[ApS] Creation of the corporate website for the 17th of May association

Óscar Pina Gracia

Resumen– El presente proyecto se ha llevado a cabo mediante la modalidad de aprendizaje por servicio (ApS), que pretende que los estudiantes se formen ayudando a solucionar un problema real propuesto por una empresa o entidad. En este caso, la asociación 17 de mayo ha solicitado la creación de una página web corporativa desde cero. Esta asociación se centra en ayudar a resolver diferentes problemas de la comunidad LGTBI, así como a promocionar diferentes actividades relacionadas con este colectivo para darle visibilidad y concienciar a la sociedad sobre su situación. Este proyecto pretende realizar un proceso de ingeniería del software completo desde la captura de requisitos, pasando por el diseño, el testing y la puesta en marcha de dicha web para así facilitar el trabajo de la asociación.

Palabras clave– Web, LGTBI, PHP, MySQL, HTML, Bootstrap5, PHPMailer, Actividades, Entidades, Noticias, Usuarios.

Abstract– The present project has been carried out through the *Aprenentatge per Servei* (ApS) modality, which aims for students to train by helping to solve a real problem proposed by a company or entity. In this case, the 17th of May association has requested the creation of a corporate website from scratch. This association focuses on helping to solve different problems within the LGBTQ+ community, as well as promoting different activities related to this group to give visibility and raise awareness in society about their situation. This project aims to carry out a complete software engineering process, starting from requirements gathering, going through design, testing and launching of the website, in order to facilitate the work of the association.

Keywords– Web, LGTBQ+, PHP, MySQL, HTML, Bootstrap5, PHPMailer, Activities, Entities, News, Users.

1 INTRODUCTION - CONTEXT OF THE PROJECT

THE project has been carried out using the *Aprenentatge per Servei* modality, whose goal is for students to learn and train by helping to solve a real problem for a company or entity. With this premise, the 17th of May association requested the creation of its own corporate website. This association's main task is to help members of the LGBTQ+ community by providing them with information and support, and at the same time promoting the different events and activities held in Catalonia. With the idea of facilitating this promotion work for the association, a software engineering process will be established to build the association's website from scratch, which must contain spaces for the activities proposed by the various LGBTQ+ entities in Catalonia, as well as relevant news for the community and the promotion of other entities to make their work known to more people.

• Contact e-mail: 1560789@uab.cat
• Student's specialisation: Software Engineering
• Work supervised by: Alicia Fornés (Computer Science)
• Course: 2022/23

2 OBJECTIVES

The main and ultimate objective of the project is the creation of the website for the association, and to achieve this, following a software engineering process, the following objectives will be required:

- Conduct a study of the different web technologies that exist today and evaluate which could be useful for the project.
- Participate in the taking and writing of the website requirements.

- Analyse the different development methodologies that exist and select the most appropriate for the type of project to be carried out.
- Create a flexible and yet realistic plan to develop the project.
- Design and implement the website using the MVC paradigm.
- Create a database to store all the information related to the website.
- Test the website by carrying out User Acceptance Testing so that end users with different computer knowledge can ensure that a user-friendly and error-free interface is achieved.

3 STATE OF THE ART

There are numerous corporate websites belonging to different LGTBQ+ associations and entities. However, this project also intends that the web serve to have in a single place the information related to all the acts and activities that take place in Catalonia regarding LGTBQ+ matters.
Related to this, there is a website called *Quedeque* [1], belonging to the City Council of Barcelona, which serves the purpose of showing activities that take place in the city, as well as information about the different associations located there. The idea of the Quedeque website is similar to what is wanted to be achieved in this project, but with the main difference that the website of the 17th of May association will also extend this to activities and entities outside the scope of the city of Barcelona and reach the rest of the towns in Catalonia.

4 METHODOLOGY

The selected development methodology and the reasons for choosing it will be defined below, as well as the different technologies that have been decided to be used in this project will also be detailed.

