Facultade de Informática
UNIVERSIDADE DA CORUÑA

END-OF-DEGREE THESIS
COMPUTER ENGINEERING DEGREE
MENTION IN SOFTWARE ENGINEERING

Euro-Inf Bachelor awarded by EQANIE

Design and implementation of an e-commerce web application for a food store

Student: Sara Carril Méndez

Direction: Fernando Bellas Permuy

A Coruña, September 09, 2022

.

To my parents, without whom reaching here would not have been possible.

Acknowledgements

In the first place, to my parents, for their encouragement at all stages of my life. To my mother, for always finding the right words to cheer me up and never allowing me to give up. To my father, for transmitting me how proud he is and encouraging me to choose this profession. I love you both.

To my classmates and friends during these four years. Especially to Gema, Anxo and Juan, I couldn´t have better friends to walk through this path. We did it!

To all the professors, for transmitting their knowledge, it was an honor to learn from you.

And, last but not least, to my mentor Fernando Bellas, for the confidence reposed in me, his dedication and good advice.

Abstract

Over the last few years, technology has experienced an exponential growth, fuelled in part by the COVID-19 epidemiological situation. This circumstance has resulted in a remarkable change in society's consumption habits, as more and more consumers are choosing to purchase goods over the Internet. Derived from this behavior, companies must adapt to the demands of the new reality, to gain a foothold in the market and maintain their competitiveness.

Related to the previous case, in this final degree project, it has been designed and developed a web application for online seafood sales. The implemented software will allow users to sign up and login into the platform, in which they will have access to the different products from the catalogue, being able to make their purchases, pay them online and view a record of the orders placed.

To achieve the previously stated objectives, the developed web application has been implemented based on a layered architecture, more in details, following the client-server model and using Java, Spring Boot and Hibernate for the back-end and JavaScript, React and Redux for the front-end.

Resumo

A continua evolución e expansión tecnolóxica ao longo dos últimos anos e a situación epidemiolóxica da COVID-19 propiciaron novos hábitos de consumo na sociedade actual. Cada vez son máis os consumidores que optan por comprar bens a través de Internet. Derivado deste comportamento, as empresas deben adaptarse ás esixencias da nova realidade, para lograr un oco no mercado e manter a súa competitividade.

Vinculado a esta casuística, neste traballo de fin de grao deseñouse e desenvolveuse unha aplicación web para a venda en liña de produtos do mar. O software implementado permitirá o rexistro e acceso de usuarios á plataforma, na que terán acceso aos diferentes produtos do catálogo, podendo realizar pedidos, pagalos en liña e consultar un histórico dos mesmos.

Para acadar os obxectivos anteriormente mencionados, implantouse unha aplicación web baseada nunha arquitectura en capas, seguindo concretamente o modelo cliente-servidor, destacando o uso de Java, Spring Boot e Hibernate para o lado do back-end e JavaScript, React e Redux para o front-end.

Keywords:

- Web Application
- E-commerce
- Java
- JavaScript
- CSS
- Bootstrap
- MySQL
- React
- Redux
- Spring Boot

Palabras chave:

- Aplicación web
- Comercio electrónico
- Java
- JavaScript
- CSS
- Bootstrap
- MySQL
- React
- Redux
- Spring Boot

Contents

List of Figures

List of Tables

Chapter 1

Introduction

In this first chapter of the memory, we will focus on the motivation and objectives pursued with the deployment of the e-commerce web application.

1.1 Motivation and context

Today's society follows a frenetic pace, people live increasingly faster, overwhelmed by lack of time and hoping that all problems could be rapidly and efficiently resolved. That is why, in many sectors e-commerce has overcome traditional businesses. Going to an establishment, standing in lengthy queues to pay for products, carrying bags on the way home, etc. is a waste of time and discomfort that the current society cannot afford. On the contrary, not having geographical or time restrictions, the flexibility in payment methods and the possibility of purchasing without the need to leave home, are the main factors that have led to the success of this trade. In addition, e-commerce provides advantages not only for people, but also for companies; this commerce allows businesses to acquire a strong strategic position in today's market and to reach broader market niches. This is why this type of trade is perceived as a business opportunity capable of modernizing business models of enterprises, assisting them to continue growing by getting international customers. As Bill Gates said, "If your business is not on the Internet, then your business will be out of business."

On the other hand, the idea of this final degree project arises from the demands of a family business, specialized in the manufacture and elaboration of several types of sea products (cod, bivalves and octopus). Currently, the company operates at a wholesale level, with small and medium-sized companies, restaurants and supermarkets among their clients. These customers are provided with wholesale products via telephone orders and bank payment methods that enable financing, in accordance with the final price of the order placed, such as confirming, transfer, promissory notes, etc. Seeking its expansion to new market segments and expanding the existing lines of business, it intends to reach the general public by offering its products on the Internet, in addition to growing into new market categories and enhancing current business lines. Inspired by this context, in this end-of-degree project, a web application that allows the online sale of the company's products to the final consumer will be developed.

1.2 Objectives

Taking into account the aforementioned in the introduction section 1, the objective of this final degree project is the development of a web application that enables the online purchase of seafood products. It is intended that the website provides facilities and a good user experience, but that also helps the company to expand globally, not just limiting itself to the local. All of it, without neglecting the application of programming good practices to obtain a robust software. In more depth, the points that are meant to be touched within the development are detailed below:

- **User management**: there are three different user types that can access the online application. In the first place, **unregistered users**, who will access the application to make inquiries about the catalog, recipes or to contact the company's management with any questions. They will also have the choice to sign up or login into the application, abandoning their anonymity and becoming registered users in the system. The **registered users**, will be able to change their password, update their profile information, and log out. Last but not least, a position for an **administrator** has been created, to let employees of a business to manage the items in the catalog or the purchases made by clients via the web application. Additionally, users will be able to authenticate (via OAuth) using their Google credentials.

- **Catalog management**: Users will be capable of seeing the entire catalog of products offered for sale and a detailed view of each one with its specifications (price, sale unit, description, etc.) In addition, they could also add these products to the cart so they can buy them later.

- **Payment management**: Users will be able to make payments straight from the web application thanks to the usage of **Paypal** as a payment method.

- **Orders record**: Users will be able to view their purchases made through the website as well as the details of each of the orders. In addition, they will also be able to view and download a PDF ticket of their buyouts.

- **Legal compliance**: As it is intended to achieve a web page as faithful to reality as possible, the web page will be adequate, in compliance with the GDPR (General Data

Protection Regulation), the Law 34/2002 on ECISSA (E-Commerce and Information Society Services), and other legislation in force in these sectors.

- **Application administration**: The administrators will have the ability to change already existing products and add new ones to the catalog. They will also be able to examine any orders that clients have placed via the online application and change its status.

1.3 Global vision of the system

The developed web application is made up of the front-end (client), the web part with which the users interact and the back-end (server), in charge of connecting with the MySQL database and of the business logic. More in detail, the back-end has been implemented in Java also using Spring Boot, and as it is shown on the Figure 1.1, it is subdivided into two layers:

- **Model layer**: This layer is responsible for data access and contains the business logic of the application.
- **REST (Representational State Transfer) service layer**: It defines a REST API (Application Programming Interface) and therefore responds to different types of requests that arrive from the front-end.

On the other hand, the front-end layer is a web application SPA (Single Page Application), with the goal of providing consumers with a more fluid experience. In this kind of applications, all its screens are displayed on the same page, without having to reload the browser every time the user switches views. It has been implemented using JavaScript, React and Redux. The front-end of the application is made up of:

- **Service Access Layer**: When a user interacts with the page, for example by clicking on a button, this layer is responsible for doing back-end calls.
- **UI (User Interface) layer**: this layer manipulates the DOM (Document Object Model) tree of the page displayed by the browser; that is, it modifies the page to show to the user the desired view when pressing the button.

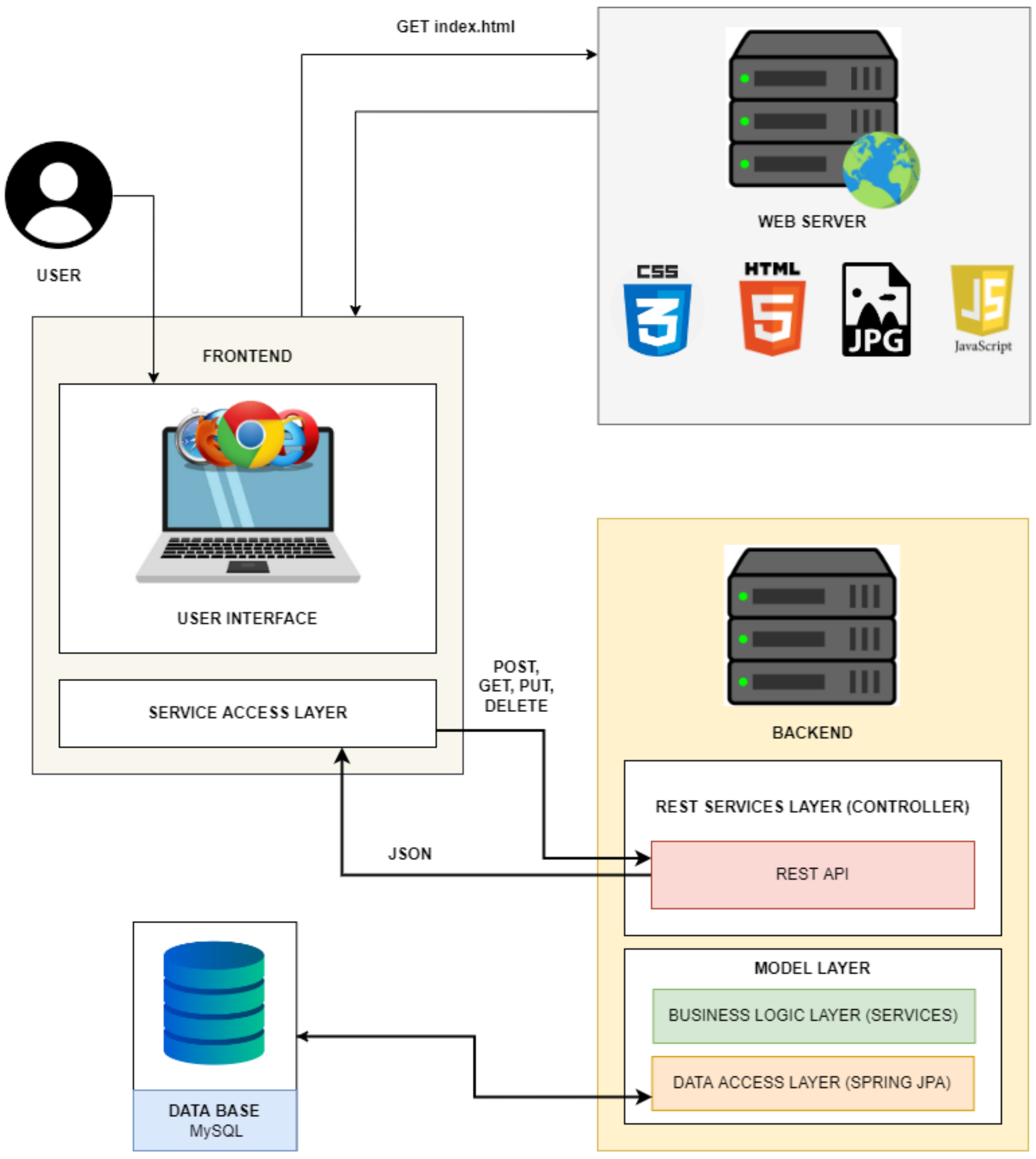


Figure 1.1: General architecture of the application

Chapter 2

State of the art

Nowadays, more and more companies are deciding to take a leap towards digitization and start their journey in the online product market. That is why there are numerous examples of web applications for online sales from different sectors: fashion, food, technology, services, etc. However, it is true that none of the existing pages on the market adapts perfectly to the demands of another specific company, so a custom development is necessary in each circumstance.

2.1 Alternatives

In this section of the report, different e-commerce web applications related to the one developed for this final degree project, are detailed, as they have served as great inspiration throughout its development:

- **Pescanova** [1]: is a Spanish multinational specialized in the capture and sale of seafood. The corporate group is present in the markets of 19 countries and is supported by approximately 10,000 employees worldwide. It is currently the first fishing company in Spain. Part of the large volume of business that this company has every day comes from its online store, where customers can consult all the products they offer, place orders and pay for them online by credit or debit card through the Abanca POS (Point of Sales). In addition to making purchases, customers may browse recipes and cooking suggestions on the website; therefore, in each visit Pescanova adds value to its customers.

  This company has been taken as an example, specifically because of its significance on a commercial level, it belongs to the same industry as the produced web application and this website has many strong points such as easy navigation, responsive design and good usability and organization, advantages shown in figure 2.1

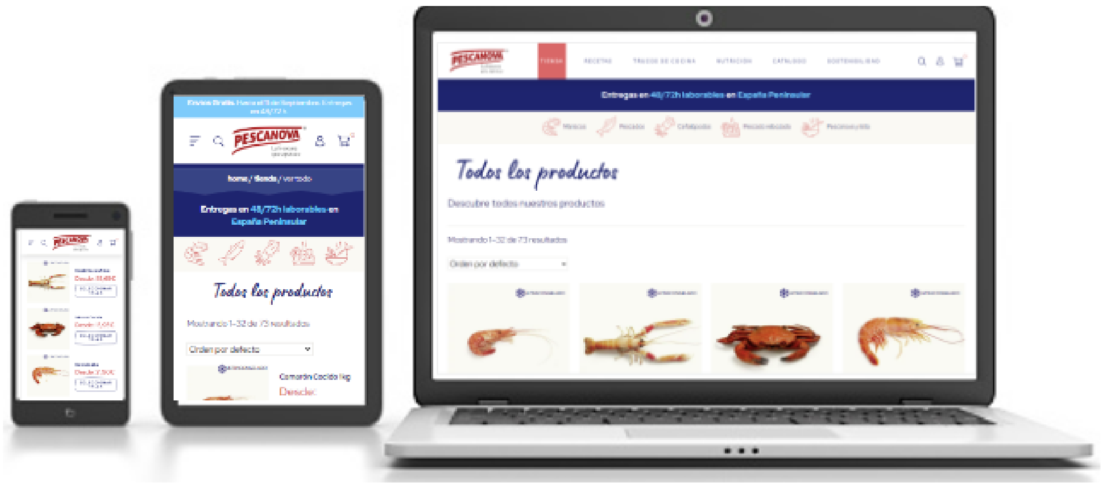
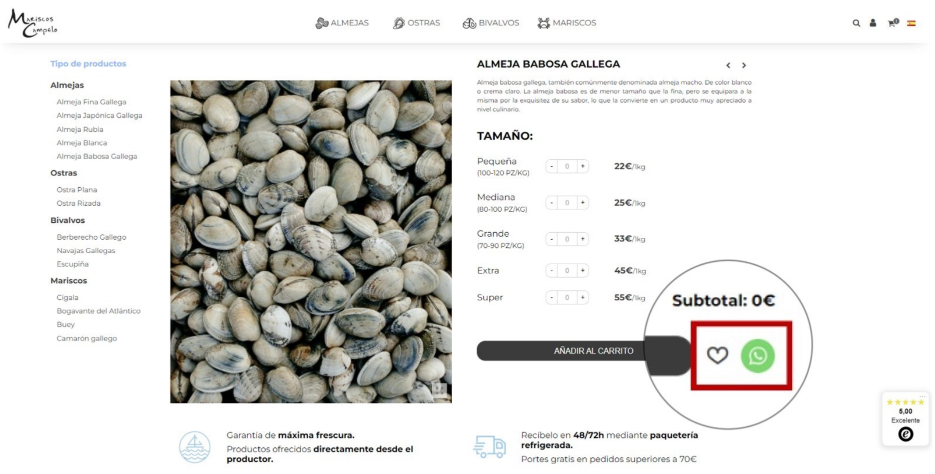

PESCANOVA
Entregas en 48/72h laborables en España Peninsular
Todos los productos
Descubre todos nuestros productos
Orden por defecto

Figure 2.1: Pescanova responsive online catalog

- **Maricos Campelo** [2]: is a Galician processing shellfish company and mollusk treatment plant. It offers a website where customers may browse a catalog of its items, add them to the cart and pay for the purchases online, by transfer, cash on delivery or credit card. This e-commerce shop is simple in terms of web design and user interfaces, which is a very significant non-functional requirement in terms of the page's user experience. In addition, there are certain use cases to emphasize that were not seen in the other choices considered, firstly, it allows to add products to favorites and consult them later, and it also provides direct WhatsApp communication with their business to solve doubts or any problem by simply picking a button. In figure 2.2 are shown both functionalities from the user interface point of view.



