Software Solution for Food Additives Management

Tiago Padrão

Thesis presented to the School of Technology and Management in the scope of the Master in Information Systems.

Supervisors:

Rui Pedro Lopes

Isabel C.F.R. Ferreira

Bragança

2019

Software Solution for Food Additives Management

Tiago Padrão

Thesis presented to the School of Technology and Management in the scope of the Master in Information Systems.

Supervisors:

Rui Pedro Lopes

Isabel C.F.R. Ferreira

Bragança

2019

Dedicatória

Dedico este trabalho aos meus pais que sempre me deram tudo, permitindo-me alcançar os meus objetivos e por fazerem de mim o Homem que sou hoje.

Resumo

Numa altura em que a sociedade está cada vez mais atenta ao que consome, os aditivos naturais têm ganho maior interesse por parte dos consumidores e da indústria. Devido a alguma controvérsia nos anos noventa e no início do milénio, os aditivos alimentares ainda são vistos com alguma incerteza pela população que, em grande parte, desconhece a sua finalidade, origem e segurança. A informação encontrada na Internet é ainda escassa, difícil de localizar e, muitas vezes, provém de fontes não confiáveis.

Desmistificar e informar a população sobre os aditivos que são adicionados à nossa mesa é uma responsabilidade que requer acesso constante, simples e integrativo, permitindo que a população entenda facilmente o que está a consumir. Existem ainda empresas que necessitam de informação relativamente às quantidades de aditivos alimentares legalmente autorizadas, disponível no site da Autoridade Europeia para a Segurança Alimentar, mas condensada e de difícil acesso.

O objetivo deste projeto consistiu no desenvolvimento de uma solução que agrega os dados fundamentais dos aditivos alimentares e as suas possíveis utilizações numa aplicação móvel de livre acesso. Esta solução é suportada por uma base de dados relacional que armazena as características mais importantes dos aditivos e as classes a que pertencem, bem como uma lista de categorias alimentares, de modo a especificar cuidados e restrições existentes na utilização de certos aditivos em determinadas categorias.

Palavras-Chave: Aditivos Alimentares, Código E, Alternativas Naturais.

Abstract

In a time where society is increasingly interested in what it consumes, natural additives have been gaining attention from consumers and industry. Due to some controversy in the nineties and two thousands, food additives are still viewed with some mistrust by the population, who is largely unaware of their purpose, origin and safety. The information found on the Internet is still scarce, difficult to find, and, in some cases unreliable.

Demystifying and informing people about the additives that are added to our table is a responsibility that requires constant, simple and integrative access, allowing the population to easily understand what they are consuming. Furthermore, some companies require information regarding the legally authorised quantities of food additives, which is condensed and difficult to access on the European Food Safety Authority website.

This project presents a solution that aggregates the fundamental data of food additives and their possibilities of use in an open-access mobile application. This solution is supported by a relational database that stores the most important characteristics of additives and the classes to which they belong, as well as a list of food categories, to specify the warnings and restrictions on the usage of certain additives in some categories.

Keywords: Food Additives, E-Number, Natural Alternatives.

Agradecimentos/Acknowledgments

Agradeço à Fundação para a Ciência e a Tecnologia (FCT, Portugal) e ao FEDER no âmbito do programa PT2020 pelo apoio financeiro ao CIMO (UID/AGR/00690/2019). Este trabalho foi financiado pelo Fundo Europeu de Desenvolvimento Regional (FEDER) através do Programa Operacional Regional Norte 2020, no âmbito do Projeto NORTE-01-0145-FEDER-023289 (DeCodE) e projeto Mobilizador Norte-01-0247- FEDER-024479: ValorNatural.

Antes de mais, agradecer ao Centro de Investigação de Montanha pelo acolhimento, em especial à Professora Isabel Ferreira pela oportunidade de participar neste projeto, bem como toda a ajuda prestada ao longo do seu desenvolvimento e aquando da escrita desta dissertação. Agradecer também à Professora Lillian Barros pela orientação e ajuda ao longo de todo este processo.

Ao Professor Márcio Carocho pela disponibilidade e apoio prestado ao longo de todo este projeto e acima de tudo pela sua amizade.

Um agradecimento especial ao meu orientador e amigo Professor Rui Pedro Lopes, agradecer esta oportunidade, e todo o apoio e conhecimento que me transmitiu durante este percurso, não só ao longo de todo este projeto, mas também ao longo da Licenciatura em Engenharia Informática e do Mestrado em Sistemas de Informação. Sem a sua ajuda, nada teria sido possível.

Um forte abraço ao meu colega e grande amigo Bernardo Lopes, pela sua ajuda não

só no desenvolvimento deste projeto mas também por todo o apoio e amizade ao longo destes anos académicos.

Um agradecimento especial aos meus pais pela força e motivação e a todos os meus familiares que sempre me acompanharam ao longo da minha vida.

Por último, mas não menos importante, agradecer a todos os meus amigos, em especial aos Siga, por estarem sempre do meu lado.

Um sincero obrigado a todos!

Contents

List of Tables

List of Figures

List of Acronyms

ADI Acceptable Daily Intake. 6, 13, 14, 31

API Application Programming Interface. 2, 39, 41, 54

CIMO Mountain Research Center. 2, 30

DTO Data Transfer Objects. 39

EFSA European Food Safety Authority. 5, 6, 8, 9, 13, 14

EU European Union. 6, 9, 14

FAO Food and Agriculture Organization of the United Nations. 6, 13

FDA Food and Drug Administration. 5, 8, 9

GCD Grand Central Dispatch. 53

GUID Global Unique Identifier. 41

HTTP Hypertext Transfer Protocol. 39, 41

IDC International Data Corporation. 16

IDE Integrated Development Environment. 21, 52

IPB Polytechnic Institute of Bragança. 2

JSON JavaScript Object Notation. 53

MVC Model View Controller. 43, 44, 52

SDK Software Development Kit. 17

SOA Service-Oriented Architecture. 37

USA United States of America. 5, 9

W3C World Wide Web Consortium. 23

WHO World Health Organization. 6

Xcode Apples IDE. 21

XML Extensible Markup Language. 23, 24

Chapter 1

Introduction

Food has a vital role in human life. Although the need to feed has kept itself immutable over the years, the way we consume foodstuffs has witnessed deep transformations [1]. The urgent matter to feed the increasing world population has built a need to enhance food production. This process must achieve two fundamental purposes: food must have the highest quality standards and it has to be as cheap as possible. Appearance, texture, taste and microbiological safety are required to be preserved during the whole shelf-life, relying on substances named food additives. These molecules play different roles in foods, ranging from sweetening to preserving and colouring foods. The benefits of these substances are undeniable and so no country forbids them entirely, however, some of them are still enveloped in controversy [2].

The accessible data concerning food additives is distributed throughout many sources and in multiple locations, some of them of insufficient reliability, and sometimes related to general news, reporting the danger of its consumption. It is necessary, as far as possible, to gather, relate and also integrate this information.

Mobile computing is nowadays more present in our daily routines. This can be explained by the great evolution that occurred in this area over the last decade and caused the emergence of a variety of new mobile devices and operating systems, increasingly more sophisticated and powerful [3], [4].

Mobile devices are contributing to facilitate access to information and services regardless of when and were [5], [6], thus, this dissertation proposes a mobile software solution for food additives management, that aims to contribute to improving the access, detail and reliability of information.

The base information is retrieved from several sources, online and offline, containing a list of food categories to specify the existing cautions and restrictions. The integration of data allows to start with a complete and detailed list of additives and associated information. The application is structured in two main componentes: the server, making its functionality available through open Application Programming Interface (API) based on RESTful services that grant access by diverse applications, operating systems and technologies, and a mobile application, providing faster, convenient and more accessible browsing anywhere.

The development of this solution is joint teamwork that for dissertation purposes was divided into two distinct parts. The division consists of distributing the Client-side and the Server-side into two separate projects. As far as this dissertation is concerned, the Client-side will be explained and focused.

1.1. Context

This project has been conducted at the Mountain Research Center (CIMO)[1], a multidisciplinary research unit focused on Mediterranean mountain issues, based at the Polytechnic Institute of Bragança (IPB), Portugal. within a research grant from the Project NORTE-01-0145-FEDER-023289: DeCodE "Bring natural colorants and preservers into a real alternative to artificial additives through a strategy of open data and experience based research", presented on Appendix Research Grant. CIMO follows an approach based on an applied research strategy covering issues and expertise that goes from Nature to Products, but this project was a pioneering initiative on the Information Technology (IT) field.

[1] `http://www.cimo.ipb.pt`

This research work is a small fraction of a large mobilizing project named **Valor Natural**, which has several copromoters from academy and agrofood industry.

Further detailed information about it can be found on the project's website[2]. The current research work is described in particular within the following chapters 3 and 4.

The project also had a scientific contribution, a publication in the "XIV Encontro de Química dos Alimentos", entitled *Descodificar os "E": plataforma online de acesso aberto de aditivos alimentares* [7].

1.2. Objectives

This research work intended to study and develop a solution that allows a quick search of the most relevant information of each food additive, as well as its conditions of use in specific categories of foods. This solution needed to incorporate two access methods, web and mobile applications.

Initially, an analysis of the current open access systems was made. The main features and characteristics of each solution were identified to the purpose of comparing them, and thus obtain an overview of the current state of the art.

Next, the system use cases were explored, and a survey of the requirements to be supported, both functional and non-functional. Thus, to support these demands, an architecture was developed and allowed to speed up the implementation of the solution.

Lastly, the most appropriate technologies were chosen according to the desired functionalities and system implementation, thus, proceeding to the development of both web and mobile applications.

1.3. Outline

The remainder of this dissertation is organized in five more chapters, which are succinctly defined beneath.

[2] `http://www.cimo.ipb.pt/valornatural2`

- **Chapter 2** represents an overview of the theory of additives and needs behind this project. In this Chapter, some existing solutions and different approaches are reviewed and studied. Additionally, a brief analysis of mobile application development is given, alongside some theory on Web Services, to better understand their purpose.

- The following chapter, **Chapter 3**, analyses the significant challenges of the systems conception. It details all the essential requirements, which results in functional and non-functional requirements, and a global architecture that comprehends these needs. Moreover, RESTful Web Services are explained, and finally, some iOS application development characteristics are presented.

- **Chapter 4** the final solution is presented. The technology used throughout the implementation and concluding with the mobile app interface, parsing page by page with its corresponding features and app functionality.

- At last, **Chapter 5**, presents the conclusions obtained in the realization of this project and introduces some details to be acknowledged in future work to enhance the solution.

Chapter 2

Background

This chapter provides an overview of the main concepts that sustain this research work, and it is focused on the main concepts and technologies. Section 2.1, describes the basic concepts of food additives, presenting their purpose, definition and the existing classes and categories of food additives. The purpose of Section 2.2 is to review the current state of the art of the existing market applications, providing an analysis of their advantages and disadvantages.

Section 2.3 describes the basic concepts and technologies essential for the development of mobile applications.

Finally, Section 2.4 introduces Web Services and their purpose, additionally the two most used approaches are explained.

2.1. Food Additives

Nowadays consumers in modern countries expect a wide variety of foodstuffs to be available throughout the year. The shelves on supermarkets are stocked with fresh fruit, vegetables, meats and fish from many different countries, prepared meals for reheating, preserved meat, and many others. All of these products have to endure the journey from

harvest to store and, once acquired, they must endure on peoples shelf for their convenience. Food additives are vital to enable the food industry in all its forms to make foods that meet these increasingly challenging demands [8]. The two largest food additive regulators worldwide are European Food Safety Authority (EFSA) and Food and Drug Administration (FDA) of the United States of America (USA).

Food additives have a core definition that is unanimous among authors and authorities, which is the legal definition adopted by the European Union (EU), stated by EFSA [9], [10]. For better understanding, this legal definition can be adapted to:

> "Food additives are substances added intentionally to foodstuffs to perform certain technological functions, for example to colour, to sweeten or to help preserve foods."

This definition can be found on the legislation [11] and it can be resumed to any substance added to food to perform a particular technological function. Food additives are used either to facilitate or to complement a wide variety of production processes in the modern food supply [12]. Some food additives and their uses can be seen on Table 2.1.

The Codex Alimentarius, or "Food Code" is a collection of standards, guidelines and codes of practice adopted by the Codex Alimentarius Commission. The Commission, is the central part of the joint Food and Agriculture Organization of the United Nations (FAO) and World Health Organization (WHO) Food Standards Programme and was established to protect consumer health and promote fair practices in food trade at an international level [13].

In the EU, all food additives, whether approved or not, are preceded by the letter "E" (representing Europe) and a distinct number [2]. This codification was extended to the Codex Alimentarius Commission to easily identify food additives around the world, regardless of whether they are approved for use or not. A huge effort was set in motion to gather information toward the creation of a single database of legal additives to be used within the EU, and in the Regulation 1129, of 2011, all the approved additives, as well as their Acceptable Daily Intake (ADI), are listed [14]. The ADI is the starting point

Additives	Function
Colors	Improves food appeal. Replace lost colors.
Preservatives	Increase shelf life. Retard bacterial growth.
Antioxidants	Prevent spoiling. Maintain color and taste.
Emulsifiers and Stabilizers	Binding. Prevent separation of ingredients.
Thickeners	Improve the texture of food. Assist water being mixed in.
Flavor enhancers	Boost product flavors.
Sweeteners	Increase sweetness.
Flavoring	Add extra flavor. Replace loss of flavor in processed food.

Table 2.1: Food additives and their functions

to establish the maximum amount of a certain additive to be included in each foodstuff, which can differ from a few milligrams to *quantum satis*, and expressed as milligrams of additive per body weight in kilograms, per day. *Quantum satis* is a Latin term employed by the EFSA which determines that there is "no maximum numerical level specified and substances shall be used in accordance with good manufacturing practice, at a level not higher than necessary to achieve the intended purpose" [11]. Under EU legislation, food additives must be authorised before they can be used in foods.

2.1.1 Controversy Around Food Additives

The first half of the twentieth century saw a significant increase in the use of food additives as the adoption of processed foods raised. Initially, the use of food additives was not controlled, and a number of these were subsequently shown to pose safety concerns [9].

Some consumers regard food additives as unsafe, especially due to scandals and scares that took place in the nineties and two thousands [15].

Some examples that lead to consumer mistrust are:

- It is widely accepted that the excess of intake of nitrite is dangerous and has deleterious effects on human health [16].
- Surveys in Belgium and New Zealand have reported danger to individuals who consume high quantities of sulfited wine [17].
- Some authors consider that the consumption of lollipops with brilliant and patent blue colorants by children is dangerous due to their absorption through the lingual mucosa [18].
- The compound E104, Quinoline yellow, has been reported to cause urticaria, asthma, rashes, and hyperactivity [19].

Nevertheless, like any other aspect of society, the evolution of science brings new data to the discussion table. Thus, the development of the mobile application detailed in this work will contribute to the enlightenment of people regarding the additives they consume in food.

2.1.2 Origin of the Additives

Although there is no clear distinction between synthetic, artificial or natural additives from the two biggest regulating bodies, namely EFSA for the European Union, and the FDA for the United States of America the need for this discrimination is quite useful for consumers, relying on the fact that they prefer natural additives over artificial ones [2], [20].

Furthermore, food additives are rigorously tested and periodically re-assessed by EFSA. This re-evaluation brings an opportunity to increase research toward the use of natural

additives, while clarifying doubts that still might persist in food producing companies, the scientific community, regulating organizations and consumers.

Additionally, the developed database has a field that provides the consumer with information regarding the provenance of food additives, which will allow them to search for a natural counterpart if they are before an artificial or synthetic one.

2.1.3 Classification of Food Additives

Food additives are classified in groups and subgroups, or, as defined in this project, classes [2]. This classification is not unanimous among the regulation authorities around the world. Within the EU, EFSA provides a classification of 26 functional classes of additives: sweeteners, colorants, preservatives, antioxidants, carriers, acids, acidity regulators, anticaking agents, antifoaming agents, bulking agents, emulsifiers, emulsifying salts, firming agents, flavor enhancers, foaming agents, gelling agents, glazing agents, humectants, modified starches, packaging gases, propellants, raising agents, sequestrants, stabilizers, thickeners, and flour treatment agents [14].

The FDA approach, on the other hand, narrows down the number of additives allowed in the USA to 18 groups[1] : preservatives, sweeteners, color additives, flavors and spices, flavor enhancers, fat replacers (and components of formulations used to replace fats), nutrients, emulsifiers, stabilizers and thickeners, binders, texturizers, pH control agents and acidulants, leavening agents, anti-caking agents, humectants, yeast nutrients, dough strengtheners and conditioners, firming agents, enzyme preparations, gases [2].

This challenge to unify a standard classification on legislation between authorities and the different additives that are permitted or not was an inspiration to create this database, to have an integrated definition and information. The classification is performed depending on each additive function in food, the approach adopted on this solution consists dividing them into six main classes with a brief explanation to each one below (Figure 2.1).

[1] `http://www.fda.gov/overview-food-ingredients-additives-colors`

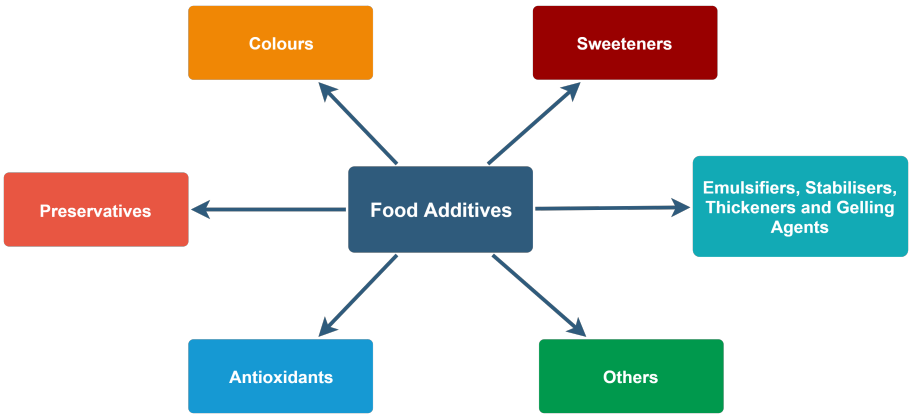


Figure 2.1: Additive Classes

- **Colours**

  Colour additives may be added to food, drugs, cosmetics and certain medical devices to confer colour [21]. Food colour additives are used to enhance the visual attributes of foods. Their application is particularly controversial, partly because the colour is perceived by some as a means of deceiving the consumer about the nature of the food, but also because some of the most brightly coloured products are those directed at children. As with all additives, their use is strictly controlled and authorised only where a case of need is proven [8].

