UNIVERSIDAD DE CASTILLA-LA MANCHA
ESCUELA SUPERIOR DE INFORMÁTICA

COMPUTER SCIENCE ENGINEERING

SPECIFIC TECHNOLOGY OF
INFORMATION TECHNOLOGIES

DEGREE FINAL PROJECT

DYN Print: Automated Printing System for Copy Centres

Ruth Rodríguez-Manzaneque López

July, 2018

DYN PRINT: AUTOMATED PRINTING SYSTEM FOR COPY CENTRES

UNIVERSIDAD DE CASTILLA-LA MANCHA

ESCUELA SUPERIOR DE INFORMÁTICA

Information Technologies and Systems

TECNOLOGÍA ESPECÍFICA DE
INFORMATION TECHNOLOGIES

TRABAJO FIN DE GRADO

DYN Print: Automated Printing System for Copy Centres

Autor: Ruth Rodríguez-Manzaneque López

Director: David Vallejo Fernández

Director: Carlos González Morcillo

July, 2018

Ruth Rodríguez-Manzaneque López

Ciudad Real – Spain

E-mail: Ruth.RodriguezManzaneque@alu.uclm.es
Teléfono:

TRIBUNAL:

Presidente:

Vocal:

Secretario:

FECHA DE DEFENSA:

CALIFICACIÓN:

PRESIDENTE	VOCAL	SECRETARIO
Fdo.:	Fdo.:	Fdo.:

Resumen

La integración del comercio electrónico en los negocios más recientes resulta una estrategia de innovación y de adaptación a un mercado cada día más competitivo en el que es clave alcanzar al mayor número de clientes y ofrecer una calidad de servicio mejor. En negocios modernos puede resultar un proceso más fácil, sin embargo, en negocios tradicionales es más complicada la adaptación y la concienciación de sus beneficios debido a que su modelo de negocio ya está asentado y produce los resultados esperados.

El negocio de las copisterías, en su mayoría, aún trabaja con un modelo tradicional en sus tiendas, de forma que surgen aspectos que afectan de forma negativa a su productividad, como la acumulación inesperada de clientes y de trabajos solicitados en la copistería y el excesivo tiempo que puede llevar realizar algunos pedidos. Son factores que pueden afectar negativamente tanto a la copistería como a sus clientes.

El diseño y el desarrollo de una solución tecnológica y digital que automatice el proceso tradicional de las copisterías es el cometido de este Trabajo Fin de Grado (TFG), cuyo objetivo principal es el de aumentar la productividad de las copisterías y, de la misma forma, aumentar el nivel de satisfacción del cliente con respecto al servicio que estas empresas ofrecen. El sistema supone una alternativa al proceso tradicional que incorpora una transformación digital del mismo.

Este sistema a desarrollar, llamado DYN Print, reunirá las características necesarias para ofrecer el servicio a través de diferentes dispositivos, dado que contará con un diseño *responsive* y con una interfaz intuitiva que garantice un correcto funcionamiento a todo tipo de usuario. Por otro lado, este sistema también dispondrá de la escalabilidad y disponibilidad necesaria para prestar sus servicios a un gran número de usuarios, puesto que se desplegará de forma que esté en funcionamiento para varias copisterías tradicionales.

Abstract

The integration of e-commerce in the newest businesses is a strategy of innovation and adaptation to an increasingly competitive market in which it is essential to reach the greatest number of customers and offer a better quality of service (QoS). In modern businesses it can be an easier process, however, in traditional businesses it is more complicated to adapt and become aware of their benefits because their business model is already established and produces the expected results.

The copy centre business, for the most part, still works with a traditional model in its stores, resulting in aspects that negatively affect its productivity, such as the unexpected accumulation of customers and jobs requested in the copy centre and the excessive time it can take to carry out some orders. These factors can adversely affect both the copy centre and its customers.

The design and development of a technological and digital solution that automates the traditional process of the copy centres is the purpose of this degree project, whose principal objective is to increase the productivity of the copy centres and, in the same way, to increase the level of customer satisfaction with respect to the service that these companies offer. The system is an alternative to the traditional process that incorporates a digital transformation of the same.

This system to be developed, called DYN Print, will have the necessary characteristics to offer the service through different devices, since it will have a *responsive* design and an intuitive interface that guarantees a correct operation to all types of users. In addition, the system will also have the scalability and availability needed to serve a large number of users, as it will be deployed in such a way as to be operational for multiple traditional copy centres.

Acknowledgements

A *ella*, por ser el apoyo incondicional que siempre necesito y necesitaré a mi lado. No existen palabras para agradecer el hecho de que siempre esté ahí.

A *él*, por hacerme fuerte y enseñarme a no rendirme.

A *ella*, por creer en mí, en nosotras y en el poder que tenemos de luchar por nuestros sueños.

A *ellos*, por acompañarme durante estos cuatro años y convertirlos en un recuerdo lleno de sonrisas, esfuerzos, alegrías y encuentros dificiles de olvidar.

A *ellas*, por vivirnos, apoyarnos y disfrutarnos día tras día y noche tras noche, sin dejar de cantar y de bailar.

A *ellos*, por guiarme y enseñarme durante el trayecto que ha supuesto este proyecto y por confíar en mí desde el primer momento.

Ruth Rodríguez-Manzaneque López.

A mi padre.

Contents

List of Tables

List of Figures

List of Listings

List of Acronyms

AI	Artificial Intelligence
AJAX	Asynchronous JavaScript And XML
ARP	Address Resolution Protocol
AWS	Amazon Web Services
API	Application Programming Interface
CERN	European Organization for Nuclear Research
CDN	Content Delivery Network
CSS	Cascading Stylesheets
DevOps	Development Operations
DHCP	Dynamic Host Configuration Protocol
DNS	Domain Name System
DOM	Document Object Model
FTP	File Transfer Protocol
HTML	HyperText Markup Language
HTTP	Hypertext Transfer Protocol
HTTPS	Hypertext Transport Protocol Secure
ICMP	Internet Control Messaging Protocol
IDE	Integrated Development Environment
IoT	Internet of Things
IP	Internet Protocol
IT	Information Technology
JSON	JavaScript Object Notation
MIME	Multipurpose Internet Mail Extensions
OSI	Open System Interconnection
PaaS	Platform as a Service
PDF	Portable Document Format

RAM	Random Access Memory
SaaS	Software as a Service
SDK	Software Development Kit
SMS	Short Message Service
SQL	Structured Query Language
SSL	Secure Sockets Layer
TCP	Transmission Control Protocol
TLS	Transport Layer Security
UCLM	University of Castilla-La Mancha
UI	User Interface
URL	Uniform Resource Locator
UUID	Universally Unique Identifier

Chapter 1

Introduction

TECHNOLOGY and information technology can integrate their advantages into almost any type of business. Taking into account the context in which you are working, it is possible to make an analysis of the problems and possible solutions. A software system can bring the expected benefits to even the most traditional retailers. Throughout this chapter, the context of e-commerce in terms of web and mobile technologies is discussed, in addition to presenting a possible solution to the automation process of the copy centre market.

1.1 Adaptation of enterprises to e-commerce

Nowadays it is rare not to contemplate the option of making some of the most daily purchases over the Internet, to go to a business that makes available an online store, or even to a company whose business is entirely conducted online. People are aware that it is possible to make these purchases in a traditional shop, but they are also aware that, for reasons of convenience, availability and even economic issues, they can bet on e-commerce.

Years ago, the first online shops appeared, full of insecurities and rejection by the majority of people, especially those who distrusted the Internet itself. Over the years, this has been a field that has **gained the trust of its public** to a greater or lesser extent, a process that has been researched in [Bha00] taking as an example the users who are customers of e-brokerage trade.

After all, it is difficult to find Spanish companies that have not wanted to take the step in this **digital transformation towards online technologies**, as they are aware of the benefits they bring. The evolution of this trade-oriented digital transformation is studied in [DPGE04], which assesses the risks and benefits that can occur in Spanish companies, taking into account the sector in which they operate and the market in which they compete.

It is important not to forget the problems that can still be found in this type of trade, such as the hesitant security provided by Internet applications. The amount of private data being handled is immense, and frauds continue to exist, most recently enhanced by the social engineering that goes along with distributed malware.

Despite this, the number of companies that offer their products or services through the web and the number of sales on these trading platforms are increasing by leaps and bounds,

a statistic that affects not only those businesses that have been born on the web itself, but also those that have traditionally opted for the digital transformation. Another of the difficulties faced by this new sales system is the **loyalty of its consumers**, a factor that increases its complexity on the web due to the ease of navigation, the anonymity of buyers and the threat of constant abandonment. In [Gef02] there is a study of how some of the traditional processes of customer acquisition through e-commerce can be applied, trying to counter these risk factors that arise when managing e-commerce.

It is a challenge for **small and medium sized companies** to face these risks in order to seek **innovation in a competitive and constantly growing market**. As detailed in [FPAR15], in this type of business it is essential to create a strategic plan that considers e-commerce as a means of creating new sales opportunities and new distribution channels, since one of the important advantages of electronic commerce is the attraction of customers whose location is not the same as that of the company, so that the scope is greater.

But these are not all limitations and complications, as **e-commerce offers advantages** to be taken into account, such as reducing costs for the company or increasing the quality of the service offered to customers. These are some of the advantages defined in [NMC13], where the term m-commerce is also mentioned.

M-commerce is a type of e-commerce that refers to the online sale of products and services through portable devices, such as a mobile phone or a tablet. With the advent of 3G and 4G smartphones and telecommunications, this approach has grown dramatically. It is a means of communication that has not only contributed to the evolution of electronic commerce but has also provided an improvement in the relations between companies and their customers and suppliers, as explained in [KS03].

Taking into account the statistics highlighted in the *Mobile Report in Spain and in the world 2017*[1], the **mobile phone is the most widely used device with Internet access in Spain**. Focusing on the online sales statistics also reflected in this report, in 2017 it was estimated that 34% of online sales were made via a mobile device and would increase by 200% in the coming years compared to sales via a conventional computer. In the figure 1.1[2] it can be seen that in 2016 there was already a growth in transactions on mobile devices.

These data reveal the importance of framing m-commerce among the strategy options that Spanish companies take, as they translate into a likely increase in profits, since it can be observed an increase in the confidence of users when conducting business transactions through these mobile devices.

[1] http://mktefa.ditrendia.es/informe-mobile-espac3b1a-mundo-2017

[2] https://go.ey.com/2uyRjmQ

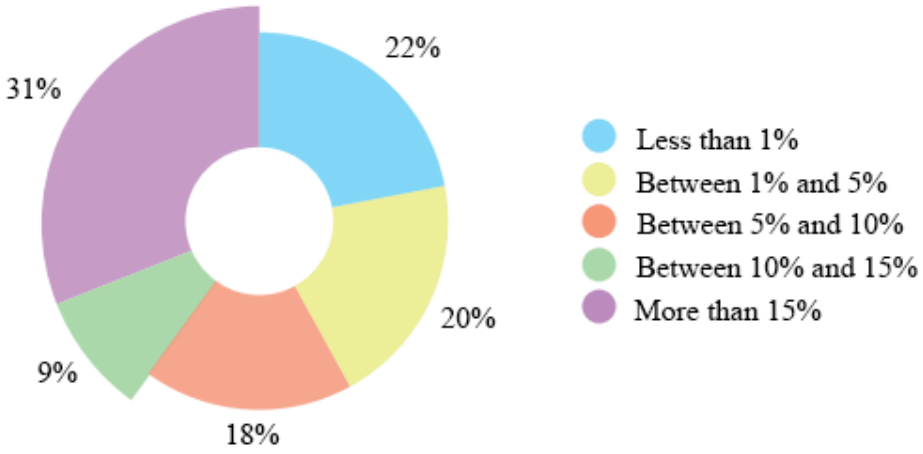

Figure 1.1: Percentages of turnover from 2016 mobile phones

1.2 Challenges in the reprographics market

From the point of view of the reprographics business, it is known that it is a fairly traditional market that has not opted for the digital transformation mentioned in the previous section until relatively recently. Nowadays, there are online applications that offer these services. However, they are not numerous and most of them only offer their services in the big cities of the country. Therefore, it is still a market that has not been further integrated with current technologies.

In a copy centre, the traditional process of dealing with customer requests is **slow and sometimes unproductive**. The factor that creates these characteristics is mainly the time spent waiting by both customers and employees.

Given a successful copy centre, it is normal that, at certain times of the day, there is a **large number of customers waiting for being attended**. Depending on the order to be placed and the quantity, each customer will take time to be attended and receive the result. The time customers are waiting in line at copy centre can be excessive.

On the other hand, for the employee or employees working in the copy centre, not only must they go client by client collecting the files they intend to work with from the storage devices or from the email, but **printers, as they usually have more than one running, have a runtime that also impacts on the total time** it takes to complete an order, as it also depends on the number of files or content with which they work. The employee must print out the documents and hand them to the customer one by one, regardless of the amount of time involved. There is also **time involved in the order payment process**, as dealing with small amounts (cents) per number of pages or colour format can sometimes take longer to prepare the corresponding money change.

All these factors directly influence the estimated **response time since the time the customer arrives at the copy centre until he receives the result of his request**. As mentioned above, there are copy centres that have integrated into their services tools that can help to a greater or lesser extent to speed up this process, such as making available to customers a computer in the store from which they can send documents to the copy centre employee. However, the print queue remains the same and so does the payment process as well. Because of the analysis of this problem related to copy centres, a technological solution can be proposed that can provide the necessary time reduction in the process.

1.3 Proposal design and motivation

The digital transformation of a business is not a simple process, as an analysis and strategy must be carried out at company level, and the **objectives, advantages and potential risks** that may arise must be identified. There are a variety of technologies and methodologies to carry out this process. A procedure is explained in [Lom15] in order to facilitate understanding of this change in business and production model.

Productivity is a key point in the copy centre business, as the benefits depend directly on it. The development of an online software system capable of increasing the productivity of the business and its employees, as well as reducing the problems mentioned above, such as **reducing the time spent in production**, is a commitment to innovation, contemplating an automation of the traditional model.

The technologies used for the development of this system are not the most important, but the **reproduction and integration of the printing service of the copy shops through a web application**. It is essential that the traditional process is not only reflected in the functioning of the system, but also improved in a way that attracts more customers and improves the level of quality of service (QoS).

The approach and development of DYN Print, the system discussed in this work, is attributed to the need for the copy centre to deal with the above-mentioned problems: a software project that is able to give customers the ability to save time and make their orders anytime, anywhere. On the other hand, the system should offer copy centres a more automatic and fluid management of the orders they receive, as well as the possibility of reducing the printing queue time and the management of the corresponding payments.

Offering this service online would be **complementary to the service normally offered in the copy centres**. The priority given to customers going to the copy centre will continue to exist, but it reduces the abundance of customers and consecutive orders that may arise at certain times.

The payment process is also a factor that benefits costumers. The management of a **virtual balance would replace the traditional payment of orders**, so that the customer does not need to worry about the payment when collecting the results as well as the employees.

DYN Print is a system whose development is focused mainly on its use through mobile devices. As already mentioned, the level of transactions through this device is growing rapidly and taking advantage of this situation would be beneficial to increase the productivity in copy centres. Most people have a mobile device they use every day, so having this service available to them may increase the level of customer satisfaction.

1.4 Structure of the document

Listed below are the chapters contained in this document along with a brief explanation of what can be discussed in them:

- **Chapter 2 - Objectives:** Both the main objective of this project and the more specific objectives are clearly detailed in terms of the service DYN Print will provide.
- **Chapter 3 - State of the art:** A compilation and analysis of the concepts and technologies that will be integrated into the development of this project is presented. It also includes a comparison and analysis of the applications similar to this project that are currently in operation.
- **Chapter 4 - Working Methodology:** The planning and development methodologies to be implemented in the development of this project are explained in this chapter, along with the hardware and software technologies that will be used.
- **Chapter 5 - Architecture:** This chapter will focus on the design of the system architecture and its implementation, focusing on the different functionalities it offers. In addition, diagrams are presented to help understand its structure in terms of modules, data model and classes implemented.
- **Chapter 6 - Results:** The results obtained after the development are presented in this application. Both an analysis of the completed functionality and screenshots of the application are presented. There is also a description of the work phases that have been carried out.
- **Chapter 7 - Conclusions:** Finally, this last chapter analyses the fulfilment of the objectives of the project, proposes different ideas for its improvement and also includes a personal conclusion regarding the development of this project.

Chapter 2

Introducción

La tecnología y la informática son capaces de integrar sus ventajas casi en cualquier tipo de comercio. Teniendo en cuenta el contexto en el que se trabaja, se puede realizar un análisis de los problemas y de las posibles soluciones. Un sistema software puede dar los beneficios esperados hasta a los comercios más tradicionales. A lo largo de este capítulo, se discute el contexto del comercio electrónico en cuanto a tecnologías web y móvil, además de presentar una posible solución a la problemática del mercado de las copisterías.

2.1 Adaptación de las empresas al comercio electrónico

Hoy en día se hace raro no contemplar la opción de realizar algunas de las compras más cotidianas a través de Internet, acudir a un negocio que pone a disposición una tienda online, o incluso a alguna empresa cuya actividad comercial se desarrolla completamente de forma online. La gente es consciente de que existe la posibilidad de realizar estas compras en un comercio tradicional, pero también es consciente de que, por razones de comodidad, de disponibilidad e incluso de cuestiones económicas, puede apostar por el comercio electrónico.

Hace años aparecían los primeros comercios online, plagados de inseguridades y rechazo por parte de la gran mayoría de personas, sobre todo para aquellos que desconfiaban del propio Internet. A lo largo de estos años, es **un campo que se ha ido ganando la confianza de su público** en menor o mayor medida, un proceso que se ha investigado en [Bha00] tomando como ejemplo a usuarios que son clientes del comercio sobre el corretaje electrónico.

Después de todo, es difícil encontrar a compañías españolas que no hayan querido dar el paso en esta **transformación digital hacia las tecnologías online**, ya que son conscientes de los beneficios que éstas les aportan. La evolución que ha experimentado esta transformación digital enfocada al comercio se estudia en [DPGE04], donde se evalúan los riesgos y los beneficios que se pueden producir en las compañías españolas teniendo en cuenta el sector en el que se encuentran y en el mercado donde compiten.

No hay que olvidar los problemas que aún se pueden encontrar en este tipo de comercio, como la titubeante seguridad prestada por las aplicaciones en Internet. La cantidad de datos privados que se manejan es inmensa, además de que siguen existiendo los fraudes, últimamente potenciados por la ingeniería social que se lleva a cabo junto con el malware distribuido.

A pesar de ello, el número de empresas que ofrecen sus productos o servicios a través de la web y el número de ventas en estas plataformas de comercio incrementa a pasos agigantados, una estadística que no solo afecta a los comercios que han nacido en la propia web, sino también para aquellos tradicionales que han apostado por la transformación digital. Otra de las dificultades que afronta este nuevo sistema de venta es la **lealtad de sus consumidores**, un factor que incrementa su complejidad en la web debido a la facilidad de navegación, a la anonimia de los compradores y a la amenaza de abandono constante. En [Gef02] se realiza un estudio de cómo puede aplicarse algunos de los procesos tradicionales de captación de clientes a través de un comercio electrónico, tratando de combatir contra estos factores de riesgo que surgen al gestionar un comercio electrónico.

Para las **medianas y pequeñas empresas** es todo un reto afrontar estos riesgos con tal de buscar la **innovación en un mercado competitivo y que está en constante crecimiento**. Tal y como se detalla en [FPAR15], en este tipo de empresas es fundamental crear un plan estratégico en el que se contemple el comercio electrónico como un medio para crear nuevas posibilidades de venta y nuevos canales de distribución, pues una de las ventajas importantes del comercio electrónico es la captación de clientes cuya localización no sea la misma que la de empresa, de forma que el alcance es mayor.

Pero no todo son limitaciones y complicaciones, pues **el comercio electrónico ofrece unas ventajas a tener en cuenta**, como puede ser la reducción de costes para la empresa o un incremento de la calidad del servicio ofrecido a los clientes. Son algunas de las ventajas que se definen en [NMC13], donde también se menciona el término m-commerce.

M-commerce es un tipo de comercio electrónico que hace referencia a la venta online de productos y servicios a través de dispositivos portátiles, como puede ser un teléfono móvil o una Tablet. Es un enfoque que, con la llegada de los smartphones y las telecomunicaciones 3G y 4G, ha ido creciendo desorbitadamente. Es un medio de comunicación que no solo ha aportado a la evolución del comercio electrónico, sino que ha proporcionado una mejora en las relaciones entre las empresas y sus clientes y proveedores, tal y como se explica en[KS03].

Teniendo en cuenta las estadísticas remarcadas en el *Informe Mobile en España y en el mundo 2017*[1], **el teléfono móvil es el dispositivo con acceso a Internet más usado en España**. Centrándose en las estadísticas de ventas online que también se reflejan en este

[1] `http://mktefa.ditrendia.es/informe-mobile-espac3b1a-mundo-2017`

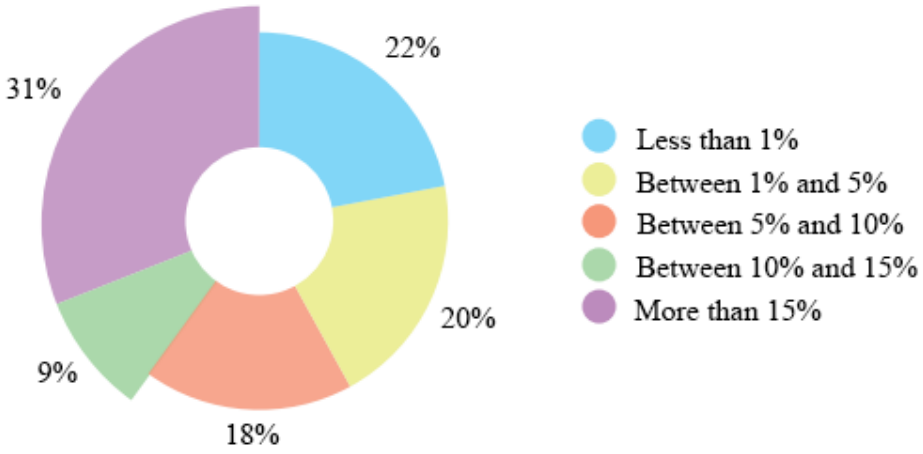

Figure 2.1: Percentages of turnover from 2016 mobile phones

informe, en el año 2017 se estimó que un 34% de las ventas online se realizaron a través de un dispositivo móvil y que experimentarían un incremento de un 200% en los próximos años con respecto a la venta a través de un ordenador convencional. En la figura 2.1[2] se puede observar como ya en el año 2016 se experimentó un crecimiento de las transacciones en dispositivos móviles.

Estos datos revelan la importancia de enmarcar el m-commerce entre las opciones de estrategia que tomen las empresas españolas, pues se traducen en un probable incremento de los beneficios obtenidos, ya que se puede observar un aumento de la confianza de los usuarios a la hora de realizar transacciones de negocio a través de estos dispositivos móviles.

2.2 Dificultades en el mercado de la repografía

Desde el punto de vista del comercio de la reprografía, se sabe que es un mercado bastante tradicional que no ha apostado por la transformación digital mencionada en la sección anterior hasta hace relativamente poco. Hoy en día se conocen aplicaciones online que ofrecen estos servicios, sin embargo, no son numerosas y además la mayoría sólo ofrecen sus servicios en las grandes ciudades del país. Por tanto, aún es un mercado que no se ha integrado en mayor medida con las tecnologías actuales.

En una copistería, el proceso tradicional que se lleva a cabo **para atender a las solicitudes de los clientes es lento y a veces, poco productivo**. El factor que crea estas características es, principalmente, el tiempo de espera por parte tanto de los clientes como de los empleados.

Teniendo en cuenta una copistería con éxito, es normal que a ciertas horas del día se cree **una gran afluencia de clientes a la espera de ser atendidos**. Dependiendo del pedido

[2] https://go.ey.com/2uyRjmQ

que vayan a realizar y de la cantidad del mismo, cada cliente tardará más o menos en ser atendido y en recibir el resultado. El tiempo que los clientes están esperando en la cola de las copisterías puede llegar a ser excesivo.

Por otro lado, en cuanto al empleado o empleados que trabajan en la copistería, no sólo tienen que ir cliente por cliente recopilando los archivos con los que pretende trabajar desde los dispositivos de almacenamiento o desde el correo electrónico, sino que **las impresoras, pues suelen tener más de una en funcionamiento, tienen un tiempo de ejecución que también influye en el tiempo total que se tarda en completar un pedido**, pues también depende de la cantidad de archivos o de contenido con el que se trabaja. El empleado debe imprimir los documentos y entregárselos al cliente de uno en uno, independientemente de la cantidad que sea o el tiempo que conlleve. También hay tiempo que se debe invertir en el proceso de pago del pedido, pues al tratarse con cantidades pequeñas (céntimos) por número de páginas o formato del color, a veces puede tardarse más en preparar el cambio correspondiente.

Todos estos factores influyen directamente en **el tiempo de respuesta que se estima desde que el cliente llega a la copistería hasta que recibe el resultado de su demanda**. Como se ha dicho anteriormente, hay copisterías que sí han querido integrar en sus servicios herramientas tecnologías que pueden ayudar en menor o mayor medida a agilizar este proceso, como poner en disponibilidad de los clientes un ordenador en la tienda desde el cual pueden enviar los documentos al empleado de la copistería, sin embargo, la cola de impresión sigue siendo la misma y el proceso de pago también.

A raíz del análisis de esta problemática relacionada con las copisterías, se puede plantear una solución tecnológica que sea capaz de aportar la reducción de tiempo necesaria en el proceso.

2.3 Planteamiento y motivación de una propuesta

La transformación digital de un comercio no es un proceso sencillo, ya que se debe realizar un análisis y una estrategia a nivel de empresa, **identificar los objetivos, las ventajas y los posibles riesgos que pueden surgir**. Hay diversidad de tecnologías y metodologías para llevar a cabo este proceso, un procedimiento que se explica en [Lom15] de forma que se facilita la comprensión de este cambio de modelo de negocio y productivo.

La productividad es un punto clave en el negocio de las copisterías, pues los beneficios dependen directamente de ella. El desarrollo de un sistema software online capaz de aumentar la productividad del negocio y de sus empleados además de reducir los problemas mencionados anteriormente, como **la reducción del tiempo empleado en la producción**, es apostar por la innovación contemplando una automatización del modelo tradicional.

Las tecnologías empleadas para el desarrollo de este sistema no son lo más importante, sino **la reproducción e integración del servicio de impresión de las copisterías** a través de una aplicación web. Es fundamental que el proceso tradicional no solo se vea reflejado en el funcionamiento del sistema, sino mejorado de forma que atraiga a más clientes y mejore el nivel de calidad del servicio.

El planteamiento y desarrollo de DYN Print, el sistema discutido en este trabajo, se atribuye a la necesidad de las copisterías de combatir contra la problemática expuesta. Un proyecto software que sea capaz de dar a los clientes la posibilidad de ahorrar tiempo y realizar sus pedidos en cualquier momento y desde cualquier lugar. Por otro lado, que el sistema ofrezca a las copisterías una gestión más automática y fluida de los pedidos que reciban, así como una posibilidad de reducir el tiempo de cola de la impresión y en la gestión de los correspondientes pagos.

El ofrecer este servicio de forma online sería **complementario al servicio que se ofrece normalmente en las copisterías**. La prioridad que tienen los clientes que van a la copistería seguirá existiendo, pero reduce la abundancia de clientes y de pedidos consecutivos que puedan surgir en momentos determinados.

El proceso de pago también es un factor que se ve beneficiado. La gestión de un **saldo virtual sustituiría el pago tradicional de los pedidos**, de forma que daría independencia al cliente en la realización de múltiples pedidos sin necesidad de preocuparse.

DYN Print es un sistema cuyo desarrollo estará enfocado principalmente a su uso a través de dispositivos móviles. Como ya se comentó, el nivel de transacciones a través de este dispositivo crece rápidamente y aprovechar esta situación sería beneficioso para aumentar la productividad de las copisterías. La mayoría del público dispone de un dispositivo móvil que utiliza a diario, por tanto, disponer de este servicio a través de él podría aumentar el nivel de satisfacción del cliente.

2.4 Estructura del documento

A continuación, se listan los capítulos contenidos que componen este documento junto con una breve explicación de lo que se podrá debatir en ellos:

- **Capítulo 2 - Objetivos:** Tanto el objetivo principal de este proyecto como los objetivos más específicos se detallan claramente en cuanto al servicio que ofrecerá DYN Print.
- **Capítulo 3 - Estado del arte:** Se presenta una recopilación y un análisis de los conceptos y las tecnologías que se integrarán en el desarrollo de este proyecto. También incluye una comparativa y análisis de las aplicaciones similares a este proyecto que actualmente están en funcionamiento.
- **Capítulo 4 - Método de trabajo:** Las metodologías de planificación y de desarrollo que se van a implementar en el desarrollo de este proyecto se explican en este capítulo,

junto con las tecnologías hardware y software que es utilizarán.

- **Capítulo 5 - Arquitectura:** Este capítulo se centrará en el diseño de la arquitectura del sistema y su implementación centrándose en las diferentes funcionalidades que ofrece. Además, se presentan diagramas que ayudan a la comprensión de su estructura en cuanto a módulos, modelo de datos y clases implementadas.
- **Capítulo 6 - Resultados:** Los resultados obtenidos tras el desarrollo se presentan en esta aplicación. Se presenta tanto un análisis de las funcionalidades completadas junto con capturas de la aplicación. También se incluye una descripción de las fases de trabajo que se han llevado a cabo.
- **Capítulo 7 - Conclusiones:** Para finalizar, en este último capítulo se analiza el cumplimiento de los objetivos del proyecto, se plantean diferentes ideas de mejora para el mismo y además incluye una conclusión personal en cuanto al desarrollo de este trabajo de fin de grado.

Chapter 3

Objectives

AFTER detailing the focus and motivation of this project, this chapter specifies the objectives to be achieved. First, a general objective of the system is synthesised, followed by objectives relating to other more specific aspects to be achieved.

3.1 General objective

The development of this degree project is inspired by the necessity of having an alternative to the traditional system of the copy centres. DYN Print has as principal objective to provide an increase of the copy centre's productivity and of the client's satisfaction level thanks to the automation of the traditional process of printing files, by replacing it with an automated online process. The printing process through this kind of system will assume an enhancement of the problems related to the traditional one.

3.2 Specific objectives

Above-mentioned the general objective of this project, consecutively the necessary sub objectives to accomplish it are going to be specified:

- **Time reduction:** time spent in printing documents in a physical copy centre could be reduced **thanks to the digitalization of the process**. On one side, the client will experience a reduction of the time which he spends in the copy centre queue, the time he is waiting for the documents to be printed and the time he spends to pay for the service. On the other side, the copy centre employees will feel a reduction of the time they spend doing their job too. They will save time when selecting the documents from the storage device of the clients, selecting the printing configuration, waiting for the printers to be ready for printing new documents and preparing the change of the money for the client.
- **Intuitive interface:** it is essential that the web application offers to the users an intuitive and easy experience when they use it. To accomplish that goal, the application must have an interface which **satisfies basic levels of usability and affordability**, where the important elements are emphasised. The users must know unfailingly how to proceed to complete the order they are performing or how to know where to find

their balance quantity, their order history and other useful functionalities of the system. The interface does not have to present complications for users in order not to produce frustrations when they want to upload documents and to choose the printing configuration. Thus, the users who give up during the process and an application which offers a bad user level experience are avoided.

