MenuServe
Free·the·Service

Business and Development plan for a SaaS based Restaurant Menu System

Master Thesis
Màster d'Aplicacions multimedia
Area of Information Technologies

June 16th, 2016

Author:
Iñaki Bigatà Fages
Consultor:
Sergio Schvarstein Liuboschetz
Lecturer:
David García Solórzano

UOC
Universitat Oberta de Catalunya

Acknowledgements

Thanks to Guillem for helping me come up with the idea, for his knowledge of the guts of information technologies and for all the support.

Thanks to Raúl for everything I learn at work and all the support during “rush hours”.

Thanks to Simon for helping me with the touch that the logotype needed.

Thanks to Yago and Pol for introducing me to the world of applications, entrepreneurism and a lot more. And to Pablo for always demonstrating that the attitude is what matters.

Abstract

The hospitality sector is very well adapting to information technology, both at management and at customer reach. However, the management tools for restaurants are often too complex and very specific, making its implementation slow and expensive. This translates into that the vast majority of small and medium establishments do not use such systems or use only a tiny fraction of them, losing a lot of time and a lot of information that could contribute to the business.

Therefore, this project focuses on the design and prototyping of a restaurant management system on the cloud. This system will allow restaurants to manage all their information, points of sale, ordering system and kitchen, and even provide a tool to their customers so they can also place electronic orders, everything from any smart devices available.

The result is a platform accessible through the Internet, affordable and accessible from almost any device with an Internet connection. That way, restaurants will have a working integrated management system in less than a day and without leaving their premises. The use of devices for orders increases performance by two and the use of devices for customers could reduce by one third the cost of wait staff. Of course, all information generated accounts for key reports oriented to increase business performance.

The process of analysis and design has allowed us to validate the creation of a management system for restaurants and to determine the good accessibility of the system. Besides, the fact that we are offering a generic but adaptable and very accessible platform to restaurants allows us to differentiate on the market and could lead to a good reception within the sector.

Keywords

Restaurant Management System, Electronic Menu, Software as a Service, Multi-Platform.

Resum

El món de l'Hosteleria s'està adaptant molt bé a les noves tecnologies de la informació, tant per a la gestió com per la difusió al públic. Tot i així, les eines de gestió dels restaurants solen ser complexes i molt específiques, fent que la seva implementació sigui lenta i cara. Això fa que la gran majoria de petits i mitjans establiments estiguin molt limitats, perdent un munt de temps i informació que podrien aportar moltíssim al seu negoci.

Per això, que aquest projecte es centra en el disseny i prototipatge d'un Sistema de gestió en el núvol. Aquest sistema permetrà als restaurants gestionar tota la seva informació, els terminals de punt de venta, la cuina i les comandes, i fins i tot proporcionar una eina per a que els propis clients puguis fer comandes, tot des de dispositius electrònics propis o adquirits.

El resultat es un sistema disponible des d'internet, econòmic i accessible des de gairebé qualsevol dispositiu amb connexió a internet. Així, els restaurants podran disposar d'un sistema integrat de gestió en menys d'un dia i sense moure's del local. L'ús de dispositius per a les comandes augmenta per dos el rendiment dels cambrers i l'ús de dispositius per als clients podria reduir a un terç el cost en personal cara el públic. Per descomptat, tota la informació obtinguda permet obtenir informes clau per a augmentar el rendiment del negoci.

El procés de anàlisis i disseny, a més, permet valorar que la creació d'un sistema de gestió de restaurants per a tothom és viable i usable per als potencials clients. I, a més, el fet de ser un sistema genèric, adaptable i accessible per als restaurants, pot diferenciar-lo i podria aconseguir una bona acollida dins del sector.

Paraules clau

Sistema de Gestió de Restaurant, Menú Electrònic, Software com a servei, Multi-plataforma.

Table of Contents

List of Figures

List of Tables

1 Introduction

1.1 Introduction

The *Business and development plan for a SaaS based Restaurant Menu Management System* is a project that aims to the design and plan the development of a Management System for the Hospitality industry based on a Software as a Service business model and to analyse its business value in order to create a company that provides this services.

A Restaurant Management System (RMS) is a set of tools embedded in the same platform that allows Restaurant and other Hospitality Business Managers to centralize all their management needs in a single window. As other similar systems like a Human Resource System, a RMS allows different tasks like Managing the menus and orders, having an inventory, facilitate the accounting tasks or allocating employees in a schedule.

This kind of Management Systems until a few years ago, used to be specific, in house platforms that needed to be installed in a company server, and be maintained buy the costumer. This situation, although, is changing very quickly with the somewhat new technologies as Could Computing [1]. Cloud Computing makes reference to internet-based software that can be accessed from different devices sharing its information and configurations, making software “hardware non dependant”. With this concept, a platform as could be a Management System does not need to be stored and installed in the costumer system, it can be hosted by the company that offers the software and costumers will only pay a fee to use that software as it were a service. And here we have the Software as a Service [2] concept, allowing the costumers to have a wider range of action and performing the tasks with the desired platform without the needs of investment on hardware or maintenance that would make a system not worth to use.

There are some reasons why a project as *MenuServe* can be an interesting project within the *Master Universitari en Aplicacións Multimedia.* The main one is that concentrates almost all of the knowledge acquired during the curse of the subjects that this Master degree has. When engaging on a professionalizing Master Degree, the students seeks to gain knowledge and experience that will allow him to be better in his career and to evolve as a professional in the new technologies and business models that are revolutionizing the markets. Entering more in this concept of applying the knowledge acquired in the different subjects, the *BDRSM* project demands skills in strategic planning, project management, information and architecture design, promotion and web positioning, content providers, web and multimedia design, user oriented design, prototyping and so on.

BDRSM not only provides an excellent way of applying the acquired knowledge, will also allow the student to get deeper knowledge in new business models that are trend in the multimedia and software sectors, like SaaS. This business model based on SaaS has a set of advantages that can make it very profitable [3] like a reduced time for benefits, lower costs of implementation (delivery vs. service) and high scalability, and it has been adopted by the big companies of the internet sector as Google, Amazon, Microsoft or Sales Force. Then, performing a project that gets deeper knowledge of what is starting to be a proven effective way of doing business is a huge motivation.

BDRSM is interesting not only because offers the chance of creating a complex and viable software program; it opens a door to create a full brand and company that could made its way to real life and provides all the bases to uplift a career.

1.2 Definition

The main objective with the project *Business and development plan for a SaaS based Restaurant Menu Management System* is to design and plan a Restaurant Menu Management system that will focus on the process of petition and delivery of the meals to the costumers and managing the incomings and logistics that this actions my need (i.e. inventory, billing or kitchen order) by using a cloud platform that restaurants can take advantage of by using internet. This means, the restaurants will be able to use any hardware configuration that they want, as PCs, tablets, smart phones or laptops, to access the platform.

The design and plan the *Restaurant Menu Management System* (*MenuServe* from now on) allows the creation of a company that will manage, maintain and provide this platform as a service to the different costumers. Meaning that the project itself will not be only the design of this tool, but to create a Business Plan on how to manage the company and exploit this tool.

MenuServe will be, then, a software application or platform that will act as tool where restaurants and other Hospitality business will be able to manage all the logistics related to interact with their customers. Following this principle, MenuServe will allow the waiters to note the customer petitions of each table in a Smartphone or tablet, will allow the kitchen stuff to see those petitions in a computer, will allow to see the accountancy of the restaurant later on and will also let the restaurant have an "open" menu for the customers to directly ask for service or meals. Using a closed Wi-Fi network or QR Codes, customers of the restaurant will be able to "log in" to their table and see a menu and other options like asking for a waiter to come or asking the bill. Or they may be logging in by a tablet set up in each table of the restaurant. And of course, all of these options will be configurable by the manager of the establishment.

Usually, this set of platforms called Management Systems offers very specific functionalities to the Management Office of a business, and its use is usually limited to a small number of users or *managers* inside a company. There are plenty of Restaurant Management Systems that are already in use. But here is where *MenuServe* can differentiate itself from their competitors in order to acquire a better market share. *MenuServe* offers a totally new way of having access to a RMS by being based in Cloud Computing, where the restaurants can log in, manage their own site, setting up their own menu, and create users that will have this account as a "parent". That means that there can be different users for waiters or managers for example. And second big differentiator is the ability to open this software interaction with the clients, allowing the restaurant to provide an "Open Menu" to its customers so they can directly interact with the restaurant personnel, increasing the efficiency and reducing costs.

And last but not least, a cloud system provides the ability to obtain big data. This means being able to gather statistics and analytics in order to improve the restaurant revenue; get to know the most demanded products or dishes and the customer's tendencies or improve their business positioning.

The use of Service as a Cloud has a big impact in both *MenuServe* as a company and the restaurants as customers. As said before, the costumers can get a powerful tool without a big investment and be able to customize it for their own. And for *MenuServe* as a company, it will allow to offer different subscription plans regarding the size of the restaurant, or which modules it would use (i.e. a small business only needs the Menu ordering module but not the accountancy). Without forgetting about big data analytics, that could be used and sold as marketing and competitive intelligence.

1.3 Objectives

Once understood the concept, it will be easier to set up the goals for the *Business and development plan for a SaaS based Restaurant Menu Management System* project. *MenuServe* itself is a very ambitious project with a very wide scope, even if it is a defined and delimited concept. This is the main reason why to limit the work to be done in the Master Thesis.

The objectives for the project BPRMMS are:

1. Design a Restaurant Menu System that allows the restaurants and other hospitality to simplify the wait staff work and to offer a tool for management of the business with a cloud-based platform.
2. Establish the information architecture in order to have a "virtual" restaurant with the menu items, employees' users and metrics, establishments and table structures, kitchen orders, point of sale functionality, smart order menu and reports.
3. Establish the functionality and mechanism so MenuServe customers can create an "Open Menu" that can be used by their customers.
4. Establish a secure system to avoid undesired manipulation of the data by external or internal parties.
5. Use the knowledge acquired in the master's degree to establish the technologies that will be used to provide a wide range of hardware compatibility by using a cloud-base system.
6. Use the knowledge acquired in the master's degree to create a HQ prototype of the system that can show how the system would work in real devices and in real situations.
7. Obtain an analysis of the costs and viability of the project.

1.4 Project structure

The project is formed by a Thesis document that will gather all the information and requirements, and it is also formed by a prototype of the product.

The thesis will contain the following items:

- Market Research
  - Performing an analysis of the existent products and how the restaurants use them. This part of the project adds the State of Art and determine which the most important features to implement are.
- Functional design of the project:
  - Given the marketing research and the concept of *MenuServe*, the conceptual design will be formed by a list of specifications that will be set to the product, the main workflows that will have. In this stage, all the roles and their interaction will be defined, the main functionalities and requirements, phases of development and the life cycle. Prior to this, user stories will be used to explain the main decisions made.
- Information and system architecture
  - Data structure and technologies that will have MenuServe. This section will not only provide the information structure of menu items, employee's users and metrics, establishments, table, kitchen orders, point of sale functionality, smart order menu and reports but will relate all of them and create the diagrams to parameterize all the information. This means the definition of objects and attributes, and its properties. This section will also define which technologies will be used to create the product in order to provide a wide range of hardware compatibility and a secure environment.
- User interface design:
  - Using the information created until this point, the design of the platform will be defined. This design will take in accountancy the concept of the product, the user stories, the information structure and the existing standards in multimedia design in order to create the user interface.
- Viability and costs:
  - This last section will gather the estimations and costs of performing the actual project bases on the technology and requirements defined in the points above. By the costs that can arise to produce the product, and the design of monetizing plans, a viability conclusion can be set.

The prototype is a proof of concept system using the basis established in the Thesis. It will be used to validate the concept of *MenuServe* and provide a demonstration of what it can be. This phase will help to prove the concept of the product and to understand, alongside with the viability study, if it has the potential to become a reality.

1.5 Initial planning

The original planning of the project may be viewed on this following table.

Table 1 - Master's Thesis Plan

Task	Duration (days)	Start	End
PAC 3	27,00	29/03/2016	25/04/2016
Market research of RMS and MenuServe.	3,00	29/03/2016	31/03/2016
Functional design: Feature and requirement specification	7,00	01/04/2016	07/04/2016
Functional design: Cloud system and big data.	4,00	08/04/2016	11/04/2016
Definition of roles and workflows.	3,00	12/04/2016	14/04/2016
User Stories.	2,00	15/04/2016	16/04/2016
Risk assessment.	2,00	17/04/2016	18/04/2016
Definition of the technologies to be used and why (information, communication, authentication, etc.)	6,00	19/04/2016	24/04/2016
PAC 4	28,00	26/04/2016	23/05/2016
Information architecture specifications.	6,00	26/04/2016	01/05/2016
Application architecture specifications.	3,00	02/05/2016	04/05/2016
UI definition and wireframes.	3,00	05/05/2016	07/05/2016
High Quality designs.	9,00	08/05/2016	16/05/2016
UX Designs testing	2,00	17/05/2016	18/05/2016
PoC preparation (prototype).	5,00	19/05/2016	23/05/2016
PAC 5	21,00	24/05/2016	13/06/2016
PoC implementation (prototype).	13,00	24/05/2016	05/06/2016
Cost estimation	2,00	06/06/2016	07/06/2016
Monetization	2,00	08/06/2016	09/06/2016
Viability study	3,00	10/06/2016	12/06/2016
Thesis	76,00	29/03/2016	13/06/2016
Thesis writing	76,00	29/03/2016	13/06/2016
Corrections	1,00	13/06/2016	14/06/2016
Defending	1,00	14/06/2016	15/06/2016

1.6 Cost estimation of the project

The project of MenuServe is an ambitious set of modules that together, with a strong back end, offers a complete suite for Restaurant Management. The development of this project may become a costly task to perform, in terms of money, as it will involve more than one professional working alongside for a long period of time. Furthermore, in order to run a company like this, there will be more professionals involved on this, like a salesman, providers, marketing campaigns, etcetera,

The yearly development costs below are estimated for the initial stages of the project. As can be seen on table 2, the estimated yearly costs for the first year will be a bit over 200k€. This will include the development of an MPV, running real time tests, advertisement and acquisition of the first clients. Below, a detailed justification of the estimated expenses is provided. To do so, the costs have been divided in four different categories.

Category	Concept	Yearly cost
Human Resources	Front end developer	40,000.00 €
	Back end developer	40,000.00 €
	Sys admin	40,000.00 €
	Salesman	50,000.00 €
	Designer	5,000.00 €
Operational	Office expenses	15,600.00 €
	HW expenses	900.00 €
	Material	5,000.00 €
	Licences	150.00 €
Marketing	Online ads	8,400.00 €
	Campaigns	3,600.00 €
Administration	Administration	500.00 €
Total		209,150.00 €

Table 2 - Cost estimation

Development

For the development of the SaaS products, a front end and a back end developers are required. Given the proposed salary, the potential developers to be hired are expected to have about 5 years of experience. The same applies for the systems administrator.

A salesman is also accounted in order to negotiate and arrange deals with restaurants. Part of its salary will depend variables, therefore it's budget is a little bit higher.
Finally, an external graphical designer will be hired to design the front-end.

Operational

This section includes the expenses required to operate the company. For the office expenses, an office of 90m2 in the middle of Barcelona has been considered, including electricity, wifi and others.
As hardware expenses, for the first years Amazon Web services is the best suitable approach, since it allows pay per use. For the estimation, a monthly average storage of 150GB has been considered, alongside a total of 1.500 hours of use of a "*m4.2xlarge*" server.
The material expenses account for the office material, it equipment and devices required for demos.

Marketing

As for marketing, a strong online campaign is issued to acquire users (potential restaurant costumer), so a strong user avail can be presented to the restaurant clients. Moreover, some budget is booked for discount campaigns.

Administration

Expenses for administration include accounting, invoicing, fiscal declarations and HHRR administration among other. This tasks can be externalized on the first stages of the company.

2 Analysis

2.1 State of Art and Market Research

The project "Business and development plan for a SaaS based Restaurant Menu Management System" (BPRMMS), aims to the design and plan the development of a Menu Management System for the Hospitality industry based on a Software as a Service business model and to analyse its business value in order to create a company that provides this services.

In order to fully define the path for this project and understand what is needed to achieve its goal, an environment analysis needs to be done. This analysis includes learning more about what is a Restaurant Management System (RMS), which are its core functionalities and the needs that these systems aims to cover, which is the actual market and competence and the strategies that are being followed, hardware requirements, between others. This set of information not only offers visibility of the ecosystem of these systems, but will also provide key data in order to design a product that has enough potential in ab hypothetic market.

2.1.1 Restaurant Management System (RMS)

A Restaurant Management System (RMS) is a set of tools embedded in a platform that allows Restaurant and other Hospitality business to centralize all their management needs in a single place and enables them to offer better services to their customers and facilitate the work of their manpower. As other similar systems like a Human Resource System, a RMS allows different tasks like managing the menus, orders and serves, performing a full control of inventory, facilitate the accounting tasks, allocating employees in a schedule or providing all sort of reports.

RMS tends to be really complex platforms that offer a wide range of functionalities that usually aims to mid-large restaurants and food-chains. There are plenty of RMS systems in the market and for general; all offers at least these functionalities [1]:

- Accounting Management
- Customer Database

- **Food Costing**
- Inventory Management
- **Kitchen Management**
- **Table Management**
- Wait staff Management
- POS (Point Of Sale)

From all the features listed above, the Point of Sale is the most complex and is usually sold as an independent solution that can have some other RMS features added. In order to create a useful Menu Management System for restaurants, MenuServe will need to incorporate a POS style system, so transactions, billing and other essential actions can be performed. More detail about POS will be provided in the following sections.

All the functionalities of a RMS may have more or less deepness in levels of customization or information, but always tends to be present. All this functions are essential in order to allocate all management needs of a big Restaurant. This, though, extends the complexity of the product and usually excludes the small restaurant, which is why the basic features that would provide a fully competent product for small business are highlighted in bold.

.

2.1.2 POS (Point of Sale)

In a business/retail point of view, the POS is the place and time where a transaction is completed. Is, then, the point at which the customer pays for the services or goods that have been provided to him and the business prepares an invoice for what the customer has to pay for, and collects this cash in some way (cash, credit card, ticket, etc.). Once the payment is performed, the business will provide the customer with a receipt of this payment. [2]

Restaurants are every day more frequently using software POS to perform the action described before. This POS can be seen in a wide variety of establishments, not only Restaurants and Hospitality business but in a wide range of retail shops. Usually, software based POS integrates different hardware depending on the business type, so for example, a grocery shop may have a barcode scanner, and a restaurant may have an only a touchscreen that allows the wait staff to select the orders of the customers. In general, all POS have done manage a cash drawer in order to safely protect the payments in cash.

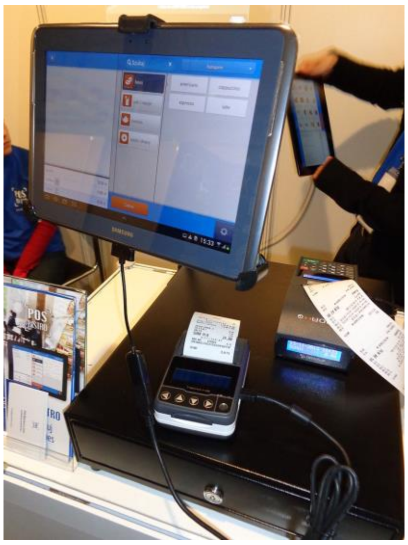


Figure 1 - Example of a POS (Wikipedia)

In the last few years, cloud based POS have extended in the market, offering great value in a very innovative way. With the new advances in internet communications and the appearance of the new mobile devices like smartphones and tablets, the cloud-based software is living an exponential growth. With that in mind, cloud based POS are operative system and platform independent and can run in almost any modern device, reducing the price of its deployment to a level never before seen [3]. With the new growth of developers and information technologies, attaching compatible hardware to this cloud based POS is not a difficult task, and the business owners can have a wide range of devices to acquire. Also, Cloud-Based systems offer a lot of advantages, being the easiness to install, the possibility of centralise data, the chance to use almost any new hardware, being able to access from almost anywhere or the lower entry-costs the main ones.

2.1.3 RMS vs POS

Powerful Restaurant Menu Systems usually requires a big initial investment on terms of hardware, training, installation and communication. For big restaurant chains, this investment is very rentable as they tend to work with a huge amount of information, stock control, employees, hiring processes and so on. But for Small to Middle Business (SMBs), big RMS do not tend to be used as they tend to make thinks more complex than needed. That is why SMBs are more often attracted by the use of POS oriented to hospitality business that usually brings some RMS functionalities.

From the point of view of a small restaurant or bar, a RMS system does not make any sense, but he still has the managing needs that this RMS offers. That is why the POS oriented to this kind of business are having a huge impact and growth in the latest years. For example, the all the POS that we have reviewed before, offers a set of functionalities as inventory tracking, expenses reports, and other features that all hospitality owners need to do. All of this in a price much more reduced and with a much lower entrance barrier than RMS systems. A talk with some restaurant owners can make these points more visible [4]. The tasks that have more error risk or are more time consuming are:

- Table and order tracking at payment.
- Coordination from the kitchen and wait staff.
- Employee supervision.
- Closure reporting.
- Keep track of all transactions, food and ticketing.
- Human errors.
- Obtain visibility of the accounting and costumers.

Both a RMS and a POS can help with all this issues, but POS is more adequate in the majority of cases for the easiness of use, installation and price.

Spain has approximately 300,000 restaurants, bars and coffee shops and almost 240.000 of them are from independent owners (data from 2009) [5]. That means that almost the 80% of the restaurants of Spain are form independent owners and consequently, a POS platform will be what they would need and a full RMS would be quite less demanded. Other countries have slightly more independent-chain restaurant ratio, but at least in Europe, there are always more independent places than chain foods (Spain 80-20%, Italy 72-28%, Germany 68-32%, France 65-35% [6]).

Aiming now to the development of *MenuServe* and the project BPRMMS, where to aim the functionalities is clear. At least from a national and European point of view, the market share is dominated by small to mid-business and for that, POS for Hospitality restaurants makes more sense. More dedicated RMS features can be added to the product in other development stages or with a higher license costs.

2.1.3.1 ***Hardware needs***

POS and RMS need specific hardware to provide all their functionalities. For example, an employee charging for people on a point of sales would need to print a receipt and an invoice if the customer requires it. It may also need to access a secure cash drawer or to scan some item’s barcode.

With a software POS, hardware is needed and in more depth than a traditional point of sale. That means that, at least, a computer or similar device must be set up so the employee can interact with the system.

Alongside all the analysed systems and the ones used in local restaurants, the minimum hardware needed is:

- Computer, tablet or similar device.
- A printer, connected or not to the POS.

In order to have a full working POS system, the hardware need will increase to:
- A printer connected to the main system.
- A cash drawer in order to securely access to the cash.

As seen in the systems available that have been described in the previous sections, the hardware compatible tends to be flexible for the POS main system, allowing the use of tablets or computers, but do not tend to be so on for the additional hardware. Lots of POS software require using specific tablets like iPads, or only accepts the hardware that they sell alongside the system. A lot of restaurants using these systems have had to buy the specific hardware that the vendor specifies of the software required. This is an aspect that can be very differentiator for MenuServe if advertised correctly, as should be able to be used in almost all the hardware with a web browser and should pair with all the regular brands for hospitality hardware.

2.1.4 Menu Systems

One can have an idea how POS are used in today's restaurants just by going to pay in some establishments. They tend to be a computer with a touch monitor, and a car drawer and printer attached to them, and that's it. When tracking the orders for payments, the employees must translate all the notes from a table to the system by performing a manual entry, making them perform twice the same job. This action not only slows down the process but opens windows for poor data entry or human errors, limiting the extended functionalities of POS. A lot of times, all wait staff uses the same computer or charges bills from a same table into different ones, limiting the control of employees and the information of how client acts. It also leaves full open doors to employee manipulation of the configuration, orders and cashing both coincidently or by mistake.
Orders are still made with pencil and notebook, that means that the transaction from table to kitchen is also difficult to coordinate, and a lot of times some orders are missed and not charged according.

MenuServe wants to offer a way of reducing these costs by not only offering a POS, but offering a Menu system as well, where the waiters will only need to use a device for everything, and once on the POS, transfer the data easily and without mistakes. This solves every problem described above, and not only that, offers a better way to attend customers by being able to take notes and send them to kitchen without moving.

This kind of services are already existing embedded in full RMS, but are more difficult to find in small and affordable POS. MenuServe tries to offer that in a more accessible way and for all restaurants.

2.2 Competition analysis

2.2.1 Restaurant Management System (RMS)

RMS are used in a wide number of restaurants, especially in big chains where organization and centralization of the information is a key to success. Some of the available RMS on the market are:

CLOVER [7]**:** Clover is a full RMS that offers all the functionalities mentioned before. It offers user management, tax declaration, refunds, closing reports, discounts and a set of "apps" that can be installed to the main platform to extend functionalities and an API for developers, so it can be adapted to the business requirements if needed. The version MINI is available as a standalone POS.
Price: Under quotation.

R.M.S – JoLoMo [8]**:** This RMS is based for Android and offers a small but very detailed set of features like food costing, customer database, Kitchen and inventory management and POS. It offers the chance to cast the serve on a Chromecast that can be placed on the kitchen of the restaurant.
Price: 75$/month.

MIRUS [9]: Announced as Restaurant Business Intelligence software, Mirus offers not only functions of RMS but a full set of data analysis and reports of costs, ROI, stakeholders, forecasts and marketing intelligence. It is intended to large food companies and its features include multi-restaurant management, HR and loss prevention. It also has support for tablets and smartphones.

Other interesting RMS are True Restaurant Management, Tally or Aldeo.
.

2.2.2 POS (Point of Sale)

As mentioned before, there are a lot of POS systems for Restaurants and Hospitality business. Some of the most interesting or used for the small and medium business [10] are listed below:

REVEL [11]**:** Cloud based POS that offers POS for fast-food, restaurants and retail businesses. It is based in solutions for iPads, and integrates with variety of hardware. It is announced as the fastest "install and go" POS and offers a big list of extra features like gamification, social media link, self-service kiosk, mobile order takes, employee management between others. Revel also provides compatible hardware for the system. The main constraint is that works only for iPad, limiting the range of hardware that can be used.
Price: Not announced.

BREADCRUMB [12]**:** Cloud based POS oriented to Hospitality services. It is similar to Revel, and offers functionalities like Offline mode, Menu customization and provides with full reports. It has a very well designed user interface, but it also has the constraint of working only for iPad, limiting the range of hardware that can be used.
Price: Not announced.

CLOVER STATION [7]**:** Clover's POS that can act as an independent solution or as a part of its RMS. It is provided with specific hardware that limits the minimum price, but offers a solution that works straight out of the box.
Price: $645-$1.500 (with hardware).

EHOPPER [13]**:**Free cloud POS that provides a lot of features for free. They offer the possibility to acquire a kit of all the hardware required (Samsung Tablet, Stand, Drawer and printer) and a payment carrousel.
Price: Free ($699 with hardware).

LOYVERSE [14]: Another cloud based POS totally free. It allows creating products, categories, users with different privileges, managing a customer base, getting sales reports, and it comes with a companion app for smartphones that can be used as a customer app. This app allows customers to give feedback or participate in a customer loyalty campaign that the business owner set up, for example. The main constraint is that the POS can only be accessed from an Android device (although the

customer app is iOS and Android, and the back office can be accessed via web) and it is not oriented to Restaurants, and aims to retail stores. LOYVERSE has a technological approach really similar to the one that MenuServe is looking.
Price: Free.

Other interesting POS software are Agora, Toast or VendHQ and LightSpeedHQ.

3 Business Model

3.1 Software as a Service

MenuServe has been imagined to be a cloud based platform. This translates into that the platform itself will hosted by the owner's company and will be accessible to all its customers through internet, providing a very versatile system with a very low cost per user. Adding a license management to this application distribution strategy, MenuServe fits very well in what is known as Software as a Service or SaaS.

This added benefits of SaaS, that are listed in the next section, acts as well as excellent selling points.

3.1.1 Benefits of SaaS in MenuServe

Providing a software as a service through internet has a number of benefits both for the company that offers the solution and the customer that will use this solution. This benefits are often not only related to the retribution and acts as differentiators and added value of the company to the potential customers.

The following sections offers an overview of these benefits that MenuServe will take advantage of:

3.1.1.1 *Lower cost of entry*

With SaaS, the cost of entry for customers is very low. Users of the system may register themselves into the system and get access to a platform, out of the box, without any added cost. There is, although, a need of hardware, but in any case, this hardware is not ad-hoc, customers have the ability to use a wide range of devices that they may have or may acquire at very low prices when compared to ad-hoc hardware sold by the company.

Additionally, the amount of time that customers need to invest into having a RMS available may be reduced from weeks to hours, without the need to hiring anyone.

From the vendor point of view, the price for a new customer is also very low, minimizing the investment needed to do in providing a system, deploying it, or maintaining it, as all customers will be accessing to the same environment.

Lower costs of maintenance and lower costs for the customers to get to use the systems is an indeed a very good differentiator and provides benefits from the very beginning.

3.1.1.2 *Pricing adapted to the customer needs*

Software as a Service provides very robust pricing models, allowing the customers to have very accurate predictions of the costs that the system may generate. Even if a customer has intentions to scale, it always has the visibility of which will be the expenses and will be able to budget according to it, without the risk factor that an IT project usually carries.

Furthermore, providing a set of licenses will proportionate to the customer the ability to adapt to what it can afford. Probably, MenuServe will help him to improve the business and allow him to afford more.

3.1.1.3 *Security and reliability*

In SaaS models, the provider is the part in charge of maintaining the platform up and running. This means that the customer will not have to worry for if the system is out of date or not compatible with the new version of android.

The provider is also in charge of security and back up, removing this responsibility from the customer without any added costs. There is no need from the restaurants to pay for an external backup system, for example, as MenuServe will take care of this.

3.1.1.4 *High adoption rates*

SaaS applications are usually accessed through internet, via web browsers that users are already familiar with. This makes them feeling more comfortable with the software and thus, having a lower learning curves as there is no need to prepare an environment, install a software or learn all the new and unfamiliar navigation.

This easiness to use, added to the low cost entry of the systems, resolves into a higher adoption rate by the side of users. This means more people starting to use the product and sticking to it.

3.1.1.5 *Integration with third party*

SaaS software, thanks to depend on a stable internet connectivity, and using standard web and communication APIS, can be easily integrated with third party software very easily and almost automatically for the end user.

For example, MenuServe will be integrated with Social Media through Facebook, foursquare, Yelp or the Fork between others via the APIs that those companies provide. This will help the restaurant owners to control the reviews and to add the information of this sources to their reports, obtaining even better stats and feedback from the data they collect.

Furthermore, this integration will also affect the users of the restaurant, as OpenMenu will also be integrated with social networks, generating another source of content from the restaurant to the social networks, all by the customers of the restaurant.

3.1.1.6 *Easy to experiment and improve*

A new feature in SaaS means a new feature to everyone, or to everyone intended for it. This situation also allows companies to test and prototype new features already on customers, having a platform to test concepts and experiments already with the end customers.

This will reduce the costs of testing new features and will help MenuServe to validate new concepts before investing too much money and effort to them while always creating added value.

3.1.1.7 ***Works anywhere***

Last but not least, since MenuServe is hosted in the cloud and accessible over the internet, users can access it via mobile devices wherever they are connected. This means, the ability to use any hardware that more fits the restaurant interests.

For example, even if a Restaurant invested a lot of money in a RMS some time ago, that has become obsolete, they may reuse part of the hardware they acquired to use MenuServe, without any added cost.

3.2 Monetization and income sources

A very important part of any project is the monetization plan, as it will be the base component of the ROI of the company and the base to generate an income, the main goal of any business and/or company. It is obvious, then, that a company needs to generate more income that the costs it has to maintain its life.

The business model of MenuServe combines different factors as licenses, consulting, data analysis and hardware distribution to provide a susceptible basis of income to maintain a profitable growth of the company itself.

3.2.1 Free to use

MenuServe is planned to be a free to use platform, where every user will be able to register and start using the system as soon as they have completed all the registration steps. The decision of making a software like MenuServe free to use is one of the biggest business oriented decisions of all, as it impacts on the whole cycle of life of the product, the marketing, product penetration and income generation.