- **Preservatives**

  Preservatives are probably the single most essential class of additives, as they perform an important purpose in the safety of the food supply [22]. They are used alone or in conjunction with other methods of food preservation. Most importantly, they delay bacterial degradation, which can lead to the production of toxins and cause food poisoning. Thus they contribute to a clear consumer benefit in keeping food safe over the shelf-life of the product, which itself may be extended by their use and thus meet the demands of modern lifestyles [8].

- **Antioxidants**

  Antioxidants are substances that reduce the oxidative deterioration that leads to

rancidity, loss of flavour, colour and nutritive value of foodstuffs. Thus, antioxidants are used to extend shelf-life [23]. The rate of deterioration can differ considerably and it is influenced by the presence of natural antioxidants and other components, the amount of oxygen available, and sensitivity of the substance to oxidation, light and temperature, for instance. Antioxidants cannot restore oxidised food, they can only delay the oxidation process. As oxidation is a chain reaction process, it needs to be retarded as early as possible [8].

- **Emulsifiers, Stabilisers, Thickeners and Gelling Agents**

  The purpose of emulsifiers is to facilitate the production or maintenance of homogeneous mixing of ingredients that normally would not mix, namely fat and water. This mixing of the aqueous and lipid phases is then sustained by stabilisers. These additives are vital in the production of mayonnaise, chocolate products and fat spreads, for example [8]. In addition to this function, the term stabiliser is additionally used for substances that can stabilise, retain or enhance an existing colour of foodstuff and substances that increase the binding capacity of the food, allowing the binding of food pieces into reconstituted food [24].

  Thickeners are substances which increase the viscosity of a foodstuff, while gelling agents are substances which confers texture through a gel formation [25].

- **Sweeteners**

  Sweeteners, as their name implies, perform, undoubtedly, an obvious function, to impart a sweet taste in foodstuffs and as table-top sweeteners [26].

- **Others**

  Colours and sweeteners are very explicit, well-defined classes of additives and, because of the nature of their function, are subject to specific legislation. All other classes of additive remain under the overall topic of "miscellaneous" [8].

Besides these classes, there is a food category system, a tool for assigning food additive uses. This system applies to all foodstuffs and it is used to simplify the reporting of food

additive uses for assembling and constructing [27]. The food category system can be seen in detail on Appendix Food Category System, it is based on eighteen main food categories that are further divided into subgroups up to a maximum of four levels.

2.2. Case Studies

Food additives are the target of countless myths, leading to mistakes in choosing the products to use in everyday life. There is a lack of resources that inform consumers about their characteristics.

During this work, numerous sources of information were studied, composing a general overview of the available market. The purpose of this study was to analyse flaws, lack of information or sometimes incorrect information, the complexity of use, and above all, to analyse their strengths. This screening helped shape the development of the database explained in this work.

Although the solution to be developed is a mobile application, the research method was divided into two distinct platforms, web and mobile.

2.2.1 Web Applications

For the development of the project, several sites were analysed, not only for the information they contained but also for their design and appearance. However, only four web applications were structured enough to allow a deep understanding and further analysis, which were:

- `http://www.aditivos-alimentarios.com`
- `https://webgate.ec.europa.eu/foods_system`
- `http://www.fao.org`
- `http://www.efsa.europa.eu`

All of the websites listed above are food additives databases. The lack of user experience in all of them was notorious, except for "Aditivos Alimentarios". The design and usability of the Spanish website were appropriate to the needs of the client. It was possible to search for an additive number, additive name and additive toxicity level, as demonstrated on Figure 2.2. It also had some interesting features, such as leave a comment on each additive, see the latest ten commentaries and the five most searched additives. Besides that, there was a lack of information and the data was not certified or from a reliable source.

Figure 2.2: Website Aditivos Alimentários

The second website, the "Food Additives Database", has a lot of information, such as ADI value, several links for the user to navigate from the additive to other additives of the same class and the additive applications. However, despite the reliability of the information, since its source is EFSA, it is difficult to access such data and understand what it means.

The third one is the main website for FAO which has updated news and articles about food security. Apart from that, it has a food additive index, that contains an index of individual food additives. Clicking on a food additive opens its specifications. These

specifications contain the food additive description, sources, synonyms and functional uses, yet, to access this data, the user needs to always open a PDF file. This process is stressful and exhausting if the user wants to search for several additives, therefore, it is not recommended.

Finally, the most important one, EFSA's website, has legal and uniformed information that will be used to populate the database developed for the solution. More specifically from the Commission Regulation (EU) No 1129/2011. This document contains an extensive table that aggregates all legal additives. This is also the document that lays down all legislation of food additives within the EU, and that is widely used in other impoverished and developing countries, due to lack of funds to conduct the analysis themselves. These additives are presented, first by order of their E-number and then in a table with all authorised food additives and conditions of use in food categories. A web crawling was made on that table to retrieve the category number, the E-number, name, ADI, footnotes, restrictions and exceptions corresponding to each additive, further information about these last three properties will be explained following the dissertation.

2.2.2 Mobile Applications

A search was performed in both Android and Apple app stores to gather as much as possible on mobile gestures and design layout of the application. Since the data was already aggregated, the mobile study was purely an initiation to the solution development. The following were the most important apps analysed:

- Food Additives
- Food Additives Checker
- E-Codes Free: Food Additives

All of the apps presented above show a similar approach. They embedded an extensive list of additives and, by selecting them, the respective detailed information opens, such

as origin, restrictions and side effects. The information presented by these applications is not verified, so they can contain some unreliable data.

The "Food Additives" app, is only available on the Google Play Store and it is the most complete app from those in the study. The features of this app include the possibility to mark the additives as favourites so that the user can access them whenever he wants. It is also possible to add commentaries to each additive, for the community to interact. It has the level of toxicity divided in colours, so it is easier to identify, although, the way the information is displayed is not very user-friendly, mainly due to the additive description being too extensive and grouped, causing some reading difficulty.

The "Food Additives Checker" is only available on App Store. It has a comprehensive list of additives that can be sorted by alphabet, E-number, category and danger level and a much more appealing layout, comparing with the other two. But, on the additive list, the colour of the additive is according to the level of toxicity, which makes the application a bit confusing, as seen on Figure 2.3, thus, the additive detail page is very well designed. The search can only be done by E-number or food additive name and it has seven different languages.

"E-Codes Free: Food Additives" is the app with fewer features, it has a simple and old fashioned layout that, sometimes, becomes confusing, being that the application is from 2013. Besides the basic features of listing additives and see their details it allows the user to rate each additive.

After this market review, it was possible to create a general concept of what is available to the public for both web and mobile platforms. It has been found that no mobile applications are containing reliable information available, although some are well done, the information may not be correct. The design of existing applications is also a bit confusing, consequently, making difficult the understanding of the food additive in focus, which should be simple and easy.

On Chapter 4 it is possible to see some features or design ideas that were obtained from this study, as there is no platform to fully meet the objectives intended, the solution build needed to be capable of filling those gaps.

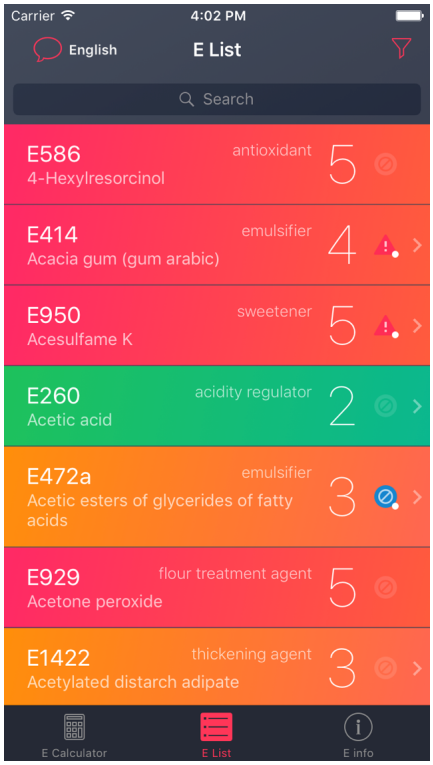


Figure 2.3: Mobile Application Food Additives Checker

2.3. Mobile Application Development

Mobile applications have become an important part in the life of the modern man. The involved devices have become indispensable companions in our lives. Both mobile devices and applications provide significant advantages to their users, in terms of portability, location awareness, and accessibility. Besides this advantages, the use of mobile apps for health and well being promotion has grown exponentially in recent years [28].

There are two predominate operating systems in today's mobile market: Google's Android and Apple's iOS. According to an 18 Jun 2019 International Data Corporation (IDC) industry report[2], Android continues to dominate the worldwide smartphone market, grabbing 86.7% of the share, leaving the remaining 13.3 % to the iOS market. The remaining of this section comprehends a brief comparison between the two major operating

[2]https://www.idc.com/promo/smartphone-market-share/os?fbclid=IwAR1BzYyQyAwWvbu35zu0KsQ5K8xu25nfDZe6G7H2531Kx5vkwYfB6ZanxKw

systems and the types of existing mobile applications.

2.3.1 Mobile Architecture

Central to every development platform is its Software Development Kit (SDK), which enables third party developers to deliver applications operating on their specific platform. This kit includes, among other things, libraries, debuggers and handset emulators. Different approaches are taken by the existing platforms when it comes to sharing their SDK with developers. Some have chosen to disclosure the entire SDK source code and OS, whereas others have chosen to restrict the access as much as possible [29]. In the general case, mobile platform technologies can be viewed as a set of hierarchical layers. Higher levels contain sophisticated services, and lower layers contain the necessary technologies on which the apps rely. Android and iOS platforms contain four well defined layers [30], as seen in Figure 2.4.

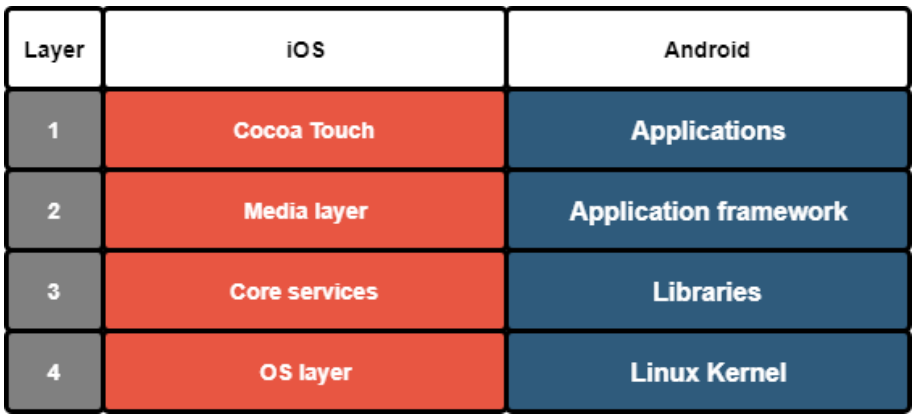


Figure 2.4: iOS and Android Architecture

This layered architecture represents the level of abstraction, the amount of complexity by which a system is viewed or programmed, with the higher level layers more abstracted and the lower level layers more fundamental, closely tied to the hardware. It goes without saying that the higher level layers rely on the lower level layers for some of their functionality.

Regarding iOS architecture, the Cocoa Touch layer is the topmost layer for the iOS

architecture. It contains some of the critical frameworks native iOS applications rely on, with the most prominent being the **UIKIT** framework[3].

The **Cocoa Touch** layer defines the necessary application infrastructure and provides several vital technologies, such as multitasking and touch-based input. As I mentioned, iOS applications rely heavily on the UIKit framework. Native iOS applications cannot operate if they are not linked against the UIKit and the **Foundation** frameworks[4]. The UIKit framework, or UIKit for short, is designed to the iOS platform. UIKit provides the infrastructure for graphical, event-driven iOS applications. The core aspects that are specific to iOS are also taken care of by it, such as multitasking, push notifications, and accessibility. The Cocoa Touch layer provides developers with a large number of high level features, such as Auto Layout, printing, gesture recognizers, and document support. In addition to UIKit, it contains the Map Kit, Event Kit, and Message UI frameworks, among others.

The **Media Layer** provides all multimedia services that are used within the iPhone and other iOS devices, handling graphics, audio, and video. The Media layer contains numerous frameworks including the Assets Library framework to access the photos and videos stored on the device, the Core Image framework for image manipulation through filters, and the Core Graphics framework for 2D drawing.

The **Core Services layer** is in charge of managing the essential system services that native iOS applications use. The Cocoa Touch layer depends on this layer for some of its functionality. The Core Services layer also provides a number of indispensable features, such as block objects, Grand Central Dispatch[5], In-App Purchase, and iCloud Storage. One of the most welcomed additions to the Core Services layer is ARC or Automatic Reference Counting. ARC is a compiler-level feature, introduced in 2011 alongside the release of iOS 5, that simplifies the process of memory management.

Most of the functionality provided by the three higher level layers are built on the **Core OS layer** and its low level features. It also interacts directly with the hardware.

[3] `https://developer.apple.com/documentation/uikit`
[4] `https://developer.apple.com/documentation/foundation`
[5] `https://developer.apple.com/documentation/DISPATCH`

Vendor	Operating Systems	Programming Language	Development Environment	Application Store
Apple	iOS	Objective-C / Swift	Xcode	App Store
Google	Android	Java / Kotlin	Eclipse / Android Studio / InteliJ IDEA	Google Play

Table 2.2: Comparison of mobile platforms

The Core OS layer provides a handful of frameworks that an application can use directly, such as Security frameworks.

2.3.2 Operating Systems

A mobile platform (or operating system) consists of the software responsible for controlling and supporting a mobile device. Following, a summary of the two major native mobile platforms, iOS, and Android, is presented. The mobile app deployment is made through special distribution app stores, with or without fees. Additionally, Table 2.2 provides a comparison among those platforms [31].

- **iOS**

  Developed by Apple, iOS, formerly known as iPhone OS [32], is a mobile operating system derived from Mac OS X. iOS was launched specifically for the iPhone in 2007, but in 2010, with the growing success of the iPhone, Apple chose to rename and apply the same operating system in its other device offerings, including the iPad, iPod Touch and Apple TV [33]. Its release revolutionized the mobile devices market since it provided user experience and applications with unique and high quality [34].

  iOS applications are written in Objective-C or Swift and can be downloaded in the App Store, Apple's application market. Further, some Swift characteristics are detailed on Chapter 4.

  To publish an app on Apple's App Store, the developer needs a developer account,

which costs US$ 99. Apple also has a more extensive process for reviewing apps before they are released on the store, an effort to determine whether they are reliable, perform as expected, and are free of offensive material[6].

- **Android**

  Android is a mobile operating system developed by Google. It is based on a modified version of the Linux kernel and other open-source software and is designed originally for touchscreen mobile devices such as smartphones and tablets [32]. It was initially created by the company Android Inc., which in 2005 was purchased by Google. In September 2008 was released the first Android phone and since then, this operating system suffered several updates with new versions and support for various mobile devices from different vendors [35].

  Android applications are written mostly in Java, a language adopted by many developers around the world, a fact that contributes to its large community of developers. Furthermore, Android's application market is named Google Play, where users can download Android apps.

  To publish an app on Google Play Store, it is required to have an additional developer account, US$25 is the registration fee to complete the process.

2.3.3 Types of Mobile Applications

These days, there is strong growth in new software development tools and applications for smart mobile devices. The theory *write once*, *run anywhere* cannot be employed when developing native apps, the most suitable choice for developers or corporations is the cross-platform mobile development. Cross-platform development simplifies the maintenance, and deployment processes and saves development time and effort.

The sudden growth in the mobile market motivated the implementation of cross-platform software development environments that could produce the development more

[6] `https://www.developer.apple.com`

accessible and more efficient. The vital purpose of cross-platform mobile app development is to achieve native app performance and run on as many platforms as possible. The main categories of applications provided by these software environments are native, web, and hybrid [30].

- **Native Mobile Application**

  For the development of a Native app, it is required to use an Integrated Development Environment (IDE) that provides the necessary development tools for building and debugging the applications. Native apps require a high level of experience and technological know-how than other types of applications, therefore, are more challenging to develop, and it is designed to run in a specific operating system of a mobile service [31]. For instance, to develop this solution, an iPhone native application, it was mandatory the usage of Apples IDE (Xcode), and it runs only on his specific platform, iOS.

  The principal advantage of native apps is high performance and ensuring a more deep user experience as developers use native device UI. Source code is efficient, with fast performance, consistent look and feel and full access to the underlying platform hardware and data [30].

  Although there are some cons to native apps, as they are higher cost compared to other types of apps, this is due to the need of creating the same app for other platforms, with different programming languages, separate support, and maintenance for different types of apps resulting in the higher product price.

- **Web Mobile Application**

  A mobile web application consists of a typical Internet application that resides on a server and could be accessible through the Internet. Every time the application is executed, it is downloaded and processed locally. Furthermore, it is an application designed to fit the screens of most mobile devices and written as web pages using languages supported by browsers, like JavaScript, CSS, and HTML5. For this

reason, this kind of applications can be accessed from any device that has Internet access and a compatible browser [31].

