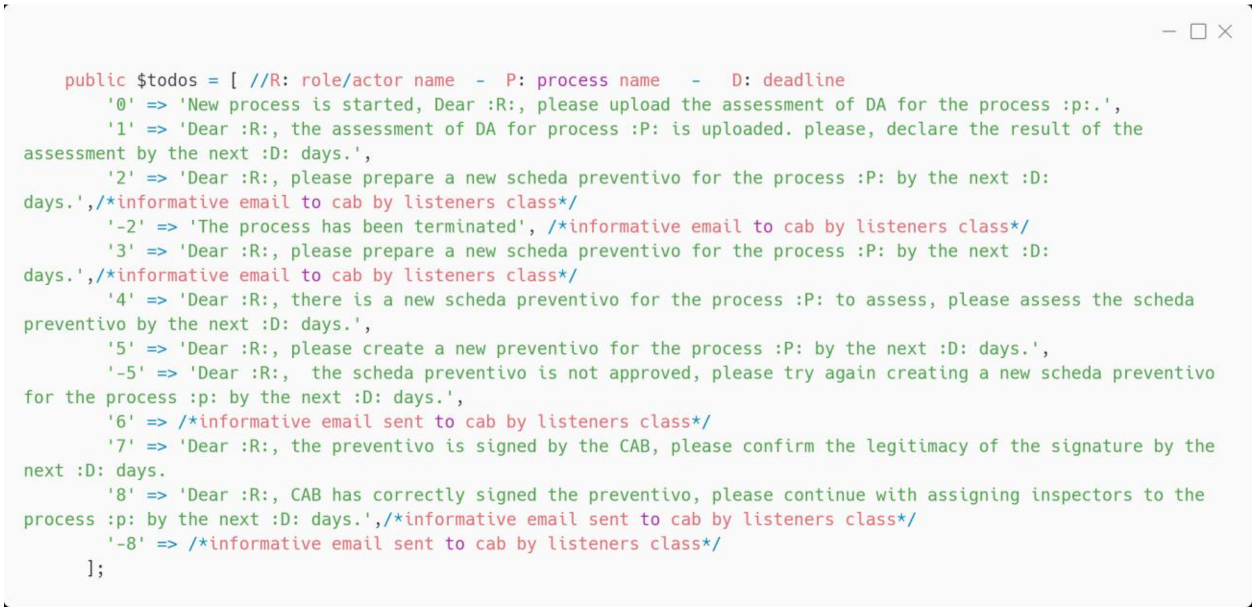


```
public $statuses = [
        '0' => 'process started',
        '1' => 'acceptance letter creation',
        '2' => 'acceptance letter declaration: acceptance with reserve',
        '-2' => 'acceptance letter declaration: rejection',
        '3' => 'acceptance letter declaration: acceptance',
        '4' => 'SchedaPreventivo created',
        '5' => 'SchedaPreventivo approved by DDT',
        '-5' => 'SchedaPreventivo rejected by DDT',
        '6' => 'Preventivo created by STD',
        '7' => 'Preventivo signed by CAB',
        '8' => 'The signiture of CAB is confirmed by STD',
        '-8' => 'The signiture of CAB is NOT confirmed by STD',
        '9' => 'The inspectors are assigned',
```

Figure 12 the statuses function

Numbers in a whole number set have been indexed to each checkpoint, meaning that number zero, refers to starting of a process (creation of a new process by STD) and progresses by one step with advancement to a new checkpoint.

Negative values of the numbers indexed to the checkpoints represent the situation when a checkpoint hasn't gone to a good end. A negative checkpoint often means the repetition of problematic checkpoints and tasks. Nonetheless, some checkpoints don't require a negative number assignment due to the action's nature, for example, STD uploading the acceptance letter (assessment of DA) is considered to be one of the non-repetitive tasks.



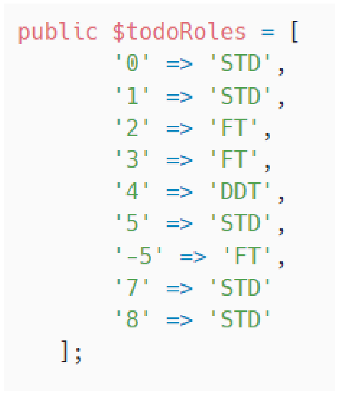
```
    public $todos = [ //R: role/actor name  -  P: process name   -   D: deadline
        '0' => 'New process is started, Dear :R:, please upload the assessment of DA for the process :p:.',
        '1' => 'Dear :R:, the assessment of DA for process :P: is uploaded. please, declare the result of the
assessment by the next :D: days.',
        '2' => 'Dear :R:, please prepare a new scheda preventivo for the process :P: by the next :D:
days.',/*informative email to cab by listeners class*/
        '-2' => 'The process has been terminated', /*informative email to cab by listeners class*/
        '3' => 'Dear :R:, please prepare a new scheda preventivo for the process :P: by the next :D:
days.',/*informative email to cab by listeners class*/
        '4' => 'Dear :R:, there is a new scheda preventivo for the process :P: to assess, please assess the scheda
preventivo by the next :D: days.',
        '5' => 'Dear :R:, please create a new preventivo for the process :P: by the next :D: days.',
        '-5' => 'Dear :R:,  the scheda preventivo is not approved, please try again creating a new scheda preventivo
for the process :p: by the next :D: days.',
        '6' => /*informative email sent to cab by listeners class*/
        '7' => 'Dear :R:, the preventivo is signed by the CAB, please confirm the legitimacy of the signature by the
next :D: days.
        '8' => 'Dear :R:, CAB has correctly signed the preventivo, please continue with assigning inspectors to the
process :p: by the next :D: days.',/*informative email sent to cab by listeners class*/
        '-8' => /*informative email sent to cab by listeners class*/
      ];
```

Figure 13 to-do messages function

Please note that ":R:" used in Figure 13 to-do messages function refers to the Figure 14 to-do roles function and ":P:", ":D:" refer to process title, and remaining days to the deadline of the ongoing task respectively.

In checkpoints 2, -2, and 3 there's also the need to send emails to CAB to update them on the outcome of DA.

Furthermore, in checkpoints 6 and -8 there's only the need to send emails to CAB as a reminder to sign the Preventivo, as a result, checkpoints 6 and -8 are not mentioned in the todoroles function (Figure 14 to-do roles function).



```
public $todoRoles = [
        '0' => 'STD',
        '1' => 'STD',
        '2' => 'FT',
        '3' => 'FT',
        '4' => 'DDT',
        '5' => 'STD',
        '-5' => 'FT',
        '7' => 'STD'
        '8' => 'STD'
   ];
```

Figure 14 to-do roles function

Splitting the process into checkpoints results in excellent record-keeping and transparency capability for the underdevelopment software. In order to align with the integrity of the accreditation system, once a checkpoint has passed, it's considered irreversible.

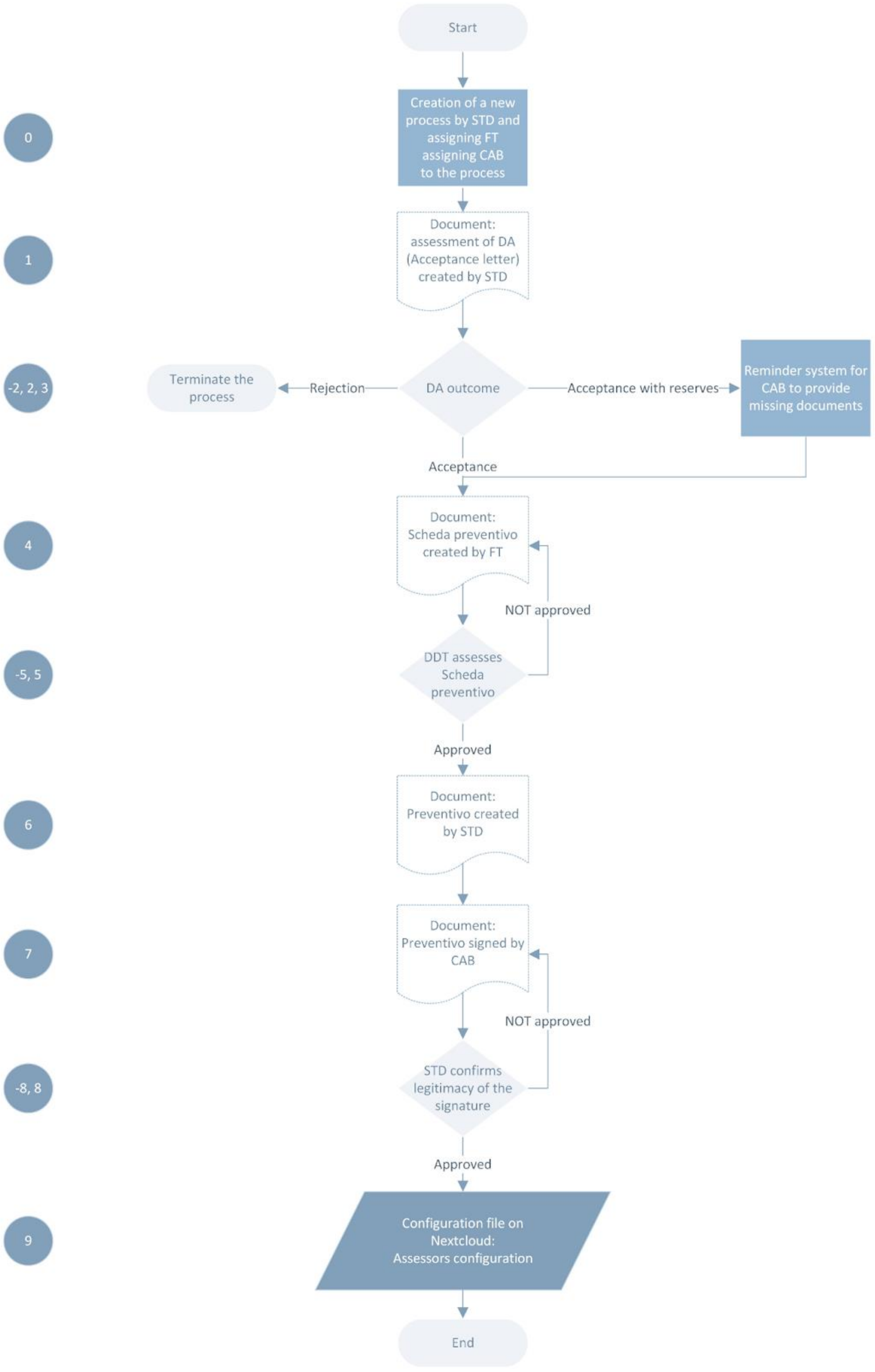

Start
0
Creation of a new process by STD and assigning FT assigning CAB to the process
1
Document: assessment of DA (Acceptance letter) created by STD
-2, 2, 3
Terminate the process
Rejection
DA outcome
Acceptance with reserves
Reminder system for CAB to provide missing documents
Acceptance
4
Document: Scheda preventivo created by FT
NOT approved
-5, 5
DDT assesses Scheda preventivo
Approved
6
Document: Preventivo created by STD
7
Document: Preventivo signed by CAB
NOT approved
-8, 8
STD confirms legitimacy of the signature
Approved
9
Configuration file on Nextcloud: Assessors configuration
End