The main reason to make a platform free to use in its simpler license is to be able to have a big market penetration, ensuring a big amount of customers that can provide a great value both in business expertise and data gathering. MenuServe will offer its main functionalities for free, providing a high value platform that the users will find useful and practical. With a good customer satisfaction, the reach on the market can become viral, gathering a big customer portfolio.

In order to have success with this model, MenuServe will offer a proper system for free, with limitations in extra features but not in the core of the application, so a customer will be able to run the restaurant without the need of paying. Although, but paying a very affordable fee a month, the restaurant can enjoy a great number of benefits both in functionalities and business improvement. A free mode will, then, allow restaurants to easily test the system and validate it before they decide if it is worth it.

All the modules will be accessible for free, although there will be a limitation of some features and in number of users or multiple devices running at once.

The list of limitations of the free to use version will be:

- No advance reporting
- No access to market data
- No business growth reports
- No business trends reports
- No integration with social network data in the reports (but yes on the OpenMenu)
- No tracking of employee performance
- Not tracking of Kitchen performance
- Limitation to 50 menu items (enough to almost all small business).
- Limitation to 20 options (enough to almost all small business).
- No role configuration (only pre-sets)
- Limited number of waiters connected at the same time (2 maximum)
- Limited number of POS connected at the same time (1 maximum)
- Limited customization of the Look and Field (only main logo and pre-set colours)
- No integration with other payment methods like PayPal.
- Standard support (3 days SLA).
- No customization

All this features will enable extra functionalities, more customization and adaptability of the system and, the most important, all the KPIs and business indicators that may lead to a growth of business. The main goal is to let the restaurants use the system, enjoy it, understand how powerful it is, eventually, let them realize how much more they can achieve by paying a small amount every month.

3.2.2 Licensing

MenuServe will have different license types, depending of the amount of features will a restaurant require. This will allow every customer to pick the one that mostly fits their needs.

BASIC	**PLUS**	**PRO**
✓ No hardware restrictions ✓ Basic reporting ✓ Access to all modules ✓ Max. 2 simultaneous waiters ✓ Max. 1 simultaneous PoS ✓ Maximum 50 menu items ✓ Basic support ✓ Basic configuration ✓ Limited appearance customization	All BASIC features plus: ✓ Advance reporting ✓ Employee tracking ✓ Advanced configuration ✓ Up to 100 item menus ✓ Max. 5 simultaneous waiters ✓ Max. 1 simultaneous PoS ✓ Advanced support ✓ Full appearance customization	All PLUS features plus: ✓ Advance market analysis ✓ Business growth suggestions ✓ Social Media monitoring ✓ Mobile payment methods (PayPal, NFC, etc.) ✓ Max. 8 simultaneous waiters ✓ Max. 2 simultaneous PoS ✓ Pro support ✓ Personalized support ✓ Set up consultant
FREE	120€ monthly by restaurant (save up to 8% paying yearly)	250€ monthly by restaurant +20€ extra user / +40€ extra POS
CUSTOM LICENSE		
If you have a bigger business but still want to take advantage of MenuServe expertise and advanced statistics, you can contact us for a personalized quote. Our Sales team will be happy to assist you in any doubt regarding personalized development, implementation and licenses for special cases.		

Table 3 - MenuServe proposed licenses

This table shows a first approach of the licensing model stablished for MenuServe. Prior to a commercialization of the product, there would be the need of performing an exhaustive analysis of the pricing for accurate estimations.

3.2.3 Upselling

Upselling is a sales technique where the customers are presented with more expensive items when registering or effecting a payment in order to make the sale more profitable. This point involves marketing in order to offer better value offer to the customers.

MenuServe will use this technique in order to explain why bigger licenses are better and what they can achieve, while offering a discount by purchasing a license yearly. In this point, is important to let the customer know that with MenuServe, this first investment on a platform will make his business more efficient thus, making them save money at the end of every month even when paying an extra fee for the license.

In this point, for example, all new registers will be given a free week of personalized support, so they can solve their doubts and understand more the product. This will increase the interest of the customer and its satisfaction, increasing the sale opportunity.

3.2.4 Consulting and reporting

MenuServe will take advantage of all the data that generates to have an extensive intelligence analysis of the market of hospitality. Thanks to that, it will be possible to offer consulting services like business growth, restaurant chains, restaurant image, etcetera.

Thanks to the experience acquired with the consulting and the support to clients, and added to the value of all the big data generated, MenuServe will be able to create and sell reports and informs to third party companies working in sectors of marketing, branding, consulting among others.

Consulting and reporting will create a good income base by taking in advantage information and resources already existing in the company.

3.2.5 Customized platforms

MenuServe can make use of its scalable platform and consulting expertise in order to create personalized solutions for big companies. In the early stages of the company, this is a risky business, that is why this situation should be avoided until MenuServe is assented.

Once MenuServe is more mature, although, it could provide big restaurant chains with an affordable internal Restaurant Management System customized to the client requirements. This projects are costly, so that would create the need to use providers in many occasions, but the business incomes are certainly big and can help MenuServe petition even better on the market.

3.2.6 Third party hardware

Another important point of income for MenuServe will be the possibility to resell hardware of other providers. With this in mind, MenuServe can have agreements with hardware providers to act as selling partners, providing the customers of MenuServe with cheap hardware and earning a margin while doing so.

The sales team will be in charge of stablishing contacts, and the products will be tested to validate that they are 100% functional with the platform. If so, this hardware can be advertised on the website for the customers that needs to acquire both the software platform and the hardware to run it.

This is a very profitable income model as the revenue generated by this has small risk and almost no cost associated to it.

3.3 Revenue estimation

The last step of the Business Model is a revenue estimation based on a hypothetical case, that will try to reflect the reality to have this first picture of the business value.

In order to perform the first predictions of income for this licenses, we will use the Pareto principle [15], that states that, for many events, roughly 80% of the effects come from 20% of the causes. This means, that we will account as payer users the 20% of the total. And from this payer users, the 20% will be PRO licenses.

The following numbers are an estimate over 1.000 users globally. Being a SaaS based platform that may operate worldwide and given that only Spain has around 300.000 restaurants, a group of 1.000 users seems as an achievable number of users after a proper marketing placing and sales effort.

Concept	Fee	Users	Amount (monthly)	Amount (yearly)
BASIC	- €	800	- €	- €
PLUS	100.00 €	160	16,000.00 €	192,000.00 €
PRO	250.00 €	40	10,000.00 €	120,000.00 €
Total		1000	26,000.00 €	312,000.00 €

Table 4 - Yearly license income estimation

The table 3 shows the estimated income by licenses with a base of 1000 users both by month and by year. Even though this is a very early estimation, the numbers are favourable as with this amount, the implementation investment is covered.

Adding to this numbers, there should be an estimation of the other revenue sources. To have an approximate idea of the income that could be generated from these other sources, we have taken into account that a small restaurant would need around 1.500€ in hardware and the 30% of those would be margin, and that the revenue of reports and consulting would be around the 5% of the amount in licenses and hardware. This last assumption is very in accurate but will help us have a first approximation in numbers to a real case.

Concept	Users	Price	Amount (yearly)
Licenses	200	various	312,000.00 €
hardware	50	450.00 € (margin)	22,500.00 €
		Subtotal	334,500.00 €
Consulting	-	5% over total	16,725.00 €
Reporting	-	5% over total	16,725.00 €
		Total	367,950.00 €

Table 5 - Income estimation

The table 4 show us an approximate value of yearly income under a situation with 1000 users acquired. This income goes up to 409.000€ in cash, that would lead us to have big margins of benefit once the costs of the project are deducted.

It has to be said that this estimation is pretty rough, as there will be probably more income coming from Consulting and Reporting, but getting a number of 1.000 licenses may require a big marketing expenses. Even though, if the numbers look as favourable as this ones, a big investment on marketing might be very feasible.

There is clearly a possible business on this sector, as the product is accessible, there is a big market of restaurants and establishments all over the world, and thanks to a SaaS model are accessible. But this numbers cannot be taken as finals, as this would lead to a speculation that will not necessarily be true.

If or that, that in the list of further work to do after this thesis, will be the performing of an extensive business analysis and revenue assessment.

4 Functional Design

The functional design for MenuServe is intended to specify the requirements and the how the system will work, without going into detail of technical, software or hardware specifications. The functional will, then, describe the logical system flow, data organization, system inputs and outputs, processing rules, and operational characteristics of the product from the user's point of view.

This functional design is divided between two sections:

- Client side, where it takes in accountancy the specifications of the different applications (also called modules) that creates the MenuServe suite.

- Cloud side, that defines how the context of the applications or modules will fit on a cloud system and how the content will be provided to the users.

4.1 Client side

4.1.1 Requirements

As a part of the State of Art and Market research, a series of requirements or "musts" have been detected in what would have to be every Restaurants/ Menu Ordering System. This following section is intended to list these full requirements to provide a first glance of what MenuServe should be able to perform. This information will be, in following steps, evaluated with different potential customers (users) to validate its value.

4.1.1.1 *Restaurant Environment Diagram*

In order to understand the needs of the restaurant, a picture of their environment need to be procured. With a picture of how the restaurant is structured, it becomes easier to detect and list the requirements of a successful Restaurant Menu System.

The graphic shown on the Figure 1 aims to provide an overview of how a restaurant is structured in a single glance. It is appreciable that the basic structure of the restaurant is divided in 4 sections that require inputs and outputs from the other sections in order to function as a whole. The sections described in the diagram are:

- **Kitchen**: in order of receiving the petitions, preparing them and delivering it back to the wait staff.
- **Dining Area**: where the wait staff is in charge, providing attendance to the customers and taking the needed notes.
- **Point of Sale**: where the information is structured in order to charge the customer for the services provided.
- **Back Office**: or administration part of the restaurant in all its aspects.

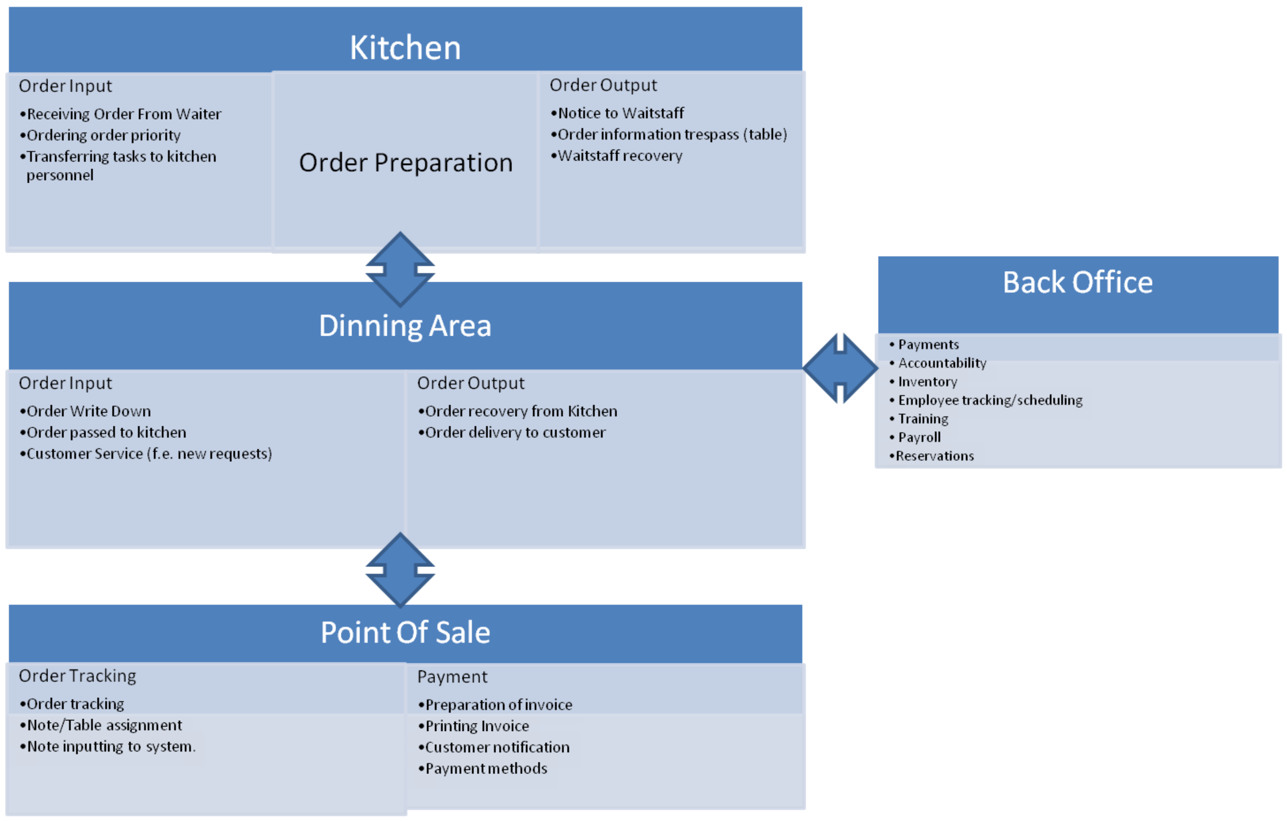


Figure 2 - Restaurant Structure Diagram

4.1.1.2 *Requirement Table*

The Figure 1 shows the basic structure and how the information fluctuates from the different parts of a Restaurant. By following the provided structure and the product description of MenuServe, the set of requirements that will need to be accomplished can be listed.

The following list of requirements sowed on Table 1 is directly related to the different sections of a restaurant and how they will interact with it. The requirements, then, will be realized in the sections visible on the Logic Diagram for MenuServe in the Figure 2.

Table 6 - Functional Requirements

R.	Requirement
1	Menus created with the system must be visible from any device regarding screen size.
2	Customers may be able to access the menu without the need of installing the application (via web browser)
3	Mobile Application must be independent of the platform (same app for Windows/iOS/Android)

4	Mobile Application and Web Version must support Multilanguage.
5	Every menu item must be able to have a multimedia support, description and price.
6	Every menu item must be able to have additional options (i.e. side dish, cooking level, comments).
7	Quantity of a menu item must be editable, in order to perform multiple orderings.
8	Customer must receive feedback for customer approval of order.
9	Customer must have a button to request attendance by a waiter.
10	Customer must be able to edit or cancel orders before submitting the order to the queue.
11	Customer must be able to see all its ordered items.
12	Mobile Application must accept an always logged in mode (Kiosk)
13	Customer identification can be provided both by temporary passwords or QR code.
14	Wait staff must accept user log in for verifiability of the customer.
15	Waiter is assigned to the table that attends. Waiter cannot directly see listed tables/bills that are not attending. Waiters can look for them to add quick edits.
16	Wait staff must be able to add, edit or delete menus and orders in their assigned tables.
17	Wait staff must be able to relocate menu/orders in case customer changes table.
18	PoS must provide list of all table and invoices.
19	PoS must have the option to consult and edit each table ordering.
20	PoS must be able to select payment method, assign charged import, and close bill.
21	PoS must be able to print invoices.
22	Wait staff device must have the same options as PoS for assigned tables (except when HW required)
23	Kitchen Device can be unattended or interactive depending on customer hardware limitations.
24	In Kitchen device, Items may be shown listed by categories, (i.e. Drinks, cold dishes, hot dishes, sauces) and may be ordered (i.e. By Cooking Time, By Table).
25	In Kitchen Device, items should be able to mark as ready and expired (when interactive device)
26	In Kitchen Device, items must show its name, table that ordered, wait staff name, time since ordered.
27	Administrator must be able to create, edit, read and delete (full control) of menu items.
28	Administrator must be able to create, edit, read and delete (eating) table item.
29	Administrator must be able to create, edit, read and delete users.
30	Administrator must have access to all saved information and reports (i.e. efficiency and performance).
31	Administration must have access to Open Menu configuration (colours, logotypes).
32	System will follow a SaaS model. The system must be multitenant and multiuser, and it has to allow different restaurants and their data structure to coexist.

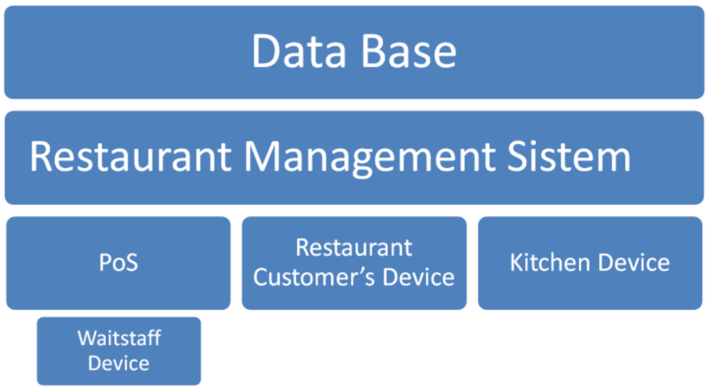


Figure 3 - MenuServe Logic Diagram

On the Figure 2, the scope of each of the modules that takes part on the system are showed in a structural way. The following list will provide extended context for each of the boards in the Logic Diagram provided. It is aligned regarding dependency from its upper item in the list:

- **Data Base** contains all the information stored in the system.
- **Restaurant Management System** is the administration application layer. From this module, all others can be administrated. It is also the module where data is created, updated or deleted in order to set up the system.
    - **PoS** module, data regarding orders and billing is showed here. Payment point.
        - **Wait staff Device** is the module running on wait staff device.
    - **Customer Device** is the module running on the customer device in case OpenMenu is enabled. It can run from an app or website.
    - **Kitchen Device** is the application module that enables interaction with Kitchen and the rest of the restaurant.

4.1.2 System Roles

MenuServe is intended to be multitenant, meaning that multiple restaurants can run the system with their own data, and multi-user, meaning that every restaurant have to be able to administrate a set of users. The number of users that a restaurant can have may be controlled by the provider of MenuServe for monetizing purposes (licensing).

4.1.2.1 ***Super Administrator***

This role will be only held by the provider of the system and will be able to do the same actions as the Restaurant Administrator (described in point 3.2) but for each of the Restaurants on the system. This role will have total access to the system, and will be able to set up limitations to other users (roles).

4.1.2.2 ***Restaurant Administrator***

The admin is the most important role as for the correct function of the system. The administrator is a super user that will be able to control and edit the roles of all other users that have him as a parent (Requirement 29). The administrator responsibility will be to set up the system as desired once the deployment/acquisition is done, and will have the maximum authority level in creating, modifying and deleting the menu items and restaurant table layout (Requirements 27 and 28). It will also be able to set up different modules, as allowing the OpenMenu (R.30), setting how the Kitchen will receive the data (R.23), setting the system's languages (R.4) or accessing to the reports for efficiency and performance.

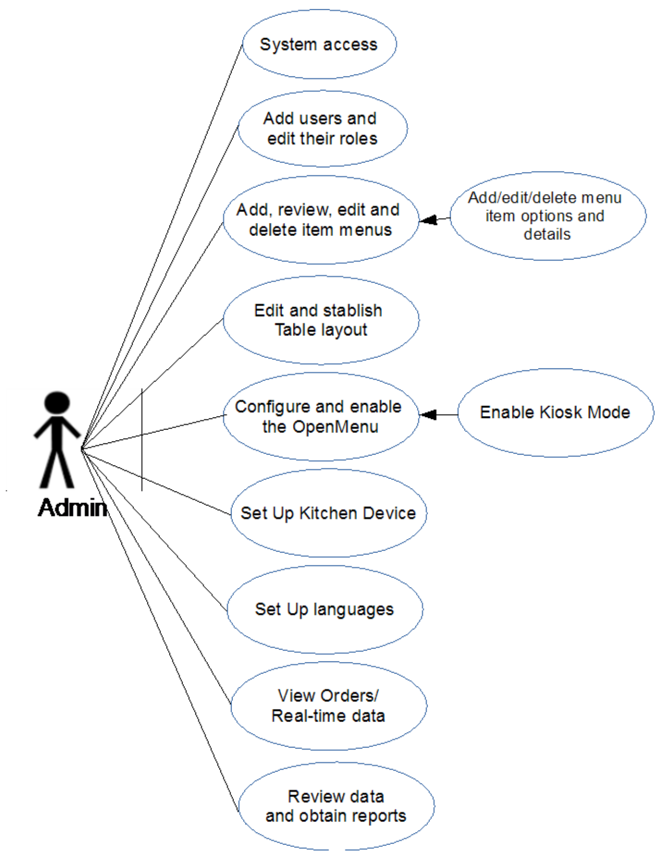

System access
Add users and edit their roles
Add, review, edit and delete item menus
Add/edit/delete menu item options and details
Edit and stablish Table layout
Configure and enable the OpenMenu
Enable Kiosk Mode
Admin
Set Up Kitchen Device
Set Up languages
View Orders/ Real-time data
Review data and obtain reports

Figure 4 - Administrator User Case Diagram

4.1.2.3 *Wait staff*

Another important actor of the system is the wait staff. Every member of the wait staff present on the restaurant will have one device to operate; it can be either a Smartphone or a tablet with the proper app installed (R. 1). Each user may be able to log in to the system with a user/password that could be the same for all (with limited tracking) or one by waiter.

The first action a waiter will do is to assign the new arrived customer to a table/bill that will be then assigned to this waiter (R. 15).

The actions that a member of the wait staff can perform are to introduce orders into the system (R.17), edit or perform changes to the client orders if required (R.16) or to the full customer bill, for example if they wish to relocate to another table (R. 17), verify the login of the users if *OpenMenu* is enabled (R.13 and R.14) or accept customer orders to be sent to the kitchen (R. 8).

The wait staff will be also in charge of using the PoS; both with their device in hand or by approaching the PoS Device, where they will be able to verify the bill (R.19), select the payment method and introduce the result payment information (R. 20) and print the receipt and invoice, in this case on the PoS Device itself.

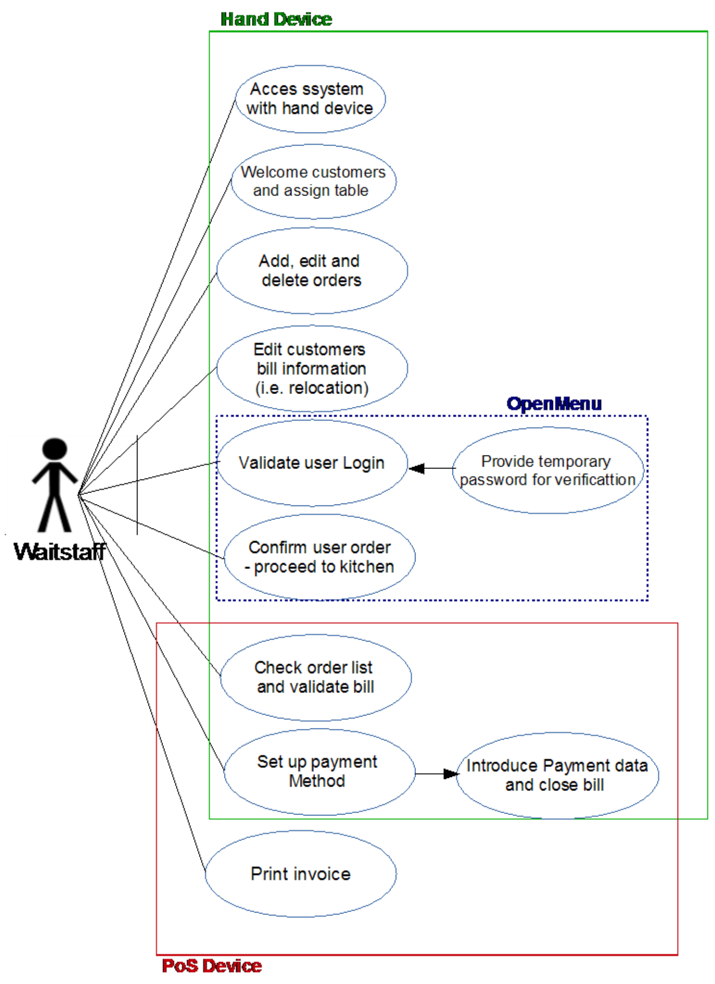

Hand Device
Acces ssystem with hand device
Welcome customers and assign table
Add, edit and delete orders
Edit customers bill information (i.e. relocation)
OpenMenu
Validate user Login
Provide temporary password for verificattion
Confirm user order - proceed to kitchen
Waitstaff
Check order list and validate bill
Set up payment Method
Introduce Payment data and close bill
Print invoice
PoS Device

Figure 5 - Wait staff User Case Diagram

4.1.2.4 ***Kitchen Staff***

Kitchen Device is intended to show and provide the information that the wait staff inputs to the Kitchen Staff so they can prepare and have all items ready. The Kitchen Device can act both unattended or with human interaction, needing only a Kitchen role if the second option is enabled.

When working unattended, the administrator will be in charge of setting up what information (R.26) and how the orders will be shown to the Kitchen Staff members (R.24). The wait staff will be in charge of marking the items as delivered to the table and the status ready will not apply.

By contrary, when the Kitchen Staff can interact with the Kitchen Device, they will be able to change the order by themselves via drop down menus (R.24), and will be able to mark as ready the orders or to mark them expired by themselves (R.25). The wait staff will still be in charge of marking the items as delivered to the table and.

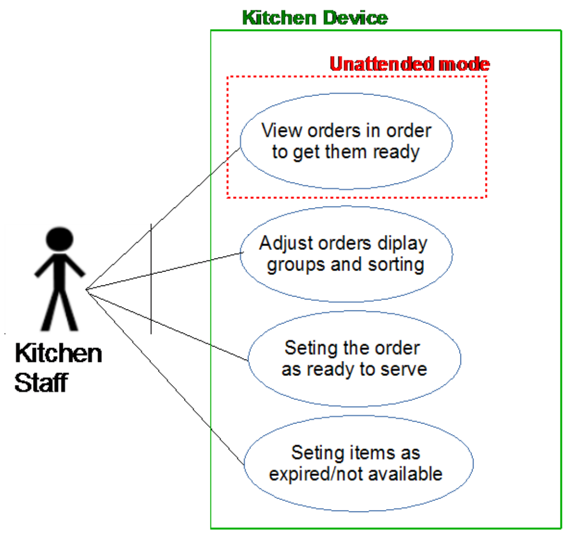

Kitchen Device
Unattended mode
View orders in order to get them ready
Adjust orders diplay groups and sorting
Seting the order as ready to serve
Seting items as expired/not available
Kitchen Staff

Figure 6 - Kitchen Staff User Case Diagram

4.1.2.5 ***Customer***

When MenuServe, a part of acting as a PoS and ordering system for the restaurant staff, can also open in what have been called *OpenMenu*. This is a functionality of the system that allows customers to use their own devices to interact with the restaurant and wait staff in order to place orders and receive useful information. If administrator has decided to activate *OpenMenu* (R.31) and configured the table layout, the customers will be able to access the application. The customers may use their mobile phone to access either a website (R.2) or using MenuServe mobile application to access the system.

Once they accessed, they may log in to their table by using a QR code (may be the same as for the website) or via a temporary password provided by the wait staff and given automatically by the system when checking in a table (both R.13 and R.14 at once). The restaurant could also enable tablets or “kiosks” on each table that customers could later use as an input point. In this case the wait staff would activate the kiosk when assigning the customers to a table.

From their devices, customers will be able to perform actions as choose language (R.4), navigate through all items of the menu and seeing its price and some multimedia support to identify it (R.5), selecting items and its options if available and adding to the order list (R.6 and R.7) or placing this order list once they have decided that everything is ok (R. 10). After the order is placed, customer can see all items ordered and its price individually and in total (R. 11). Customer can also see if these orders have been processed and cancel them if they have not (R.8), and if there is any incidence or request, the customer can always request a waiter through the system (R.9).

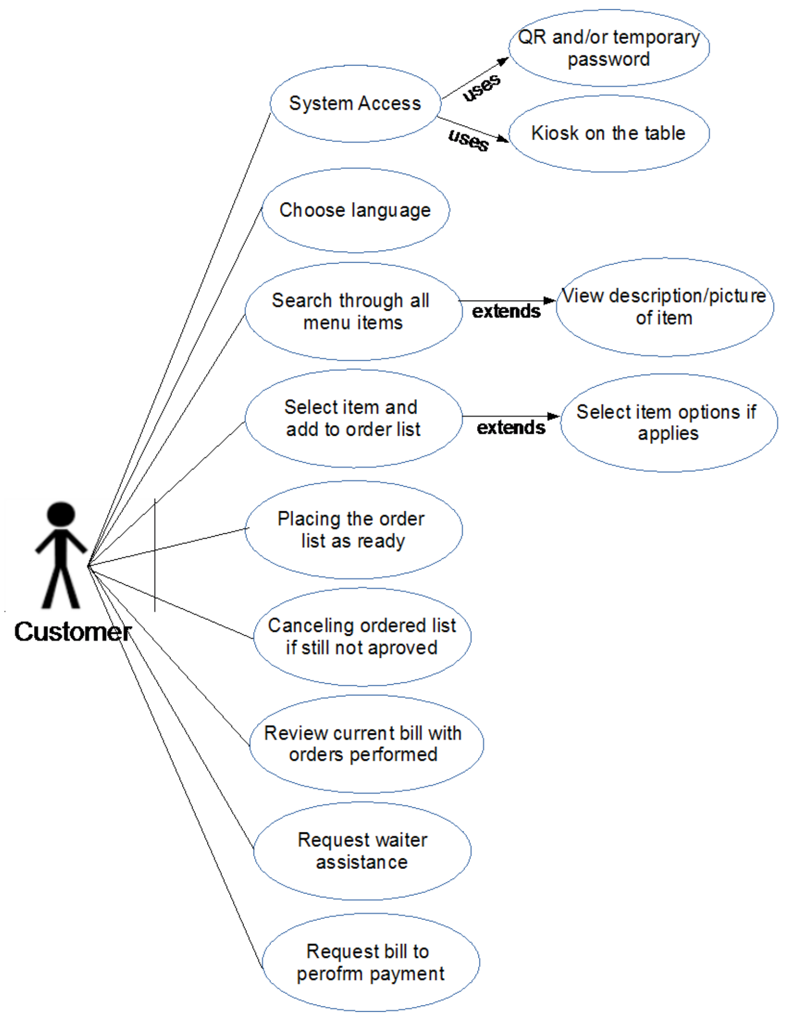

QR and/or temporary password
System Access
uses
uses
Kiosk on the table
Choose language
Search through all menu items
extends
View description/picture of item
Select item and add to order list
extends
Select item options if applies
Customer
Placing the order list as ready
Canceling ordered list if still not aproved
Review current bill with orders performed
Request waiter assistance
Request bill to perofm payment

Figure 7 - Customer User Case Diagram

4.1.3 Non-Functional Requirements

Apart of the functional related requirements, MenuServe will need to fulfil a series of requirements that will be transparent for the final user (restaurants and customers of these restaurants) or not related on how the system has to work.

Many of these requirements will be fully designed and more extended on the following chapters of the thesis. This section may be useful for understanding the needs and concerns of a system with such a scale.

4.1.3.1 *User Interface*

MenuServe is intended to be a system based on a Software as a Service model, meaning that users will be able to use the system without almost any human interaction with the provider. In terms of use of the platform, that means that a big portion of the users will start using the system without any prior training or contact with it. Especially third party users (referring to the customers of the restaurant), that will use MenuServe as a service from the Restaurant itself and not from the restaurant's provider.

We can, then, identify three actors that will interact with the system in a user experience point of view, the restaurant administrator, the restaurant staff and the restaurant customers.

The administrator will have the most access to trainings and documentation related to MenuServe, allowing the user interface of the administration module to be more complex in order to gain functionality. Note that this does not mean that the system should not be easy and intuitive, but may become a bit more complex without the risk of user abandonment.

For the restaurant staff module, the system must be intuitive, fast and easy to use in order to maximise wait staff performance and avoid possible mistakes. Regarding look and feel, it may maintain a standard scheme without the need to personalize.

On the *OpenMenu* module, aimed to the customers, there has to be a special attention to easiness of use, in the same approach of the wait staff module, but combining with an appealing design that may be customizable in terms of look and feel and branding, in order to be as similar as possible to the restaurant's branding.

Through all the system, User Experience is a must that may be guaranteed by proper Quality Assurance during the Testing Phase. This will take special attention in User Interface design both in design quality and user-friendliness. That meaning to provide a consistent design through the entire platform, and use the most standardized elements, actions and feedback to these actions as possible.