One of the significant disadvantages of web applications compared to the native is related to the more inferior user experience they provide. Because access to physical resources of the device, such as buttons, GPS or camera, is limited, the performance, responsiveness and even the look and feel offered are lower than those offered by native applications [36].

- **Hybrid Mobile Application**

  Hybrid apps come from the fusion between native and web applications, they are built with web technologies and then wrapped in a platform-specific shell that allows them to be installed like native ones. They are built using multi-platform web technologies such as Xamarin, React Native, Ionic, and Angular.

  A clear advantage is the time taken and the difficulty in developing the application, also, a single base code for all platforms grants low cost maintenance and smooth updates. Even so, hybrid applications lack in performance, speed, and overall optimization in comparison to native apps, for instance. Also, there are specific design issues due to app inability to look in the same way on two or more platforms [37].

  HTML5 improves video and media support, animations and graphics, provides geolocation and allows offline operation through the use of the cache. This way, HTML5 allows more proximity between the two types of applications described above, through the creation of hybrid applications that aim to combine the best of both worlds. Nevertheless, the specification of HTML5 is still in progress, which implies that this language is not entirely defined and still lacks the definition of standards to be universally adopted by browsers. Nevertheless, the use of hybrid applications is seen as an area with future, existing several tools and frameworks that use these technologies [36].

Figure 2.5[7] summarizes the explanation provided above about the three types of mobile Apps.

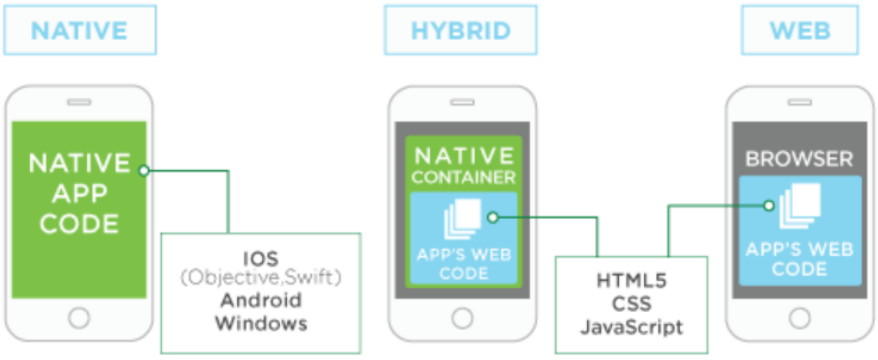


Figure 2.5: Mobile Apps At a Glance.

2.4. Web Services

Web services make software application resources available over networks using standard technologies. Because web services are based on standard interfaces, they can communicate even if they are running on different operating systems and are written in different languages. For this reason, they are an excellent approach for building distributed applications that must incorporate heterogeneous systems over a network[8].

The World Wide Web Consortium (W3C)[9] is an international organization aimed to design and develop Web standards, guidelines and protocols that support the World Wide Web as a secure communication tool used to provide services and create content. W3C defines a Web service as a software system designed to support interoperable machine-to-machine interaction over a network[10].

The following topics outline the most common approaches to implement Web services: SOAP and REST.

[7] https://medium.com/appzio/for-dummies-web-hybrid-and-native-explained-e6b1a8194fd8
[8] https://docs.oracle.com/cd/E13224_01/wlw/docs102/guide/webservices/conBasicWebServiceTechnologies.html
[9] https://www.w3.org
[10] https://www.w3.org/TR/wsdl.html

2.4.1 SOAP

SOAP originally held for "Simple Object Access Protocol" when it was a part of the original specification for SOAP version 1.1. With the current version of 1.2, it is just called SOAP. According to the W3C[11], SOAP is defined as "a lightweight protocol intended for exchanging structured information in a decentralized, distributed environment. It uses Extensible Markup Language (XML) technologies to define an extensible messaging framework providing a message construct that can be exchanged over a variety of underlying protocols. The framework has been designed to be independent of any particular programming model and other implementation specific semantics".

Therefore SOAP is a protocol specification for exchanging structured XML messages over Internet based protocols. As illustrated in Figure 2.6, a SOAP message consists of three general elements, a vital, the Envelope, an optional, the Header and a mandatory, the Body. SOAP envelope, encapsulates these messages in another XML structure, which contain metadata, SOAP headers, to process the SOAP message.

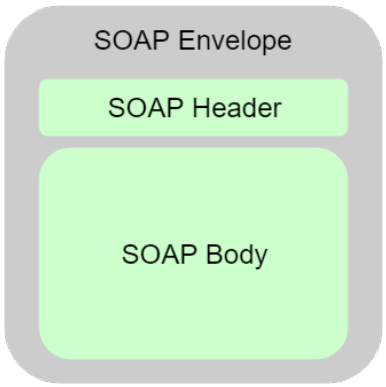


Figure 2.6: SOAP Message Construct

Due to the nature of its encapsulated messages, service consumers and providers can exchange invocation requests and responses through heterogeneous systems and different transport protocols. The metadata contained in the SOAP headers provide addressing regardless of the transport protocol used, this means, that transport protocols are only

[11] https://www.w3.org/TR/soap12-part1

tunneling protocols to carry out communication, ignoring any semantic element that the protocol could provide to use the service interaction [38].

2.4.2 REST

REST is an acronym for REpresentational State Transfer, initially introduced by Roy T. Fielding in his 2000 Doctoral Dissertation [39]. Fielding's Dissertation covers the core research for understanding modern network-based software through architectural styles. His research is significant in today's network-based systems and architectures [40].

REST is a set of architectural constraints used to implement RESTful services. A service conforming to the REST constraints is often referred to as being "RESTful". These constraints can be summarized in the following points [38] [41]:

- **Client-Server:** this feature is commonly found in Web apps. A server with a set of services available listens for requests to those services. A client, who wants to run a specific service in an available server, sends a request to the server. The server behavior is either reject or execute the requested service and return a response to the client;

- **Stateless:** this constraint is also imposed by the REST style, concerning the interaction between client and server. Communication must be done without storing any state on the server,i.e., every request from a client to the server must hold all information necessary for it to be understood. Therefore, session states, when needed, must be fully supported on the client;

- **Cache:** By using cache, it's possible to reduce the impact of the downside brought by performance reduction. It also requires that data from a response, coming from a request to the server, to be marked as cacheable or noncacheable If an answer is set as cacheable, then it will be reused in response to future equivalent requests;

- **Interface/Uniform contract:** REST architectural style is distinguished from

other network-based styles on this central feature, it emphasis on a uniform interface between components,i.e., client and server. To obtain a uniform interface, REST defines four interface requirements:

1. Identification of resources;
2. Manipulation of resources through representations;
3. Self-descriptive messages;
4. Hypermedia as the application state mechanism.

- **Layered system:** a layering feature as added to REST style, to improve the scalability requirement of the Internet. Multilayer systems use layers to separate different units through their responsibilities. The main disadvantage of this model is the addition of overhead and latency in the processed data, reducing performance. On a network-based system that supports caching, this drawback can be mitigated;

- **Code-on-demand:** this item, the last in the set proposed by the REST style, is an optional feature. REST allows clients to have the ability to download and execute code on the client side directly. With it, it simplifies the client-side and focuses on extensability, in contrast, reduces visibility. A known practical example of Code-On-Demand is Adobe Flash: A user (client) requests a Web page which contains a link to an SWF (Adobe Flash file format) using a web browser. After the request, the Web page is transported to the client machine together with an SWF and executed.

Services designed through the RESTful approach expose their functionality as Web resources, where each resource is addressed with a unique URL: a user can directly access a specific resource by the associated URL or traverse the offered functionality through a hierarchical structure [42].

2.4.3 HTTP Methods

The HTTP protocol has 9 different methods. Only 6 of them are widely used: GET, POST, PUT, DELETE, HEAD, and OPTIONS. Although HEAD and OPTIONS are special methods, HEAD is used to return only the headers of the response and OPTION is for getting allowed methods on a resource [43]. The others are used for operating with resources.

There is only one method that is *idempotent*, POST. An *idempotent* HTTP method is an HTTP method that can be called many times without different outcomes. It does not matter how many times the method is called, the result is expected be the same. Table 2.3 summarizes a brief definition of the methods.

Verb	**Action**
GET	Used to retrieve a representation of a resource. It is a read-only, idempotent, and safe operation.
PUT	Used to update a reference to a resource on the server and it is idempotent as well.
POST	Used to create a resource on the server based on the data included in the body request. It is the only nonidempotent and unsafe operation of HTTP.
DELETE	Used to remove a resource on the server. It is idempotent as well.
HEAD	Like GET but returning only a response code and the header associated with the request
OPTIONS	Used to request information about the communication options of the addressed resource (e.g., security capabilities such as CORS[12])

Table 2.3: HTTP Common Methods [44]

Due to the requirements of the solution (see Section 3.1), REST has characteristics that give several advantages over SOAP in the context of the development of this solution, thus matching technological needs. It [38]:

[12]Cross-Origin Resource Sharing (CORS) is a mechanism that uses additional HTTP headers to tell browsers to give a web application running at one origin, access to selected resources from a different origin [45].

- Requires little infrastructure support apart from HTTP and XML which are supported by most programming languages, operating systems, servers;
- Proven scalability, simplicity, and low performance overhead;
- The simplicity derived from a uniform interface avoids burdening APIs, thus facilitating the maintenance and growth of services. REST applications run in the World Wide Web using the HTTP protocol to transfer data. It is not necessary to implement an additional layer over the ISO/OSI stack [46]. Furthermore, REST does not require any toolkit to be installed on client machines.
- REST defines a uniform interface for any resource, and it does not require WSDL to define it (although some human-readable documentation is still needed) and only requires knowing the resource structure and semantics.

Chapter 3

Architecture

On this chapter, specifications that support the systems development are discussed. These requirements can be divided into two groups: functional and non-functional. Section 3.1 describes both of those groups in depth. Additionally, a systems use case is analysed on Subsection 3.1.3, where the target public is established.

Section 3.2 details the system architecture and its main components, without considering the technologies and tools used for the implementation - those are described in Chapter 5.

The class diagram used on this solution is presented on Section 3.3, alongside an explanation on RESTful Web Services on Section 3.4, regarding information such as defining the URL design used on this solution.

Lastly, Section 3.5 presents the architectural pattern used on the mobile application, and some very important aspects when programming an iOS software.

3.1. Requirements Analysis

This section aims to identify the needs of this project, dividing them into functional and non-functional requirements. This study focuses on the tasks that determine the demands

or conditions to meet the project.

Behavioral requirements are specifications of user interactions with a system, they describe all the cases where the system uses the functional requirements. These are captured in use cases, that can be seen in Subsection 3.1.3. Functional requirements are supported by non-functional requirements, which impose constraints on the design or implementation, such as performance requirements, security, or reliability. Generally, functional requirements are expressed in the form "system must do <requirement>", while non-functional requirements take the form "system shall be <requirement>" [47].

Functional requirements may include calculations, technical details, data manipulation and processing, and other specific functionality that define what a system is assumed to perform [48].

Furthermore, this analysis was made for the complete system, that covers both Web and Mobile applications. Some of the requirements outlined beneath are only a demand for the Web application.

3.1.1 Functional Requirements

Functional requirements should incorporate functions performed by specific screens, outlines of work flows performed by the system, and other business or compliance requirements the system must meet [49]. Moreover, CIMO defined these requirements when the proposal was elaborated, and they have been further studied and improved throughout the early stages of development.

- **Accounts with different access levels**

  Only on the Web app the user can create an account, although, both Web and Mobile apps have login implemented. Those accounts have the possibility of providing different content in the future, depending on the target audience.

- **Complete Search Engine**

The search engine should be very complete, allowing searching for Additive name or E-number, additive Class name, Food Category name and any word that is part of the details, whether additives, classes or food categories. All of this, regardless of the language in which the research is performed.

- **Additive, Additive Classes & Food Categories Listing**

  The fundamental objective of this project is to be able to list all Additives, Additive Classes, and Food Categories. On the Additives list, displayed their E-number, name, the class that they belong, their origin, and evaluation. The Additive Classes display their name, while Food Categories present their number and name.

- **Access Additive Details**

  The user should be able, in all Additives, to access additional information besides the one shown on the additives listing, such as the group that it belongs, their typical applications and some observations.

- **Quick Search of a Specific Additive in Scientific Databases**

  In the Additive page, there should be possible to use direct links to scientific databases with the current additive already searched.

- **Access to Conditions of Use of a Specific Additive in Food Categories**

  Also on the Additive page, the user should be able to access the conditions of use of the current additive on food categories, if they exist namely, the ADI value, some details, and their restrictions or exceptions.

- **Access Authorized Additives for a Specific Food Category**

  On the Food Category page, it should be possible to access the inverse feature stated above, i.e., see which additives are authorized for that Food Category, with the same information, ADI value, some details, and their restrictions or exceptions.

- **Filter Additives by Classes**

On the Additive listing, it should be possible to filter that same list by Class of Additives.

- **Link Related Information**

  The web app navigation is essential for the users to find what they intend quickly. Therefore, all related data is connected to access it easily.

- **Internationalization**

  The website should be fully internationalized, supporting, at least the languages previously chosen: Portuguese, English, and Spanish. These three languages, as mentioned earlier, have been taken into account when developing the search engine.

- **Access to Scientific Publications**

  On the website, it should be possible to easily access to papers or other scientific publications regarding each food additive, through notable scientific databases.

- **Administration Area**

  The system should have a dedicated area where administrators can manage all existing data such as additives, additive classes, food categories and also registered users. This area should be available exclusively on the website.

3.1.2 Non-Functional

Non-functional requirements aggregate all solution requirements that do not match the features. In this subsection, particular attention is paid to quality requirements, as some of them are particularly important to the success of the project.

- **Performance**

  This is particularly important as a user can get thousands of data in one query, so the system needs to ensure proper data access performance.

- **Reusability**

  Given the structure of the information to display, it is vital to be able to reuse components such as lists or tables. This ensures consistency in data presentation and greater reuse of code.

- **Robustness**

  The solution needs to be robust in overall, i.e. when the crawling is performed, if, for some reason unrelated to the system one additive can't be loaded, the crawling continues, and only that additive fails. Also, when searching on the database, the system can't fail. Furthermore, some information is only available in the default language, and, if that is the case, even tho the search is performed in another language, the default information must be displayed, so that the user does not receive incomplete information.

- **Usability**

  As described above, it is very important to acquire the public interest, so that the solution becomes their election application. Thus, a perfect interaction with both apps is vital and it is important that the user experience is the most simple as possible.

- **Service-Oriented Architecture**

  The system must be easily used in multiple platforms, as a service provider. This can be simplified by following a service-oriented architecture, based on REST web services [50].

- **Integrability**

  The solution needs to be constituted by a Web and Mobile applications, thus the system must be developed to easily integrate them on the same ecosystem with no problems.

- **Documentation**

Establishment of documentation to support the use of services and tools developed to help future work.

- **Readability**

  As mentioned earlier, the goal is to list a lot of information, this is a clear objective from the outset, an easy and quick reading of the information presented so as not to confuse users.

3.1.3 Use Cases

As mentioned above, use cases are effective techniques to express the functional requirements of a system straightforwardly and efficiently to learn. Use cases are mainly composed of natural language sentences that represent the system behaviour. This process of defining a system's response is always a critical point, due to the inherent ambiguities originating from the different possible interpretations of those sentences [51].

To fully understand the use cases of the system, it is necessary to understand the intended audience of the system. The purpose of the solution is to provide detailed and reliable information about food additives to the general public but may also serve as a reference for researchers or companies wishing to work with additives. Therefore, the system is intended to serve three distinct entities: Consumers, Researchers, and Industry. This solution will provide to these entities the following possibilities:

- **Consumers**

  Who wish to consult immediate information on a particular additive, such as its function, natural alternatives and potential risks;

- **Researchers**

  Who wish to access relevant information such as maximum authorised doses or related scientific articles;

- **Industry**

Who is looking for a comprehensive database of food categories and restrictions on permitted additives.

Although there is this difference between entities that will make use of the system, the three groups behave similarly from a functional point of view, since the information will be all freely accessible.

With this point of view, a single actor, Customer, will be considered and composed of those three entities, Consumers, Researchers and Industry. Besides, it is essential to have system management and administration processes, which will be the responsibility of a new actor, the Administrator this actor, besides his management actions, will also have access to all features that are available to the Customer. With this approach, Figure 3.1 illustrates the elaborated diagram that presents the central use of the solution, with various actions and the corresponding actors.

Although some of these actions are previously explained in the functional requirements described in Subsection 4.1.1, the following are the actions available to each actor:

- **Customer**

  Querying food additives is the main functionality of the system, so the Customer can see the list of available additives and see the corresponding details for each one. Apart from this functionality, the listing of additive classes and their relationship to each additive is also crucial. In this way, the Customer can consult the listing of additive classes and even filter them by the desired Class. To complete the fundamental actions, the Customer can consult existing food categories in the same way as previous ones consultation of the list of existing categories, with access to authorized additives that constitute those same categories. Customer can view scientific articles related to the additive from major scientific databases, such as *PubMed*, *ScienceDirect*, or *Google Scholar*. This actor can also perform full-text searches, available through a search bar present on the website.