ALMEJA BABOSA GALLEGA

Almeja babosa gallega, también comúnmente denominada almeja macho. De color blanco o crema claro. La almeja babosa es de menor tamaño que la fina, pero se equipara a la misma por la exquisitez de su sabor, lo que la convierte en un producto muy apreciado a nivel culinario.

TAMAÑO:

Pequeña (100-120 PZ/KG)	- 0 +	22€/1kg
Mediana (80-100 PZ/KG)	- 0 +	25€/1kg
Grande (70-90 PZ/KG)	- 0 +	33€/1kg
Extra	- 0 +	45€/1kg
Super	- 0 +	55€/1kg

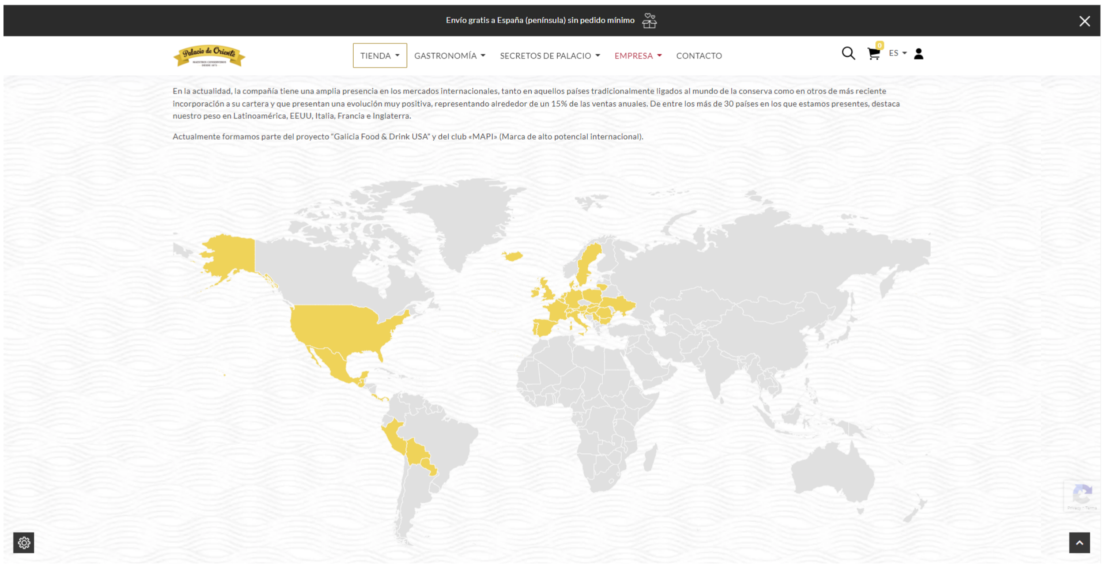
Subtotal: 0€

Garantía de **máxima frescura.**
Productos ofrecidos **directamente desde el productor.**

Recíbelo en **48/72h** mediante **paquetería refrigerada.**
Portes gratis en pedidos superiores a 70€

Figure 2.2: Mariscos Campelo catalog, highlighting the functionalities of favorites and contact.

- **Palacio de Oriente** [3]: is the oldest canning company in Spain, specializing in bonito from the north, light tuna and mussels from the Galician estuaries. Once more, this website provides the selling of its products while taking into account other common use cases for online retailers. After studying this website, it has been detected that it is not implemented using internationalization, a highly unfavorable aspect of this company's presentation since, as seen in figure 2.3, the marketing spots are spread out throughout practically the whole of Europe and part of America, so it does not encourage international consumers having a positive shopping experience. However, the very exact adherence to the law and the inclusion of all user-required legal information were noted as positive characteristics of this page.



En la actualidad, la compañía tiene una amplia presencia en los mercados internacionales, tanto en aquellos países tradicionalmente ligados al mundo de la conserva como en otros de más reciente incorporación a su cartera y que presentan una evolución muy positiva, representando alrededor de un 15% de las ventas anuales. De entre los más de 30 países en los que estamos presentes, destaca nuestro peso en Latinoamérica, EEUU, Italia, Francia e Inglaterra.

Actualmente formamos parte del proyecto "Galicia Food & Drink USA" y del club «MAPI» (Marca de alto potencial internacional).

Figure 2.3: Palacio de Oriente points of sale.

2.2 Conclusions

The market is swamped with e-commerce web apps, including the ones for the food industry and other sectors, which can be used as a model or source of inspiration while developing this project, due to the commonality in terms of use cases.

Prior to development, the pages indicated above on the section 2.1 were examined, in order to take lessons from other companies past errors or to gain insight into how the top businesses in the industry create their web applications. What has been detected and concluded is that, despite the fact that they are virtually all well-known and worldwide enterprises, none of them execute internationalization, which means that the goal of reaching the broadest possible audience is not fulfilled. Additionally, some imperfections were also detected, which highlights the importance of carrying out very exhaustive testing processes in this kind of applications.

On the other hand, positive aspects and development ideas have been clearly drawn, for example to carry out a good user experience design by providing clear and simple interfaces, by using bolder colors to draw attention to the page's most important elements, not compelling users to have to give more than three clicks to get their objectives, assisting consumers in avoiding errors warning them about possible mistakes and helping to recover if some problem takes place, in addition to establishing an adaptive design, which allows users to connect from any device.

Chapter 3

Methodology

THIS chapter will go into depth about the methodology that has been employed for the development of the project, including what it entails, its key features, and how it has been applied.

3.1 Development methodology

The development of a computer application is a complex challenge that requires careful thought, great effort, planning and rigorous order. Not working in an organized manner has traditionally had negative effects, such as the software crisis of the late 1970s, time in which between 30% and 40% of the software developed was not usable, due to a lack of preparation and structure.

The majority of software issues are caused by overly optimistic planning or due to subpar work because of inadequate testing, or unplanned project changes. In order to prevent these issues, projects must be carried out in accordance with methodologies that allow for flexibility, autonomy, and efficiency, which in turn lowers costs and boosts productivity.

This project was carried out utilizing an approach that combines some aspects of linear and agile methodologies: an iterative and incremental methodology. The primary phases followed during the development based on this technique are outlined below.

- **Preliminary analysis**: A preliminary analysis phase is the first step in the incremental and iterative development. During this step, the requirements that will identify the division into iterations are gathered, the best technologies to be used in the project are studied, some of the comparable applications that are already available on the market are examined, and mock-ups are created for those use cases that call for them.

- **Product development**: The development of the product is carried out through iterations and in each of them a set of requirements will be implemented. Each iteration begins with the product developed in the previous iteration and when it is done, it must provide an usable product that increases the functionalities of the previous product. The steps of analysis, design, implementation, and testing make up each iteration; they are described in more detail below:

  - **Analysis**: In the first place, it is analyzed what the program must perform precisely, what the specific objectives are and how the iteration will be approached to meet these goals.
  - **Design**: This step involves choosing the overall structure of the program to be developed as well as researching potential implementation alternatives. Additionally, mock-ups will be used to define how information and functionality will be presented to the user.
  - **Implementation**: Once the requirements and functionalities of the system are understood and its structure has been designed through the analysis and design, these functionalities begin to be implemented with the aim of obtaining a fully functional but unfinished product in terms of the system's overall goals.
  - **Testing**: This stage is completed concurrently with the development, with the goal of catching faults sooner that, absent testing, would not have been discovered until later stages and would have had a considerably greater detrimental effect on the project's progress.

- **Final testing**: Once all the iterations of the project have been carried out and a final product has been obtained, more exhaustive tests have been run, with the aim of testing the complete component, rather than just one part or the performance of each iteration, in order to identify issues or improvements.

Next, a more graphic and illustrative representation of the iterative and incremental methodology employed during the project's development is presented in figure 3.1.

3.1.1 Advantages and disadvantages of incremental iterative methodology

Given that each project has unique conditions and features, it is probable that the methodology will not be suitable to the demands in all aspects of development. The benefits and drawbacks of the chosen methodology are then outlined.

- **Advantages**:
  - A functioning program is obtained from the beginning, so the user does not have to wait until the conclusion of development to start using the app.

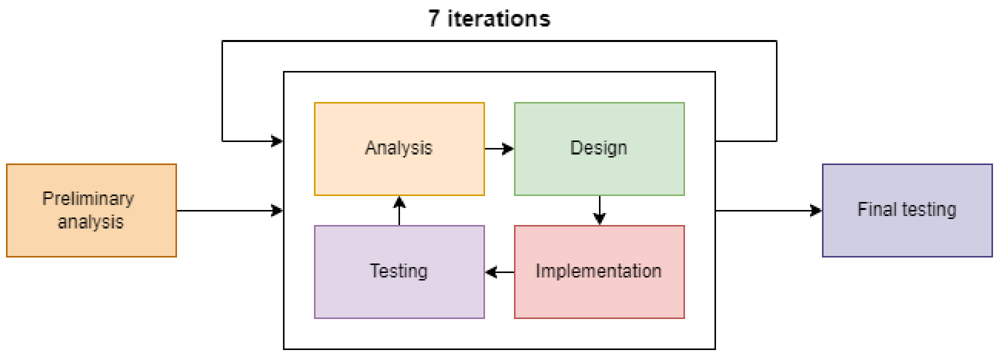


Figure 3.1: Iterative incremental methodology diagram

- Reduces the likelihood of discovering bugs in critical parts of the software too late because crucial iterations are delivered first, so they undergo a higher number of tests.
- It is a very adaptable methodology to the changing demands of the project and allows a flexible development of the project.
- It is possible to readily evaluate the project's progress in each iteration, making it easier to refocus the project if substantial deviations from the estimate occur.

- **Disadvantages**:
  - The system design requires proactive attention and may require significant modifications as iterations proceed.
  - It might be challenging to determine the top or most valuable features, since the requirements for a realistic project with clients are often changeable.

3.1.2 Methodology justification

By the end of this section, a better understanding of why this iterative methodology has been chosen for this project would be gained. In addition to the iterative and incremental methodology, other agile methodologies, such as SCRUM, have been considered when deciding which methodology to use throughout the project's development.

In terms of the SCRUM methodology, it is a framework for agile software development that enables the use of an incremental development approach while simultaneously providing a significant amount of flexibility in response to shifting needs. However, the aim of this methodology is to provide a set of best practices for teamwork. Additionally, this methodology includes a number of roles that must fall on various persons, such as the Product Owner or Scrum Master, and meetings like the Daily scrum meetings, sprint planning meetings, sprint retrospective, etc. must also be held. These qualities are not appropriate for this project because the development team is made up of a single developer, so, carrying out the aforementioned actions would be pointless. Additionally, SCRUM is separated into different sprints, with a very similar duration between each of them, generally lasting a month or two weeks, depending on how big the project is, so since there is no full-time dedication to this project due to labor reason, this feature is not appropriate.

Last but not least, the iterative-incremental methodology was selected for this project due to its adaptability. The functionalities are organized in iterations, which facilitates the organization and temporary control over the project, this is very similar to the SCRUM sprints, but without the requirement of meetings, roles, or iteration duration. Additionally, because the use cases are so well defined and because this project is not a real project with rapidly changing use cases, the drawbacks of the technique mentioned in point 3.1.1 do not apply to it.

Chapter 4

Global Requirements Analysis

In this chapter of the report, the necessary functionalities and use cases will be analyzed, as well as the different actors involved in the system.

4.1 Actors

The actors presented below stand in for the different roles that users who access the application can take on. There are three primary user groups in the application developed for this project: unauthenticated users, that are the primary visitors to the web application, customers and administrators. More information about each of these groups, as well as the actions they may perform on the application, is provided below.

4.1.1 Unauthenticated users

Any user who accesses the application and has not through the authentication procedure is considered as an anonymous client. This actor can be authenticated in the system or register, if he does not already have an account, obtaining the status of ”Registered customer,” as indicated in the following point. Furthermore, this sort of profile will be allowed to access the application’s current static consultation pages, such as the recipe book, home, privacy policies, cookies, etc. as well as consult the product catalog and the information of each one of the products.

4.1.2 Customers

These kinds of actors have signed in or registered within the system, demonstrating that they have successfully completed the authentication procedure. They will be capable of performing all the actions listed above for anonymous users and, in addition, they will be able to browse their order information, add items to the shopping cart, and make purchases.

4.1.3 Administrators

This role corresponds to the actor who has been authenticated with the admin role. This sort of actor can only log in through the application for security reasons; registration is handled by a developer through the database. The administrator will be able to manage client orders and update the catalog.

Figure 4.1 depicts a hierarchy diagram of the previously described actors.

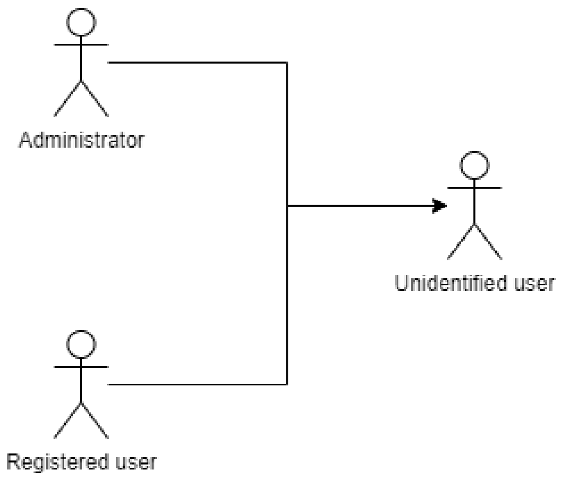


Figure 4.1: Actor hierarchy diagram

4.2 Requirements

4.2.1 Non-functional requirements

Non-functional requirements are properties or qualities that the product must have. They do not relate directly to the specific services supplied by the system, but rather to the system's properties: security, and availability, for instance.

- **Security**: A non-functional need that was taken into consideration for the application was security management. First and foremost, in terms of user security and online accesses, passwords are encrypted in the database so that administrators do not have access to them. Furthermore, every application access is done via a signed access token, in JWT (JSON Web Token) format, which allows the back-end to identify the user securely. On the other hand, access to the application's many URLs (Uniform Resource Locator), are regulated by roles based on the kind of user (unregistered user, registered user, or administrator), as shown in the following example, in which access to the consultation of all orders is restricted to administrators only:

```
        .antMatchers(HttpMethod.GET,
    "/shopping/ordersAll").hasRole("ADMIN")
```

- **Simplicity**: To maintain the web page's simplicity, three color hues and the same typeface style were mostly utilized. Furthermore, more prominent tones have been employed to represent the vital characteristics, while the remainder of the elements, have been left as neutral tones, among other things.
- **Usability**: The created website offers simple pages and menus, quick downloads and intuitive functionalities for the user, attempting to provide user-accessible functionalities that requires no more than three clicks to be completed.
- **Language**: Spanish, English, and Galician internationalization are available, so users visiting the created application could understand the information displayed. In order to do this, the user's browser's default language is automatically recognized and the language that best suits is presented.