Figure 15 the process's flowchart

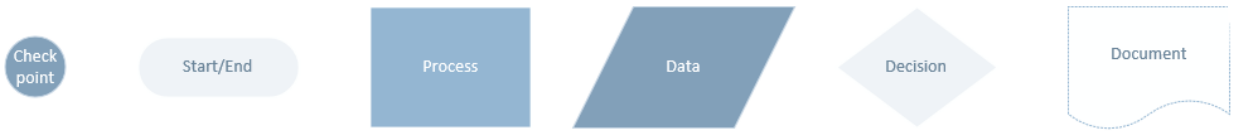
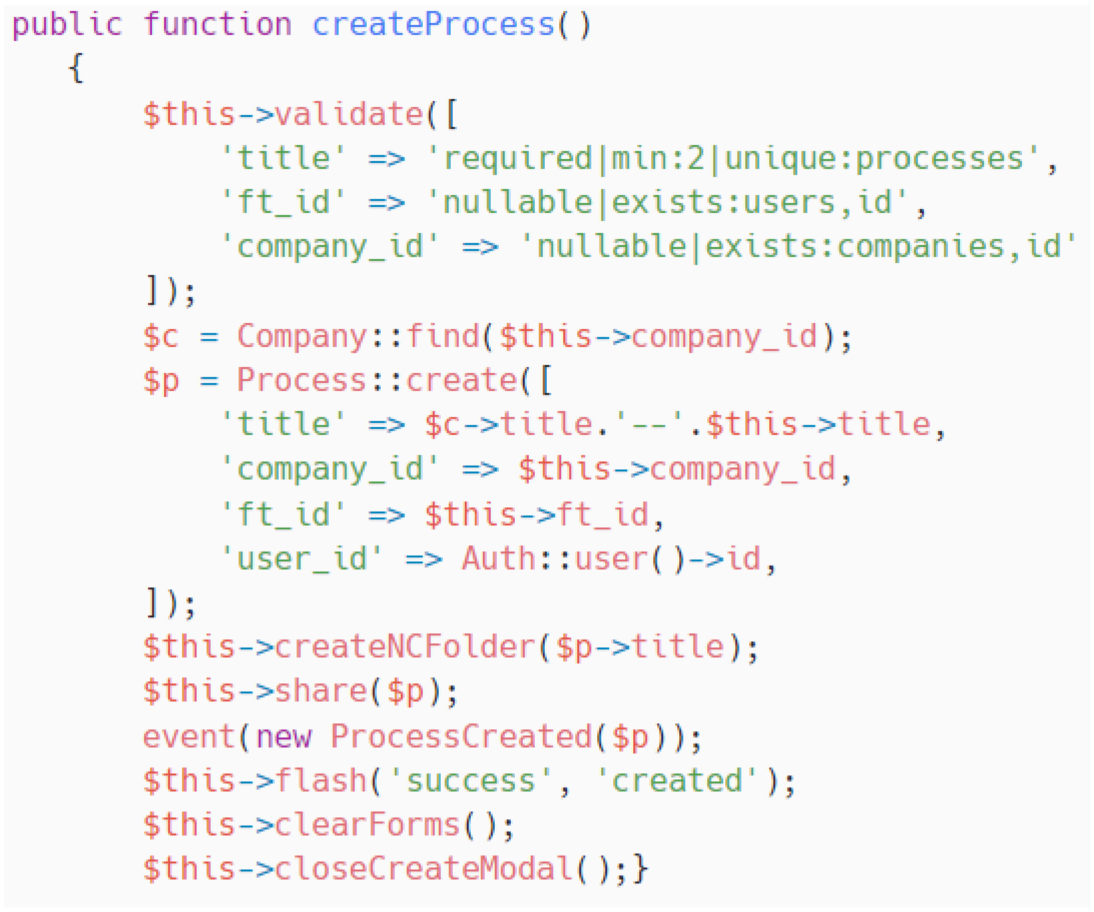


Figure 16 Flowchart's ledger

Figure 15 the process's flowchart covers the part of the process that is implemented in this thesis.

4.3 Checkpoint zero

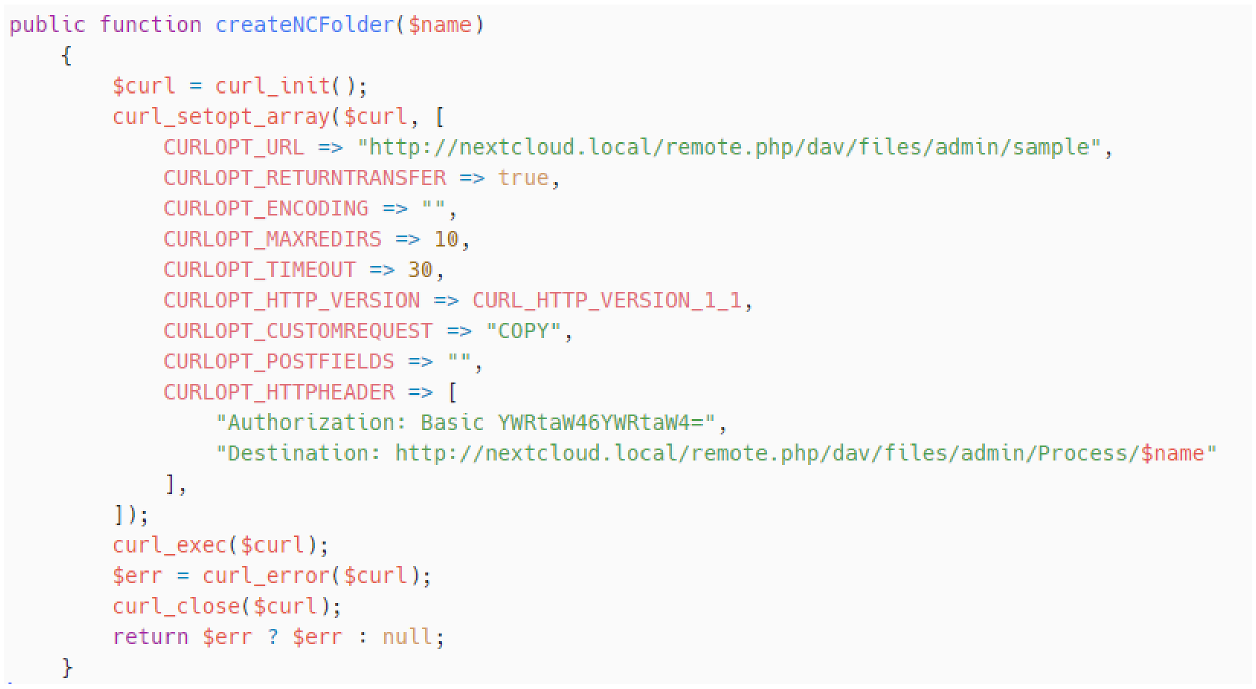
Checkpoint zero is starting of a new process by STD, and assigning an FT and CAB to the process. Figure 17 Create/add a new process function is programmed in the processes class of the livewire library of the controller component. Figure 17 Create/add a new process function represents the back-end code of Figure 11 Processes panel, add modal.



```
public function createProcess()
   {
       $this->validate([
           'title' => 'required|min:2|unique:processes',
           'ft_id' => 'nullable|exists:users,id',
           'company_id' => 'nullable|exists:companies,id'
       ]);
       $c = Company::find($this->company_id);
       $p = Process::create([
           'title' => $c->title.'--'.$this->title,
           'company_id' => $this->company_id,
           'ft_id' => $this->ft_id,
           'user_id' => Auth::user()->id,
       ]);
       $this->createNCFolder($p->title);
       $this->share($p);
       event(new ProcessCreated($p));
       $this->flash('success', 'created');
       $this->clearForms();
       $this->closeCreateModal();}
```

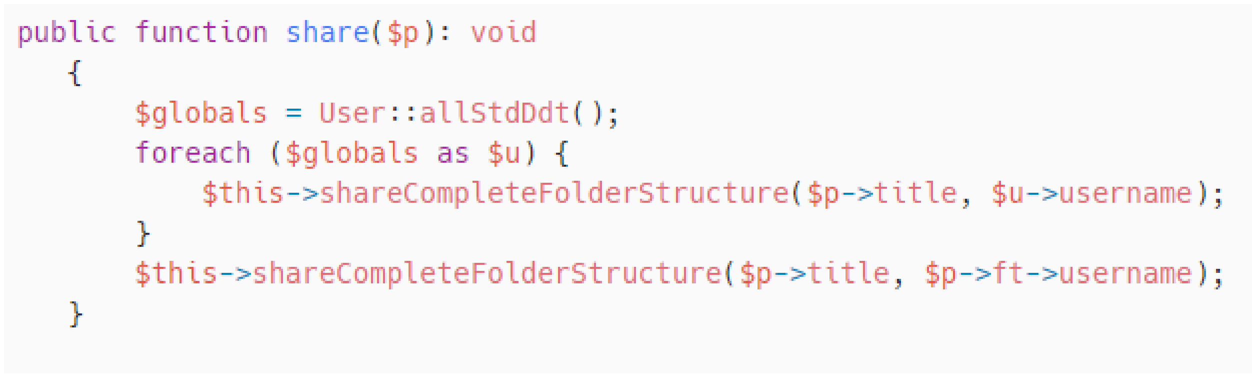
Figure 17 Create/add a new process function

Figure 17 Create/add a new process function Calls the Figure 18 creating new folder structure function to create a folder with a definite structure as it was described in section 2.7, the folder structure. The create process function Figure 17 Create/add a new process function also calls share function Figure 19 share function, to share the process's folder with STD, DDT, and the assigned FT.



```
public function createNCFolder($name)
    {
        $curl = curl_init();
        curl_setopt_array($curl, [
            CURLOPT_URL => "http://nextcloud.local/remote.php/dav/files/admin/sample",
            CURLOPT_RETURNTRANSFER => true,
            CURLOPT_ENCODING => "",
            CURLOPT_MAXREDIRS => 10,
            CURLOPT_TIMEOUT => 30,
            CURLOPT_HTTP_VERSION => CURL_HTTP_VERSION_1_1,
            CURLOPT_CUSTOMREQUEST => "COPY",
            CURLOPT_POSTFIELDS => "",
            CURLOPT_HTTPHEADER => [
                "Authorization: Basic YWRtaW46YWRtaW4=",
                "Destination: http://nextcloud.local/remote.php/dav/files/admin/Process/$name"
            ],
        ]);
        curl_exec($curl);
        $err = curl_error($curl);
        curl_close($curl);
        return $err ? $err : null;
    }
```

Figure 18 creating new folder structure function



```
public function share($p): void
    {
        $globals = User::allStdDdt();
        foreach ($globals as $u) {
            $this->shareCompleteFolderStructure($p->title, $u->username);
        }
        $this->shareCompleteFolderStructure($p->title, $p->ft->username);
    }
```

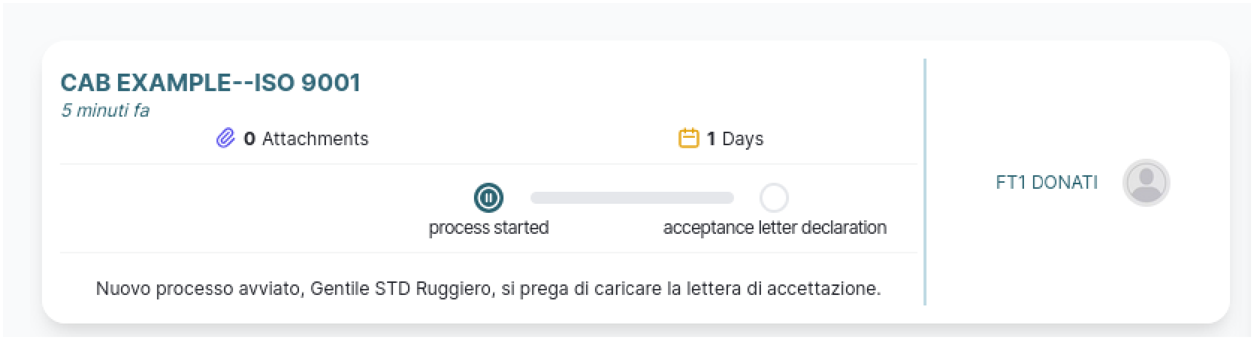
Figure 19 share function

Figure 19 share function calls Figure 20 share folder structure function, the latter functions gets the process title and usernames from the former function and posts a curl command to share the folder structure.

```
public function shareCompleteFolderStructure($process, $username)
    {
        $curl = curl_init();
        $process = str_replace(' ', '%20', $process);
        curl_setopt_array($curl, [
            CURLOPT_URL => env('NEXT_CLOUD_DOMAIN')."/ocs/v2.php/apps/files_sharing/api/v1/shares?
path=/Process/{$process}&shareType=0&shareWith={$username}&permissions=15",
            CURLOPT_RETURNTRANSFER => true,
            CURLOPT_ENCODING => "",
            CURLOPT_MAXREDIRS => 10,
            CURLOPT_TIMEOUT => 30,
            CURLOPT_HTTP_VERSION => CURL_HTTP_VERSION_1_1,
            CURLOPT_CUSTOMREQUEST => "POST",
            CURLOPT_POSTFIELDS => "",
            CURLOPT_HTTPHEADER => [
                "Authorization: Basic ".
base64_encode(env('NEXT_CLOUD_USER').':'.env('NEXT_CLOUD_PASSWORD')),
                "OCS-APIRequest: true"
            ],
        ]);

        curl_exec($curl);
        $err = curl_error($curl);
        curl_close($curl);
        return $err ? $err : null;
    }
```
Figure 20 share folder structure function

4.4 Checkpoint 1

Uploading the assessment of DA (acceptance letter)


CAB EXAMPLE--ISO 9001

5 minuti fa

0 Attachments

1 Days

process started

acceptance letter declaration

Nuovo processo avviato, Gentile STD Ruggiero, si prega di caricare la lettera di accettazione.