- **Administrator**

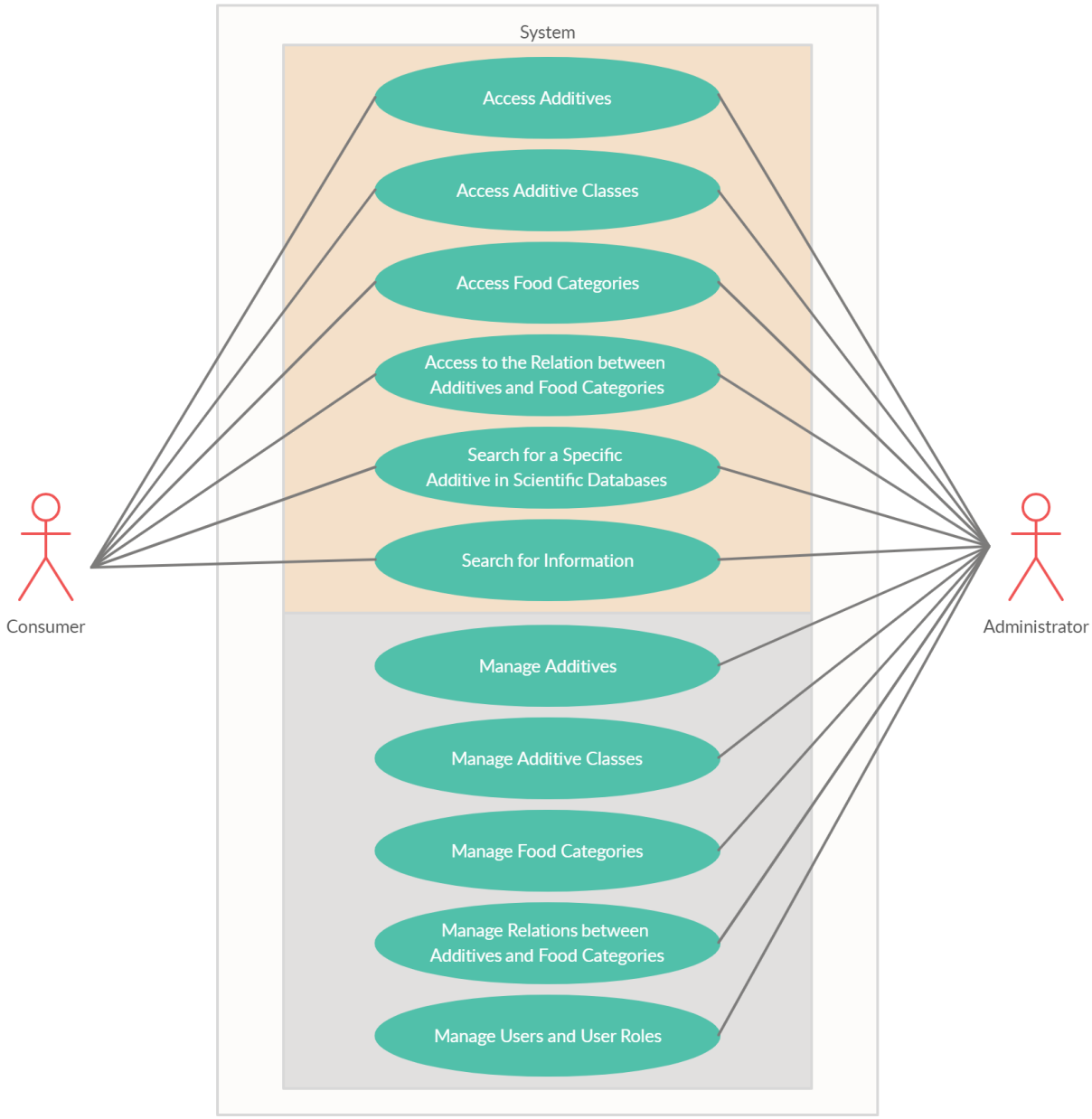


Figure 3.1: System Use Cases

Besides every action that the Customer may perform, the Administrator has even more. He is also able to manage all additive data. By adding new information, updating existing data, or improving the translations of each additive. The same functionality is available for both Additive Classes and Categories. However, the actions performed on these two entities are uncommon. As I recall, the information is updated by the competent authorities, and the definition of Classes and Categories is unlikely to be changed. Still, this possibility exists, and the solution holds it. This actor manages the relationship between Food Additives and Food Categories, which include options for adding, updating and removing observations and restrictions specific to each pair of food additive-categories. This action also provides for the management of maximum permitted doses for each case. Finally, the Administrator also has the power to list registered users in the system, change their role or delete their account.

3.2. Global Architecture

The system architecture, represented in Figure 4.2, was developed to fulfil all demands analysed in the requirements outlined in the prior section. Therefore, the solution to be implemented favours the client-server architecture, where there can be multiple clients. This architecture is organised in several layers, interconnected. This allows a centralised server where the clients send the requests while the server responds to those requests [52].

The Service-Oriented Architecture (SOA) enables a system to provide loosely coupled services to remote consumers or clients [40]. Features must be exposed through accessible client services. Thus, it follows a client-server model - where the server provides the services, and the client accesses the data through them or provides data to the server. The subsequent subsections describe this architecture in detail.

The Client side encapsulate most of the presentation part of the system, thus, the main function of the clients is to present the system data through graphical interfaces. However, as can be observed, the system is divided into two different individual clients.

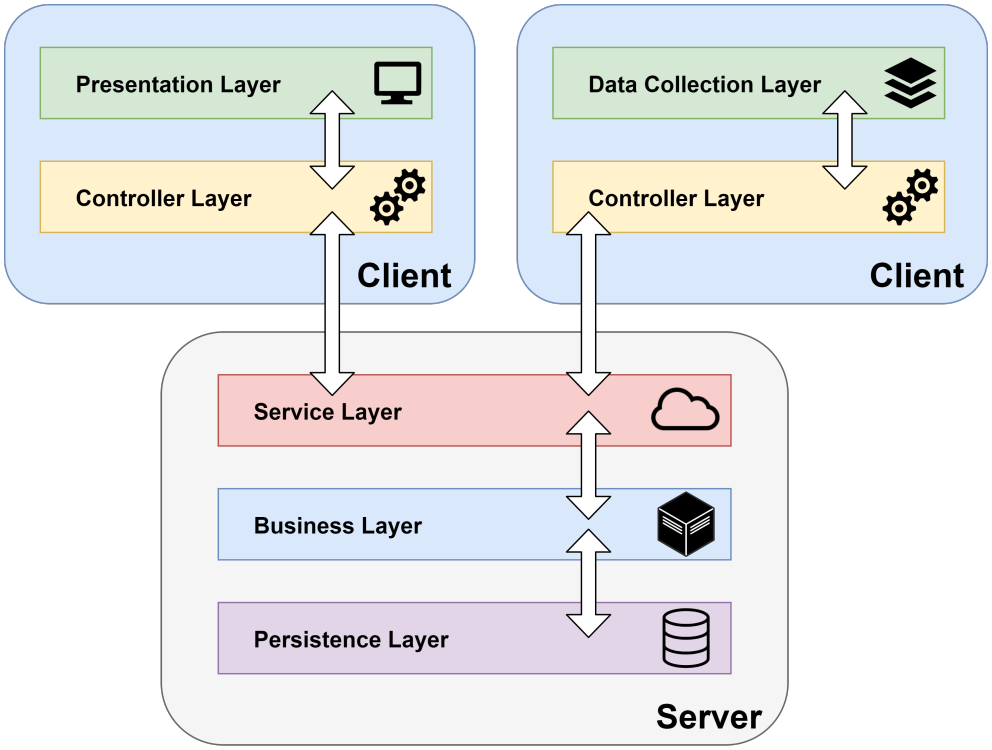


Figure 3.2: System Architecture

This is because there are clients with another purpose, such as web scraping tools, whose function is to collect data from several sources outside the system and send it to the server for storage. Therefore, we can admit clients that have a Controller layer interconnected with a presentation layer or a Data Collection layer.

The responsibility for managing the components that make up the graphical interface lies with the Presentation layer. In turn, the Data Collection layer is in charge of analyzing a source of information and collecting the necessary data. However, both types of clients use the Controller layer, which consumes RESTful services to retrieve data from the server or send collected data.

The Server, which is mainly where the backend core is embedded, encompasses three sub-layers:

1. Service Provider;

2. Business;

3. Persistence.

The Services layer includes the RESTful API that provides clients with access to the required application functionality. The middle layer is responsible for all the business logic of the system, thus linking services to the Persistence layer. Finally, to this layer belongs the responsibility of database accesses, both for reading and writing data. All of these elements will be detailed in the following subsections.

3.3. Data Model

Given the analysis performed above, it would be necessary to think of a database that could meet all the demands of the system. The created entity structure is adapted for efficient data storage in a relational database. However, for objects manipulation and their use in web services, this structure is not feasible, due to the separation of attributes between base entities and its translation which is a different entity. With this thought, the creation of auxiliary objects named Data Transfer Objects (DTO) was necessary. This brings together the attributes of both entity types, offering a higher level of abstraction to service clients, and, when the web services make a request, the response brings the data with the desired language. These objects are represented in Figure 3.3.

3.4. RESTful Web Services

This solution implements the REST architecture over Hypertext Transfer Protocol (HTTP). HTTP is a stateless client/server transport protocol used for retrieving hypertext documents. This protocol specifies eight methods as discussed on Subsection 2.4.3, though only four are mainly used in RESTful services. These four methods are: GET, POST, PUT and DELETE and allow implementing a CRUD (Create, Retrieve, Update, and Delete) interface to access any hyperlinked resource [38]. When creating a REST Web service, these steps should be followed [38], [53]:

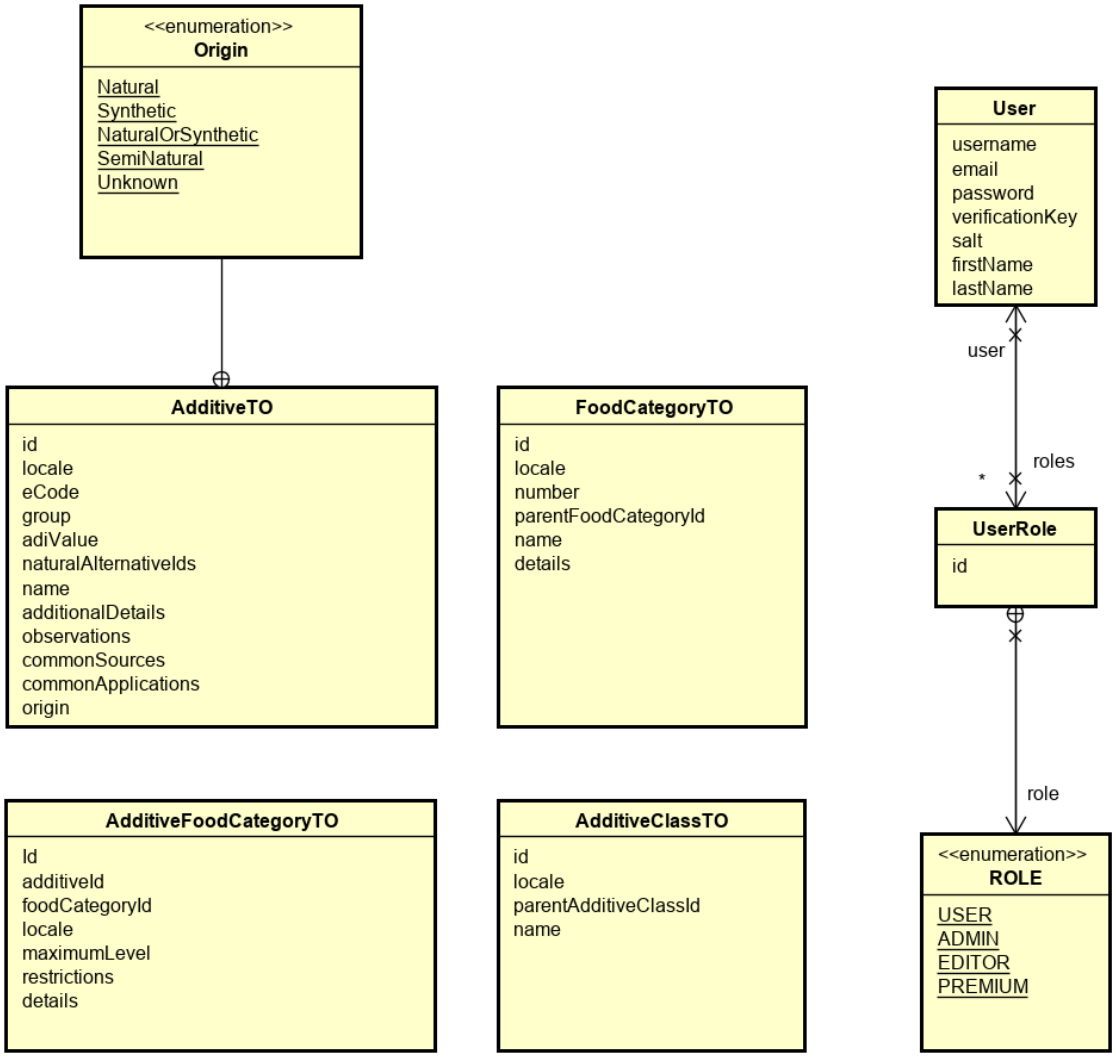


Figure 3.3: Class Diagram of DTOs

1. identify which resources should be published;
2. design resources URIs;
3. decide which representations will be available;
4. define which methods are available for each resource with a description of its effect;
5. list the possible responses (HTTP codes and result);

Each of these steps is fully explained in the following subsections, with the exception of topic number 5, that was discussed on Subsection 2.4.3

3.4.1 Resource Identification and URL Design

This solution is implementing REST over HTTP, the Global Unique Identifier (GUID) used are URLs which are built with the following syntax [54]:

```
<scheme name>://<host>/<version>/<path>?<query>
```

Scheme name refers to the protocol used to interpret the URL (in this case HTTP), the **host** refers to the server domain, while the **path** contains the information needed to locate the resource to be retrieved in the host.

The **version** is a symbolism that represents the API version that is being used. It allows the update to another version of the API, without removing any of the content of the previous version, making it deprecated.

The **query** part is optional, and it must not be used to identify resources. An example of a complete URL is:

```
http://www.domain.com:8080/v1/additives
```

Identification of resources is carried out in the same way that objects are identified in a system when using object-oriented paradigm. That is, dividing reality in concepts which are important for the system and which will encapsulate the needed data and application

state [55]. REST relies on the author to define the URL that best describes the resource he is creating, so it is very important to choose representative names. Bellow are some examples of resources and their identifiers, whereas URLs are fictitious:

- A list of persons:
  `http://www.domain.com/v1/persons/`
- The quantity of red sports cars:
  `http://www.domain.com/v1/vehicle/car/sports/red/qty/`
- The most viewed comment:
  `http://www.domain.com/v1/comments/mostviewed`

The URLs created for this solutions Web Services can be seen on Section 4.2.

3.4.2 Methods Description

Resources are manipulated through representations transfers through a uniform interface addressed by the resource identifier. Each representation has to implement a CRUD interface to achieve the uniform interface required by the REST architecture. HTTP protocol defines eight methods or verbs [39] (Subsection 2.4.3), four of which allows defining such interface: GET, PUT, POST and DELETE. 3.1 shows the correspondence of HTTP methods and CRUD actions.

Any representation has to accept all these four methods, although this does not imply they have to be implemented. Read-only resources, for instance, would implement GET requests while the rest of methods would return a suitable response code, as described on Subsection 4.3.2.

3.4.3 Listing Responses

It must be clearly stated which possible responses can be sent to the client depending on the request received. HTTP defines five categories of responses [39]:

Method	CRUD	Description
POST	Create	Creates a new resource
GET	Retrieve	Retrieve a representation
PUT	Update	Updates a resource
DELETE	Delete	Deletes a resource

Table 3.1: CRUD Correspondence with HTTP Methods

- 1xx Informational
- 2xx Success
- 3xx Redirection
- 4xx Error in the client side
- 5xx Error in the server side

Each of these groups contains some predefined HTTP codes which allow sending more information in the response. The most usual responses concerning the methods described above (section 3.2.3) are shown in Table 3.2.

It is possible to define new HTTP codes and responses, although it is not the best procedure, hence not being recommended. If a client receives an HTTP code it does not know, it will consider it as an X00 response code (where X is the category where the new HTTP code is defined). Besides the HTTP code and message response, the server has to send the proper representation in the case of GET requests or the URL of the resource created (in the Location header field) for POST requests. It is obligatory for servers to properly use HTTP response codes as they are described in [39]. Clients should at least understand the code responses described in Table 3.2.

Code	Methods	Description
200	GET	Ok (request was successful); the representation is sent to client
201	POST, PUT	Created (The location of the new resource is in Location header)
	PUT	Created (if the resource already existed, then content is updated)
204	GET, PUT, DELETE	No content (request was successful; the response body is empty)
400	All	Bad request (Bad formed request)
401	All	Unauthorized (the client needs to be authenticated)
404	All	Not found (the resource is not in the system)
405	All	Method not allowed

Table 3.2: Usual HTTP responses codes

3.5. Mobile Application

The architectural pattern encouraged by Apple on the iOS platform is Model View Controller (MVC). Model View Controller is one of the most popular presentational architectural designs, is used to create desktop, mobile and web applications. Its purpose is to provide a simple separation of concerns for an app that embeds a user interaction component. One of the most crucial aspects of this pattern is that the application should work and fulfil all its requirements even if the View and Controller layers are removed. The data manipulation and the business logic should reside the Model, and it should not be affected in any way by those other layers [56].

Apple's MVC [57] version is different from the common concept. When the classic theory of MVC arises, there wasn't the notion of mobile development. Therefore, to compensate for this fact, Apple created an extension of the classic MVC, called Cocoa version of MVC, that would be better suited for mobile and desktop applications.

This new model consists of the same three layers, Models, View and Controller, the only aspect that changes are the interaction and the data flow. The layers are more

decoupled, and the emphasis is on the Controller, as can be seen in Figure 3.4.