4.2.2 Functional requirements

Functional requirements define a function of the software system or its components. The use cases demonstrate the behavioral demands for each functional requirement. They are complemented by non-functional requirements, which are concerned instead on design or implementation. A brief mock-up is then presented in those use cases where it is deemed pertinent and each use case for the system will be explained in more detail.

- **Use cases for unauthenticated users**
  - **UC.001 - User sign up**: An anonymous user can sign up for the application by filling out the form available for this purpose, entering their name, last name, email address, phone number, and the national identity document number. An exception will be issued informing the user if any of these data is entered improperly, and the procedure won't be finished until the problems are fixed.

- **UC.002 - User login**: An anonymous user who has previously registered in the application will be able to log in, either by entering the relevant platform login credentials (email and password). To accomplish this activity successfully, the application username and password must be correct; otherwise, an error message will be shown.
- **UC.020 - User login with a Google account**: Each and every user will be able to authenticate in the app by logging in with their Google account. They must choose the "Sign in with Google" option and enter their email and password in the Google pop-up window.
- **UC.006 - User contact with the administration through the contact form**: Users can contact the application manager for any requests, questions or ideas, by providing a name, surname, email address, subject, and body of the message in the contact form.
- **UC.007 - View all the products from the catalog**: The products in the catalog will be visible to every user that visit the online application. An image of the product, the name, and the price will be displayed for each of them, as it is illustrated in figure 4.2.

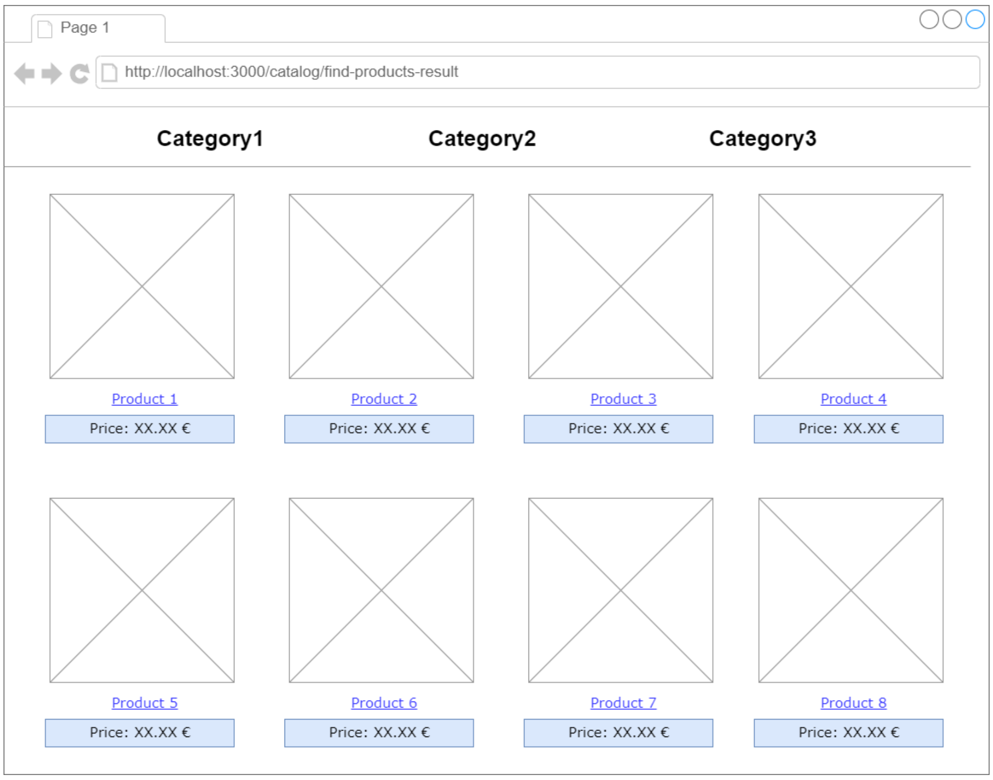
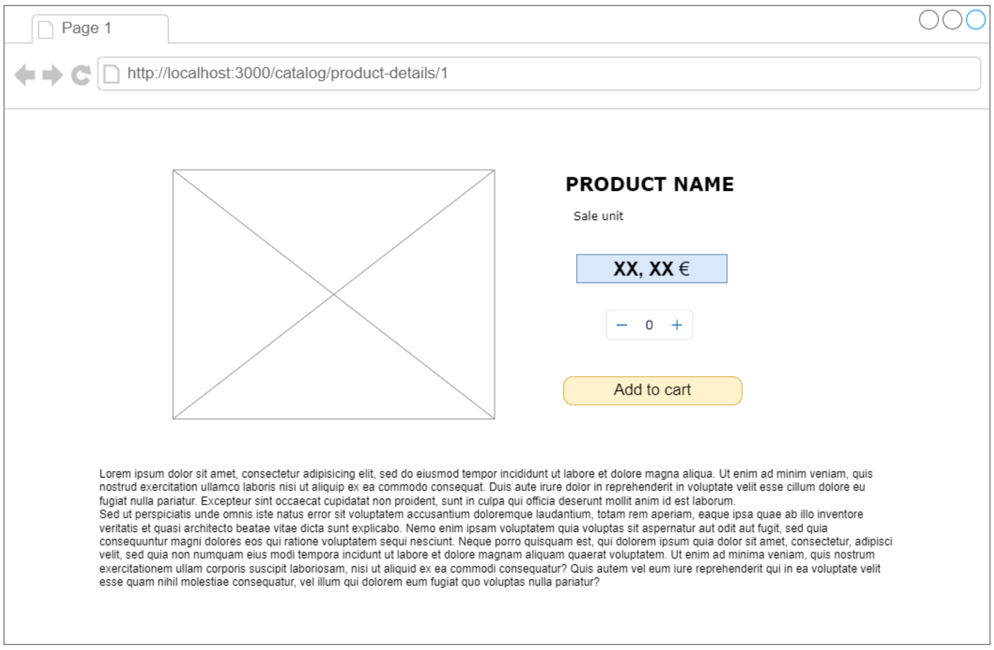


Figure 4.2: Mock-up products from the catalog

– **UC.008 - Filter products by category**: Every user is able to filter the catalog products by category (Codfish, octopus or seafood).

– **UC.009 - View product details**: All application users, including clients and administrators, will be able to choose a product from the catalog and view its information, as shown in figure 4.3.



Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.
Sed ut perspiciatis unde omnis iste natus error sit voluptatem accusantium doloremque laudantium, totam rem aperiam, eaque ipsa quae ab illo inventore veritatis et quasi architecto beatae vitae dicta sunt explicabo. Nemo enim ipsam voluptatem quia voluptas sit aspernatur aut odit aut fugit, sed quia consequuntur magni dolores eos qui ratione voluptatem sequi nesciunt. Neque porro quisquam est, qui dolorem ipsum quia dolor sit amet, consectetur, adipisci velit, sed quia non numquam eius modi tempora incidunt ut labore et dolore magnam aliquam quaerat voluptatem. Ut enim ad minima veniam, quis nostrum exercitationem ullam corporis suscipit laboriosam, nisi ut aliquid ex ea commodi consequatur? Quis autem vel eum iure reprehenderit qui in ea voluptate velit esse quam nihil molestiae consequatur, vel illum qui dolorem eum fugiat quo voluptas nulla pariatur?

Figure 4.3: Mock-up product details

– **UC.025 - About us page**: By choosing the "About us" option in the top menu, any user may consult a quick introduction about the company that the page is about.

– **UC.026 - Privacy policy page**: By clicking on the related link in the footer, any user may consult the privacy policy page, which states the procedures taken by a business or organization to ensure the secure and ethical use of user or client data that is gathered in the course of a business engagement.

– **UC.027 - Cookies policy page**: Any user, authenticated or not, can examine the cookie policy page by looking for the related link in the footer from anywhere on the website. The cookies policy page aims to inform users about their use, so that they are aware of the purpose, duration and owner of the cookies.

– **UC.028 - Shopping Terms and Conditions page**: By checking for the relevant link in the footer, accesible from any page of the website, any user, authenticated or not, can examine the Shopping Terms and Conditions page. The terms of the contract between the seller and the user are spelled out in full on this page and serve as the basis for the sale of the products.

  - **UC.029 - View all the cooking recipes**: By clicking on the "Recipes" part of the menu, any user, authenticated or not, will be able to see a list of the recipes accessible in the system.
  - **UC.030 - View cooking recipe details**: Any user who accesses the system will be able to view the details of a recipe, including the ingredients and method of preparation. To do so, it is necessary to navigate to the "Recipes" area of the menu and choose one.

- **Use cases for authenticated users**
  - **UC.003 - User logout**: An authorized user will be able to log out from his session and would thereafter become an anonymous user.
  - **UC.004 - User information update**: A authenticated user will have the ability to change their name, last name, the national identity document number and the phone number.
  - **UC.005 - Change user password** : An authenticated user can change their password by accessing the "My profile" section, it will be required to enter the old password and the new one twice, to ensure that the user did not make a mistake when entering it. If the user enters an incorrect old password or the two new passwords do not match, an error message will be shown.

- **Use cases for client users**
  - **UC.010 - Add products into the shopping cart**: A customer can add a variety of goods to their shopping cart. The user must visit the product information, choose the quantity, and click the "Add to basket" button in order to accomplish this functionality.
  - **UC.011 - Delete products from the shopping cart**: As long as there is one item in the basket, an authenticated customer can remove any of the products from the shopping cart.
  - **UC.012 - Update products from the shopping cart**: As long as there is one item in the basket, an authorized consumer can adjust the quantity to shop of any item in the shopping cart, introducing firstly the number of units to purchase and then saving this action through the green shopping cart button.
  - **UC.013 - View products from the shopping cart**: An authenticated consumer can examine the goods in her/his shopping basket. If the user's cart is empty, a warning will be presented.

- **UC.014 - Search for user orders**: An authenticated customer will be able to consult the orders made recently, for this purpose, a table will be provided with the order identifier, the date of execution, the total price and the shipping address. If the user has not made any order through the application, a warning will be displayed indicating such information.
- **UC.015 - View order details**: An authenticated customer will be able to consult the details of the orders made, to achieve this, they must pick the identification of the order they wish to consult from the table and it will display the date of purchase, the order identifier, the shipping address and a summary with the information of the products purchased, as shown in the figure 4.4

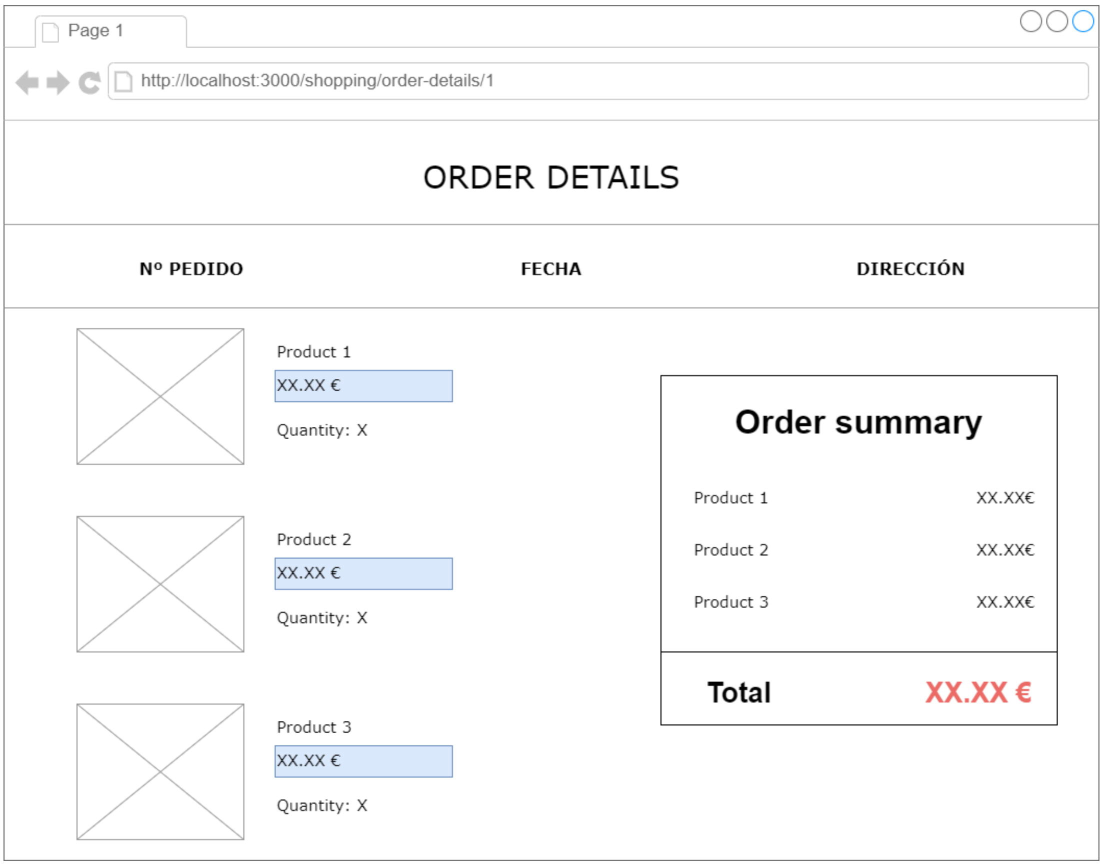


Figure 4.4: Mock-up order details

- **UC.016 - Download order ticket**: By clicking on the ”Download order ticket” option, an authenticated client who has previously placed an order in the program can download or consult the purchase receipt in pdf format.
- **UC.018 - Pay an order with a real payment method**: A website user who has successfully verified their identity and desires to make a purchase, will have the

option to pay for it through the web application. In the first place, the shipping address of the package will be requested. For this, the user must complete his/her address, postal code, country, province and city. Next, payment buttons will be displayed, and the user can choose the payment method between Sofort, Paypal or credit or debit card. If Paypal or Sofort is chosen, it will be necessary to log in to the application through a pop-up window, on the other hand, if making the payment with a card is desired, it must be completed the required data: card number, expiration date, security code, etc.

- **Use cases for admin users**
  - **UC.021 - Product details edition**: By using the "Update product" button in the product details page, website managers can change the name, description, image, further information, unit of sale, or price of any of the goods in the catalog.
  - **UC.022 - Order status management**: The website administrators will be able to change the status of an order placed by a specific user by selecting the order to update, choosing a state from the list of options in the picker and then pressing the save button to put the changes into effect.
  - **UC.023 - Overview of the list of the purchases made by all of the clients using the web application**: Administrators will be able to view the orders and details made by any user in the program.
  - **UC.024 - Add products to catalog**: Administrators will be able to add new goods to the catalog, including the following information: product name, description, price, category, image, supplementary product information, and unit of sale.

Chapter 5

Planning

In this section, the different phases into which the project has been divided and the time frame of the different tasks will be presented. In addition, the cost of the web application is detailed, taking as a reference the resources involved, both, human and technical.

5.1 Development planning

Once the overall analysis of the application was completed, as it is described on the section 4 of this document, the next step was to do a planning of how all the incorporated functionalities were intended to be addressed. For doing so, the project was divided into seven iterations, following an iterative and incremental development model. Each iteration will be an independent module in which new functionalities and improvements will be implemented. Finally, in the last iteration, a software that implements all the requirements is achieved.

