• U C •

Thesis
Master in Informatics Engineering
Software Engineering

Solving an everyday problem: A web platform

Micael Pereira dos Reis

Thesis Advisors
Sara João
António Leitão

Department of Informatics Engineering
Faculty of Sciences and Technology
University of Coimbra

July 2016

Thesis
Master in Informatics Engineering
Software Engineering

Solving an everyday problem: A web platform

Micael Pereira dos Reis

Thesis Advisors
Sara João
António Leitão

Jury Members
Professor Mário Rela
Professor Álvaro Rocha

Department of Informatics Engineering
Faculty of Sciences and Technology
University of Coimbra

July 2016

Abstract

Nowadays, society enjoys and values when everything is delivered with minimal effort, without the need for people to leave the comfort of their homes. For this reason, home delivery services have been growing, and now a large set of businesses offer this option. This growth gave room for the idea behind this project. It consists on the development of a web platform for HYP, that works as an platform where bakeries can subscribe to show and sell their products to the users. The users can then search for bakeries and their products, and place orders. Afterwards, the products are delivered to their homes in a date that they have chosen.

The devolopment of this project was divided into a few phases, starting by a study of related work. In this phase, the market competitors, web technologies and usability techniques were studied and analyzed. Then, the planning for the whole project was made, the methodology to be used was chosen as well as the web technologies and the productivity tools. After this phase, all requirements were defined together with HYP, and the system architecture as well as the other artifacts that constitute the system were described. When this last phase was completed, the development of the application started, by doing the initial configurations and implementing the functionalities and everything else needed for the platform to work properly. Finally, tests were performed to check the functionalities that needed fixing or improvements, the security of the platform as well as the usability of it.

This document details the phases of this project, the architectural choices and others details relevant to the development of the platform.

Keywords: HYP, Web Platform, Software Engineering, Development, Tests, Home Delivery, Bakery, Commerce

Resumo

Cada vez mais a sociedade prefere e valoriza quando tudo lhes é entregue pessoalmente sem que tenham de fazer qualquer tipo de esforço, e sem que tenham de sair do conforto das suas casas. Devido a isto, os serviços de entrega ao domicílio têm vindo a crescer, e agora, muitos negócios já integraram este tipo de serviço. Este crescimento fez com que a ideia por de trás deste projeto surgisse. O projeto consiste no desenvolvimento de uma plataforma web para a HYP. Esta irá funcionar como uma loja online onde padarias podem subscrever ao serviço de forma a poderem vender e mostrar os seus produtos aos utilizadores. Estes podem procurar por produtos e padarias e fazer encomendas dos produtos que desejam. As encomendas serão depois entregues nas suas casas nas datas que pretenderem.

O desenvolvimento deste projeto foi dividido em diferentes fases, tendo começado com o estudo de tudo o que está relacionado com o projeto. Nesta fase, foram estudados os possíveis rivais neste mercado bem como as tecnologias web e técnicas de usabilidade usadas atualmente. Posteriormente, o planeamento do projeto foi realizado, tendo sido definido a metodologia a ser aplicada e escolhidas as tecnologias e ferramentas de produtividade a serem usadas. Depois desta fase, os requisitos foram definidos em conjunto com a HYP, a arquitetura do sistema foi descrita juntamente com todos os outros artefactos que constituem o sistema. Depois desta fases iniciais estarem completas, o desenvolvimento da aplicação foi iniciado. Primeiramente, as configurações inicias necessárias foram executadas e após isto deu-se inicio á implementação das funcionalidades e todo o resto necessário para que a plataforma funcione corretamente. Por fim, os testes foram executados de maneira a que possíveis erros e bugs durante a implementação ou possíveis falhas de segurança fossem detetados e corrigidos, bem como de forma a verificar a usabilidade da plataforma.

Este documento detalha as fases do projeto, as escolhas de arquitetura e outros aspetos relevantes do desenvolvimento da plataforma.

Palavras-chave: HYP, Plataforma Web, Engenharia de Software, Desenvolvimento, Testes, Entregas ao Domicílio, Padaria, Comércio

Acknowledgements

I would like to start by thanking my family, especially to my parents, for all the support they have given me throughout the years I have spent in university. Without them none of this would have been possible.

I also want to thank Sara João for giving me this opportunity of making my internship and thesis at HYP, for the privilege of being guided by her during this dissertation and for all the corrections and tips she gave me while writing this dissertation.

To my other adviser, António Leitão I want to thank for the corrections, for the advices and tips he gave me and most important for his friendship.

I could not fail to thank Daniel Barbosa, João Ribeiro, Mariana Alfafar and everyone else at HYP for the good reception and for everything they have taught me and for all the help they gave me.

I would also like to thank my friends, Bruno Pedroso, Pedro Matias, João Sá, Alexandre Jesus, João Cerveira, Tiago Mateus, Ana Rute and João Marques for all the friendship and support they have shown, not only in this past year, but in all my academic years.

To these and all who, although have not been mentioned here, marked and supported me during these six years, thank you!

Contents

List of Figures

List of Tables

Acronyms

BDD Behavior-Driven Development

DAU Daily Active Users

DBMS Database Management System

ER Entity-Relationship

MVC Model-View-Controller

MVP Minimum Viable Product

TDD Test-Driven Development

SDLC Software Development Life Cycle

WAU Weekly Active Users

Chapter 1

Introduction

Waking Bread is a web platform that allows users to order products from bakeries to be delivered at their homes. The bakeries subscribe to the platform to be able to have a personal page on it, in which they can show and sell their own products to the users. The users can search for bakeries that make deliveries at their address and order products to be delivered at their homes, at the date they want.

In this chapter an introduction of the project is done, for a better understanding on what it consists, to whom this project is destined and their main goals. The introduction is divided into four subsections. The first describes the company to whom this project is destined, the second describes the motivation behind this project, the third describes the goals for the project, and the last one does a brief description of what contains each chapter on this document.

1.1 HYP

HYP was created in January 2015 in Coimbra, Portugal, by a team of 3 highly-experienced Computer Engineers and Multimedia Designers with abounding technological ideas to develop, from their own websites and applications, to clients' ideas and projects, covering areas such as consulting, web and mobile development, design, user experience and marketing.

Given the different areas reached, the company stands out from its competitors and tries to create conditions to help its clients to positively thrive their brand/business with great support. Instead of only offering a single specific service (only design, marketing or engineering...), HYP wants to be a breath of fresh air by providing a wide range of services and to perfectly marry Design and Computer Engineering.

1.2 Motivation

The need for convenient solutions for the everyday tasks has been growing. With that, services that provide home delivery have been growing in popularity and have been requested more and more each day. Nowadays, the users value convenience and comfort above other things and are more sedentary than ever. They want everything to be delivered at their homes to avoid their own effort on getting those things. This need gave room for new types of service to emerge and from it came the idea of a new service, that does not exist at the time in Portugal, that could help deliver fresh products from a bakery into the homes of everyone.

1.3 Goal

The main goal of this project is to develop a web platform that allows users to order products from bakeries that are in the system. Users have to be able to register and login in the platform, check their profile, search for products and bakeries and order products to be delivered at their address in a date they chose. The bakeries should be able manage their own products, page and employees, see the orders they have, and check their sales statistics.

1.4 Outline

This document is divided in seven chapters: Related Work, The Project, Planning, System Description, Implementation, Tests and Conclusions.

Related Work (Chapter 2)

This chapter presents the performed studies related to this project, such as a market analysis to get to know the competitors, a web technologies study and a study of usability techniques and tests.

The Project (Chapter 3)

This chapter does a description of what this project consists, defines its objectives and the approach to be followed in the development.

System Description (Chapter 4)

On this chapter the definition of all requirements is presented as well as other artifacts that represent the description of the system, such as its architecture.

Planning (Chapter 5)

This chapter contains the tasks and Gantt diagram for each semester, the description of the methodology used in the project, the technological choices made as well as the productivity tools used.

Implementation (Chapter 6)

The implementation chapter contains the description and explanation of everything that was developed for this web platform, from the user functionalities to the admin area options.

Tests (Chapter 7)

In this chapter the tests performed to test the platform in terms of security, functionalities, database and usability are presented.

Conclusions (Chapter 8)

Finally, in this chapter the final conclusions are taken and the future work is presented.

Chapter 2

Related Work

In this chapter a study is made about topics that are related to this project. It starts by doing a market analysis, describing the direct competitors present in the market and some similar services. Then an analysis and description is made of the web technologies that are being used nowadays in the development of web applications. In the last section of this chapter a study is performed about the usability techniques and tests used in web applications that help improve the user experience.

2.1 Market Analysis

Before starting a new business or service it is always important to know what kind of competition is present in that specific market, so that we can have knowledge of how and where they operate as well as having an understanding of what they are doing wrong and their disadvantages.

This subsection does an analysis of possible competitors of this project that are present in the market, by describing them and showing their pros and cons. In the end there is also a comparison table between the competitors and our web platform.

To chose the competitors, a strict search was conducted, in the beggining of this project, to find services equal or similar to Waking Bread. After the search five websites that differ from each others in a variety of aspects were chosen to be analysed.

2.1.1 Rota da Espiga

This website[1] belongs to a unique bakery located in Queluz, Portugal. It allows users to buy a variety of products from them, to be delivered in the morning at their homes. The location for the deliveries is restricted to the Lisbon area. The payment is done by bank transfer and the distribution is free. Image 2.1 shows the home page of this website.

Pros

- Appealing and intuitive website;
- Free delivery;
- Favorite products;

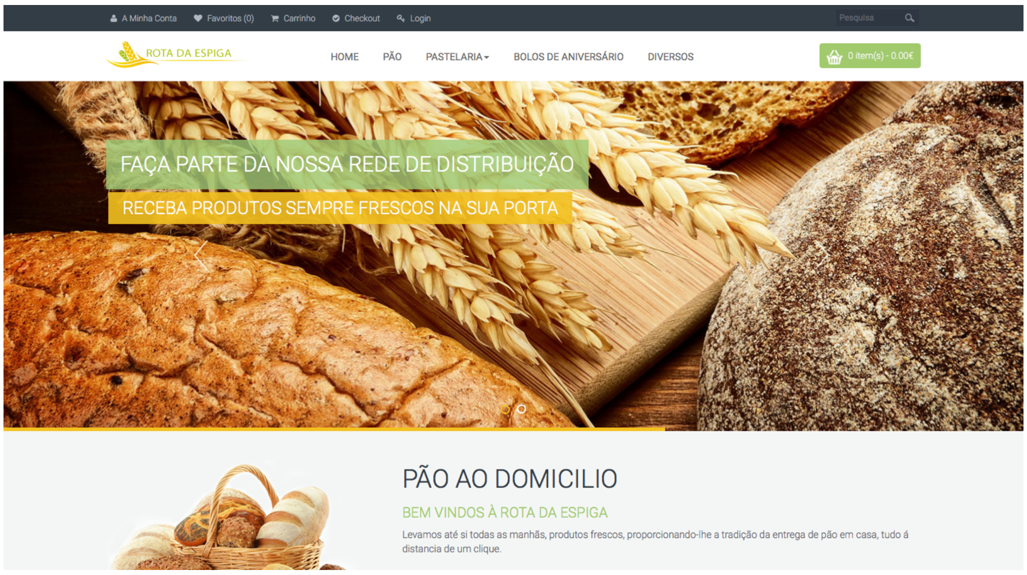
[1]More information at: www.http://www.rotadaespiga.pt/

- Comparison between products;
- Search for products;
- Categories for products;
- Add to cart system;
- Payment by bank transfer;
- Daily, weekly of monthly deliveries.

Cons

- Delivery range is restricted;
- Only products from one bakery;
- Only one payment method;
- Little information about the products;
- Requires security deposit.

In conclusion this website is well designed and constructed, has some great features and overall it is a good website. The main problems are the restricted range for deliveries and only having products from one single bakery.



Levamos até si todas as manhãs, produtos frescos, proporcionando-lhe a tradição da entrega de pão em casa, tudo á distancia de um clique.

Figure 2.1: Home Page of Rota da Espiga

2.1.2 Pao da Quinta

This website[2] is merely a website of a bakery from Palmela, Portugal. It allows users to buy bread only from this bakery to be delivered at home, by ordering online or by a phone call. It has two payment methods: bank transfer or coupons. The delivery range is short,

[2]More information at: http://www.paodaquinta.com.pt/

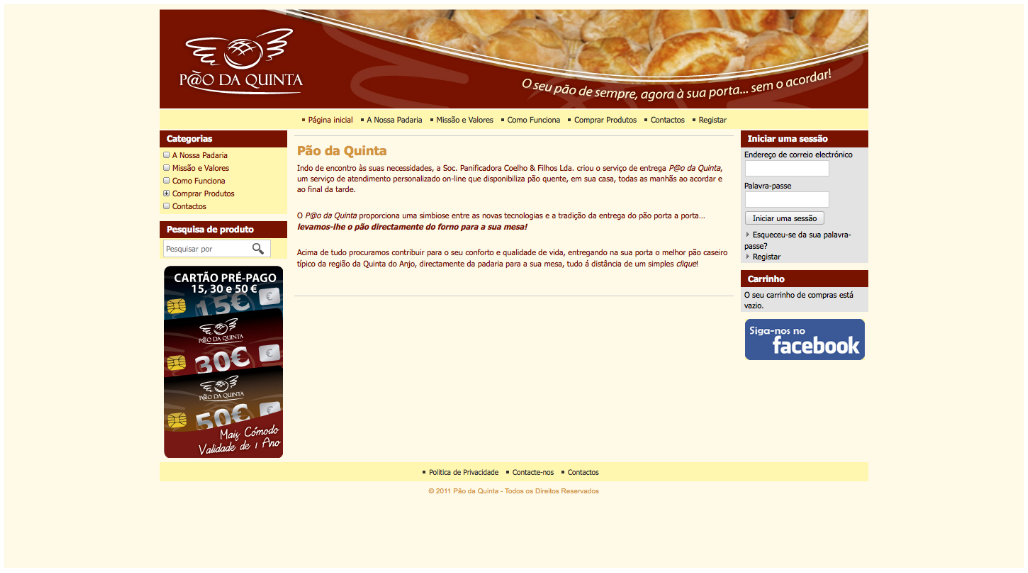
and restricted to the area where the bakery is located. The representation of the home page of this website is presented on image 2.2.

Pros

- Possibility to order products on the website or by phone call;
- Two payment methods;
- Payment by coupons;
- Add to cart system;
- Search for products;
- Free delivery until certain range.

Cons

- Website not appealing and intuitive;
- Delivery range is restricted and small;
- Only products from one bakery;
- Only sells breads;
- Lack of features.



Pão da Quinta

Indo de encontro às suas necessidades, a Soc. Panificadora Coelho & Filhos Lda. criou o serviço de entrega *P@o da Quinta*, um serviço de atendimento personalizado on-line que disponibiliza pão quente, em sua casa, todas as manhãs ao acordar e ao final da tarde.

O *P@o da Quinta* proporciona uma simbiose entre as novas tecnologias e a tradição da entrega do pão porta a porta... ***levamos-lhe o pão directamente do forno para a sua mesa!***

Acima de tudo procuramos contribuir para o seu conforto e qualidade de vida, entregando na sua porta o melhor pão caseiro típico da região da Quinta do Anjo, directamente da padaria para a sua mesa, tudo á distância de um simples *clique*!

O seu carrinho de compras está vazio.

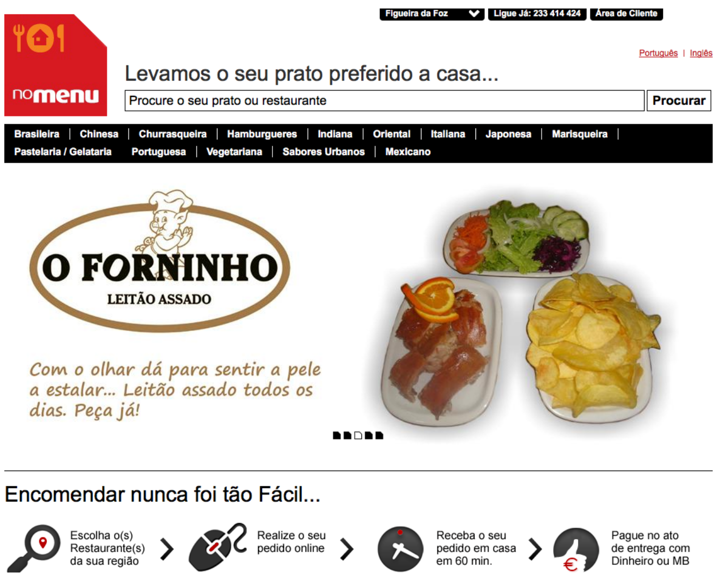
© 2011 Pão da Quinta - Todos os Direitos Reservados

Figure 2.2: Home Page of Pao da Quinta

In conclusion this website has some good functionalities, such as the payment method by coupons and the fact that the products can be bought by phone call or online. The problem is that it does not have a good design, lacks on features, only sells bread and has a really short delivery range.

2.1.3 noMENU

The platform NoMenu[3] is a home delivery system for restaurants, ice creams, flowers and more. With this service the restaurants are not responsible for delivering the orders. Instead, this service has workers who are responsible for going to the restaurants pick up the orders and then delivering them to the users. It started with only a few restaurants but it has now a great set of restaurants and works in many cities throughout Portugal. It gives the users two payment methods, three order methods and multiple hours for delivery. On image 2.3 the home page of this platform is represented.


Figueira da Foz
Ligue Já: 233 414 424
Área de Cliente
Português | Inglês
nomenu
Procure o seu prato ou restaurante
Procurar
Brasileira | Chinesa | Churrasqueira | Hamburgueres | Indiana | Oriental | Italiana | Japonesa | Marisqueira | Pastelaria / Gelataria | Portuguesa | Vegetariana | Sabores Urbanos | Mexicano
O FORNINHO
LEITÃO ASSADO
Escolha o(s) Restaurante(s) da sua região
Realize o seu pedido online
Receba o seu pedido em casa em 60 min.
Pague no ato de entrega com Dinheiro ou MB

Levamos o seu prato preferido a casa...

Com o olhar dá para sentir a pele a estalar... Leitão assado todos os dias. Peça já!

Encomendar nunca foi tão Fácil...

Figure 2.3: Home Page of noMENU

Pros

- Good search method;
- Multiple hours for delivery;
- Two payment methods;
- Three order methods;
- Add to cart system;
- Multiple addresses;
- Large set of delivery locations.

Cons

- Website not appealing;
- Restaurants and other business do not have a personal page;
- Little information about the products and business;
- Delivery is not free, and delivery fee differs from location to location;

[3]More information at: http://www.nomenu.pt/

- Lack of features.

In conclusion this platform is well developed, and the proof of that is its growth. It gives users a large set of business to order from, such as restaurants and florists. It also provides multiple payments and order methods, and works on a large group of locations. The main disadvantages of this service are the lack of features and the fact that the website is not appealing and intuitive.

2.1.4 GrubHub

GruHub[4] is the most similar service to this project, with the main difference being working with restaurants instead of bakeries. Restaurants can subscribe to be on the platform and users can search for products from many restaurants. It works in a numerous number of cities on United States of America, and has a really large set of restaurants in the system. This services gives the user the option to pick up the order or to get it delivered and allows them to rate and comment the restaurants. Image 2.4 shows the home page of GrubHub.

Pros

- Appealing and Intuitive website;
- Advanced search;
- Ratings and comments;
- Add to cart system;
- Mobile application;
- Large set of delivery locations and restaurants;
- Categories for products;
- Option to pick up the order.

Cons

- Too slow;
- Only one payment method;
- Does not allow to search without having inserted a valid address.

In conclusion this platform has a great design that is appealing and intuitive and also has mobile applications. Overall it is a great service with a great set of useful features such as an advanced search, and possibility to rate and comment. However, the system is sometimes slow and takes a long time to process some requests and also requires for the user to insert a valid address to see and search for restaurants.

[4]More information at: https://www.grubhub.com/

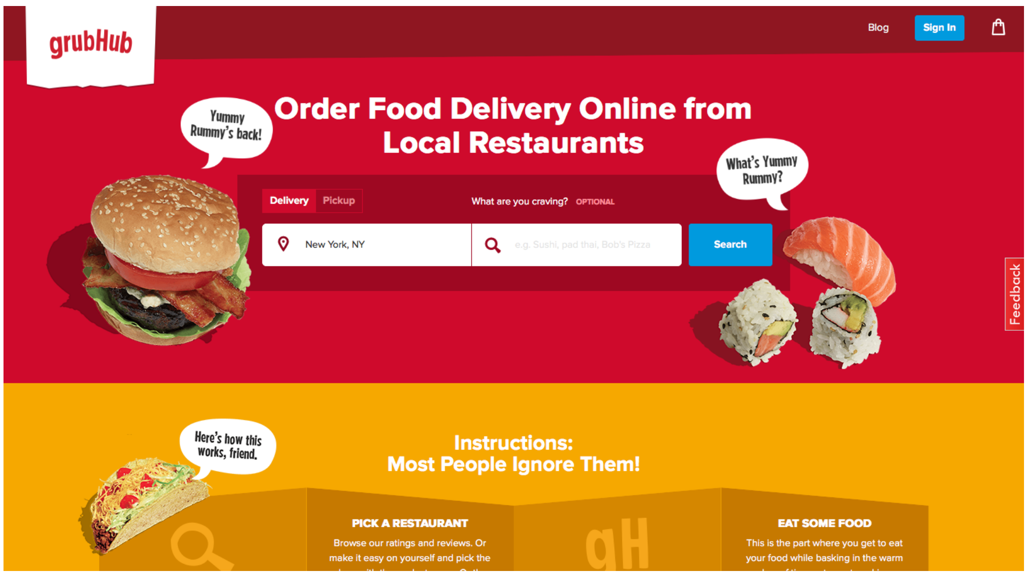

grubHub
Blog
Sign In
Yummy Rummy's back!
Order Food Delivery Online from Local Restaurants
What's Yummy Rummy?
Delivery
Pickup
What are you craving? OPTIONAL
New York, NY
e.g. Sushi, pad thai, Bob's Pizza
Search
Feedback
Here's how this works, friend.
Instructions: Most People Ignore Them!
PICK A RESTAURANT
Browse our ratings and reviews. Or make it easy on yourself and pick the
gH
EAT SOME FOOD
This is the part where you get to eat your food while basking in the warm

Figure 2.4: Home Page of GrubHub

2.1.5 Foodler

Foodler[5] is a web platform that just like GrubHub works with restaurants instead of bakeries and allows the restaurants to subscribe to the platform to sell their products. Users can search for the restaurants and then order meals from them. This platform works in almost every big city across the US and in one city in Canada. It also provides a mobile application for Android and iOS and gives the user the choice to either pick up the order themselves or for it to be delivered at home. Image 2.5 shows the home page of Foodler.

Pros

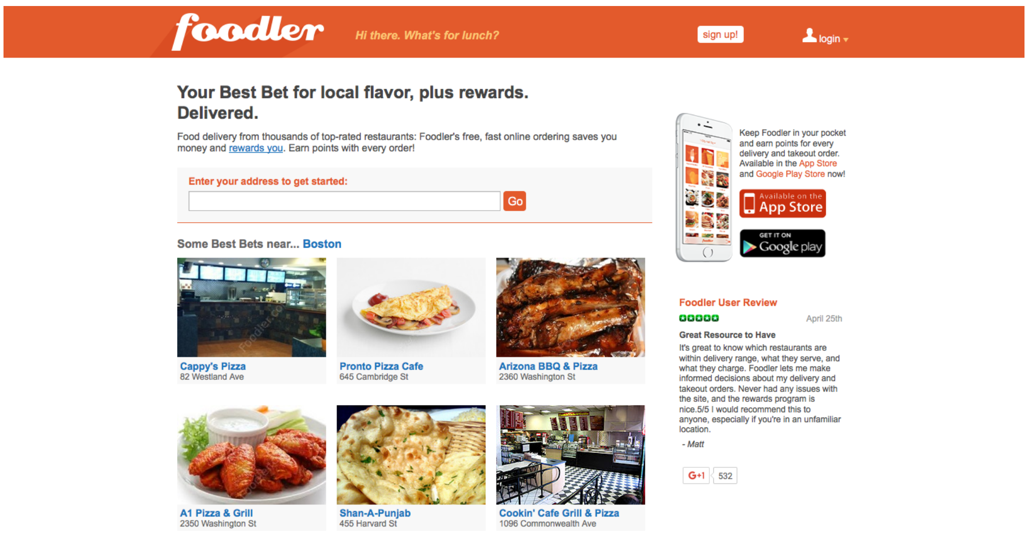
- Ratings and comments;
- Add to cart system;
- Mobile application;
- Large set of delivery locations and restaurants;
- Option to pick up the order;
- Variety of payment methods.

Cons

- Website not appealing and intuitive;
- Simple search;

In conclusion this platform is really complete, with a lot of good features and being complemented with mobile applications. Even though the platform does not provide an advanced search, it still works well and is efficient. The platform also provides the possibility to comment and rate the restaurants and a variety of payments methods that the users can use. The most evident problem of the platform is the fact that the design of it is outdated and not appealing, and some things are hard to figure out.

[5]More information at: https://www.foodler.com/



Your Best Bet for local flavor, plus rewards. Delivered.

Food delivery from thousands of top-rated restaurants: Foodler's free, fast online ordering saves you money and rewards you. Earn points with every order!

Enter your address to get started:

Some Best Bets near... Boston

Cappy's Pizza
82 Westland Ave

Pronto Pizza Cafe
645 Cambridge St

Arizona BBQ & Pizza
2360 Washington St

A1 Pizza & Grill
2350 Washington St

Shan-A-Punjab
455 Harvard St

Cookin' Cafe Grill & Pizza
1096 Commonwealth Ave

Keep Foodler in your pocket and earn points for every delivery and takeout order. Available in the App Store and Google Play Store now!

Foodler User Review

April 25th

Great Resource to Have

It's great to know which restaurants are within delivery range, what they serve, and what they charge. Foodler lets me make informed decisions about my delivery and takeout orders. Never had any issues with the site, and the rewards program is nice.5/5 I would recommend this to anyone, especially if you're in an unfamiliar location.

- Matt

Figure 2.5: Home Page of Foodler

2.1.6 Synthesis

To have a better understanding of the differences between each website analysed and Waking Bread, the following table 2.1 was created. It does a comparison between Waking Bread and the other websites analysed on different aspects.

	Rota da Espiga	**Pao da Quinta**	**noMenu**	**GrubHub**	**Foodler**	**Waking Bread**
Type	Application	Application	Platform	Platform	Platform	Platform
Service	Bakeries	Bakeries	Multiple Services	Restaurants	Restaurants	Bakeries
Multiple Businesses	No	No	Yes	Yes	Yes	Yes
Appealing Design	Yes	No	No	Yes	No	Yes
Intuitive Design	Yes	No	Yes	Yes	No	Yes
Delivery Range	Small	Small	Medium	Large	Large	Large
Advanced search	No	No	Yes	Yes	No	Yes
Rating	No	No	No	Yes	Yes	Yes
Comments	No	No	No	Yes	Yes	No
Categories	Yes	No	Yes	Yes	Yes	Yes
Pick Up Option	No	No	No	Yes	Yes	Yes
Multiple Payment Methods	No	Yes	Yes	No	Yes	Yes

Table 2.1: Market Analysis Comparison

After the analysis of these websites and checking the table above, it is possible to draw some conclusions. There are not many web applications or platforms that do what this project aspire. Most of the websites found during our research have an outdated design and are not intuitive while others do not have many features and do not provide the users with enough information about the bakeries and their products. The other main problem found on our research was that most of the websites are restricted to one bakery and only perform deliveries in one location and with a short range.

GrubHub and Foodler were clearly the best platforms analysed, and are the closest websites to Waking Bread. However, they still have their limitations and problems and most important they only work with restaurants and Waking Bread is destined to work with bakeries.

In conclusion Waking Bread has the possibility to stand out in the market and to be successful, as long as it meets the requirements established in the beginning of the project and keeps adapting to the demands and market trends.

2.2 Web Technologies

The development of a web application is accomplished through the use of different types of web technologies. Typically these web technologies are divided into four areas that when together make a web application. These areas are: back-end, front-end, database, and web server.

These section describes some of the most popular and better web technologies and frameworks for each of the four areas. This technologies and frameworks were found during a research conducted in the early stage of this project.

The technological choices made for this project as well as the comparison between these technologies and frameworks can be seen in the section 5.3.

2.2.1 Back-End

The Back-End of a web application or platform is what gives support to the front-end. Its function is to deal with the requests made by the users on the front-end of the application. This is achieved by being in constant contact with the database and by manipulating the data and manage it to provide an adequate response to the request made in the front-end.

To help with this task, there are a large number of frameworks that contribute for a better development and functioning of a web platform. These frameworks help with the development of web platforms by providing some helpful functionalities that can reduce the time and difficulty needed to implement the web platform and by improving their performance.

This subsection describes four of the most popular and better frameworks available nowadays, all based in different programming languages. These frameworks were chosen due to the fact that they all have a big share market in this area, they are based on different programming languages, and they all are great choices to develop this platform.

Laravel [1] [2]

The most used language for back-end frameworks is PHP [3]. In this language there is a large number of different frameworks, and as show on the survey realized by SitePoint [4], Laravel is by far the most popular and favorite PHP framework.

Laravel is an open source framework and had its initial release in 2011 and as mentioned above is based on PHP. It folows the MVC architecture model [5] and was released under the MIT License. Laravel also supports four popular DBMS: PostgreSQL[6], SQL

Server, MySQL[7] and SQLite[8].

Its main objective is to deliver an expressive and elegant syntax for those that value these attributes. Like its competitors, Laravel brings many features that help with some tasks, such as authentication and sessions.

Django [9]

Born in 2005, Django is an open source web framework based in Python, that has been growing ever since, having now a great and big community that provides helpful tutorials and books. It strives to be as fast as possible during the development phase, while still being extremely secure and scalable.

It uses the MVC architecture, that in Django is called MVT (Model-View-Template), and supports a large number of databases including Oracle, PostgreSQL and MySQL. In terms of functionalities, Django has a large set, providing features for the authentication, administration and more, while still developing and implementing new functionalities for each release.

With security being one of the main goals, Django can deal with SQL injection, cross-site request forgery and other major types of security attacks.

Node.js [10]

Based on JavaScript, Node.js is a recent framework that had its initial release in 2009 and is described as a JavaScript runtime. Unlike the other frameworks, Node.js uses an event driven, non-blocking I/O model designed to build scalable network applications, as cited in [10]. Also, Node.js is under the free MIT License.

Node.js applications can run in Windows, Linux or OS X, and has a built-in library with multiple modules that help with the development. Due to the use of the Google V8 JavaScript engine and being a JavaScript runtime, Node.js is very fast and lightweight.

Ruby on Rails [11] [12]

Ruby on Rails was created in 2005 and was released under the MIT License. It is a open source framework developed in the programming language Ruby, that uses the MVC architecture model.

Rails main goal is to allow the developer to write less code and still be able to accomplish the required objective. It also has two main ideals: do not repeat yourself, meaning that you should not repeat the same piece of code multiple times in the application, and that conventions shall prevail to the detriment of configurations, in order to end the creation of big and complex configurations files.

To help accomplish its goal, Ruby on Rails uses gems (ruby programs and libraries) that help the development of certain functionalities, such as authentication, debug and more. Also, Rails provides a group of commands that allow the construction of a web application in a short period of time.

In terms of support, the community of Ruby on Rails is a really large one, with a large number of tutorials, books and examples available, and with a large amount of useful gems that were developed by the community, available to be used.

2.2.2 Front-End

Front-End is the interface of a website that the user sees and interacts with. It is accomplished through multiple languages such as HTML, CSS and JavaScript. These three technologies are the main ones to develop a website:

HTML (HyperText Markup Language) [13] is used to construct web pages, allowing them to have buttons, images, links and more. CSS (Cascading Style Sheets) [14] is a language that has the purpose of giving the websites some style and color, and that define how the HTML elements are presented. JavaScript [15] is a dynamic programming language used to make dynamic websites with a responsive interface in order to improve user experience.

Nowadays, there is a large number of frameworks that help improve the front-end of websites and their implementation. Since this is not the focus of this thesis and the author was not the responsible for doing the front-end of the web application, this section does only a brief description of the most popular frameworks available.

Sass [16]

Sass (Syntactically Awesome StyleSheets) is an extension of CSS for Ruby that allows the use of inline imports, variables, add nested rules, along with other things. It can be written in two different syntaxes: Sass and SCSS. The first one is the oldest one, and is known to be easier to read and to write. The second one is the main syntax used in Sass 3 and is an extension of CSS, meaning that every CSS code is a valid SCSS code.

Less [17]

Less is a dynamic stylesheet language that extends CSS and allows writing CSS code in a better way by combining variables, functions and more. It also runs in both client-side (browsers) and server-side(Node.js).

Susy [18]

Susy is a framework for Sass to help creating the layout of the website. The framework provides different grids that can be used together or separated to create the desired layout.

Bootstrap [19]

As their website describes it, "Bootstrap is the most popular HTML, CSS, and JS framework for developing responsive, mobile first projects on the web.". It is a open source framework that has design templates for most of HTML elements, responsive grids and has some useful JavaScript plugins. Bootstrap also allows the creation of responsive websites in a simply and easy way.

Foundation [20]

Foundation is a responsive front-end framework, created to make the task of creating responsive websites easier. It has several HTML templates and code snippets to help with the development and includes optional JavaScript extensions and a responsive grid.

2.2.3 Database

The book "Database Management Systems" [21] defines a Database Management System (DBMS) as a "software designed to assist in maintaining and utilizing large collections of data." They are used to validate and store the data of a web application in a way that becomes easier to retrieve and manipulate it.

There is a vast group of database managements systems available in the market that the user can choose from. The DBMS's have different database models, with the most common one being the Relational DBMS. This model relies on tables with a specific and unique name that have a fixed number of attributes, each one with a data type. When data is inserted in the table, a row is created with the values for each attribute.

According to [22], the four biggest Relational DBMS's in the January 2015 rankings are: Oracle, MySQL, Microsoft SQL Server and PostgreSQL. These four DBMS's have been around for a long time and have always been the most popular ones.

This subsection analysis and describes MySQL, PostgreSQL and also SQLite that is the default DBMS used in Ruby on Rails, leaving behind Oracle and Microsoft SQL Server since these two DBMS are not open source and require a license to be used. Also, only Relational DBMS were considered since the author only has experience with this kind of DBMS.

PostgreSQL [6]

PostgreSQL is a open source DBMS that has been around for almost twenty years. It runs on all the most important operating systems and has a strong reputation in the area, winning some awards for best DBMS.