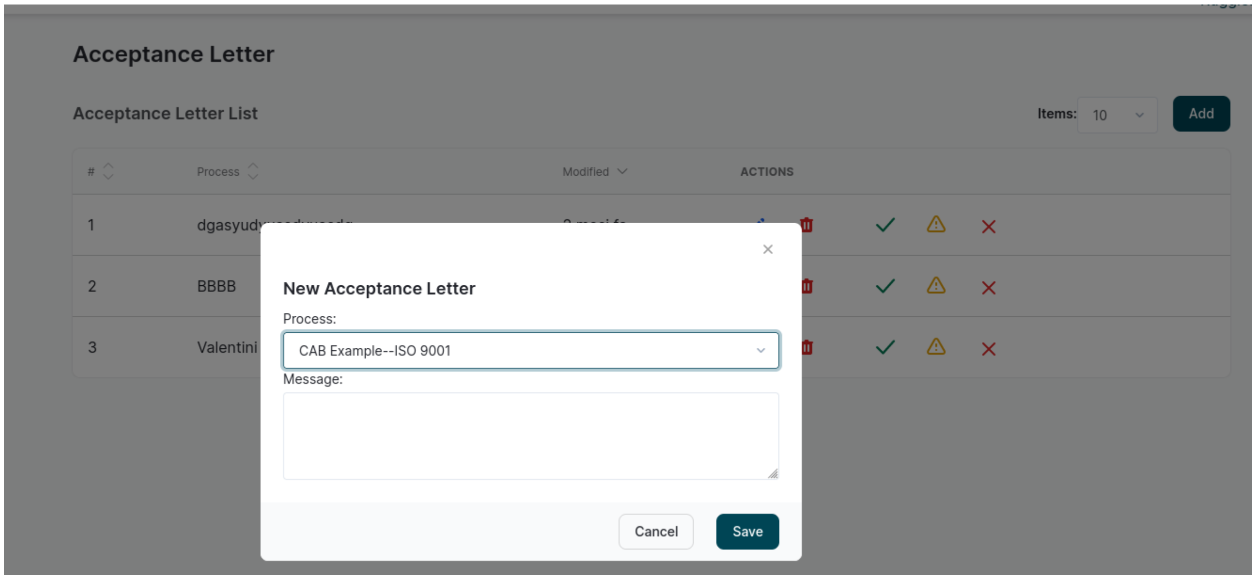
FT1 DONATI

Figure 21 Dashboard panel of STD

Anytime at this point, a new message, in accordance with Figure 13 to-do messages function, checkpoint zero (please note that the to-do messages are translated to English and reported in Figure 13 to-do messages function, while the to-do messages shown in the screenshots of dashboard panel are in Italian), will appear in the dashboard panel of STD in the web app, telling that uploading the acceptance letter will push the process to the next checkpoint (Figure 21). This reminder should go away once this checkpoint is passed.

The acceptance letter can be uploaded directly from the web app interface or STD can upload it onto the destination path over the Nextcloud. In case STD chooses to upload the file using Nextcloud, the update will be detected in a maximum of one minute, which is studied in uploading files in the next step.

For this step let’s get into the details of uploading a file using the web app. the acceptance letters class of the livewire library powers the acceptance letter panel Figure 22 which is the interface to upload and also to declare the assessment of DA.



Acceptance Letter

Acceptance Letter List

Items: 10 Add

#	Process	Modified	ACTIONS
1	dgasyud		
2	BBBB		
3	Valentini		

New Acceptance Letter

Process:

CAB Example--ISO 9001

Message:

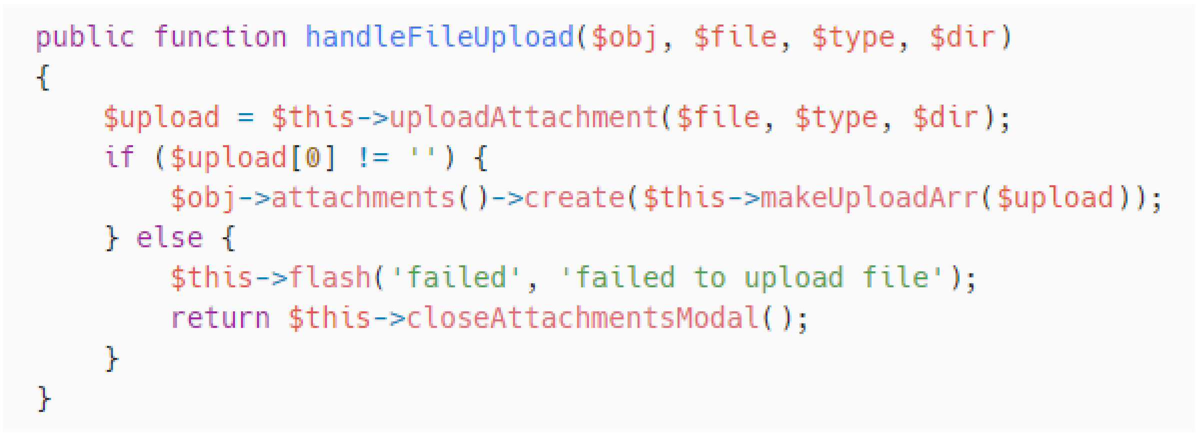
Figure 22 Acceptance letter panel of STD

The attachfiles function Figure 23 is programmed in the acceptance letters class and passes the file uploaded to handlefileupload function (Figure 24).

```
    public function attachFiles(AcceptanceLetter $al)
    {
        $this->validate([
            'acceptance_letter' => 'required|mimetypes:pdf|mimes:application/pdf',
            'attachments.*' => 'mimetypes:pdf|mimes:application/pdf'
        ]);
        if (!is_null($this->acceptance_letter)) {
            $this->handleFileUpload($al, $this->acceptance_letter, 'acceptance-letter', $al->name .
'/00acceptance_letter');
            event(new AcceptanceLetterUploaded($al));
        }
        foreach ($this->attachments as $a) {
            $this->handleFileUpload($al, $a, 'acceptance-letter', 'attachments');
        }
        $this->closeAttachmentsModal();
        $this->flash('success', 'All files uploaded');
    }
```

Figure 23 Attach/upload files function

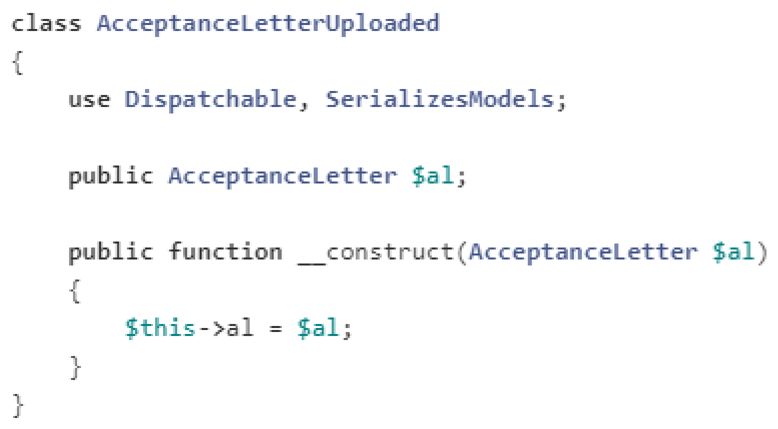
There are methods like Figure 23 in other classes in which there is uploading of a file. These functions index the file and pass the information like the destination directory to the handlefileupload function Figure 24 which is programmed in the base component class of livewire.



```
public function handleFileUpload($obj, $file, $type, $dir)
{
    $upload = $this->uploadAttachment($file, $type, $dir);
    if ($upload[0] != '') {
        $obj->attachments()->create($this->makeUploadArr($upload));
    } else {
        $this->flash('failed', 'failed to upload file');
        return $this->closeAttachmentsModal();
    }
}
```

Figure 24 handlefileupload function

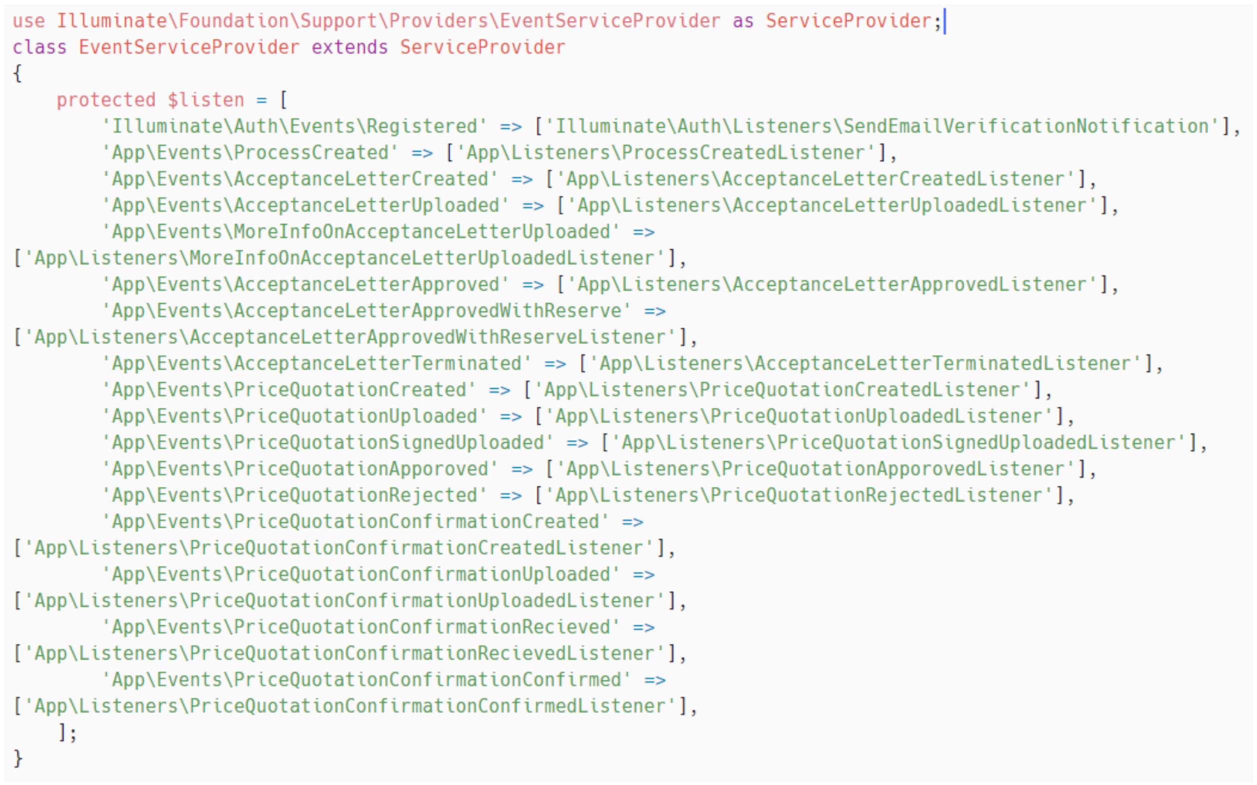
In Figure 23 an event name acceptanceletteruploaded (Figure 25 event acceptanceletteruploaded) is fired after using the handlefileupload method.



```
class AcceptanceLetterUploaded
{
    use Dispatchable, SerializesModels;

    public AcceptanceLetter $al;

    public function __construct(AcceptanceLetter $al)
    {
        $this->al = $al;
    }
}
```