An approach to Material Design will be used as a base for the User Interface. The Figure 7 below shows a sample of the design bases form Material Design.

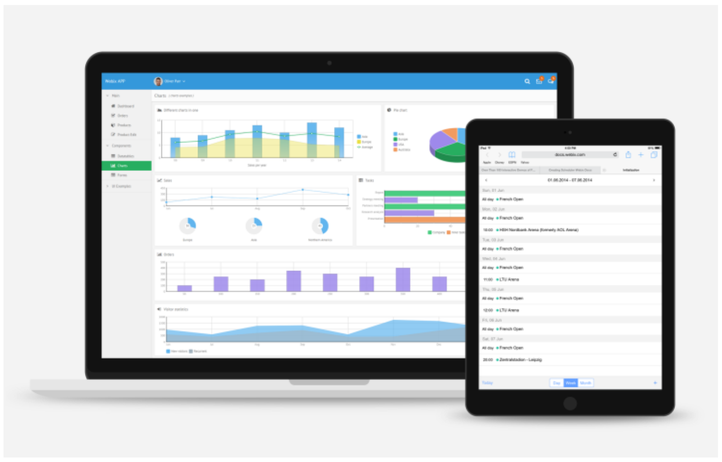


Figure 8 - Material Design Web Application.
Source: Wikipedia

As one of the key points of MenuServe is this capability to run in a wide range of devices both from the restaurant and from the customers, providing a full range of compatible devices is a must. In order to fulfil this important requirement, the design must be fully responsive and adapt to different devices like Smartphone, tablet or desktop PC. During the testing, a set of cross-browser and cross-platform tests must be performed to ensure a proper functionality.

Regarding responsiveness on small screen devices (six or less inches), the application must be able to run in portrait mode as is the most comfortable way to interact with the device.

4.1.3.2 *Security*

Security is a major concern and a risk factor on every cloud based application, as all the sensitive data may be accessible through the trusted users from wherever they are connected. The servers will count with the most advanced security measures as SSL certification, traffic inspection, Firewalls and 24/7 system monitoring, ensured by the MenuServe enterprise and its Hosting provider.

Not all the security risks, but, fall on the server side on the system. As MenuServe will be in charge of managing the Restaurant information, orders, and billing, the impact of an external user using the system could be high. This is why the verifiability of the users and its data must be guaranteed at any time. In order to do so, there are different security measures that have been taken in consideration.

- **User credentials**: Every restaurant user will have a personal and non-transferable username and password that must be used to access the system.
- **User roles**: Different roles will have different permissions onto the system, avoiding that a leak on a user can cause considerable damage.
- **QR Codes**: Can be re-generated periodically and contain a time limited token. Staff may use a token to check in to the systems when their shift starts, and customers may use a QR code to access to its *OpenMenu*. This QR code can be generated automatically when assigning a new table and the code can be read from the waiter device screen.
- **Temporary password**: The same as QR codes, temporary passwords contains a time limited token that customers can use to access the system. Temporary passwords may be generated when assigning a new table, and can be configured to create easy to remember passwords by using a dictionary
- **Wait staff verification**: If a Kiosk (tablet on every table) is available, wait staff must activate it once the customers are assigned. This process of verification is implicit when using auto-generated Tokens (QR Code or Temporary Passwords).

4.1.3.3 *High Availability*

Restaurants using MenuServe will be relying on the system to perform all the actions related to its day a day business. This means that it is necessary to offer a high availability system that will minimize the downtime in case of any issue. In order to grant this maximum availability of the system, there will be actions taken from the server side and the application side.

From the server side, a high availability system must be provided, with server scalability and node balancing so the system can maintain its functionality even when one or more nodes are down. There also have to be a 24/7 monitoring of the system both by technicians and monitoring systems like Greylog.

From the application side, if the restaurant’s connectivity fails, the PoS and the Kitchen devices must be able to maintain the information on the system so the orders may be preceded and the bills may be accounted without the need of the Wait staff hand device. In this case, the PoS may act a classic PoS system, allowing the wait staff to introduce the commands from a paper to the system just before generating the invoice.

4.2 Cloud system functional design

The Cloud System is intended to handle all requests from the different modules of MenuServe and proportionate all the needed server side functionality in order to provide an availability and continuity to the system at any time.

As a cloud based application, MenuServe will be accessed from different sources that will request information. The server part of MenuServe has to be able to proportionate this information and to process it in any way that is necessary, centralising the information on the server side so the interaction between devices, modules and roles it is always controlled, tracked and proportioned in a satisfactory way.

The following lines provide a functional description of the cloud system. This definition is by nature smaller than for the application part due the server side will be only in charge of processing the information and will not provide direct functionality to the application layer. Figure 1 shows how the logical diagram from the server side and data base side.

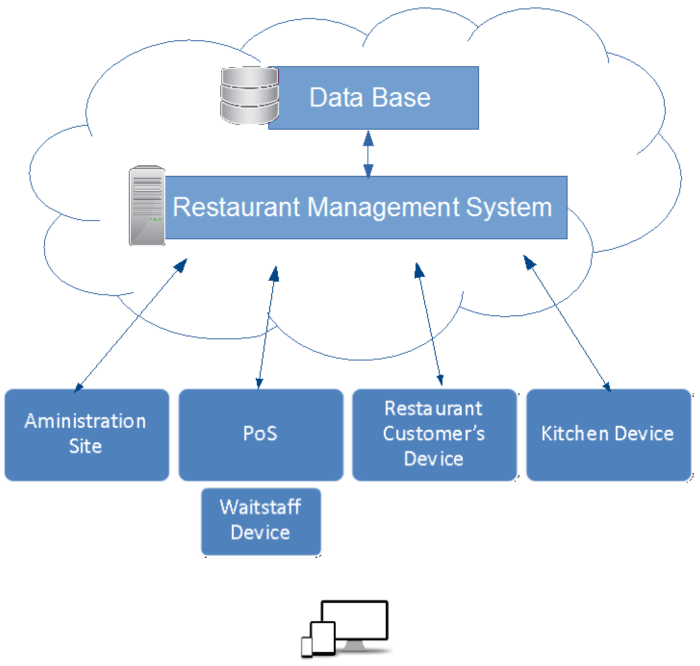


Figure 9 - Cloud Logic Diagram

4.2.1 Cloud system Requirement Table

RC.	Requirement
1	System will follow a SaaS model.
2	The system must be multitenant and multiuser, meaning that different restaurants and their data structure has to coexist.
3	The platform must provide an user management system.
4	The user management system must allow roles and permissions.
5	The user management system must allow date ranges for roles/permissions (licensing).
6	The cloud system must record all actions performed by a user in an application log
7	The cloud system must record all warnings and errors in an error log
8	The communications between modules and platform must be able to work both in “one way” communication and bidirectional communication.
9	The business layer in the platform has to proportionate an API with all methods in order to provide and/or save the data of the modules.
10	The business layer must provide a response for any performed action (i.e. success or error).

Table 7 - Cloud System requirements

4.2.2 Cloud system Description

The Table 1 lists the requirements for the Cloud part of MenuServe. The “Cloud System” references the server side of MenuServe and it will be the platform in charge of providing all the content to the users of MenuServe and to the customers of the restaurant.

As MenuServe is a SaaS based application (RC.1), there is the need to have a multitenant system, allowing different data structures and independent administrators for each of these structures, in order to proportionate “virtual” instances for each restaurant (RC.2). From each restaurant will hang a set of users that will have a variety of roles and permissions over its parent (the restaurant), that is why the system must also be multiuser (RC.2).

To fulfil the Requirement RC.2, the system must provide a way of managing the users. This system, a part of providing the registration and identification of the users (RC.3), must be able to control the roles and permissions of each one, by assigning it to its specified "virtual" environment (restaurant), and showing them the options that it can see (RC. 4). For example, when a user with the role of a Waiter logs in to the system from the web based application, the platform has to show to him the user interface intended for the waiter (with the options to see tables assigned, orders, notifications, etc.) and only for the restaurant he is assigned to (RC. 4).

This user management system must be able to manage licenses and the range of time for each of these permissions. For example, if a Restaurant has acquired a license that allows them to have more detailed reports, the user management system must be in charge to proportionate this option to the users when accessing the administration panel and to limit the access to it once the license has expired (RC.5).

In order to obtain reports and big data that can proportionate benefits both to MenuServe as a company and to its clients as an added value, all actions and processes that have taken place in a restaurant needs to be saved to the system (RC. 6). That means the existence of logs or historic that will keep track of the sales, orders, transactions, items sent to the kitchen, etcetera. With that information, all kinds of reports will be able to be done, both by restaurant and by bigger segments, depending if the report is demanded by a user (restaurant) or is an internal report (MenuServe Inc.).
This will also proportionate a powerful *auditability* tool, allowing third party entities to certificate our system if it is required in a near future. And, in order to save data on our servers, a License and Agreement will be mandatory and users will have to read and accept it when registering.

For internal tracking and customer support, an error log must exist. Every time there is an error when processing data or executing some functionality, the platform will write this error to a log, allowing the detection of the error and its solution (RC.7).

By analysing the functional description of the Application part of MenuServe, there are two possible situations regarding the interaction between the client and the server. The most common communication to the server will be one way, that means that the application will perform a request, and then the server side will respond to this request and proportionate the data, without the need of an open channel of communication. In other words, only one side will be listening, the server.
In some cases, although, there will be the need of an open communication between client and server, where the applications must be listening to anything that may come from the server, to be able to react in real time. For example, the ordering information going between the Wait staff/Customer Module and the Kitchen must work in real time, arriving to the destination device without this one having to "request" if there is something new (RC.8).

The business layer will be the API, on server side, that will provide the set functionalities that are key for the MenuServe application to work, and with what the client side will perform its requests(RC.9). The API will be in charge of:

- Provide the website and web application for the different modules that users will access form their browsers or specific applications.
- Gather the information on the databases and provide it to the pages of the application in order to populate them.
- Receive all data entered by the used and register on the corresponding database or storage system. That can mean adding new entries, updating existing one or deleting them.
- Manage the user permissions and enable only the functionalities that each user can view/use.
- Save a record of each action performed and populate the logs.

4.2.3 Non-functional Requirements

The Server Side centralizes the application functionality, as all the data is sent to de platform, it is processed as needed, stored in the database and returned to the related modules if needed.

This "Cloud System" will not provide the UI for the Mobile Based Client, as this will be preloaded on the Device and not downloaded as an HTML. The data that populates this application will be provided by the "Cloud System".

As described in the Non-Functional Requirements on the PAC 3, the server must provide advance security and high availability in order to maintain the correct functionality at any time without compromising any data. The application will use standard security measures to ensure its security against threads, that includes:

- **Phishing:** An SSL connection will be used to prevent phishing attacks on the cloud system. The SSL certificates allows the server both to validate the authenticity of the information and to decrypt the received requests.
- **Denials of Service**: In order to neutralize such kind of attack, the MenuServe platform will have

a redundant server with firewalls that can be used to block traffic from IP addresses being used to bombard the platform with requests.

- **Virus**: An antivirus system should be used by the restaurants administrator to quarantine his computer against viruses, worms and Trojan horses. This generic malware cannot affect the data of the server as it is encrypted and “isolated” by tenant, but could log the passwords of the systems the administrator uses.
- **Spying software**: Antispyware systems should be used to block and remove this malware from administrator computers.

Some other measures implemented to ensure the security of MenuServe “Cloud System” are:

- **Log trail**: all system components will register the main activities done by the different users, so they can be traced if required.
- **Timeouts**: users logged into the system have a limited time to perform their activities. In case of inactivity, after certain configurable time the session is lost and they are requested to log in again.
- **Firewalls**: these commercial security elements prevent external attacks and allow to detect them to be able to react in time.
- **Secure architecture**: the system is deployed in a secure architecture which limits the access to the different services/servers only to the appropriate elements in the data centre (by using VLANs, DMZs…). In example, the database can only be accessed by the platform that can only be accessed through the firewall; it is not possible to access the database directly from outside.

5 Technology Design

5.1 Information Architecture

The platform of MenuServe is, in its roots, a Content Management System (CMS) with complex and very focalized modules to perform very specific operations related to a business field. This simplification of the concept of MenuServe is intended in order to proportionate a simplest view of how the Information Architecture of MenuServe might be.

The main goal of this section is to describe the Database information architecture. In order to do so, the easiest way is to proportionate a view of the information through the entities and the properties of this entities that will be used to provide the functionality to the platform.

5.1.1 Entities and properties

A description of entities and their properties may be find in the following lines. These entities will have standard database properties as unique identifiers (ID), timestamp and other information that will not be reflected in this section. If the complete information regarding the database want to be acquired, please refer to the section 3.2 Entity Relational Diagram

The main entities for MenuServe are:

5.1.1.1 *Company Master*

The company master contains 1 to N restaurants, and it allows the system to have multiple restaurants by company, allowing multiple locals sharing data.

- Company ID
- Active
- Name
- Description

- CIF
- Main Currency
- Telephone
- Address
- Email
- Contact ID

5.1.1.2 ***Licenses***

Each restaurant/company may have a different use license that must be tracked and know by the system:

- License ID
- License type
- Created on
- Expiration Date
- Price

5.1.1.3 ***Restaurant***

Restaurant and its local information. This entity will be unique by restaurant, and will act as the Restaurant profile. The main properties will be:

- Restaurant ID
- Company ID
- Contact ID
- Configuration ID
- License ID
- Active
- Name
- Description
- Currency
- Restaurant telephone
- Restaurant address
- Restaurant email
- Website
- Facebook
- Open menu flag
- Kiosk Mode flag
- Temp password type
- Auto-accept time

5.1.1.4 ***Restaurant Settings***

Settings of the restaurant:

- Configuration ID
- Number of tables
- Maximum seating
- Dinner table layout

- Logotype
- Colour scheme
- Devices layout

5.1.1.5 *Users - Customer*

Information of a particular user and/or customers. The table can share information both from users and persona.

- Name and Surname
- Job position
- Username
- Password
- Email
- Role
- Picture

5.1.1.6 *Roles*

Roles by user

- Name of the role
- Access flag (BO, POS, Kitchen)
- Flag “view all tables”
- Flag “edit bill details”
- Picture

5.1.1.7 *Dinner Tables*

The Restaurant and billing system will be organized by tables. Each table needs to be defined in order to proportionate an environment to work for the wait staff device. The key properties are:

- Restaurant
- Number of the table
- Zone/Floor of the table
- URL (Open Menu)
- Occupied status
- Assigned wait staff
- Seated people (users)
- Assigned bill

5.1.1.8 *Items*

Each of the dishes and items of the menu of the restaurant must be entered on the system before they are eligible on the corresponding menus. The main properties of this entity are:

- Name
- Description
- Availability
- Characteristics
- Type (Dish/Drink/Dessert)
- Image
- Price
- Allergens
- Colour on POS
- List of options

5.1.1.9 *Item Options*

Menu items might have options as side dish, ingredients, etcetera. As one item might have multiple option items, it has been defined as an independent entity that can be populated with sets of option. Each option will have different choices. Each item may have different options to choose (i.e. side dish + ingredients + cooking level):

- Name of the option (i.e. cook level)
- Option items (cooked, raw)
- Max number of this option selectable
- Added value

5.1.1.10 *Menus*

A restaurant might offer predefined menus with a closed price that might be available only in determined occasions, or might want to prepare a special combination of choices depending of the day of the week. This entity sets up this menu and they menu items related.

- Restaurant
- Name
- Description
- List of groups (i.e. starters/main dish/desert)
- (true/false)
- Items in this group
- Price
- Schedule
- Availability

5.1.1.11 *Kitchen Settings*

Each of the dishes and items of the menu of the restaurant must be entered on the system before they are eligible on the corresponding menus. The main properties of this entity are:

- Unattended mode flag
- Default order
- Default group
- Kitchen can mark ready flag
- Kitchen device URL

5.1.1.12 *Bill*

Each of the dishes and items of the menu of the restaurant must be entered on the system before they are eligible on the corresponding menus. The main properties of this entity are:

- Bill number
- Bill status
- Table
- Date
- Waiter
- Item list
- Subtotal
- Discount
- Taxes
- Total amount
- Take away flag (no table or waiter assigned)
- Comment
- Payment method
- Payed amount (if cash)
- Change (if cash)

5.1.1.13 *Bill Item*

Each of the dishes and items of the menu of the restaurant must be entered on the system before they are eligible on the corresponding menus. The main properties of this entity are:

- Item type
- Bill related
- Quantity
- Options selected
- Status (new, accepted, preparing, ready to serve, served)

5.1.2 Entity Relational Diagram

A relationship between entities and the tables that holds their information may be better understood with a visual representation. The diagram below intends to represent the relationship of this tables in a friendlier manner (bigger resolution image in Annex III):

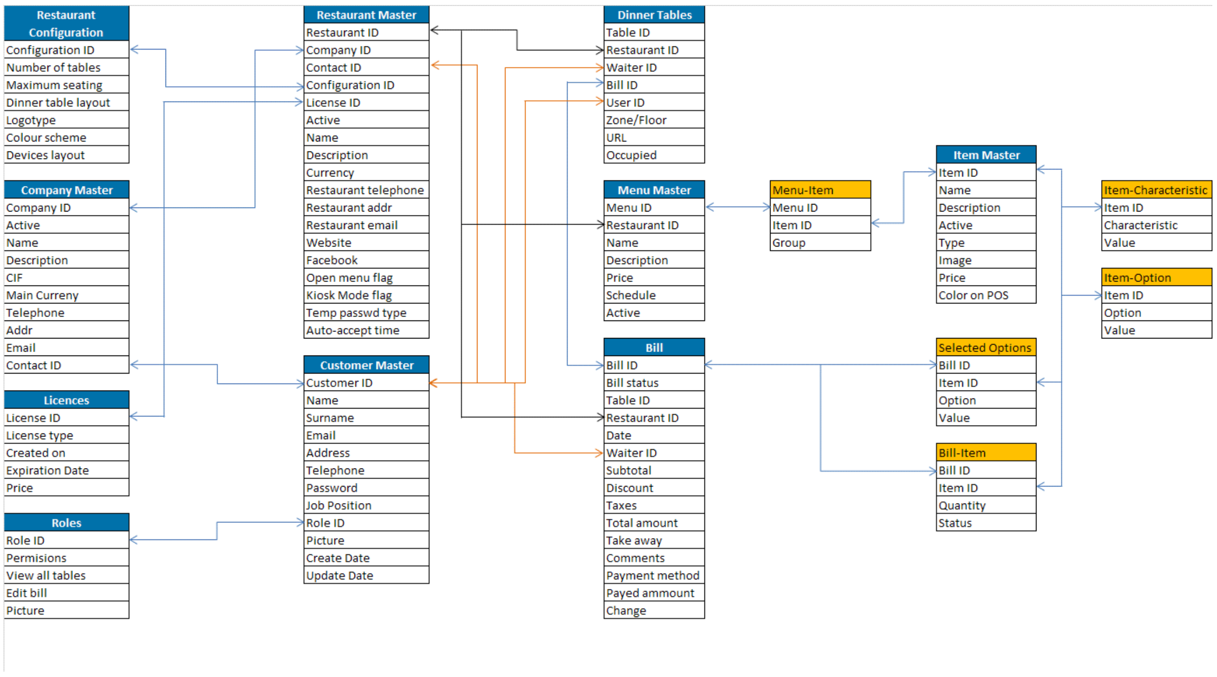


Figure 10 - Entity Relationship Diagram

5.2 Application Architecture

The following section describes the actions performed over the server side for each interaction between the modules and the RMS. It does not intend to provide deep detail about the technologies used but to provide the functional understanding for the application to communicate with the server.

The following sub-sections enumerate the modules (User Layer) that will need interaction with the cloud database. This interaction will be managed by the RMS that acts as business layer, on the server side. This summarizes in that the RMS will be the system in charge of providing the communication and functionality between modules and database.

4.2.1 - Administration
4.2.2 - Point of Sale
4.2.3 - Wait staff (Tablet/Smartphone)
4.2.4 - Customer
4.2.5 - Kitchen

5.2.1 Administration

The administration module provides, mainly, CMS (Content Management System) functionalities, as it acts as the central point to configure the system. It is intended to be as user friendly as possible, minimizing the configuration options to the minimum so it can be powerful and easy to use at the same time.

The communications between the client and the server will be done asynchronous, done by requests of the client to the server, that answers back on demand. There is no need for a bidirectional communication (i.e. web socket), as there is no “true real-time” operations in this module. With a reasonable refresh time, we can manage a less occupied communication channel without affecting the functionality at all.

The following communications and functions will be provided in the Administration module.:

5.2.1.1 *Access to the system*

- **Page request**: The access to the system will be done over the HTTPS protocol, to a certain website, provided by MenuServe and shared by the Administrator, PoS and Kitchen devices. Wait Staff can also access the system through a web application in case the Mobile Specific app is not supported by the device. When the client device requests a new page, the Front End Layer on the Server side will provide all the HTML5 files.
- **Authentication**: User will introduce its username/email and password. This data will be encrypted and sent to the server as a request. The server will decrypt this information and validate that username and password is valid. If it does, the system will provide an answer to the client with the information of the user and its permission and role. If it does not, the server will answer with an error.
  - **Roles/Permissions**: Every request made by the client will contain the session information. The system will always validate, on the server, the permissions of the user before answering back. This ensures an extra security layer if someone is trying to manipulate the client.

5.2.1.2 *Users, Tables/Layout, Menus and Menu Items*

In order to manage all information of their own restaurant, Administrators may be able to create, read, update and remove entities. This kind of control over the information it is called CRUD
.

- **Create**: The client requests an insert of the new data, providing the information for each required field. Server side receives this data and save it in the corresponding tables, assigning a unique ID for its entity.
- **Read**: The client requests the data when it is needed to fulfil a page, the server answers with the requested information.
- **Update**: The client requests an update to the server side with edited information of an entity, the message contains the ID of the item to update and the new data to set up.
- **Remove:** The client requests to remove an item. The message sent contains the ID of the object. The server deletes the information of the item in all the related tables.

5.2.1.3 *Open Menu Configuration*

- **Access settings:** The client requests the actual settings. The server answers with the requested information and it is updated on the client screen.
  - The information regarding Look & Feel of the Open Menu (colours, logotype) are located in sub-page. Related data is requested when accessing this sub-page.
- **Update:** Once settings are modified and saved, the client requests an update with the new information on the message. The server receives this information and updates the setting tables for the specific restaurant.

5.2.1.4 *Set Up Kitchen Device*

- **Access settings:** The client requests the actual settings. The server answers with the requested information and it is updated on the client.
- **Preview:** The client application uses the data received to render a preview of the arrangements.
- **Save:** Once settings are modified and saved, the client requests an update with the new information on the message. The server receives this information and updates the setting tables for the specific restaurant.

5.2.1.5 *Set Up languages*

- **Access languages**: The client requests the list of enabled languages. The server answers with the requested information and it is rendered on the client screen.
- **Adding languages:** The client requests a new language to be included, providing the ID of this language. Server side receives this data and save it in the corresponding table.
- **Enable/Disable languages:** The client provides a request message with the language ID and its property true/false (enabled/disabled). The system updates the language table. Languages cannot be deleted, only disabled, to avoid massive data loosing.

5.2.1.6 *Active Bills*

- **Access data**: The client requests the data needed to fulfil a page, the server answers with the active bills open in the bar and basic information of each of those bills.
- **Update active bills**: The client is set up to request an update to the server every 2 minutes. The server answers back with the new information that is then shown to the screen.
- **Bill details**: When clicking on a bill, a new sub-page is loaded. The client requests full information of this bill. The server answers back with the information that is then rendered on screen. The client requests and update every minute and answers with the updated data or with a “no updates” message.
- **See closed bills**: When clicked, a new page is loaded with the closed bills of the day. When accessing, the client requests all bill information from a day, by default the actual day of the client device.
- The server answers back with a list of all Bills. The administrator has the chance to change the day requested and reload the list of bills from past days.

5.2.1.7 *Statistics and Reporting*

- **Access Dashboard**: The client requests the data needed to fulfil a page, the server answers with the active bills open in the bar and basic information of each of those bills.
- **Basic reporting:** The client application requests the basic report data (average customers, average income, most requested dishes…) to the system by providing only the restaurant ID, client time and the parameter to request the basic report. The client answers back with all de data related for the last 15 days. The client will then display this information with different charts and tables.

  - **Setting up segments:** The client allows selecting different segments of information, to provide most detailed information. The segments options are provided by the server when accessing the report, in order to limit the report to the restaurant depending on the license plan.

- This segments can include timing (expand/shrink date range) or metric (show only number of clients). When the administrator set up a segment, the client requests the new data by requesting it to the server providing on the message the information of the segment.
- **Advanced report**: If the restaurant has advanced reporting available, it can access to more reporting pages than only the report. This pages allows extended reporting by predefined metric, that includes but are not limited to the basic reporting ones. In the same procedure that the basic reporting, when accessed, the client system requests the information of the report to the server. It is then returned to the client and rendered on screen. The segmentation possibilities are present.
- **Custom reporting**: The Administrator system may be set up to allow advanced, custom reporting. This page will allow the administrator with the proper license to select the metrics that want to see on screen and its segments, and then launch a query. The client system then, will sent a message to the server with the list of metrics and segment that wants to receive, and the server will answer with the related information, that will be rendered on screen.

5.2.2 Point of Sale

The Point of Sale device must provide a platform where every restaurant's employee can use to attend the customer's billing requests and to fulfil, at any time, the ordering information if, by some reason, the device of some waiter is not properly working.

Unlike the Administration module, the PoS must ensure a bidirectional communication with the server side at any time, to avoid any mistake at the moment of billing the customers. It will, also, provide asynchronous communication when a bidirectional communication is not required.

The following communications and functions will be provided in the Point of Sale module.:

5.2.2.1 *Access the system*

- **Page request**: The access to the system will be done over the HTTPS protocol, to a certain website, provided by MenuServe and shared by the Administrator, PoS and Kitchen devices. Wait Staff can also access the system through a web application in case the Mobile Specific app is not supported by the device.
- **Authentication**: User will introduce its username/email and password. This data will be encrypted and sent to the server as a request. The server will decrypt this information and validate that username and password is valid. If it does, the system will provide an answer to the client with the information of the user and its permission and role. If it does not, the server will answer with an error.
    - **Roles/Permissions**: Every request made by the client will contain the session information. The system will always validate, on the server, the permissions of the user before answering back. This ensures an extra security layer if someone is trying to manipulate the client.
- **Dashboard**: When logged in, the system will show large buttons to select each functionality according to the information acquired by the system when logging in. Large buttons are intended to proportionate better touch-screen user experience.
- **Menu Items**: The list of menu items is requested to the server and when received, stored in the client side cache to increase performance. By taking advantage of the bidirectional communication, if there is any update to the items, the client system will request an update to the server.

5.2.2.2 *Real Time Billing system*

- **Real time data**: The client system will stablish a bidirectional connection with the server platform via WebSocket protocol. With a full-duplex communication channel, both client and server side can be the requester or answerer, allowing notifications, always up to date data and events to be displayed on the client without the need of this client application requesting any information. This results in the client side listening for any incoming message a displaying it as requested.
- **Active bills**: The basic information of every bill will be updated automatically every time the server provides updated information. The WebSocket will be used to provide basic information of every order and bill, but more detailed information can be obtained by accessing the bill.
- **Add bill:** The client allows adding bills to the system. When adding a bill, it is required to enter at least the Waiter and the table number, all other data is automatically filled or left by default. When accepted, the client sends an asynchronous request with the provided data to the server, the server creates the new bill entity and answers with a successful or error message. The new bill is listed automatically on the real time bill list thanks to the WebSocket.
- **Bill details**: When clicking on a bill, a new sub-page is loaded. Once the bill is accessed, the most detailed information will be requested to the server asynchronously. The server answers back with the information that is then rendered on screen. The client will request the new detailed information every time that receives an update of the basic information via de WebSocket, granting this way a real time information. The bill details are modifiable following the criteria below:
    - **Add:** The client has the list of all menu items loaded. Every time one item is selected, their ID and information such as the quantity and the options are saved in local storage and added to the onscreen bill. This full bill is provided to the system when the bill is approved o the PoS, the system returns either a success message or an issue when this action is performed.
    - **Modify:** Bill items can be modified. By selecting an item on the onscreen bill, a pop-up is rendered with the details of this ordered item (i.e. quantity, side dish, options). This information can be edited and saved. The same as when adding items, the updated bill is sent to the server via post, and the server saves the information in the corresponding database.
    - **Delete:** Items of the bill may be deleted. When the bill is approved, the information is sent to the server, who updates the corresponding tables. It does not delete items on the table but mark them as “disabled”, in order to keep track of every action performed.
- **See closed bills**: When clicked, a new page is loaded with the closed bills of the day. When accessing, the client requests all bill information from a day, by default the actual day of the client device.

5.2.2.3 *Payment and bill closing*

This process is mainly done on the client side until all the information has been provided.

- **Start payment flow:** The payment methods and flow are executed on the client side. Once the waiter presses the "PAYMENT" button, modal windows is showed with the payment options. The client application sends a request to the server with the bill ID and the flag that the bill is pending for payment, for tracking procedures.
- **Invoice (bill) printing:** When the button "PAYMENT" has been pressed, the invoice is automatically printed on the default's system printer. In a standard PoS configuration, this will be a thermal printer. The invoice can be printed independently at any time without having to carry the payment.
- **Setting payment method**: The payment method can be selected on this screen, the client shows on screen buttons to select the payment method (by Card, by Cash, by mobile application or others). It also allows to calculate the share of the bill between a selectable number. If "Cash" has been selected, the next screen will let the waiter enter the amount the customer has provided in order to calculate the exact change.
- **Receipt printing and bill closing**: If the payment method has been accepted, a receipt for the transaction will be printed. This action also sets up the bill as closed. MenuServe follows the standard for PoS and USB Cashier Drawers, adding the "Bell" character at the end of the Invoice [16] [17] in order to let this Cash Drawer open allowing the waiter to provide the change. At this point, the client system requests an update via post to the server, providing all the payment information to the server.
- The Cash Drawer can only be opened when a payment has been done and its invoice is printed. This ensures both internal and external security of the Restaurant's physical cash.

5.2.3 Wait Staff Device

The Wait Staff device is the key tool for waiters and waitress of the restaurant. It must allow them to do their job the easiest and fastest way possible without any setback. An intuitive interface to navigate through and allow the waiter to note quickly every action is, together with providing the right functionality, the key factors of this module.

As in the PoS, the Wait Staff module must ensure a bidirectional communication with the server side at any time, to avoid any delay between actions and allowing waiters/res to react quickly to any change. This ensures an error free environment and reduces human mistakes. It will, also, provide asynchronous communication when a bidirectional communication is not required.

The following communications and functions will be provided in the Wait Staff module.:

5.2.3.1 *Access the system*

- **Page request**: The access to the system will be done via a specific MenuServe application that can be downloaded for free from the application distribution platforms Android, iOS and Windows. Wait Staff can also access the system through a web application over HTTPS in case the Mobile Specific app is not supported. In this last case, the device browser must be compatible with JavaScript and HTML5.
- **Authentication**: User will introduce its username/email and password. This data will be encrypted and sent to the server as a request. The server will decrypt this information and validate that username and password is valid. If it does, the system will provide an answer to the client with the information of the user and its permission and role. If the role of the user is Wait Staff, the application will log in to the system.
- **Menu Items**: The list of menu items is stored on the device memory to maximize the performance and communication speeds. By taking advantage of the bidirectional communication, if there is any update to the items, the client device will request an update to the server.