To carry out the division of labour, it has been considered essential to complete firstly the functionalities that implied a greater risk, for example, those user stories that something else depends on, rather than the smallest ones, in case something unforeseen could arise at any point of the development, in such a way that there would be margin enough for problem solving. In addition, the development process for each one of the iterations consists of detailed analysis, design, implementation and testing, following a waterfall life cycle model, in order to rectify as soon as possible the errors detected during the implementation phase and avoiding a higher software cost or the unnecessary growth of development times.

Next, each of the iterations in which the project has been divided are detailed:

- **Iteration 0**: This iteration served as a learning stage, expending time for research and education into the key technologies that would be used in the application, in order to choose the most appropriate ones based on accurate criteria. For instance, at this time, the benefits and drawbacks of different payment gateways were considered, and Paypal was ultimately chosen over Redsys. Paypal provides easier implementation, more secure testing, and extremely thorough documentation, among other benefits, to developers. The end user's comfort was a key factor in the decision to use it, as it accepts all major credit and debit cards and is a well-known payment method that gives users confidence when making purchases. If the user has a Paypal account, they can pay without entering all of their card information each time they make a purchase. If they don't have an account, they can still enter their information in the same way as they would with a traditional virtual POS.

In addition to the above, as discussed in section 2 of this report, time was spent analyzing other e-commerce sites to make the best possible design, guaranteeing a good user experience. Additionally, alternative technologies were studied; some were incorporated in the project, such as Material UI or Bootstrap, which had been tested by reading the documentation and writing basic test cases. However, other technologies, were eventually abandoned owing to time constraints or non-priority use cases in the web site.

- **Iteration 1**: The objective of this first iteration was the development of use cases related to the user management in the web application, not only the back-end, but also in the front-end, and taking into account from the beginning the quality of the design and the look and feel. In addition, anticipating future work, a registry of users by roles was also left pre-established, to allow differentiation between client users and platform administrator users. The use cases developed during this iteration are the following ones:
  - UC.001: User sign up
  - UC.002: User login
  - UC.003: User logout
  - UC.004: User information update
  - UC.005: Change user password
  - UC.006: User contact with the administration through the contact form
- **Iteration 2**: In this second iteration, most representative use cases of an e-commerce page were developed. This iteration was the most complete of the entire development, since the use cases present in it are of vital importance for the development of the rest of the web application. The developed user cases are numbered below:
  - UC.007: View all the products from the catalogue.
  - UC.008: Filter products from the catalogue in categories (octopus, codfish, seafood)

  - UC.009: View product details
  - UC.010: Add products into the shopping cart
  - UC.011: Delete products from the shopping cart
  - UC.012: Update products from the shopping cart
  - UC.013: View products from the shopping cart
  - UC.014: Search for user orders
  - UC.015: View order details
  - UC.016: Download order ticket

- **Iteration 3**: During this iteration, only the use cases related to real online payments within the application have been developed. This use cases are detailed below:
  - UC.018: Pay an order with a real payment method
- **Iteration 4**: In this section of the development, an integration with OAuth2 has been implemented, both on the client side and on the server side, allowing the user, once this iteration is finished, to register and authenticate in the web application developed with a Google account. The use following use case was implemented during this iteration:
  - UC.020: User login with a Google account
- **Iteration 5**: During this development iteration, the most specific tasks of design and adaptation of the web page to the applicable law were developed. Additionally, the implementation of several static sites like a recipe display have been included in this iteration.
  - UC.025: About us page
  - UC.026: Privacy policy page
  - UC.027: Cookies policy page
  - UC.028: Shopping Terms and Conditions page
  - UC.029: View all the cooking recipes
  - UC.030: View cooking recipe details
- **Iteration 6**: In this iteration of the project development, the use cases contemplated are related to the administrative part of the web page, considering the front-end and back-end developments. Achieving once the iteration is finished, the existence of an administrator profile that manages the orders and the products of the catalog exposed on the web.

  - UC.021: Catalog product edition.
  - UC.022: Order status management.
  - UC.023: Overview of the list of the purchases made by all of the clients using the web application.
  - UC.024: Add products to catalog.

- **Iteration 7**: This iteration refers to the report's writing as well as the repair and enhancement of aspects discovered in the application during the review.

5.2 Temporary planning

The temporary planning procedures used at the time of development are described in depth in this section of the report. It should be noted that due to job obligations, several iterations have been prolonged over time while maintaining the predicted total number of hours.

The project's delivery date of September 2022 has been factored into the time estimate. Thinking about this, the project was planned, with each iteration being tried individually. The time frame for each iteration, as well as the estimations made at the start of each one, are shown below.

Iteration	Start date	End date	Estimation (h)	Real time (h)
Iteration 0	20/04/2022	30/04/2022	20	28
Iteration 1	01/05/2022	15/05/2022	55	69
Iteration 2	16/05/2022	20/06/2022	115	133
Iteration 3	21/06/2022	26/06/2022 12	35	27
Iteration 4	27/06/2022	10/07/2022	55	46
Iteration 5	11/07/2022	20/07/2022	35	37
Iteration 6	21/07/2022	14/08/2022	50	47
Iteration 7	15/08/2022	09/09/2022	65	72
TOTAL	20/04/2022	09/09/2022	430	459

Table 5.1: Temporary planning and real time spent

As can be observed from an examination of the findings displayed in the preceding table, the first development iteration was the one that deviated the most in relation to the expected time. This was owing to the fact that more time than anticipated had to be committed to the look and feel part of the application due to the lack of knowledge of the use of Bootstrap and the Material-UI libraries. Furthermore, the report writing process included a review of all the application's use cases, thus enhancements occurred that were built concurrently and prolonged the length of this last iteration. On the other hand, there were also some favorable transient variances, since a higher complexity was estimated for the Paypal API and OAuth, because they were new technologies that had not previously been worked with. It should also be noted that the first two iterations were the longest in terms of time, since they concentrated the highest number of use cases and the most risky ones, opting to organize it in this manner in order to handle any unanticipated occurrences early on rather than later where there may not have been enough time to correct the faults.

In conclusion, the most significant deviations from the plan occurred at the beginning of the project, so there was enough time to recover from these unforeseen occurrences. However, taking all of the periods into account as a whole, it can be determined that there were no substantial deviations that hampered the project's progress, given that negative deviations were somewhat offset by other positive ones.

5.3 Costs

By the time of calculating the cost of the project, human and technical resources used within the time frame covered by the project, have been taken into account. A summary of the different costs considered are detailed below:

5.3.1 Human resources

During the development of this web application, the collaboration of two human resources has been needed:

- **Project manager**: this role was carried out by the director of this final degree project, Fernando Bellas. He was in charge of defining the scope of the project and the planning, supervision and correction throughout the duration of the project. Therefore, taking into account the previous tasks, the overall amount of time spent by the project manager has been approximately 30 hours, divided between follow-up meetings and the correction of this final report. On the other hand, after a research on different employment social networks, we can conclude that the average cost of a senior project manager is approximately €45 per hour.

- **Developer**: the development work of the web application was carried out by the student, conducting the design, development, implementation, documentation and testing of each of the iterations of the project. In addition, the student also performs an analyst-developer role, studying the best technologies to be used to achieve each of the objectives set, looking for the greatest alternatives to resolve the needs raised and solving the problems that have arisen on the fly. Taking into consideration the work experience of the analyst-developer we will consider it as a junior developer with an average salary of €30 per hour.

5.3.2 Technical resources

The technical resources employed in this project are presented in the following points:

- **Computer**: the computer used for the development of this web application was an Asus Vivobook Pro with an Intel i7 processor and 32GB of RAM (Random Access Memory) memory, with a cost value of €1,500.00
- **Mouse**: for greater comfort and speed, a mouse with an approximate value of €12 has also been used.
- **Secondary screen**: a secondary 21.5 inches screen was used for the development, in addition to the one of the computer, having an approximate cost of €125.00
- **Software**: all the tools used for the development of this project were open-source. Although the development environment used was free due to the student license, if this project was developed for a commercial purpose, an annual license would cost €180.29 including VAT (Value Added Tax).

5.3.3 Total cost of the project

The hours invested in the web application by each of the human resources and a proportional part of the technical resources used, will be taken into account to calculate the total cost of the project.

- To calculate this percentage account, the depreciation method is going to be used. Following actions listed below, show how the maths behind this computation has been carried out:
  1. Since full time has not been allocated to the project, the first step is to calculate the number of hours required per week if the commitment were full-time:

$$1\text{ week } = (8 * 5) = \textbf{40 hours}$$

2. The number of weeks and months that the project would have covered in case of full time dedication is determined below.

$$\text{Temporary full-time project coverage} = (459/40) \approx \textbf{12 weeks} = \textbf{4 months}$$

3. For calculation the depreciation it is considered that the technological equipment will have a useful life of approximately 3 years (except for the license cost of the IDE used) and as the previous calculations reveal, that the full-time project lasted 4 months:

$$\text{Technical resources cost} = \frac{€180.29}{4} + \left(\frac{€125 + €12 + €1500}{3 \text{ years} * 52 \text{ weeks/year}} * 12 \text{ weeks}\right) \approx \textbf{€171.00}$$

- The developer's and the project manager's salaries are then determined:

$$\text{Project manager cost} = \ 30 \text{ hours} * 45€/\text{hour} = \textbf{€1350}$$

$$\text{Developer´s salary} = \ 459 \text{ hours} * 30€/\text{hour} = \textbf{€13770}$$

- Finally, the project's overall cost is calculated by summing the earlier monetary results:

$$\boxed{\textbf{Total cost} = \ €171.00 + €1350.00 + €13770.00 = \textbf{€15291.00}}$$

Chapter 6

Technological fundamentals

In this chapter we will set out the technologies, frameworks and libraries employed during the implementation of this project.

6.1 Technologies used in the back-end

- SQL (Structured Query Language) [4]: is a domain-specific language designed for storing, manipulating and retrieving data stored in relational database systems.
- JPQL (Java Persistence Query Language) [5]: is an object-oriented query language which instead of querying against the database model, uses the persistence cache to access entities kept in a relational database.
- **Java**: is a general-purpose, concurrent, strongly typed, object-oriented programming language. It was originally designed for embedded multi-platform network applications so it is extremely portable, thanks to JRE (Java Runtime Environment), a software layer that allows to execute source code in different environments.
- JWT [6]: is an open JSON-based standard, which allows to securely transmit signed information between the client and server. JWTs can be signed using a secret or a public/private key pair to ensure that the information remains unaltered once the token is issued.
- **Spring Boot** [7]: is an open source Java-based framework that allows to build standalone and production ready spring applications. With the help of a set of tools, Spring Boot aims to make it simple to build and configure Spring applications.
- **Hibernate** [8]: is an open source object relational mapping tool that provides a framework to speed up the mapping between object-oriented domain models and relational databases for web applications.

- **JUnit** [9]: is a free unit testing framework for Java programming language, used for write and execute automated tests.
- **Maven** [10]: is an open source project management tool based on POM (Project Object Model). It allows to automate the building process of a project, dependencies and documentation.
- **MySQL** [4]: is one of the most popular open source SQL database management systems developed, distributed, and supported by the Oracle Corporation.
- **Postman** [11]: is an Application Programming Interface (API) platform which simplifies the building, testing, update and the use of APIs by allowing to send HTTP (Hypertext Transfer Protocol) requests to a REST API.

6.2 Technologies used in the front-end

- CSS (Cascading Style Sheets) [12]: is a simple design language used for adding styles to a web page. It allows to separate the web page content from its visual representation.
- **JavaScript**: is an object-oriented interpreted computer programming language which implements dynamic and interactive web content, greatly improving user interaction within the web page. JavaScript is most known for being a web-based language, being native to the browser, one of the main reasons for which it is the most commonly used programming language in the world.
- HTML (Hyper Text Markup Language) [13]: is a standard markup language which allows the creation and structure of web pages, telling the browser how to display the content.
- **Bootstrap** [14]: is a free, open source front-end development framework for the creation of responsive and mobile-first websites. It includes HTML and CSS responsive templates for buttons, tables, modals, tables, etc. providing visual elements with preset style and behavior, in a way that speeds up the development process of websites.
- **React** [15]: is a JavaScript development library created by Facebook. This library provides an extensive, fast, declarative, flexible, and simple development by building UI as a tree of small pieces called components.
- **Redux** [16]: is an open source library that works on the front-end of JavaScript, employed to manage applications state. It helps React components to communicate with each other through an unidirectional data flow model.

- **Material - UI** [17]: is an open-source, front-end framework for React components that provides a simple, customizable, and accessible library of React components.
- **npm** [18]: is a package manager for the JavaScript programming language consisted of a command line client used to publish, discover, install, and develop node programs and dependencies.
- **Paypal Payments API** [19]: it is a REST API which allows to develop different type of payments use cases such as authorize payments, refund payments, show the payment information, etc.
- **EmailJS** [20]: is a service tool that allows to send emails directly from the JavaScript client-side, whithout the need to develop the backend.
- **OAuth2** [21]: is an authorization framework that provides users with a safe way to access online services without putting their credentials at risk. It delegates the user's authentication to the service that hosts the user's account and authorizes third-party applications to access said user's account.

6.3 Other technologies

- **LaTeX** [22]: is a high-quality free software package for typesetting documents.
- **IntelliJ IDEA** [23]: is an IDE (Integrated Development Environment) for developing Java programs and designed to maximize developer productivity providing tools such as code analysis, clever code completion, etc.
- **Git** [24]: is a version control system generally used for managing source code in software development. It allows to maintain a record of project changes and revert them into a specific point or version.
- **Dia** [25]: is a open source diagramming software.
- **Draw.io** [26]: is an open source diagramming tool for collaborative creation of diagrams such as flowcharts, wireframes, UML (Unified Modeling Language) diagrams, organizational charts, etc.
- **Overleaf** [27]: is a online, real time collaborative editor for documents written in the LaTeX markup language.

Chapter 7

Iterative development

The project's most important development-related issues are covered in depth in this section of the report. The structure of the web application will be presented first, followed by the implementation details on how each iteration was accomplished.

7.1 Application structure

7.1.1 Back-end structure

This subsection of the report deals with implementation issues of the back-end, made up of the data access layer, the business logic layer and the REST layer. Next, the structure used for this project is described in depth.

- **/src/main/java**: This project directory includes the majority of the source code for the web application, which manages the data access layer, the business logic layer, and the REST layer. This directory is then broken into folders, dividing the project into smaller components, as seen below.
  - **/model**: The source code for the data access layer and the business logic layer are found in this directory. These two levels denote the folder's subdivision, which will be discussed further below:
    - **/entities**: This folder houses the application data access layer. It is composed by the entities, which act as the building blocks and serve as the bases for gathering the data that will be reflected in the database system, and by the DAOs (Data Access Object), which provide the necessary methods for inserting, updating, deleting, and consulting information, achieving the separation in the application between the business logic part and the data access part.

        * **/exceptions**: The many exceptions issued by the back-end when some functionality does not behave as intended are saved in this directory, protecting the user from more serious issues.
        * **/services**: The business logic layer, which contains the data processing performed by the application, is located here.
    - **/oauth2**: This folder contains all of the source code related to a user's login and registration through Google, as mentioned in point 7.6 below.
    - **/rest**: This package contains the back-end's REST layer. It is built up of REST controllers and DTOs (Data Transfer Object), which allow information to be gathered from many sources and condensed into a single basic class.
    - **Application.java**: This application file has the purpose of serving as the program's starting point.

- **/src/main/resources**: The application.yml file in this package includes the various back-end configurations, such as the Google OAuth server data, the ports on which the application operates, the permissible redirect URLs, and so on. It also includes a subfolder containing the internationalization of the error messages displayed by the application in the languages Spanish, Galician, and English.

