uc3m | Universidad Carlos III de Madrid

University Degree in Computer Science and Engineering
2018-2019

Bachelor Thesis

"Development of a web application to facilitate access to clients for catering"

Eduardo Boronat López

Tutor

Telmo Zarraonandia Ayo

Leganés, 2019

ABSTRACT

A web application is going to be implemented and designed in this thesis. I called the application UCOME and it is a platform specially thought for companies that have catering services like restaurants or cafes; or even it can be adapted to any restaurant. The main idea is that users can seat wherever they want because the administrator of the application configured the scenario of his local before and then, users will be able to order food without calling the waiter through their laptops or mobile phones.

UCOME is developed with Spring Boot and it consists of three main pages:

- Administrator Page: the administrator of the application will be able to configure the scenario of the local, the login preferences (LDAP, Office 365…), the food menu, events, social networks, statistics, etc.
- Customers Page: the customers will choose the seat where they want to eat, and they will order the food.
- Employees Page: the employees of the local will receive the orders made by the customers and they will carry the food to their seats.

All the data is stored in a MySQL database which provides the necessary capacity for this application.

Keywords: Client-server systems; Web design; Web 2.0; Food industry.

DEDICATION

In these lines I would like to thank everyone who has helped me in this stage.

Thanks to my colleagues and friends who have made it easier and have supported me in every moment. Thanks to my family for being always at my side and thanks to all my professors who have taught me most of my knowledge, with special mention to my tutor Telmo Zarraonandia.

Thank you all.

CONTENIDO

LIST OF FIGURES

LIST OF TABLES

1 INTRODUCTION

1.1 Context

Nowadays web applications have become the most demanded alternative for software development by all types of companies. They are the technological basis of modern companies, both for small and for the largest multinationals, because they can solve business problems providing efficiency and productivity; or they create platforms in which the users and customers interact with the company in an agile and simple way. These applications have helped companies to automate and simplify processes; and to have a better contact with clients and suppliers of the company. In the same way they have improved these activities, they can also be used to improve the catering service, which is what this project is going to be focused on.

We must differentiate between web applications and websites. Websites are static and its objective is to provide information to the user, so there is no interaction between the user and the site. Web applications are dynamic and there is an interaction between the user and the application, so it is constantly changing. Therefore, websites are usually cheaper because less time is invested as they don’t use programming languages or databases; although it is true that nowadays most of websites are not totally static. For example, most of them have a form where the user sends a message to the administrators.

Web applications are used by accessing to a web server by means of the internet or an intranet through a browser. Therefore, it is not needed to install any application into the computer or smartphone because users connect to the server where the system is located.

Native apps are also in constant progress. They are installed into smartphones or tablets, so they are developed to be used in a particular platform or device. Both native apps and web apps have a lot of advantages, but it was chosen to make a web application because for a native app two implementations must be done, one for Android and another one for iOS. Then, development cost is bigger, while a single web application can be accessed from all devices if it is responsive. In addition, from the users’ point of view, web applications do not occupy memory on the phone and they are not needed to update. However, native apps can offer push notifications, that would be very useful for UCOME to keep users informed about upcoming events or new dishes, so it would be a good decision to consider as future work.

1.2 Motivation of Work

Every sector in the society faces daily the progress of the technological world, but in the catering sector there exist a lot of businesses with the traditional procedure of ordering at the bar or requesting to the waiter. We have the opportunity to adapt to new technologies and achieve greater results. Therefore, a system can be implemented to improve some problems that are faced by companies' cafeterias or cafes. For example, when the number of customers increases, the service gets worse and it is always difficult to locate who ordered the food. This system will allow users to make orders with their phones or laptops and the waiter will process them in a computer respecting the time of arrival. This will cause an improvement in the quality of the service, being more organized, effective and comfortable.

The possibility of choosing your seat from your place of work and to have your order in your seat when arriving; or having the statistics of your dishes, offers major advantages for both the clients and the company. Clients will have a greater level of satisfaction, while companies will have logistic improvements among others.

The knowledge acquired during the degree will be applied, especially programming, web design and project management aspects. All with a global business vision.

1.3 Goals

The purpose of this project is to create a complete system for the food orders management, as an alternative to the classical service that is currently being used through a waiter.

The main goal can be defined as the implementation of a project that improves the catering services in a restaurant/cafeteria providing a simple and intuitive interface for the user, so that all the process of ordering and delivering food is completed in the simplest way without requiring a technical profile to the user. In addition, the project will help to complement the knowledge and skills acquired in the degree on development of web applications, more specifically in the Spring Framework and its modules.

The objectives of the theoretical part are:

- Installation and configuration of Spring with Maven: how to use Spring, configuring it with Spring Boot and adding dependencies with Maven.
- Study MVC Framework of Spring: learn the architecture Model-View-Controller.
- Study Spring Security: learn how to provide both authentication and authorization to users in the application.
- Study of all the tools and APIs that are going to be used: Google API, Microsoft Graph, Azure Active Directory, PayPal API, LDAP, Moodle API, Pusher, etc.

- Study of the technologies never used until this project: Swing, AJAX, Thymeleaf.
- Study of the deployment of Spring Boot applications and deployment options on different platforms in a production environment.

The objective of the practical part is to apply all the concepts learned to be able to do in a satisfactory way the web application.

1.4 Structure of work

In the present work, each of the parts carried out will be explained, as well as the results obtained. The structure has been organized as detailed below, with the objective that the reader obtains an incremental knowledge of the design as the reader progresses through the project.

After this brief introduction, the contents of each chapter will be detailed below.

- **Chapter 1** Introduction: it contains a brief introduction about web applications, the motivation of the project, goals to be developed, how the work is structured, a list of abbreviations and the regulatory framework through which the work is governed. In the latter, the professional responsibilities and data protection laws will be explained.
- **Chapter 2** Related work: a complete study of the web evolution and the main web development technologies is carried out to reach a final decision of which technology is best suited to the project. After that, a software architecture decision will be taken, and the chosen technologies to be used will be explained in detail. Finally, the application will be compared with another one existing in the current market.
- **Chapter 3** System analysis: system functionalities according to the role; and functional and non-functional requirements will be described.
- **Chapter 4** System design: the structure of the project, the system architecture, the technologies used for the user interfaces design and the physical data design will be explained for a whole understanding of the system.
- **Chapter 5** Development: it is the most extensive part. All the technologies, programs, configurations made, APIs and libraries used are explained in detail.
- **Chapter 6** Socio-economic environment: the social, economic and environmental impact due to web applications is evaluated; and a detailed budget with the time invested and resources used in the project is calculated.
- **Chapter 7** Conclusions: it is concluded by providing a vision of the work done and detailing the goals achieved, as well as the improvements and future lines of work.

Finally, the bibliography used and the administrator', users' and employees' manuals as annexes will be defined in the final pages of the document. These annexes will help as a summary of all the functionalities that the application does and how the application looks like with screenshots of each section.

1.5 Regulatory framework

All works done on any topic present a political-legal environment that must be known. In this section the criteria that affects the development of the work will be described, as well as the standards applicable to it.

One of the topics to be discussed in this section is the professional responsibility known as Public Liability, which according to Article 1902 of the Civil Code for engineers, if any work, application or device which may harm society or may cause any damage is done, then the responsible party is obligated to repair the damage caused.

General Data Protection Regulation (GDPR) is the European regulation according to the natural people protection in the treatment of their personal data. It was applied on May 25th, 2018; but it entered into force May 25th, 2016. Any company belonging to the European Union must enforce this law. In Spain, it rendered the LOPD of 1999 obsolete, being replaced by the Organic Law 3/2018.

In the Organic Law 3/2018, 5th of December, is detailed how to protect the sensible data of the natural people. Those articles, as far as possible, will be respected in the application.

1.6 List of abbreviations

AAD	Azure Active Directory
AD	Active Directory
AJAX	Asynchronous JavaScript and XML
ApacheDS	Apache Directory Server
API	Application Programming Interface
ASCII	American Standard Code for Information Interchange
AWS	Amazon Web Services
CA	Certification Authority
CMS	Content Management System
CN	Common Name
CRUD	Create, Read, Update, Delete
CSS	Cascading Style Sheets
DAO	Data Access Object

DC	Domain Component
DN	Distinguished Name
GDPR	General Data Protection Regulation
GUI	Graphical User Interface
HTML	HyperText Markup Language
HTTP	HyperText Transfer Protocol
HTTPS	HyperText Transfer Protocol Secure
IDE	Integrated Development Environment
IIS	Internet Information Services
IMAP	Internet Message Access Protocol
iOS	iPhone Operating System
IoT	Internet of Things
JAR	Java Archive
Java EE	Java Enterprise Edition
JCP	Java Community Process
JNDI	Java Naming and Directory Interface
JPA	Java Persistence API
JRE	Java Runtime Environment
JS	JavaScript
JSON	JavaScript Object Notation
JSP	JavaServer Pages
JSTL	JSP Standard Tag Library
LDAP	Lightweight Directory Access Protocol
LDIF	LDAP Data Interchange Format
LMS	Learning Management System
LOPD	Ley Orgánica de Protección de Datos de Carácter Personal
MOODLE	Modular Object-Oriented Dynamic Learning Environment
MVC	Model-View-Controller

OAuth	Open Authorization
OGNL	Object-Graph Navigation Language
ORM	Object-Relational Mapping
OU	Organizational Unit
PHP	Hypertext Preprocessor
PKCS	Public-Key Cryptography Standards
POM	Project Object Model
POP	Post Office Protocol
REST	Representational State Transfer
RoR	Ruby on Rails
RSA	Rivest, Shamir, Adleman
SMTP	Simple Mail Transfer Protocol
SPI	Serial Peripheral Interface
SQL	Structured Query Language
SSL	Secure Sockets Layer
SSO	Single Sign-On
TLS	Transport Layer Security
UC3M	Universidad Carlos III de Madrid
UI	User Interface
UID	User Identifier
UPN	User Principal Name
URI	Uniform Resource Identifier
URL	Uniform Resource Locator
VAT	Value Added Tax
WAR	Web Application Archive
WWW	World Wide Web
XHTML	Extensible HyperText Markup Language
XML	Extensible Markup Language

2 RELATED WORK

The technologies used in web development have evolved a lot in the last decade. There are many applications, frameworks, libraries, architectures that are constantly improving with the release of new versions. This progress has also been produced in the administration of systems, hosting services, scalability techniques and monitoring.

This evolution has allowed developing websites and applications with very different technologies and tools.

In this section how web is evolving, and the most known technologies used for web development will be explained and compared to justify the decision of the chosen ones to be used in the project; and the software architecture based in microservices will be compared with the monolithic architecture. After that, the key points of Spring Framework and Maven will be explained and finally, the application will be compared with another one in the current market.

2.1 The Evolution of the Web

WWW (World Wide Web) or Web is an information system composed by interconnected hypertext documents that are accessible through the internet.

The web has been constantly evolving from its creation in the year 1991 until nowadays. In this period four stages of web can be well differentiated:

- Web 1.0: static unidirectional pages with an informative character where the user could not interact with.
- Web 2.0: social webs were developed where users could exchange information through blogs, forums and social networks. Internet became a global phenomenon.
- Web 3.0 or data web: it is an intelligent web that uses cloud services and web applications to provide services to the users. They can be accessed from almost any device with internet connection. In addition, semantic data is used to create an understandable language which allows more intelligent searches.
- Web 4.0: it is in progress. It is an open, connected, predictive and intelligent web thanks to deep learning and machine learning techniques. The systems will be able to process information in a similar way than a human and all the devices will be connected (IoT). This will lead us to situations in which for example the people can reserve a hotel room through a virtual assistant, who will know all our preferences and likes.

It is incredible how technology has advanced in the last twenty years and we cannot imagine what will come in the future.

2.2 Web development technologies

In the server-side, many technologies for developing web applications can be used. Foremost among them are Java EE, PHP, ASP.NET, Ruby on Rails or new technologies like Go or Node.js. All of them will be compared, drawing conclusions to finally make a final decision.

2.2.1 Java EE

Java EE is a programming platform on the basis of Java. It is standardized by JCP (Java Community Process), so providers must meet certain requirements. There are some independent implementations such as Spring or Struts, which do not follow Java EE specifications, but they include most of the solutions offered by Java EE and they allow a coupling with many other frameworks.

Every Java EE web application must be executed in an application server such as Glassfish, Tomcat, JBoss, Jetty, etc., while Spring is who joins the core APIs (Servlet, JPA, JMS…) together.

2.2.1.1 Spring vs Struts

The main differences between Spring and Struts can be seen in the following table: [1]

TABLE 2.1. COMPARATIVE ANALYSIS BETWEEN SPRING AND STRUTS

Characteristics	Spring	Struts
Architecture	Layered	Not layered
Framework	Lightweight	Heavyweight
Integrating	Easy integration with ORM and JDBC	Manual coding needed
Flexibility	Spring MVC is more flexible	Less flexible
Coupling	Loosely coupled	Tightly coupled

For developing an enterprise application, I would choose Spring because it is a loosely coupled framework, so it will make the application more reusable, distributed and robust.

2.2.2 PHP

PHP is a programming language suitable for web development and it is embedded into HTML. It is used to generate dynamic webpages and it is usually integrated with Apache and MySQL. There are a lot of frameworks for the web applications development such us Laravel, Symfony, Zend, Codelgniter and CakePHP.

A comparative table between Java and PHP is created to see which technology more advantages could have for the type of application thought.

TABLE 2.2. COMPARATIVE ANALYSIS BETWEEN PHP AND JAVA

Characteristics	**PHP**	**Java**
Costs	Cheaper	More expensive
Performance	Good	Good
Code	Interpreted	Compiled
Security	Less secure	More secure
Concurrency	Multiprocessing	Multithreading
Development time	Less	More
Portability	Portable	Portable
Support	Good	Good

Both technologies have competitive advantages and are very used, so they have a big community behind. It is difficult to decide in favor of one of them, but considering our project I would choose Java because I give more importance to the security and robustness than cost. In addition, the web application will use threads to control the time that the user is eating and to be listening to the database log file, as will explained throughout the document. Therefore, Java is a better approach because it will be faster as the memory of subprocesses is faster shared than the communication between processes.

2.2.3 ASP.NET

ASP.NET is a web application framework developed by Microsoft. ASP.NET is stuck on the Microsoft platform due to compatibility problems. It can only be developed in a computer with Microsoft Windows and at the time of deployment it only works in the Microsoft IIS server, which is a huge disadvantage for this project because it must focus on an easy deployment in the companies' servers and there are more popular ones than the IIS server like Apache. [2]

2.2.4 Ruby on Rails

RoR is a web application framework written in the high-level programming language Ruby. Its syntax is very simple and intuitive being a good option for beginners. [3]

RoR has some advantages over frameworks based on Java for the development of web applications. RoR uses less lines of code, so it allows improving bug fixing; but Java provides a better performance, it has a high popularity and the ability to migrate to any other app, database or platform that could be of interest in the project for the future.

2.2.5 Go and Node.js

In 2009 appeared two technologies: Go and Node.js. Go is a programming language developed by Google and Node.js is a JavaScript run-time environment. Both are the growing technologies that are gaining a lot of popularity nowadays among the developers and they could substitute the traditional server-side languages like Java and PHP in the future because both show a very good performance. For a web application Node.js could be a better choice than Go because Go is focusing more on concurrency and speed, so it is more used as a scripting language.

The drawback of these two technologies is that they have lack of maturity in the market and we have to make sure that the selected packages have a recent activity. In addition, there won’t be as much documentation and common errors that are solved in internet as Java.

2.2.6 Decision

We have seen that there are a lot of technologies for the development of web applications; and only the most popular ones have been explained. This does not mean that there are no more technologies that could fit into the project.

I cannot say that a technology is better than another, but I have considered the size of the project, its structure, its costs and the type of deployment desired; and from my point of view Java is the programming language that matches my business goals and requirements best. Therefore, I will use Spring as the framework for developing the web application.

2.3 Software Architecture

2.3.1 Monolithic Architecture

Monolithic applications are those in which the software is structured in very coupled functional groups, involving aspects related to the presentation, processing and storage of information. They are implemented within a single software component.

The first software applications used this architecture and even though new alternatives have been developed, monolithic applications continue being the most used because they are easy to develop, easy to deploy, efficient and fast in their execution and management, but as a consequence, they are less flexible to new work environments or types of applications.

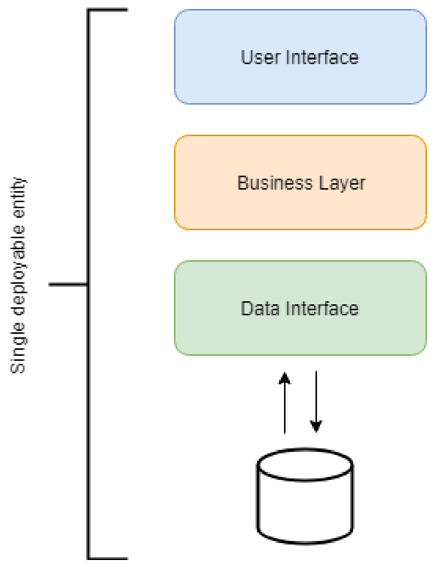


Fig. 2.1. Monolithic Architecture

2.3.2 Microservices Architecture

The purpose of this architecture is the development of the applications as sets of small services that are executed independently and are communicated through light mechanisms like HTTP or REST APIs.

These services are developed around business capabilities and can be implemented independently and completely automated. Services can be written in different programming languages and can use different data storage technologies.

Microservices adapt perfectly to the requirements of agility, scalability and reliability.

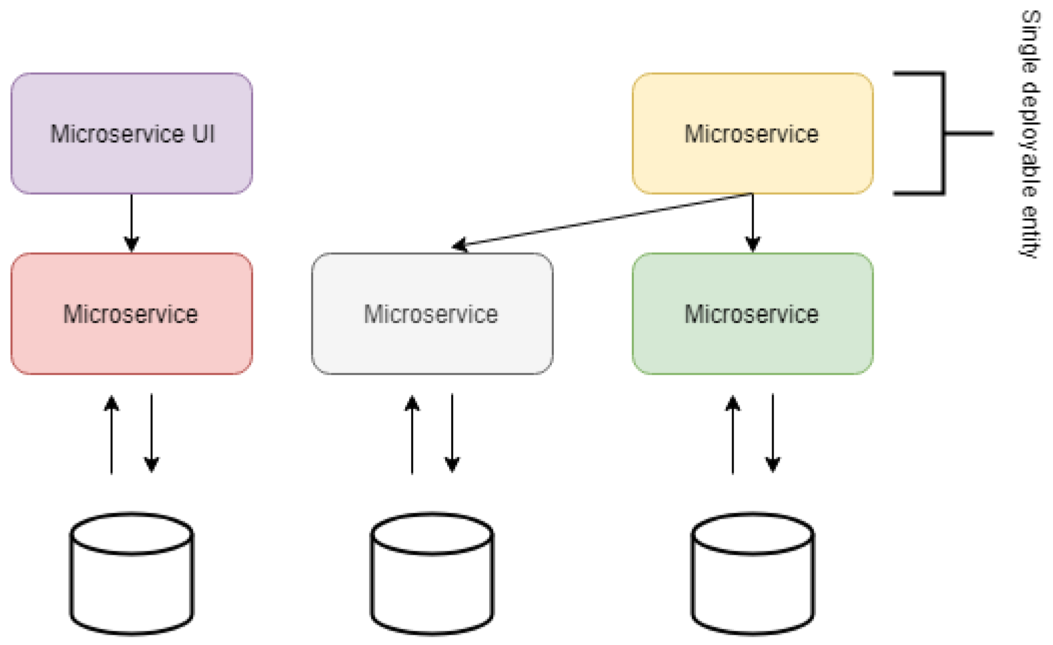


Fig. 2.2. Microservices Architecture

2.3.3 Comparison

Unlike the traditional monolithic applications, in which everything is integrated into a single piece, the microservices are independent and work together to carry out the same tasks.

In the table below, it is possible to observe the main differences between both architectures to understand how a monolithic application and one based on microservices affect the different aspects of software development. Finally, in the next section it will be explained which one is selected for this project considering the advantages and disadvantages.

TABLE 2.3. MONOLITHIC VS MICROSERVICES ARCHITECTURE

Characteristics	Monolithic	Microservices
Deployment	One single deployment	One deployment per microservice
Language	One programming language	Different programming languages
Understandable	Easy to understand	Difficult to understand as a whole, but easy to understand a service
Testing	Easy to test as a single unit	Distributed deployment could block tests
Maintenance	Easy to manage changes (up to a size)	Changes can be done quickly
Design	Centralized	Decentralized

Scalability	Reduce performance	Easy to scale
Data storage	One single database model	No one single database model
Development	Could become heavy	Continuous improvement
Team	Big	Small
Size	One big-size file	Some small-size files
Transactions	Simple	No transactions

2.3.4 Decision

After having seen the advantages and disadvantages of both architectures, the monolithic one was chosen. The reasons are:

- This web application will be sold to companies and restaurants, and it is good to have a single deployable file. It will be much easier, handy and fast to deploy it.
- This application is not a huge one and therefore, not too complex, so there is no need to divide it into microservices.
- One single database model is enough to handle all the data.
- It will not grow a lot. Maybe are added just some requests of the clients.

“Don’t even consider microservices unless you have a system that’s too complex to manage as a monolith. The majority of software systems should be built as a single monolithic application. Do pay attention to good modularity within that monolith, but don’t try to separate it into separate services”. [4]

“If you can’t build a well-structured monolith, what makes you think microservices is the answer?”. [5]

2.4 Spring Framework

2.4.1 Introduction

Spring is an open source framework used for the development of Java applications. Spring is divided into different modules and each one helps in different needs. Therefore, its modular aspect makes it flexible and configurable for any kind of application. [6]

2.4.2 Versions

The first version was written by Rod Johnson and was published in the book "Expert One-to-One J2EE Design and Development" in October 2002.

- The first version of Spring (1.0) was released in March 2004.
- Spring 2.0 was released in October 2006.
- Spring 3.0 was released in December 2009.
- Spring 4.0 was released in December 2013.

Each one has their respective subversions. Nowadays, the last stable version is Spring 5.0.0 and was released in September 2017, which include improvements like a JDK update, a Core update, functional programming with Kotlin, a reactive programming model and testing improvements because it supports JUnit 5.

2.4.3 Characteristics

The Spring modules can be summarized in the image below and each of them will be explained.

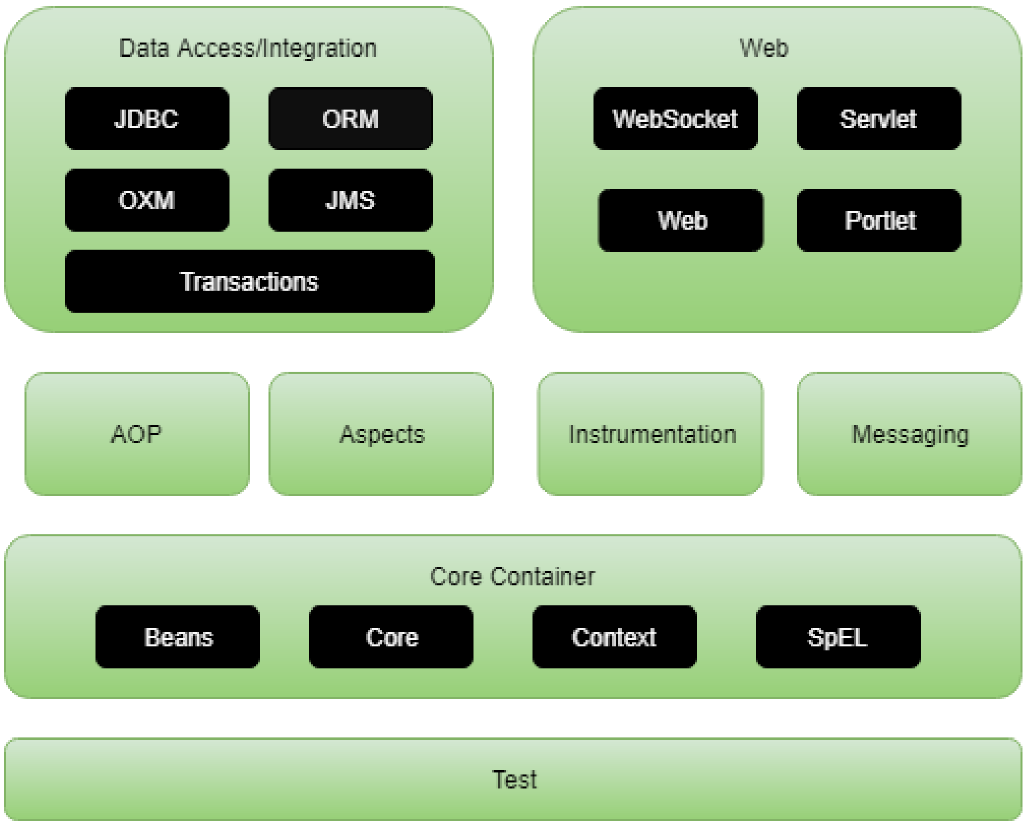


Fig. 2.3. Spring Framework Modules

In the Core Container is the Bean Factory, which is the main container of Spring. It is in charge of Dependency Injection. The Application Context is also located there. It is based on Bean Factory and extends its functionality with support for i18n, life cycle events and better integration with AOP. Finally, Expression Language module provides an expression language for querying.

AOP (Aspect Oriented Programming) allows to develop method-interceptors and pointcuts to cleanly decouple code of the transversal functionalities.

The Data Access/Integration layer consists of a JDBC-abstraction layer, the ORM module that provides integration layers for popular object-relational mapping APIs, the OXM module that supports Object/XML mapping implementations for JAXB, Castor, XMLBeans; JiBX and XStream; the JMS (Java Messaging Service) module that allows the exchange of messages and the Transactions module for the transaction management for classes that implement special interfaces.

Web layer provides special classes oriented to web development and integration with technologies like Struts and JSF. It contains the Spring MVC implementation used in this project.

Finally, the Test module allows testing the Spring components with JUnit or TestNG. [7]

2.4.4 Spring Boot

Spring Boot is a technology used inside the Spring Framework specially designed for the agile development of applications. As seen before, Spring has a lot of modules which involve many configurations. Spring Boot facilitates all these configurations, especially in an initial stage.

The main characteristics of Spring Boot are:

- It aims at facilitating creating Spring applications with minimum configuration.
- Microservices oriented.
- Works with Maven, Ivi and Gradle.
- It has an embedded Tomcat or Jetty, so the deployment is quite fast using an über-jar, which is a package of the application and its dependencies in a single and executable JAR.
- Eliminates possible conflicts related with the dependency’s versions thanks to the autoconfiguration.

In conclusion Spring Boot is an extension of Spring itself to make the development, testing, and deployment more convenient. [6]

2.4.5 Decision

Spring Boot has been used in this project for some reasons. First, it brings a lot of tools that eliminate some complexities and considerably facilitates the configuration of the project. Then, the embedded server is included, so it avoids complexity in application deployment. Mixing this with the decision of making a monolithic application, a really fast deployment is achieved, being good for clients; and it is also very fast to test when developing the application. Finally, using Maven it is quite simple to add the dependencies in a POM.xml.

2.5 Maven

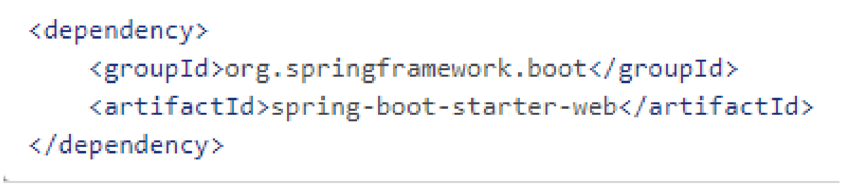
Maven was created in 2001 and it is an open source tool whose objective is to simplify build processes. Before Maven was invented, the developer had to spend a lot of time trying to understand how to compile and how to generate executables from source code. Each project used to have a developer just to configure the build process.

Maven builds the projects in some stages:

- Validate that the project is correct.
- Compile
- Test the source code using a unitary testing framework.
- Package the compiled code and transform it in .jar or .war format. UCOME is packaged in a .war.
- Integration test: process and deploy the code in an environment where integration tests can be executed.
- Verify that the packaged code is valid and fulfil quality requirements.
- Install the packaged code in the local repository of Maven, to use it in other projects as dependencies.
- Deploy the code in an environment.

Maven makes the dependencies management between modules and libraries versions very easy. Project modules must be set in a configuration file of the project called pom.xml. [8]

For starting using Spring Boot with Maven the most important dependency, which is written in the pom.xml is the following one:

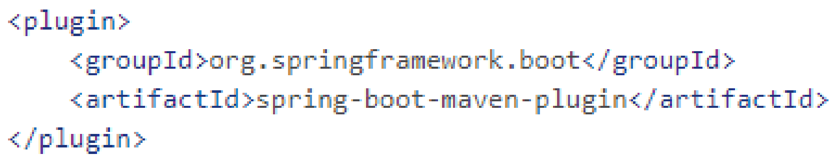


```
<dependency>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-web</artifactId>
</dependency>
```

Fig. 2.4. Spring Boot Dependency

With this dependency an embedded Tomcat will be used to deploy our application and it also provides spring web dependencies like spring MVC.

The Spring Boot Maven plugin will also be defined in the POM and when packaging the application, an über-jar will be created. It also searches the main class to run it.



```
<plugin>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-maven-plugin</artifactId>
</plugin>
```

Fig. 2.5. Spring Boot Plugin

As seen, configuring Spring Boot with Maven is very easy. When the packaging format is specified (war or jar), and some dependencies are set, the application can be deployed in the embedded Tomcat and can be accessed for example from `http://localhost:8080`. These are the basic dependency and plugin to deploy a very simple application, but of course, in this project have been used much more dependencies than here explained, which will be analysed later.

2.6 Similar applications in the current market

The most similar application found in the market is called "mr.noow" [9]. It is an iOS or Android app that shows you the nearest restaurants (around 40 in Madrid) and it is possible to buy the food from another place. When arriving to the restaurant, the food will be there waiting for you.

The web application developed in this project differs from it because it is more focused on companies' cafeterias and restaurants. Each one deploys the application in a server and it is more customizable. Single Sign-On are configurable depending on the login mechanisms used in the company (LDAP, Office365, Google, Aula Global in case of the UC3M). Then, UCOME is accessible from mobile phones and laptops through a browser without the need of downloading an app, while mr.noow is only accessible through the app. Even though, an app could be fine to develop for UCOME as future work.

UCOME allows users to see upcoming events and a photo of each dish, and to configure the restaurant scenario to allow users to choose the seat he/she wants. These functionalities are not possible with mr.noow.

3 SYSTEM ANALYSIS

3.1 System functionalities

The application will have different functionalities, which are divided according to the type of user accessing the application, so there are three different pages: administrator’s page, customers’ page and employees’ page.

In this section the functionalities will be defined in a summarized form for the reader to have a global vision of all the functionalities of the system, before getting into functional and non-functional tables.

3.1.1 Neither signed up nor logged in users features

Users who are in the index page will be able to sign up if they haven’t any account yet; or to log in through Office365, Google, Aula Global, LDAP or UCOME.

3.1.2 Administrator features

The administrator can:

- Configure the restaurant: introduce how many square meters the restaurant has, then a matrix will be created. Each clicked cell represents a seat.
- Register and delete employees. They will be able to access the application in the employees’ page.
- Register and delete events. They will be seen by the customers.
- Register and delete food dishes.
- Register social networks to be seen by customers.
- LDAP synchronization: auto registration and synchronization of users into UCOME from LDAP.
- See the statistics of the dishes.