Figure 25 event acceptanceletteruploaded

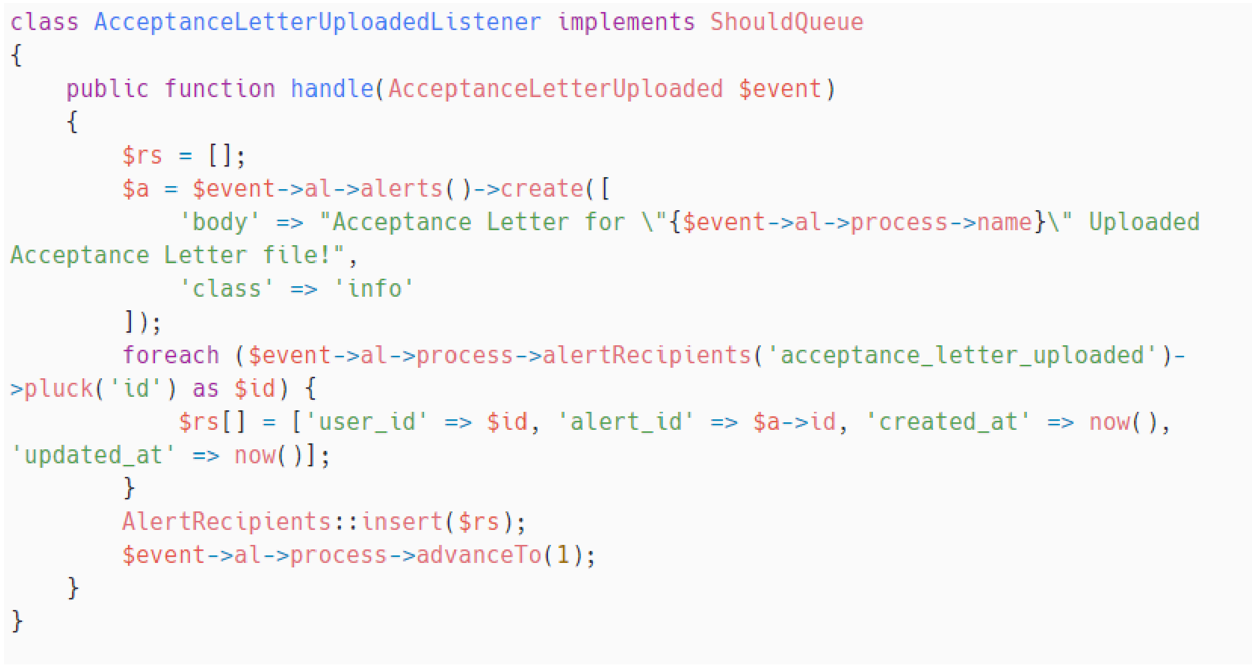
Every event triggers a specific listeners class (Figure 26 Events-Listeners relation). This is done thanks to the EventServiceProvider class of the illuminate library.



```php
use Illuminate\Foundation\Support\Providers\EventServiceProvider as ServiceProvider;
class EventServiceProvider extends ServiceProvider
{
    protected $listen = [
        'Illuminate\Auth\Events\Registered' => ['Illuminate\Auth\Listeners\SendEmailVerificationNotification'],
        'App\Events\ProcessCreated' => ['App\Listeners\ProcessCreatedListener'],
        'App\Events\AcceptanceLetterCreated' => ['App\Listeners\AcceptanceLetterCreatedListener'],
        'App\Events\AcceptanceLetterUploaded' => ['App\Listeners\AcceptanceLetterUploadedListener'],
        'App\Events\MoreInfoOnAcceptanceLetterUploaded' =>
['App\Listeners\MoreInfoOnAcceptanceLetterUploadedListener'],
        'App\Events\AcceptanceLetterApproved' => ['App\Listeners\AcceptanceLetterApprovedListener'],
        'App\Events\AcceptanceLetterApprovedWithReserve' =>
['App\Listeners\AcceptanceLetterApprovedWithReserveListener'],
        'App\Events\AcceptanceLetterTerminated' => ['App\Listeners\AcceptanceLetterTerminatedListener'],
        'App\Events\PriceQuotationCreated' => ['App\Listeners\PriceQuotationCreatedListener'],
        'App\Events\PriceQuotationUploaded' => ['App\Listeners\PriceQuotationUploadedListener'],
        'App\Events\PriceQuotationSignedUploaded' => ['App\Listeners\PriceQuotationSignedUploadedListener'],
        'App\Events\PriceQuotationApporoved' => ['App\Listeners\PriceQuotationApporovedListener'],
        'App\Events\PriceQuotationRejected' => ['App\Listeners\PriceQuotationRejectedListener'],
        'App\Events\PriceQuotationConfirmationCreated' =>
['App\Listeners\PriceQuotationConfirmationCreatedListener'],
        'App\Events\PriceQuotationConfirmationUploaded' =>
['App\Listeners\PriceQuotationConfirmationUploadedListener'],
        'App\Events\PriceQuotationConfirmationRecieved' =>
['App\Listeners\PriceQuotationConfirmationRecievedListener'],
        'App\Events\PriceQuotationConfirmationConfirmed' =>
['App\Listeners\PriceQuotationConfirmationConfirmedListener'],
    ];
}
```

Figure 26 Events-Listeners relation

As it is seen in Figure 26 Events-Listeners relation, the acceptanceletteruploaded event triggers Figure 27 AcceptanceLetterUploadedListener class.



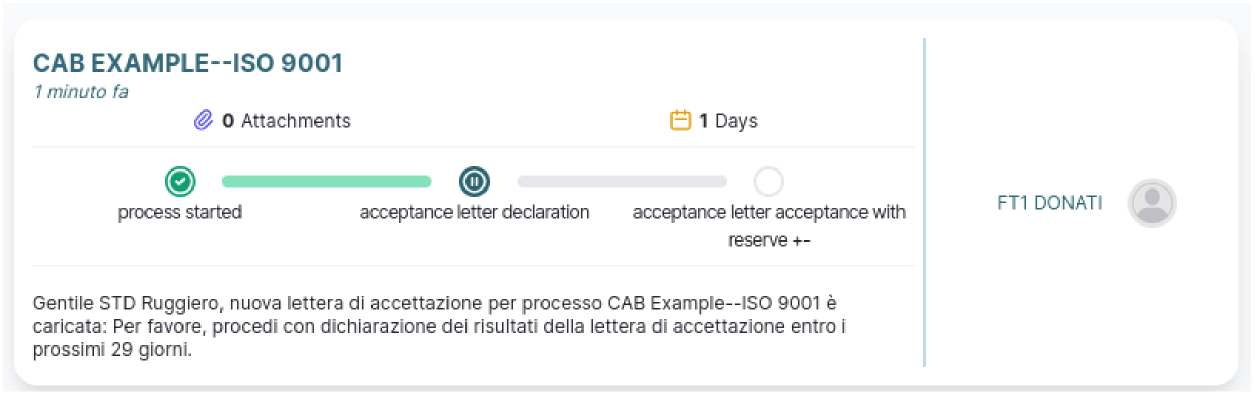
```php
class AcceptanceLetterUploadedListener implements ShouldQueue
{
    public function handle(AcceptanceLetterUploaded $event)
    {
        $rs = [];
        $a = $event->al->alerts()->create([
            'body' => "Acceptance Letter for \"{$event->al->process->name}\" Uploaded
Acceptance Letter file!",
            'class' => 'info'
        ]);
        foreach ($event->al->process->alertRecipients('acceptance_letter_uploaded')-
>pluck('id') as $id) {
            $rs[] = ['user_id' => $id, 'alert_id' => $a->id, 'created_at' => now(),
'updated_at' => now()];
        }
        AlertRecipients::insert($rs);
        $event->al->process->advanceTo(1);
    }
}
```

Figure 27 AcceptanceLetterUploadedListener class

Listener classes are responsible for the automatic notification system. They cause a pop-up message on the web app to STD, DDT, and the assigned FT.
Moreover, Figure 27 AcceptanceLetterUploadedListener class is responsible for advancing the process to the next checkpoint (checkpoint 1).

4.5 Checkpoints 2, 3, -2

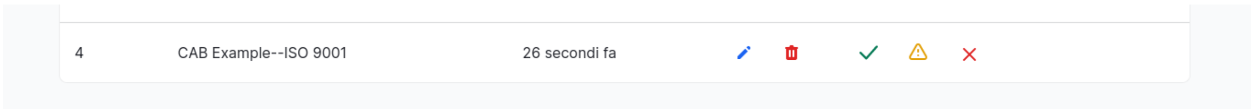
After the assessment of DA is uploaded and the first checkpoint is reached, the web app will ask STD to declare the assessment's outcome (in accordance with checkpoint1, Figure 13 to-do messages function). The message is available in Figure 28 Dashboard panel of STD.



Gentile STD Ruggiero, nuova lettera di accettazione per processo CAB Example--ISO 9001 è caricata: Per favore, procedi con dichiarazione dei risultati della lettera di accettazione entro i prossimi 29 giorni.

Figure 28 Dashboard panel of STD

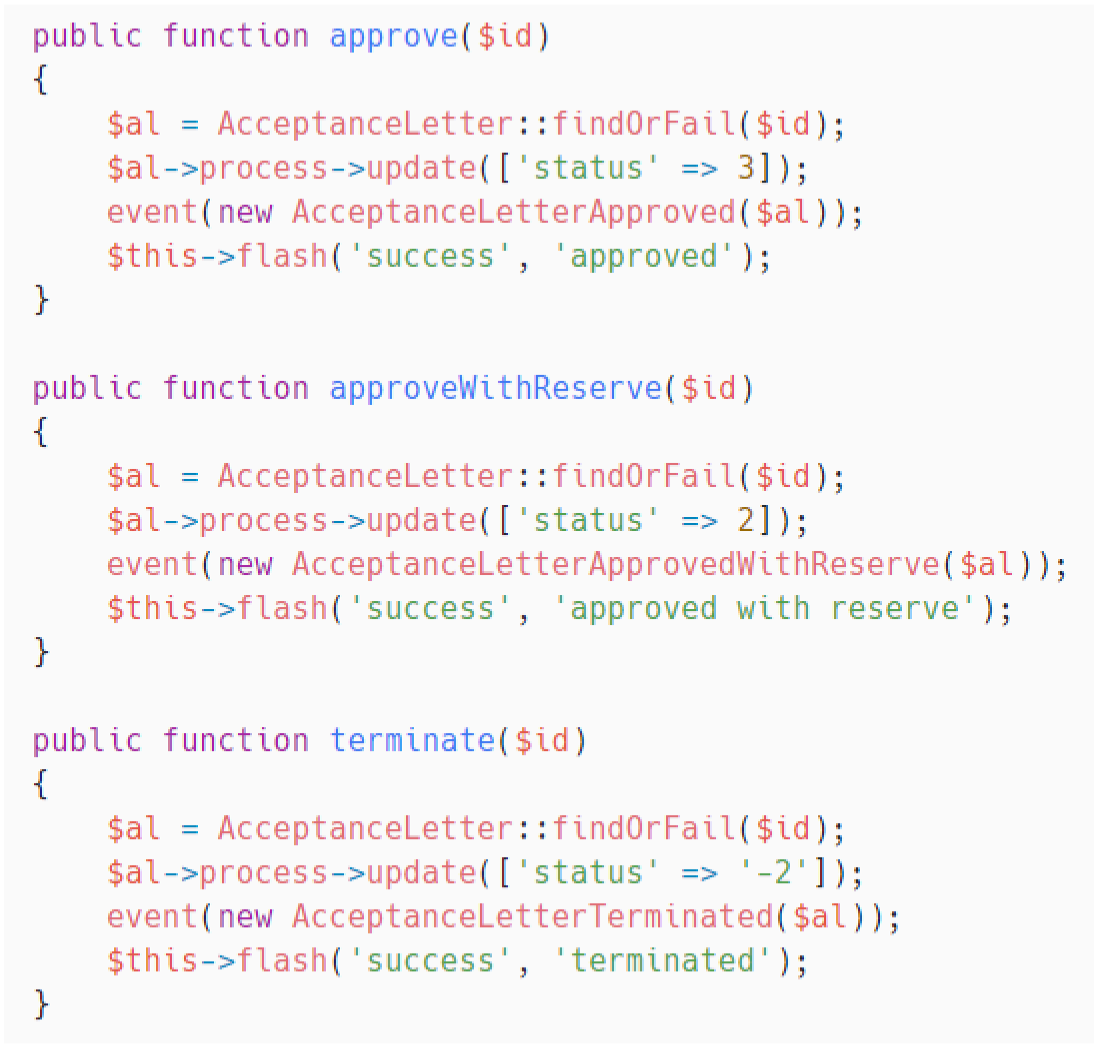
The screenshot of the interface available for STD to declare the status of the response to DA is presented in Figure 29 Acceptance letter panel of STD.



4	CAB Example--ISO 9001	26 secondi fa	✎	🗑	✓	⚠	✕

Figure 29 Acceptance letter panel of STD

The functions for approving (the green tick button Figure 29), approving with reserve (the yellow triangle sign Figure 29), and terminate (the red cross Figure 29) the process are presented in Figure 30 Approve/acceptance, acceptance with reserve, termination/rejection functions. The functions reported in Figure 30 are programmed within the acceptance letter class of the livewire library of the controller component.



```
public function approve($id)
{
    $al = AcceptanceLetter::findOrFail($id);
    $al->process->update(['status' => 3]);
    event(new AcceptanceLetterApproved($al));
    $this->flash('success', 'approved');
}

public function approveWithReserve($id)
{
    $al = AcceptanceLetter::findOrFail($id);
    $al->process->update(['status' => 2]);
    event(new AcceptanceLetterApprovedWithReserve($al));
    $this->flash('success', 'approved with reserve');
}

public function terminate($id)
{
    $al = AcceptanceLetter::findOrFail($id);
    $al->process->update(['status' => '-2']);
    event(new AcceptanceLetterTerminated($al));
    $this->flash('success', 'terminated');
}
```

Figure 30 Approve/acceptance, acceptance with reserve, termination/rejection functions

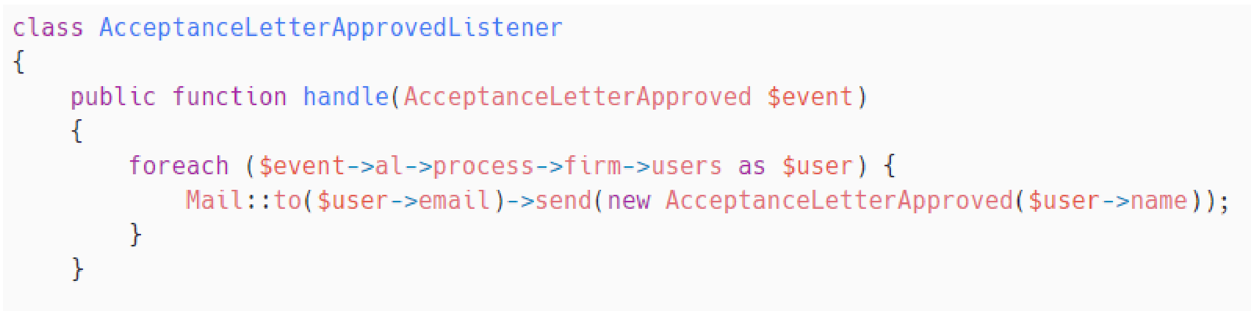
Functions approve and approveWithReserve proceeds to checkpoints 3 and 2 respectively. Both checkpoints 3 and 2 have the same to-do message for the assigned FT to process. (Figure 13 to-do messages function and Figure 14 to-do roles function).

