UNIVERSIDAD POLITÉCNICA DE VALENCIA

Escuela Técnica Superior de Ingeniería Informática

“Utilización de un marco de persistencia objeto relacional en el diseño de aplicaciones web multicapa”.

PROYECTO FINAL DE CARRERA (PFC)

Presentado por:

Pedro Alfaro Fernández

Dirigido por:

Juan Sánchez Díaz

Valencia Julio 2012

Índice:

1. Introducción

Esta es la memoria descriptiva del proceso de desarrollo del proyecto final de carrera de Ingeniería Informática realizado en la Universidad Politécnica de Valencia.

Realizado por Pedro Alfaro Fernández, dirigido y supervisado por Juan Sánchez Díaz.

1.1. Motivación

Profesionalmente me interesaba profundizar en la utilización de "object relational mapper" (de ahora en adelante ORM) no solo debido a ser herramientas potentes que simplifican y mejoran el trabajo del programador sino también debido a la aparición en relativamente poco tiempo de muchos sistemas ORM y proyectos que hacen uso de estas herramientas en su desarrollo.

La utilización de arquitecturas multicapa en el diseño de aplicaciones Web permite independizar la capa de interfaz de usuario, de la capa de negocio y de la capa de persistencia o almacenamiento de datos, de modo que las modificaciones en el sistema quedan confinadas a la capa objeto de modificación. Esta característica permite obtener aplicaciones más robustas y más fácilmente modificables.

Se ha elegido una tienda web online debido a la enorme expansión que ha experimentado el comercio online y que permite utilizar una temática bien conocida claramente enfocada a gestionar una Base de Datos y un sistema persistente. Así mismo el hecho de ser un escenario conocido permitirá ver más claramente las diferencias en el proceso de desarrollo de la aplicación.

1.2. Objetivos

Personalmente el objetivo era la ejecución propiamente dicha de un desarrollo para aprender el manejo de las tecnologías (como los ORMs) y como afectarían al desarrollo habitual de software. Utilizando el framework Symfony junto con algunas de sus herramientas así como el ORM Doctrine.

El objetivo principal del proyecto es utilizar un marco de persistencia objeto relacional en el diseño de una aplicación multicapa en la cual se emplea PHP como lenguaje de programación. Como caso de estudio para evaluar la utilidad del marco de persistencia se implementará una aplicación web que gestionará una tienda de compras en línea en Internet.

Se busca aplicar un sistema de mapeo objeto relacional para la gestión de la persistencia en las aplicaciones Web multicapa estándar añadiendo las capas necesarias y utilizando bibliotecas necesaria para ello. Creando de facto una base de datos orientada a objetos "virtual" que sobre la base de datos relacional permita su utilización de forma más natural.

1.3 Contexto

Es necesaria la existencia de diferentes tipos de usuarios con diferentes permisos. Los usuarios quedan así clasificados en **Administrador** el encargado de la gestión general del portal, **Clientes** aquellas personas registradas que van a realizar compras y **Visitantes** personas sin registrar que solo desean consultar o hojear los productos.

1.3.1 Estructura del documento

Introducción: Capitulo que presenta el proyecto realizado, dando detalles sobre los objetivos y motivaciones del mismo y expone el planteamiento del problema.

Especificación de requisitos: Capitulo que explica y recoge los diferentes requisitos que tiene este proyecto. Me guiaré por estándares que permitan la extracción de todos estos requisitos de forma fiable y rigurosa.

Análisis: Capitulo que enseña las fases de análisis que han sido necesarias para obtener el modelo conceptual. El centro de este capitulo orbita alrededor del Diagrama de clases de la aplicación.

Diseño: Capitulo que muestra los diseños realizados, las metodologías utilizadas y el modelo relacional de la base de datos.

Implementación: Capitulo que describe y comenta las diferentes herramientas y tecnologías utilizadas en la implementación del proyecto.

Bibliografía: Apartado donde se reflejan las principales fuentes consultadas para la elaboración de este documento.

Manual de instalación: Capitulo que expone las técnicas de evaluación y pruebas del sistema.

2. Especificación de requisitos

La especificación de requisitos del sistema que se pretende construir está basada en el estándar IEEE830-1998, esta especificación puede ser interpretada como un contrato entre clientes y desarrolladores.

2.1 Introducción

Esta especificación de requisitos pretende marcar de forma clara, concisa y completa el funcionamiento, las funcionalidades y características del software a desarrollar.

2.1.1 Propósito

El propósito de esta especificación de requisitos es reunir y plasmar el comportamiento del sistema software a desarrollar, de tal forma que todas las personas involucradas en el proyecto puedan tener conocimiento sin ambigüedades de lo que se va a desarrollar.

2.1.2 Ámbito del sistema

El producto software que se va a desarrollar es una aplicación web de gestión y venta de productos vía Internet, donde los usuarios podrán consultar información de los diferentes productos y comprarlos, así como también permitirá la gestión de los productos de la tienda.

2.1.3 Definiciones, acrónimos y abreviaturas

Autenticación o Autentificación es la confirmación de algo o alguien como auténtico. Con este término nos referimos al acto de verificar que la persona que se conecta a una aplicación es la que se espera, y debe tener los derechos, permisos y/o privilegios que le corresponden.

PHP es un acrónimo de *Hypertext Pre-processor*, es un lenguaje de programación interpretado, diseñado originalmente para la creación de páginas web dinámicas.

HTML, siglas de HyperText Markup Language (*Lenguaje de Marcado de Hipertexto*), es el lenguaje de marcado (junto con el texto, incorpora etiquetas o marcas que contienen información adicional acerca de la estructura del texto o su presentación) predominante para la elaboración de páginas web. Es usado para describir la estructura y el contenido en forma de texto, así como para complementar el texto con objetos tales como imágenes.

IEEE corresponde a las siglas de (Institute of Electrical and Electronics Engineers) en español Instituto de Ingenieros Eléctricos y Electrónicos, una asociación técnico-profesional mundial dedicada a la estandarización, entre otras cosas.

DB acrónimo de Data Base.

Interfaz se refiere a una conexión física y funcional entre dos aparatos o sistemas independientes, en el caso de Interfaz de usuario es el medio con que el usuario puede comunicarse con una máquina, en nuestro caso nuestra aplicación software.

ERS o SRS acrónimo del castellano Especificación de Requisitos Software.

Symfony es un completo framework para aplicaciones web de proyectos PHP.

ORM del inglés Object-Relational mapping, mapeo objeto-relacional en castellano, es una técnica de programación para convertir datos entre el sistema de tipos utilizado en un lenguaje de programación orientado a objetos y el utilizado en una base de datos relacional, utilizando un motor de persistencia.

Doctrine es un ORM para PHP que ofrece persistencia para objetos PHP.

Apache Es un servidor muy popular.

Twig engine de plantillas para PHP que incluye symfony.

MVC o modelo vista controlador es un modelo de abstracción en el desarrollo del software.

CSS del inglés *Cascading Style Sheets, son hojas de estilo en cascada,* es un lenguaje usado para definir la presentación de un documento estructurado escrito en HTML o XML.

2.1.4 Referencias

[1] Estándar IEEE830-1998 (1998)

[2] Méndez G. 2008. Especificación de Requisitos según el estándar de IEEE 830. Universidad Complutense de Madrid Fuente:

2.1.5 Visión General del Documento

Este documento está conformado de tres secciones que son la Introducción, la Descripción General y los Requisitos Específicos. En esta primera sección se procura proporcionar una visión general de lo que es el documento de especificación de requisitos. En la segunda sección se da una descripción general del sistema a construir, para conocer sus funciones principales, los datos requeridos, y sus restricciones, entre otras cosas que afecten a su desarrollo, aunque no se entra en los detalles de cada uno de estos factores y, por último, en la tercera sección se definen los pormenores de los requisito que nuevo sistema debe satisfacer.

2.2 Descripción General

2.2.1 Perspectiva del producto

Se pretende conseguir una aplicación que solo requiera un navegador web para utilizarla. Buscando un uso intuitivo y sencillo. La aplicación será optimizada para Mozilla Firefox y para Google Crome.

2.2.2 Funciones del producto

A continuación enumeraremos las funcionalidades principales del sistema, cada una de ellas será explicada con más detalle en la sección 2.3.3. Se han añadido códigos a las funcionalidades para facilitar su trazabilidad.

- Ver información relevante sobre el portal (Inicio F001, Contacto F002, Ayuda F003, Privacidad F004)
- Registrarse como nuevo usuario. (F005)
- Iniciar sesión en la aplicación con un usuario previamente registrado. (F006)
- Ver información personal. (F007)
- Borrar cuenta de cliente. (F008)
- Cerrar Sesión. (F009)
- Listado de contenido incluyendo búsquedas por palabras clave, precios, categoría… permitiendo la revisión de los productos de forma detallada. (F010a, F010b)
- Gestión de artículos para añadir artículo nuevo, modificar datos artículo existente y eliminar artículo. (F014, F015, F016, F017)
- Añadir elemento/s al carrito de la compra. (F011)
- Revisar carrito de la compra (F012)
- Comprar contenido del carrito de la compra. (F013)
- Listado para gestión de pedidos. (F014)

- Listado de clientes.(F015)

2.2.3 Características de los Usuarios

El sistema software a desarrollar tendrá un sistema de autenticación que permitirá identificar los diferentes roles y así ofrecer una funcionalidad concreta dependiendo del rol. Esta característica es esencial entre otras causas por razones de seguridad.

Usuario anónimo o invitado: es aquel que no ha iniciado sesión en el sistema y que tiene únicamente acceso al catálogo de productos, a sus características y puede gestionar su carrito de la compra.

Usuario registrado o cliente: Son los usuarios que ya se han registrado y están ya autenticados. Tienen acceso a prácticamente todo lo que el "invitado" pero además pueden comprar y revisar sus datos.

Usuario administrador o gestor: Es el usuario encargado de gestionar el portal y sus contenidos, de tal forma que puede añadir nuevo material así como modificar el existente.

2.2.4 Restricciones generales

Para acceder al portal no será necesario ningún hardware específico, será necesario solamente acceso a internet y un navegador.

El administrador o gestor podrá alterar parte del contenido de la base de datos usando la propia aplicación pero nunca modificar la estructura de la BD.

Se debe tener un sistema seguro que permita almacenar de forma segura la información de los clientes.

2.2.5 Supuestos y dependencias

La aplicación requerirá de un servidor apache que debe tener instalados symfony2.

2.2.6 Requisitos Futuros

La posible ampliación del portal así como el posible estudio para la mejora del rendimiento en caso de que el portal tuviese más visitas de las esperadas, todo esto debería ser estudiado de forma independiente al proyecto actual.

2.3 Requisitos específicos

En la figura 2.3 podemos ver el Diagrama de casos de uso del sistema que se va a desarrollar. Los diagramas de casos de uso son uno de los artefactos de UML.

Figura 2.3 Casos de uso del sistema.

Podemos apreciar tanto los tipos de usuarios como “casos de uso” que podrán realizar. Estos casos de uso son también las funciones que aparecen en las secciones: Funciones del producto (2.2.2), así como en Funciones (2.3.2).

Las claves más reseñables del modelo son:

La herencia entre usuarios que permite ver que usuarios más “importantes” pueden realizar las mismas funcionalidades que los “menos importantes”.

Es destacable también la redefinición de uno de los casos de uso, ya que es un escenario relativamente similar y utiliza gran parte de la funcionalidad pero adaptada a los casos de uso concretos del gestor.

Se puede apreciar en el modelo que los roles de usuario pueden variar utilizando los casos de uso: Registro, Inicio Sesión (Log in) y Cerrar Sesión (Log out).

2.3.1 Funciones

Área común a todos los usuarios. Será una serie de contenidos que podrán ver todos los usuarios sin ninguna variación.

F001 Ver Inicio, como inicio se mostrara toda una serie de información referente al portal. Esta información deberá poder ser consultada por todos los tipos de usuarios.

F002 Ver Contacto, tendrá toda la información referente al “comercio” así como las formas de contacto y localización física.

F003 Ver Ayuda, permitirá al usuario ver información relativa al portal que le permita aclaraciones sobre su uso y posibles problemas.

F004 Ver detalles de privacidad, permitirá conocer los detalles sobre la privacidad que permite el portal.

Usuario anónimo o invitado. Cualquier persona que acceda al portal sin haber sido autenticado correctamente pertenecerá a este rol.

F005 Registrarse como usuario, como primero de los pasos requeridos para comprar en la tienda se necesita toda una serie de información sobre el usuario, así como un nombre y contraseña que lo identifique. Se mostrara un formulario de información a rellenar por el usuario, que tendrá una serie de campos requeridos.

F006 Iniciar sesión en la aplicación, con un usuario previamente registrado (dejando de ser usuario anónimo). Para poder realizar esta funcionalidad es pre-requisito que anteriormente se realice el paso F001 para que el sistema tenga la información relativa al usuario, así como su nombre y contraseña que permita su autenticación.

F010a Listar los productos. Mostrará un listado de los productos disponibles. Desde aquí se podrá matizar la búsqueda según distintos criterios. Así como la ordenación.

F010b Revisar contenido de los productos que están disponibles en la web.

F011 Añadir elemento/s al carrito de la compra. El sistema mantendrá la información del carrito de la compra y permitirá al cliente añadir elementos de forma iterativa.

F012 Revisar carrito de la compra. El sistema permitirá ver el contenido del carrito de la compra mostrando los datos básicos de los productos seleccionados así como los precios. También permitirá eliminar algún producto no deseado de la lista.

Usuario registrado o cliente. Previamente ha de haberse identificado para ser un usuario registrado o cliente, para lo que en otra sesión anterior deberá haberse registrado.

El usuario registrado tendrá acceso a gran parte de funcionalidades iguales a las de usuario anónimo, como serian: Listado de productos (F010a, F010b), Añadir elementos y revisar el carrito de la compra (F011, F012).

F007 Ver la información personal del usuario.

F008 Borrar cuenta de cliente. Permitirá al usuario que lo desee, eliminar sus datos del portal en caso de que no quiera utilizar el servicio nunca más o esté preocupado por su privacidad.

F009 Cerrar Sesión. El sistema deberá cerrar la sesión del usuario devolviendo el sistema al estado de usuario anónimo y requiriendo de nuevo el inicio de sesión en caso de querer volver al estatus de usuario registrado o administrador.

F013 Comprar contenido del carrito de la compra. Permitirá al cliente realizar la compra del contenido que ha ido acumulando en el carrito de la compra.

Usuario Administrador o Gestor. El administrador deberá autenticarse para dejar de ser un usuario anónimo y tener acceso a las secciones del portal relativas a la gestión del mismo.

F009 Cerrar Sesión.

F014 Listado y búsqueda de artículos. El sistema deberá permitir al Administrador revisar la lista de productos del portal y desde aquí usar otras funcionalidades: Añadir, modificar o descatalogar productos.

F015 Añadir artículo nuevo. Permitirá introducir en el sistema productos junto con sus características.

F016 Modificar datos artículo existente. El sistema mostrará los datos de un producto concreto permitiendo al gestor modificar los detalles del producto.

F017 Eliminar artículo. El gestor podrá descatalogar productos del portal haciéndolos desaparecer a los ojos de los usuarios pero permitiendo ver los detalles desde administración así como desde el histórico de pedidos.

F018 Listado para la gestión de pedidos, ver información detallada y marcar detalles como si ha sido ya enviado o no.

F019 Listado de clientes para poder revisar sus datos así como pedidos etc…

He realizado la estructuración basándome en los distintos usuarios pues considero que al ser diferentes los requisitos de cada uno permite una clara diferenciación.

2.3.2 Requisitos de Rendimiento

En principio se espera una apertura rápida del portal y no se espera una gran cantidad de usuarios conectados simultáneamente por lo que no hay requisitos de rendimiento claramente definidos.

2.3.3 Requisitos Lógicos de la Base de Datos

Los requisitos lógicos de la BD vienen marcados por las tecnologías a utilizar. Se ha de buscar un diseño que permita obtener el máximo beneficio del ORM y permita mapear correctamente.

Los ORM se utilizan para cambiar la forma tradicional en la que las capas de persistencia se venían haciendo hasta su aparición. Existe una diferencia de concepción entre los datos en un entorno de programación orientada a objetos y los datos almacenados en una base de datos relacional SQL. Esa diferencia se puede intentar evitar usando bases de datos objeto-relacionales, pero también se puede recurrir a un ORM.

Un ORM no hace cambiar mucho la idea de la Base de Datos pero hace recomendable un planteamiento claro desde el principio enfocado a facilitar un mapeado lo mejor diseñado posible, ya que las prioridades han cambiado.

Aun así se intentará explotar las virtudes propias de una BD relacional siempre que no genere inconvenientes en el uso del ORM.

2.3.4 Atributos del Sistema

Se buscará un sistema centrado en ser fiable y seguro. Se utilizará un sistema de log in para garantizar la Autenticación del usuario y así poder delimitar su acceso como corresponda en cada caso.

3. Análisis

Para los modelos utilizados en la etapa de análisis se ha utilizado el lenguaje unificado de modelado (UML), que es un estándar que se ha convertido en uno de los mas usados para especificar y documentar sistemas de información [9]. En particular los requisitos funcionales se han descrito mediante un modelo de casos de uso, la estructura estática del sistema con un diagrama de clases, mientras que para las interacciones internas en el sistema se han empleado diagramas de secuencia.