Known by its stability, reliability and extensibility, PostgreSQL includes most of the data types available, supports almost every relational database features and even has some exclusive features. In terms of support, there is a large active community that helps resolve any problem.

MySQL [7]

MySQL being the most popular open source database is not an accident, and it is one of the best DBMS there is. Developed and distributed by Oracle, MySQl had its initial release twenty years ago.

MySQL excels in terms of performance and scalability, and is also easy to use. Like PostgreSQL, it includes most of the data types, supports a lot of features and runs in many operating systems. Also, MySQL is an extremely safe and secure database.

SQLite [8]

Released in the year 2000, SQLite is not a client-server DBMS such as PostgreSQL or MySQL, instead it is a library that is embedded into an application.

SQLite comes by default in Ruby on Rails and is extremely portable, with the database being entirely saved on a single file. This makes the database a great choice for the developing and testing phases of a project and for applications that require portability.

2.2.4 Web Server

As cited in [23], a "web server is a system that delivers content or services to end-users over the internet. A web server consists of a physical server, server operating system (OS) and software used to facilitate HTTP communication." This means that a web server is a server that on the hardware side its function is to store all the files that are related to the website and deliver them to the users, and on the software side its function is to receive and interpret the HTTP requests that were made by the end-users.[24]

Nowadays there are four major web servers, Apache that is the most popular one and has the most share market, Nginx that is the second largest and serves some popular websites such as Netflix and GitHub, IIS that is a web server developed by Microsoft and used by them, and Google web server [25].

Since the last two web servers are not open sourced, only the first two were analysed. Below, a description of the two open source web servers is presented.

Apache [26] [27]

The Apache HTTP Server, commonly known as Apache, is a collaborative open source HTTP server and was launched in 1995. Since then, it has been the most popular and most used web server in the world.

Created to be robust, secure, efficient, and to have a lot of useful features, the Apache HTTP Server has continuously been developed by a wide range of volunteers that communicate through the Web to do all the work related to the project, from the development of the server to writing its documentation.

About the server, it uses a multi-process and multi-thread architecture, implements the latest protocols available, allows people to write their own modules with the help of their API, is completely free of licences, runs on almost all operating systems and it has a lot of specifications that the user can configure.

Nginx [28] [29]

Nginx is an open source HTTP server and reverse proxy server. Created in 2002 by Igor Sysoev, Nginx has been growing exponentially ever since and is now serving some big and known websites.

Originally, the server was developed to serve a Russian website with high-traffic. Therefore the need for it to be stable, efficient and with a good performance. Nowadays, it is known to have all those qualities and also having a good feature set, performing well while consuming few resources and having an easy configuration.

The server was build in a particular way, that differs from other servers. Traditionally the servers have multiple threads or multiple processes to handle the requests, but Nginx uses the asynchronous multi-process event driven architecture. This means that Nginx uses small and predictable pieces of memory under load making the architecture more scalable. This way, and thanks to the Nginx high performance and efficiency, the server can still handle a large amount of requests at the same time, without any problems. Also, Nginx, just like Apache, can run in almost any operating systems.

2.3 Usability

The book "Human Factors for Informatics Usability" [30] defines usability as "the capability in human functional terms to be used easily and effectively by the specific range of users, given specified training and user support, to fulfill the specific range of tasks, within the specific range of environmental scenarios". In shorter words "the capability to be used by humans easily and effectively".

The problem with this definition is that usability is more than these two properties. There are multiple attributes that help defining usability. In the book "Usability Engineering" [31] by Jacob Nielsen, a well-known expert in this area, usability is defined by five main attributes: Learnability, Efficiency, Memorability, Errors and Satisfaction;

- **Learnability** - How easily can users accomplish basic tasks in their first time trying;
- **Efficiency** - How fast can users perform a task after they have learned it;
- **Memorability** - How easy is it for users that do not have seen the design for a period of time, to start performing tasks again;
- **Errors** - How many errors they make and how easily can they overcome their errors;
- **Satisfaction** - How happy are the users with the design.

So, why is usability important? A web platform with good usability is more appealing and intuitive, and users value that above everything. If a website does not have a good usability, users tend to stop using it and search for better solutions. As cited on [32], "The first law of e-commerce is that if users cannot find the product, they cannot buy it either". This sentence could not be more true. If a home page is not explicit about what exactly the website is, or if the users take to long to find what they are looking for on the website, they simply leave.

Nowadays, users are more demanding and are always looking for the best solution to solve their problem, making usability a key aspect on a web platform. There are some usability techniques and rules that can help improving a web platform, and that every website should follow. Article [33] describes some techniques that should be followed:

- Home page should clearly explain what the website is and does in one sentence;
- The tittle needs to have a good visibility;
- Main task of the website should be emphasized;
- The website should have a good search box;
- New features should be highlight;

- Visited links should be noticeable;
- Text should be easy to read and not too long;
- Avoid anything that can be interpreted as an advertisement.

There are other rules that should be followed in addition to those mentioned above, such as the user should do the minimum number of steps possible to complete a task, the eyes and mouse of the user should not cross the entire window multiple times, the information should be concise and readable for everyone, all features should be implemented to not mislead the user, filling the screen with unnecessary information should be avoided and users should receive feedback of every action they perform in the system. Also, when there is a need to present a large quantity of text to the user, the most important information should be in the first two paragraphs. [34]

In order to evaluate the usability of web platforms there are some evaluation techniques that can be performed. Article [35] describes some evaluation techniques, such as the Heuristic Evaluation, Feature Inspection, Standards Inspection and Cognitive Walkthroughs. For instance, Heuristic Evaluation judges a web application based on some heuristics such as, Feedback, Simple and Natural Dialog, Consistency and more. Standards inspection is done by having an expert on some interface standard doing a evaluation. These techniques are easy to execute and some of them do not require an expert on the subject.

There is also a set of practical tests that should be executed to test the usability of the web platform. For example, the web platform should be tested by a variety of people from different age groups and occupations in order to have better feedback of the most common errors that they encounter when using the platform, and on the amount of effort they have to put on to perform each task. Also, these tests should be divided into two phases, wherein the first phase the user should not have any information on how to perform any task, while on the second phase the user should know how to perform the tasks.

These evaluation techniques and tests should always be performed in every development of a web platform since they contribute for a better usability and help improve the user experience.

Chapter 3

The Project

In the modern times, users strive for solutions that can help them do and achieve their goals without any effort. They want convenience and comfort in their everyday tasks, and are more and more sedentary each day.

Bill Gates once said "I will always choose a lazy person to do a difficult job because a lazy person will find an easy way to do it.". This quote can be applied to other scopes. For instance, lazy persons are giving room for new types of services to rise and grow in this society, due to the fact that they want to do as little as possible, and do not mind paying for a service that they could do.

One type of service that have been growing due to this need, is home delivery services. These services has been requested more each day and a lot of business are adopting it. This project came from an idea to solve a simple problem by using this type of service as a solution. The original problem was:

"How can I have fresh bread in the morning without the
need for me to leave home to get it?"

The solution was to create a platform where bakeries could subscribe and have a page with their products to sale. Then users could register and order products from the bakeries available to be delivered at their homes at the dates they choose. From this solution to that simple problem, Waking Bread was born.

Waking Bread is a web platform that implements this solution in the best way possible, by having a large number of useful functionalities that help improving the user experience. These functionalities were debated with a bakery that was made a partner of this project in an initial phase. This partnership helped us understanding what bakeries expect and want in an platform of this type.

With this approach, the platform can give the users a more broad and wide number of bakeries and products to order from, instead of some web applications that only belong to a specific bakery. Also, this method can be applied to everywhere as long as there are bakeries in that place that are subscribed to this platform.

3.1 Objectives

This project has a set of key objectives that should be accomplished by the end of it, in order for it to be a success. The objectives are:

- Learn how web platforms operate and function;
- Successfully learn how to develop web platforms;
- Work in team;
- Follow the HYP code guidelines;
- Contribute with ideas to improve the platform;
- Be active during the project meetings;
- Implement at least one functional payment method;
- Use a geolocation service;
- Implement all the must functional requirements;
- Code should be ready to implement other languages;

This objectives were defined together with HYP and are a must requirement in the project.

3.2 Collaboration

The web platform developed in this project is being developed in collaboration with other person. The person in question is a master's degree student in design and multimedia also doing his internship for his master thesis on HYP.

While the author's obligations to this project are to develop the back-end of it and everything related to it, his obligations are to study the image of this project and define it, and also develop the front-end of the platform.

Chapter 4

System Description

This chapter does an overview description of the system. First, the system requirements are described by user stories and use cases, the priority of the functional requirements is presented and the non-functional requirements are described. Then, an analysis of the system architecture is made, the database architecture is represented in the form of an entity relationship diagram and the flow of the website is represented in a navigation diagram. At last, the wireframes created are presented.

4.1 User Stories

User Stories are a simple way of representing the system requirements in a user's point of view. They are represented as a group of sentences written in an casual language describing what the user needs or wants to do in the system.

Nowadays there are a couple of formats to do the User Stories, but in this project we stick in the traditional way, that takes this form: [36] [37]

As a <role>
I want to <goal>
So that <reason>

This format tries to respond to the questions of "who" wants this action, "what" is the action and "why" is the action needed.

4.1.1 User

1. Login
    (a) **As a** user **I want to** login with my email **so that** I can access the website with my account.
    (b) **As a** user **I want to** login with Facebook or Google **so that** I can access the website with my account quickly.
    (c) **As a** user **I want to** retrieve my lost password **so that** I can avoid losing my account.
2. Register
    (a) **As a** user **I want to** register **so that** I can have an account.

   (b) **As a** user **I want to** register with Facebook or Google **so that** I can register quickly.

3. Home Screen
   (a) **As a** user **I want** to see if it is possible to deliver something in my address **so that** I do not waste time registering.
   (b) **As a** user **I want to** have a navigation menu **so that** I can see and experience the available features.
   (c) **As a** user **I want to** see the most popular bakeries **so that** I can know which ones are the best.
   (d) **As a** user **I want to** see the promotions in the home screen **so that** I do not miss them.
   (e) **As a** user **I want to** see the popular products **so that** I can know what products are popular during this particular season.
   (f) **As a** user **I want to** see the best selling products **so that** I can check them.
   (g) **As a** user **I want to** see my favorites products / bakeries **so that** I can access them easily.

4. Profile and Settings
   (a) **As a** user **I want to** edit my profile **so that** I can change my email.
   (b) **As a** user **I want to** edit my profile **so that** I can change my password.
   (c) **As a** user **I want to** edit my profile **so that** I can change my personal data.
   (d) **As a** user **I want to** edit my profile **so that** I can add my allergies.
   (e) **As a** user **I want to** edit my profile **so that** I change my addresses.
   (f) **As a** user **I want to** edit my addresses **so that** I can select the main one.
   (g) **As a** user **I want to** see my profile **so that** I can see my personal data.
   (h) **As a** user **I want to** see my profile **so that** I can check my purchases.
   (i) **As a** user **I want to** see my purchases **so that** I can see how much I have been spending.
   (j) **As a** user **I want to** see my purchases **so that** I can see how many products I bought in the past month.
   (k) **As a** user **I want to** see my favorite bakeries **so that** I can edit them.
   (l) **As a** user **I want to** see my favorite product **so that** I can edit them.

5. Search
   (a) **As a** user **I want to** search for a bakery **so that** I can find one to order from.
   (b) **As a** user **I want to** search for a product **so that** I can find what I want to buy.
   (c) **As a** user **I want to** add filters to my search **so that** I can easily find what I am looking for.
   (d) **As a** user **I want to** search for available promotions **so that** I can save money.
   (e) **As a** user **I want to** sort the search **so that** I can see the cheaper products first.

   (f) **As a** user **I want to** find a bakery in my region **so that** I can find the closest to me.

6. Purchase

   (a) **As a** user **I want to** have more than one payment method **so that** I can choose my favorite one.
   (b) **As a** user **I want to** select the delivery date **so that** I can purchase products for next week.
   (c) **As a** user **I want to** buy multiple products **so that** I can be happy.
   (d) **As a** user **I want to** get my receipt **so that** I can add it to my expenses for taxes purposes.
   (e) **As a** user **I want to** have to confirm my purchase **so that** I can check if everything is alright before ordering.
   (f) **As a** user **I want to** edit the comments area **so that** I can inform the bakery of any important notes.

7. Bakery

   (a) **As a** user **I want to** see the bakery page **so that** I can check their personal information.
   (b) **As a** user **I want to** see the photos of the bakery **so that** I can check the conditions they present.
   (c) **As a** user **I want to** see the address of the bakery **so that** I can know where it is.
   (d) **As a** user **I want to** check the products of the bakery **so that** I can see if they have my favorite cake.
   (e) **As a** user **I want to** see their schedule **so that** I can check their opening hours.
   (f) **As a** user **I want to** see a bakery's delivery schedule **so that** I can see if they can delivery bread when I need.
   (g) **As a** user **I want to** see the promotions they have **so that** I can check if they interest me.
   (h) **As a** user **I want to** see the bakery's specialty of the house **so that** I can see if I like it.
   (i) **As a** user **I want to** see their payment methods **so that** I can check if they accept the one I want to use.
   (j) **As a** user **I want to** see their classification **so that** I can know what people think about it.
   (k) **As a** user **I want to** see comments by other users **so that** I can see what people say about it.
   (l) **As a** user **I want to** check their location in google maps **so that** I can know exactly where it is.
   (m) **As a** user **I want to** classify the bakery **so that** I can give my feedback.
   (n) **As a** user **I want to** classify the products **so that** I can give my feedback.
   (o) **As a** user **I want to** comment about the bakery **so that** I can give my opinion.

(p) **As a** user **I want to** purchase a product from the bakery **so that** I can be happy.

8. Client Area

    (a) **As a** user **I want to** know how the client area works **so that** I can subscribe to the service.
    (b) **As a** user **I want to** know the advantages of subscribing to the service **so that** I can see if they are compatible with my budget.
    (c) **As a** user **I want to** know the conditions of the service **so that** I can see if they are aligned with mine.
    (d) **As a** user **I want to** see the contact form **so that** I can subscribe to the service.
    (e) **As a** user **I want to** be redirected to the client area **so that** I can login in there.
    (f) **As a** user **I want to** send a message **so that** I can know more about the service.

9. About

    (a) **As a** user **I want to** see information about the service **so that** I can learn more about it.
    (b) **As a** user **I want to** know how the service works **so that** I can check if it interests me.
    (c) **As a** user **I want to** know what payment methods are available **so that** I can check if the one I prefer is available.
    (d) **As a** user **I want to** see the contact of the service **so that** I can contact them to know more.
    (e) **As a** user **I want to** be able to send a message to the support team **so that** I can have my question answered.
    (f) **As a** user **I want to** be able to send my feedback to the team **so that** I can help improve the service.

4.1.2 Bakery

1. Login

    (a) **As a** bakery **I want to** login with my email **so that** I can access the website with my account.
    (b) **As a** bakery **I want to** retrieve my password **so that** I can avoid losing my account.

2. Pre-registration

    (a) **As a** user **I want to** pre-register **so that** I can have my bakery in the website.

3. Profile and Settings

    (a) **As a** bakery **I want to** see my profile **so that** I can evaluate if it needs changes.

(b) **As a** bakery **I want to** edit my profile **so that** I can change my personal data.
(c) **As a** bakery **I want to** edit my profile **so that** I can change my delivery schedule.
(d) **As a** bakery **I want to** edit my profile **so that** I can change the available payment methods.
(e) **As a** bakery **I want to** edit my profile **so that** I can change my address.
(f) **As a** bakery **I want to** edit the specialties of the house **so that** the user can see what our best products are.
(g) **As a** bakery **I want to** upload photos **so that** I can show the users my bakery.
(h) **As a** bakery **I want to** edit my photos **so that** I can remove some old ones.
(i) **As a** bakery **I want to** create workers **so that** I can give the sign in credentials to my employees.
(j) **As a** bakery **I want to** change my region **so that** I can do deliveries there.
(k) **As a** bakery **I want to** add another region **so that** I can make more deliveries.

4. Products

   (a) **As a** bakery **I want to** add a product to my inventory **so that** I can offer the users my newest product.
   (b) **As a** bakery **I want to** add a photo of the product **so that** the user can see how it looks like.
   (c) **As a** bakery **I want to** assign a category to my product **so that** the user can easily find it.
   (d) **As a** bakery **I want to** assign a sub-category to my product **so that** the user knows to which sub-categories the product belongs.
   (e) **As a** bakery **I want to** remove a product **so that** I can prevent the user from purchasing a product that we do not produce anymore.

5. Promotions

   (a) **As a** bakery **I want to** put a product on promotion **so that** I can sell more.
   (b) **As a** bakery **I want to** edit the promotion **so that** I can define the date of the promotion.
   (c) **As a** bakery **I want to** define the conditions of the promotion **so that** I can determine when is the promotion valid.

6. Consult Sales

   (a) **As a** bakery **I want to** consult the sales **so that** I can see how much money we made is the past two weeks.
   (b) **As a** bakery **I want to** consult the sales **so that** I can see what is our best selling product.
   (c) **As a** bakery **I want to** change the order of the sales **so that** I can see the first products sold.
   (d) **As a** bakery **I want to** generate a graph **so that** I can see my earnings of the past month.

7. Deliveries
   (a) **As a** bakery **I want to** see the deliveries I have for tomorrow **so that** I can avoid missing any.
   (b) **As a** bakery **I want to** know the places of the deliveries **so that** I can trace a route.
   (c) **As a** bakery **I want to** know the contact of a client **so that** I can contact him in case there is a problem in the delivery.
   (d) **As a** bakery **I want to** know the special orders **so that** I can prepare them in advance.
   (e) **As a** bakery **I want to** see deliveries for the next days in different views **so that** I do not get confused.
8. Employees
   (a) **As a** bakery **I want to** see my employees **so that** I can know who has access to the website.
   (b) **As a** bakery **I want to** add an employee **so that** he can check the deliveries.
9. Contract
   (a) **As a** bakery **I want to** see my actual contract **so that** I can know when it expires.
   (b) **As a** bakery **I want to** renew my contract **so that** I can sell even more.

4.1.3 Admin

1. Manage Bakeries
   (a) **As a** admin **I want to** add a bakery **so that** I can attract more users.
   (b) **As a** admin **I want to** list all the bakeries **so that** I can check them.
   (c) **As a** admin **I want to** see a bakery **so that** I can check their contract information.
   (d) **As a** admin **I want to** check the sales of the bakeries **so that** I can see which one is the best seller.
   (e) **As a** admin **I want to** select a bakery **so that** I can change their contract.
2. Manage Users
   (a) **As a** admin **I want to** list all users **so that** I can check them.
   (b) **As a** admin **I want to** search for a user **so that** I can ban him.
3. Manage Workers
   (a) **As a** admin **I want to** add a worker **so that** I can assign it to a bakery.
   (b) **As a** admin **I want to** list all workers **so that** I can check them.
   (c) **As a** admin **I want to** search for a worker **so that** I can see in which bakery is he on.
4. Manage Categories

   (a) **As a** admin **I want to** add categories **so that** the bakeries can have more categories to choose from.
   (b) **As a** admin **I want to** add sub-categories **so that** the bakeries can have more sub-categories to choose from.
   (c) **As a** admin **I want to** remove categories **so that** I can remove one that is not being used anymore.
   (d) **As a** admin **I want to** remove sub-categories **so that** I can remove one that is not being used anymore.

5. Manage Payments
   (a) **As a** admin **I want to** add payments methods **so that** the user can have more options to pay for their purchases.
   (b) **As a** admin **I want to** remove payments methods **so that** I can remove one that is no longer available.

4.2 Use Cases

The Use Cases were created to define the functional requirements of the actions by goal, pre and post conditions, and the main and secondary flow of events. The table 4.1 refers to the Use Cases of the User, table 4.2 to the Bakery and table 4.3 to the Admin. [38]

User					
Name	**Goal**	**Pre-conditions**	**Post-conditions**	**Main flow of events**	**Secondary flow of events**
1. Register	Allow the user to create an account in the website	None	An account for the user is created	The **user** indicates their intention to register The **system** presents the fields to be filled The **user** fills the fields The **system** creates the account	If the user is already created or the fields are not correctly filled: The **system** indicates that there is something wrong The **user** fills the fields again The **system** creates the account if it is valid. Otherwise repeat
2. Login	Allow the user to do purchases in the website	The user needs an account	The user has access to all the user's functionalities	The **user** indicates their intention to login The **system** presents the text boxes The **user** fills the text boxes with their login information The **system** logs the user and redirects them to the home page	If the login is not valid: The **system** indicates that the login is incorrect The **user** inserts their login information The **system** logs the user if the information is valid. Otherwise repeat
3. See profile	Allow the user to see his personal information	The user has to be logged in	None	The **user** indicates their intention to see their personal information The **system** presents the information The **user** sees their personal information	None
4. Edit profile	Allow the user to edit his personal information	The user has to be logged in	The user's personal information is updated	The **user** indicates their intention to edit his personal information The **system** presents the user with the fields to edit The **user** edits the information The **system** validates the information and saves it	If the user inserts invalid information: The **system** indicates that there is invalid information The **user** changes the information to be valid The **system** validates the information and saves it. Otherwise repeat
5. See purchase history	Allow the user to see their purchase history	The user has to be logged in	None	The **user** indicates their intention to see their purchase history The **system** presents the user with the information The **user** sees their purchase history	None
6. Search for product	Allow the user to search for a specific product	None	A list of products that match the search is showed	The **user** indicates their search The **system** presents the user with the search results	None
7. Search for bakery	Allow the user to search for a specific bakery	None	A list of bakeries that match the search is showed	The **user** indicates their search The **system** presents the user with the search results	None

User					
Name	**Goal**	**Pre-conditions**	**Post-conditions**	**Main flow of events**	**Secondary flow of events**
8. Order product	Allow the user to order products	The user has to be logged in	An order is created	The **user** inserts products to the shopping bag The **user** indicates their intention to order the products in the shopping bag The **system** presents the user with the check-out page The **user** indicates their intention to order The **system** presents the user with the payment page The **user** pays the order The **system** creates the order	None
9. Check bakery page	Allow the user to see the page of a bakery	None	None	The **user** indicates their intention to see a bakery page The **system** presents the user with the bakery page The **user** sees the page	None
10. Reset shopping bag	Allow the user to cancel his current shopping bag	The user has to be logged in The user has to have inserted at least one product in the shopping bag	The shopping bag is reset	The **user** indicates their intention to reset the shopping bag The **system** resets the shopping bag	None
11. See information	Allow the user to see information about the website	None	None	The **user** indicates their intention to see information about the website The **system** presents the user with the information The **user** sees the information about the website	None

Table 4.1: User Use Cases

Bakery					
Name	**Goal**	**Pre-conditions**	**Post-conditions**	**Main flow of events**	**Secondary flow of events**
1. See orders	Allow the bakery to see their next deliveries	The bakery has to be logged in	None	The **bakery** indicates their intention to see the deliveries The **system** presents the bakery with the deliveries	None
2. See products	Allow the bakery to see their products	The bakery has to be logged in	None	The **bakery** indicates their intentions to see her products The **system** presents the bakery with their products	None
3. Add Product	Allow the bakery to add a product to be sold	The bakery has to be logged in	A new product is added	The **bakery** indicates their intention to add a new product The **system** presents the bakery with the form to add a new product The **bakery** fills the form The **system** check the form and adds the product	If the form has fields not correctly filled: The **system** indicates that there is something wrong The **user** fills the form again The **system** adds the product if it is valid. Otherwise repeat
4. See bakery profile	Allow the bakery to see their page	The bakery has to be logged in	None	The **bakery** indicates their intention to see their page The **system** presents the bakery with their page	None
5. Edit bakery profile	Allow the bakery to edit their page	The bakery has to be logged in	The bakery page is updated	The **bakery** indicates their intention to edit her page The **system** presents the bakery with the fields to edit The **bakery** edits the information The **system** checks the fields and saves it	If the page has fields not correctly filled: The **system** indicates that there is something wrong The **user** fills the fields again The **system** checks the fields and saves it if valid. Otherwise repeat
6. See bakery statistics	Allow the bakery to see their sales statistics	The bakery has to be logged in	None	The **bakery** indicates their intention see their sales statistics The **system** presents the bakery with their sales statistics	None
7. See contract	Allow the bakery to see their current contract information	The bakery has to be logged in	None	The **bakery** indicates their intention see their contract information The **system** presents the bakery with their contract information	None
8. Manage employees	Allow the bakery to add or remove employees	The bakery has to be logged in	A employee is created or deleted	The **bakery** indicates their intention to add or remove a employee The **system** adds or removes a employee	None

Bakery					
Name	**Goal**	**Pre-conditions**	**Post-conditions**	**Main flow of events**	**Secondary flow of events**
9. Register bakery	Allow the user to pre-register their bakery	None	A pre-register is sent to the administrator of the website	The **user** indicates their intention to pre-register their bakery The **system** presents the user with the form to fill The **user** fills the form The **system** validates the form and saves it	If the user inserts invalid information: The **system** indicates that the form is not filled correctly The **user** fills the form again The **system** validates the form and saves it. Otherwise repeat

Table 4.2: Bakery Use Cases

Admin					
Name	**Goal**	**Pre-conditions**	**Post-conditions**	**Main flow of events**	**Secondary flow of events**
1. Manage bakeries	Allow the admin to add bakeries	The admin has to be logged in	A bakery is created	The **admin** indicates their intention to add a bakery The **system** presents the admin with a form to add a bakery The **admin** fills the form The **system** checks the form and adds the product	If the form has fields not correctly filled: The **system** indicates that there is something wrong The **user** fills the form again The **system** adds the product if it is valid. Otherwise repeat
2. Manage bakeries contract	Allow the admin to renew a contract of a bakery	The bakery needs to have a contract	The bakery contract is updated	The **admin** indicates their intention to renew a contract of a bakery The **system** presents the admin with a list of bakeries The **admin** chooses the bakery The **system** renews the contract of the bakery	None
3. Manage users	Allow the user to ban a user	None	The user is banned	The **admin** indicates their intention to ban a user The **system** presents the admin with a list of users The **admin** chooses the user The **system** bans the user	None
4. Manage Workers	Allow the admin to add a worker	None	A worker is created	The **admin** indicates their intention to add a worker The **system** presents the admin with a form to add a worker The **admin** fills the form The **system** checks the form and creates a worker	If the form has fields not correctly filled: The **system** indicates that there is something wrong The **user** fills the form again The **system** creates the worker if it is valid. Otherwise repeat
5. Manage Categories	Allow the admin to add categories and sub-categories	None	A category or sub-category is added	The **admin** indicates their intention to add a category The **system** presents the admin with a form to add a category The **admin** fills the form The **system** adds a category	None
6. Manage Payments	Allow the admin to add a payment method	None	A payment method is added	The **admin** indicates their intention to add a payment method The **system** presents the admin with a form to add a payment method The **admin** fills the form The **system** adds a payment method	None

Table 4.3: Admin Use Cases

4.3 Functional Requirements

This section lists all Functional Requirements that are expected to be developed and assigns them a level of priority according to their importance in the project.

There are 3 levels of priority that can be assign to the requirements:

- Must - critical and crucial requirements that the system must have;
- Should - important requirements that the system should have, but not necessary for the system to function properly;
- Could - requirements that the system could have to improve the user experience.

The lists of Functional Requirements for the User, Bakery and Admin are presented below in the following tables 4.4, 4.5 and 4.6, respectively.

User		
Action	**Requirement**	**Priority**
1. Login	1.1. Login with email	Must
	1.2. Login with *Facebook*	Could
	1.3. Login with *Google*	Could
	1.4. Retrieve password	Must
2. Register	2.1. Register with email	Must
	2.2. Register with *Facebook*	Could
	2.3. Register with *Google*	Could
	2.4. Registration confirmation	Should
	2.5. Captcha verification	Should
3. Home Screen	3.1. Menu bar	Must
	3.2. Show popular bakeries	Must
	3.3. Show best sellers	Must
	3.4. Show favorites	Must
	3.5. Show popular products	Should
	3.6. Check existence of bakeries in a location	Should
	3.7. Show promotions	Could
4. Profile and Settings	4.1. Show info	Must
	4.2. Change profile data	Must
	4.3. Change password	Must
	4.4. Change email	Must
	4.5. Add, edit and remove addresses	Must
	4.6. Select main address	Should
	4.7. Consult purchases history	Must
	4.8. Add and remove favorite bakery	Should
	4.9. Add and remove favorite product	Should
	4.10. Consult purchases stats	Could
5. Search	5.1. Search for bakeries	Must
	5.2. Search for products	Must
	5.3. Add and remove search filters	Must
	5.4. Go to bakery page	Must
	5.5. Change order of search	Should
6. Bakeries	6.1. Show info	Must
	6.2. Show products	Must
	6.3. Show promotions	Should

User		
Action	**Requirement**	**Priority**
	6.4. Show location in google maps	Could
	6.5. Show specialties of the bakery	Should
	6.6. Select products to purchase	Must
	6.7. Rate bakery	Should
	6.8. Rate product	Should
	6.9. Comment bakery	Could
7. Checkout	7.1. Show products	Must
	7.2. Select delivery dates	Must
	7.3. Select delivery address	Must
	7.4. Select payment method	Must
	7.5. Select delivery hour	Should
	7.6. Write observations	Should
	7.7. Confirm purchase	Must
	7.8. Cancel purchase	Must
	7.9. Pay	Must
8. Client Area	8.1. Show information	Must
	8.2. Show contact	Must
	8.3. Show pre-registration form	Must
	8.4. Redirect to bakeries login	Must
	8.5. Send message to the team	Should
9. about	9.1. Show information	Must
	9.2. Show contact	Must
	9.3. Send message to the team	Should

Table 4.4: User Functional Requirements

Bakery		
Action	**Requirement**	**Priority**
1. Login	1.1. Login with email	Must
	1.2. Retrieve password	Must
2. Pre-registration	2.1. Pre-register	Must
	2.2. Captcha verification	Must
3. Profile and Settings	3.1. Show bakery profile	Must
	3.2. Change profile data	Must
	3.3. Change password	Must
	3.4. Change email	Must
	3.5. Add and remove specialities of the bakery	Should
	3.6. Enable or disable payment methods	Should
	3.7. Add and remove photos	Should
	3.8. Select range of delivery	Must
4. Products	4.1. Show products	Must
	4.2. Add, edit and remove products	Must
	4.3. Add photos to products	Must
	4.4. Assign categories and sub-categories to products	Must
	4.5. Add and remove promotions	Should
	4.6. Add search filter	Should
	4.7. Change order of the products	Could
5. Consult Sales	5.1. Show sales history	Must
	5.2. Change order of sales history	Should
	5.3. Generate graph of sales	Must

Bakery		
Action	**Requirement**	**Priority**
	5.4. Generate PDF with sales	Should
6. Consult Sales	6.1. Show deliveries for specific day	Must
	6.2. Generate PDF with deliveries	Must
	6.3. Slide between dates	Must
	6.4. Show special orders	Should
7. Employees	7.1. Add and remove employees	Must
	7.2. Show employees	Must
8. Contract	8.1. Show contract information	Must
	8.2. Renew contract	Should

Table 4.5: Bakery Functional Requirements

Admin		
Action	**Requirement**	**Priority**
1. Manage Bakeries	1.1. Add bakeries	Must
	1.2. List all bakeries	Must
	1.3. Show bakery information	Must
	1.4. Add search filter	Should
	1.5. Change order of search	Should
2. Manage Users	2.1. List users	Must
	2.2 Ban user	Should
3. Manage Workers	3.1. Add workers	Must
	3.2. List workers	Red
	3.3. Show worker information	Red
	3.4. Add search filter	Should
	3.5. Change order of search	Should
4. Manage Categories	4.1. Add categories or sub-categories	Must
	4.2. List all categories and sub-categories	Must
5. Manage Payments	5.1. Add payment method	Must
	5.2. List all payment methods	Must

Table 4.6: Admin Functional Requirements

4.4 Non-Functional Requirements

The Non-functional Requirements are related to the quality and operation of the system, rather than the functionalities themselves. They describe and try to quantify some characteristics such as how well the system behaves, how reliable it is or how fast it performs. [39]

These requirements were defined by HYP and they are represented in the following tables 4.7 and 4.8. The first table represents the process requirements and the second table represents the product requirements.

Attribute	Requirement	Description	Goal	Priority
1. Implementation	1.1. Programming Language	Language in which the project needs to be developed	Ruby	Must
	1.2. Web Application Framework	Framework to be used to build the web application	Ruby on Rails	Must
	1.3. Database System	Open Source DBMS that should be used to store the data of the project	PostgreSQL	Should
	1.4. Front-End Developing Languages	Programming languages that need to be used to implement the front-end of the web application	Javascript / HTML / SASS	Must
	1.5. Version Control System	System that controls the different versions of the application during its development	Git	Must
	1.6. Git Repository	Git repository manager where the project should be hosted	GitLab	Should
2. Standards	2.1. General	Programming standards that HYP uses in the development of their applications	Follow the standards	Must
	2.2. Specific	Code conventions and standards used in Ruby on Rails	Follow the standards	Should
3. Organization	3.1. Tasks manager	Tool to help managing and organizing the tasks of the project	Trello	Must
	3.2. File storage	Service used to store and synchronize the files of the project	Google Drive	Must

Table 4.7: Process Non-Functional Requirements

Attribute	Requirement	Description	Goal	Priority
1. Performance	1.1. Max Response Time	Time that takes the application to respond to a request	8 seconds [40]	Should
	1.2. Throughput	Number of operations that the system can handle per second	6000 requests [41]	Should
2. Reliability	2.1. High Availability	Expected percentage of uptime of the system per year	99% [42]	Should
	2.2. Failure Rate	Average number of work hours between failures	400 hours	Could
3. Security	3.1. Unauthorised Access	Prevent users from accessing pages that they are not allowed	User can not access pages that they do not have access to	Must
	3.2. Payments	The payments should be dealt by external entities	PayPal / MBNet	Must
	3.5. Encrypted Data	All the critical information stored in the database should be encrypted	Passwords need to be encrypted	Must
	3.5. Attacks	Types of attacks that the system should prevent	Session / Cross-Site Request Forgery / SQL Injection / Cross-Site Scripting	Must
4. Privacy	4.1 Data Privacy	The user information should be private to others users	Users can not see other users information	Must

Attribute	Requirement	Description	Goal	Priority
5. Recover	5.1. Backup Data	All data stored in the database should have a backup	Keep a backup of the data at all times	Could
6. Portability	6.1. Mobile Devices	The application functions on mobile devices browsers with a appropriate design	Responsive web platform	Should
7. Compatibility	7.1. Multiple Browsers	The web application can function properly in the most popular browsers	Chrome / Firefox / Safari / Edge / Opera	Should
8. Robustness	8.1. Reboot Time	Amount of time that takes for the system to restart after a failure	30 minutes	Should
9. Maintainability	9.1. System Architecture	The application follows an architecture model	Model-View-Controller	Must
	9.2. Code Organization	The code follows a developing standard	Ruby on Rails standard	Must
11. Extensibility	11.1. Features	It is easy to add new features in the system	Code needs to be organized and easy to understand	Should
12. Readability	12.1. Understandable	The text is easy to understand for everyone	Do not use complex and complicated words	Should
13. Usability	13.1. Effort	The effort that takes a user to accomplish an action should be the smallest possible	Maximum of four clicks to execute a key action	Should
	13.2. Learn	The learning time needed for a user purchase a product	30 minutes	Should
	13.3. Information	The user receives feedback when he does something within the application	Errors and notices messages	Must

Table 4.8: Product Non-Functional Requirements

4.5 System Architecture

This section describes the system architecture used in the development of this project. Since the project was developed in Ruby on Rails, and this framework uses the Model-View-Controller architecture, it is only logical that this project uses this architecture as well.