After FT prepares and uploads Scheda Preventivo, checkpoints 2 and 3 will both advance to checkpoint 4, however, there is a need to do a set of actions in checkpoint 2 to check if CAB has procured the missing or miss-compiled documents within the deadline. There are two conditions essential for this point.

First there is the event-listener logic, in which uploading of files to the anticipated folder (integrazioni alla domanda folder inside the acceptance letter folder inside the process's folder) on Nextcloud will be detected (using 4.6.3 parse function) and then an event will be fired which will trigger a specific listener class. The listener class will informs the web app and its users, in particular STD, of the events that has happened.

But what if CAB doesn't deliver the missing or miss compiled document until the deadline? Here there's the need for a job to check the specific folder and report if no modification were made until the deadline, a warning should depart to CAB (automatic sending of the warning message is not implemented, however, there's a job class developed (section 4.6.2) which runs every minuet and report back what has happened on Nextcloud).

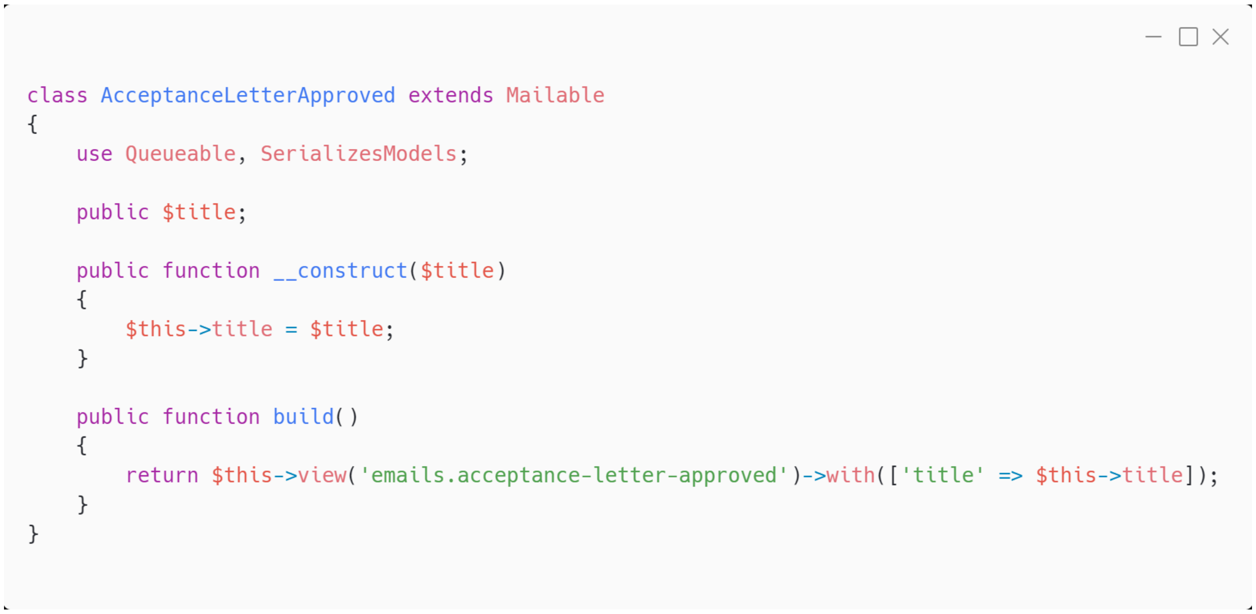
Each of the three functions reported in Figure 30 Approve/acceptance, acceptance with reserve, termination/rejection functions fire an event. These evets can be traced using Figure 26 Events-Listeners relation to their relative listener class. Apart from automatic pop up notification, the listener classes linked to the events that were fired by Figure 30 Approve/acceptance, acceptance with reserve, termination/rejection functions send an email to CAB (the email address of the user who is assigned to company that is involved in ongoing accreditation process). Let's take a look at Figure 31 AcceptanceLetterApprovedListener class which is triggered by STD clicking on the green tick button in Figure 29 Acceptance letter panel of STD (Figure 30 Approve/acceptance, acceptance with reserve, termination/rejection functions fires AcceptanceLetterApproved event, and this event triggers Figure 31 AcceptanceLetterApprovedListener class).



```
class AcceptanceLetterApprovedListener
{
    public function handle(AcceptanceLetterApproved $event)
    {
        foreach ($event->al->process->firm->users as $user) {
            Mail::to($user->email)->send(new AcceptanceLetterApproved($user->name));
        }
    }
```

Figure 31 AcceptanceLetterApprovedListener class

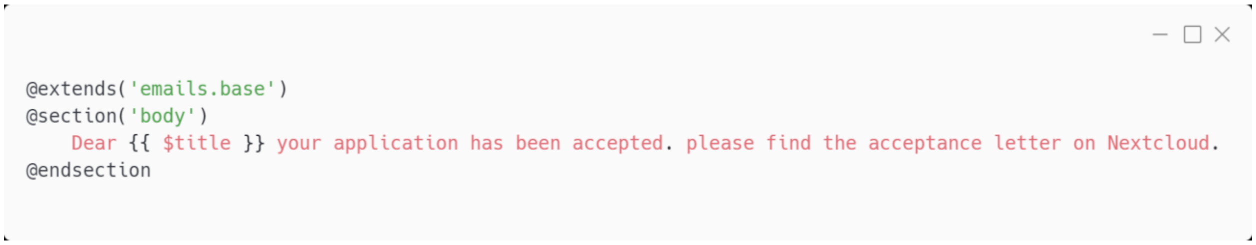
Figure 31 AcceptanceLetterApprovedListener class passes the event's title to Figure 32 AcceptanceLetterApproved class and the recipient user's email address to mail facade.



```php
class AcceptanceLetterApproved extends Mailable
{
    use Queueable, SerializesModels;

    public $title;

    public function __construct($title)
    {
        $this->title = $title;
    }

    public function build()
    {
        return $this->view('emails.acceptance-letter-approved')->with(['title' => $this->title]);
    }
}
```

Figure 32 AcceptanceLetterApproved class

The build function of Figure 32 AcceptanceLetterApproved class uses the view method to send the anticipated message. (Figure 33 AcceptanceLetterApproved email view)

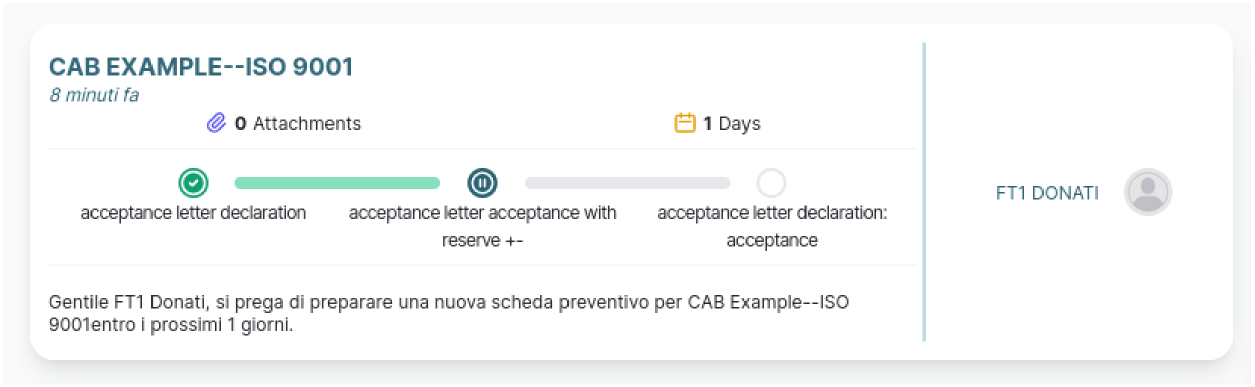


```
@extends('emails.base')
@section('body')
    Dear {{ $title }} your application has been accepted. please find the acceptance letter on Nextcloud.
@endsection
```

Figure 33AcceptanceLetterApproved email view

4.6 Checkpoint 4

Uploading Scheda preventio or the internal price quotation file.



CAB EXAMPLE--ISO 9001
8 minuti fa

0 Attachments   **1** Days

acceptance letter declaration — acceptance letter acceptance with reserve +- — acceptance letter declaration: acceptance

Gentile FT1 Donati, si prega di preparare una nuova scheda preventivo per CAB Example--ISO 9001entro i prossimi 1 giorni.

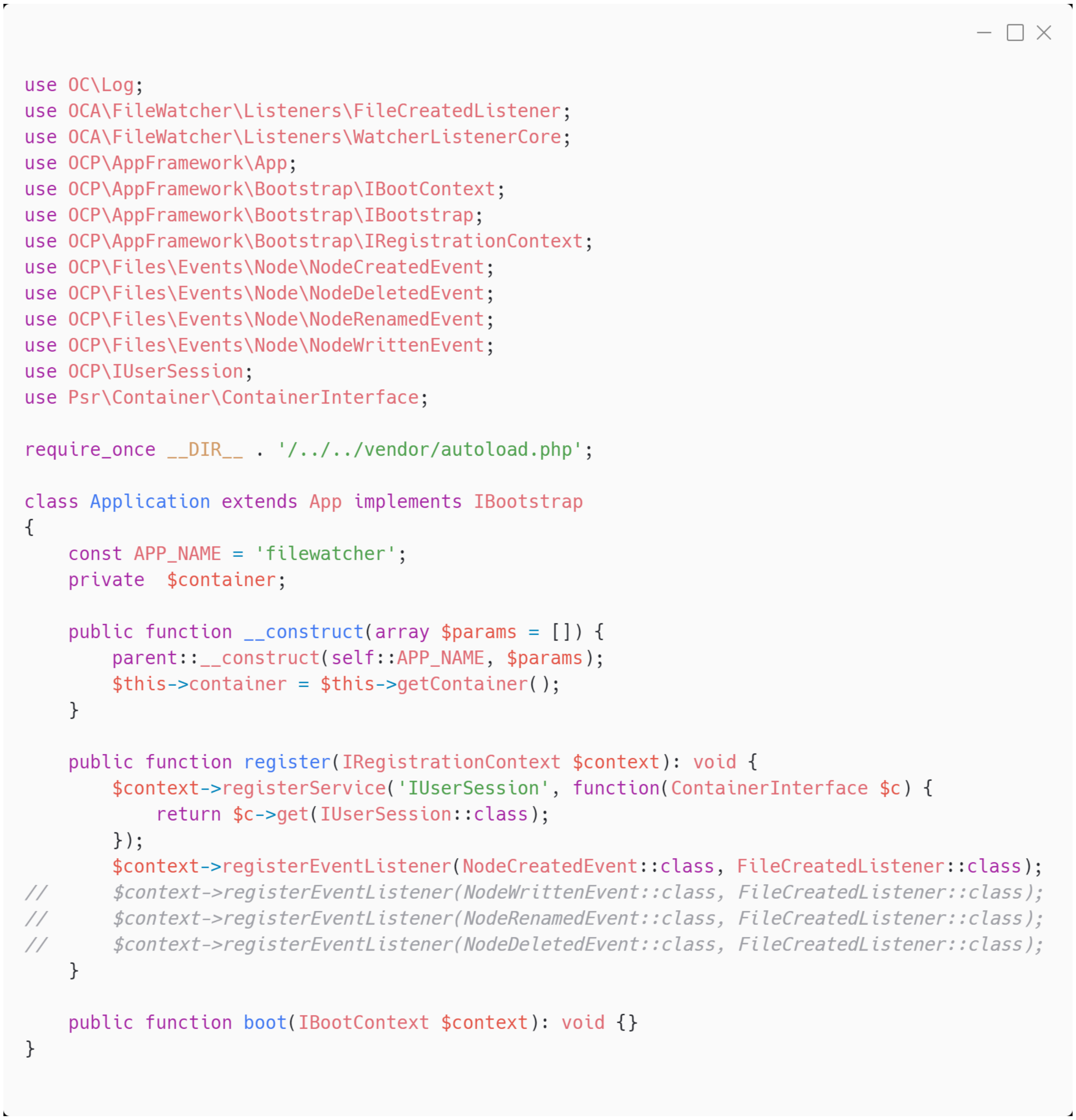
FT1 DONATI

Figure 34 Dashboard panel of FT

According to Figure 13 to-do messages function and Figure 14 to-do roles function, checkpoints 2 and 3 (acceptance and acceptance with reserve) web app sends a message to the assigned FT of the process to continue with preparing Scheda Preventivo (Figure 34 Dashboard panel of FT).
Let’s discuss uploading Scheda Preventivo by using Nextcloud. FT proceeds to the process’s folder on Nextcloud and uploads Scheda Preventivo document to the 01_scheda_preventivo folder. There are two classes in the web app (4.6.2 and 4.6.3) to be able to update the process when a file is uploaded using Nextcloud. But first File watcher app (4.6.1) registers the event that has happened on Nextcloud and publishes it on the Redis server.