- **Scalability:** there is a proposal to offer the DYN Print services not to a single copy centre, but to a lot of them simultaneously. **To manage the needed scalability to serve all the requests done by the clients of different copy centres**, it is necessary to develop the system with an infrastructure capable of responding to all the requests in a reasonable time and with the lowest number of availability failures. Thus, the scalability level must grow up while the number of requests increase, and the level of availability must satisfy the use of the application.
- **Proactiveness:** DYN Print is a system which needs to support a notification system in order to maintain all the users aware of how their orders are being processed. This project claims a **proactive role to notify the users** when their orders have been processed by the copy centre and when the orders are ready to be collected by the clients. Therefore, the copy centre employees will be notified when new orders are performed to provide the service as soon as possible to satisfy their clients.
- **Multi-device support:** Since the project is based on a mobile-first development approach, because the DYN Print application will be mostly used through a mobile phone, it is critical to **elaborate a responsive interface which is adapted to the mobile screens**. To accomplish this level of responsiveness by the application, the DYN Print interface is going to be oriented from the beginning to mobile screen dimensions and the usage tests are going to be performed with a mobile phone device.

Chapter 4

State of the art

To go through this degree project, a previous investigation about the printing process which is used in physical copy centres, about other web applications which offer a similar service and about the technologies and the tools which are pretended to be used in order to develop this project has been necessary. In this section, **all concepts acquired through this investigation are defined** and they are analysed highlighting the most influential points for the degree project. The figure 4.1 presents a conceptual map with the mainly terms of this chapter.

4.1 Web technologies

Nowadays, what we know as the Internet has become what it is thanks to a long process which has not been discovered and developed by a single person. The group of **pioneer people which are responsible for the Internet development**, which is almost connecting all the world, is formed by Lawrence Roberts, Robert Kahn, Vinton Cerf and Tim Berners-Lee. Among the researches and the work done by these people, the technologies that take part in almost every web application project can be found: the ARPANET[1] network, the TCP/IP protocol[2], the HTML[3] mark language, the Hypertext Transfer Protocol (HTTP)[4] protocol and the uniform resource locator URL[5].

Thanks to the set of these technologies, the Internet has experienced many transformations and changes in order to become the system which, according to the Premio Príncipe of Asturias of Technical and Science Investigation[6], is changing the world and the way it communicates and improves at a scientific and social level. In [CJD16], **the evolution and the socio-economic impact that the Internet has given to the society** from the day it emerged until the recent years are specified. At a more detailed level, it mentions topics like the things which were expected before the Internet became a stable technology, its history, its financial, social and political impact and the possible risks and regulations.

[1] http://www.walskium.es/magazine/tecnologia/arpanet-el-origen-de-internet/

[2] https://docs.oracle.com/cd/E19957-01/820-2981/ipov-10/

[3] https://developer.mozilla.org/es/docs/Web/HTML

[4] https://developer.mozilla.org/en-US/docs/Web/HTTP

[5] https://developer.mozilla.org/en-US/docs/Learn/Common_questions/What_is_a_URL

[6] http://www.fpa.es/es/premios-investigacion-cientifico-tecnica

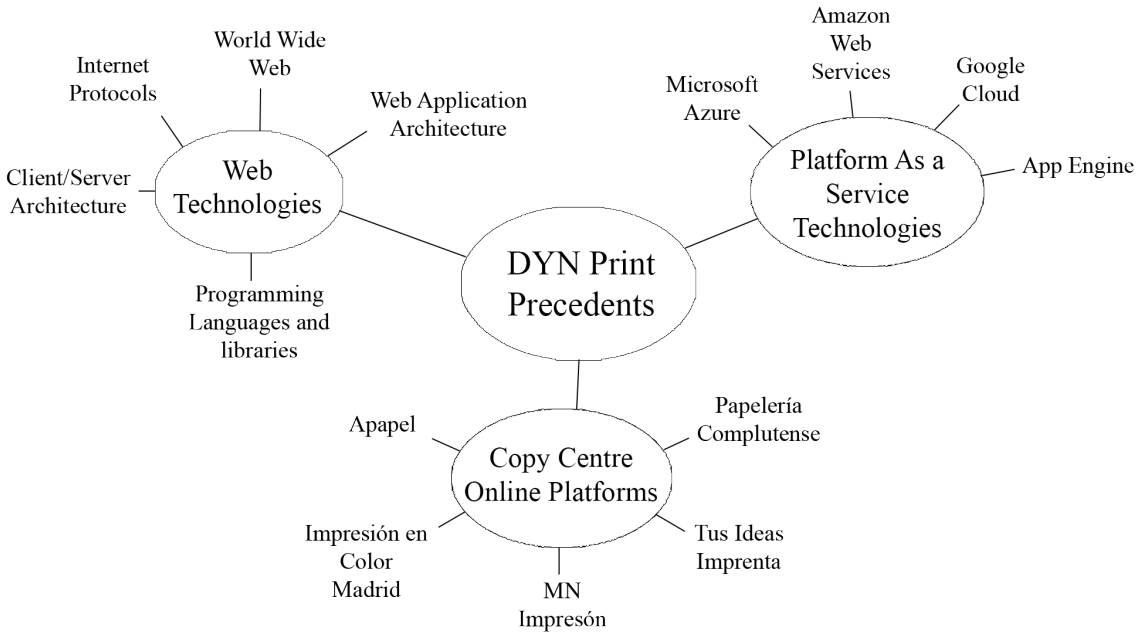

Figure 4.1: Conceptual map of the precedents.

The **Internet communications are based on the TCP/IP protocol** and it is composed by a set of levels which are similar to the OSI model, as it is shown in the figure 4.2. This protocol allows the information exchange independently in which system the information is stored. TCP/IP contains a list of well-known protocols like the File Transfer Protocol (FTP), the Domain Name System (DNS), the HTTP protocol, the Address Resolution Protocol (ARP), the Dynamic Host Configuration Protocol (DHCP) or the Internet Control Messaging Protocol (ICMP). These protocols belong to different levels of the TCP/IP model depending on their functionalities. The concepts and the knowledge related to the TCP/IP protocol are demonstrated in [For09].

4.1.1 World Wide Web

The World Wide Web, what everyone knows as Web, is a hypertext document distribution system which is accessible from the Internet through a web browser. **This technology was developed by Tim Berner-Lee**[7] while he was working at the European Organization for Nuclear Research (CERN)[8], between 1989 and 1990. Originally, it was a project for the use of the company, but later the use of the World Wide Web became free for everyone which could access it.

The history of the Web is specified in detail in [TM01], which is written by the very inventor of the Web, Tim Berner-Lee, and where he tries to explain the functionality of the technology and how it must be understood, from its commercial proposes and its social

[7] https://www.w3.org/People/Berners-Lee/

[8] https://home.cern/

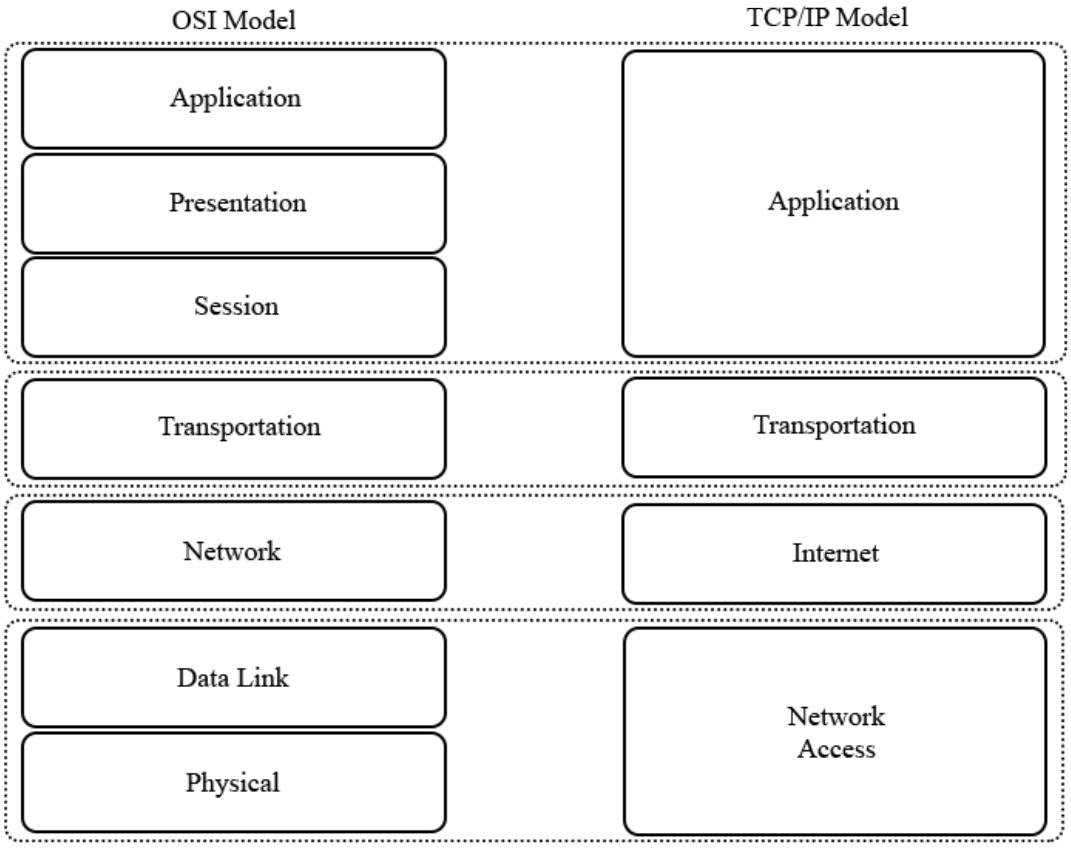

Figure 4.2: TCP/IP model vs OSI model

forces.

As it was mentioned above, to access these hypertext documents where the multimedia information we need is shown, it is necessary to use **web browser**. A web browser is a software application which allows a user to access the information stored at the World Wide Web. Every web page and its multimedia information are identified by a URL, which allows the web browser to display that single web page that the user wants to see.

The most popular web browsers that have been recently experimented a huge progression are Google Chrome[9], Mozilla Firefox[10] and Safari[11]. These web browsers are available at desktop computers, mobile phones and tablets. The **communication between a web browser and the server where the required resources are stored** is performed through the HTTP. It downloads the HTML web pages from the server and display them in the screen. The navigation between the displayed web pages is done through the hyperlinks.

[9] https://www.google.es/chrome/index.html
[10] https://www.mozilla.org/es-ES/firefox/new/
[11] https://www.apple.com/es/safari/

Nowadays, the web browsers are constantly receiving updates to ensure the security of the personal and navigation information and to integrate new functionalities which offer the users better ways of interacting with the web pages.

4.1.2 Client/Server architecture

There is a specific type of network architecture called client/server architecturem whose characteristicas and funtionalities are explained in [A.92]. In this kind of architecture, **every computer or process in the network is a server or a client**. A server can be distinguished from a client because it delivers and manages most of the resources and services which will be consumed by the client. The client can be a web browser, which is prepared to maintain a communication with the server through the HTTP protocol. When a user tries to access a web page with its URL through a web browser, it sends a request to the server to retrieve the HTML document which specifies the web page and the functionalities programmed in the scripts of the web application.

This communication is performed between the client and the server. The server can interpret the requests sent by the client. Depending on the kind of request, the server will manage determined resources and it will provide to the client the necessary services. Some advantages of this architecture are explained in detail in the following list:

- Diverse processors in the network can execute different processes of the application, accomplishing a desired level of concurrency.
- It provides a capacity of increasing the number of clients which are requesting the server services without actually affecting the performance of the application. This is called **horizontal scalability**.
- Also, it can provide **vertical scalability**, which consists in migrating a server which needs a higher velocity and performance in order to attend the client requests.
- The client can access the server information and services independently of where the server is installed or is deployed.

This architecture allows to provide a functionality at three different levels:

- **Presentation logic:** it is in charge of the application input and output data. Some of their functions are: to obtain information from the user, to send the user information to the application logic for processing it, to receive the processing results and to present them to the user.
- **Application logic:** it oversees managing the data at the processing level. It is like a bridge between the data and the user. Some of their functions are: to receive the input data from the presentation logic, to interact with the data logic to execute the business rules that the application must follow and to send the processing results to the client.

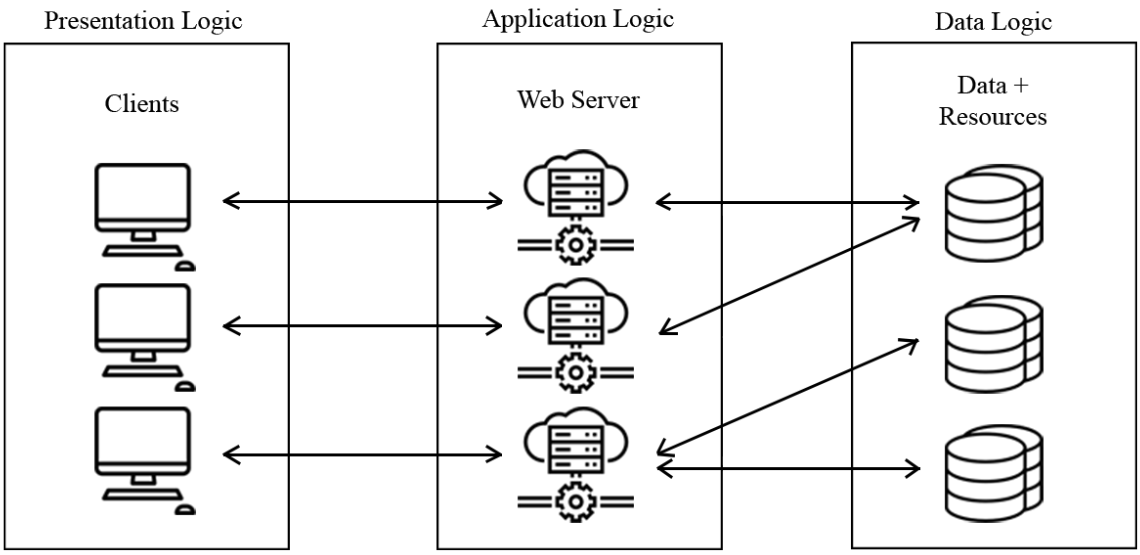

Figure 4.3: Client/Server architecture levels

- **Data logic:** it is in charge of managing the data at the storage level. Some of the functions are: to storage the data, to retrieve the data, to maintain the data and to ensure the data integrity.

In the figure 4.3, the arrangement of the three levels and the communications among them are represented.

4.1.3 Web applications and their architecture

A web-based application is a **special type of the client/server architecture**, where the client, the server and the HTTP protocol are standardized, and the application programmers do not have the need of creating them.

The user will interact with the web client to request resources of the web server through the HTTP protocol. The client is usually formed by HTML code which is the structure of the web page and by scripts implemented in Javascript. The **web browser mission is to interpret the HTML pages and the resources retrieved from the web server** (images, sounds, videos, etc...). The web server is permanently waiting for client-side requests.

The full process, since the user requests a web page until the web browser shows it, is the following one:

1. The user specifies the URL in the web browser.
2. The client establishes a connection with the web server.
3. The client requests a web page or the desired resource.
4. The web server sends the web page or the resource, if there is not an error.

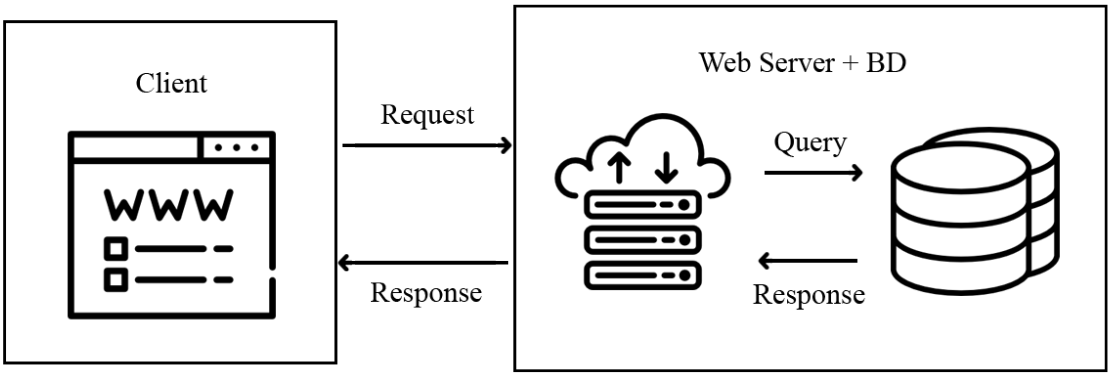

Figure 4.4: All in one server architecture

5. The client interprets the web page and identifies the resources that it needs from the server.
6. The connection between the client and the server is closed.
7. The web page is showed to the user.

The connection only lasts until the transmission of the data ends because **it is important to avoid wasting bandwidth while the web page is received**. When the user wants to access another web page or resource, then another connection is opened. Also, **every connection between the client and the server is independent from the others**.

The design of this web application architecture has been defined through the years, whose process is explained in detail in [R.02]. The web applications can have different architectures depending on the structure of the server. The most common architectures are:

- **All in one server:** it is composed by a single computer hosting the HTTP service, the application and data logics and the data. A scheme of this type is presented in the figure 4.4; the web server and the database are together and they maintain a communication with the client.
- **Separated data server:** considering the previous architecture, this one separates the data logic and the data from the HTTP service and they are managed in an independent database. In the figure 4.5, the separation between the web server and the database can be observed. The web server performs queries to the database and it transfers the response to the client.
- **All in one server with an application service:** this architecture adds an application service together with the HTTP service and the database. The web server, the database and the application service are together and they keep a communication with the client.

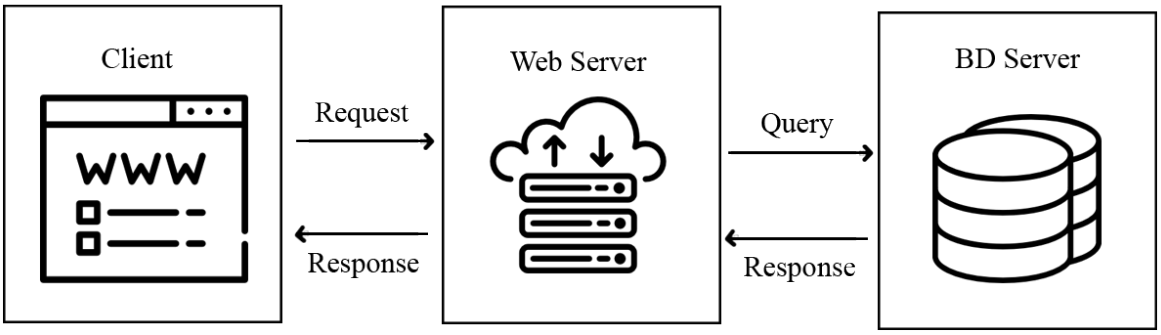

Figure 4.5: Separated data server architecture

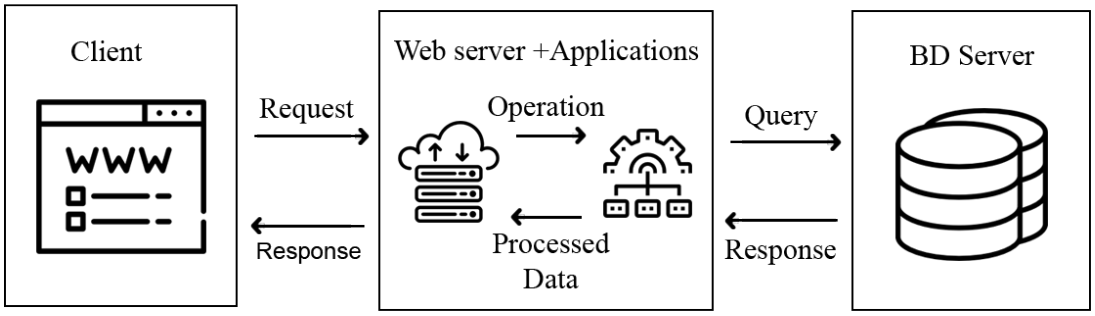

Figure 4.6: Separated data server with an application service architecture

- **Separated data server with an application service:** considering the previous architecture, the data logic and the database are separated from the HTTP service and the application service. As it can be observed in the figure 4.6, the database is separated from the web server and the application service. In this case, the application service is the one which performs the queries to the database and it transfer the processed data to the web server, which is responsible for retrieving it to the client.
- **All separated:** the functionalities of the HTTP service, the application service and the data logic are separated. The figure 4.7 presents this type of architecture, where the web server, the database and the application service are isolated from each other, but they keep the same communication as the previous architectures.

4.1.4 Programming languages and libraries

Nowadays, at the client-side of a web application, there is a programming language which is predominant over the rest: Javascript. In the development of the client-side, **it is necessary a programming language that implements the communications with the server-side**, to perform requests to the resources or the services and to send the information that the server-side needs to process.

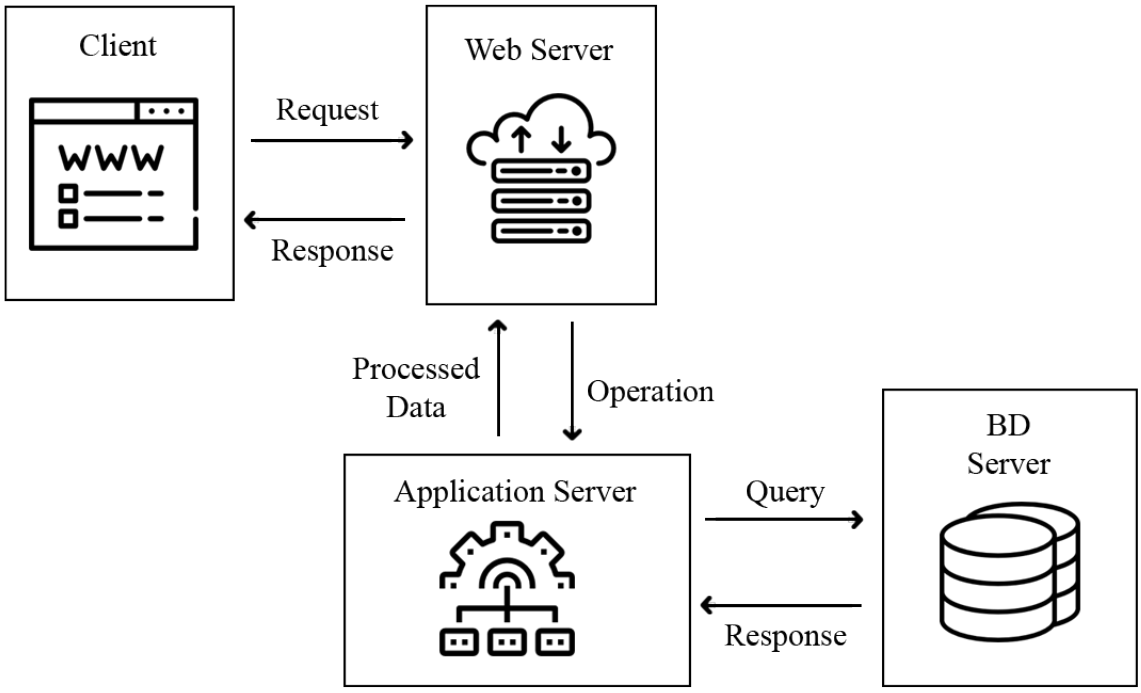

Figure 4.7: All separated architecture

Also, web applications need an interface for the web page which will be displayed through the web browser. The technologies mainly used for this purpose are the mark-up language HTML, to organize the structure of the web page, and the graphic design language CSS, to define the appearance of the web page.

Javascript

Javascript[12] is a programming language which is known as the scripting language for the development of web pages, but it can be used for other non-browser environments. **Dynamic, prototype-based, supporting object-oriented, imperative and declarative** are some of the principal characteristics that define this programming language.

It has first-class functions, which means that defined functions can be passed as arguments to other functions, can be returned as values from other functions and can be assigned to a variable. A prototype-based programming is a type of object-oriented programming; thus, the inheritance concept is used when referring to the behaviour of the objects. There are objects defined as prototypes which are extended objects from the original one. All the characteristics and elements of this programming languages are explained in deep in [Fla11].

Javascript is mainly used to implement the behaviour of the web page, **creating functions which respond to the events that occur from the user interaction**. The functions that can be covered are, for example, to validate forms and to retrieve information from them,

[12] `https://www.javascript.com/`

to create objects to support the functionalities of the client-side, to manage variables in the session or local storage and to send data to the server-side with the JavaScript Object Notation (JSON) notation.

This programming language is popular because it has **many useful libraries to the development of the web application**. The libraries can be easily installed through the npm[13] package management or they can be included in the Javascript script through a Content Delivery Network (CDN) link. The most significant libraries and frameworks which are used in the recent years are:

- **React[14]:** this library is focused in the **user interface development**. It provides an easy way of developing the interfaces and it defines different states views of the application. The library is flexible in terms of adapting the data to the components changes. Also, the code implemented by React is simpler to understand and easier to debug. The components of this library are encapsulated in such a way that allow to exchange data without worrying about the state of the Document Object Model (DOM).

- **Angular[15]:** this framework is used to **build mobile and desktop web applications and native mobile and desktop applications in HTML and Javascript**. It has a core and optional functionalities which can be import into the application. The applications built in Angular are composed by modules which enable the use of Bootstrap. There is a main module called the root module and optional modules called the feature modules. Then, there are components which define the views of the application. These vies are a set of screen elements with are related to the data or logic of the program. Likewise, the components use the services, which are in charge of providing the functionalities needed that are not part of the views. The codes implemented in Angular assure a high level of optimization and good performance for the application.

- **Ember[16]:** it is a Javascript framework which **takes care of the heavy implementations that are usually tedious**. The key is to give common tasks done to save time when the developer is programming the application. So, the huge advantage of this framework is to save time on the development process, allowing to focus on the interface or the additional features.

- **Three.js[17]:** this library makes easy **to create and to use 3D components on the application**. It provides the tools to display the 3D scenes, to create 3D scenes directly from the code and rendering them. Also, it allows to create animations with the objects of the scene.

[13] https://www.npmjs.com/
[14] https://reactjs.org/
[15] https://angular.io/
[16] https://www.emberjs.com/
[17] https://threejs.org/

- **Axios**[18]**:** This library creates an Application Programming Interface (API) to **perform HTTP request from the client**. Javascript already provides functions to make request to server or external resources, but Axios integrates those functions and provides an easier way to make the requests, with only calling a single function with some simpler parameters. It allows the application to cancel requests, to intercept requests or responses and to transform the JSON data.
- **PDF.js**[19]**:** it is a **Portable Document Format (PDF) viewer for a Javascript application** and it is built with HTML5. It provides a way of parsing and rendering PDF documents in web platforms. This library has different layers: the core layer, which parses and interprets the PDF documents, the display layer, which provides the document information, and the viewer layer, which allows to display the document content in the interface of the application.

HTML

The HyperText Mark-up Language is used to elaborate the structure of a web page. It defines the contents like the text, the images, the videos and other media. External scripts and resources can be included to complement the functionalities of the web page through a reference to the location of the script files. It is important to know that **HTML defines the content of the web page, but not its functionality**. The implementation of the different functions available in the HTML5 language are specified in [LSA11].

The web browsers are in charge of interpreting the HTML programmed pages, which are usually considered as plain text when they are submitted through the communications with the server-side.

HTML is programmed through labels and it has some essential components: the elements and their attributes, the data types and the document type declaration. The distribution of these components is illustrated in the figure 4.8. The elements have two basic properties: the attribute and the content. The structure of the HTML elements is shown in the figure 4.8. Some of the special elements are:

- **<head>:** it defines a section of the document where the metadata is provided to the application. It usually includes the title of the web page, with the element <title>, links to necessary scripts and to stylesheets.
- **<body>:** it represents the web page content itself. This element is unique in every HTML document.
- **<article>:** it represents a content which can be reused in the application, so it is independent of other elements. It can be a forum message, a magazine article or a

[18] https://github.com/axios/axios
[19] https://mozilla.github.io/pdf.js/

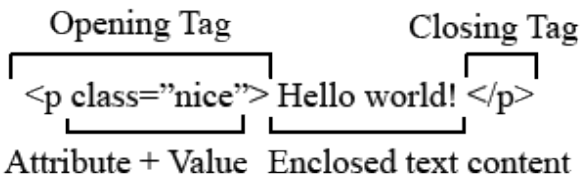

Figure 4.8: Components of a HTML element

newspaper article.

- **<section>:** it defines a generic section of the document. It allows to determine which contents correspond to one part of the document or to another one. Theoretically, it is used to structure the document.
- **<p>:** it is used to structure the text of the document in paragraphs.
- ** :** it divides the document into sections or it can group the contents.
- **:** it is like a container in line, it serves up to apply a style to the text or group elements in line.
- **<img>:** it represents an image inside the document.

The attributes of the elements are additional values which provides a configuration or a behaviour to them. Not all types of attributes can be applied to every element. Therefore, there are attributes that can only give a configuration to certain types of elements. For example, the attribute align can be applied to many types of elements, such as tables, bodies, images, columns, or rows, and the attribute language can be only applied to the script element. A special attribute of the elements is the ID, which is a unique identifier for the element throughout the document. Another fundamental attribute is the class attribute, which specifies a class for the element, and that class can be applied to different elements so that the class style is applied to all elements with it.

CSS

The Cascading Style Sheets or CSS is the language which defines **how the web pages are presented in terms of the visual style**. The styles defined with CSS can be included in the HTML files through the proper command and then, the style will be applied to the web page. Also, every element of the HTML file can have its own style programmed with CSS. The CSS3 technology applied to web design is explained in its web documentation, but a more extensive and detailed version can be found at [Gas11].

A CSS file is structured by blocks with some elements: the selector, the declaration, the properties and the values. An example of a block is shown in the figure 4.9. An example of a CSS block is shown in the figure.

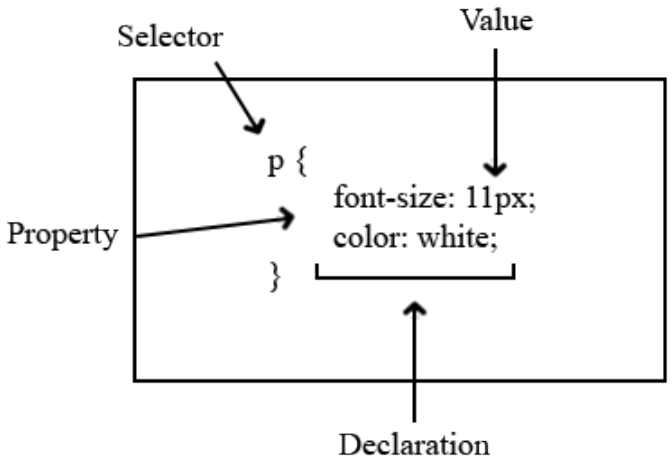

Figure 4.9: Scheme of a CSS block

- **Selector:** it specifies which element or elements from the HTML file will be affected by the style define inside that block.
- **Declaration:** it defines a single rule that specifies which property is to be styled with the defined value.
- **Property:** it is part of the declaration. Each element of the HTML document may have different properties, as not all elements have the same properties. Some examples of properties are: color, width, height, border, font-size, etc...
- **Property value:** after the property, the value is specified. The value chooses the appearance for such a property from the many possibilities that exist.