4.1 Selected methodology

After analysing different development methodologies, it has been decided to use **extreme programming** (XP). This methodology is very suitable for projects carried out by a single person and with tight deadlines, as it allows for a more dynamic development process and the creation of many versions with short development cycles [2].
It also provides greater flexibility in terms of changing requirements, as it is made clear that it is normal and even desirable for these requirements to undergo modifications as the project progresses and takes shape. With this methodology, it is also advocated for good communication and feedback with the client, as the client is involved in the project to give feedback on what they like and what they don't like about what has been developed so far.
Another positive aspect in its favour is that it supports simplicity of code, emphasising the importance of documenting the code well instead of filling it with comments that will become outdated as soon as the code is modified. This documentation involves the refactoring of the code to improve it as it expands, making it easier to maintain in turn; and also by using variable and function names that make it clear to anyone reading the code what the purpose or goal of these is [3].

4.2 Server-side technologies

- *PHP 8.1*: to develop the code that will run on the server side, it has been decided to use the PHP programming language, as it is a versatile and well-documented language that has been used for decades in web creation and is still being used today. In case the association wants to expand the website later on, it will be very easy to find a PHP developer to help them do it. Additionally, in regards to the technical aspect PHP has good performance and is very easy to integrate with the front-end HTML and CSS.
- *MySQL*: as for the database to store the website information it has been decided to use MySQL, which is a relational and open-source database that is among the most widely used in the world. Thanks to its large community, it will be much easier to maintain.
- *XAMPP*: is a software that integrates a database manager and a local Apache server that will be used to host the website temporarily while the code is being implemented.
- *Hostinger*: is a paid web server that will be where the web is permanently hosted. It has relatively low prices and provides a lot of resources while having an intuitive and easy-to-use interface. Additionally, the association already had a Hostinger account contracted, so it will be easier to use in the future as they already know the tool.

4.3 Client-side technologies

- *Bootstrap*: is an open-source framework that allows the use of predefined classes to easily create responsive web pages with great flexibility. It uses HTML, CSS, and JavaScript internally to implement various components, leading to cleaner code and simultaneously saving a lot of time for the developer.

4.4 Web and desktop tools

- *PHPStorm*: is an integrated development environment (IDE) specialized in PHP code and other web development technologies like HTML, CSS and JavaScript. It also integrates Git for easy and quick code upload and updates to a GitHub repository.
- *Mailtrap*: is a web tool that allows capturing the emails that our website tries to send even when we are in localhost and the emails cannot be truly sent due to the lack of an SMTP server. In this way, the functionalities related to emailing users can be tested without having to configure the server where the website will finally be hosted.

- *Epigeon*: is a web software that allows creating and sending newsletters to the desired email addresses. This software will be used so that the association can continue sending newsletters here but connecting the information about the subscribers stored in our database through CSV file imports.
- *GitHub*: it will be used to store the finished website in a private repository as a backup so that the website code is not only on the Hostinger server. The idea is to upload the project to the main branch with my personal account when the project is completed and transfer the repository privileges to an association's account.
- *MS Project*: is a Microsoft desktop product used to manage projects. It has tools for creating and assigning tasks to different resources and also allows creating Gantt charts for better visual and effective planning adjustments.

5 PLANNING

With regards to the project planning, it was decided to separate the two most important parts of the implementation. On one hand, once the requirements capture and design were completed, the back-end of the website would be started. The initial idea was to complete the back-end of the most important parts of the website first and then start programming the entire front-end visual part. Although this was largely achieved, there were some back-end parts that were delayed due to the difficulty they entailed and that's why it was decided to leave these more tedious parts for last, once the majority of both the back-end and front-end were completed.

The initial task planning and its corresponding Gantt chart can be seen in *Appendix A.1* and *A.2* respectively.

6 REQUIREMENTS' ANALYSIS

The first step in starting a software engineering process is to gather requirements. There are various techniques for extracting software requirements, and in this case, the following technique was chosen:

6.1 Interviews

In this case, the interviews were carried out in a telematic way joining with the tutor and a member of the association, whom we will call John. At first, John was asked to explain in general terms what he intended to achieve with the website, that is, what the main objectives of it would be. During this phase, I tried not to interrupt him and as John was speaking, I took schematic notes of all the important points he was commenting on.

Once he finished explaining, we moved on to another phase where I asked John questions to refine aspects that were not clear or were vaguely described. There were certain parts of the web that were not well thought out in terms of how they would be carried out, so by offering suggestions and trying to ask about elements that typically contain websites that John may not have mentioned, most of the requirements were satisfactorily extracted.