5.2.3.2 *Attending customers*

- **Create a new bill:** The client platform is in charge of providing the information for this new bills to the system. The data is sent to the server via post request with a message containing the table assigned and number of guests (entered by the user) and the date, waiter assigned and other relevant information (automatically gathered). The server returns a confirmation message if created successfully.
- **Bill list**: The client system will stablish a bidirectional connection with the server platform via WebSocket protocol. The list of bills assigned to the wait staff will be automatically updated. When there is an external update on a Waiter's bill, the server side will make use of the stablished WebSocket to ensure the client side can update the information and provide a visual feedback to the waiter.
- **Bill details/items**: When clicking on a bill, a new sub-page is loaded. Once the bill is accessed, more detailed information will be requested to the server via GET request. The server answers back with the information that is then rendered on screen. The client will request the new detailed information every time that receives an update of the basic information via de WebSocket. The bill details are modifiable following the criteria below:
  - **Add:** The client system has the list of all menu items loaded. Every time one item is selected, their ID and information such as the quantity and the options are saved in local storage and added to the onscreen bill. The application will have a button allowing the client to provide this information to the server via post request.
  - **Modify:** Bill items can be modified. By selecting an item on the onscreen bill, a pop-up is rendered with the details of this ordered item (i.e. quantity, side dish, options). This information can be edited and saved. The same as when adding items, the updated bill is sent to the server via post, and the server saves the information in the corresponding database.
  - **Delete:** Items of the bill may be deleted. When the bill is approved, the information is sent to the server, who updates the corresponding tables. It does not delete items on the table but mark them as "disabled", in order to keep track of every action performed.
  - **Delivered:** Specially when the Kitchen device is working unattended, items status must be updated by the wait staff. An update request may be sent to the system with the bill ID, item ID and the new status. The client application may provide a button to enable this status change.
- **Edit bill information:** The client application may request an update of the bill entity information. The request is the same as for creating a new bill, but additionally, the bill ID is provided. The server side receives this information and updates the bill description table.
- **Delivering ready item:** When an order for a table is ready to serve, the client application will receive a notification, via WebSocket. The client device will also receive a Push Notification via

OSPNS (Operating system push notification service) when the specific mobile application is used. [18]

5.2.3.3 *Open Menu*

When the Open Menu is enabled, the workflows for opening a bill and confirming orders of customers are a bit different. The application layer will check this value (Open Menu true/false) when effectuating any action. It will also validate if Kiosk Mode is on or not.

If the value of Open Menu is TRUE, the following additional actions will be performed on the system:

Create a new bill with KIOSK MODE

- **Activating Kiosk:** The process of activating a Kiosk is almost transparent. When a bill is created, the workflow is exactly the same as when there is no Open Menu. The only difference is that the server will enable the Kiosk of the table assigned when the value "Kiosk Mode" is true.
- The Kiosk may be locked down at any time from the Bill Detail screen. The WebSocket will provide this functionality in real time.

Create a new bill with KIOSK MODE

- **Provide/Receive temporary password:** When the Open Menu is enabled for the Customer Devices, the system will provide a temporal password for this the bill to be accessed from an external device. After the Bill Opening process have been performed, the server will return not only a success message but a temporary password and/or a QR image on the wait staff device, depending on the configuration. The system is loaded with a dictionary that allows to assign easy to remember passwords for each table and bill.
  When logging in, the system will match the table from where the customers connect (by a QR/web address referring to this table) and the password provided. This double factor will ensure security.

Add items to bill

- **Confirm User orders to kitchen:** When a customer has ordered via Open Menu, a notification will appear on the Waiter Device (both via app using WebSocket and via Push Notification). The notification will provide a link to the bill, allowing the wait staff to read the recent added orders. When accepted, the client application will send a request with the ID of the table and the flag "approve order". The server side will then treat the new orders as performed by a wait staff and will follow the usual workflow.

5.2.3.4 *Point of Sale*

- **Start payment flow:** The payment methods and flow are executed on the client side. Once the waiter presses the “PAYMENT” button, modal windows is showed with the payment options. The client application sends a request to the server with the bill ID and the flag that the bill is pending for payment, for tracking procedures.
  - **Limitations:** This option can only be performed once the bill has been printed (on the PoS).
- **Setting payment method**: The payment method can be selected on this screen, the client shows on screen buttons to select the payment method (by Card, by Cash, by mobile application or others). It also allows to calculate the share of the bill between a selectable number. If “Cash” has been selected, the next screen will let the waiter enter the amount the customer has provided in order to calculate the exact change.
  - **LIMITATIONS:** The Wait Staff will still have to use the PoS to print the Receipt, to provide change to the customer if the payment method requires it, and to close the bill.

5.2.4 Kitchen Device

The kitchen device has the main functionality to show on screen the incoming orders, so the Kitchen Staff may have all the information needed to prepare the dishes and provide them fast and in a reliable way. A restaurant that is able to provide with the orders to its customers fast and efficiently has a great added value.

The Kitchen Device must ensure a bidirectional communication with the server side at any time, to have always the most updated list of orders on screen. Real time data means that the Kitchen staff can priories properly when preparing the dishes and can communicate with the wait staff from the same place (when attended approach).

The following communications and functions will be provided in the Kitchen module:

5.2.4.1 *Access to the system*

- **Page request**: The access to the system will be done over the HTTPS protocol, to a certain website.
- **Authentication**: User will introduce the username/email and password. This data will be encrypted and sent to the server as a request. The server will decrypt this information and validate that username and password is valid. If it does, the system will provide an answer to the client with the information of the user and its permission and role. If it does not, the server will answer with an error.

5.2.4.2 *Unattended Approach*

The unattended approach is intended to be used on screens and TVs that does not allow a direct interaction. This device needs to be connected to a PC, for example the PoS, via Chromecast, cable or similar.

- **View Orders:** Once logged in, the client will stablish a bidirectional connection with the server side via WebSocket, this will ensure that all data is received by the client in real time, without the need of the client to continuously request the data. Although a GET request every one minute may be a perfect valid approach, it needs to reload the page, while a WebSocket allows the data to stay on screen even when a connectivity error appears, without any interruption of the order list. The server provides the client with the information to render the order list according to the ordering criteria saved on the server.

5.2.4.3 *Attended Approach*

The attended approach is intended to be used with an interactive device such as an Android TV, computer, tablet, etc. as it requires the input of the Kitchen Staff to complete the workflows.

- **View Orders:** The orders are displayed on screen with the same criteria of the unattended approach. The main difference with this one is that the screen of the client will show action areas that may be used to interact with it. The order preferences are loaded from the server also.
- **Ordering/Grouping items:** The client offers two select items to choose both grouping and ordering of the order list. When a filter is applied, the client applies the new criteria on the onscreen order list, all of this process is executed on client side as the bidirectional communication only updates the order list and not how it is displayed. Although, the new ordering information is posted to the server so it may be loaded when logging in to the system again.
- **Order ready:** When clicking on an order, the ready button is enabled. This action will send to the server the order ID and the flag "ready", and the server will be in charge of providing notification to the waiter/res assigned to the table that this order is assigned to.
- **Expired item:** The client can communicate to the system that an item is expired. This communication message needs the Item ID and the flag "expired". The server will be in charge of updating the menu item information from PoS, wait staff device and Open Menu. The wait staff will receive a notification on its device to communicate the issue to the customer and the not processed items expired will be removed from the bill.

5.2.5 Open Menu/Customer Device

This functionality allows customers gain control of the service that is provided to them via their devices or a Kiosk on their table. Customers can bypass waiter/res in the act of ordering the dishes they wish for, validate their bill, or get detailed information on the menu of the restaurant.

Customers will be able to access this information both with an Open Menu application or directly from a website, increasing the compatibility and user friendliness. The Open Menu will take advantage of WebSocket in order to communicate with the server bidirectional the possible issues, but a big amount of the actions will be done on the client side and after posted to the server via asynchronous requests.

5.2.5.1 *Enrolment/Access the system*

Via Customer Device:

- **Page request**: The access to the system will be done via a specific MenuServe application available for Android, iOS and Windows, or over a website via HTTPS. In any case, a webpage must be provided with a specific parameter assigned to each table. When accessing to the requested webpage, if the application is available on the customer device, it will be launched; if not, a website will be loaded. Bothe the application and webpage, when accessing to the URL provided, will be sending a request to the server. The server will then provide back the information of the restaurant and the table where they are logging in.
- **Authentication**: To authenticate to the system, the table may only be logged in when it has an active bill assigned. If not, the system will provide an error. Once the server has returned the information to the client and the front End has been loaded, it will be required to provide the temporary password provided by the wait staff when assigning the table and opening the bill. When entering the temporary password, it will be encrypted with the SSL certificate on the client device and sent to the server via request message. If the password is valid, the system will return a session token and allow the client to request the FE and menu information.

Via Kiosk:

- **Access the system:** The Kiosk is already connected to the system and listening to the server through the WebSocket. It will be enabled for use when it receives a notice from the server that the bill has been opened.

5.2.5.2 *Enrolment/Access the system*

- **Menu Items**: The list of menu items is requested to the server via GET and when received, stored in the client side cache (web browser) or in the device storage (when mobile application) to increase performance.
- **Navigation**: The navigation over the menu is performed on client side. When an item is clicked, a more detail pop up will load. At this moment, the client requests detailed information of the item, as an example, the picture and description.
- **Ordering**: The systems uses a "basket" system where the orders can be saved on a list before posting them to the server. The client saves the ID of each item, quantity and option for every unit on the device memory (or cookies when using a web browser). When placing the order, all the list of selected items and its information is post to the server. The server answers back with a success message. If success, the information is stored as ordered items.
- **Reviewing ordered items (bill)**: The client has the possibility to review the full list of ordered items. This list items are updated automatically via the WebSocket, that provides information of the Item ordered, quantity and its status. The details of price and other information is loaded on the device memory, so there is no point in send it using the WebSocket channel.
  When there is a status change, a notification is showed on screen in real time thanks to the WebSocket protocol.
- **Editing orders:** Orders can be cancelled by the customer if their status are still pending of confirmation. When done, the client updates the information of the item on the bill and the server informs the wait staff device with the new information. Items that are already approved may only be deleted by wait staff.

5.2.5.3 *Wait Staff assistance*

- **Request waiter/res:** Customers may use the client application to request for the wait staff assistance. When the defined action to do so is performed on the client, this communicates the system with a flag of “waiter request” and the information of the table. The server will receive this information and send it to the wait staff device as a notification. Thanks to Web Sockets, the wait staff will receive it instantaneously and it will be listed on its notification queue with the proper message.
- **Request bill:** The same process can be performed to request the bill and, consequently, request the closing of this “virtual” bill.

5.3 Platform Architecture

The platform architecture aims to describe the technologies that will be used in order to provide a fully functional cloud system and the application modules.

5.3.1 Hardware architecture

The hardware elements necessary to run the system in a data centre will be:

- **Firewalls** connected to an Internet Service Provider on one end and to load balancers on the other.
- **Load Balancers** working in parallel, connected to the front-end servers. The load balancing is carried out at IP level and configured to use sticky sessions.
- **Database Servers** connected to a **disk array** or **Storage Area Network**.
- **Switches and networking** – All required switches to configure VLANs and interconnect the previously listed devices.
- **Infrastructure to support the virtual servers.**
- **Storage SAN** – Storage Area Network with 5TB of storage space.

In addition, the virtual infrastructure will require the following virtual servers:

- **Front-end Servers**, running Apache, connected to the application servers.
- **Application Servers,** running Tomcat and the system, connected to the database.
- **Monitoring Server** for searching, monitoring and analysing machine-generated big data and network metrics.

Firewalls and load balancers do not need to be dedicated (they could be shared in the data centre used). Servers should be dedicated due to security reasons. In the data centre, a backup system is also needed.

All the infrastructure provided above may be provided by a third party, for example Amazon. Specially for an initial phases of the life of MenuServe, there is no need to invest on an ad-hock data centre. The architecture is intended to be highly and easily scalable in order to be able to attend the demand at any time. The system allows redundancy of the system, to provide a backup system is one fails.

High availability and reliability are key factors in adding value to the product and providing an excellent user experience for restaurants.

The figure below shows the architecture diagram in a most detailed level and it follow IT best practices in architecture design.

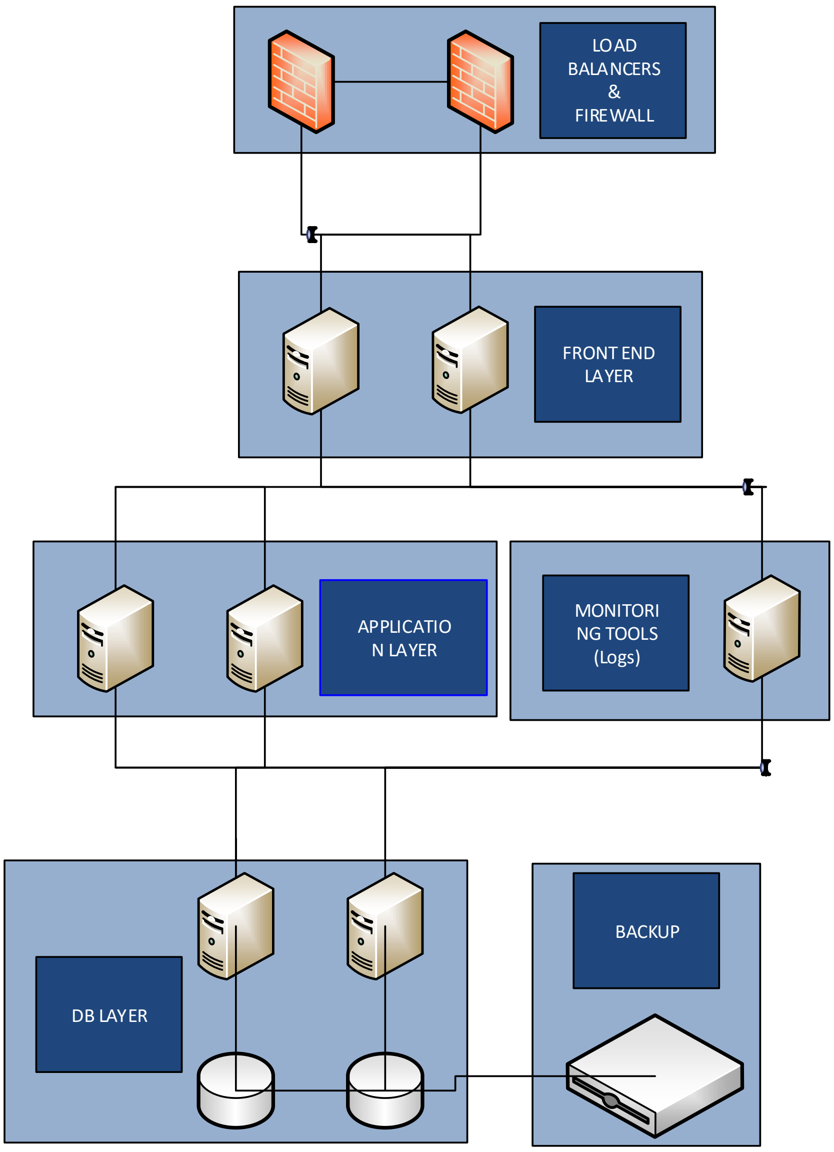


Figure 11 – Server infrastructure Diagram

5.3.2 Software Architecture

The software architecture of MenuServe needs to use different frameworks depending on the module and layer of each component, in order to fulfil all its requirements. The main environments that can be found are:

- Server Operative System
- Application Layer (server side)
- Web Application
- Mobile Applications

Each of this modules count with a front end, application layer, database and monitoring tool.

- **Front End:** User interface of the application and in charge of providing both feedback to the user and a way to interact with it.
- **Application layer:** Framework in charge of providing the infrastructure or environment where the Front End can be executed or acquired.
- **Database:** Platform providing the storage of the information required or generated by the application.
- **Monitoring**: Each application needs to be monitored in order to gather big-data and provide analytics, operational reports and security support.

For the software architecture, the initial proposal is as follows.

Software Type	Front-end	Application	Database	Monitoring
Server OS	CentOS 7 64	CentOS 7 64	Oracle Enterprise Linux 64	CentOS 7 64
Client Web App	HTML5 AngularJS	Apache HTTP Server	Cookies Local Storage	Graylog
Mobile App	HTML5 AngularJS	Cordoba Hybrid Mobile Application	Local Storage	Graylog
Application Layer	Apache OpenSSL	Tomcat Java PHP	Oracle Database or MySQL	Graylog

Table 8 - Software Architecture

5.3.3 Redundant System for Recovery

MenuServe will have a redundant system that will act as a disaster recovery unit. It has to be identical to the main system used and must be updated with the latest information at any time. In the unlikely case of the system failing, the functionality of MenuServe can be switched to the second platform in order to continue the service with the minimum impact possible.

For the initial phases of the life of the product, this both instances will be hosted by Amazon but deployed in two physically separated data centres. This will grant us that if the system is failing by a physical situation, the backup platform will not be affected.

An important requirement for a redundant system is that its database must be synchronized in real time with the primary system database to minimize the time required to switching from primary to redundant platforms.

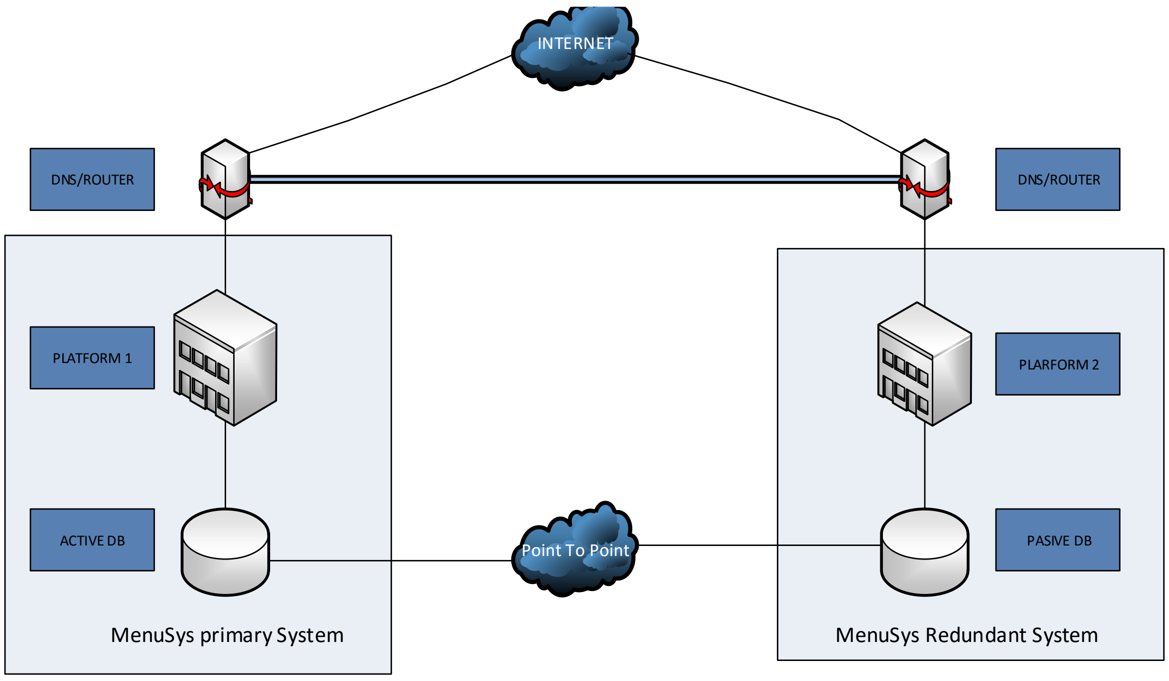


Figure 12 - Redundant System Diagram

6 Visual Design and User Interface

An application can be very well designed in its interiors and be very efficient in every aspect of how it treats the data, communicates between modules and processes all this information. But if its visual aspect is not appealing, and the most important, it is not designed having in mind that people will need to use the application in the easiest way possible, the application itself is doomed to fail.

This is a very short rationalization of a true situation, but as simple as it may seem, an uncountable number of times really good concepts and applications have been cancelled or abandoned because the users were not able to understand how it was supposed to handle. When users get frustrated, they stop using an application, because it provides the feeling that it is creating more problems than doing what it tries to, solve a problem.

In addition to this logical concept of how an application is handled (how it orders and displays the data, which messages shows, which navigation paths provides, etc.), there is another aspect that cannot be underestimated and it is the visual quality of it. Nowadays, caring a computer everywhere is the norm, everybody installs and uninstalls software by thousands, uses an infinity of applications and exchanges information via electronic devices every day. The big competence in the sector has carried the applications not only to be easy to use, but to be attractive to the eye while doing it. This is the main reason User Experience is a must in every application that goes out.

6.1 User Experience Guidelines

Design guidelines has been created in order to provide a base ground on how to “standardize” the main functionalities and interactions between man and computer. A regular user will interact with many applications every day, and will expect them to be easy to use and not needing a training for every one of them; if it is too hard to understand or remember, he will ditch it. It makes sense then to use the same ways of interaction that users already know onto new created applications, homogenising the inputs and outputs of an application.

This homogenization translates into design guidelines. These guidelines are usually provided by the manufacturers of the operative systems, looking for a base of interaction that aligns through all the system and its apps so the user will find the overall product more appealing, and as a last instance, more susceptible to keep using the system.

As this guideline are provided by companies, like Google Inc. for Android or Apple Inc. for iOS, they may change from one system to another. For example, Android has a dedicated back button, while iOS does not have one, this may provoke inconsistences between these two companies’ guidelines. This is the main reason why those rules are called *Guidelines* and not a *Standard*, although, when having a general look, almost all bases of interaction are similar.

MenuServe uses, as guidelines to base its design, Material Design. Although, it is not limited to this guideline and may follow other similar design tendencies in order to provide the best user experience possible.

6.1.1 Material Design

Material Design is a design base or language created by google, and that provides the guidelines to design applications and websites visuals according to what Google defines as the fusion between the classic principles of good designs with the innovation and possibilities of new technology.

According to Wikipedia [19], Material Design is:

> “**Material Design** (…) makes more liberal use of grid-based layouts, responsive animations and transitions, padding, and depth effects such as lighting and shadows. Designer Matías Duarte explained that, "unlike real paper, our digital material can expand and reform intelligently. Material has physical surfaces and edges. Seams and shadows provide meaning about what you can touch." Google states that their new design language is based on paper and ink”

Material Design is an extension of the recent “flat design” tendencies that have been gaining popularity last years. It is a very minimalistic design, and this allows it to be very adaptable to the basis of responsive design, allowing the apps to “fluidly” adapt to almost any screen size on the market. This is key point of Material Design, as MenuServe is intended to be as widely usable as possible, so it needs to adopt a design base that allows it to be correctly displayed in all the devices that may use it. Material Design, in this point, adapts perfectly to the objectives of MenuServe.

Figure 13 - Representation of Material Design Responsive design

The minimal approach of Material Design also provides great visual outputs, allowing the user to rapidly detect the points of attention and the items that calls for an action, providing that way a very intuitive interaction between user and application. The use of flat colours also provides a great base to allow the customers adapt the MenuServe to their colours, providing this way an added value of personalization that will adapt to the needs of the client.

By last, and as an advantage more of design bases themselves that Material Design exclusively, having a “design language” available will allow MenuServe to provide a strong visual homogeneity between the different modules.

The following sections will show some of the design patters used in the design of MenuServe and that are shared with Material Design.

6.2 Navigation Tree

MenuServe is a RMS solution that offers not only an application, but a suite of management for restaurants that aims to cover all their basic needs just outside the box, and allowing the restaurant owners to take control of the application and start using it as soon as they register.

Following that concept of a solution suite, MenuServe has 5 main modules that interacts between them but they also act as individual applications by their own. In this point, not only is important to provide a visual alignment between them but to provide similarity of behaviour in every module. Following this point, the main idea of the design of the screens of MenuServe is to maintain the maximum amount of simplicity, providing the least amount of actions to provide each functionality.

This is very important not only to provide an intuitive platform but to speed up the performance of the workers that will be using it to provide a service to the final customers. Is at that point where a fast and reliable designed application will make the difference.

In order to provide a simple but fast navigation through each module, all sections will have menu that will allow fast change to other important sections of the module. Navigation also has been designed to follow logic workflows and correlate actions with its next steps.

The following sub-sections shows the main screens of each module through their navigation tree.

6.2.1.1 *Back office (Administration)*

The navigation three of the back office module:

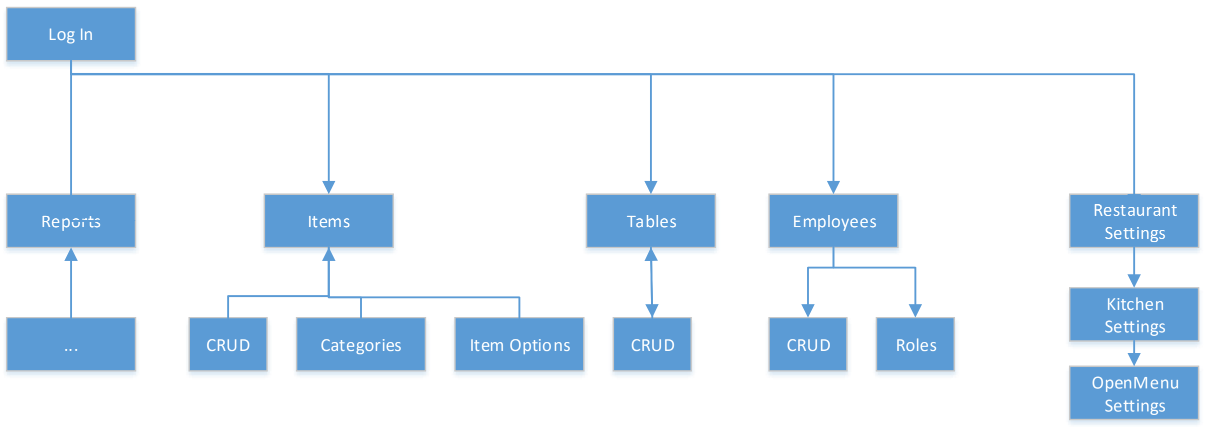


Figure 14 - Back Office navigation tree

6.2.1.2 *Kitchen Device*

The kitchen's device navigation three is very simple. Nerveless, this simplicity does not mean that this module is dispensable, as it is one of the key features of MenuServe and is the one intended to keep communications running between Kitchen and table zone.

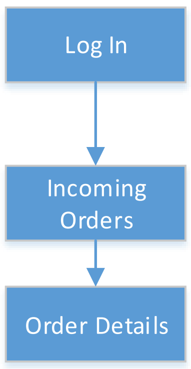


Figure 15 - Kitchen Device Navigation Tree

6.2.1.3 ***Point of Sale***

The navigation three of the POS:

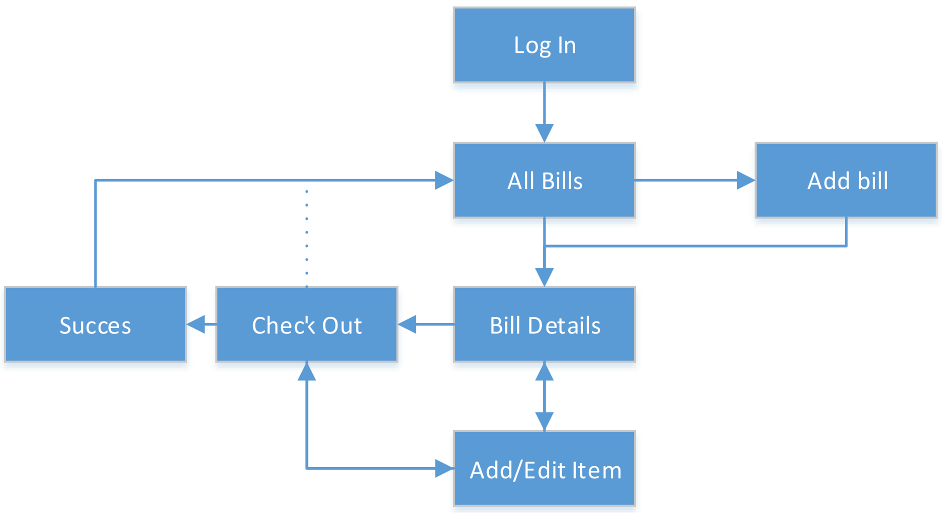


Figure 16 - POS Navigation Tree

6.2.1.4 ***Wait Staff Device***

The navigation three of the Waiters Device:

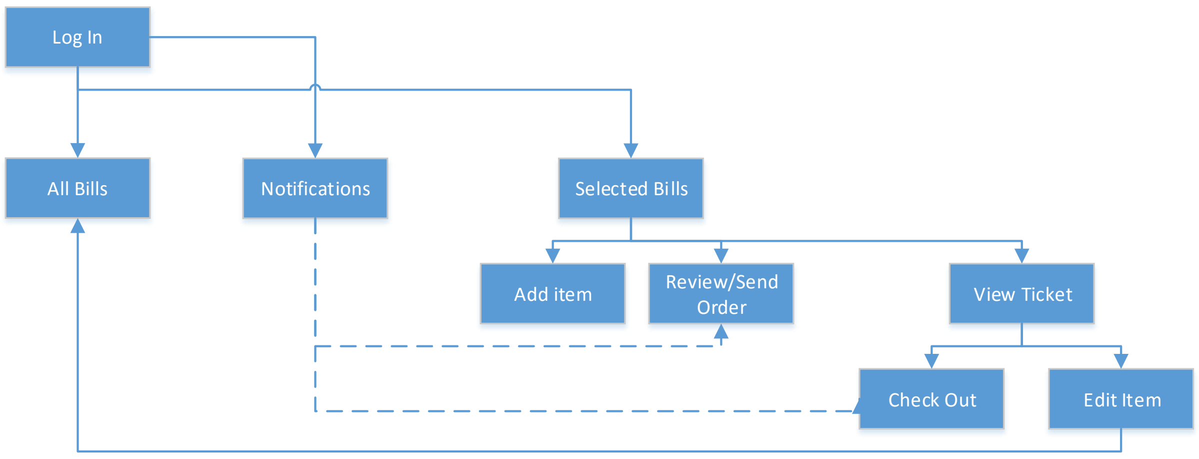


Figure 17 - Wait Staff Device Navigation Tree

6.2.1.1 ***Open Menu***

The navigation tree of the OpenMenu is quite similar to the Wait Staff device, but only focused on the ordering, as the OpenMenu automatically logs in to a particular table, not using the rest of the features that the Wait Staff module does.

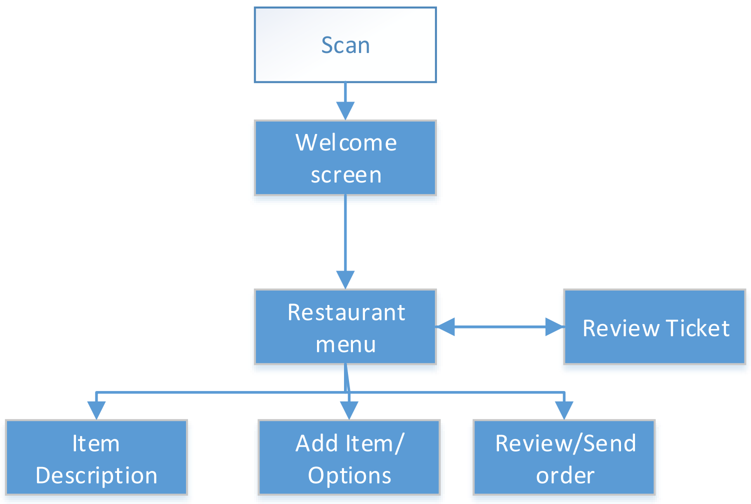


Figure 18- Open Menu Navigation Tree

6.3 Colour and Logotype

A very important part of the design of an application, or set of applications in the case of MenuServe, is to provide a visual identification through a logotype and a colour palette. This visual identification will also provide a coherence between the different modules and facilitate the navigation through the system and the way the users understand the inputs and the outputs.

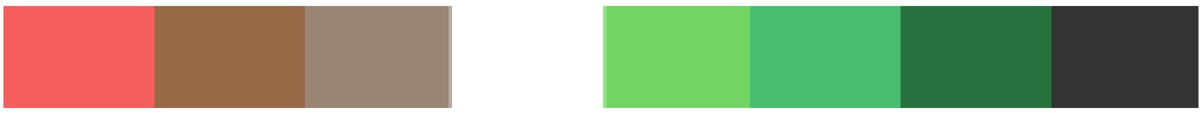


Figure 19 - MenuServe Colour Palette

MenuServe is intended for restaurants, so the colour palette that has been selected reflects colours that may be found in food. Warm colours, greens and browns, that reflects the colour of food and therefore activate the subcontinent response of hungry, and relating this colours to restaurants. This soft natural colours and tones that encourage users to relax and, especially green, is a great colour to communicate freshness and healthy intentions.