The selector of the block has different types. The above-mentioned selector referred to the types of elements in the document, but a higher level of specification can be reached. The **element selector** is simply specified with the element name, for example, to affect all paragraphs the selector would be p. The **identification selector**, which corresponds to the ID attribute of a document element, applies the style to the element with that ID. The **class selector** applies the values to all elements of the document that have the class defined as an attribute. Finally, the **attribute selector** applies the style to all elements that you specify a value for that attribute.

Bootstrap

Once Javascript, HTML and CSS have been briefly explained before, it is important to mention the web framework called Bootstrap[20]. **This framework is a set of functions that involves all the above-mentioned technologies**. It provides templates that contains specialized design form elements like the typography, the forms, the buttons, the menus and other

[20] http://bootstrapdocs.com/v3.3.0/docs/

	Chrome	Firefox	Internet Explorer	Opera	Safari
Android	✔ Supported	✔ Supported	N/A	✖ Not Supported	N/A
iOS	✔ Supported	N/A		✖ Not Supported	✔ Supported
Mac OS X	✔ Supported	✔ Supported		✔ Supported	✔ Supported
Windows	✔ Supported	✔ Supported	✔ Supported	✔ Supported	✖ Not Supported

Figure 4.10: Bootstrap supported web browsers

components of the HTML document. The important characteristics of these templates is that they all are responsive, and it is mostly used for mobile-first development approaches for the web.

Bootstrap can be included in applications via CDN links. To work properly, it is essential that the JQuery framework is included in the project. Bootstrap is designed to fit the newest desktop and mobile browsers. For older browsers it may be a different or poor design. Bootstrap is designed to fit the newest desktop and mobile browsers. For older browsers it may be a different or poor design. In the figure 4.10 you can find the browsers that support and those that do not support the framework.

Bootstrap is characteristic for its grid system. This system provides the responsive component of the framework because it **automatically scales according to the size of the device viewport**. This system consists of 12 columns that occupy the screen of the device, and the content of the web page must be distributed among these columns, so that each content must be assigned several columns, proportionally, without exceeding the maximum of 12 columns. Within each column rows can be defined, and it can be more than one.

In addition, the columns have **different sizes to suit the possible screens of the devices**. Available sizes are, from smallest to largest: xs, which fits any container width, sm, whose width is 750 pixels, md, whose width is 970 pixels and lg, whose width is 1170 pixels. Also, columns can be nested in different sizes. There are several predefined components in Bootstrap that feature modern designs and easy accessibility. These components can be used in the HTML document by simply defining the name of the class that corresponds to it in the class attribute of the element.

4.2 Platform As a Service technologies

Since the discovery of the term Software as a Service, there are a lot of facilities which help the companies to develop applications. It is a **software distribution model which allows the companies to save their data and their resources in an external server**, and its maintenance and its support are managed by the company responsible for providing this kind of service. It is an Internet-based strategy because the company that hires this service

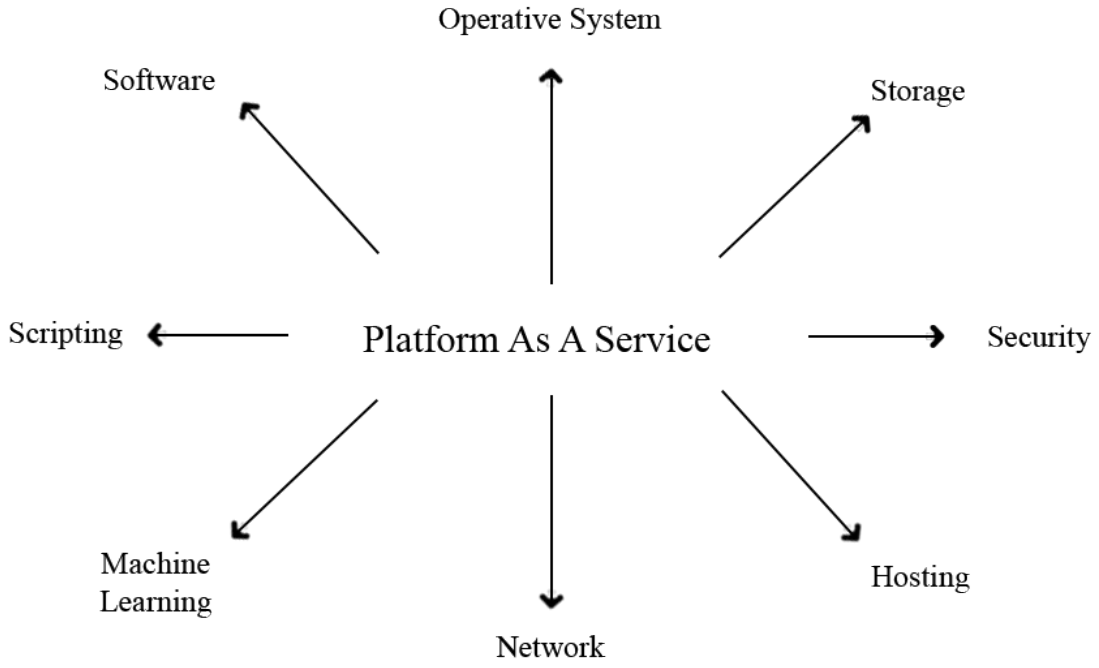

Figure 4.11: PaaS provided services

will manage and access its data, its resources, its processes and its results through a web browser from anywhere, independently that **the server is working in a different part of the company itself**. According to [AD13], there are highlighted advantages regarding the Software as a Service (SaaS), both for the clients and the developers of the application:

- The clients does not have to install any kind of application. Also, the hardware of the client device is not relevant to the performance of the system.
- The clients does not need to be worried about the data stored of the system, about doing security copies or about losing the data due to failures of the server.
- It is easier to use SaaS when a group of users need to interact with the system in a collective form.
- A copy of the software is executed in a server with a strict operative system chosen by the developer, thus compatibility problems when files are distributed are avoid.
- There are adaptability and flexibility when developers can update and change their software whenever they want as long as external APIs are not affected and without disturbing the functionalities to the clients.

As a conclusion, SaaS provides a way of developing applications and the developers of them do not need to worry about developing, hosting, updating or maintaining servers or storage systems.

Companies in charge of providing Software as a Service has take a step forward and has developed what is called **Platform as a Service**, which not only provides what it is above-mentioned, but also it handles tasks from editing code to debugging, deployment, runtime and management of the system. It is usually used as a web-based application-development platform and its functionalities and its main advantages are specified at [G.08].

PAAS provides preconfigured functionalities to the developers, these functionalities are presented in the figure 4.11. They can choose which functions they want in their web application depending on their needs and they can disable those functions that are not needed in the development of the application.

There is not infrastructure which have to be managed by the developers, it is in charge of the PAAS providers, who also offer technical support and they are always updating their technologies to provide new functionalities which improve them. Sometimes, they try to collaborate with the developers to take new ideas for the improvement. Nowadays, some of the relevant services provided by a PAAS are: operating system, scripting environment of the server, database management system, server software, technical support, data storage, network access, design and development tools and hosting.

AS every cloud service, the **PAAS services provided are usually billed to the client depending on the use of the service**. There is a cost reduction to the providers since the physical infrastructure is shared among the clients.

Hereunder, some of the most relevant PAAS in the recent years are presented.

4.2.1 Windows Azure

Microsoft Azure[21] is a cloud service integral group which offers to the developers and the Information Technology (IT) professionals the opportunity to create, to implement and to manage applications through the Microsoft global network, a process which is extensively explained in [Ric99].

This Microsoft service provides an increase on the developer's productivity due to the flexibility on selecting the operating system, the programming language, the data base type and the type of the device where the application will work. The available programming languages are Node.js, Java or .NET and the application can be implemented in development tools like Visual Studio, Visual Studio Code or Visual Studio Mobile Center, which usually include automated functions to interact and to manage the application through Microsoft Azure directly from the tool.

Azure Cloud Services can also provide many useful services to the application. Depending on the category that the developers need to implement in their application, there are a lot of functions that Microsoft Azure makes available to their clients, some of them are pre-

[21] https://azure.microsoft.com/es-es/

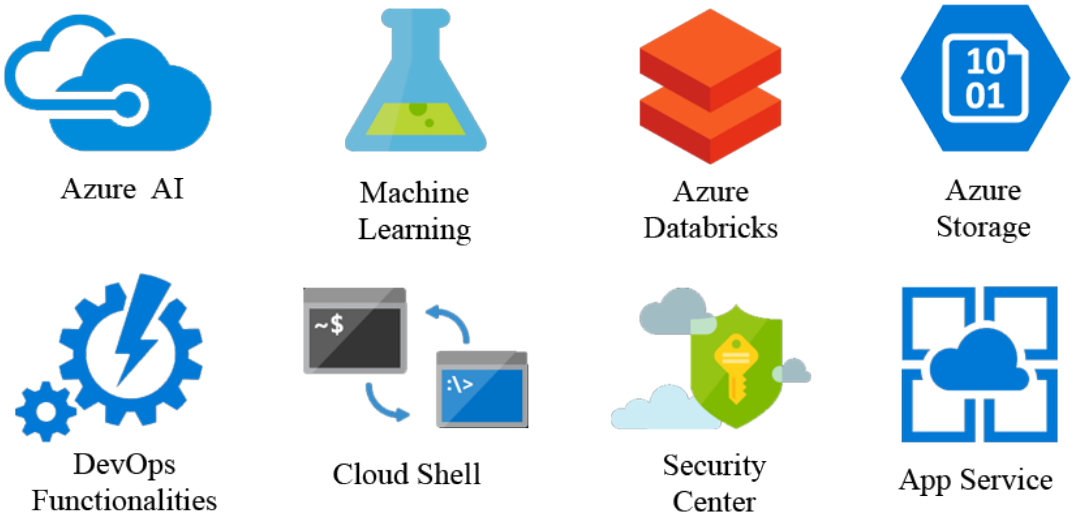

Figure 4.12: Some of the Microsoft Azure services

sented in the figure 4.12. These products and these functions are related to fields such as follows: the Artificial Intelligence (AI) and Machine Learning techniques, the data analysis, the database management, the Development Operations (DevOps) functionalities, the management tools, the mobility and migration management, the data security, the multimedia storage, the Internet of Things, the network functionalities and more.

Some of the successful projects that have bet for using Microsoft Azure are the following ones:

- **Patrus Transportes Urgentes**[22]**:** this company decided to reduce their cost using the cloud functions and the cloud storage instead their physical option.
- **Atento**[23]**:** a company which has invested in the artificial intelligence and the cognitive services of Microsoft Azure to improve the client relation industry.
- **Hospital CIMA**[24]**:** a health institution which bet for the cost reduction migrating their services to the cloud using the Microsoft Azure products.
- **Intermex Wire Transfer**[25]**:** a finance company which had the need of create a digital channel to improve its business and its relationship with the clients.

4.2.2 Amazon Web Services

Amazon Web Services (AWS)[26] from Amazon is a cloud services platform which offers a lot of functionalities related to fields like the cloud computing and the cloud storage. These

[22] http://www.patrus.com.br/
[23] http://atento.com/es/
[24] https://www.hospitalcima.es/es/
[25] https://corporate.intermexonline.com/
[26] https://aws.amazon.com/es/

services can help the companies to have scalable projects and to grow up in a competitive market. [Mur08] demonstrates how applications integrating the Amazon Web Services can be programmed.

Every cloud platform of this kind always tries to provide as much tools and services as possible to the developers. The AWS is ready to offer its services to self-employed workers, to small and medium-sized enterprises and to big corporations, since it provides possibilities to reach the desired scalability level and the amount of storage needed by the company in accordance with its own growth.

Some of the given services which belong to development fields are the following ones:

- **Cloud computing:** it is a service that provides the development of IT activity through the cloud in a secure and scalable manner. This service creates instances of web applications so that they are active for users as long as necessary. The choice of different operating systems and software packages offers a wide range of development options.
- **Cloud datastore:** currently, most companies demand a great capacity of data collection and analysis, trying to achieve a high level of security, simplicity and scalability. This AWS service can offer those features to retrieve information quickly and from anywhere.
- **Business applications:** an outstanding tool of AWS is the communications service it offers to companies, so that they can make calls, videoconferences, chats and share content inside and outside the company. It includes artificial intelligence to assist in online meetings.
- **Mobile applications management:** this service provides the opportunity to create mobile applications quickly and scalable, all from the cloud. AWS ensures quality in the development and services involved in the application. They are available for Android and iOS.
- **Internet of things:** the Internet of Things (IoT) section allows users to connect their home devices to the AWS online network, so that they are safer and easier to interact with. In addition, its operation can be easily monitored from a mobile phone.
- **Development tools:** thanks to this service, it is possible to monitor the resources hosted in the cloud. In addition, it provides a visualization of the entire system implemented.

There are many successfully projects which use the AWS services:

- **The Kellogg Company**[27]**:** it uses AWS to deliver spend analysis and data simulations in minutes.

[27] `https://www.kelloggcompany.com/`

Figure 4.13: Worldwide locations of the Google services

- **Spotify**[28]**:** it uses the AWS scalable IT to support the usage peaks and to launch new features faster to the users.
- **Airbnb**[29]**:** the mobile application of this company is running on AWS and it scales automatically to support the demand of the application.
- **Brisol-Myers Squibb**[30]**:** this company used AWS to build a secure and self-provisioning research portal.

4.2.3 Google Cloud

Google did not last to get into the cloud like Amazon. From Blogger[31] to Youtube[32], Google has applications and tools over all the Internet with different useful functionalities. Google had already cloud tools from editing documents online to printing documents form different devices. Google Cloud[33] is the stake from Google to get a place in the cloud development.

Through the technology that Google Cloud offers, developers can **create applications without worrying about the system infrastructure, the data storage, the database management or the system deployment**. It allows quickness and scalability, which is a huge benefit for applications whose companies are growing over time.

The workload of the Google Cloud applications is distributed over **many different loca-**

[28] https://www.spotify.com/es/
[29] https://www.airbnb.es/
[30] https://www.bms.com/es
[31] https://www.blogger.com
[32] https://www.youtube.com
[33] https://cloud.google.com/

tions of the world, as it can be seen in the figure 4.13[34]. Google Cloud provides functionalities which will extend the development of any application from many different fields:

- **Computing resources:** from the deployment of virtual machines to a platform to develop applications which are totally administrated from the web:
  - **Compute Engine[35]:** this service offers virtual machines which are executed in the Google centres and they are connected through the worldwide network. The instances of the virtual machines have scalability to balance the workload.
  - **App Engine[36]:** this tool allows the developers to design modern web and mobile applications in the cloud. It is explained in more detail in the next section because of its large capacity.
- **Database and storage:** Google Cloud provides different kinds of storage structures to adapt it to the requirements of the application. From MySQL databases to globally-scalable NoSQL databases.
  - **Cloud Datastore[37]:** it is a NoSQL document database which is built for applications that needs automatic scaling and high performance. This database includes automatic transactions and provides high availability of reads and writes of the stored entities. It also has an Structured Query Language (SQL)-like query language to ease the query operations. There is a limit of the size to 1MB for the entity and its properties.
  - **Cloud Storage[38]:** it is a RESTful file storage which avoids the size limitation problem of the Cloud Datastore. However, there are some features which are similar to the Cloud Datastore, like the high performance, the scalable data and the availability of the data.
- **Network:** the Google private network is very important because it connects all the locations where the Google services are working and ready to deliver to the clients. Google Cloud uses a software-defined networking and distributed technologies for these services in order to offer a high-speed, privately and reliably network. Google Virtual Private Cloud[39], Google Cloud DNS[40] and Google Cloud CDN[41] are some of the provided functionalities related to the network.
- **Big Data:** because of the large amount of data that is generated these days, Google Cloud provides functions for managing this highly unstructured, raw data with a large-

[34] https://cloud.google.com/about/locations/
[35] https://cloud.google.com/compute/
[36] https://cloud.google.com/appengine/
[37] https://cloud.google.com/datastore/
[38] https://cloud.google.com/storage/
[39] https://cloud.google.com/virtual-network/
[40] https://cloud.google.com/dns/
[41] https://cloud.google.com/cdn/

scale volume. The BigQuery[42] function allows to query massive volumes of data with an expected speed and the Cloud Dataflow[43] concedes to process and to execute data processing patterns.

- **Machine Learning:** Google Cloud includes an AI which provides modern machine learning services in a scalable, fast and easy way. There are functionalities both for unskilled developers and for proficient ones.

- **Development tools:** a collection of tools and libraries are presented to ease the development process from the web browser or the command line.

  - **Google Cloud SDK[44]:** it is a set of tools to manage the resources and the hosted applications on Google Cloud from the command line. Google Compute Engine, Google Cloud Storage, Google BigQuery and other functionalities can be accessed with it and can be run interactively.
  - **Cloud Tools[45]:** Google Cloud can be incorporated into different Integrated Development Environment (IDE)s, which enable a big range of functions that can be directly run when developing the application without worrying about the command lines. Visual Studio[46], Eclipse[47], IntelliJ[48] and Android Studio[49] are the IDEs where the tools are available.

Some of the successfully projects which have been using Google Cloud services to face the challenging online market are the following ones:

- **Khan Academy[50]:** it enables the company to focus on other tasks which differ from the administration and the deployment of the application.

- **Broad Institute[51]:** because of the inevitable increase of the data produced by their researches, this company needed to a system to professionally and securely manage the data.

- **Rovio[52]:** Google App Engine allows them to launch games without worrying about the servers and the maintenance.

[42] https://cloud.google.com/bigquery/
[43] https://cloud.google.com/dataflow/
[44] https://cloud.google.com/sdk/
[45] https://cloud.google.com/tools/docs/
[46] https://www.visualstudio.com/es/
[47] https://www.eclipse.org/
[48] https://www.jetbrains.com/idea/
[49] https://developer.android.com/studio/
[50] https://es.khanacademy.org/
[51] https://www.broadinstitute.org/
[52] http://www.rovio.com/

4.2.4 Google App Engine

Google App Engine, from now on GAE, is one of the services provided by Google, its speciality is focus on the **development of web or mobile applications** with the advantages of offering the developers all the facilities needed to avoid complications related to, for example, the backend infrastructure, the data storage management, the debugging management or the network issues.

GAE has as set of characteristics that automatically and effectively allows the creation of applications. Some of these principal characteristics are as follows:

- The possibility of using the **most known programming languages** to develop the applications: Node.js, Java, Ruby, C Sharp, Go, Python and PHP.
- GAE provides a Docker container that allows that the **execution time of the application can be flexible** to include any external library or framework in the development.
- There is an **administration environment**, which can be accessed from a web browser by the developers, allowing them to focus on the implementation on the code instead of the infrastructure issues.
- The developers are provided with a tool that can perform the **system debugging**, the **monitorization** of the system state and an **analysis** of the application performance.
- GAE has a system that provides a **version control of the different instances of the application**. The developer can upload and host some versions at the same time and administrate them through the administration environment.
- A **division of the traffic** is handled, where the requests to the different version are separately managed. A/B tests and incremental deploys can be performed thanks to this characteristic.
- Also, GAE provides a **high level of security** for the applications. The developer can define access rules with the App Engine firewall to protect the application. Secure Sockets Layer (SSL) and Transport Layer Security (TLS) certificates are predetermined in every of the applications.
- Finally, GAE uses an increasing environment of the Google Cloud Platform services and development tools, which can be integrate in the applications.

GAE has a **daily limitation of resources usage**. The developers can use the functionalities and the services, but their consumption is bounded by a fee. If the consumption limitations are overcome, then Google will charge the proportional quantity of the consumption.

The development of the application with the company computers is allowed thanks to the installation of the Software Development Kit (SDK) for App Engine. It enables the computers for developing, deploying and managing the applications in GAE. There are **two types of available environments: standard and flexible**. The flexible environment automatically

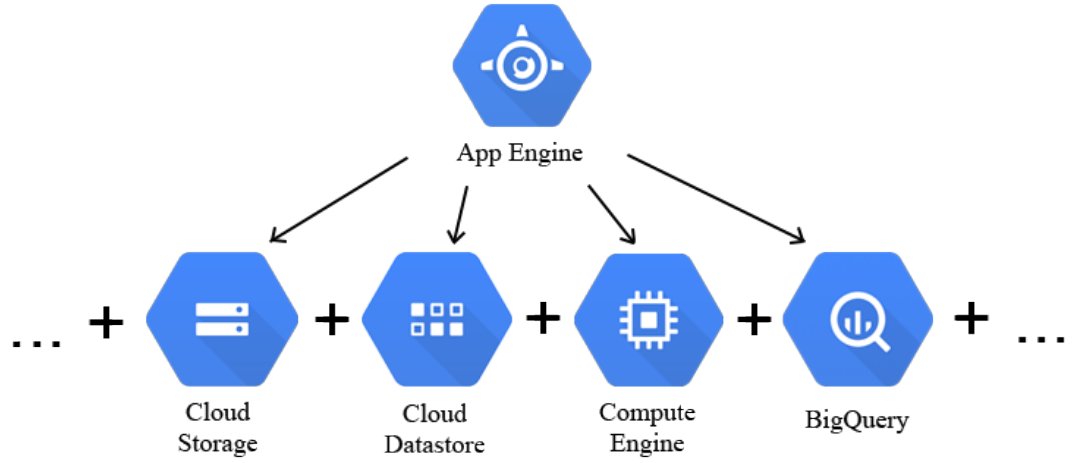

Figure 4.14: Google Cloud functions for App Engine

provides scalability to the applications while the load balance is managed. On the other side, the standard environment provides container instances which are running in the Google's infrastructure. The containers are configured with the runtime of the selected programming language and with the App Engine Standard APIs.

GAE has other functionalities related to the necessities of a web or mobile application apart from the ones defined before: a task queue management where processes can be listed in order to be executed in background, a blobstore management where blob data can be stored and a memory cache service to provide a faster accessibility of the application data.

Google Cloud Platform features can be integrated in all the GAE applications, as it is shown in the figure 4.14. Thus, the developers can be implemented the required requests to those services. For example, if the application needs to store large files, then the application can request the Google Cloud Storage service; if the developer needs to use virtual machines, it can request the Compute Engine service; if the developer wants to use an SQL database, then the Cloud SQL are available to be requested. All these features can be directly managed from the administration environment of GAE.

Some authors discuss an extensive guidance on how to get involved in the development of Google App Engine and all the features it offers in [San12]. By installing the **Google Cloud SDK**, the basic management tools for the application will be available on the developer's computer, as well as the **specific libraries for the programming language** in which he or she wants to develop the application. It also has features for accessing basic services such as Google Datastore or Google Storage. One of the most important parts of a GAE application is the YAML configuration file. This file maps the values of the application so that it defines its structure. In the listing 4.1 there is an example of this type of file, where first the values

related to the application information are specified, such as the programming language and its version, then the mapping of the handlers and finally the libraries are included.

```
application: clock
version: 1
runtime: python27
api_version: 1
threadsafe: true

handlers:
        - url: .*
                script: main.application

libraries:
        - name: webapp2
                version: "2.5.1"
```

Listado 4.1: Example of a yaml file

A very important aspect of GAE is the database model it manages. **It can be understood as a database of objects**. An object within the database is called an entity. Each entity is identified by a unique key. You can even reference other objects by saving a key to another object in an entity. An entity key has two parts: the kind, which is in charge of categorizing the object, and the ID, which will be a random number created by the database to identify it. Once this key has been created it can not be changed again and will be used to access the entity and its properties. As mentioned, each entity has its properties, which can have a variety of types: string, integer, a reference to another object, a date, boolean... As can be seen, the data model offered by GAE does not resemble the generally known relational model. However, this does not mean that it does not provide a quality and unique service.

Another prominent service in Google App Engine is caching, called memcache. The main advantage it offers is the **reduction of time to access the database entities**. Therefore, Google is committed to developing high-performance applications that mitigate the cost of data access through this service, offering a Random Access Memory (RAM) memory that is accessed before the entities. The structure of memcache is none other than key-value pairs. So, the developer only has to define a key and a value. The value can be retrieved with the key.

In a GAE application, it is very common for the client to perform requests to the server, so that the user receives the answer as quickly as possible. There will always be processes in an application that require more time than estimated to perform a functionality. Google App Engine offers a **queue of tasks that run in the background**, so the user does not have to wait for a full run to get a response. A task can be executed independently of the code that has commanded it to be executed. The terms defined are: the producer, who is responsible for creating a task and queuing it, and the consumer, who is the process for carrying out the

task, separate from the producer. When the consumer successfully completes the task, it then removes the task from the queue and leaves room for new tasks to arrive.

Very often an application requires external Internet services. In order to do this, requests need to be made to these services. GAE offers a **scalable and easy way to carry out these requests by using its Fetching URLs service**. This service can make requests using either HTTP or Hypertext Transport Protocol Secure (HTTPS) protocol. It has a restriction on the types of requests you can make, as it only allows the execution of GET, PUT, DELETE and HEAD requests. In addition, the request can transmit parameters, a message body and headers. However, there is also a limit on the size of the request of up to 5 MB. On the other hand, you also have a deadline to receive a response from the required service. An example where the Fetch URL library is used in Python language is shown in the listing 4.2.

```
from google.appengine.api import urlfetch
# ...
        try:
                newsfeed = urlfetch.fetch('http://ae-book.appspot.com/blog/atom.xml/',
                    allow_truncated=False, follow_redirects=False,deadline=10)
                newsfeed_xml = newsfeed.content
        except urlfetch.Error, e:
                # Handle urlfetch error...
```

Listado 4.2: Example of the URL Fecth API in Python

4.3 Copy centre online platforms

As detailed in the chapter 1, the e-commerce has obtained an important role in the current market and it is providing huge benefits for the companies. A lot of Spanish copy centres have taken the plunge into the e-commerce and they have decided to expand their businesses betting on the web technologies and the Internet. With that, they can offer their services through a multi device available web application. Some of the cases which are more relevant for this degree project are Apapel, a start up from Murcia, or TusIdeas, a company from Madrid which goes beyond an online copy centre.

Hereunder, different online platforms which provides a complete printing service are presented, paying attention on the characteristics which assume a source for the design and the basic funtionalities concepts of DYN Print.

4.3.1 Apapel

Apapel[53] is an online copy centre from Murcia which counts on the prize of the online shop with the most innovate service, from the Cecarm Prizes 2017[54]. Below is a detailed description of what the main aspects of your web application look like and how they work:

[53] https://www.apapel.es/

[54] http://www.cecarm.com/premios-cecarm

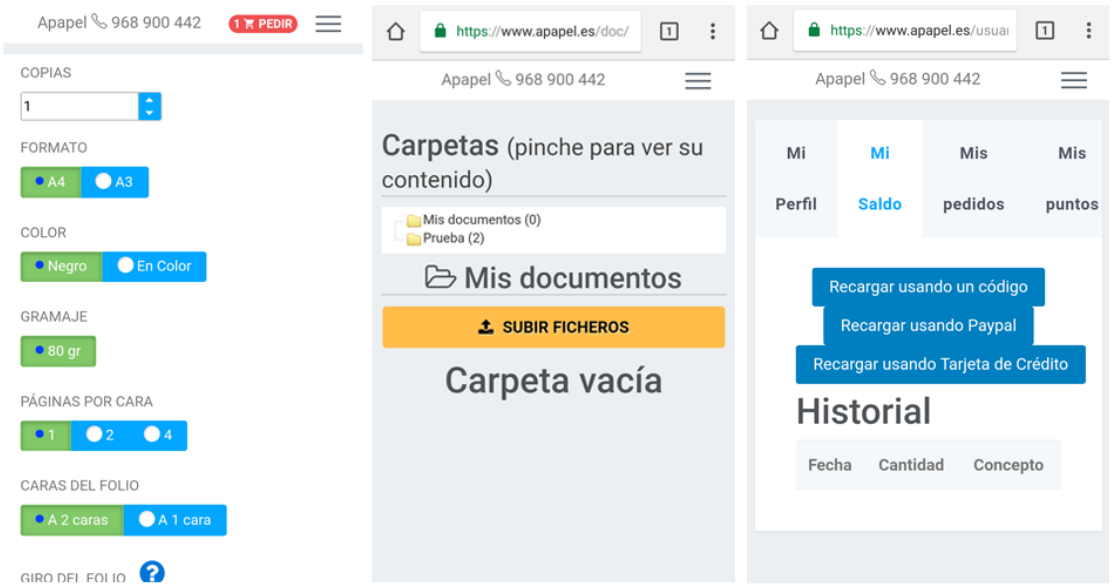

Figure 4.15: Apapel responsive interface

- **Design:** the application counts on a principal page where all the relevant information and the functionalities are presented, like the price of the prints and the copy centre addresses. Furthermore, a button to start directly the printing process is outstanding. There is an available menu which has several sections with the information that has to do with how to perform an order through the page, with the information about the company, about how to contact them, another with some help information and a section related on the user profile.

- **Printing process functionality:** once the user has uploaded his documents, Apapel provides a shopping cart where these documents which are pretended to be printed are stored. Then, printing configuration for the documents is specified (number of copies, page format, colour, finished...). The final price is shown over all the process, specifying the cost per page and the taxes. Subsequently, the user chooses the copy centre where their files will be printed and where he is going to pick up them. Also, the user must choose the way he wants to pay the service. At all times, the user can cancel the printing process, so the shopping cart will be empty again.

- **User account management:** it is mandatory for making orders to be logged in. The options to log in are through Google + or with credentials created by the user. With this last option, the user must specify an email, a password, his name and his surname, and a mobile phone number.

- **Order management:** : in the user profile section, there is an order history where all orders in process or already completed are listed. The user can check them at any time and the information related on the order can be found too, like the order ID, its state, the date when it was ordered, its final price, its printing configuration and where the

user must pick up the result or where it was picked up.

- **Notifications:** notifications in Apapel can be received from different ways: from the web page itself, because in the user profile there is section dedicated to the notifications, from the email and from the mobile phone number through a message.
- **Payment process:** the user has different ways to perform the payment of his orders: through a recharge of his balance with a code provided by the copy centre which specified an amount of money to use in the application, through a credit card or through a Paypal account.
- **Adaptability:** the web page has a responsive interface, which can be seen in the figure 4.15, providing that way a available and usable navegation through a smartphone.