- **/src/test/java**: All tests completed on the back-end of the application are saved in this directory.

- **/src/sql**: The SQL directory contains the MySQL code for creating all of the tables required to store the application data, as well as the code for directly performing insert queries while building the application.

- **pom.xml**: The construction of the application back-end is managed by Maven, for this, it is necessary to define all the dependencies of the project in the POM file. When creating the program, this XML (Extensible Markup Language) file makes it simpler for the executable file to include everything it needs to function.

7.1.2 Front-end structure

- **/src**: The majority of the project's front-end source code is located in the /src folder. Below is a list of every sub-components.

    - **/backend**: The service access layer enabling REST API calls to the application back-end is contained in this folder.

  - **/constants**: The most frequently used constants are saved in a file in this folder during the front-end development process, promoting code simplicity and favoring organization.
  - **/i18n**: This directory includes internationalization files in Galician, Spanish, and English that allow the language to be adapted to the user's browser.
  - **/img**: Static pictures used in the application's user interface are saved here.
  - **/modules**: The modules section mostly includes the user interface code for the back-end use cases. This directory is separated into additional sub-directories (catalog, documents, receipts, shopping, data-Protection, and users) to facilitate proper organization. It also includes two more folders: app, which generates the application layout, and common, which builds the user interfaces for use cases common to multiple functionality.
  - **/store**: The primary Redux logic is located in this folder. The RootReducer it is also included in this directory, it is the application's standard reducer and imports all reducers by creating an object with all stateful properties.
  - **index.js**: This application directory is set for internationalization, which achieves language automation by extracting it from the user's browser. Additionally, it renders the App.js component of the front-end in the DOM tree.

- **/public**: This folder contains several application icons as well as an HTML code file including, among other things, executable script references and style sheets to be utilized in the application.

- **package.json**: The libraries and versions on which the front-end depends are listed in this file.

7.2 Data model

The many entities that are modeled in the relational database of the access layer to the server-side application data are shown in figure 7.1. Below are descriptions of each entity:

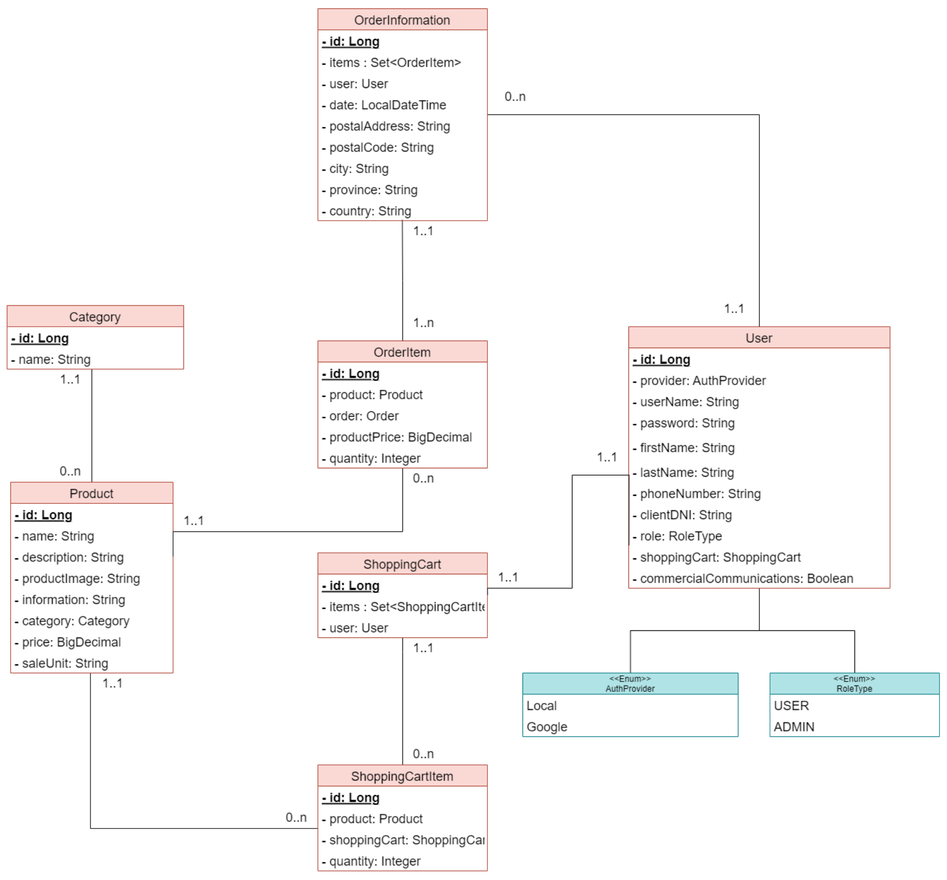


Figure 7.1: Data model diagram

In accordance to the data model in the preceding figure 7.1, a data dictionary is then displayed.

- **User**: This entity maintains the information of all users who register on the application, whether they are administrators, customers or anonymous users.
  - **id** (Long): Unique user registration key.
  - **provider** (AuthProvider): The kind of registration or login that the user choose to access the site.
  - **userName** (String): User's email
  - **password** (String): The application's user's login password

  - **firstName** (String): Stores the first name of the user
  - **lastName** (String): Stores the surname of the user
  - **comercialComunications** (Boolean): Parameter that specifies whether or not the user wishes to receive commercial communications through his email.
  - **phoneNumber** (String): Stores the phone number of the user.
  - **clientDNI** (String): Stores the national identity document number of the user.
  - **role** (RoleType): Stores the user membership on the app.

- **RoleType**: Enumeration that represents the user's membership on the app. It might be customer or administrator.
- **AuthProvider**: Enumeration that indicates the user's login or registration type on the app. It might be done locally or via Google.
- **Category**: Entity illustrating the several categories that a product registered in the system may fit into.
  - **id** (Long): Unique category registration key.
  - **name** (String): Stores the name of a category
- **Product**: Entity that stores the information about the products that make up the catalog.
  - **id** (Long): Unique product registration key.
  - **name** (String): Stores the name of a product.
  - **description** (String): Represents the description of a product.
  - **price** (BigDecimal): Stores the product price.
  - **category** (Category): A reference to the category associated to this product.
  - **productImage** (String): Holds the URL of the repository where the product image is stored.
  - **information** (String): represents the information relative to a product.
  - **saleUnit** (String): Contains the product's unit of sale; for instance, a pack of three cod loins.
- **ShoppingCart**: Entity that links each of the user's accounts to a shopping cart.
  - **id** (Long): Unique shopping cart registration key.
  - **user** (User): A reference to the used associated to this shopping cart.

- **ShoppingCartItem**: Entity that models a product and quantity added to the shopping cart.
  - **id** (Long): Unique shopping cart item registration key.
  - **product** (Product): A reference to the product associated to this shopping cart item.
  - **quantity** (Integer): Keeps track of how much of a product the consumer wants to buy.
  - **shoppingCart** (ShoppingCart): A reference to the shopping cart associated to this shopping cart item.
- **OrderInformation**: Entity that keeps track of the broad details of each of the user orders.
  - **id** (Long): Unique order registration key.
  - **user** (User): Unique identifier of the user placing the order.
  - **date** (Local Date Time): Keeps track of the day and time when a certain order was placed.
  - **postalAddress** (String): Territorial constituency to which a specific order must be delivered.
  - **postalCode** (String): Stores the postal code to which a given user´s purchase should be delivered.
  - **city** (String): It is saved the city to which a certain user's purchase should be delivered.
  - **province** (String): Keeps track of the province to which a certain order should be shipped.
  - **country** (String): Stores the country to which a specific order should be sent.
- **OrderItem**: This entity maintains details on each item that makes up an order.
  - **id** (Long): Unique order item registration key.
  - **product** (Product): A reference to the product associated to this order item.
  - **productPrice** (Big Decimal): Attribute that describes the cost that was paid for a product in a certain order.
  - **quantity** (Integer): Keeps track of how many units of a product the consumer has bought.
  - **order** (Order): A reference to the order cart associated to this order item.

7.3 Iteration 1

7.3.1 Analysis

The creation of the back-end directory structure indicated in point 7.1 above, serves as the initial stage in this iteration. Once the preceding stage was completed, the development began with the implementation of the use cases related to user management and the look and feel of all of them. Finally, once this iteration was accomplished, a completely functional application was obtained, that allowed the access to anonymous users, clients or administrators. The following use cases were addressed:

- UC.001: User sign up
- UC.002: User login
- UC.003: User logout
- UC.004: User information update
- UC.005: Change user password
- UC.006: User contact with the administration through the contact form

7.3.2 Design and implementation

The first step for the development of user management was to begin with the implementation of the back-end, in the first place, the necessary entities were established to record the information of the users who access the application and in the same way, the User table was created in database to store such data. JPA/Hibernate is used to implement the entities, which aids in connecting the database attributes to the entity ones, specified in the data access layer.

UC.001 - User sign up:
The user registry was created once the general features for any user management functionality were gained. Next, the DAOs were constructed. In most circumstances, the essential data may be acquired just by using naming conventions, as demonstrated in the following code, since the DAO provides the appropriate methods to query, insert, update, and remove the information.

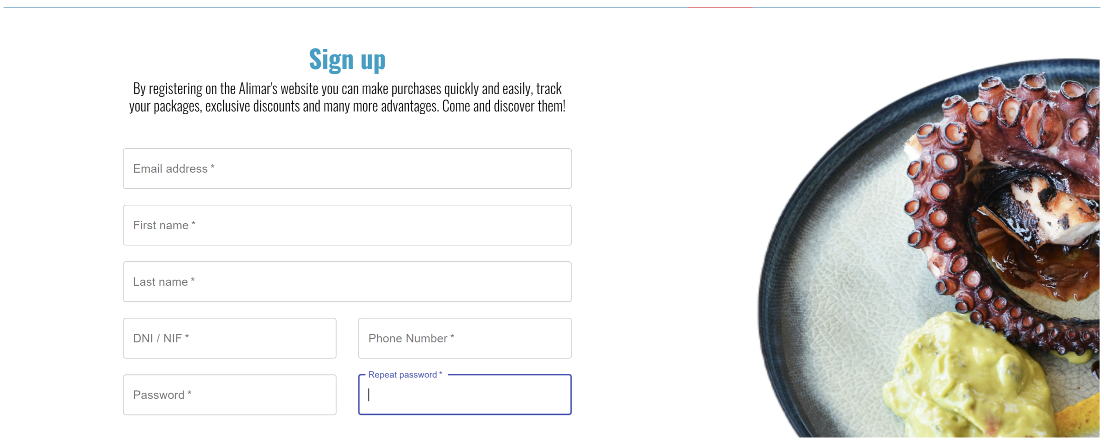
```
public interface UserDao extends
PagingAndSortingRepository<User, Long> {

  boolean existsByUserName(String userName);

  Optional<User> findByUserName(String userName);

  Optional<User> findById(Long id);

}

```

However, DAOs also enables the application to become independent of the database access method, guaranteeing that adjustments to database administration do not have an impact on the system as a whole.

Upon completion of this implementation, the service (sign Up) was implemented with the logic corresponding to the creation of the user. Since services need access to DAOs to read, update, insert, or delete information, dependency injection is used on services, annotating each DAO with @Autowired. Additionally, a @Service annotation is added to the newly constructed service so that controllers may access such implementations. On the other hand, noteworthy that DAOs shouldn't be annotated with @Repository because when utilizing Sping data they are automatically transformed into beans. Regarding how the service will really be provided, the reasoning followed for this functionality was first to check that the user does not already exist in the database, if so, the new user is created with all of the data; otherwise a DuplicatedUserException is thrown. Next, the REST controller and the DTOs were developed with a POST request for the creation of new users. Once the back-end is finished, the design of the front-end continues little by little, first a mock-up is created and finally progressing by generating the interface in the application. To do this, first of all, a post request is created that calls the back-end service. Following that, a function is constructed in which the components of the user interface are implemented, among them, a TextField from the Material-UI library, in which the data related to the name, surnames, email, ID, etc. phone number and password is input, as it is shown on the figure 7.2.

UC.002 - User login:

The usage of the standard JWT was selected for the implementation of user authentication, as in this way, the back-end does not have to maintain state between requests, being the client who supplies the token in each request that is made. To perform the authentication, a process very similar to the registration was carried out, creating a POST operation in the back-end and calling it from the front-end. In the user interface of this use case, the user will have to



Sign up

By registering on the Alimar's website you can make purchases quickly and easily, track your packages, exclusive discounts and many more advantages. Come and discover them!

Email address *

First name *

Last name *

DNI / NIF *

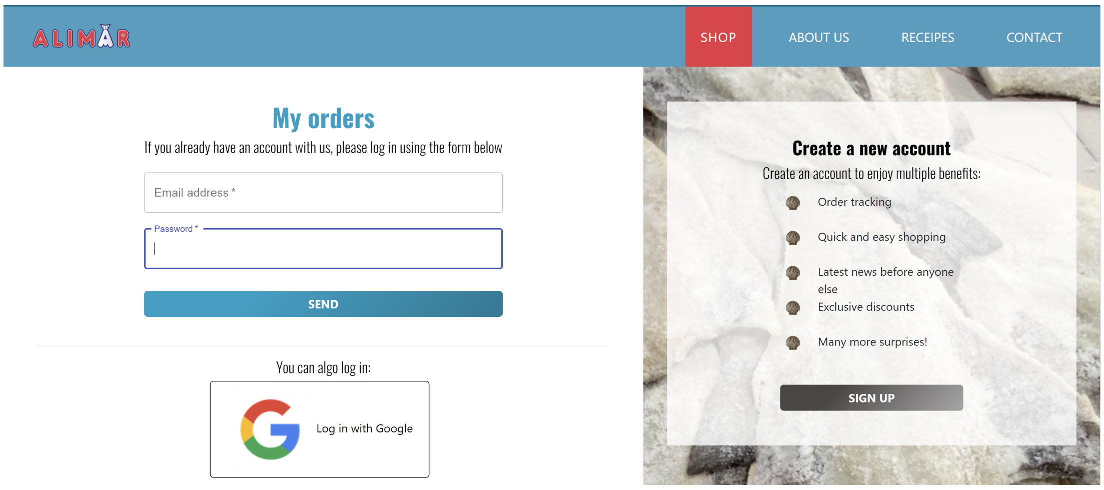
Phone Number *

Password *

Repeat password *

Figure 7.2: User case - User sign up

input their registered email and password. Next, the credentials will be compared to those stored in the database, if they are correct, a web token is generated and it will be stored in the local storage of the browser, enabling the user to avoid authenticating until after 24 hours, after that time, is when the session expires. A sequence diagram of this process is shown in figure 7.4. If the data does not match the information stored in the database, an exception will be thrown, warning the user of this error. In terms of the front-end, based again on the previous use case, two TextField type inputs were incorporated to allow the user to enter the data in the interface, as seen in the figure 7.3. It should be noted that parallel to the previous development, the front-end files that allow the rendering and design of the application, such as the Body, Header, Footer, etc., were also implemented.



My orders

If you already have an account with us, please log in using the form below

Email address *

Password *

You can algo log in:

Create a new account

Create an account to enjoy multiple benefits:

- Order tracking
- Quick and easy shopping
- Latest news before anyone else
- Exclusive discounts
- Many more surprises!