Model-View-Controller (MVC) architecture [5] is one of the most popular architectures to develop web platforms. This architecture divides an platform into three different parts:

- **Model** - this component job is to manage the data and to notify the Views and Controllers when some change has been made. It also has to answer to requests from the Controller;
- **View** - the View generates a visual representation to present to the users, and answers to the requests from the controller;
- **Controller** - this component connects the three components. It sends requests to the Model to update their state or to retrieve information and sends request to the View to change the presentation of the Model;

In the following image 4.1 a representation of the MVC architectural model is presented.

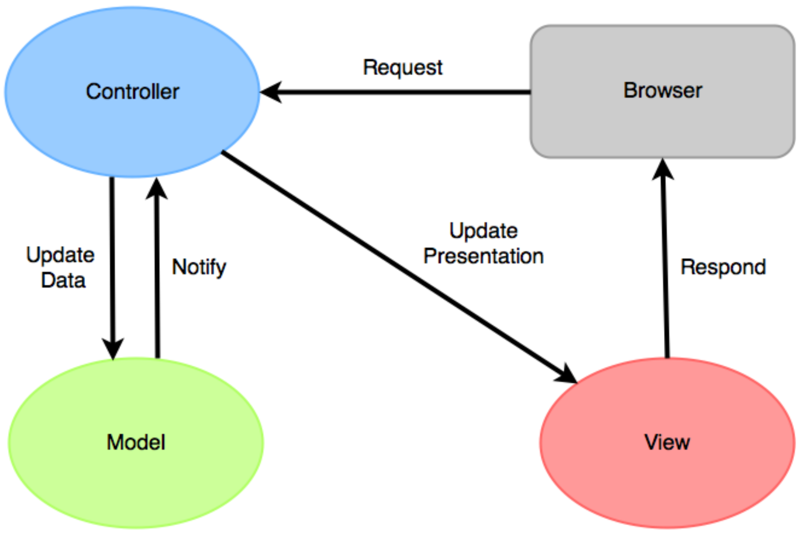


Figure 4.1: MVC Architectural Model

With this understanding of how the MVC architecture works, it is possible to comprehend how it is applied to this project. In this project the Model is used to manage the interaction of the system with the database in order to store, update, retrieve, remove and validate the data. It also contains all business logic operations and states of the platform. The View refers to the front-end of the platform. In other words, everything that is displayed to the user in the browser. This display of data is achieved through views that present the content in format of HTML5, XML, etc. Finally the Controller is the main component in the architecture. It is the one responsible for controlling everything in the project by interacting with the Models and the Views simultaneously. Initially it receives requests from the browser and then starts to deal with them by processing data from the Models and give this data to the Views to be presented.

4.6 Entity-Relationship Diagram

In order to show how the data of this project is organized and related, a Entity-Relationship Diagram was created. This diagram has entities composed by attributes that are connected with each other through relationships. Those relationships are used to describe the dependencies between the entities. With this diagram we can have a better visualization of how the database of the application will be.

The image 4.2 has the representation of the Entity-Relationship Diagram of this project. For the purpose of making the diagram as simple an easy to understand and interpret as possible, it is color coded:

- **Red Entities** - entities related to the user
  - **User** - entity containing the user information;
  - **Shopping Bag** - this entity contains the products that are inside the user shopping bag.
- **Green Entities** - entities related to the products
  - **Product** - entity containing the products information;
  - **Category / Sub-Category** - these entities contain the category and sub-categories that the product belongs to.
- **Blue Entities** - entities related to the bakeries
  - **Bakery** - this entity contains the information of the bakeries;
  - **Worker** - this entity contains information about the workers of a bakery;
  - **Contract** - entity containing the contract information of a bakery;
  - **Promotion** - entity that contains the active promotions of a bakery;
  - **Delivery Hour** - this entity contains the delivery hours of a bakery;
  - **Delivery Tax** - this entity contains the delivery taxes of a bakery;
  - **Photo** - this entity contains the photos that are inside the gallery of a bakery;
  - **Rating** - this entity contains the ratings given by the users of a bakery;
  - **Notification Request** - entity containing the requests to notify the users.
- **Orange Entities** - entities related to the purchases
  - **Purchase** - entity that contains the purchases information;
  - **Purchase Product** - entity containing the quantity of each product of a purchase;
- **Grey Entities** - other entities;
  - **Admin** - this entity contains the admins information of the platform.
  - **Address** - entity that contains both the users and the bakeries addresses;
  - **Payment** - entity containing the payment methods available in the platform.

Noteworthy, not all attributes that will be in the final database are represented in this diagram, since that the Ruby on Rails framework creates some attributes to help with their processes, and the attribute ”id” is not presented in the diagram for the purpose of not overload the diagram with unnecessary information.

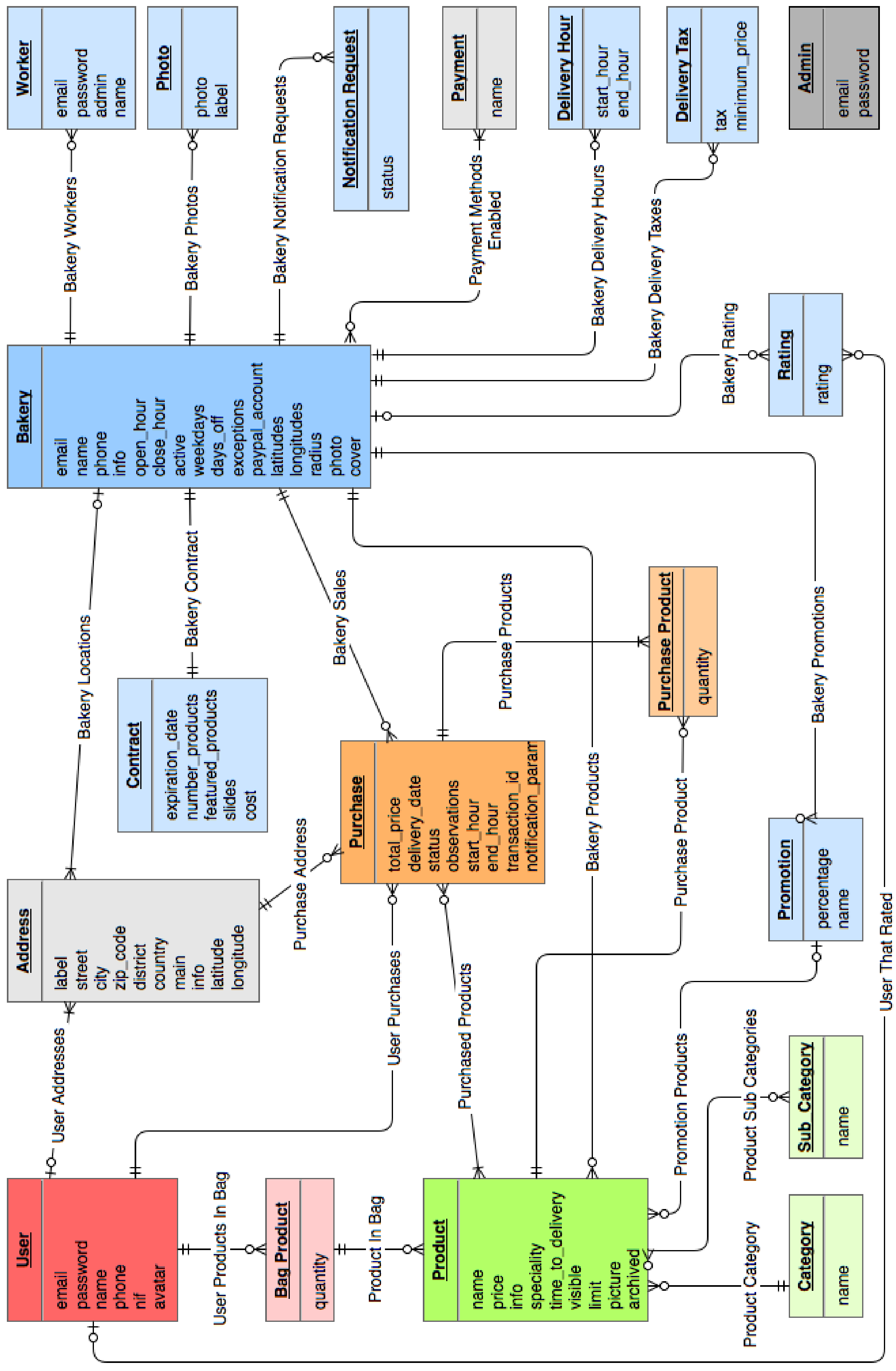


Figure 4.2: Entity Relationship Diagram

4.7 Navigation Diagram

A Navigation Diagram shows how to navigate within an application. In our case, the diagram shows how to navigate between the pages of the website. This diagram has two main objectives: present the web pages available, and show how the user can navigate between them.

The image 4.3 shows the Navigation Diagram created for this project. This diagram has all the main and most important connections between pages, missing only some of them for the purpose of making the diagram more readable and simple.

To better understand the diagram, it is divided in three main areas: the User that represents what the normal user will see and have access, the Bakery that represents what a client that has a bakery registered in the application has access to, and the Admin that represents what the Admin of the application can control and access.

In the first area, the blue and light blue rectangles represent the pages that can be access at all times, in other words, pages that can be access through the menu bar, with the exception of the light blue rectangles that are only accessible when the user is logged. The grey rectangles are pages that are not always accessible and that have other restrictions.

In the Bakery area both the orange and light orange rectangles represent which pages the admins of the bakery have a direct access to all the time. The light orange rectangle represents the pages that the employees of the bakery have access.

Finally, in the last area there are no connections represented between the pages since they all have access to each others, so there is no need to fill the diagram with connections if there are no restrictions.

4.8 Wireframes

This section presents all the wireframes created for the website. They represent the schematics of the pages and show how the elements are arranged on them. In other words, they are a basic and simple layout of what the pages will be, and are created for the main reason of showing how the platform will function and to test in a early stage of the project, the usability of the platform.

The first wireframe created refers to the first page that a user will see when opening the website, and it can be seen in the image 4.4. The others Wireframes created are divided in two parts, "User" and "Bakery", and can be seen in the subsections of this chapter.

It should be noted that some wireframes can be become outdated when compared to the current version of the website, because since they were made in a early stage of the project, some pages can have small changes to the initial idea.

4.8.1 User

This subsection has all of the created wireframes related to the area of the website for the user. In total, thirteen wireframes were created, that represent the pages of the website for the common user.

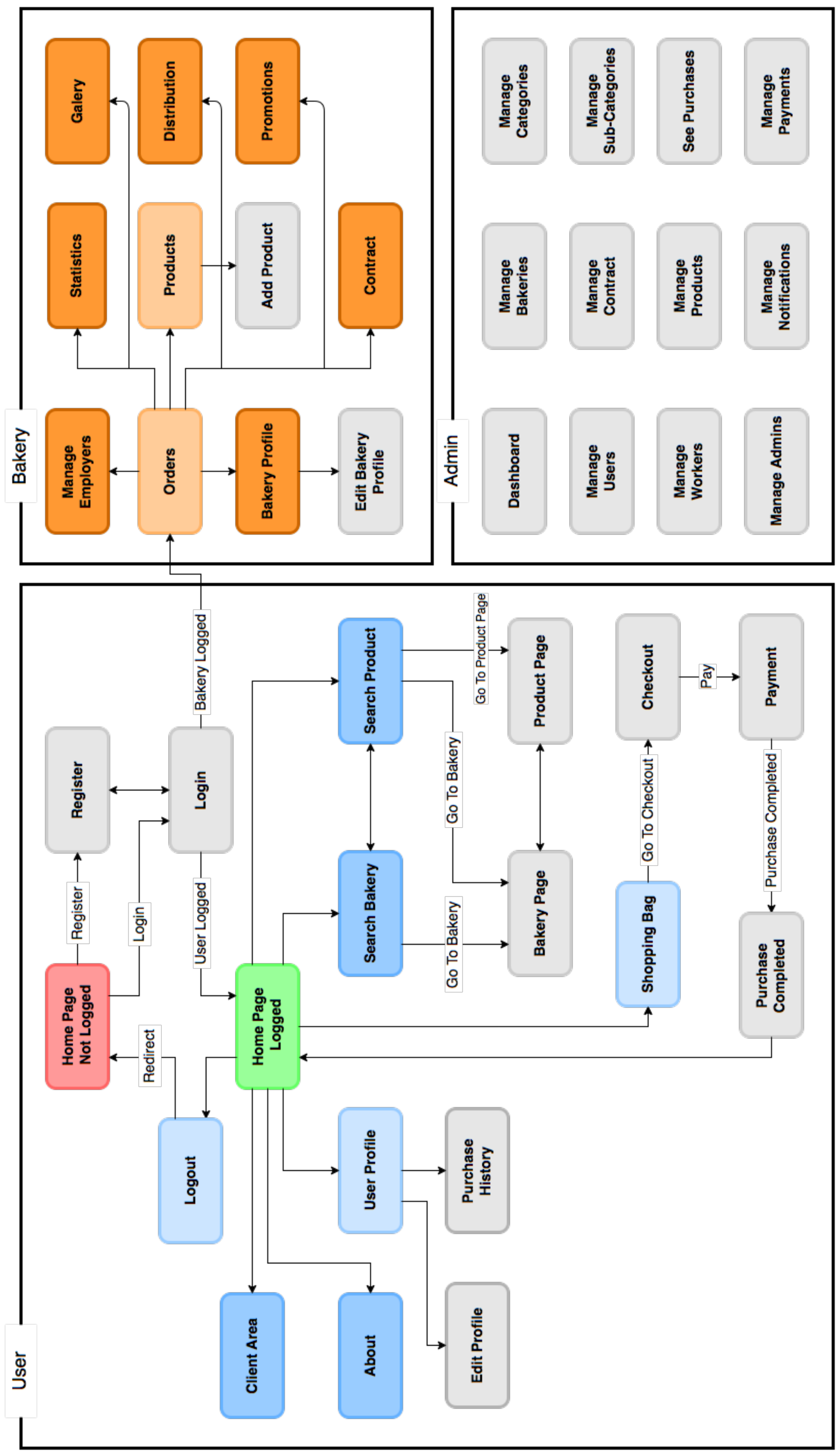


Figure 4.3: Navigation Diagram

Image 4.5 represents the first page that the users will see when they sign in, the home page. Image 4.6 represents the login where the user can access with their account and image 4.7 represents the page where the user can register in the website. When the user is logged to the website they have access to their profile. They can check their profile or check their purchases history. This actions are represented in images 4.8 and 4.9. The search for a bakery is represented on image 4.10 and the bakery page is represented on image 4.11. The search for a product is represented on image 4.12 and the way to buy or see a product is represented in the image 4.13. On the image 4.14 is represented how the user can check which products are in the shopping bag or how to reset it, and image 4.15 shows how the checkout page is. Finally, the about and client area pages are represented on images, 4.17 and 4.16, respectively.

4.8.2 Bakery

At last, in this subsection, all of the created wireframes related to the area of the website for the bakery are presented. There were created eight wireframes in total that represent the website that a Bakery has access to.

Image 4.18 represents the page when a Bakery can pre-register for the website. After the bakery signs in, they have access to the page where the list of orders is, image 4.19. The profile and statistics of the sales are represented in the images 4.20 and 4.21, respectively. Image 4.22 shows where the bakery can see their products, and the image 4.23 shows how the bakery can add more products to the inventory. Finally on image 4.24 is represented how the admin of the bakery can see and add more employees to the platform and the image 4.25 represents where the bakery can check their contract.