3.1.3 Customers features

These users can:

- Modify and delete profile.
- See top dishes.
- Buy food dishes.
- See upcoming events.
- Access social networks.
- Reserve a seat or a table.

- Free seat.
- See shopping cart, delete dishes from the shopping cart and pay food dishes.
- See purchases.

3.1.4 Employees features

Employees will be able to:

- Receive orders in real time seeing in a visual way what ordered, who and where is seated.
- Modify profile.
- Access social networks.

3.2 Functional requirements

Functional requirements represent how the system should react when an event is triggered. They specify a behaviour that give the programmer the list of functionalities to be implemented and how they must work under certain conditions.

TABLE 3.1. FR 01 - CONFIGURE RESTAURANT SCENARIO

ID	FR-01
Name	Configure restaurant scenario
Trigger	The administrator enters the square meters and clicks on create/restart/accept buttons.
Prerequisites	The administrator must be authenticated. The administrator must be in the server-side.
Description	1. If “create”, the system should paint a matrix with the dimensions set and the user should be able to click on the squares. 2. If “restart” the system should delete the matrix, being possible to be recreated. 3. If “accept”, the system should save the matrix created.

TABLE 3.2. FR 02 - EMPLOYEES REGISTRATION

ID	FR-02
Name	Employees registration
Trigger	The administrator registers an employee through a form.
Prerequisites	The administrator must be authenticated. The employee’s email must not be already registered.
Description	1. The system should check for valid fields. 2. The system should save the employee into the database. 3. The system should send an email to the employee with the new password.

TABLE 3.3. FR 03 - EMPLOYEES DELETION

ID	FR-03
Name	Employees deletion
Trigger	The administrator deletes an employee.
Prerequisites	The administrator must be authenticated. Some registered employee.
Description	1. The system should delete the employee from the database.

TABLE 3.4. FR 04 - EVENTS REGISTRATION

ID	FR-04
Name	Events registration
Trigger	The administrator registers an event through a form.
Prerequisites	The administrator must be authenticated.
Description	1. The system should check for valid fields. 2. The system should save the event into the database.

TABLE 3.5. FR 05 - EVENTS DELETION

ID	FR-05
Name	Events deletion
Trigger	The administrator deletes an event.
Prerequisites	The administrator must be authenticated. Some registered event.
Description	1. The system should delete the event from the database.

TABLE 3.6. FR 06 - FOOD DISHES REGISTRATION

ID	FR-06
Name	Food dishes registration
Trigger	The administrator registers a dish through a form.
Prerequisites	The administrator must be authenticated.
Description	1. The system should check for valid fields. 2. The system should save the dish into the database.

TABLE 3.7. FR 07 - FOOD DISHES DELETION

ID	FR-07
Name	Food dishes deletion
Trigger	The administrator deletes a dish.
Prerequisites	The administrator must be authenticated. Some registered dish.
Description	1. The system should delete the dish from the database.

TABLE 3.8. FR 08 - SOCIAL NETWORKS REGISTRATION

ID	FR-08
Name	Social networks registration
Trigger	The administrator registers a social network.
Prerequisites	The administrator must be authenticated.
Description	1. The system should save the social network into the database. 2. The system should enable the social network's icon of the footers.

TABLE 3.9. FR 09 - LDAP SYNCHRONIZATION

ID	FR-09
Name	LDAP synchronization
Trigger	The administrator registers/synchronizes the users from an LDAP Directory into UCOME.
Prerequisites	The application.yaml file must be configured. The administrator must be authenticated.
Description	1. The system should save into the database the non-existing users and an email should be sent to them with the new generated password. 2. The system should update a user in the database if the LDAP attributes have been modified. 3. The system should not do anything if the user already exists in UCOME and its LDAP attributes have not been modified.

TABLE 3.10. FR 10 - STATISTICS CONTROL

ID	FR-10
Name	Statistics control
Trigger	The administrator clicks on the statistics button.
Prerequisites	The administrator must be authenticated. The administrator must be in the server-side.
Description	1. The system should show the quantity sold of every registered dish and the total revenues.

TABLE 3.11. FR 11 - FILTERING STATISTICS

ID	FR-11
Name	Filtering statistics
Trigger	The administrator filters by date the statistics.
Prerequisites	The administrator must be authenticated. The administrator must be in the server-side.
Description	1. The system should show the quantity sold of the dishes and the revenues in the date range specified.

TABLE 3.12. FR 12 - USER REGISTRATION

ID	FR-12
Name	User registration
Trigger	The user registers an account in UCOME through a form.
Prerequisites	The user's email must not be already registered.
Description	1. The system should check the fields. 2. The system should save the user into the database.

TABLE 3.13. FR 13 - SIGN IN WITH OFFICE 365

ID	FR-13
Name	Sign in with Office 365
Trigger	The user logs in through Office 365.
Prerequisites	The application.yaml file must be configured. The user must be registered. The scenario must be configured.
Description	1. The system should redirect to Microsoft login page. 2. The system should redirect to the UCOME user's account after a successful authentication.

TABLE 3.14. FR 14 - SIGN IN WITH GOOGLE

ID	FR-14
Name	Sign in with Google
Trigger	The user logs in through Google.
Prerequisites	The application.yaml file must be configured. The user must be registered. The scenario must be configured.
Description	1. The system should redirect to Google login page. 2. The system should redirect to the UCOME user's account after a successful authentication.

TABLE 3.15. FR 15 - SIGN IN WITH AULA GLOBAL

ID	FR-15
Name	Sign in with Aula Global
Trigger	The user logs in through Aula Global.
Prerequisites	The user must be registered. The scenario must be configured.
Description	1. The system should redirect to the UCOME user's account after a successful authentication using the Moodle API.

TABLE 3.16. FR 16 - SIGN IN WITH LDAP

ID	FR-16
Name	Sign in with LDAP
Trigger	The user logs in through the LDAP chosen.
Prerequisites	The ldaps.yaml file must be configured. The user must be registered. The scenario must be configured.
Description	1. The system should redirect to the UCOME user's account after a successful authentication in the LDAP Directory chosen.

TABLE 3.17. FR 17 - SIGN IN WITH UCOME

ID	FR-17
Name	Sign in with UCOME
Trigger	The user/administrator/employee logs in through UCOME.
Prerequisites	The user/admin/employee must be registered. The scenario must be configured.
Description	1. The system should redirect to the UCOME user's/administrator's/employees' account after a successful authentication.

TABLE 3.18. FR 18 - MODIFY PROFILE

ID	FR-18
Name	Modify profile
Trigger	The user/employee modifies his profile through a form.
Prerequisites	The user/employee must be authenticated.
Description	1. The system should check for valid fields. 2. The system should update the user/employee in the database.

TABLE 3.19. FR 19 - DELETE ACCOUNT

ID	FR-19
Name	Delete account
Trigger	The user deletes his account.
Prerequisites	The user must be authenticated.
Description	1. The system should delete the user from the database.

TABLE 3.20. FR 20 - SEE TOP DISHES

ID	FR-20
Name	See top dishes
Trigger	The user navigates to the top dishes section.
Prerequisites	The user must be authenticated.
Description	1. The system should show the top 8 most purchased dishes.

TABLE 3.21. FR 21 - SEE UPCOMING EVENTS

ID	FR-21
Name	See upcoming events
Trigger	The user navigates to the events section.
Prerequisites	The user must be authenticated.
Description	1. The system should show the events registered in the database.

TABLE 3.22. FR 22 - ACCESS SOCIAL NETWORKS

ID	FR-22
Name	Access social networks
Trigger	The user/administrator/employee clicks on the social network icon in the footer.
Prerequisites	The user/administrator/employee must be authenticated.
Description	1. The system should redirect the user/administrator/employee to the social network profile.

TABLE 3.23. FR 23 - SEAT RESERVATION

ID	FR-23
Name	Seat reservation
Trigger	The user reserves a seat.
Prerequisites	The user must be authenticated. The seat must be free. The user must not have already a seat reserved.
Description	1. The system should save the user's seat location in the database. 2. The system should mark the seat as occupied. 3. The system should allow the user to buy dishes.

TABLE 3.24. FR 24 - TABLE RESERVATION

ID	FR-24
Name	Table reservation
Trigger	The user reserves a table through a form.
Prerequisites	The user must be authenticated.
Description	1. The system should send an email to the administrator with the information.

TABLE 3.25. FR 25 - ADD DISHES TO CART

ID	FR-25
Name	Add dishes to cart
Trigger	The user adds food dishes to the shopping cart.
Prerequisites	The user must have a seat reserved.
Description	1. The system should add the dishes to the user's cart.

TABLE 3.26. FR 26 - REMOVE DISHES FROM THE CART

ID	FR-26
Name	Remove dishes from the cart
Trigger	The user removes food dishes from the shopping cart.
Prerequisites	The user must have a seat reserved.
Description	1. The system should remove the dishes from the user's cart.

TABLE 3.27. FR 27 - BUY THE DISHES OF THE CART

ID	FR-27
Name	Buy the dishes of the cart
Trigger	The user buys the dishes of the shopping cart through PayPal.
Prerequisites	The application.yaml file must be configured. The user must have a seat reserved.
Description	1. The system should redirect to the PayPal login page to proceed to the payment. 2. The system should save the purchase into the database. 3. The system should record the transaction into the database replication log file. 4. The system should read in real time the replication log file and send all the information of the purchase to the employees' page.

TABLE 3.28. FR 28 - FREE SEAT

ID	FR-28
Name	Free seat
Trigger	The user frees his seat.
Prerequisites	The user must have a seat reserved.
Description	1. The system should update the seat of the user in the database. 2. The system should mark the seat as free. 3. The system should ask the user to free his seat when he has been eating for thirty minutes and he didn't free his seat. If the user does not answer the message, he will have another opportunity to answer in ten minutes and if no answer is provided, the system should free his seat.

TABLE 3.29. FR 29 - SEE PURCHASES

ID	FR-29
Name	See purchases
Trigger	The user navigates to the purchases section.
Prerequisites	The user must be authenticated.
Description	2. The system should show all the purchases of the user taking them from the database.

3.3 Non-functional requirements

These requirements are those related with quality attributes that must be met so that the functionalities can be used correctly by the users, handling the application without problems. Alternatively, they define system restrictions.

TABLE 3.30. NFR 01 - INTERNET CONNECTION

ID	NFR-01
Name	Internet connection
Description	It is necessary to have internet connection to access the web application.

TABLE 3.31. NFR 02 - BROWSER

ID	NFR-02
Name	Browser
Description	It is necessary to have a web browser to access the web application.

TABLE 3.32. NFR 03 - BROWSERS COMPATIBILITY

ID	NFR-03
Name	Browsers compatibility
Description	The application will be compatible with Google Chrome, Firefox, Opera, Safari and Mozilla navigators.

TABLE 3.33. NFR 04 - SERVER

ID	NFR-04
Name	Server
Description	The application will be deployed in a server with enough memory to support a number of concurrent users of at least the number of seats in the restaurant.

TABLE 3.34. NFR 05 - FRAMEWORK

ID	NFR-05
Name	Framework
Description	The application will be developed with the Spring framework.

TABLE 3.35. NFR 06 - PROOGRAMMING LANGUAGE

ID	NFR-06
Name	Programming language
Description	The application will be developed with the programming language Java.

TABLE 3.36. NFR 07 - IDE

ID	NFR-07
Name	IDE
Description	The application will be developed in the Eclipse IDE.

TABLE 3.37. NFR 08 - DATABASE

ID	NFR-08
Name	Database
Description	The application will use MySQL as database.

TABLE 3.38. NFR 09 - MANUALS

ID	NFR-09
Name	Manuals
Description	A manual for the users, administrator and employees will be delivered.

TABLE 3.39. NFR 10 - EFFICIENCY

ID	NFR-10
Name	Efficiency
Description	The system will respond the 95% of the requests in less than 4 seconds.

TABLE 3.40. NFR 11 - APPEARANCE

ID	NFR-11
Name	Appearance
Description	The interface will be responsive and attractive.

TABLE 3.41. NFR 12 - USABILITY

ID	NFR-12
Name	Usability
Description	The system will be easy to use with simple and intuitive elements.

TABLE 3.42. NFR 13 - PORTABILITY

ID	NFR-13
Name	Portability
Description	The system can be transferred from its current hardware and software environment to another.

TABLE 3.43. NFR 14 - SECURITY

ID	NFR-14
Name	Security
Description	The system will have a login system in which users will have roles and each role offers some functionalities.

TABLE 3.44. NFR 15 - RELIABILITY AND AVAILABILITY

ID	NFR-15
Name	Reliability and availability
Description	The system will be available in the restaurant/cafeteria working hours.

TABLE 3.45. NFR 16 - INTEGRITY

ID	NFR-16
Name	Integrity
Description	The system will check the incorrect data inputs.

TABLE 3.46. NFR 17 - PRIVACY

ID	NFR-17
Name	Privacy
Description	The system will comply the current legislation.

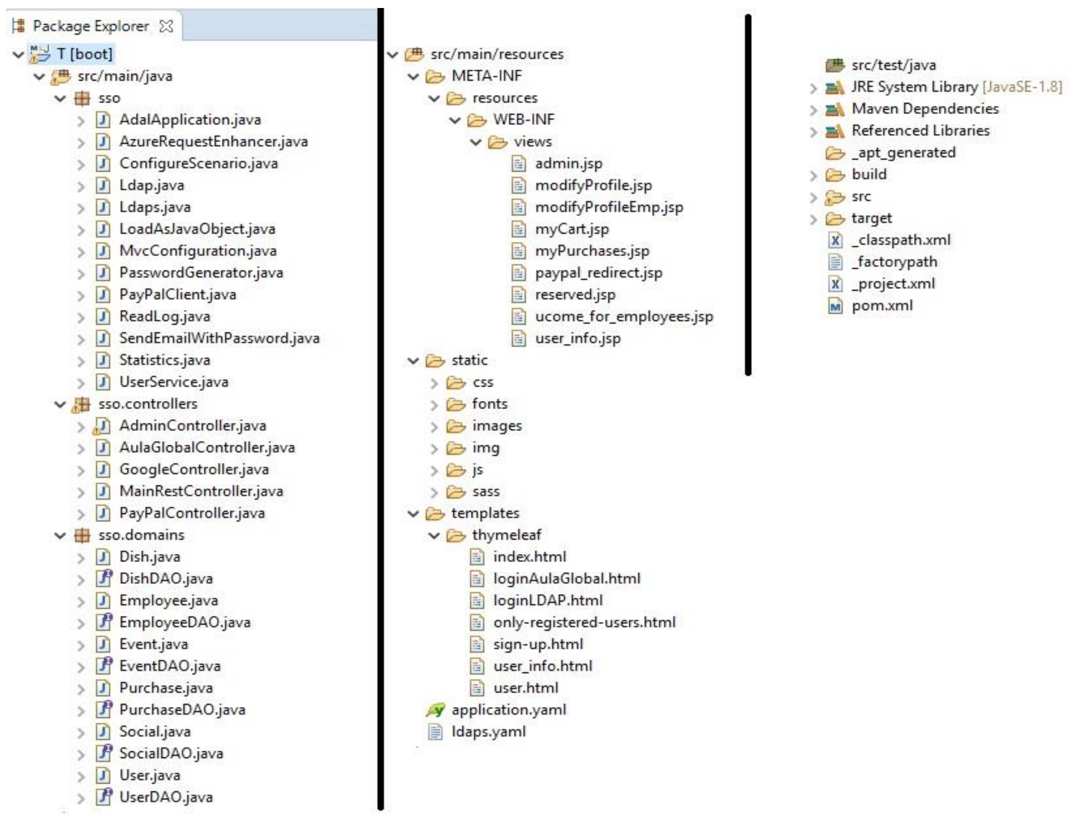
4 SYSTEM DESIGN

4.1 Project structure

The Spring Boot project is organized in folders. The `/src/main/java` folder contains all the java files separated by packages. The package `sso` has the main java classes and some needed configuration classes, `sso.controllers` has all the controllers that use the MVC pattern and `sso.domains` contains all the entities and interfaces used by Spring Data for the Object Relational Mapping (ORM). These concepts will be explained throughout the document.

Inside `/src/main/resources` folder are the views and thymeleaf folders, which include the views of the MVC pattern that are the interfaces seen by the users. Bootstrap files, images used in the application, stylesheets (files with `.css` extension) and JavaScript (with extension `.js`) files are in the static folder. It is worth highlighting the `application.yaml` and `ldaps.yaml` files, which will be the ones that the administrators of the application must edit for multiple purposes: single sign-on, LDAPs, database, payment, etc. configuration.

Finally, note the `pom.xml` file that is the main unit of a Maven project. It contains information about the project, sources, tests, dependencies, plugins, versions, etc.



```
Package Explorer
T [boot]
  src/main/java
    sso
      AdalApplication.java
      AzureRequestEnhancer.java
      ConfigureScenario.java
      Ldap.java
      Ldaps.java
      LoadAsJavaObject.java
      MvcConfiguration.java
      PasswordGenerator.java
      PayPalClient.java
      ReadLog.java
      SendEmailWithPassword.java
      Statistics.java
      UserService.java
    sso.controllers
      AdminController.java
      AulaGlobalController.java
      GoogleController.java
      MainRestController.java
      PayPalController.java
    sso.domains
      Dish.java
      DishDAO.java
      Employee.java
      EmployeeDAO.java
      Event.java
      EventDAO.java
      Purchase.java
      PurchaseDAO.java
      Social.java
      SocialDAO.java
      User.java
      UserDAO.java
  src/main/resources
    META-INF
      resources
        WEB-INF
          views
            admin.jsp
            modifyProfile.jsp
            modifyProfileEmp.jsp
            myCart.jsp
            myPurchases.jsp
            paypal_redirect.jsp
            reserved.jsp
            ucome_for_employees.jsp
            user_info.jsp
    static
      css
      fonts
      images
      img
      js
      sass
    templates
      thymeleaf
        index.html
        loginAulaGlobal.html
        loginLDAP.html
        only-registered-users.html
        sign-up.html
        user_info.html
        user.html
    application.yaml
    ldaps.yaml
  src/test/java
  JRE System Library [JavaSE-1.8]
  Maven Dependencies
  Referenced Libraries
  _apt_generated
  build
  src
  target
  _classpath.xml
  _factorypath
  _project.xml
  pom.xml
```

Fig. 4.1. Project Structure

4.2 System Architecture

4.2.1 Spring Web MVC

Spring Web MVC framework is the one that makes a separation between the web views and the Java application behind it, while communicating them with effectiveness and in a simple way. That is, it provides Model-View-Controller (MVC) architectural pattern. In addition, it integrates with other Spring Framework components than could be necessary. [10]

4.2.1.1 MVC pattern

Three elements are defined in this pattern: model, view and controller, related according to the following diagram:

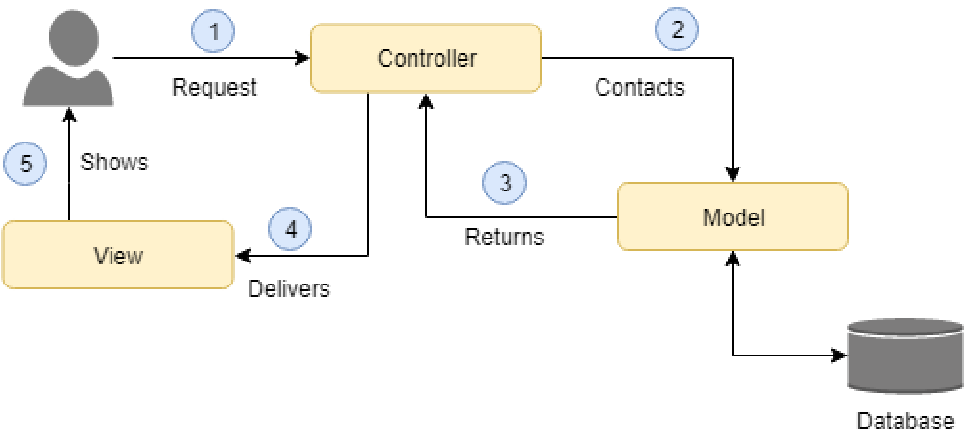


Fig. 4.2. MVC pattern components diagram

The functions of the three components are:

- Model: it contains the data to be displayed in the view. Everything related to the access and management of data is done by the model.
- View: graphical user interface. It displays the data obtained from the model to the user.
- Controller: it connects the model with the view. It is in charge of obtaining the data of the model to send it to the view. It also receives the events that the user generates in the view and performs the requested changes in the model.

4.2.1.2 MVC implementation in Spring

In the application, the views are `.html` (Thymeleaf) or `.jsp` files, the controllers are Java classes that receives HTTP requests and generates responses; and the model are the classes that contains the user requested data. The model will connect with the MySQL database.

Spring Web MVC is designed around a `DispatcherServlet` that handles the HTTP requests and responses.

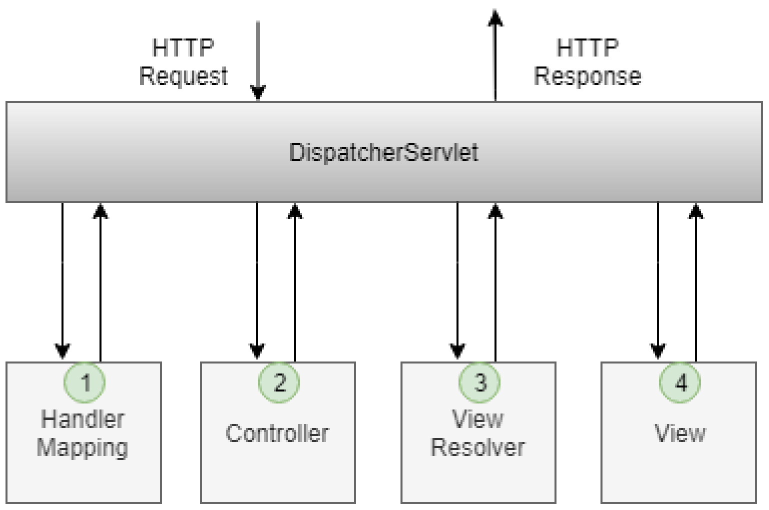


Fig. 4.3. Spring Web MVC diagram

The browser sends a request and the `DispatcherServlet` consults the `Handler Mapping` for calling the appropriate `Controller`. The `Controller` performs all the business logic and will return a view name to the `DispatcherServlet`, that will have to associate a view with the view name. This will be done with the help of the `View Resolver`, which must be edited because UCOME uses two technologies to render a view: Thymeleaf and JSP (explained in the next section). Finally, the `DispatcherServlet` sends the data model to the view and it is displayed on the browser to be seen by the user. [11]

4.3 User Interfaces Design

4.3.1 JSP and Thymeleaf

The application uses JSP and Thymeleaf. Thymeleaf is being used for the pages that the user sees before logging in UCOME, and JSP after logging. Both are technologies that can be used to render a view from Spring models.

4.3.1.1 JSP

JavaServer Pages (JSP) allows to divide a webpage into dynamic and static parts by combining dynamic code written in Java and static content written in HTML or XML. Therefore, it has a great compatibility with Spring. The Java code is embedded within the HTML with `<% … %>` tags and it is executed in the server side.

To execute JSP pages, it is needed a Web server with a Web container that comply with the JSP and Servlet specifications. The embedded Tomcat used by Spring Boot accomplish it.

JSP pages are files with the extension `.jsp` and is compiled into a Servlet the first time is accessed.

There are three ways for inserting Java code in a JSP page [12]:

- Expressions `<%= expression %>`: the expression is evaluated; its result is converted to String and it is inserted in the output.
- Scriptlets `<% code %>`: the code is executed inside the method `service()` of the servlet generated.
- Declarations `<%! code %>`: they are inserted in the generated servlet body, outside the methods.

4.3.1.2 Thymeleaf

Thymeleaf is a Java library which provides an XML/XHTML/HTML5 template engine that fits very well for working with MVC view layer of web applications. It allows to work with different expressions types [13]:

- Variables expressions: are the most used. Specified in OGNL (Object-Graph Navigation Language). Represented by `${…}`.
- Selection expressions: allows to reduce the expression length if an object is assigned through a variable expression. Represented by `*{…}`.
- Message expressions: to load messages and even perform the internalization of the application from properties or text files. Represented by `#{…}`.
- Link expressions: to create URLs that can have parameters or variables. Represented by `@{…}`.
- Fragments expressions: to divide the templates into smaller ones and to load them only when needed. Represented by `~{…}`.

4.3.1.3 Comparison

The advantages of JSP and Thymeleaf will be described in the following table being compared with the alternative technology.

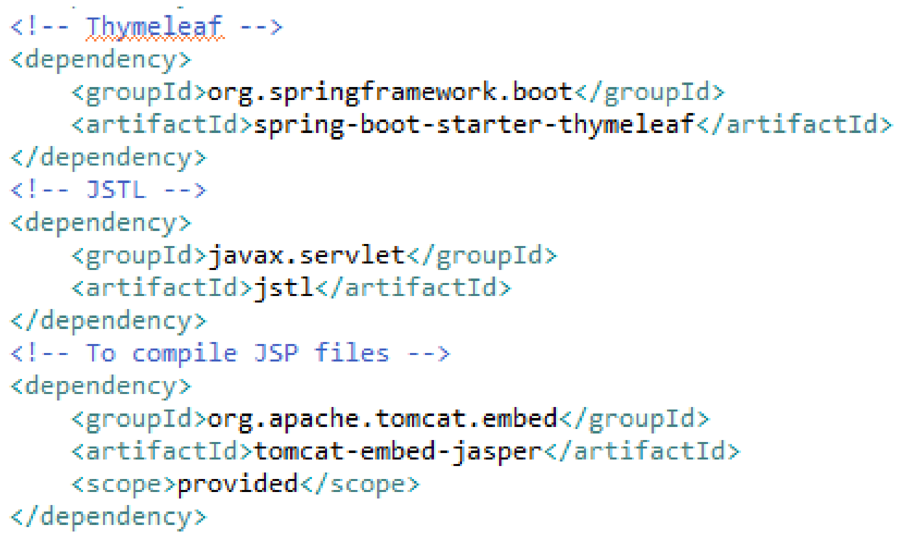
TABLE 4.1. JSP AND THYMELEAF ADVANTAGES

Advantages	
JSP	**Thymeleaf**
Older technology with a long way behind.	Prototype without executing the application.
Templates processing is faster than the Thymeleaf processing speed.	No need to re-deploy the application if templates are modified. In JSP it is needed to re-deploy.
Easier to learn if HTML and Java are already known.	If Thymeleaf is used with Spring, the Spring dialect is more powerful than the tags library of JSTL.

4.3.1.4 View Resolver

Maps view names to actual views, and it is enabled to render models in the browser. As the web application is using Thymeleaf and JSP simultaneously, it is needed to configure the `InternalResourceViewResolver` bean.

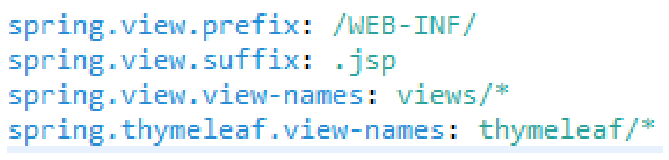
For that, the following dependencies must be added:



```
<!-- Thymeleaf -->
<dependency>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-thymeleaf</artifactId>
</dependency>
<!-- JSTL -->
<dependency>
    <groupId>javax.servlet</groupId>
    <artifactId>jstl</artifactId>
</dependency>
<!-- To compile JSP files -->
<dependency>
    <groupId>org.apache.tomcat.embed</groupId>
    <artifactId>tomcat-embed-jasper</artifactId>
    <scope>provided</scope>
</dependency>
```

Fig. 4.4. JSP and Thymeleaf dependencies

Then, in the application.yaml file set Thymeleaf view names and JSP configuration that will be used in a new configuration class.

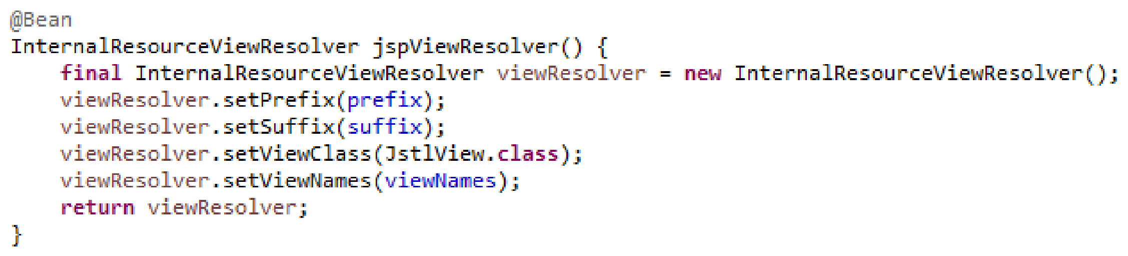


```
spring.view.prefix: /WEB-INF/
spring.view.suffix: .jsp
spring.view.view-names: views/*
spring.thymeleaf.view-names: thymeleaf/*
```

Fig. 4.5. Spring views configuration - application.yaml.

As seen in the Fig. 4.1., `jsp` files are in the `/WEB-INF/views` folder and Thymeleaf files (.html) in `/templates/thymeleaf` folder. Both in `/resources` folder.

Finally, the bean is configured in the new configuration class marked with the `@Configuration` annotation for the view resolution for JSP pages [14]:



```
@Bean
InternalResourceViewResolver jspViewResolver() {
    final InternalResourceViewResolver viewResolver = new InternalResourceViewResolver();
    viewResolver.setPrefix(prefix);
    viewResolver.setSuffix(suffix);
    viewResolver.setViewClass(JstlView.class);
    viewResolver.setViewNames(viewNames);
    return viewResolver;
}
```

Fig. 4.6. View resolution for JSP pages

Now, the controller can serve:

- JSP pages. Example:

  `return new ModelAndView("views/admin");`

- Thymeleaf pages. Example:

  `return new ModelAndView("thymeleaf/sign-up);`

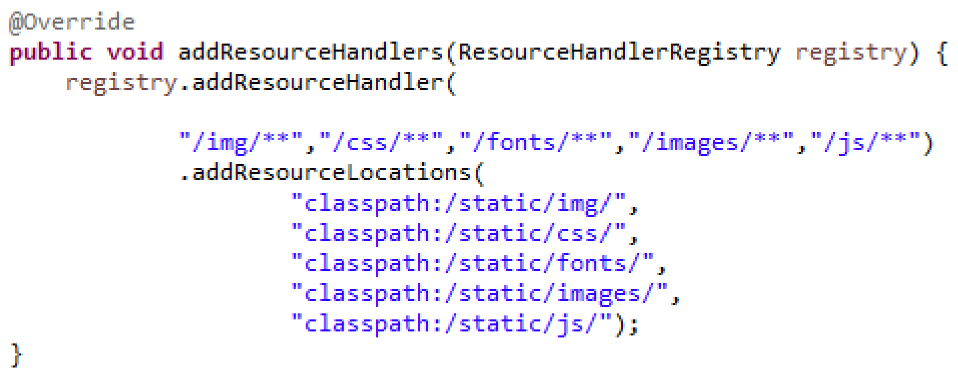
4.3.2 Bootstrap

Bootstrap is a CSS framework used to create user interfaces, which are most of them adaptable to any type of devices and screens. It simplifies the creation of web designs combining CSS, HTML and JavaScript. These characteristics make it an excellent option to develop responsive and mobile-first web applications. In addition, it is perfectly integrated with the main JavaScript libraries like JQuery, used in this project too. [15]

In the client side, Bootstrap is used as the Front-End because it was very important for UCOME application to be accessible not only through computers, but smartphones too, so that the seat reservation and purchases can be done from anywhere.