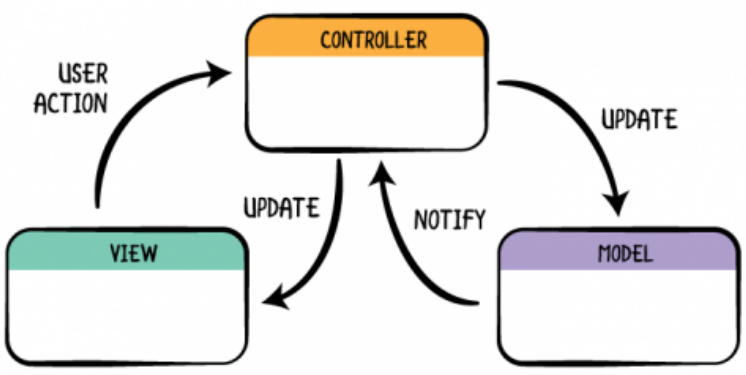


Figure 3.4: Apples Model-View-Controller Architectural Overview

At the centre of every iOS application, the View Controllers act as bridges between the data of the application (Model) and the user interfaces (View). Each one of these objects is intended to be separated of the other, and each performs a specific role:

- The **Model** is where the data resides. It incorporates things like persistence, model objects, parsers, managers, and networking.
- The **View** layer is the face of every app. Its classes are often reusable as they dont contain any domain-specific logic.
- The **Controller** mediates between the view and the model via the delegation pattern. In an ideal scenario, the controller entity wont know the concrete view its dealing. Instead, it will communicate with an abstraction via a protocol.

Auto Layout

The Auto Layout is a powerful feature present in iOS development. Auto Layout dynamically calculates the size and position of all the view in the view hierarchy, based on constraints set on those views[1]. For instance, if the developer puts a button and implements a horizontally centered constrain with an Image View and so that the button's

[1] `https://developer.apple.com/library/archive/documentation/UserExperience/Conceptual/AutolayoutPG/`

top edge always remains 5 points below the image's bottom. If the image's view or position changes, the button's position is automatically adjusted to match the constraint previously established.

This constraint-based strategy allows the developers to build interfaces that are dynamic and can respond to both internal and external changes.

- **External Changes**

  External changes occur when the size or shape of the superview (a view that contains other views) changes. On each change, the layout of the view hierarchy must be updated to best use the available space. These changes can arise because:

  - User resizes the window;
  - User rotates the device;
  - If the app wants to support different size classes;
  - If the app wants to support different screens.

  Most of these changes can happen at runtime, and they require a dynamic response from the app. Others, like support for various screen sizes, represent the app adapting to different environments. Even though the screen size wont typically change at runtime, building an adaptive interface lets the app run equally well on an iPhone 4S, an iPhone XS, or even an iPad.

- **Internal Changes**

  Internal changes occur when the size of the views or controls in the user interface change, for example:

  - The content displayed on the app changes;
  - The app supports internationalization;
  - The app supports Dynamic Type.

These commonly happen in apps that display text or images, when the content changes, the new content may require a different layout than the default. The internationalization can affect the app in several ways, the need to adapt to other languages affect the label size and consequently, the space it requires. When the language is changed, not only the text is altered but also the position of elements, changing the layout of the app dynamically. Finally, if the app supports Dynamic Type, the user can change the font size used on the app. This, obviously adjusts the height and width of any textual element present in the interface. If these changes are made on runtime, the fonts and layout must adapt.

Swift Package Manager

The Swift Package Manager is a tool for managing the distribution of Swift code. Its integrated with the Swift build system, that addresses challenges such as automate the process of downloading, compiling, and linking dependencies. This tool is included in all versions since Swift 3.0[2].

Although not used in the development of this particular solution, it is a very relevant tool in iOS development. The following points explain the basic concepts that drive the functionality of the Swift Package Manager:

- **Modules** Swift sorts out code into **modules**. Every module determines a namespace and implements access controls on which parts of that code can be utilized outside of the module. A program may have the majority of its code in a single module, or it might import different modules as *dependencies*. At the point when a different module with a bit of code that takes care of a specific issue is done, that code can be reused in different circumstances[2].

- **Packages**

  A **package** consists of Swift source files and a manifest file. The manifest file,

[2] `https://swift.org/package-manager`

called *Package.swift*, defines the packages name and its contents using the *PackageDescription* module. A package has one or more targets. Each target specifies a product and may declare one or more dependencies.

- **Products**

  A target may build either a library or an executable as its product. A library contains a module that can be imported by other Swift code. An executable is a program that can be run by the operating system.

- **Dependencies**

  A targets dependencies are modules that are required by code in the package. A dependency comprises of a relative or absolute URL to the source of the package and a lot of prerequisites for the version of the package that can be utilized. The role of the package manager is to lessen coordination costs via automating the way toward downloading and building the majority of the conditions for a project. This is a recursive procedure: A dependency can have its very own dependencies, every one of which can likewise have dependencies, shaping a dependency chart. The package manager downloads and builds everything that is expected to fulfill the entire dependency graph.

Drill Down

During the development of this mobile solution, it was adopted the Drill Down navigation philosophy. Thus, it implies that the user interface follows a Drill Down architecture. This forces the user to select content after content to get the intended target [34]. The challenge of this interface architecture is to be able to display enough and correct data to satisfy its needs and, at the same time, saving the user's time. Being this an unpleasant mode of present information to the user, because sometimes the app usability becomes painful, the need to create a simple and easy process flow for the app usage was vital.

Considering this mobile app has a lot of textual information to show it wasn't easy to find a solution. But, as observed later on Section 4.3, to guarantee that all information would be shown through tables and that the little information that would be visible at first glance, in a first hierarchically phase, was enough to navigate the application in an intuitive to practical way. The Drill Down functionality typically takes a selected data and provide another list in a table correspondent to the selection previously made. This process creates a hierarchical selection, as mentioned before.

Let's look at the practical example of this solution. On the additives listing page, information that is displayed on a table, if the user selects an additive, it is automatically referred to its details, which are shown in another table. Once in the details table, you can still access the X page, which information is also shown in a table. This process is called Drill Down, and it is beneficial for devices with little viewing space.

Chapter 4

System Implementation

The desired solution was considered, various arrangements were reviewed, following the interpretation of a set of prerequisites. All of these cases have been depicted in the preceding chapters. This chapter addresses the implementation details.

Following the architecture model defined in the Section 3.2, a few technologies and tools were selected to satisfy the architectural requirements. On Section 4.1 those technologies are presented with a concise description of its purpose.

The following section, Section 4.2 illustrates the implemented Web Services, with their specification and features.

Additionally, Section 4.3, introduces the final perspective of this mobile solution, with the graphical representation of each existent page, their function, and the central system features.

4.1. Tools & Technologies

The plan proposed in the research grant gave the developers total freedom on the technologies to be used and, therefore, were chosen only after analysis of all the demands and the current state of the art, as discussed on Section 3.1.

Swagger

Swagger is an open-source software framework backed by a vast ecosystem of tools that helps to design, build, document, and consume RESTful web services. While most developers identify Swagger by the Swagger UI tool, the Swagger toolset includes support for automated documentation, code generation, and test-case generation [58]. A use case of this tool can be seen in Figure 4.1.

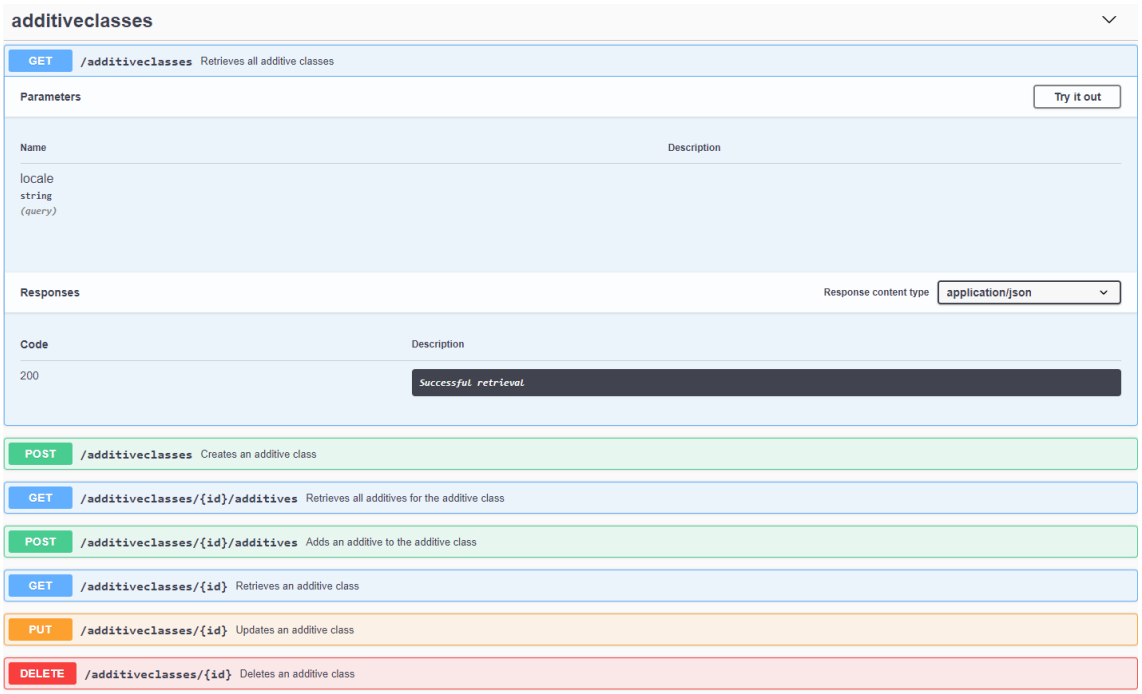


Figure 4.1: Additive Classes Class Documented in Swagger

Swift

Swift is a general-purpose programming language developed by Apple Inc[1]. It was designed to allow developers to create software for iOS, iPadOS, macOS, watchOS, tvOS, basically all platforms from Apple [59]. It arises as an alternative to Objective-C. Although Objective-C would also be a valid option, Swift was the chosen language to develop the

[1] `http://www.apple.com`

software for this mobile solution.

Swift was launched on June 2, 2014, and became an open-source language on December 3, 2015[2], and since then, it has some online editors/compilers, such as `http://www.iswift.org/playground` that supports the development of this language. Although, for app submission on App Store, a Swift stable version to be used on Xcode remains a requirement.

Swift has several advantages comparing with his predecessor, it adopted modern programming patterns and strived to present a more straightforward and clear syntax, that makes it less error-prone. Swift requires less code, provides various speed advantages during development and in performance is faster than Objective-C [60]. Despite all these factors that favour Swift, the primary consideration was the willingness to learn and work with this language was the motivation behind this choice.

Xcode

Xcode is an IDE for macOS containing a suite of software development tools developed by Apple for developing software for macOS, iOS, iPadOS, watchOS, and tvOS. First released in 2003, the latest stable release is version 11.0 and is available via the Mac App Store free of charge for macOS Mojave users [61].

Core Data

Core Data is a framework developed and maintained by Apple, that manages the object life cycle and Object Graph. Object Graph is nothing more than a collection of objects that are connected with one another. Whereas a normal data model such as a UML Class diagram details the relationships between classes, the Object Graph relates their instances. But Core Data is much more than that. The framework adds a number of other compelling features, such as input validation, data model versioning, and change tracking. Is the M in MVC, Core Data typically decreases by 50 to 70 percent the amount of code

[2] `https://developer.apple.com/swift/blog/?id=34`

needed to write to support this Model Layer. It can optionally persist the object graph to disk and it also offers a powerful interface for searching the object graph it manages[3].

JSONSerialization

As mentioned on Subsection 2.3.1, the Foundation framework is vital for almost every iOS application. This framework contains the Foundation Objects, that correspond to the essential data types, such as NSString, NSNumber, NSArray, NSDictionary, or NSNull.

Object serialization is converting structured data into sequential representation. In other words, preserve the current state (attributes, values, etc.) of objects to use them later to reconstruct the objects. Serialization is used for data transfer between apps via network or within the same device. Besides that, it is used for storing data in a database or persisting in a file [62].

That said, JSONSerialization is class to convert JavaScript Object Notation (JSON) to Foundation objects and convert Foundation objects to JSON[4].

Grand Central Dispatch

A thread is a path of execution of the smallest sequence of programmed instruction that can be managed independently by the operating system. This said, a thread is a path of execution within a process, and a process can contain multiple threads. The idea is to achieve parallelism by dividing a process into numerous threads - multithreading [63]. A problem arises when dealing with multithreads, thread synchronization. This is essentially when two or more threads within a program, there may be a situation when multiple threads try to access the same resource, and finally, they can produce unexpected result due to concurrency issues[5]. To deal with this problem, Grand Central Dispatch (GCD) was created.

[3] `https://developer.apple.com/library/archive/documentation/Cocoa/Conceptual/CoreData`
[4] `https://developer.apple.com/documentation/foundation/jsonserialization`
[5] `https://www.tutorialspoint.com/java/java_thread_synchronization.htm`

GCD is a technology developed by Apple Inc. to optimize application support for systems with multi-core processors and other symmetric multiprocessing systems [64]. The fundamental idea is to move the management of the thread pool out of the hands of the developer, and closer to the operating system. This low-level API allows developers to write multithreaded applications without the need to manage threads at all. All developers must do is to define tasks and leave the rest to GCD [65], submitting work to dispatch queues maintained by the system.

4.2. Web Services

The Client-side has two layers, the Presenting Layer, and the Controller Layer. The previous section already described the key technologies and tools used. One of the responsibilities of the Controller Layers is to consume web services.

The consumption of client-side web services in Swift is executed through a native language class, the *URLSession*, a complete networking API provided by Apple, for uploading and downloading content[6].

4.2.1 Accessory Services

On the following subsection, Subsection 4.2.2, are represented all Web Services that feed the solution, both mobile and web applications. Below are some essential details to consider when requesting specific resources:

- **Internationalization and localization**

  Some resources admit the **locale** query string. Provided values must correspond to two-letter language tags. Currently available languages are **EN**, **PT**, and **ES**. If invalid or absent, the default locale is used (EN). The response will always contain information about the language of the retrieved content in the **Content-Language** header.

[6] `https://developer.apple.com/documentation/foundation/urlsession`

- **Pagination**

  Pagination is available on certain resources and consists of two query strings: **offset** and **limit**. Offset starts at 0 for the first page of results and each increment corresponds to the next page. Limit specifies the number of results per page and defaults to 25 if it isn't included in the request. The total number of available results will be included in the response in the form of the **X-Total-Count** header.

- **Sorting**

  Sorting can be achieved by appending sort=field&order=ord to the query string. Order can be **ASC** or **DESC** and defaults to **ASC** if it isn't provided:

  GET http://localhost:8083/v1/additives?sort=id&order=DESC

- **Filtering**

  Filtering is not yet supported. Filtering is available for certain resources by appending *filter = field = value* to the query string. Multiple filters can be comma separated and the following operands are available: <, >, =, <=, >= and **!=**:

  GET http://localhost:8083/v1/additives?filter=id>=5,origin!=Synthetic

4.2.2 List

As mentioned before on Section 3.4.1, with the goal of homogenizing the web services definition, an URL structure was defined. This helps to better understand what the web service is calling. This said, all web services start are preceded by a base URL, or path, followed by the version (*'/v1/'*), followed by the name of the resource pretended. The next element in the URL can be the method to call or the id, depending on the web service's definition. The following tables represent the available web services for this solution.

Additives

Method	URL	Description
GET	/additives/{id}/foodcategories	Retrieves all food category relationships for the additive
POST	/additives	Creates an additive
GET	/additives	Retrieves all additives
GET	/additives/{id}/additiveclasses	Retrieves all additive classes for the additive
GET	/additives/{id}	Retrieves an additive
PUT	/additives/{id}	Updates an additive
DELETE	/additives/{id}	Deletes an additive
GET	/additives/ecode/{ecode}	Retrieves an additive

Table 4.1: Additives Services

Additive Classes

Method	URL	Description
GET	/additiveclasses/{id}/additives	Retrieves all additives for the additive class
POST	/additiveclasses/{id}/additives	Adds an additive to the additive class
GET	/additiveclasses	Retrieves all additive classes
POST	/additiveclasses	Creates an additive class
GET	/additiveclasses/{id}	Retrieves an additive class
PUT	/additiveclasses/{id}	Updates an additive class
DELETE	/additiveclasses/{id}	Deletes an additive class

Table 4.2: Additive Class Services

Food Categories

Method	URL	Description
GET	/foodcategories/{id}/additives	Retrieves all additive relationships for the category
GET	/foodcategories	Retrieves all food categories
POST	/foodcategories	Adds a food category to the database
GET	/foodcategories/{id}	Finds a food category by id
PUT	/foodcategories/{id}	Updates an existing food category
DELETE	/foodcategories/{id}	Deletes a food category
GET	/foodcategories/number/{number}	Finds a food category by number

Table 4.3: Food Categories Services

Additive Food Categories

Method	URL	Description
GET	/additivefoodcategories	Retrieves all additive food categories
POST	/additivefoodcategories	Creates an additive food category
GET	/additivefoodcategories/{id}	Retrieves an additive food category
PUT	/additivefoodcategories/{id}	Updates an additive food category
DELETE	/additivefoodcategories/{id}	Deletes an additive food category

Table 4.4: Additive Food Categories Services

Search

Method	URL	Description
GET	/search	Retrieves search results for a given query

Table 4.5: Search Services

Users

Method	URL	Description
GET	/users	Retrieves all users in the database
POST	/users	Adds a user to the database
POST	/users/login	Login
GET	/users/{username}	Retrieves a user by its email
PUT	/users/{username}	Updates an existing user
DELETE	/users/{username}	Deletes a user

Table 4.6: Users Services

User Roles

Method	URL	Description
GET	/userroles/{id}	Retrieves a role
DELETE	/userroles/{id}	Deletes a role
GET	/userroles	Retrieves all user roles
POST	/userroles	Adds a role to a user

Table 4.7: User Roles Services

4.3. Mobile Application Interface

On this section, an overview of the mobile solution is presented. It analyses and details the main system features, describing page by page, along with illustrations on the final product. The layout adopted was based on the simplicity of use of the application itself the arrangement of the elements together with the necessity to read so much textual information.