This palate of colours is also used by reference of the restaurant industries online as could be “The Fork” or “trip Advisor” and has proven to be very valid in this context. Although the visual identification of MenuServe if using similar palettes than big companies may be reduced, the market that attacks MenuServe is very different than the one these companies attack. Furthermore, MenuServe is a tool for the restaurants to use, allowing the full customization of the colours scheme, so in this way, MenuServe does not have the same marketing requirements regarding colour than a company that its main customers are the big public.

This colours may also be found on the logotype of the MenuServe, specially the soft green and the soft brown, providing a sensation of calm and relax, everything is under control.

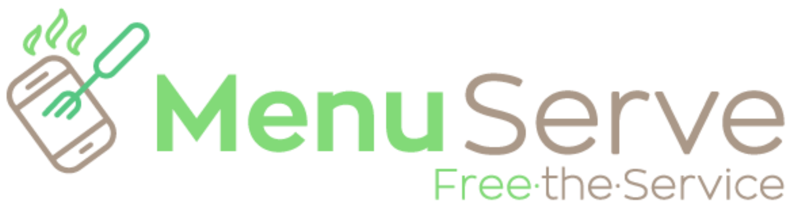


Figure 20 - MenuServe Logotype

The logotype of MenuServe represents a cocking fork, used to prepare meat in barbeque, coal or hot fire, and a mobile device, that is the base device that makes MenuServe more accessible than any other similar system.

The design of the logotype if flat and horizontal, allowing it to be used in documents, websites and other documents both digital and physic without problems. It carries the slogan of MenuServe, “Free the service”, but this can be added or removed depending on the situation.

The logotype also follows a minimalistic design, providing an identifying a unique Icon that is offers an idea of what the application is about and can act as the logotype itself without the need of the title. For mobile applications, it will be really useful.

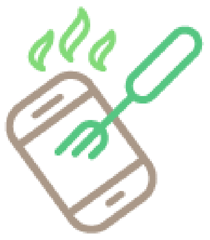


Figure 21 - MenuServe Icon

The logotype of MenuServe it has been performed with Adobe Illustrator CC, a program specialized in vectored design. The font used is ZonaPro, a sans serif font very similar to OpenSans o Roboto that offers a modern look and on line with the actual tendencies.

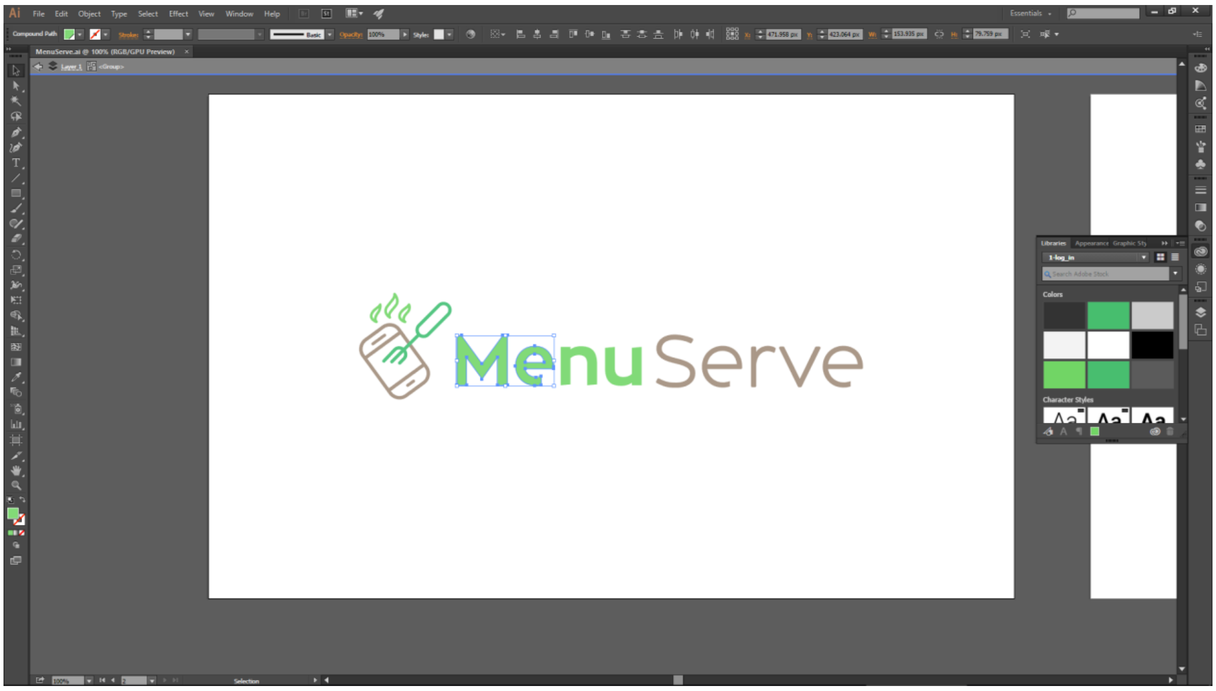


Figure 22 - Screenshot of the logotype in Adobe Illustrator

6.4 Wireframes

Before starting with the definition of HQ design for MenuServe, concept drawings of each page have been made. This process is called wireframing and allows the designers not only to make their minds in order on how to design an application, but to detect any missing part of the functional definition and to detect any flaw in the workflows.

A wireframe [20] is a visual definition or scheme that represents the skeleton of a website or application. Its main purpose is to compose the basic composition of each page of an application in order to provide with a first design of what will be the final page. This wireframe lacks of a visual definition in terms of colour, typographic and visual effects and are oriented to provide the page organization, the functionalities that it will offer and its behaviour. In other words, the wireframes focus on what the application will do, not how it will look.

The wireframes can be done in paper or in specific software, both being as valid. For the case that concerns this document, MenuServe wireframes have been created by hand. The main motivation in order to do the wireframes with pencil and paper is that this media is more versatile, allowing faster changes and better performance.

Approximately 46 wireframes have been created between the 5 modules of MenuServe. When working with high number of pages as in this case, is very common that when designing a page in a more advanced workflow, the designer realises that three is something missing in a previous wireframe screen. Using paper and a pencil allows for a fast change of this wireframe that needs an update without requiring a big effort in time.

Below, you can find some examples of wireframes performed. In order to see all the wireframes created and ordered by module and workflow, refer to ***Annexes IV – Wireframes***.

6.4.1.1 *Administration module*

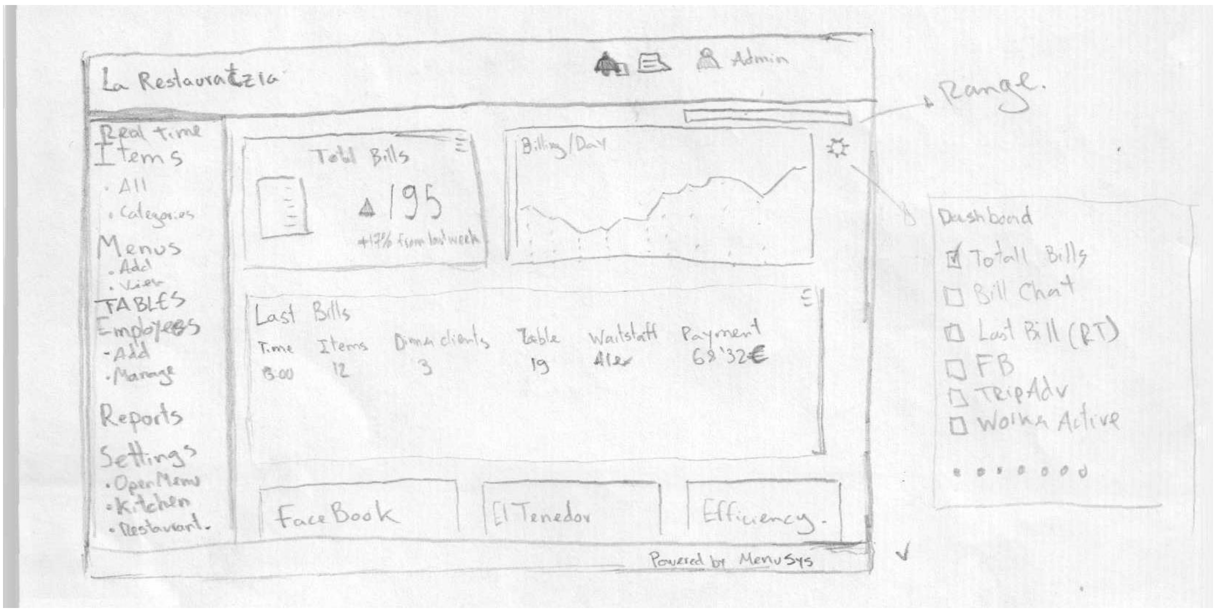


Figure 23 - Wireframe 1

6.4.1.2 *Kitchen Module*

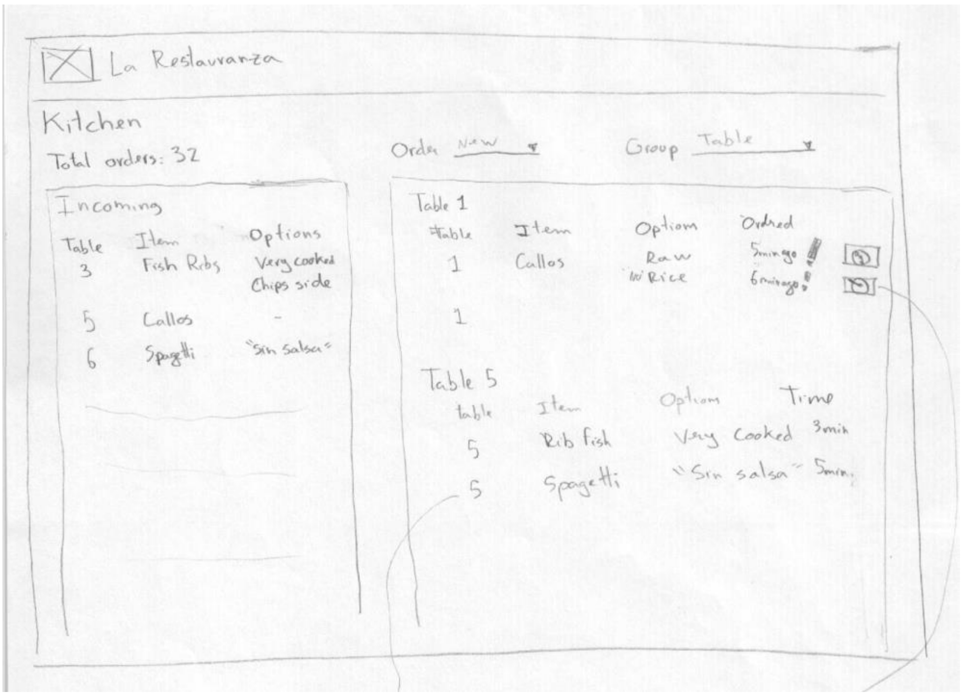


Figure 24 - Wireframe 2

5.4.1.3 *POS Module*

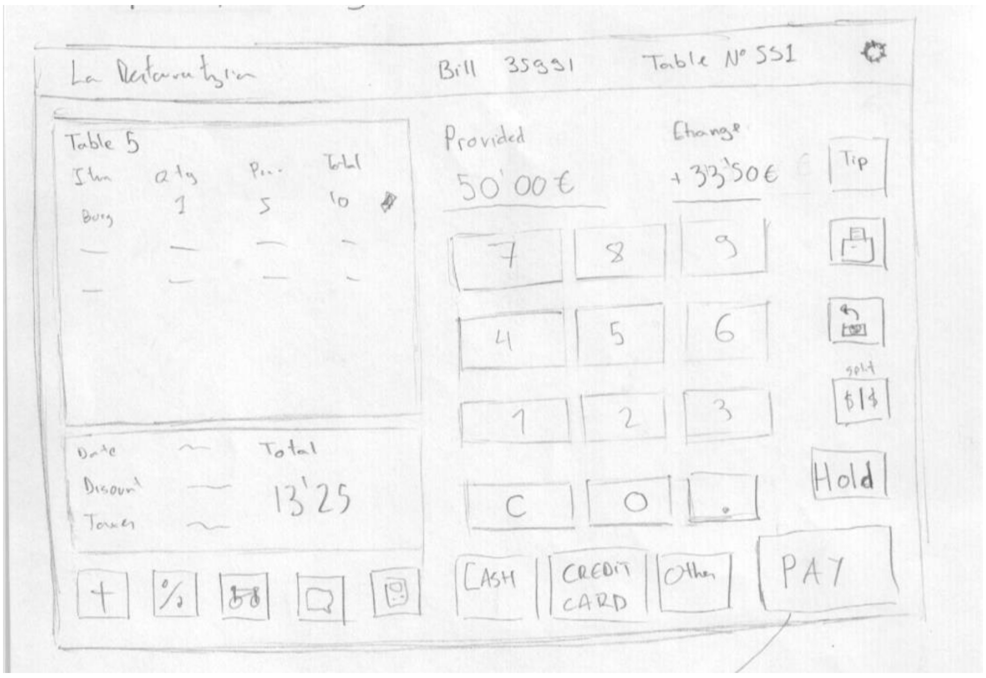


Figure 25 - Wireframe 3

6.4.1.3 *Wait Staff and Open Menu*

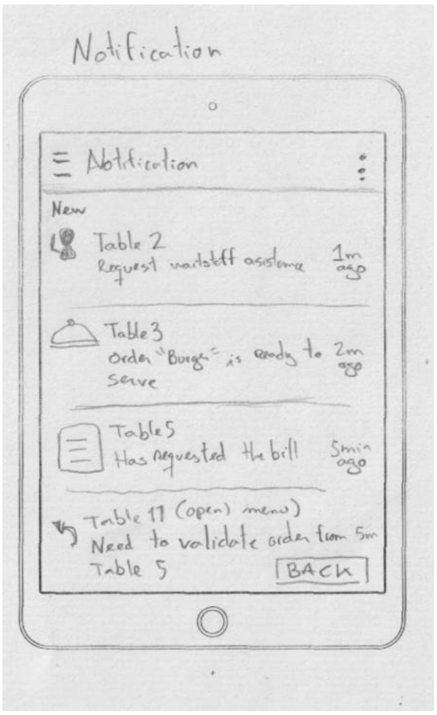


Figure 26 - Wireframe 4

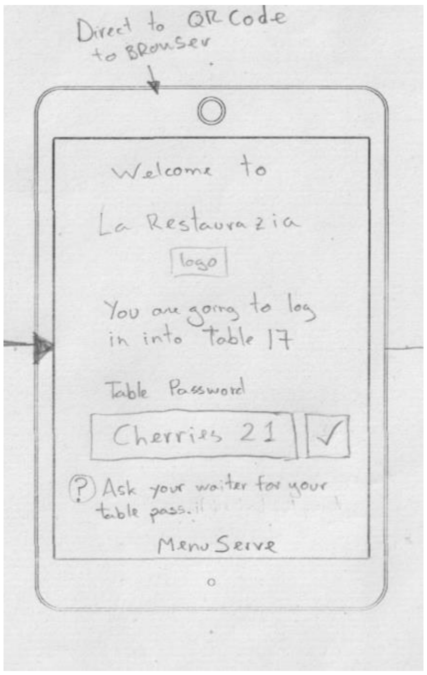

Direct to QR Code to Browser
Welcome to
La Restaurazia
logo
You are going to log in into table 17
Table Password
Cherries 21
✓
Ask your waiter for your table pass.
Menu Serve

Figure 27 - Wireframe 5

6.5 High quality designs

The logical path after creating the wireframes of an application is to use the generated documentation in order to create high quality design so the different screens to obtain a view of how may the final product look one is ready. This process helps identify the most common design patterns, validate the concept and to settle the colour palette and design language selected for the project.

The process of creating high quality designs is complicated and time consuming, especially in early stages where the main building blocks has not yet been created. This building blocks can be elements like input forms, menus, buttons and so on.

But in order to create a prototype of MenuServe, this tasks of generating high quality designs are mandatory. Although, a prototype is intended to provide an idea of how the application works and its main workflows, so there is no need to create designs for every page on the MenuServe suit. This process of designing a big amount of screens allows to generate all the visual information required to extrapolate to pages that has not been prototyped yet.

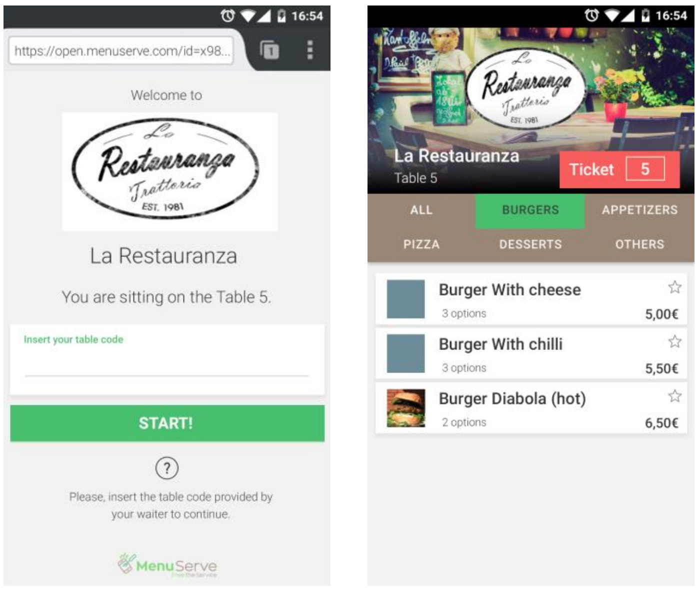
In total, to generate the full prototype that goes through the 5 modules of MenuServe, there has been rendered 81 high quality images of pages of the applications, enough to settle the design and how the application will look like.

Thankfully, internet is full of resources and that is a great help in order to create designs. For the initial task of creating the main building blocks, the help of UI kits has been used. In total, only one of this UI kits have been used as a base for starting with the design, although in a very small scale because the specifications for MenuServe where not matching in any generic UI kit.

The UI kit used is Ultra Free Admin Dashboard [21] UI by Mushfiqul Islam, it is distributed free under no license and it has mainly provided the proportions of elements and the base shadow effects for the Material Design.

For the icons, it has been used Icons8, a repository of more than 28.000 icons. The icons are free to use with an attribution license if it is a document or by linking the webpage on your application and/or website.

The designs have been made on Adobe Photoshop CC, and each screen is a different document. The main building blocks are vectors, both squares and lines, and typography. There is also bitmaps for icons and images.



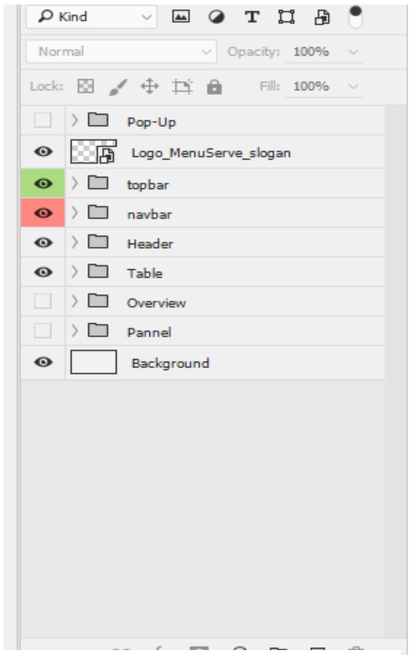
You are sitting on the Table 5.

Please, insert the table code provided by your waiter to continue.

Burger With cheese	3 options	5,00€
Burger With chilli	3 options	5,50€
Burger Diabola (hot)	2 options	6,50€

Figure 28 – High quality designs of OpenMenu module

The figure 28 shows an example of a high quality design, to be concrete, the access screen and the Ordering screen of Open Menu module.



Normal Opacity: 100% Fill: 100%

Figure 29 – Layer panel

Each Photoshop document has every object ordered in folders and layers, allowing the designer to activate or inactivate items. For example, pop-ups and notification screens are usually included in the same screen file that makes this modal panel to jump, and it is hided or showed depending on which image is going to be rendered.

Figure 29 shows an example of this layer configuration in a Photoshop document.

Photoshop also provides flexibility in order to edit the texts one they are situated on screen, so it is easy to change information very quickly. Moreover, it provides a full suite of options to align grids, guides, and duplicate objects to create patterns and other design requirements.

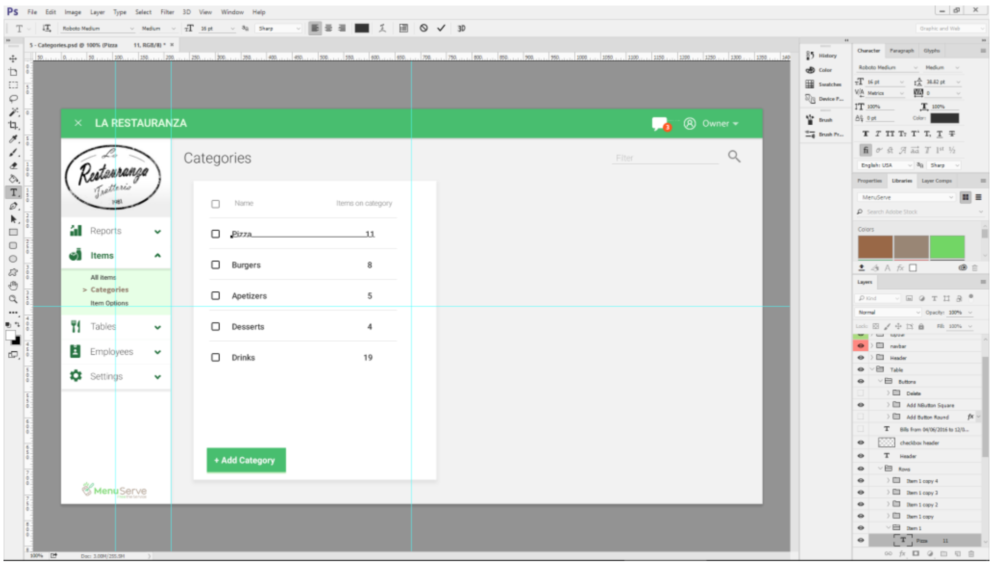


Figure 30 - Edition screen in Photoshop 1

The figure 30 shows a screen capture of a Photoshop CC while editing a page of the Administration module of MenuServe. We can appreciate the layer tree on the bottom right of the screen. We can also appreciate the colour palette shown on the right panel and the grid lines (in blue) that allows to align the different objects.

As said before, each page of the High Quality prototypes has been created following other screens, so the design patterns and building blocks that has been created on the initial stages of the high quality designs acts as a base for all the screens.

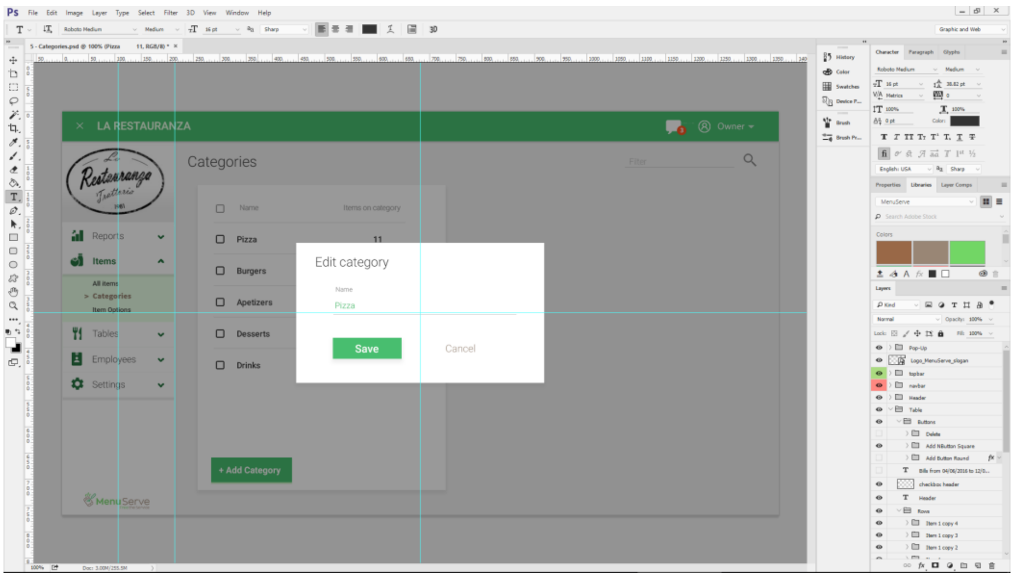


Figure 31 – Edition screen in Photoshop 2

The figure 31 shows the same screen as the figure 30 but with the pop-up layer active.

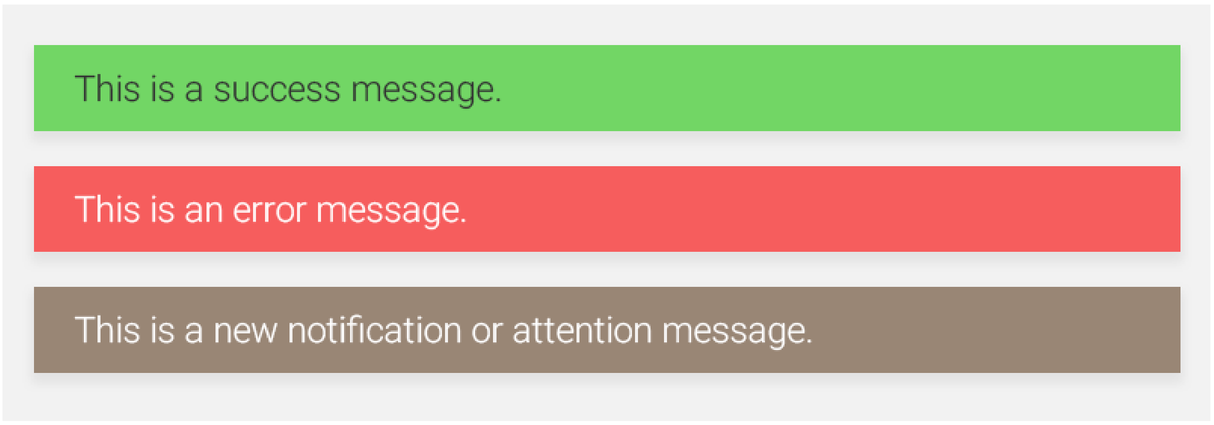
6.5.1 Design patterns

There are some designs patterns that are recurrent and widely used. While performing the high quality designs, this design patterns are used as building blocks for other screens in process.

Some other design patterns than defines the belabour of an item in determined circumstances and are more difficult to show when performing a prototype, like for example the behaviour of a form when there is a wrong input field. In this cases, the design pattern is usually designed once and not applied to all the screens that may use it. This allows to explain the behaviour of some situations that may not always happens and, as a consequence, will not need to be added in every screen.

We may find some examples below:

6.5.1.1 *Notification messages*


This is a success message.

This is an error message.

This is a new notification or attention message.

Figure 32 – Notification messages

When performing an action in the application, as form example creating a new item on the Back Office, the user needs to receive feedback from its actions. In this situation, after the administrator has filled the forms and submitted the information, the system will return a message providing him with a feedback of how the operation went.

When creating an item and saving it, the messages could be:

Success:	“The item has been created successfully”.
Error:	“There has been an error creating the item. Please try again”.
Notification:	“The item you created is not set as active. Users will not see it until its enabled”

6.5.1.2 *Form Input*

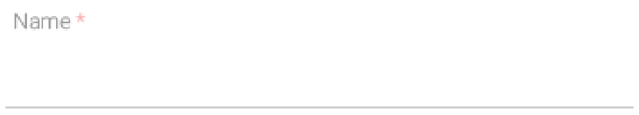
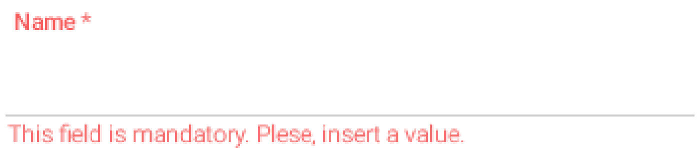


Figure 33 – Input field

Forms are tricky, especially when there are mandatory fields that needs to fulfil a series of criteria. That is why is important to set how the form will acknowledge the user that something is missing. For MenuServe, the field will get red and a message of the error will appear under the input.

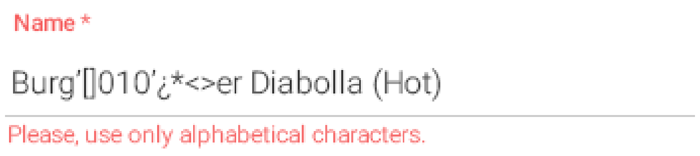
If the field is mandatory and has not been filled:



This field is mandatory. Plese, insert a value.

Figure 34 – Mandatory field warning

If the field has a specific restriction that has not been fulfilled:



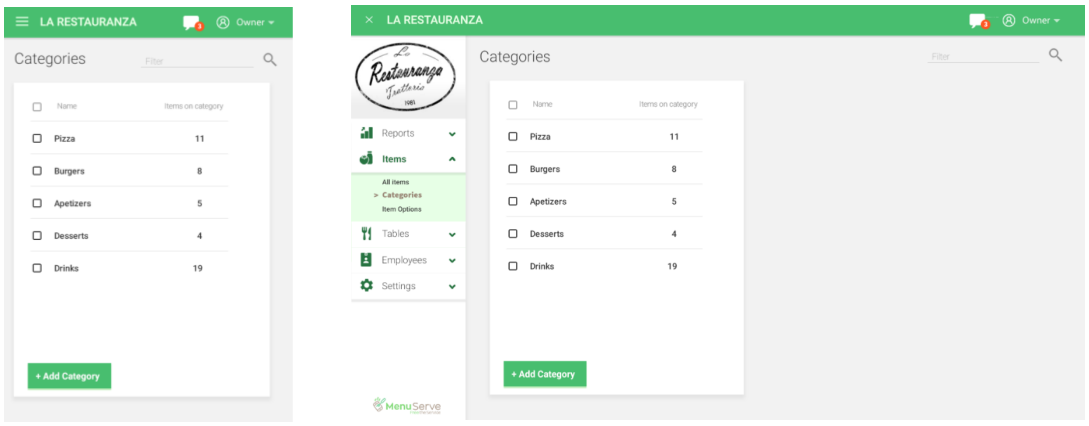
Burg'[]010'¿*<>er Diabolla (Hot)

Please, use only alphabetical characters.

Figure 35 – Bad format warning

6.5.1.3 *Responsive designs*

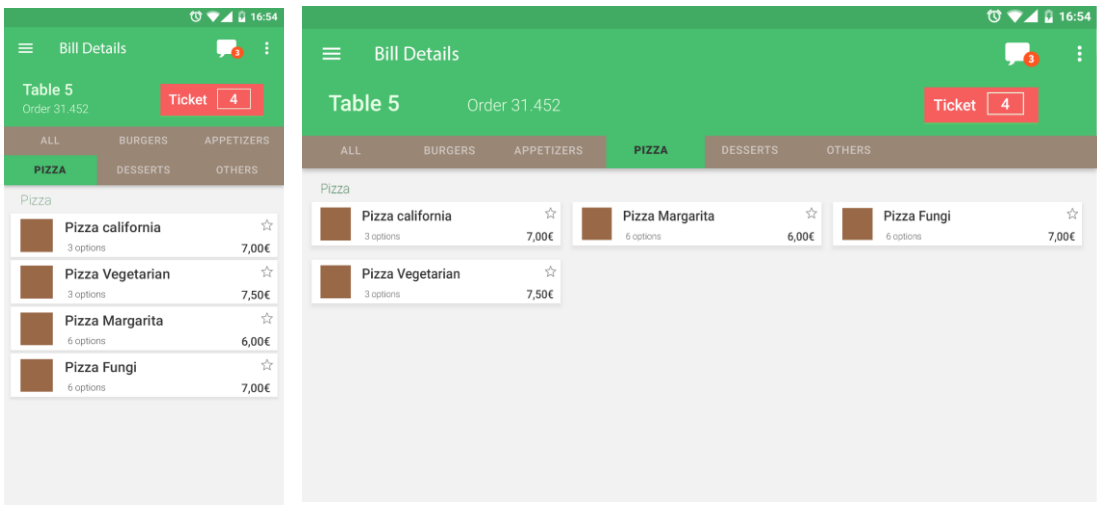
MenuServe has been designed to be fully responsive to fit to one of its main requirements and key selling points, being usable from almost any device. Thanks to Material Design though, making a responsive design is not difficult as this design language has been designed from its base with this concept in mind.