4.3.2 Papelería Complutense

Papelería Complutense[55] has been providing a copy centre service for more than 30 years in Albacete. In its web application, as of providing the classical process of uploading documents to send them to the copy centre and to print them, it allows the users to buy other kind of copy centre elements, like office material, academic material and more technical material. In the following list, the different features of the application are detailed in terms of functionality and appearance:

- **Design:** in its principal page, the company shows specific information about the products and the services which they can offer. Also, there are some offers and recent news about the most sold products or the changes in prices. Through the principal menu and the available buttons, the user can access different sections where they can order documents to print them or they can purchase the offered products of the online shop. There are eye-catching images to attract the user attention and to give him the important information.
- **Printing process functionality:** to order the print of the documents, there is a specific form where the user must introduce the personal data (name, email and mobile phone number). Then, the parameters about the printing configuration must be specified. Finally, the files are uploaded and are sent to the copy centre.
- **User management:** the user must be signed up in the web to purchase the products of the shop and to upload his documents. To accomplish it, there is another registration form where the user only must introduce his email. Shortly, the user will receive an email with a password to log in in the web. The information about the user profile can be changed through the corresponding section of the menu.
- **Order management:** in the user profile section, an history of the orders done by the identified user can be found. It contains the finished orders and they are classified by

[55] `https://www.papeleriacomplutense.com/`

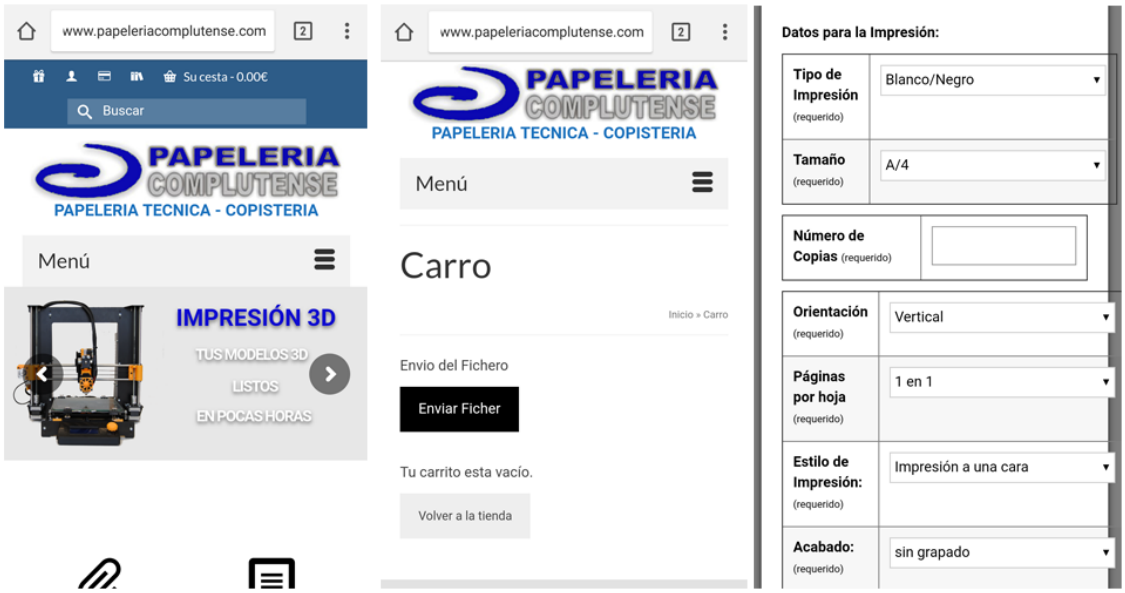

Figure 4.16: Papelería Complutense responsive interface

the type of the ordered product.

- **Notifications:** in this web page, no notification system is available.
- **Payment process:** the payment process is not online, so the user must pay when he picks up the documents at the copy centre. If the user chooses the option of home delivery, he has to pay when the product arrives.
- **Adaptability:** this application has a responsive web design to make it easier to use from a smartphone. The mobile interface is shown in the figure 4.16.

4.3.3 Tus Ideas Imprenta Online

Tus Ideas[56] is a company from Madrid. It is situated in the centre and it has not only copy centre functions, but it offers other services or products like the production of web designs, services related to online SEO and graphic design offers. However, the printing service is going to be highlighted because it is in an independent page of the company page. Its online shop has an e-commerce design which shows a big product catalogue and a shopping cart where the products are storage before the user purchases them. Now the functionality and appearance of the application will be explained:

- **Design:** T¡this online copy centre has a fresh design with clear colours which offers clarity in order to navigate through it. In the principal page, there are highlighted images about the products and some icons which visualizes the most important information about the business and its functionalities. Then, a brief catalogue is showed with the most outstanding products. Finally, there are some text paragraphs where the

[56] https://www.tusideas.es/tienda/

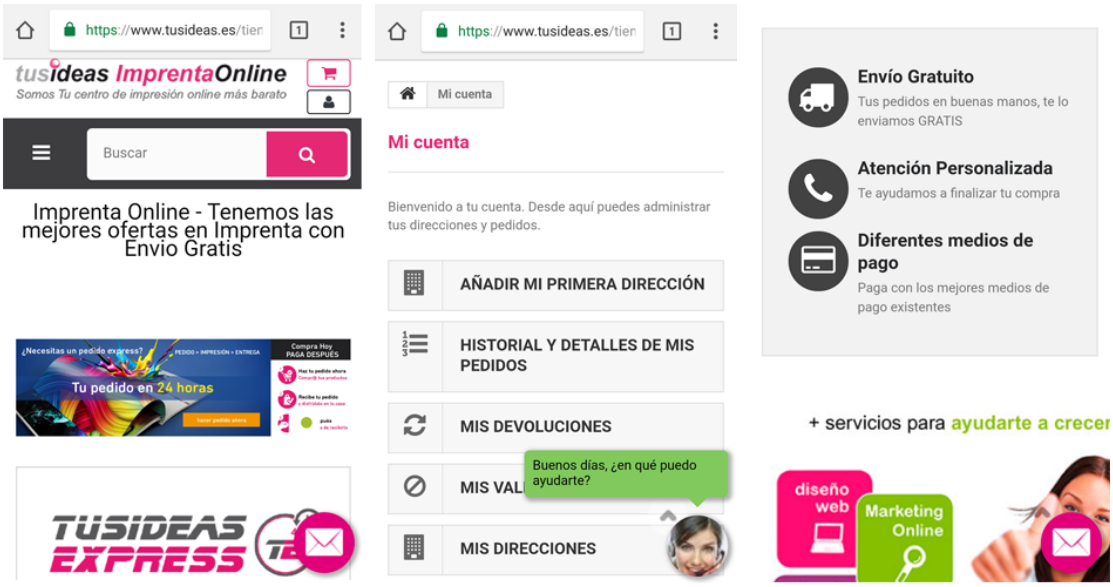

Figure 4.17: Tus Ideas responsive interface

working methodology of the company and the services provided are explained. Also, there is a floating bot in a corner of the page which offers its help to the user.

- **Printing process functionality:** Tus Ideas is a company focused on printing specific products like visit cards, brochures, posters, calendars, seas, etc... Thus, the printing service that is being studied in this section is not offered through this web.
- **User management:** to order products of the catalogue, it is mandatory that the user is signed up or logged in the web. In the registration process, some personal data must be introduced: the name and the surname, the DNI, the mobile phone number, the personal address, a password and the news subscription options. In the profile user section, new addresses can be added where the products can be sent, and the personal data of the user can be changed too.
- **Order management:** : in the user profile section, there is an history with the orders done by the user since the creation of his account. In this history, the date of the order and its state are specified.
- **Notifications:** a notification section is in the profile user section where the user can see the alerts related to the orders he has done. These notifications only work in the web page.
- **Payment process:** the payment of the orders can be done through different possibilities which are online. The user can pay with a Paypal account, with a credit card or with a bank transference.
- **Adaptabilidad:** the application design is responsive, and it provides an easy and comfortable navigation with a mobile device. It is presented in the figure 4.17.

4.3.4 Impresión En Color Madrid

This copy centre[57] whose office is in Madrid offers a completely online service. This means that all the products and services which are sold by this company through its web application. From singular copies to posters and doctoral thesis. The final product is sent to the personal address of the user. The functionality and design of this application is explained in detail below:

- **Design:** in the principal page, a large menu with the list of services which are offered by the company can be found. In every of the menu sections, a service is described with the different types and the prices. There are huge text paragraphs followed by little images. Also, there are some buttons in the initial page which lead the user to start the printing process directly. Finally, some companies which have worked with Impresión En Color are showed and there is information about the offered services focusing on a kind of business: education centre, consultancies, marketing agencies, construction company...
- **Printing process functionality:** Impresión En Color offers an online copy centre which has different options. Firstly, a table with the prices is showed to the user to know the printing rate. Secondly, a variety of recommendations about how should be the documents which are going to be uploaded, like the advisable file format and the resolution of the images belonging the document. Lastly, the user must upload the documents through the different process that are offered by the company. One option is through the web itself filling a form with the personal data of the user and uploading the documents with a size limitation of 16 MB. Other options are available to avoid the size limitation. An option is to upload the documents through an external system called WeTransfer[58], whose behaviour is showed in the figure 4.18. This system has a size limitation of 2 GB, which is so much larger than the first option. Ultimately, the company has a FTP folder which can be downloaded to the computer. The user only needs to put the documents into that folder and the copy centre will receive them, without any size limitation.
- **User management:** there is no need to be registered or logged in the web to use the printing service. A registration user system is not available.
- **Order management:** There are two ways of delivering the orders to the users: picking up the final documents or the products at the copy centre in Madrid or with the home delivery service. At no time the user can not see his orders and their state are.
- **Notifications:** through the web there is not a notification system, but it is possible that the company uses the user email or the user mobile phone number to contact him.

[57] http://www.impresionencolor.com/impresion-digital.html
[58] https://wetransfer.com/

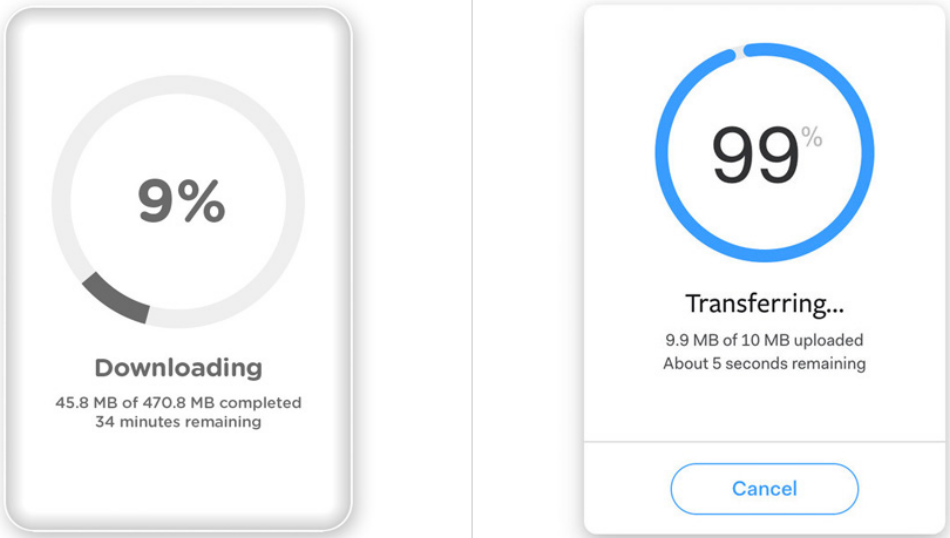

Figure 4.18: WeTransfer interface

- **Payment process:** Since a payment mechanism is not offered through the web, the user must pay when he receives the product or when he goes to the copy centre to pick up the product.
- **Adaptability:** Impresión En Color page has not a responsive design, so its web contents and its functionalities are not well accessible from a device whose resolution is smaller than a computer screen.

4.3.5 MN Impresión

MN Impresión[59] is a copy centre from Castellón which has taken the advantages of the web technologies too. A way to complement its physical shop which already offers the printing offers at that city. Unlike the above-mentioned copy centres, this company has developed a mobile application which is available in the Play Store and it is used to send the documents which the user wants to print from its smartphone. Finally, each element of the application is explained in terms of its appearance and functionality:

- **Design:** MN Impresión web presents a modern design which uses mostly images to attract the attention of the user, being more visual and easier to navigate for him. In the initial page, a slider shows the present offers and the more significant events of the company. Then, there are brief descriptions of the provided services.

[59] http://www.mnimpresion.com/

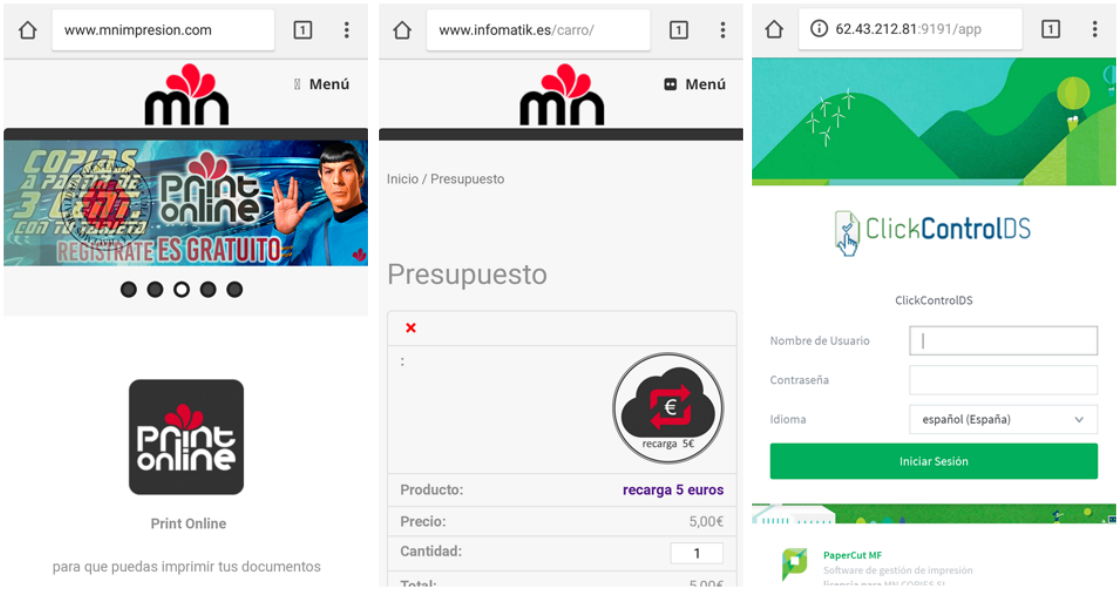

Figure 4.19: MN Impresión responsive interface

- **Printing process functionality:** MN Impresión uses two software system to provide the printing service. Firstly, it offers an external web system called ClickControlDS[60], which belongs to the International Software PaperCut[61] company. This system allows the user prints his documents in any of the available printers which belong to MN Impresión. To start the printing process is mandatory to have a user account. When the user has logged in, a system interface appears where the account information and some order statistics are showed, followed by a menu where there are some useful sections. One of them is to initiate the process of printing documents. The first thing the user must do is to choose the printer he wants to use. Then, he has to upload the documents whose file format must be PDF and to specify the number of copies to be printed. When the process is finished, the application itself shows an order report which expounds its information, its price and its state. On the other hand, MN Impresión has its own desktop application to carry out the orders. From its web page the user can download the application, which allows the user to upload files with a format different from PDF and without a size limitation.
- **User management:** the user nick and the password of the account which is necessary to start printing in the application is provided by the copy centre employee at the physical shop. Thus, it is not possible to create an account directly from the web application to use the service.
- **Order management:** ClickControlDS has a section with a order history where the user can know the order states, the information related to them (order date, its price and its format) and some statistics about them. For example, how many documents

[60] https://www.clickcontrol.es/
[61] https://www.papercut.com/

have been ordered by the user and the total number of printed pages by the user are showed.

- **Notifications:** a notification system neither is not available in the web application nor other ways to notify the user of the order states.
- **Payment process:** MN Impresión payment process is online and it works with a recharge of the virtual balance. The recharge can be of different quantities: 5 euros, 10 euros and 20 euros. It is done through an online process where the user must introduce his personal data and pay via the Redsýs service.
- **Adaptability:** the design of this web application is responsive since it adapts to different resolutions smaller than the one of a computer screen, which makes easier the navigation through the web to the users. The interface design is presented in the figure 4.19.

Chapter 5

Working Methodology

In this chapter, the working and the planification methodologies of the degree project are going to be presented and explained. Also, the hardware and software technologies used to develop the project are listed.

5.1 Working and planification methodology

Furious Koalas S.L.[1] is known as one the spin-offs which belongs to the University of Castilla-La Mancha. In cooperation with the copy centres of the city of Ciudad Real (Castilla La-Mancha), they identified a **common problem related to their kind of business** and they decided to **initiate the development of the DYN Print project**. Its development has been part of the University of Castilla-La Mancha (UCLM) program of work placements in companies.

The development of DYN Print has been considered an **agile project**, which is defined as a project which focuses on software development through iterative and incremental cycles, with a team characterized by high collaboration and well organized. There are a variety of agile methodologies for this type of project, which are identified and explained in [Gar12].

Then, the planification methodology used for the development planification of DYN Print is **inspired by the agile methodology SCRUM**. As specified in [Sch04], the transparency and the adaptation of the project are the keys of the methodology. The use of this methodology is due to the **composition of a small team** that can provide **multidisciplinary roles** within the development of the project, given that it is framed in the company where the roles of employees are also multidisciplinary when it comes to getting involved in the projects that are carried out. **Communication between members is direct and transparent**, as the meetings held each week allow for constant and fluid communication.

One of the goals of this methodology is to comply with the principles elaborated in [BKD01], which attributed to the development of this project are:

- To **value individuals and their interaction more than the processes and tools**: there is full collaboration between team members, in addition to the constant communication

[1] https://www.furiouskoalas.com/

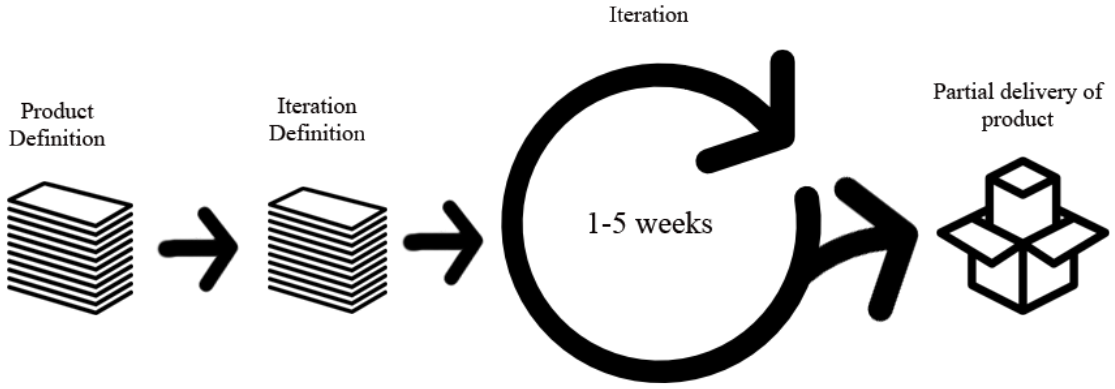

Figure 5.1: Schematic SCRUM functionality

mentioned above. This collaboration is essential to achieve a correct development of the project, as the level of support provided by the different members of the team improves the implementation of DYN Print itself.

- To **value software that works more than comprehensive documentation**: the documentation of the tools used is important for the development of DYN Print, however, the interaction and testing with the different partial versions of the system has allowed a better understanding of the operation and the needs that have arisen.
- To **value collaboration with the client more than contractual negotiation**: at the beginning of this project, an exhaustive study of the objectives and requirements that will define the product obtained is not carried out, but a lighter one is carried out and is complemented as the development progresses. Thanks to the meetings with the product owner, it is easier to define them.
- To **value the response to change more than following a plan**: since the requirements are not definitively defined, it is normal for changes to occur during development. The response and adaptation to these changes are favourable, as regular meetings are held and feedback is received at all times.

SCRUM is a methodology that is committed to the **adaptability of the project**, as it is divided into **different iterative phases** so that each one will begin when the previous one has been satisfactorily completed. In each of these iterations, it must be predicted that it will be necessary to carry out the next one, so that an estimated plan is created through defined tasks. This avoids a direct and complete execution of the product, replacing it with an iterative execution, which allows it to be more controlled and manageable. It is also an

advantage that the iterations mentioned are developed in an overlapping way so that they can influence each other, which helps due to **unforeseen events or changes** that may arise in a degree project. A process that will be explained in more detail below due to the work methodology incorporated in the project. The **user stories** defined in section 6.8.4 constitute the user needs for DYN Print. The figure 5.1 shows an outline of the philosophy that this methodology implements in the project.

The **roles defined** and the people assigned to them for this project are as follows:

- **Product owner:** David Vallejo Fernández y Carlos González Morcillo. It can be considered as the customer who wants to get the system as a result. It helps to define the requirements and features expected of DYN Print.

- **SCRUM master:** Santiago Sánchez Sobrino. He is the project manager and is the main person in charge of supervising the adapted changes and providing the necessary feedback to achieve the defined objectives.

- **Development team:** Ruth Rodríguez-Manzaneque López. It is the team in charge of the development of the system, in such a way that its members take on different roles within the process, as an analyst, designer, developer or tester.

The working methodology of the project will be incremental and iterative, therefore there will be incremental iterations, as **temporal segments whose duration varies between 1 week and 5 weeks or so**, where the development work is similar in order to compose a final product. In every of the iterations, the benefits of the project to the client are delivered and the partial product evolves (as an incremental delivery) starting from the delivered product of the previous iteration. To achieve that, a set of requirements and objectives must be established before starting the iterations. This methodology history and benefits are analysed in [CV03] and the iterations defined for this project are explained in the section 7.2.

There is an iteration in the development of the project whose objective will be to carry out a series of **unit tests to check the correct functioning of some functionalities** that are carried out on the server side. These tests will focus mainly on the functioning of the user information management, such as the virtual balance and the authentication process. This procedure verifies that these functionalities are executed correctly, so that it is verified that the movements on the balance sheet entities and users are right and that they report possible errors correctly.

5.2 Used hardware and software

In this chapter, the hardware and software elements used in the development and the deployment of this project are exposed.

5.2.1 Hardware

Considering that the computing cost is not high, then it is not necessary that the computer used to the project development has huge characteristics of processing and storage. The table 5.1 specifies the detailed characteristic of the computer used.

Characteristic	Value
Operative System	Windows 10 Pro
Computer Model	MSI CX61 2PC
Processor	Intel® Core™ i7-4712MQ CPU @ 2.30 GHz
RAM	8,00 GB
Hard Disk	S-ATA 1 TB
Graphic Card	NVIDIA GeForce 820M and Intel® HD Graphics 4600

Table 5.1: Hardware required.

A local server provided by the PaaS has been used to perform the testing of the development with a virtual non-relational database which is refreshed every time the local server is executed. Regarding the deployment of the project, it is performed with the cloud services also provided by the PaaS Google App Engine, which belongs to the Google Cloud Platform. The functionality of this technology is specified in [D.15].

Finally, as an approach of mobile-first development has been used, a smartphone has been used to test the functionalities of the system through a browser installed in it.

5.2.2 Software

The following list contains the software components which have been used for the development of DYN Print:

- Windows 10 Pro: the operative system in which DYN Print has been developed.
- JetBrains PyCharm 2017: a Python IDE for implementing the backend of the application.
- Sublime Text 3: a text editor to implement other parts of the project.
- Google Cloud SDK: a set of tools provide by Google Cloud to execute commands related to some of the Google Cloud services like Compute Engine, Cloud Storage or App Engine.
- Google Chrome 64: the web browser where the application has been tested and visualized.
- Git and Bitbucket: the distributed version control system and the hosting system to save the source code of this project.

On the other hand, the list below presents the programming languages, frameworks and libraries which have been used in the project implementation:

- Python 2.7: a interpreted high-level programming language to program the backend.
- HTML5: a standard mark-up language to compound the structure of the web pages which form the front end of the application.
- CSS3: a style sheet language to define the style of the web pages.
- Bootstrap 3.3.0: an open source toolkit for developing responsive, mobile-first systems.
- Javascript: an object-oriented based language used for implementing the necessary scrips in the front end.
- JQuery: a Javascript library to simplify the syntax and the event management.
- PDF.js: a standard-based platform to parse and to render PDF documents in Javascript.
- Webapp2: a Python web application framework to simplify the management of the request and the responses performed by the application.
- PIL: a Python library that supports the image processing and manipulation.
- pdfrw: a Python library to read and to merge PDF documents.
- unittest: a Python library which offers the preparation and execution of unitary tests.

Chapter 6

Architecture

THIS chapter will provide a detailed analysis of the modules that constitute DYN Print. An organization of these modules along with their relationships is presented to help understand the structure of the system. The different functionalities offered by the system will be defined, specifying which module each one belongs to, as well as an explanation of how its implementation has been carried out, being more detailed in terms of technical specifications. Reference is made to the user stories that are linked to each of the functionalities. In addition, an explanation of the existing models in the database is performed together with the organization at the project file system level.

6.1 General overview of the system

A top-down approach with a high level of abstraction is followed to define the DYN Print principal components, which compose the architecture of the developed system. The possible users of the system who interact with the different modules of the architecture are the copy centre clients and its employees.

The modules share components in both the client and the server side. The **connection between the different modules are performed by the Google App Engine tool**, which allow them to communicate and request the provided functionalities. In the figure 6.1, the structure explained is presented, where the relationships they have between them are showed and the layers to which they belong. In addition, the figure 6.2 shows the flow of information exchanged between these modules, as the data exchanged depends on where it is going in the system.

The implementation of the project on the Google App Engine platform allows the application to integrate a variety of technologies, as well as the use of external libraries useful for the functionalities. The **client side of the application uses typical frontend development technologies** such as Javascript, HTML and CSS, but beyond the languages, Javascript scripts use Asynchronous JavaScript And XML (AJAX) requests, web browser authorization requests, classes that develop object-oriented programming, document management is used in addition to processing and handling JSON responses received from the server.

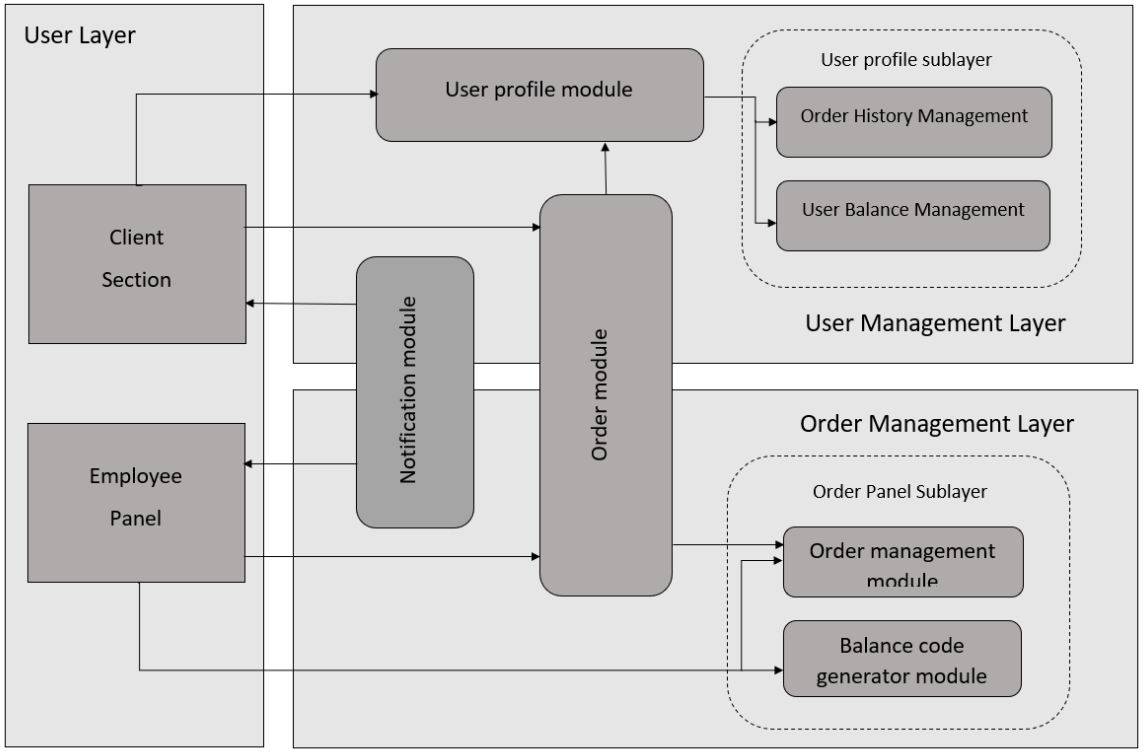

Figure 6.1: Module structure of the system

On the other hand, the server is implemented in the Python language and makes use of many of its available libraries. The **server is structured into three layers** and carries out a process that requires communication and data transmission through them. This multi-layer structure allows to have different advantages:

- Independence in the functionality of the different layers.
- Decoupling of the parts of the system, which makes it easy to make changes without affecting other layers.
- In case the needs of the system change or increase, this structure allows the scalability of each of the layers.
- It provides greater security and autonomy between functionalities.
- There is a high level of concurrence at a time when there is a high number of users.

The reception of requests from the client side is done thanks to the webapp2 library that is integrated into Google App Engine. Facilitates the reception of requests by defining the classes ordered as handlers, which can implement the corresponding get and post methods, as well as manage and process the response to be sent to the client. Google App Engine has a configuration file that defines the **mapping of the requests received**, so each request is sent to the appropriate handler for processing.

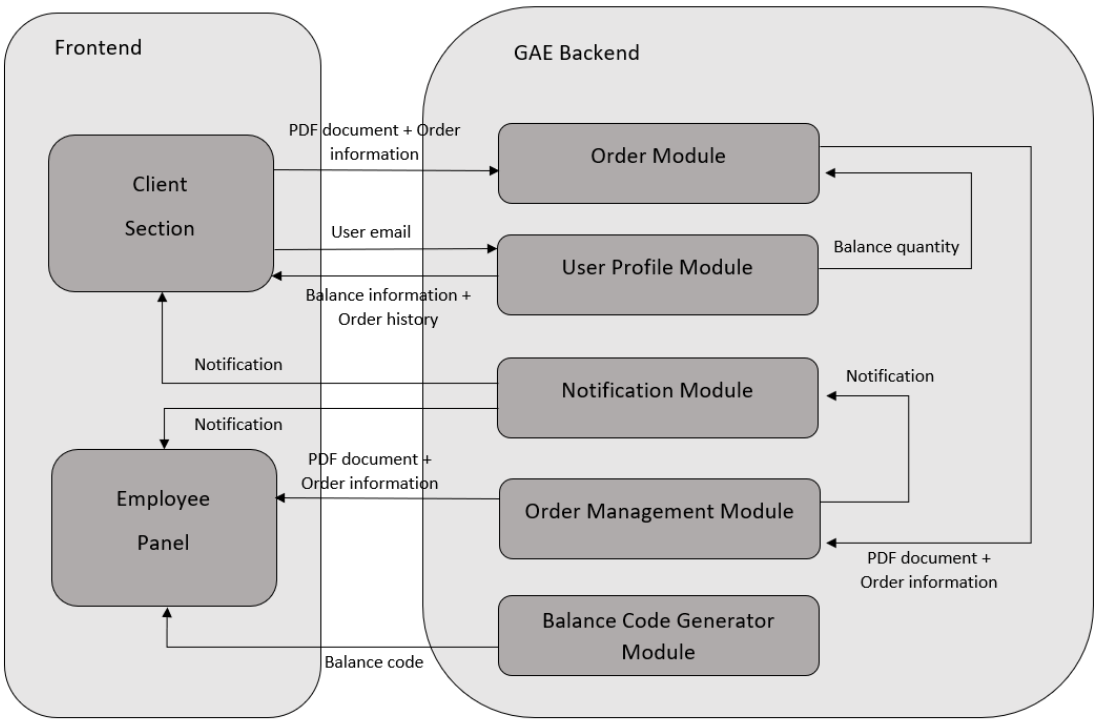

Figure 6.2: Information flow exchanged between the modules of the system

For the system classes that are in charge of the database accesses, they implement the ***Singleton* design pattern**, which allows a single instance of this class to be created to handle the required processes. Generally, the use of this design pattern is discouraged despite the advantages it offers, which are as follows:

- The object associated with the class is **available globally**.
- Only **one instance** of the class is guaranteed.
- The instance **is only created once**, which is when it is first required to be used. This initialization is performed at runtime.
- It is possible to create **derived classes from a base class** that implements the *Singleton* pattern.