3.1 UML (*Unified Modeling Language*)

El Lenguaje Unificado de Modelado es el lenguaje de modelado de sistemas software más conocido y utilizado, lo que permitirá que más gente pueda entender este proyecto con facilidad viendo este análisis. El ser un lenguaje gráfico facilita enormemente la comprensión y permite visualizar de forma global un proyecto, ya que de otra forma sería más difícil de abordar este tipo de problemas.

RUP (Rational Unified Process) que es un proceso de desarrollo de ingeniería del software [10] es una de las metodologías que utilizan UML como notación.

3.1.1 Diagrama de Casos de uso

Los casos de uso se utilizan para dar una descripción de las funcionalidades que el sistema ofrece a los distintos tipos de usuarios.

Las funcionalidades del sistema que aparecen en los casos de uso ya fueron explicadas previamente en el apartado 2.3, pero parece importante remarcar algunas decisiones que se tomaron al realizar los diagramas.

Como se aprecia en la figura 2.3 existen 3 tipos de actores entre los cuales dos de ellos son muy similares mientras que el Administrador o “Gestor” tiene unas funcionalidades bastante diferentes a los otros, “podrá hacer diferentes actividades”. Así se decidió darle privilegios al usuario anónimo para hacer lo mismo que un cliente excepto comprar o consultar sus datos para mayor comodidad de los nuevos visitantes del portal.

La herencia con redefinición de la actividad “Consultar Catalogo” con “Gestión de catalogo” es debido a que aunque parezcan actividades completamente diferentes, en realidad los tres

actores necesitan utilizar una estructura de listados y búsquedas muy similar aunque con diferentes privilegios. Así es necesario buscar entre los productos tanto para verlos detallados (cosa que pueden hacer todos) como también para añadirlos a un carrito o para eliminarlos. Utilizando la redefinición se podrán reutilizar partes de código, agilizar el desarrollo y darle más consistencia al sistema.

3.1.2 Diagrama de Clases

El diagrama de clases describe la estructura del sistema mostrando las clases, atributos y métodos, así como las relaciones entre estas clases.

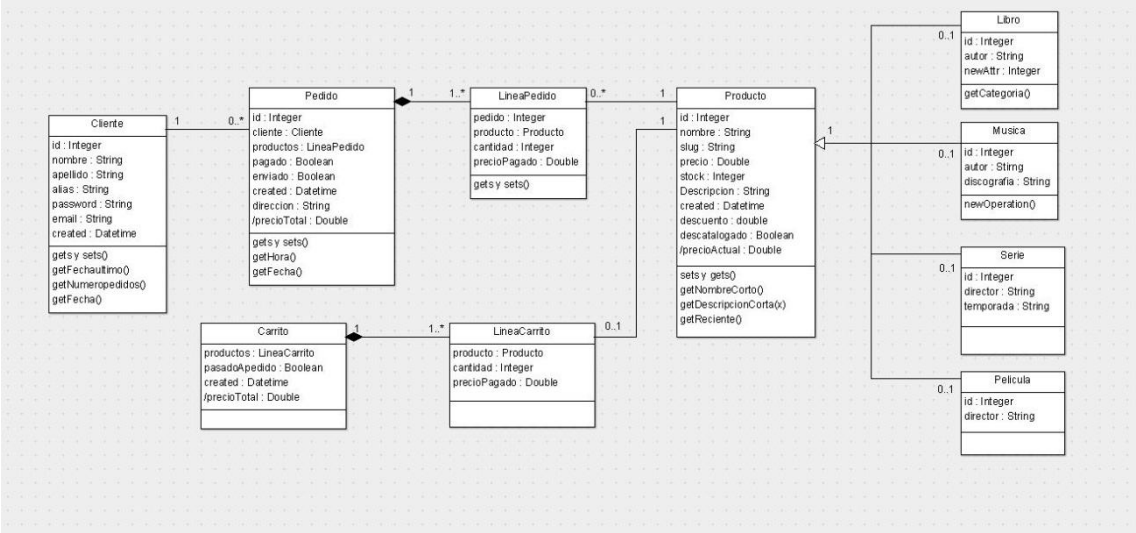


Figura 3.1.2 Diagrama de clases realizado con Agrouml.

A la hora de diseñar el sistema y especialmente el diagrama de clases es importante tener en cuenta que se pretende hacer uso de un ORM de tal forma que es importante prever como serán las relaciones de mapeado con la base de datos y saber que el servidor almacenara en memoria grandes cantidades de objetos.

Aun así el Diagrama de clases no deja de ser bastante estándar para un escenario del tipo “tienda en internet”.

Características como el hecho de que el carrito esté aparte tiene su razón de ser en que el carrito será almacenado en la Sesión del usuario, independientemente de si ha iniciado sesión o no, obviamente no será algo persistente pues estará en el Cliente y no en el servidor.

La razón por la que hay tantos indicadores id sin ser necesarios en un entorno orientado a objetos es porque al hacer el mapeado pueden resultar necesarios. Además se ha optado por incluir identificadores auto-incrementales en la base de datos que tanto la BBDD como el ORM gestionarán, pero mas tarde podrán ser utilizados. Las razones de esta decisión son:

-simplifica el sistema conceptualmente.

-en casos como “LineaCarrito” y “LineaPedido” da seguridad añadida a la no redundancia de tuplas ni datos ya que serán claves primarias dobles y además únicas, impidiendo errores, nulos y duplicados.

- en casos como la herencia de pedido que es un caso sensible en el mapeado, es necesario realizarlo con identificadores duplicados en el padre y los hijos, así como la obligatoriedad del campo discriminador “discr” que veremos en el apartado 4.3.2.

- en los repositorios de datos que aparecerán en el apartado 5 veremos que vienen incluidos unos casos básicos que con un identificador son muy interesantes. Además al incluir la propia clave en la url podemos simplificar la codificación y hacer claras las urls.

Se aprecia que el constructor y los métodos get y set normales no están en el diagrama. La razón es para facilitar la legibilidad del mismo. Aunque si aparecen aquellos que dan información tratada.

3.1.3 Escenarios y Diagramas de Secuencia

Los diagramas de secuencia de UML modelan el flujo de la lógica interna del sistema de forma visual, permitiendo documentar y validar esa lógica. Es el más popular de los modelados dinámicos que se centra en identificar el comportamiento del sistema. [8] Los escenarios son tablas de información que permiten plasmar información sobre interacciones, siendo ideales para compaginar con diagramas de secuencia.

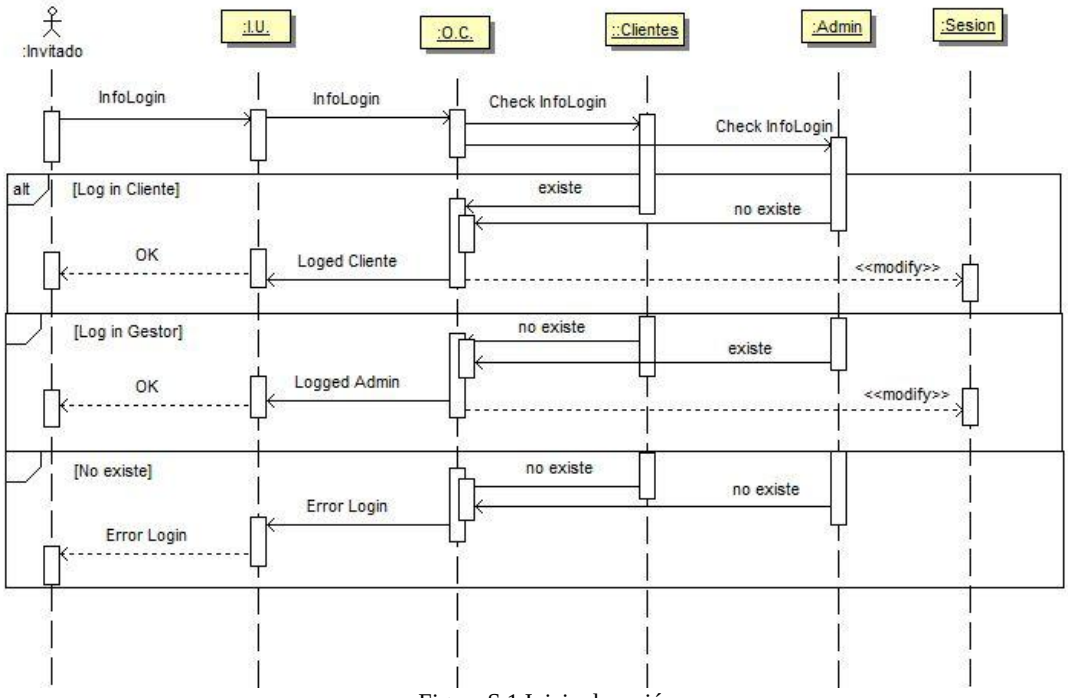


Figura S.1 Inicio de sesión

Título: Inicio de sesión (Login)

Objetivo: Autenticar a un usuario anónimo (invitado), permitiéndonos catalogarlo como usuario registrado (cliente) o usuario administrador (gestor).

Precondición: El usuario ha de ser en ese momento un usuario invitado.

Actores: Usuario anónimo (invitado).

Episodios:

1 En cualquier momento el usuario introduce los datos (nombre y contraseña).

2 El sistema intenta autenticar al usuario.

3.1 El sistema le asigna el rol correspondiente.

Excepciones:

3.2 Si el sistema no consiguió autenticar muestra un mensaje y no hace nada.

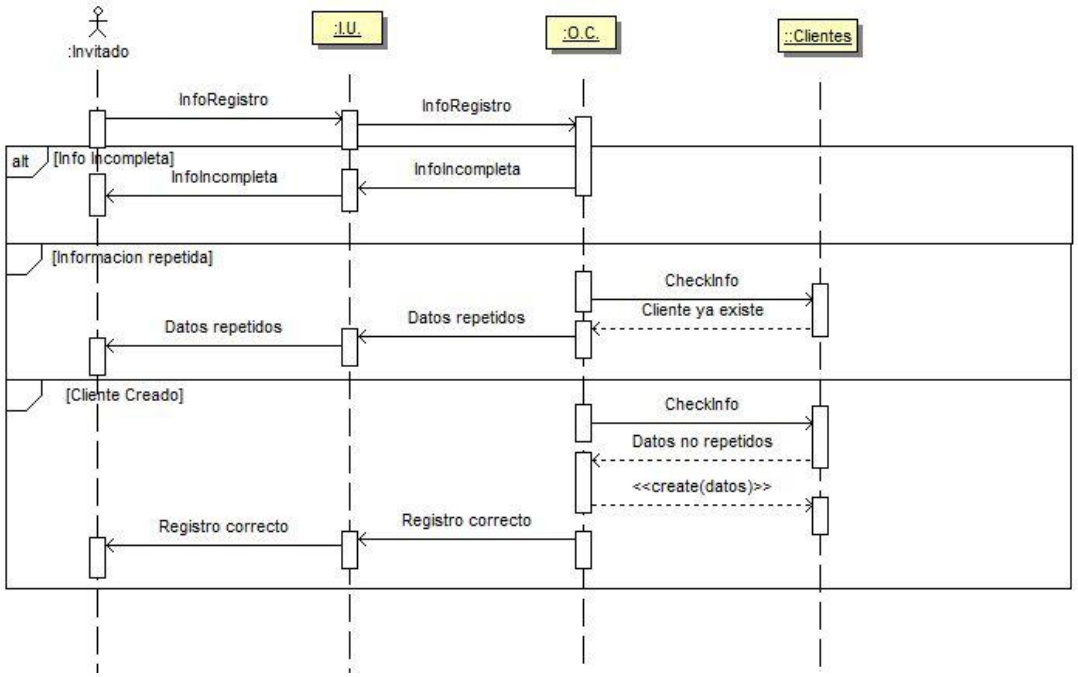


Figura S.2 Registro

Título: Registro

Objetivo: Almacenar la información pertinente sobre un potencial cliente, incluyendo los datos necesarios para posteriores autenticaciones. Para permitir mas adelante el login como usuario registrado.

Precondición: El usuario ha de ser en ese momento un usuario invitado.

Actores: Usuario anónimo (invitado).

Episodios:

1 En cualquier momento el usuario invitado puede irse al formulario de registro.

2 El formulario de registro le permite poner sus datos y enviarlos al sistema.

3.1 El sistema confirma que los datos introducidos son validos.

4 El sistema almacena todos los datos del cliente y le realiza automáticamente el login.

Excepciones:

3.2 Si los datos introducidos no son validos marca los posibles errores y vuelve a mostrar el formulario.

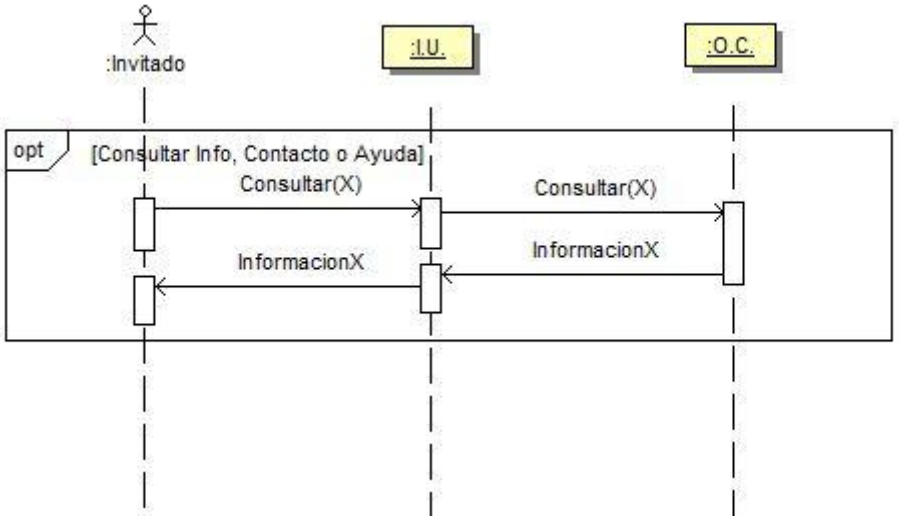


Figura S.3 Consultar información web.

Título: Consultar información web
Objetivo: Permitir a cualquier usuario de la web consultar información relativa al portal, a la compañía. Cosas como quienes somos, donde estamos, detalles sobre la política del portal o ayuda en como es su funcionamiento.
Precondición: No existen precondiciones
Actores: Usuario anónimo (invitado), Usuario registrado (cliente), Usuario gestor (Administrador).
Episodios:
1 Desde cualquier parte del portal se podría acceder a las pantallas de información y ayuda.
2 El usuario decide que información quiere consultar.

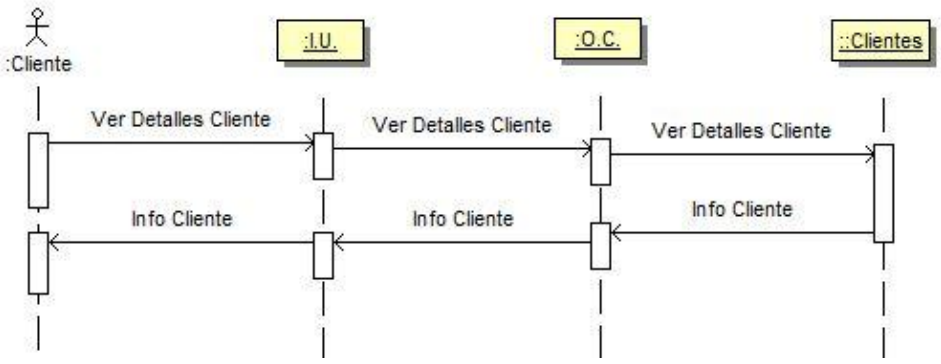


Figura S.4 Ver información personal.

Título: Ver información personal

Objetivo: Permitir al usuario ver su información personal así como un histórico de pedidos que él ha realizado permitiendo ver los detalles de sus pedidos y su estado.

Precondición: El usuario ha de estar logueado y ser un usuario registrado.

Actores: Usuario registrado (cliente).

Episodios:

1 En cualquier momento el usuario puede ir a su perfil.

2.1 El sistema muestra sus datos y sus pedidos.

3 El usuario puede navegar por el sitio para revisar detalles de sus pedidos.

Excepciones:

2.2 Si el usuario no esta logeado le pide q se logee para acceder.

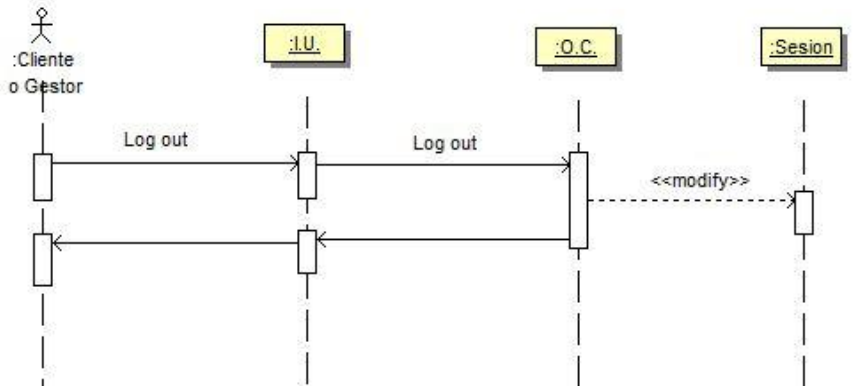


Figura S.5 Cerrar Sesión.

Título: Cerrar sesión (Log out)

Objetivo: Cambiar el status de un usuario registrado (cliente) o un usuario gestor (administrador) a usuario anónimo (invitado).

Precondición: El usuario ha de ser en ese momento un usuario registrado (cliente) o un usuario gestor (administrador).