Once the meeting was finished, I created a requirements list taking into account everything that was discussed in the meeting. Once the list was completed, we agreed to a second meeting but this time to read the requirements I had written and to go over them one by one to give approval or change certain parts that John did not like, as well as to add any more specific functionality that was left out in the previous meeting.

It should be mentioned that although many requirements remained the same as they were in the beginning, there were others that underwent some change during the development process, which is not negative given that the methodology chosen for this project encourages these changes during implementation.

6.2 Functional requirements

In the following table 1 it can be seen the list of functional requirements that was obtained after completing the interviews:

Code	Title
RF-01	Definition of 4 user types: administrator, entity, subscriber, and visitor.
RF-02	Ability to log in for administrator and entity users.
RF-03	Registering new users by the administrator user.
RF-04	Closing the session for the administrator and entity users.
RF-05	Modifying the profile for the administrator and entity users.
RF-06	Deleting user accounts by the administrator user.
RF-07	Proposing activities by the administrator and entity users.
RF-08	Notifying the administrator user and the organising entity about a new proposed activity.
RF-09	Approving activities by the administrator.
RF-10	Displaying approved activities in an agenda and calendar.
RF-11	Filtering approved activities in the agenda and calendar.
RF-12	Searching for activities, entities, news and articles through a search bar.
RF-13	Modifying the information of a proposed activity.
RF-14	Cancelling an activity from the agenda.
RF-15	Displaying the list of entities.
RF-16	Filtering the list of entities.
RF-17	Adding a new entity by the administrator user.
RF-18	Modifying data of an entity by the administrator user.
RF-19	Deleting an entity by the administrator user.
RF-20	Publishing news and articles by the administrator.
RF-21	Displaying the list of news and articles.
RF-22	Filtering the list of news and articles.

TABLE 1: FUNCTIONAL REQUIREMENTS

Code	Title
RF-23	Modifying news and articles by the administrator.
RF-24	Deleting news and articles by the administrator.
RF-25	Pinning and unpinning relevant news and articles.
RF-26	Adding a topic by the administrator user.
RF-27	Adding tags to a topic by the administrator user.
RF-28	Deleting a topic by the administrator user.
RF-29	Deleting tags by the administrator user.
RF-30	The system must be able to connect to the external newsletter software to send the information related to the user's emails and activities.
RF-31	Being able to subscribe to the newsletter by any user who has not logged in.
RF-32	Cancelling subscription to the newsletter by subscribed users.
RF-33	Changing the interface language.

TABLE 1: FUNCTIONAL REQUIREMENTS (CONTINUATION)

6.3 Non-functional requirements

On the other hand, in table 2 there is the list of non-functional requirements:

Code	Title
RNF-01	Catalan as the main language.
RNF-02	Spanish and English as alternative languages.
RNF-03	Responsive design.
RNF-04	Encrypt the password of the administrator and entity users.

TABLE 2: NON-FUNCTIONAL REQUIREMENTS

7 DESIGN

Once the requirements analysis is finished, we begin with the web design. In this phase, the web architecture will be decided, and various diagrams and prototypes will be made to detail the web's functionality and how the different components interact with each other.

7.1 Architecture

The first step was to decide the type of architecture that the web was going to have. It was decided to opt for the MVC (Model-View-Controller) paradigm using a router to distribute the different web resources.
A resource contains a web functionality, although its logic and visual presentation are not in the same file. Using the router, different web resources are called by passing parameters through the web's URI. These resource files contain calls to the controller, which is the one that manages the functionality logic and is the one who makes calls to both the model and the view. It should be noted that the controller is the only one with permission to make these calls, that is, the model cannot call the view and vice versa. On the other hand, the model focuses on performing the different CRUD (create, read, update, delete) operations in the database. The view is composed of the web's front-end and displays the information collected in a presentable and appropriate way.

7.2 Prototype design

In order to have a clearer idea of how the different screens of the website would be, it was decided to carry out a prototype design. Although this design can be done by hand, it was determined to use software in order to create prototypes of higher quality. In general, an attempt was made to create a minimalist web design so that it could be avoided having a screen that is too busy with elements that are not relevant for the purpose the user is doing at that moment, all with the goal of having a web that is as user-friendly as possible. It was also decided to use a colour palette that was not too aggressive, but rather soft, using mainly white and pastel lilac and rosy colours.