Figure 7.3: User case - User login

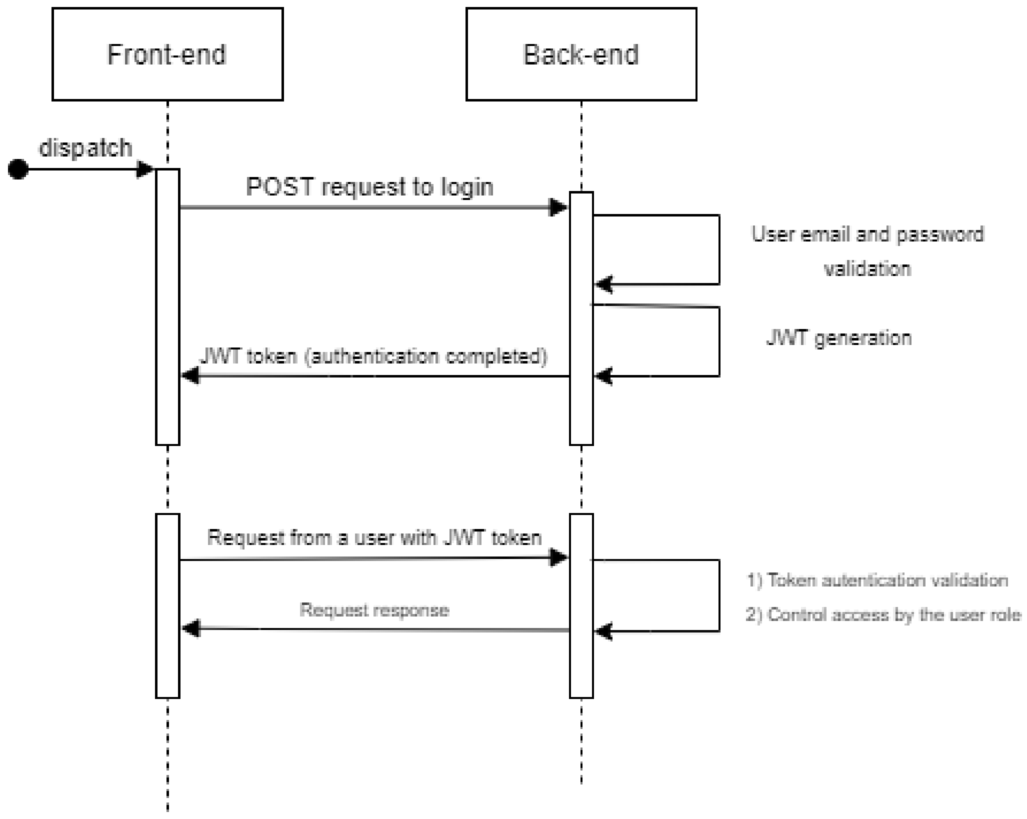
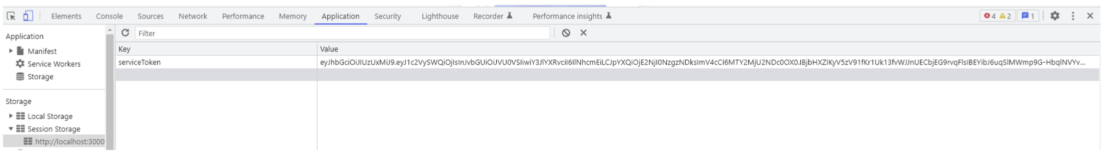

Front-end
Back-end
dispatch
POST request to login
User email and password validation
JWT generation
JWT token (authentication completed)
Request from a user with JWT token
1) Token autentication validation
2) Control access by the user role
Request response

Figure 7.4: Sequence diagram authorization with JWT

UC.003 - User logout:
Because the information is saved in a web token, as stated above in the login use case ?? and shown in figure 7.5, it must be deleted from local storage of the browser to achieve logout. This capability also does not introduce any new interface to the website other than the "Logout" button that is clicked to initiate the event.

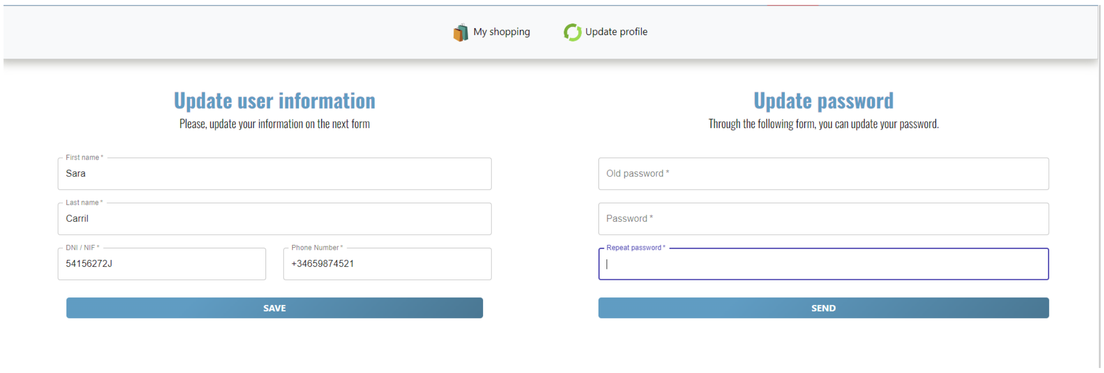


Key	Value
serviceToken	eyJhbGciOiJIUzUxMiJ9.eyJ1c2VySWQiOjIsInJvbGUiOiJVU0VSIiwiY3JlYXRvciI6IlNhcmEiLCJpYXQiOjE2NjI0NzgzNDksImV4cCI6MTY2MjU2NDc0OX0.IBjbHXZIKyV5zV91fKr1Uk13fvWJJnUECbjEG9rvqFlsIBEYibJ6uqSlMWmp9G-HbqlNVYv...

Figure 7.5: JWT login token on the browser local storage

UC.004 - User information update:
Being based on the principle of accuracy of the GDPR law, which establishes that the data gathered must be correct and duly updated, a new use case has been designed that allows the user to keep their data updated if some of them had changed over time. To construct this

use case, it was started by implementing a service that refreshes the data in the database. A PUT request is then made from the controller and called from the front-end to complete the request. In the front-end, a new page is developed that includes both, this functionality and the password change feature explained above, allowing the user to make all relevant adjustments from the same page, as shown in the figure 7.6. When changing their profile, the user can change their name, surnames, national identity document number and phone number by entering the new information in the TextField designated for that purpose.



Update user information

Please, update your information on the next form

First name *: Sara

Last name *: Carril

DNI / NIF *: 54156272J

Phone Number *: +34659874521

Update password

Through the following form, you can update your password.

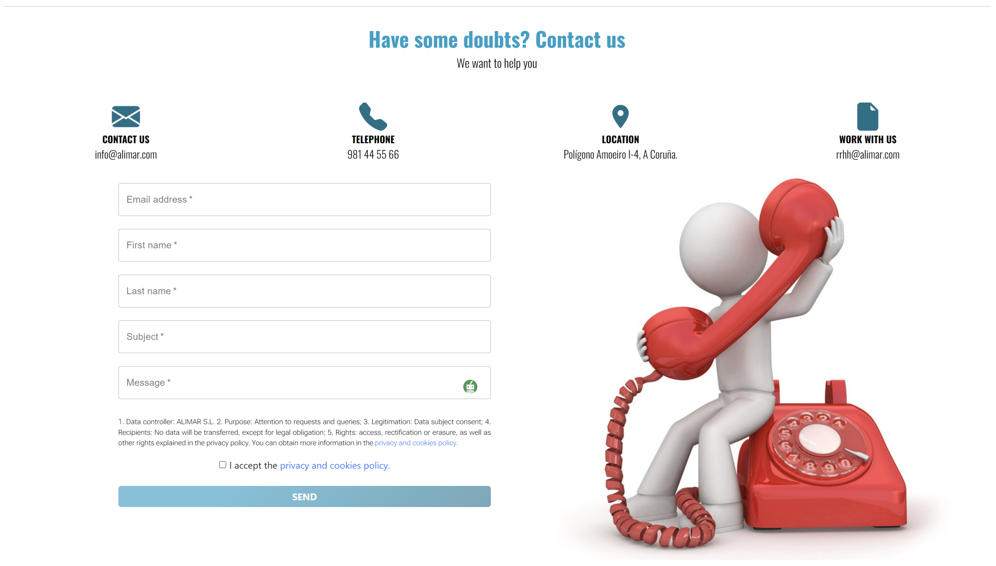
Old password *

Password *

Repeat password *

Figure 7.6: User case - User information update and change user password

UC.006: User contact with the administration through the contact form:
This use case entails allowing the user to contact the website's administrator in order to consult on doubts or other concerns. For its implementation, only the front-end part is required, since it is not a functionality that generates any data that needs to be stored in the database. The emailJS library was used to manage this use case, which required it to first access the website to construct a template, containing the format of the email to send and the variables to consider. Following that, an email is created in the front-end code by utilizing the service id, the template id and the user id, all of them provided among the emailJS website, finally is necessary to supply the variables to fill in the call. TextFields have been added to input the data, allowing the user to attach their name, surnames, email, email topic, and message, as seen in the image 7.7.



Have some doubts? Contact us

We want to help you

CONTACT US
info@alimar.com

TELEPHONE
981 44 55 66

LOCATION
Polígono Amoeiro I-4, A Coruña.

WORK WITH US
rrhh@alimar.com

Email address *

First name *

Last name *

Subject *

Message *

1. Data controller: ALIMAR S.L. 2. Purpose: Attention to requests and queries; 3. Legitimation: Data subject consent; 4. Recipients: No data will be transferred, except for legal obligation; 5. Rights: access, rectification or erasure, as well as other rights explained in the privacy policy. You can obtain more information in the privacy and cookies policy.

☐ I accept the privacy and cookies policy.

Figure 7.7: User case - User contact form

7.4 Iteration 2

7.4.1 Analysis

The goal of this second iteration is to incorporate all of the use cases of an e-commerce business, including the catalog, shopping cart and order management. All of this would result in a completely working e-commerce website, with the exception of payment administration, which would be handled in a future version. The following are the specific use cases:

- UC.007: View all the products from the catalogue.
- UC.008: Filter products from the catalogue in categories (octopus, codfish, seafood)
- UC.009: View product details
- UC.010: Add products into the shopping cart
- UC.011: Delete products from the shopping cart
- UC.012: Update products from the shopping cart
- UC.013: View products from the shopping cart
- UC.014: Search for user orders
- UC.015: View order details
- UC.016: Download order ticket

7.4.2 Design and implementation

All use cases except for the UC.016 follow an identical development to the one used in the first iteration. The shopping cart, categories, and product entities are first built. All of them have the annotation @Entity, allowing Hibernate to map each one of them to a database table. The associations between entities are also established by annotations; regarding this project, @ManyToOne, @OneToOne, and @OneToMany annotations have been used. The @JoinColumn annotation is used to provide the name of the column that represents the foreign key in both relationships marked with @OneToOne and @ManyToOne. Here is an illustration of how these annotations are used.

```
  @ManyToOne(optional=false, fetch=FetchType.LAZY)
@JoinColumn(name="shoppingCartId")
public ShoppingCart getShoppingCart() {
  return shoppingCart;
}

```

The services are then constructed utilizing dependency injection once more and the creation of the REST layer starts as the final back-end task. For its development, each application service's controller and DTOs are initially implemented. The development of the controllers makes use of a number of annotations, such as @RequestMapping to specify that all requests for the given route are processed in this class, @GetMapping or @PostMapping to specify that all requests for the given route go to the annotated method or @PathVariable to set a variable that arrives in the URL, however, it can only set basic data types like String or Long, and other less noticeable ones.

In contrast, when developing the front end, various functions are used for the retrieval and display of data in the user interface. First, selectors are used to retrieve the state of the components. In addition, there is a function called actions that responds to iterations on the user's page and enables the user to receive a response afterward. A reducer function is also present, and it runs the logic necessary to alter the state of the components. Finally, a function is developed to set the code and components for each of the aforementioned use cases, which make up the interface.

UC.016: Download order ticket:
This use case, unlike all the others implemented in this iteration, only requires frontend development. The @react-pdf/renderer library was utilized for the implementation. This library does not utilize standard HTML code to generate the components that will be included in the pdf, but instead is structured in several sorts of tags: a single document tag that has pages

within, and which in turn is made up of additional elements such as views or pictures. The internal structure of the document is initially built using these components:

```
    <Document>
        <Page size="A4" style={styles.page}>
            <Image src={Logo} style={styles.image}></Image>
            <View style={styles.section}>
                <Text>Número de pedido:</Text>
                <Text style={styles.sectionRight}>Fecha del pedido:</Text>
            </View>
            <View style={styles.section}>
                <Text>{list.id}</Text>
                <Text style={styles.sectionRight2}>{dateTime}</Text>
            </View>
            <Image src={qr_img} style={styles.image}></Image>
            <View style={styles.table}>
                <View style={[styles.row, styles.bold, styles.header]}>
                    <Text style={styles.row1}>Producto</Text>
                    <Text style={styles.row2}>Precio</Text>
                    <Text style={styles.row3}>Cantidad</Text>
                </View>
                {list.items.map(item =>
                    <View key={i} style={styles.row} wrap={false}>
                        <Text style={styles.row1}>
                            <Text style={styles.bold}>{item.productName}</Text>
                        </Text>
                        <Text style={styles.row2}>{item.productPrice} € </Text>
                        <Text style={styles.row3}>{item.quantity}</Text>
                    </View>
                )}
            </View>
            <View style={styles.totalPrice}>
                <Text style={styles.totalPriceTextPos}>Precio total</Text>
                <Text style={styles.totalPriceTextPos}>{list.totalPrice}</Text>
            </View>
            <Text style={styles.condicion}>Este ticket es imprescindible para cualquier cambio o devolución.
```

```
                        Puedes presentarlo en tu dispositivo móvil o imprimirlo.</Text>
            </Page>
        </Document>
```

Finally, to allow the user to download the ticket, a tag <PDFDownloadLink> is inserted in the function that produces the page's content, invoking the previously illustrated code. Below is the complete code that executes the button to perform said download.

```
    <PDFDownloadLink document={<OrderTicket list={order} />} fileName="Ticket"><button className="button-login-submit"><FormattedMessage id="project.global.buttons.descargarTicket"/></button>
    </PDFDownloadLink>
```

UC.013: View products from the shopping cart:
To display all of the catalog's items to users in the interface, a page separated into two primary views was chosen; first, the products and their information are displayed, using the <ShoppingCartItem> and <ShoppingItemList> functions; the second part of the page is a summary of the purchase to be made, indicating the ultimate price of the transaction and other information regarding the order. To do this, the product details are presented in a Table component with one line per product, while the summary is displayed in a container. Figure 7.8 depicts the many pieces that comprise the page:

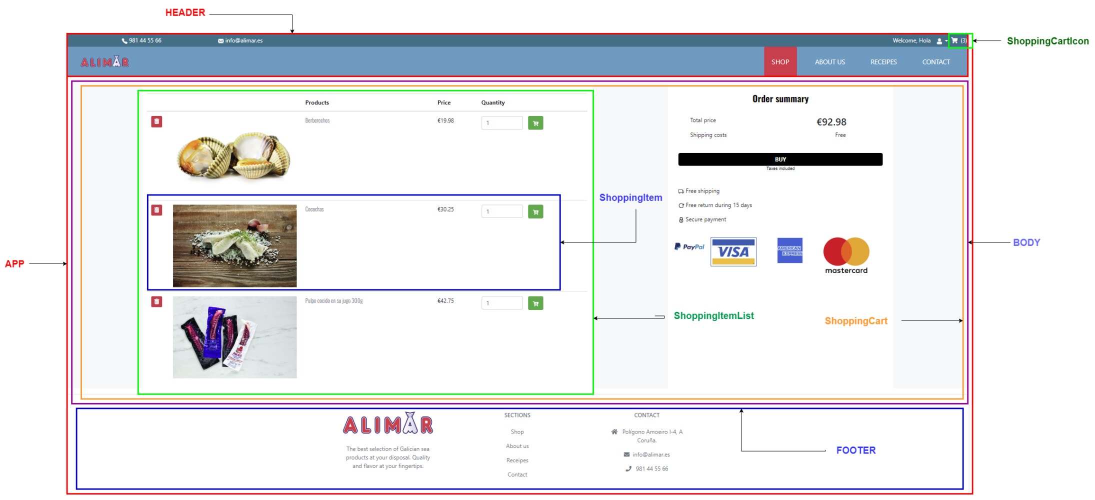


Figure 7.8: Shopping Cart page components

7.5 Iteration 3

7.5.1 Analysis

This iteration's objective is to complete the functionalities started in the previous iterations, so that the user can make an online payment, using a payment gateway. The following use case has been developed to meet this objectives:

- UC.018: Pay an order with a real payment method

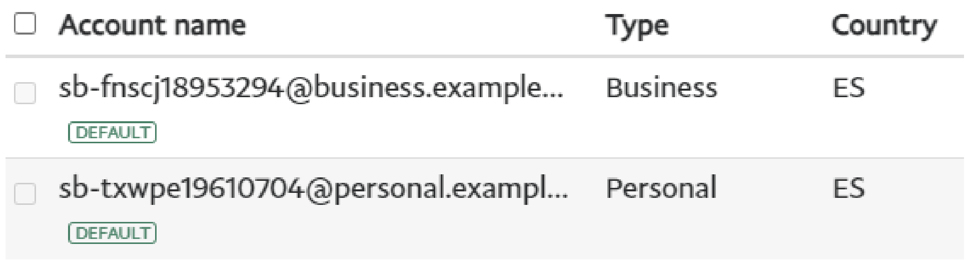
7.5.2 Design and implementation

The frontend of the application is where payments through Paypal via the website are managed. The first step was to create a developer account in Paypal Sandbox so that a necessary client-id could be obtained later, as well as two test users with bogus balances, one representing the firm and the other representing the customer, as shown in the figure 7.9. Following that, the code for the class into which the payment would be incorporated was written, and eventually, the payment gateway itself was created. To begin, the client id must be specified in the code in order to correlate the test accounts generated, with the platform payments. Additionally, the Paypal buttons must be placed, using the @paypal/react-paypal-js library for them. Within the same button, it must be specified the amount to pay, which in the case of this application is a variable value indicated by the 'totalPrice' attribute, and what behavior is desired for the successful response, in this case, an alert is sent to the user indicating that the payment was made correctly. However, in terms of the user experience while completing the payment, it might be done in a variety of ways. To begin, the user might want to use their PayPal account to make a payment. By clicking the Paypal button, a pop-up window will appear, in which the user must log in with their account and authorize the payment afterwards. Once this procedure is performed successfully, Paypal will provide an object stating that the functionality is complete. A user who does not have a Paypal account, on the other hand, can pay using a credit or debit card by providing the card number, security code, and expiration date, as well as other information about the user, such as billing address, name, etc.

7.6 Iteration 4

7.6.1 Analysis

This iteration's purpose is to allow users to register or log in to the app using their Google account. Once completed, the user who accesses the developed application will be able to select the "Log in with Google" button and register directly in the application by entering



Account name	Type	Country
sb-fnscj18953294@business.example... DEFAULT	Business	ES
sb-txwpe19610704@personal.exampl... DEFAULT	Personal	ES

Figure 7.9: Test users provided by Paypal

their credentials in the pop-up window that will appear, because Google will provide the user's data, including the name, surname, username, and email, if the user has specified them in his Google account. These are the use cases associated with this iteration:

- UC.020: User login with a Google account

7.6.2 Design and implementation

We started by generating a Google developer account for this iteration's implementation. In said account, a credential for OAuth authentication was produced, resulting in a client identity. Authorized redirect URLs, in this example 'http://localhost:8080/oauth2/callback/google,' are also noted. These URLs act as a security mechanism, allowing Google to clearly identify the person who is authenticating and preventing the services from supplanting one another. Next, the development begins by creating the AuthProvider class, to be able to detect what type of start the user has chosen. After that, the entity that retrieves the user information from the call returned by Google is developed and the services that contain the business logic of the user's start and registration are created. Also being consistent with a regular login, a functionality to update the user's information is incorporated, in case they modify something in their Google account, it could be modified also on this web application. In order to authenticate the user, further calls are performed between the client and the Google server. Web tokens are once again utilized for them. The process of the calls begins with a request from the built application to the Google server to access a user's resources. The user will then attempt to sign in using their Google credentials, and if successful, they will be granted access. The application then asks Google for an access token to confirm the user's authentication, ensuring that the authorized individual is who they claim to be. Finally, if everything has been properly developed, the server issues an access token, which is received from the application, granting the user access to the developed data.

7.7 Iteration 5

7.7.1 Analysis

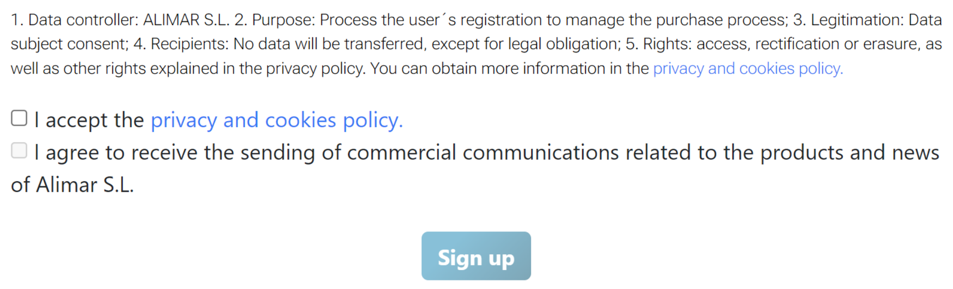
This iteration mostly comprised analysis and research, which included reviewing the many regulations pertaining to this form of electronic trade. The goal of this iteration is to create a web application that meets the legal standards of Spain, the nation in which the fake corporation is based. To accomplish total adaptation, many new pages were required to be developed, setting the requirements of access to the website and supplementing others, as outlined below. The following are the primary usage cases:

- UC.025: About us page
- UC.026: Privacy policy page
- UC.027: Cookies policy page
- UC.028: Shopping Terms and Conditions page
- UC.029: View all the cooking recipes
- UC.030: View cooking recipe details

7.7.2 Design and implementation

The front-end is responsible for the majority of the work in all use cases, as it is in control of for providing information to the user through the web interface. As previously stated, in addition to the use cases indicated above, modifications have been made on those sites where personal data is input, such as the contact form and the registration form, to comply with the obligation of information and information by layers, as it is demonstrated in the figure 7.10. This entails informing all users of the time of data conservation, the purpose for which they are gathered, the legal basis, the forecast or non-prediction of data transfer, the name of the person in charge of the treatment, and a reference to the exercise of their rights. To do this, it is necessary to disclose this information by layers in a clear and unambiguous manner that the user can view at a glance without having to enter another tab.

UC.026: Privacy policy page:
The goal of this new application page is to inform the user about the methods and practices employed on the page, giving them confidence in the treatment of their data. To meet the goals of this page, the data of the person responsible for the treatment must be established in it, in this instance fictitiously established in point 1, as indicated in the figure 7.11. The user must


1. Data controller: ALIMAR S.L. 2. Purpose: Process the user´s registration to manage the purchase process; 3. Legitimation: Data subject consent; 4. Recipients: No data will be transferred, except for legal obligation; 5. Rights: access, rectification or erasure, as well as other rights explained in the privacy policy. You can obtain more information in the privacy and cookies policy.

☐ I accept the privacy and cookies policy.
☐ I agree to receive the sending of commercial communications related to the products and news of Alimar S.L.

Figure 7.10: Layered information

also be informed of the purpose of the use of their data and legitimacy. In the case of this web application, the legitimization is the execution of a contract since in order to sell the objects it is necessary to know who they are sold to, where they should be shipped, etc. Furthermore, the goal varies based on the data involved; in the case of the purchase, the objective is contract execution, but in the case of the contact form, the purpose is to address the user's worries; all of this information is established in point 2 of the image 7.11. In addition, the user must be informed if transfers to other firms or nations are planned; in this situation, no transfers are planned because it is a single company. Finally, point 4 of the picture 7.11 states that the user must be notified of how to use their rights to correction, access, and cancellation of authorization.

UC.027: Cookies policy page:
A new page in the back-end rendered from App.js was constructed to incorporate the privacy policy and cookies. The goal of this page is to inform users about the use of cookies, what they are, their purpose, duration, and how to deactivate them, so that they can consent to their installation or not, knowing what they consist of and the type of personal data they will capture, record, and transmit to their owner. Cookies are specifically used on this website to log in with Google credentials using OAuth2, therefore this information is established in the privacy and cookies policy, as the picture 7.12 shows. Furthermore, the privacy and cookies policy must be available from any portion of the website, which is why the links have been placed in the footer.

UC.028: Shopping Terms and Conditions page:
Finally, it is recommended that the terms and conditions of purchase be established on the page in order to define the purchase circumstances under which the Products are sold, which will comprise the contract to be signed between the Seller and the User. Unlike the other papers, there are no legislative requirements for necessary material to be established in this document; nonetheless, it must fulfill legal requirements while avoiding abusive terms. For

example, the scope of application, prices, and delivery conditions could be indicated, such as whether the prices include VAT or not, how long shipments are expected to take, what payment methods can be used on the web, and returns, which legally require a period of 14 days, but there is no legal obligation to accept returns in the case of this sector, when selling perishable food products. In conclusion, the paper defines the purchasing regulations to safeguard both the supplier and the user.

7.8 Iteration 6

7.8.1 Analysis

This iteration intends to provide use cases that let the administrator control the web application's purchases and items. By the end of this iteration it is aimed to provide an administrator-only page where they may add and update goods and change the status of orders. The use cases that are included are shown below.

- UC.021: Catalog product edition.
- UC.022: Order status management.
- UC.023: Overview of the list of the purchases made by all of the clients using the web application.
- UC.024: Add products to catalog.

7.8.2 Design and implementation

A development order similar to that described in the sections 7.3 and 7.4 was followed for the implementation of the aforementioned use cases. Since the entities and DTOs required for these use cases had previously been created in earlier stages, services were created first.

UC.021: Catalog product edition:
The page produced for the visualization of the product information has been utilized to generate the front-end of this use case, hiding or revealing items depending on the user's membership. To carry out the use case in question, the user will access the details of the product to update, as it is shown in figure 7.13; after, a form will be presented with the product's current information will be presented, so that the administrator could modify necessary data, among the information that can be modified are: the product's name, description, image, information, unit of sale, and price.

7.9 Iteration 7

This last iteration of the end-of-degree project's major goal is the construction of this memory; however, throughout the writing process, all of the application's pages were inspected and minor problems or improvements were identified and addressed, as shown below.

- **Page's appearance**: The login page's appearance has been changed because it was poorly organised and lacked a responsive design. In addition, payment management has also been improved, which means that after a user inputs shipping information and advances to payment, they cannot change them since, for example, if we added shipping expenses, they would not be managed effectively.
- **Success message**: A success message has been incorporated to inform the user when the use case UC.006 - User contact with the administration through the contact form, is completed and the message is sent correctly.
- **Design changes**: Minor design changes to the web app to improve its appearance and make it more realistic.
- **Error in the information displayed**: An error detected on the purchase page has been modified, when displaying the user information, if the user had registered through Google, the name and surnames appeared in the name field, leaving the surname field empty.

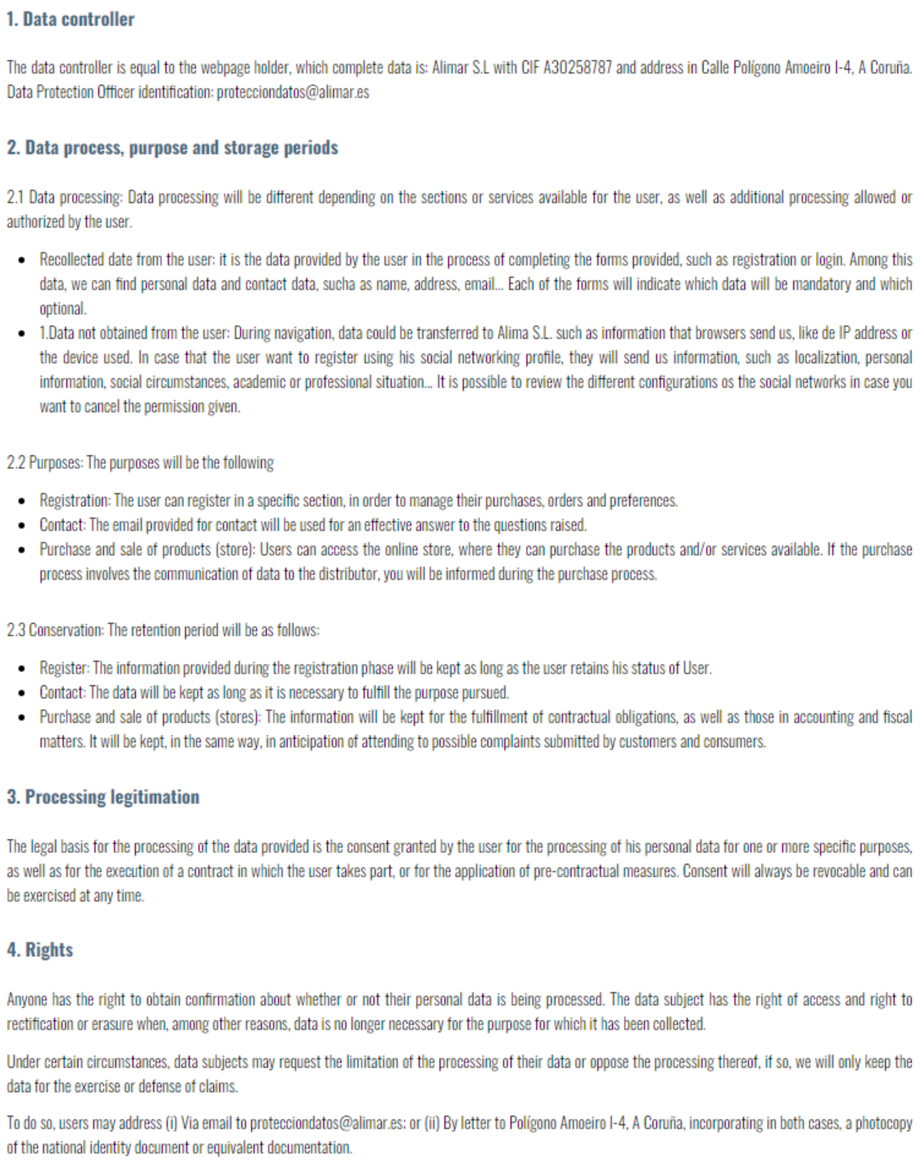
7.10 Tests

Throughout the project's development, tests were performed on all components, allowing to discover and solve problems before they had a detrimental influence on the project. Several sorts of testing have been performed, depending on the tested components.

- **Automated tests**: Backend components, particularly entities and services, were subjected to unit testing. The JUnit tool was used for this, which provides a set of libraries used in programming to do unit tests in Java applications. Positive tests, which strive to check that the software works properly, as well as negative tests, which attempt to test error instances and verify that the program throws the right exceptions, have been performed, as the one it is shown above. For this, @Transactional-annotated test classes were constructed, ensuring that each of the test methods is performed in a transaction and they end with a rollback, reversing changes and preventing data contamination between tests and other information kept in the system. It should be mentioned that the tests are carried out on an auxiliary database dedicated just to the tests in order to achieve the same goal.

```
  @Test
public void testSignUpNONValidEmail() throws
  DuplicateInstanceException, InstanceNotFoundException,
  IncorrectEmailException {

  User user = createUser();
  user.setUserName("sara");

  assertThrows(IncorrectEmailException.class, () ->
  userService.signUp(user));

}
```

- **Acceptance Tests**: After development was completed, acceptance tests were performed to ensure that the application meet all of the specified criteria. These tests were carried out in the presence of the developer and the project teacher, in such a way that the project was validated, albeit some modifications for the implementation were offered.