Name	Items on category
Pizza	11
Burgers	8
Apetizers	5
Desserts	4
Drinks	19

Figure 36 – Responsive design (web)

The figure 36 shows an example of how the administrator website adapts to different screen sizes. To do so, the menu disappears on small devices, and can be activating by pressing the left top of the screen, as a mobile application.



Pizza california	3 options	7,00€
Pizza Vegetarian	3 options	7,50€
Pizza Margarita	6 options	6,00€
Pizza Fungi	6 options	7,00€

Figure 37 – Adaptive design (application)

The figure 37 shows another example of responsive design. For this case, both are screenshots of the Wait Staff device application, one in a smartphone and the other in a tablet.

6.5.1.4 *Accessibility*

All the designs of MenuServe have been designed following the Web Content Accessibility Guidelines (WCAG), on its AA level. This convention specifies the best practices and requirements to allow a proper accessibility of the application.

Following this guideline allows making the application more accessible, until a point, to people with difficulties on sight or movement, that has the same right to work and use the advantages of MenuServe than everybody else.

The base guidelines [22] are:

Perceivable

Information and user interface components must be presentable to users in ways they can perceive.

- **Guideline 1.1**: Provide text alternatives for any non-text content so that it can be changed into other forms people need, such as large print, braille, speech, symbols or simpler language.
- **Guideline 1.2**: Time-based media: Provide alternatives for time-based media.
- **Guideline 1.3**: Create content that can be presented in different ways (for example simpler layout) without losing information or structure.
- **Guideline 1.4**: Make it easier for users to see and hear content including separating foreground from background.

Operable

User interface components and navigation must be operable.

- **Guideline 2.1**: Make all functionality available from a keyboard.
- **Guideline 2.2:** Provide users enough time to read and use content.
- **Guideline 2.3**: Do not design content in a way that is known to cause seizures.
- **Guideline 2.4**: Provide ways to help users navigate, find content, and determine where they are.

Understandable

Information and the operation of user interface must be understandable.

- **Guideline 3.1**: Make text content readable and understandable.
- **Guideline 3.2**: Make web pages appear and operate in predictable ways.
- **Guideline 3.3:** Help users avoid and correct mistakes.

Robust

Content must be robust enough that it can be interpreted reliably by a wide variety of user agents, including assistive technologies.

- **Guideline 4.1:** Maximize compatibility with current and future user agents, including assistive technologies.

7 Prototype

After the definition of the different modules, its navigation tree, the extensive wireframe and the creation of the high quality designs, the last step is gathering all this information that has been generated and creating a prototype. According Wikipedia, a prototype is “an early sample, model, or release of a product built to test a concept or process or to act as a thing to be replicated or learned from It”. And this is the main goal intended to achieve in this stage of the project.

This prototype, then, is intended to offer an accurate view of how MenuServe will work in case it is developed, and to provide better understanding on how it will look, it will behave, and which are the main workflows of this RMS. Because the functionalities exposed in the prototype are just a predefined workflow and does not allow to interact outside this specified workflow, the correct term to call this exercise is a Visual Prototype.

As OpenMenu has 5 modules, there has been multiple separated prototype modules, that can be executed separately, but that altogether completes the most common use cases that can be found. The prototype modules or sections that has been created are the following ones.

1. Admin
2. Wait Staff Device
3. Kitchen
4. POS
5. Open Menu
6. Wait Staff Device with Open Menu

The prototype sections have been created with the high definition images that has been generated during the last stage of the UI design. Each of the screens has been personalized to fit in a user case, allowing the sequence of images to create the workflows designed.

To assemble the prototypes and providing them with interaction and continuity, JustInMind has been used. JustInMind is a free to use wireframe and prototyping software with a wide range of possibilities.

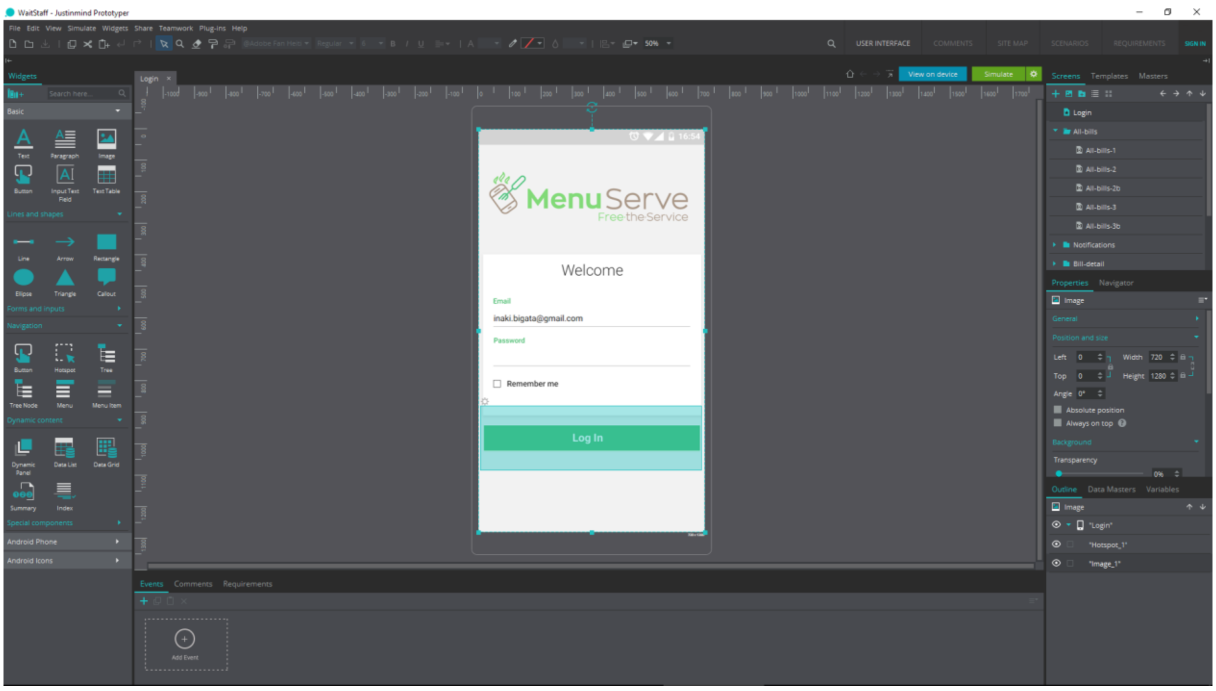


Figure 38 – JustInMind main screen

In the case of MenuServe, the main purpose of JustInMind it has been to join the different screens generated via Photoshop and provide them with interactive parts in order to create a prototype. The figure 38 shows an actual screenshot of the prototyping of MenuServe with JustInMind.

The prototypes have been exported to HTML and can be reproduced by any modern browser. This files can be find in the delivered files alongside the project. To run them, it will be needed to enter the folder of every module and execute “index.html”.

The following subsections will introduce three use cases, and provide a walkthrough of those prototypes. At any time of this walkthrough, the module of the prototype used will be written.

All three use cases are set on a fictional restaurant called “La Restauranza”, serving mainly burgers and pizza. Any similarities with reality are purely a coincidence.

A video is also attached on the deliverables of the project, widely explaining each flow.

Note: If stuck or not knowing how to proceed, by double-clicking on the prototype screen, the interactive (clickable) parts will highlight.

Note 2: You may relate each prototype module with the title of each of the following subsections by the number in parentheses following the title name.

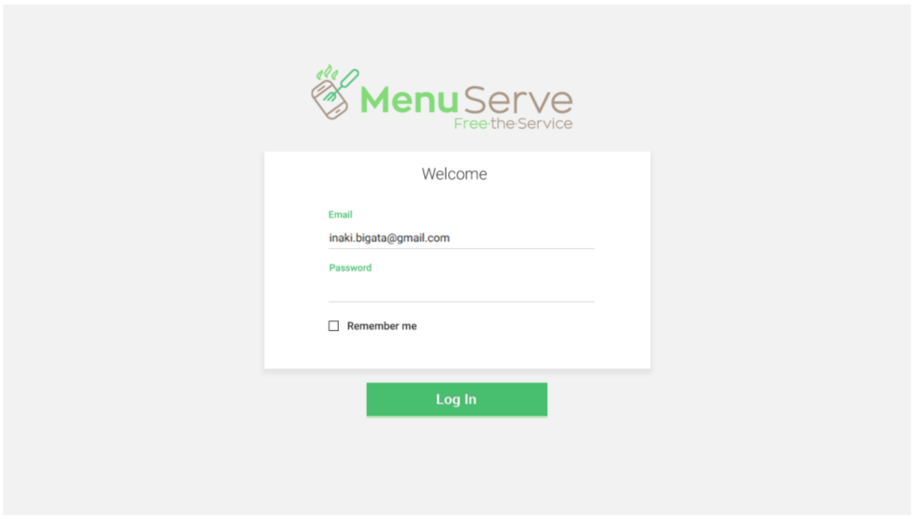
7.1 Use case 1: Setting up a Restaurant

As a restaurant owner, the user will require to configure and customize the restaurant information and all its context so their employees may use MenuServe effectively. This first use case shows how an owner of a fictional restaurant can fulfil this task of creating items, adding employees or changing the settings of the restaurant.

In that case, the restaurant has already been created, and the owner logs in to perform some changes to the system and add a new employee. This use case offers an excellent base to show the easiness and intuitiveness of the system.

7.1.1 Admin Prototype Module (1)

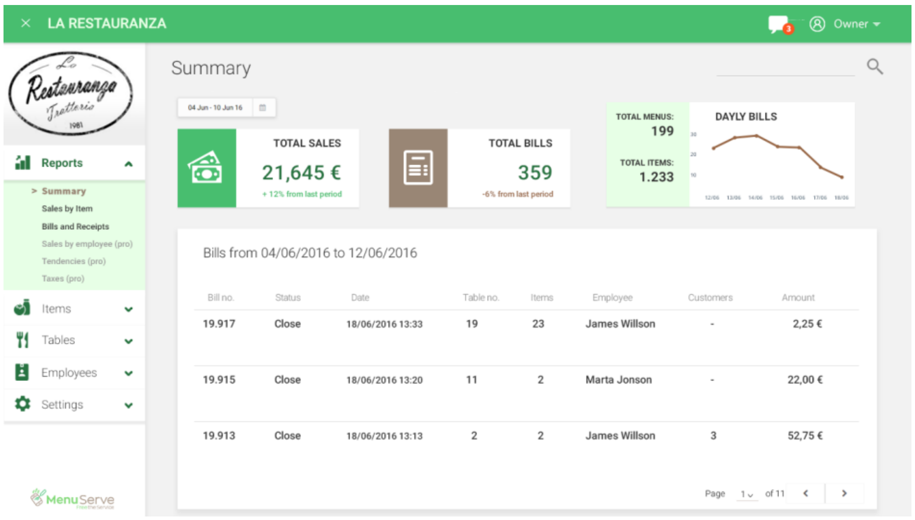
7.1.1.1 ***Log in***



inaki.bigata@gmail.com

The first thing to be done when accessing MenuServe is to log in into the system. By introducing the username and password, MenuServe will automatically log the user in its own restaurant.

7.1.1.2 *Reporting*



Bills from 04/06/2016 to 12/06/2016

Bill no.	Status	Date	Table no.	Items	Employee	Customers	Amount
19.917	Close	18/06/2016 13:33	19	23	James Willson	-	2,25 €
19.915	Close	18/06/2016 13:20	11	2	Marta Jonson	-	22,00 €
19.913	Close	18/06/2016 13:13	2	2	James Willson	3	52,75 €

Page 1 of 11

Once logged in, the system will redirect the user to the summary page, where it will offer a quick view of how is the restaurant going on, with total sales, total bills, a chart and the last receipts. The period in which this info is displayed could also be adapted.

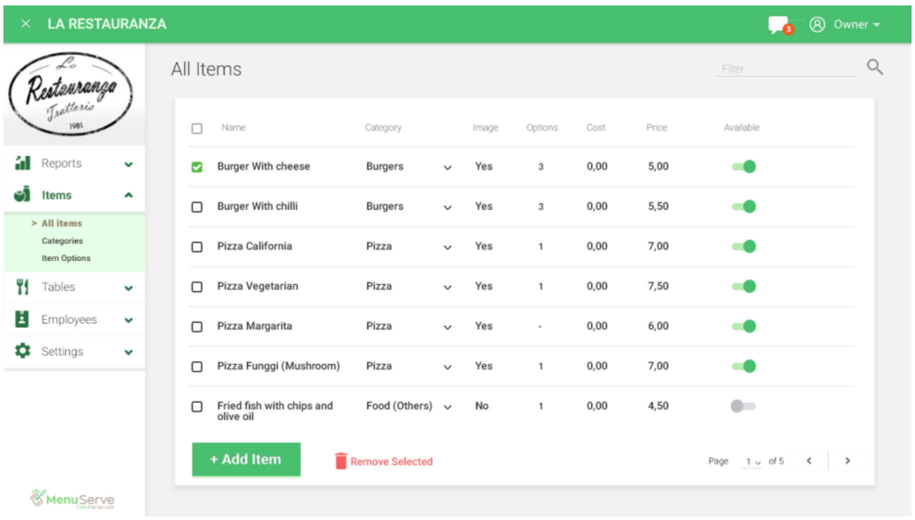
In here, the left menu can be seen with the logotype of the restaurant a quick access to all the sections that this Back Office has.

The prototype continues by clicking on Items.

7.1.1.3 *Items*

This section of the Back Office will display a list of all the available items created on this restaurant. This is a CRUD page (create, review, update, delete), and each row of the list is an already existing item.

Although is not available on the prototype, on all CRUD pages of the system clicking a row will open an edit page for the item. By selecting the item, a delete button will appear, and a new item can be created by clicking on the “Add item” button.



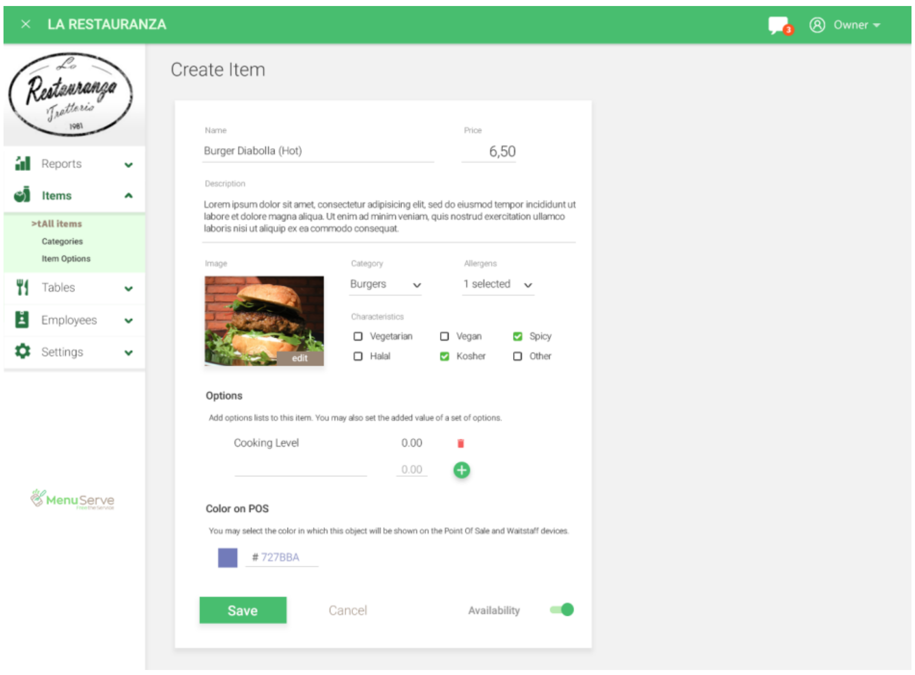
All Items

☐	Name	Category	Image	Options	Cost	Price	Available
✓	Burger With cheese	Burgers	Yes	3	0,00	5,00	●
☐	Burger With chilli	Burgers	Yes	3	0,00	5,50	●
☐	Pizza California	Pizza	Yes	1	0,00	7,00	●
☐	Pizza Vegetarian	Pizza	Yes	1	0,00	7,50	●
☐	Pizza Margarita	Pizza	Yes	-	0,00	6,00	●
☐	Pizza Funggi (Mushroom)	Pizza	Yes	1	0,00	7,00	●
☐	Fried fish with chips and olive oil	Food (Others)	No	1	0,00	4,50	○

+ Add Item   Remove Selected   Page 1 of 5

The process on the prototype for its specific page is selecting all, selecting one item, clicking on delete, cancelling the delete, and adding an item.

7.1.1.4 *Add an item*



Create Item

Name: Burger Diabolla (Hot)
Price: 6,50

Description: Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat.

Image
Category: Burgers
Allergens: 1 selected

Characteristics: ☐ Vegetarian ☐ Vegan ✓ Spicy ☐ Halal ✓ Kosher ☐ Other

Options
Add options lists to this item. You may also set the added value of a set of options.
Cooking Level 0.00
0.00

Color on POS
You may select the color in which this object will be shown on the Point Of Sale and Waitstaff devices.
727BBA

Save   Cancel   Availability

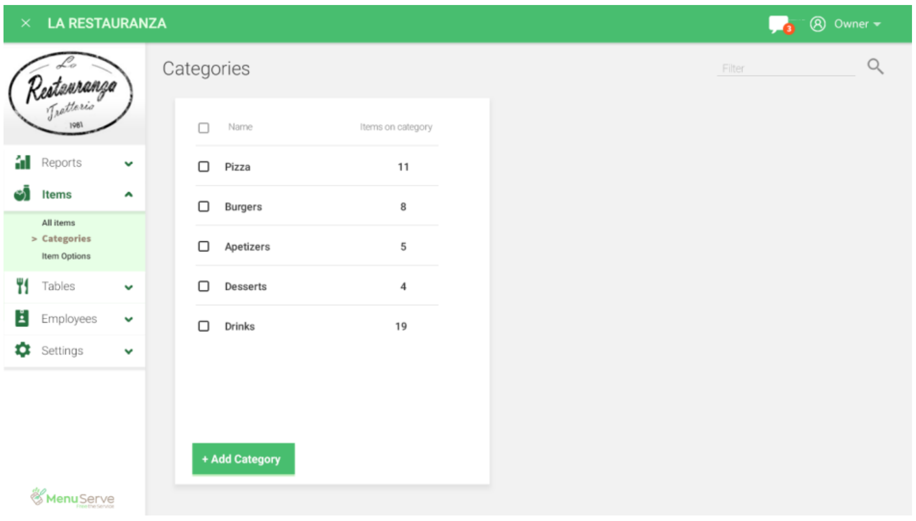
The add item page, where all the information of the item can be added and into the system. Every information is important to be entered either to allow a fast categorization on MenuServe or in order to keep the health standards.

The category item will allow MenuServe to categorize the item on the POS and Wait Staff device later on. The allergens and characteristics of the item allow the customers to exclude the food they cannot or do not want to eat.

By last, there is the options to be added. Options are a different entity that is created in the back office, and allows deep personalization of the items by adding logical items that later one the Waiter or the customer will be able to choose to personalize the item. This would allow the owner of the restaurant to create, for example, menus in an early stage of MenuServe where the daily menu system would not be still deployed.

When clicking on save, the item is added.

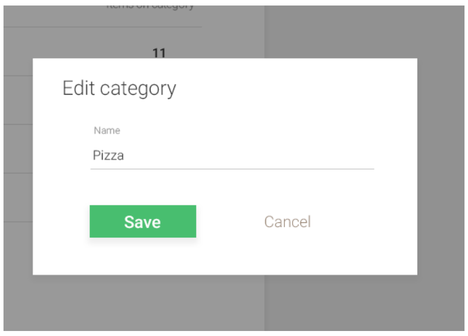
7.1.1.5 ***Categories***



Categories

Name	Items on category
Pizza	11
Burgers	8
Apetizers	5
Desserts	4
Drinks	19

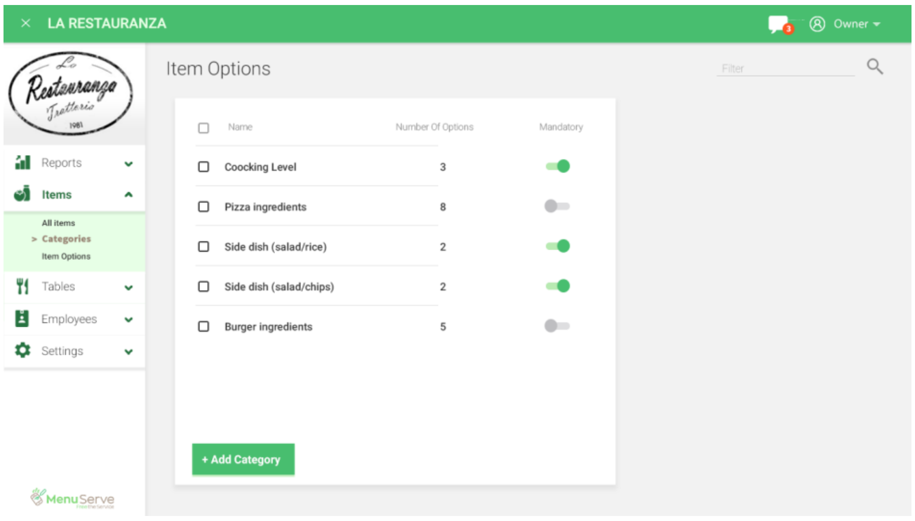
Following the same concept of CRUD, categories can be added, deleted or edited.



Edit category

Name: Pizza

7.1.1.6 *Item Options*

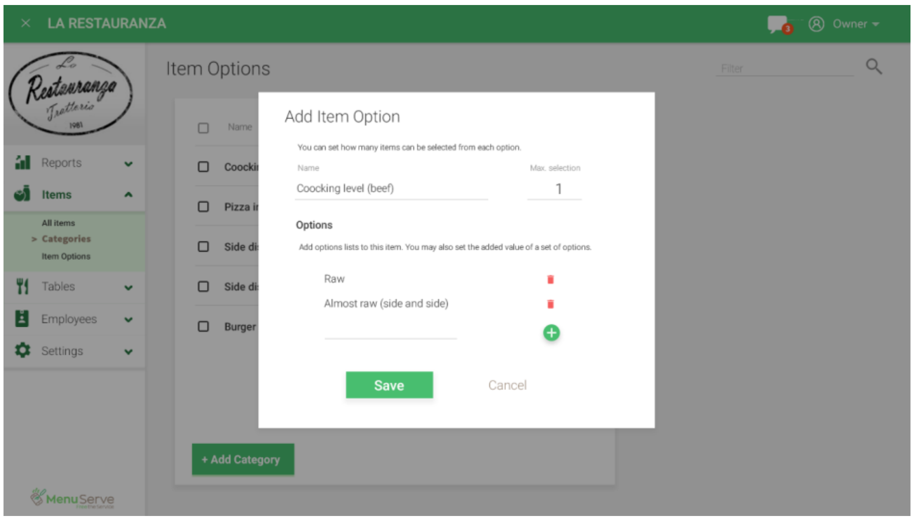


Item Options

Name	Number Of Options	Mandatory
Coocking Level	3	●
Pizza ingredients	8	○
Side dish (salad/rice)	2	●
Side dish (salad/chips)	2	●
Burger ingredients	5	○

Item options, as mentioned below, can be also edited from the Back Office. Each option can be the added to as many menus as desired, and offers a higher level of personalization. From the table itself, some option may be edited (mandatory).

Selectable items of an Option can be set and they can be defined as mandatory or not, to expand the customization. In example, a daily menu may have all items mandatory.



Add Item Option

You can set how many items can be selected from each option.

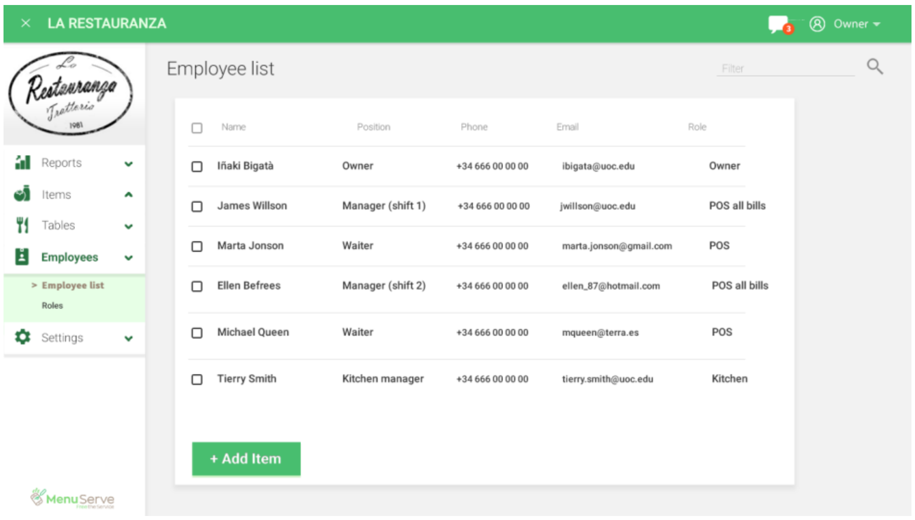
Name: Coocking level (beef)
Max. selection: 1

Options

Add options lists to this item. You may also set the added value of a set of options.

Raw
Almost raw (side and side)

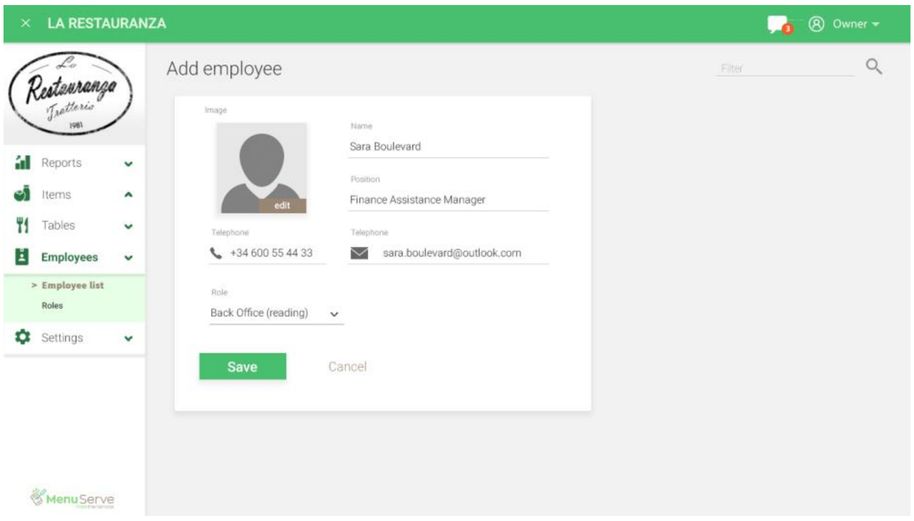
7.1.1.7 *Employees*



Employee list

Name	Position	Phone	Email	Role
Iñaki Bigatà	Owner	+34 666 00 00 00	ibigata@uoc.edu	Owner
James Willson	Manager (shift 1)	+34 666 00 00 00	jwillson@uoc.edu	POS all bills
Marta Jonson	Waiter	+34 666 00 00 00	marta.jonson@gmail.com	POS
Ellen Befrees	Manager (shift 2)	+34 666 00 00 00	ellen_87@hotmail.com	POS all bills
Michael Queen	Waiter	+34 666 00 00 00	mqueen@terra.es	POS
Tierry Smith	Kitchen manager	+34 666 00 00 00	tierry.smith@uoc.edu	Kitchen

As items, employees can be added to the system. Each employee will mean a new user onto the system, that may or may not have permission rights. When creating this employee, the role of it can be selected, providing them with more or less privileges depending its position or main duties.



Add employee

Image

Name: Sara Boulevard

Position: Finance Assistance Manager

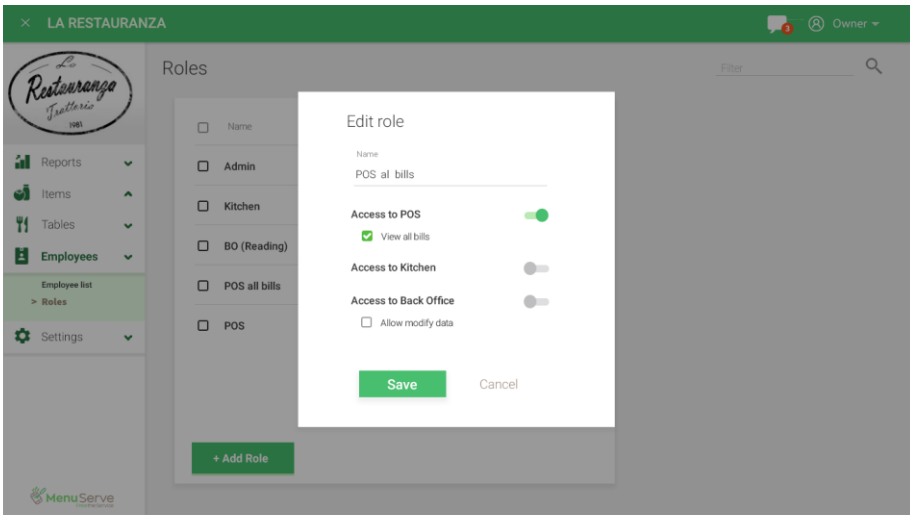
Telephone: +34 600 55 44 33

Telephone: sara.boulevard@outlook.com

Role: Back Office (reading)

The add/edit page of the employee also allows to add an image, and edit its contact information in case the owner needs to contact them.

7.1.1.8 ***Roles***



Edit role

Name: POS al bills

Access to POS: on
[✓] View all bills

Access to Kitchen: off

Access to Back Office: off
[ ] Allow modify data

Save | Cancel

Roles play an important part of MenuServe, as there is an active security measure. The owner will be able to view, add and edit the roles of the system, personalizing them to the requisites of the restaurant.

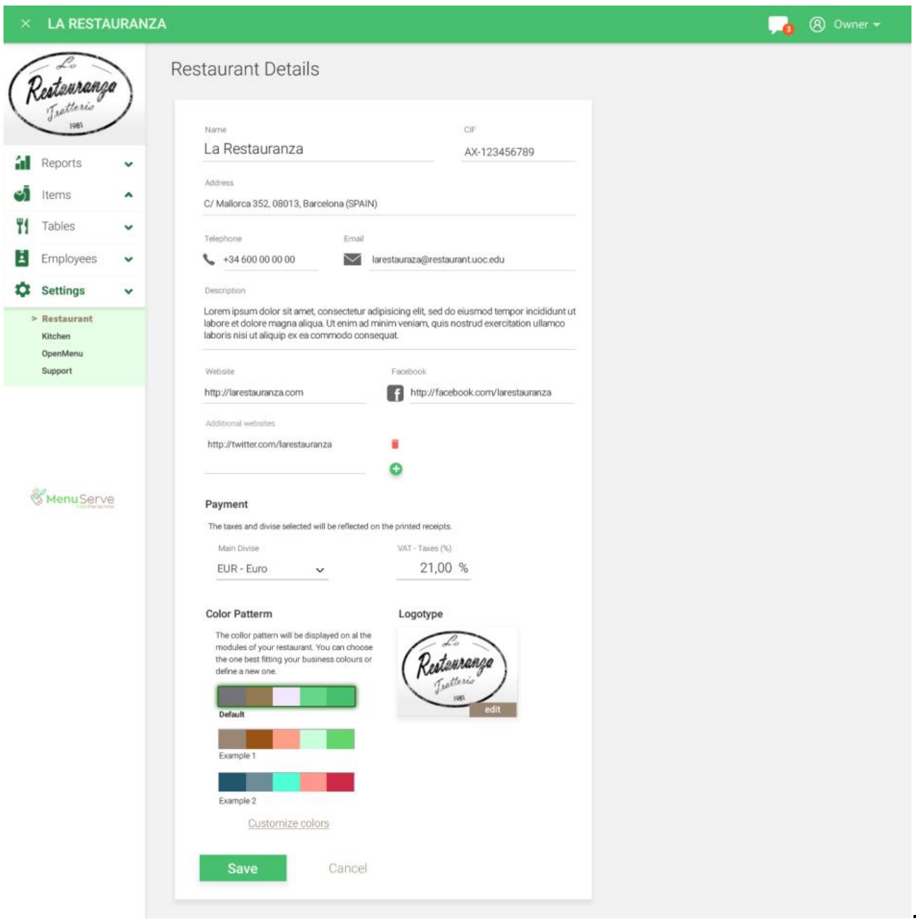
For example, Waiters can be limited to the POS (Point of Sale + Wait Staff), and can be either limited to their bills or may have access to all the bills of the restaurant with their own device. In the physical POS, all bills will always be active.