In order to illustrate these prototypes and as an example, the following figure 1 shows the design of a prototype representing how the page where the activities are added would look like:

Fig. 1: Prototype of add activity

7.3 Diagrams

To summarise how the different functionalities and actors interact with each other, it was crafted a use case diagram for the complete web which it can be seen in figure 2:

As well as a diagram with the database structure that can be seen in figure 3:

8 IMPLEMENTATION

The next step in the engineering process is implementation, which involves developing the code for the website, trying to follow the initial plan, and making changes as needed.

8.1 Description of the development process

To this end, as seen in the initial planning (Fig. 1), it was decided to start with the back-end. To make the back-end functional without being hosted on a real web server and to easily implement and test the code, it was resolved to use an Apache server with XAMPP to test the PHP code locally.

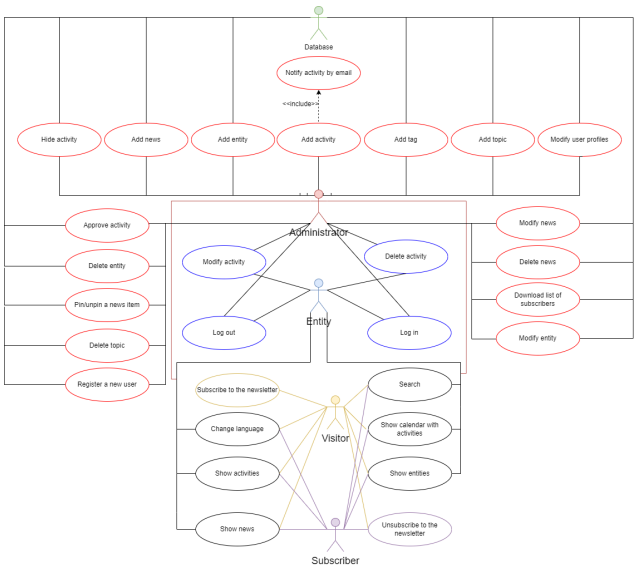

Fig. 2: Use case diagram

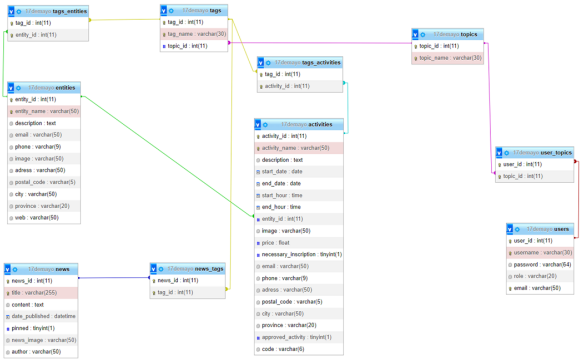

Fig. 3: Database structure

```php
<?php

session_start();

$action = $_GET['action'] ?? null;
if (!isset($_SESSION['lang'])) {
    $idioma = "cat";
    $_SESSION['lang'] = "cat";
}
else $idioma = $_SESSION['lang'];
$lang_file = file_exists( filename: "public/lang/$idioma.ini")
    ? "public/lang/$idioma.ini" :
    "public/lang/cat.ini";
$GLOBALS['translation'] = parse_ini_file($lang_file);

switch ($action) {
    case 'modify_activity':
        require_once(__DIR__ . '/rsc_modify_activity.php');
        break;
    case 'modify_news':
        require_once(__DIR__ . '/rsc_modify_news.php');
        break;
    case 'delete_news':
        require_once(__DIR__ . '/rsc_delete_news.php');
        break;
    case 'list_tags_topics':
        require_once(__DIR__ . '/rsc_list_tags_topics.php');
        break;
    case 'delete_tags':
        require_once(__DIR__ . '/rsc_delete_tags.php');
        break;
    case 'delete_topics':
        require_once(__DIR__ . '/rsc_delete_topics.php');
        break;
    case 'change_pinned':
        require_once(__DIR__ . '/rsc_change_pinned.php');
        break;
    case 'news_detail':
        require_once(__DIR__ . '/rsc_news_detail.php');
        break;
```

Fig. 4: First lines of the router index.php

Once this was done, first of all, all the necessary directories were created to separate all the parts of the website correctly (model, view, controller, css, js, img...), and the router was created in the root of the project. The router can be seen in the following figure 4 which contains the necessary variables so that the translation works correctly as well as the different cases that can be called once a resource is requested:

Once the general structure of the web was created, the process of creating the controllers for the most basic web functionalities began, leaving the more complicated ones that caused problems for later in order not to get stuck and waste time. As I needed to call functions from the model to execute database queries, I created the model files, with one file for each main table (I have not separated in different files the tables that consist of foreign keys from other tables). As for the views, at this early stage of implementation I simply created files with basic HTML code without any style for the sake of performing tests of the PHP code as I was making it.