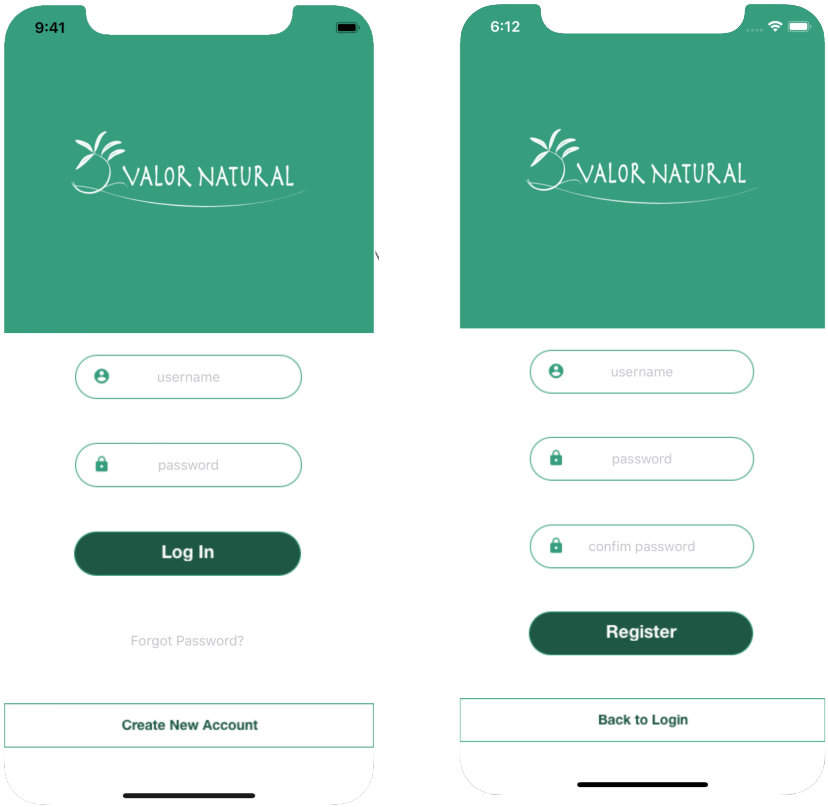
4.3.1 Login & Register

To be able to access the application and use its features, the user needs to be logged in. The Login can be performed through the *Login Page* represented in Figure 4.2 (a), if the current user does not have an account created he can do it, by filling the data fields on the *Register Page*, as illustrated on Figure 4.2 (b).

As soon as the application is accessed, it is evident the presence of a *Tab Bar*, navigation element, in the interface. Through this Tab Bar, it is possible to navigate quickly and practically among all the pages and features available. Another essential component of navigation is the *Nav Bar*, which allows the user to go back to the previous page and also to provide information. Tab Bar's first option is the additives page, described below, in Subsection 4.3.2.

4.3.2 Additives

Figure 4.3 represents the first highlight page of the app. Once the user gets a successful login, it is redirected to this page, the *Additive Listing Page*. As soon as the user accesses this page, he is then able to browse through the list of all additives in the database and access the detail of the intended additive. Furthermore, it is also possible to search for any E-number or additive name, the return of that search is a list of one or more additives that correspond with the inserted search expression, whose details can be accessed as well.



(a) Login Page (b) Register Page

Figure 4.2: Login and Register Pages

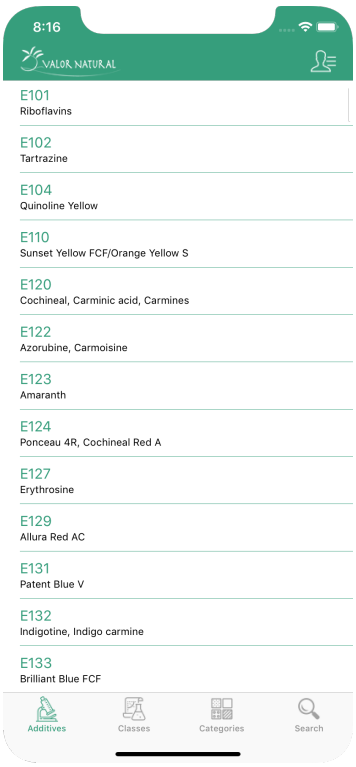


Figure 4.3: Additives Listing Page

The details outlined in the paragraph above, leads to the *Additive Details Page*, represented on Figure 4.4. This page enables the user to examine the chosen additive, observing some properties with more detail.

This additional information is displayed through a table containing the additive class and group that the current additive belongs, the origin of the additive, the common sources of the additive, its common applications, some useful observations (such as side effects) and finally the additive evaluation.

Lastly, the user can go further and find the stipulations that exist between the current additive and a specific Food category. He can access that relation trough the button placed in the bottom of the page, that leads to the *Related Food Categories Page*, illustrated on Figure 4.5 (a).

Once the user is on this page, he is presented with a list of all Food Categories. This list contains their correspondent number and name that the additive is related. After that, the user can choose which Food Category conditions he wants to see by selecting

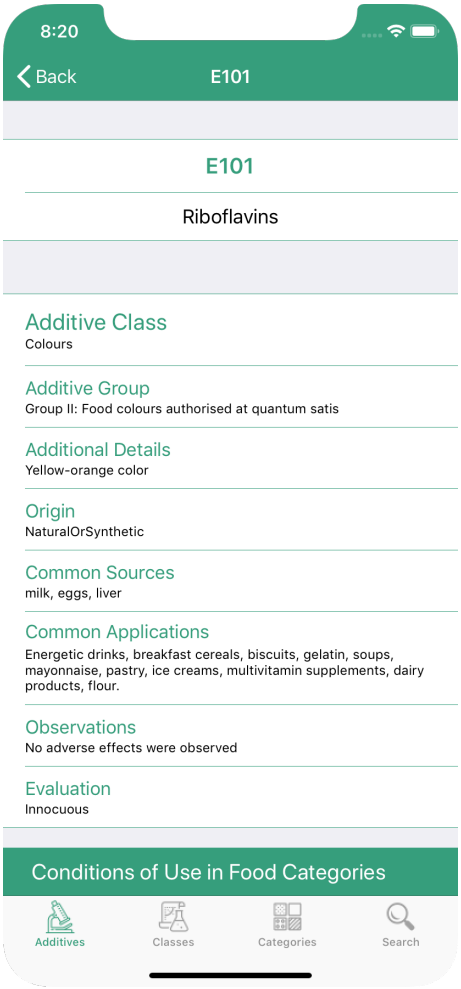
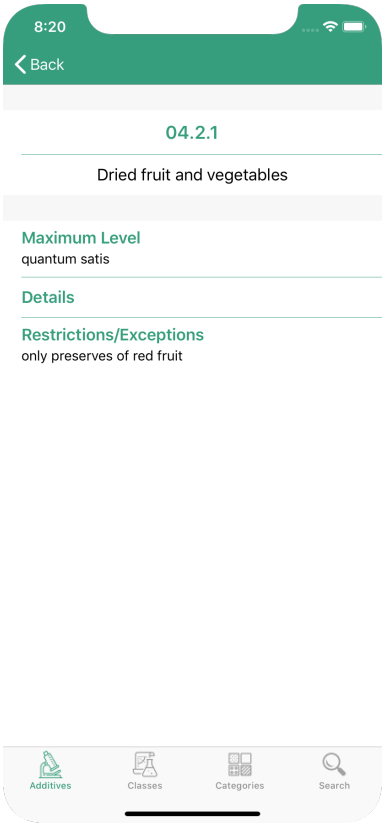


Figure 4.4: Additive Details Page

(a) Related Food Categories



(b) Conditions Of Use

Figure 4.5: Conditions of Use in Food Categories Pages

any of the Food Category displayed, being redirected to the *Conditions Of Use Page*, illustrated on Figure Figure 4.5 (b). On this page, the information displayed consists of the number and name of the chosen Food Category and the respective additive conditions of use. Namely, the maximum level permitted, some details between that condition, and Restrictions or Exceptions found. With all of this process, terminates the Drill Down relative to additives.

4.3.3 Additive Classes

The second option on the Tab Bar is the *Additives Class Listing Page*. This page, as seen on Figure 4.6, represents a table containing the six existing Classes on the database. The user can browse through that table and select the desired Class. Moreover, it is also possible to search for the name of any of the Classes, the outcome of that search is a list of one or more classes that correspond with the inserted search expression, whose additives belonging to that Class can be accessed as well.

After the user selects the intended Class, he accesses the *Additive Class Detail Page*. This page contains a list, in the form of a table, of all additives presented on the database, that are incorporated into the selected Class. This page can be observed in Figure 4.7.

While on this page, the user can choose the desired additive and, after that, the following process is similar to that described in the Subsection above, Subsection 4.3.2.

4.3.4 Food Categories

As a third option on the Tab Bar, is presented the *Food Categories Listing Page*, on it is displayed a table aggregating all of the existing Food Categories, as seen of Figure 4.8. Those can be reached one by one, leading to the associated details, as shown on Figure 4.9 (a). Additionally, it is also possible to search for the name or number of any Food Category present in the database. The result of that search is a list of one or more Food Categories that correspond with the inserted search expression and can be accessed as well.

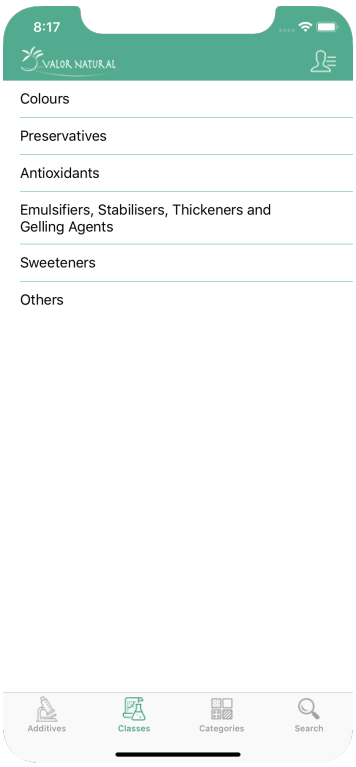


Figure 4.6: Additive Classes Listing Page

The *Food Category Details Page* shows the number and name of the desired Food Category. Besides that information, its details are also displayed. Additionally, there is the option to search for the Additives who have conditions of use that belong to that specific Food Category.

This option is available through the button on the bottom of the page named "Additives subject to Restrictions", and it redirects the user to the page represented on Figure 4.9 (b). This page lists all additives and follows the same flow described on Subsection 4.3.2.

4.3.5 General Search

The last option on the Tab Bar is the general search, illustrated on Figure 4.10. On this page, the user has the opportunity to search for literally anything related to Additives. It is possible to search for the same specifications as on previous pages, but it has some

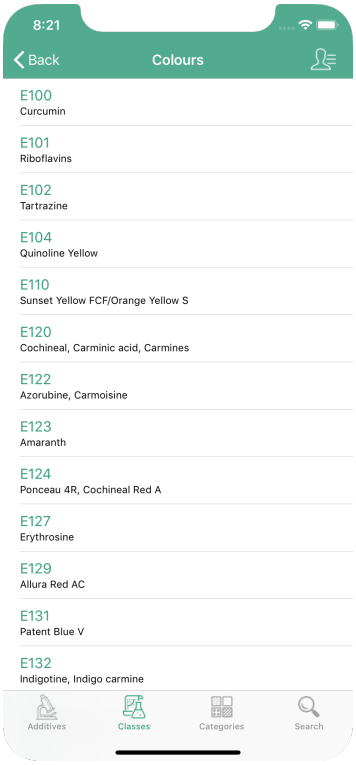


Figure 4.7: Additive Class Detail Page

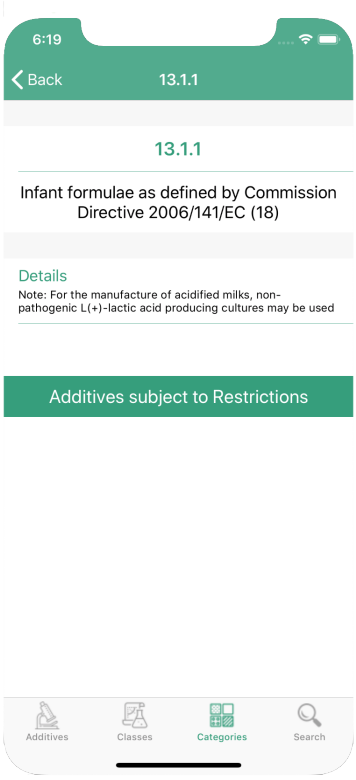
unique features. In addition to aggregate the options that only existed individually, i.e. additives page only allows additives search, it has the benefit of allowing a search for any expression that is present in the database. This expression may be described in a description of a food category, the origin of an additive, etc. The result of this search can be returned in the form of Additives, Additive Food Category or Additive Class, or sometimes all of the above.

On this stage, the app is ready for deployment, although it can benifit from an usability study. Regarding the user interaction with the app, it is not yet possible to obtain any kind of feedback besides the developer and a restrict group of people which were members of the project or were in a closed environment during the development.

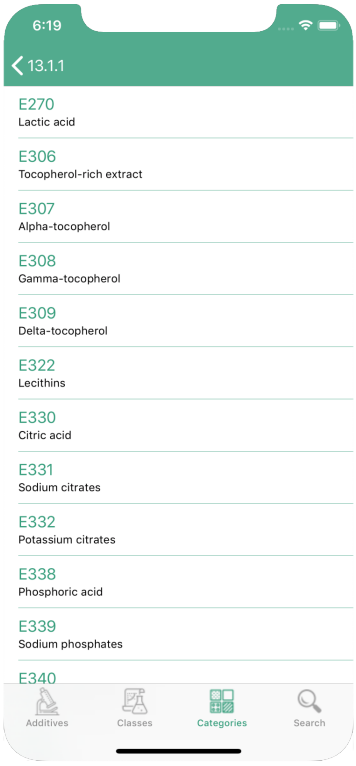
These tests, although they befell with people that, for obvious reasons, previously knew the intent of the app, it was possible to obtain some valuable information. It was possible to identify some imperfections that were present and some behaviours that were not supposed to happen.

Figure 4.8: Food Categories Listing Page

Nonetheless, it still requires feedback from users that are unfamiliar with the app or its purposes and, from users that do not comprehend what an additive is. With that, it will be possible to establish if the system is intuitive and incorporates all the needs that it is supposed to. Simply after those tests and its possible corrections, fundamentally at the interface level, the application will be completed.



(a) Food Category Details



(b) Additives Subject to Restrictions Listing

Figure 4.9: Food Categories & Related Additives

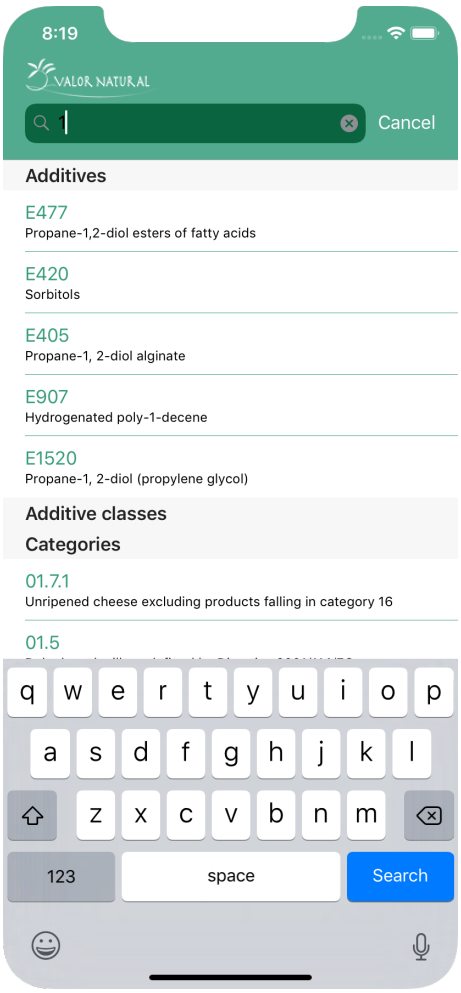


Figure 4.10: General Search Page

Chapter 5

Concluding Remarks

In the previous chapters, numerous aspects of the system conception were described, from the requirement analyses to the implementation. Although most of the requirements were already defined in the research grant proposal, a review of the problem and the various solutions on the market was necessary. The development of the system was reviewed in detail in chapters 3 and 4, with an explanation of the surrounding architecture, its design and focusing on the key features that make the solution so important.

The project was developed in collaboration with the researchers of CIMO, from the initial discussions on the prerequisites of the system as well as in its development process. This collaboration led to new approaches, diverse thoughts of the problem and the use of methodologies to analyze the proposal with the help of specialists from different knowledge areas.

One of the most significant challenges to the project development was the analysis that needed to be made and the system concept. Defining all the technical demands for the problem, which has a lot of scientific specifications demanded a considerable effort to understand all the issues and requirements needed for its conception. The implementation of all of this also has some interesting challenges to be considered, such as the learning of a new programming language (Swift) and the behaviour of an iOS ecosystem, to the mobile solution implementation.

It is expected that the most significant scientific contribution will, in fact, be the resulting product of the developed solution. The future of food additives is of grave importance for humanity. It is related to the well being of the entire human population. Without them, it would not be possible to maintain the current selection and quality of foods. The concern about misinformation regarding food additives is also fueled by the blurred separation between natural and synthetic additives. With this solution, the central purpose will always be to inform consumers or the industry, to clarify and demystify any problems that may exist around food additives.

Throughout the development process, some ideas have emerged for potential future improvements. Some of them unrealistic, others considerably impressive, that could be a motive of distinction for the solution. One of these would be the development of an image recognition feature that would allow the user to scan the label of a food product package in which the application would automatically recognize the additives listed and immediately refer to their details.