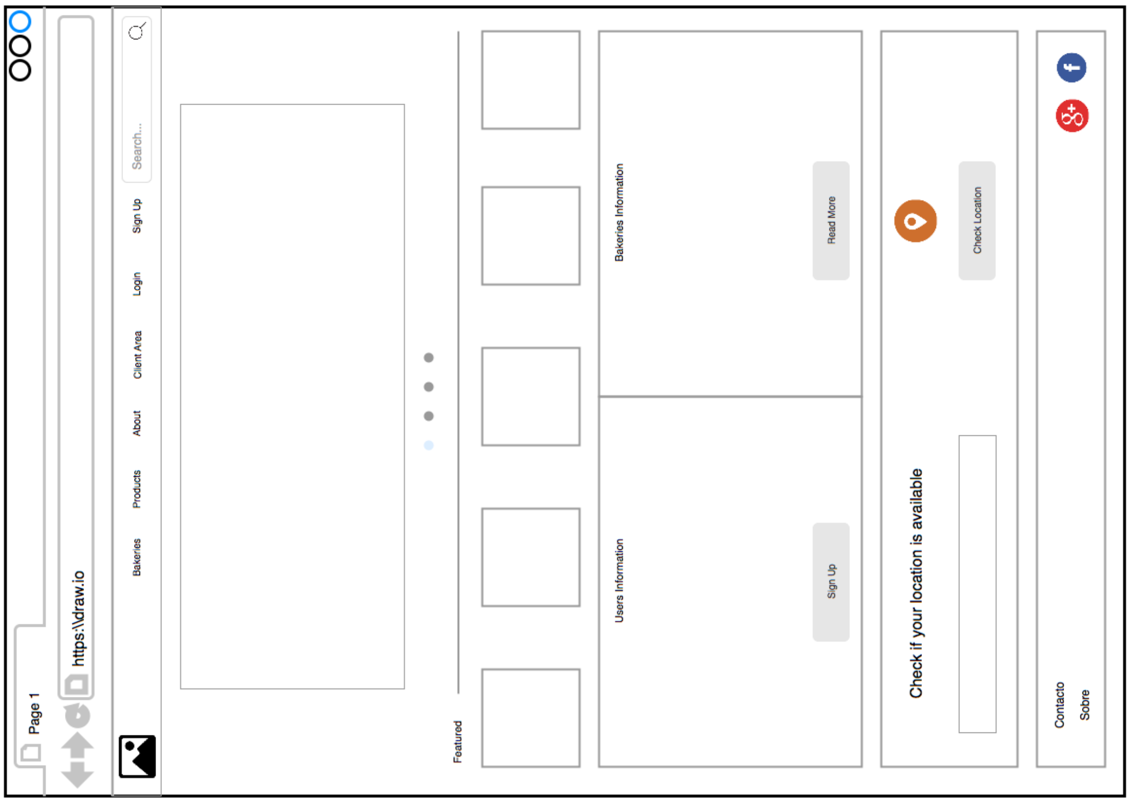


Figure 4.4: Home Page Not Logged

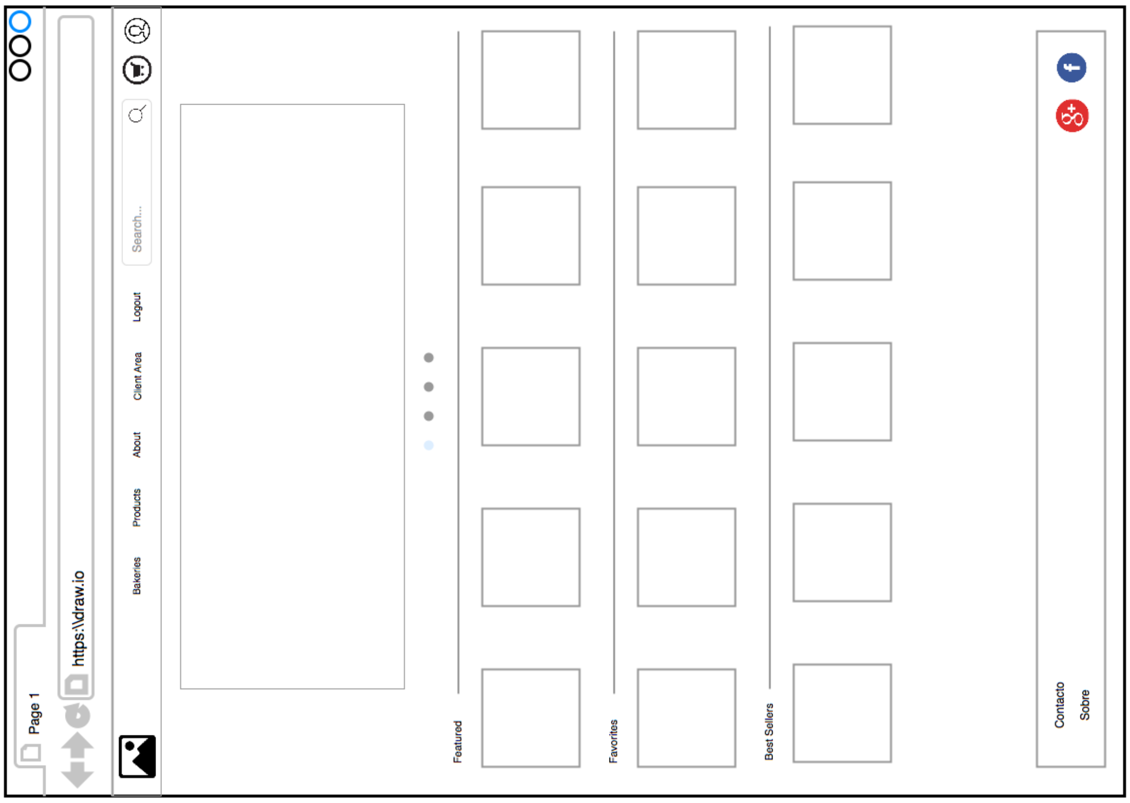


Figure 4.5: Home Page Logged

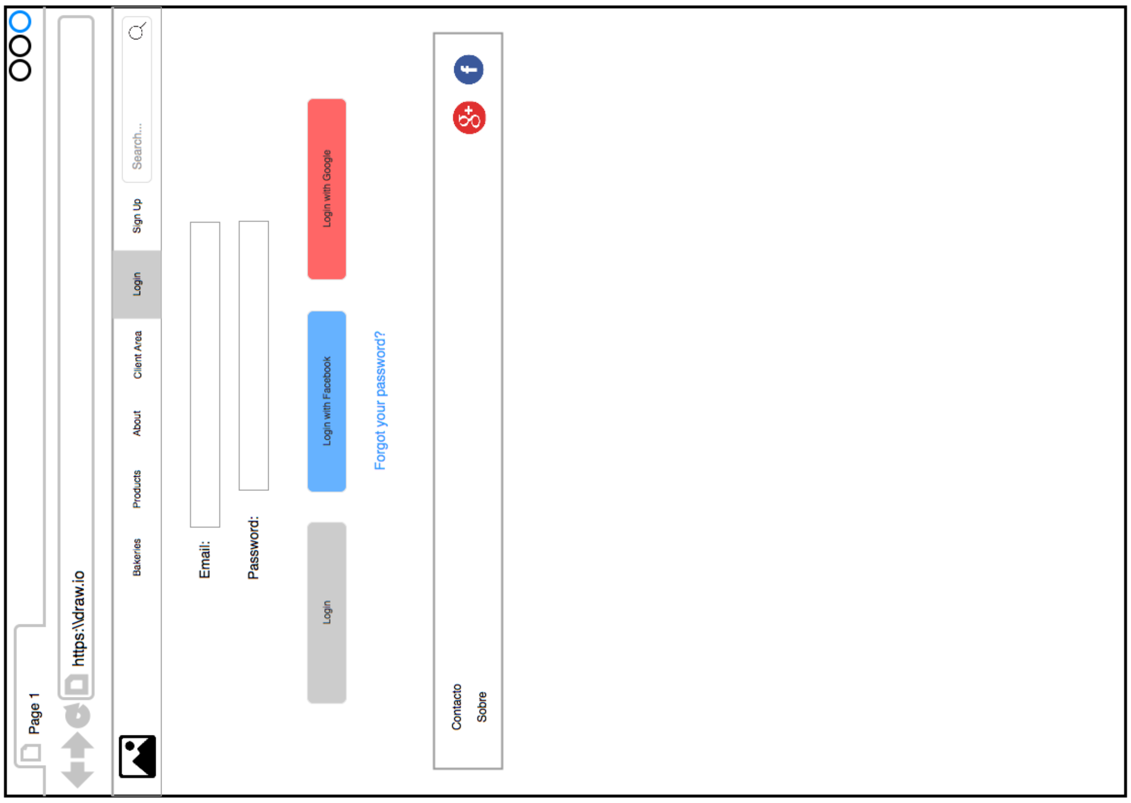


Figure 4.6: User Login

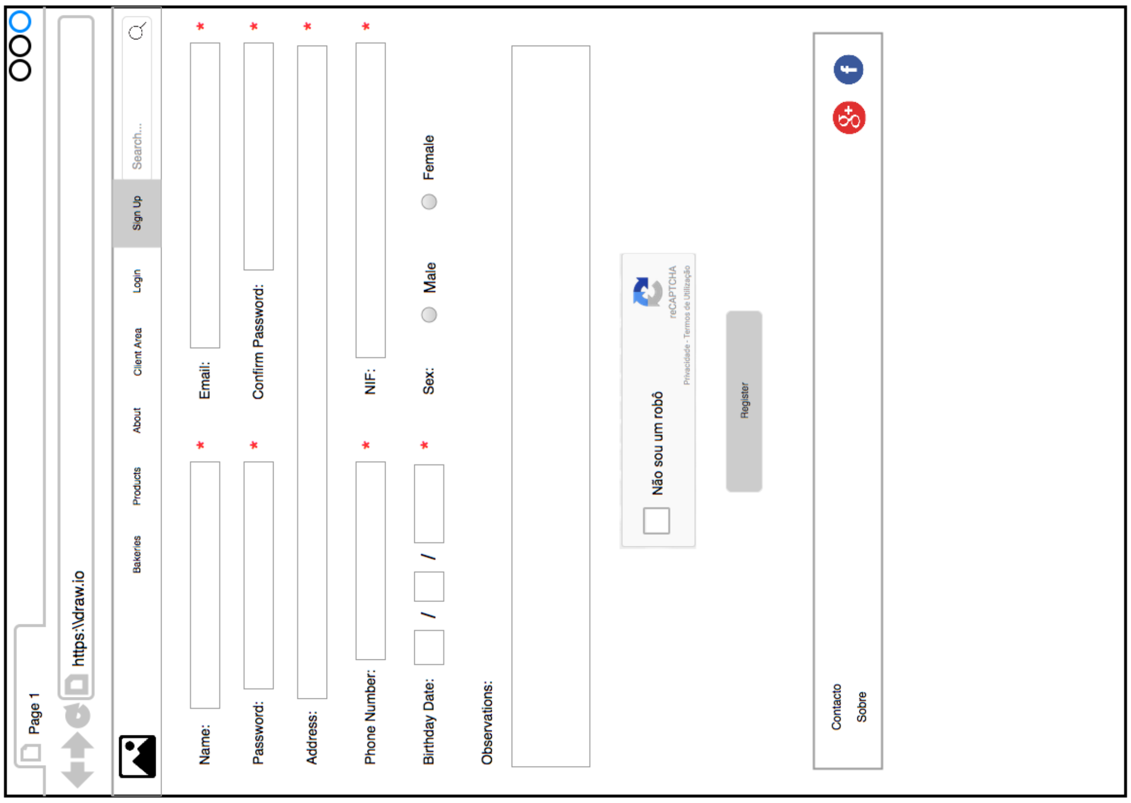


Figure 4.7: User Registration

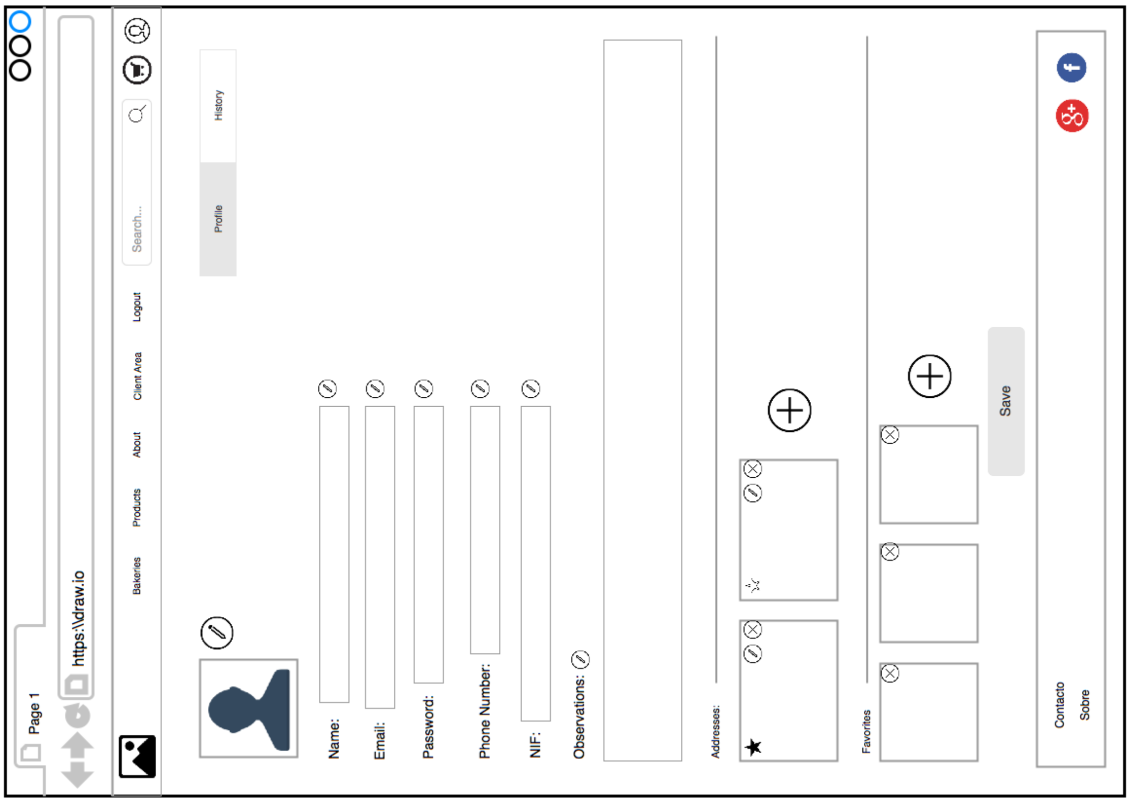


Figure 4.8: User Profile

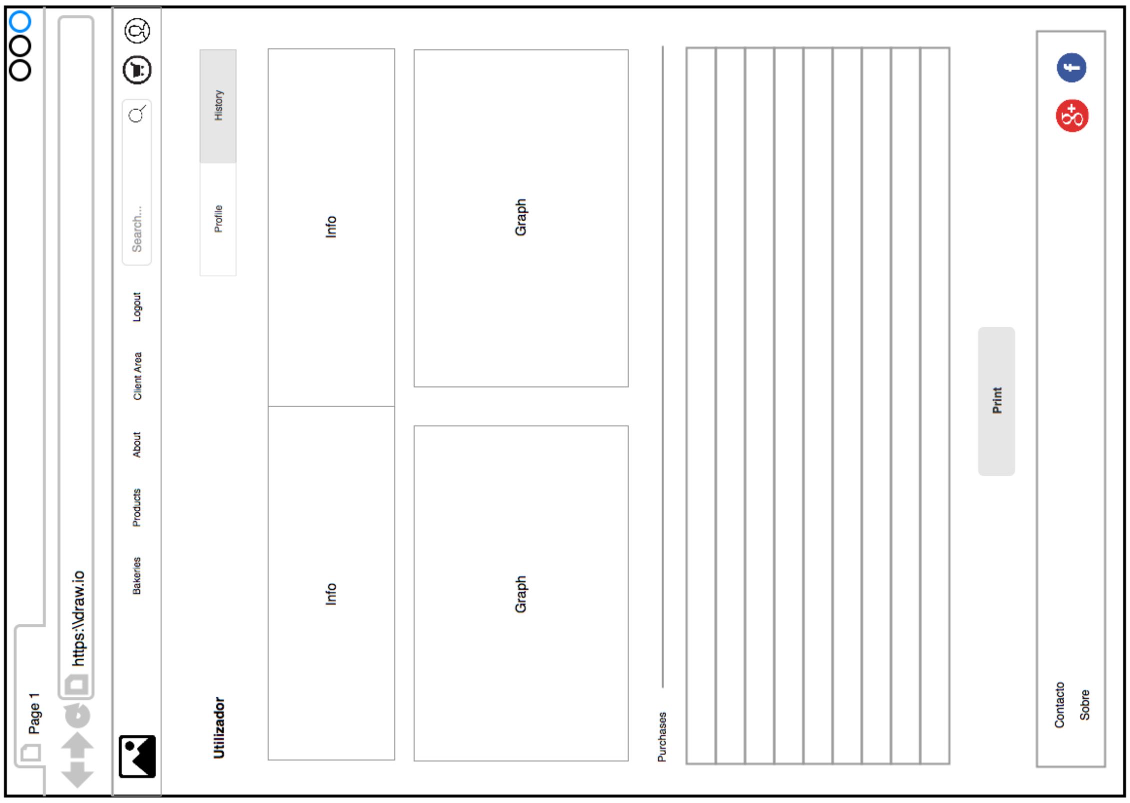


Figure 4.9: User Purchases History

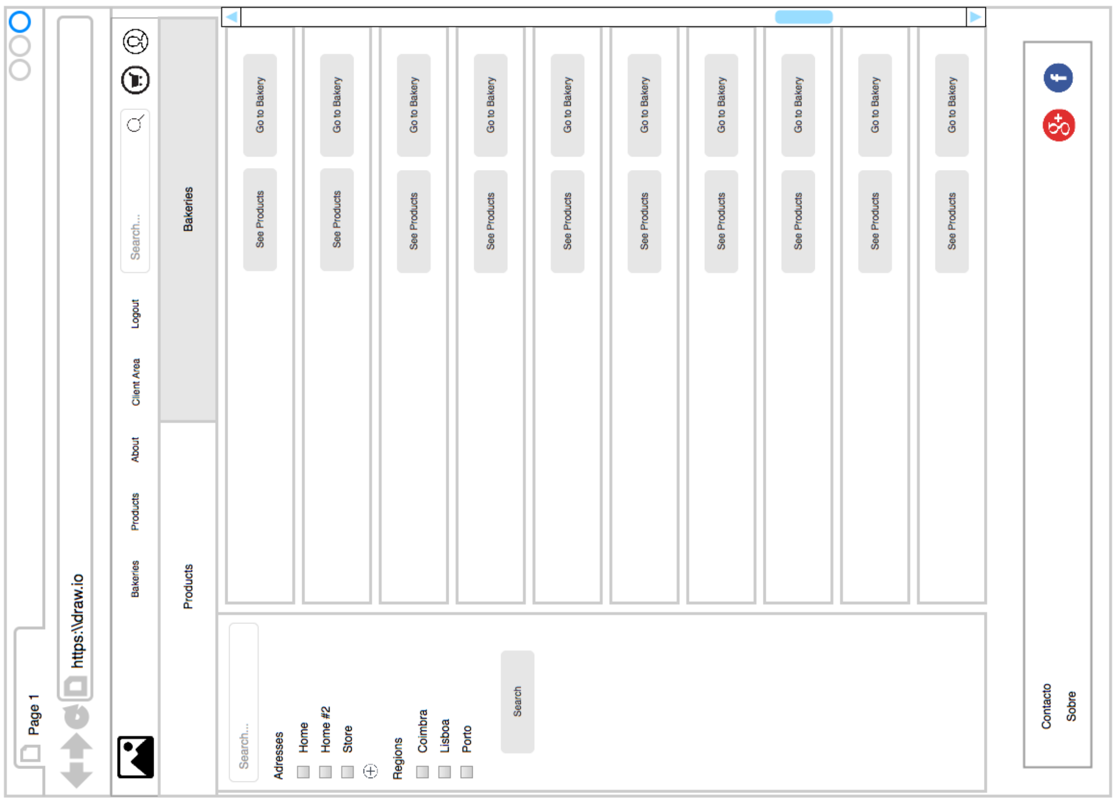


Figure 4.10: Search Bakery

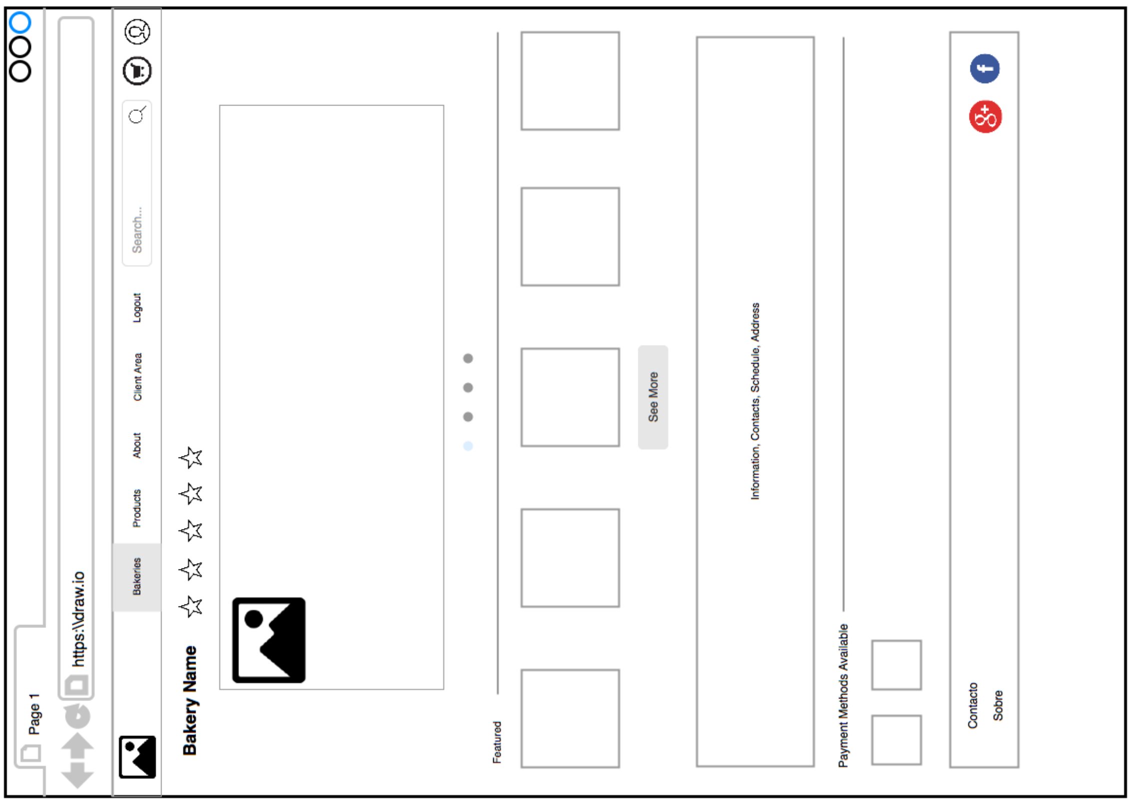


Figure 4.11: Bakery Page

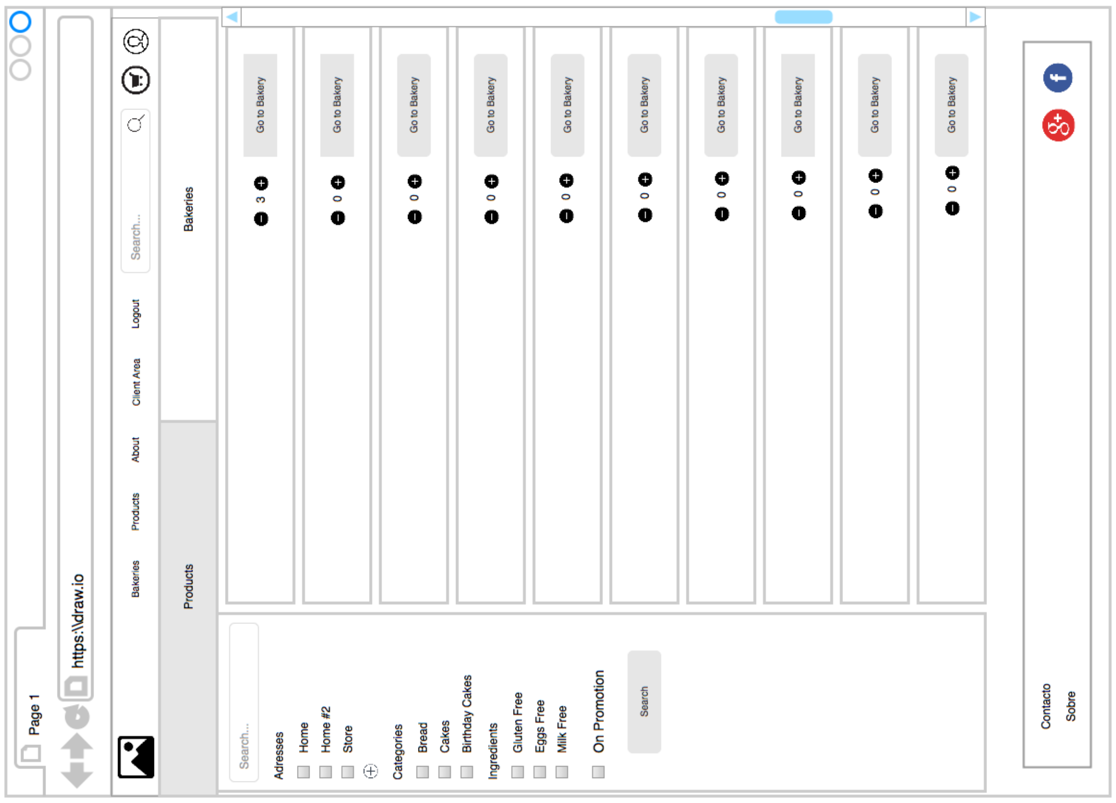


Figure 4.12: Search Product

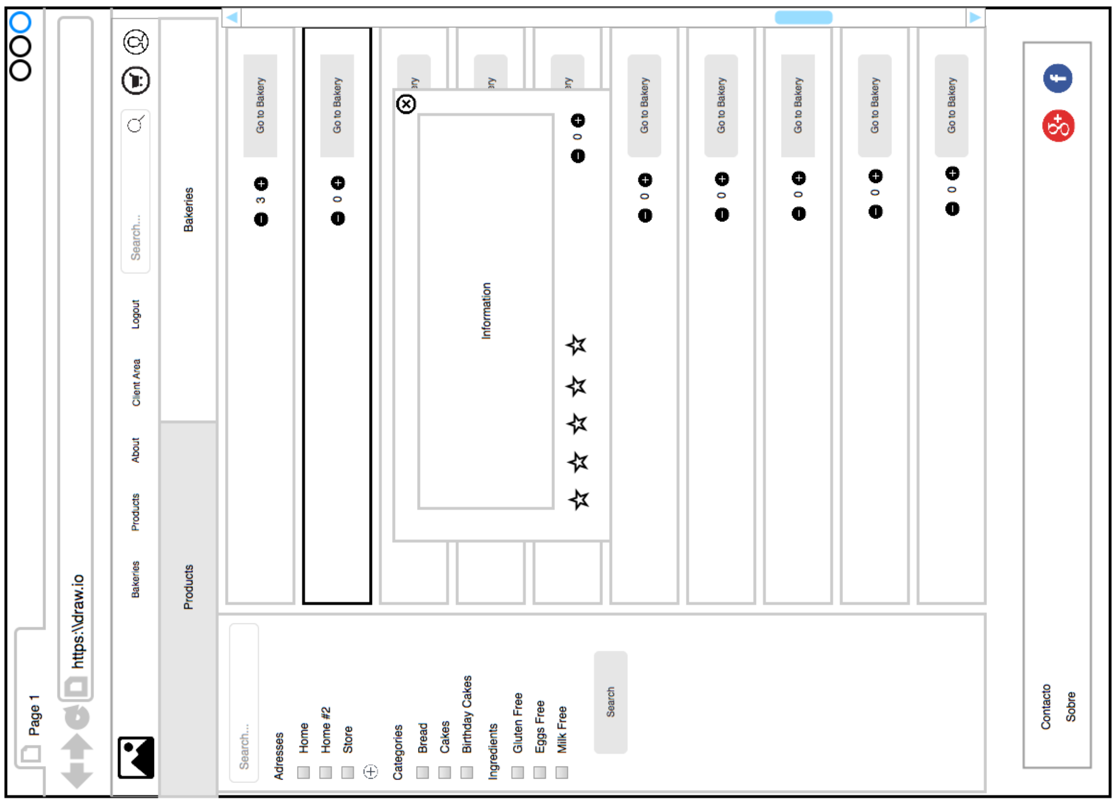


Figure 4.13: See / Buy Product

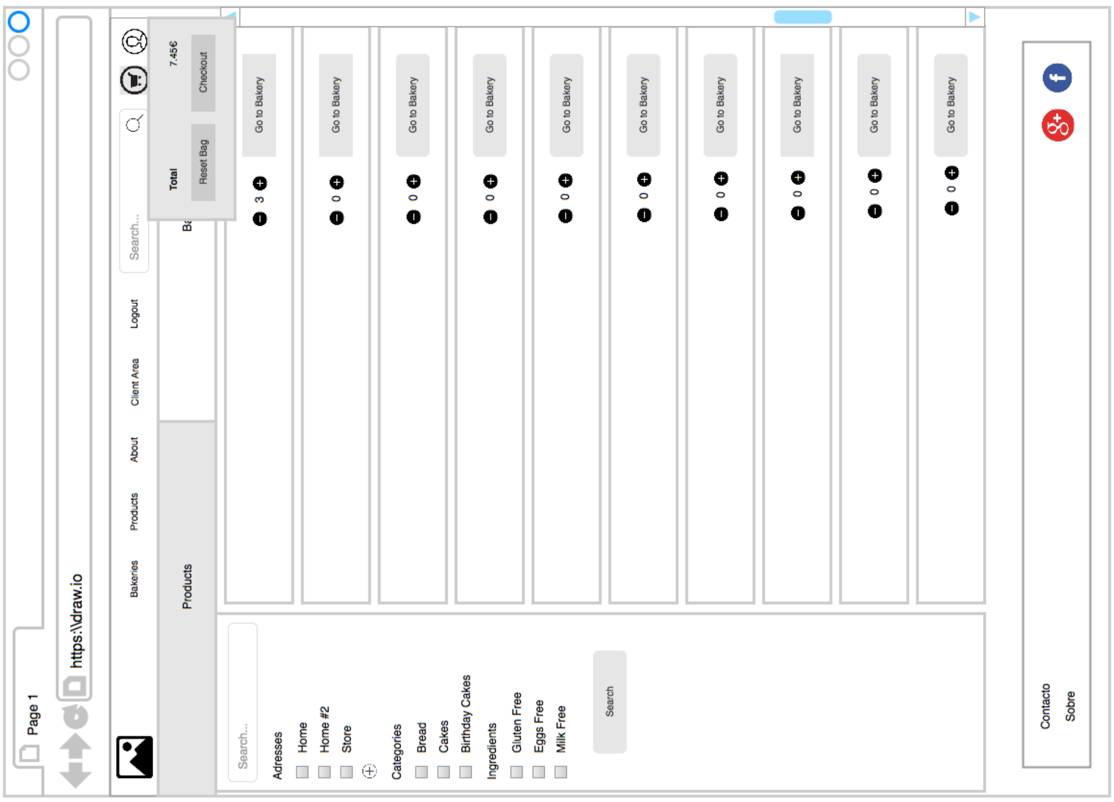


Figure 4.14: Reset / Check Bag

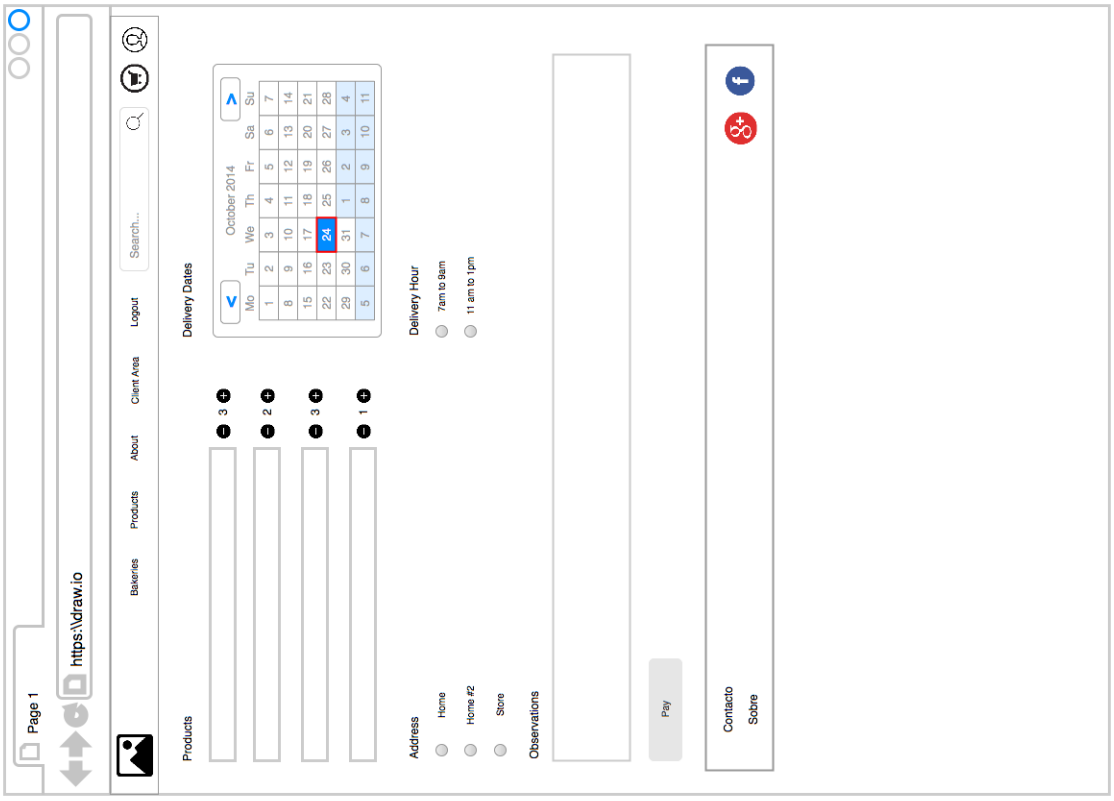


Figure 4.15: Checkout

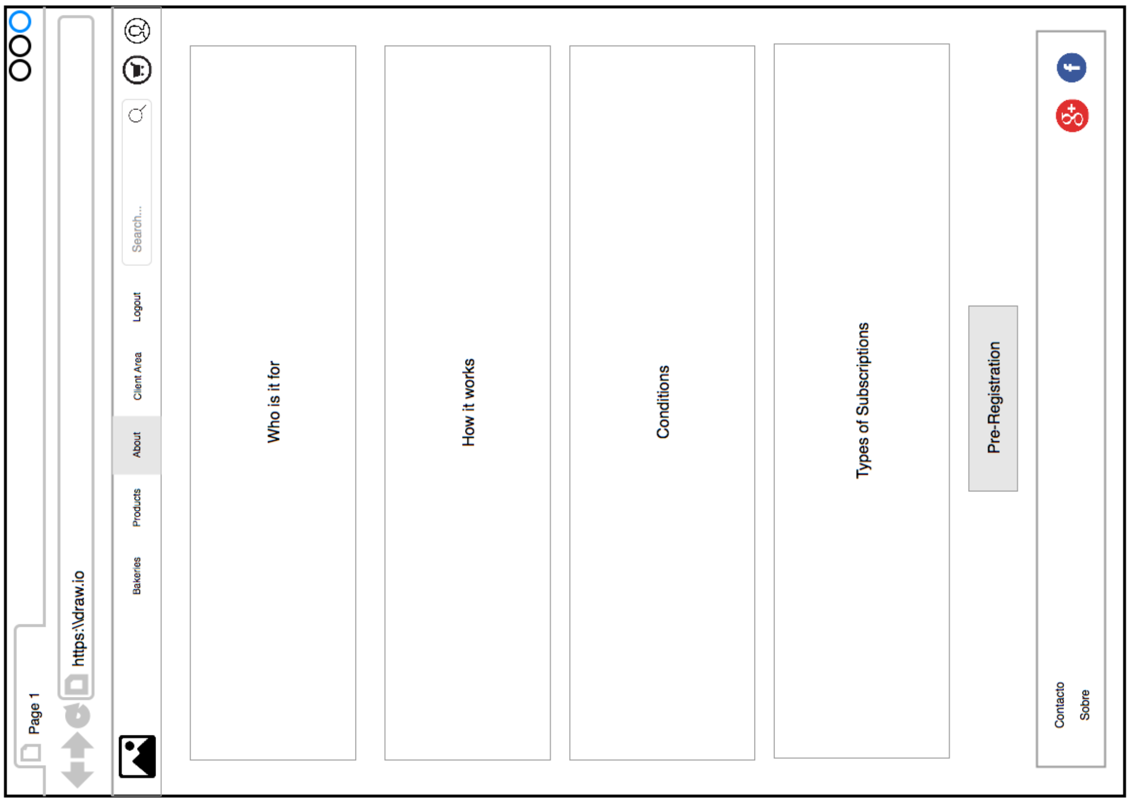


Figure 4.16: Client Area

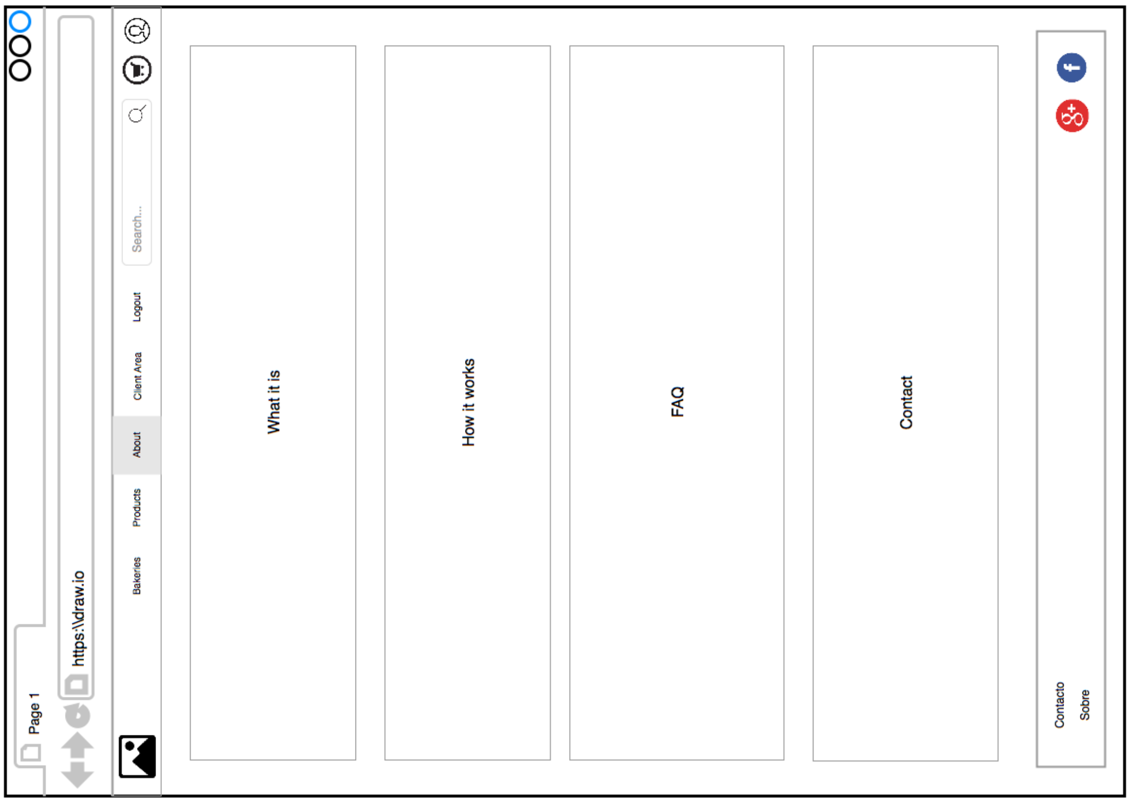


Figure 4.17: About

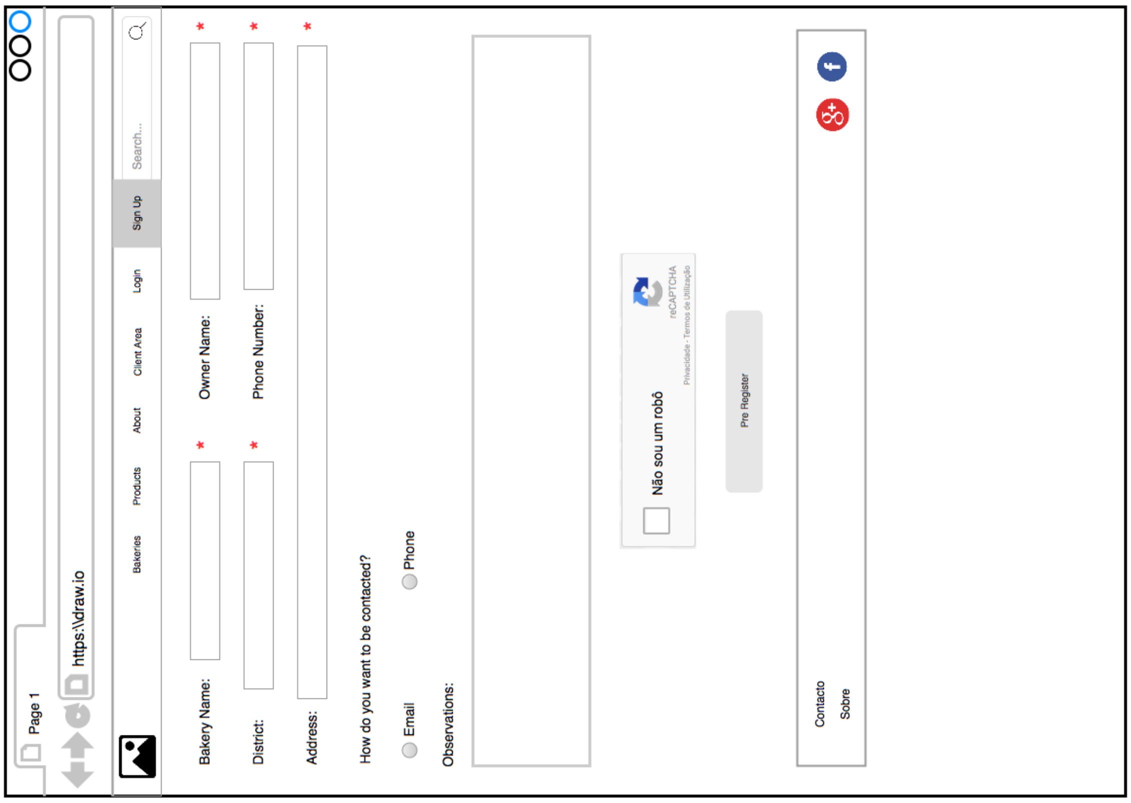


Figure 4.18: Bakery Pre-registration

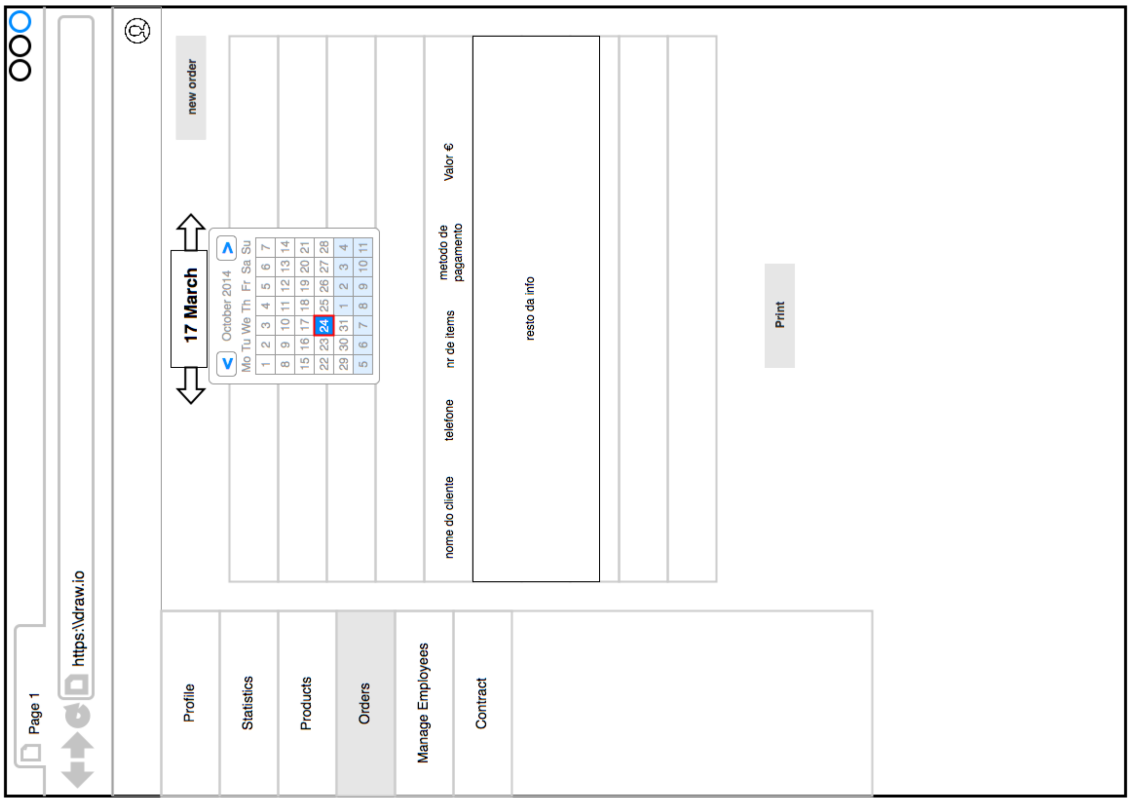


Figure 4.19: Bakery Orders

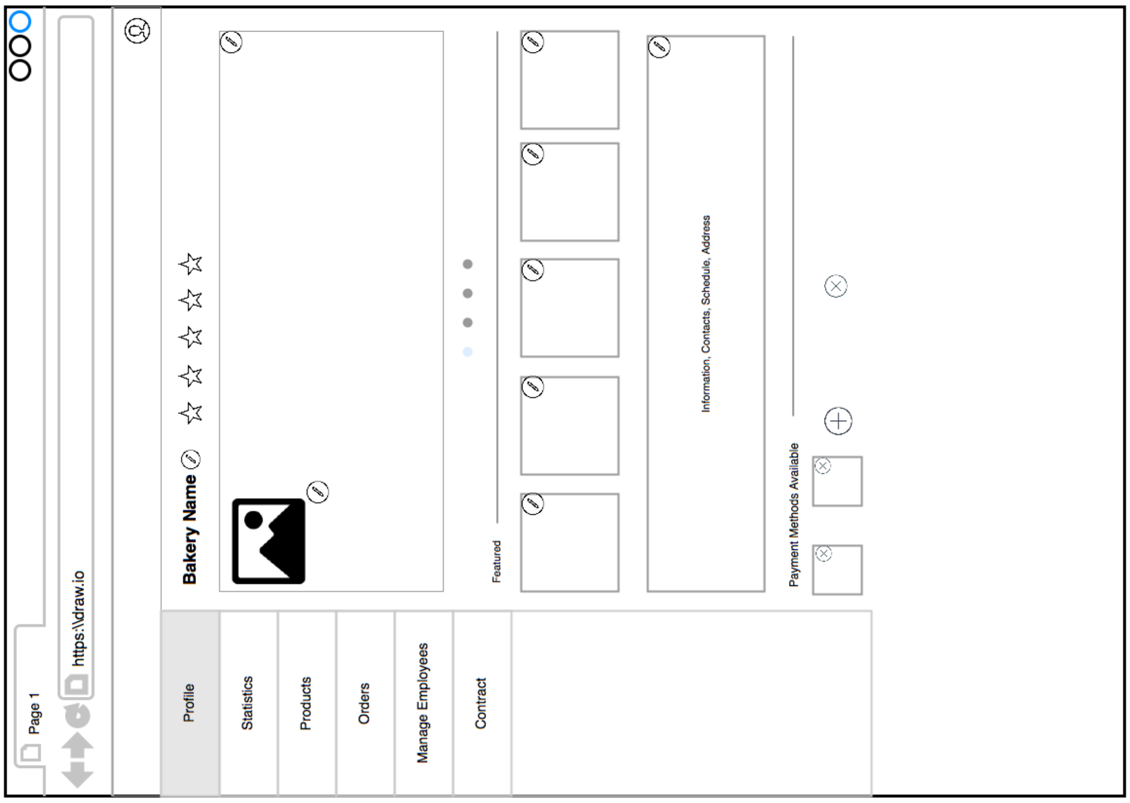


Figure 4.20: Bakery Profile

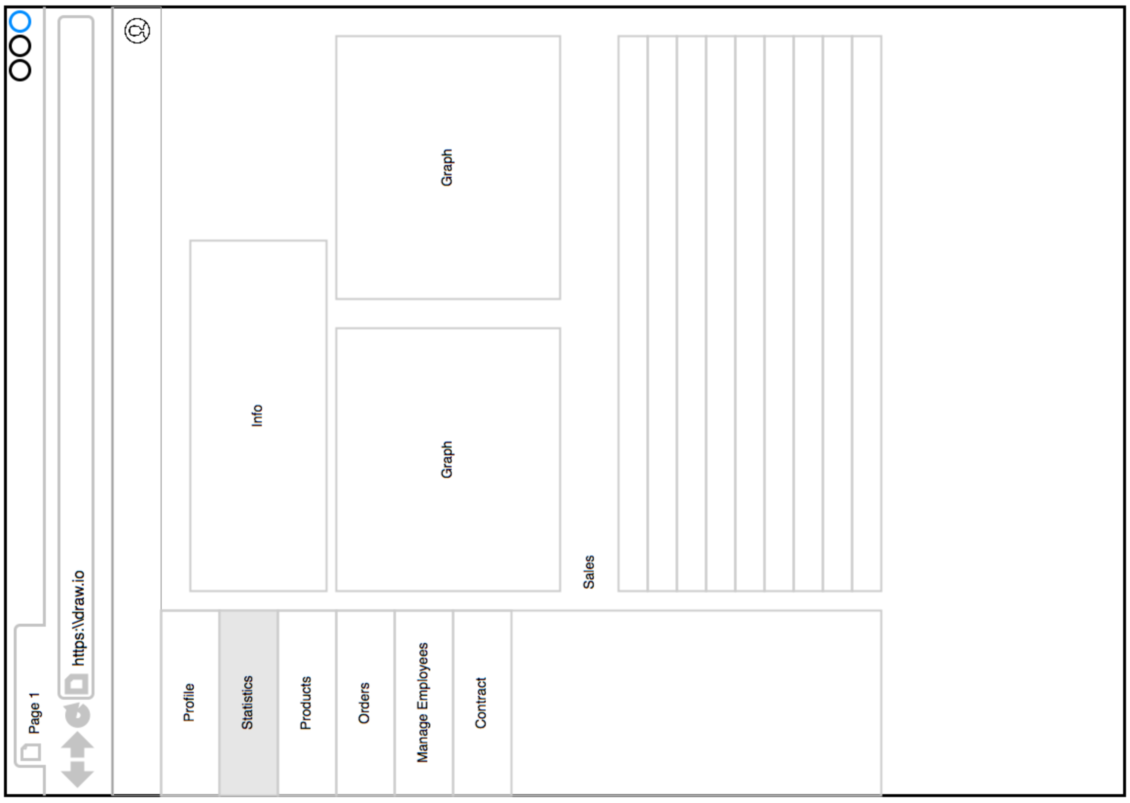


Figure 4.21: Bakery Statistics

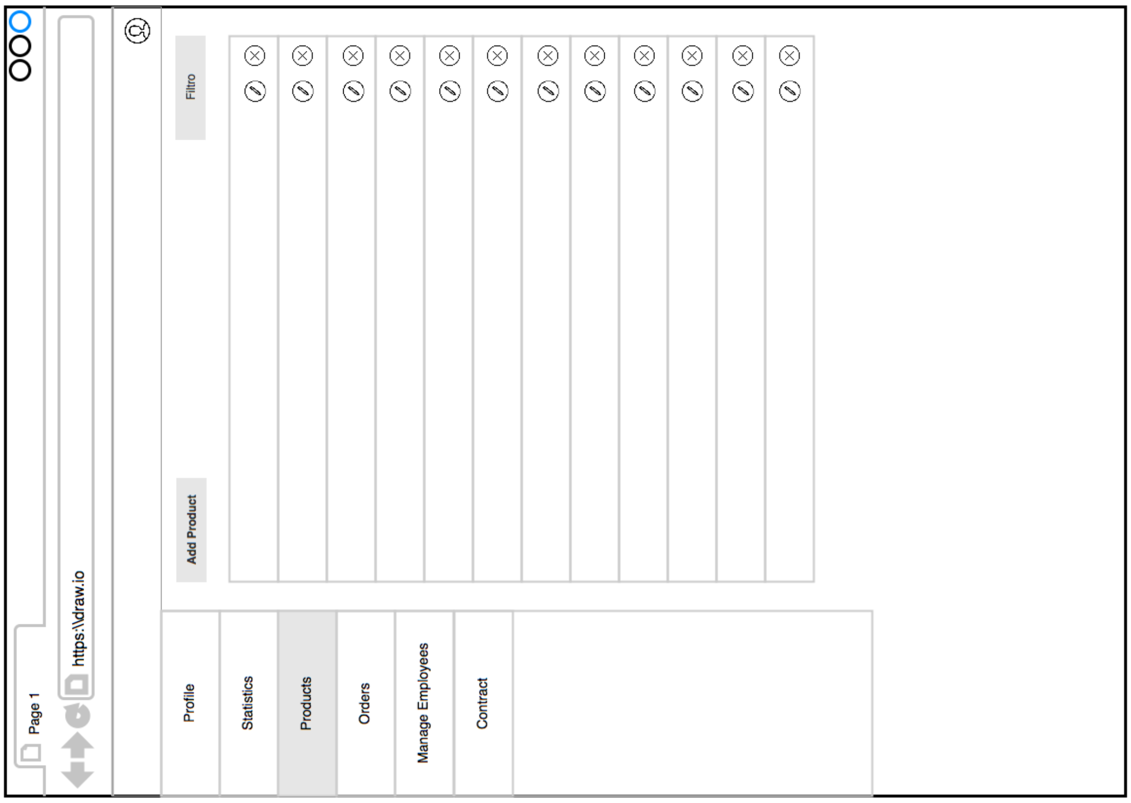


Figure 4.22: Bakery Products

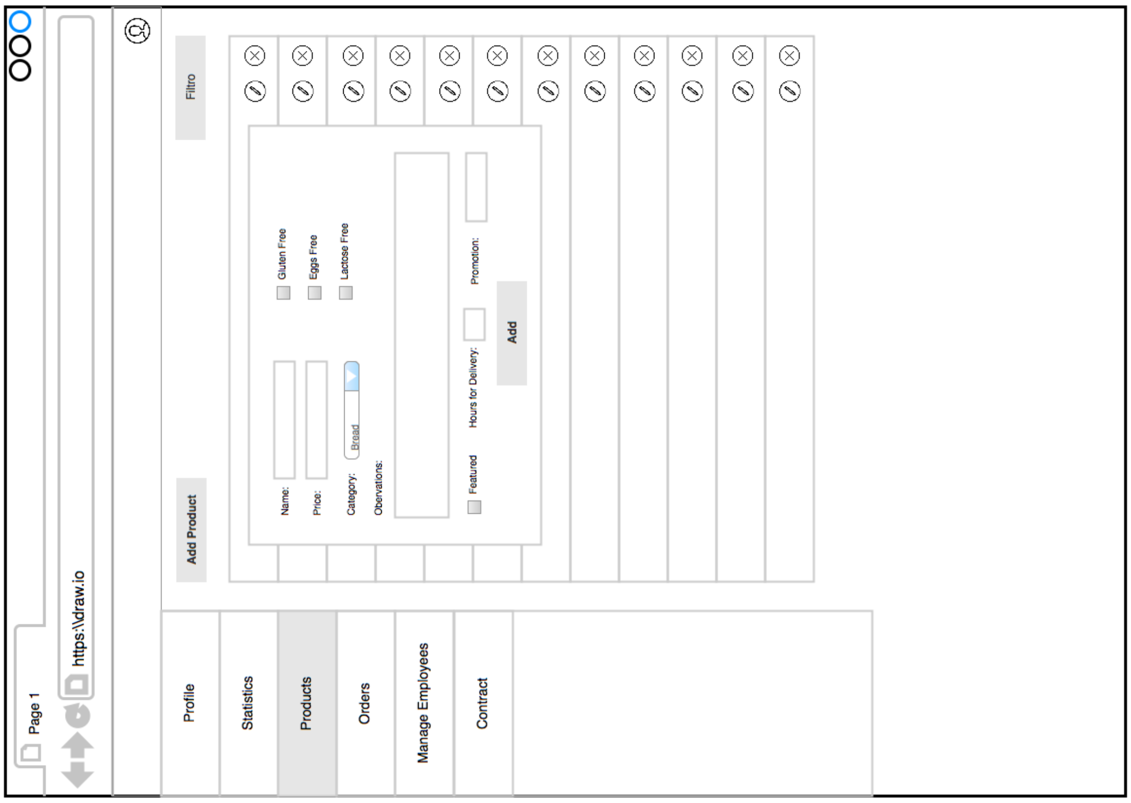


Figure 4.23: Add Product

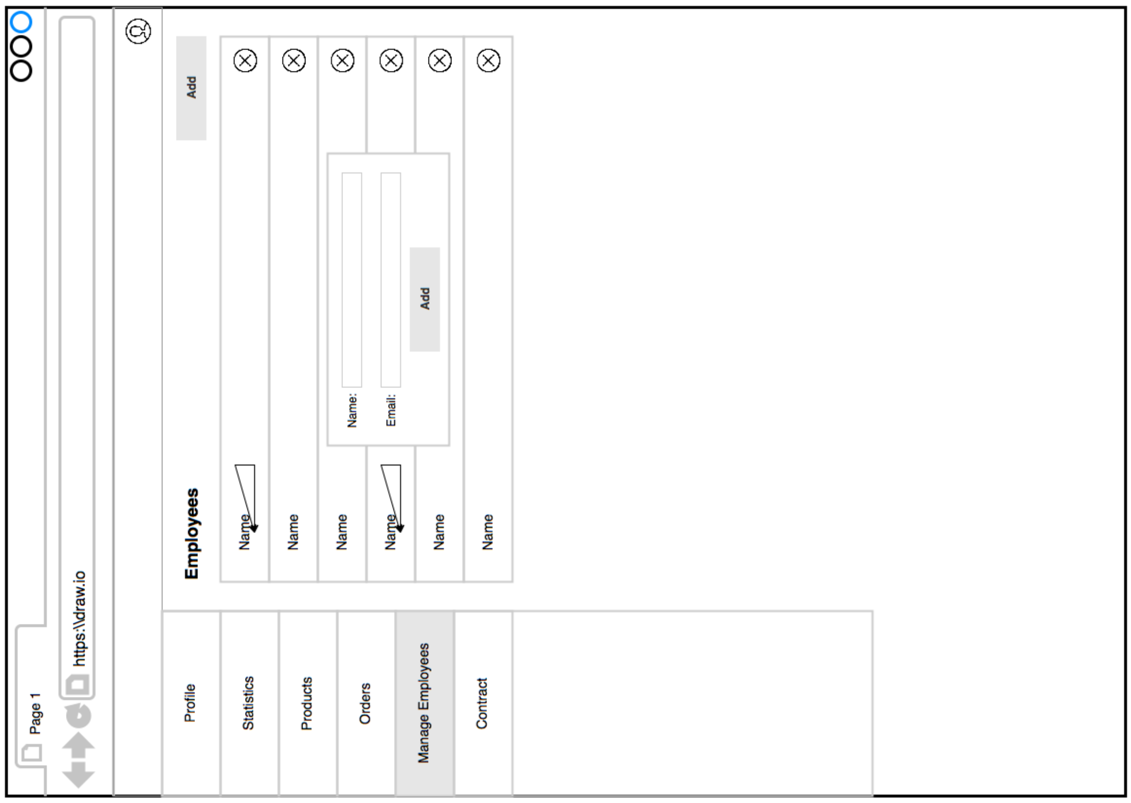


Figure 4.24: Manage Employees

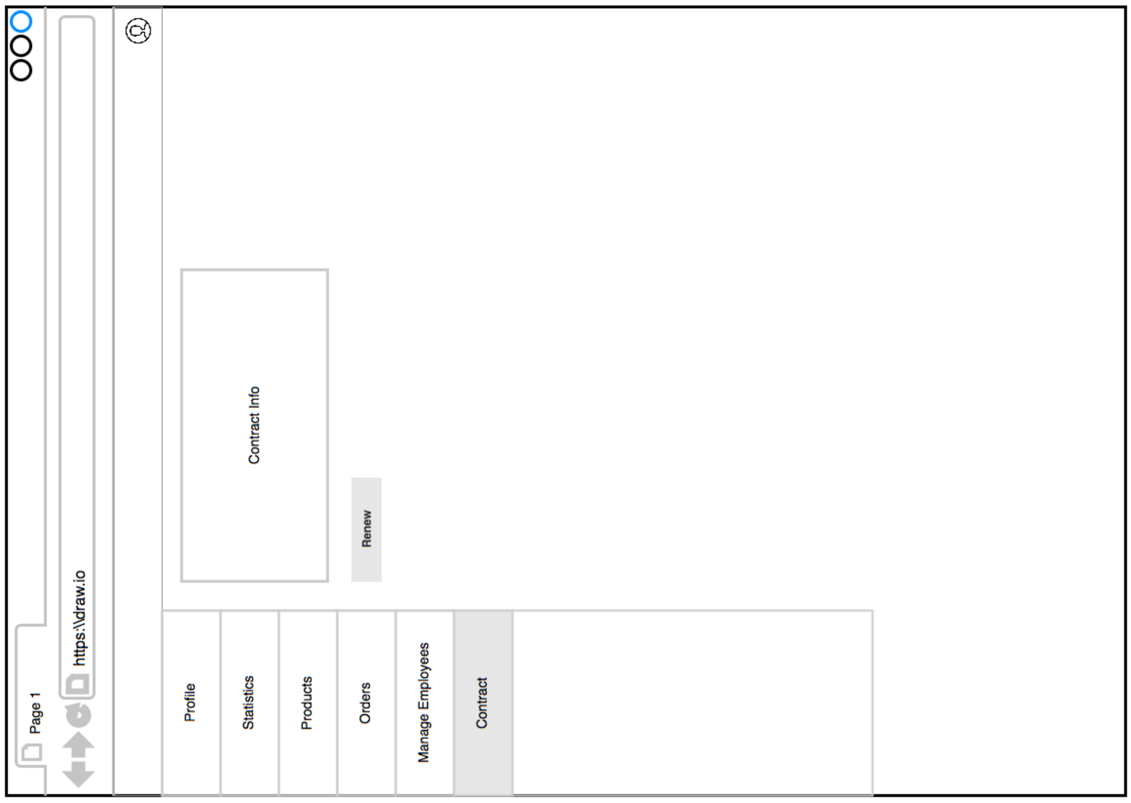


Figure 4.25: Bakery Contract

Chapter 5

Planning

This chapter describes how the planning for this project was made and the methodology used during its development. The planning was divided in three phases, the tasks made in the first semester, the tasks made in the second semester and the changes made, during the second semester, to the initial plann. It also describes the most important technological choices made for developing the platform, and which productivity tools were used.

5.1 Temporal Planning

The temporal planning for this project is described in this section and is divided into two phases, the first semester and the second semester. First the main tasks for each semester are presented and described, and then a Gantt diagram representing the schedule of the corresponding semester is presented.

5.1.1 First Semester

The first semester consisted mainly on planning and making decisions of how the project would be developed. The work for the semester was divided into five main tasks:

- Related Work
- Planning
- System Description
- Development
- Interim Delivery

For a better understanding of what these tasks are exactly, a brief description of each one is done below and image 5.1 shows the Gantt diagram of the first semester, representing the schedule for the mentioned tasks and their subtasks, showing the start and finish date expected for each task and subtask.

Related Work

This task refers to the study of similar web applications and platforms that are currently present in the market, the study of the web technologies used nowadays, and the study of usability techniques that are being used in web platforms.

Planning

This task consists on planning when and how the work would be done. The tasks for each semester were described, along with a Gantt diagram for each semester. It also consisted on defining the methodology that would be used in the project and choosing the web technologies to be used.

System Description

This task refers to the description of the system, including the description of the requirements in the form of user stories and use cases, the attribution of a priority on both functional and non-functional requirements, the description of the system architecture, the design and definition of the database and the representation of a navigation diagram and wireframes.

Development

This task, in this semester, involved the learning phase of Ruby on Rails, the basic setup and initial configuration of the web platform, the creation of the admin area, the development of both bakery and user login, the sign up of the user, and functionality of a bakery adding a product to their inventory.

Interim Delivery

This last task involved the write of the interim delivery of this thesis and the preparation for the interim presentation.

5.1.2 Second Semester

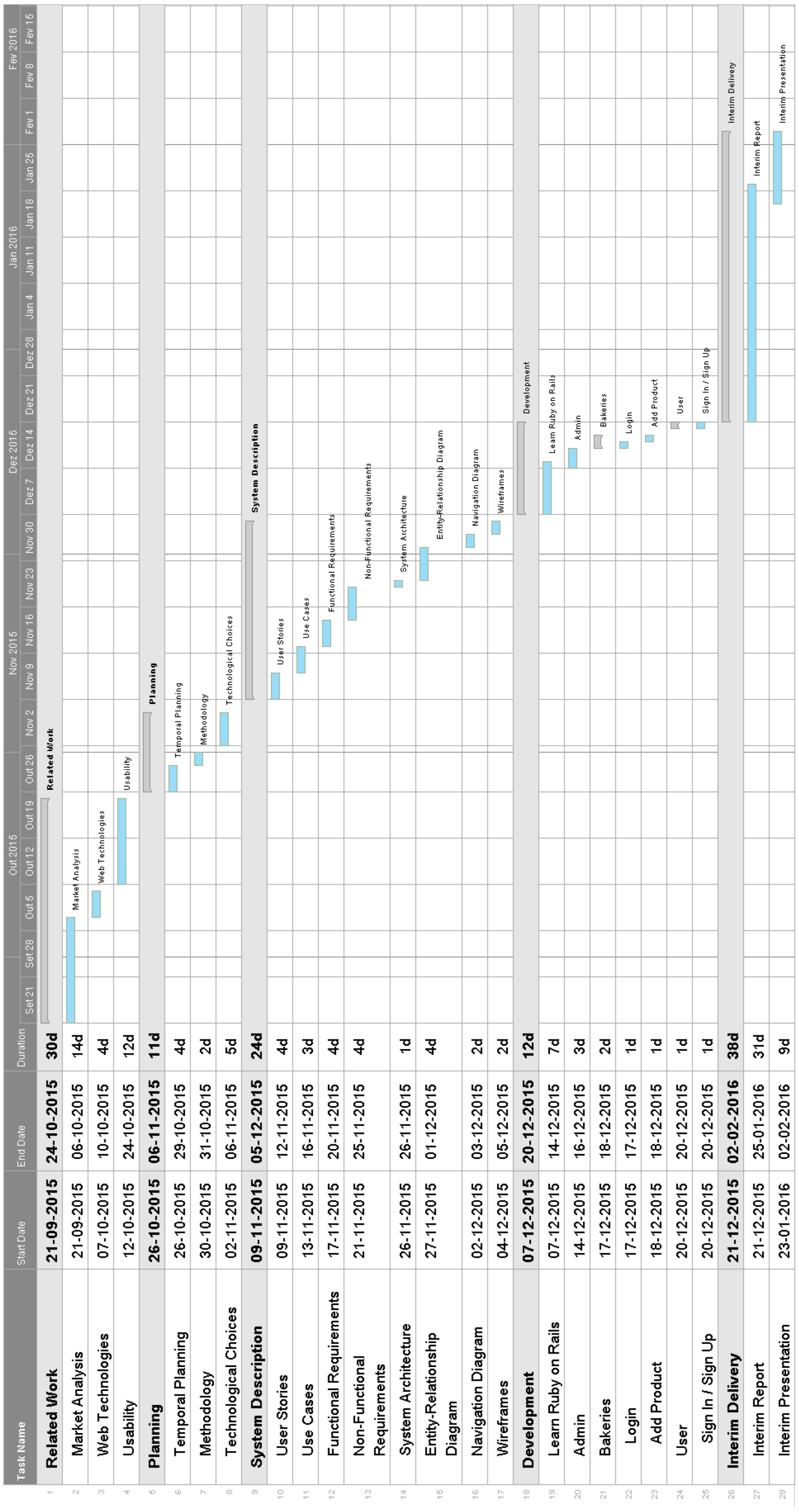
The second semester will consist mainly on developing and testing the platform. The work for the semester is divided into 3 main tasks:

- Development
- Tests and Improvements
- Final delivery

A brief description of what consists each one of these tasks can be seen below. Also a Gantt diagram on image 5.2, presents the schedule for the tasks and subtasks planned for the second semester, showing their duration, start date and finish date.

Development

This task consists on finishing the development already started on the first semester. It includes developing all the user functionalities such as, user profile and search, developing the rest of the bakeries functionalities such as, profile and sales statistics, and developing everything that involves the purchases and payments such as, orders and payments with PayPal. It also includes finishing the development of the admin area.



	Task Name	Start Date	End Date	Duration
1	**Related Work**	**21-09-2015**	**24-10-2015**	**30d**
2	Market Analysis	21-09-2015	06-10-2015	14d
3	Web Technologies	07-10-2015	10-10-2015	4d
4	Usability	12-10-2015	24-10-2015	12d
5	**Planning**	**26-10-2015**	**06-11-2015**	**11d**
6	Temporal Planning	26-10-2015	29-10-2015	4d
7	Methodology	30-10-2015	31-10-2015	2d
8	Technological Choices	02-11-2015	06-11-2015	5d
9	**System Description**	**09-11-2015**	**05-12-2015**	**24d**
10	User Stories	09-11-2015	12-11-2015	4d
11	Use Cases	13-11-2015	16-11-2015	3d
12	Functional Requirements	17-11-2015	20-11-2015	4d
13	Non-Functional Requirements	21-11-2015	25-11-2015	4d
14	System Architecture	26-11-2015	26-11-2015	1d
15	Entity-Relationship Diagram	27-11-2015	01-12-2015	4d
16	Navigation Diagram	02-12-2015	03-12-2015	2d
17	Wireframes	04-12-2015	05-12-2015	2d
18	**Development**	**07-12-2015**	**20-12-2015**	**12d**
19	Learn Ruby on Rails	07-12-2015	14-12-2015	7d
20	Admin	14-12-2015	16-12-2015	3d
21	Bakeries	17-12-2015	18-12-2015	2d
22	Login	17-12-2015	17-12-2015	1d
23	Add Product	18-12-2015	18-12-2015	1d
24	User	20-12-2015	20-12-2015	1d
25	Sign In / Sign Up	20-12-2015	20-12-2015	1d
26	**Interim Delivery**	**21-12-2015**	**02-02-2016**	**38d**
27	Interim Report	21-12-2015	25-01-2016	31d
28	Interim Presentation	23-01-2016	02-02-2016	9d

Figure 5.1: Gantt Diagram - First Semester

Tests and Improvements

This task includes all the tests and final improvements of the platform. The tests consist on testing all functionalities individually, testing the system in general by testing how we would react to certain events, and testing the usability by realizing different types of tests to the interface with people from a variety of ages and knowledge to see the amount of effort they needed to accomplish something. The improvements and testing will occur at the same time because during the testing phase there could be a need to correct some bugs and to improve the usability of the platform.

Final Delivery

This task involves the writing of the final version of this thesis and the preparation for the final presentation.

5.1.3 Changes To The Initial Plan

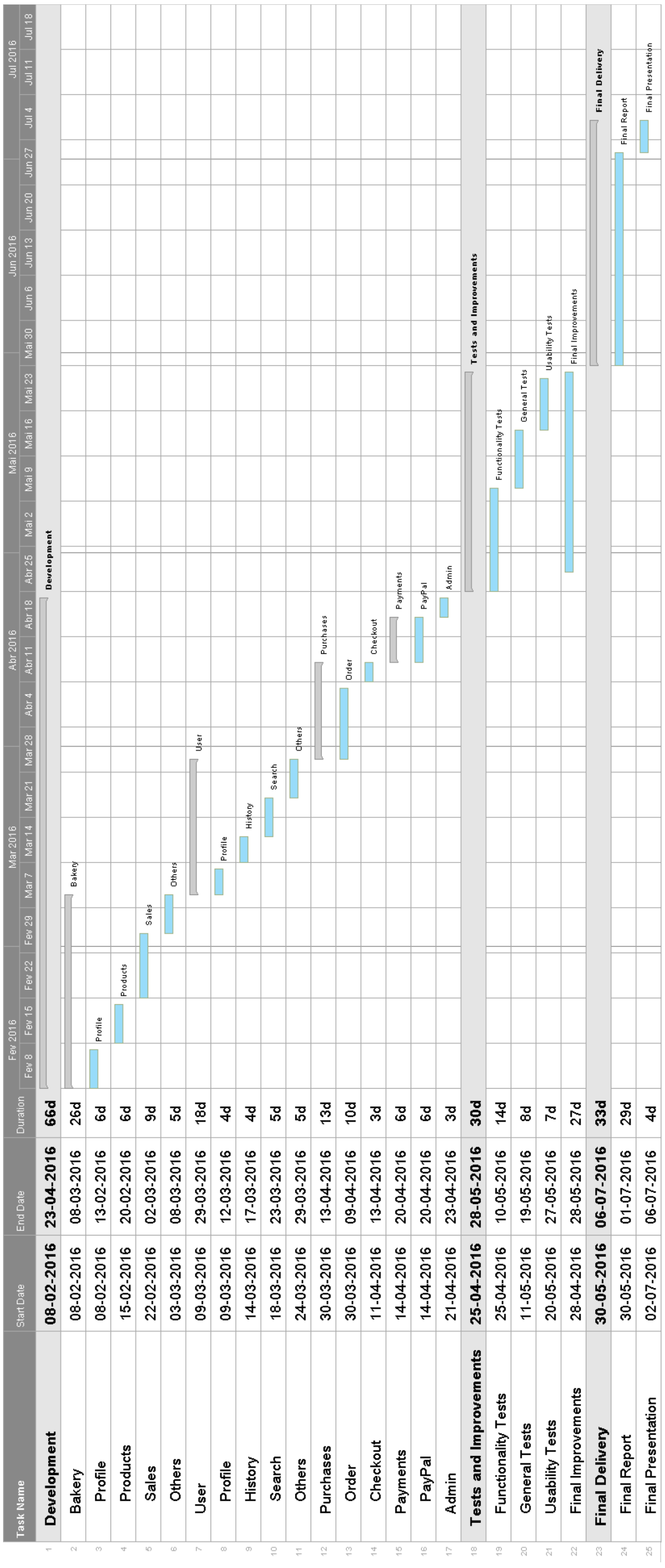
Since things do not always go according to plan, due to many different aspects that may interfere with the normal development of a project, it is necessary to analyze the plan initially made to see if it was followed to the letter and if everything went smoothly according to the plan. If this was not the case, then it is necessary to identify what caused this and think about what can we do in future projects to avoid this kind of problems.

After checking our initial plan for the second semester, we found that there was a delay on the development of the web platform. The development phase took longer to finish and instead of being completely done by April 23, like it was initially planned, it only ended on May 13 of 2016. This forced the author to have less time for the testing and improvements phase and to write this thesis. Therefore, the testing and improvements phase only started on 14 May instead of April 25 and was finished by Jun 11. The remaining time, until the delivery of this thesis, was spent writing this dissertation.

By analyzing and interpreting the process and the actual work that was done in the second semester, is possibly to identify two main problems that caused this delay.

The first problem was not having enough experience to do a project of this dimension, with so many requirements to be met, which caused the planned time for the implementation process to be shorter than expected. The reason why this happened was because the large number of functionalities that needed to be implemented were not properly taken in account and time needed for each one took longer than expected.

The second problem that caused a delay on this project, was due to the lack of work made by the Design and Multimedia student that was working on the front-end of this project. This caused delays on the development phase, because the author ended up taking on some functionalities and usability decisions that could otherwise be made together with this team mate. The author also spent some time implementing a simple design on the website so that it could be tested by the partner bakery which would have been avoided if the team mate had properly done his job. However, tests with the partner bakery still managed to be made, evaluating only functionalities and not usability / design. These tests resulted on the development of some additional features, such as the filter and the summary present on the Orders page which will be later explained on section 6.3.3.



	Task Name	Start Date	End Date	Duration
1	**Development**	**08-02-2016**	**23-04-2016**	**66d**