Despite the advantages that have been pointed out, this pattern leads to a series of **long-term disadvantages**, which is why its use is discouraged. One of the main problems of using this pattern is the global access that is carried out on the class instance, which can lead to **problems in the debugging and concurrency of the code**. On the other hand, the single instance initialization can provide complexity and response time to the application that is negative for its performance.

There are solutions to avoid the disadvantages offered by the *Singleton* pattern. However, for the development of this project, the **advantages offered by at least one first version of**

the system have been more valued.

Hereunder, the different modules that implement and execute all the above are briefly explained:

- **Order module:** this module is the main module of the application for the client side, as it allows clients to upload their files and place the corresponding order with their parameters. The ordering module provides the system with the necessary user information and processes it on the server side to correctly create the order in the system. The system's order form receives the information provided by the customer and makes it available for the copy shop employees to receive and complete. It involves the implementation of the main functionality of the system, the complementation and operation of which will be further detailed below.

- **User profile module:** it is like a module for managing user information related to the application. This module manages the features that are useful for customers. It calculates the customer's current balance and allows you to add an amount to the balance. In addition, it retrieves all orders placed by a customer with order information. This information is useful for the client because it is essential to have knowledge about the entities that are related to him/her in the system. To make an order, the customer needs to know if he or she has the necessary balance and also has the possibility of increasing the quantity. On the other hand, being able to have a follow-up of the orders he or she has placed, whether completed or pending, and information regarding each order to identify it is important for users as it avoids ignorance.

- **Notification module:** as defined in the chapter 3, the system has a proactive role in informing the different users of DYN Print. The system must notify the customers when orders are printed at copy centres, so that the customers know when they need to go there. On the other hand, the system must notify the employees when the customers demand new orders. The method of notification for both will be through a web browser notification API, however, the customer will receive additional notifications via email. It is important that users of the system are aware of the changes that are being made in the main entities and that are the basic operation of the application.

- **Order management module:** when the customer places an order, the document associated with the order is retrieved for the employees, allowing them to send it to the print queue. Files that are already printed are listed so that employees know which orders are pending to be picked. The system must guarantee a correct reception of the orders and the processing of the document content must be faithful to that uploaded by the customer. In addition, it is also necessary for the system to track orders according to their status, so this module also provides a list of orders that are pending to be picked up by the customer.

Module	Basic functionalities
ID	US_AUTH (Table 6.8.4).
Problem	It is necessary to create a user in the application related to every client of the copy centres to manage and to identify the orders made. Also, there are users with special privileges, who are the employees of the copy centres and who need to access the private panel of the application. Every user must identify to use the application through an authentication process.
Approach	To avoid saving risky data about the client, like his password, there will not be an authentication process belonging to DYN Print. Instead, an authentication process through third-party companies is going to be used. Thanks to the Firebase authentication system, the users can identify through different suppliers: Google +, Facebook and Twitter. Firebase will take care of the OAuth2 process and the management of the user session.

Table 6.1: User authentication functionality

- **Balance code generator module:** finally, this module is accessed by the employees to generate a balance code associated with a quantity of money that the clients must pay to get one. This balance code allows the clients to increase their balance quantity in the application for making orders.

6.2 User authentication

The user management was approached from the beginning in such a way that the **registration in the application of the users was totally independent of the copy centre**, whose development process is specified in the table 6.1. The user does not necessarily have to visit the copy centre beforehand to log in the system. Therefore, the system module that authenticates and registers such users must be consistent and secure.

6.2.1 Google +, Twitter and Facebook authentication

At the beginning of the project, the decision was made to save risks when storing information related to each user of the system, so that in order to identify each of them it would only be necessary to save **their profile name and their e-mail address belonging to an existing account of the different third-party providers: Google +, Twitter and Facebook**. Therefore, the system will not implement any proprietary process by which the user registers but will only have to access through an authorization and identification gateway obtained from the providers.

In this gateway, the user will authorize the DYN Print application to access the data: the name of that profile and the e-mail address. Once authorized, the application will either create the user in the system, if it has not already been identified, or simply identify and create a session for the user.

With the logged in session, the user will be able to access the different sections of the application where they can use its functionalities. If no session exists, **the application will deny users the access and will forward them directly to the page where they need to log in**. For sections of the application that are exclusive to employees, in addition to requiring a user's login, the user is also required to have an administrator role.

6.2.2 OAuth 2 authorization and Firebase authentication

The process of obtaining the profile data will be carried out via the **OAuth 2 authorization protocol**, which authorizes third parties, in this case DYN Print, to access data such as the profile name and e-mail address. This protocol follows the next process to obtain the data:

1. An **authorization code** is requested for the resource assigned by the company that is going to provide the profile data.
2. With this authorization code, the system requires an **access token** the company's resource.
3. When the system already has the access token, then it will give the system **the permissions to access the user's data**. If the token obtained is valid and has not expired, then the requested resource will return the requested data correctly.

It is a process that requires several direct requests to different resources and although there are libraries prepared to facilitate the process and make it more intuitive, problems arose when validating the facilitated access tokens. Therefore, it was finally decided to use the Firebase Authentication tool. The authentication is one of the main services offered by Firebase, whose process of implementation and adaptation to applications is specified in [Yah17].

Firebase[1] is a Google platform that can be easily integrated with Google App Engine services. In this case, its integration has been necessary for the authentication process. Some of the advantages it offers over the use of libraries that implement the OAuth 2 protocol are as follows:

- Availability of a **specific User Interface (UI) for the authentication process** that handles the necessary flows. It is open source and can be customized to fit the application design.
- Authenticate users through their email address and password, without having to store them in the application.
- Integrates different providers with federated identity: **Google, Facebook, Twitter and GitHub**. They all have the same Firebase authentication procedure.

[1] https://firebase.google.com/

To integrate Firebase into the application, it was necessary to **register the Google App Engine project in the Firebase console**. This way, Firebase can know all the information related to the project and apply its functions to it. Authentication must then be enabled; there is a section for this function in the console that lists the users registered in the application, configures the available access methods and provides templates for the emails sent.

In the case of DYN Print, Google, Facebook and Twitter access methods have been enabled, as mentioned above. It is essential to **add the address where DYN Print is hosted in authorized domains**, so that redirections to the application that are made in the authorization process are allowed.

Once the authentication function is enabled, the service must be integrated on the client-side of DYN Print. The implementation is carried out with the Javascript language, and through the guide provided by the official Firebase documentation, the necessary imports and the code that will activate Firebase are added to the pages. For this implementation it is necessary to have the **key of the application** as well as other data that can be found in the console. An example of the code which is necessary to initialize Firebase in the web application is in the listing 6.1.

```
<script src="https://www.gstatic.com/firebasejs/4.10.1/firebase.js"></script>
<script>
  // Initialize Firebase
  // TODO: Replace with your project's customized code snippet
  var config = {
    apiKey: "<API_KEY>",
    authDomain: "<PROJECT_ID>.firebaseapp.com",
    databaseURL: "https://<DATABASE_NAME>.firebaseio.com",
    storageBucket: "<BUCKET>.appspot.com",
    messagingSenderId: "<SENDER_ID>",
  };
  firebase.initializeApp(config);
</script>
```

Listado 6.1: Inicialization of Firebase in Javascript

When the application is initialized through Firebase, the web application already has the possibility to access the services provided. For the authentication service, an authentication provider will be created and the scope defining the data to be accessed from the profile will be defined. Afterwards, you only need to call the function that manages the authentication process where the user will enter their credentials and give the necessary permissions. This process can be done through a pop-up or through redirects.

When the authentication process is complete, the same function receives the required data and gives the possibility to manage it. **DYN Print collects the user's email and profile name and sends them to the server** for identification within the system. The flow of this functionality is shown in the figure 6.3.

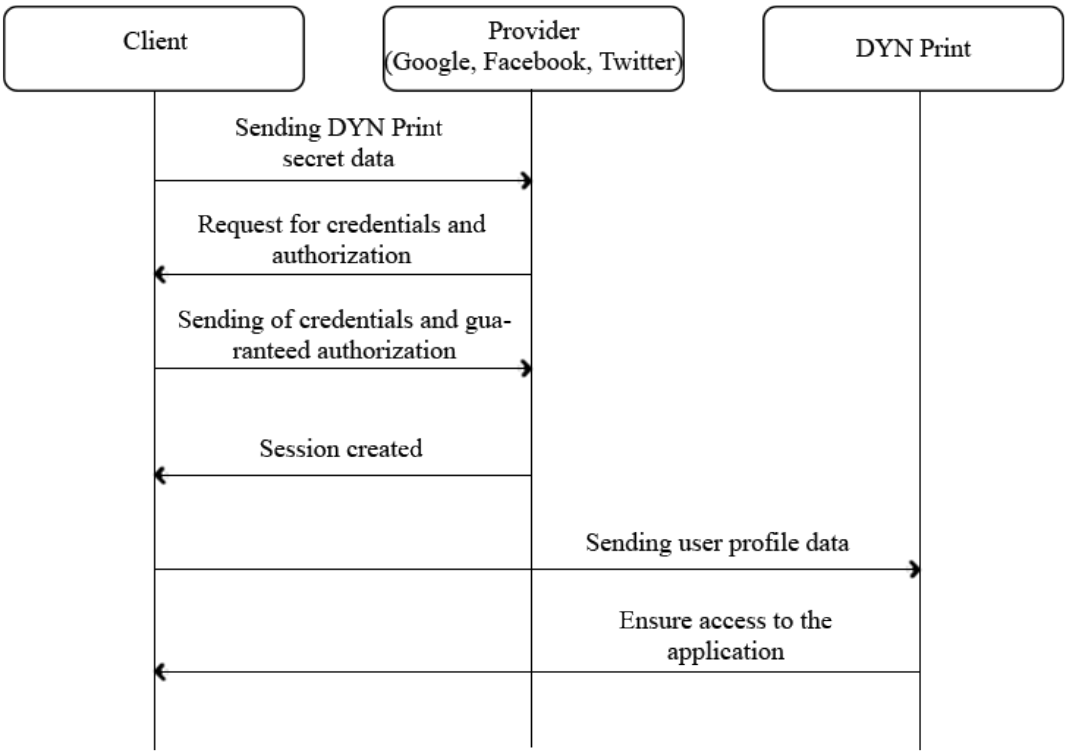

Figure 6.3: User authentication flow

6.2.3 User and session management

When the client side sends the user's data to the server, the server is responsible for identification. The user data are sent through an AJAX request implemented in Javascript. First, **it is checked whether the user already exists in the database**, in which case a response is simply identified and sent to the customer to confirm his or her identification. If it does not exist in the database, then the server will **create a new user**.

For new users, their e-mail address and profile name are specified, but also their role is defined, which by default is 'user', and their balance is created, which for new users is zero.

When the new user is created, the confirmation is also sent to the client-side. When it receives this information, **a session is then created in the web browser** linked to Firebase, so that on every page where Firebase is enabled, it will check to see if a session is logged in or not. This session restricts access to DYN Print to users who are not logged in. When the system identifies a user's session, you get an object that refers to the user and the information in their profile is easily accessible.

For privileged access to the employee application section, the **role property** has been implemented in each user's entity in the database. The customers of the copy centre have

Module	Order module.
ID	US_UPLOAD (Table 6.8.4).
Problem	The main functionality of DYN Print is to send documents through the web browser so that copy centre employees can send them to the print queue. This functionality is the core of the application and should allow you to attach PDF documents, choose the printing parameters that the customer wants for their order and see a preview of the document along with a summary of the order. Once the customer confirms the order, it is created in the system along with its characteristics and the contents of the document. Finally, to speed up the printing process, an additional page is created that will be added to the beginning of the PDF content, where the parameters, the user to whom the order belongs, and an identifier will be specified.
Approach	The selection of the document to be uploaded and the required parameters will be made through a form on the client side. When the order is sent to the server, the server will store the document content in the Google Cloud Storage service, and then create one entity for the order that refers to the customer's user and another for the associated document in the Google Cloud Datastore service, so that everything is linked.

Table 6.2: File upload functionality

the role of 'user', as the basic user of the application, however, the employees will have the role of 'admin', which will allow them to access this section. To create users with the role of 'admin', it is necessary to do so through the Google App Engine console. In this way, when a user identifies himself/herself and accesses the private section of DYN Print, the application itself first **checks the user's role**, in order to know whether to guarantee or deny him/her access.

6.3 Uploading files

The most fundamental functionality of DYN Print is the creation of orders by the customer, whose development process is specified in the table 6.2. A process that will be started when the user decides to select the file from the system. The main objective for this process is to design a **flow of mechanisms that are intuitive and easy for the user to understand**, since it depends on the user whether the application can provide the service for which it is intended.

6.3.1 Uploading form

Once the user is logged in, he or she is presented with the ability to select PDF documents from the file system of the device to which he or she is connected. Once you have selected the document, the process of initiating an order begins. The document must be in **PDF format** to avoid possible complications with the structure of the document. The HTML language itself offers a type of component that allows access to the system files, while Javascript offers an

event capture mechanism through which it receives the document that has been selected with the component.

The user is then presented with a form through which he or she selects the **characteristics of the order related to the printing parameters** that the copy centre employee must take into account when printing the documents. In the interface, the simplest and most responsive ways of selecting these parameters have been implemented so that the user does not have any complications when selecting them. Each of these characteristics has **default values and a range of possibilities**:

- **Color format:** its predetermined value is the grey scale. The possible values are: grey scale and color.
- **Number of copies:** the predetermined value is 1, and the user can choose any value from 1 to 50 copies.
- **Style:** the predetermined value is double-sided style, and the possible values are: doubled-sided style or one-sided style.
- **Page format:** the predetermined value is A4, and the possible values are: A4 and A3.
- **Finished:** the predetermined value is without finished, and the possible values are: without finished, stapled or bounded.
- **Scale:** the predetermined value is without scale, and the possible values are: without scale, scale of 2, scale of 4 or scale of 8.

With the order parameters selected, the process will continue to show a **preview of the document content** using a Javascript library called PDF.js[2]. In addition, a summary of the order is included where the user can check which parameters have been selected and see the **final price of the order**, which is calculated taking into account the number of pages of the PDF and the color format chosen.

The user will have the possibility to return to the configuration in case of a mistake or, on the contrary, to place an order. The system will **deny the request if the user's balance does not have the required amount**, which is checked on the server side by accessing the user's balance. The process of creating an order can be cancelled at any time, and if the user mistakenly tries to leave the page or go to another section, the browser will ask for confirmation to avoid the loss of the order.

While the user makes the order, the client-side is responsible for creating a Javascript object whose variables are the order and document data that are completed throughout the process. Finally, when the user decides to send the order, this object data, which includes the document content, the user's mail, the printing parameters and the price, is sent via an

[2]https://mozilla.github.io/pdf.js/

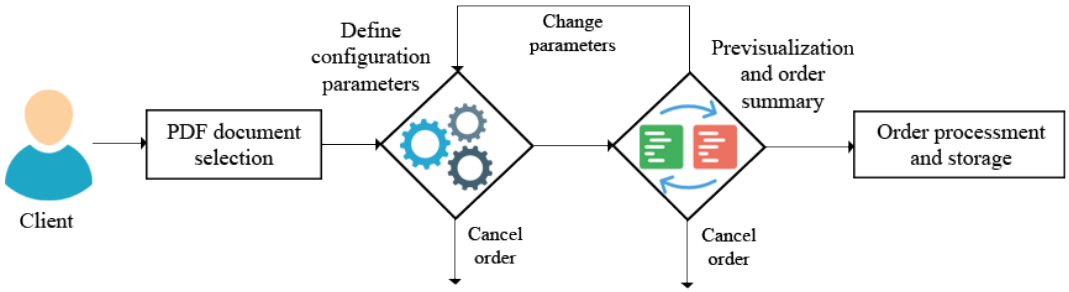

Figure 6.4: Uploading a document functionality flow diagram

AJAX request to the server. A flow diagram with the explained process is shown in the figure 6.4.

In the interface buttons are added that indicate the possibility of uploading documents from providers such as **Dropbox or Google Drive**, however, are features that have not been carried out, leaving as the only option to upload documents from the device's file system.

6.3.2 Files management

When order-related data is received on the server, the server has several tasks to perform:

1. An **identifier for the order** is created from the date it is uploaded to the server and a code generated by the Python library called Universally Unique Identifier (UUID).
2. An **element is created in Google Cloud Storage**, whose path corresponds to the ID, which saves the content of the PDF uploaded by the user.
3. A **task is created in the background** and added to the queue.
4. The function of the background job is to create a **blank page where the print parameters are written along with the order ID to identify the order**. This page will be added to the beginning of the original PDF content.
5. An **entity is created in Google Cloud Datastore** that stores a reference to the Cloud Storage key that identifies the uploaded document.

Using the task queue using the **Task Queue API**[3] provided by Google App Engine is necessary to reduce response time for the user once the order is sent from the client-side. This task queue is managed using the deferred library[4] that is integrated into the Google App Engine development environment.

Google Cloud Storage has been used for the storage of files uploaded by users due to the **limited space defined in the Google Cloud Datastore** service. This limitation disappears

[3]https://cloud.google.com/appengine/docs/standard/python/taskqueue/
[4]https://cloud.google.com/appengine/articles/deferred

Module	Order management module.
ID	`US_COMPLETE_ORDER` (Table 6.8.4).
Problem	Orders placed by customers must be managed by employees, as they are responsible for adding them to the print queue and for managing which orders are pending to be printed or picked up. The system should inform employees of the open purchase orders on the client-side. On the other hand, the server is responsible for retrieving orders from the database depending on their status and sending them to the client-side along with their content.
Approach	The orders to be submitted on the customer side for the employees are added to a list. There is a list for each order status: *To be printed* and *to be picked*, so that, from the *to be printed* list, employees can select the orders they will send to the print queue, and when the print is confirmed, the orders will be added to the *to be picked* list, where employees will have to select the orders that have already been picked and inform the system that they have been completed. When the pages corresponding to these two lists are loaded, the server immediately retrieves from the Google Cloud Datastore service the orders whose status corresponds to each list and sends them to the client-side, who adds them to the list.

Table 6.3: Order handling functionality

in Cloud Storage, which can generate a reference key that can be stored in Cloud Datastore entities.

6.3.3 Order management

The document is saved in the application's database, but an entity that refers to the order is also generated. This entity stores both the key of the document that has been saved in the database, as well as the printing parameters, the price of the order and the key of the user who has made it. In this way, the entities of the documents are differentiated from those of the orders.

This entity model **makes each order unique because of the identifier**, which is the same as the document identifier, creating a unique connection between document and order. This avoids problems of duplication between documents and orders.

6.4 Order management by the employees

The private section available to copy centre employees has two important sections: a section containing a **list of orders that have documents to be printed**, and a section containing a **list of orders whose document has been printed but not yet picked up** by the customer in the copy centre. These sections constitute the functionality to manage the orders from the copy centres, whose development process is detailed in the table 6.3.

6.4.1 Section of documents to be printed

This section is responsible for collecting the orders whose documents are to be printed in the copy centre. To facilitate its management, an interface has been created on the client-side that incorporates a list where orders are added. The order page is reloaded from time to time so that new orders are added to the list along with those that have not yet been dispatched.

When the user accesses this section, the browser will request from the server a **list of the orders whose status is listed as pending to be printed**. The server is responsible for retrieving the content and data of the order through requests to Google Cloud Datastore.

The list only shows the order ID, which consists of its date and an identifier. It is possible to click on the orders and have the content of the document to which they are associated downloaded to the computer of the employee who is connected.

To open the window that will start the document printing process, select the item from the list corresponding to the order using a checkbox and send it to print. To speed up this process, it has been implemented the **possibility of selecting more than one element from the list at a time**, so that when they are sent to print, a new window opens for each document.

6.4.2 Section of documents pending of collection

Once the employee has printed the order documents, they disappear from the previous list and are added to the list where the orders are to be picked up by the customers of the copy centre. The orders listed here only need to be completed when they have been delivered.

When this section loads, the browser returns to make a request to the server requesting the **orders whose status is pending to be collected**. The server, in the same way as before, retrieves the orders from the database and sends the information to the browser, so that it includes them in the list.

Unlike the previous section, there is no possibility to download the contents of the document associated with the orders. Only the order IDs are listed. What is the same as in the previous section is that each item in the list can be selected individually or in its entirety, so that group orders can be completed or individually.

When an order is deemed completed, then its **status changes to *completed*** and it is no longer listed in either section.

6.4.3 Printing process

Employees must send documents to the **print queue via their computer's web browser**. Due to security reasons implemented in current browsers, each document opened in **the web browser must be sent for printing one by one**, preventing the sending of sets of documents, as well as the selection of printing parameters from outside the print window. This security feature is a problem to ensure a faster printing process.

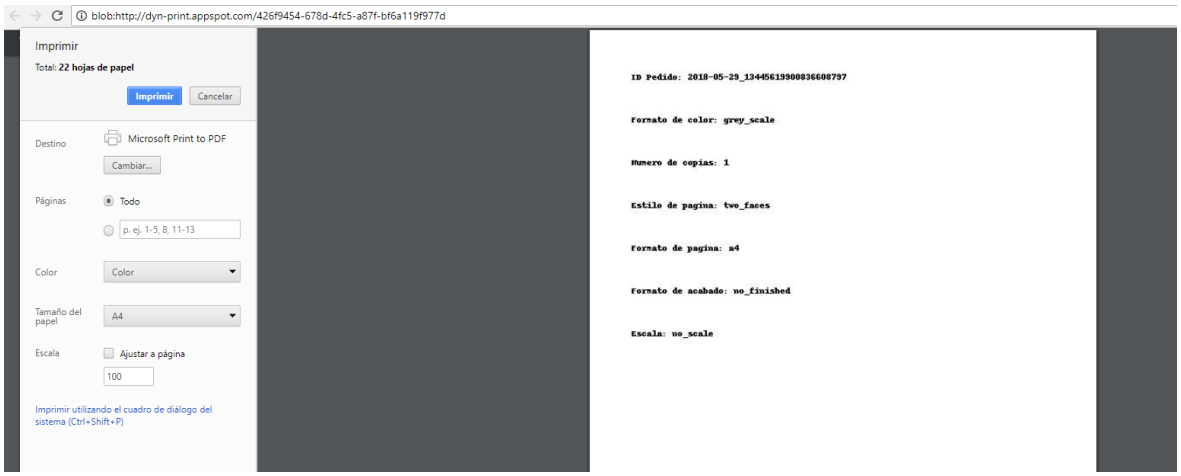

Figure 6.5: Preview of the configuration page.

The solution that has been provided to the problem is based on attaching a **first page to the contents of the document that reflects the printing parameters and the order ID**, so that in the windows that start the printing process and that shows a preview of the document, this page is shown with the information, making it easier for the employee to select parameters from the browser. The figure 6.5 shows how the configuration page shows the parameters defined for the order first.

This task of assembling the content of the PDF and the new configuration page is done by the server, with the help of the library that deals with PDF documents called pdfrw[5]. At the beginning of this implementation, the use of the most popular library called pyPDF[6] was considered, however, when delegating this process to a background task, it **exceeded the allowed response time**.

It was proposed the idea of bringing **together in the same document all the orders whose printing parameters were the same** and each document is separated by the page where the information of the order appeared, so that they were all sent followed to the printing queue avoiding going one by one through the browser.

6.5 Notification management

The system is responsible for **notifying both copy centre customers and employees**. The development process of this module is specified in the table 6.4. Each user type of the system will be notified accordingly in different ways. An outline of this system can be seen in the figure 6.6.

[5]https://github.com/pmaupin/pdfrw
[6]https://github.com/mstamy2/PyPDF2

Module	Notification module.
ID	`US_USER_NOTIFICATION and US_EMPLOYEE_NOTIFICATION` (Table 6.8.4 and table 6.8.4).
Problem	The system needs to be able to inform users of order status changes that exist in the system. To do this, it is necessary to inform customers when the order has been printed and is ready for pickup and employees when new orders are placed in the system.
Approach	Firstly, to alert employees via the desktop on behalf of the customer, the Notifications API provided by the Javascript language will be used, which is a standard feature for most browsers, allowing a notification to appear on the desktop to warn of the arrival of new orders when the list of pending orders is updated. Secondly, the customer will be notified in several ways. When the order changes from print pending to pick-up pending status, an email will be sent to tell the customer. In addition, a notification will be created on the server, as an entity of the Google Cloud Datastore service, associated with that client and sent to the notification panel in the user profile section. The customer side will be responsible for notifying the user through the Notifications API and through the application interface.

Table 6.4: Functionality of the notification alert

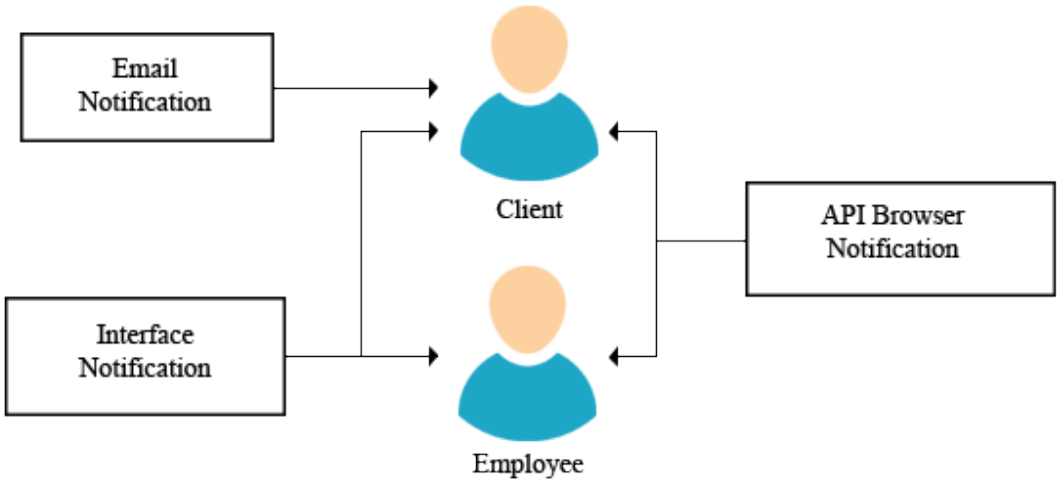

Figure 6.6: System notification schema

6.5.1 Employee notification system

The notifications for the employees are made through the sections where the orders are listed. It is the client-side that manages it through a **notification API that is available for the latest web browsers**, both for desktop and mobile versions.

This API is implemented with Javascript and requires the **authorization of the web browser user** to be able to use it. Therefore, when the pages are loaded, a request is made to the user

to accept the authorization. If it is denied, then the notifications cannot appear. Once the browser is authorized, notifications can already appear on the desktop of the computer or mobile phone, even if the user does not have the application window in the foreground.

Notifications will be created when the client-side receives new orders from the server. There is a variable in the **web browser session storage** which keeps track of the orders that have already been listed in the section, so that the browser will notify when this variable increases in value, that is, when new orders are sent by the server.

In the listing 6.2, the implementation of the verification of the existence of new orders with the help of this variable can be seen, along with the creation of the corresponding notifications.

```
if (sessionStorage.getItem('num_documents') === null) {
    sessionStorage.setItem('num_documents', count.toString())
} else if (sessionStorage.getItem('num_documents') !== count.toString()){
    if (count !== 0){
        var notification = new Notification((count - sessionStorage.getItem("num_documents")) + "
            documentos disponibles.");
        sessionStorage.setItem('num_documents', count.toString());
    }
}
```

Listado 6.2: Creation of desktop notifications for employees

6.5.2 Customer notification system

For copy centre customers the notification system is more extensive. It should be emphasized that customer **notifications will be created when the copy centre employees have printed their orders**. Therefore, when the order changes from *print pending* to *pick-up pending* status, then that notification will be generated.

The difference with employee notifications is that for customer notifications an entity is created in the database of that notification whose properties are the key to the user's entity for which the notification was created, the status of the notification: *read* or *not read* by the user and the associated order. The default state at creation is *not read*.

At the same time that this entity is created, the user is sent an e-mail from the application informing him/her of the order that is ready to be picked up. These emails are sent by the server using the Google App Engine **Mail Python API** library[7], which allows you to create and send the message to the specified email using just a couple of functions, with a custom header and message body.

On the other hand, the user now has a series of available notifications, which are sent to the client-side to notify the user through the browser or application interface. In each page of the

[7] https://cloud.google.com/appengine/docs/standard/python/mail/

Module	Balance code generator module.
ID	US_BALANCE_CODES (Table 6.8.4).
Problem	In order for DYN Print to allow the user to place an order, it is necessary that his or her virtual balance belonging to the system has enough money to deduct the amount required for the order. The price of the order is calculated depending on the number of pages of the document and the configuration selected by the customer. The system will have a functionality that generates codes associated with an amount of money that the customer will have to pay to add this amount to his virtual balance.
Approach	Employees will have a private form where they will have to enter the customer's email address and select the amount of money for the server to generate a code using the UUID module, creating an entity for it in the Google Cloud Datastore service. This code will be associated with the customer's user and will have a status that will reflect whether the code is used or unused. Once created, it will be sent to the browser for the employee to view and transmit to the copy centre customer.

Table 6.5: Balance management functionality

application, there is a mechanism that checks, by sending a request to the server, if the user has unread notifications available, so that in almost every page of the application, a **desktop notification** like the one created for employees will be created and showed and also, **through the page interfaces**, alerting that in the profile section dedicated to notifications there are notifications pending to be read.

When the user accesses this section of the profile, then the client requests the server for the notifications, which returns a list with the order ID and the name of the document that is ready to be picked up. The elements of this list are added to a list in the section and allow the user to confirm their reading. When the user confirms the notifications, the server changes the status of the notifications to read.

It was proposed to implement a system of **notifications via Short Message Service (SMS) messaging** from mobile phones. However, by not storing the user's phone number, it has been discarded.