Once most of the functionalities were finished (back-end), it was determined to start implementing the front-end part so that the web would start to look better and I could show the project to the members of the association so they could evaluate the current state of the web at that point. This was done despite the fact that the plan was to have finished all the back-end before continuing with the front-end, but it was thought that it would be more productive to change it and continue with the front-end to advance more quickly with the web and leave the functionalities that were taking longer to bring out for the end.

Later, the functionalities that were taking longer and had been left aside to continue with the front-end were resumed when the visual part of the web was almost finished. At that point, parts such as sending emails from the web when adding activities, using tags for activities, news and entities, etc., could continue in a way that both the back-end and front-end of those functionalities were made as soon as they were needed.

While it would have been desirable to be able to do all the tasks according to the plan, this did not pose any burden on the development, as making code as you go along can also be an efficient way to program, as you have the code from the other part more recent and you don't have to make any change of context from one part of the code to the other.

8.2 Uploading the web to the server

Once the great majority of functionalities were implemented it was time to upload the website to the Hostinger server. In order to do that, we used the account that the association already possesses and uploaded the files of the project. Then I had to change the version of PHP that Hostinger was interpreting as at the beginning nothing was

working due to this factor. Once I discovered that problem, I moved on to create the database. This was done exporting the database I already had in my localhost and import it in Hostinger creating first a user for the database with admin privileges.
After this, only one thing was missing regarding this part, and it was the installation of an SSL certificate so that the website can establish https connections. This was done using the Hostinger interface in a very effortlessly way.

9 Testing

The final stage of the software engineering process is testing, which involves testing the code in order to detect bugs and improve the software's usability and perfect it. For this project, it has been decided to perform User Acceptance Testing.

9.1 Testing with final users

Given the nature of the project and the fact that the website will be used by individuals with technical and non-technical knowledge, it has been decided to have two people testing the website. The first person is a 24-year-old computer engineering student, and the second is a member of the association who has no technical knowledge in computer science, but only user-level knowledge. Both individuals will give their impressions on possible improvements and report any bugs they find after testing the website. They will also test the website on different devices to test the responsiveness of the page.
The member of the association, who we will call John, tested the different functionalities of the website and recommended making certain improvements to reduce errors and improve usability. Although the list of improvements was quite extensive, due to the lack of time, as this message was communicated when only a few days remained before the final delivery of the project, only some of the reported issues were fixed. A summary of the errors that were fixed thanks to this testing process will now be provided:

- The activities were supposed to be hidden when their celebration date had passed, as before this point they simply stayed there until the administrator removed them.
- Images were not being uploaded to the server because, when moving from the local server where the implementation was done to the Hostinger server, the absolute paths where the files were stored had changed.
- Topics that did not have associated tags simply did not show, so it was requested that when creating a topic, a tag with the same name be automatically created for it.
- It was also suggested that the description fields of the entities and activities have a broader restriction of characters and, with the aim of improving usability, that all fields with character restrictions show visually what the maximum was.
- Finally, it is worth mentioning that it was also changed so that when an activity was modified by an entity when it had already been approved by the administrator to appear in the list of activities, the administrator would be asked again so that they would have to approve it again.
On the other hand, the computer engineering student, in addition to a large part of the contributions that John had already mentioned, also recommended that the hamburger button icon that replaced the menu when the screen was made small should include the three links that were outside (who we are, agenda, and contact) so that everything looked much more compact and those loose links would not be left on top of the button.

10 Results

In this section, the final results of the website will be discussed, providing images of how the most important functionalities have finally turned out, as well as detailing how they could be expanded and improved in the future.