2	Bakery	08-02-2016	08-03-2016	26d
3	Profile	08-02-2016	13-02-2016	6d
4	Products	15-02-2016	20-02-2016	6d
5	Sales	22-02-2016	02-03-2016	9d
6	Others	03-03-2016	08-03-2016	5d
7	User	09-03-2016	29-03-2016	18d
8	Profile	09-03-2016	12-03-2016	4d
9	History	14-03-2016	17-03-2016	4d
10	Search	18-03-2016	23-03-2016	5d
11	Others	24-03-2016	29-03-2016	5d
12	Purchases	30-03-2016	13-04-2016	13d
13	Order	30-03-2016	09-04-2016	10d
14	Checkout	11-04-2016	13-04-2016	3d
15	Payments	14-04-2016	20-04-2016	6d
16	PayPal	14-04-2016	20-04-2016	6d
17	Admin	21-04-2016	23-04-2016	3d
18	**Tests and Improvements**	**25-04-2016**	**28-05-2016**	**30d**
19	Functionality Tests	25-04-2016	10-05-2016	14d
20	General Tests	11-05-2016	19-05-2016	8d
21	Usability Tests	20-05-2016	27-05-2016	7d
22	Final Improvements	28-04-2016	28-05-2016	27d
23	**Final Delivery**	**30-05-2016**	**06-07-2016**	**33d**
24	Final Report	30-05-2016	01-07-2016	29d
25	Final Presentation	02-07-2016	06-07-2016	4d

Figure 5.2: Gantt Diagram - Second Semester

5.2 Methodology

In every software development there is a need to choose a suitable Software Development Life Cycle (SDLC) methodology that defines how the developing process is managed and executed, by dividing it into few phases, such as planning, implementation or design. Nowadays, there are two major sets of methodologies, Traditional SDLC and Agile.[43]

Traditional SDLC refers to methodologies such as Waterfall and V-Model that follows four sequential phases. The first phase starts by defining the requirements and the second phase is when the system architecture is designed. After the two first phases are completed, the implementation phase starts. In it, all the implementation needed to construct the software is made. The last phase is the testing phase, where the software is tested for errors and bugs. The two last phases often are mixed together in order to deal with errors in an early stage of the implementation, as it is referred in the article [43].

The Agile development methods are based on iterative and incremental development and they focus on communication and interaction. This means that the process phases are executed multiple times, allowing a faster adaptation when requirements are changed. Agile has a couple of different methodologies, such as Scrum, Extreme Programming, Crystal Methods and more. [44]

The most popular methodologies for each Traditional SDLC and Agile are Waterfall and SCRUM, respectively. These two articles, [43] [45], do a comparison between these methodologies. They show that Waterfall can provide faster results on larger projects where the requirements are well defined and documented in the beginning while the Agile methods like SCRUM works better when the requirements are still being defined during the development and when customer interaction is bigger.

Waterfall is sequential, is based on phases, has a better documentation and is recommended to larger projects. Also, the requirements need to be set on the initial phase, and a contract or plan is required. It biggest disadvantage is not being able to deal well with changes. The phases of a Watefall model are well described and documented on this article [46].

Scrum is not sequential, works on sprints and is recommended to small and medium size projects. Despite the lack of documentation, it can deal with changes with ease, does tests on every iteration and has a higher customer involvement. Also, can make estimates easier to get and better.

Our Choice

Since this project had all requirements well documented and defined in its initial phase, and the author did not expect any major changes in the requirements during its development, the methodology used was Waterfall.

Waterfall model was chosen over an Agile method even though the project had a future client that gave us feedback on what he expected that the platform should provide, and that would test the application when a Minimum Viable Product (MVP) was completed.

This happened because the MVP was planned to only be finished by the end of the development of this project, and even then, we did not expected big changes on the requirements, only small ones that would not be relevant. The other reason for this choice was that, if we had opted out by and agile method, where the requirements are being established throughtout the development, there might not have been enough time to complete the project, due to the time that the author would have to spend on a back to back communication with the client. This way, by establishing all requirements in the beggining of the project, a lot of time was saved, time that came to be very valuable during the development phase.

Waterfall Model In This Project

The image 5.3 represents the Waterfall methodology used in this project.

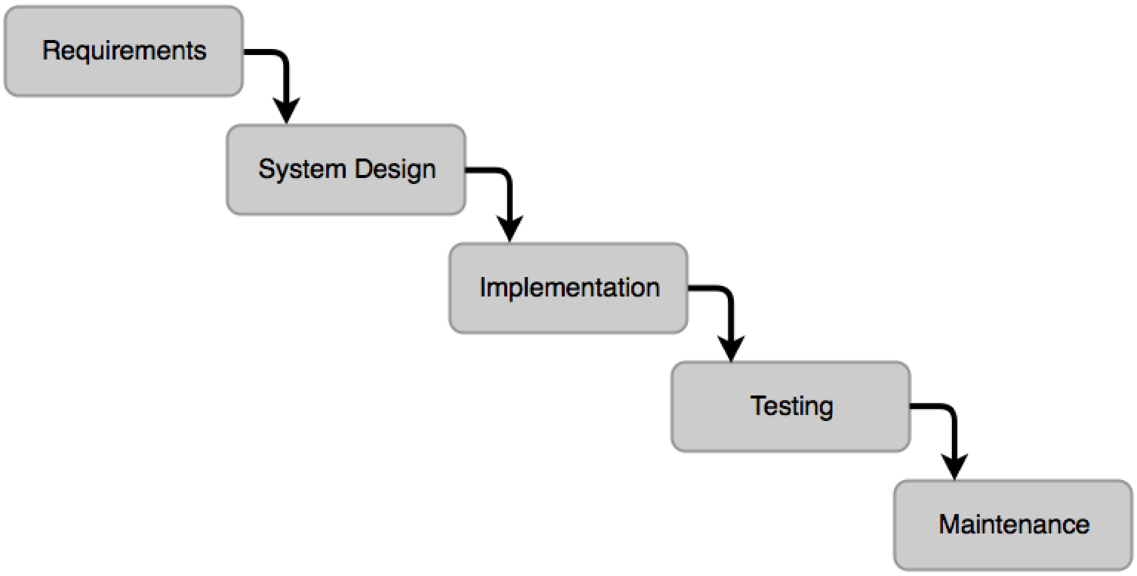


Figure 5.3: Waterfall Methodology

First, on the Requirements phase, all requirements, both functional and non-functional, needed to be well discussed and defined along side with the future client and the people responsible for the project at HYP. Then, on the System Design phase the system architecture and planning of the project would be defined, the database designed and the wireframes created. After these two initial phases, the Implementation phase starts, where the functionalities for the Bakery, User and Admin are implemented. When this last phase is completed, the Testing phase starts. In it, all tests are executed to ensure that the platform is validated and verified, and most important, ready to be used. For last, is the Maintenance phase where every error or bug is resolved and improvements are made.

Note that the Implementation and Testing phases often overlap in this project, since that when a functionality was implemented, some basic tests were executed to ensure that the functionality was working properly.

5.3 Technological Choices

This section does a comparison between the technologies that were used in this project and the other available and viable technologies described in chapter 2.

5.3.1 Back-End

In order to develop the back-end of the web platform, a web framework had to be chosen. This subsection compares the four web platform frameworks described in the section "Web Technologies" on chapter 2 and reveals the chosen framework for this project.

Comparison

In comparison, Node.js is the faster of the four but lacks in popularity and its community and documentation is still small, making its learning more difficult. Also, developing a web platform in Node.js takes more time than in the other three, since that Node.js does not have as many libraries as them.

These three articles [47] [48] [49] analyze and compare both Django and Ruby on Rails, and their conclusions are the same, both Django and Ruby on Rails are excellent choices to develop a web platform and their differences are not that big. Their learning time is identical, they are both popular, and while Ruby on Rails has a slightly better community and their web platform are easier to deploy, updates on Django are not as difficult as in Ruby on Rails.

Finally, when comparing Laravel with Django or Ruby on Rails, there is a slight difference between them. Laravel is not as secure, because of the security issues that PHP has and PHP loses in almost every category when compared to the other two frameworks, as can be seen in this article [49]. This causes Laravel to lose against Django and Ruby on Rails, even though Laravel also provides features to help improving the development time like the other frameworks.

The following table 5.1 does shows a comparison betweeen these frameworks in some key aspects.

	Node.js	**Laravel**	**Django**	**Ruby On Rails**
Language	Javascript	PHP	Python	Ruby
Architecture	Multiple	MVC	MVC	MVC
Developing Time	Medium	Small	Small	Small
Performance	Great	Good	Good	Good
Security	Good	Good	Great	Great
Popularity	Low	High	Medium	Medium
Release Year	2009	2011	2005	2005

Table 5.1: Back-end Frameworks Comparison

Our Choice

Our choice fell on Ruby on Rails because this was the only tecnical constraint of this project, since HYP does their projects with this framework. Even though this was a constraint, if we compare Ruby on Rails with the others frameworks is visible that Ruby on Rails is much faster than Node.js in terms of developing the web platform and learn the framework, it is overall better than Laravel, and compared to Django there was not a big difference between the two. With this we can verify that however Ruby on Rails was a constraint, it still is a great technologial choice.

5.3.2 Front-End

To develop the Front-End of the web platform, multiple programming languages were used. Since this is not the focus of this thesis and the author was not the responsible for this part in the web platform, this section is just a brief description of the languages that were used.

The Front-End was developed in the common languages HTML5, JavaScript and JQuery. As for the design and appearance of the web application, instead of using CSS, we choose to use Sass and Susy.

Sass [16]

Sass (Syntactically Awesome StyleSheets) is an extension of CSS for Ruby that allows the use of inline imports, variables, add nested rules, along with other things.

Susy [18]

Susy is a framework for Sass to help creating the layout of the website. The framework provides different grids that can be used together or separated to create the desired layout.

5.3.3 Database

This subsection compares the DBMS's described in the section "Web Technologies" on chapter 2 and reveals which database the author opted to store all the data of this project.

Comparison [50] [51]

In comparison SQLite is clearly the worst choice of the three databases for this project because is too simple and would not perform well, and in this case a client-server database is more appropriate.

Comparing MySQL and PostgreSQL, they are both safe and secure databases, they both support many features and run in plenty of different operating systems, and they both have a good support.