Another possibility of improvement would be, in a business context, the creation of various types of accounts, which would allow controlled access to multiple types of information. These account types are already implemented in the database to implement this functionality in the near future.

Bibliography

[1] M. Carocho, P. Morales, and I. C. F. R. Ferreira, "Natural food additives: Quo vadis?", *Trends in Food Science & Technology*, vol. 45, no. 2, 2015.

[2] M. Carocho, M. F. Barreiro, P. Morales, and I. C. Ferreira, "Adding molecules to food, pros and cons: A review on synthetic and natural food additives", *Comprehensive reviews in food science and food safety*, vol. 13, no. 4, 2014.

[3] M. E. Morris and A. Aguilera, "Mobile, social, and wearable computing and the evolution of psychological practice.", *Professional Psychology: Research and Practice*, vol. 43, Dec. 2012.

[4] S. Tachakra, X. Wang, R. S. Istepanian, and Y. Song, "Mobile e-health: The unwired evolution of telemedicine", *Telemedicine Journal and E-health*, vol. 9, no. 3, pp. 247–257, 2003.

[5] A. Faiola, E. L. Papautsky, and M. Isola, "Empowering the Aging with Mobile Health: A mHealth Framework for Supporting Sustainable Healthy Lifestyle Behavior", *Current Problems in Cardiology*, vol. 44, no. 8, 2019.

[6] A. Balapour, I. Reychav, R. Sabherwal, and J. Azuri, "Mobile technology identity and self-efficacy: Implications for the adoption of clinically supported mobile health apps", *International Journal of Information Management*, vol. 49, 2019.

[7] B. Lopes, T. Padrão, M. Carocho, R. P. Lopes, and I. C. F. R. Ferreira, "Descodificar os e: Plataforma online de acesso aberto de aditivos alimentares", *14º Encontro de Química dos Alimentos*, 2018. [Online]. Available: https://bibliotecadigital.ipb.pt/handle/10198/18186.

[8] M. Saltmarsh and M. Saltmarsh, *Essential guide to food additives*. Royal Society of Chemistry, 2013.

[9] L. D. Tomaska and S. Brooke-Taylor, "Food Additives: Food Additives General", in *Encyclopedia of Food Safety*, Y. Motarjemi, Ed., Waltham: Academic Press, 2014, ISBN: 978-0-12-378613-5. [Online]. Available: http://www.sciencedirect.com/science/article/pii/B9780123786128002341.

[10] G. Tarnavölgyi, "Professional and consumer attitudes towards food additives", *Doctoral (PhD) Thesis Summary, Kaposvár University, Kaposvár*, 2009.

[11] R. Smith, "Regulation (EC) No 764/2008 of the European Parliament and of the Council", en, in *Core EU Legislation*, London: Macmillan Education UK, 2015, ISBN: 978-1-137-54501-5 978-1-137-54482-7. [Online]. Available: http://link.springer.com/10.1007/978-1-137-54482-7_19.

[12] R. C. W. Welch and P. C. S. Mitchell, "Food processing: A century of change.", *British medical bulletin*, vol. 56, no. 1, pp. 1–17, 2000. DOI: 10.1258/0007142001902923.

[13] *CODEXALIMENTARIUS FAO-WHO*. [Online]. Available: http://www.fao.org/fao-who-codexalimentarius.

[14] E. Commission *et al.*, "Commission regulation (eu) no 1129/2011 of 11 november 2011 amending annex ii to regulation (ec) no 1333/2008 of the european parliament and of the council by establishing a union list of food additives", *Official Journal of the European Union L*, vol. 295, no. 4, pp. 12–11, 2011.

[15] A. Bearth, M.-E. Cousin, and M. Siegrist, "The consumers perception of artificial food additives: Influences on acceptance, risk and benefit perceptions", *Food Quality and Preference*, vol. 38, pp. 14–23, 2014.

[16] R. Cammack, C. Joannou, X.-Y. Cui, C. Torres Martinez, S. R. Maraj, and M. N. Hughes, "Nitrite and nitrosyl compounds in food preservation", en, *Biochimica et Biophysica Acta (BBA) - Bioenergetics*, vol. 1411, no. 2-3, pp. 475–488, May 1999, ISSN: 00052728. DOI: 10.1016/S0005-2728(99)00033-X. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S000527289900033X (visited on 08/30/2019).

[17] P. Cressey and S. Jones, "Levels of preservatives (sulfite, sorbate and benzoate) in new zealand foods and estimated dietary exposure", *Food Additives and Contaminants*, vol. 26, no. 5, pp. 604–613, 2009.

[18] M. Lucová, J. Hojerová, S. Paoureková, and Z. Klimová, "Absorption of triphenylmethane dyes brilliant blue and patent blue through intact skin, shaven skin and lingual mucosa from daily life products", *Food and chemical toxicology*, vol. 52, pp. 19–27, 2013.

[19] P. M. Davidson, J. N. Sofos, and A. L. Branen, *Antimicrobials in food*. CRC press, 2005.

[20] M. Carocho and I. CFR Ferreira, "The role of phenolic compounds in the fight against cancer–a review", *Anti-Cancer Agents in Medicinal Chemistry (Formerly Current Medicinal Chemistry-Anti-Cancer Agents)*, vol. 13, no. 8, pp. 1236–1258, 2013.

[21] G. A. Burdock, *Encyclopedia of food & color additives*. CRC Press, 2014.

[22] H. A. Abdulmumeen, A. N. Risikat, and A. R. Sururah, "Food: Its preservatives, additives and applications", en, 2012. DOI: 10.13140/2.1.1623.5208. [Online]. Available: http://rgdoi.net/10.13140/2.1.1623.5208 (visited on 09/04/2019).

[23] F. Shahidi, "Antioxidants in food and food antioxidants", *Food/nahrung*, vol. 44, no. 3, pp. 158–163, 2000.

[24] *Emulsifiers, Stabilisers, Thickeners, Gelling Agents - Shellac, Waxes, Gums, Resins, Menthol & Zein.* [Online]. Available: `https://www.afsuter.com/application/emulsifiers-stabilisers-thickeners-gelling-agents/` (visited on 09/05/2019).

[25] EU, "European parliament and council directive no 95/2/ec of 20 february 1995 on food additives other than colours and sweeteners", *Official Journal of the European Commission*, vol. 61, pp. 1–40, 1995.

[26] *Sweeteners.* [Online]. Available: `https://www.efsa.europa.eu/en/topics/topic/sweeteners`.

[27] W. H. Organization *et al.*, "Codex alimentarius: General standard for food additives.", *Codex Alimentarius: general standard for food additives.*, 2011.

[28] F. Nayebi, J.-M. Desharnais, and A. Abran, "The state of the art of mobile application usability evaluation", in *2012 25th IEEE Canadian Conference on Electrical and Computer Engineering (CCECE)*, IEEE, 2012, pp. 1–4.

[29] A. Holzer and J. Ondrus, "Trends in mobile application development", in *International Conference on Mobile Wireless Middleware, Operating Systems, and Applications*, Springer, 2009, pp. 55–64.

[30] S. Xanthopoulos and S. Xinogalos, "A comparative analysis of cross-platform development approaches for mobile applications", in *Proceedings of the 6th Balkan Conference in Informatics*, ACM, 2013, pp. 213–220.

[31] A. Ribeiro and A. R. da Silva, "Development of mobile applications using a model-driven software development approach", *Instituto Superior Técnico Lisbon, Portugal*, 2014.

[32] S. Helal, R. Bose, and W. Li, "Mobile platforms and development environments", *Synthesis Lectures on Mobile and Pervasive Computing*, vol. 7, no. 1, pp. 1–120, 2012.

[33] K. Johnson, Y. Li, H. Phan, J. Singer, and H. Trinh, "The innovative success that is apple, inc.", 2012.

[34] B. Fling, *Mobile design and development: Practical concepts and techniques for creating mobile sites and Web apps*. O'Reilly Media, Inc, 2009.

[35] S. Allen, V. Graupera, and L. Lundrigan, *Pro smartphone cross-platform development: iPhone, blackberry, windows mobile and android development and distribution*. Apress, 2010.

[36] A. Charland and B. Leroux, "Mobile application development: Web vs. native", *Communications of the ACM*, vol. 54, no. 5, pp. 49–53, 2011.

[37] W. Jobe, "Native apps vs. mobile web apps.", *International Journal of Interactive Mobile Technologies*, vol. 7, no. 4, 2013.

[38] L. J. Oliva Felipe, "Design and development of a rest-based web service platform for applications integration", Master's thesis, Universitat Politècnica de Catalunya, 2010.

[39] R. T. Fielding and R. N. Taylor, *Architectural styles and the design of network-based software architectures*. University of California, Irvine Doctoral dissertation, 2000, vol. 7.

[40] B. C. Henry, "Restful services–applying the rest architectural style", PhD thesis, Regis University, 2011.

[41] R. C. d. C. Gonçalves, "Restful web services development with a model-driven engineering approach", PhD thesis, 2018.

[42] R. Alexandre Peixoto de Queirós, A. Simões, M. T. Pinto, and S. Patnaik, Eds., *Code Generation, Analysis Tools, and Testing for Quality:* en, ser. Advances in Computer and Electrical Engineering. IGI Global, 2019, ISBN: 978-1-5225-7455-2 978-1-5225-7456-9. DOI: 10.4018/978-1-5225-7455-2. [Online]. Available: http://services.igi-global.com/resolvedoi/resolve.aspx?doi=10.4018/978-1-5225-7455-2.

[43] V.-C. Nguyen, X. Qafmolla, and K. Richta, "Domain Specific Language Approach on Model-driven Development of Web Services", en, *Acta Polytechnica Hungarica*, vol. 11, no. 8, p. 18, 2014.

[44] B. Selic, "The pragmatics of model-driven development", *IEEE software*, vol. 20, no. 5, pp. 19–25, 2003.

[45] *What are idempotent and/or safe methods? - The RESTful cookbook*. [Online]. Available: `http://restcookbook.com/HTTP%20Methods/idempotency/`.

[46] B. Svendsen and M. Roehne, "Data networks and open system communications", *TELEKTRONIKK*, vol. 90, pp. 186–186, 1994.

[47] P. Loucopoulos, "Requirements engineering", in *Design process improvement*, Springer, 2005, pp. 116–139.

[48] B. Lightsey, "Systems engineering fundamentals", DEFENSE ACQUISITION UNIV FT BELVOIR VA, Tech. Rep., 2001.

[49] R. Fulton and R. Vandermolen, *Airborne Electronic Hardware Design Assurance: A Practitioner's Guide to RTCA/DO-254*. CRC Press, 2017.

[50] L. Bass, P. Clements, and R. Kazman, *Software architecture in practice*. Addison-Wesley Professional, 2003.

[51] A. Fantechi, S. Gnesi, G. Lami, and A. Maccari, "Applications of linguistic techniques for use case analysis", *Requirements Engineering*, vol. 8, no. 3, pp. 161–170, 2003.

[52] S. Oluwatosin, "Client-Server Model Haroon", 2014.

[53] C. Pautasso, O. Zimmermann, and F. Leymann, "RESTful Web Services vs. Big Web Services: Making the Right Architectural Decision", en, *Web Engineering*, p. 10, 2008.

[54] T. Berners-Lee, R. Fielding, L. Masinter, *et al.*, *Uniform resource identifiers (uri): Generic syntax*, 1998.

[55] C. Larman, *Applying UML and patterns: an introduction to object oriented analysis and design and interative development*. Pearson Education India, 2012.

[56] D. Dobrean and L. Diosan, "Model view controller in ios mobile applications development",

[57] *Model-View-Controller*. [Online]. Available: `https://developer.apple.com/library/archive/documentation/General/Conceptual/CocoaEncyclopedia/Model-View-Controller/Model-View-Controller.html`.

[58] *The Best APIs are Built with Swagger Tools | Swagger*. [Online]. Available: `https://swagger.io/` (visited on 10/09/2019).

[59] A. Inc, *Swift.org*, en. [Online]. Available: `https://swift.org`.

[60] M. Wilde, M. Hategan, J. M. Wozniak, B. Clifford, D. S. Katz, and I. Foster, "Swift: A language for distributed parallel scripting", *Parallel Computing*, vol. 37, no. 9, pp. 633–652, 2011.

[61] *Xcode - Apple Developer*. [Online]. Available: `https://developer.apple.com/xcode/` (visited on 10/09/2019).

[62] M. Philippsen and B. Haumacher, "More efficient object serialization", in *International Parallel Processing Symposium*, Springer, 1999.

[63] L. Lamport, "How to Make a Multiprocessor Computer That Correctly Executes Multiprocess Programs", *IEEE Transactions on Computers C-28*, vol. 9, Sep. 1979.

[64] A. Yao, S. Strombeck, and J. Chi, "Development of a mobile manufacturing system with pda and plc", *The International Journal of Advanced Manufacturing Technology*, vol. 25, no. 7-8, 2005.

[65] V. Nahavandipoor, *Concurrent Programming in Mac OS X and iOS: Unleash Multicore Performance with Grand Central Dispatch*. O'Reilly Media, Inc., 2011.

Chapter A

Food Category System

ANNEX B

FOOD CATEGORY SYSTEM

PART I: Food Category System

01.0 Dairy products and analogues, excluding products of food category 02.0

- 01.1 Fluid Milk and Milk Products
  - 01.1.1 Fluid Milk (plain)
  - 01.1.2 Other Fluid Milk (plain)
  - 01.1.3 Fluid Buttermilk (plain)
  - 01.1.4 Flavoured Fluid Milk Drinks
- 01.2 Fermented and renneted milk products (plain),
  - 01.2.1 Fermented milks (plain)
    - 01.2.1.1 Fermented milks (plain), not heat-treated after fermentation
    - 01.2.1.2 Fermented milks (plain), heat-treated after fermentation
  - 01.2.2 Renneted milk (plain)
- 01.3 Condensed milk and analogues (plain)
  - 01.3.1 Condensed milk (plain)
  - 01.3.2 Beverage whiteners
- 01.4 Cream (plain) and the like
  - 01.4.1 Pasteurized cream (plain)
  - 01.4.2 Sterilized and UHT creams, whipping and whipped creams, and reduced fat creams (plain)
  - 01.4.3 Clotted cream (plain)
  - 01.4.4 Cream analogues
- 01.5 Milk powder and cream powder and powder analogues (plain)
  - 01.5.1 Milk powder and cream powder (plain)
  - 01.5.2 Milk and cream powder analogues
- 01.6 Cheese and analogues
  - 01.6.1 Unripened cheese
  - 01.6.2 Ripened cheese
    - 01.6.2.1 Ripened cheese, includes rind
    - 01.6.2.2 Rind of ripened cheese
    - 01.6.2.3 Cheese powder (for reconstitution; e.g. for cheese sauces)
  - 01.6.3 Whey cheese
  - 01.6.4 Processed cheese
    - 01.6.4.1 Plain processed cheese
    - 01.6.4.2 Flavoured processed cheese, including containing fruit, vegetables, meat, etc.
  - 01.6.5 Cheese analogues
  - 01.6.6 Whey protein cheese
- 01.7 Dairy-based desserts (e.g. pudding, fruit or flavoured yoghurt)
- 01.8 Whey and whey products, excluding whey cheeses
  - 01.8.1 Liquid whey and whey products, excluding whey cheeses

- 01.8.2 Dried whey and whey products, excluding whey cheeses

02.0 Fats and oils, and fat emulsions

- 02.1 Fats and oils essentially free from water
  - 02.1.1 Butter oil, anhydrous milkfat, ghee
  - 02.1.2 Vegetable oils and fats
  - 02.1.3 Lard, tallow, fish oil, and other animal fats
- 02.2 Fat emulsions mainly of type water-in-oil
  - 02.2.1 Butter
  - 02.2.2 Fat spreads, dairy fat spreads and blended spreads
- 02.3 Fat emulsions mainly of type oil-in-water, including mixed and/or flavoured products based on fat emulsions
- 02.4 Fat-based desserts excluding dairy-based dessert products of food category 01.7

03.0 Edible ices, including sherbet and sorbet

04.0 Fruits and vegetables (including mushrooms and fungi, roots and tubers, pulses and legumes, and aloe vera), seaweeds, and nuts and seeds

- 04.1 Fruit
  - 04.1.1 Fresh fruit
    - 04.1.1.1 Untreated fresh fruit
    - 04.1.1.2 Surface-treated fresh fruit
    - 04.1.1.3 Peeled or cut fresh fruit
  - 04.1.2 Processed fruit
    - 04.1.2.1 Frozen fruit
    - 04.1.2.2 Dried fruit
    - 04.1.2.3 Fruit in vinegar, oil, or brine
    - 04.1.2.4 Canned or bottled (pasteurized) fruit
    - 04.1.2.5 Jams, jellies, marmalades
    - 04.1.2.6 Fruit-based spreads (e.g. chutney) excluding products of food category 04.1.2.5
    - 04.1.2.7 Candied fruit
    - 04.1.2.8 Fruit preparations, including pulp, purees, fruit toppings and coconut milk
    - 04.1.2.9 Fruit-based desserts, incl. fruit-flavoured water-based desserts
    - 04.1.2.10 Fermented fruit products
    - 04.1.2.11 Fruit fillings for pastries
    - 04.1.2.12 Cooked fruit
- 04.2 Vegetables (including mushrooms and fungi, roots and tubers, pulses and legumes, and aloe vera), seaweeds, and nuts and seeds
  - 04.2.1 Fresh vegetables, (including mushrooms and fungi, roots and tubers, pulses and legumes, and aloe vera), seaweeds and nuts and seeds
    - 04.2.1.1 Untreated fresh vegetables, (including mushrooms and fungi, roots and tubers, pulses and legumes (including soybeans), and aloe vera), seaweeds and nuts and seeds
    - 04.2.1.2 Surface-treated fresh vegetables, (including mushrooms and fungi, roots and tubers, pulses and legumes, and aloe vera), seaweeds and nuts and seeds