This section, although, will only be available on the payed versions, as the regular version will only offer roles by default, limiting its flexibility.

7.1.1.9 ***Restaurant Settings***

From this screen, the information relating to the restaurant itself can be set up and edited. Between this information, we can find all the related public information as it can be the name of the restaurant, its description, the contact information, address, the website, and the social media links.

The social media links will allow the extensive reports if the restaurant owns a payed license



Restaurant Details

Name: La Restauranza
CIF: AX-123456789
Address: C/ Mallorca 352, 08013, Barcelona (SPAIN)
Telephone: +34 600 00 00 00
Email: larestauraza@restaurant.uoc.edu
Description: Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat.
Website: http://larestauranza.com
Facebook: http://facebook.com/larestauranza
Additional websites: http://twitter.com/larestauranza

Payment

The taxes and divise selected will be reflected on the printed receipts.

Main Divise: EUR - Euro
VAT - Taxes (%): 21,00 %

Color Patterm

The collor pattern will be displayed on al the modules of your restaurant. You can choose the one best fitting your business colours or define a new one.

Default
Example 1
Example 2
Customize colors

Logotype

edit

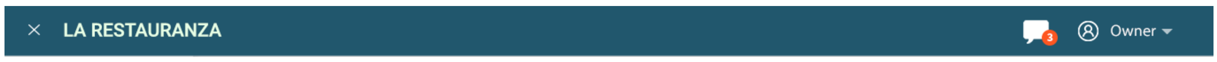
Save Cancel

.

This section also contains internal details of the restaurant, as it can be the main currency that will be used or the percentage of taxes (VAT) that may be required when ticketing.

Other important details in this settings are that the restaurant can be customized, so the owner of the restaurant will be able to adapt both the logotype and the colour scheme of the restaurant to adapt to its corporate image. Changing the colour scheme will automatically change the display colour in all the modules of MenuServe.

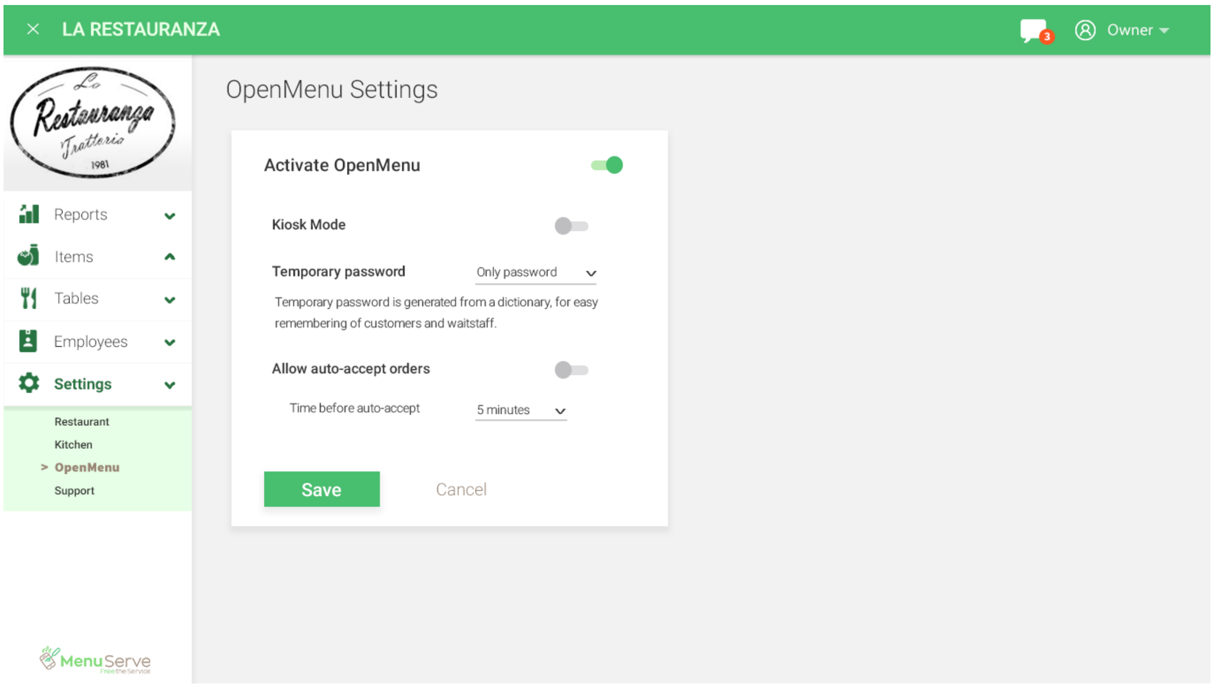
An example of the header in blue colour:



7.1.1.10 *Open Menu Settings*

The last part of this use case is the configuration of the OpenMenu. From this page, all the functionality of Open Menu can be enabled or disabled.

If enabled, there are complementary options that may me set up, like kiosk mode, auto accept orders or the security of the OpenMenu access by the customers of the restaurant.



OpenMenu Settings

Activate OpenMenu

Kiosk Mode

Temporary password: Only password

Temporary password is generated from a dictionary, for easy remembering of customers and waitstaff.

Allow auto-accept orders

Time before auto-accept: 5 minutes

Save Cancel

7.2 Use case 2: Waiter attending a table

MenuServe is a tool focused in improving the workflows and efficiency of restaurants, is because of this reason that MenuServe needs to excel as an ordering system for restaurants. This means, that a waiter must be able to attend the customers, note their orders with his/her device, send it to the kitchen and at the end, be able to charge the customer.

In this use case, the waiter takes an order from customers with his device and sends it to the kitchen and the kitchen module effectively shows the new order. Later on, the waiter approaches the POS to charge the customers by cash.

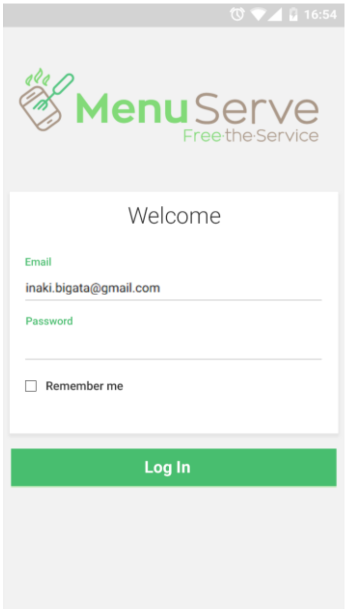
7.2.1 Wait Staff Module (1)

7.2.1.1 *Log in*

The first thing to be done when accessing MenuServe is to log in into the system. By introducing the username and password, MenuServe will automatically log the user in its own restaurant.

This log in will be made by each waiter with their own email and the password that they have set up.

When users are created on the back office, an email is sent to them in order to set up their passwords.



7.2.1.2 ***Active bills***

When logged in, the waiters will have a wide view of all the active tables. This way, they will easily change from one table to another quickly and effective.

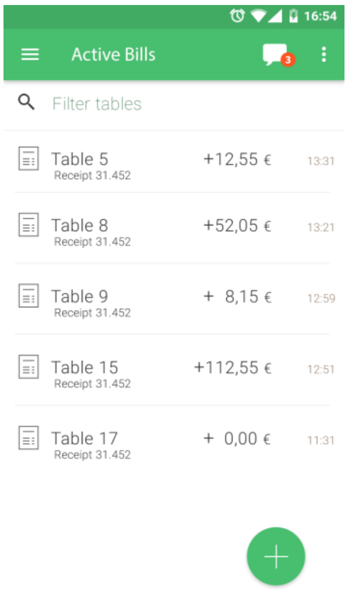

16:54
Active Bills
Filter tables

Table 5 Receipt 31.452	+12,55 €	13:31
Table 8 Receipt 31.452	+52,05 €	13:21
Table 9 Receipt 31.452	+ 8,15 €	12:59
Table 15 Receipt 31.452	+112,55 €	12:51
Table 17 Receipt 31.452	+ 0,00 €	11:31

In the bottom right part of the screen, and following Material Design baselines, there is the button to create a new bill. This will be used to create a new ordering bill when new customers are welcomed into their table.

There is an always present notification item in the top right of the screen, so if there is anything new that pops up, they can be aware of it. If there are unread notifications, it will show the number on screen.

.

7.2.1.3 ***Notifications***

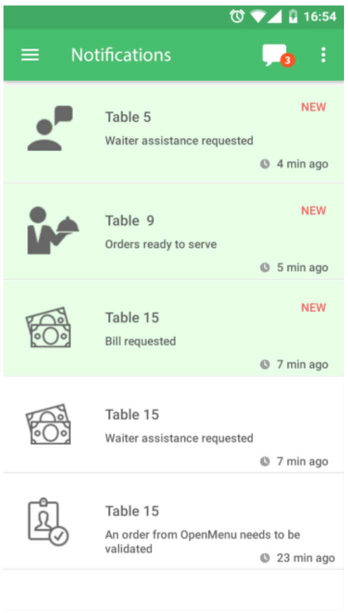

16:54
Notifications

Table 5 — Waiter assistance requested — NEW — 4 min ago
Table 9 — Orders ready to serve — NEW — 5 min ago
Table 15 — Bill requested — NEW — 7 min ago
Table 15 — Waiter assistance requested — 7 min ago
Table 15 — An order from OpenMenu needs to be validated — 23 min ago

The notification screen will show the last messages received by the system, and can inform of situations like assistance requested by a table, order ready to be served, bill requested or an OpenMenu order pending to be approved if the "auto-accept" is not enabled.

When clicked, notifications will provide you with a way of solving the issue.

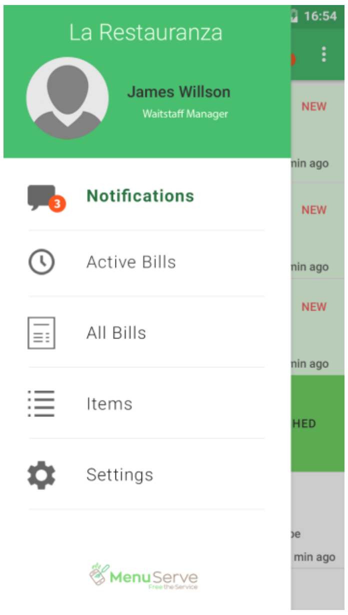
If a notification is swipe to the left, its options will appear, allowing them to dismiss it.

7.2.1.4 *Navigation*

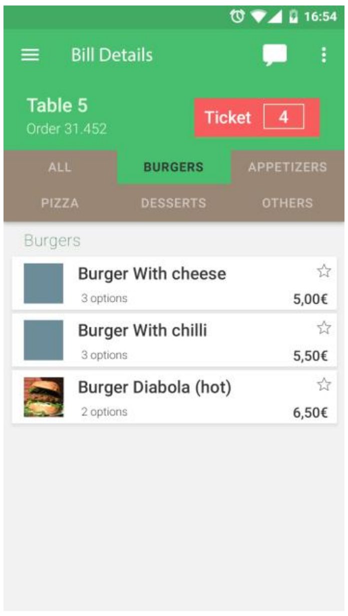
A part of the navigation by buttons, the Wait Staff device has a lateral menu that appears by pressing the top left part of the screen or by swiping from right to left of the screen.

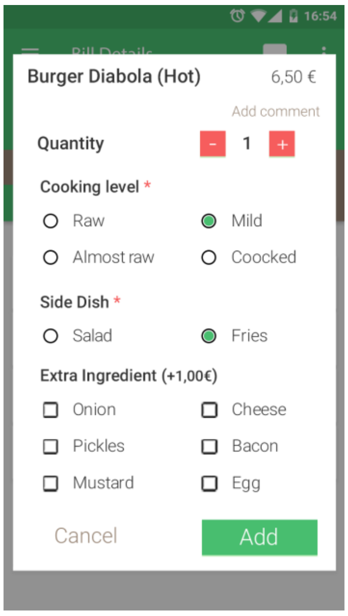
This menu provides information and allows to navigate to the other sections of the application.

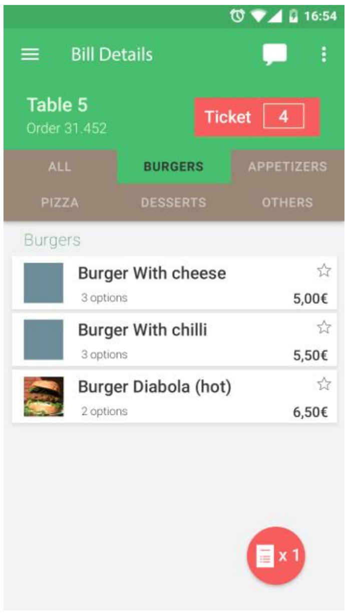
It also provides access to settings, where a Bluetooth printer may be configured, in example.



7.2.1.5 *Taking Orders*





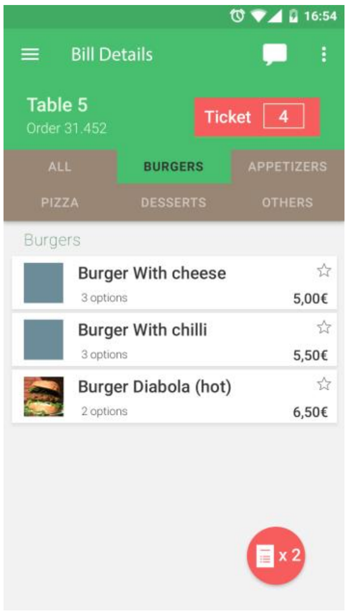


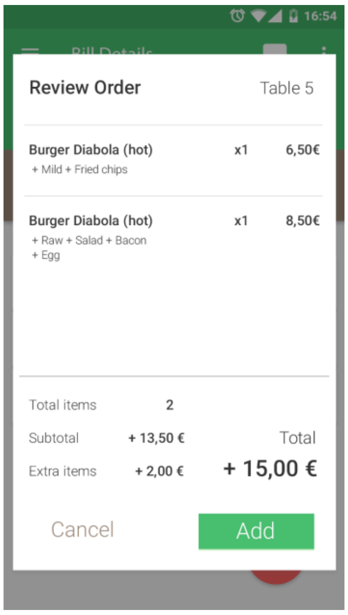
When accessing to a bill, the ordering screen will appear. This screen allows the waiters to navigate through all the items or to filter them by the categories created on the Back Office.

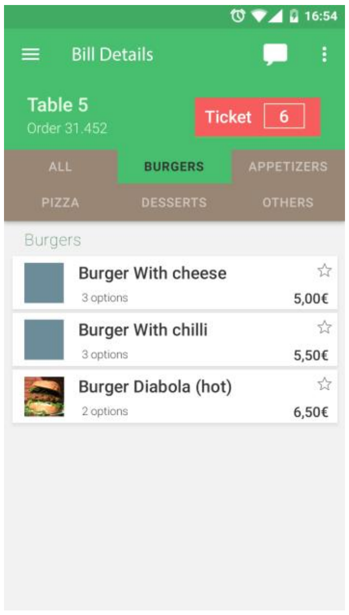
By pressing the three dot on the top of the screen, the additional options menu opens, allowing the waiter to search by name for example. The user may also mark items as favourite, and they will appear in a specific section, so for example if everybody orders something, it can be set as a fast item to select.

One an item is tap, a modal screen of options will open, allowing the waiter to specify its options, if any, it will also allow him to define the quantity of this product with that options. Once the item is added to the order, on the bottom right of the application will appear a “ticket” icon with a number on it, indicating how many items are on the order and waiting to be sent to the kitchen.

7.2.1.6 *Sending the Order to Kitchen*





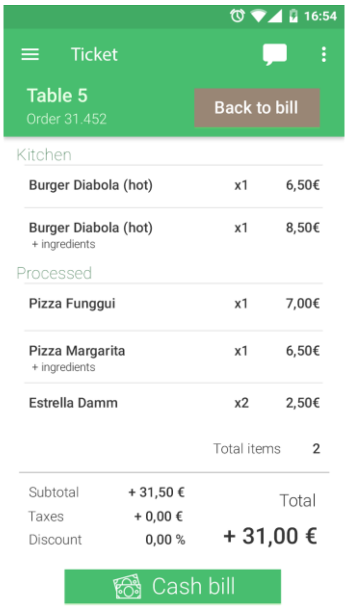


Once the items requested by the costumers has been added to the order by the waiter, he or she will be able to send this items to the kitchen. To do so, a waiter can type on the “order” button on the bottom right of the screen.

By taping on that button, a modal screen with the “Review order” will be displayed. The waiter can, then, validate the order, edit its items or send it to the kitchen.

Once it has been sent, the “order” button will disappear of the bottom right of the screen and the number of items marked there will be added on the “Ticket” button.

7.2.1.7 *Reviewing the ticket*


16:54
Ticket
Table 5
Order 31.452
Back to bill

Kitchen

Burger Diabola (hot)	x1	6,50€
Burger Diabola (hot) + ingredients	x1	8,50€

Processed

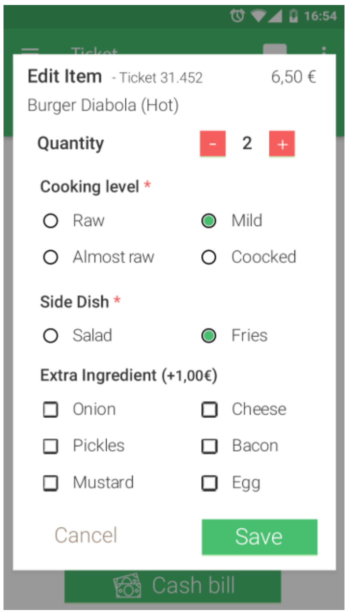
Pizza Funggui	x1	7,00€
Pizza Margarita + ingredients	x1	6,50€
Estrella Damm	x2	2,50€

Total items 2

Subtotal	+ 31,50 €
Taxes	+ 0,00 €
Discount	0,00 %

Total
+ 31,00 €

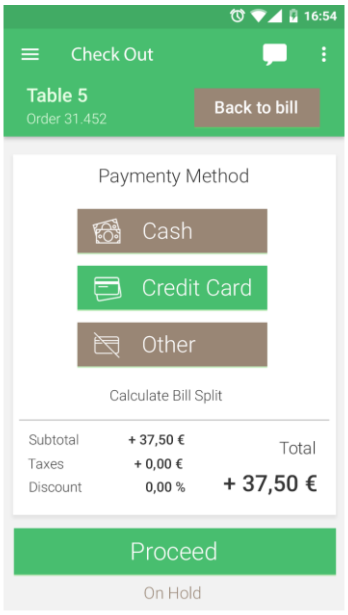
Cash bill


16:54
Edit Item - Ticket 31.452 6,50 €
Burger Diabola (Hot)
Quantity - 2 +
Cooking level *
Raw [✓ Mild]
Almost raw Coocked
Side Dish *
Salad [✓ Fries]
Extra Ingredient (+1,00€)
☐ Onion ☐ Cheese
☐ Pickles ☐ Bacon
☐ Mustard ☐ Egg
Cancel Save
Cash bill

By clicking on the ticket button on the “Bill details” screen, the system will go to the Ticket page. On this page, all the orders from that bill are displayed, together with its status. For example, the ordered items a minute ago are marked as kitchen, and the older ones, already served, are marked as processed. This screen also offers a simple representation of the bill itself, with the subtotals and total amount.

From this screen, items already existing on the bill can be modified. That way, if there has been an error, it can be modified before proceeding to charge the customer.

7.2.1.8 *Checkout from the Wait Staff device*



Waiters can initiate the process of checkout from their devices. This is especially useful when the customer is paying with credit card or via mobile application, when no change has to be provided. In those situations, if the bill is already printed, the waiter will not need to go to the POS.

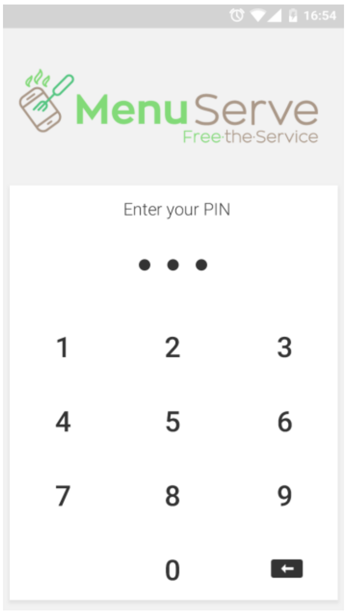
Notice that there is the chance to calculate the division of the bill by a specified number of customers paying together.

In the specific case of the prototype, the customers would like to pay in cash, so a visit to the POS will be mandatory. The waiter can leave the payment on hold or go back to bill to edit the ticket or add more items.

7.2.1.9 *Security of the Wait Staff Device*

MenuServe can be configured to offer better security for the Wait Staff devices. If this security is active, whenever the waiter blocks the device and unlocks it again, MenuServe will ask for a PIN code.

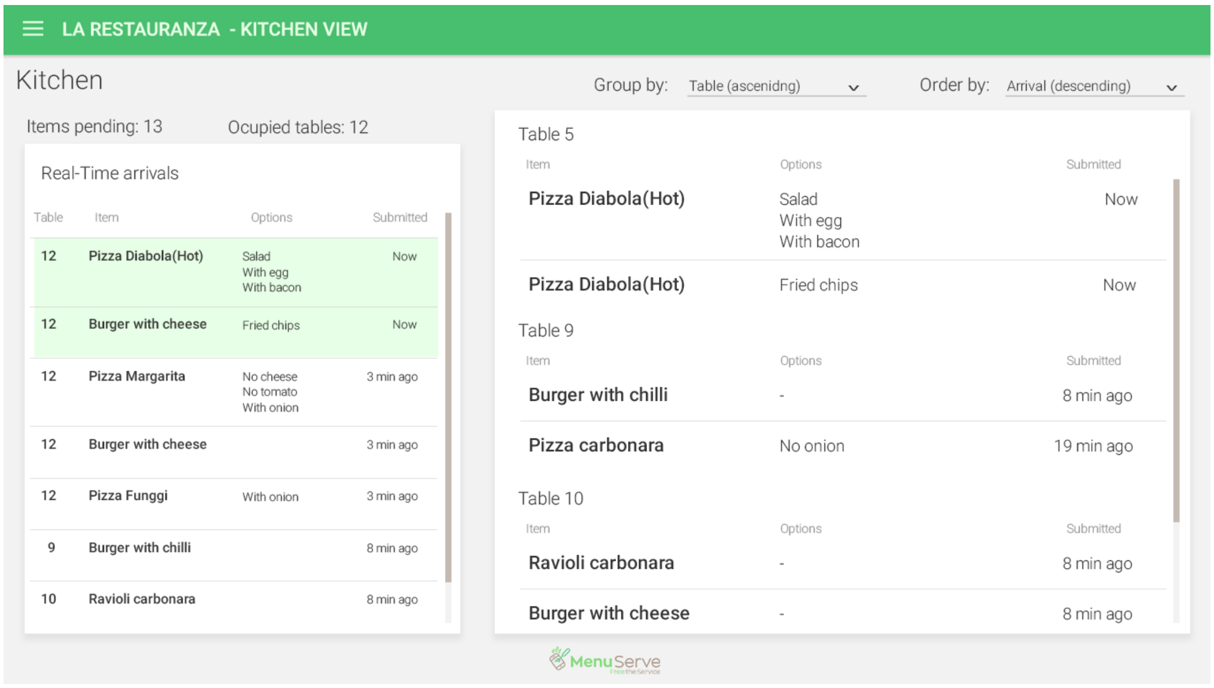
This way, if the device is stolen or left to the reach of third parties, they will not be able to mess with the system and lose information.



7.2.2 Kitchen Module (3)

This is the module that is displayed on the Kitchen of a restaurant, and it is intended to show display the incoming orders so the kitchen personal con start working on them as fastest and effectively as possible.

When there is a lot of information on screen, there is an automatic scroll that will display all the items, similar to the message board on banks or train stations, allowing the Kitchen staff to not miss any detail.



LA RESTAURANZA - KITCHEN VIEW

Kitchen

Group by: Table (ascenidng)
Order by: Arrival (descending)

Items pending: 13
Ocupied tables: 12

Real-Time arrivals

Table	Item	Options	Submitted
12	Pizza Diabola(Hot)	Salad With egg With bacon	Now
12	Burger with cheese	Fried chips	Now
12	Pizza Margarita	No cheese No tomato With onion	3 min ago
12	Burger with cheese		3 min ago
12	Pizza Funggi	With onion	3 min ago
9	Burger with chilli		8 min ago
10	Ravioli carbonara		8 min ago

Table 5

Item	Options	Submitted
Pizza Diabola(Hot)	Salad With egg With bacon	Now
Pizza Diabola(Hot)	Fried chips	Now

Table 9

Item	Options	Submitted
Burger with chilli	-	8 min ago
Pizza carbonara	No onion	19 min ago

Table 10

Item	Options	Submitted
Ravioli carbonara	-	8 min ago
Burger with cheese	-	8 min ago

The screen above shows the Kitchen module in action, and align to the use case that has been described. In this point of the workflow, the waiter has submitted the data and it is displayed on the incoming table on the left in a greenish background, to inform about the arrival of a new item. It can also be set up to display a notification sound for every new entry. In the case that an order was cancelled, this greenish background would change onto a red one, displaying the text cancelled.

This kitchen module has two tables, the small one on the left shows all the incoming items in arrival order, and it is used as an information table of all that is happening in real time.

The table on the right, in the other hand, offers a more customizable view more adaptable to the needs of each restaurant's kitchen. As it is intended to be the main table where kitchen staff will get the information, it is displayer in a bigger text, enough to be seen from far away when using a regular screen

(32"). Although it is responsive and can be displayed with an interactive tablet or any device with internet connectivity.

This second table, allows to readjust the grouping g of the items and the order itself. These two variables allow a big personalization of the display.

7.2.3 Point of Sale (4)

7.2.3.1 *Bill details*

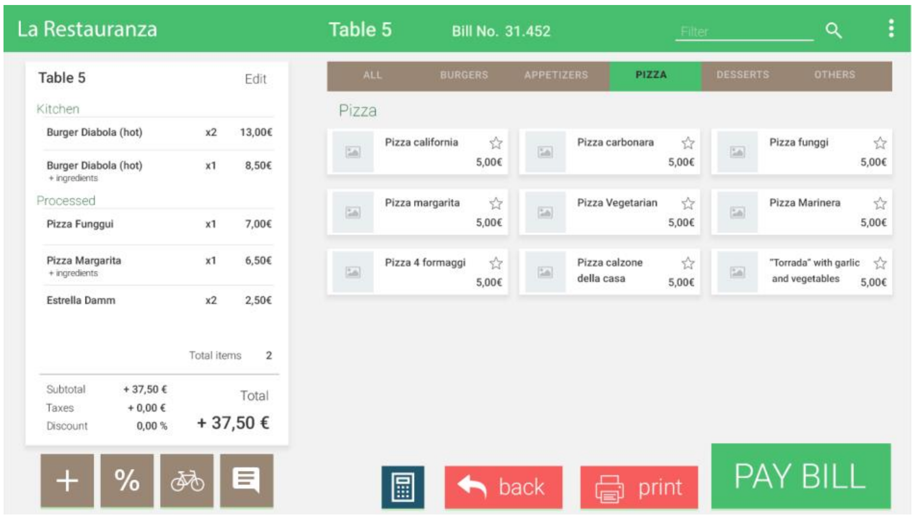

La Restauranza

Table 5 Edit

Kitchen		
Burger Diabola (hot)	x2	13,00€
Burger Diabola (hot) + ingredients	x1	8,50€
Processed		
Pizza Funggui	x1	7,00€
Pizza Margarita + ingredients	x1	6,50€
Estrella Damm	x2	2,50€

Total items 2

Subtotal	+ 37,50 €
Taxes	+ 0,00 €
Discount	0,00 %

Total + 37,50 €

Table 5 Bill No. 31.452

Pizza california 5,00€	Pizza carbonara 5,00€	Pizza funggi 5,00€
Pizza margarita 5,00€	Pizza Vegetarian 5,00€	Pizza Marinera 5,00€
Pizza 4 formaggi 5,00€	Pizza calzone della casa 5,00€	"Torrada" with garlic and vegetables 5,00€

Following the use case number two, when the waiter needs to charge a customer, it can use the POS system, especially when there is cash involved and the cash drawer has to be opened. As the different modules are connected to the RMS, although, all the data available in one device will be also available, instantaneously on the other. This way, the waiter will be able to retrieve the bill on the POS and charge the customer without having to enter the data again to the system.

The screen above shows the structure of the POS bill edit page. Very similarly to the Wait Staff device, here there is the possibility to add items, but there is more information on screen (it can be set up as bigger or smaller). In this case, and taking advantage of a bigger screen, the ticket is included on the left side, together with more options in the shape of buttons.

From left to right, the bottom buttons allow to add an amount to the bill without specifying product, add a discount to the bill, mark the bill as take away, add a comment to this bill, open the calculator, go back to all the bills, print the ticket and start payment flow.

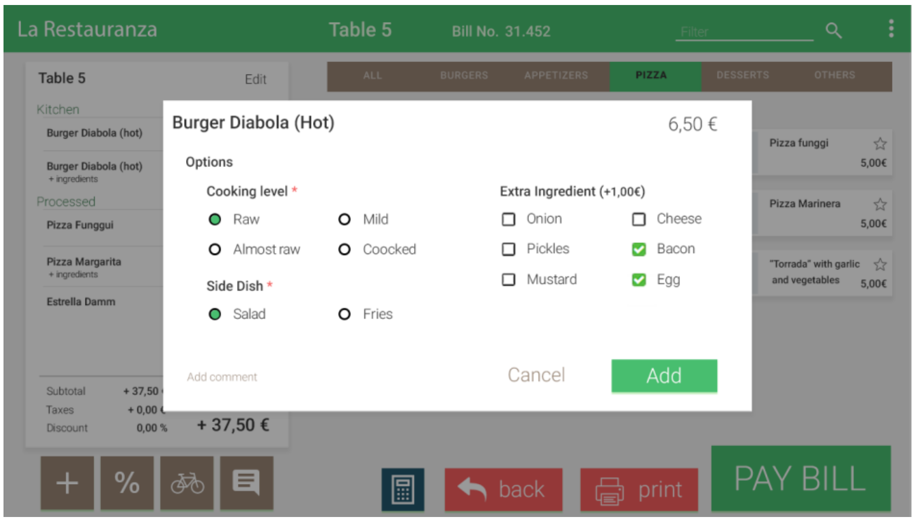


Table 5

Kitchen
Burger Diabola (hot)
Burger Diabola (hot) + ingredients
Processed
Pizza Funggui
Pizza Margarita + ingredients
Estrella Damm

Subtotal	+ 37,50 €	
Taxes	+ 0,00 €	+ 37,50 €
Discount	0,00 %	

Burger Diabola (Hot) 6,50 €

Options

Cooking level *: [✓ Raw] Mild Almost raw Coocked

Side Dish *: [✓ Salad] Fries

Extra Ingredient (+1,00€): Onion, Cheese, Pickles, [✓ Bacon], Mustard, [✓ Egg]

Pizza funggi	5,00€
Pizza Marinera	5,00€
"Torrada" with garlic and vegetables	5,00€

The screen above shows how the options to add an item to the ticket looks like. They are, again, similar to the Wait Staff module but taking advantage of a bigger screen.

7.2.3.2 ***Checkout***

Once the bill has to be payed, the waiter will start the checkout process. By doing so, the POS will show the checkout screen.