PostgreSQL was made to deal with large databases while the MySQL was originally written to connect low-level data structures as mentioned in the article [52] and has been evolving ever. Because of this, MySQL is faster and easy to configure than PostgreSQL, but PostgreSQL is more reliable and deals better with concurrency as mentioned in the article [50].

Our Choice

The author chose PostgreSQL since for this project it works perfectly and even though MySQL is a little faster, in this project, the difference of speed and performance would not be noticed. Also, the author already has experience working with PostgreSQL, which reduces the learning time.

5.3.4 Web Server

This subsection compares the two most popular web servers: Apache and Nginx, and explains why Nginx was chosen for the project. It also describes other tools used to help in the interaction with Ruby on Rails and with the deployment of the website.

Comparison

In comparison, Nginx is known to perform faster than Apache because it does not need to create new processes or threads to handle each request, making its memory and CPU usage much smaller.

An interesting article [53] tests the performance of the two web servers in three different scenarios: the first two focusing on response time and memory usage and the last scenario focused on the efficiency of both servers.

In the first scenario it is tested how the servers react to a continuous request for an HTML page. The results show that as of 5000 requests per second the memory usage of Apache starts increasing and the memory usage of Nginx stays almost the same. In terms of response time, both servers increase linearly until 5000 requests per second and after that Nginx increases two double the response time while Apache increases to thrice the response time.

The second scenario tests how the servers react to a continuous request from two clients for an image. In this case, after 7000 requests per second the memory usage of Apache increases up to 40% more compared to Nginx, and the response time of Apache rises exponentially where as Nginx increases is response time up to twice.

The last scenario tests the efficiency of both servers by increasing the request rate for an image. The test reveals that for a small number of requests the error rate is almost nil, but when the request rate is increased, the Nginx suffers a greater increase on the error rate compared to Apache.

In conclusion, both servers seem to perform well when dealing with less requests per second but when the number of requests rises, Nginx presents a better response time and memory usage due to is architecture, but shows a bigger error rate. This proves that Nginx is better than Apache in terms of scalability and responsiveness but has less efficiency.

Our Choice

Nginx was chosen since it uses less memory, is faster than Apache and performs well when there is a large amount of request being made. Even though Nginx has less efficiency than Apache, in this project it is not expected a really high-traffic so that it can make a difference. Also, Nginx is easy to configure and simple to use.

Other Technologies

To help in the deployment of the website in the server, two more technologies were used. This technologies were: Unicorn and Capistrano.

Unicorn is, as cited in their website [54], an "HTTP server for Rack applications designed to only serve fast clients on low-latency" and was used because Nginx can not serve Ruby web apps out-of-the-box. Basically, Unicorn helps connect the Rails rack application and Nginx, with Nginx being the fast client. Also, Unicorn has some helpful features for Rails servers.

Capistrano is a Ruby remote server automation and deployment tool as cited in their website [55] designed to help with the deployment of web applications in the servers. This technology allows you to deploy the web application directly from the GitHub repository to the server via SSH and has many useful functions.

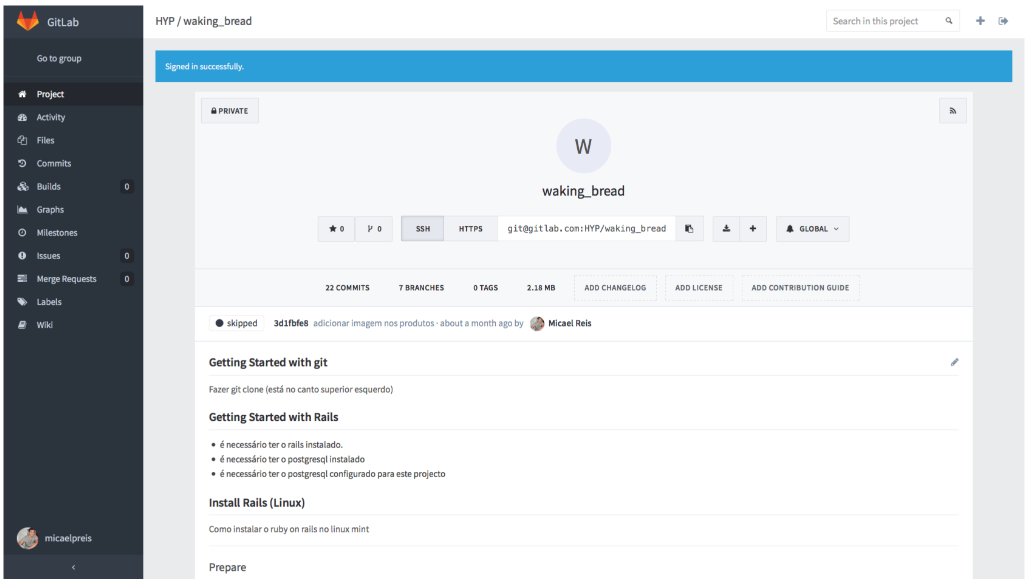
5.4 Productivity Tools

This section describes the productivity tools that are being used during the development of this project. This tools help with the planning, organization, implementation of the web platform and writing this thesis.

5.4.1 Git

This first tool is being used as a version control system of the web platform, Git. This systems allows the control of different versions of the application during its development.

Since Git requires a repository where the system is hosted, the author choose GitLab, the Git repository used by HYP. This repository has all of the code versions developed for the web platform stored and can be seen in the following figure 5.4.



Signed in successfully.

`git@gitlab.com:HYP/waking_bread`

skipped 3d1fbfe8 adicionar imagem nos produtos · about a month ago by Micael Reis

Getting Started with git

Fazer git clone (está no canto superior esquerdo)

Getting Started with Rails

- é necessário ter o rails instalado.
- é necessário ter o postgresql instalado
- é necessário ter o postgresql configurado para este projecto

Install Rails (Linux)

Como instalar o ruby on rails no linux mint

Prepare

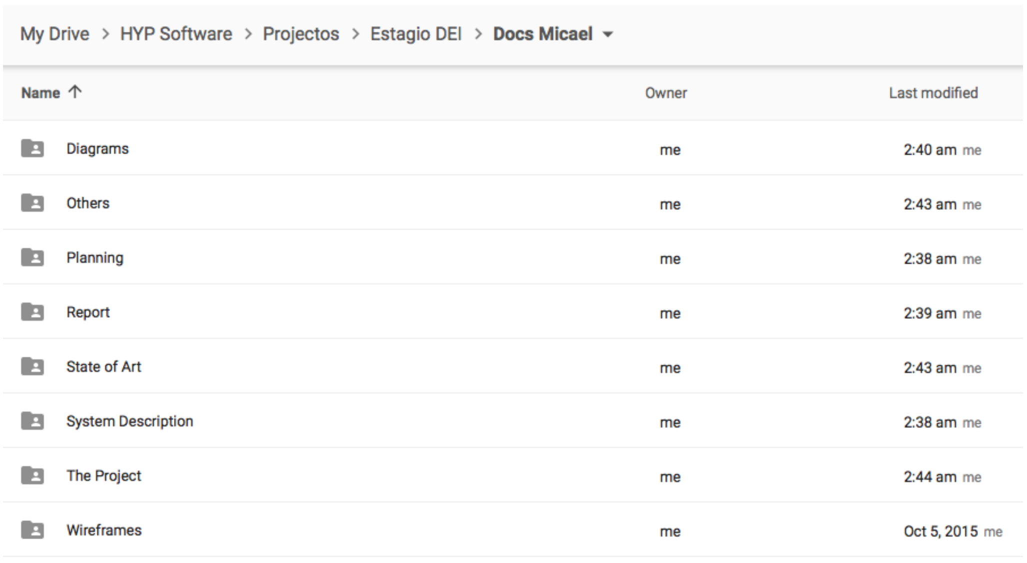
Figure 5.4: Git Repository

5.4.2 Google Drive

Google Drive is the service used to store and synchronize the files of this thesis. It is where the initial versions of the report were created such as the files with the features originally thought for the web platform, and all the diagrams and wireframes are saved there. In figure 5.5 the main folder of the project is represented along with all subfolders.

5.4.3 Trello

This productivity tool works as a tasks manager for the whole project. Trello organizes the projects into boards, and inside them there are a set of lists with multiple cards (tasks).

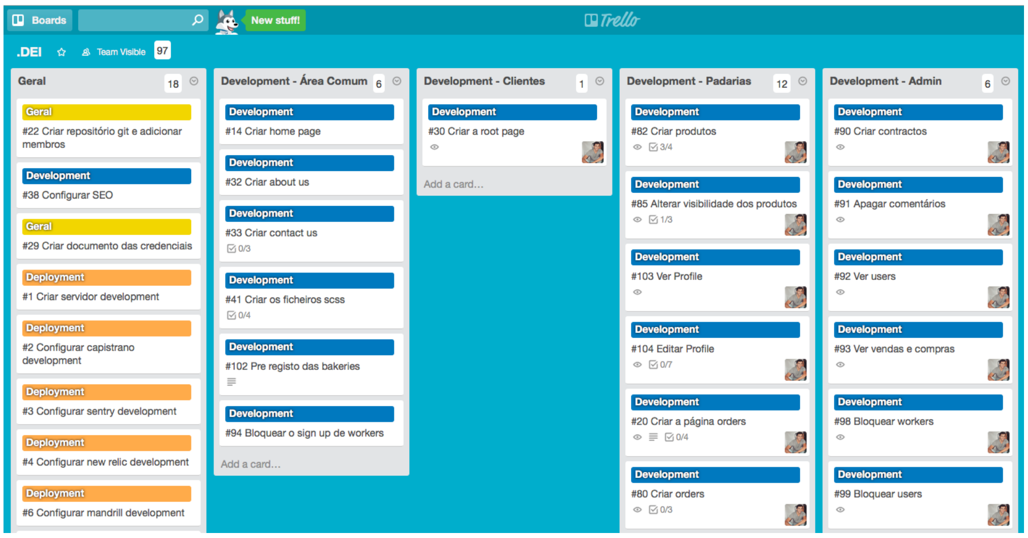


My Drive > HYP Software > Projectos > Estagio DEI > **Docs Micael**

Name ↑	Owner	Last modified
Diagrams	me	2:40 am me
Others	me	2:43 am me
Planning	me	2:38 am me
Report	me	2:39 am me
State of Art	me	2:43 am me
System Description	me	2:38 am me
The Project	me	2:44 am me
Wireframes	me	Oct 5, 2015 me

Figure 5.5: Google Drive

This cards can then be assigned to the people that are on the project and they can have a checklist, a due date, comments and even attachments. Usually there is a list called "Done" where the cards that have been completed care moved to. In the figure 5.6 the board created for this project can be seen.



Geral 18
- Geral — #22 Criar repositório git e adicionar membros
- Development — #38 Configurar SEO
- Geral — #29 Criar documento das credenciais
- Deployment — #1 Criar servidor development
- Deployment — #2 Configurar capistrano development
- Deployment — #3 Configurar sentry development
- Deployment — #4 Configurar new relic development
- Deployment — #6 Configurar mandrill development

Development - Área Comum 6
- Development — #14 Criar home page
- Development — #32 Criar about us
- Development — #33 Criar contact us — 0/3
- Development — #41 Criar os ficheiros scss — 0/4
- Development — #102 Pre registo das bakeries
- Development — #94 Bloquear o sign up de workers

Add a card...

Development - Clientes 1
- Development — #30 Criar a root page

Add a card...

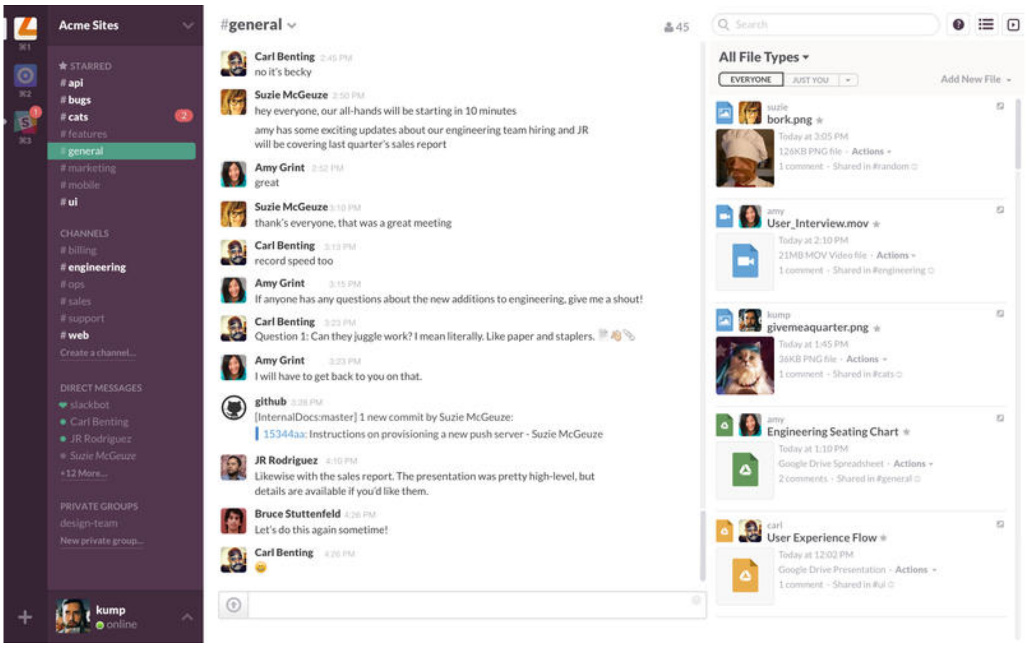
Development - Padarias 12
- Development — #82 Criar produtos — 3/4
- Development — #85 Alterar visibilidade dos produtos — 1/3
- Development — #103 Ver Profile
- Development — #104 Editar Profile — 0/7
- Development — #20 Criar a página orders — 0/4
- Development — #80 Criar orders — 0/3

Development - Admin 6
- Development — #90 Criar contractos
- Development — #91 Apagar comentários
- Development — #92 Ver users
- Development — #93 Ver vendas e compras
- Development — #98 Bloquear workers
- Development — #99 Bloquear users

Figure 5.6: Trello

5.4.4 Slack

Since this is a collaborative project, it is necessary to use some communication tool to communicate within the team and with the mentors of this thesis. The chosen tool is Slack since HYP uses this tool and the author already as experience with it. Slack allows the creation of private group conversations, to send direct messages to team members, import attachments and more. Figure 5.7 shows the interface of a typical project channel.



Carl Benting 2:45 PM
no it's becky

Suzie McGeuze 2:50 PM
hey everyone, our all-hands will be starting in 10 minutes
amy has some exciting updates about our engineering team hiring and JR will be covering last quarter's sales report

Amy Grint 2:52 PM
great

Suzie McGeuze 3:10 PM
thank's everyone, that was a great meeting

Carl Benting 3:13 PM
record speed too

Amy Grint 3:15 PM
If anyone has any questions about the new additions to engineering, give me a shout!

Carl Benting 3:23 PM
Question 1: Can they juggle work? I mean literally. Like paper and staplers.

Amy Grint 3:23 PM
I will have to get back to you on that.

github 3:28 PM
[InternalDocs:master] 1 new commit by Suzie McGeuze:
15344aa: Instructions on provisioning a new push server - Suzie McGeuze

JR Rodriguez 4:10 PM
Likewise with the sales report. The presentation was pretty high-level, but details are available if you'd like them.

Bruce Stuttenfeld 4:26 PM
Let's do this again sometime!

Carl Benting 4:26 PM
😀

Figure 5.7: Slack

5.4.5 Draw.io

Draw.io is a diagram editor that allows the creation of multiple types of diagrams. This tool was integrated with Google Drive, meaning that every diagram created is saved in the Google Drive project folder. On it, all the diagrams and wireframes used in this thesis are created. Figure 5.8 represents the interface of this productivity tool.

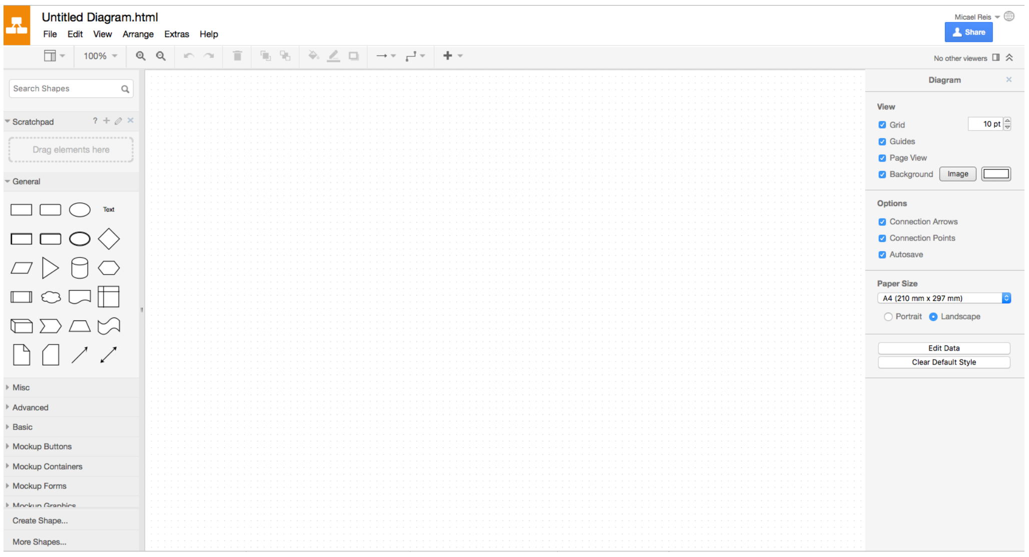
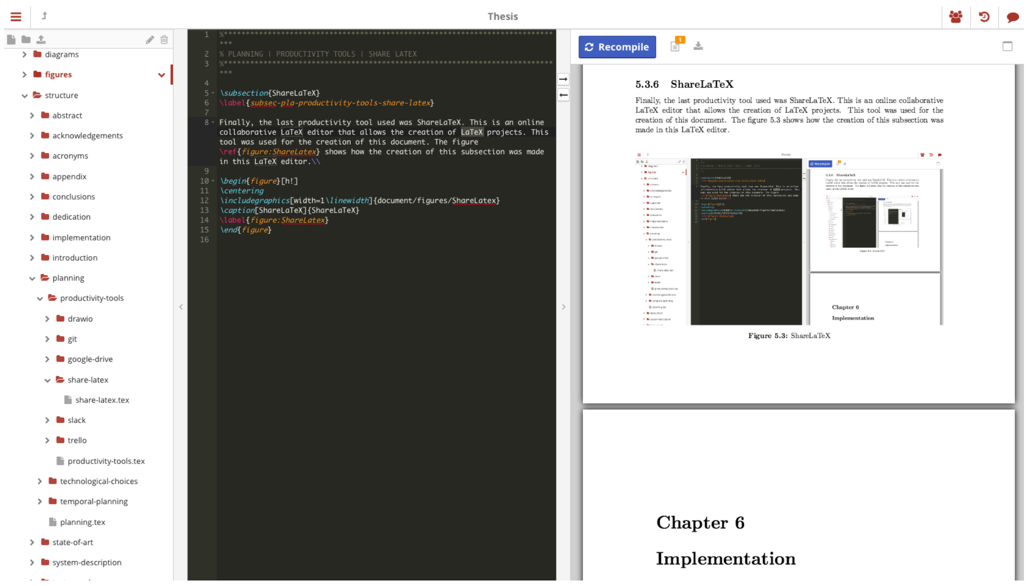


Figure 5.8: Draw.io

5.4.6 ShareLaTeX

Finally, the last productivity tool used is ShareLaTeX. This is an online collaborative LaTeX editor that allows the creation of LaTeX projects. This tool is being used for the creation of this document. The figure 5.9 shows how the creation of this subsection is made in this LaTeX editor.



```
%*****************************************************************************
***
% PLANNING | PRODUCTIVITY TOOLS | SHARE LATEX
%*****************************************************************************
***

\subsection{ShareLaTeX}
\label{subsec-pla-productivity-tools-share-latex}

Finally, the last productivity tool used was ShareLaTeX. This is an online
collaborative LaTeX editor that allows the creation of LaTeX projects. This
tool was used for the creation of this document. The figure
\ref{figure:ShareLatex} shows how the creation of this subsection was made
in this LaTeX editor.\\

\begin{figure}[h!]
\centering
\includegraphics[width=1\linewidth]{document/figures/ShareLatex}
\caption[ShareLaTeX]{ShareLaTeX}
\label{figure:ShareLatex}
\end{figure}
```

5.3.6 ShareLaTeX

Finally, the last productivity tool used was ShareLaTeX. This is an online collaborative LaTeX editor that allows the creation of LaTeX projects. This tool was used for the creation of this document. The figure 5.3 shows how the creation of this subsection was made in this LaTeX editor.

Figure 5.3: ShareLaTeX

Chapter 6

Implementation

Figure 5.9: ShareLaTeX

Chapter 6

Implementation

In this chapter everything related to the implementation of the web platform is described and explained. First a descripton of how the implementation process was organized and executed is presented. Then, all functionalities implemented are described along with an explanation of how they were implemented. This part was divided into three diferent sections for a better visualization and understanding: user, bakery and admin. At last, other key features implemented that are used in more than one subsection mentioned above, are presented and explained.

6.1 Process

During the implementation of this web platform, there was a simple process that was followed to help improve the efficiency of the development and to help keep everything organized and up to the schedule.

Before the process could start, all functionalities that needed to be implemented were set in Trello cards properly organized, see section 5.4.3. These cards would later be updated as they were being used and completed. These functionalities were established with the help of the functional requirements defined on section 4.3. Also, for the main tasks a date limit was established with the objective of ensure that the work was being done and that the deadlines set on the planning phase were met.

After this initial action, the process created was ready to be followed. This process consisted on eight actions that were being executed over and over again until the implementation was completed. The process was the following:

1. Choose the task to be implemented;
2. Set the label "Working On" on the task;
3. Create a checklist inside the task with every thing that needed to be done, in order for it to be completed;
4. Start implementententing the task;
5. Check the elements of the checklist as they were being met;
6. Finish implementententing the task;
7. Remove the label "Working On" of the task;

8. Move the task to the "Done" list.

Action 1, choose the task to be implemented, followed a criteria of developing the main tasks and their dependencies first, and only after that, the less relevant tasks were chosen. To identify which tasks were main tasks, the author used the use cases defined on section 4.2.

Action 2 and 7 had the objective of letting the people in charge of the project and its other collaborator know what the author was doing at all times, since the label allowed the task being implemented be easily identified.

Action 3 and 5 were used only on some bigger tasks that had a lot of sub-tasks to be implemented or done. The checklist helped the author so taht nothing was forgotten.

The action 4 and 6 are pretty much self explanatory, they refer to the beginning and end of a task implementation.

Finally the action 8 is the last thing happening in this process. It happens when the task is completely done and is moved to the list "Done" on Trello. This is the list that contains all tasks previously completed.

In conclusion, this process allowed the author to keep everything well organized and to keep track of everything that was being done, since it is possible to check when each task was completed, when each element of the checklist was checked and what task was being implemented at the time.

6.2 User

This section has as an objective, to explain and describe the functionalities developed for the User. It starts by listing the pages that the user has access to. Then, the processes of searching for bakeries or products and make and order are explained in the subsections 6.2.1 and 6.2.2, respectively. In the end, the remaining features developed are briefly described in the section 6.2.3.

In total there are 13 pages destined only to the user, not counting with the login and registration pages. Some of these pages can be accessed only when the user is logged in the platform and others can be accessed at all times. The pages are:

- Homepage;
- Profile;
- Edit profile;
- Purchase history;
- Search bakery;
- Search product;
- Bakery page;
- Product page;

- Shopping bag;
- Checkout;
- Purchase completed;
- About;
- Client Area.

The pages related with the login and registration are described on the subsection 6.5.1.

Before proceeding to the following subsections, is worth noting that the images of the platform that are shown, do not have the final design and layout implemented. The design and layout present in the pages is the one made by the collaborator of this project, and since he did not made a good job, his work will not be the final one, and the design and layout of the platform will be completely reformulated.

6.2.1 Search For Bakery / Product

One of the most important things in this platform, was the ability to make smart and advanced searches, since it is very important that the user can find what they are looking for, without the need to spend a long time looking for it. With this in mind, the search process developed tries to be simple while still being affective and efficient.

There were two different searches developed in the platform, one to search for bakeries and another to search for products. This distinction happened due to the fact that the user might want to search for a specific bakery or for a specific product and having these two searches they can find what they desire more quickly. Also worth nothing that both searches are always available to the user, whether they are logged in or not.

Search for Bakery

The process of searching for a bakery is a easy and simple one because is was only two fields, the normal text input field and the address field. Note that these two fields can be used together or separately.

The text field is used to search for a bakery by its name. The system looks for bakeries that have the string searched in any part of its name. Meaning, that the string searched does not need to be found separately as a single word to return results, it can be inside a word. Also, to help improving this search, capitalize letters are ignored.

Next, a example of a simple search is shown to help understanding what was now explained.

Search: Bake

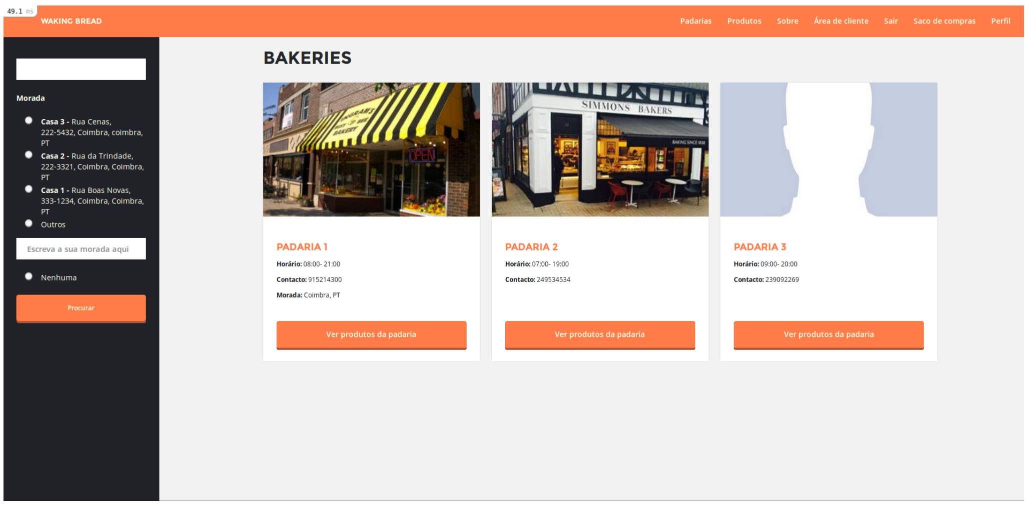
Possible results: Bakery / bake / Bake / bakeries / BAKERY

Unlike the text field, the address field changes depending on whether the user is logged in or not. If the user is not logged in to the platform, the address field is a simple text input where the user can type an address. In case the user is logged in, the address field is a group of radio buttons composed of the user addresses, a text input for the user to insert other address and also a option for none. The user can then choose one of their addresses, input a new one or choose none.

The purpose of this address field is so that the user can search only for bakeries that do deliveries in the address chosen, preventing bakeries, that do not do deliveries in their location, from showing up in the search.

At last, when the user does a search, the bakeries that match the search criteria are presented to the user. In them, the user can see some information about the bakery and a button that redirects the user to a page with all products from that bakery. Also, when clicking in a bakery, the user will be redirected to the bakery page.

The image 6.1 shows the search bakery page when the user is not logged in.



Casa 3 - Rua Cenas, 222-5432, Coimbra, coimbra, PT
Casa 2 - Rua da Trindade, 222-3321, Coimbra, Coimbra, PT
Casa 1 - Rua Boas Novas, 333-1234, Coimbra, Coimbra, PT
Outros
Nenhuma

PADARIA 1
Horário: 08:00- 21:00
Contacto: 915214300
Morada: Coimbra, PT

PADARIA 2
Horário: 07:00- 19:00
Contacto: 249534534

PADARIA 3
Horário: 09:00- 20:00
Contacto: 239092269

Figure 6.1: User Area - Search for Bakery

Search for Product

Searching for a product has a little but more complexity than the search for bakeries has. Besides the fields of the search above, that work exactly the same way, this search adds three more fields to it. The fields are: categories, sub-categories and promotion.

These fields are added to this search since it is expected for the bakeries to have a lot of different products from different categories, sub-categories as well as having products in promotion, and these fields provide a good way to filter the results returned. Now that we know why were these fields added to this search, it is time to explain how they work.

The categories field consists on a group of select boxes for every category of the products. The user can choose to not select any category, select some of them or even select all of them. If the user were to select none, the search would return products of any categories, if the user selected some of the categories, the search would return products of only those categories and in the last case, where the user selects all categories, the result is the same as not select any.

At first sight the field of sub-categories seams to work the same way as the categories field, since it consists on a group of select boxes for every sub-category of the products. However, that is not the case due to the fact that a product can belong to multiple sub-categories, but it can only belong to one category. So, in this case, if the user were to select two sub-categories, the result would return products that belong to at least those two sub-categories at the same time. In the case of the user selecting every sub-category, the search would look for products that belong to every sub-category.

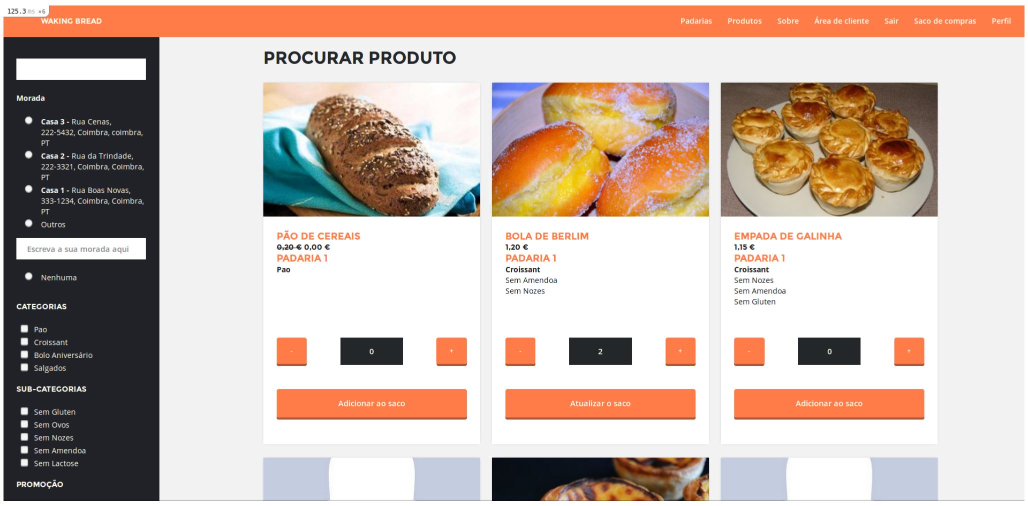
The last field is a single select box that when selected makes the search return only products that are in promotion, and ignore all the other products.

In short, these fields when used together, allow the user to search for products that have a specific name, that can be delivered in their location, that belong to a certain category and sub-categories and that are in promotion. These are all very valuable filters that can help the user find what they want quickly and effortless.

After doing a search, it is visible in the resulting products, their name, photo and price, the bakery and category that they belong, and their sub-categories. In case the user is logged in to the platform, a form to add the product to the bag is also shown on each product.

To help with the usability of the platform, the bakery name is a link that redirects the user to that bakery page, the category name is a link that does a search for products of that category and the sub-categories, such as the category, are links that do a search for producs of the sub-category clicked. Also, when clicking on the product the user is redirected to the product page.

The image 6.2 presents the search product page when the user is logged in.



Casa 3 - Rua Cenas, 222-5432, Coimbra, coimbra, PT

Casa 2 - Rua da Trindade, 222-3321, Coimbra, Coimbra, PT

Casa 1 - Rua Boas Novas, 333-1234, Coimbra, Coimbra, PT

PÃO DE CEREAIS
~~0,20 €~~ 0,00 €
PADARIA 1
Pao
0

BOLA DE BERLIM
1,20 €
PADARIA 1
Croissant
Sem Amendoa
Sem Nozes
2

EMPADA DE GALINHA
1,15 €
PADARIA 1
Croissant
Sem Nozes
Sem Amendoa
Sem Gluten
0

Figure 6.2: User Area - Search for Product

6.2.2 Make An Order

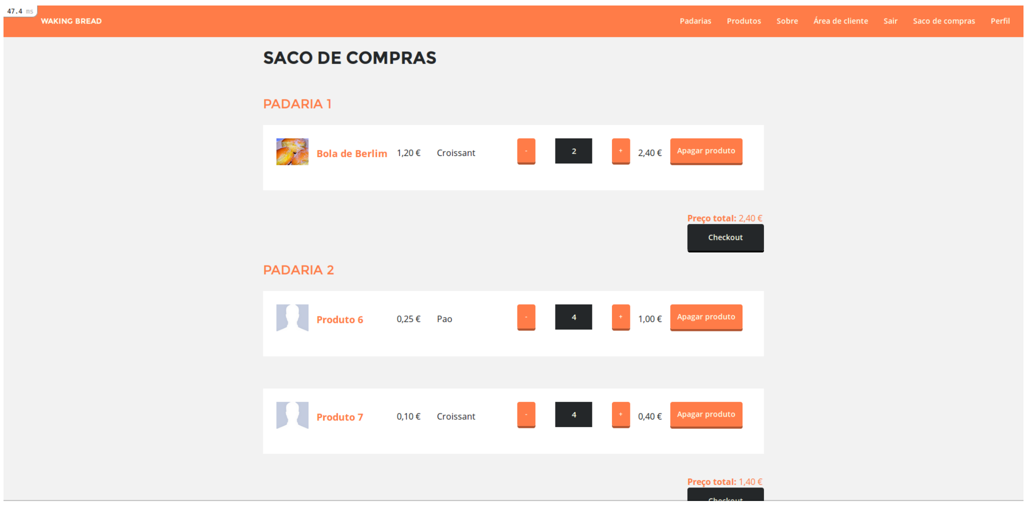
The process of making and order is the most important one in the platform. It is the main goal, and the main action that the user can do. Therefore, it was important to make it as easy and efficient as possible, and not deviate much from the process that is used on most platforms that require this type action since it is already proven to be effective.

The process created to make an order followed these 5 steps:

1. Add products to the bag;
2. Check the products in the bag;
3. Go to checkout;
4. Choose delivery options on checkout;
5. Pay

In the first step, the user needs to add products to their virtual bag. For that, the platform offers two ways for them to do that. Either they add the products in the search product page or in the product page. To add the products the user needs to select the quantity and then click the button to add the product to the bag, or in case the product is already in it, to update the quantity of the product in the bag.