Actores: Usuario registrado (cliente) o un usuario gestor (administrador).

Episodios:

1 En cualquier momento el usuario registrado (cliente) o un usuario gestor (administrador) puede hacer log out.

2 El sistema le cambia el estatus del usuario a invitado.

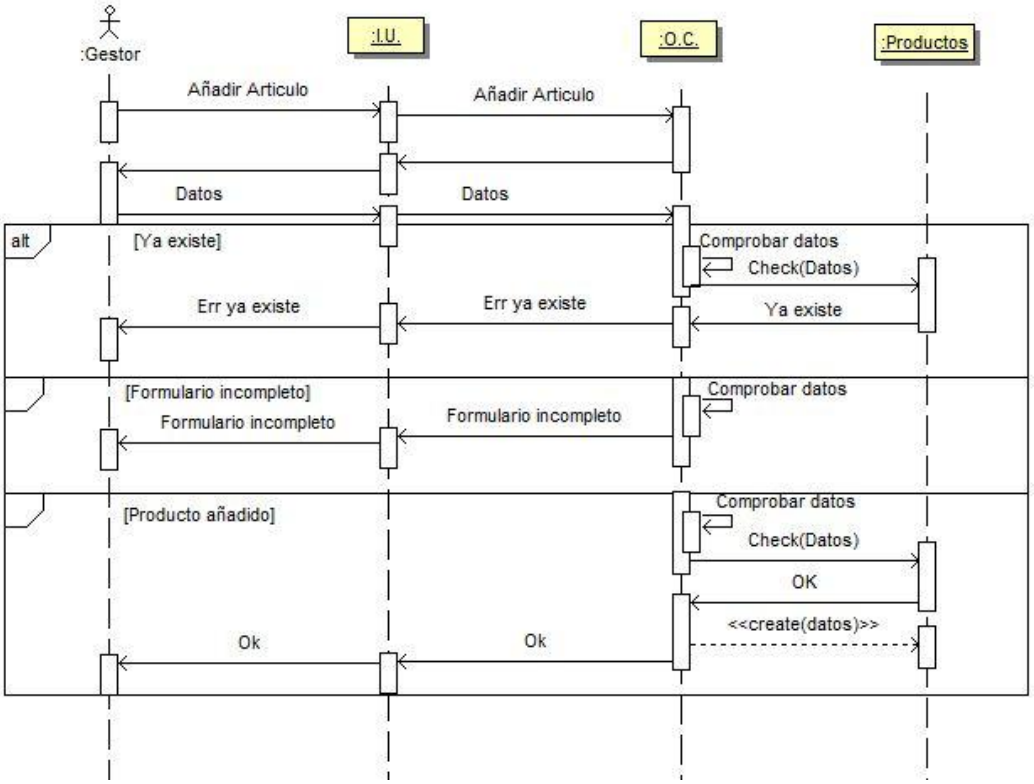


Figura S.6 Añadir Producto.

Título: Añadir producto

Objetivo: Permitir al administrador añadir nuevos productos para poner a la venta a través del portal.

Precondición: El usuario ha de ser un usuario gestor (administrador).

Actores: Usuario gestor (administrador).

Episodios:

1 El administrador entra al formulario de añadir producto.

2 Introduce los datos del producto y los envía al sistema.

3.1 El sistema confirma que los datos introducidos son validos.

4 El sistema almacena el nuevo producto.

Excepciones:

3.2 Si los datos introducidos no son validos o el producto ya existe, marca los posibles errores y vuelve a mostrar el formulario.

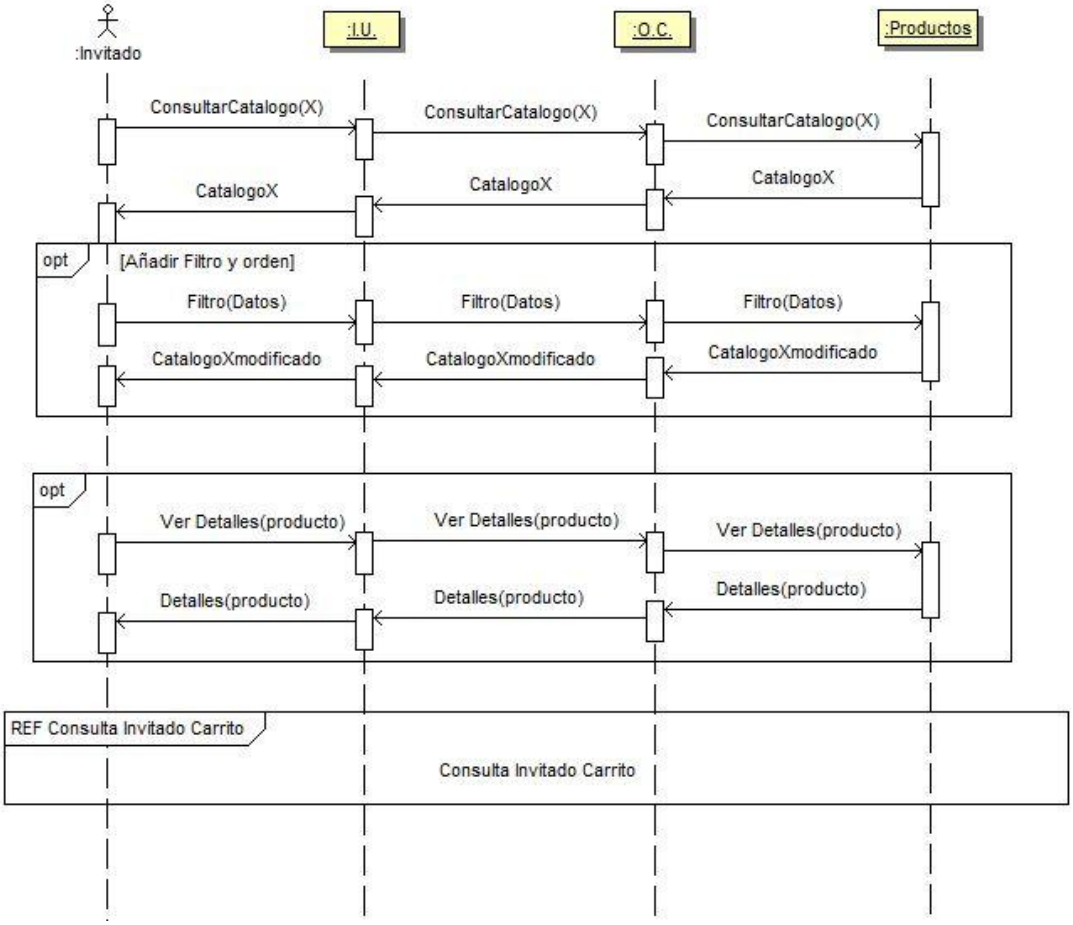


Figura S.7 Consultar catalogo invitado.

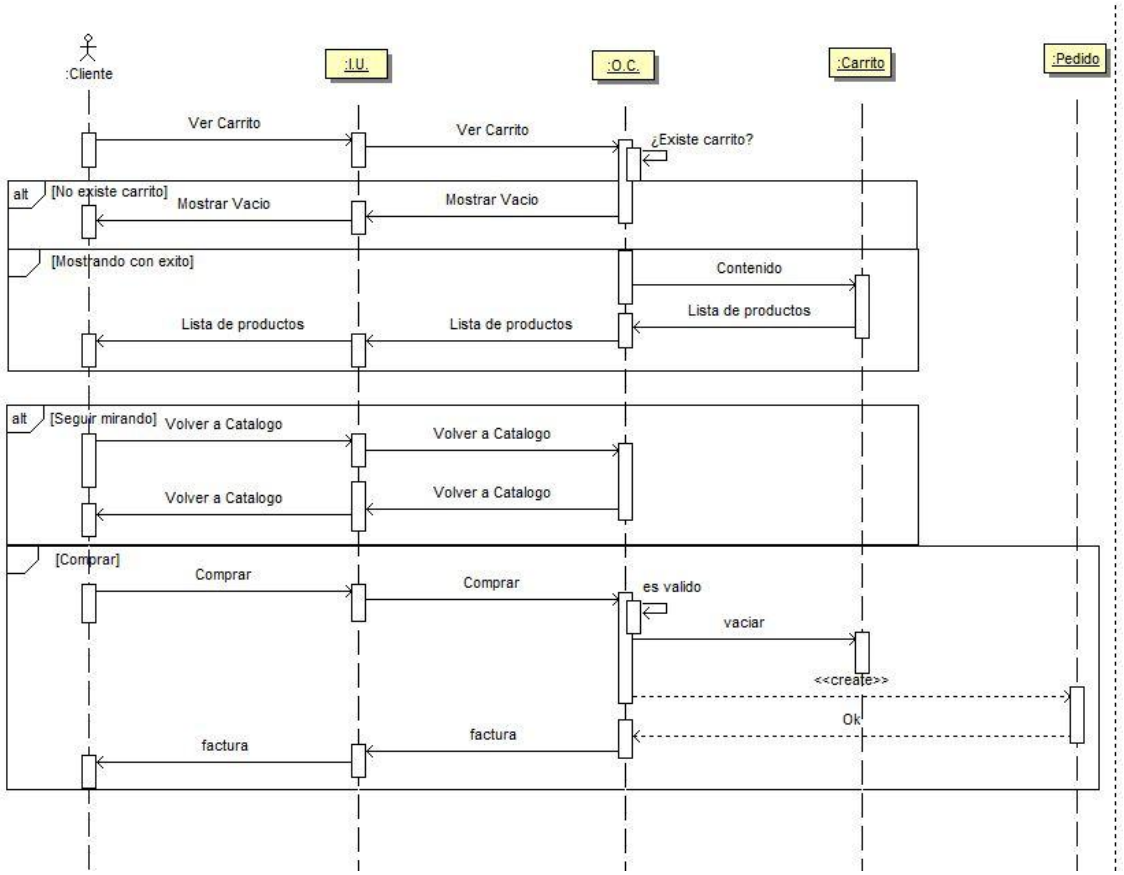


Figura S.8 Consultar carrito, gestión y compra.

Título: Consultar catalogo. Añadir al carrito y ver detalles del artículo.

Objetivo: Permitir al usuario ver listados de artículos así como ver los detalles de artículos y añadir esos productos a su carrito de la compra, para su posterior adquisición.

Precondición: El usuario ha de ser un usuario anónimo (invitado) o usuario registrado (cliente).

Actores: Usuario anónimo (invitado) o usuario registrado (cliente).

Episodios:

1 El usuario entra a la sección de productos del portal.

2 Busca manualmente o con ayuda del buscador el/los productos que desea.

3.1 El usuario añade un producto a su carrito.

3.2 El usuario va a ver los detalles de un producto concreto.

- Puede desde dentro de los detalles del producto añadirlo a su carrito o volver al catalogo.

4 El escenario termina cuando el invitado o cliente decide salir de esta sección ya sea viendo el catalogo, cerrando el navegador, etc…

Excepciones:

3.1.1 Si no hay ese producto en stock muestra un error y devuelve a donde estaba el usuario.

Título: Gestión carrito y compra.

Objetivo: Permitir al usuario conocer el estado de su carrito de la compra. También permitir ver los precios de los productos elegidos, las cantidades y poder eliminar elementos así como comprar el carrito.

Precondición: El usuario ha de ser un usuario anónimo (invitado) o usuario registrado (cliente).

Actores: Usuario anónimo (invitado) o usuario registrado (cliente).

Episodios:

1 El usuario puede ver el estado resumido de su carrito en todo momento. Pero para ver el carrito de forma detallada así como comprar el carrito deberá ir a la sección de carrito.

2 En la sección carrito el sistema le mostrará todos los detalles del carrito.

3.1 El usuario modifica las cantidades.

3.2 El usuario compra el carrito de la compra en su estado actual.

3.3 El usuario decide vaciar el carrito de la compra.

3.4 El usuario vuelve al catalogo o sigue navegando por el portal sin perder el contenido de su carrito.

4 En caso de compra con éxito el sistema mostrara una factura detallada de su pedido.

Excepciones:

3.2.1 Si el sistema encuentra errores, mostrara esos errores y volverá al carrito.

3.2.2 Si el sistema detecta que el usuario no es un usuario registrado o cliente le mandara automáticamente a log in para que pueda autenticarse y comprar.

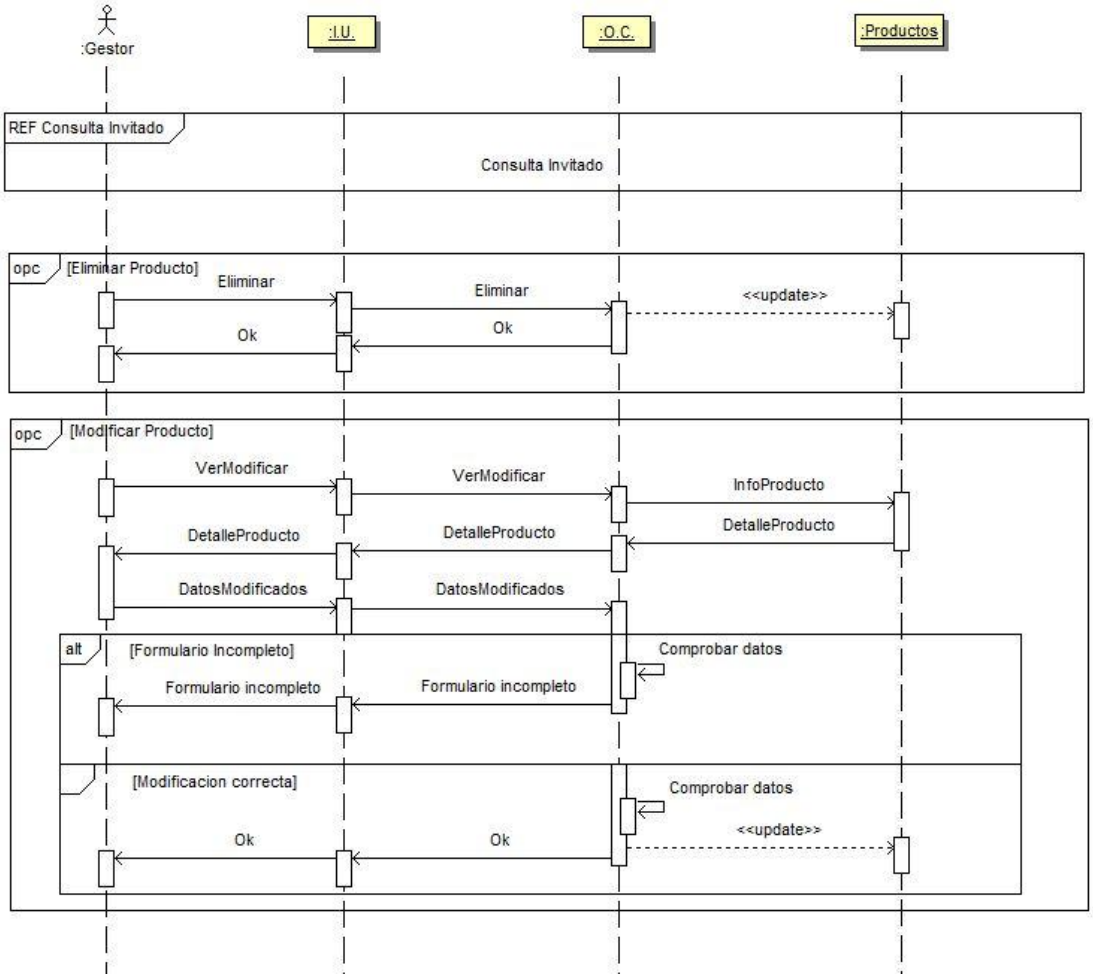


Figura S.10 Gestión de artículos.

Título: Gestión de artículos. Modificación y eliminación.

Objetivo: Permitir al administrador ver listados de artículos así como detalles de artículos que puedan ser modificados y eliminar del catalogo los artículos que por distintas causas se desee dejar de comercializar.

Precondición: El usuario ha de ser un usuario gestor (administrador). El/los productos que se desee modificar o eliminar del catalogo deben existir previamente en el sistema.

Actores: Usuario gestor (administrador).

Episodios:

1 El administrador entra a gestión de artículos.

2 Busca manualmente o con ayuda del buscador el/los productos que desea.

3.1 El usuario clica en modificar para ir al formulario de un producto concreto.

- Modifica los datos que desea en el formulario y envía los datos.