6.6 Balance code management

Since the orders made have a cost, **each user disposes of a virtual balance which must have the necessary amount to order it**. There is a module in charge of this functionality and its development process is detailed in the table 6.5. The way to increase the amount of this virtual balance must be exercised through random codes that are generated in the copy centres. This module is in charge of generating these codes so that the employee can give them to the customers who buy them.

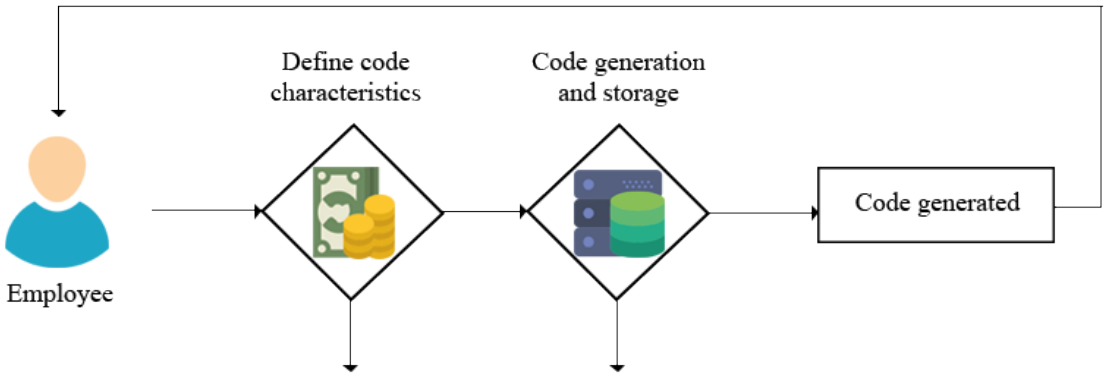

Figure 6.7: Flow diagram of the process of generating a balance code

6.6.1 Code generator and storage

The generation of the balance codes is carried out by the employees of the copy centres. In the private section for them, there is a sub-section with a **form that will collect the necessary data to generate the codes**. It is required to select the amount of the credit, whose possibilities are 5 euros, 10 euros or 20 euros, and also to specify the email address of the user to whom the generated code will be sold.

Specifying the customer's email address is a mandatory condition to prevent the codes sold from being used by other users who have not paid for them. What is not a required condition is that the user who buys the balance code has already registered in the system, as soon as he or she makes the first identification he or she will be registered immediately and when he or she uses the code he or she will already be identified in the system.

Once the form has been filled in and the request for the code is sent, the server is responsible for creating the code, which is generated with the **Python UUID library**. At the same time, an entity is created in the database for each code. The properties of such an entity are the code itself, the user it is associated with and the status of the code, which can be used or not. The default value when creating the code is unused. State property is required to prevent users from using the codes more than once.

The code is returned to the client-side and the browser shows it to the employee, who must give it to the customer so that he or she can debit his or her virtual balance.

The entire process specified above is shown in the figure 6.7.

6.7 Client profile

In DYN Print there are two areas within the **user profile section that are related to order information and the balance of the connected user**. These areas constitute the module in charge of the functionalities related to the user's profile and in the table 6.6 it is specified how the development process is. The user can access them to see the status of their orders

Module	User profile module.
ID	US_USER_PROFILE (Table 6.8.4).
Problem	The clients of DYN Print must have access to the information related to their profile. The important features about their profiles are the user balance and the history of their orders. The system must retrieve this information from the server and deliver it to the clients.
Approach	In the client side, there are two sections where the user can get the information he wants to know. One section is for the virtual balance, where the current quantity of the balanced belonging to the logged in user is retrieved from the entity of the user in the Cloud Datastore and where the user can add a quantity to his balance through a form in which he writes the balance code provided by the copy centre. The other section is for the order history, where a list is showed with all the orders made by the current user. Every order listed has the useful information about it, like the date, the price, the document name and the state of the order. The server is in charge of retrieving all the orders linked with the client user from the Cloud Datastore.

Table 6.6: User profile management functionalities

and the status of their balance.

6.7.1 Balance of the client

The section dedicated to the virtual balance offers different functionalities. First, when the user accesses this section, the browser makes a request to the server to recover the **amount of the user's current balance**, so that it knows at all times when money is left accumulated, because without having enough balance it is not allowed to place orders that exceed the amount.

On the other hand, this section is where **the amount is added to the balance through the codes generated** by the copy centre. After the customer has purchased it in the copy centre, he must enter the code using a form, which is sent to the server and the server is responsible for carrying out the necessary checks.

First of all, the server checks that the entity of that code exists in the database. The code is then verified to be linked to the user who intends to use it. And finally, it is checked that the code entered has not been used before.

If all these checks are successful, the **code status is changed** to used and the amount associated with it is charged to the linked user's balance.

It was proposed the idea of including the possibility of adding money to the balance through the **Paypal system**, so that it would be an online mechanism. However, this possibility was not completed, leaving the only option of recharging the balance through the codes.

Client-side structure			
/			
	static/	This folder contains all the templates that are loaded to the requests of the different web pages that compose the application.	
		css/	Folder containing the style sheet.
		images/	Folder containing the images used in the application.
		scripts/	Container folder with the function definitions that implement the frontend behaviour, communications with the backend and the processing of the data retrieved from the backend.

Figure 6.8: Client-side system structure

6.7.2 Order history of the client

The other section mentioned is for the **history of orders placed by the connected user**. When the user accesses this section, the browser makes a request to the server to return a list of all orders that are linked to the user, regardless of their status.

The details of each order that will be specified in the history are: the name of the document associated with the order, the status of the order, the date when the order was placed and the price of the order. This data is added as an item to a list that will appear in the section. The **order information is updated dynamically** each time the user reloads the page.

When the status of the orders is complete, the list item is highlighted in green so that the user can identify the orders that are still pending.

6.8 Organization of the project

The section on the organization of the project contains an **extensive description of the structure of the system**, in terms of the scheme followed by the files that implement the system, as well as that of the database it uses, with the help of tables and diagrams that facilitate understanding of it.

6.8.1 System structure

In this section an analysis of the **file structure of the system** will be carried out, in order to differentiate which part belongs to the client side of the structure and which to the server

Server-side sructure		
/		
	app.yaml	Configuration file and mapping of handlers for requests to the Google App Engine application.
	index.yaml	File with the indexes of the queries to the application database.
	appengine_config.py	Configuration file to allow the use of libraries external to those included by Google App Engine.
	constants.py	File collection of used keys and general information messages in the application.
	utils.py	File that implements generic functionalities used at a general level in the application.
	test_runner	File implemented in bash for the execution of the tests.
	handlers/	Folder containing the handlers in charge of managing the requests received from the client side and composing the necessary responses.
	manager/	Folder containing the manager files in charge of accessing the entities required by the requests.
	models/	Folder containing the models in the database.
	lib/	Folder containing the external libraries used in the application.
	tests/	Folder containing the implementation of some unit tests.

Figure 6.9: Server-side system structure

side.

In the table 6.8, you specify the structure of the **DYN Print frontend**.

On the other hand, in the table 6.9, the **backend structure** of the application is specified.

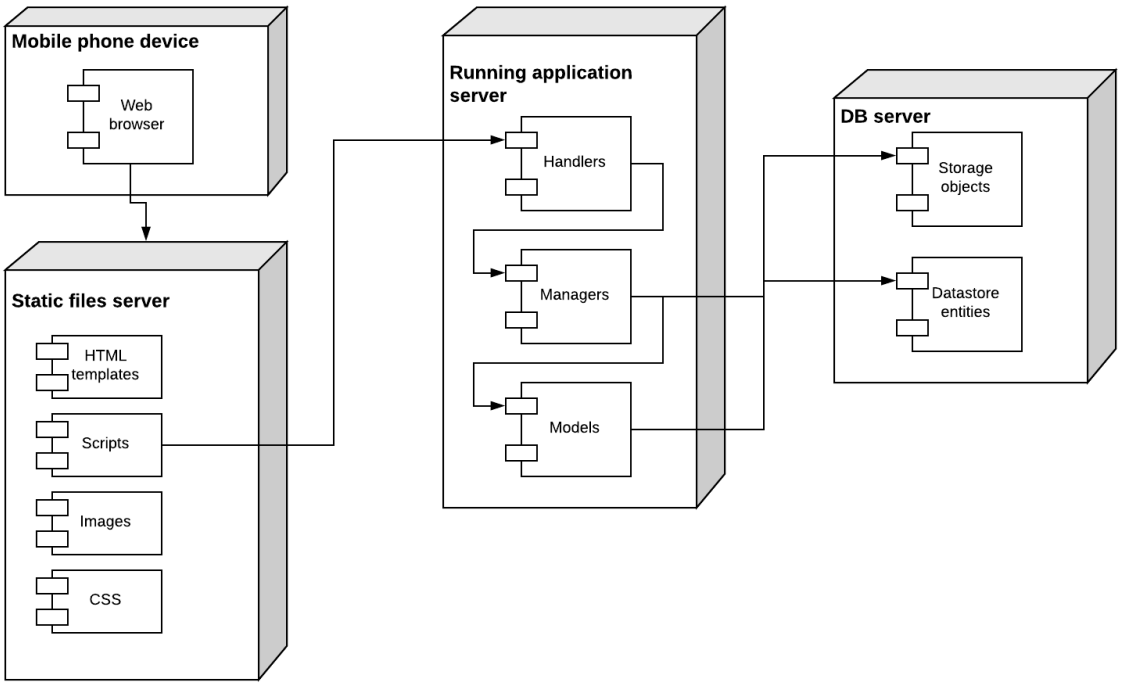

Figure 6.10: Deployment diagram of the system

Once a definition of the system structure has been made, it is essential to have a **generic view of how it is distributed**, taking into account that the infrastructure is offered by Google App Engine as well as the server-side organization scheme. This visual scheme is provided in the display diagram presented in the figure 6.10.

The DYN Print server side runs on instances created by Google App Engine, which are running on the world's distributed infrastructures on dedicated Google servers. The back-end scheme that GAE offers for the development of web applications is divided into **three fundamental layers: handlers, managers and models**.

- The **handlers** are in charge of receiving the requests that the server receives from the frontend, so that they constitute a way to access the server resources.
- **Managers** are in charge of providing a bridge between the requests and the information they demand, so that access to the application's database entities is made through them. These classes implement the Singleton design pattern.
- Finally, the **models** implement the database entities themselves, whose classes also include the queries made to the entity itself.

6.8.2 Front/end organization

The DYN Print frontend interface is organized in **HTML templates**, which are associated with each view of the different sections of the application. Along with these templates, the frontend has different **scripts in charge of the Javascript functionalities**, as well as

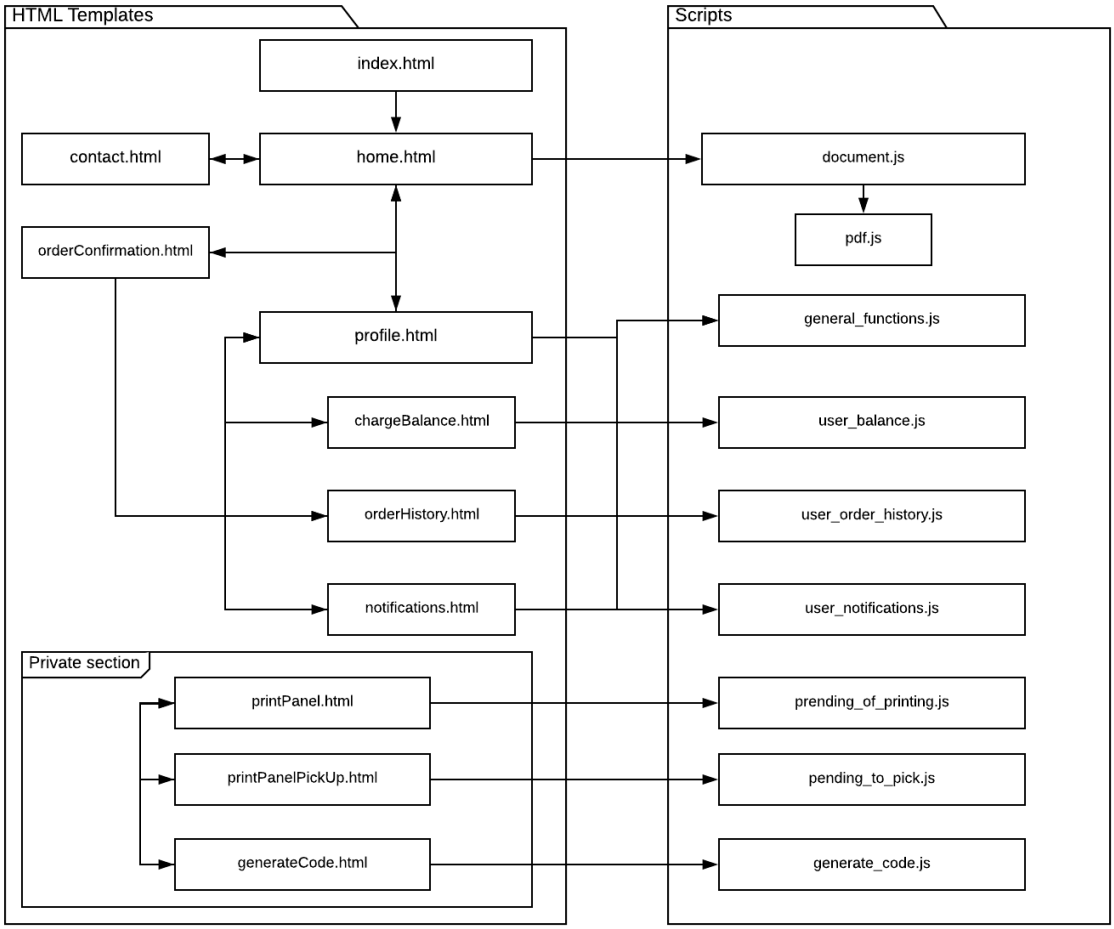

Figure 6.11: Scheme of the HTML templates of the system

performing the necessary interactions with the backend through AJAX requests. HTML templates use the **Bootstrap framework**, which applies responsive components to them. The figure 6.11 shows the relationship between the different templates available through the implementation of hyperlinks and also the relationship with the scripts that use each of them to maintain communication with the server.

On the other hand, the backend is formed by the classes that constitute the levels of handlers, managers and models. The figure 6.12 shows the **set of handlers that are implemented in the server**, so that the requests and their parameters received by each one of them are identified and the answers that are composed together with the corresponding parameters.

6.8.3 Database structure

As mentioned in the subsection 4.2.4, the Google App Engine database has an **object-based structure** which has some features that have great advantages, as well as being different from traditional approaches such as relational. GAE has several storage services, of which those used in DYN Print are Google Cloud Datastore and Google Cloud Storage

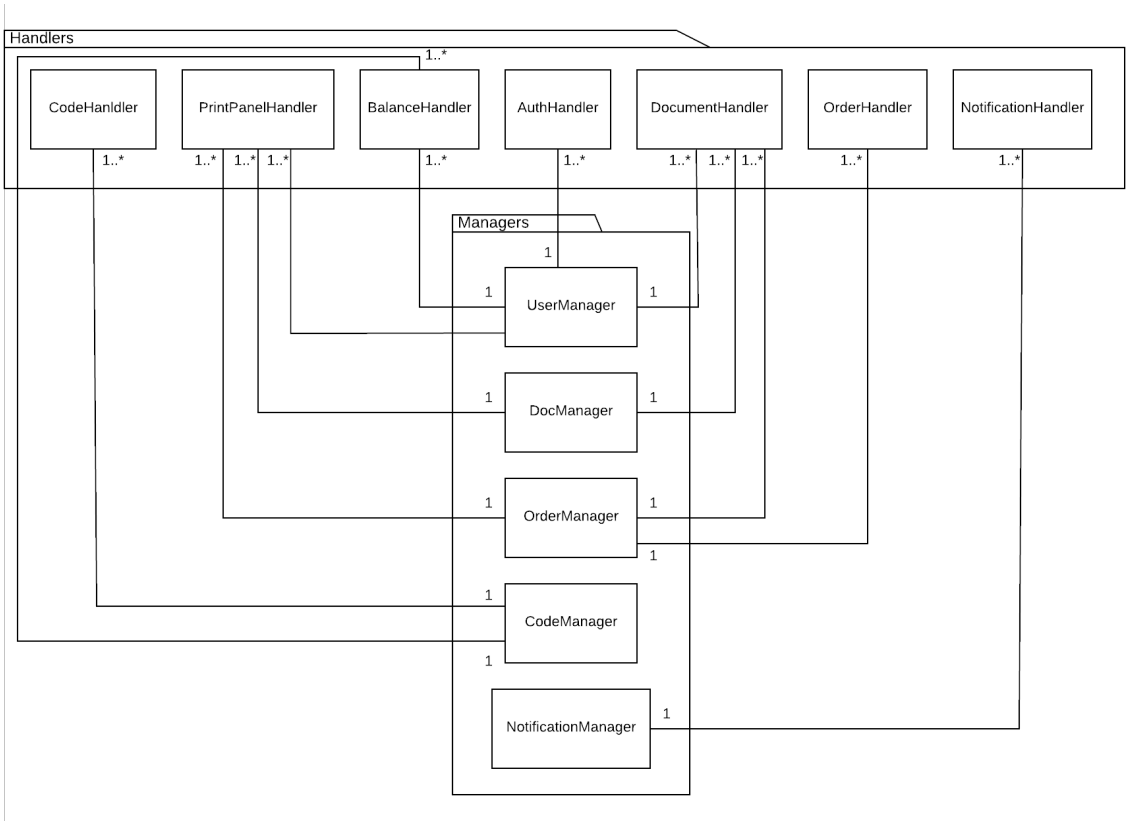

Figure 6.12: Class diagram of the handlers and managers

(which has no size limitation on its data).

The models that will be stored in Google Datastore, which belong to the backend models level, are classes that inherit from the class provided by Google App Engine ndb.Model. There are models that have a relationship between them, so that **a model contains a property that is a reference to an entity of another model**. These references are obtained thanks to the type of property called Reference that can be used with the unique keys generated when creating the entities.

Each property of **the entity has different attributes which define some characteristics that will affect the entities**. For example, for the reference property, the kind attribute can be specified, so that the reference will be made to entities that are of the kind defined in that attribute. Another example, for the String property, is the attribute choices, which defines the possible values that can be assigned to the property, restricting the assignment of another value that is not among those possible values.

The figure 6.13 shows all the **models that have been created for DYN Print** together with their properties and corresponding types.

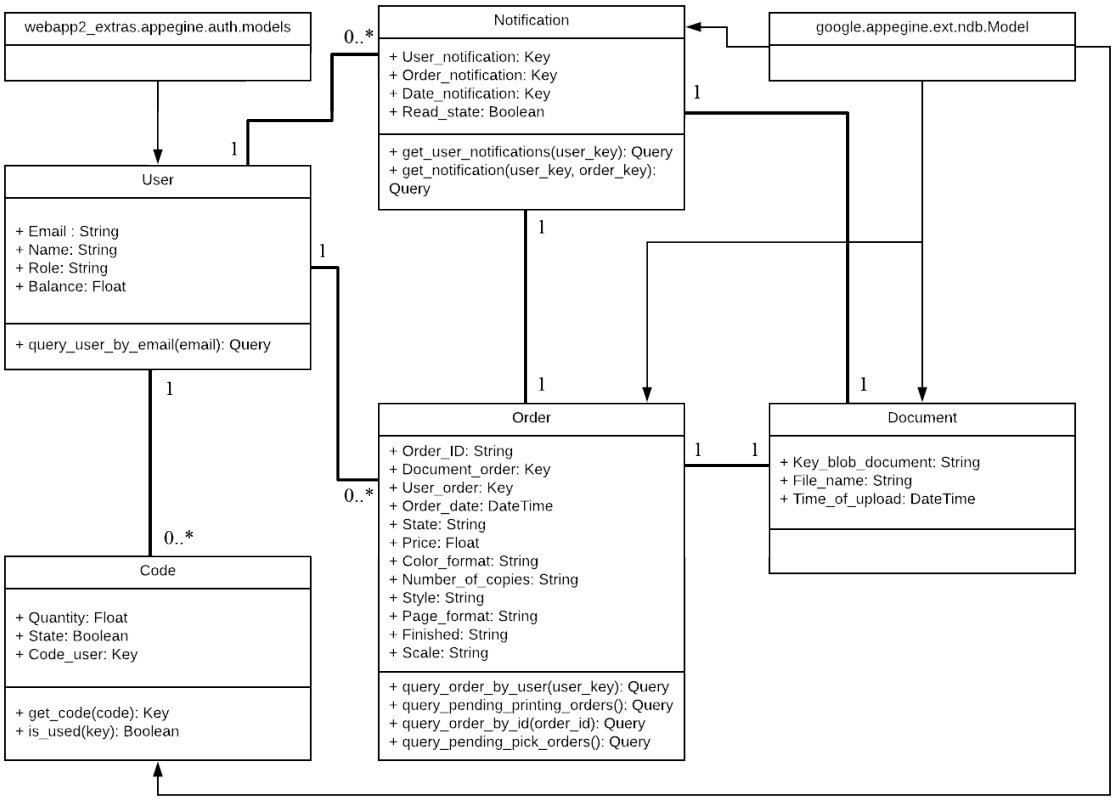

Figure 6.13: Class diagram of the models

On the other hand, the Google Cloud Storage service has had to be used to store the content of the documents, as most of them **exceed the 1MB size limit set for Google Cloud Datastore**. In Google Cloud Storage, objects are stored in a container that is installed in one of the locations where Google's servers work. Each application has a certain number of containers and each of them has a predefined route. In order to save objects in these contents it is necessary to define this path next to the name of the object to be saved, previously opening an access to this container.

Google Cloud Storage objects can be assigned a **content type**, which makes it easier to manage. For DYN Print, the objects that will be stored in this service are exclusively of the Multipurpose Internet Mail Extensions (MIME) application/pdf type, therefore, when dealing with them on the server it will be quite similar to file management in the Python language.

6.8.4 User stories

In this section, the definitions of the **user stories** are included.

ID	US_AUTH
Role	As... Client
Functionality	I want... to identify me in the system.
Utility	To... make orders and see information about my orders and my profile.
Approval	Authentication with Google +, Facebook and Twitter are allowed. There is not an own authentication process of the system. The user has not fill any form to sign up in the system.

ID	US_UPLOAD
Role	As... Client
Functionality	I want... to upload my files and select the printing parameters.
Utility	To... send my orders to the copy centre.
Approval	It will allow the client users to upload their files from the device, with PDF format. The parameters of the file are selected through a form. The parameters must be the format color, the number of pages, the style of the page, the format page, the finished and the scale of the content. A previsualization of the file content is showed along with a summary of the order information.

ID	US_COMPLETE_ORDER
Role	As... Employee.
Functionality	I want... to send the files to the queue of the printed and to change the state of the orders.
Utility	To... complete the orders of the clients.
Approval	A mechanism to send the files to the printer queue by the web browser and to select the parameters from it must be implemented. The employee needs to know which orders are pending of being printed and which ones are pending of being collected. A way of completing the orders – changing the state of the order to completed – which are pending of being collected by the clients when they are picked up.

ID	US_USER_NOTIFICATION
Role	As... Client.
Functionality	I want... to know when my order is ready to be collected.
Utility	To... inform the clients when the order state change to pending of being picked up.
Approval	A notification system that allows the client users to know when they have to go to the copy centre to pick up the results of the order will be included. The notification will alert the client users with an email, with the application interface and with a notification process of the web browser.

ID	US_EMPLOYEE_NOTIFICATION
Role	As... Employee.
Functionality	I want... to be notified when new orders are sent to the
Utility	To... send the corresponding files to the printer
Approval	There will be a mechanism of informing the employees, with the notifications API of the web browser, when new orders are sent to the system and their state is pending of being printed.

ID	US_BALANCE_CODES
Role	As... Employee.
Functionality	I want... to create balance codes for the clients.
Utility	To... sell the balance codes to the clients and to allow them to increase their virtual balance.
Approval	There will be a form which will allow the employees to create new codes. They have to introduce the quantity of the code and the email of the client user.

ID	US_USER_PROFILE
Role	As... Client.
Functionality	I want... to get the information of my profile.
Utility	To... know the quantity of my balance, to increase its quantity and to see a list of my orders.
Approval	In the profile section of the application, a subsection to consult the balance information and another subsection to see the order history are included. The subsection of the user balance will allow him to see the current quantity of the balance and to increase the quantity by introducing a balance code previously bought in the copy centre. The balance code must be associated with the user of the client. The subsection of the order history will list all the orders that the logged in user made. This list will collect the information about the order like the date, the price, the state and the name of the uploaded file.

Chapter 7

Results

THE results obtained during the development of this project are going to be specified in this chapter, after the definition of its objectives, the planification and working methodologies and the architecture and problems which compose DYN Print were established.

Firstly, the different functionalities of the DYN Print application are presented through their interface with some figures which show it and a briefly explanation of how they work. Secondly, the results regarding the working flow of the development process to produce the application are explained iteration by iteration. Every iteration has its own tasks which are classified and explained in detail.

7.1 Functionality and interface of the project

The development has been done through a special environment created by the Google App Engine service where all the functionalities are working to execute the application and to test it on the computer used for development. After the process is complete, then the deployment of the application at the Google servers is performed. The usage of the deployed application has its limitations due to the service fee which is hired by the developers. The service requests, the instance execution time and other Google App Engine features are bounded depending on the fee.

The hosting of the application is one of the provided services, so when the application is deployed, Google provides an URL where DYN Print is available to the users:

```
https://dyn-print.appspot.com/index.html
```

Hereunder, the capabilities of DYN Print following the figures representing them are specified. It is important to note that the interface of the application has been developed in the Spanish language because the main audience of DYN Print will be of that nationality.

- **Authentication:** the authentication of the user can be achieved through different alternatives, but all of them use a social media account, as it is shown in the figure 7.1. The available social media are Google+, Twitter and Facebook. In every of the options, the social media company request the user for permissions to provide the required user

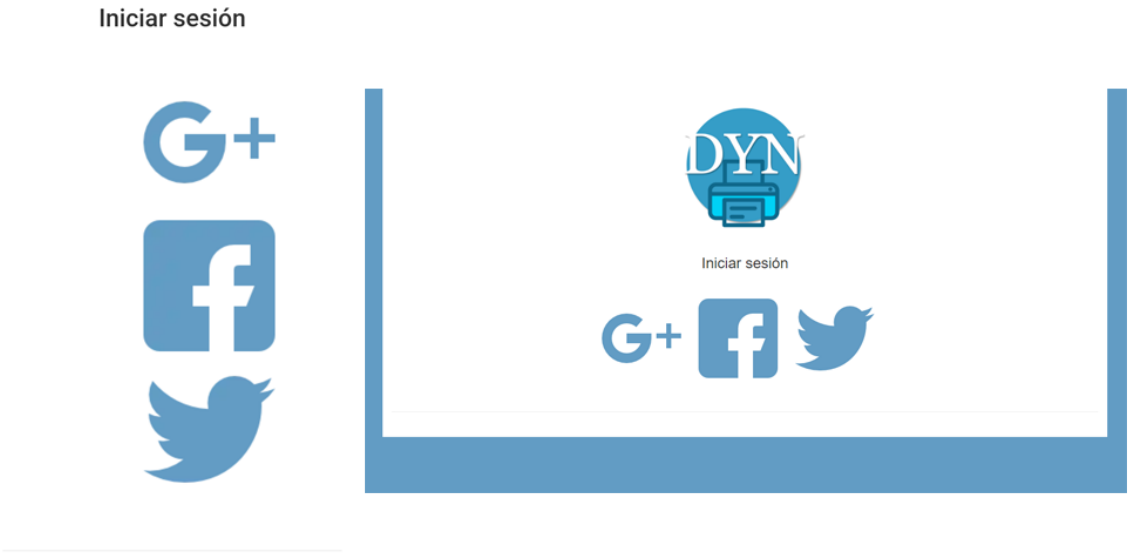

Figure 7.1: Authentication interface.

Figure 7.2: Selecting the file and the paramenters selection interfaces.

information to DYN Print. The information that the application will use is the user email and the name of the user profile. If the user gives the permissions, then a form is presented where the user introduces his credentials. Once the user guarantees authorization and logs in, the application itself redirects them to the DYN Print home page, where the user already has permissions to start ordering and managing their profile.

- **Upload a document and make an order:** files can be uploaded from the device file system and they must be uploaded one by one. The format of the documents to be uploaded must be PDF. After selecting the desired file, the user must choose the printing parameters. The choice of parameters is an important step in the process, the

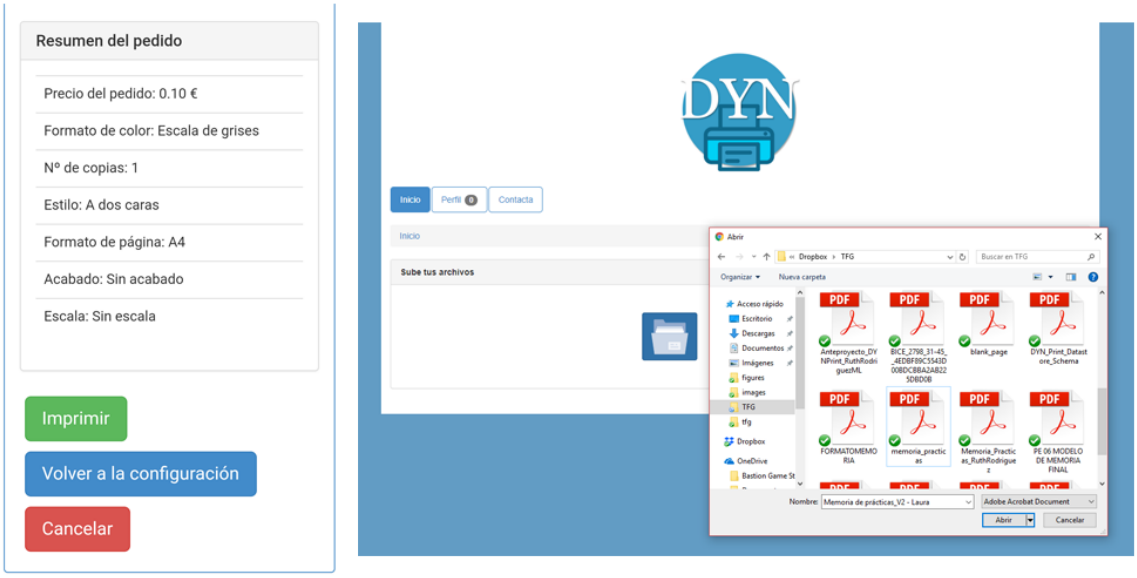

Figure 7.3: Order summary interface.