4.6.1 File watcher application

Nextcloud’s built-in functions and APIs, particularly the elements of OCP API, are orchestrated in developing a new plug-in for Nextcloud. This app has no user interface and works in the background on Nextcloud.
Figure 35 Application class of File-watcher app registers creation of a file on Nextcloud. It’s also possible to register other events, such as deleting or renaming of a file, but for the scope of this project only the events that refer to file creation are considered. These events are registered in the NodeCreatedEvent class.



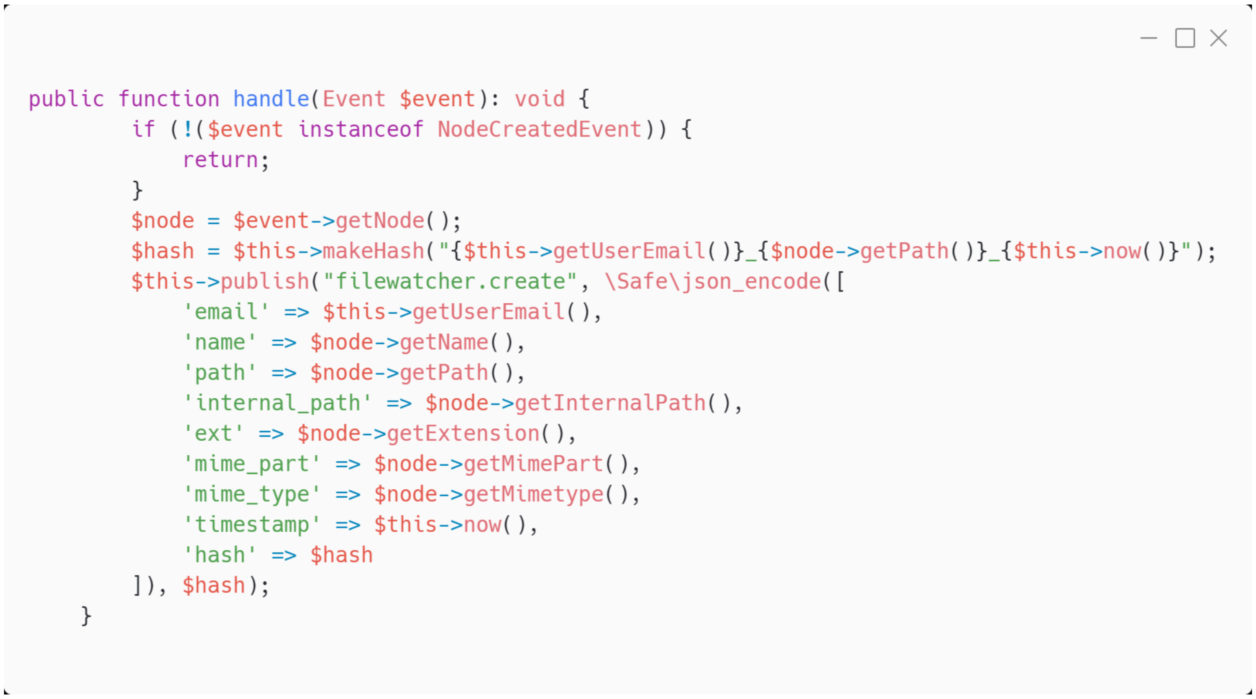
```php
use OC\Log;
use OCA\FileWatcher\Listeners\FileCreatedListener;
use OCA\FileWatcher\Listeners\WatcherListenerCore;
use OCP\AppFramework\App;
use OCP\AppFramework\Bootstrap\IBootContext;
use OCP\AppFramework\Bootstrap\IBootstrap;
use OCP\AppFramework\Bootstrap\IRegistrationContext;
use OCP\Files\Events\Node\NodeCreatedEvent;
use OCP\Files\Events\Node\NodeDeletedEvent;
use OCP\Files\Events\Node\NodeRenamedEvent;
use OCP\Files\Events\Node\NodeWrittenEvent;
use OCP\IUserSession;
use Psr\Container\ContainerInterface;

require_once __DIR__ . '/../../vendor/autoload.php';

class Application extends App implements IBootstrap
{
    const APP_NAME = 'filewatcher';
    private  $container;

    public function __construct(array $params = []) {
        parent::__construct(self::APP_NAME, $params);
        $this->container = $this->getContainer();
    }

    public function register(IRegistrationContext $context): void {
        $context->registerService('IUserSession', function(ContainerInterface $c) {
            return $c->get(IUserSession::class);
        });
        $context->registerEventListener(NodeCreatedEvent::class, FileCreatedListener::class);
//      $context->registerEventListener(NodeWrittenEvent::class, FileCreatedListener::class);
//      $context->registerEventListener(NodeRenamedEvent::class, FileCreatedListener::class);
//      $context->registerEventListener(NodeDeletedEvent::class, FileCreatedListener::class);
    }

    public function boot(IBootContext $context): void {}
}
```

Figure 35 Application class of File-watcher app

Figure 36 handle function of FileCreatedListener class gets the events information registered on NodeCretedEvent (by Figure 35 Application class of File-watcher app) and extends WatcherListenerCore class. Figure 36 handle function of FileCreatedListener class serializes the events information into a json format and creates a hash to ensure uniqueness of the batch, then publishes it on the Redis server using Figure 37 publish function of WatcherListenerCore class.

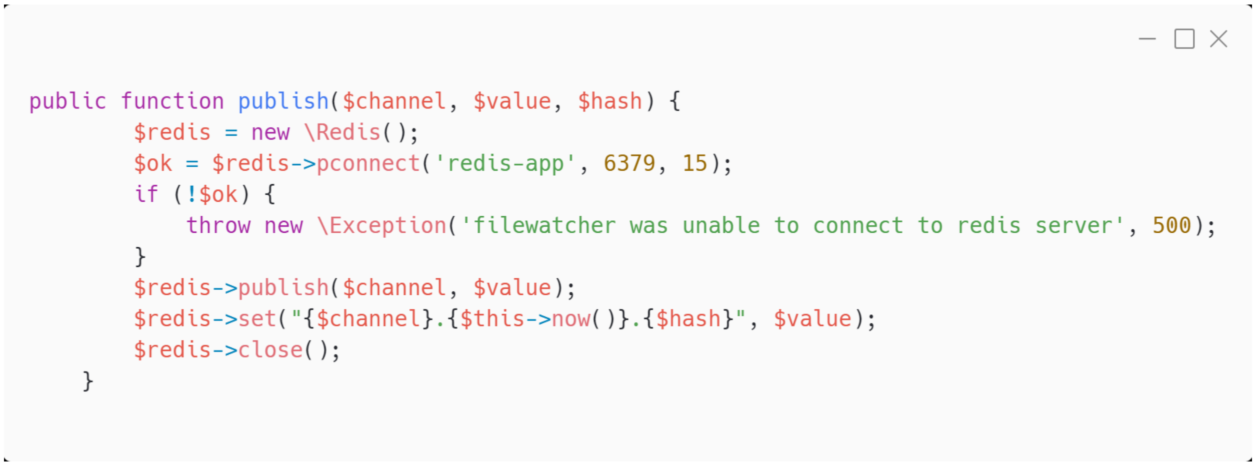
Moreover, information regarding the author of the event (User name, Email address) are not registered by Figure 35 Application class of File-watcher app alone.

Figure 38 getUser and getUserEmail function of WatcherListenerCore class are used in Figure 36 handle function of FileCreatedListener class to get and publish the aforementioned not registered information.



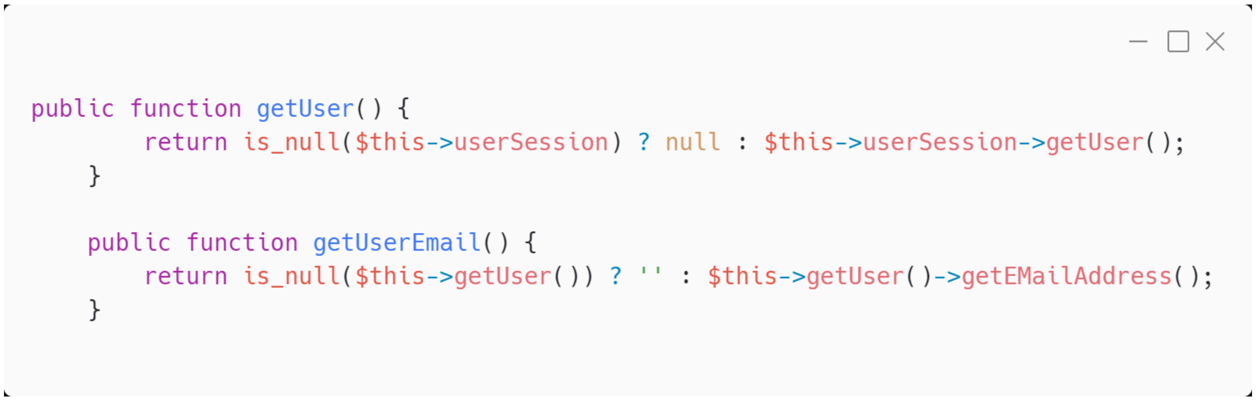
```
public function handle(Event $event): void {
        if (!($event instanceof NodeCreatedEvent)) {
            return;
        }
        $node = $event->getNode();
        $hash = $this->makeHash("{$this->getUserEmail()}_{$node->getPath()}_{$this->now()}");
        $this->publish("filewatcher.create", \Safe\json_encode([
            'email' => $this->getUserEmail(),
            'name' => $node->getName(),
            'path' => $node->getPath(),
            'internal_path' => $node->getInternalPath(),
            'ext' => $node->getExtension(),
            'mime_part' => $node->getMimePart(),
            'mime_type' => $node->getMimetype(),
            'timestamp' => $this->now(),
            'hash' => $hash
        ]), $hash);
    }
```

Figure 36 handle function of FileCreatedListener class



```
public function publish($channel, $value, $hash) {
        $redis = new \Redis();
        $ok = $redis->pconnect('redis-app', 6379, 15);
        if (!$ok) {
            throw new \Exception('filewatcher was unable to connect to redis server', 500);
        }
        $redis->publish($channel, $value);
        $redis->set("{$channel}.{$this->now()}.{$hash}", $value);
        $redis->close();
    }
```

Figure 37 publish function of WatcherListenerCore class



```php
public function getUser() {
        return is_null($this->userSession) ? null : $this->userSession->getUser();
    }

    public function getUserEmail() {
        return is_null($this->getUser()) ? '' : $this->getUser()->getEMailAddress();
    }
```

Figure 38 getUser and getUserEmail function of WatcherListenerCore class

4.6.2 FileWatcherTrigger Class

This class is represented with all of its dependencies in Figure 39 FileWatcherTrigger class. This class is programmed to do the job of connecting to the Redis server every minute and gets the report of the events that have happened on Nextcloud and convert it to json format and pass the data to the FileWatcherParser class (4.6.3). In the end, the successfully processed data is deleted from the database to avoid future unwanted reprocesses.