There are some free and premium templates for web designers and developers that are very useful to start designing the webpage or web application from a theme and not from scratch. In this application a free template called "Foodee" was used as a basis. [16]

To use Bootstrap inside the Spring Boot Project, the static content of the template such as CSS, JavaScript and images are placed inside the `src/main/resources/static` folder. As the `@EnableWebMvc` annotation was used in a `@Configuration` class for using Thymeleaf and JSP together as explained in the previous section, Spring Boot's MVC auto-configuration was disabled. This means that Spring Boot MVC must be manually configured to serve static resources.



```java
@Override
public void addResourceHandlers(ResourceHandlerRegistry registry) {
    registry.addResourceHandler(

            "/img/**","/css/**","/fonts/**","/images/**","/js/**")
            .addResourceLocations(
                    "classpath:/static/img/",
                    "classpath:/static/css/",
                    "classpath:/static/fonts/",
                    "classpath:/static/images/",
                    "classpath:/static/js/");
}
```

Fig. 4.7. Static resources handler configuration

4.4 Physical Database Design

4.4.1 MySQL

MySQL is the open source relational database management system used in this project. It is multithread and multiuser allowing to be used by some people at the same time, even perform queries at the same time.

The packages used are MySQL Community Server and MySQL Workbench:

- MySQL Workbench is the graphical tool to manage or create databases. [17]
- MySQL Community Server is the database engine. [18]

4.4.1.1 Database structure

The database consists of 7 tables. These tables contain the information of the users, the purchases, the dishes, the social networks, the events and the employees.

We have a many-to-many relationship between the purchases and the dishes, since each purchase can have multiple dishes, and each dish can be in multiple purchases. Therefore, a junction table called `purchase_dish` is made, so we could see all the dishes purchased by a user or how many times a dish was purchased. This will be used for calculating the statistics.

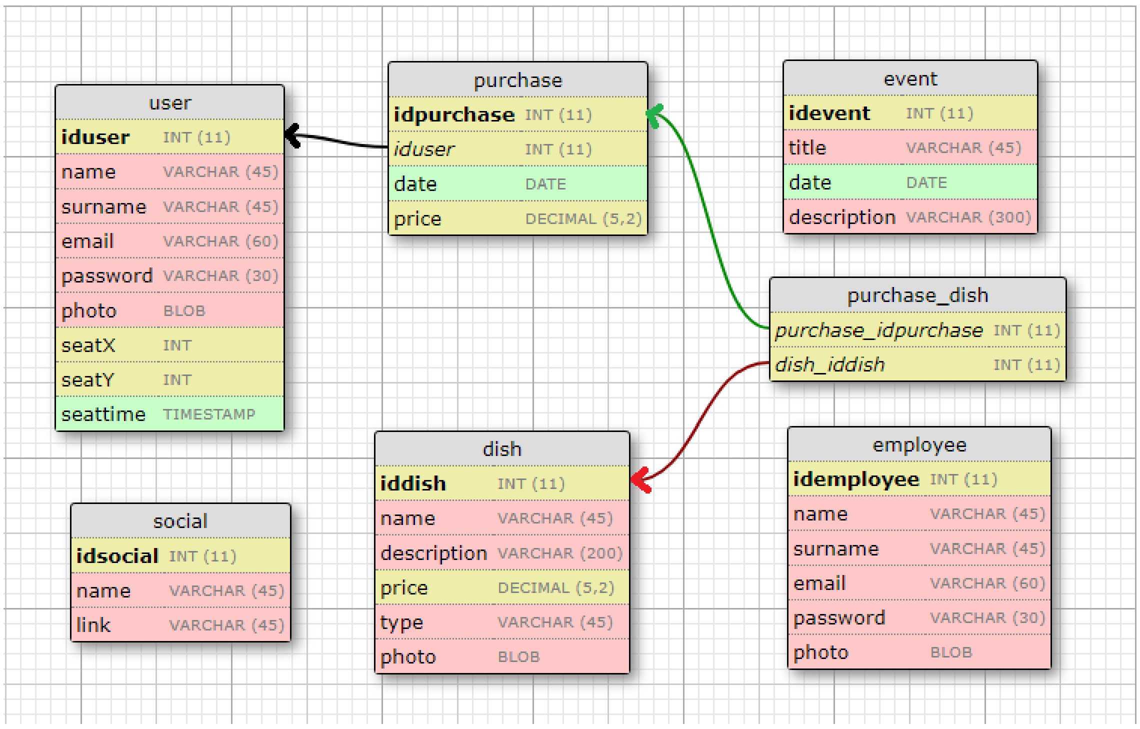
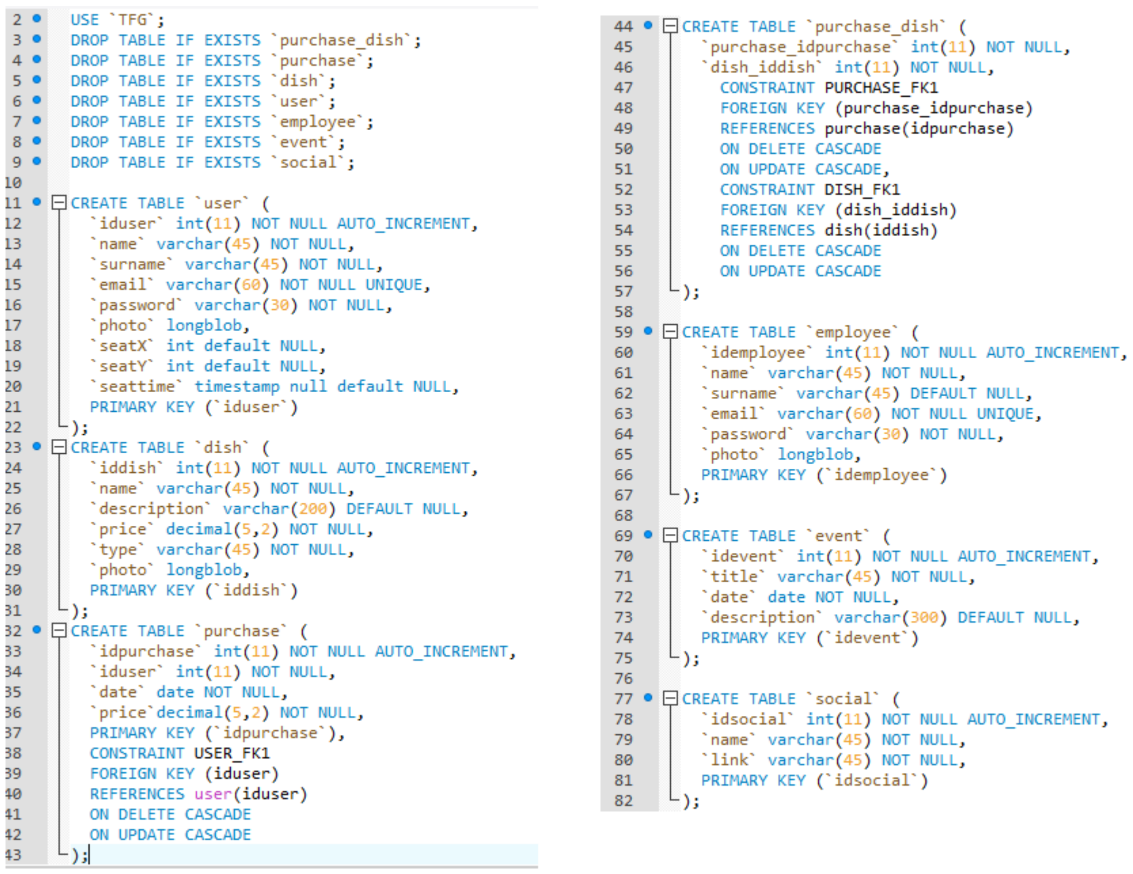


Fig. 4.8. Database diagram

In the MySQL database script, all the tables are created with its corresponding primary key and foreign keys. It has been used on delete/update cascade because if for example a user is deleted, all their purchases will be deleted too; or if a dish is deleted, this dish will be deleted from the purchases because it does not exist anymore.



```sql
USE `TFG`;
DROP TABLE IF EXISTS `purchase_dish`;
DROP TABLE IF EXISTS `purchase`;
DROP TABLE IF EXISTS `dish`;
DROP TABLE IF EXISTS `user`;
DROP TABLE IF EXISTS `employee`;
DROP TABLE IF EXISTS `event`;
DROP TABLE IF EXISTS `social`;

CREATE TABLE `user` (
  `iduser` int(11) NOT NULL AUTO_INCREMENT,
  `name` varchar(45) NOT NULL,
  `surname` varchar(45) NOT NULL,
  `email` varchar(60) NOT NULL UNIQUE,
  `password` varchar(30) NOT NULL,
  `photo` longblob,
  `seatX` int default NULL,
  `seatY` int default NULL,
  `seattime` timestamp null default NULL,
  PRIMARY KEY (`iduser`)
);
CREATE TABLE `dish` (
  `iddish` int(11) NOT NULL AUTO_INCREMENT,
  `name` varchar(45) NOT NULL,
  `description` varchar(200) DEFAULT NULL,
  `price` decimal(5,2) NOT NULL,
  `type` varchar(45) NOT NULL,
  `photo` longblob,
  PRIMARY KEY (`iddish`)
);
CREATE TABLE `purchase` (
  `idpurchase` int(11) NOT NULL AUTO_INCREMENT,
  `iduser` int(11) NOT NULL,
  `date` date NOT NULL,
  `price`decimal(5,2) NOT NULL,
  PRIMARY KEY (`idpurchase`),
  CONSTRAINT USER_FK1
  FOREIGN KEY (iduser)
  REFERENCES user(iduser)
  ON DELETE CASCADE
  ON UPDATE CASCADE
);
CREATE TABLE `purchase_dish` (
  `purchase_idpurchase` int(11) NOT NULL,
  `dish_iddish` int(11) NOT NULL,
    CONSTRAINT PURCHASE_FK1
    FOREIGN KEY (purchase_idpurchase)
    REFERENCES purchase(idpurchase)
    ON DELETE CASCADE
    ON UPDATE CASCADE,
    CONSTRAINT DISH_FK1
    FOREIGN KEY (dish_iddish)
    REFERENCES dish(iddish)
    ON DELETE CASCADE
    ON UPDATE CASCADE
);

CREATE TABLE `employee` (
  `idemployee` int(11) NOT NULL AUTO_INCREMENT,
  `name` varchar(45) NOT NULL,
  `surname` varchar(45) DEFAULT NULL,
  `email` varchar(60) NOT NULL UNIQUE,
  `password` varchar(30) NOT NULL,
  `photo` longblob,
  PRIMARY KEY (`idemployee`)
);

CREATE TABLE `event` (
  `idevent` int(11) NOT NULL AUTO_INCREMENT,
  `title` varchar(45) NOT NULL,
  `date` date NOT NULL,
  `description` varchar(300) DEFAULT NULL,
  PRIMARY KEY (`idevent`)
);

CREATE TABLE `social` (
  `idsocial` int(11) NOT NULL AUTO_INCREMENT,
  `name` varchar(45) NOT NULL,
  `link` varchar(45) NOT NULL,
  PRIMARY KEY (`idsocial`)
);
```

Fig. 4.9. Database script

4.4.2 ORM in Spring

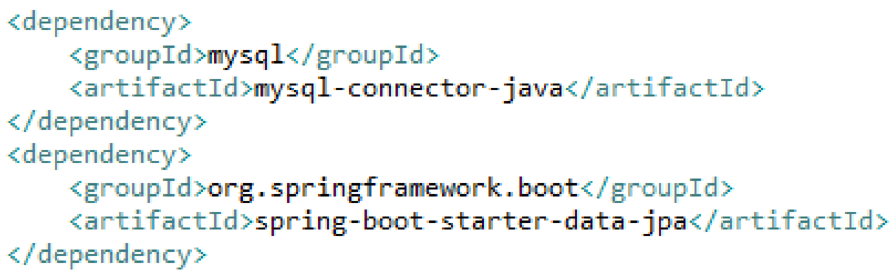
Object-Relational Mapping (ORM) is a technique used in object-oriented programming languages that maps the relational database tables into programming objects called entities. This technique will be used by Spring Data explained below.

4.4.2.1 Spring Data

Spring Data is a Spring module whose objective is to simplify the data persistence against different information repositories. Spring Data can be used with JPA, MongoDB, Redis, Solr, etc. In this project JPA is used. [6]

Java Persistence API (JPA) is a persistence API developed for Java which handles relational data following ORM pattern. Spring Data JPA allows to implement the ORM layer in a simple way and reduce the necessary lines of code implementing automatically the repository interfaces. The interface used in this project is CRUD Repository, which will be explained soon.

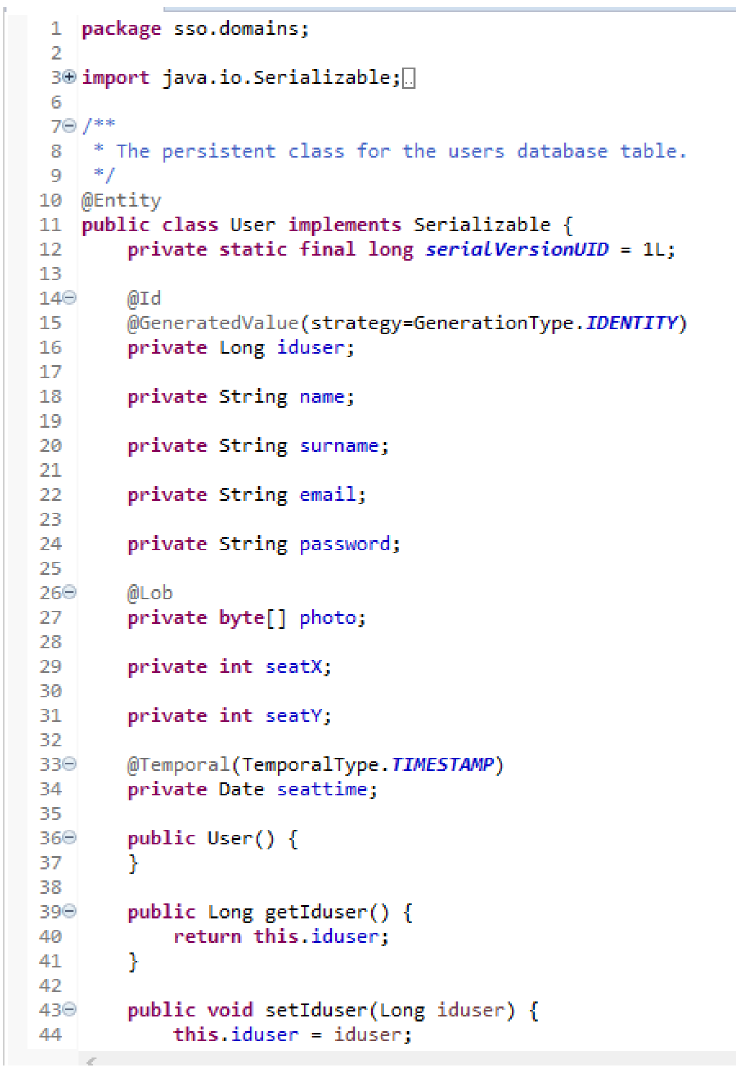
For starting using Spring Data JPA it is necessary to add two new dependencies in pom.xml: Spring Data JPA module and relational database module which is MySQL in this project.



```xml
<dependency>
    <groupId>mysql</groupId>
    <artifactId>mysql-connector-java</artifactId>
</dependency>
<dependency>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-data-jpa</artifactId>
</dependency>
```

Fig. 4.10. Spring Data JPA and MySQL dependencies

After that, the next step is to define the entities. For that, a Java class is created with the structure of that entity. The image below represents the user entity:



```java
package sso.domains;

import java.io.Serializable;

/**
 * The persistent class for the users database table.
 */
@Entity
public class User implements Serializable {
    private static final long serialVersionUID = 1L;

    @Id
    @GeneratedValue(strategy=GenerationType.IDENTITY)
    private Long iduser;

    private String name;

    private String surname;

    private String email;

    private String password;

    @Lob
    private byte[] photo;

    private int seatX;

    private int seatY;

    @Temporal(TemporalType.TIMESTAMP)
    private Date seattime;

    public User() {
    }

    public Long getIduser() {
        return this.iduser;
    }

    public void setIduser(Long iduser) {
        this.iduser = iduser;
```

Fig. 4.11. User Entity

The User entity contains all the fields created in the MySQL database table named user because when running the application, the table user will be mapped with this User entity.

Each field in the entity has their corresponding getters and setters.

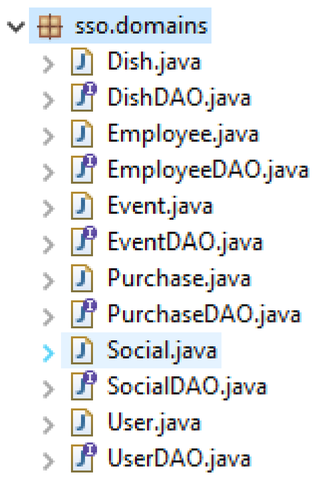
All the entities are defined with the annotation `@Entity` to indicate that they will be used by Spring Data. The name of the table must be the same as the name of the entity for the mapping. If not, the annotation `@Table` must be used to indicate the name of the table to map.

The primary key will be indicated with the annotation `@Id` and `@GeneratedValue` is used to fill these ids automatically by JPA.

There are more annotations that have been used in this project like `@Lob` which indicates that the type is blob, or `@Temporal` to indicate if we are referring to date, time or timestamp. `@Column` used if the name of the table attribute is different from the one written in the entity.

Using all these annotations, the database structure will be defined making sure that there is a correct traceability between data. The ORM layer generated by Spring JPA handles these operations.

Finally, one entity per each table is created, except the `purchase_dish` table which is indicated with the `@ManyToMany` annotation.

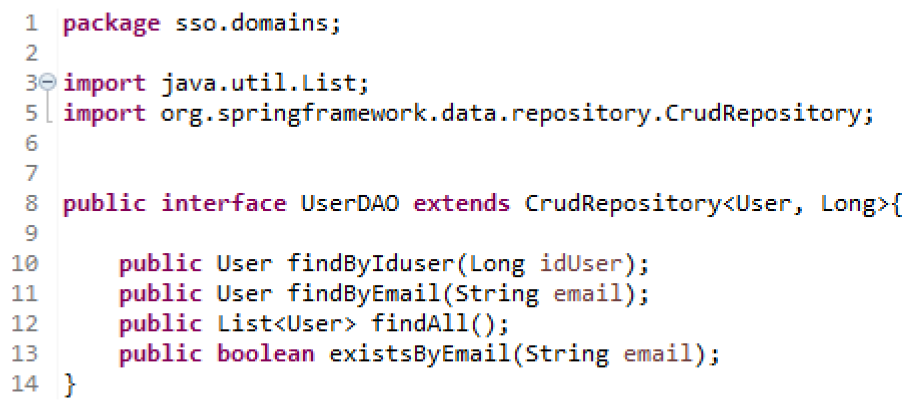


```
sso.domains
  Dish.java
  DishDAO.java
  Employee.java
  EmployeeDAO.java
  Event.java
  EventDAO.java
  Purchase.java
  PurchaseDAO.java
  Social.java
  SocialDAO.java
  User.java
  UserDAO.java
```

Fig. 4.12. All project entities and DAOs

Next step is to create a DAO linked to the entity. The DAO is a specific interface for each entity that extends a repository, which will make very easy to perform the main operations with the database.

The following DAO works with the User Entity:



```java
package sso.domains;

import java.util.List;
import org.springframework.data.repository.CrudRepository;


public interface UserDAO extends CrudRepository<User, Long>{

    public User findByIduser(Long idUser);
    public User findByEmail(String email);
    public List<User> findAll();
    public boolean existsByEmail(String email);
}
```

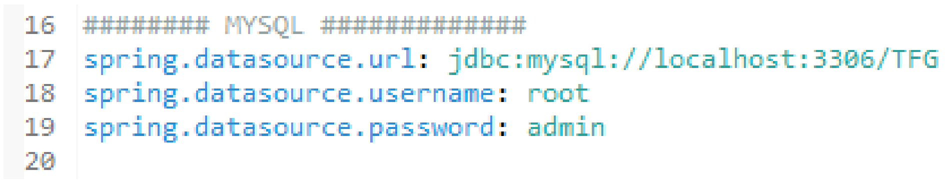
Fig. 4.13. User DAO - CRUD Repository

UserDAO extends the interface CrudRepository which has the CRUD operations (Create, Read, Update, Delete) used to find, save, update and delete records from the User table. The parameters are User and Long, that indicates the type of the entity and the primary Key data type.

To define specific queries, methods are created in the interface, as seen in the image above. Since JPA implements the interface, it is very important to follow a names structure which JPA understands.

At the start of the Spring context, an implementation class will be created with the necessary functionality to cover the base CrudRepository methods, plus the added by us.

Finally, a connection with the database must be configured in the application.yaml file. The location of the database and the credentials of how to access it are set.



```yaml
######## MYSQL #############
spring.datasource.url: jdbc:mysql://localhost:3306/TFG
spring.datasource.username: root
spring.datasource.password: admin

```

Fig. 4.14. Database connection configuration

5 DEVELOPMENT

5.1 Development environment

To develop the application the following work environment has been used:

- Windows 10.
- Java 8.
- Apache Maven 3.6.0.
- Eclipse IDE 2018-12.
- Google Chrome.
- MySQL Workbench 6.3 CE.
- MySQL Community Server 5.5
- Apache Directory Studio.

5.2 Used technologies

In this section, the remaining technologies that have been used to develop some project functionalities are explained.

5.2.1 JQuery

JQuery is an open source JavaScript library that simplifies JavaScript programming. There exists a lot of plugins created by developers that can be found in internet that solve concrete situations like a responsive menu, a carrousel of images, a photo gallery, etc. [19]

For using JQuery is as easy as locate the `.js` file in the `src/main/resources/js` folder and then reference it by means of a `<script>` tag. For example:

```
<script src=”js/jquery-3.2.1.min.js”></script>
```

5.2.2 AJAX

AJAX (Asynchronous JavaScript and XML) is a technique that allows a client and a server to exchange information asynchronously and it is not necessary to refresh the webpage to see what returned the server; unlike when for example the button of a form is pressed and the page is refreshed to show you a message saying that the form was sent.

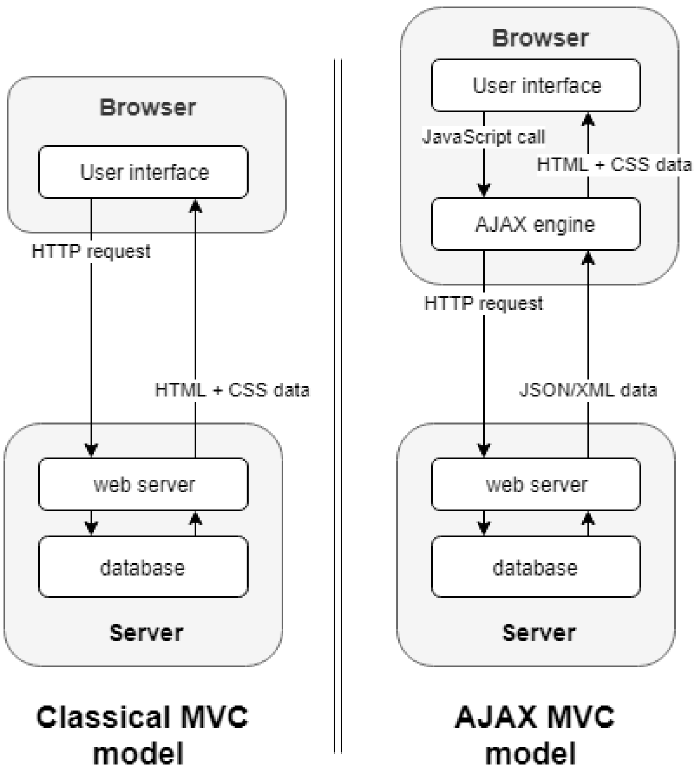


Fig. 5.1. MVC based in AJAX

With the AJAX technique the server will not send a complete view that refresh the web page as the classical model did. It will send back a JSON or XML as response and it will be stored in a JavaScript variable used to modify something in the already existing view.

This technique will be used in the Spring Boot project for the employees' page. When someone buys a dish, it will appear in the employees' page automatically in real time, without refreshing the page by the employees to see if there are new purchases to serve those dishes.

The most common way of making AJAX calls is with JQuery. It provides a method called `$.ajax()`, which will make the AJAX request.

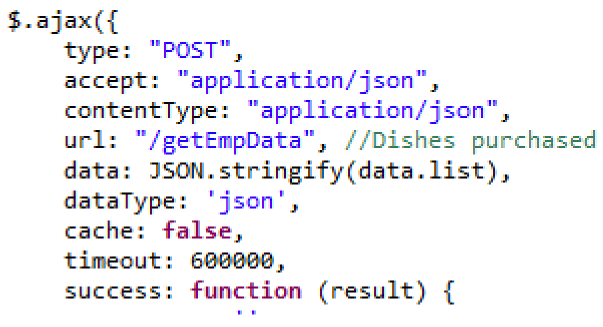

```
$.ajax({
    type: "POST",
    accept: "application/json",
    contentType: "application/json",
    url: "/getEmpData", //Dishes purchased
    data: JSON.stringify(data.list),
    dataType: 'json',
    cache: false,
    timeout: 600000,
    success: function (result) {
```

Fig. 5.2. AJAX request with JQuery

The above code will make a POST request, in which a purchase id (`data.list`) is passed. It will be caught by the controller and will return the dishes information of that purchase as JSON. On success, a function will be called with the result as parameter and it will be used to draw in the employees' screen the information of the dishes.

5.2.3 Pusher

Pusher is a cloud service to manage connections and messages sending through WebSockets (bidirectional communication channel). It encapsulates WebSockets implementation saving time because it is not necessary to make a new infrastructure and it is possible to automatically scale according to the number of connections and the number of sent messages. [20]

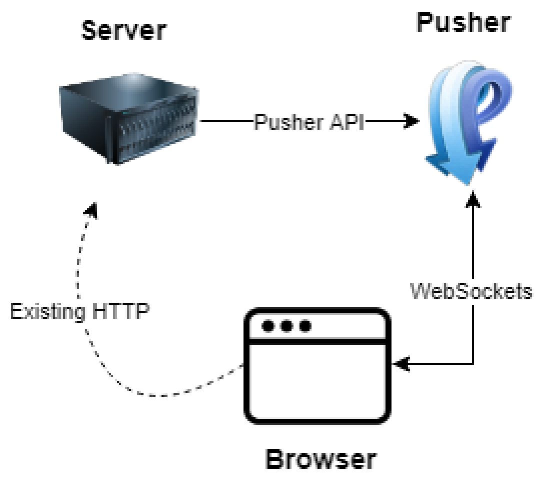
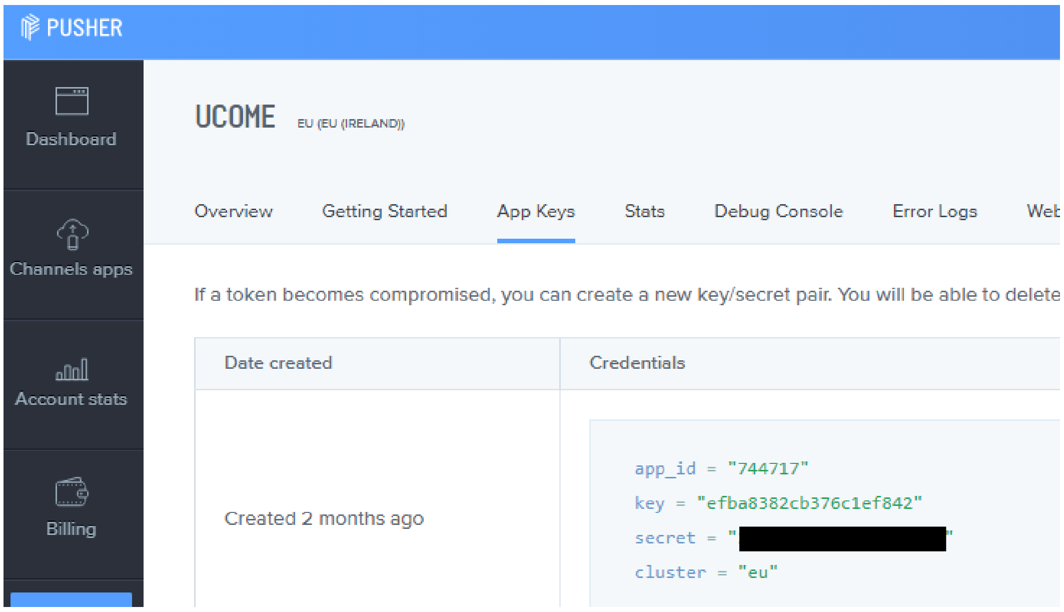


Fig. 5.3. Pusher working diagram

First step to start using Pusher is to create a new account, register an application and some credentials will be obtained to be used in the code to establish a connection.



If a token becomes compromised, you can create a new key/secret pair. You will be able to delete

Date created	Credentials

Created 2 months ago	app_id = "744717"
	key = "efba8382cb376c1ef842"
	secret = "[redacted]"
	cluster = "eu"

Fig. 5.4. Pusher - App registration

Then, the Pusher dependency is added in pom.xml file.

```
<dependency>
    <groupId>com.pusher</groupId>
    <artifactId>pusher-http-java</artifactId>
    <version>1.0.0</version>
</dependency>
```

Fig. 5.5. Pusher dependency

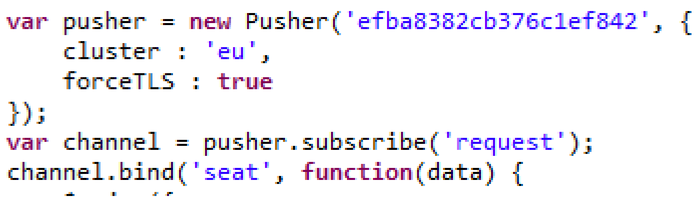
UCOME uses Pusher to send a message to all the clients that have been seated for more than thirty minutes asking them if they continue eating or not. This is a control mechanism that is used because the diners could have forgotten to free their seat after finishing eating. If they confirm that they continue eating, no operation is performed and if they confirm that they finished, their seat is freed. In case the user does not answer, he will have two attempts to answer or his seat will be freed.

From Java, this is the way a message is sent to Pusher through the Pusher API:

```
Pusher pusher = new Pusher(pusher_app_id, pusher_key, pusher_secret);
pusher.setCluster(pusher_cluster);
pusher.setEncrypted(true);
pusher.trigger("request", "seat", Collections.singletonMap("message",
```

Fig. 5.6. Pusher server code

And the JavaScript client will take the message being subscribed to the channel:



```
var pusher = new Pusher('efba8382cb376c1ef842', {
    cluster : 'eu',
    forceTLS : true
});
var channel = pusher.subscribe('request');
channel.bind('seat', function(data) {
```

Fig. 5.7. Pusher client code

5.2.4 Swing

Java Swing is a GUI (Graphical User Interface) tool that allows creating widgets for Java applications. There are two main basic elements for the creation of graphical interfaces [21]:

- Containers: elements that contains other components like buttons and text fields.
- Components: elements that are added into containers.