04.2.1.3 Peeled, cut or shredded fresh vegetables, (including mushrooms and fungi, roots and tubers, pulses and legumes, and aloe vera), seaweeds and nuts and seeds

04.2.2 Processed vegetables (including mushrooms and fungi, roots and tubers, pulses and legumes, and aloe vera), seaweeds, and nuts and seeds

04.2.2.1 Frozen vegetables (including mushrooms and fungi, roots and tubers, pulses and legumes, and aloe vera), seaweeds and nuts and seeds

04.2.2.2 Dried vegetables (including mushrooms and fungi, roots and tubers, pulses and legumes, and aloe vera), seaweeds, and nuts and seeds

04.2.2.3 Vegetables (including mushrooms and fungi, roots and tubers, pulses and legumes, and aloe vera), and seaweeds in vinegar, oil, brine, or soybean sauce

04.2.2.4 Canned or bottled (pasteurized) or retort pouch vegetables (including mushrooms and fungi, roots and tubers, pulses and legumes, and aloe vera), and seaweeds

04.2.2.5 Vegetable (including mushrooms and fungi, roots and tubers, pulses and legumes, and aloe vera), seaweed, and nut and seed purees and spreads (e.g. peanut butter)

04.2.2.6 Vegetable (including mushrooms and fungi, roots and tubers, pulses and legumes, and aloe vera), seaweed, and nut and seed pulps and preparations (e.g. vegetable desserts and sauces, candied vegetables) other than food category 04.2.2.5

04.2.2.7 Fermented vegetable (including mushrooms and fungi, roots and tubers, pulses and legumes, and aloe vera) and seaweed products, excluding fermented soybean products of food categories 06.8.6, 06.8.7, 12.9.1, 12.9.2.1 and 12.9.2.3

04.2.2.8 Cooked or fried vegetables (including mushrooms and fungi, roots and tubers, pulses and legumes, and aloe vera), and seaweeds

05.0 Confectionery

05.1 Cocoa products and chocolate products including imitations and chocolate substitutes

05.1.1 Cocoa mixes (powders) and cocoa mass/cake

05.1.2 Cocoa mixes (syrups)

05.1.3 Cocoa-based spreads, incl. fillings

05.1.4 Cocoa and chocolate products

05.1.5 Imitation chocolate, chocolate substitute products

05.2 Confectionery including hard and soft candy, nougats, etc. other than food categories 05.1, 05.3, and 05.4

05.2.1 Hard candy

05.2.2 Soft candy

05.2.3 Nougats and marzipans

05.3 Chewing gum

05.4 Decorations (e.g. for fine bakery wares), toppings (non-fruit), and sweet sauces

06.0 Cereals and cereal products, derived from cereal grains, from roots and tubers, pulses, legumes and pith or soft core of palm tree, excluding bakery wares of food category 07.0

06.1 Whole, broken, or flaked grain, including rice

06.2 Flours and starches (including soybean powder)

06.2.1 Flours

06.2.2 Starches

- 06.3 Breakfast cereals, including rolled oats
- 06.4 Pastas and noodles and like products (e.g. rice paper, rice vermicelli, soybean pastas and noodles)
  - 06.4.1 Fresh pastas and noodles and like products
  - 06.4.2 Dried pastas and noodles and like products
  - 06.4.3 Pre-cooked pastas and noodles and like products
- 06.5 Cereal and starch based desserts (e.g. rice pudding, tapioca pudding)
- 06.6 Batters (e.g. for breading or batters for fish or poultry)
- 06.7 Pre-cooked or processed rice products, including rice cakes (Oriental type only)
- 06.8 Soybean products (excluding soybean-based seasonings and condiments of food category 12.9)
  - 06.8.1 Soybean-based beverages
  - 06.8.2 Soybean-based beverage film
  - 06.8.3 Soybean curd (tofu)
  - 06.8.4 Semi-dehydrated soybean curd
    - 06.8.4.1 Thick gravy-stewed semi-dehydrated soybean curd
    - 06.8.4.2 Deep fried semi-dehydrated soybean curd
    - 06.8.4.3 Semi-dehydrated soybean curd, other than food categories 06.8.4.1 and 06.8.4.2
  - 06.8.5 Dehydrated soybean curd (kori tofu)
  - 06.8.6 Fermented soybeans (e.g. natto, tempe)
  - 06.8.7 Fermented soybean curd
  - 06.8.8 Other soybean protein products

07.0 Bakery wares

- 07.1 Bread and ordinary bakery wares and mixes
  - 07.1.1 Breads and rolls
    - 07.1.1.1 Yeast-leavened breads and specialty breads
    - 07.1.1.2 Soda breads
  - 07.1.2 Crackers, excluding sweet crackers
  - 07.1.3 Other ordinary bakery products (e.g. bagels, pita, English muffins)
  - 07.1.4 Bread-type products, including bread stuffing and bread crumbs
  - 07.1.5 Steamed breads and buns
  - 07.1.6 Mixes for bread and ordinary bakery wares
- 07.2 Fine bakery wares (sweet, salty, savoury) and mixes
  - 07.2.1 Cakes, cookies and pies (e.g. fruit-filled or custard types)
  - 07.2.2 Other fine bakery products (e.g. doughnuts, sweet rolls, scones, and muffins)
  - 07.2.3 Mixes for fine bakery wares (e.g. cakes, pancakes)

08.0 Meat and meat products, including poultry and game

- 08.1 Fresh meat, poultry, and game
  - 08.1.1 Fresh meat, poultry and game, whole pieces or cuts
  - 08.1.2 Fresh meat, poultry and game, comminuted
- 08.2 Processed meat, poultry, and game products in whole pieces or cuts

08.2.1 Non-heat treated processed meat, poultry, and game products in whole pieces or cuts

08.2.1.1 Cured (including salted) non-heat treated processed meat, poultry, and game products in whole pieces or cuts

08.2.1.2 Cured (including salted) and dried non-heat treated processed meat, poultry, and game products in whole pieces or cuts

08.2.1.3 Fermented non-heat treated processed meat, poultry, and game products in whole pieces or cuts

08.2.2 Heat-treated processed meat, poultry, and game products in whole pieces or cuts

08.2.3 Frozen processed meat, poultry and game products in whole pieces or cuts

08.3 Processed comminuted meat, poultry, and game products

08.3.1 Non-heat treated processed comminuted meat, poultry, and game products

08.3.1.1 Cured (including salted) non-heat treated processed comminuted meat, poultry, and game products

08.3.1.2 Cured (including salted) and dried non-heat treated processed comminuted meat, poultry, and game products

08.3.1.3 Fermented non-heat treated processed comminuted meat, poultry, and game products

08.3.2 Heat-treated processed comminuted meat, poultry, and game products

08.3.3 Frozen processed comminuted meat, poultry, and game products

08.4 Edible casings (e.g. sausage casings)

09.0 Fish and fish products, including molluscs, crustaceans, and echinoderms

09.1 Fresh fish and fish products, including molluscs, crustaceans, and echinoderms

09.1.1 Fresh fish

09.1.2 Fresh molluscs, crustaceans, and echinoderms

09.2 Processed fish and fish products, including molluscs, crustaceans, and echinoderms

09.2.1 Frozen fish, fish fillets, and fish products, including molluscs, crustaceans, and echinoderms

09.2.2 Frozen battered fish, fish fillets and fish products, including molluscs, crustaceans, and echinoderms

09.2.3 Frozen minced and creamed fish products, including molluscs, crustaceans, and echinoderms

09.2.4 Cooked and/or fried fish and fish products, including molluscs, crustaceans, and echinoderms

09.2.4.1 Cooked fish and fish products

09.2.4.2 Cooked molluscs, crustaceans, and echinoderms

09.2.4.3 Fried fish and fish products, including molluscs, crustaceans, and echinoderms

09.2.5 Smoked, dried, fermented, and/or salted fish and fish products, including molluscs, crustaceans, and echinoderms

09.3 Semi-preserved fish and fish products, including molluscs, crustaceans, and echinoderms

09.3.1 Fish and fish products, including molluscs, crustaceans, and echinoderms, marinated and/or in jelly

09.3.2 Fish and fish products, including molluscs, crustaceans and echinoderms, pickled and/or in brine

09.3.3 Salmon substitutes, caviar and other fish roe products

09.3.4 Semi-preserved fish and fish products, including molluscs, crustaceans and echinoderms (e.g. fish paste), excluding products of food categories 09.3.1 - 09.3.3

09.4 Fully preserved, including canned or fermented fish and fish products, including molluscs, crustaceans, and echinoderms

10.0 Eggs and egg products

10.1 Fresh eggs

10.2 Egg products

10.2.1 Liquid egg products

10.2.2 Frozen egg products

10.2.3 Dried and/or heat coagulated egg products

10.3 Preserved eggs, including alkaline, salted, and canned eggs

10.4 Egg-based desserts (e.g. custard)

11.0 Sweeteners, including honey

11.1 Refined and raw sugars

11.1.1 White sugar, dextrose anhydrous, dextrose monohydrate, fructose

11.1.2 Powdered sugar, powdered dextrose

11.1.3 Soft white sugar, soft brown sugar, glucose syrup, dried glucose syrup, raw cane sugar

11.1.3.1 Dried glucose syrup used to manufacture sugar confectionery

11.1.3.2 Glucose syrup used to manufacture sugar confectionery

11.1.4 Lactose

11.1.5 Plantation or mill white sugar

11.2 Brown sugar excluding products of food category 11.1.3

11.3 Sugar solutions and syrups, also (partially) inverted, including treacle and molasses, excluding products of food category 1.1.1.3

11.4 Other sugars and syrups (e.g. xylose, maple syrup, sugar toppings)

11.5 Honey

11.6 Table-top sweeteners, including those containing high-intensity sweeteners

12.0 Salts, spices, soups, sauces, salads and protein products

12.1 Salt and salt substitutes

12.1.1 Salt

12.1.2 Salt substitutes

12.2 Herbs, spices, seasonings, and condiments (e.g. seasoning for instant noodles)

12.2.1 Herbs and spices

12.2.2 Seasonings and condiments

12.3 Vinegars

12.4 Mustards

12.5 Soups and broths

12.5.1 Ready-to-eat soups and broths, including canned, bottled, and frozen

12.5.2 Mixes for soups and broths

12.6 Sauces and like products

12.6.1 Emulsified sauces and dips (e.g. mayonnaise, salad dressing, onion dips)

12.6.2 Non-emulsified sauces (e.g. ketchup, cheese sauce, cream sauce, brown gravy)

12.6.3 Mixes for sauces and gravies

12.6.4 Clear sauces (e.g. fish sauce)

12.7 Salads (e.g. macaroni salad, potato salad) and sandwich spreads excluding cocoa-and nut-based spreads of food categories 04.2.2.5 and 05.1.3

12.8 Yeast and like products

12.9 Soybean-based seasonings and condiments

12.9.1 Fermented soybean paste (e.g. miso)

12.9.2 Soybean sauce

12.9.2.1 Fermented soybean sauce

12.9.2.2 Non-fermented soybean sauce

12.9.2.3 Other soybean sauces

12.10 Protein products other than from soybeans

13.0 Foodstuffs intended for particular nutritional uses

13.1 Infant formulae, follow-on formulae, and formulae for special medical purposes for infants

13.1.1 Infant formulae

13.1.2 Follow-up formulae

13.1.3 Formulae for special medical purposes for infants

13.2 Complementary foods for infants and young children

13.3 Dietetic foods intended for special medical purposes (excluding products of food category 13.1)

13.4 Dietetic formulae for slimming purposes and weight reduction

13.5 Dietetic foods (e.g. supplementary foods for dietary use) excluding products of food categories 13.1- 13.4 and 13.6

13.6 Food supplements

14.0 Beverages, excluding dairy products

14.1 Non-alcoholic ("soft") beverages

14.1.1 Waters

14.1.1.1 Natural mineral waters and source waters

14.1.1.2 Table waters and soda waters

14.1.2 Fruit and vegetable juices

14.1.2.1 Fruit juice

14.1.2.2 Vegetable juice

14.1.2.3 Concentrates for fruit juice

14.1.2.4 Concentrates for vegetable juice

14.1.3 Fruit and vegetable nectars

14.1.3.1 Fruit nectar

14.1.3.2 Vegetable nectar

14.1.3.3 Concentrates for fruit nectar

14.1.3.4 Concentrates for vegetable nectar

14.1.4 Water-based flavoured drinks, including "sport," “energy,” or "electrolyte" drinks and particulated drinks

14.1.4.1 Carbonated water-based flavoured drinks

14.1.4.2 Non-carbonated water-based flavoured drinks, including punches and ades

14.1.4.3 Concentrates (liquid or solid) for water-based flavoured drinks

14.1.5 Coffee, coffee substitutes, tea, herbal infusions, and other hot cereal and grain beverages, excluding cocoa

14.2 Alcoholic beverages, including alcohol-free and low-alcoholic counterparts

14.2.1 Beer and malt beverages

14.2.2 Cider and perry

14.2.3 Grape wines

14.2.3.1 Still grape wine

14.2.3.2 Sparkling and semi-sparkling grape wines

14.2.3.3 Fortified grape wine, grape liquor wine, and sweet grape wine

14.2.4 Wines (other than grape)

14.2.5 Mead

14.2.6 Distilled spirituous beverages containing more than 15% alcohol

14.2.7 Aromatized alcoholic beverages (e.g. beer, wine and spirituous cooler-type beverages, low-alcoholic refreshers)

15.0 Ready-to-eat savouries

15.1 Snacks - potato, cereal, flour or starch based (from roots and tubers, pulses and legumes)

15.2 Processed nuts, including coated nuts and nut mixtures (with e.g. dried fruit)

15.3 Snacks - fish based

16. Prepared foods

Chapter B

Research Grant

INSTITUTO POLITÉCNICO DE BRAGANÇA

Bolsa de Investigação - Licenciatura

Encontra-se aberto concurso para a atribuição de 1 Bolsa de Investigação - Licenciatura no âmbito do Projeto NORTE-01-0145-FEDER-023289: DeCodE "Tornar os corantes e conservantes naturais numa alternativa real aos aditivos artificiais através de uma estratégia de informação aberta e investigação aplicada baseada na experiência".

Área Científica: Informática

Requisitos de admissão: Licenciatura em Engenharia Informática; adequação da formação e experiência do candidato aos objetivos do trabalho a desenvolver; experiência em linguagem de programação Swift para iOS, bases de dados NoSQL e serviços RESTful.
Será ainda valorizado o domínio da língua inglesa (escrita e comunicação oral).

Plano de trabalhos: O presente plano tem como principais tarefas: 1) implementação de uma base de dados sobre aditivos alimentares na categoria dos corantes, utilizando uma ferramenta de referência; 2) Desenvolvimento de um aplicativo móvel em Swift para iOS com acesso remoto por RESTful.

Legislação e regulamentação aplicável (Regime de atividade): Lei Nº. 40/2004, de 18 de Agosto (Estatuto do Bolseiro de Investigação Científica); Regulamento da Formação Avançada e Qualificação de Recursos Humanos no Regulamento de Bolsas e Investigação da Fundação para a Ciência e a Tecnologia em vigor. (http://www.fct.pt/apoios/bolsas/docs/RegulamentoBolsasFCT2015.pdf).

Local de trabalho: O trabalho será desenvolvido em Bragança, no Centro de Investigação de Montanha, sob a orientação dos Professores Rui Pedro Lopes e Pedro Bastos.

Duração da bolsa: A bolsa terá a duração de 12 meses, com início previsto para 1 de outubro de 2017.

Valor do subsídio de manutenção mensal: O montante da bolsa corresponde a € 745,00, conforme tabela de valores das bolsas atribuídas diretamente pela FCT, I.P. no País (http://www.fct.pt/apoios/bolsas/valores), a ser pago mensalmente por transferência bancária.

Métodos de seleção: Os métodos de seleção a utilizar serão os seguintes:
a) Avaliação curricular e sua adequação para o trabalho em causa (70%);
b) Experiência em programação em Swift para iOS e modelação e implementação de base de dados NoSQL (30%);

Os três melhores candidatos, de acordo com estes critérios, poderão ser convocados para entrevista presencial e a sua classificação será ponderada no item a).
Composição do Júri de Seleção: O júri é constituído por Prof. Doutora Isabel C.F.R. Ferreira (presidente), Prof. Doutor Rui Pedro Lopes (vogal) e Prof. Doutor Pedro Bastos (vogal). Suplentes: Doutora Lillian Barros (vogal) e Prof. Doutor Rui Abreu (vogal).

Forma de publicitação/notificação dos resultados: Os resultados finais da avaliação serão publicitados através de lista ordenada *por nota final obtida* afixada no Centro de Investigação de

Montanha do Instituto Politécnico de Bragança, sendo o(a) candidato(a) aprovado(a) notificado(a) através de correio eletrónico.

Prazo de candidatura e forma de apresentação das candidaturas: O concurso encontra-se aberto no período de 12 de agosto a 29 de agosto de 2017, inclusive.
As candidaturas devem ser formalizadas, obrigatoriamente por correio eletrónico, acompanhadas dos seguintes documentos: carta de candidatura indicando a motivação para se candidatar a esta bolsa, cópia do Bilhete de Identidade/Cartão de Cidadão/ documento de identificação, *Curriculum Vitae* detalhado, certificado de habilitações e outros documentos comprovativos considerados relevantes.

As candidaturas devem ser remetidas por correio eletrónico (exclusivamente), com aviso de leitura, para:
Prof. Isabel C.F.R. Ferreira
Email: iferreira@ipb.pt
Centro de Investigação de Montanha
Instituto Politécnico de Bragança
ESA-IPB, Campus de Santa Apolónia
5300-253, Bragança