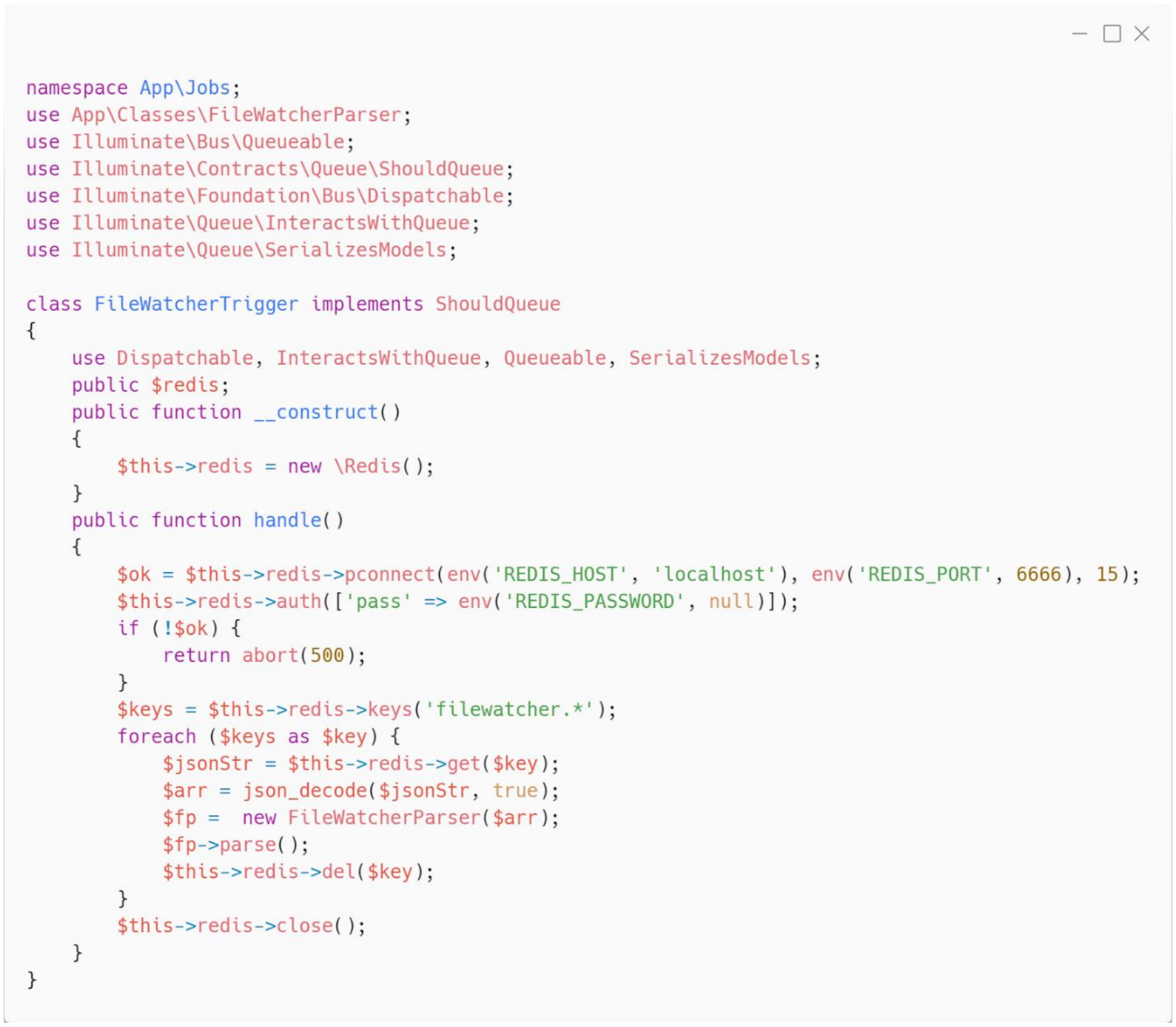
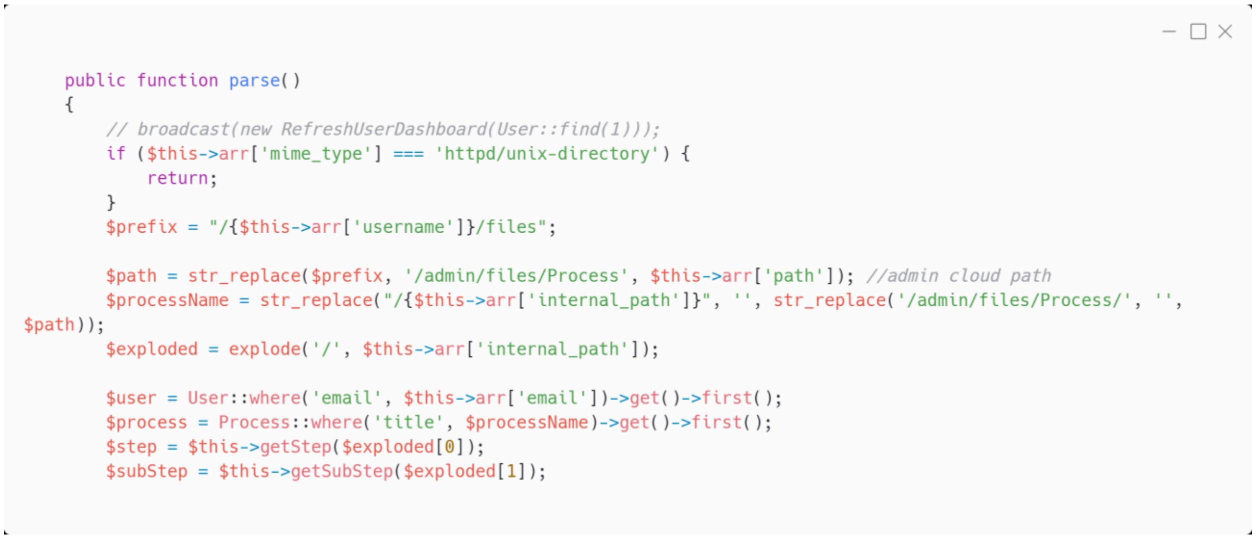

```php
namespace App\Jobs;
use App\Classes\FileWatcherParser;
use Illuminate\Bus\Queueable;
use Illuminate\Contracts\Queue\ShouldQueue;
use Illuminate\Foundation\Bus\Dispatchable;
use Illuminate\Queue\InteractsWithQueue;
use Illuminate\Queue\SerializesModels;

class FileWatcherTrigger implements ShouldQueue
{
    use Dispatchable, InteractsWithQueue, Queueable, SerializesModels;
    public $redis;
    public function __construct()
    {
        $this->redis = new \Redis();
    }
    public function handle()
    {
        $ok = $this->redis->pconnect(env('REDIS_HOST', 'localhost'), env('REDIS_PORT', 6666), 15);
        $this->redis->auth(['pass' => env('REDIS_PASSWORD', null)]);
        if (!$ok) {
            return abort(500);
        }
        $keys = $this->redis->keys('filewatcher.*');
        foreach ($keys as $key) {
            $jsonStr = $this->redis->get($key);
            $arr = json_decode($jsonStr, true);
            $fp =  new FileWatcherParser($arr);
            $fp->parse();
            $this->redis->del($key);
        }
        $this->redis->close();
    }
}
```

Figure 39 FileWatcherTrigger class

4.6.3 FileWatcherParser class

Figure 40 Parse function of FileWatcherParser class takes the directory path of the file that was created and constructs a variable array ($exploded). The first element of such array refers to the name of the folder that the new file was uploaded to. It's needed to mention that the first element of variable exploded is going to be one of the four folder names mentioned in section 2.7, moreover, there might be folders inside the four folders inside the process's folder.

For example, a folder named "integration of the application" inside the folder 00_acceptance_letter can be considered for storing the files related to acceptance with reserve or even rejection cases. The second array of variable exploded refers to the names of the folders located within this layer of process's folder structure.

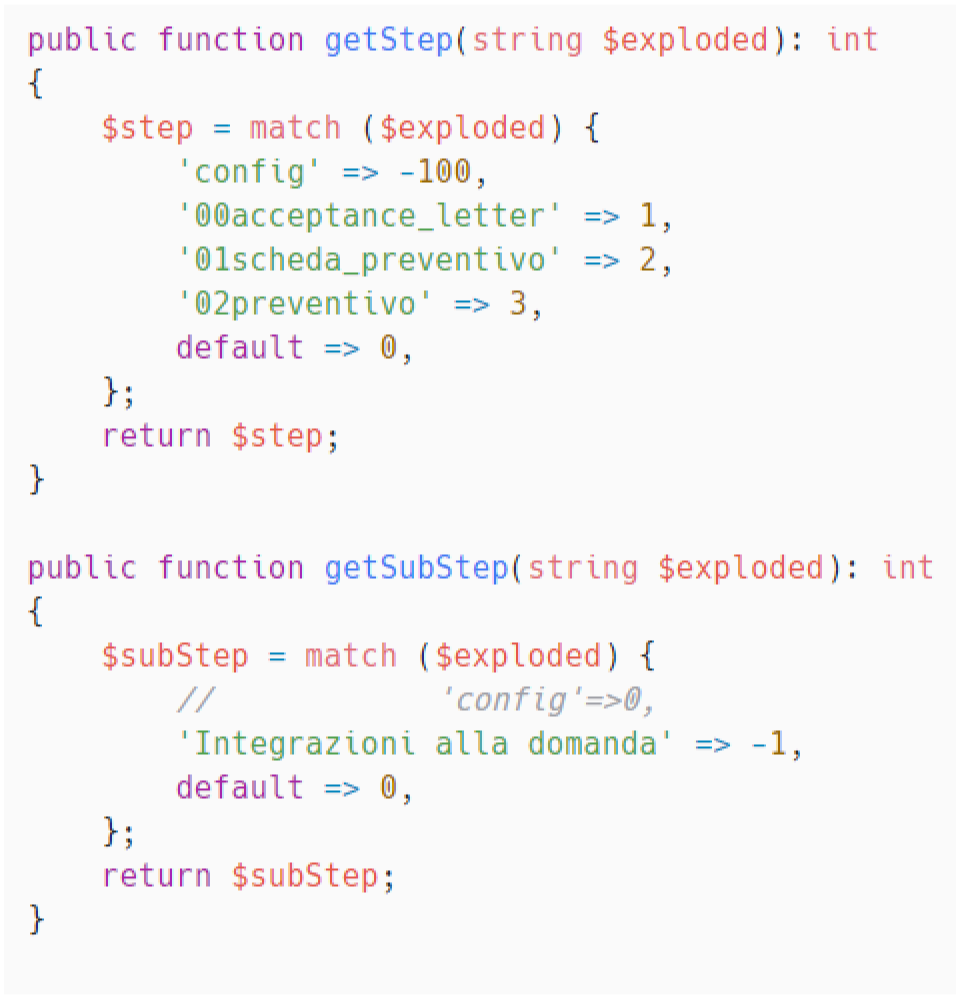


```
    public function parse()
    {
        // broadcast(new RefreshUserDashboard(User::find(1)));
        if ($this->arr['mime_type'] === 'httpd/unix-directory') {
            return;
        }
        $prefix = "/{$this->arr['username']}/files";

        $path = str_replace($prefix, '/admin/files/Process', $this->arr['path']); //admin cloud path
        $processName = str_replace("/{$this->arr['internal_path']}", '', str_replace('/admin/files/Process/', '',
$path));
        $exploded = explode('/', $this->arr['internal_path']);

        $user = User::where('email', $this->arr['email'])->get()->first();
        $process = Process::where('title', $processName)->get()->first();
        $step = $this->getStep($exploded[0]);
        $subStep = $this->getSubStep($exploded[1]);
```

Figure 40 Parse function of FileWatcherParser class

Please note that Figure 40 Parse function of FileWatcherParser is not concluded is going to be discussed further. Let’s take look at the last couple of lines of Figure 40 Parse function of FileWatcherParser.

Figure 41 getStep and getSubStep functions of FileWatcherParser class takes the first and second element of the variable exploded respectively, and matches them with the anticipated folder names and returns variables step and substep respectively. The variable step and sub-step are recalled by the if-clauses that are programmed in the rest of Figure 40 Parse function of FileWatcherParser class.

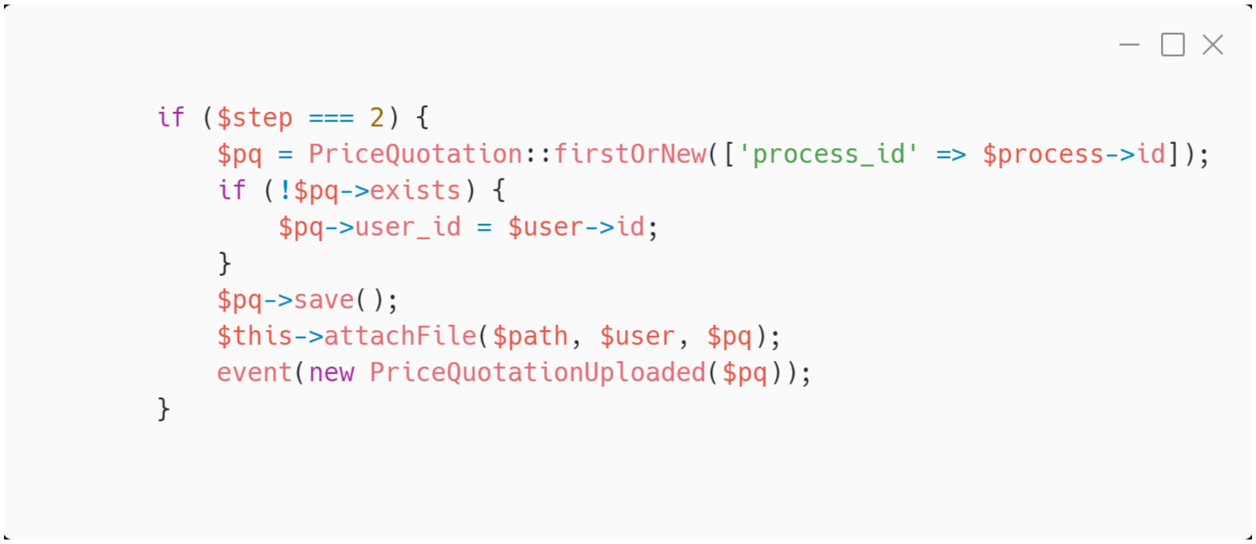


```php
public function getStep(string $exploded): int
{
    $step = match ($exploded) {
        'config' => -100,
        '00acceptance_letter' => 1,
        '01scheda_preventivo' => 2,
        '02preventivo' => 3,
        default => 0,
    };
    return $step;
}

public function getSubStep(string $exploded): int
{
    $subStep = match ($exploded) {
        //              'config'=>0,
        'Integrazioni alla domanda' => -1,
        default => 0,
    };
    return $subStep;
}
```