Swing is used to create the scenario where the administrator sets the square meters that the local has and a matrix with those dimensions will appear. He will be able to click on the squares which mean that there is a seat there. Therefore, the restaurant/cafeteria tables will be represented. After that, the user could choose any free seat and it will become occupied.

The containers used are:

- JFrame: it is the main container and the most used one. It represents the main window.
- JPanel: it allows the creation of independent panels where other components are added.

They are composed by the following components:

- JButton: implementation of a push button that performs an action when pressed.
- JLabel: text or image displayed in the frame.
- JTextField: text field for obtaining data.
- JSeparator: horizontal or vertical dividing line.

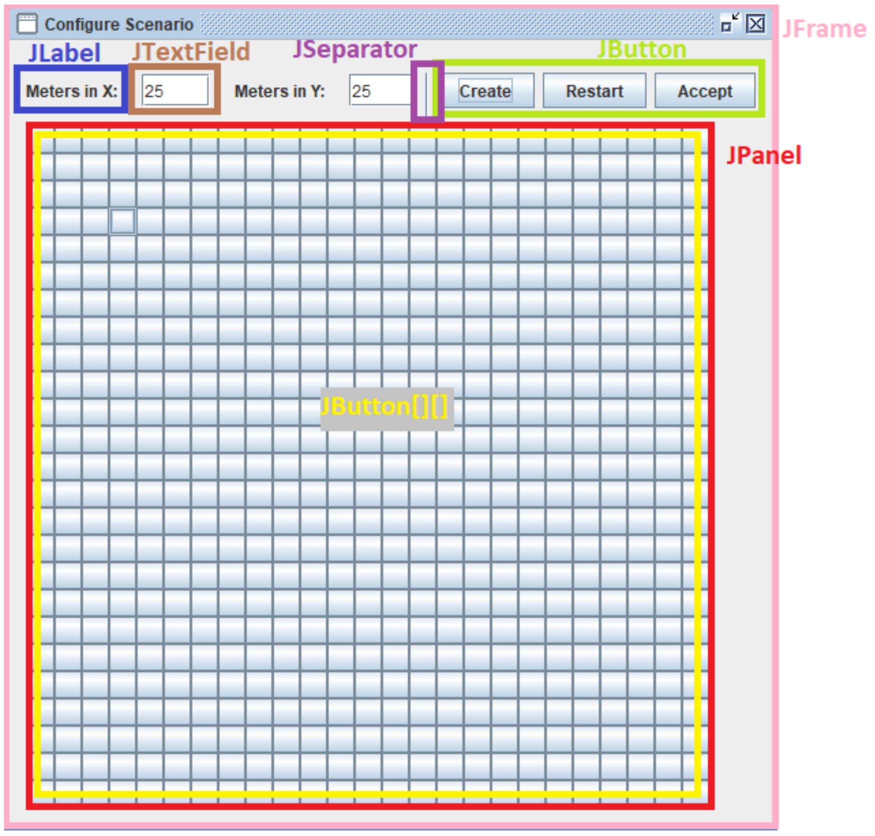


Fig. 5.8. Java Swing - Configure scenario

These frames created with Swing are only accessible in the server. The client side won't be able to see them. This is a disadvantage because it is needed to be in the server computer.

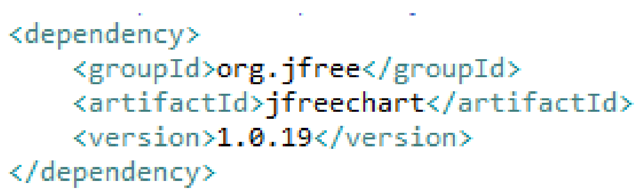
5.2.5 JFreeChart

Swing has no graphing and charting packages. Then, the JFreeChart library is used for making a bar chart in which the statistics of the dishes are shown. That is, how many times a dish was purchased giving to the administrator a global vision by providing him information about the most purchased dishes and the less purchased ones. The charts are also classified by colours indicating to which type the dish belongs to (seafood, pasta, meats, etc.). The final purpose of this is to suggest improvements and to think why some dishes are not being sold.

The total revenues are displayed too. It defines how much money the cafeteria/restaurant earned by selling the dishes.

The administrator can filter the data by dates. Therefore, he will be able to see what dishes sold and how much earned in a certain interval of time, for example, per month.

For using JFreeChart the following dependency must be added:



```xml
<dependency>
    <groupId>org.jfree</groupId>
    <artifactId>jfreechart</artifactId>
    <version>1.0.19</version>
</dependency>
```

Fig. 5.9. JFreeChart dependency

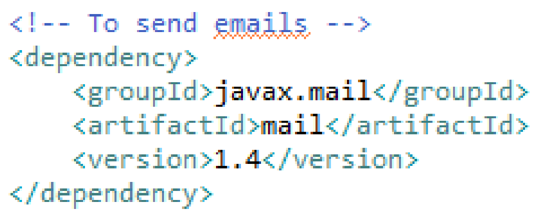
Then, a JFreeChart is created with the dataset to be displayed. It is inserted into a `ChartPanel` of the JFreeChart library and this panel will be added into a `JPanel`, which, in turn, is inside a `JFrame`. Both from Java Swing.

5.2.6 JavaMail

UCOME sends emails with the new password to the employees registered by the administrator and to the users that were synchronized from LDAP by the administrator too.

For that, JavaMail is used. It is an API that facilitates the sending and reception of emails through the SMTP, POP3 and IMAP protocols. SMTP (Simple Mail Transfer Protocol) is used for the outgoing mail management, while POP3 (Post Office Protocol) and IMAP (Internet Message Access Protocol) are used for receiving the incoming mail.

For using JavaMail in the Spring Boot project, the following dependency must be added:



```xml
<!-- To send emails -->
<dependency>
    <groupId>javax.mail</groupId>
    <artifactId>mail</artifactId>
    <version>1.4</version>
</dependency>
```

Fig. 5.10. JavaMail dependency

UCOME will communicate with the SMTP server of our service provider (ISP). This SMTP server will leave the message in the SMTP server of the recipient to be taken through POP or IMAP by the receiver.

This process is carried out in Java by getting an instance of the `Session` class of JavaMail. For that, the method `Session.getDefaultInstance()` is called passing the connection properties as parameter that are explained in the following table:

TABLE 5.1. JAVAMAIL CONNECTION PROPERTIES

Property	Value	Description
mail.smtp.host	smtp.gmail.com	Google SMTP server.
mail.smtp.user	sender	Username of the sender.
mail.smtp.auth	true	Use authentication through user and password.
mail.smtp.starttls.enable	true	Secure connection to the SMTP server through TLS.
mail.smtp.port	587	Secure SMTP port of Google.

After that, the message is created by instantiating the `MimeMessage` class, which will receive the session generated before and the fields from, to, subject and text are fulfilled.

Finally, the message is sent with an instance of the `Transport` class that models a message transport. [22]

5.3 Authentication

UCOME is thought and designed to facilitate users to log in without the need to use or remember the access username and password. There are four ways for users to log in: through LDAP, Office365, Google and Aula Global.

5.3.1 LDAP

LDAP (Lightweight Directory Access Protocol) is an application-level protocol which allows to perform queries on a directory service to search information in a network. A directory service is a database in which information is stored and organized. This hierarchically organized structure of the objects in the directory is achieved with the implementation of LDAP that defines how to access to the directory. LDAP directories are not relational databases. They are optimized for a good performance on reading.

A directory service executes the client-server model, that is, if a client wants to access to some information, he will not access directly to the database, but he will contact a process in the server side. This process makes the query and the information is returned to the client.

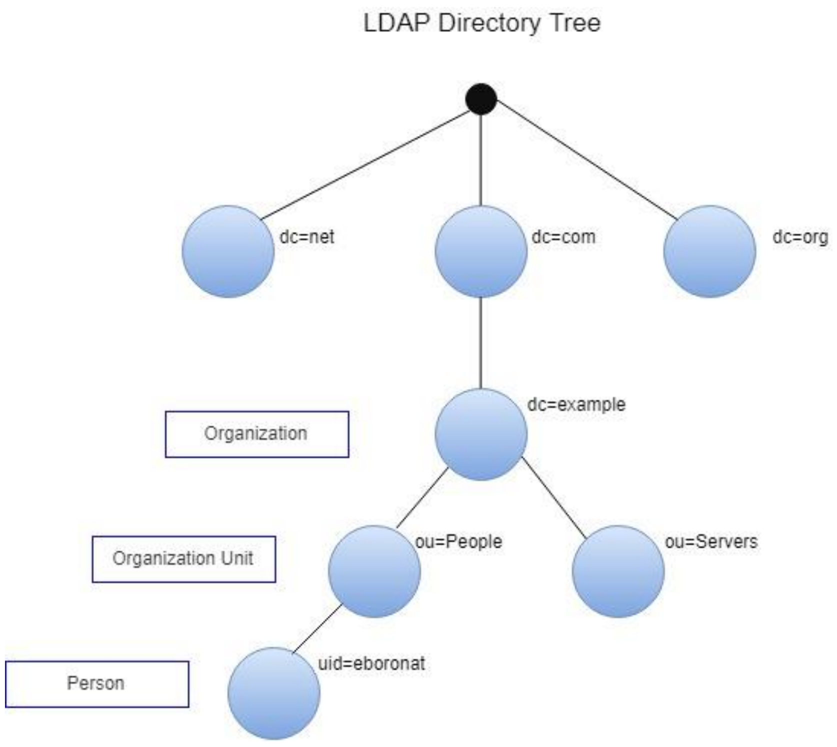


Fig. 5.11. LDAP Directory structure

Each blue circle in the image above represents an entry and, in this way, an organized and distributed tree is created to be able to perform queries. The structure of a LDAP directory is the following:

- Entries: collection of attributes with a Distinguished Name (DN) used as unique identifier of an entry. For example, in the image above, the DN of `eboronat` would be: `dn: uid=eboronat,ou=People,dc=example,dc=com.`
- Attributes: properties of the entries. Some attributes are name, surname, mail, photo, etc. Some attributes of an entry are mandatory and other optional.

For the import and export of data independently of the LDAP server that is being used, the LDAP Data Interchange Format (LDIF) format is used. It represents the entries in ASCII text. [23]

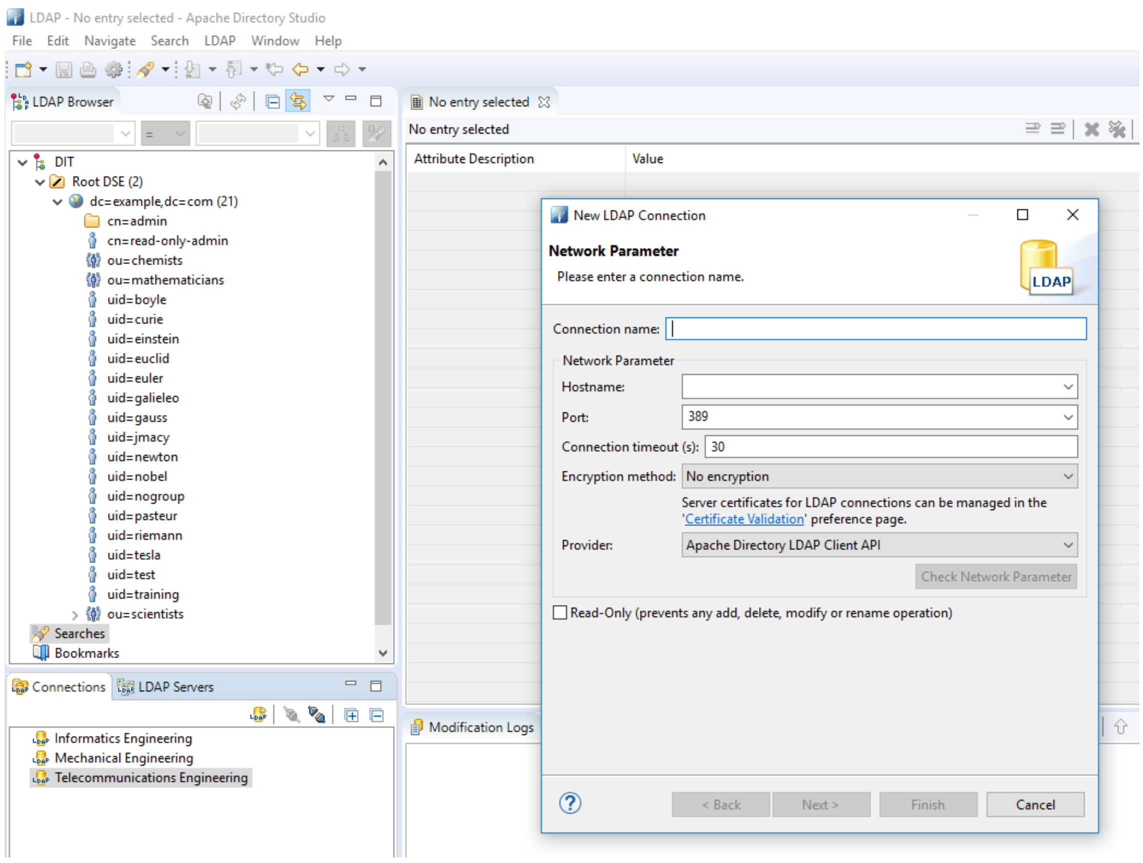
Most of the big enterprises use these directory services for multiple purposes: organizing employees' data, accessing to some services with the same password, access management to a corporative intranet, CMS (Content Management System) authentication through LDAP, etc. The biggest advantage of LDAP is that companies can access the LDAP directory from almost any computing platform. [24]

As this project focusses mainly on companies' cafeterias and restaurants, it was decided to implement authentication to the application through LDAP. Therefore, all the companies that have directories in which employees are organized, will be able to access to UCOME without the need of remembering the application password. They will log in with his LDAP username and password.

5.3.1.1 Apache Directory Studio

Apache Directory Studio is a software to display and read an LDAP Directory Tree, but it also can be used to create and to launch a new LDAP server (ApacheDS), in which a new directory tree is created and it is allowed creating, modifying or deleting entries. [25]

This software is then used to create a new LDAP server for creating our own directory and two online LDAP servers found in the internet are connected. The purpose is to test the application against these three LDAP servers.



DIT
Root DSE (2)
dc=example,dc=com (21)
cn=admin
cn=read-only-admin
ou=chemists
ou=mathematicians
uid=boyle
uid=curie
uid=einstein
uid=euclid
uid=euler
uid=galieleo
uid=gauss
uid=jmacy
uid=newton
uid=nobel
uid=nogroup
uid=pasteur
uid=riemann
uid=tesla
uid=test
uid=training
ou=scientists
Searches
Bookmarks

Informatics Engineering
Mechanical Engineering
Telecommunications Engineering

Please enter a connection name.

Connection name:
Network Parameter
Hostname:
Port: 389
Connection timeout (s): 30
Encryption method: No encryption
Server certificates for LDAP connections can be managed in the 'Certificate Validation' preference page.
Provider: Apache Directory LDAP Client API
Read-Only (prevents any add, delete, modify or rename operation)

Fig. 5.12. Apache Directory Studio

In the image above, it is shown the main features used in Apache Directory Studio. On the bottom left of the image there are two tabs: Connections and LDAP Servers. In the second one, the ApacheDS server is created in localhost on the port 10389. After that, a connection of this new server is opened, and some entries are added with my classmates' information. This connection was called Informatics Engineering. The other two connections are opened from two online LDAP servers for testing purposes and to be sure that the authentication works with different LDAP Directories structures.

Coming back to the image, above these two tabs, the Directory Tree is displayed, and it is possible to read, edit and delete entries (except the online ones [26] [27] that are only-read directories). These users are the ones who will be able to log in into UCOME.

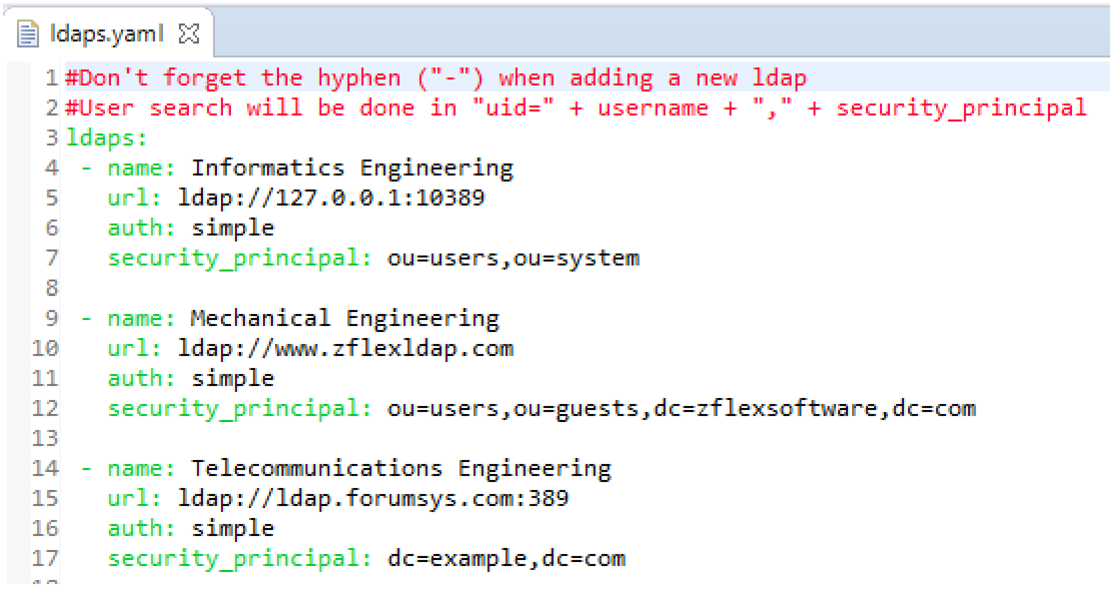
For adding a new connection, the window of the middle of the image will appear. Here the network parameters must be set, as well as the credentials of an authorized user to see the tree in case it is not open for everybody. For the two online servers the indicated parameters were written [26][27] and in the case of the own server the hostname is localhost; and the port is 10389. Authentication parameters are also needed, so the admin and his password are written.

5.3.1.2 Login

In the web application there will be a drop-down button to select the name of the LDAP (established by the administrator in `ldaps.yaml`) in which the user can login through. The number of options in the drop-down will be automatically increased when the admin adds a new LDAP in the file `ldaps.yaml`, located in the folder `/WEB_INF/classes` of the war file delivered to the client (UCOME application buyers).

To add a new LDAP the admin must add the LDAP fields:

- Name: name that will appear in the drop-down. Example: LDAP1.
- Url: LDAP URL.
- Auth: authentication type.
- Security_principal: where the LDAP users' entries are located.



```yaml
#Don't forget the hyphen ("-") when adding a new ldap
#User search will be done in "uid=" + username + "," + security_principal
ldaps:
  - name: Informatics Engineering
    url: ldap://127.0.0.1:10389
    auth: simple
    security_principal: ou=users,ou=system

  - name: Mechanical Engineering
    url: ldap://www.zflexldap.com
    auth: simple
    security_principal: ou=users,ou=guests,dc=zflexsoftware,dc=com

  - name: Telecommunications Engineering
    url: ldap://ldap.forumsys.com:389
    auth: simple
    security_principal: dc=example,dc=com
```

Fig. 5.13. ldaps.yaml configuration file

The end-users will simply select his LDAP from the drop-down button, he will sign in using his LDAP username and password; and he will be directly redirected to his UCOME account.

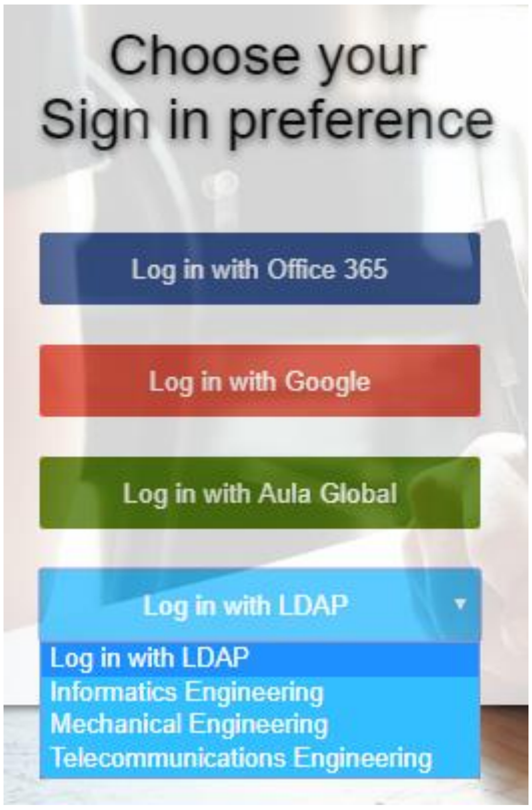


Fig. 5.14. LDAP login drop-down button

5.3.1.3 JNDI

JNDI (Java Naming and Directory Interface) consists of a Java API interface and a Service Provider Interface (SPI), being able to connect with a big variety of directories services. SPI allows you to interact with almost every type of naming or directory service including LDAP.

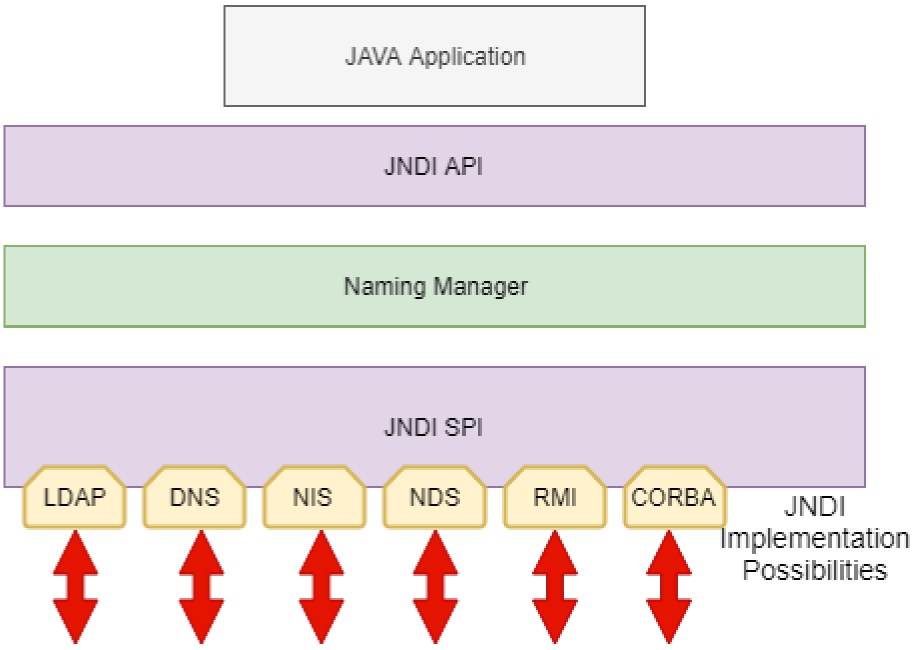
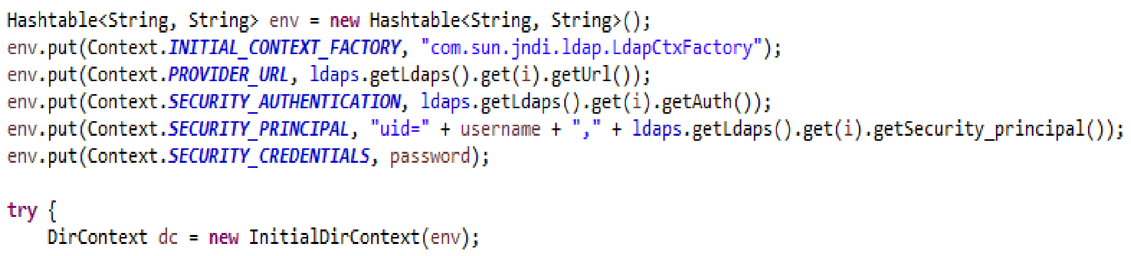


Fig. 5.15. JNDI Architecture

Once configured the `ldaps.yaml` file, the application can be launched. All the LDAPs configuration will be stored as objects and when the end-users try lo login, the authentication will be done with the selected LDAP from the UCOME index page.

In the code below, it is shown how the authentication is done. A `hashtable` is created with the connection parameters and an `InitialDirContext` instance is created. If it is successfully created means that the user entered correctly his credentials.



```
Hashtable<String, String> env = new Hashtable<String, String>();
env.put(Context.INITIAL_CONTEXT_FACTORY, "com.sun.jndi.ldap.LdapCtxFactory");
env.put(Context.PROVIDER_URL, ldaps.getLdaps().get(i).getUrl());
env.put(Context.SECURITY_AUTHENTICATION, ldaps.getLdaps().get(i).getAuth());
env.put(Context.SECURITY_PRINCIPAL, "uid=" + username + "," + ldaps.getLdaps().get(i).getSecurity_principal());
env.put(Context.SECURITY_CREDENTIALS, password);

try {
    DirContext dc = new InitialDirContext(env);
```

Fig. 5.16. JNDI - LDAP Connection

A description of the parameters used to connect to the LDAP are explained in the following table:

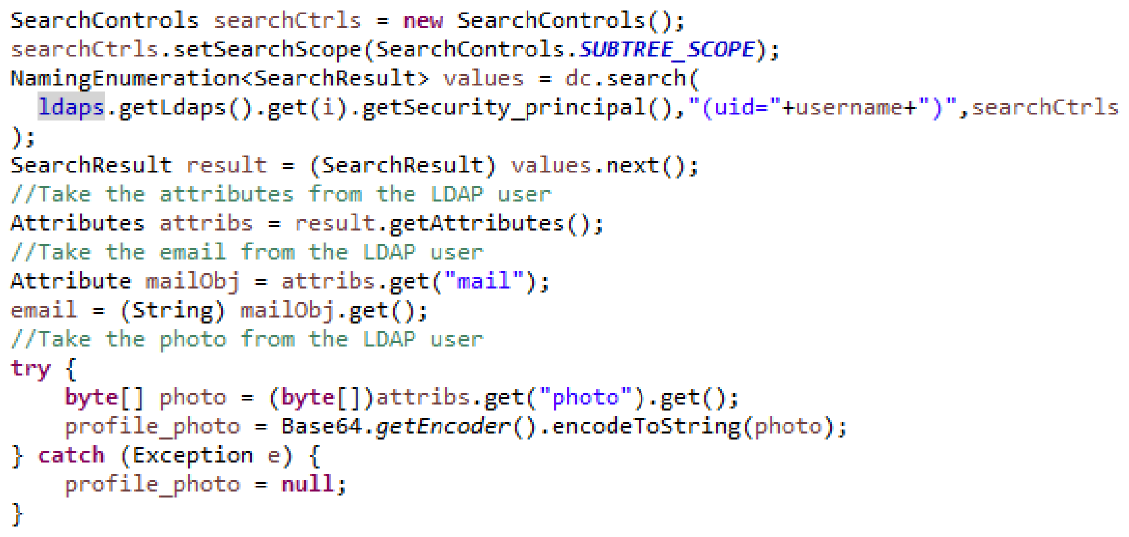
TABLE 5.2. JNDI - CONNECTION PARAMETERS DESCRIPTION

JNDI Constant	**Description**
INITIAL_CONTEXT_FACTORY	Factory class for LDAP contexts
PROVIDER_URL	LDAP service URL
SECURITY_AUTHENTICATION	Authentication type. It can be none, simple or strong
SECURITY_PRINCIPAL	Distinguished Name (DN) of the user
SECURITY_CREDENTIALS	User password

It must be mentioned that companies could use very different Directory Tree structures. For example, Active Directory (AD), which is the Windows directory service implementation, use the type attribute `userPrincipalName` (UPN), which users log in with [28]. This attribute is composed by the username + @domain. In those cases, small changes in the application must be done. JNDI `SECURITY_PRINCIPAL` constant would be in this case the UPN and in the `ldaps.yaml` file the attribute `domain` would be defined instead of `security_principal`. It is good because any user in the tree could authenticate without depending on the user tree location as before.

In conclusion, a previous contact with the client company should be established in order to know which LDAP directory tree structure they have and do some small changes in case they need them.

JNDI is also used to take the photo and the email of the user stored in the directory. It is mandatory to have the attribute "mail" because a linkage between the LDAP user and the UCOME account must be established.



```
SearchControls searchCtrls = new SearchControls();
searchCtrls.setSearchScope(SearchControls.SUBTREE_SCOPE);
NamingEnumeration<SearchResult> values = dc.search(
  ldaps.getLdaps().get(i).getSecurity_principal(),"(uid="+username+")",searchCtrls
);
SearchResult result = (SearchResult) values.next();
//Take the attributes from the LDAP user
Attributes attribs = result.getAttributes();
//Take the email from the LDAP user
Attribute mailObj = attribs.get("mail");
email = (String) mailObj.get();
//Take the photo from the LDAP user
try {
    byte[] photo = (byte[])attribs.get("photo").get();
    profile_photo = Base64.getEncoder().encodeToString(photo);
} catch (Exception e) {
    profile_photo = null;
}
```

Fig. 5.17. JNDI - search for attributes

The image above shows how to access to the attributes. A search of the user is performed indicating the username as filter and where is in the tree located (baseDN). Then, all the attributes are obtained, and the mail and photo are taken.

5.3.1.4 LDAP Synchronization

The administrator of the application will have the functionality to create all the contacts from a LDAP Directory into UCOME. If the contact already exists and any field was modified, it will be updated into UCOME. If all the fields are the same, no operation will be performed.

The following LDAP attributes will be taken to create the users in UCOME with them: `mail` (will be the email in UCOME), `displayName` (will be the name in UCOME) and `sn` (will be the surname in UCOME). If `displayName` is empty in the LDAP, it will be the same as `mail`.

The administrator in charge of the synchronization must edit the `application.yaml` file where the following properties must be completed:

- originLdap: origin the users will be taken to be synchronized from.
- usersToSynchronize: email of the users that the administrator wants to synchronize. If "all", all the users from the origin will be synchronized.

An LDAP user with read permissions is needed to access the directory. These properties are related to him:

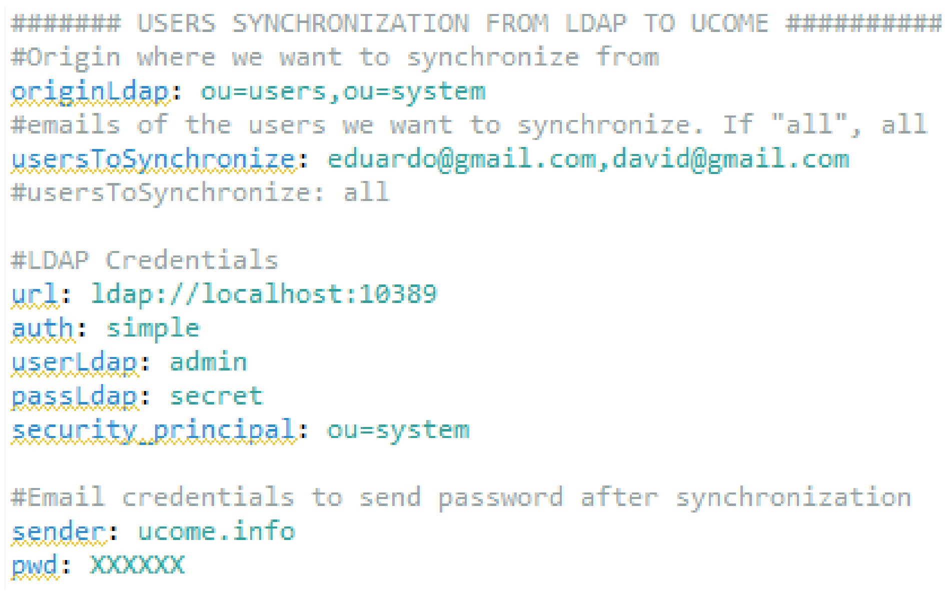
- url: LDAP URL to be accessed.
- auth: authentication type.
- userLdap: user to access LDAP with read permissions.
- passLdap: LDAP password of the previous user.
- security_principal: baseDN (base Distinguished Name) of the previous user.