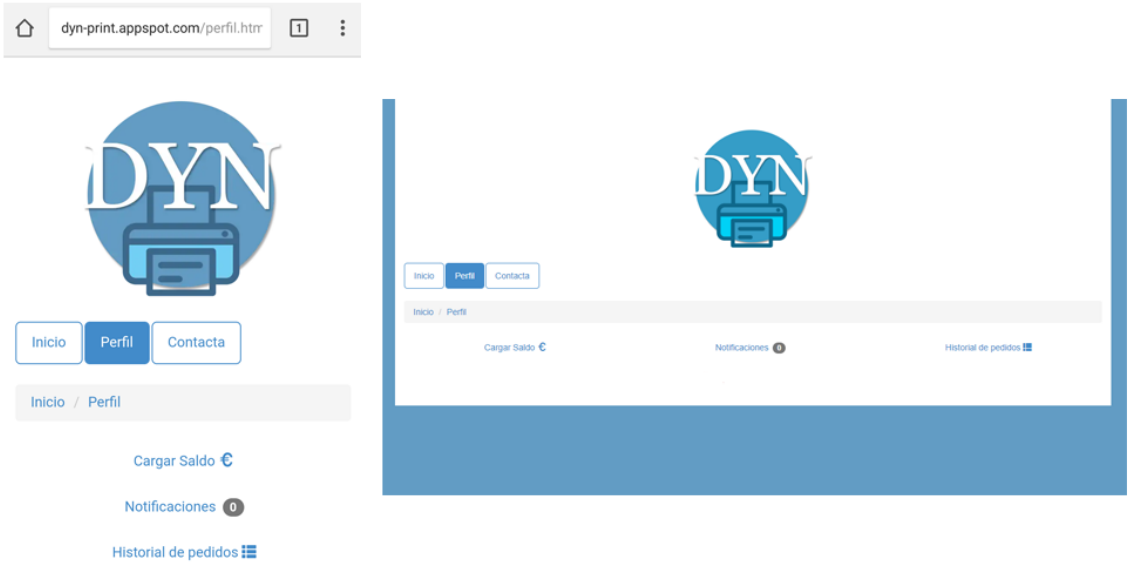

Figure 7.4: Profile section interface.

interface makes it easy to choose the desired values. The user has to choose the color format, the page format, the number of copies, the page style, the desired finished and the scale. Finally, a resume of the order is showed which includes the final price. The total price of the order is calculated taking into account the number of pages that constitute the document and the colour format of the pages. Also, a previsualization of the file is available for the user if he needs to see the result before sending it. All this process can be observerd in the figures 7.2 and 7.3.

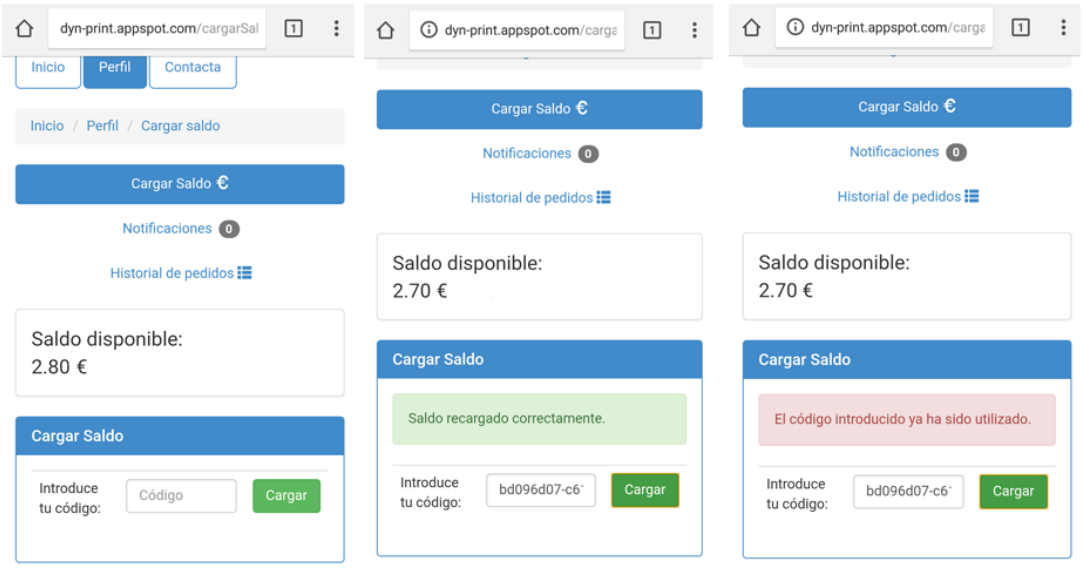

Figure 7.5: User balance section interface.

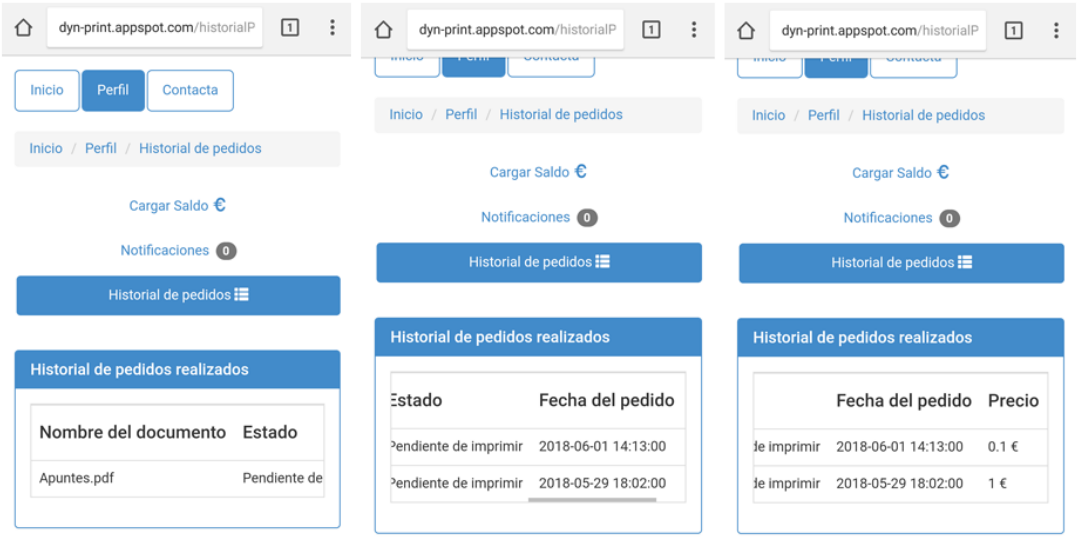

Figure 7.6: Order history of the user section interface.

- **User profile section:** there is a profile section available for the user where he can access to information related to his orders, his balance and his notifications. This section is just a gateway to the different subsections.

- **User balance charge:** the user must buy a balance code in the copy centre in order to increase his balance in the application. This balance is necessary to perform the orders because if the application detects that the user has not enough money in the balance to make an order, then the system will reject the order. In this section, shown in the figure 7.5, the user can see which the quantity of his balance. Also, he can introduce a code to increase the quantity. The code must be linked with the user and it can be repeated.

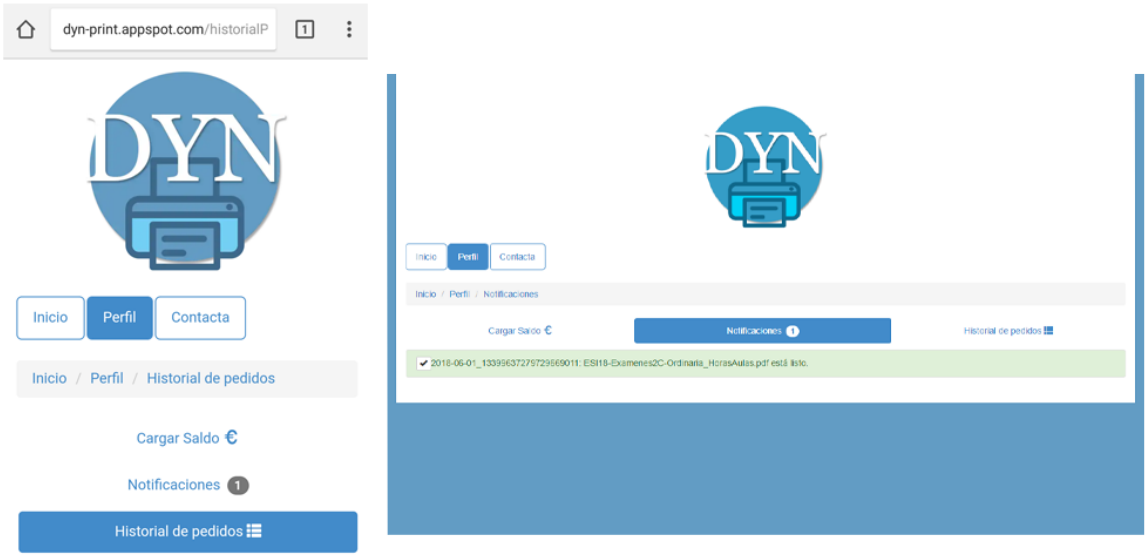

Figure 7.7: Notification section interface.

- **Order history panel:** another section in the user profile is the order history, where all the orders performed by the current user are listed along with all the information related to it, like the date, the state or the price. This list is constantly synchronized as soon as changes in orders are detected, so that the order status changes as soon as there is an update. The interface allows you to differentiate which orders are complete and which are not because they are highlighted differently. Its interface can be seen in the figure 7.6.

- **Notification panel:** there is a section in the user profile panel, whose interface is shown in the figure 7.7, which lists the notifications linked to the user specifying the name of file and the ID of the order in order to identify it easily. These notifications are generated when the files ordered by the users are printed and are ready to be picked up in the copy centre. The notifications that appear in this list can be deleted so that the user confirms that he or she has read them, and a confirmation dialog is presented before being removed from the list.

- **Balance code generator:** there is a private panel only available to users with permissions, which are the copy centre employees. This panel, which can be seen in the figure 7.8, is composed of three section, one of them is the balance code generator which produces random codes linked with the email client. These codes are a recharge to the virtual balance of the user to whom the code will be sold. With them they will be able to increase the amount of the virtual balance in order to make orders. Entering the user's email address is mandatory to establish a link between the code and the user, so that only the user can use it. Every code has a quantity, which can be five euros, ten euros or twenty euros.

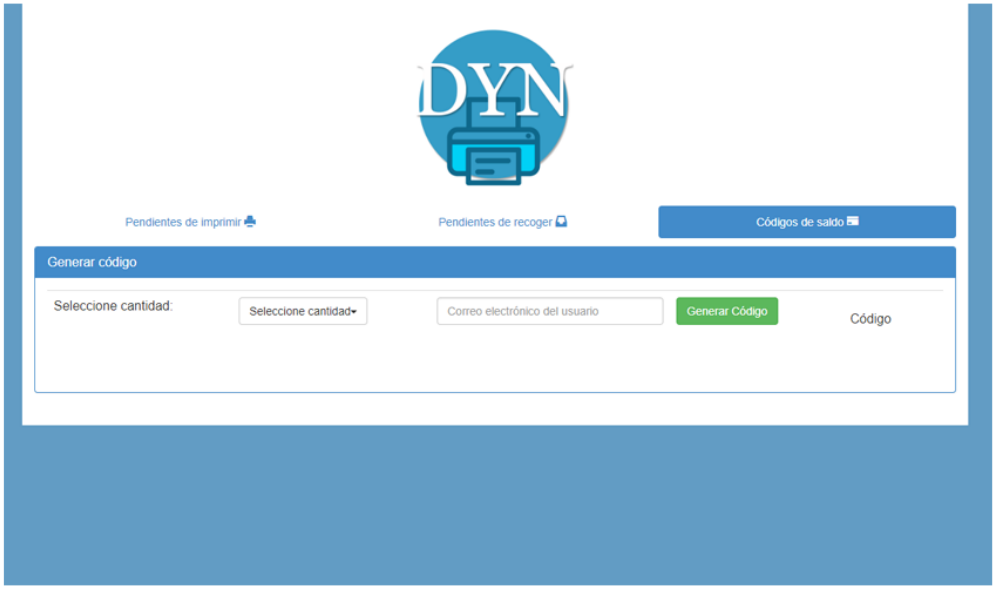

Figure 7.8: Balance code generator section interface.

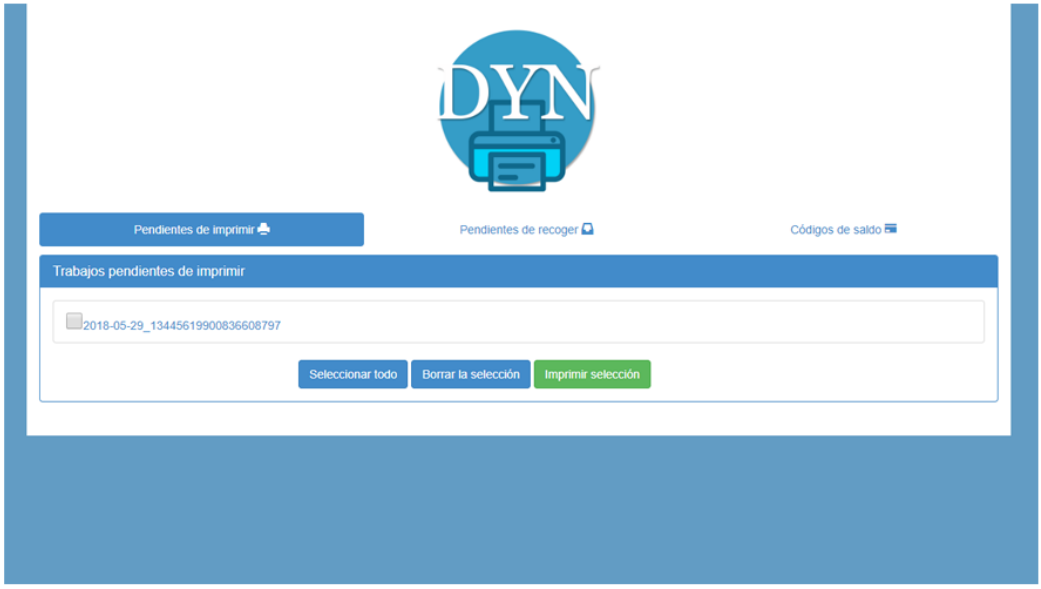

Figure 7.9: Panel of pending files to be printed.

- **Panel with files pending of being printed:** another section in the private panel is the list with the orders which are pending of being printed by the copy centre. Its interface is shown in the figure 7.9. The employee must choose the orders and select the print button. Then, for every order, a window in the web browser will open where the employee must choose the parameters and then send the document to the printer. It is not possible to send more than one document at a time to the printer queue due to the browser security system. They must therefore be sent one by one, although the interface allows all orders to be selected so that a window is automatically opened for each of them. It is also possible to download the contents of the orders to the employee's computer. When this process is complete, then the order disappears from

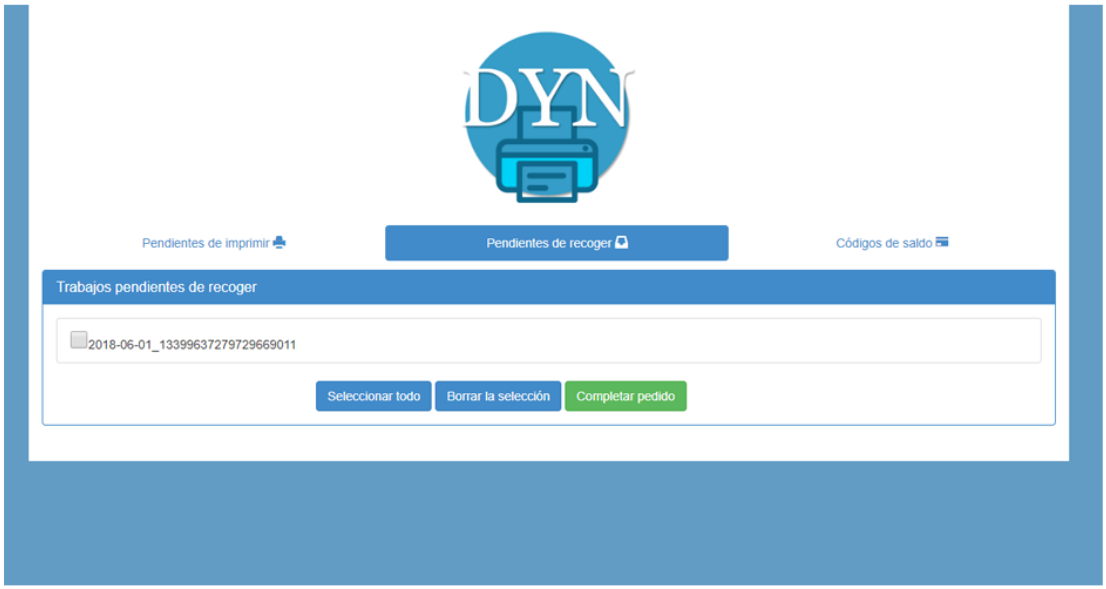

Figure 7.10: Panel of pending files to be collected.

the list. The prices that are set for orders cannot be set through this section, but are made on the server side of the system.

- **Panel with files pending of being picked up:** when the files are sent to the printer, then the state of the order changes and the files are listed in the other section of the private panel. A section whose list has the orders which are pending of being picked up by the users. An example can be seen in the figure 7.10. In this list the orders appear next to their ID, so that the employee can quickly identify them. In the configuration page that is attached to the orders there is the ID, so as soon as it is delivered to the user it is necessary to look at the identifier and locate it in this list to complete the order in the system.

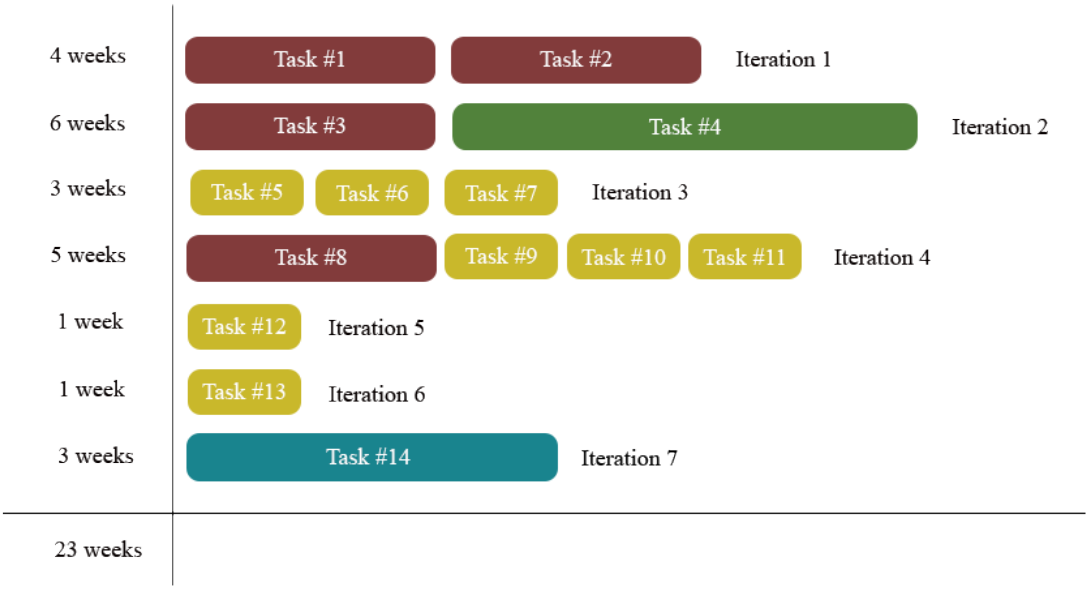

Figure 7.11: Visual representation of the work distribution

7.2 Work distribution

Consecutively, the iterations which was planned for the development of this project are explained one by one in detail to know how the required tasks have been achieved. A visual representation of the period of each iteration together with its corresponding tasks has been integrated in the figure 7.11, which also shows a count of the weeks invested in the development of the project. The implementation of DYN Print has been carried out from December 2017 to June 2018, taking into account that some weeks were not entirely dedicated to the development of the project. They are estimated at around 23 weeks and 7 hours per day, which means a work of about 35 hours a week.

7.2.1 Iteration 1: Preview of technologies

Task #1	A research about the technologies and the printing process.
Duration	2 week and 70 estimated hours.
A research about how the printing process and the printing configuration was made talking to some of the copy centres administrators, hence the requirements to develop the online printing process were defined. Furthermore, there was a documentation process about the available technologies and the optimal methodologies to develop the project. The decisions to use a cloud infrastructure for the system, to use an incremental and iterative working methodology and a project management methodology based on SCRUM and to follow a mobile first approach were made.	

Table 7.1: Iteration 1, task 1.

Task #2	A research about similar applications.
Duration	2 week and 70 estimated hours.
A work of researching about similar current working applications to the pretended project was accomplished. In this research, the different printing functionalities, the user management, the document management, the interface design, the payment process and the notification process provided by the applications were profoundly studied and documented.	

Table 7.2: Iteration 1, task 2.

7.2.2 Iteration 2: Interface design and implementation

Task #3	Drawing the interface schemes.
Duration	2 week and 70 estimated hours.
The design of the interface started having as key point that it must be responsive and must have a high level of usability to offer the user the best comfortable and useful navigation and usage of the application, mainly from their mobile devices. The first step was to draw the design of the different interfaces belonging to thedifferent processes of the system considering the use cases defined in the previous iteration and the interfaces of the existing responsive web applications studied previously. In consequence, the prototypes of the interface were obtained with the details about the responsive components considering the possible device screens and the convenience of the users when using the application. Also, it was essential to arrange the parameters of a printing process and how the it would be managed through the interface.	

Table 7.3: Iteration 2, task 3.

Task #4	Implementation of the interface.
Duration	4 weeks and 140 estimated hours.
The interface implementation was started. Instead of using a template which adjusts to the idea, its implementation was made totally from the beginning with HTML5, CSS3 and the framework Bootstrap. Finally, the suitable tests were performed in order to prove the adaptability of the interface in the device screens and to ensure the correct flow along the different pages.	

Table 7.4: Iteration 2, task 4.

7.2.3 Iteration 3: Upload and download modules implementation

Task #5	Familiarization with the technologies.
Duration	1 week and 35 estimated hours.
Google App Engine is a PAAS provided by Google which offers a lot of documentation about its services and its functions. A deep familiarization with this documentation and with the offered basic functionalities was performed before starting with the main modules of DYN Print. Some implemented examples were consulted to understand the behaviour of the components of the PaaS. Then, basic communications between the client and the server were developed to learn how they work in addition to validating the interactions with the Google App Engine services.	

Table 7.5: Iteration 3, task 5.

Task #6	Implementation of the upload module.
Duration	1 week and 35 estimated hours.
This task was focus on the functionality where the user must select the document which is going to be uploaded, then the printing parameters must be selected, and a resume of the order is presented to the user along with a previsualization of the document pages. The user must upload the documents and set its parameters one by one to simplify the process for the user. The documents are uploaded along with the parameters related to the order: number of copies, color format, page format, page style, finished and scale, which are collected by a form from the client-side. After defining the parameters of the order, a previsualization of the document is presented to the user with a Javascript library and the parameters defined previously are resumed to allow the user to confirm the order. This functionality needed the integration of the Google Datastore and the Google Cloud Storage by the server side. The Google Cloud Storage to upload the document itself because of its size and the Google Datastore to save an entity related to the order.	

Table 7.6: Iteration 3, task 6.

Task #7	Implementation of the download module.
Duration	1 week and 35 estimated hours.
For the server part of the system, some functional tests have been implemented to look through its functionalities and to provide a feedback for them. This part of the development procedure has been important to prove that the processes which include information of the system users work properly and the server deliver the processed data to the client in the correct way. These unit tests have been implemented with the help of a Google App Engine library and a special Python testing library.	

Table 7.7: Iteration 3, task 7.

7.2.4 Iteration 4: User management system implementation

Task #8	Authentication process implementation.
Duration	2 weeks and 70 estimated hours.
The development of the system is an incremental process because the functionalities are added continuously along with the already implemented ones. Thus, the authentication module is added to the system created in the tasks #6 and #7. The authentication of the user in the system is performed through several social media accounts, like Google+, Twitter and Facebook. The first one to be implemented was the Google+ option. To avoid complications with the management of the OAuth2 process, it was decided to use the services of Google Firebase, which allows the application developers only worry about the communication between the client and the server and selecting which data form the profile user is needed. Only the e-mail and the name of the users are saved in the system to avoid saving their passwords. A user entity is saved in the Google Datastore which collects the user email, name and its role inside the system. Users must be identified in order to use the functionality of uploading documents. In this way, every user is linked to the document that he uploads. Also, in the print panel, it is necessary that the user has the admin role and the privileges to access it. These privileges are provided to the copy centre employees.	

Table 7.8: Iteration 4, task 8.

Task #9	Balance management system implementation.
Duration	1 week and 35 estimated hours.
The next step was to integrate the balance management for the user profile and the order management. A balance is necessary to control the order prices and it is associated to every single user. A user must buy a balance code from the copy centre, which is generated from the print panel, where the employee must introduce the email user to synchronize it in the server. Moreover, the server is the attendant to generate the code number and to serve it to the client. In the application, a profile section is implemented where some other subsections are available to the user. In this task, the subsection related to the balance is implemented to presents to the user its current quantity of the balance and allows the user to introduce his code and to charge a new quantity in order to make orders. Conversely, the price of every order is calculated and presented along with the parameters chosen by the user. This order price is discounted to the user balance if the order finally is uploaded. An entity of an order related to the user and to the document is created in the Google Datastore.	

Table 7.9: Iteration 4, task 9.

Task #10	Order history management system implementation.
Duration	1 week and 35 estimated hours.
Other subsection available in the profile section is the order history. Once the orders are generated containing all the information related to it, then all the orders linked to a user can be retrieved to the client. The history order subsection was implemented in order to presents the user a list with his orders. This list contains information about the order like the date when it was performed, the price, the name of the document and the state of the order. Moreover, the state of the order can be changed between the following values: pending of printing in the copy centre, pending of picking up from the copy centre, completed order and cancelled order.	

Table 7.10: Iteration 4, task 10.

Task #11	Notification system implementation.
Duration	1 week and 35 estimated hours.
In this task, a notification system was implemented because of the necessity of informing, both the users and the employees, about the changes of the orders performed in the application. Firstly, the notifications when new orders are listed in the print panel were implemented through the Notification API standard, so the employees are informed through desktop notifications. Secondly, when an order is printed and is ready to be picked up by the corresponding client, the server is in charge of sending an email to the user to inform him of the order state change. On the other hand, a notification subsection in the profile section was implemented to notify the users through the application itself. This part of the application also uses the Notification API. The use of badges in the design of the web page allows the user to know when he has received a notification too.	

Table 7.11: Iteration 4, task 11.

7.2.5 Iteration 5: Optimization

Task #12	Optimization of the implementation.
Duration	1 week and 35 estimated hours.
When all the basic functionalities of the system were implemented, then it was necessary to perform some optimizations in order to reduce the response time of the system and to assure a reduction of the time when printing the documents from the web. Recent web browsers only allow the users to select the parameters and to print documents one by one. In order to improve the quickness of the process, it was decided to create a PDF page with the order information along with the parameters and to merge it to the rest of the PDF content. In this way, the first page of the document will have the information and selecting the parameters will be faster for the employees to print the documents. Google App Engine provides a service to program background processes. The Task Queue API was used in the server side to avoid an excessively response time when the user uploads his documents in the application. Thus, when the user uploads the document through the web, the confirmation is presented but the server creates a background task where the document is being merged and uploaded to the Google Cloud Storage.	

Table 7.12: Iteration 5, task 12.

7.2.6 Iteration 6: Testing

Task #13	Testing of some of the server-side functionalities.
Duration	1 week and 35 estimated hours.
For the server part of the system, some functional tests have been implemented to look through its functionalities and to provide a feedback for them. This part of the development procedure has been important to prove that the processes which include information of the system users work properly and the server deliver the processed data to the client in the correct way.	

Table 7.13: Iteration 6, task 13.

7.2.7 Iteration 7: Documentation

Task #14	Completing the documentation of the TFG.
Duration	3 weeks and 105 estimated hours.
Ultimately, the documentation of the project has been completed taking into account all the research which was accomplished at the beginning of the project and during the development of the different modules along with the required information about the degree project. A deep research about the Google App Engine services and their proper implementation into DYN Print has been performed. Besides that, the documentation regarding the external libraries concerned in the project has been achieved from some specialized books and information found in the Internet.	

Table 7.14: Iteration 7, task 14.

7.3 Analysis of the project

In this section, the analysis of the evolution of the version control of the application will be carried out together with the analysis of the cost invested in its development.

7.3.1 Versioning

The DYN Print project has had a monitoring of the changes made in its development through the distributed version control system Git, so that the files that make up the system have been constantly replicated with new features added. This version control provides security to the development because a copy of the system is always available from any device.

The repository system that integrates Git that has been used in the development of this project is Bitbucket. Thanks to the integration of their services in the development IDE, the commits have been easily and fluently carried out. Below are different graphs representing statistics on the commits made.

The figure 7.12 shows the relation of the month of the year by the number of committees carried out, where you can see that the months that have been most devoted to development have been April and May.

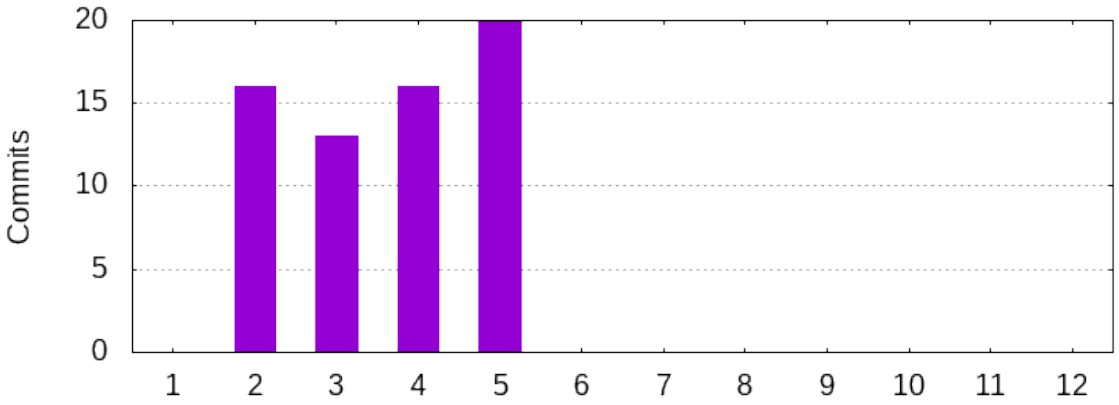

Figure 7.12: Number of commits per month

On the other hand, in the figure 7.13 it can be observed the distribution of the performed commits over the past 32 weeks.

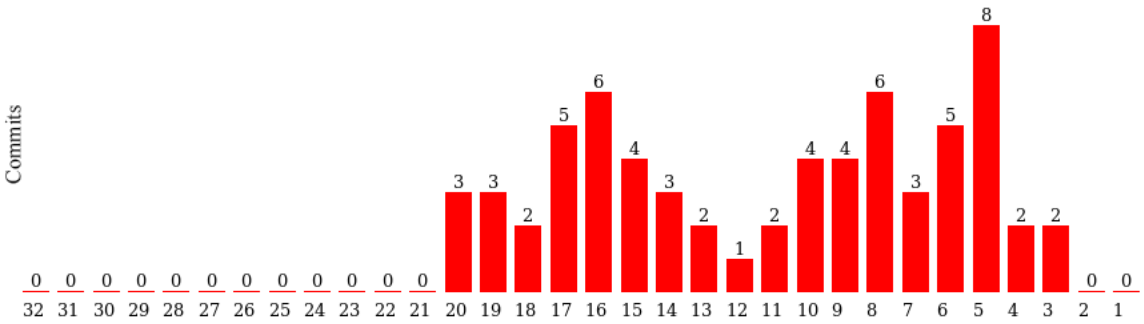

Figure 7.13: Number of commits for the past 32 weeks

7.3.2 Cost analysis

Considering that the average gross salary for a junior developer in Spain is around 24,000 euros per year, then it is estimated that the cost per hour of the day for this project is 12.25 euros. Taking into account that this project has been carried out in around 805 hours, since 23 weeks with 35 hours of work each were estimated, then **the salary calculated for the developer of this project is 9,861.25 euros**. On the other hand, the hardware resources used, and their cost are reflected in the table 7.15.

Resource	**Cost**
MSI CX61 2PC	699 euros
Huawei P9	359.99 euros
Monitor Acer V196HQLab 18,5" LED	69 euros
Acer Aspire XC-230	409 euros
HP K1500 Keyboard	9.99 euros
HP x1500 USB Mouse	6.99 euros
Total	1,553.97 euros

Table 7.15: DYN Print hardware resources cost

As mentioned in the section 4.2.4, the use of the Google App Engine tool is free of charge with a limit related to the number of transactions and the time spent on the resources offered. For this project, the free subscription fee has been used so that the limitations have not been exceeded, so no project costs have been allocated to using Google App Engine.

Therefore, the **total cost of the project considering all the above is 11,415.22 euros**.

Chapter 8

Conclusions

In this final chapter, a specification is made of the fulfilment of the objectives set out in the chapter 3, together with a list of possible lines of improvement to be integrated and a personal conclusion after the completion of the project.

8.1 Achieved objetives

With the development of DYN Print it has been possible to establish the provision of the printing service through an online platform whose performance meets the main objective of this project, since the entire process is automated so that the contribution of documents, the payment of orders and their management can be done through the application. The level of customer satisfaction increases due to the time and travel savings they experience, in addition to a simple and intuitive interface that facilitates the process. On the other hand, employee productivity increases by saving time in the order execution.

The degree of satisfaction with the degree to which the specific objectives defined for this project have been achieved will be analysed:

- **Time reduction:** to achieve a reduction in the time spent in the traditional printing process in copy centres through the automation of the same.

  Achieved? Yes. The printing process can be carried out through the online application from a device outside the copy centre. It is only necessary to have a web browser installed and to have the necessary amount in the virtual balance by purchasing a code beforehand. By performing this process from the browser, the customer saves time both in waiting for the queues of the copy centre, and in waiting for the documents to be printed at the same time, as they can upload the documents from their device at any time of the day. On the other hand, the employee saves time so that he or she does not have to wait for the printers to be ready to deliver the order at the time it is ordered and to prepare the money exchange. He or she can send incoming orders to print at the right time and deliver them without having to worry about payment. The fulfilment of this objective is reflected in the development of the `US_UPLOAD` and `US_COMPLETE_ORDER` user stories.

- **Intuitive interface:** it is necessary to implement an interface that is simple and intuitive to use for all types of users of the system, as it is intended to facilitate the ordering process, to avoid abandoning the process.

  Achieved? Yes. A research has been carried out on the current web applications and their interfaces in order to carry out one that is suitable for the facilitated use of DYN Print. The elements that the HTML language and the Bootstrap framework offer for the implementation of interfaces have served to achieve fluency in the navigation of the application, in order to highlight the elements necessary to help the user understand the process and its continuity, as well as providing elements that facilitate the choice of the printing configuration parameters. In addition, mechanisms have also been implemented by which the order process can be cancelled at any time, return to the configuration in case of errors and an alert system that asks for confirmation to avoid possible losses of the process.

- **Scalability:** DYN Print is designed as an online application that not only provides its service to a single copy centre, but to more than one at the same time, so that each user of each copy centre can access the service. Therefore, it is essential that the application supports the necessary scalability to handle all requests made by these users.

  Achieved? Yes. By implementing and deploying DYN Print in Google App Engine, the scalability and availability required by the application to respond to user requests is ensured.

- **Proactiveness:** to keep DYN Print users informed of updates applied to system orders, a notification system is required to define the proactivity of the application.

  Achieved? Yes. Compliance with this requirement has been achieved through the implementation of the `US_USER_NOTIFICATION` and `US_EMPLOYEE_NOTIFICATION` user histories.

- **Multi-device support:** it is estimated that DYN Print users use the application through a mobile device, therefore, it is necessary to develop a responsive interface that adapts to the screens of these devices.

  Achieved? Yes. Thanks to the use of the Bootstrap framework, the interface that has been developed for DYN Print accomplishes this objective, adapting its components to the different screen sizes in which the application can be accessed.

8.2 Future work

This section will define a series of ideas that can be applied to this project in order to improve and evolve it:

- The file format that DYN Print uses to create an order is exclusively PDF. Therefore, a module can be integrated to **perform a format transformation**, such as from docx to

PDF or from odf to PDF. There are many online applications that transform document formats, so when you send them a document and specify which format you want to transform it into, it sends back the response with the document in the new format. These requests could be implemented in the application when it detects that the file uploaded by the client does not have the PDF format. A more suitable option available is to perform this conversion from the server side using the LibreOffice program.

- To perform an **optimization of the print queue generated in the application** since web browsers do not allow more than one document to be sent to the printer queue at a time for security reasons. As detailed in the subsection 6.4.3, the idea can be based on collecting orders with the same configuration parameters in such a way as to put together a global document, where each document in an order is separated using the order information page. The server would be responsible for this task when sending orders with pending status to the employee, so that the characteristics of the orders are checked. Orders that have the same characteristics will be put together generating a new joint document, whose identifier can be created again just like an entity in the database. The employee will receive these files taking into account that they belong to different users. It is possible that there may be orders that have unique characteristics, therefore, there will be no need to join them with others. However, most orders have the same features, so more time will be saved when sending them to the printer queue, as they will be sent at the same time instead of one at a time.

- To allow the obtaining of virtual balance through the **integration of the Paypal payment service**, so that the users do not need to go to the copy shop to buy the balance codes in order to place orders in the application. To integrate the Paypal service, the company that manages it offers an API that facilitates its installation in the application, so that it creates a secure gateway for the customer when making payments. This service would be available to users through the section of the profile where you load balance. To use Paypal you only need to enter the credentials of the account and determine the amount you want to add to the balance. Once payment is authorized, the server will add this amount to the user's balance.

- To allow the user to choose their documents to **start an order from the Dropbox and Google Drive providers**, so that it is not necessary to have the documents on the device itself but in the cloud, as mentioned in the subsection 6.3.1. For the case of Google Drive it would be easier because users would be asked for additional permissions to access their Google Drive file folders when logging in with their Google Account. Its use is in the client side with the help of the API provided by Google Drive. On the other hand, Dropbox not only offers an API for the client side, but also provides it for the server. The Dropbox API would be in charge of the authorization process and access to the stored files.

8.3 Personal conclusion

Thanks to the development of this project I have been able to establish and better understand the knowledge acquired during my degree, especially the knowledge related to the intensification taken, Information Technologies. In addition to enriching this knowledge, the development of DYN Print has also given me the opportunity to acquire new development concepts, as well as the technologies used, which will undoubtedly serve me in the future.

From the very beginning, the development of this project has been a challenge for me, having to face the use of technologies that until now I had not had the opportunity to use. After the progress of the development period, I can confirm that I am more willing to continue learning concepts and procedures related to the technologies used.

Despite having dedicated as much time as possible to this project, the scope for improvement is still wide, but it undoubtedly leaves me with a feeling of effort, dedication and motivation that will allow me to improve and reach the level I have been fighting for since I found a passion and admiration in Computer Engineering.

Chapter 9

Conclusiones

En este capítulo final, se lleva a cabo una especificación del cumplimiento de los objetivos planteados anteriormente en el capítulo 3, junto a una enumeración de las posibles líneas de mejoras a integrar y a una conclusión personal tras la realización del proyecto.

9.1 Objetivos conseguidos

Con el desarrollo de DYN Print se ha podido establecer la prestación del servicio de impresión a través de una plataforma online cuyo rendimiento cumple con el objetivo principal de este proyecto, ya que todo el proceso está automatizado para que la aportación de documentos, el pago de los pedidos y su gestión se pueda realizar a través de la aplicación. El nivel de satisfacción del cliente aumenta debido al ahorro de tiempo y de desplazamientos que experimentan, además de una interfaz sencilla e intuitiva que facilita el proceso. Por otro lado, la productividad de los empleados aumenta al ahorrar tiempo en la ejecución del pedido.

Se va a proceder a realizar un análisis del nivel de satisfacción con el que se han alcanzado los objetivos específicos definidos para este proyecto.

- **Reducción de tiempo:** conseguir una reducción del tiempo empleado en el proceso de impresión tradicional en las copisterías a través de la automatización del mismo.

  ¿Conseguido? Sí. El proceso de impresión se puede llevar a cabo a través de la aplicación online desde un dispositivo ajeno a la copistería. Tan sólo es necesario tener instalado un navegador web y tener la cantidad necesaria en el saldo virtual comprando un código previamente. Al realizar este proceso desde el navegador, el cliente ahorra tiempo tanto en la espera de las colas de las copisterías, como en esperar a que los documentos estén impresos en ese mismo momento, ya que puede subir los documentos desde su dispositivo en cualquier momento del día. Por otro lado, el empleado ahorra tiempo de forma que no tiene que esperar a que las impresoras estén listas para entregar el pedido en el momento en el que se ha pedido y en preparar el cambio de dinero. Puede mandar los pedidos recibidos a imprimir en el momento oportuno y entregarlos sin necesidad de preocuparse del pago. El cumplimiento de este objetivo se refleja en el desarrollo de las historias de usuario `US_UPLOAD` y `US_-`

COMPLETE_ORDER.

- **Interfaz intuitiva:** es necesario implementar una interfaz que sea sencilla e intuitiva de usar por parte de todos los tipos de usuarios del sistema, ya que se pretende facilitar el proceso de pedido, de forma que se evite el abandono del proceso.

  ¿Conseguido? Sí. Se ha realizado una investigación sobre las aplicaciones web actuales y sus interfaces para llevar a cabo una que se adecue al uso facilitado de DYN Print. Los elementos que el lenguaje HTML y el framework Bootstrap ofrecen para la implementación de interfaces han servido para conseguir una fluidez en la navegación de la aplicación, para poder destacar los elementos necesarios para ayudar al usuario a entender el proceso y su continuidad, además de aportar elementos que faciliten la elección de los parámetros de configuración de impresión. Además, también se han implementado mecanismos por los cuales se puede cancelar el proceso de pedido en cualquier momento, volver a la configuración en caso de equivocaciones y un sistema de alerta que pide confirmación para evitar posibles pérdidas del proceso.

- **Escalabilidad:** se plantea DYN Print como una aplicación online que no solo preste su servicio a una copistería en singular, sino a más de una al mismo tiempo, de forma que cada usuario de cada copistería acceda al servicio. Por tanto, es imprescindible que la aplicación soporte la escalabilidad necesaria para atender a todas las peticiones realizadas por dichos usuarios.

  ¿Conseguido? Sí. Al implementar y desplegar DYN Print en Google App Engine, se garantiza la escalabilidad y la disponibilidad requerida por la aplicación para satisfacer las peticiones realizadas por los usuarios.

- **Proactividad:** para mantener a los usuarios de DYN Print informados de las actualizaciones aplicadas a los pedidos del sistema, es necesario un sistema de notificación que defina la proactividad de la aplicación.

  ¿Conseguido? Sí. El cumplimiento de este requisito se ha conseguido gracias a la implementación de las historias de usuario US_USER_NOTIFICATION y US_EMPLOYEE_NOTIFICATION.

- **Soporte multi dispositivo:** se estima que los usuarios de DYN Print usen la aplicación a través de un dispositivo móvil, por tanto, es necesario que se desarrolle una interfaz *responsive* que se adapte a las pantallas de dichos dispositivos.

  ¿Conseguido? Sí. Gracias a la utilización del framework Bootstrap, la interfaz que se ha desarrollado para DYN Print cumple este objetivo, adaptando sus componentes a los diferentes tamaños de pantallas en las que se puede acceder a la aplicación.

9.2 Trabajo futuro

En este apartado se definirán una serie de ideas que pueden aplicarse a este proyecto de forma que supongan una mejora y evolución:

- El formato de archivos que DYN Print usa para crear un pedido es exclusivamente PDF. Por tanto, puede integrarse un módulo encargado de realizar una **transformación de formato**, como puede ser de docx a PDF o de odf a PDF. Existen gran cantidad de aplicaciones online que transforman los formatos de los documentos, de forma que al enviarles un documento y especificarles a qué formato se quiere transformar, envía de vuelta la respuesta con el documento en el nuevo formato. Estas peticiones podrían implementarse en la aplicación cuando detectase que el archivo subido por el cliente no tiene el formato PDF. Una opción más adecuada es la de realizar esta conversión desde el lado del servidor haciendo uso del programa LibreOffice.
- Realizar una optimización de la cola de impresión que se genera en la aplicación puesto que los navegadores web no permiten el **enviar más de un documento a la vez a la cola de la impresora** por motivos de seguridad. Tal y como se detalla en la subsección 6.4.3, la idea puede basarse en recopilar los pedidos que tengan los mismos parámetros de configuración de forma que junten un documento global, donde cada documento de un pedido se separe gracias a la página de información sobre el pedido. De esta tarea se encargaría el servidor a la hora de enviar los pedidos con estado de pendiente al empleado, de forma que se comprueben las características de los pedidos. Los pedidos que tengan las mismas características se juntarán generando un nuevo documento conjunto, cuyo identificador puede crearse de nuevo igual que una entidad en la base de datos. El empleado recibirá estos archivos teniendo en cuenta que pertenecen a diferentes usuarios. Cabe la posibilidad que haya pedidos que tengan características únicas, por tanto, no habrá necesidad de juntarlos con otros. Sin embargo, la mayoría de los pedidos tienen las mismas características, por tanto, ahorrará más tiempo aún a la hora de enviarlos a la cola de la impresora, ya que los enviará a la vez en vez de uno en uno.
- Permitir la obtención de saldo virtual a través de la **integración del servicio de pago Paypal**, de forma que los usuarios prescindan de la necesidad de ir a la copistería a comprar los códigos de saldo para poder realizar pedidos en la aplicación. Para integrar el servicio de Paypal, la empresa que lo gestiona ofrece una API que facilita su instalación en la aplicación, de forma que crea una pasarela segura para el cliente a la hora de realizar los pagos. Este servicio estaría disponible para los usuarios a través de la sección del perfil donde carga saldo. Para utilizar Paypal tan solo necesitaría introducir las credenciales de la cuenta y determinar la cantidad que quiere sumar al saldo. Una vez autorizado el pago, el servidor se encargaría de sumar dicha cantidad al saldo del usuario.

- Permitir que el usuario pueda **elegir sus documentos para empezar un pedido desde los proveedores Dropbox y Google Drive**, de forma que no sea necesario tener los documentos en el propio dispositivo sino en la nube, tal y como se ha mencionado en la subsección 6.3.1. Para el caso de Google Drive sería más sencillo debido a que los usuarios, al iniciar sesión con su cuenta de Google, se le pedirían los permisos adicionales para acceder a sus carpetas de archivos de Google Drive. Su implementación también se llevaría a cabo en el lado del cliente con la ayuda de la API que Google Drive facilita. Por otro lado, Dropbox no solo ofrece una API para el lado del cliente, sino que también la facilita para el servidor. La API de Dropbox se encargaría de realizar el proceso de autorización y de acceso a los archivos almacenados.

9.3 Conclusión personal

Gracias al desarrollo de este proyecto he conseguido asentar y entender mejor los conocimientos adquiridos a lo largo de la carrera, en especial los conocimientos relacionados con la intensificación tomada, Tecnologías de la Información. Además de enriquecer estos conocimientos, el desarrollo de DYN Print también me ha brindado la posibilidad de adquirir nuevos conceptos de desarrollo, así como de las tecnologías utilizadas, los cuales, sin duda, me servirán en un futuro.

Desde el primer momento, el desarrollo de este proyecto ha supuesto para mí un reto, teniendo que afrontar el uso de tecnologías que hasta ahora no había tenido la oportunidad de usar. Tras el transcurso del periodo de desarrollo, puedo confirmar que termino con más ganas de seguir aprendiendo conceptos y procedimientos relacionados con las tecnologías utilizadas.

A pesar de haber dedicado el mayor tiempo posible a este proyecto, el margen de mejora aún es amplio, pero sin duda deja en mí un sentimiento de esfuerzo, dedicación y motivación que me permitirá mejorar y alcanzar el nivel por el que llevo luchando desde que encontré en la Ingeniería Informática una pasión a la vez que nua admiración.

APPENDIXES

Appendix 1: Google App Engine application configuration file

EVERY application implemented in Google App Engine with the Python programming language needs a configuration file, named app.yaml, that collects the main features of the project, as well as the location of the system resources that will process the requests received on the server. In addition, this file can also define the libraries and their versions to be used that are available in the Google App Engine development environment.

For the defined characteristics, it is possible to include the runtime language, the application version and the application ID. The handlers are then defined, which are the applications in Python in charge of handling the requests received from the specified url. In the script field you define the mapping to the application.

On the other hand, static system files can also be mapped in this section, so that when the application runs, the static files are uploaded to a cache for this type of files. The builtins option refers to the handlers that the Google App Engine Python SDK includes for some common features. For DYN Print, the deferred handler has been enabled to use the tasks in the background.

The listing 1 shows the app.yaml file configured for the DYN Print application.

```
runtime: python27
api_version: 1
threadsafe: true

libraries:
  - name: webapp2
    version: "2.5.1"
  - name: PIL
    version: "1.1.7"

builtins:
  - deferred: on

handlers:
  - url: /
    script: handlers.auth.application

  - url: /signUpIn
    script: handlers.auth.application

  - url: /uploadDocument
    script: handlers.upload_document.application

  - url: /printPanel
    script: handlers.print_panel.application

  - url: /balance
    script: handlers.user.application

  - url: /generateCode
    script: handlers.user.application

  - url: /orders
    script: handlers.user.application

  - url: /userNotification
    script: handlers.user.application

  - url: /css
    static_dir: static/css

  - url: /images
    static_dir: static/images

  - url: /scripts
    static_dir: static/scripts

  - url: /_ah/queue/deferred
    script: google.appengine.ext.deferred.deferred.application
    login: admin

  - url: /(.*\.html)
    mime_type: text/html
    static_files: static/\1
    upload: static/(.*\.html)
```

Listado 1: app.yaml for DYN Print

Appendix 2: Interface Mockups

BEFORE starting the implementation of the DYN Print interface, a series of pencil mock-ups were made on paper to get an idea of the initial design. These prototype interfaces were designed with the screen size of a mobile phone in mind, as development would focus on prioritizing the use of users through these devices. In addition, the interfaces that were defined were in line with the basic functionalities that were established for the system from the beginning.

The final design of the interface differs from these mockups, showed in the figures 1, 2 and 3, in that as they were implemented, the needs and requirements of the system changed.

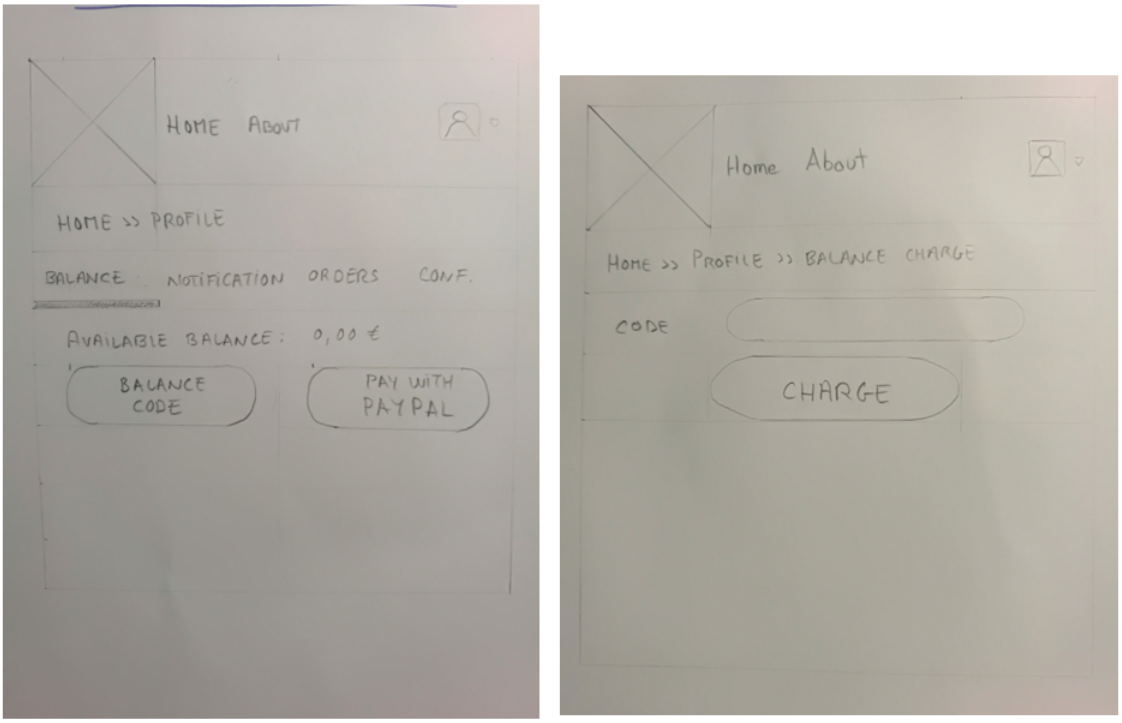

Figure 1: Mockups of the balance management interface

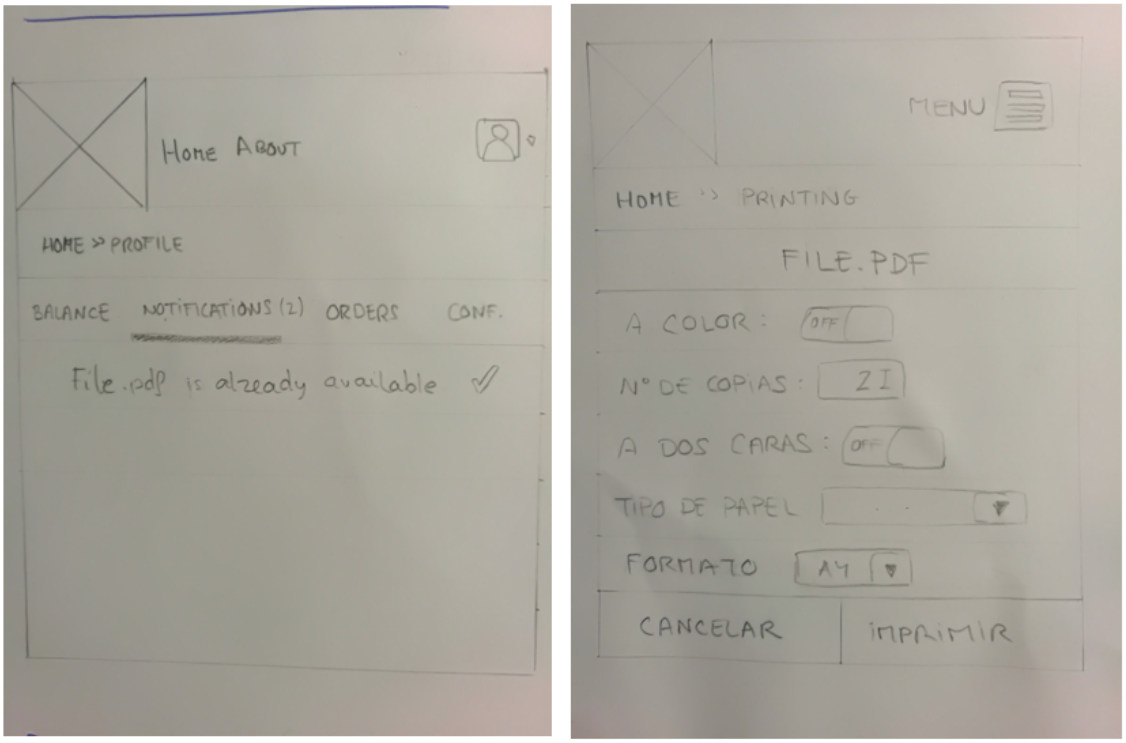

Figure 2: Mockups of the file configuration and notification interfaces

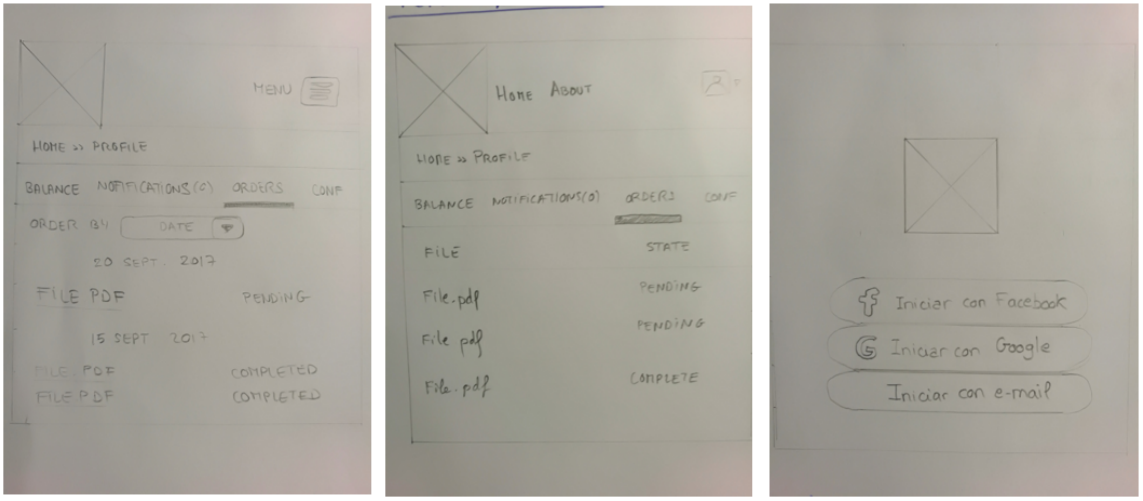

Figure 3: Mockups of the order history and log in interfaces

References

[A.92] Berson A. *Client/Server Architecture (J. Ranade Series on Computer Communications)*. McGraw-Hill, 1992.

[AD13] Fox A. y Patterson D. *Engineering Software as a Service: An Agile Approach Using Cloud Computing*. Strawberry Canyon LLC, 2013.

[Bha00] A. Bhattacherjee. Acceptance of e-commerce services: the case of electronic brokerages, July 2000. url: `https://ieeexplore.ieee.org/abstract/document/852435/`.

[BKD01] Van Bennekum A.-Cockburn A. Cunningham W. Fowler M. Grenning J. Highsmith J. Hunt A. Jeffries R. Kern J. Marick B. Martin R. C. Mellor S. Schwaber K. Sutherland J. Beck K., Beedle M. y Thomas D. *Manifiesto for Agile Software Development*. Agile Alliance, 2001.

[CJD16] Fenton N. Curran J. y Freedman D. *Misunderstanding the Internet (Communication and Society)*. Routledge Taylor & Francis Group, 2016.

[CV03] Larman C. y Basilli V.R. Iterative and Incremental Development: A Brief History, Junio 2003. url: `https://ieeexplore.ieee.org/abstract/document/1204375`.

[D.15] Sanderson D. *Programming Google App Engine with Python: Build and Run Scalable Python Apps on Google's Infrastructure*. O'Reilly Media, 2015.

[DPGE04] A. Del Pino González y Vázquez Yánez E. El comercio electrónico en España. *Economic Newsletter of the ICE*, 2798:31–45, 2004.

[Fla11] David Flanagan. *JavaScript: The Definitive Guide: Activate Your Web Pages (Definitive Guides)*. O'Reilly Media, 2011.

[For09] Behrouz A. Forouzan. *TCP/IP Protocol Suite (Mcgraw-hill Forouzan Networking)*. McGraw-Hill Education, 2009.

[FPAR15] Jiménez-Naranjo H. V. Fernández-Portillo A., Sánchez-Escobedo M. y Hernández-Mogollón R. La importancia de la Innovación en el Comercio Electrónico. *Universia Bussiness Review*, 47:106–125, 2015.

[G.08] Lawton G. Developing Software Online with Platform-as-a-Service Technology, Junio 2008. url: https://ieeexplore.ieee.org/abstract/document/4548165.

[Gar12] Javier Garzás. *Gestión Ágil de Proyectos Software*. Kybele Consulting, 2012.

[Gas11] Peter Gasston. *The Book of CSS3: A Developer's Guide to the Future of Web Design*. No Starch Press, 2011.

[Gef02] David Gefen. Customer Loyalty in E-Commerce. *Journal of the Association for Information Systems*, 3:27–51, 2002.

[KS03] Zahn C. Kumar S. Mobile Communications: Evolution and Impact on Business Operations. *Technovation*, 23(6):515–520, 2003.

[Lom15] Luis Lombardero. *Trabajar en la era digital: Tecnología y competencias para la transformación digital (Acción empresarial) (Spanish Edition)*. LID Editorial, 2015.

[LSA11] Peter Lubbers, Frank Salim, y Brian Albers. *Pro HTML5 Programming: Powerful APIs for Richer Internet Application Development (Expert's Voice in Web Development)*. Apress, 2011.

[Mur08] James Murty. *Programming Amazon Web Services: S3, EC2, SQS, FPS, and SimpleDB*. O'Reilly Media, 2008.

[NMC13] S. Jagannath Niranjanamurthy M., Kavyashree N. y Dharmendra C. Analysis of E-Commerce and M-Commerce: Advantages, Limitations and Security issues. *International Journal of Advanced Research in Computer and Communication Engineering*, 2(6):2360–2370, 2013.

[R.02] Fielding R. Principled design of the modern Web architecture. *ACM Transactions on Internet Technology (TOIT)*, 2(2):115–150, 2002.

[Ric99] Jeffrey Richter. *Programming Applications for Microsoft Windows (Dv-Mps General)*. Microsoft Press, 1999.

[San12] Dan Sanderson. *Programming Google App Engine: Build & Run Scalable Web Applications on Google's Infrastructure*. O'Reilly Media, 2012.

[Sch04] Ken Schwaber. *Agile Project Management with Scrum (Developer Best Practices)*. Microsoft Press, 2004.

[TM01] Lee-Bernes T. y Fischetti M. *Weaving the Web: The Original Design and Ultimate Destiny of the World Wide Web by Its Inventor*. DIANE Publishing Company, 2001.

[Yah17] Houssem Yahiaoui. *Firebase Cookbook: Over 70 recipes to help you create real-time web and mobile applications with Firebase*. Packt Publishing, 2017.

Este documento fue editado y tipografiado con LaTeX empleando
la clase **esi-tfg** (versión 0.20180627) que se puede encontrar en:
`https://bitbucket.org/arco_group/esi-tfg`