Then, when they finish adding the products, they need to go to the shopping bag. To do this, they need to click on the menu button to go to the shopping bag. When in there, they can check the products that are in the bag, change their quantity, delete the products from the bag or if everything is the way they want, go to checkout. Since they can not do a order with products from different bakeries, the products are arranged by bakery. On the image 6.3 a shopping bag page can be seen.



SACO DE COMPRAS

PADARIA 1

Bola de Berlim	1,20 €	Croissant	- 2 +	2,40 €	Apagar produto

Preço total: 2,40 €

Checkout

PADARIA 2

Produto 6	0,25 €	Pao	- 4 +	1,00 €	Apagar produto
Produto 7	0,10 €	Croissant	- 4 +	0,40 €	Apagar produto

Preço total: 1,40 €

Figure 6.3: User Area - Shopping Bag

The third step just requires that the user, when in the shopping bag, clicks the button to go to checkout. It needs to be noted that there may be more than one checkout button, and when this happens, the user needs to click on the button that corresponds to the bakery that we wants.

The fourth step of this process is the biggest and probably the most important one. In it, the user needs to choose everything about the delivery of the order that they are about to make. First they need to check the products that they are about to order and the total price of the order, so that they do not make any mistake, then they need to chose the date or dates for the delivery, next they can choose the address where they want the delivery to be made as well as the delivery hour, and finally, if they thinks it is necessary, they can write an observation about the order. The image 6.4 presents the checkout page.

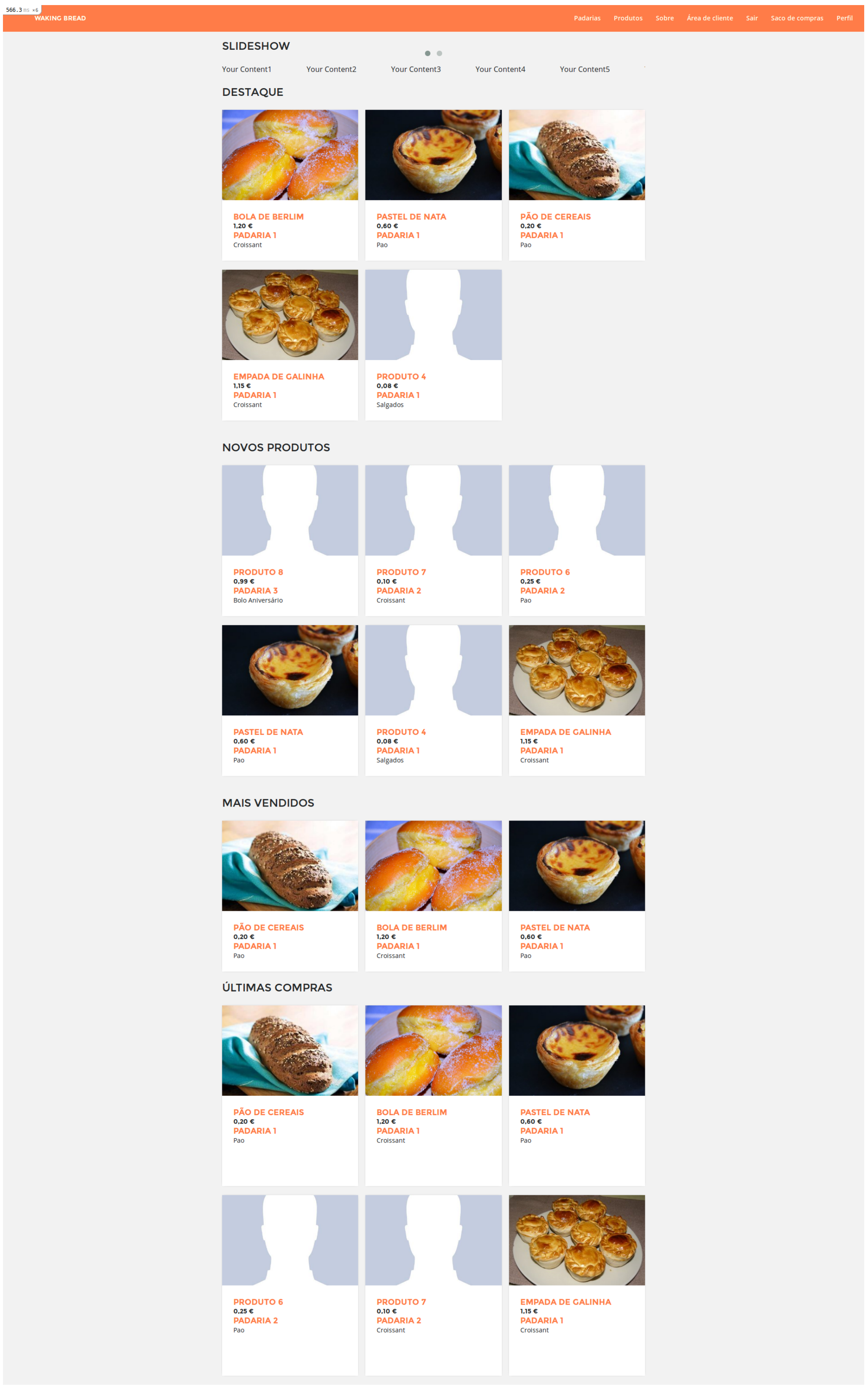

WAKING BREAD
Padarias
Produtos
Sobre
Área de cliente
Sair
Saco de compras
Perfil
SLIDESHOW
Your Content1
Your Content2
Your Content3
Your Content4
Your Content5
DESTAQUE
BOLA DE BERLIM 1,20 € PADARIA 1 Croissant
PASTEL DE NATA 0,60 € PADARIA 1 Pao
PÃO DE CEREAIS 0,20 € PADARIA 1 Pao
EMPADA DE GALINHA 1,15 € PADARIA 1 Croissant
PRODUTO 4 0,08 € PADARIA 1 Salgados
NOVOS PRODUTOS
PRODUTO 8 0,99 € PADARIA 3 Bolo Aniversário
PRODUTO 7 0,10 € PADARIA 2 Croissant
PRODUTO 6 0,25 € PADARIA 2 Pao
PASTEL DE NATA 0,60 € PADARIA 1 Pao
PRODUTO 4 0,08 € PADARIA 1 Salgados
EMPADA DE GALINHA 1,15 € PADARIA 1 Croissant
MAIS VENDIDOS
PÃO DE CEREAIS 0,20 € PADARIA 1 Pao
BOLA DE BERLIM 1,20 € PADARIA 1 Croissant
PASTEL DE NATA 0,60 € PADARIA 1 Pao
ÚLTIMAS COMPRAS
PÃO DE CEREAIS 0,20 € PADARIA 1 Pao
BOLA DE BERLIM 1,20 € PADARIA 1 Croissant
PASTEL DE NATA 0,60 € PADARIA 1 Pao
PRODUTO 6 0,25 € PADARIA 2 Pao
PRODUTO 7 0,10 € PADARIA 2 Croissant
EMPADA DE GALINHA 1,15 € PADARIA 1 Croissant

Figure 6.4: User Area - Checkout

After this last step the user can proceed to pay the order. Currently the platform only supports one payment method, PayPal. Therefore, when the user clicks on the button to pay, they will be redirected to PayPal to complete the payment. When the payment is finished, the user is redirected to the page Purchase Completed, and this whole process is completed.

Before finishing this subsection is important to note two things. First, in the future there will be more payment methods, wherefore when clicking on pay, the user will be redirected to a page to select the payment method instead of being directly redirected to PayPal. Second, the receipts and bills are delivered together with the order.

6.2.3 Other Features

In order to provide the user with the best experience possible, a lot of useful features were implemented beyond those already spoken in the last two subsections. To better explain and describe these features implemented, this subsection will be divided by the user pages.

Homepage

The platform has two different homepages, one for when the user is logged in an other for when they are not.

The homepage when the user is not logged is the first thing that a user will see when visiting the platform for the first time. Therefore, it was crucial that the homepage provides key information about the platform, explaining clearly what it is and what it does.

While doing the initial planning phase of this project we reached the conclusion that it would be important for the user to check if there are bakeries that do deliveries to their locations without the need for them to register on the platform. So, this feature was implemented on the homepage with the help of the Google Maps Javascript API, allowing the user to check for bakeries in their location. The use of this API is explained on the section 6.5.

Also, in this homepage a pre-registration form is present so that the bakeries that desire to join the platform can get in touch with the admins of the platform and then proceed to the contract negotiations.

As for the homepage for when the user is logged in. It can be seen on image 6.5.

In this homepage, the need to show the type of information present in the other homepage is nil. Instead, there are other type of information that should be present to the user, such as the last purchases, new products and more.

Therefore, we chose to split it into five sections:

- Slideshow;
- Featured products;
- New products;
- Best sellers;
- Last purchases;

The slideshow is where bakeries will be featured. It is a simple slideshow that will show special photos of bakeries, that can do deliveries to a user address, with the objective of promoting them. Note that not every bakery has this opportunity, only the ones that included that in their contract.

The featured products is a grid that shows a certain number of products, such as the next two sections, that are featured, and that belong to a bakery that can make deliveries in at least one user address. Like the slideshow the bakery can only have featured products if it is included in the contract.

The new products section shows new products from bakeries that can deliver in a user address so that the user can know when a product is created.

The best sellers shows the most sold products from the past fifteen days. Thus, the user learns about the products that are more popular to the other users. Such as the previous three sections, in this one, the products shown are only from bakeries that do deliveries in a user address.

The last section shows the last products purchased by the user. This way, if they want to order the products again, they can easily find them right on the homepage.

Noteworthy that the products showed in this page, in addiction to the fact that when clicked the user is redirected to the product page, also have links for their bakery and category. Meaning that when the user clicks on the bakery name, they are redirected to the the bakery page and when they click on the category of the product, the user is redirected to the search product page listing only products of that category.

Profile

When the user is logged in, they have the possibility to check their profile. In that page they will find all their personal information, from the email or name to their addresses. Additionally, from there they have access to edit their profile and check their purchase history.

Edit profile

On edit profile, the user is allowed to change all their personal information, from the email or avatar to the their contact. Note that in order to make changes, the user needs to provide their current password, otherwise this changes will not be made.

To edit their addresses, the user is provided with a form for each address, and with other form to add a new one.

Since it is important to the bakery to not only have the address but also have the coordinates of the location, besides the forms mentioned before, there are also two maps in this page.

The first map has the location of the user addresses marked in it. The user can then hide or show each marker for a better visualization, and move the markers around inside the map. This last feature allows the user to change the coordinates of an address, by moving the marker, because sometimes the Google Maps API can not determine exactly where the exact location of an address is. Also, the markers have an info window, containing the address, that appears when the user clicks on them.

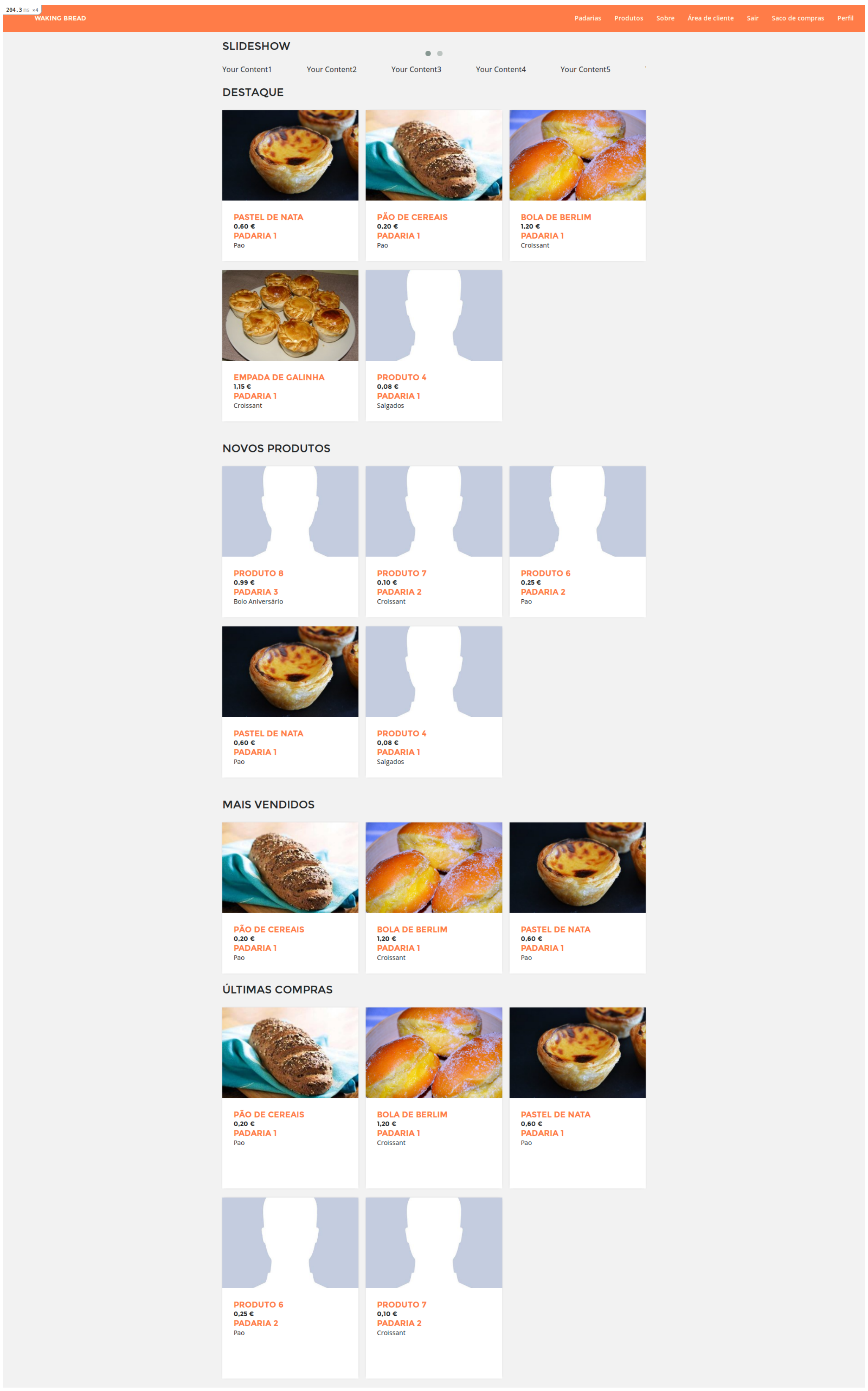


Figure 6.5: User Area - Homepage User Logged In

The second map is for when a user wants to add a new address. Due to the fact that the Google Maps API may not get the actual location from the address fields, the user is obliged to exam the form fields so that the platform can check the address inserted and show it on the map. Thus, the user can check if the location they inserted is correct and was well interpreted by Google Maps.

If so, then the user can create the address, otherwise, they can move the marker if the address interpreted was just a little bit off or change the value of the fields and try examine the address again.

The use of the Google Maps API is explained on section 6.5.

Purchase history

This page has as its objective to let the user know what their last purchases were and how much they have been spending. In it the user will see a list of all their last purchases, showing how must the purchases cost, what products were bought in each purchase, when the purchase was made and more.

Bakery page

On the bakery page the user can consult the bakeries information, their photo gallery, the payment methods that they have available, their addresses as well as some products.

The addresses represent the location of their stores and are shown in text format and in a map for a better visualization.

In the case of the products, since only a small set of products are shown in this page, it is important that the ones selected are the most relevant. Therefore, the selection process followed a simple criteria. First the featured products, then the products that are specialties of the bakery, and finally the rest of the products.

Finally if the user wants to check all products of the bakery, they can do so by clicking the button that will redirect them to a page showing every product of the bakery.

Product page

The product page is a simple page that shows all the information about a product, from the price to the time required for it to be delivered. Additionally, if the user is logged in, a form to add the product to the bag is shown.

About

The about page shows all the information about the platform, what it is, how it works, who is it destined for and more.

Client Area

Lastly, this page is where the bakeries can get more information about how they can sell their products in the platform, the conditions given to them when they join the platform and more.

6.3 Bakery

On this section the features that the platform provides to the bakeries are described and explained. First, all pages developed are listed and then the subsections 6.3.1, 6.3.2, 6.3.3 and 6.3.4 describe and explain the features developed on the main pages. At last, the subsection 6.3.5 does a brief description of the remaining features implemented on the pages not mentioned in the other subsections.

The bakery area consists of 11 pages, excluding the pages related to the login that will be described on the section 6.5.1. All of these 11 pages require the worker to be logged in into the bakery area. These pages are:

- Profile;
- Edit Profile;
- Edit Worker;
- Statistics;
- Orders;
- Distribution;
- Products;
- Promotions;
- Gallery;
- Manage Employees;
- Contract.

In addiction to these pages, there are other pages related to some of them, such as the page Products that includes the pages New Product, Edit Product and Show Product. A better description of these pages can be seen in the following subsections.

Due to the fact that the worker can be an admin or just a regular employee, there are some restrictions while accessing these pages. While the admin has access to every page, the worker only has access solely to the Orders, Products and Edit Worker pages.

Note that, as in the previous section, the images shown on this section do not have the final design and layout implemented, due to the same reasons.

6.3.1 Manage Products

The possibility to add products and see the orders are the two major features that the platform needed to provide to the bakeries, and without them the platform could not work.

This subsection explains and describes how the bakeries can manage their products. But, before explaining this process it is necessary to know the attributes that a product has.

The attributes are the following:

- **Name** - name of the product;
- **Price** - price of the product;
- **Picture** - picture of the product;
- **Info** - any extra info that the bakery might find relevant about the product;
- **Category** - category of the product;
- **Sub-categories** - sub-categories to which the products belongs;
- **Specialty** - if the product is a specialty of the bakery or not;
- **Time to delivery** - time required after the purchase for the product to be delivered;
- **Limit** - maximum number of units per purchase;
- **Visible** - if the product is visible or not to the users;
- **Featured** - if the product is a featured product of the bakery or not.

The process of managing the products is simple and easy to understand. When a worker goes to the Products page, they will first see a list of all the products of the bakery, showing also a few of the most important attributes of the products. Besides this list, the worker will also see a button to add new products, three buttons on the top of the page that help with the visualization of the products, and then, for each product on the list there is also three other buttons.

By clicking on the button to add a new product, the worker is redirected to a page with a form, for the worker to fill, with a field for each one of the attributes mentioned above. After they are done filling the form they can add the product.

The three buttons on the top of the page are intended to let filter the products by their visibility. This lets the worker see only the products that are visible to the users, the hidden products or all of them.

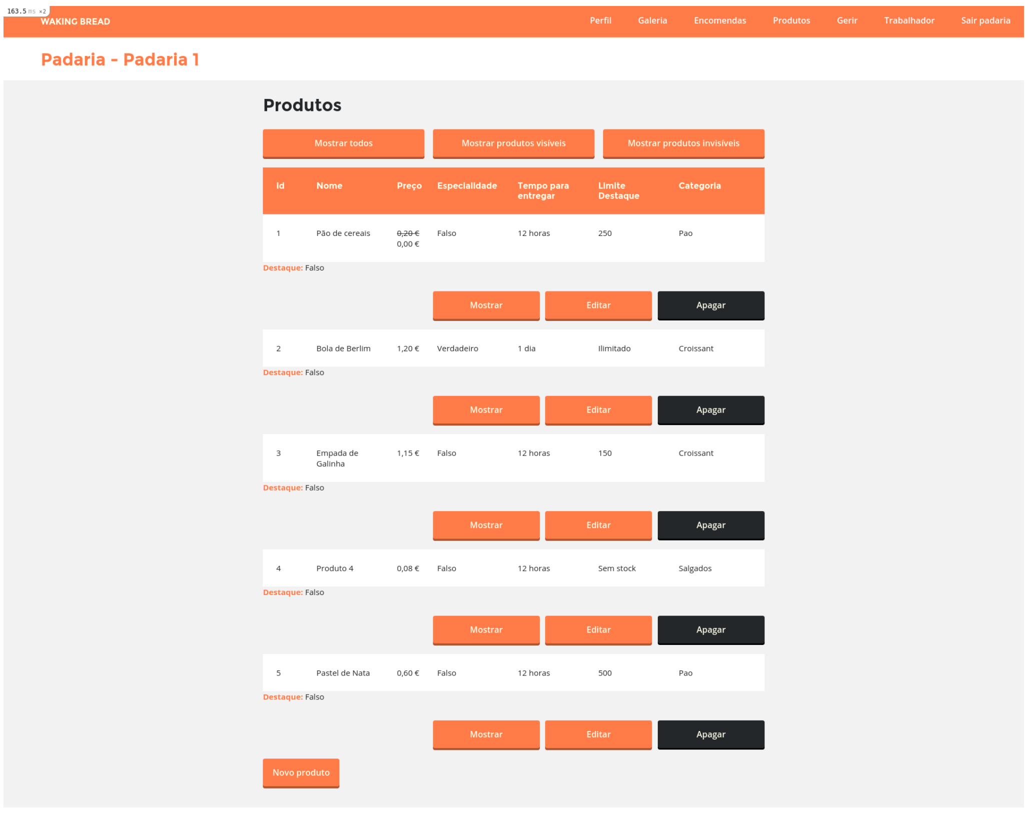
Finally, the other three buttons present on each one of the products, are the actions that the worker can perform for each product. These actions are:

- **Show** - redirects the worker to a page with all the information about the product;
- **Edit** - redirects the worker to a page where they can edit the product information;
- **Delete** - deletes the product after confirmation.

With all these functionalities, the bakery has a perfect control on the management of the products they want to sell. Image 6.6 shows the Products page.

6.3.2 Manage Distribution

Since the beggining of the project, the author noted that there were a lot of aspects that the bakeries should be able to manage, in order to better control the distribution process. They needed to set their delivery hours, delivery taxes, delivery dates as well as the delivery areas. So, to manage all of these, the Manage Distribution page was created. This



Padaria - Padaria 1

Produtos

Id	Nome	Preço	Especialidade	Tempo para entregar	Limite Destaque	Categoria
1	Pão de cereais	~~0,20 €~~ 0,00 €	Falso	12 horas	250	Pao
2	Bola de Berlim	1,20 €	Verdadeiro	1 dia	Ilimitado	Croissant
3	Empada de Galinha	1,15 €	Falso	12 horas	150	Croissant
4	Produto 4	0,08 €	Falso	12 horas	Sem stock	Salgados
5	Pastel de Nata	0,60 €	Falso	12 horas	500	Pao

Destaque: Falso (for each product)

Figure 6.6: Bakery Area - Manage Products

page can be seen on image 6.7.

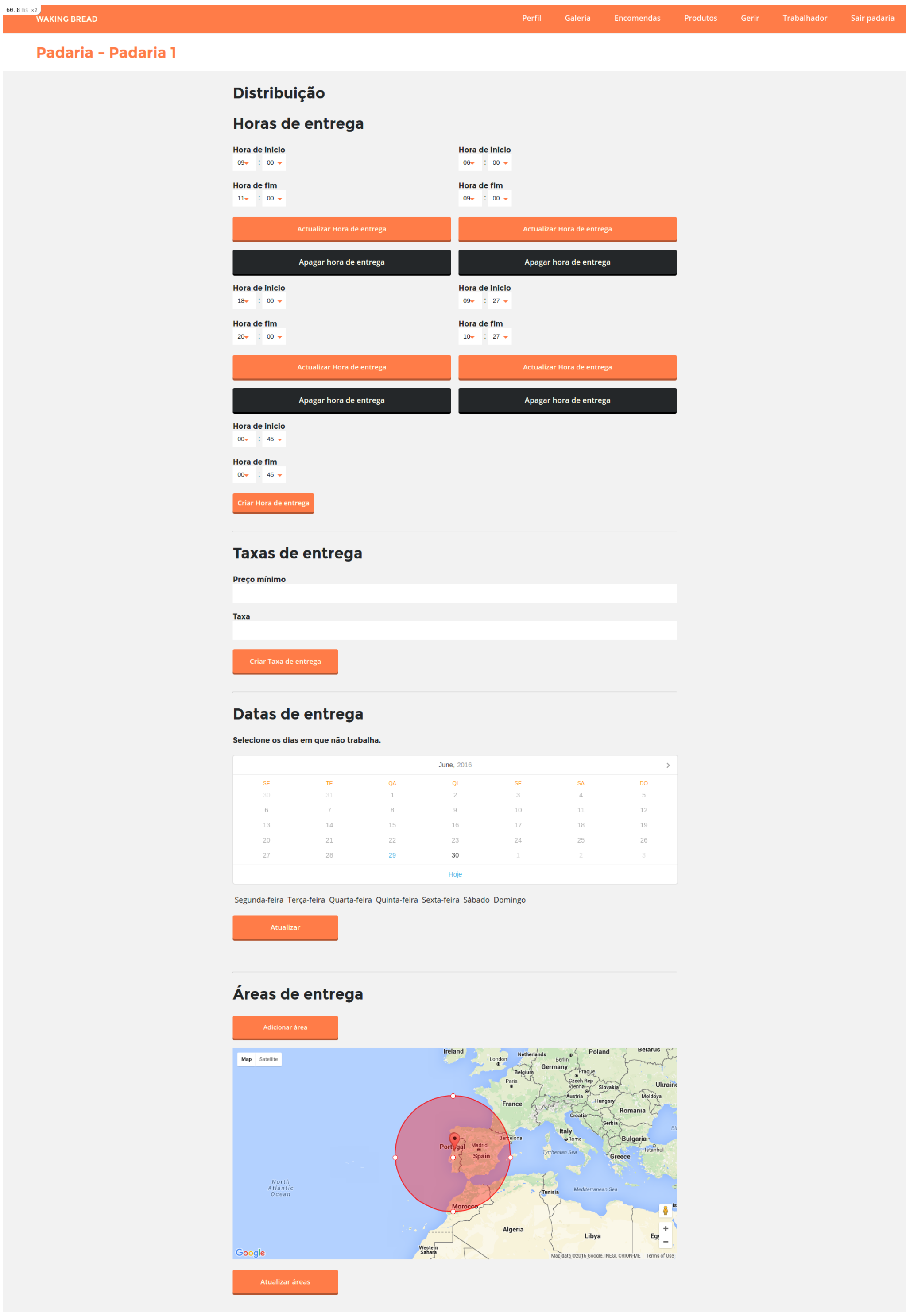
If a bakery does deliveries more than one time a day, it is important for them to let the user choose which one they prefer. Due to this, the platform lets the bakeries add, update or delete delivery hours. So, in case they want to add or update a delivery hour, they just need to set the start hour and end hour of the delivery.

Since the platform does not apply any kind of delivery tax on the purchases, this needs to be done by the bakeries themselves. They can do so by setting the tax price and the minimum price for the tax to be applied on the purchase. Also, they can update or delete the delivery taxes.

Due to the fact that the bakeries do not work every single day of the year, or even the week, the platform needed to provide a way for them to choose the days that they do not work. To do this, a calendar is presented to the admins of the bakery, where they can select specific days to not work. In addiction, there is also present a checkbox for each day of the week for them to select their days off.

At last, the bakeries needed to select the areas where they could deliver. The author chose to implement this feature by presenting the admins of the bakeries a map, with their bakery locations marked, where they can add areas of delivery. The areas are circles with adjustable radius that the bakeries that move around in the map, making the area that the circle covers, available for delivery. This feature was implement using the Google Maps API.

Note that this page can only be access by admins of the bakery.



Padaria - Padaria 1

Distribuição

Horas de entrega

Hora de inicio 09 : 00	Hora de inicio 06 : 00
Hora de fim 11 : 00	Hora de fim 09 : 00
Hora de inicio 18 : 00	Hora de inicio 09 : 27
Hora de fim 20 : 00	Hora de fim 10 : 27

Hora de inicio 00 : 45

Hora de fim 00 : 45

Taxas de entrega

Preço mínimo

Taxa

Datas de entrega

Selecione os dias em que não trabalha.

June, 2016

SE	TE	QA	QI	SE	SA	DO
30	31	1	2	3	4	5
6	7	8	9	10	11	12
13	14	15	16	17	18	19
20	21	22	23	24	25	26
27	28	29	30	1	2	3

Hoje

Segunda-feira Terça-feira Quarta-feira Quinta-feira Sexta-feira Sábado Domingo

Áreas de entrega

Figure 6.7: Bakery Area - Manage Distribution

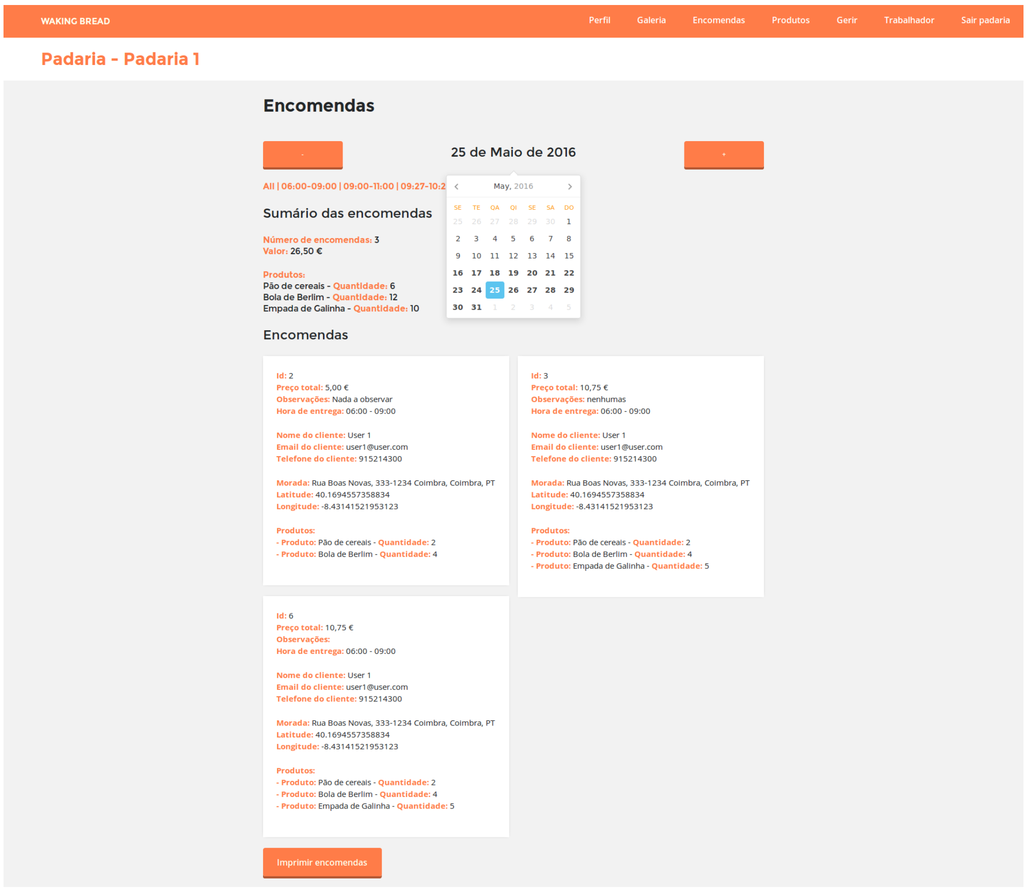
6.3.3 Check Orders

Being this one of the main features on the bakeries area, the Orders page had to be simple to use and understand, and able to facilitate the delivery process as much as possible.

In the initial phase of the project, the author found that the best way to show the orders that the bakery needed to deliver, was to show them by day. Meaning that the bakery should be able to scroll between days to see the deliveries of those days.

Later during the testing phase, the usability tests led to some improvements that could be done to facilitate this process for the bakeries, such as filter the orders by delivery hour.

After the initial implementation and the final improvements, the page was completed and can be seen on image 6.8.



Padaria - Padaria 1

Encomendas

25 de Maio de 2016

All | 06:00-09:00 | 09:00-11:00 | 09:27-10:2

Sumário das encomendas

Número de encomendas: 3
Valor: 26,50 €

Produtos:
Pão de cereais - Quantidade: 6
Bola de Berlim - Quantidade: 12
Empada de Galinha - Quantidade: 10

Encomendas

Id: 2
Preço total: 5,00 €
Observações: Nada a observar
Hora de entrega: 06:00 - 09:00

Nome do cliente: User 1
Email do cliente: user1@user.com
Telefone do cliente: 915214300

Morada: Rua Boas Novas, 333-1234 Coimbra, Coimbra, PT
Latitude: 40.1694557358834
Longitude: -8.43141521953123

Produtos:
- Produto: Pão de cereais - Quantidade: 2
- Produto: Bola de Berlim - Quantidade: 4

Id: 3
Preço total: 10,75 €
Observações: nenhumas
Hora de entrega: 06:00 - 09:00

Nome do cliente: User 1
Email do cliente: user1@user.com
Telefone do cliente: 915214300

Morada: Rua Boas Novas, 333-1234 Coimbra, Coimbra, PT
Latitude: 40.1694557358834
Longitude: -8.43141521953123

Produtos:
- Produto: Pão de cereais - Quantidade: 2
- Produto: Bola de Berlim - Quantidade: 4
- Produto: Empada de Galinha - Quantidade: 5

Id: 6
Preço total: 10,75 €
Observações:
Hora de entrega: 06:00 - 09:00

Nome do cliente: User 1
Email do cliente: user1@user.com
Telefone do cliente: 915214300

Morada: Rua Boas Novas, 333-1234 Coimbra, Coimbra, PT
Latitude: 40.1694557358834
Longitude: -8.43141521953123

Produtos:
- Produto: Pão de cereais - Quantidade: 2
- Produto: Bola de Berlim - Quantidade: 4
- Produto: Empada de Galinha - Quantidade: 5

Figure 6.8: Bakery Area - Check Orders

So, on this page, the workers can see the orders of the selected day, with their complete information. Then, they can scroll between days on two possible ways. They can either click on the buttons to go to the next or previous day, or click on the current selected day and a calendar will appear with the days that have orders signalized.

If they desire to check only the orders that have a specific delivery hour, instead of showing all of them, they can do so by clicking on the delivery hour they want and the orders will be filtered.

In addiction, a summary of all orders of the selected day is presented. This lets the bakers know the quantities of each product that they need to make, so that they can satisfy every order, without the need for them to check each order individually and do the calculations themselves.

At last, the platform gives the bakeries the possibility of printing the orders with all the information necessary. To do this, the worker needs to click on the button to print the orders, and they will be redirected to a new blank page with all the orders listed.

6.3.4 Check Statistics

In order for the bakeries to know how their business is going, so that they can have a better control over it, the need for a page that could show their sales as well as some statistics was crucial. That page was the Statistics page.

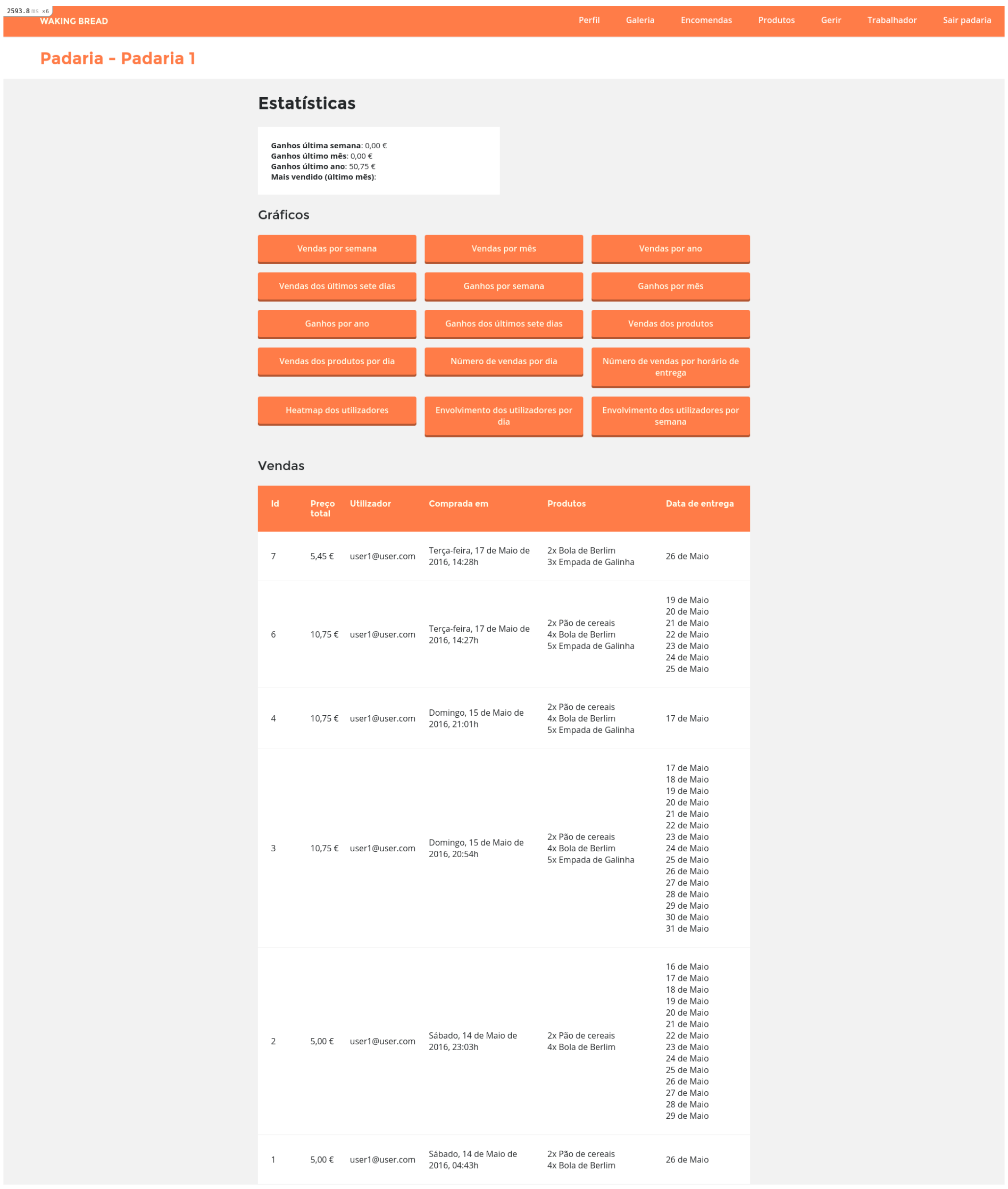
In it, the admins of the bakery can see some simple statistics, such as how much they made in the last week and month, some graphs and also a list of all their last sales. The image 6.9 shows how this page.

The graphs that are available to the bakeries are:

- **Sales by week / month / year** - shows the number of sales by week, month and year;
- **Sales of the past 7 days** - shows the number of sales by day during the last seven days;
- **Earnings by week / month / year** - shows how much the bakery has been making by week, month and year;
- **Earnings of the past 7 days** - shows how much the bakery has been making by day during the last seven days;
- **Product sales** - shows a pie chart with the most sold products;
- **Product sales by day** - shows the sales of each product by day of the week;
- **Number of sales by day** - shows the total number of sales for each day of the week;
- **Number of sales by delivery hour** - shows the total number of sales for each delivery hour of the bakery;
- **Users heatmap** - shows a heatmap of where the bakery users are located;
- **User engagement by day** - shows the percentage of users that were active by day;
- **User engagement by week** - shows the percentage of users that were active by week;

While most of these graphs were fairly easy to implement, the last three were more difficult, and were not as trivial as the others. To develop the users heatmap, the author had to use the Google Maps API, and to develop the last two graphs, the author had to study and use two user engagement metrics. These metrics were the DAU (Daily Active Users) and WAU (Weekly Active Users). They calculate the percentage of unique active users by day and week, respectively. In our case, the users were only considered to be active, in a particular day or week, if they have done a purchase from the bakery.

Lastly, the list of sales shows all of the bakery sales ordered by purchase date. Each element of the list has the total price payed by the user to the bakery, the user that made the purchase, the products sold and their quantities as well as the delivery dates.



Padaria - Padaria 1

Estatísticas

Ganhos última semana: 0,00 €
Ganhos último mês: 0,00 €
Ganhos último ano: 50,75 €
Mais vendido (último mês):

Gráficos

Vendas por semana
Vendas por mês
Vendas por ano
Vendas dos últimos sete dias
Ganhos por semana
Ganhos por mês
Ganhos por ano
Ganhos dos últimos sete dias
Vendas dos produtos
Vendas dos produtos por dia
Número de vendas por dia
Número de vendas por horário de entrega
Heatmap dos utilizadores
Envolvimento dos utilizadores por dia
Envolvimento dos utilizadores por semana

Vendas

Id	Preço total	Utilizador	Comprada em	Produtos	Data de entrega
7	5,45 €	user1@user.com	Terça-feira, 17 de Maio de 2016, 14:28h	2x Bola de Berlim 3x Empada de Galinha	26 de Maio
6	10,75 €	user1@user.com	Terça-feira, 17 de Maio de 2016, 14:27h	2x Pão de cereais 4x Bola de Berlim 5x Empada de Galinha	19 de Maio 20 de Maio 21 de Maio 22 de Maio 23 de Maio 24 de Maio 25 de Maio
4	10,75 €	user1@user.com	Domingo, 15 de Maio de 2016, 21:01h	2x Pão de cereais 4x Bola de Berlim 5x Empada de Galinha	17 de Maio
3	10,75 €	user1@user.com	Domingo, 15 de Maio de 2016, 20:54h	2x Pão de cereais 4x Bola de Berlim 5x Empada de Galinha	17 de Maio 18 de Maio 19 de Maio 20 de Maio 21 de Maio 22 de Maio 23 de Maio 24 de Maio 25 de Maio 26 de Maio 27 de Maio 28 de Maio 29 de Maio 30 de Maio 31 de Maio
2	5,00 €	user1@user.com	Sábado, 14 de Maio de 2016, 23:03h	2x Pão de cereais 4x Bola de Berlim	16 de Maio 17 de Maio 18 de Maio 19 de Maio 20 de Maio 21 de Maio 22 de Maio 23 de Maio 24 de Maio 25 de Maio 26 de Maio 27 de Maio 28 de Maio 29 de Maio
1	5,00 €	user1@user.com	Sábado, 14 de Maio de 2016, 04:43h	2x Pão de cereais 4x Bola de Berlim	26 de Maio

Figure 6.9: Bakery Area - Check Statistics

6.3.5 Other Features

Now that all the main features were explained in the last 4 subsections, it is time to describe the features implemented in the remaining pages.

Profile

If an admin of the bakery is logged in, they have the possibility to see the bakery profile. In it, they can see every information about it, such as the email, contact number and more. Also, in this page they can go to the Edit Profile page and request a user notification.

The user notification request is a request that the bakery does to the admins of the platform, so that they send a notification, via email, to the users, informing them about the bakery. Then, if the admins accept the request, a email will be sent to the users that are inside the delivery range of the bakery.

Edit Profile

This page is destined to the admins of the bakery, so that they can edit the profile of their bakery. They can change the photo and cover of the bakery, edit the name, email and all of the remaining fields.

The process of editing the bakery addresses works exactly the same way as the process of editing the user addresses explained on the subsection 6.2.3.

Edit Worker

The Edit Worker page is where the workers of the bakery, being admin or not, can change their login information. To do so, they need to provide their current password and then they can change both their email and password.

Promotions

On this page, the admins of the bakery can manage the promotions of the bakery. The promotions work as a percentage that when applied to a product, it reduces the price by that percentage.

They can add new promotions and edit or delete promotions listed on the page. The actions of adding and editing a promotion redirect the admin to other pages, the Add Promotion and Edit Promotion, respectively. These pages have a simple form, for the user to fill or edit, with the information about the promotion.

To apply the promotion, the admins need to go the the Products page, and edit the products that they want to be in promotion.

Gallery

The gallery of the bakery is where the admins can add or remove photos of the bakery. All the photos can be seen on this page and each one has a label given by the admin.

The admins are able to add new photos, edit the existent ones or remove them. When adding or editing a photo, the admins are redirected to the pages Add Photo and Edit Photo, respectively. On this pages, the admins can choose the photo they want to upload and write the corresponding label.

Manage Employees

On this page, the admins of the bakery can invite their employees to the platform, so that they can have access to the orders and products. Also, they can manage their workers in this page. They can change their names, set them as admins of the bakery or delete them in case they are no longer employees of the bakery.

To edit a worker, the admin is redirected to a Edit Worker page, where he is presented with a form to edit the fields of the worker. The invitation system was developed with the help of Devise. An explanation of Devise is present on the section 6.5.1.

Contract

At last, the page Contract is where the admins of the bakery can find all the information about their current contract.

6.4 Admin

In this section, all functionalities developed for the admin area of the web platform are described. This area was developed with the help of a Ruby on Rails framework named Active Admin[1]. This is a framework that generates a very functional backoffice for applications in Ruby on Rails.

The admin area consists on 12 pages that help the admins of the platform manage data of both bakeries and users and also visualize important information. The pages are:

- Dashboard;
- Check Products;
- Check Purchases;
- Manage Admins;
- Manage Bakeries;
- Manage Categories;
- Manage Contracts;
- Manage Notification Requests;
- Manage Payment Methods;
- Manage Sub-Categories;
- Manage Users
- Manage Workers.

To access these pages, the admins need to login to their specific area on the platform. After they successfully sign in, they are redirected to the homepage of the admin area, the Dashboard.

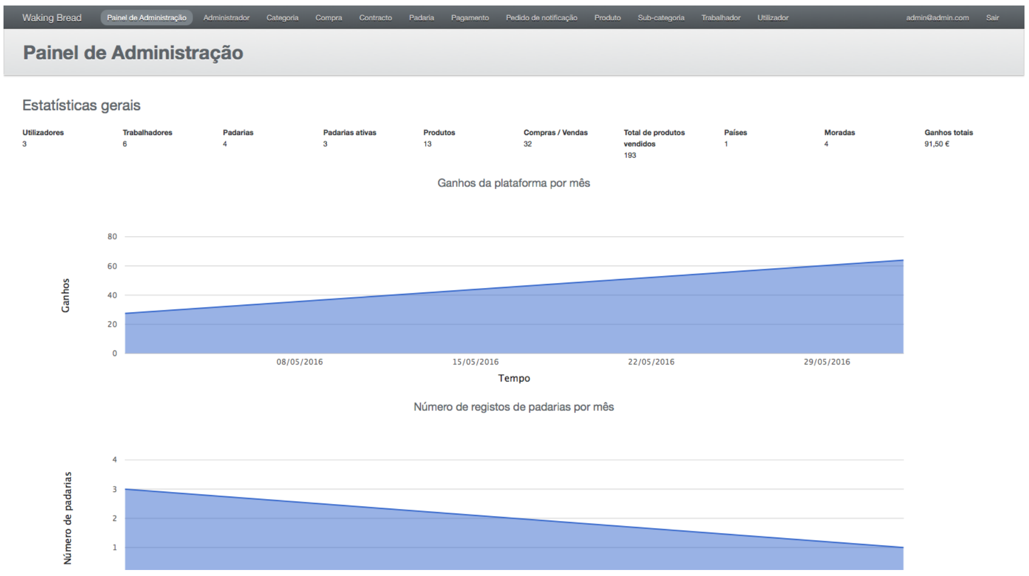
This page is the most important one of the admin area. In it, statistics and graphs are presented to the admins to help them visualize and understand how their and the bakeries business is going. The graphs presented in this page are the following:

- **General Graphs:**
  - Earnings of the platform by month;
  - Number of bakery registrations by month;
  - Number of user registrations by month.

[1]More information at: http://activeadmin.info/

- **Bakery Graphs:**
  - Sales by bakery;
  - Sales by week and month;
  - Earning by week and month;
  - Number of products of the bakeries.
- **Product Graphs:**
  - Product categories;
  - Product sub-categories.
- **Other Graphs:**
  - Countries represented.

The image 6.10 shows a partial of the Dashboard page as well as the menu of the Admin Area.



Painel de Administração

Estatísticas gerais

Utilizadores	Trabalhadores	Padarias	Padarias ativas	Produtos	Compras / Vendas	Total de produtos vendidos	Países	Moradas	Ganhos totais
3	6	4	3	13	32	193	1	4	91,50 €

Figure 6.10: Admin Area - Dashboard

To finalize this section a brief explanation of the 11 remaining pages is presented. It is worth noting that these pages are all very similar to each other since they have the same layout. An example of one of these pages can be seen on image 6.11. In addiction, the lists present in these pages can be downloaded in three different formats: CSV, XML and JSON.

Check Products

This page is where the admins can consult every single product present in the platform. They have the possibility to filter or order them by price, name, bakery, category and more.

Check Purchases

In this page, it is possible to visualize every purchase that has made in the platform. Also, it is possible to order and filter these purchases by date, bakery, user and more.

Manage Admins

This page is destined to add more admins to the platform since initially only one admin is created. Also, it is in here that the admins can change their login information or even remove other admins from the platform.

Manage Bakeries

As the name suggests this is the page where the admins can manage the bakeries. They can add new bakeries to the platform, remove them if necessary and even check their information. In addiction they can visualize all bakeries as well as filter and order them for a better visualization.

Noteworthy that, to add new bakeries, they must first contact the admins of the platform through the pre-registration form and then agree to the terms of the contract. In order to the process of adding a new bakery be complete, the admin needs to add a new contract and a new worker to the bakery, and assign the payment methods that the bakery has available.

Manage Categories

This page is where the admins manage the product categories present in the platform. They have the possibility to add new categories, remove or edit existent ones, and also visualize all of them.

Manage Contracts

In this page the admins can create new contracts to assign to the bakeries. They can also visualize all current and past contracts and even edit or remove them. To facilitate the visualization of the contracts, the admins can filter and order them by expiration date, bakery and more.

Manage Notification Requests

This page is where all notification requests made by the bakeries are presented. The admins have the possibility to accept the request and send a notification to the users that are inside the bakery delivery range, or ignore the request if for instance they have sent another request recently.

Manage Payment Methods

This page is destined to manage the payment methods that the platform supports. In it the admins can add new payment methods as soon as they are implemented in the platform or remove one if it stops being supported by it.

Manage Sub-Categories

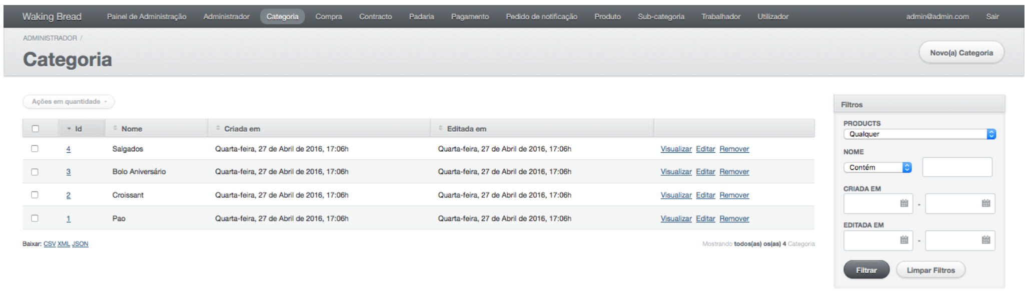
Similar to the page Manage Categories, this page has the same function and properties with the difference being that it deals with product sub-categories instead of product categories.

Manage Users

In this page the admins can check the public information of the users of the platform.

Manage Workers

Lastly, this page is where the workers information is presented. The admins can add, remove or edit workers when necessary. Additionally they can order and filter them by bakery, name, whether they are admins of their bakery or not, and more.



Id	Nome	Criada em	Editada em	
4	Salgados	Quarta-feira, 27 de Abril de 2016, 17:06h	Quarta-feira, 27 de Abril de 2016, 17:06h	Visualizar Editar Remover
3	Bolo Aniversário	Quarta-feira, 27 de Abril de 2016, 17:06h	Quarta-feira, 27 de Abril de 2016, 17:06h	Visualizar Editar Remover
2	Croissant	Quarta-feira, 27 de Abril de 2016, 17:06h	Quarta-feira, 27 de Abril de 2016, 17:06h	Visualizar Editar Remover
1	Pao	Quarta-feira, 27 de Abril de 2016, 17:06h	Quarta-feira, 27 de Abril de 2016, 17:06h	Visualizar Editar Remover

Baixar: CSV XML JSON

Mostrando todos(as) os(as) 4 Categoria

Figure 6.11: Admin Area - Manage Categories

6.5 Other Features

Besides the features mentioned in the last three sections there were also other features implemented that may not be visible at first sight, or that were just not explained. In this section all of those extra features are explained and described.

6.5.1 Logins and Registrations

The authentication for both the user and the bakeries as well as the invitation system for the employees of the bakeries was accomplished with the help of a Ruby on Rails gem. The gem was Devise[2] and it helps generating a functional, complete and secure authentication system on a simple and fast way with only a little effort required on the part of the developer.

In addiction to the login and registration page that this gem helped generating, it also helped generating the pages needed to reset a password, invite someone to the platform or to confirm the registration.

[2]More information at: https://github.com/plataformatec/devise

6.5.2 Notifications and Alerts

In a web platform it is essential that a user knows at all times what action he just performed, and the status of it. If it went well or if something went wrong and the action could not be done. Because of this it is necessary for the platform to give some kind of feeback on the actions the user performs.

This platform does that by always showing a notification or an alert when a user or bakery performs an action within the platform.

6.5.3 Pagination

The platform has a pagination system implemented in multiple pages, both in the user area and in the bakery area. This prevents the users and the bakeries from doing a lot of scrolling to check information. Instead they can just navigate between pages. This pagination system was accomplished with the help of the paginator Kaminari[3].

6.5.4 Mailer

One of the things that can not be seen at first sight is the mailer system that the platfrom has implemented on it. This system serves three purposes:

- Send an email to the users when a bakery notification request is accepted;
- Send an email with the information of a purchase that a user just made to the bakery;
- Send an email to the admins of the platform with the information of a pre-registration made by a bakery.

This mailer system can later be used to send other kind of notifications via email.

6.5.5 Google Maps Javascript API

Finally the Google Maps Javascript API[4]. This API was used in many ways and in many pages of the platform, and really helped implementing some crucial features.

The great documentation and code samples provided by it, proven to be of a great help while implementing the features that required this API. Also, the libraries and services that it provides avoided extra work hours. The services, libraries and other features used and where they were used on the platform are:

- **Places Library** - used when a user needs to write an address, such as in the homepage or in the searches, to show suggestions of addresses while they types their address;
- **Drawing On The Map Service** - used to show the markers on the maps, insert an info window on the markers, draw the circles on the map while adding the delivery ranges of the bakeries and more.
- **Geolocation Service** - used to get the current geolocation of a user / bakery when they try to add an address;

[3]More information at: https://github.com/amatsuda/kaminari
[4]More information at: https://developers.google.com/maps/documentation/javascript/

- **Geocode Service -** used to get the latitude and longitude of an addresses when the user or bakery tries to add or update a address;
- **Heatmap Layer -** used to show the heatmap of the locations of the bakery users;

Beyond this, some events of this API were also used to help with some tasks, like detecting the current position of a marker when it is moved.

Chapter 7

Tests

In this chapter the tests executed on the web platform are presented. It starts with the functional tests, then the security tests and then the database tests. All of these tests will be described and explained how they were performed, and in the end, a final conclusion on the results obtained after the testing is presented.

At last, the process of how the usability tests were performed is described.

7.1 Functional

The functional tests are used to check if every functionality developed on the web platform is correctly implemented. Besides the traditional testing that a developer is used to doing while developing, it is essential that more detailed unit tests are implemented. To do this type of tests the author chose to use the Ruby on Rails testing tool Rspec[1].

This tool is the most used testing tool for Ruby on Rails, mostly because it is simple and easy to use and its syntax is easy to understand and read. Rspec is oriented towards the idea of a Behavior-Driven Development (BDD) instead of the more traditional idea, Test-Driven Development (TDD).

A brief difference between these two ideas is that, while TDD focus is to test the features implemented to see if they work, BDD focus is to test how the components of the application should behave.

So, with RSpec it was possible to create tests for five key aspects of the Ruby on Rails platform:

- **Controllers** - test if the controller is working properly, and if no error occurs while performing their actions;
- **Models** - test if a model accepts the right data and if returns error if the data is corrupted or wrong;
- **Views** - checks if any error occurs when a view is being rendered;
- **Requests** - test to check if a request for a page succeeds or not, and if the results obtained is the correct;

[1]More information at: http://rspec.info/

- **Routing** - tests to see if the routings are working properly;

These tests are after run to verify if they passed or not. This, allows the author to check if there was any error in the any aspect of the platform, and if so where it occurred.

After creating and running these tests, was possible to see that there were some things that were not perfectly implemented and that were causing errors, such as Models accepting data they should not accept and problems with controllers that were not passing the correct information to the views. These things were promptly resolved until they pass on the tests.

7.2 Security

For the security tests to be successful, there were two things that needed to be tested:

- Check if users or workers could not access pages that they do not have access to;
- Check if there is any security flaw in the platform.

To verify the first situation, the author logged as an user, admin of a bakery, worker of a bakery, and admin of the platform and checked individually every single page of the platform to see if all permissions were properly defined.

The second situation was accomppished with the help of a gem for Ruby on Rails called Brakeman[2]. This gem is a security scanner, that scans for possible security vulnerabilities in the platform, such as SQL injections, mass assignments and more.

After running this gem, the results obtained showed that there is no security flaw detected in the platform.

7.3 Database

Testing the database used by the platform consisted on verifying three aspects:

- Check if the data is correctly stored;
- Check whenever some information is changed or deleted on the platform, if the data is changed or deleted in the database of if it stays the same;
- Check if the connections in the database are correct.

To verify the first two aspects, the author started by using the platform intensively, making every single request possible to the database more than one time. Then, he went to check if the data in the database was not corrupted and if it was still consistent.

The last aspect, was accomplished by generating the ER diagram of the database or comparing it with the ER diagram created during the planing phase of this project.

The results obtained on both verifications was favorable.

[2]More information at: http://brakemanscanner.org/

7.4 Usability

The usability tests with a test group of people from different ages and occupations was part of the initial plan for this project. However, due to the second problem mentioned on the subsection 5.1.3 these tests could not be done.

Therefore, the only usability tests made were the ones with the partner bakery of this project. On them, the responsible of the bakery was conducted during his interaction with the platform in order to understand if there were things that needed fixing or features that should be implemented.

This test resulted on the development of some additional features, such as the filter and the summary present on the Orders page.

Later, when the platform is online and has a better design, more usability tests should be done, such as the ones mentioned on the section 2.3.

Chapter 8

Conclusions

This report describes the different phases of a software engineering process, in this case a web platform. This web platform basically consists of an online shop that allows users to order products from a bakery to be delivered at their home. Its main goal is to provide fresh products in the morning for everyone, everywhere, without the need to leave home.

In the first phase of this project, three studies were made about things related to it. First a market study was done in order to understand if there was already any type of competition in this particular area, or if there was a similar service in any other area. Then other study was conducted to get a better understanding of what types of technologies are being used to develop web platforms, and to know their specific characteristics and advantages. At last, a study was conducted with the purpose of get a better understanding of what kind of usability techniques are being used nowadays in web platforms.

After this phase, the planning for the whole project was created. The tasks for each semester were defined along with a Gantt diagram of each semester and the methodology used in the project was chosen. Other choices were also made, such as the technologies in which the platform was developed and the productivity tools used to help with the different aspects of the development.

The requirements were defined in different ways in order to have more detailed requirements. The database was designed as a navigation diagram as well, wireframes for each page of the web platform were created and the system architecture was defined.

During the implementation phase, all functionalities to make the web platform functional were developed and are fully operational. This process begun with the implementation of the bakery area, then the user area and it ended with the implementation of the admin area.

After the implementation phase the unit and functional tests were made, began the testing phase. This phase helped finding some errors and bugs of the platform, what leaded to make some improvements.

For the future, there are a couple of features that should be implemented, and most important, the application needs to get a new and improve design so that it can be deployed and ready to use.

8.1 Future Work

With this being a big and complex web platform, there are always space for improvements that can benefit the end user. This section describes some of the improvements, already thought by the author, that can be implemented in the platform to help improving some aspects of it as well as some things that need to be done, such as the deployment and the design of the platform.

8.1.1 Design

As it is obvious at this point, the current design of the platform is far from being perfect and even functional. Because of this, the first thing that should be done after this thesis is completed, is redo the whole design and make the platform clean, beautiful and usable.

8.1.2 Deploy

The deployment of the web platform stills needs to be done, as soon as the design is finished. The platform will be put and configured on the servers system used by HYP. After that, the usability tests should be executed, with the help of a group of people from different ages and occupations.

8.1.3 Features

There are a lot of features that can still be implemented in the future, with the objective of making the platform better. Some of the features thought by the author, that can be implemented are:

- Favorite products and bakeries;
- Comparison between products;
- Order searches;
- A recommendarion system, that would show product and bakery recommendations to the user;
- Pick up in store option when making a purchase;
- Possibility to the users to give a feedback on the purchase to the bakery;
- Subscribing service that allows the user to receive an email as soon as there bakeries that can deliver in the location requested;
- Notification service that notifies the user when a deliver is made.

References

[1] Laravel. Love beautiful code? we do too, 2016. Available online in https://github.com/laravel/laravel, last consulted on 10/01/2016.

[2] GitHub. Laravel, 2016. Available online in https://laravel.com/, last consulted on 10/01/2016.

[3] BuiltWith. Framework technologies web usage statistics, 2016. Available online in http://trends.builtwith.com/framework, last consulted on 10/01/2016.

[4] SitePoint. The best php framework for 2015: Sitepoint survey results, 2016. Available online in http://www.sitepoint.com/best-php-framework-2015-sitepoint-survey-results/, last consulted on 10/01/2016.

[5] Glenn E Krasner, Stephen T Pope, et al. A description of the model-view-controller user interface paradigm in the smalltalk-80 system. *Journal of object oriented programming*, 1(3):26–49, 1988.

[6] PostgreSQL. Postgresql: The world's most advanced open source database, 2015. Available online in http://www.postgresql.org/, last consulted on 06/11/2015.

[7] MySQL. Mysql, 2015. Available online in https://www.mysql.com/, last consulted on 06/11/2015.

[8] SQLite. Sqlite home page, 2015. Available online in https://www.sqlite.org/, last consulted on 06/11/2015.

[9] Django. The web framework for perfectionists with deadlines, 2016. Available online in https://www.djangoproject.com/, last consulted on 10/01/2016.

[10] Node.js Foundation. Node.js, 2016. Available online in https://nodejs.org, last consulted on 10/01/2016.

[11] Ruby on Rails. Web development that doesn't hurt, 2016. Available online in http://rubyonrails.org/, last consulted on 10/01/2016.

[12] Ruby on Rails Guides. Getting started with rails, 2016. Available online in http://guides.rubyonrails.org/getting_started.html, last consulted on 11/01/2016.

[13] W3Schools. Html introduction, 2016. Available online in http://www.w3schools.com/html/html_intro.asp, last consulted on 15/01/2016.

[14] W3Schools. Css introduction, 2016. Available online in http://www.w3schools.com/css/css_intro.asp, last consulted on 15/01/2016.

[15] Tutorialspoint. Javascript overview, 2016. Available online in http://www.tutorialspoint.com/javascript/javascript_overview.htm, last consulted on 15/01/2016.

[16] Sass. Sass: Syntactically awesome style sheets, 2016. Available online in http://sass-lang.com/, last consulted on 10/01/2016.

[17] Less. Getting started, 2016. Available online in http://lesscss.org/, last consulted on 16/01/2016.

[18] Susy. Susy: Power tools for the web, 2016. Available online in http://susy.oddbird.net/, last consulted on 10/01/2016.

[19] Bootstrap. The world's most popular mobile-first and responsive front-end framework, 2016. Available online in http://getbootstrap.com/, last consulted on 16/01/2016.

[20] Foundation. The most advanced responsive front-end framework in the world, 2016. Available online in http://foundation.zurb.com/, last consulted on 16/01/2016.

[21] Raghu Ramakrishnan and Johannes Gehrke. Database management systems. 2000.

[22] Db-Engines. Db-engines ranking - popularity ranking of database management systems, 2015. Available online in http://db-engines.com/en/ranking, last consulted on 06/11/2015.

[23] Techopedia. What is a web server?, 2015. Available online in https://www.techopedia.com/definition/4928/web-server, last consulted on 28/10/2015.

[24] Mozilla Developer Network. What is a web server, 2015. Available online in https://developer.mozilla.org/en-US/Learn/What_is_a_web_server, last consulted on 28/10/2015.

[25] Netcraft. April 2015 web server survey, 2015. Available online in http://news.netcraft.com/archives/2015/04/20/april-2015-web-server-survey.html, last consulted on 28/10/2015.

[26] Apache. The apache http server project, 2015. Available online in http://httpd.apache.org/, last consulted on 28/10/2015.

[27] Apache. Apache httpd server wiki, 2015. Available online in http://wiki.apache.org/httpd/, last consulted on 28/10/2015.

[28] Nginx. Nginx, 2015. Available online in http://nginx.org/, last consulted on 29/10/2015.

[29] Nginx. Welcome to nginx wiki's documentation!, 2015. Available online in https://www.nginx.com/resources/wiki/, last consulted on 29/10/2015.

[30] Brian Shackel. Usability-context, framework, definition, design and evaluation. *Human factors for informatics usability*, pages 21–37, 1991.

[31] Jakob Nielsen. *Usability engineering*. Elsevier, 1994.

[32] Jakob Nielsen. Usability 101: Introduction to usability, 2016. Available online in https://www.nngroup.com/articles/usability-101-introduction-to-usability/, last consulted on 20/01/2016.

[33] Jakob Nielsen. Top 10 guidelines for homepage usability, 2016. Available online in `https://www.nngroup.com/articles/top-ten-guidelines-for-homepage-usability/`, last consulted on 20/01/2016.

[34] Jakob Nielsen. F-shaped pattern for reading web content, 2016. Available online in `https://www.nngroup.com/articles/f-shaped-pattern-reading-web-content/`, last consulted on 20/01/2016.

[35] Jakob Nielsen. Usability inspection methods. In *Conference Companion on Human Factors in Computing Systems*, CHI '94, pages 413–414, New York, NY, USA, 1994. ACM.

[36] User Stories. User stories: An agile introduction, 2016. Available online in `http://www.agilemodeling.com/artifacts/userStory.htm`, last consulted on 14/01/2016.

[37] Mike Cohn. *User stories applied: For agile software development*. Addison-Wesley Professional, 2004.

[38] Use Cases. System use cases: An agile introduction, 2016. Available online in `http://agilemodeling.com/artifacts/systemUseCase.htm`, last consulted on 14/01/2016.

[39] Lawrence Chung and Julio Cesar Sampaio do Prado Leite. On non-functional requirements in software engineering. In *Conceptual modeling: Foundations and applications*. Springer, 2009.

[40] Nielsen Norman Group. Website response times, 2016. Available online in `https://www.nngroup.com/articles/website-response-times/`, last consulted on 14/01/2016.

[41] Market Interactive. The ruby web benchmark report, 2016. Available online in `http://www.madebymarket.com/blog/dev/ruby-web-benchmark-report.html`, last consulted on 10/01/2016.

[42] M. Ebbers and International Business Machines Corporation. International Technical Support Organization. *Introduction to the New Mainframe: Large-scale commercial computing*. IBM redbooks. IBM, International Technical Support Organization, 2006.

[43] Yu Beng Leau, Wooi Khong Loo, Wai Yip Tham, and Soo Fun Tan. Software development life cycle agile vs traditional approaches. In *International Conference on Information and Network Technology*, volume 37, pages 162–167, 2012.

[44] David Cohen, Mikael Lindvall, and Patricia Costa. Agile software development. *DACS SOAR Report*, (11), 2003.

[45] M Mahalakshmi and M Sundararajan. Traditional sdlc vs scrum methodology–a comparative study. *International Journal of Emerging Technology and Advanced Engineering*, 3(6):192, 2013.

[46] Youssef Bassil. A simulation model for the waterfall software development life cycle. *CoRR*, abs/1205.6904, 2012.

[47] Ben Askins and AA Gree. A rails/django comparison. In *The Open Source Developers Conference Papers*, 2006.

[48] Jonathan S Linowes. Evaluating web development frameworks: Rails and django. *Parkerhill Technology Group LLC. Februari*, 2007.

[49] Klaus Purer. Php vs. python vs. ruby–the web scripting language shootout. In *Seminar aus Programmiersprachen*, pages 103–119, 2009.

[50] DigitalOcean. Sqlite vs mysql vs postgresql: A comparison of relational database management systems, 2015. Available online in `https://www.digitalocean.com/community/tutorials/sqlite-vs-mysql-vs-postgresql-a-comparison-of-relational-database-management-systems`, last consulted on 06/11/2015.

[51] DB-Engines. System properties comparison mysql vs. postgresql vs. sqlite, 2015. Available online in `http://db-engines.com/en/system/MySQL%3BPostgreSQL%3BSQLite`, last consulted on 06/11/2015.

[52] Tim Conrad. Postgresql vs. mysql vs. commercial databases: Its all about what you need, 2006.

[53] P. Prakash, R. Biju, and M. Kamath. Performance analysis of process driven and event driven web servers. In *Intelligent Systems and Control (ISCO), 2015 IEEE 9th International Conference on*, pages 1–7, 2015.

[54] Unicorn. Unicorn: Rack http server for fast clients and unix, 2015. Available online in `http://unicorn.bogomips.org/`, last consulted on 06/11/2015.

[55] Capistrano. A remote server automation and deployment tool written in ruby, 2015. Available online in `http://capistranorb.com/`, last consulted on 06/11/2015.