Finally, as the administrator is who registers the users through this synchronization, a random password is generated for each of the new users. This password will be sent to their email from a Gmail account. For that, the following properties must be written:

- sender: username of the Gmail account (usually administrator username).
- pwd: password of the Gmail account.

For testing, a new Gmail account was created simulating the administrator email: ucome.info@gmail.com

However, this password that is sent is not needed for these users because they will be able to log in through LDAP now. It is the main purpose of this synchronization.



```
####### USERS SYNCHRONIZATION FROM LDAP TO UCOME ##########
#Origin where we want to synchronize from
originLdap: ou=users,ou=system
#emails of the users we want to synchronize. If "all", all
usersToSynchronize: eduardo@gmail.com,david@gmail.com
#usersToSynchronize: all

#LDAP Credentials
url: ldap://localhost:10389
auth: simple
userLdap: admin
passLdap: secret
security_principal: ou=system

#Email credentials to send password after synchronization
sender: ucome.info
pwd: XXXXXX
```

Fig. 5.18. LDAP Synchronization - application.yaml

5.3.2 Azure Active Directory

As with LDAP servers, the application is also integrated with Azure Active Directory (Azure AD or AAD) because a lot of companies have acquired it as the identity platform for managing users and providing secure access to their applications.

AAD allows to have a unique identity through a set of applications registered inside Azure. Therefore, a Single Sign-On (SSO) has been implemented.

5.3.2.1 Office 365

Office 365 can be free installed being a student of the university; and Office 365 subscribers have automatically an Azure AD tenant. That is why I could install it and test the implementation.

5.3.2.2 Spring Security OAuth2

Azure AD is responsible for verifying the identity of users. For that, the OAuth 2.0 protocol is used. It is an authorization protocol that allows users to give access to their information to third-parties, who will not know user credentials.

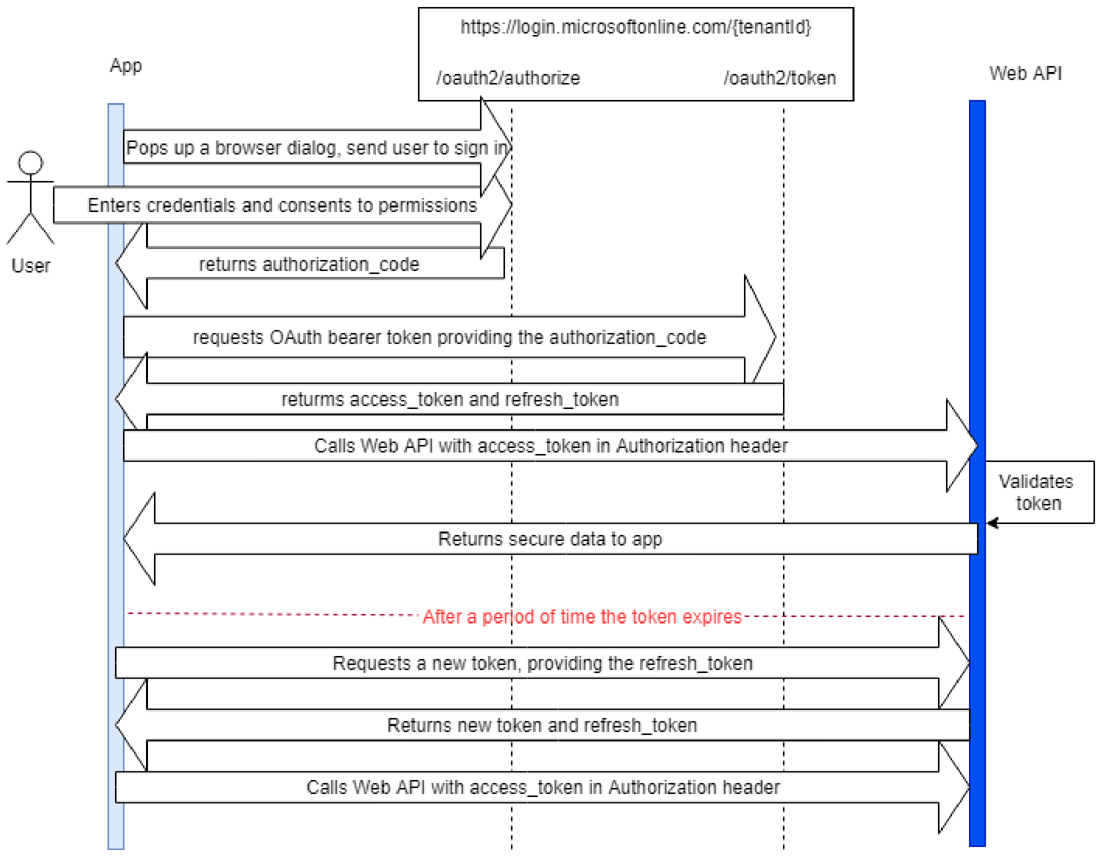
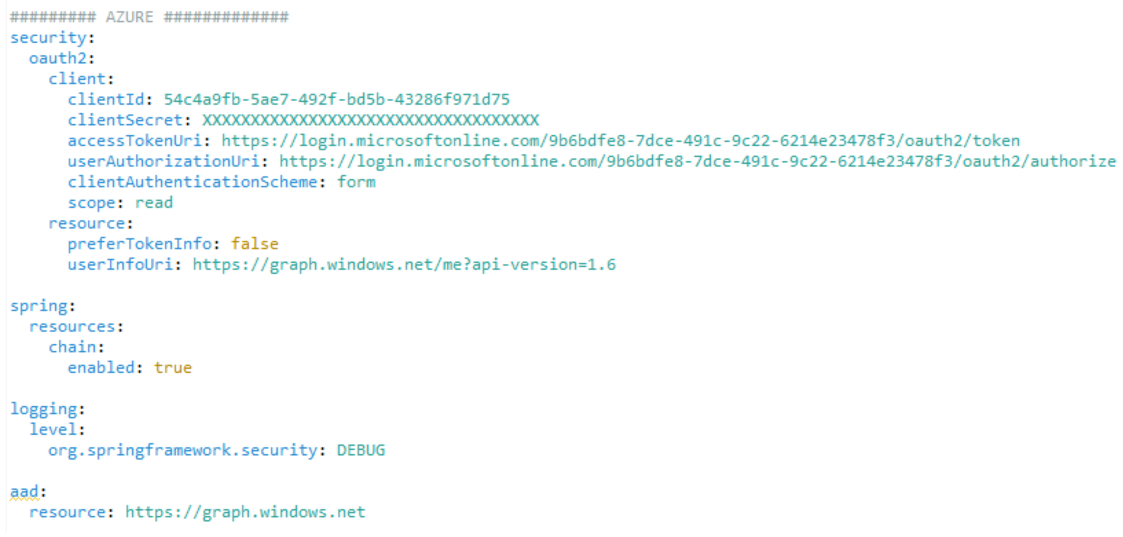

https://login.microsoftonline.com/{tenantId}
/oauth2/authorize
/oauth2/token
App
Web API
User
Pops up a browser dialog, send user to sign in
Enters credentials and consents to permissions
returns authorization_code
requests OAuth bearer token providing the authorization_code
returns access_token and refresh_token
Calls Web API with access_token in Authorization header
Validates token
Returns secure data to app
After a period of time the token expires
Requests a new token, providing the refresh_token
Returns new token and refresh_token
Calls Web API with access_token in Authorization header

Fig. 5.19. OAuth 2.0 authorization flow

This process using the OAuth 2.0 protocol is implemented by the Spring Security module in a simple way. For that, in the main Spring Boot class the `@EnableOAuth2Sso` annotation is needed and it is set which URL is the login page.

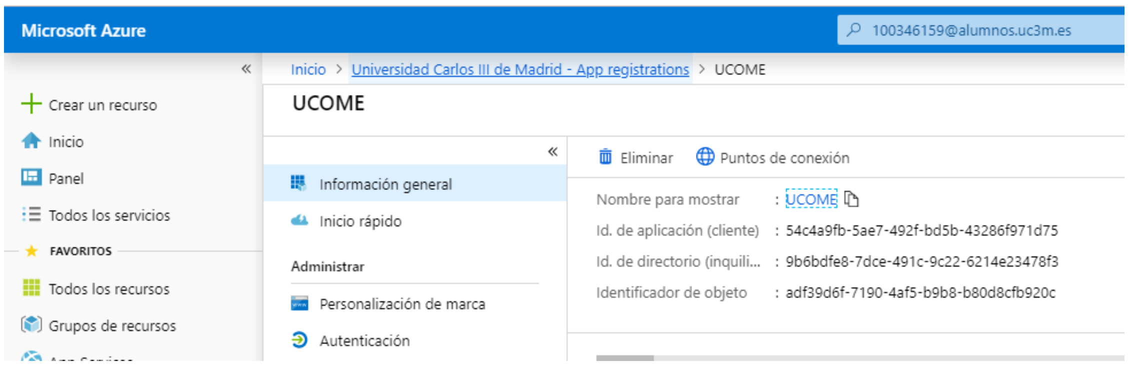
Then, in the configuration file (`application.yaml`) the following properties are defined inside specific tags like `clientId`, `clientSecret`, etc. that Spring Security understands:



```yaml
######### AZURE #############
security:
  oauth2:
    client:
      clientId: 54c4a9fb-5ae7-492f-bd5b-43286f971d75
      clientSecret: XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
      accessTokenUri: https://login.microsoftonline.com/9b6bdfe8-7dce-491c-9c22-6214e23478f3/oauth2/token
      userAuthorizationUri: https://login.microsoftonline.com/9b6bdfe8-7dce-491c-9c22-6214e23478f3/oauth2/authorize
      clientAuthenticationScheme: form
      scope: read
    resource:
      preferTokenInfo: false
      userInfoUri: https://graph.windows.net/me?api-version=1.6

spring:
  resources:
    chain:
      enabled: true

logging:
  level:
    org.springframework.security: DEBUG

aad:
  resource: https://graph.windows.net
```

Fig. 5.20. Azure Active Directory Credentials - application.yaml

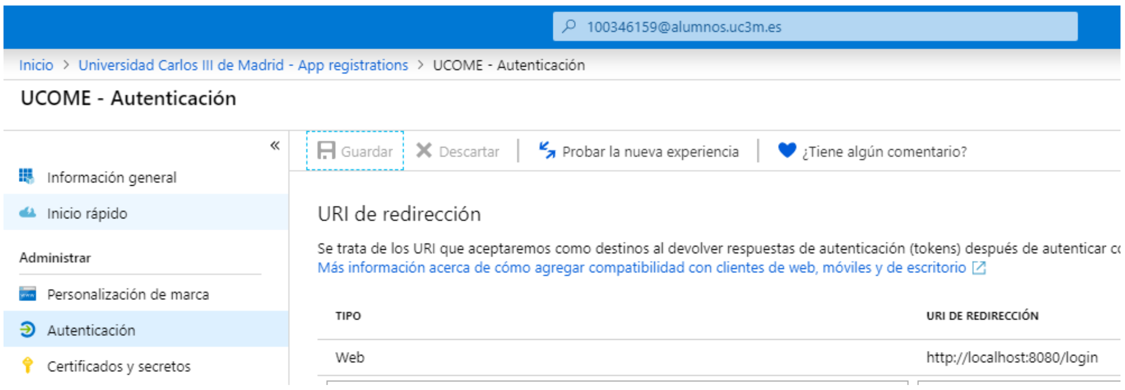
These credentials are obtained from the Azure Active Directory when registering an application:



Nombre para mostrar	: UCOME
Id. de aplicación (cliente)	: 54c4a9fb-5ae7-492f-bd5b-43286f971d75
Id. de directorio (inquili...	: 9b6bdfe8-7dce-491c-9c22-6214e23478f3
Identificador de objeto	: adf39d6f-7190-4af5-b9b8-b80d8cfb920c

Fig. 5.21. Application registration - AAD

A redirection URI must also be provided because when UCOME redirects to Microsoft, it must know where to send the authorization code after logging:



URI de redirección

Se trata de los URI que aceptaremos como destinos al devolver respuestas de autenticación (tokens) después de autenticar c

Más información acerca de cómo agregar compatibilidad con clientes de web, móviles y de escritorio

TIPO	URI DE REDIRECCIÓN
Web	http://localhost:8080/login

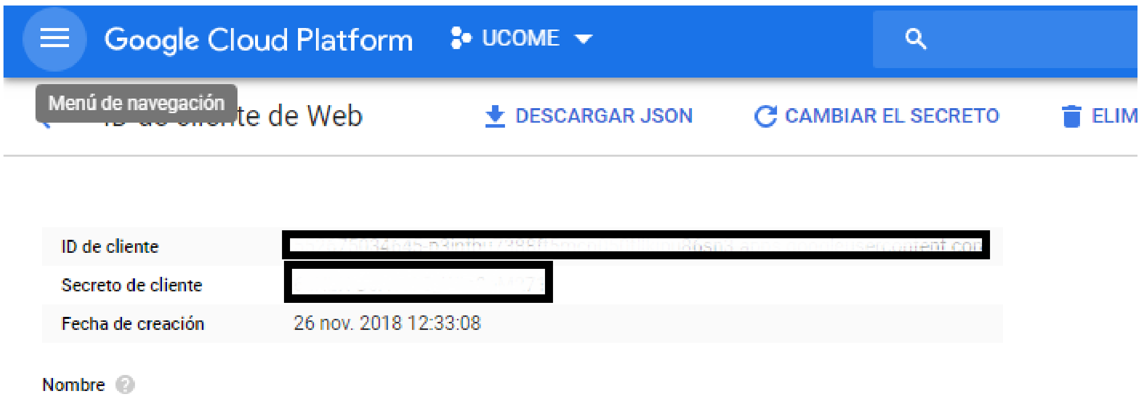
Fig. 5.22. Redirection URI - AAD

Spring Security oversees all the process described in Fig. 5.19 and it will do an automatic redirect to Microsoft login page when clicking on the Office 365 login button. After a successful login, an authorization code will be returned to our application that Spring Security will use to obtain a token that is used to call the AD Graph API where the email of the logged user is get from; and then, it is now possible to do the redirection to the user's UCOME account.

5.3.3 Google

Everybody who has a Google account will be also able to authenticate in UCOME through it with the single sign-on mechanism. The OAuth 2.0 protocol is used for authentication and authorization [29], as was done in Office 365 in the previous section. Therefore, the OAuth2 authorization flow is the same as the one described in the image Fig. 5.19.

First, the OAuth 2.0 credentials must be obtained from the Google API Console for developers after creating a new project.

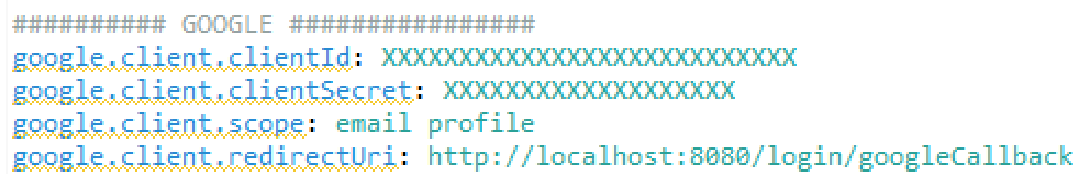


ID de cliente	[redacted]
Secreto de cliente	[redacted]
Fecha de creación	26 nov. 2018 12:33:08

Nombre

Fig. 5.23. Google API Console – OAuth 2.0 Credentials

These generated credentials must be written in the `application.yaml` file for the application to use them.



```
########## GOOGLE ################
google.client.clientId: XXXXXXXXXXXXXXXXXXXXXXXXXXX
google.client.clientSecret: XXXXXXXXXXXXXXXXXXX
google.client.scope: email profile
google.client.redirectUri: http://localhost:8080/login/googleCallback
```

Fig. 5.24. OAuth 2.0 Credentials - application.yaml

The `scope` property controls the set of resources that the access token will permit. The email and profile resources will be requested for getting the user email for redirecting to his UCOME account and the photo of the user. The `redirectUri` property is the URI that Google Authorization Server will return after logging and granting permissions by the user.

After that, the logic of the OAuth 2.0 protocol is developed in the Spring Boot application. When the user clicks on the option to log in through Google, a redirection to Google will be done where the user logs in with his Google account. After logging in, the user will be asked to grant the permissions that the application is requesting (`scope` property). If the user grants the permissions, an authorization code will be obtained from the Google Authorization Server and the application will use it to obtain the access token.

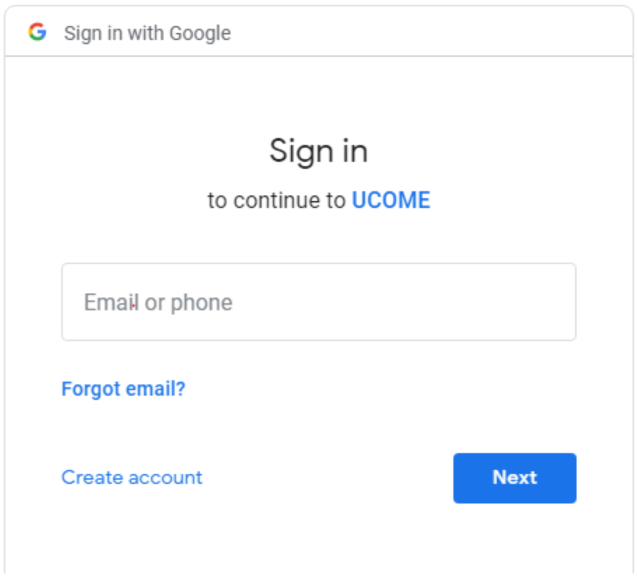


Fig. 5.25. OAuth 2.0 - Google redirection

Finally, the token is sent in the authorization header of a HTTP request for requesting the email and the user's photo to the Google+ API.

5.3.4 Aula Global

Aula Global is a Moodle application integrated by UC3M for the management of teaching resources for students and teachers. Moodle is an LMS (Learning Management System), which is used to manage all the communications between professors and students, to distribute the subjects' contents or to make assessments to students. Therefore, all the people from the university is constantly accessing to Aula Global and it was thought as a good idea to integrate UCOME with Aula Global, in case this system developed in this project is acquired by the university cafeteria. In addition, very small changes could be introduced to adapt UCOME to any other university or institution that uses Moodle.

In the application, the user should enter his username and password of Aula Global and a call to the script located in `/login/token.php` is requested [30]. If authentication is successful, then a token is received, which will be used to make another request to obtain the user's photo. Therefore, the requests will look like:

```
https://www.aulaglobal.uc3m.es/login/token.php?username=USE
RNAME&password=PASSWORD&service=ag_mobile

https://aulaglobal.uc3m.es/webservice/rest/server.php?wstok
en=TOKEN&wsfunction=core_webservice_get_site_info
```

After authenticating, the user will be redirected to his UCOME account.

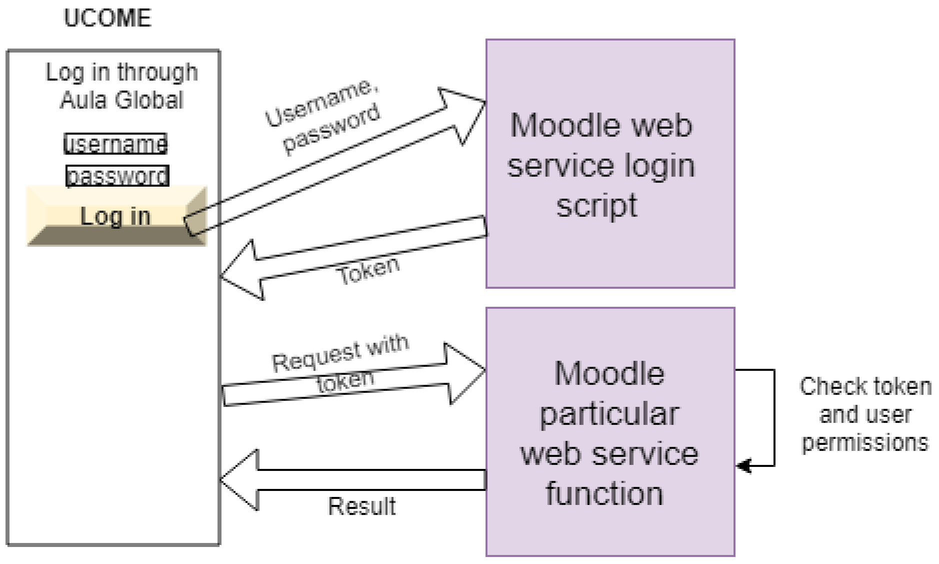

UCOME
Log in through Aula Global
username
password
Log in
Username, password
Moodle web service login script
Token
Request with token
Moodle particular web service function
Check token and user permissions
Result

Fig. 5.26. Moodle web services infrastructure functioning

5.4 Payment

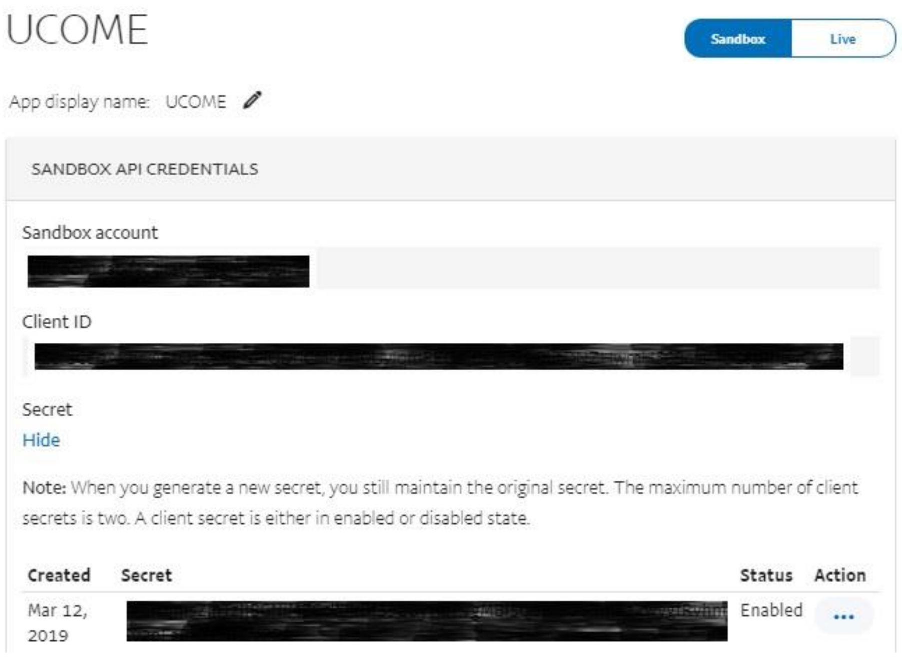
5.4.1 PayPal Sandbox

PayPal is a secure payment platform that was born because making online payments was insecure. Banking information was provided to online forms and someone (fraud online store or an external person) could access to this data. With PayPal the credit card is linked with your PayPal account and through an email and a password the payments are done, and no credit cards information is provided to the online stores.

PayPal Sandbox is used by developers to integrate PayPal in their applications. It allows to test payments with non-real money. When everything is working right, the Sandbox API Credentials are changed by the “good” ones (Live API Credentials) and the payments are now real. In this project only testing API Credentials are used.

When registering in PayPal Developer page two Sandbox Accounts are obtained: facilitator and buyer. Each one is identified by an email and a password and they simulate the PayPal vendor and buyer accounts.

Then, an App must be created in PayPal Developer page to receive REST API credentials for testing and live transactions (the ones mentioned before).


UCOME

App display name: UCOME

SANDBOX API CREDENTIALS

Sandbox account

[redacted]

Client ID

[redacted]

Secret

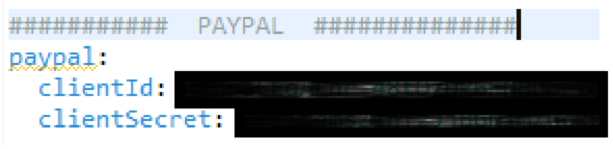
Hide

Note: When you generate a new secret, you still maintain the original secret. The maximum number of client secrets is two. A client secret is either in enabled or disabled state.

Created	Secret	Status	Action
Mar 12, 2019	[redacted]	Enabled	...

Fig. 5.27. PayPal Developer - App Registration

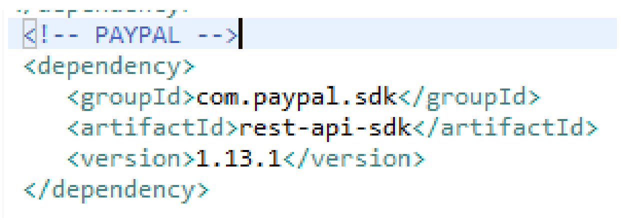
The Client ID and the Client Secret generated are placed into the `application.yaml` file to be used by the application. Each client company must put his app credentials that will be linked with the company’s PayPal selling account.



```
###########   PAYPAL   ##############
paypal:
  clientId: [redacted]
  clientSecret: [redacted]
```

Fig. 5.28. PayPal Credentials in application.yaml

A new dependency must be added to deal with PayPal items, transactions, payments, etc.

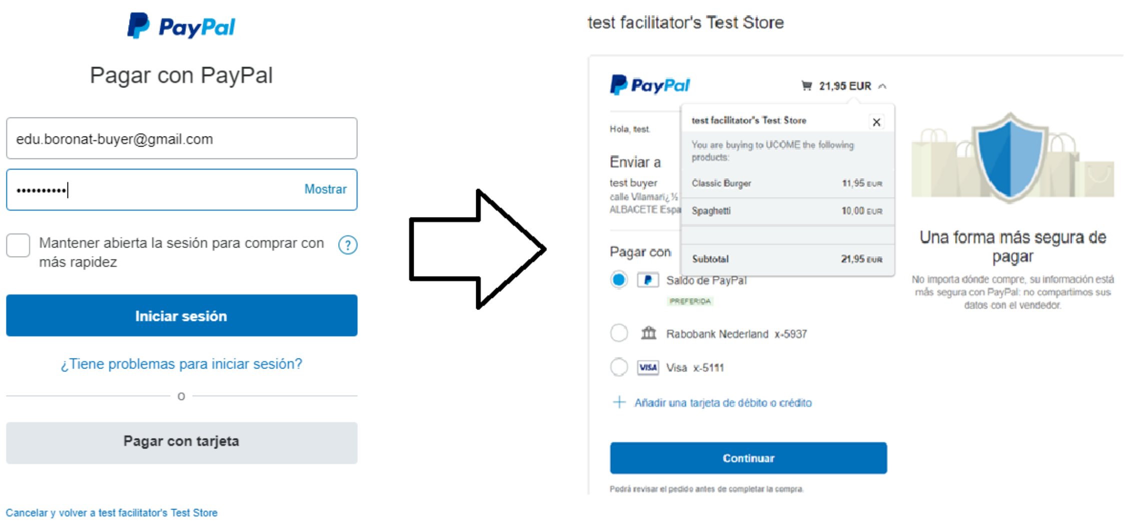


```
<!-- PAYPAL -->
<dependency>
    <groupId>com.paypal.sdk</groupId>
    <artifactId>rest-api-sdk</artifactId>
    <version>1.13.1</version>
</dependency>
```

Fig. 5.29. PayPal Dependency

In the Spring Boot project two classes are created to make this payment process. The first class will have the logic of building payments and transactions objects, while the second class is a controller that will catch the `/paypal/make/payment`, `/paypal/complete/payment` and `/paypal/cancelled` requests. The process is the following:

First, when all the chosen dishes are in the cart and the PayPal button is pressed a call to `/paypal/make/payment` is requested and caught by the controller. A new payment is created involving a transaction with the total price and the list of dishes; and the redirection URLs in case of success or error. A call to the PayPal API Server is done indicating the `clientId` and `clientSecret` established before in the `application.yaml` file and it will return an `approval_url`. UCOME will redirect the user to this `approval_url` where he will log in into his PayPal account (for testing is the buyer sandbox account previously created) and he will be able to confirm the order.


PayPal

Pagar con PayPal

edu.boronat-buyer@gmail.com

•••••••••• Mostrar

Mantener abierta la sesión para comprar con más rapidez

Iniciar sesión

¿Tiene problemas para iniciar sesión?

o

Pagar con tarjeta

Cancelar y volver a test facilitator's Test Store

test facilitator's Test Store

PayPal 21,95 EUR

Hola, test.

Enviar a

test buyer
calle Vilamari¿½
ALBACETE Espa

test facilitator's Test Store

You are buying to UCOME the following products:

Classic Burger	11,95 EUR
Spaghetti	10,00 EUR
Subtotal	21,95 EUR

Pagar con

Saldo de PayPal PREFERIDA

Rabobank Nederland x-5937

Visa x-5111

Añadir una tarjeta de débito o crédito

Continuar

Podrá revisar el pedido antes de completar la compra.

Una forma más segura de pagar

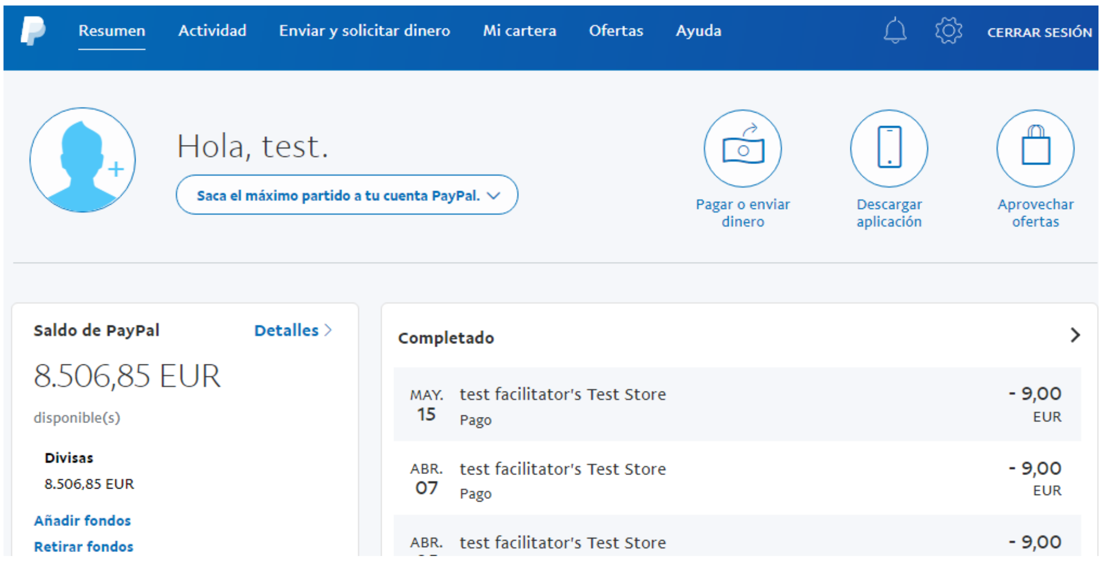
No importa dónde compre, su información está más segura con PayPal: no compartimos sus datos con el vendedor.