10.1 Tags and topics

With the objective of being able to classify the different activities, entities and news into easily identifiable groups, it was decided to create tags. All these previous elements can have various tags associated with them. Furthermore, to facilitate the sending of newsletters from the Epigeon software used by the association, it was also decided to create the topics that serve to encompass a few related tags. This way, a few topics can be had so that the user who wants to subscribe can select the ones that interest him or her the most, and the association sends the newsletter according to his or her preferences, without the need to create newsletters for each of the tags, since there would be too many.
This functionality could be improved in the future by using some drag&drop mechanism to change the tags of one topic to another and also by merging the creation of tags and topics on a single screen. The following figure 5 is the tag management page as it has finally turned out:

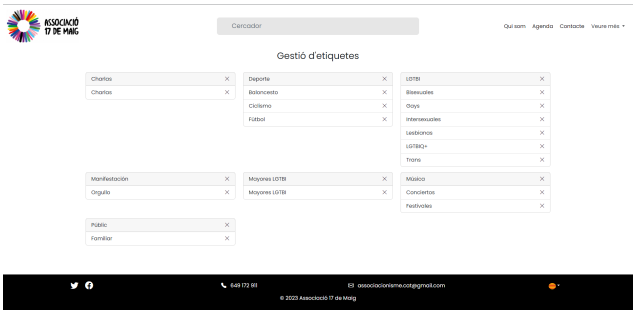

Fig. 5: Tag management

10.2 Activities

This functionality allows both administrators and entities to propose activities related to the LGBTQ+ community. In order for the activities to appear on the agenda for everyone, the administrator must first approve them, thus avoiding someone putting activities that the 17th of May association does not consider appropriate to promote on its web-

site. This functionality could be complemented with a calendar that shows the planned activities for each day. This was planned to be done but finally, due to lack of time and the complexity it entailed, it was decided to prioritise other parts of the website such as sending emails to first notify the administrator that someone had proposed a new activity and it was pending approval, and to the entities to send them the code to be able to modify the activities.
The process of modifying the activities has been done through the introduction of a code since the association asked for only one entity user and that they would give the different entities that wanted to publish activities on the website the credentials to that user. The problem with this was that it created a problem since anyone who had logged in with the entity user could modify all the activities proposed by other entities. To solve this, a code was generated at the time of adding the activity and sent to the email of the entity proposing it to be able to modify it in the future. Below in figure 6, you can see an example of how the screen to add activities looks like.

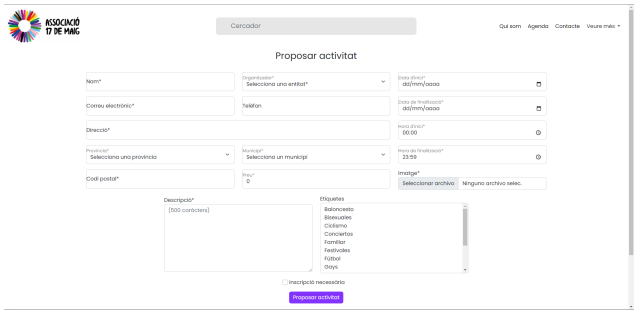

Fig. 6: Add activity

In the figure 7 it can be seen the detail of an activity clicked within the schedule section:

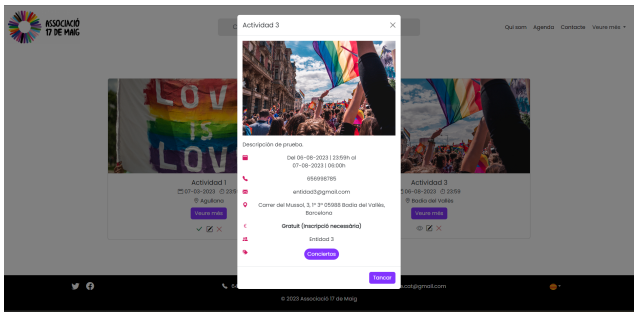

Fig. 7: Activity detail

10.3 News

Regarding the news section, a news listing has been implemented that is ordered based on the publication date, meaning the newest appear first. In addition to this basic functionality, the option to pin a news item by clicking on a star has also been implemented, which makes those news items always appear at the top of this section.
All news items are associated with tags to know before reading what the news will be about. Although this could be improved in the future by allowing clicking on a tag to show all news related to that tag. In the following figure 8, a specific news item can be observed:

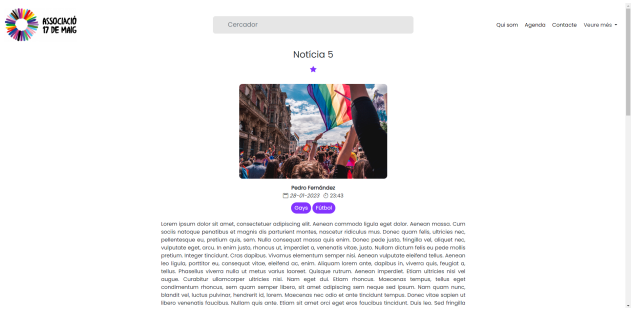

Fig. 8: News detail

10.4 User management

Another aspect of the web's structure is user management, as it plays a very pivotal role in the rest of the functionalities. The system operates according to roles, meaning that there are certain parts of the web that are only accessible if you log in with a user with specific privileges. On this website, there are three types of roles: the administrator, the entity, and the visitor. The administrator can freely modify any activity, entity, or news, as well as delete or approve them. In addition, they can create other administrator or entity users. Users with the visitor role are users who have not logged in and can only see information on the web and cannot modify or delete anything. On the other hand, users with the entity role can see the same as visitor users but also have the option to propose activities and only modify the ones they have proposed. Due to the restriction imposed by the association members that they did not want to have to create entity users for each of the associations, but instead wanted to give the same user account to every entity to be able to propose activities, the system to delete and modify each user's data was slightly changed. What was done was that an administrator user can modify or delete their own account and all entity-type accounts, but cannot modify or delete the accounts of other administrator-type users. Besides, it is worth mentioning that there is another type of user called subscriber but the only difference between this one and a visitor is that we store the email and the topics that he or she likes in the database, but he or she can see and use the same functionalities as a visitor user.
In the future, I would recommend that the association change their stance on this issue and allow each entity to create a user for themselves, as it would avoid security problems if each of them has their own account. In figure 9 you can see an example of the screen that a administrator user sees when logged in, where they can modify and delete their own account, as well as the other existing entity-type account.

Fig. 9: User list seen as an administrator user

10.5 Search bar

Finally, among the most relevant web functionalities, we have the search bar, which allows users to search for approved activities, entities, and news by title, description, city, or province. From this same search section, it is also possible to access the functionalities of modifying activities, entities, and news, as well as approving activities or hiding them in the same way as in their respective sections for listing activities, entities, and news, as long as you are logged in as an administrator user. This makes it much easier to find what you want to modify when there are many elements added in these sections. It also allows regular users to search for activities near them, which can be very convenient instead of having to look at the entire list of activities showing some of them that are too far away in another province.
This functionality could be improved by also allowing you to search by tags and allowing for a more extensive filtering, although the association did not consider it very important and preferred to prioritise other parts of the web due to a planning error that resulted in a lack of time in the final moments of the project. Figure 10 shows an example of a search that produces results of news and activities:

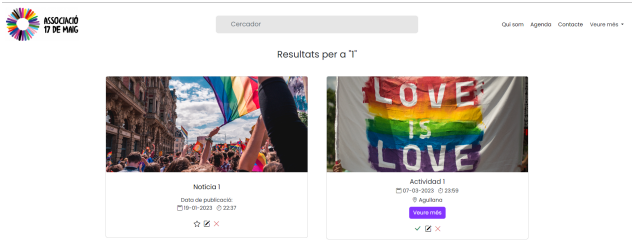

Fig. 10: Search example

11 Conclusions

To conclude this project, the degree of compliance with the objectives that were set at the beginning of the project must be reviewed. It was proposed first of all to carry out a satisfactory analysis of requirements, where through the use of interviews with the members of the association these requirements were able to be extracted without any issue.
A planning was also developed that was especially fulfilled for the first parts of the project but was somewhat neglected towards the end of it. Even so, I consider that it has been quite positive to be clear from the beginning about the dates and limits to carry out the different parts of the project, although the planning could have been improved.
On the other hand, the chosen methodology of extreme programming has allowed considerable flexibility when carrying out the implementation, which has been substantially beneficial for the outcome of the website.
Likewise, a correct development of the web has been carried out using the MVC model, which is one of the most widespread model in the present date and which will allow the web to grow in the future, having an easy structure for other developers to expand it. In the same way, a MySQL database has also been implemented, being that is widely used by the community and considering the size of the web I think it was a perfect choice, since it would not have been worth it to use NoSQL since this web does not need to store massive amounts of information.
Finally, it should be noted that with the help of a member of the association and a colleague from the university, it has been possible to test the different parts of the website with the aim of improving its usability and minimising the number of errors.
That is why it can be said that the main objectives of the project have been achieved, although perhaps with better planning other not so important web functionalities could have been also achieved as enhancers of the web. In any case, the website is ready to carry out the main functions for which the association requested it, which is basically to be able to promote LGTBQ+ activities and entities as well as publish news and articles to help the members of the collective; and all of this has been achieved.

The domain where the website will be stored is: https://www.associacionisme.cat

Acknowledgements

I would first of all like to thank my parents and grandmother for all the support they have given me throughout my studies, especially in the early years when I was at UPC and I did not see how I could finish this career. Also, in this sense, I want to thank my best friend Mª Àngels, with whom I have finished the career together and without her I would never have been able to finish it because she was the one who encouraged me to continue and not to drop it. Furthermore, by always being together we have helped each other a lot to study and pass all the subjects we have done. I also want to mention my partner, who has been very patient and understanding during these past few months when I have been very busy with this project but always encouraged me a lot and I am very excited to start this new stage of my life by his side. Finally, I want to thank my tutor Alicia for all the advice she has given me to improve this work and for always being available for me in a very close way.

References

[1] ≪Quedeque≫. https://quedeque.barcelona/ (accessed 2nd October 2022).

[2] Asana, ≪Las 12 metodologías más populares para la gestión de proyectos • Asana≫, Asana. https://asana.com/es/resources/project-management-methodologies (accessed 8th October 2022).

[3] ≪Extreme programming≫, Wikipedia, the free encyclopedia. 30th May 2022. Accessed: 8th October 2022. [Online]. Available in: https://en.wikipedia.org/wiki/Extreme_programming

APPENDIX

A.1 Task planning

Id	Modo de tarea	Nombre de tarea	Duración	Comienzo	Fin
1		**Web asociación 17 de may**	**115 días**	**vie 16/09/22**	**jue 23/02/23**
2		Análisis de requisitos	16 días	vie 16/09/22	vie 07/10/22
3		Redacción del informe inicial	2 días	vie 07/10/22	dom 09/10/22
4		**Back-end**	**31 días**	**lun 10/10/22**	**dom 20/11/22**
5		Diseño y arquitectura del	6 días	lun 10/10/22	lun 17/10/22
6		Desarrollo del back-end	28 días	jue 13/10/22	dom 20/11/22
7		Redacción del informe de progreso I	3 días	jue 10/11/22	dom 13/11/22
8		**Front-end**	**39 días**	**lun 21/11/22**	**jue 12/01/23**
9		Diseño de prototipos	6 días	lun 21/11/22	lun 28/11/22
10		Desarrollo del front-end	33 días	mar 29/11/22	jue 12/01/23
11		Redacción del informe de progreso II	3 días	jue 15/12/22	dom 18/12/22
12		Testing y corrección de errores	17 días	jue 12/01/23	vie 03/02/23
13		Redacción informe fina	10 días	mié 25/01/23	mar 07/02/23
14		Preparación defensa TF	12 días	mié 08/02/23	jue 23/02/23

Fig. 11: Task planning

A.2 Gantt's diagram

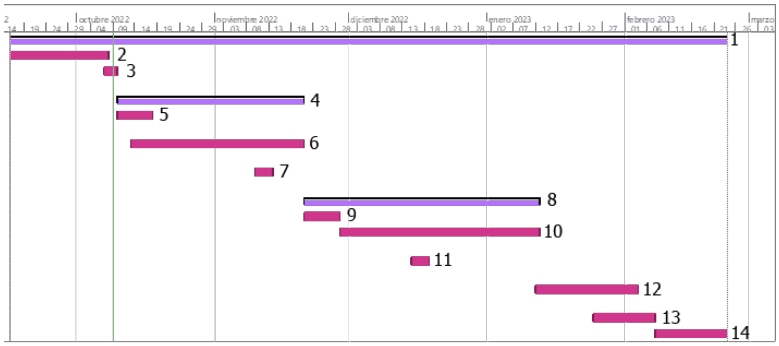

Fig. 12: Gantt's diagram related with the tasks in Appendix A.1