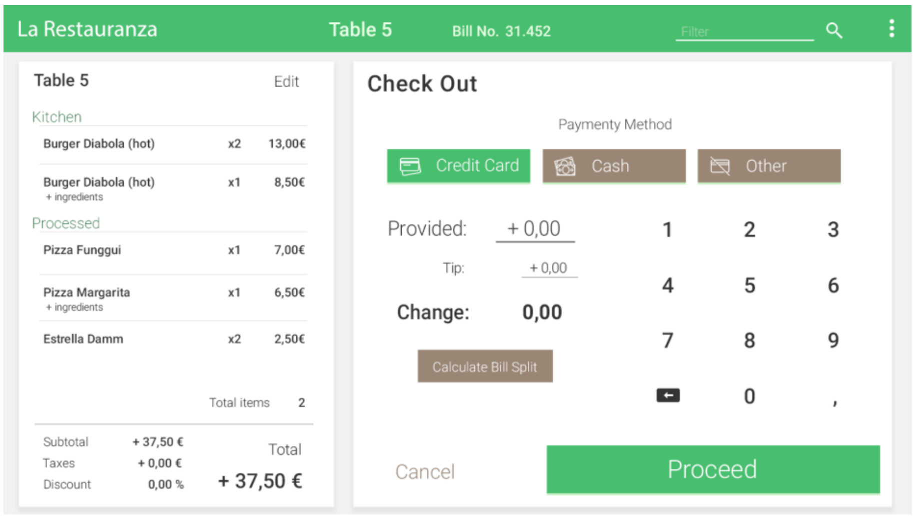


Table 5

Kitchen		
Burger Diabola (hot)	x2	13,00€
Burger Diabola (hot) + ingredients	x1	8,50€
Processed		
Pizza Funggui	x1	7,00€
Pizza Margarita + ingredients	x1	6,50€
Estrella Damm	x2	2,50€
Total items		2

Subtotal	+ 37,50 €	Total
Taxes	+ 0,00 €	+ 37,50 €
Discount	0,00 %	

Provided: + 0,00
Tip: + 0,00
Change: 0,00

This checkout screen displays the final ticket on the left, with all the information related to the bill that will be charged to a customer, and the payment method on the right.

In this screen, the waiter would enter the payment type, this case cash, and the amount provided by the customer. The application would show the change needed and then, once the button proceed is pressed, the bill could be printed and the cash drawer would open.

At this point, being the customer charged and the money saved, the bill would be automatically marked as closed.

7.3 Use case 3: OpenMenu access and ordering

OpenMenu is an extra feature of MenuServe, that allows users to access the system and perform the data entry by themselves without the need of a waiter. This means that they will be able to navigate through the menu of the restaurant with their devices or with a device set by the local on each table.

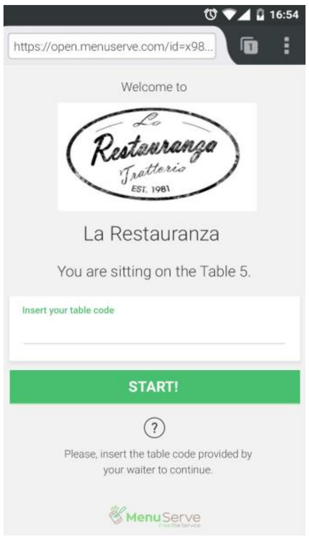
With this feature, the work of the waiters can be eased, allowing the restaurant to save in personnel specially in the shifts that there is less influx.

This feature is intended to evolve according to the reception of the public. That is why, for the first versions, the system will look very similar to the Wait Staff application, but with limited permissions and being available from web browsers.

This use case shows how the user would log into the system and order a meal, and how the waiter would be able to see this order and accept it when the “auto-accept” option is not enabled.

7.3.1 OpenMenu module (5)

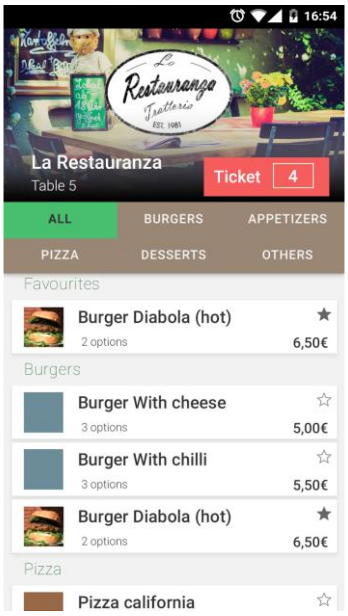
7.3.1.1 *Accessing to the system*

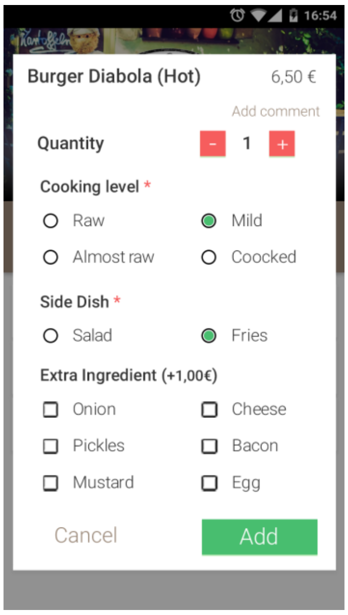

16:54
https://open.menuserve.com/id=x98...
Welcome to
La Restauranza
You are sitting on the Table 5.
Insert your table code
START!
Please, insert the table code provided by your waiter to continue.
MenuServe

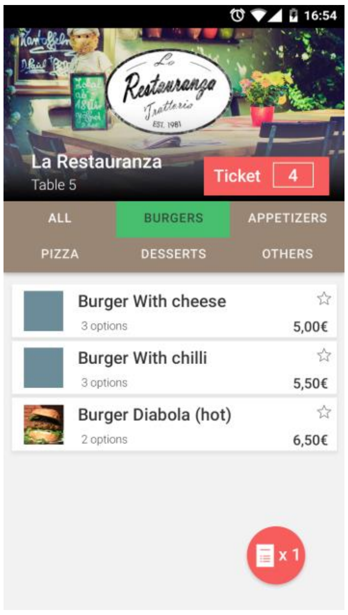
The system may be accessed through a website that the restaurant can advertise on its local or through QR codes set up in every table. Once the waiter has accommodated the customers in, he/she will provide them with a unique code that will allow this customer to log into the system to their table.

The screen on the left shows a web browser tab with the logotype of the restaurant and an input field where the customer will be able to introduce its table password, provided by the waiter.

7.3.1.2 *Ordering a meal*


16:54
La Restauranza
Table 5
Ticket 4
ALL
BURGERS
APPETIZERS
PIZZA
DESSERTS
OTHERS
Favourites
Burger Diabola (hot) 2 options 6,50€
Burgers
Burger With cheese 3 options 5,00€
Burger With chilli 3 options 5,50€
Burger Diabola (hot) 2 options 6,50€
Pizza
Pizza california



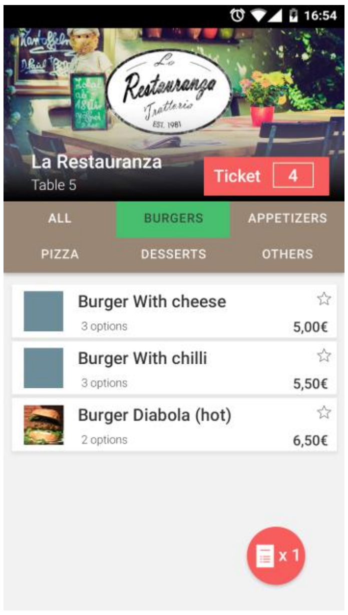

16:54
La Restauranza
Table 5
Ticket 4
ALL
BURGERS
APPETIZERS
PIZZA
DESSERTS
OTHERS
Burger With cheese 3 options 5,00€
Burger With chilli 3 options 5,50€
Burger Diabola (hot) 2 options 6,50€
x 1

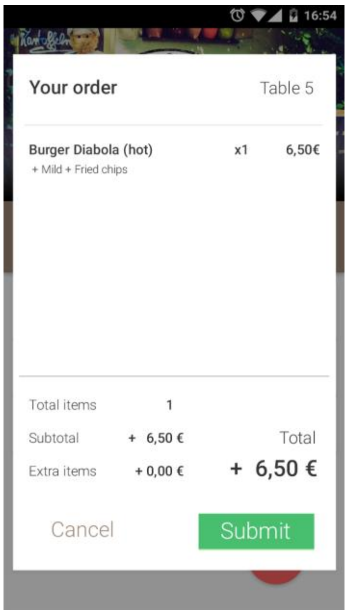
The workflow of ordering a meal on the OpenMenu system is very similar to the one for the Wait Staff application. In this case, although, it is limited to the customer's table.

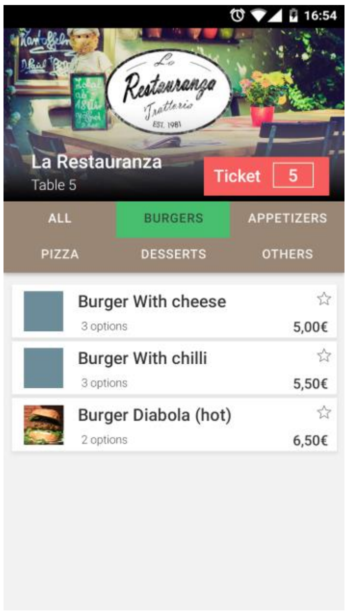
The looks of the OpenMenu are, as it is already visible, more appealing to the view. This is mainly achieved by adding an image of the restaurant (customizable from the back office). The process of selecting and reviewing your order is very similar.

Another important point of the OpenMenu application, although, is that allows the customers to see an item details page, where they can have more detail of a specific item on the menu and help them decide for it.

7.3.1.3 *Accepting an order*







Again, following the same principles of the Wait Staff device, the clients can review their order and submit it to the system.

At the same time, the customer has the ability to check not only its order but its already submitted items on the ticket, the same way the waiter would do. He/she can do that by pressing Ticket. On this screen, the customer may be able to request for assistance of the waiter or to request the bill.

7.3.2 Wait Staff module (6)

7.3.2.1 *Reviewing and accepting an OpenMenu order*

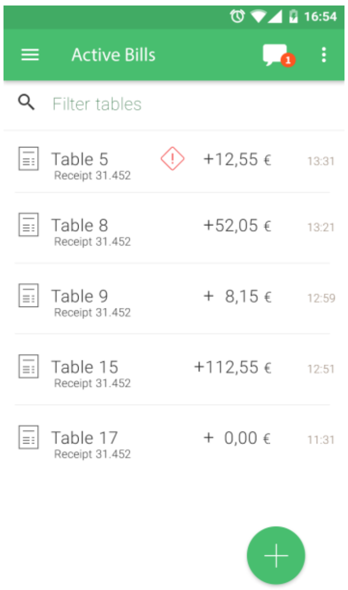

16:54
Active Bills
Filter tables

Table 5 Receipt 31.452	+12,55 €	13:31
Table 8 Receipt 31.452	+52,05 €	13:21
Table 9 Receipt 31.452	+ 8,15 €	12:59
Table 15 Receipt 31.452	+112,55 €	12:51
Table 17 Receipt 31.452	+ 0,00 €	11:31

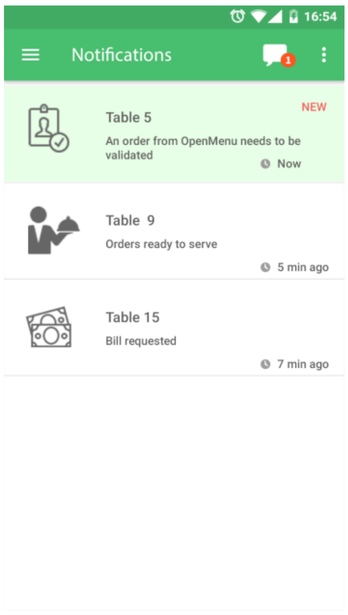
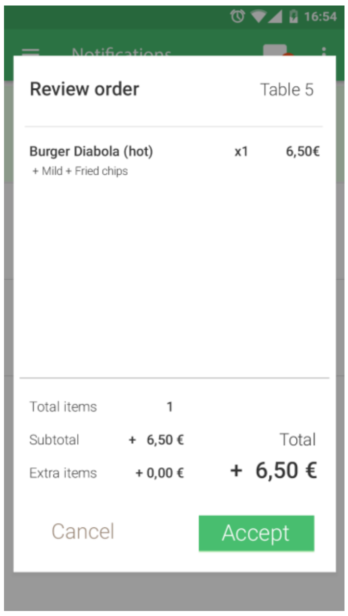

16:54
Notifications

Table 5 NEW
An order from OpenMenu needs to be validated
Now

Table 9
Orders ready to serve
5 min ago

Table 15
Bill requested
7 min ago


16:54
Review order Table 5

Burger Diabola (hot) x1 6,50€
+ Mild + Fried chips

Total items 1
Subtotal + 6,50 €
Extra items + 0,00 €
Total + 6,50 €

Cancel Accept

Once a customer has submitted an order, the waiter in charge of that table will receive a notification on its device. Thanks to that, the waiter can instantly know that there is a pending action to perform.

The first screen on the left shows all the list of active bills, and, because there is a pending item on the Table 5, there is an exclamation point. Moreover, the notification icon on the top right has a red circle with a number 1 on it, indicating that there is a new notification.

If the waiter enters to the notification screen, on the screenshot on the middle, he will see a new notification, highlighted on green. This, together with the icon of the notification, will help the waiter detect that there is a new action to do and this action is to validate an order that has been sent through OpenMenu.

On the last point, the waiter would click on the notification and a pop up to review the order would appear. Here, the waiter can make sure that the order makes sense, and accept it. Once accepted it, the order will go to the Kitchen device and follow the same process described in the use case 2.

8 Conclusions

8.1 Conclusions

Once the project *Business and Development plan for a SaaS based Restaurant Menu System* is over, it is fair to say that it has turned to be something bigger and more complex than initially expected. The ambition of the platform, involving different modules and roles is, indeed, a complicated system to define as there are many factors involved and many functionalities to take in account for. Even tough, it has been a very interesting project to work on it, as it is a perfect reflexion of the real world, where there are very complex needs to face and very complex solutions to define.

This is a project that has not only fulfilled its objective of demonstrate the knowledge learned into the master degree but has allowed me to take this acquired skill into a new level and learning while doing it. There is a vast amount of time invested on the analysis, definition and creation of all the sections of this project and there all served a bigger propose of creation a viable solution for a market, that could perfectly be exploited with further work.

Through the different parts of this project, it has been clear that creating a new product is not an easy task and a lot of sacrifices have been made in order to being able to obtain a viable solution that fits the features and objectives. Creating a product like MenuServe requires a set of skills that varies a lot from one to another; from the analysis of a situation and market, to performing the tasks of a business analyst defining a functional solution, going through the task of a software analyst by defining how this solution should behave and should be implemented, being part of the user interface and prototype creation using all the knowledge generated from all the other tasks, and finally performing the task of a product owner by creating a presentation and a demonstration of the product. I am glad to say that I have enjoyed all the parts of this project, even though a lot of times the effort needed to perform this tasks have been very high.

The first thing I have learned is that focusing too much in details is bad when you have a big scope in mind, as time is crucial and losing time performing a task will mean not having it to perform another one. This problem has affected the development of the project, focusing too many time on the definition itself and letting the prototype to the last task, losing the point of view of the lecturer as there were no more partial deliverables on it. In that part, I specially enjoy and I have skills for prototyping, so the damage may have been a bit mitigated, but in any case, it has been an issue that should be avoided in the future. Through the process of performing this process I have had the chance of applying knowledge acquired from work and from personal experiences, that lead me to perform what I consider a good job in establishing the bases of a product. And I realized that I especially like this task of business analysis and definition of new applications and solutions, as involves the same amount of technological skills than creative skills.

During the execution of the project, there are things that could have been done better. Specially in the market research and analysis parts of it. This was mostly clear when trying to perform the revenue estimation, as it is a very difficult task to do and there is a lot of unknowns that have to be really analysed to get close to a real estimation. This is why a further work to this project would be to perform an extensive analysis of the revenue in order to assess the viability of the project.

I especially enjoyed MenuServe and I think that the business model, joined with a well-developed and well tested product might be an excellent business if counting with the proper investment.

The main objective of the project itself was the creation of a business plan and the development base for a restaurant menu system, and I think it have been accomplished in a whole. The document proportionates the base on how to base the business and provides a very good picture of how the application can be, how it will work and how it will feel.

It is true, though, that some parts of the initial scope have been left out due the lack of time. Specially the marketing approach for the system and an extensive viability study. There have been two main factors to that issue, the first, as explained before, it has been focusing too much in some parts of the definition, but the second one have been the hardest of all.

I have performed this thesis during a point of my professional career where my workload was on its tops. My work requires me to travel a lot and to fulfil tight deadlines, as its core business is electoral modernization, and deadlines cannot be moved. This, in addition to familiar issues near the delivery dates of the project translated in lots of sleeping hours lost, and days and weekends of work.
This situation has led to the delay in the tasks to be performed during the project, and pushing all the heavy amount of work to the end of the available time. It has been a key indicator than the methodologies followed to control the tasks where to loose, and for next projects, a better task split will have to be done to be able to keep more focused on what really matters.

At the end, although, I am proud of the extra effort as the project has fulfilled its main objective and the outcome of the work done is satisfying.

MenuServe is indeed a very interesting project that I would really love to continue outside the master's premises. After months of working on it, the outcome is a well-structured platform that can be the base for a real product. The chance of working on a SaaS based product has allowed me to deepen in yet another trend of the sector of information and to imagine a situation where MenuServe would be very profitable.

MenuServe is a restaurant Menu System with lots of potential, but this is the tip of the iceberg. I strongly believe, but, that whit further work and the right people alongside, this project can become reality one day.

8.2 Further work

There is still lots of tasks and work to do in order to have a chance that MenuServe becomes a reality. The main objective of this further work is to assent all the bases to obtain financial aid and start a company under this product plan.

Please find below a list of the further work to advance in the stablish objective:

- Perform an extensive viability study to attach to this document.
- Perform a project plan an extensive cost estimation.
- Validate the prototype with different professionals of the hospitality sector, to detect all workflow flaws and missing opportunities.
- Add customer management to the system
- Add inventory tacking to the system
- Perform a marketing plan.
- Create a backlog of features and ideas for the project, in other to gather all the possible features to be added in the future.
- Perform a legal analysis of the system to stablish the base for the licenses and for the legislations that a system like MenuServe should follow.
- Create a complete business case and a product presentation and demonstration with all the knowledge gathered.
- Meet with investors.

Bibliography

[1] “Capterra - Restaurant management System,” 26 3 2016. [Online]. Available: http://www.capterra.com/restaurant-management-software/. [Accessed 26 03 2016].

[2] M. Authors, “Point Of Sale,” Wikipedia, 3 5 2003. [Online]. Available: https://en.wikipedia.org/wiki/Point_of_sale. [Accessed 27 3 2016].

[3] Scandit, “Mobile Point-of-Sale Apps: Redefining the Retail Industry,” Scandit, 10 5 2013. [Online]. Available: http://www.scandit.com/2013/05/10/mobile-point-of-sale-apps-redefining-the-industry/. [Accessed 28 3 2016].

[4] M. Parpal, “foodservicewarehouse,” 14 6 2015. [Online]. Available: http://www.foodservicewarehouse.com/blog/importance-point-sale-pos-system/. [Accessed 28 03 2016].

[5] T. G. o. Canada, “Foodservice Profile Document - Page 7 Table 2,” 2011. [Online]. Available: https://www.gov.mb.ca/agriculture/market-prices-and-statistics/trade-statistics/pubs/spain_foodservice_en.pdf.

[6] “Traffic share of independent and chain restaurants worldwide in 2010,” 2010. [Online]. Available: http://www.statista.com/statistics/204582/market-share-of-independent-and-chain-restaurants-worldwide/. [Accessed 28 03 2016].

[7] “RMS,” [Online]. Available: https://www.clover.com/.

[8] “R.M.S.,” [Online]. Available: http://www.jolomosoftware.com/RMS_New/.

[9] “RMS,” [Online]. Available: http://www.mirus.com.

[10] S. Miles, “7 Cloud-Based POS Systems for SMBs,” Street Fight Mag., 4 2 2013. [Online]. Available: http://streetfightmag.com/2013/02/04/7-cloud-based-pos-systems-for-smbs/. [Accessed 27 3 2016].

[11] “REVEL - POS,” [Online]. Available: http://revelsystems.com/features/for-quick-service/.

[12] “Breadcrumb - POS,” [Online]. Available: https://breadcrumb.com/.

[13] “POS,” eHopper, [Online]. Available: http://www.ehopper.com.

[14] “POS,” LOYVERSE, [Online]. Available: https://loyverse.com/en/products/pos/.

[15] V. authors, “Pareto principle,” Wikipedia, [Online]. Available: https://en.wikipedia.org/wiki/Pareto_principle. [Accessed 09 06 2016].

[16] A. C. Drawer, “https://www.cashdrawer.com/,” 2002. [Online]. Available: https://www.cashdrawer.com/wp-content/uploads/documents/serial_pro_users_guide.pdf. [Accessed 12 05 2016].

[17] A. University, “acmepointofsale.com,” 12 6 2012. [Online]. Available: http://acmepointofsale.com/acmehelpmobile/cashdrawer.htm. [Accessed 11 5 2016].

[18] Unknow, “Push Notifications Explained,” UrbanShip, [Online]. Available: https://www.urbanairship.com/push-notifications-explained.

[19] Wikipedia, “Material Design,” 12 06 2016. [Online]. Available: Material Design (codenamed Quantum Paper)[1] is a design language developed[when?] by Google. Expanding upon the "card" motifs that debuted in Google Now, Material Design makes more liberal use of grid-based layouts, responsive animations and transitions,. [Accessed 12 06 2016].

[20] Wikipedia, “Wireframing,” 12 06 2016. [Online]. Available: https://en.wikipedia.org/wiki/Website_wireframe.

[21] M. Islam, “Ultra Free Admin Dashboard UI,” 29 05 2016. [Online]. Available: https://www.behance.net/gallery/16433957/Ultra-Free-Admin-Dashboard-UI-.

[22] V. authors, “Web Content Accessibility Guidelines,” 12 06 2016. [Online]. Available: https://en.wikipedia.org/wiki/Web_Content_Accessibility_Guidelines.

[23] “What Goes Into a Functional Specification?,” [Online]. Available: http://www.bridging-the-gap.com/functional-specification/. [Accessed 2016 04 29].

[24] “Sample Functional Specifications,” [Online]. Available: http://getsp.sbisite.com/SBI/IT/Shared%20Documents/IT%20FuncSpec.pdf. [Accessed 2016 04 28].

[25] “User Case Diagrams,” [Online]. Available: http://www.agilemodeling.com/artifacts/useCaseDiagram.htm. [Accessed 2016 04 28].

[26] “dependencies and constraints an introduction,” 2015 10 13. [Online]. Available: http://www.girlsguidetopm.com/2014/10/dependencies-and-constraints-an-introduction/. [Accessed 2016 05 30].

[27] Wikipedia, *[IMAGE] JavaScript UI widgets library,* 2015.

[28] L. Manovich, The Language of New Media, Cambridge: MIT Press, 2011.

[29] C. Anderson, “The Man Who Makes the Future: Wired Icon Marc Andreessen,” 24 4 2012. [Online]. Available: http://www.wired.com/epicenter/2012/04/ff_andreessen/.

Annexes

Annex I – Glossary

ERM - Enterprise Resource Manager
KPI – Key Performance Indicator
POS – Point of Sale
RMS - Restaurant Management System
SaaS – Software as a Service
SLA – Service-level agreement
VAT – Value Added Tax

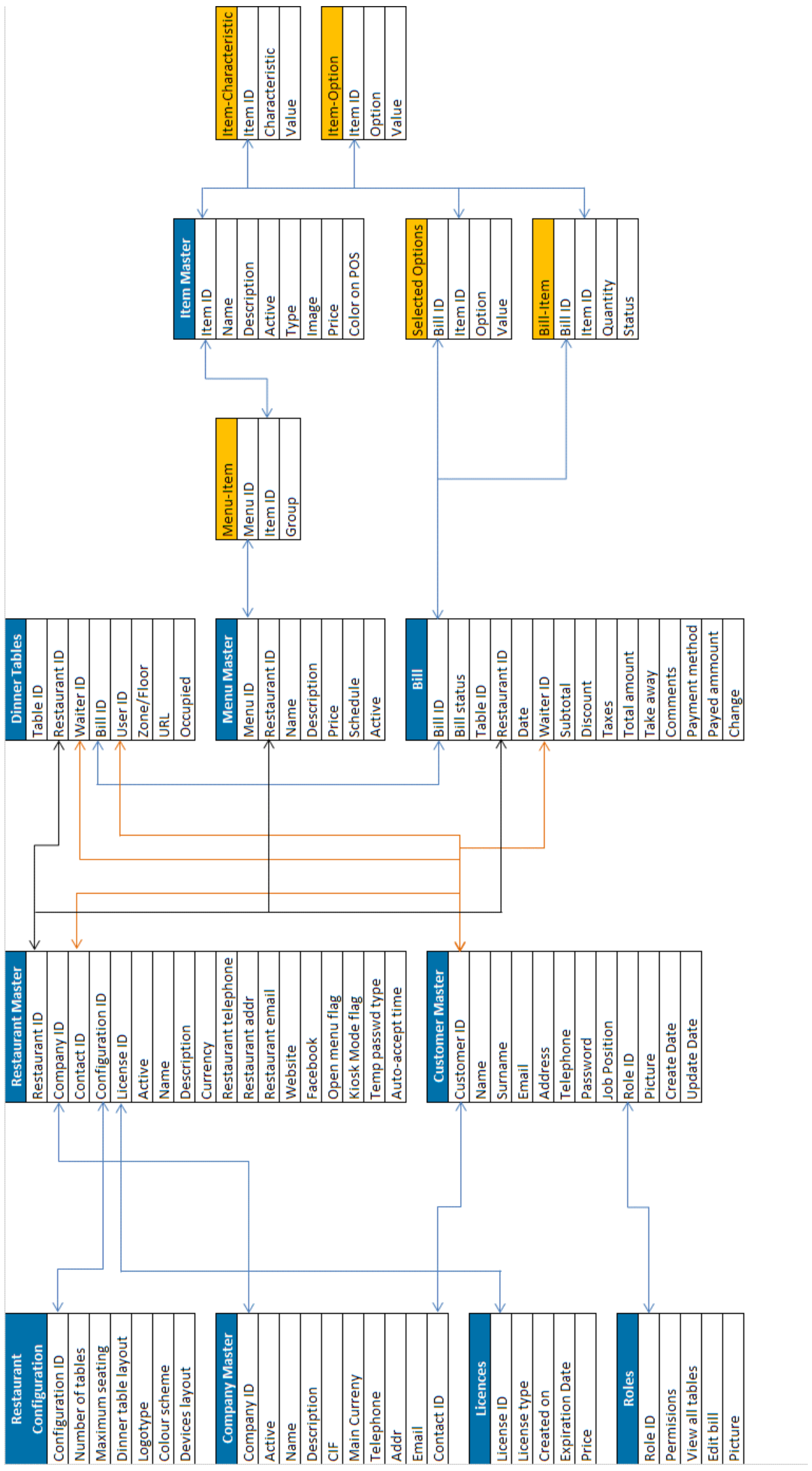
Annex II – Deliverables

The list of deliverables created for this project may be found below:

- Wireframes
- HQ designs (final)
- Raw HQ design (in Photoshop .PSD format)
- Raw Logotype design (in Illustrator .AI format).
- Icon set used on the designs
- Prototype project file (In JustInMind .VP format)
- Prototype in HTML (final)
- Entity Relationship Diagram (.xls)

All this items can be downloaded through the link provided on the attachment when this project has been delivered. If you are a third party reader and may be interested in having access to this materials, you can contact the author at inaki.bigata@gmail.com.

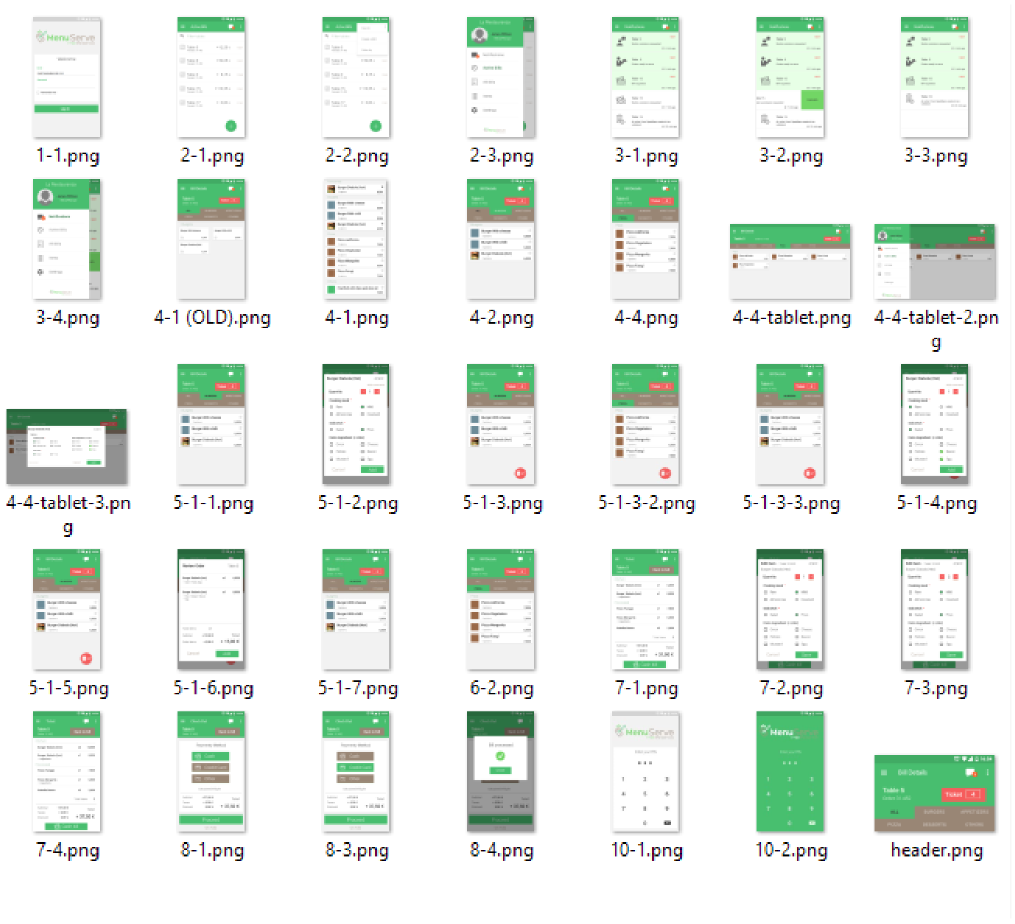
Annex III – Entity Relationship Diagram



Annex IV – Wireframes

Wireframes may be found on the deliverables of the project. If you are a third party reader and are interested in having access to this materials, you can contact the author at inaki.bigata@gmail.com.

Annex V – High Quality Designs



There has been a handful of High quality designs for MenuServe. As attaching all as annexes would increase the size of the document to an unbearable level, there have been decided to add them as an external link.

If you are a third party reader and are interested in having access to this materials, you can contact the author at inaki.bigata@gmail.com.

Annex V – CV

Iñaki Bigatà Fages

Email:	inaki.bigata@gmail.com
LinkedIn:	http://www.linkedin.com/in/inakibigata/

Work Experience

Pre-Sales Demo Specialist (Demos & Prototypes) at Scytl Secure Electronic Voting
Full-time, February, 2015 - Current (1 year 5 months)

Software Engineer at Accenture Technology Solutions
Full-Time, November, 2013 - February, 2015 (1 year 4 month)

Field Hockey Coach at Linia 22 H.C.
Part-time, September, 2006 - June, 2009 (2 years 10 month)

Promoting Manager at Sweatbox Night Group.
Part-time, September, 2006 - June, 2008 (1 year 10 month)

Education

Universitat Obvert de Catalunya
Master's degree, Multimedia applications, 2014 - 2016

AGH University of Science and Technology
Bachelor's Thesis, Department of Computing Science, 2012 - 2012

Universitat Politècnica de Catalunya
Bachelor's degree, Telecommunications Engineering, spec. in Image and Sound, 2008 - 2012