Fig. 5.30. PayPal payment confirmation

In the confirmation window the requested dishes can be seen, as well as their prices. It will also be possible to choose a payment method (PayPal Balance or credit card).

After confirming the order, PayPal will redirect to the success or error URL that was inserted into the payment object. If success, it will be `/paypal/complete/payment` that will receive two parameters from the PayPal API Server: `paymentID` and `payerID`.

Finally, a new call to the API is done with these two parameters to execute or complete the PayPal payment that the payer approved. Completed sandbox transactions can be seen in the Sandbox PayPal webpage (https://www.sandbox.paypal.com/signin), that is an identical instance of the live PayPal production site. Facilitator will see incoming payments and buyers will see their purchases.


Resumen Actividad Enviar y solicitar dinero Mi cartera Ofertas Ayuda CERRAR SESIÓN

Hola, test.

Saca el máximo partido a tu cuenta PayPal.

Pagar o enviar dinero

Descargar aplicación

Aprovechar ofertas

Saldo de PayPal Detalles

8.506,85 EUR

disponible(s)

Divisas

8.506,85 EUR

Añadir fondos

Retirar fondos

Completado

MAY. 15	test facilitator's Test Store Pago	- 9,00 EUR
ABR. 07	test facilitator's Test Store Pago	- 9,00 EUR
ABR.	test facilitator's Test Store	- 9,00

Fig. 5.31. Buyer Sandbox PayPal account

The previous detailed process to purchase a product can be summarized into the following image for a better understanding:

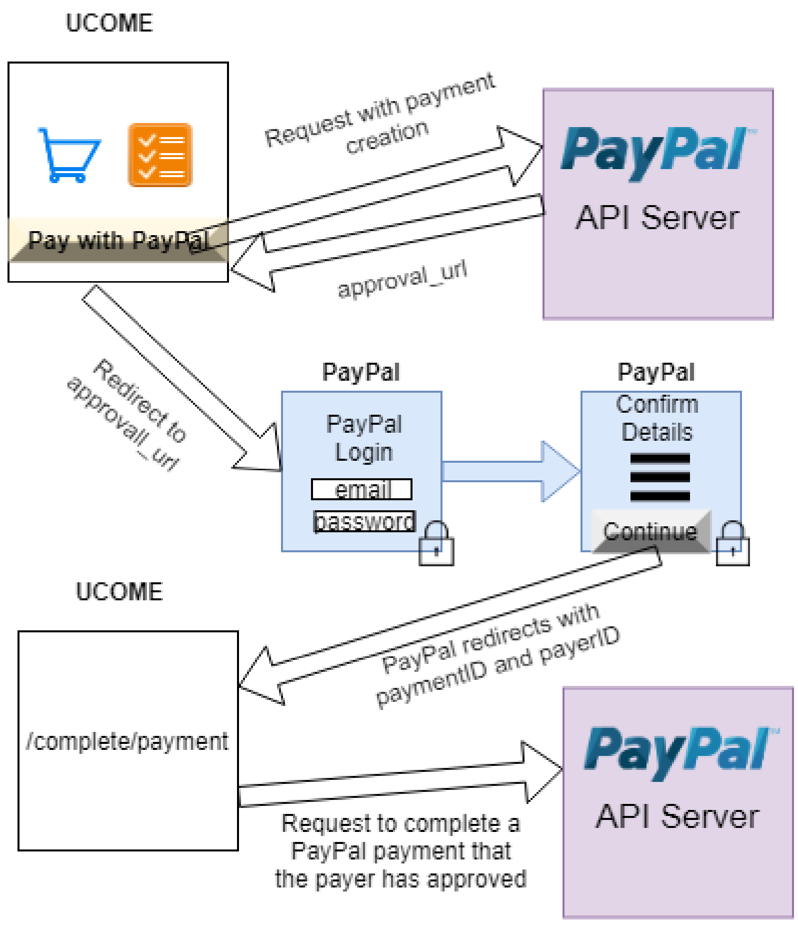

UCOME
Pay with PayPal
Request with payment creation
PayPal
API Server
approval_url
Redirect to approvall_url
PayPal
PayPal Login
email
password
PayPal
Confirm Details
Continue
UCOME
PayPal redirects with paymentID and payerID
/complete/payment
Request to complete a PayPal payment that the payer has approved
PayPal
API Server

Fig. 5.32. UCOME PayPal payment process

5.5 Order delivery

Once a purchase is made through PayPal it is registered in the database and it must be served by an employee of the restaurant/cafeteria. Therefore, the database record must be taken as soon as it goes in to start preparing the order.

For that, the replication log of the database is used. It records every change to the database. There are other methods like polling (see if something changed in the database every X seconds) or to make a trigger that executes a method when the database changes, but these ones are not good approaches because polling wastes resources and triggers hurts performance. The replication log is then the most robust one.

The following diagram explains all the process done:

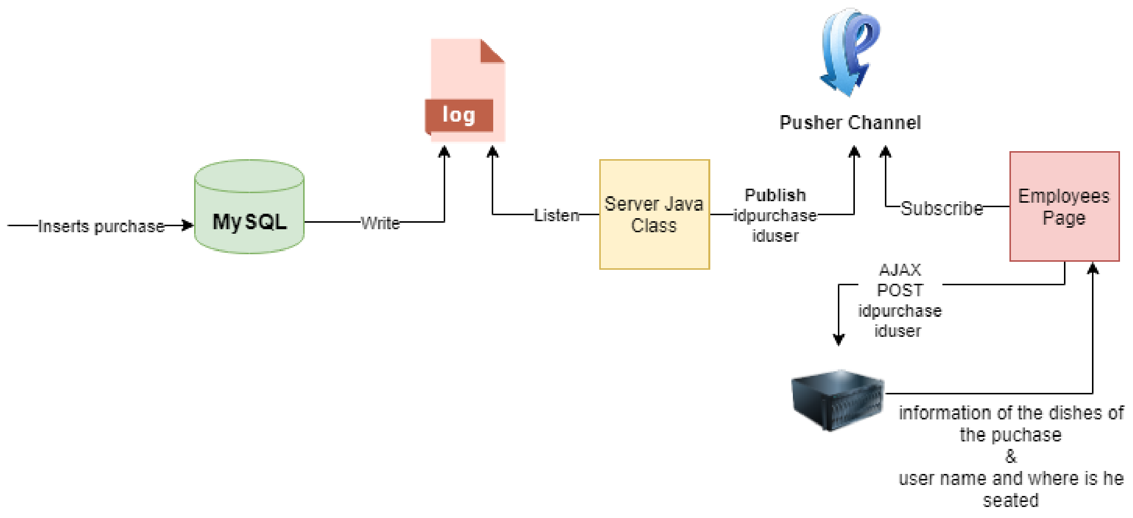

log
Pusher Channel
Inserts purchase
MySQL
Write
Listen
Server Java Class
Publish idpurchase iduser
Subscribe
Employees Page
AJAX POST idpurchase iduser
information of the dishes of the puchase & user name and where is he seated

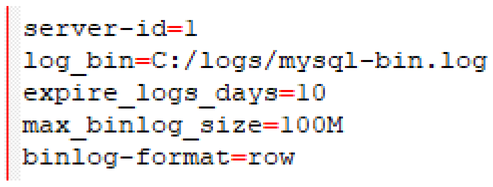
Fig. 5.33. Order delivery process

When a user buys the food, the purchase is inserted into the database and the replication log is written with this insertion. A java class will be listening to the log file in the background because it is executed as a thread and it will detect the new insertion. Then, this java class will publish to Pusher the `purchaseid` and the `userid` taken from the log. After that, the employees' page will subscribe in the Pusher channel and will obtain the two identifiers that will be sent through an AJAX call to the server in order to receive the information of the dishes and the user from it. The employee will see this information in the screen and will be able to carry the order to the seat where the user is.

The reason for using the replication log file and not to send directly the purchase information from the server to the employees' page is because in a future maybe it could be possible to make purchases from other server or other location and as long as the purchase is stored in the purchase table of the database, it will be detected and showed in screen. In addition, having a replication log is good in case there are any error in the database.

5.5.1 MySQL replication

For generating the replication log file, replication must be enabled in the MySQL configuration file `C:\ProgramData\MySQL\MySQL Server 5.5\my.ini:`



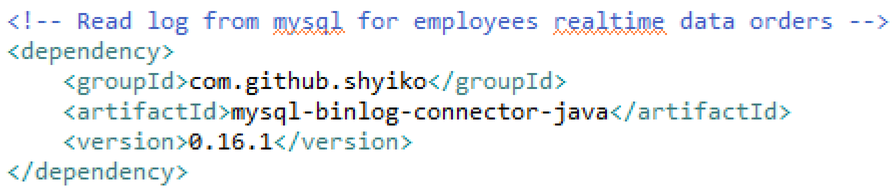
```
server-id=1
log_bin=C:/logs/mysql-bin.log
expire_logs_days=10
max_binlog_size=100M
binlog-format=row
```

Fig. 5.34. MySQL replication configuration

After restarting the MySQL server, it is possible to check if the replication is enabled with the command `show master status;` from the MySQL Workbench. If not, the following command should be executed:

```
GRANT REPLICATION SLAVE, REPLICATION CLIENT ON *.* TO
‘root’@’localhost’;
```

Finally, for reading the log file from the java class, one dependency must be used to provide the project with a connector.



```
<!-- Read log from mysql for employees realtime data orders -->
<dependency>
    <groupId>com.github.shyiko</groupId>
    <artifactId>mysql-binlog-connector-java</artifactId>
    <version>0.16.1</version>
</dependency>
```

Fig. 5.35. Replication log connector dependency

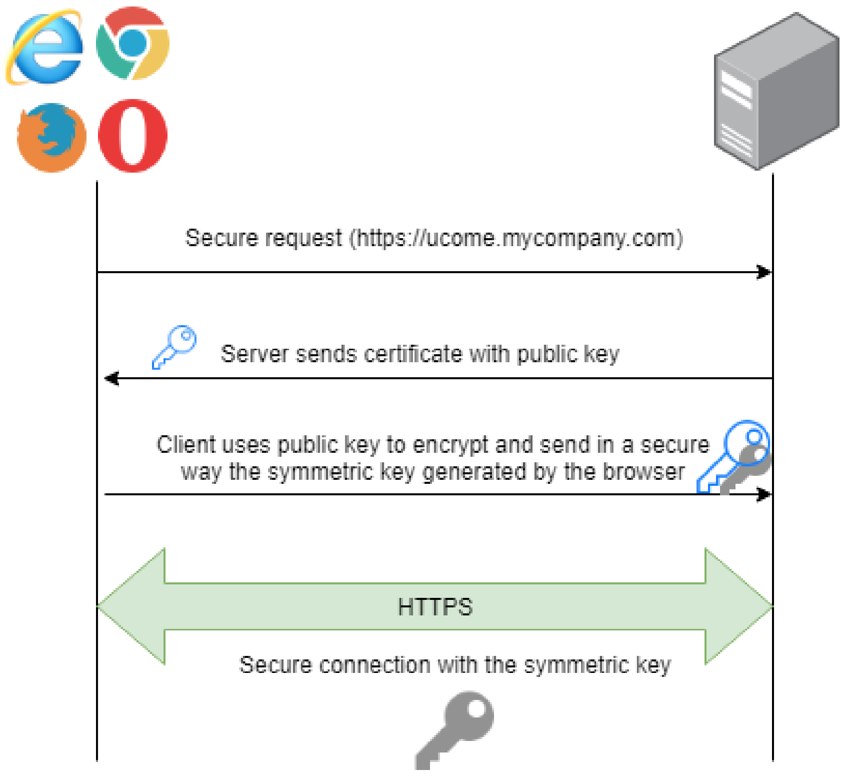
The java class will be connected to the log file through the following line of code:

```
BinaryLogClient client = new BinaryLogClient(“localhost”,
3306, “root”, “admin);
```

And the `client` will register an event listener that will detect the new insertions.

5.6 HTTPS

HTTPS is a secure version of HTTP (Hypertext Transfer Protocol). HTTPS is based on one of the two cryptographic protocols SSL/TLS that provide confidentiality and integrity between the endpoints of a communication. Both use a public key and a private key to encrypt the data.



Secure request (https://ucome.mycompany.com)
Server sends certificate with public key
Client uses public key to encrypt and send in a secure way the symmetric key generated by the browser
HTTPS
Secure connection with the symmetric key

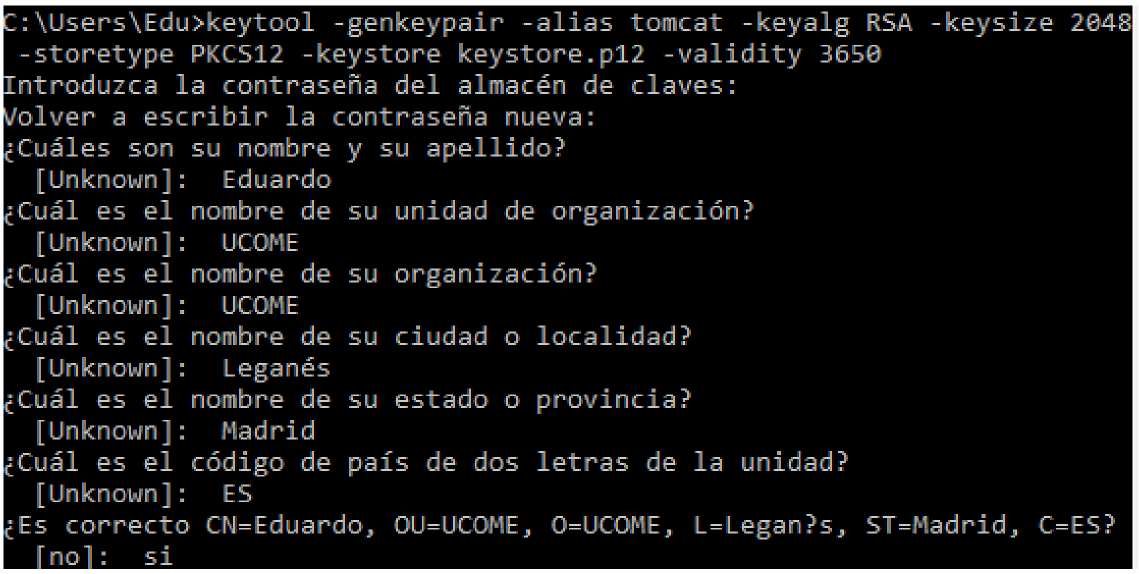
Fig. 5.36. HTTPS connection diagram

When the user makes a request through HTTPS, the server will respond with the SSL certificate that includes the public key. The browser checks that the Certification Authority (CA) is trustworthy and generates a symmetric key that is cyphered with the public key of the server. In this way, it is only possible to decrypt it with the private key of the server. When the server decrypts it, both sides have the same symmetric key and they will encrypt all the transmitted data with it.

Compared to HTTP, HTTPS makes it possible to ensure the server authentication and that the data exchange between the user and the web is encrypted (confidentiality), while ensuring the integrity of the transmitted data, preventing it from being intercepted and manipulated by a man in the middle.

The web applications run through HTTP by default. There could be UCOME clients that request the application running under HTTPS. They could configure the server where the application is deployed to use SSL, but if not, a configuration has been added to use UCOME under HTTPS. For that, an SSL self-signed certificate has been generated and it is used for testing purposes. As it is self-signed, the browser will show a security warning saying that it is not a trusted page because it is not issued by a Certificate Authority (CA). Clients should request one from a CA.

For creating the self-signed certificate Keytool is used, which is a certificate management utility that comes with every Java Runtime Environment (JRE). A pair of keys are created (public and private) for generating the SSL certificate and storing it in a keystore. The command used is the following:



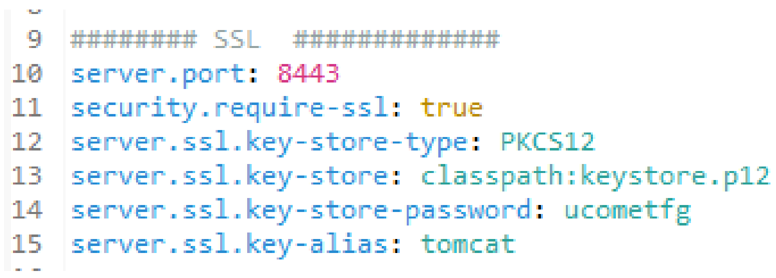
```
C:\Users\Edu>keytool -genkeypair -alias tomcat -keyalg RSA -keysize 2048
 -storetype PKCS12 -keystore keystore.p12 -validity 3650
Introduzca la contraseña del almacén de claves:
Volver a escribir la contraseña nueva:
¿Cuáles son su nombre y su apellido?
  [Unknown]:  Eduardo
¿Cuál es el nombre de su unidad de organización?
  [Unknown]:  UCOME
¿Cuál es el nombre de su organización?
  [Unknown]:  UCOME
¿Cuál es el nombre de su ciudad o localidad?
  [Unknown]:  Leganés
¿Cuál es el nombre de su estado o provincia?
  [Unknown]:  Madrid
¿Cuál es el código de país de dos letras de la unidad?
  [Unknown]:  ES
¿Es correcto CN=Eduardo, OU=UCOME, O=UCOME, L=Legan?s, ST=Madrid, C=ES?
  [no]:  si
```

Fig. 5.37. Keytool - Keystore generation

- -genkeypair: generates the pair of keys.
- -alias: indicates the alias of the certificate, which is used by the SSL/TLS layer.
- -keyalg RSA -keysize 2048: cryptographic algorithm used (RSA) and key size (2048 bits).
- -storetype PKCS12: keystore format type. PKCS12 is an industry standard format.
- -keystore keystore.p12: name of the keystore.
- -validity 3650: number of days in which the certificate is valid.

After executing above command, it will ask for some information as seen in the image. The generated keystore will be copied inside the `resources` folder of the project.

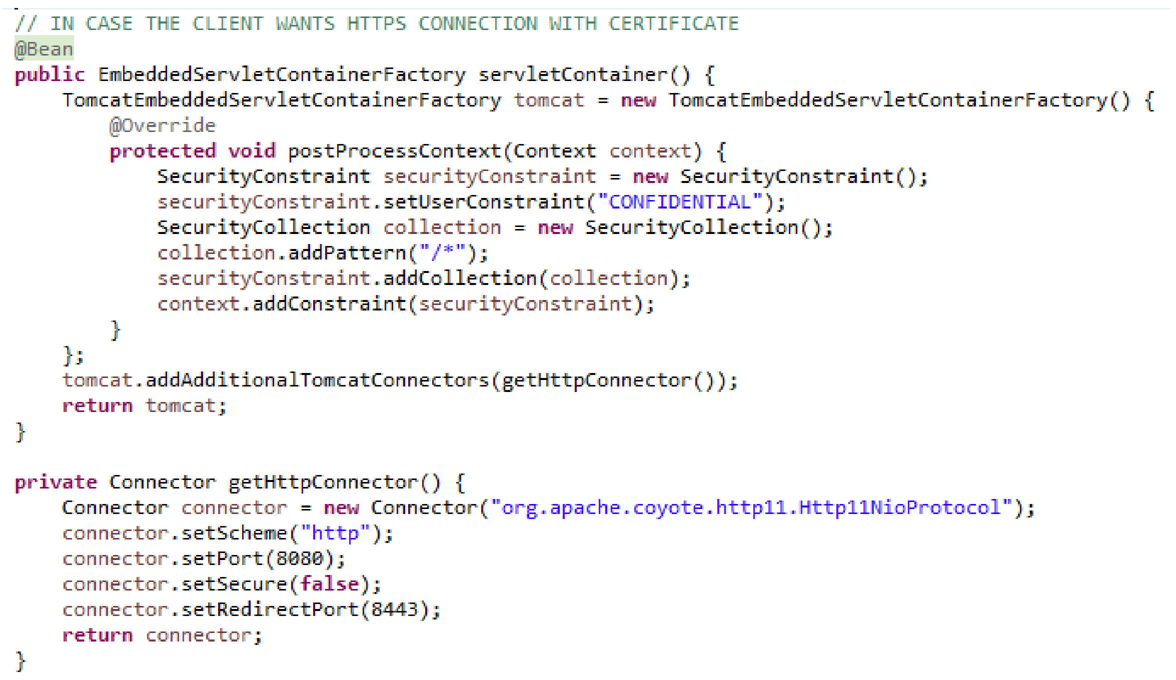
After that, the `application.yaml` file must be edited with the certificate information. The following image has the values of the testing certificate created before:



```
######## SSL  ##############
server.port: 8443
security.require-ssl: true
server.ssl.key-store-type: PKCS12
server.ssl.key-store: classpath:keystore.p12
server.ssl.key-store-password: ucometfg
server.ssl.key-alias: tomcat
```

Fig. 5.38. SSL configuration - application.yaml

HTTPS will be established through the port 8443, but some class configuration is added to redirect HTTP (8080) traffic to HTTPS (8443), so that the full site becomes secured. For that, below configuration is needed:



```
// IN CASE THE CLIENT WANTS HTTPS CONNECTION WITH CERTIFICATE
@Bean
public EmbeddedServletContainerFactory servletContainer() {
    TomcatEmbeddedServletContainerFactory tomcat = new TomcatEmbeddedServletContainerFactory() {
        @Override
        protected void postProcessContext(Context context) {
            SecurityConstraint securityConstraint = new SecurityConstraint();
            securityConstraint.setUserConstraint("CONFIDENTIAL");
            SecurityCollection collection = new SecurityCollection();
            collection.addPattern("/*");
            securityConstraint.addCollection(collection);
            context.addConstraint(securityConstraint);
        }
    };
    tomcat.addAdditionalTomcatConnectors(getHttpConnector());
    return tomcat;
}

private Connector getHttpConnector() {
    Connector connector = new Connector("org.apache.coyote.http11.Http11NioProtocol");
    connector.setScheme("http");
    connector.setPort(8080);
    connector.setSecure(false);
    connector.setRedirectPort(8443);
    return connector;
}
```

Fig. 5.39. HTTP to HTTPS redirection bean

A connector at 8080 port is added and redirected to 8443. Then, any request to 8080 through HTTP will be redirected to 8443 through HTTPS.

6 SOCIO-ECONOMIC ENVIRONMENT

6.1 Socio-economic impact

The world is continuously evolving around the new technologies and we cannot conceive it without them and their functionalities. These devices are part of our daily life, having a social, environmental and economic impact.

Web applications have been integrated into all kinds of services such as restaurants, transports, shops, etc. Any business uses these applications or web pages. They help us to perform daily tasks and to reach a bigger audience, while also allowing us to manage work in real time.

Starting with the social impact, UCOME facilitates the clients of the catering businesses, to be able to manage their diet, to request and pay their food easily and quickly, and to choose their seat through the application. Therefore, it will contribute to the society with a better quality of the service in which it is being used.

Regarding the environmental impact, the computer emits between 52 and 234 gr of CO2 per hour considering a power between 80 and 360 watts [31]. Apart from that, as a software product has been developed, it does not require any other energy than electricity.

Finally, this application would have a significant economic impact if we some factors:

- The time that the restaurant would dedicate to find a table for its diners would be considerably reduced, thus being able to invest this time in other tasks such as preparing the food.
- Workers would be more productive because they would be more rested and less stressed.
- Diners would have more time to be consuming since the waiting time would be greatly reduced. Therefore, there would be a greater number of sales (coffee, desserts...).
- There would be a greater control over the dishes that are consumed in the restaurant, being able to adjust the budget to the maximum when managing the purchases, preventing the restaurant from running out of stock.
- On the other hand, the entrepreneur will need less employees, having a negative impact on the creation of jobs.

6.2 Budget

The cost of the web application described in this project amounts to **seven thousand eight hundred fifty-five euros.**

In this amount the time devoted to research and to write this document have been included, detailing the hours of work of the Software Developer and the Project Manager shown in the following table:

TABLE 6.1. COST OF REASEARCH TIME AND DOCUMENT GENERATION

Main parts	**Hours**	**Cost per hour**	**Total**
1. Introduction	Software Developer=8	8 €	64 €
	Project Manager=2	45 €	90 €
2. Status of the question	Software Developer=20	8 €	160 €
	Project Manager=2	45 €	90 €
3. System analysis	Software Developer=10	8 €	80 €
	Project Manager=1	45 €	45 €
4. Development	Software Developer=270	8 €	2.160 €
	Project Manager=20	45 €	900 €
5. Socio-economic environment	Software Developer=10	8 €	80 €
	Project Manager=1	45 €	45 €
6. Conclusions and bibliography	Software Developer=5	8 €	40 €
	Project Manager=1	45 €	45 €
TOTAL	Software Developer=**323**	8 €	**2.584 €**
	Project Manager=**27**	45 €	**1.215 €**

The necessary hardware and software have also been considered for the budget calculation:

TABLE 6.2. HARDWARE COST

Hardware	**Cost per unit**	**Nº of units**	**Amortization**	**Total**
Personal computer	800 €	1	20%	**160 €**

An amortization is applied to the personal computer according to the duration of the project that is seven months. Then, an amortization of the 20% is applied.

TABLE 6.3. SOFTWARE COST

Software	Cost per unit	Nº of units	Total
Windows 10 EDU	0 €	1	**0 €**
Office 365 Education	0 €	1	**0 €**
Programs and tools	0 €	-	**0 €**

Fortunately, Windows 10 and Office 365 are free for students and all the software used was free because they were open source tools, or the free version was used.

In the same way, the cost of the programming time is detailed.

TABLE 6.4. PROGRAMMING COST

Programming	Hours	Cost per hour	Total
Spring Boot application	487	8 €	**3.896 €**

The Spring Boot application has 46 KLOC (thousands (kilo) of lines of code).

Adding all the totals, the cost obtained is:

TOTAL = 2.584 + 1.215 + 160 + 3.896 = 7.855 €

Finally, the VAT (value-added tax) must be added to the project. It is currently taxed at 21% in Spain.

7 CONCLUSIONS

7.1 Main conclusions

After having studied Spring, Spring Boot and the main modules of this framework; and having implemented a complete web application with it, I can conclude that a great simplicity is achieved with this framework and the modules that it offers are completely configurable and compatible with each other. This does not mean that it is the best option for developing web applications, but I would recommend considering it in most cases.

This end-of-degree project has been an opportunity, not only for acquiring knowledge, but for making me see that I am able to develop an entire application dedicating time and effort, what made me to achieve personal satisfaction. In addition, the development of the application has had a commercial purpose and I have focused a lot on business authentication mechanisms, what I find useful because I could find related tasks in my day to day work in the future as a computer engineer.

During the development I have realized that becoming an entrepreneur is not as difficult as most people think. If it is something you would like to be, it is enough to have a good idea and eagerness to constantly work.

Finally, although I have invested a lot of time and effort in the web application to be working as expected, trying to write good quality code and thinking about solutions when facing errors and requirements; it was worth it because I really liked it and it was a great challenge to face and achieve.

7.2 Level of goals achievement

Within the initial goals described at the beginning of the work, it can be concluded that all of them have been met. A complete system for improving the catering services has been developed.

The system has been developed in line having in mind all the most popular current internet trends such as integration with social networks, possibility to pay through payment platforms and authentication via most used mechanisms.

My idea when I started the project was to apply technologies and methodologies I have learnt in these years, so now, focusing in the methodologies, this project has been implemented using Scrum framework, using Trello as main tool since this is one of the most popular tools used by many organizations nowadays.

I have gone for Trello as I see it as a very intuitive and visual tool, based in Kanban cards that are added onto a board and shows the complete flow of each task implementation, moving them throughout the whiteboard when a stage is completed till they reach the “Done” status.

To conclude, with this project, I think I am more prepared to face real projects as I have learnt how to fit my tasks to timings, document them and challenge myself to reach the targeted objective.

7.3 Future work

Regarding future lines of work, the following improvements and supplements could be carried out:

- Development of an Android/iOS application.
- Visual design improvement in the seat management functionality.
- Option to pay in cash.
- Learn how to deploy in modern cloud computing platforms like Amazon Web Services (AWS), Docker or Openshift. It was tried, but it was only achieved with a simple Spring Boot application.
- Improvement in the Spring Security roles and privileges implementation.

A personalized treatment with customers will always be considered to know which authentication mechanism they would like to have and to help in the integration and deployment of the system. Small requested changes could also be implemented.

Regarding the commercialization of the application, there will be not a fixed price for all the companies. The price will be based on the number of seats that the restaurant/cafeteria has, and it depends also on the complexity of the personalized requests done, as well as the complexity of the application deployment in the company’s environment.

BIBLIOGRAPHY

[1] EDUCBA. “Spring vs Struts”. EDUCBA. https://www.educba.com/spring-vs-struts/ (accessed: Dec. 12, 2018).

[2] R. Petrusha, olprod and OpenLocalizationService. “Guía de .NET Framework”. Microsoft. https://docs.microsoft.com/es-es/dotnet/framework/ (accessed: Dec. 12, 2018).

[3] Ruby on Rails. “Getting Started with Rails”. Rails Guides. https://guides.rubyonrails.org/getting_started.html (accessed: Dec. 15, 2018).

[4] M. Fowler. “MicroservicePremium”. MartinFowler.com. https://martinfowler.com/bliki/MicroservicePremium.html (accessed: Dec 20, 2018).

[5] S. Brown, “Modular Monoliths”, presented in DevNexus, Atlanta, March 6, 2019. [Online]. Available at: http://www.codingthearchitecture.com/presentations/devnexus2016-modular-monoliths

[6] Pivotal Software. “Main Projects”. Spring. https://spring.io/projects (accessed: Feb. 6, 2019).

[7] Pivotal Software. “Modules”. Spring. https://docs.spring.io/spring/docs/3.0.0.M4/reference/html/ch01s02.html (accessed: Feb. 13, 2019).

[8] Maven. “What is Maven?”. Apache Maven Project. https://maven.apache.org/what-is-maven.html (accessed: Feb. 14, 2019).

[9] Mr. Noow. *Mr. Noow App Promo Video.* (July 26, 2017). Accessed: Feb. 17, 2019. [Video online]. Available at: https://www.youtube.com/watch?v=UJzTYlVjZNU

[10] Pivotal Software. “Web on Servlet Stack”. Spring. https://docs.spring.io/spring/docs/current/spring-framework-reference/web.html (accessed: Mar. 2, 2019).

[11] Tutorials Point. “Spring-MVC Framework”. Tutorialspoint. https://www.tutorialspoint.com/spring/spring_web_mvc_framework.htm (accessed: Mar. 2, 2019).

[12] I. Iborra and M.A. Lozano. “JSP Básico”. Jtech.ua.es. http://www.jtech.ua.es/j2ee/2006-2007/doc/sesion08-apuntes.pdf (accessed: Mar. 6, 2019).

[13] Thymeleaf. “Tutorial: Using Thymeleaf”. Thymeleaf. https://www.thymeleaf.org/doc/tutorials/2.1/usingthymeleaf.html (accessed: Mar. 6, 2019).

[14] S. Ajit. "Use Thymeleaf And JSP Simultaneoulsy In Spring Boot App". oodlestechnologies. https://www.oodlestechnologies.com/blogs/Use-Thymeleaf-And-JSP-Simultaneously-In-Spring-Boot-App/ (accessed: Mar. 26, 2019).

[15] Bootstrap. “Introduction”. Bootstrap. https://getbootstrap.com/docs/4.3/getting-started/introduction/ (accessed: Feb. 19, 2019).

[16] FreeHTML5.co. “Foodee: Restaurant Free HTML5 Bootstrap Template”. FreeHTML5.co. https://freehtml5.co/foodee-restaurants-free-html5-bootstrap-template/ (accessed: Feb. 19, 2019).