Figure 41 getStep and getSubStep functions of FileWatcherParser class

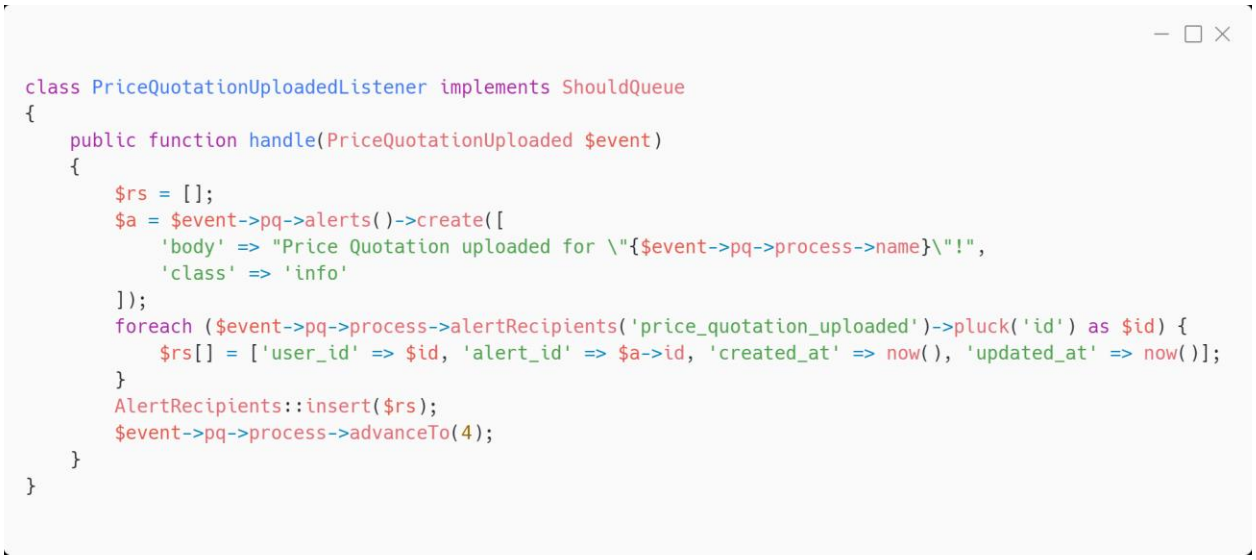
In the case of Scheda Preventivo, the variable step is set to 2. Figure 42 if-clause instance of the Parse function of FileWatcherParser class (Figure 40 Parse function of FileWatcherParser class has the right conditions to process. The uploaded file is indexed in the web app, then event PriceQuotationUploaded is fired. This event triggers Figure 43 PriceQuotationUploadedListener class.

Please note that "priceQuotation" used in the text of the code refers to Scheda preventivo.



```php
if ($step === 2) {
    $pq = PriceQuotation::firstOrNew(['process_id' => $process->id]);
    if (!$pq->exists) {
        $pq->user_id = $user->id;
    }
    $pq->save();
    $this->attachFile($path, $user, $pq);
    event(new PriceQuotationUploaded($pq));
}
```

Figure 42 if-clause instance of the Parse function of FileWatcherParser class (Figure 40 Parse function of FileWatcherParser class



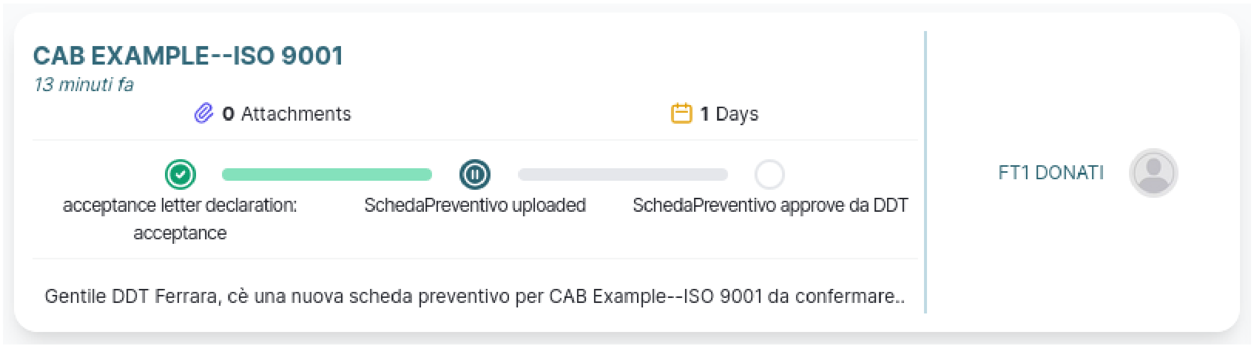
```php
class PriceQuotationUploadedListener implements ShouldQueue
{
    public function handle(PriceQuotationUploaded $event)
    {
        $rs = [];
        $a = $event->pq->alerts()->create([
            'body' => "Price Quotation uploaded for \"{$event->pq->process->name}\"!",
            'class' => 'info'
        ]);
        foreach ($event->pq->process->alertRecipients('price_quotation_uploaded')->pluck('id') as $id) {
            $rs[] = ['user_id' => $id, 'alert_id' => $a->id, 'created_at' => now(), 'updated_at' => now()];
        }
        AlertRecipients::insert($rs);
        $event->pq->process->advanceTo(4);
    }
}
```

Figure 43 PriceQuotationUploadedListener class

Figure 43 PriceQuotationUploadedListener class alerts recipients with pop up messages and pushes the process forward to the checkpoint 4.

4.7 Checkpoints 5, -5

Here DDT checks the Scheda preventivo and approves its righteousness. Figure 44 Dashboard panel of DDT guides DDT to Figure 45 Scheda Preventivo panel of DDT in order to approve or reject the Scheda preventive.



CAB EXAMPLE--ISO 9001

13 minuti fa

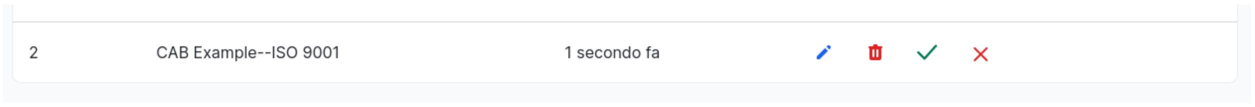
0 Attachments — 1 Days

acceptance letter declaration: acceptance — SchedaPreventivo uploaded — SchedaPreventivo approve da DDT

FT1 DONATI

Gentile DDT Ferrara, cè una nuova scheda preventivo per CAB Example--ISO 9001 da confermare..

Figure 44 Dashboard panel of DDT

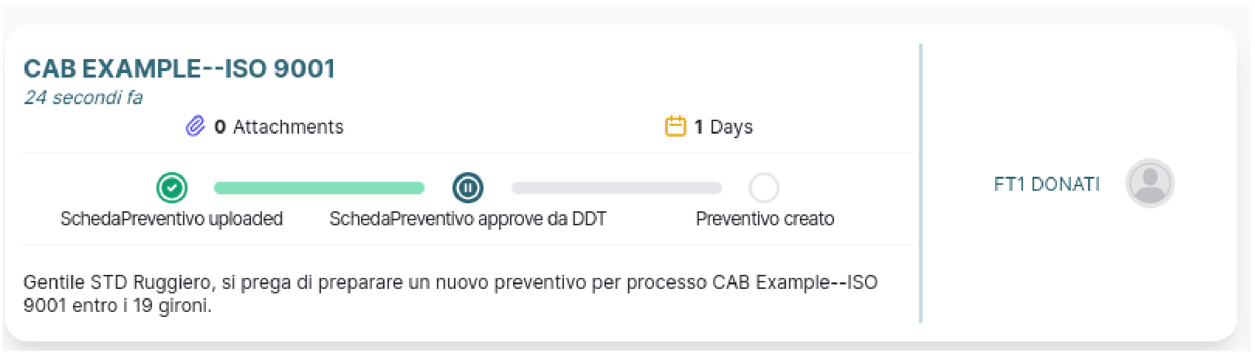


2	CAB Example--ISO 9001	1 secondo fa

Figure 45 Scheda Preventivo panel of DDT

4.8 Checkpoint 6

If DDT approves the Scheda Prevetivo, the process proceeds forward and a new to-do message will be available for STD, Figure 46 Dashboard panel of STD, telling him/her to continue with preparing a new Preventivo for the on-going process.



CAB EXAMPLE--ISO 9001

24 secondi fa

0 Attachments — 1 Days

SchedaPreventivo uploaded — SchedaPreventivo approve da DDT — Preventivo creato

FT1 DONATI

Gentile STD Ruggiero, si prega di preparare un nuovo preventivo per processo CAB Example--ISO 9001 entro i 19 gironi.

Figure 46 Dashboard panel of STD

4.9 Analyzing the new infrastructure and ecosystem

In the last couple of sections, the back-end related discussions were omitted, just for the sake of making the thesis more reader friendly, as all of the desired functionalities (process structure, deadlines, to-do messages, decision making, uploading files using the web app, uploading files using Nextcloud, automatic notifications and emails) already seem to be achieved and easily replicable.

The rest of the checkpoints are not discussed, instead let's analyze the job already done.

4.9.1 CABs representatives’ Nextcloud accounts

By following this chapter, one can question it to miss out discussing managing CABs representatives’ Nextcloud accounts. This is due to the fact that Nextcloud doesn’t have an external API by default according to its documentation to be used to create new accounts from outside of Nextcloud environment. In other words, the web app would work if there’s an account created on Nextcloud before creating the account with the same email address on the web app.

Although it might look like a minimalist approach, on the contrary, it can be a cost-effective approach since the CABs representatives are defined and limited.

Another solution can be developing yet another plug-in for Nextcloud to take care of creating new account on demand of the web app. This solution of course demands more effort.

4.9.2 Inspectors’ Nextcloud accounts

Unlike assigning FT to the process which is done through the web app, an empty but templated configuration excel file is going to be available on Nextcloud to assign inspectors to a process. This approach was suggested by engineer Pedone, and the development team seeks to implement this functionality very smoothly in particular, as this is the gateway to the next steps of the process. The template of such configuration file is provided by Engineer Pedone and covers a wide range of information columns and that can be seen as the reasoning behind the choice of using an excel configuration file over using the web app. It would be tedious to both develop and use numerous modals to give and get each piece of information over the web app.

The same situation of section 4.9.1 (for Nextcloud accounts) applies to assigning inspectors/assessors. Their number is even more restrict than number of CABs and thus favors the cost-effective approach mentioned in section 4.9.1.

5 Conclusion

The proposed solution proposed by this thesis seems to be able to help over 200 CABs and over and 80 assessors/inspectors, as well as calibration laboratories department staff in performing the accreditation process if more effort is put, in particular, to management of CABs and inspectors/assessors. Gladly, for the time being, the web page application looks promising and aligned with Accredia's demands.

After working on this project for several months, the development team suggests it would be easier (in technical terms) to manage the CABs using the web app over using Nextcloud. CABs are responsible for providing documents and the web app is responsible for delivering documents to CABs. The use case of CABs and assessors/inspectors are not the same, meaning that assessors/inspectors should have access to Nextcloud for facilitating the accreditation process (for example online collaboration on Nextcloud), however, CABs can still be managed without using Nextcloud and using the web app alone without any major drawbacks. Moreover, by using the web app to manage the CABs, CABs will have access to the web app's user interface which can to make the communications easier and more transparent.

The decision for user categorization (direct vs indirect) and to use Nextcloud to manage CABs in the process were taken by Accredia. The reasoning behind this decision for CABs can be lack of awareness of the web app (as at the time when the decision was made there was no web app developed, the web app has not yet proven to be robust, not all of the people involved in the decision making are in direct contact with the development team).

6 References

1	https://www.iaf.nu/
2	https://ilac.org/
3	https://european-accreditation.org/
4	https://www.accredia.it/
5	Rubin, Kenneth S. *Essential Scrum: A practical guide to the most popular Agile process*. Addison-Wesley, 2012.
6	Srivastava, Apoorva, Sukriti Bhardwaj, and Shipra Saraswat. "SCRUM model for agile methodology." 2017 International Conference on Computing, Communication and Automation (ICCCA). IEEE, 2017.
7	https://nextcloud.com/
8	https://w3techs.com/technologies/overview/programming_language
9	https://www.wappalyzer.com/technologies/programming-languages
10	https://kinsta.com/blog/php-frameworks/
11	https://raygun.com/blog/top-php-frameworks/
12	https://upload.wikimedia.org/wikipedia/commons/a/a0/MVC-Process.svg/
13	https://github.com/laravel/framework
14	https://packagist.org/packages/laravel/framework
15	https://belitsoft.com/laravel-development-services/10-benefits-using-laravel-php-framework
16	Stauffer, Matt. Laravel: Up & running: A framework for building modern php apps. O'Reilly Media, 2019.
17	https://github.com/illuminate/database
18	https://laravel.com/docs/8.x/eloquent
19	https://laravel.com/docs/8.x/blade
20	https://laravel-livewire.com/
21	Rossi, Fabiana, Matteo Nardelli, and Valeria Cardellini. "Horizontal and vertical scaling of container-based applications using reinforcement learning." *2019 IEEE 12th International Conference on Cloud Computing (CLOUD)*. IEEE, 2019.
22	https://www.docker.com/
23	https://redis.io/

All of the websites mentioned in the references section were last visited within June/July 2021.