1. Data controller

The data controller is equal to the webpage holder, which complete data is: Alimar S.L with CIF A30258787 and address in Calle Polígono Amoeiro I-4, A Coruña. Data Protection Officer identification: protecciondatos@alimar.es

2. Data process, purpose and storage periods

2.1 Data processing: Data processing will be different depending on the sections or services available for the user, as well as additional processing allowed or authorized by the user.

- Recollected date from the user: it is the data provided by the user in the process of completing the forms provided, such as registration or login. Among this data, we can find personal data and contact data, sucha as name, address, email... Each of the forms will indicate which data will be mandatory and which optional.
- 1.Data not obtained from the user: During navigation, data could be transferred to Alima S.L. such as information that browsers send us, like de IP address or the device used. In case that the user want to register using his social networking profile, they will send us information, such as localization, personal information, social circumstances, academic or professional situation... It is possible to review the different configurations os the social networks in case you want to cancel the permission given.

2.2 Purposes: The purposes will be the following

- Registration: The user can register in a specific section, in order to manage their purchases, orders and preferences.
- Contact: The email provided for contact will be used for an effective answer to the questions raised.
- Purchase and sale of products (store): Users can access the online store, where they can purchase the products and/or services available. If the purchase process involves the communication of data to the distributor, you will be informed during the purchase process.

2.3 Conservation: The retention period will be as follows:

- Register: The information provided during the registration phase will be kept as long as the user retains his status of User.
- Contact: The data will be kept as long as it is necessary to fulfill the purpose pursued.
- Purchase and sale of products (stores): The information will be kept for the fulfillment of contractual obligations, as well as those in accounting and fiscal matters. It will be kept, in the same way, in anticipation of attending to possible complaints submitted by customers and consumers.

3. Processing legitimation

The legal basis for the processing of the data provided is the consent granted by the user for the processing of his personal data for one or more specific purposes, as well as for the execution of a contract in which the user takes part, or for the application of pre-contractual measures. Consent will always be revocable and can be exercised at any time.

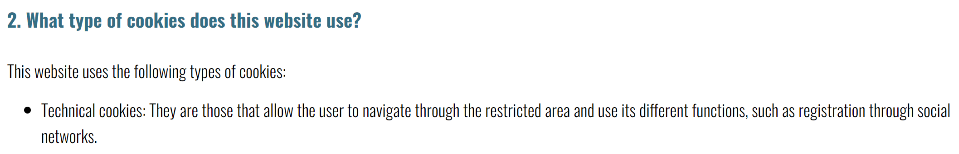
4. Rights

Anyone has the right to obtain confirmation about whether or not their personal data is being processed. The data subject has the right of access and right to rectification or erasure when, among other reasons, data is no longer necessary for the purpose for which it has been collected.

Under certain circumstances, data subjects may request the limitation of the processing of their data or oppose the processing thereof, if so, we will only keep the data for the exercise or defense of claims.

To do so, users may address (i) Via email to protecciondatos@alimar.es; or (ii) By letter to Polígono Amoeiro I-4, A Coruña, incorporating in both cases, a photocopy of the national identity document or equivalent documentation.

Figure 7.11: Privacy policy page

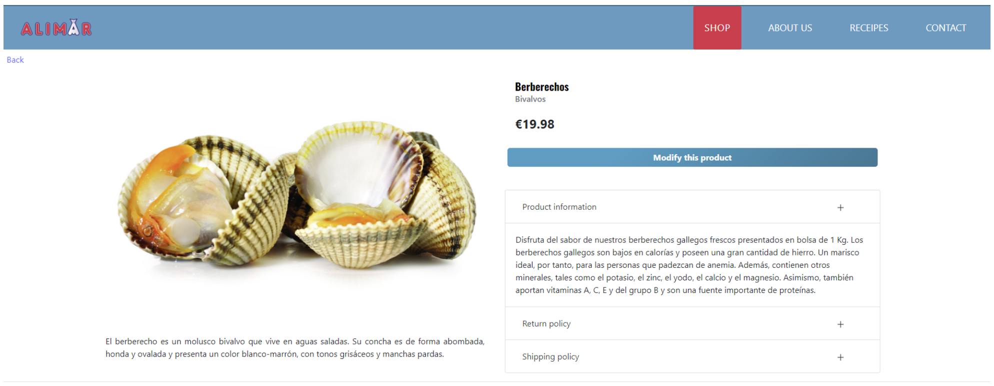


2. What type of cookies does this website use?

This website uses the following types of cookies:

- Technical cookies: They are those that allow the user to navigate through the restricted area and use its different functions, such as registration through social networks.

Figure 7.12: Cookies policy



Berberechos
Bivalvos

€19.98

Disfruta del sabor de nuestros berberechos gallegos frescos presentados en bolsa de 1 Kg. Los berberechos gallegos son bajos en calorías y poseen una gran cantidad de hierro. Un marisco ideal, por tanto, para las personas que padezcan de anemia. Además, contienen otros minerales, tales como el potasio, el zinc, el yodo, el calcio y el magnesio. Asimismo, también aportan vitaminas A, C, E y del grupo B y son una fuente importante de proteínas.

El berberecho es un molusco bivalvo que vive en aguas saladas. Su concha es de forma abombada, honda y ovalada y presenta un color blanco-marrón, con tonos grisáceos y manchas pardas.

Figure 7.13: Use case - Catalog product edition

Chapter 8

Conclusions and future work

This report's concluding chapter recapitulates the project's current state and offers the author's perspective on the work and the findings done. New areas of study and advancements are also suggested for the future.

8.1 Conclusions

The goals for this project may be conclusively determined to have been achieved, after the web application created for it has been finished. An online store that supports all use cases for product display, purchases, and administration has been developed, fulfilling the objectives established in the point 1.2 of this report.

The primary goal was to create a web application that was as similar to a genuine application as feasible. Technologies like Bootstrap or Material-UI were employed for this, which at first gave some worries but after understanding how to use them correctly, substantially assisted in achieving this aim fast and effectively.
In addition, when selecting this subject for the final degree project, the non-functional goals included learning mostly about payment gateways, because it is a topic that has not been examined during the degree and it would be interesting to acquire this knowledge in light of potential work chances down the road. The same happened with the adaptation of the application to the current legislation, although in this case, some brushstrokes were learned in one subject of the degree.

On the other hand, in terms of knowledge gained throughout this project, two categories of lessons may be distinguished: professional and personal. In the academic setting, it has been learned how to create an entire application, taking into account the analysis, design, implementation, and testing of all iterations, the use of a development methodology, and expertise in new technologies that were previously unheard of, such as Bootstrap, Paypal, OAuth2, etc. On the other side, personal development has been made in areas such as stress management, organization, and problem solving, aptitudes that are crucial for the future.

In conclusion, in light of the fact that this activity has created essential building blocks upon which to build future endeavors towards even greater professional and personal development, it can be said that it has been extremely enriching and productive.

8.2 Future work

Finally, apart from the fact that the objectives determined for said web application have been met, there is still room for improvement, being able to make a more attractive application with more value for the company in the future. Some of the functionalities outlined below were originally intended for development within the framework of this end-of-degree project, although they were finally discarded because they did not fulfil with the recommended extension or because there were other functionalities that gave the web application more value and a greater experience for the user.

- **Relative to user´s use cases**: In relation to the use cases related to user management, functionalities which allow the user to easily recover from errors could be implemented, such as a password recovery option or a show/hide password button. In addition, new register or login integration could be made with other companies like Facebook, Twitter, GitHub, etc. in case the user does not have a Google account.

- **Web application administration improvements**: The part related to the administration of the website has been minimally implemented in this project, a multitude of new functionalities could be incorporated, that would help the commercial and managerial part of the company. For instance, representative graphics could be incorporate to this part, representing the countries in which the product is best sold, which products are the most visited by customers, which are the months in which it is sold the most, which products are the ones that each client buys the most, etc. With this data, material acquisition forecasts could be improved, prognosis of future sales would be more accurate, staff could be hired in advance for campaigns, etc. All in all, it would greatly help to improve the productivity of companies.

  In addition, this website is adapted to current Spanish legislation regarding Law 34/2002, of July 11, on services of the information society and electronic commerce and other applicable laws, so that when registering an user, he can grant or not his permission to receive commercial communications. Exploiting this acceptance could be a new functionality of the website; sending personalized advertising to customers who have accept it, based on the products they consume the most by email, would favour the growth of the company´s sales.

- **Catalog visibility improvements**: Regarding the product catalog, improvements could be implemented to offer a greater experience to the user when searching for products, for example, adding product subcategories, or a keyword search engine. What is more, allowing users to add products to favorites, so they can easily find them in the future is another functionality that would add value to the user.

- **Improvements in payment methods**: Last but not least among the improvements designed for this application, are those related to payments and orders. On the one hand, a new functionality could be added to allow discount codes when doing the payment of the order. In addition, a very important improvement is payment security. Currently, payments are only managed through the front-end, trusting the object that Paypal returns when making the API call when the payment is executed. However, this call could be fake or someone could be impersonating another one else, so embedding web tokens when communicating between the back-end and the front-end would be highly recommended to verify that they are who they say. However, it is true that Paypal guarantees the protection of the buyer against payments, by not having to provide bank details, only an email and a password would be enough to make the payment. In addition, to minimize the inconvenience of the user having to input the shipping address again, it would also be useful to exchange with Paypal certain data, such as the shipping address that is already obtained in the application.

- **Order management improvements**: First of all, a new functionality added to the web application could be filters to organize the orders according to their status or other relevant characteristics. What is more, the status of the orders could be automated, through integration with the transport company, in such a way that once the order is collected in the seller company, the order state is updated automatically, for instance with the following status ”In transit”, ”Delivered”, etc. In this way, human errors would be avoided and the user would be informed in real time of the status of his order.

Appendices

Appendix A

Build and run instructions

At this point, the instructions for compiling and running the project are specified, since it is not published on any website.

It should be noted that the following software must be installed before to compilation and execution: Maven, Java JDK and MySQL are all included. Additionally, tables must be created in MySQL for both the primary database, which houses the application data, and the database used for testing; above is shown how to do it:

```
mysqladmin -u root create tfgproject -p
mysqladmin -u root create tfgprojecttest -p

mysql -u root -p
    CREATE USER 'tfg'@'localhost' IDENTIFIED BY 'tfg';
    GRANT ALL PRIVILEGES ON tfgproject.* to 'tfg'@'localhost' WITH GRANT OPTION;
    GRANT ALL PRIVILEGES ON tfgprojecttest.* to 'tfg'@'localhost' WITH GRANT OPTION;
    exit
```

The project can be assembled and carried out when the aforementioned procedures have been completed. The following actions must be taken in order to accomplish this:

- 1. Launch a terminal under the back-end's root project, where the pom.xml file is located (../backend). If this is the first time the application has been compiled, it is necessary to first run the **mvn sql:execute** command in order to create the tables in the database. The **mvn spring-boot:run** command must then be executed. Explain that just this last command will be required if the application has already been started on earlier occasions.

- 2. The front-end of the application will be launched in this second stage; if this is the first time the application has been launched, it is necessary to navigate to the root folder

(../frontend) and execute **npm install** to ensure that all the dependencies set up in the project are installed. The application will then be opened on port 3000 by the next command, **npm start**. Make it clear that if the program has previously been launched on prior instances, just this last command will be necessary.

List of Acronyms

API Application Programming Interface. 3, 29, 30, 32, 57

CSS Cascading Style Sheets. 29

DAO Data Access Object. 31, 37, 38

DOM Document Object Model. 3, 33

DTO Data Transfer Object. 32, 38, 43, 50

ECISSA E-Commerce and Information Society Services. 3

GDPR General Data Protection Regulation. 2, 40

HTML Hyper Text Markup Language. 29, 33

HTTP Hypertext Transfer Protocol. 29

IDE Integrated Development Environment. 27, 30

JPQL Java Persistence Query Language. 28

JRE Java Runtime Environment. 28

JSON JavaScript Object Notation. 28

JWT JSON Web Token. iv, 15, 28, 38, 40

POM Project Object Model. 29, 32

POS Point of Sales. 5, 22

RAM Random Access Memory. 26

Glossary

back-end It is a component of an application responsible for controlling both business logic and data. 3

bean Reusable software that eliminates the need to program each component independently. 38

front-end It is the client-side part of the application that interacts with users. 3

Mock-up It is a photo montage in which user interface concepts for use cases can be presented before they are implemented.. iv, 16

Bibliography

[1] "Pescanova website," last accessed August 2022. [Online]. Available: https://www.pescanova.es/

[2] "Mariscos campelo online store," last accessed August 2022. [Online]. Available: https://mariscoscampelo.com/

[3] "Palacio de oriente website," last accessed August 2022. [Online]. Available: https://www.palaciodeoriente.net/es

[4] "Mysql documentation," last accessed August 2022. [Online]. Available: https://dev.mysql.com/doc/

[5] "Jpql documentation," last accessed June 2022. [Online]. Available: https://docs.oracle.com/html/E13946_04/ejb3_langref.html

[6] "Jwt libraries and documentation," last accessed August 2022. [Online]. Available: https://jwt.io/

[7] "Spring boot documentation," last accessed July 2022. [Online]. Available: https://spring.io/projects/spring-boot/

[8] "Hibernate documentation," last accessed August 2022. [Online]. Available: https://hibernate.org/

[9] "Junit5 user guide," last accessed September 2022. [Online]. Available: https://junit.org/junit5/docs/current/user-guide/

[10] "Maven documentation," last accessed August 2022. [Online]. Available: https://maven.apache.org/

[11] "Postman website," last accessed March 2022. [Online]. Available: https://www.postman.com/

[12] "Css reference," last accessed May 2022. [Online]. Available: https://devdocs.io/css/

[13] "Html guide," last accessed May 2022. [Online]. Available: https://devdocs.io/html/

[14] "Bootstrap documentation," last accessed August 2022. [Online]. Available: https://getbootstrap.com/docs/4.6/getting-started/introduction/

[15] "React webpage," last accessed August 2022. [Online]. Available: https://es.reactjs.org/

[16] "Redux documentation," last accessed August 2022. [Online]. Available: https://es.redux.js.org/

[17] "Material - ui components," last accessed August 2022. [Online]. Available: https://mui.com/

[18] "Npm website," last accessed August 2022. [Online]. Available: https://www.npmjs.com/

[19] "Paypal developer website," last accessed August 2022. [Online]. Available: https://developer.paypal.com/

[20] "Emailjs developer website," last accessed August 2022. [Online]. Available: https://www.emailjs.com/

[21] "Oauth developer website," last accessed August 2022. [Online]. Available: https://oauth.net/2/

[22] "Latex documentation website," last accessed September 2022. [Online]. Available: https://www.latex-project.org/

[23] "Intellij documentation," last accessed May 2022. [Online]. Available: https://www.jetbrains.com/es-es/idea/

[24] "Git website," last accessed August 2022. [Online]. Available: https://git-scm.com/

[25] "Dia installer website," last accessed September 2022. [Online]. Available: http://dia-installer.de/

[26] "Draw.io working area website," last accessed September 2022. [Online]. Available: https://www.draw.io/

[27] "Overleaft text editing website," last accessed September 2022. [Online]. Available: https://www.overleaf.com/