[17] Oracle. “Chapter 1 General Information”. MySQL. https://dev.mysql.com/doc/workbench/en/wb-intro.html (accessed: Feb. 10, 2019).

[18] Oracle. “Chapter 1 General Information”. MySQL. https://dev.mysql.com/doc/refman/5.5/en/introduction.html (accessed: Feb. 10, 2019).

[19] The jQuery Foundation. “JQuery API”. jQuery. https://api.jquery.com/ (accessed: Apr. 2, 2019).

[20] Pusher. “Channels overview”. Pusher docs. https://pusher.com/docs/channels (accessed: Apr. 20, 2019).

[21] O. Belmonte Fernández. “Programación Avanzada. Interfaces gráficas de usuario. Swing: Contenedores y componentes”. UJI. http://www3.uji.es/~belfern/Docencia/Presentaciones/ProgramacionAvanzada/Tema3/swing.html#1 (accessed: Mar. 29, 2019).

[22] campusMVP. “Cómo enviar correo electrónico con Java a través de GMail”. campusMVP. https://www.campusmvp.es/recursos/post/como-enviar-correo-electronico-con-java-a-traves-de-gmail.aspx (accessed: May 7, 2019).

[23] J.A. Castillo. “LDAP: Qué es y para qué se utiliza este protocolo”. Profesional review. https://www.profesionalreview.com/2019/01/05/ldap/ (accessed: Dec. 28, 2018).

[24] J. Mejía Viteri, M. Gonzáles Valero, and A. España León, “Gestión de Usuarios Con LDAP (Lightweight Directory Access Protocol) para el Acceso a los Servicios Tecnológicos y a la Información en las Empresas”, *JSR*, vol. 1, n.º CITT2016, pp. 10-15, Aug. 2016. [Online]. Available at: https://revistas.utb.edu.ec/index.php/sr/article/view/84/66. Accessed: June 2019.

[25] Apache Directory Studio. “The Eclipse-based LDAP browser and directory client”. Apache Directory. https://directory.apache.org/studio/ (accessed: Feb. 27, 2019).

[26] M. Yunus. “Online LDAP Test Server”. Forum Systems. https://www.forumsys.com/tutorials/integration-how-to/ldap/online-ldap-test-server/ (accessed: Feb. 28, 2019).

[27] zFlex Software. “Free zFlex LDAP Cloud Server”. zFlex Software. https://www.zflexsoftware.com/index.php/pages/free-online-ldap (accessed: Feb. 28, 2019).

[28] G. Thomas. “Common LDAP Properties and Script Attributes List with Examples”. Computer Performance. https://www.computerperformance.co.uk/logon/ldap-attributes-active-directory/ (accessed: Mar. 1, 2019).

[29] Google Developers. “Using OAuth 2.0 to Access Google APIs”. Google Identity Platform. https://developers.google.com/identity/protocols/OAuth2 (accessed: May 12, 2019).

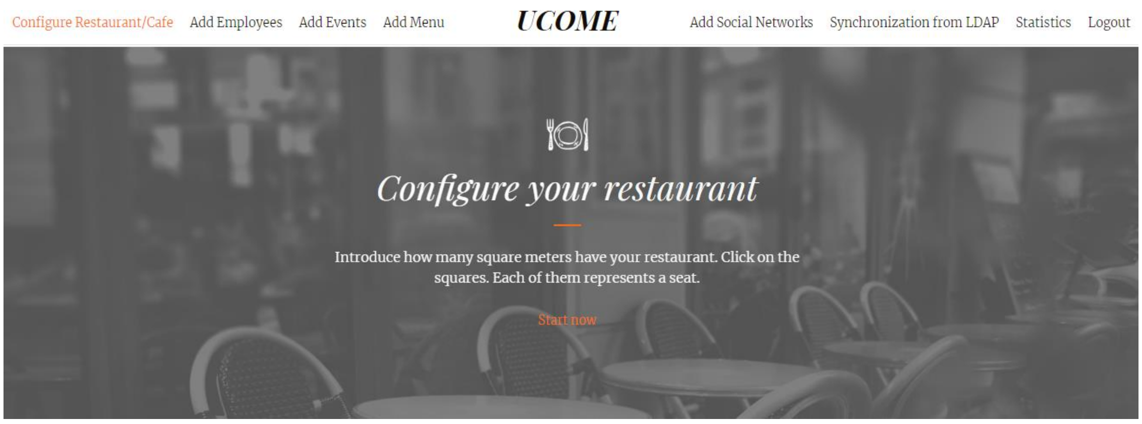
[30] Moodle. “Web services”. moodle. https://docs.moodle.org/dev/Web_services (accessed: May 17, 2019).

[31] Comisión Europea. “Lo que le cuestan sus electrodomésticos y cuánto CO2 emiten”. Comisión Europea. http://ec.europa.eu/clima/sites/campaign/pdf/table_appliances_es.pdf (accessed: Jun. 8, 2019).

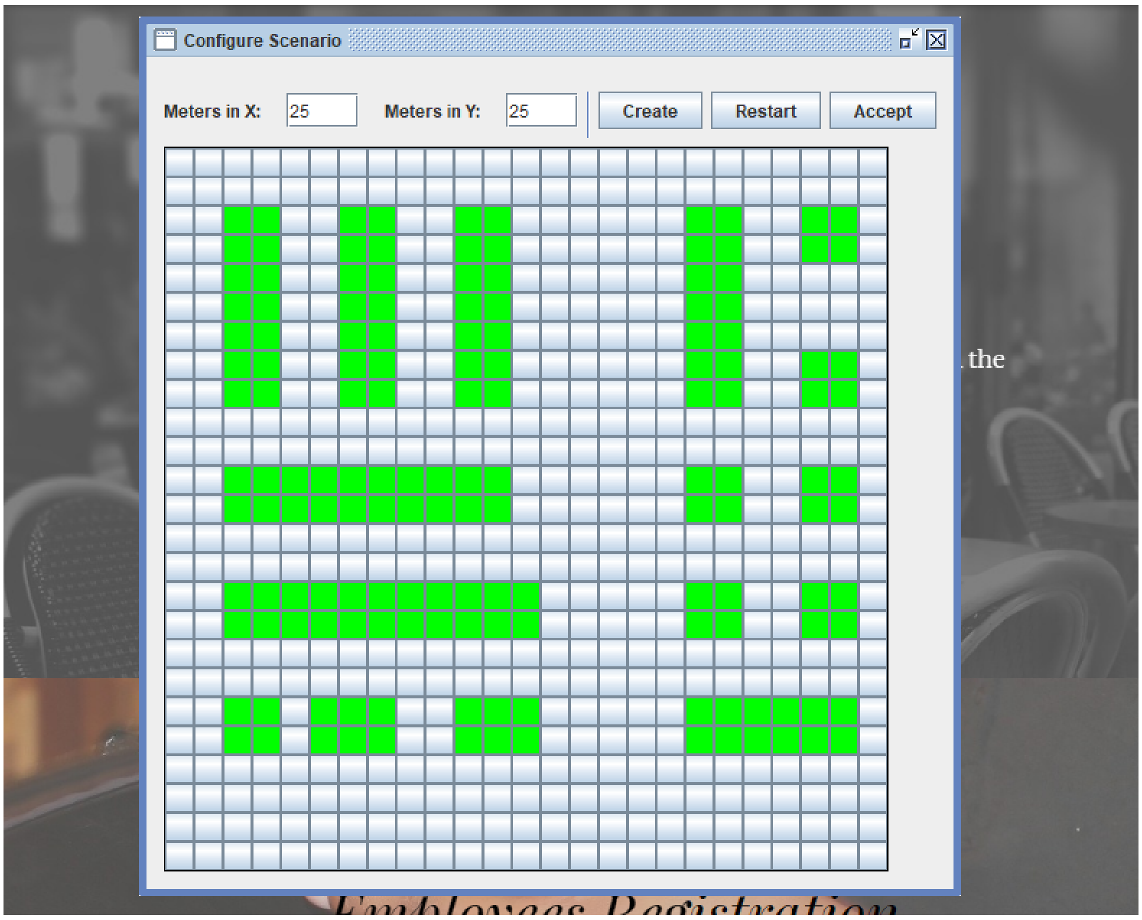
ANNEX A. ADMINISTRATOR MANUAL

When the administrator logs into the application, he will access to the administrator page where he will be able to perform the following actions:

Restaurant/cafeteria scenario configuration



Introduce how many square meters have your restaurant. Click on the squares. Each of them represents a seat.

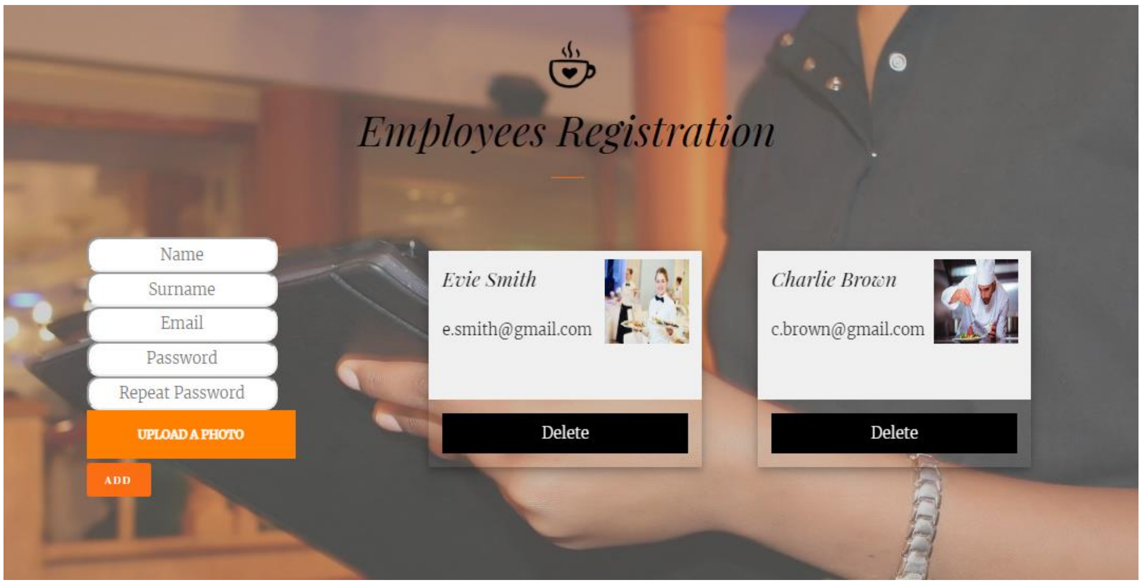


The administrator introduces the square meters of his restaurant and clicks on the squares. Each green square represents a seat the users will reserve.

Employees registration

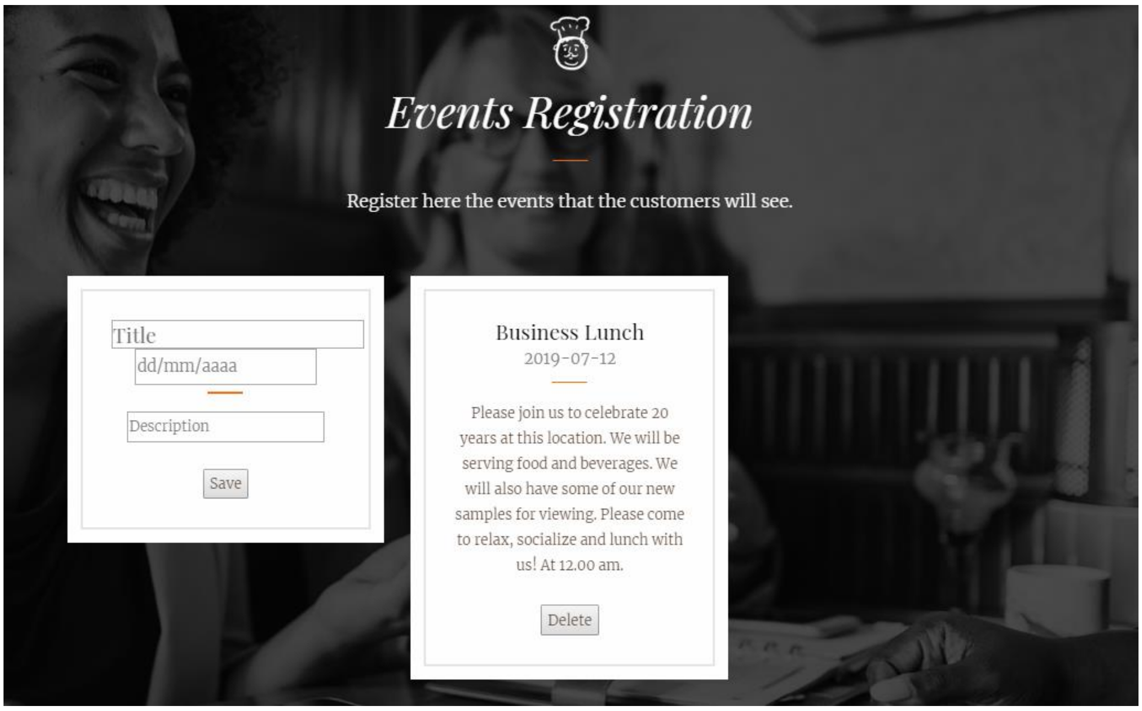
The administrator registers the employees of the business and they will be able then to access to the employees' page. An email with the password will be automatically sent to them.

It is also possible to delete an employee account from here.



Events registration

The administrator registers the events that will appear in the users' page. He will also be able to delete them.

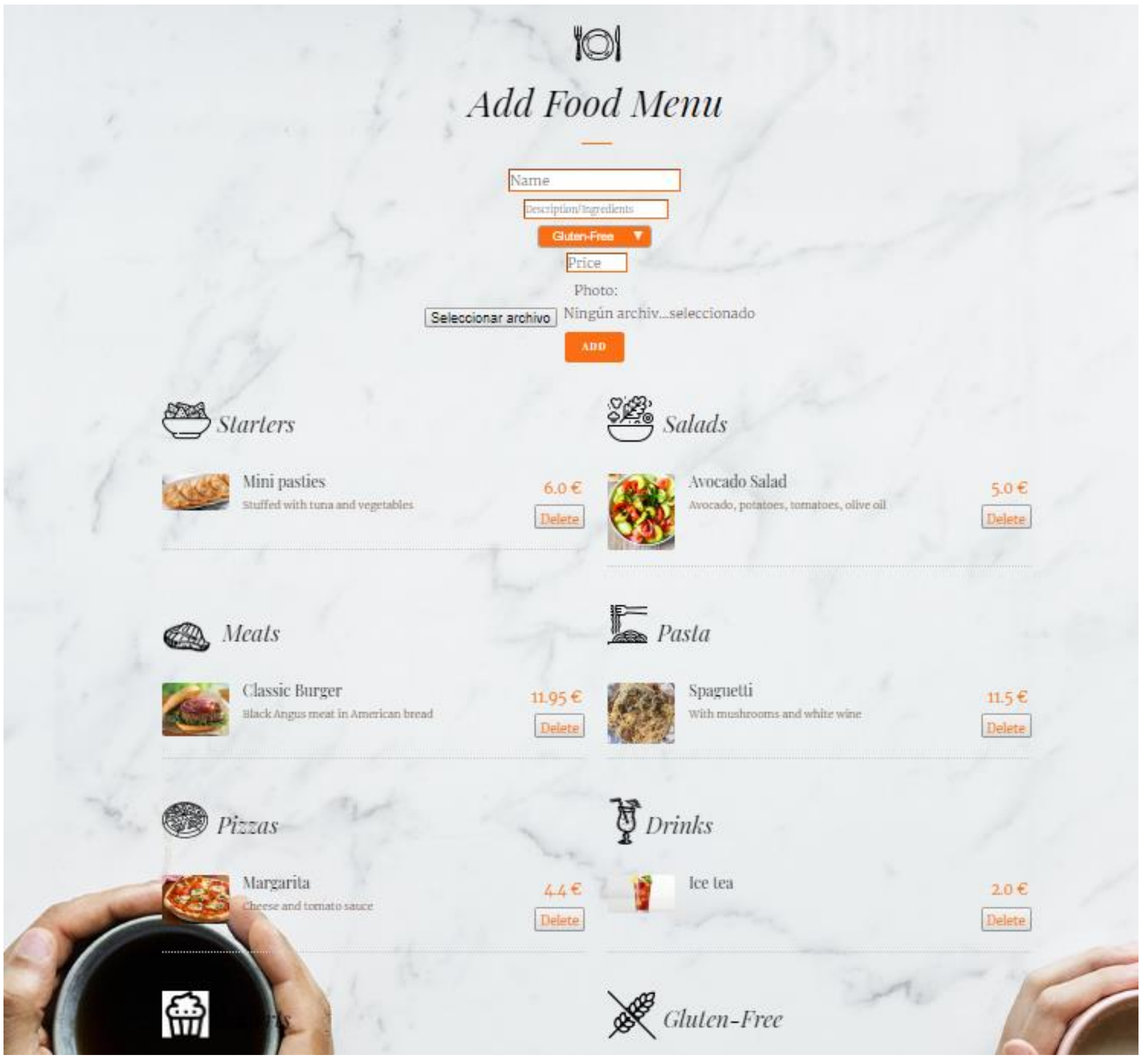


Please join us to celebrate 20 years at this location. We will be serving food and beverages. We will also have some of our new samples for viewing. Please come to relax, socialize and lunch with us! At 12.00 am.

Dishes registration

The administrator registers the food dishes that the users will buy. There are some types in which the dishes can be classified: starters, salads, meats, seafood, pasta, pizzas, vegetarian, drinks, desserts and gluten-free. If a type is not used, it will not appear in the food menu.

Dishes can be removed.



Add Food Menu

Name

Description/Ingredients

Gluten-Free ▼

Price

Photo:

Seleccionar archivo Ningún archiv...seleccionado

ADD

Starters

Mini pasties 6.0 €
Stuffed with tuna and vegetables
Delete

Salads

Avocado Salad 5.0 €
Avocado, potatoes, tomatoes, olive oil
Delete

Meats

Classic Burger 11.95 €
Black Angus meat in American bread
Delete

Pasta

Spaguetti 11.5 €
With mushrooms and white wine
Delete

Pizzas

Margarita 4.4 €
Cheese and tomato sauce
Delete

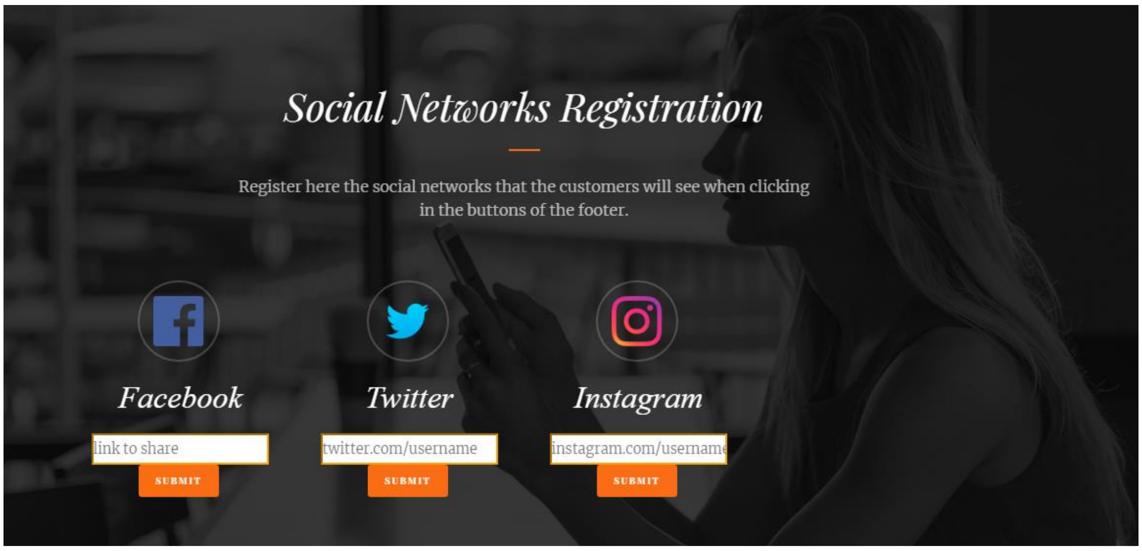
Drinks

Ice tea 2.0 €
Delete

Gluten-Free

Social networks registration

The administrator writes the profile links of the social networks in which the restaurant/cafeteria has an account and the users will access to the profiles by clicking on the logos that will appear on the footer of every pages. If a social network is not registered, it will not appear in the footer.

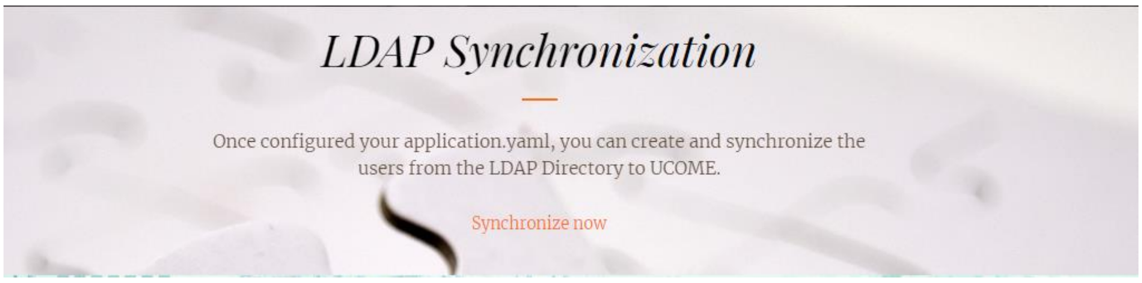


Social Networks Registration

Register here the social networks that the customers will see when clicking in the buttons of the footer.

Synchronize users from an LDAP Directory to UCOME

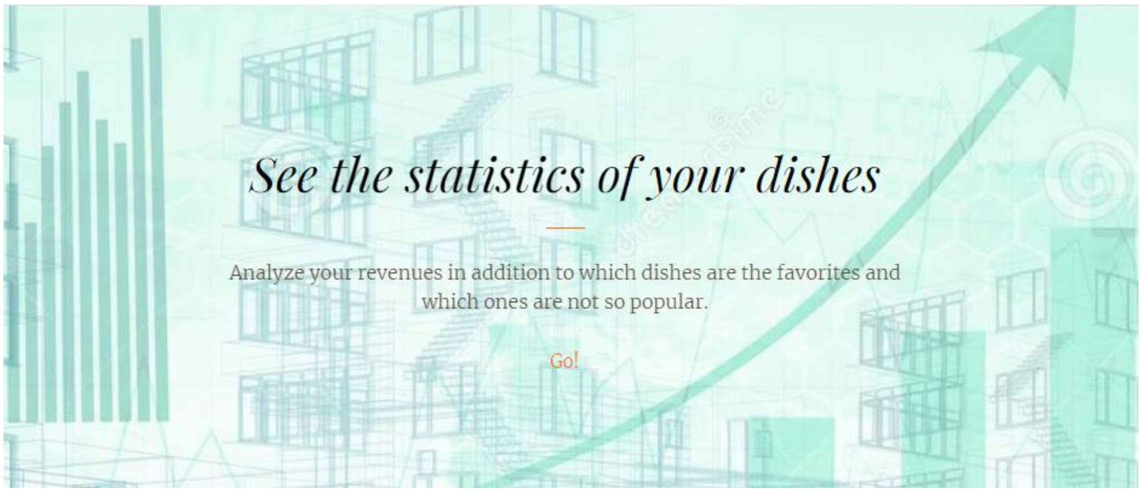
The new users (those that didn't exist in the application) will receive an email with the new generated password to access to UCOME, but they will now be able to log in through their LDAP credentials. If the user was already synchronized, no operation will be carried out, unless any field of his LDAP user was updated. Then, it will be updated in UCOME too.



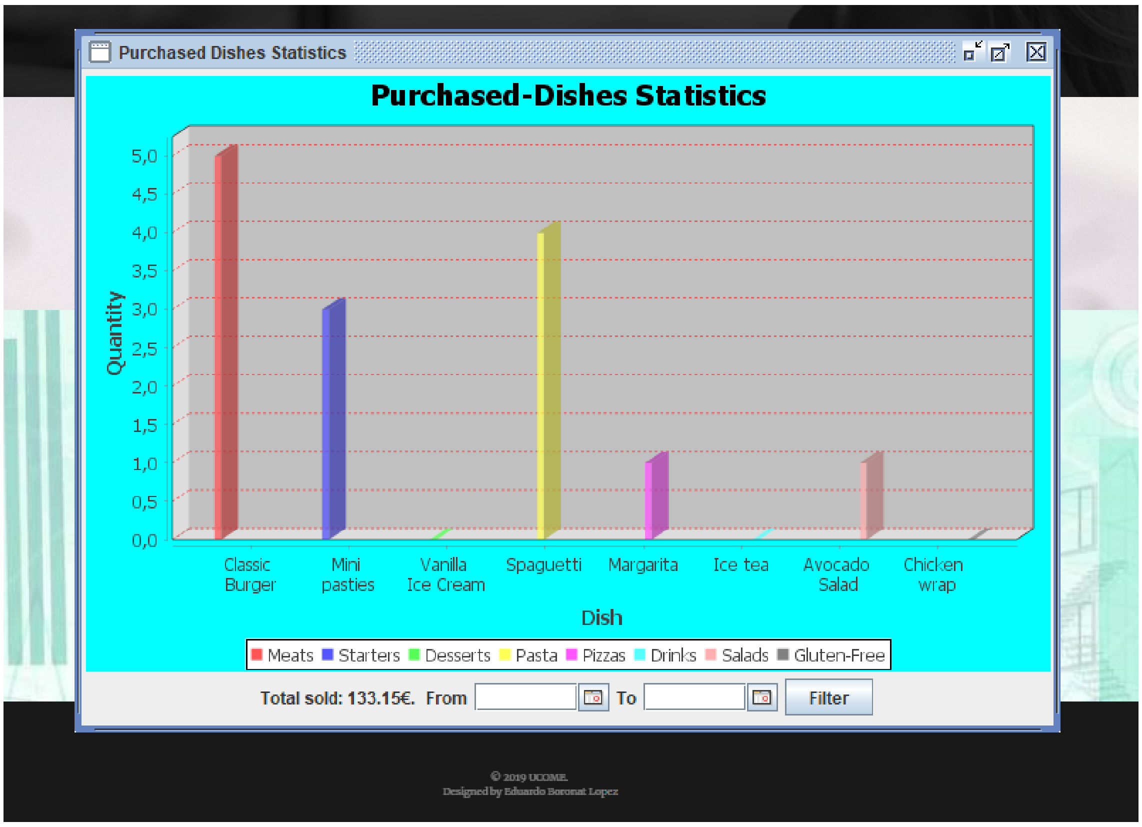
LDAP Synchronization

Once configured your application.yaml, you can create and synchronize the users from the LDAP Directory to UCOME.

Statistical control

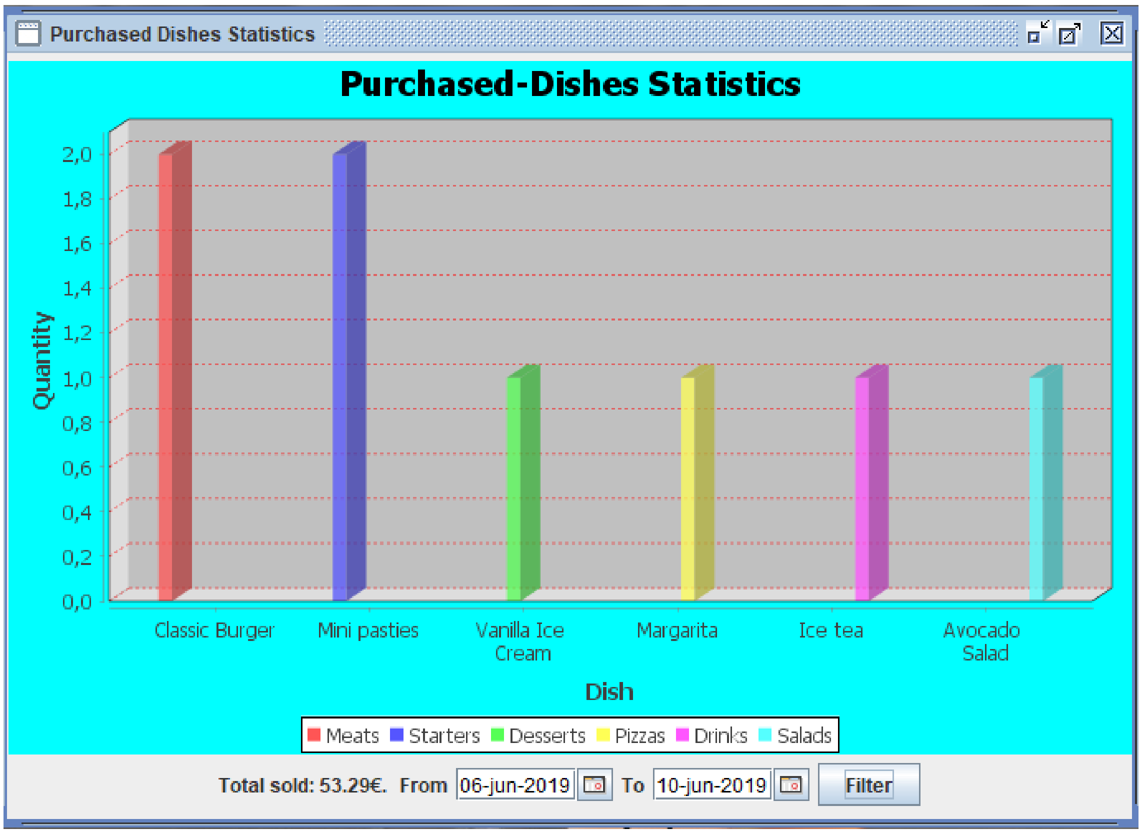

See the statistics of your dishes
Analyze your revenues in addition to which dishes are the favorites and which ones are not so popular.
Go!

The administrator sees all the dishes registered and the quantity sold of each of them. The total revenues are also displayed.


Purchased Dishes Statistics
Purchased-Dishes Statistics
Quantity
5,0
4,5
4,0
3,5
3,0
2,5
2,0
1,5
1,0
0,5
0,0
Classic Burger
Mini pasties
Vanilla Ice Cream
Spaguetti
Margarita
Ice tea
Avocado Salad
Chicken wrap
Dish
🟥 Meats 🟦 Starters 🟩 Desserts 🟨 Pasta 🟪 Pizzas 🟦 Drinks 🟥 Salads ⬛ Gluten-Free
From
To
Filter
© 2019 UCOME
Designed by Eduardo Boronat Lopez

Total sold: 133.15€.

When filtering by date, only the sold dishes and the revenues earned in that date range are displayed, allowing for example to see the revenues month by month.