3.2 El usuario saca o devuelve al catalogo el producto que desea.

4 El sistema comprueba que las actualizaciones son correctas y las realiza.

Excepciones:

3.1.1 Si los datos introducidos no son valido, marca los posibles errores y vuelve a mostrar el formulario.

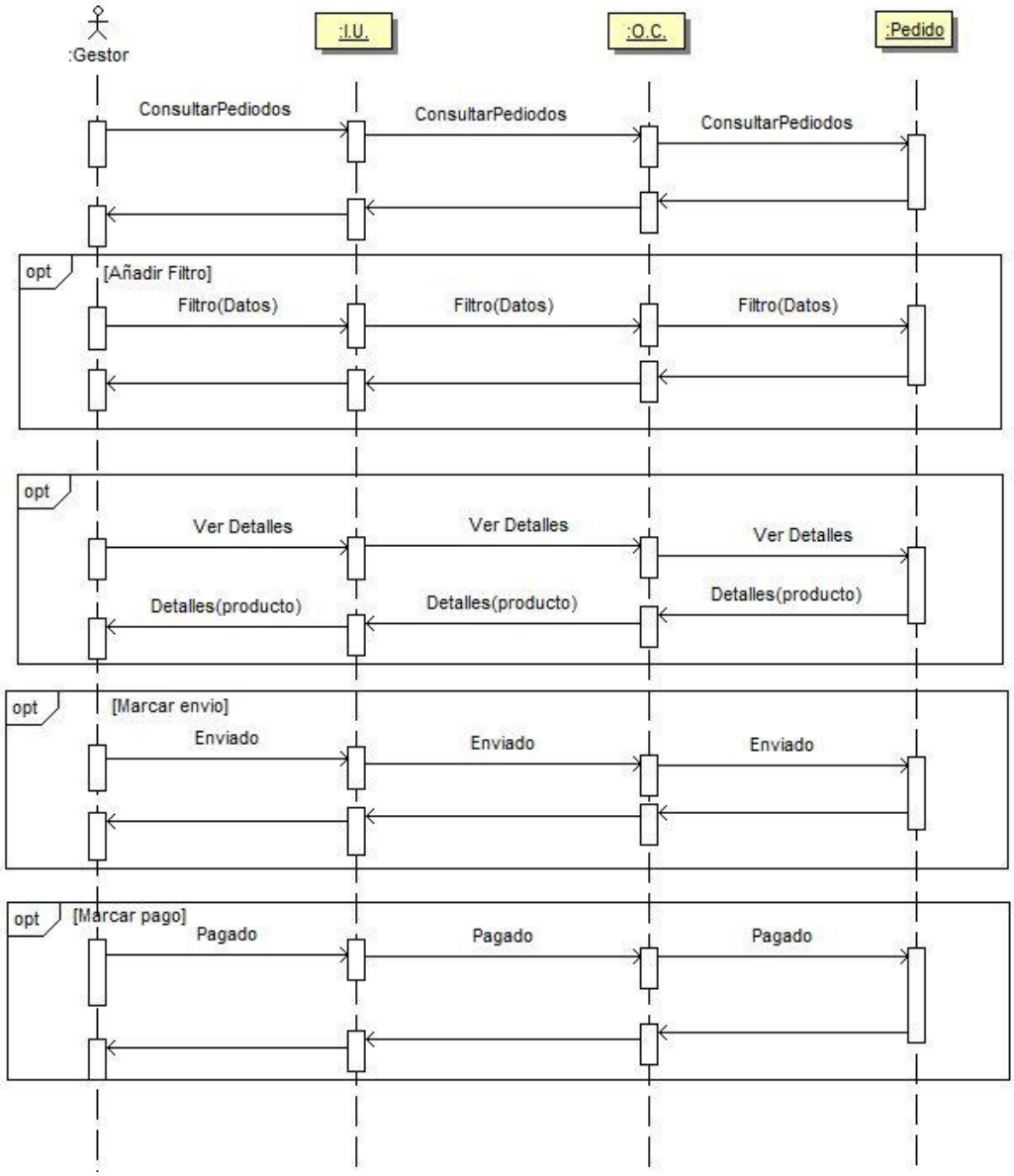


Figura S.11 Gestionar pedidos.

Título: Listar/Gestionar pedidos.

Objetivo: Permitir al usuario administrador ver listados de pedidos así como ver los detalles de los pedidos. También permitirá mostrar búsquedas informadas.

Precondición: El usuario ha de ser un usuario gestor (administrador).

Actores: Usuario gestor (administrador).

Episodios:

1 El usuario entra a la sección de pedidos del portal.

2 Busca manualmente el pedido que desea o con ayuda del buscador.

3 Mira los detalles de los pedidos que desee.

4 El usuario puede alterar los valores de enviado o pagado, ya que pueden haber ocurrido incidencias como una devolución del paquete por parte de la compañía de mensajería como un problema con el pago a posteriori.

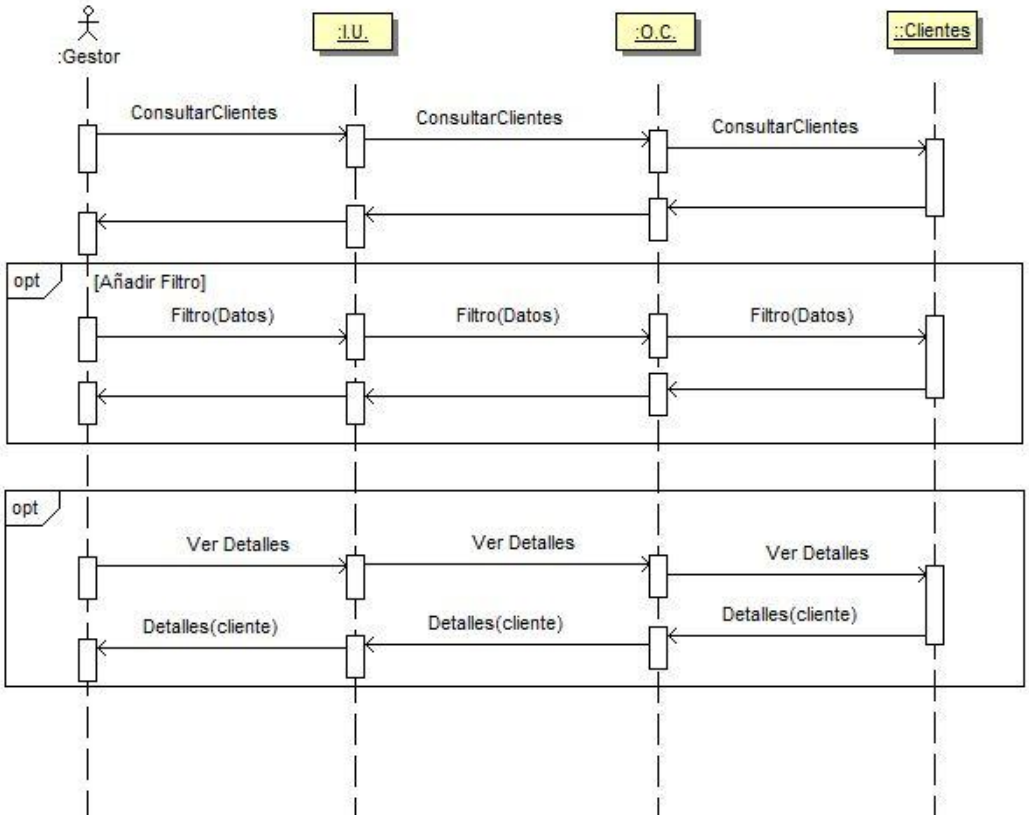


Figura S.12 Listar Clientes

Título: Listar clientes.
Objetivo: Permitir al usuario administrador ver listados de clientes así como ver los perfiles de estos clientes en detalle.
Precondición: El usuario ha de ser un usuario gestor (administrador).
Actores: Usuario gestor (administrador).
Episodios:
1 El usuario entra a la sección de clientes del portal.
2 Busca manualmente el cliente que desea.
3 Mira los detalles de los clientes que desee. (No podrá alterar los datos del cliente)

3.2 Otros Diagramas

Aunque no sean tan comunes se decidió incluir un par de esquemas que pueden aclarar los contenidos y como será la interfaz de usuario. Este tipo de diagramas no forman parte de UML:

3.2.1 Esquema de contenidos

Un esquema de contenidos es una simple representación de las agrupaciones que tiene la información. En la figura 3.2.1 se puede apreciar que la información del portal orbita en torno a 4 entes: La tienda, los productos, los pedidos y los clientes.

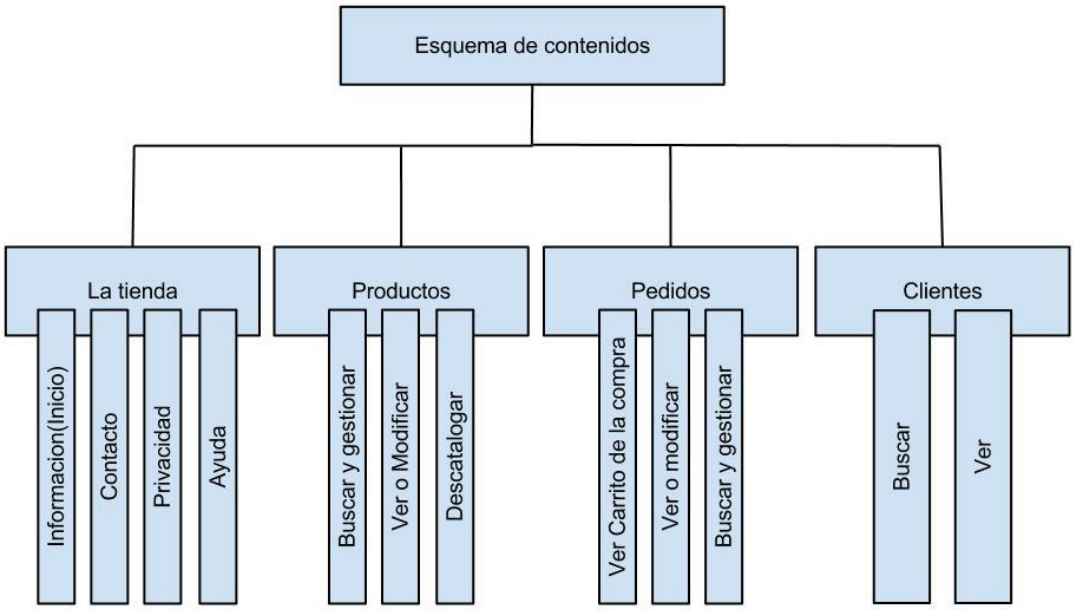


Figura 3.2.1 Esquema de contenidos. Muestra los contenidos agrupados por a que hacen referencia.

3.2.2 Vista de interfaz de usuario

Para facilitar la usabilidad se ha decidido mantener una estructura que sea común en las aplicaciones web de este tipo, ya que así el usuario encontrara familiar el entorno y lo utilizara de forma intuitiva.

La figura 3.3.2 muestra un esbozo lógico de la interfaz de usuario Web del portal, que está dividida en: Cabecera donde ira el logo y el título, menú, sub-menú, barra lateral, contenido y pie de página.

Como se observa en la figura 3.2.2, la barra lateral se colocara a la derecha pues no es una barra para menús sino más bien informativa, ese tipo de secciones suelen estar a la derecha.

El menú será el eje central de navegación y se ha decidido colocarlo en la parte superior para dejar más amplitud al contenido ya que tendrá varias secciones con listas.

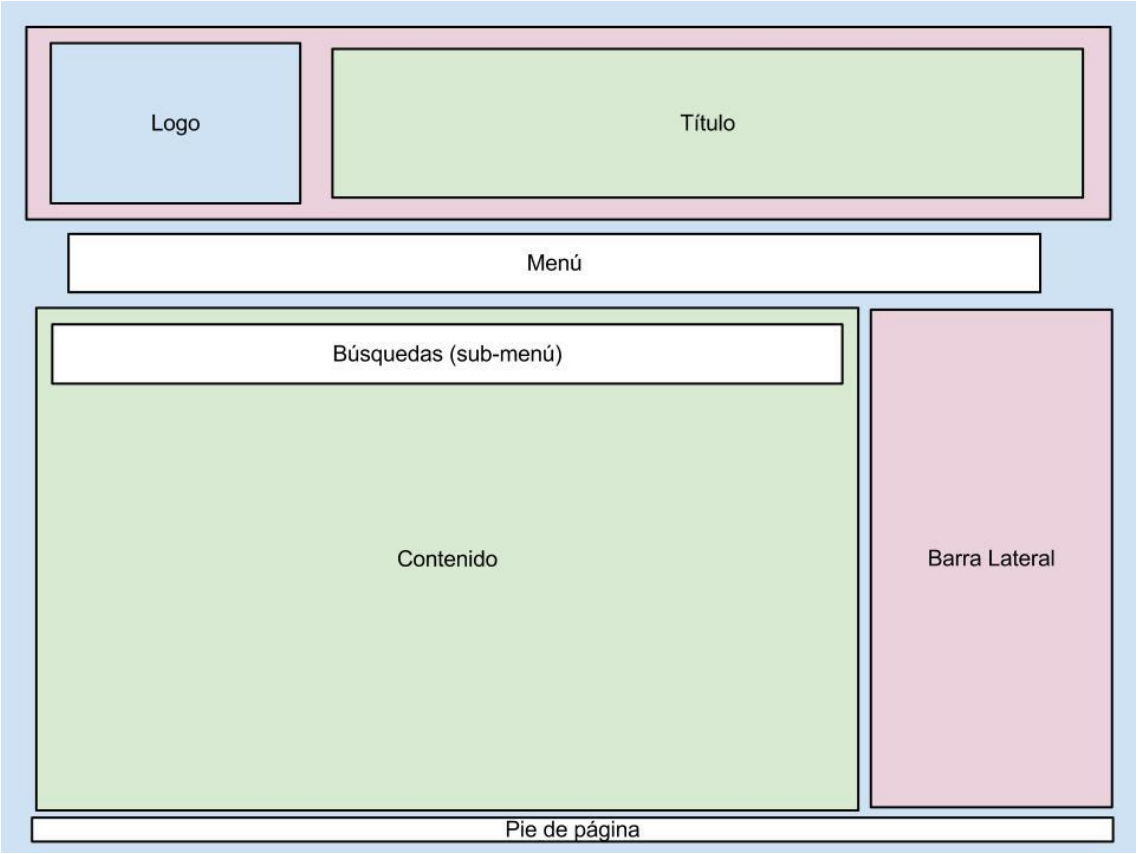


Figura 3.2.2 Se pueden apreciar las diferentes secciones que se van a emplear para la aplicación.

4. Diseño

Este proyecto al ser un proyecto web acaba como todos siguiendo la arquitectura Cliente-Servidor de varias capas por razones obvias (explicación sobre arquitectura C-S y las capas en los apartado 4.1, 4.2 y 4.3). Además se seguirá un patrón modelo vista-controlador que viene forzado en parte a las tecnologías a emplear pero que encaja perfectamente con este tipo de proyectos. Seguiremos también un modelo de capas aunque no exactamente igual al estilo clásico gracias al empleo de ORMs.

4.1 Arquitectura Cliente-Servidor

En esta arquitectura el **cliente** que normalmente es un usuario con un navegador conectado a internet, realiza tareas menores y su principal cometido es mostrar la interfaz de usuario y comunicarse con el servidor, muchas veces realizando peticiones.

El **servidor** es el que realiza el "trabajo". Almacena los cambios, realiza validaciones y manda la información necesaria del cliente. Quedándose normalmente a la espera de "conocer" las peticiones del cliente.

4.2 Modelo Vista-Controlador

El **MVC** es un modelo de abstracción en el desarrollo del software que separa las capas de presentación o interfaz de usuario y la capa de negocio. Fue creado por Trygver Reenskaug con el propósito principal de hacer de Puente entre el modelo mental que tienen los usuarios y el modelo digital que existe en el ordenador.

Simplificando:

- La vista es la información con la que interactúa el usuario.
- El controlador es el sistema haciendo todo el trabajo por detrás, comunicándose estos dos elementos por mensajes. El controlador responde a eventos.
- El Modelo es la información con la que se opera.

En las figuras 4.2.1 y 4.2.2 Se pueden ver y comparar las representaciones del MVC, tanto la originaria como la que representa el modelo en symfony.

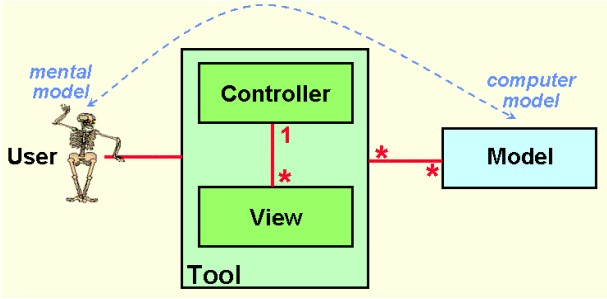


Figura 4.2.1 Concepción de como debe ser un sistema para simplificar el "Modelo" o los datos de tal forma que sean comprensibles por el usuario según su creador T. Reenskaug [1]

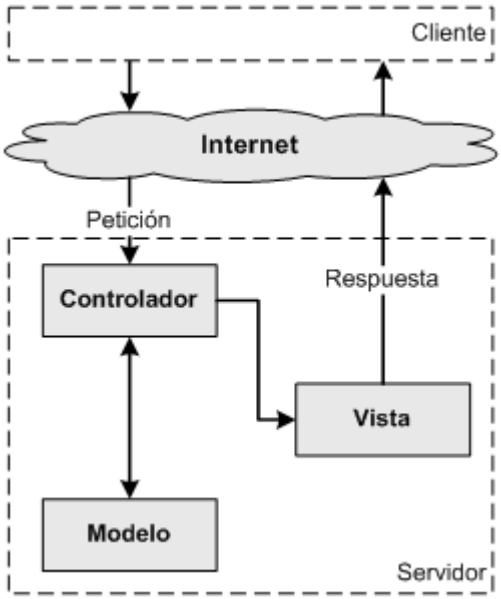


Figura 4.2.2 Representación del MVC según el manual de symfony.

El MVC encaja en la arquitectura Cliente-Servidor y también sigue las pautas de la arquitectura de capas.

4.3 Arquitectura de capas

La arquitectura de capas es muy sencilla de entender y de explicar. Se agrupan los módulos o contenidos de la aplicación en 3 grupos dependiendo de su futuro uso, esos tres grupos serán las capas.

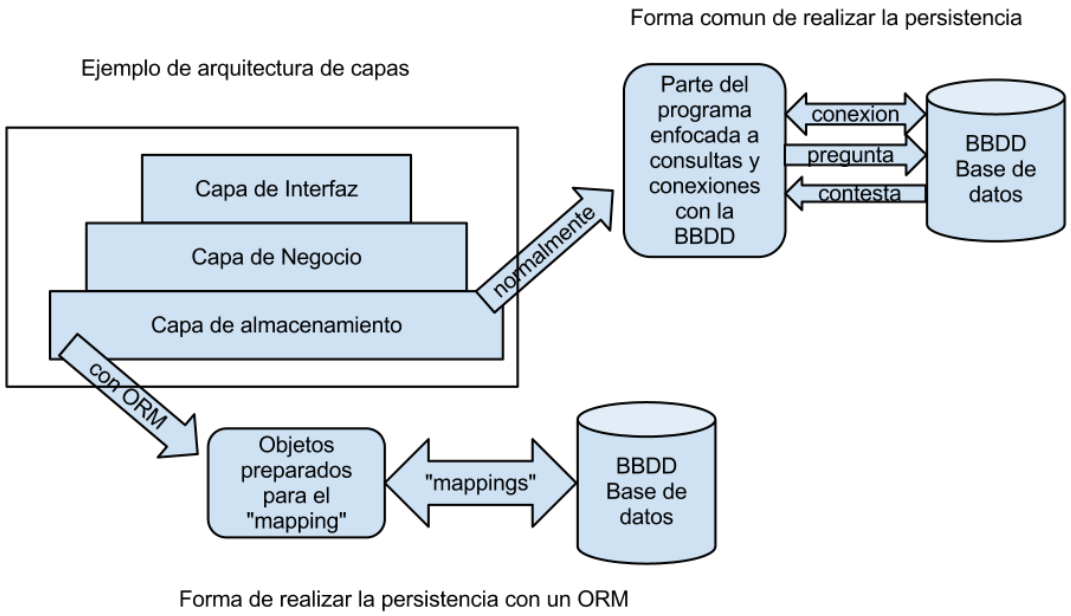


Figura 4.3 Ejemplo de arquitectura de capas con conexión a BD convencional y usando ORMs.

Se ha decidido añadir unas aclaraciones sobre como normalmente se realiza las capas de persistencia y como se realizan en el caso de utilizar un ORM como lo entiende Scott Ambler. [2]

4.3.1 Capa de presentación o Interfaz

La capa de interfaz es la encargada de “comunicarse” con el usuario. Recibe información desde el sistema y desde el usuario. En esta aplicación se usaran las herramientas que facilita symfony. El framework Symfony utiliza twig para representar la “Vista” que al fin y al cabo encaja perfectamente en lo que es la capa de presentación.

Gracias a las facilidades twig de herencia se separaran las plantillas de forma jerárquica para evitar repetir el código y darle un aspecto consistente al seguir siempre el mismo diseño.

4.3.2 Capa de Negocio o aplicación

La capa de negocio o aplicación actúa de mediados entre la capa de presentación y la capa de persistencia, contiene la implementación de la funcionalidad asociada a los casos de uso. Aunque no almacena datos todos los datos antes de ir a la capa de almacenamiento pasan por aquí para ser validados, tratados si es necesario, etc… Los datos cuando vuelven de la capa de persistencia también pasan por aquí antes de llegar a las vistas. En un MVC esta capa corresponde al Controlador. En symfony estos serían los "controllers" un tipo de clases de las que hablaremos en el apartado 5, aunque incluiría mas cosas.

4.3.2 Capa de Persistencia o almacenamiento

La capa de persistencia es la capa encargada de hacer que los datos se mantengan almacenados en la base de datos pero también es la encargada de recuperar estos datos. Como se ve en la figura 4.3, en un escenario de ORM el programador no realiza una conexión y una consulta SQL contra una base de datos. En este caso el ORM es el que realiza la conexión una vez, almacena los datos de la base de datos necesarios en memoria y los mantiene allí tomando los datos la forma de objetos. Gracias a ese mapeado que previamente se tenia hecho el ORM es capaz de transformar datos de la BBDD en datos que se pueden usar por el programa en forma de objetos, siendo estos objetos los usados generalmente en las tareas propias de la capa de negocio.

Esta forma de trabajar facilita mucho las tareas y sobre todo el trabajo del programador que podrá acceder a los datos sin la complejidad que supone tener que "sacar" y "transformar" los datos que requiere desde la BBDD. Tiene también la ventaja de que el sistema tendrá en memoria los datos sin tener que saturar de consultas a la BBDD.

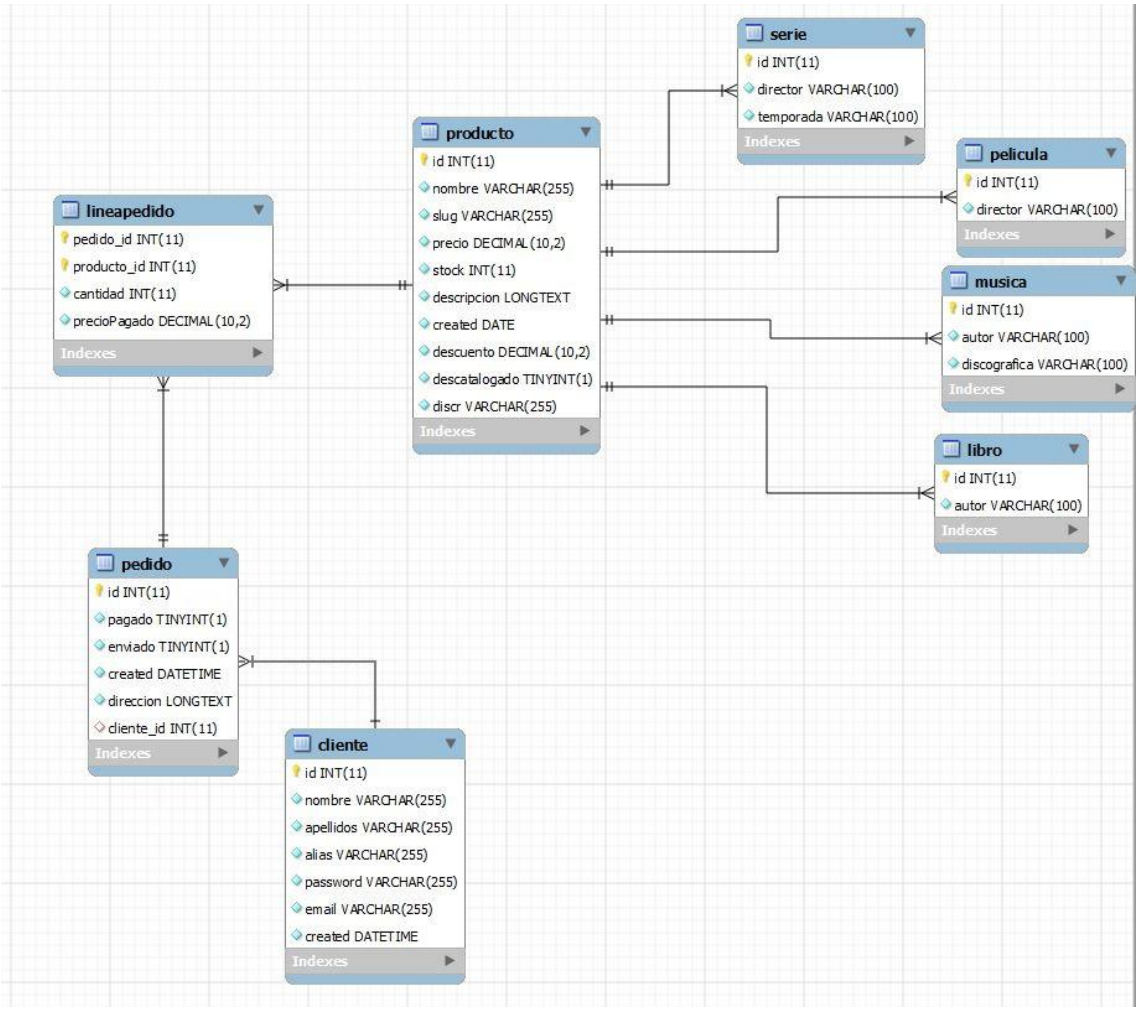


Figura 4.3.2 Esta Imagen ha sido creada con una exportación SQL desde phpmyadmin y una importación SQL a MySQL Workbench.

Se decidió no crear un diagrama en forma Entidad-Relación porque sería redundante. El diagrama de la figura 4.3.2 es en la práctica un modelo relacional con claves y claves ajenas, es más detallado y menos abstracto.

4.3.2.1 Estructura de la base de datos

Para la creación de la BBDD una aproximación muy recomendable puede ser generar las clases que correspondan en código, con su consecuente mapeado y utilizar las herramientas del ORM en cuestión (doctrine en nuestro caso) para el generado de la BBDD. Esta aproximación facilita mucho la tarea, y aunque las herramientas del ORM no son infalibles, el sistema permite cambios posteriores como la alteración de la BBDD utilizando sentencias SQL para generar las restricciones que falten o el propio SGDB. Para mas detalles ver apartado 5.3.

Como se ha mencionado en el apartado 3.1.2 el mapeado es la clave al crear una BBDD para estos sistemas ORM, se hacen imprescindibles ciertas estructuras, valores para poder crear un mapeado mas sencillo e incuso algunas veces solo se pueden crear ciertas relaciones entre clases teniendo un mapeado concreto. En el desarrollo enfocado a usar un ORM, el "mapeado" manda. Esto se puede apreciar en:

- Los identificadores de producto y sus clases herederas son necesarios y necesitan de un campo discriminador "discr" para poder hacer el mapeado de una herencia como la que nosotros necesitamos.
- Los tipos que se emplean son los que corresponden a tipos "equivalentes" en el mapeo a PHP.

Nótese que no existen las clases "Carrito" y "LineaCarrito", esto es debido a que no son persistentes. Esto se debe a que se almacenan en el cliente ya que sino sería mucha información innecesaria para el sistema que tendría que ser almacenada en la BBDD.

Como se comentaba en el análisis se decidió incluir identificadores auto-incrementales, estos serán gestionados por la BBDD y el ORM, pero más tarde los podré utilizar.

-simplifica el sistema conceptualmente. Son campos necesarios porque necesitamos no nulos y únicos para las claves.

-en "LineaPedido" da seguridad añadida a la no redundancia de tuplas ni datos ya que serán claves primarias dobles y además únicas, impidiendo errores, nulos y duplicados.

- en casos como la herencia de pedido que es un caso sensible en el mapeado (Class Table Inheritance), es necesario realizarlo con identificadores duplicados en el padre y los hijos, así como la necesidad del campo dicr que nos permitirá saber que "tabla" es la que hereda, haciendo que cada producto sea de un tipo.

- en los repositorios de datos que veremos en el apartado 5.3.3 vienen incluidos unos casos básicos que con un identificador son muy interesantes.

5. Implementación

Este capítulo se centra en las diferentes tecnologías, herramientas y lenguajes que se han utilizado y en como estas ayudan, facilitan o complican la tarea. Así como explicar como ha sido la codificación de la aplicación.

5.1 Capa de Presentación

Relativo a la Capa de Interfaz se han utilizado: Photoshop, CSSs, CSS960, HTML y plantillas Twig.

5.1.1 CSS

Cascading Style Sheets que se suele traducir por hojas de estilo en cascada es un lenguaje usado para definir los estilos y la presentación de los documentos HTML entre otros.

La razón por la que este tipo de documentos se han hecho populares es porque dejan separados los estilos de tal forma que queda modulizádo y es mucho más cómodo de gestionar y entender, además permite que toda una página web tenga los mismos estilos y también que cambiarlos sea mucho mas rápido ya que estarían “centralizados”.

Se han utilizado 3 documentos css diferentes:

- Reset.css es un documento css que fue desarrollado por Eric Meyer y que sirve para eliminar algunos valores de estilo prestablecido que los navegadores asignan a algunos elementos, evitando así ciertas inconsistencias de los mismos en la visualización del documento. Al utilizar el reset todos los valores se inicializan permitiendo que se muestre siempre lo mismo independientemente del navegador. [3]
- General.css es un documento CSS creado por mí donde pondré los estilos que se desea utilizar.
- 960.css es un documento css que permite utilizar “grids” y que explico a continuación.

5.1.2 960 Grid System

Es un documento CSS que utilizando la etiqueta div permite maquetar paginas web de forma estática. El creador ha hecho diferentes versiones. La 960 se llama así porque ocupa un máximo de 960 pixeles. Permite usar 16 columnas de 40 px o usar 12 de 60.

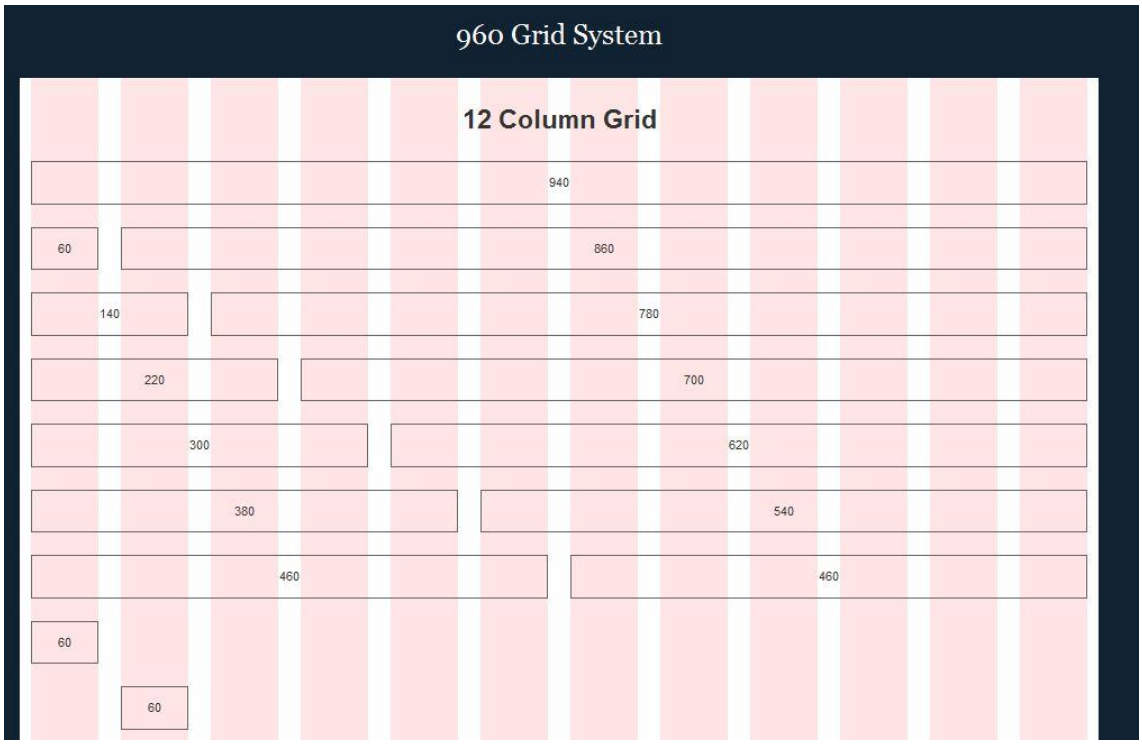


Imagen 5.1.2 Plantilla generalmente usada para trabajar con grids mostrando las medidas.

Como se puede observar en la imagen 5.1.2 se trata de montar cuadrados con contenidos, pero a diferencia de una tabla u otros sistemas estos son estáticos y no varían. Aunque no se aprecie en la imagen solo marcan el ancho de los grids, y se pueden mezclar cualquier tipo de grids en la misma línea. También permite hacer sub-grids.

El diseño de grids que propongo se explicará en la sección 5.1.3.

Existen otras versiones con mayor o menor ancho pero la más popular es la 960. Es una cifra que permite un buen equilibrio ya que aun hoy en día existe mucha gente que utiliza resoluciones de 800x600 o 1200x700, con mayores resoluciones puede parecer un error usar 960 pixeles de ancho pero el objetivo es que el tamaño sea “bueno” para la mayoría de potenciales clientes. [5]

Nota: Desde el cliente se sigue pudiendo alterar tamaños de fuente o ampliar el numero de pixeles como por ejemplo CTR+Rueda ratón que en Firefox amplia la pagina pudiendo ajustarla a lo que quiera el cliente.

5.1.3 Twig

Twig es un sistema de plantillas para la programación PHP que viene incluido en Symfony2, es open source.

En realidad Symfony2 también soporta plantillas PHP pero yo me decante por usar las plantillas twig porque son altamente recomendables al ser más sencillas, seguras y flexibles que las plantillas PHP:

- Symfony2 Traduce las plantillas twig a código PHP optimizado reduciendo en mucho la cantidad de código que generaría usando plantillas PHP.
- Twig tiene un sistema de evaluación de código que permite evitar que los usuarios puedan modificar el diseño de las plantillas.
- Twig permite definir etiquetas, funciones y servicios. [4]

Una de las características mas interesantes de Twig es que permite herencia múltiple y eso me ha permitido desarrollar un sistema de templates de tal forma que cada pantallazo de la aplicación puede tener hasta 6 plantillas twig detrás.

Por ejemplo la pantalla de Productos se genera con:

- Base.html.twig: contiene cosas generales como por ejemplo los css.(No hace referencia a imagen porque no tiene contenido visible, aunque si estructura, css…)
- Layout.html.twig: Extiende de base y tiene la estructura general de la aplicación así como la barra lateral y los títulos, también incluye el menú desde menú.html.twig. (Figura 5.1.3.1)
- listatodos.html.twig: Que extiende de layout.html.twig la estructura y la parte lateral. Tiene las cosas características de esa ventana como la búsqueda e incluye la tabla de productos que esta en lineaproductos.html.twig. (Figura 5.1.3.2)
- menuadmin.html.twig o menú.html.twig: que esta incluido dependiendo de que usuario haya pedido ver la plantilla.(Se aprecia que esta incluida en layout.html.twig en la figura 5.1.3.1)
- Lineaprodustos.html.twig: esta incluida en “listatodos” y es lista de grids con los productos y las opciones. (Figura 5.1.3.3)

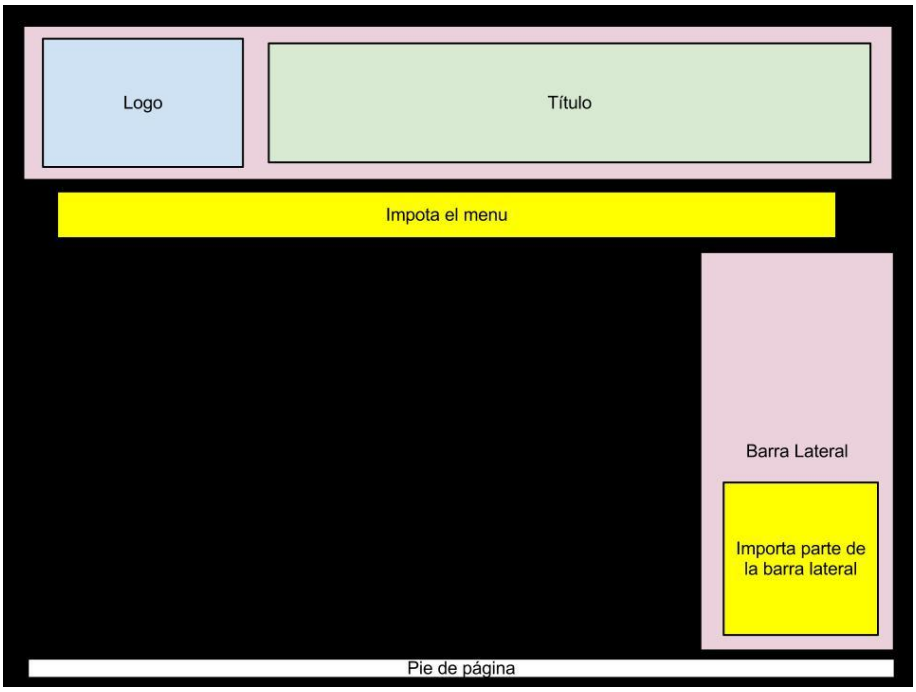


Figura 5.1.3.1 Representa el layout.html.twig

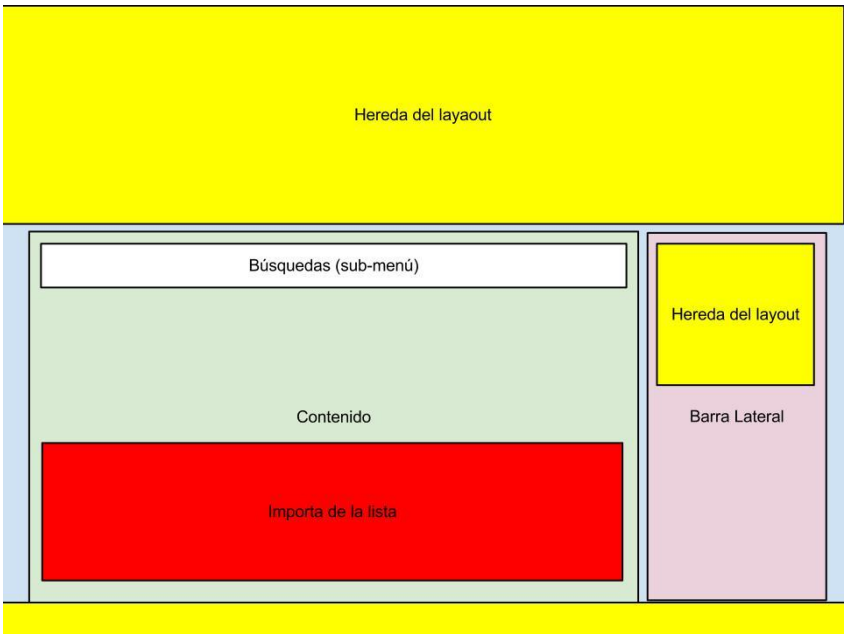


Figura 5.1.3.2 Representa la selección concreta, como por ejemplo productos.

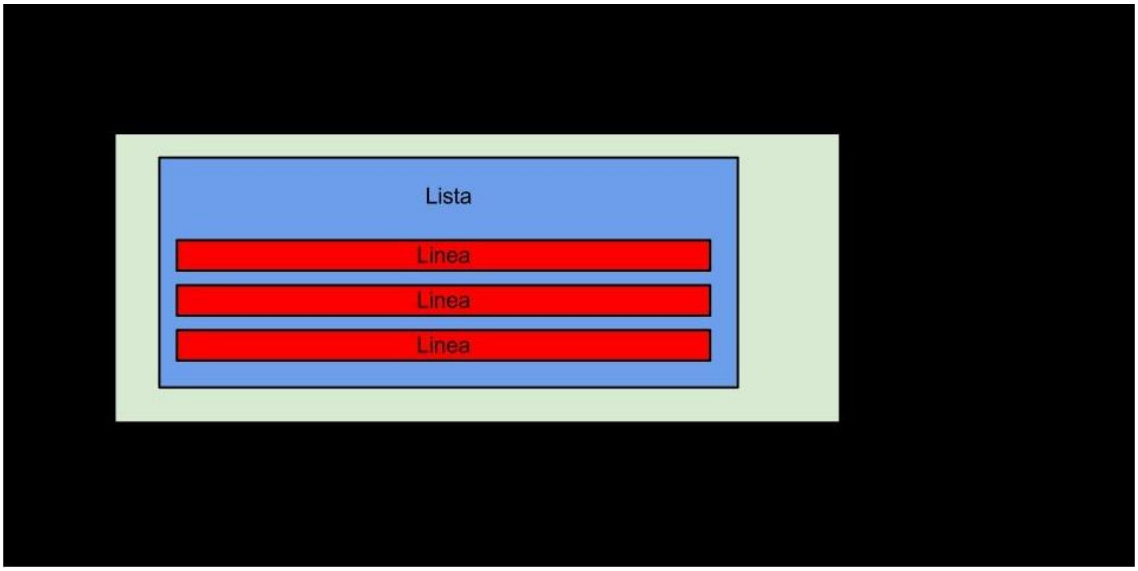


Figura 5.1.3.2 Representa la selección concreta, como por ejemplo productos.

Para que nos hagamos idea de la potencia de estas plantillas, se pueden cambiar todos los listados de productos (en los pedidos, en el carrito, o en la búsqueda de productos) de un solo plumazo editando solo un fichero. También se pueden cambiar todas las barras laterales cambiando el layout o los css cambiando base.

5.1.4 HTML

HTML son las siglas de *HyperText Markup Language* es el lenguaje usado para las paginas web y permite hipertexto en forma de etiquetas. Como cualquier página web esta utiliza HTML para la presentación final ante el usuario. Aunque todo el HTML de la aplicación en este caso esta dentro de las plantillas twig.

5.2 Capa de aplicación o capa de negocio

En la capa de negocio se ha utilizado PHP, programándolo en el IDE Netbeans usando el framework Symfony en la versión 2.0 aunque en el proceso de desarrollo se han actualizado el framework en parte por el sistema de validaciones ASSERTS que incluye symfony y que también he usado. También se ha usado yaml para realizar la estructuración del sitio así como las configuraciones sobre seguridad, autenticación y usuarios.

5.2.1 Symfony 2.0

Inicialmente la idea de Fabien Potencier era juntar varios frameworks existentes y empezó incluyendo Propel como ORM y Ruby on Rails para las plantillas. Symfony 2.0 es la segunda versión de Symfony.

"Symfony es un completo framework diseñado para optimizar, gracias a sus características, el desarrollo de las aplicaciones web. Para empezar, separa la lógica de negocio, la lógica de servidor y la presentación de la aplicación web. Proporciona varias herramientas y clases encaminadas a reducir el tiempo de desarrollo de una aplicación web compleja. Además, automatiza las tareas más comunes, permitiendo al desarrollador dedicarse por completo a los aspectos específicos de cada aplicación. El resultado de todas estas ventajas es que no se debe reinventar la rueda cada vez que se crea una nueva aplicación web."[6]

Desarrollado en PHP 5.3, es compatible con Windows, Unix o Linux. Así como acepta BBDD como MySQL, PostgreSQL, Oracle o Microsoft SQL Server. Hoy día también admite e integra herramientas para usar Propel o Doctrine como ORMs aunque cada vez mas enfocado a Doctrine 2.0. [6]

Características básicas: [6] [7]

- **Fácil de instalar y configurar en la mayoría de plataformas**, aunque es preferible para el desarrollo de grandes aplicaciones Web que para pequeños proyectos. No puedo más que asentir pero como explico anteriormente la curva de aprendizaje del uso del framework no es pronunciada pero una vez aprendidas cosas puede ser una herramienta muy potente y agilizar mucho el desarrollo. No es un framework recomendable para hacer un Proyecto final de carrera pero si es muy recomendable para trabajo en equipo de varias personas.
- **Es independiente del SGBD**, se pueden usar varios distintos como MySQL, PostgreSQL, Oracle o Microsoft SQL Server.
- **Sigue el MVC** y por lo tanto separa las lógicas del sistema consecuentemente a lo explicado en el apartado 4.2.
- **Basado en la premisa convenir y no configurar.** (CoC - Convención sobre Configuración). Se acuerda una configuración pre-establecida que permite al programador ponerse a trabajar sin tener que preocuparse por ello.
- **Usa PHP con OOP** (Programación Orientada a Objetos) y requiere PHP 5 aunque es altamente recomendable que sea la versión 5.3 o superior.
- **Sigue la mayoría de las mejores prácticas y patrones de diseño para la web.**
- **Versión estable.**
- **Fácil de extender permitiendo la integración con librerías de terceros.**

Características de automatización, Symfony automatiza la mayoría de elementos comunes de los proyectos web, como por ejemplo:

- La capa de internacionalización que incluye Symfony permite la traducción de los datos y de la interfaz, así como la adaptación local de los contenidos.
- La capa de presentación utiliza plantillas y layouts que pueden ser creados por diseñadores HTML sin ningún tipo de conocimiento del framework. Los helpers incluidos permiten minimizar el código utilizado en la presentación, ya que encapsulan grandes bloques de código en llamadas simples a funciones.
- Los formularios incluyen validación automatizada y relleno automático de datos ("repopulation"), lo que asegura la obtención de datos correctos y mejora la experiencia de usuario.
- Los datos incluyen mecanismos de escape que permiten una mejor protección contra los ataques producidos por datos corruptos.
- La gestión de la caché reduce el ancho de banda utilizado y la carga del servidor.
- La autenticación y la gestión de credenciales simplifican la creación de secciones restringidas y la gestión de la seguridad de usuario.
- El sistema de enrutamiento y las URL limpias permiten considerar a las direcciones de las páginas como parte de la interfaz, además de estar optimizadas para los buscadores.
- Los listados son más fáciles de utilizar debido a la paginación automatizada, el filtrado y la ordenación de datos.

Symfony además tiene una potente línea de comandos que facilitan generación de código, lo cual contribuye a ahorrar tiempo de trabajo. Aunque no he utilizado generadores automáticos exceptuando los de doctrine porque en la versión 2.0.0 de Symfony2 eran aun muy pobres y tenían escasa documentación, pero lo cierto es que han aumentado mucho ese tipo de herramientas. Algunos de los generadores que he visto serían de clases con mapeados incluidos, de vistas, de autenticaciones o de listados.

Symfony tiene muchas ventajas que facilitan la vida de los desarrolladores, automatizando algunas cosas o simplificando las tareas complejas. [7]

Todos los errores y las operaciones de la base de datos se muestran por el framework [7]. Permitiendo un entorno de desarrollo más informado.

5.2.2 Netbeans con symfony

Symfony 2 es un framework pensado para aplicaciones web y debido a que los servidores web son en su mayoría Linux, suelen centrarse en entornos de desarrollo para Linux. Por ejemplo a la salida de Symfony2, en el manual de instalación no estaban algunos pasos necesarios para Windows.

Otro ejemplo es que Symfony 1.4 se podía integrar en Netbeans pero solamente en Linux, la integración para Windows era mucho peor.

En el caso de Symfony 2 inicialmente no estaba integrado en ningún IDE pero debido a que en Netbeans era el IDE que había hecho esfuerzos por integrar la versión anterior, parecía que sería en un futuro el IDE que apostaría por integrar la versión 2.0.

Por ello se ha decidido usar Netbeans para la programación sobre Windows sin tener ningún tipo de integración ya que no existía ninguna aplicación que tuviese integración con Symfony2.

5.2.3 Yaml, enrutamiento, tipos de usuarios, seguridad y autenticación

Yaml (YAML Ain't Markup Language) es un lenguaje que integra symfony que en symfony se puede utilizar para varias cosas como: Definir la BBDD, hacer el mapeado de datos para Doctrine, etc… Pero hay dos funciones que Symfony usa normalmente Yaml: enrutamiento y seguridad.

El enrutamiento en symfony permite que se realice en diferentes ficheros que después el encadena uno tras otro. Debido a que yo non veía una clara diferenciación entre mis rutas yo he creado mi propio routing.yml y lo he colocado “debajo” del routing.yml genérico de symfony. Esto me permite tener mis rutas en mi Proyecto “Bundle” (el Bundle es la forma de organización que tiene symfony, es similar a los paquetes o espacios de nombres).

Un fichero de enrutamiento de symfony tiene la forma de un listado interminable de rutas de la forma:

```
producto_show:
    pattern:  /producto/show/{id}
    defaults: { _controller: PafpfcBundle:Producto:show }

estatica:
    pattern:  /sitio/{nombre}
    defaults: { _controller: PafpfcBundle:Default:estatica }
    requirements:
      nombre: inicio|contacto|privacidad|ayuda
```

El nombre de ruta: Esto permite desde twig o los controllers utilizar el nombre de la ruta directamente sin tener que recordar la url y de forma dinámica se generara, por ejemplo si cambio la ruta de routing.yml mi proyecto funcionara exactamente igual pero con otra ruta… los links no se perderán.

El pattern, patrón o ruta url: Aquí se define la ruta que usara el navegador y esta permite añadir información en forma de valores string/int cortos. Pero no Arrays o tipos complejos por razones obvias.

El controlador: es necesario pues en un MVC siempre que se hacen peticiones se hace al controlador. Así por simple que sea un controlador, ha de existir por cada petición que haya del navegador del cliente. Los controladores pueden requerir información adicional que no aparezca en el routing.

5.2.3.1 Seguridad

La seguridad se configura usando uno o varios ficheros en Yaml y permite asignar roles a clases por ejemplo. Así se simplifica mucho el trabajo. Todos mis "clientes" pueden iniciar sesión como "usuarios registrados". Sin embargo solo los administradores podrán acceder como tales, estos en nuestro caso están puestos en el fichero de seguridad que es inaccesible y no existe en la BBDD.

Symfony además cuenta con sistemas para evitar ataques XSS (cross-site scripting), y también contra CSRF (cross site request forgery) mantiene campos ocultos en los formularios. En la sección de persistencia 5.3.3 se explica porque también tiene seguridad contra "SQL Injections" gracias a Doctrine.

5.3 Capa de Persistencia

En la capa de persistencia se han utilizado facilidades del framework symfony y del ORM doctrine, utilizando para el mapeado anotaciones PHP. La base de datos que se ha usado ha sido MySQL con phpmyadmin y un servidor Apache.

5.3.1 Mapping

El mapeo en este tipo de sistemas ORM se suele hacer en XML pero symfony2 admite 3 diferentes sintaxis para hacerlo aunque a la hora de la verdad la "información" sobre el mapeo es la misma.

PHP Annotations. Pasándole un intérprete que busca tokens de una forma concreta symfony puede utilizar información del mapeado que esta contenida dentro de las mismas clases o entidades que serán mapeadas.

```
<?php
/**
 * @Entity
 * @Table(name="my_persistent_class")
 */
class MyPersistentClass
{
    //...
}
```

XML. Funciona de forma similar a Hibernate, en mi opinión es menos comodo pues tienes el mapeado aparte.

```
<doctrine-mapping>
  <entity name="MyPersistentClass" table="my_persistent_class">
      <!-- ... -->
  </entity>
</doctrine-mapping>
```

YAML. También se incluye aparte en otro fichero.

```
MyPersistentClass:
  type: entity
  table: my_persistent_class
  # ...
```

Al final la decisión de que sintaxis utilizar parece muy secundaria.

5.3.2 Creación de BBDD usando doctrine

La consola de symfony2 con doctrine permite utilizar herramientas automáticas de varios tipos. Aparte de los generadores de código automático descritos anteriormente, permite generar la base de datos a partir de las entidades y sus mapeados. Permite a través de consola de generar entidades ya mapeadas con asociaciones sencillas de forma automática.

Uno de los comandos más potentes serían **create** y **update** estos se puede utilizar con diferentes opciones como "–force" o "–dump—var", el primero realiza todas las instrucciones necesarias para alterar y crear las tablas necesarias. El segundo imprime el código SQL necesario para hacerlo.

Este generador de BBDD es muy potente y permite facilitar mucho el trabajo y sobre todo crear y alterar BBDD con muchísima rapidez.

Después podrán ser alteradas la base de datos pero siempre realizando los cambios que se deseen a mano también en las entidades.

5.3.3 Repositorios

Symfony utiliza junto con doctrine una serie de formas de acceder a los datos. Lo más normal es utilizar repositorios que son configurados aparte.

Estos tienen funciones de búsqueda que se heredan de una clase Repositorio del framework (de ahí lo interesante de tener id en la BBDD), pero aun mas interesante es crear otras búsquedas pudiendo hacer una búsqueda según tus necesidades.

Para crear búsquedas Doctrine puede hacerlo de tres formas diferentes:

Una es creando consultas nativas en SQL con el método del gestor de entidades "createNativeQuery".

También puedes usar DQL directamente con el método "createQuery"

Y la tercera opción sería crear instanciar al creador de consultas "QueryBuilder" que te permite poner por secciones lo que quieres hacer y el propio Doctrine te generara internamente la DQL y la lanzara.

La principal ventaja del DQL que se ha observado es la imposibilidad de hacer ataques "SQL Injection" porque los valores son pasados en su forma tipo, esto hace que no puedan incluirse sentencias sino solo valores. Lo malo es que puede resultar algo complejo de entender, especialmente para consultas complejas.

5.4 Otros

Relativo a ninguna de las capas: diagramas de Secuencia con bouml, diagramas de clases con Argouml, Diagrama de la BBDD usando una importación/exportación a MySQL Workbench, el resto de dibujos y diagramas con GoogleDocs Drawing.

6. Bibliografía

[1]Modelo Vista Controlador de Xerox Park Fuente: http://heim.ifi.uio.no/~trygver/themes/mvc/mvc-index.html accedida en 2012

[2] W. Ambler, Scott. Mapping Objects to Relational Databases: O/R Mapping in detail. Fuente: http://www.agiledata.org/essays/mappingObjects.html accedida en 2012

[3] Eric A. Meyer. (2011) CSS Tools: Reset CSS.

Fuente: http://meyerweb.com/eric/tools/css/reset/ accedida en 2012

[4] Manual Twig. Fuente: http://twig.sensiolabs.org/ accedida en 2012

[5] Nathan Smith. Fuente: http://960.gs/ accedida en 2012

[6] Zaninotto, F. y Potencier F. 2007. *The Definitive Guide to Symfony*. Apress. pp. 486. ISBN 978-1-59059-786-6. Versión en castelláno. Symfony la guía definitiva

[7] Jarmołowicz J., Zabierowski W., Napieralski A. 2008. Presentation of Improvements for PHP Programmers, Based on Symfony Framework. Creation of Example Portal and Description of Used Technology. *TCSET 2008 - Modern Problems of Radio Engineering, Telecommunications and Computer Science - Proceedings of the International Conference* , art. no. 5423449 , pp. 595-597

[8] W. Ambler, Scott. *Agile Model Driven Development with UML 2*. Capitulo 11. Fuente:

http://www.agilemodeling.com/artifacts/sequenceDiagram.htm accedida en 2012

[9] Fuentes-Fernández L. y Vallecillo-Moreno A. 2004. *UML and Model Engineering*. The European Journal for the Informatics Professional Vol. 5, No2, Abril 2004.

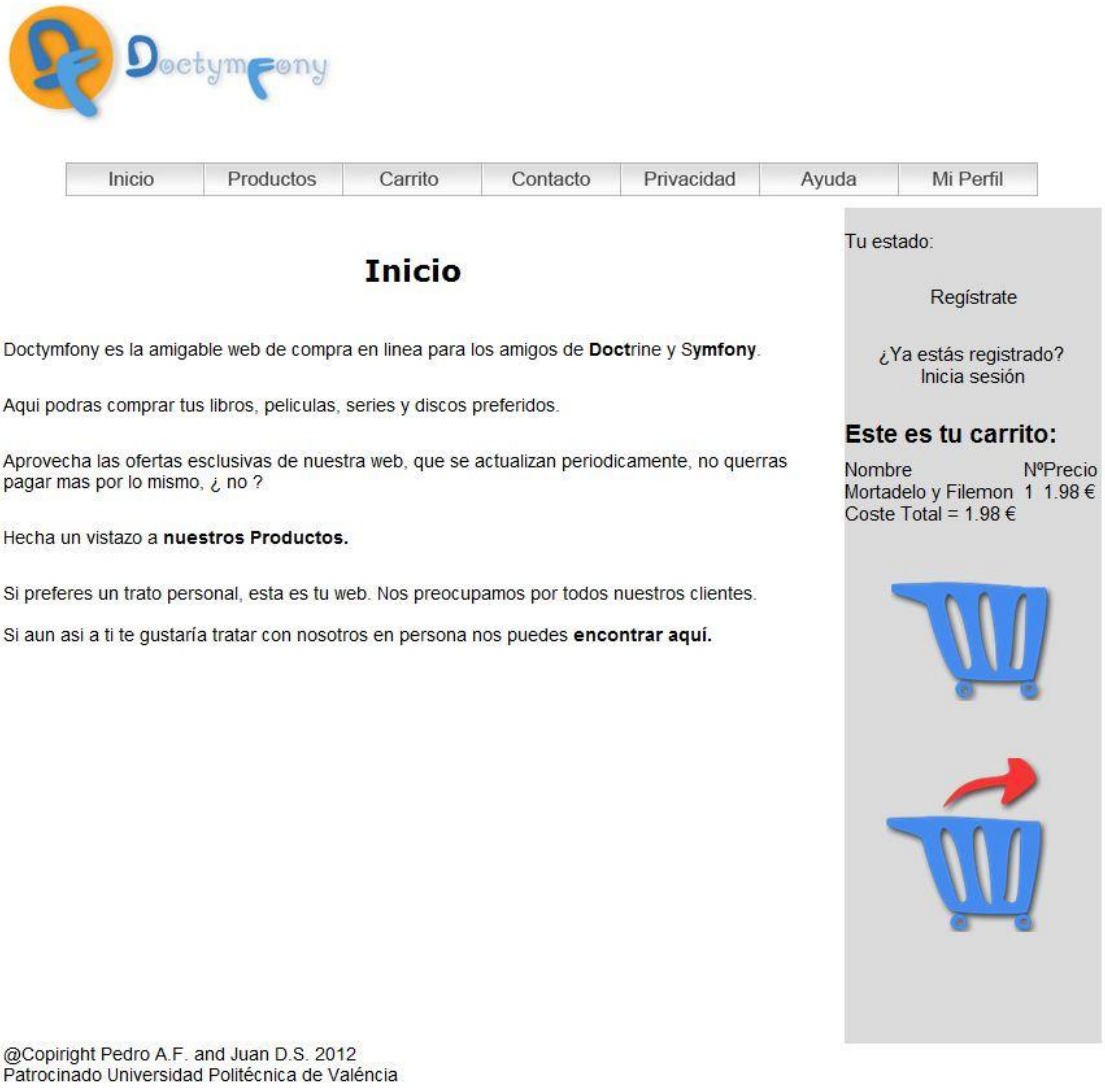
[10] Kruchten Philippe. 2004 *The Rational Unified Process an Introduction.* Pearson Education Inc.

7. Manual de usuario

La aplicación se puede acceder desde el navegador web, con la dirección url: http://localhost:2020/app_dev.php/pfc/sitio/inicio

7.1 Estructura y navegación

En la imagen inferior puedes ver la estructura de la aplicación.



Los diferentes apartados se explican a continuación:

A) Cabecera: siempre que quieras podrás hacer clic en nuestro logo o nombre la tienda para ir a la pantalla de Inicio.
B) Menú: Usa el menú para navegar entre las diferentes secciones de la web.
C) Sub-menú: Aparece cuando son necesarias opciones especiales en una sección.
D) Titulo: Junto con el nombre que aparece en la pestaña de tu navegador, te informa de en que sección estas.
E) Barra lateral: Aquí podrás ver siempre tu estado así como las opciones de autenticación. También podrás ver tu carrito de la compra.
F) Pie de página: Da información sobre la web como el copyright.

7.2 Tu estado

En la barra lateral aparece siempre tu estado.

1) Si nunca te has registrado, por favor regístrate, solo te costara unos segundos y será necesario para comprar.
2) Si ya te has registrado anteriormente por favor introduce tu alias y contraseña para acceder. Si has olvidado tu contraseña, ve a la sección de contacto y contacta con nosotros.
3) Si ya has iniciado sesión y deseas salir por favor dale a “cerrar sesión”.

7.3 Secciones:

El Menú es el centro de la navegación en el portal, úsalo para desplazarte entre las diferentes secciones, para simplificar nos referiremos al menú explicando las rutas con la forma: Menú>Apartado.

El menú de un usuario:

El menú de un administrador:

7.3.1 Inicio

Inicio es la pantalla de presentación de nuestra tienda, puedes verla siempre que quieras haciendo clic en Menú>Inicio o sobre nuestro logo o nombre.

7.3.2 Productos

Aquí tienes disponible un listado con todos los productos de la tienda. Podrás entrar desde Menú>Productos. Esta sección dispone de sub-menú para poder acotar tus búsquedas. Para ver la sección de administración de productos ir a la sección de administrador.

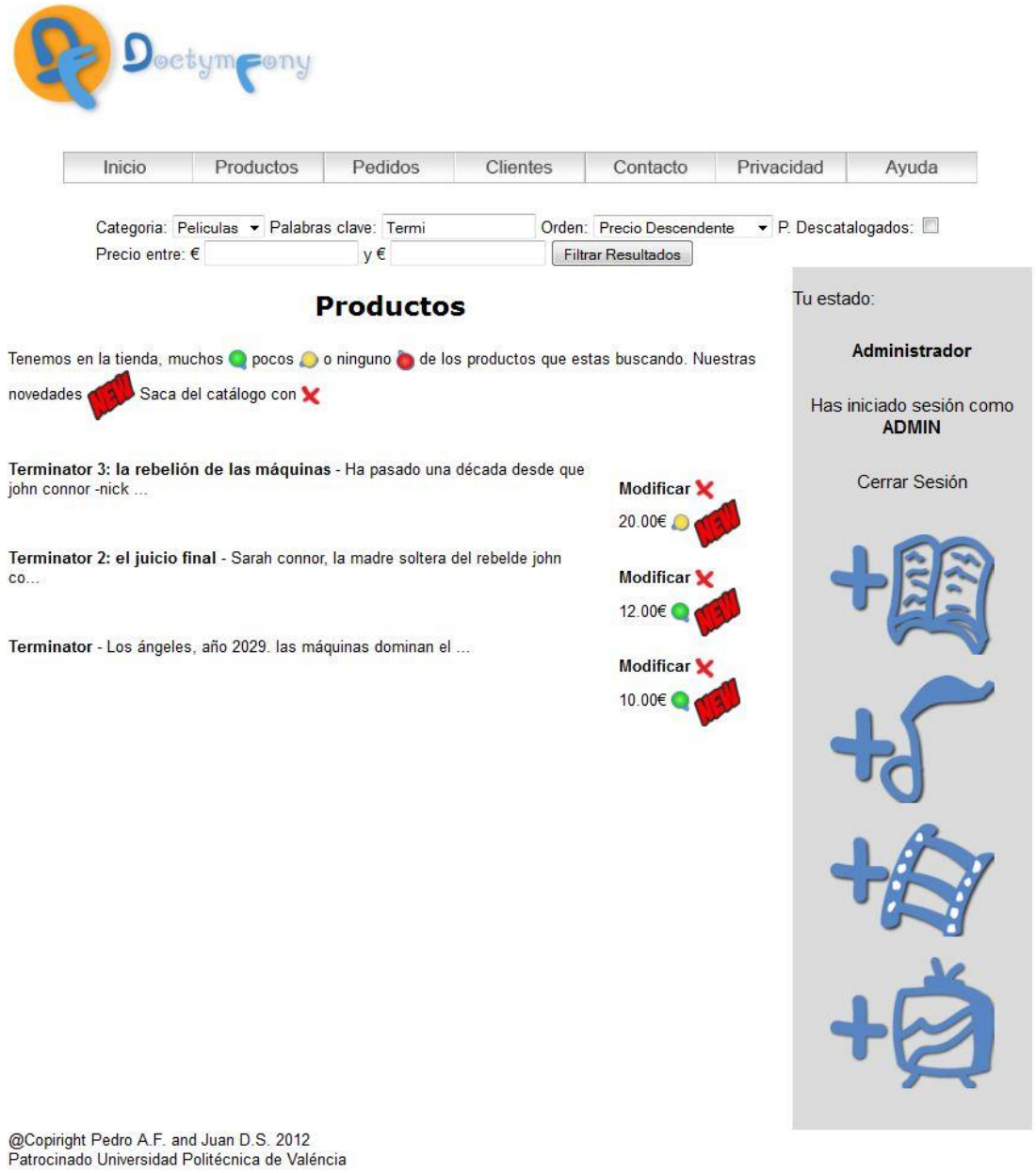


Figura 7.3.2 Muestra una búsqueda en Productos realizada por un usuario anónimo.

Búsqueda

Marca los parámetros que te interesen para la búsqueda y clica en filtrar resultados.

- Puedes marcar tu búsqueda por el tipo de producto: película, serie, música o libros. Si no marcas nada o marcas todos, buscara entre todos los productos.
- Búsqueda por palabras clave, si quieres puedes poner aquí palabras relacionadas con el producto y si están en el nombre o en la descripción te los encontrará.

- Búsqueda por rango de precios, marca el precio máximo y/o el precio mínimo para ver solo los productos cuyo precio encaje en tus preferencias.
- Ordena tu búsqueda según prefieras, por el nombre o por el precio, ya sea de forma ascendente o descendente.

Puedes hacer cualquiera de las combinaciones que nuestro buscador encontrara por ti los productos que encajen en esa descripción.

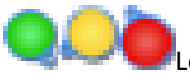
Información

La información sale en forma de una lista de elementos donde puedes ver el nombre y la descripción de todos y cada uno de los elementos y a su derecha una serie de iconos y el precio del artículo.

El precio que ves en la lista es el precio definitivo, los descuentos ya han sido aplicados.


La Lupa significa “Ver detalles de producto” y te dirige a una pantalla con toda la información detallada sobre ese producto. También puedes acceder a los detalles del producto haciendo clic sobre su nombre.


El carrito de la compra con la flecha verde permite añadir un producto al carrito.


Los colores del Semáforo te informan de la cantidad de elementos de ese producto que quedan en nuestra tienda. Verde es que tenemos en grandes cantidades, amarillo es que estamos cerca de agotarlo y rojo significa que se ha agotado y no podremos venderlo hasta recibir nuevos productos. Siendo rojo, el producto aun se podrá vender, solo tendrás que esperar a que se reponga.

Cuando un producto tiene esta etiqueta es que es un producto reciente, una novedad y lo estamos vendiendo desde hace menos de 20 días.

7.3.3 Carrito

Aquí es donde se almacenarán los productos que vayas eligiendo y podrás verlo siempre que quieras, tienes una versión reducida en la barra lateral pero en esta sección podrás ver todos los detalles. Desde aquí podrás comprar.

 Puedes acceder al carrito desde el menú o haciendo clic sobre la imagen del carrito de la barra lateral.

En el carrito de la compra puedes ver lo que has ido añadiendo, los detalles como el número de elementos añadidos y el precio de cada uno de ellos así como el precio total de una línea.

 Haciendo clic sobre el carrito añadirás un elemento a tu lista de ese producto. Si lo prefieres puedes escribir en el cuadro de texto el número de elementos que deseas añadir y hacer clic sobre el botón “send”.

También te aparece el precio de todo el carrito de la compra con la forma: Coste Total = XX.XX.

Puedes comprar todos los productos haciendo clic sobre ✔“Continuar con la compra”✔ .

O puedes vaciar el carrito haciendo clic sobre “Vaciar el carrito” o sobre la imagen :  .
Siempre puedes hacer clic sobre vaciar carrito desde la barra lateral.

Si quieres continuar añadiendo productos haz clic sobre Menú>Productos.

7.3.4 Contacto

En la página de contacto esta explicado quienes somos y donde encontrarnos, puedes llegar a ella a través de Menú>Contacto o desde Inicio a través del enlace en “encontrar aquí”.

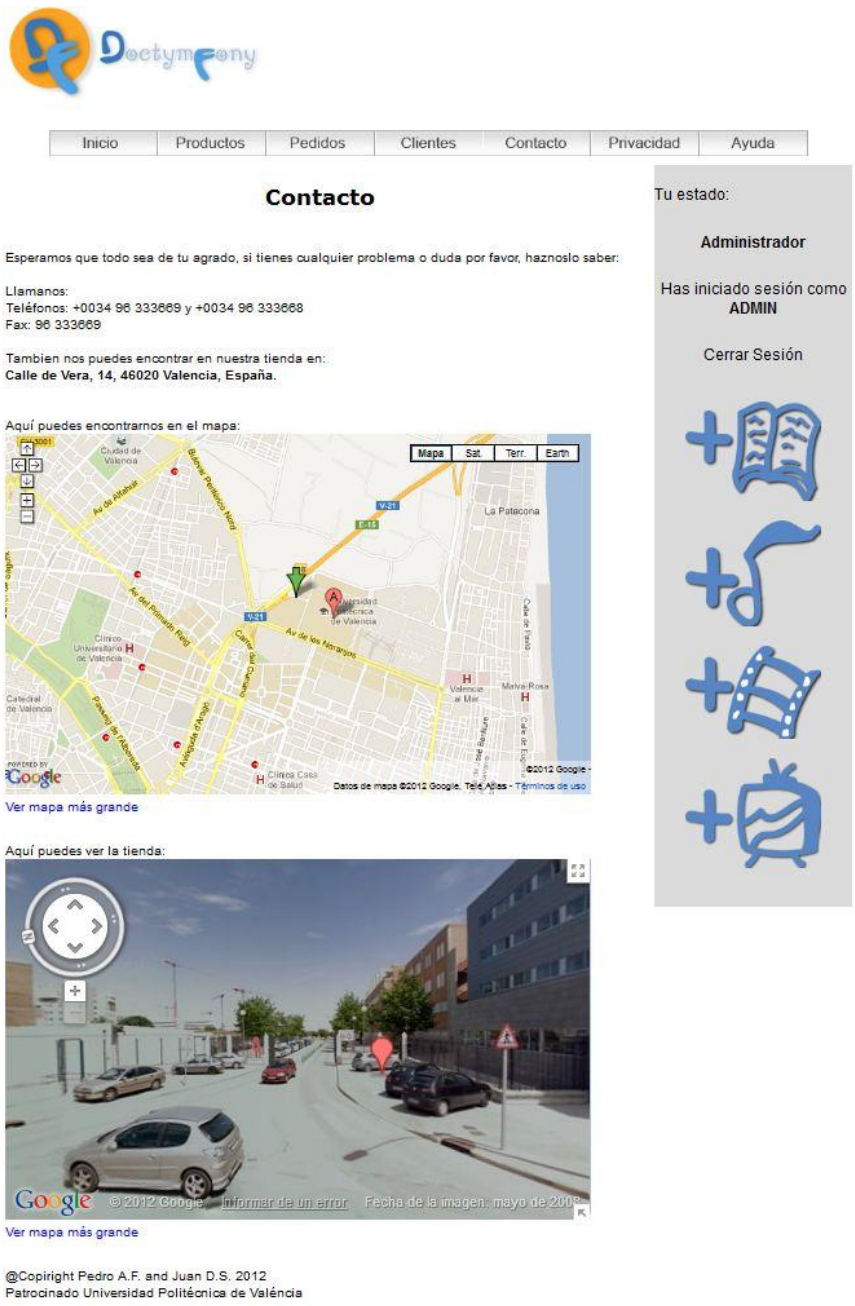


Figura 7.3.4 Muestra la página de contacto.

Puedes ver toda la información de como contactar con nosotros: llamarnos por teléfono, enviarnos un fax, mandarnos un correo electrónico, una carta o venir a vernos en persona.

7.3.5 Privacidad

Aquí puedes encontrar información sobre las políticas de la empresa a la hora de manejar tus datos. Puedes verlo haciendo clic en Menú>Privacidad o haciendo clic en cualquiera de los links referentes al copyright.

7.3.6 Ayuda

Aquí encontraras toda la ayuda para entender nuestra tienda y como moverte por ella. Incluyendo este manual de usuario.

7.3.7 Mi perfil

Aquí puedes ver información referente a tus datos en la tienda. También podrás ver todos tus pedidos y en que estado se encuentran. Permitiéndote en todo momento acceder a toda tu información. Podrás desde aquí haciendo clic sobre un pedido verlo en detalle.

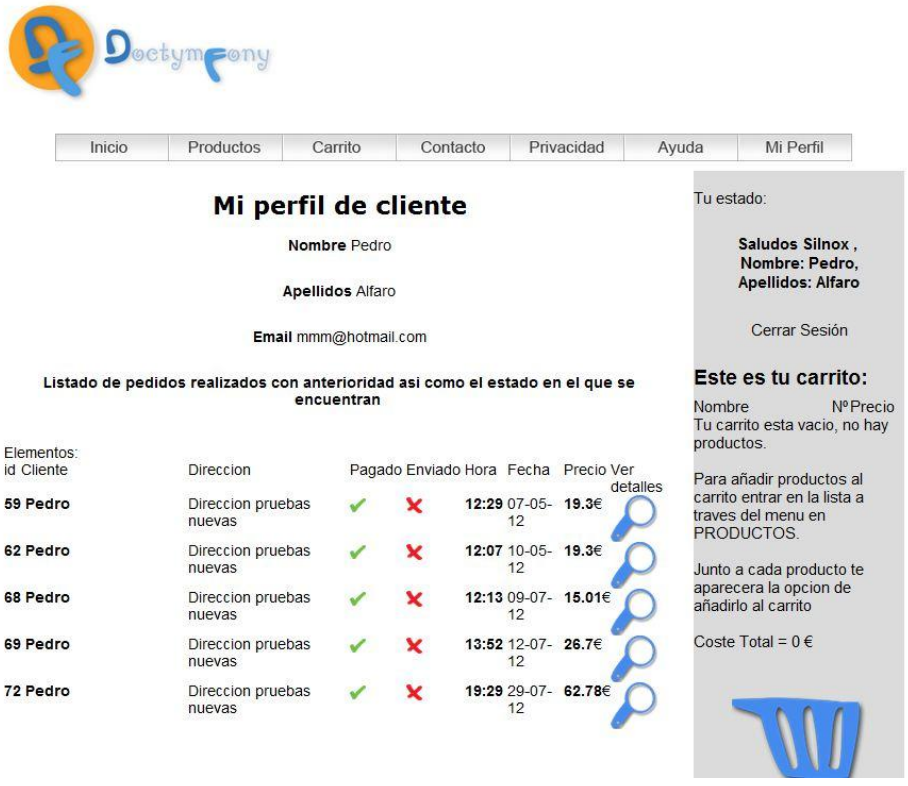


Figura 7.3.7 Mi perfil de un cliente.

7.3.8 Sección de administración

Las siguientes secciones solo son accesibles habiendo accedido a la tienda como administrador:

7.3.8.1 Administrar Productos

Aquí tienes disponible un listado con todos los productos de la tienda. Podrás entrar desde Menú>Productos. Esta sección dispone de sub-menú para poder acotar tus búsquedas.

Búsqueda

La búsqueda del administrador es idéntica a la del usuario, exceptuando la característica descatalogados que permite a un administrador buscar entre los artículos que en algún momento se eliminaron del catálogo público.

- Marca la casilla descatalogados para ver los productos que ya no están en catálogo pero que alguna vez lo estuvieron.
- Puedes marcar tu búsqueda por el tipo de producto: película, serie, música o libros. Si no marcas nada o marcas todos, buscará entre todos los productos.
- Búsqueda por palabras clave, si quieres puedes poner aquí palabras relacionadas con el producto y si están en el nombre o en la descripción te los encontrará.
- Búsqueda por rango de precios, marca el precio máximo y/o el precio mínimo para ver solo los productos cuyo precio encaje en tus preferencias.
- Ordena tu búsqueda según prefieras, por el nombre o por el precio, ya sea de forma ascendente o descendente.

Puedes hacer cualquiera de las combinaciones que nuestro buscador encontrara por ti los productos que encajen en esa descripción.

Información

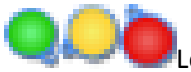
La información sale en forma de una lista de elementos donde puedes ver el nombre y la descripción de todos y cada uno de los elementos y a su derecha una serie de iconos, “Modificar” y el precio del artículo.

El precio que ves en la lista es el precio definitivo, los descuentos ya han sido aplicados.

Haciendo clic sobre el nombre de un producto podrás ver los detalles pero no editarlos.

El enlace “Modificar” te permite ir al formulario de edición de productos donde podrás cambiar todos los atributos de ese producto.

 Haz clic en el aspa roja para descatalogar un producto.

 Los colores del Semáforo te informan de la cantidad de elementos de ese producto que quedan en nuestra tienda. Verde es que tenemos en cantidad, amarillo es que estamos cerca de agotarlo y rojo significa que se han agotado.

 Cuando un producto tiene esta etiqueta es que es un producto reciente, una novedad y lo estamos vendiendo desde hace menos de 20 días.

Barra lateral

Además del estado que siempre esta presente en la barra lateral. También hay 4 imágenes que representan los cuatro tipos de productos de nuestra web con un signo más delante. Cada uno de ellos se utiliza para ir al formulario de creación de un nuevo producto del tipo correspondiente.

 Añadir un libro nuevo.

 Añadir un elemento nuevo de música.

 Añadir una nueva película.

 Añadir una nueva serie o una temporada nueva de una serie ya existente.

7.3.8.2 Pedidos

Sección del portal destinada a la administración de pedidos. Tendrás que marcar los envíos y los pagos en esta sección para mantener los datos de la tienda al día. Esta sección dispone de sub-menú para poder acotar tus búsquedas.

Búsqueda

La búsqueda de pedidos te permite dos cosas:

- Buscar un pedido en concreto usando el código de identificación que todos los pedidos tienen y es único.
- O buscar utilizando un margen de fechas, en el margen puedes poner una o dos fechas y buscará los pedidos realizados entre esas dos fechas.

Ten en cuenta que debido a que el identificador es único, no tiene sentido y no es posible realizar una búsqueda usando tanto el identificador como las fechas. Cuando hayas rellenado los campos que quieras haz clic en filtrar resultados.

Información

De cada uno de los pedidos puedes ver:

- Los nombres de los clientes que han realizado el pedido. Haciendo clic sobre ellos te llevara a la sección ver cliente donde podrás ver tanto sus datos como sus pedidos.
- La dirección de facturación del pedido.
- Si el pedido ha sido pagado ya o aún no ha sido pagado.
- Si el pedido ha sido enviado ya por mensajería o esta por enviar.
- La fecha y hora de la realización del pedido.
- El precio pagado por el pedido.
- Ver el pedido detallado haciendo clic sobre la lupa.
- Enlaces para gestión. Enviado/No enviado, Pagado/No pagado.

✔ Representa SI. Y ✘ representa NO.

Gestión

Para gestionar los pedidos el Administrador debería marcar como no pagados aquellos pedidos que han tenido algún tipo de error en el cobro. O han cancelado sus transacciones en los 3 días que dan de margen los bancos.

Cuando un pedido que ha tenido incidencias en el pago se cobra, por banco o en persona. Se debe marcar el pedido como pagado.

Cuando un pedido ha sido pagado es el trabajo del administrador enviarlo por mensajería y marcarlo como enviado en el sistema.

7.3.8.3 Clientes

Sección del portal destinada a la administración de clientes. Podrás acceder a todos los datos de los clientes de forma detallada. Esta sección dispone de sub-menú para poder acotar tus búsquedas.

Búsqueda

La búsqueda de clientes se puede hacer de dos formas:

- Usando el identificador de cliente que al igual que los pedidos o los productos tiene un número que lo identifica, y que es único para ese cliente. Poniendo ese número en el buscador se encontrará al cliente.
- Usando palabras clave el buscador intentará encontrar a los clientes que tienen alguna de esas palabras clave en el Nombre o en sus Apellidos.

Ten en cuenta que debido a que el identificador es único, no tiene sentido y no es posible realizar una búsqueda usando tanto el identificador como las palabras clave. Cuando hayas rellenado uno de los campos haz clic en “Filtrar Resultados”.

Información

Se mostrarán los siguientes datos de cada cliente.

- Nombre y Apellidos, haciendo clic en el nombre puedes ir a la pantalla para ver detalles de ese cliente.
- Alias, el nombre que utiliza dentro del sistema.
- Email, el email que utilizó para registrarse.
- Fecha de ingreso o registro
- El número de pedidos realizados hasta la fecha. (No tiene en cuenta ni si han sido pagados ni si han sido enviados)
- Fecha en la que se realizó el último pedido. (No tiene en cuenta ni si ha sido pagado ni si ha sido enviado)

7.4 Detalles

Las pantallas de detalle son las de ver pedido, ver cliente, mi perfil y ver producto. Debido a que utilizan las mismas imágenes que las secciones a las que pertenecen, no las voy a explicar en tanto detalle.

7.4.1 Ver Producto

Permite ver en detalle cualquier producto de la tienda, aunque solo permitirá ver productos descatalogados al Administrador. Sale toda la información del producto. Desde aquí se puede añadir al carrito de la compra siempre que no se sea un Administrador. Para ver detalles sobre la información gráfica, ver las secciones de Productos 7.3.2 y Administrar Productos 7.3.8.2.

Se puede acceder a “ver producto” desde:

- Cualquier pedido que lo incluya.
- Desde la sección Productos en Menú>Productos.
- Desde la barra lateral si no se es administrador y se tiene añadido ese producto al carrito.
- Desde la sección carrito en Menú>Carrito.

7.4.2 Ver Cliente

Permite ver todos los datos importantes de un cliente registrado. A un cliente registrado le permite ver sus detalles. Es igual que la pantalla de Mi perfil que sale explicada en la sección 7.3.7. Permite la navegabilidad dentro del cliente para ver pedidos, o productos de pedidos.

Se puede acceder a “Ver Cliente” desde:

- Si se es un cliente que ha iniciado sesión, desde la sección Mi Perfil en Menú>Mi Perfil.
- La sección Clientes en Menú>Clientes.
- El detalle de un pedido en ver pedido.
- La sección pedidos en Menú>Pedidos.
- El carrito en los pedidos si es que el cliente los tiene en la sección Menú>Carrito.

7.4.3 Ver Pedido

Permite ver todos los datos importantes de un pedido realizado. Permite ver los detalles del pedido, para ver la información grafica ir a sección de pedidos 7.3.8.2.

Se puede acceder a "Ver Pedido" desde:

- La sección Pedidos en Menú>Pedidos.
- EL carrito de la compra si el cliente tiene pedidos anteriores en Menú>Carrito.
- Estando en Ver Cliente, si el cliente tiene pedidos.
- Desde Mi perfil si es un cliente con pedidos anteriores en Menú>Mi Perfil.

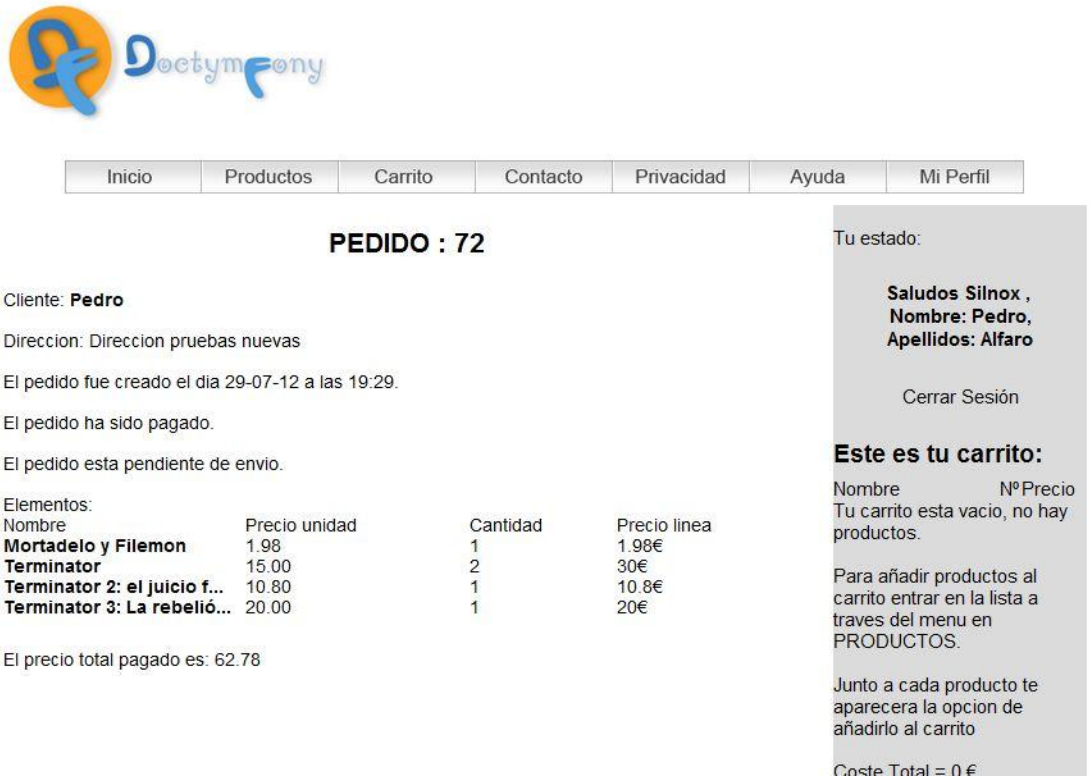


Figura 7.4.3 Ver Pedido, visitado por un cliente registrado que esta viendo pedidos ya realizados.

7.5 Manual de instalación

Preparar un Servidor Apache con PHP 5 instalado. Para ello utilizar WAMP Server que es un paquete ya preparado para Windows o LAMP Server que es el mismo paquete preparado para Linux. Ambos paquetes tienen sus propias guías de instalación que cambia según la versión. WAMP es el acrónimo de Windows Apache MySQL PHP. Asegurarse que la versión de PHP es superior a la 5.3. (http://www.wampserver.com/en/)

Una vez hecho esto, colocar la Carpeta Symfony en el directorio raíz público del servidor. (normalmente llamada www)

Configurar el Servidor como Localhost inicialmente para realizarle una prueba. Iniciar WAMP o LAMP (ejecutarlo) y teniéndolo ejecutado hacer clic en “Iniciar los servicios”.

Teniendo el Servidor funcionando probar la aplicación escribiendo la siguiente dirección URL:

http://localhost:2020/app_dev.php/pfc/sitio/inicio

Después probar la siguiente:

http://localhost:2020/ pfc/sitio/inicio

Si las dos han funcionado tienes el servidor funcionando perfectamente solo será necesario cambiar la configuración para que deje de estar configurado como localhost.