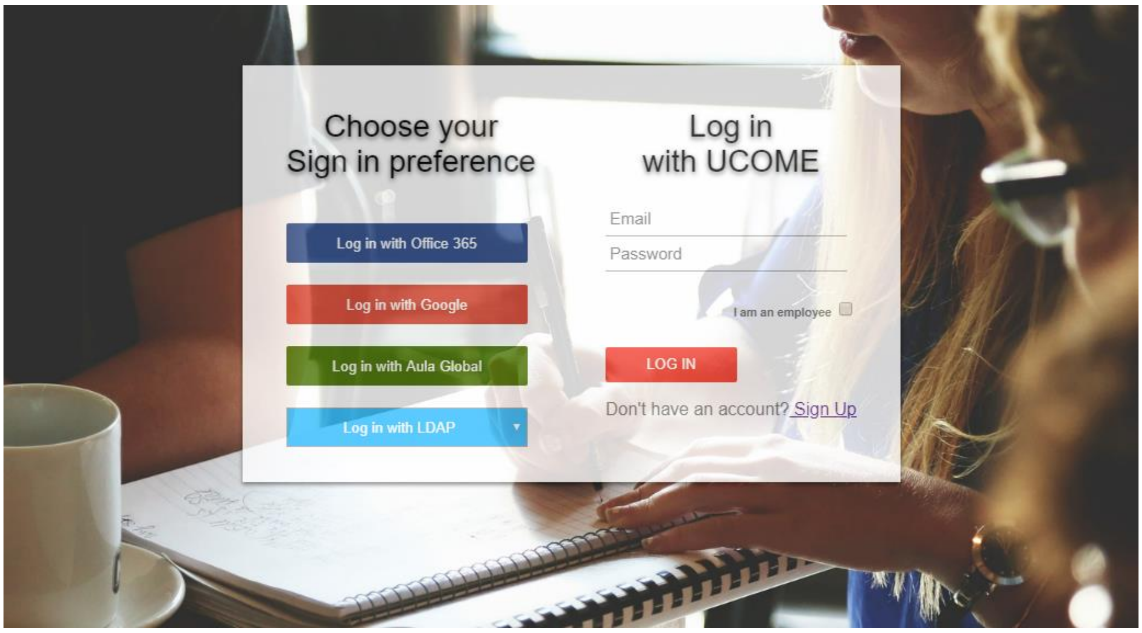


Total sold: 53.29€. From 06-jun-2019 To 10-jun-2019

ANNEX B. USER MANUAL

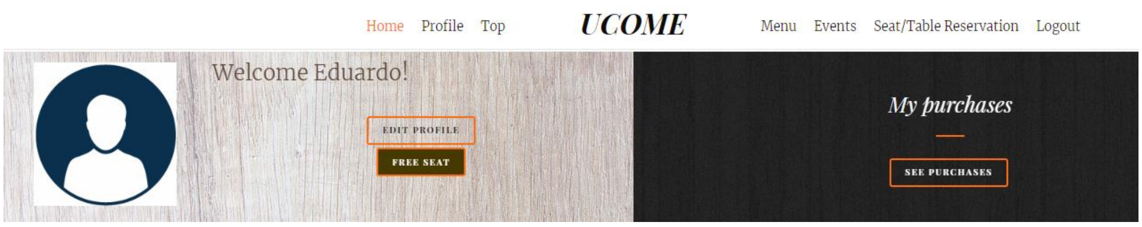
Login

The users can log in into UCOME through Office 365, Google, Aula Global, LDAP and UCOME; or sign up if they haven't done it yet.



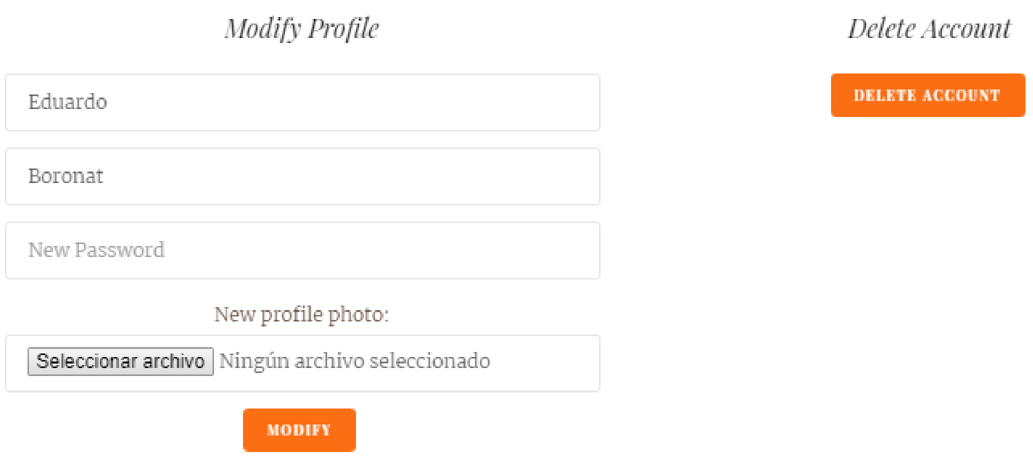
Free seat

The user clicks on the "Free seat" button when he has finished eating to make his seat available for others.



Modify profile

The user modifies his profile data or deletes his account.


Modify Profile

Eduardo

Boronat

New Password

New profile photo:

Seleccionar archivo Ningún archivo seleccionado

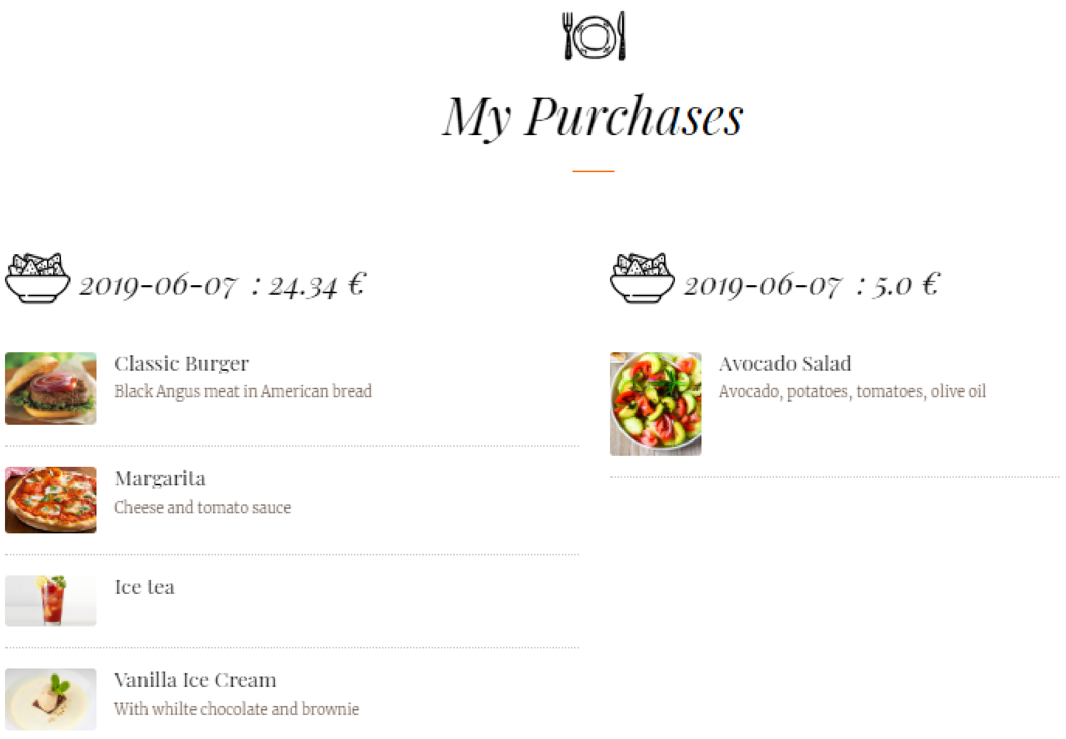
MODIFY

Delete Account

DELETE ACCOUNT

See purchases

The user sees all the purchases done, the date in which was done, and the price paid.


My Purchases

2019-06-07 : 24.34 €

Classic Burger
Black Angus meat in American bread

Margarita
Cheese and tomato sauce

Ice tea

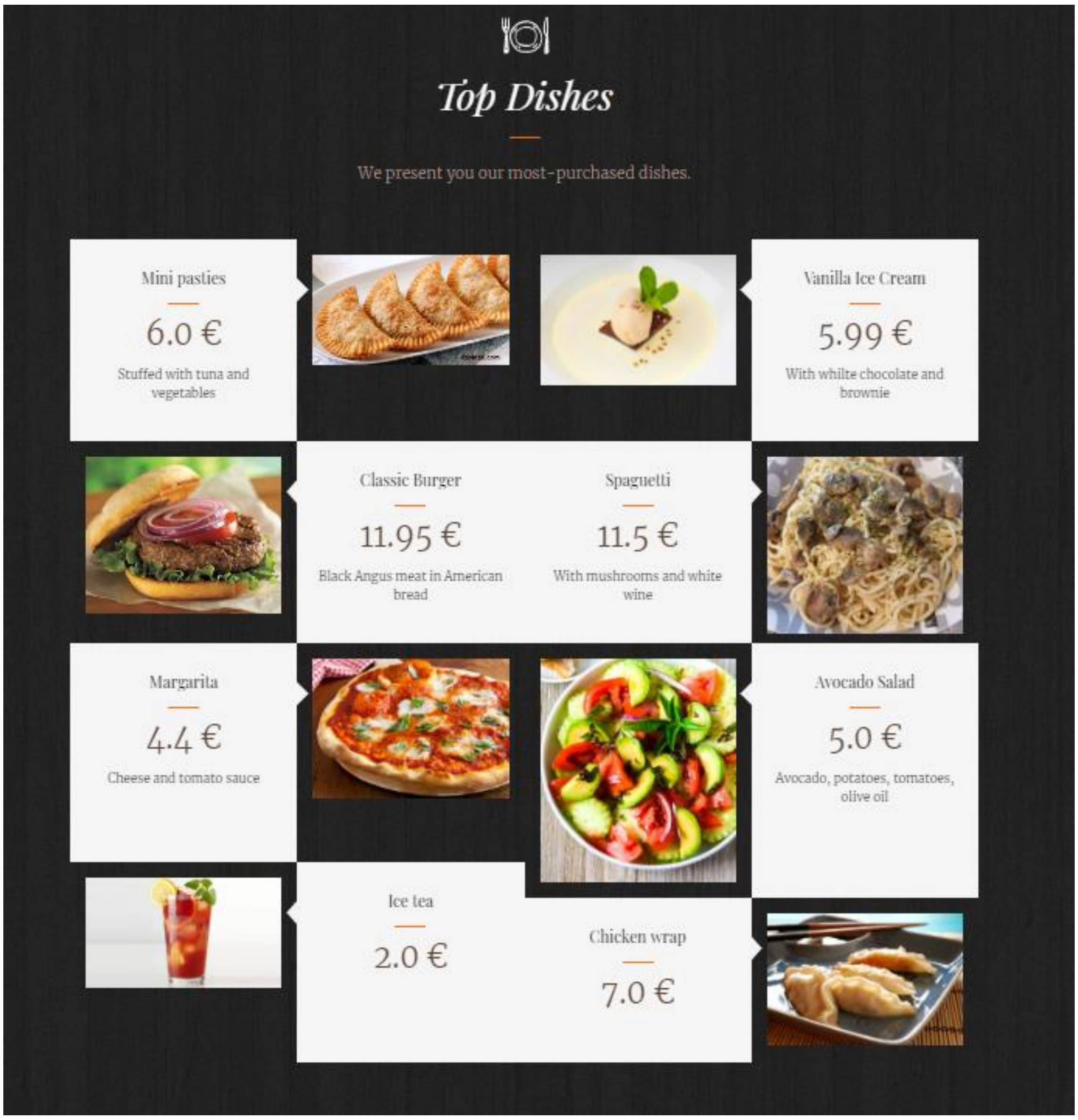
Vanilla Ice Cream
With whilte chocolate and brownie

2019-06-07 : 5.0 €

Avocado Salad
Avocado, potatoes, tomatoes, olive oil

Top dishes

The user sees the top 8 most purchased dishes.



Top Dishes

We present you our most-purchased dishes.

Mini pasties

6.0 €

Stuffed with tuna and vegetables

Vanilla Ice Cream

5.99 €

With whilte chocolate and brownie

Classic Burger

11.95 €

Black Angus meat in American bread

Spaguetti

11.5 €

With mushrooms and white wine

Margarita

4.4 €

Cheese and tomato sauce

Avocado Salad

5.0 €

Avocado, potatoes, tomatoes, olive oil

Ice tea

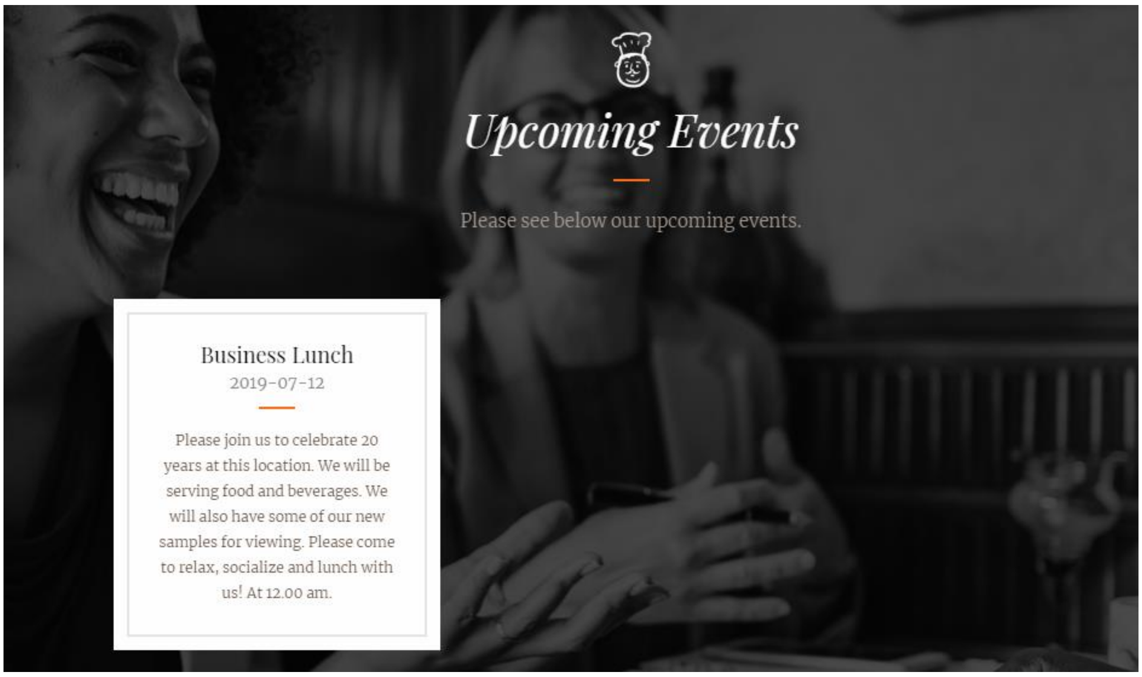
2.0 €

Chicken wrap

7.0 €

Upcoming events

The user sees the events that will be celebrated soon.



Upcoming Events

Please see below our upcoming events.

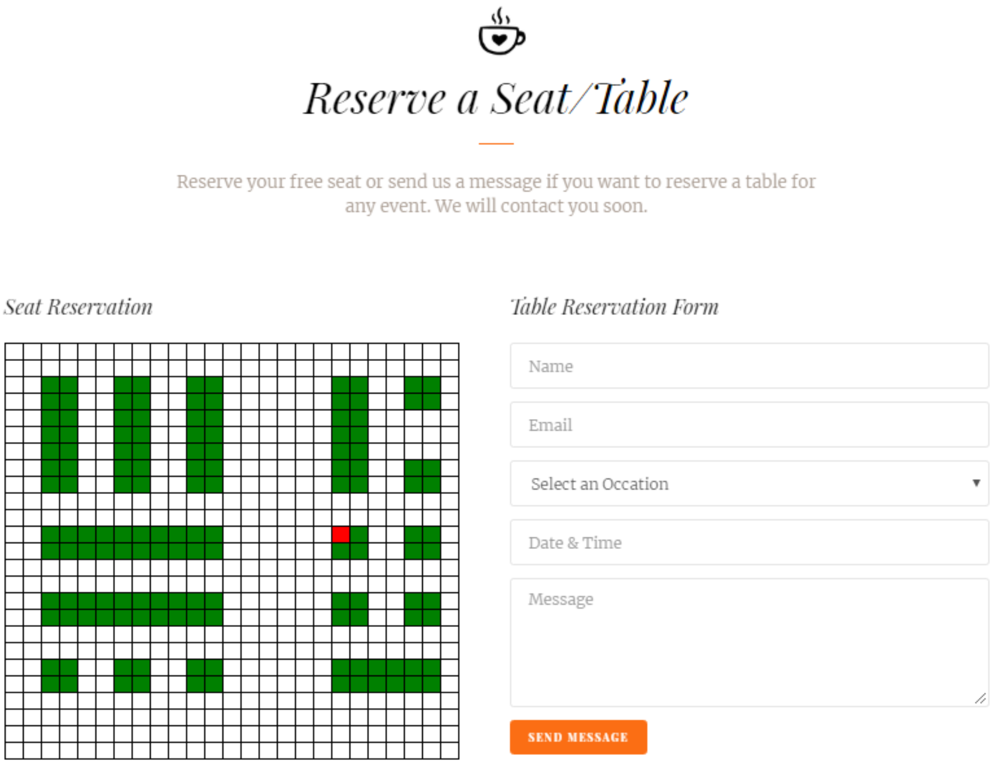
Business Lunch

2019-07-12

Please join us to celebrate 20 years at this location. We will be serving food and beverages. We will also have some of our new samples for viewing. Please come to relax, socialize and lunch with us! At 12.00 am.

Seat/table reservation

The green squares are the free seats that the user can click on to reserve that seat and the red ones are the occupied ones. A form is provided in case someone wants to reserve a full table or to celebrate any event there. An email is sent to the admin when submitting the form with the message and the information.



Reserve a Seat/Table

Reserve your free seat or send us a message if you want to reserve a table for any event. We will contact you soon.

Seat Reservation

Table Reservation Form

Name

Email

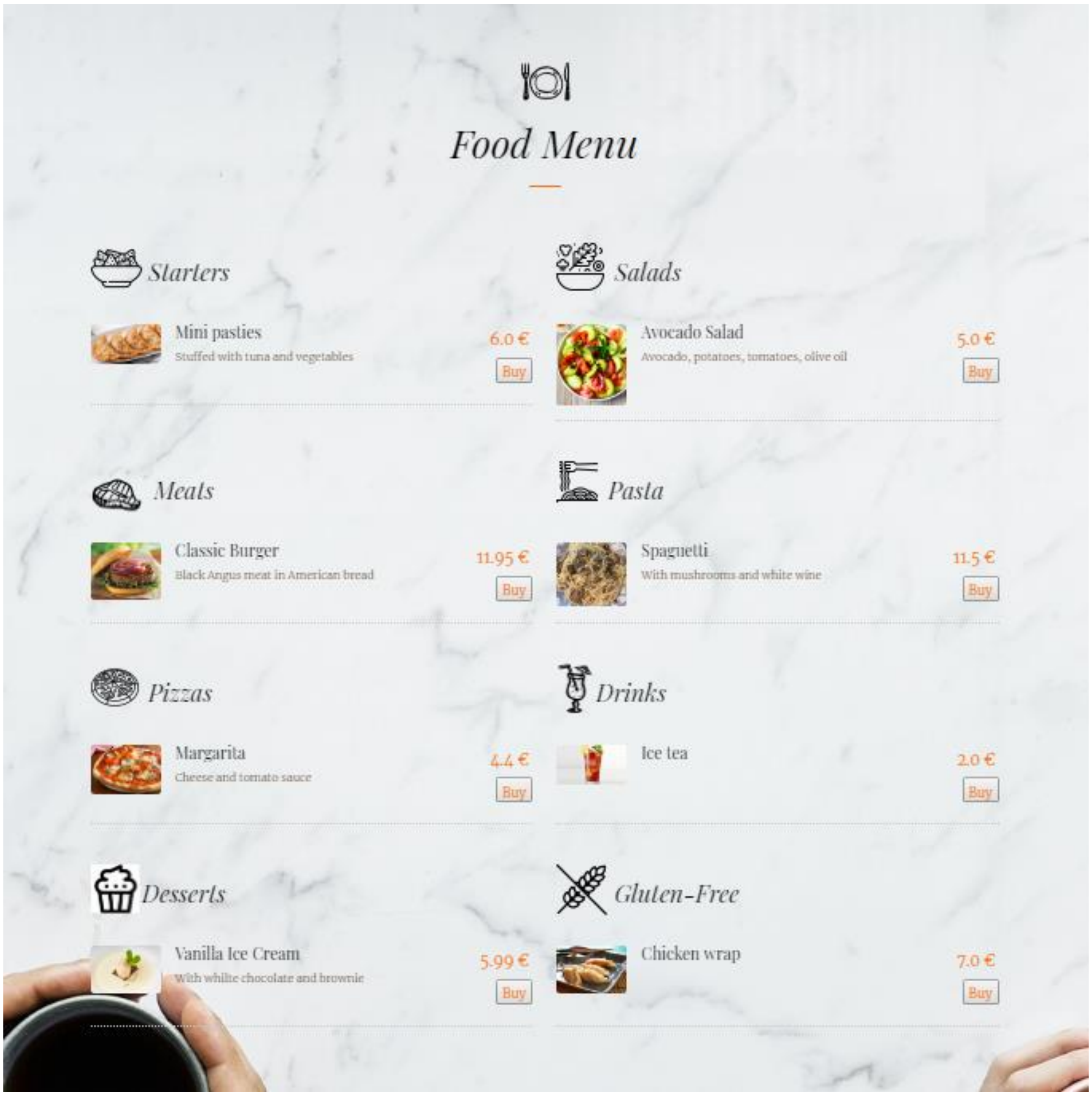
Select an Occasion

Date & Time

Message

Buy food

After reserving a seat, the user can buy the dishes and will be added into the cart.



Food Menu

Starters

Mini pasties
Stuffed with tuna and vegetables
6.0 €
Buy

Salads

Avocado Salad
Avocado, potatoes, tomatoes, olive oil
5.0 €
Buy

Meats

Classic Burger
Black Angus meat in American bread
11.95 €
Buy

Pasta

Spaghetti
With mushrooms and white wine
11.5 €
Buy

Pizzas

Margarita
Cheese and tomato sauce
4.4 €
Buy

Drinks

Ice tea
2.0 €
Buy

Desserts

Vanilla Ice Cream
With whilte chocolate and brownie
5.99 €
Buy

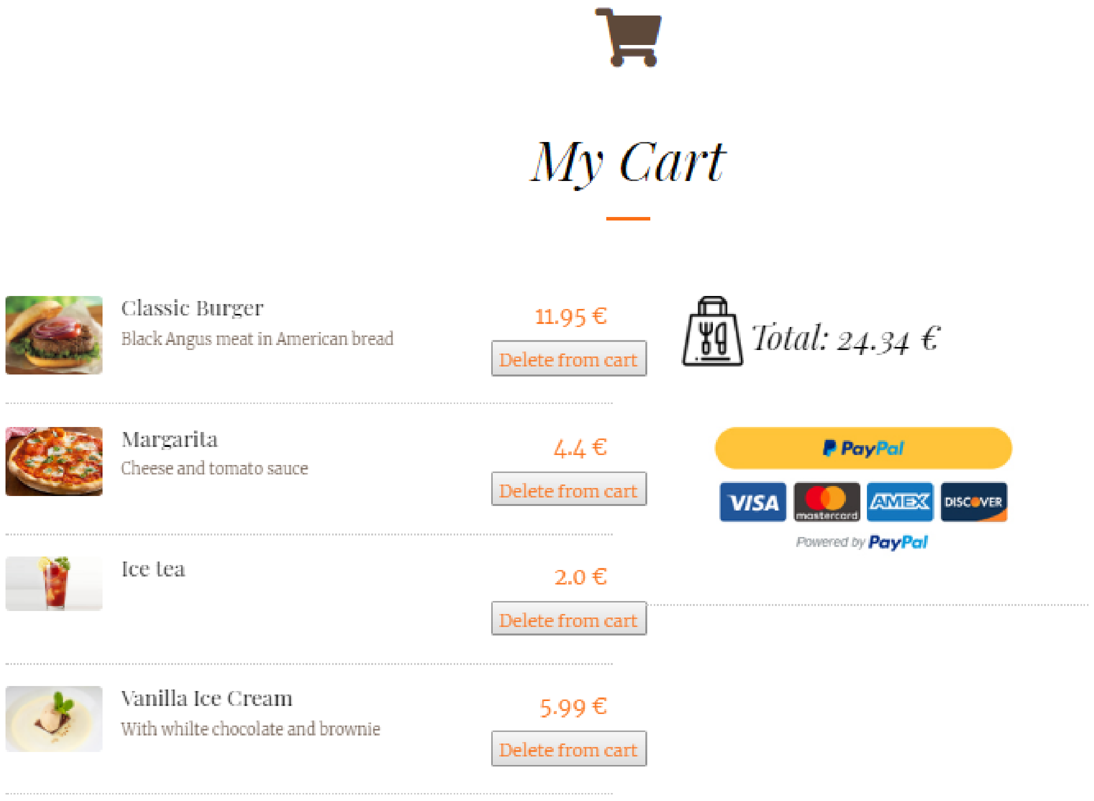
Gluten-Free

Chicken wrap
7.0 €
Buy

Payment

The user sees the dishes that he wants to buy in the cart. He can delete them from the cart.

The payment is carried out through PayPal.



My Cart

Classic Burger Black Angus meat in American bread	11.95 €	Delete from cart
Margarita Cheese and tomato sauce	4.4 €	Delete from cart
Ice tea	2.0 €	Delete from cart
Vanilla Ice Cream With whilte chocolate and brownie	5.99 €	Delete from cart

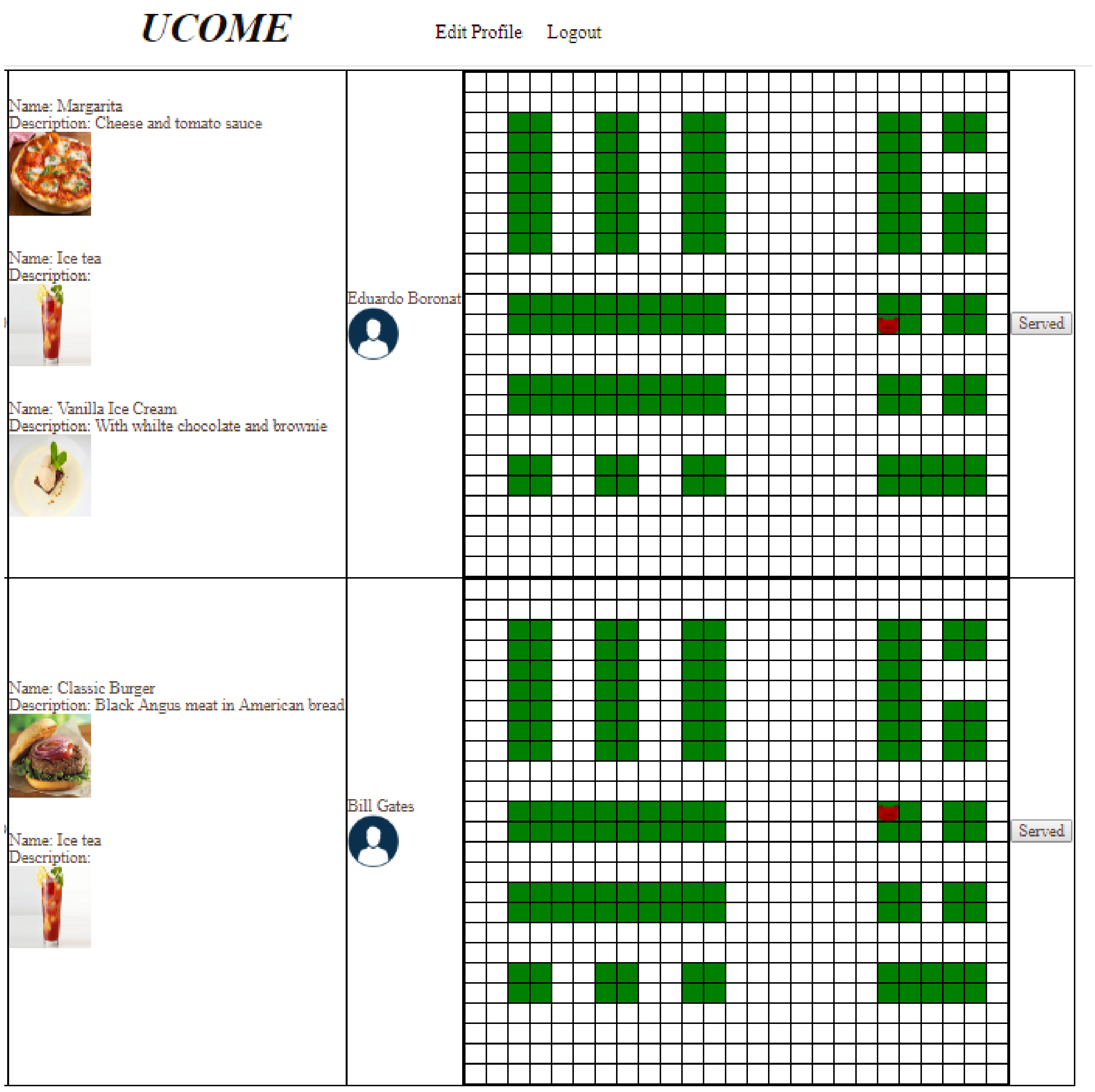
Total: 24.34 €

ANNEX C. EMPLOYEES MANUAL

When the employees log into the application, they will access to the employees' page where they will be able to perform the following actions:

Order delivery

When the users make their payments, the orders will automatically appear in the employees' page with some information: dishes ordered, name and photo of the user that made the payment and where is he located in the restaurant/cafeteria. With this information employees will be able to prepare the order and to take the order to the seat where the user is.



Name: Margarita
Description: Cheese and tomato sauce

Name: Ice tea
Description:

Name: Vanilla Ice Cream
Description: With whilte chocolate and brownie

Eduardo Boronat

Name: Classic Burger
Description: Black Angus meat in American bread

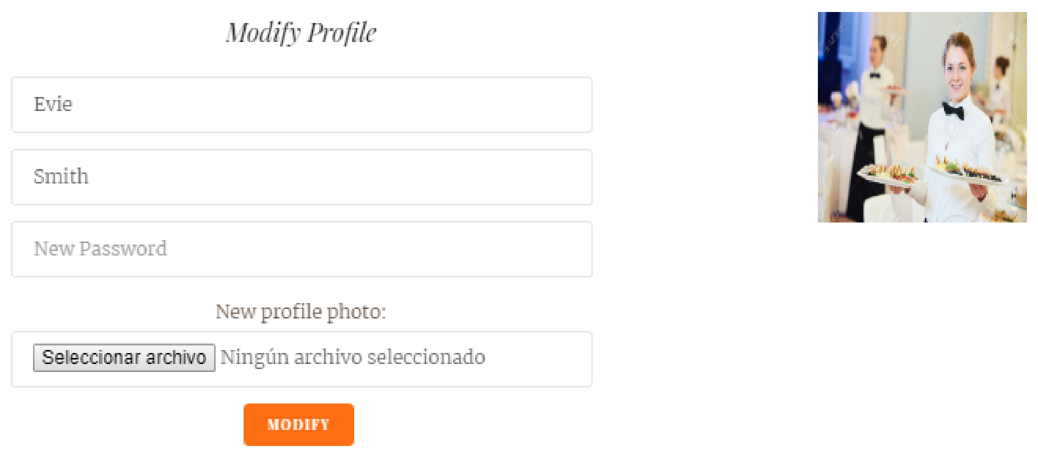
Name: Ice tea
Description:

Bill Gates

After clicking on the "Served" button, the order disappears.

Modify profile

Employees can modify their profile information.



Modify Profile

Evie

Smith

New Password

New profile photo: