Treball Fi de Grau

GRAU D'ENGINYERIA INFORMÀTICA

Facultat de Matemàtiques

Universitat de Barcelona

YOUMARKET: PLATAFORMA DE COMERCIO ELECTRÓNICO DE PROXIMIDAD

David Corto Camacho

Director: Simone Balocco

Realizado en: Departament de Matemàtica Aplicada i Anàlisi. UB

Barcelona, 15 de Junio de 2014

Índice

Abstract (Castellano)

El proyecto documentado a continuación llamado “YouMarket: Plataforma de comercio electrónico de proximidad” es una plataforma web destinada a la venta de productos de distintos comercios de una área geográfica concreta.

La principal función de la plataforma es la de ofrecer un catálogo de productos geográficamente cercanos a sus potenciales compradores. Además, está la idea de reunir tiendas que no disponen de medios para mantener una tienda online propia. De esta forma, YouMarket pretender ser el punto de entrada de muchos comercios al e-commerce.

La plataforma ha sido desarrollada pensando siempre en términos de escalabilidad y modularidad para adaptar la plataforma a diferentes necesidades futuras y diferentes escenarios.

En el presente documento se detalla el proceso de diseño, análisis, planificación y arquitectura de la plataforma proyectada.

Abstract (Català)

El projecte documentat a continuació amb nom “YouMarket: Plataforma de comercio electrónico de proximidad” és una plataforma web destinada a la venda de productes de diferents comerços d’una àrea geogràfica específica.

La funció principal de la plataforma és la d'oferir un catàleg de productes geogràficament propers a tots els seus potencials compradors. A més, existeix la idea de reunir botigues que no disposen de mitjans per a mantenir una botiga online pròpia. D'aquesta manera, YouMarket vol ser el punt d'entrada de molts comerços al e-commerce.

La plataforma ha sigut desenvolupada pensant sempre en termes d’escalabilitat i moduladitat per a adaptar la plataforma a diferentes necessitats futures i diferents escenaris.

El present document detalla el procés de disseny, anàlisi, planificació i arquitectura de la plataforma projectada.

Abstract (English)

The following project called "YouMarket: Plataforma de comercio electrónico de proximidad" is a web platform for the product sale from different stores in a specific geographic area.

The main goal of the platform is to provide a products catalog of geographically close potential buyers. There is the idea of bring to stores that don't have the means to maintain their own online store. YouMarket way claim to be the entry point of many stores to the e-commerce.

The platform has been developed in terms of scalability and modularity to adapt the platform to different future needs and different possible scenarios.

This document outlined in detail the process of design, analysis, planning and architecture of the proposed platform.

1. Introducción

El comercio electrónico generalista en España viene dominado básicamente por Amazon, a pesar de la entrada recientemente de nuevos competidores como Rakuten [1] o Alibaba [2], Amazon sigue dominando en nuestra geografía. Sus claves del éxito se pueden resumir en dos: logística y el poco margen sobre producto para mantener los precios bajos, todo basado en el gran volumen de transacciones que operan cada día.

Para el pequeño comerciante es muy difícil poder competir por precio, esa vía queda prácticamente descartada. La única forma de vender con rentabilidad y regularmente es logrando una gran fidelización del cliente, aportando para ello, un gran valor añadido al servicio prestado por vía de la atención al cliente y la proximidad de sus clientes habituales, además de ser solidarios con la sociedad que les rodea. A pesar de las dificultades de los pequeños comerciantes, cada día hay más consumidores comprando en Internet todo tipo de productos o servicios; debido a este hecho, lo más importante es tener presencia en Internet y vender de la forma más económica posible y además facilitando la interacción entre comprador y vendedor por vía de un proceso sencillo para las dos partes.

1.1. Análisis del estado del e-commerce en España

Se han analizado los datos disponibles en el Observatorio Nacional de las Telecomunicaciones y de la SI (ONTSI [3]), se detecta una pequeña desaceleración del volumen del e-commerce en la economía española. La profunda crisis económica que sufre la economía es la causa principal de este hecho, pero no todo son malas noticias: en la actualidad, más del 50% de las empresas españolas comercializa mediante esta vía (51,4%), y un 10,9% prevé recurrir a ello en los próximos 2 ó 3 años. Si se analiza por tamaño de las empresas, se puede ver que las pequeñas empresas (de 2 a 49 empleados) están 10 puntos por debajo en el uso de e-commerce, lo que da mucho margen de crecimiento y un buen momento para aportar soluciones que les permita apostar por este canal de venta analizando las previsiones de futuro.

Volumen del comercio electrónico en España:

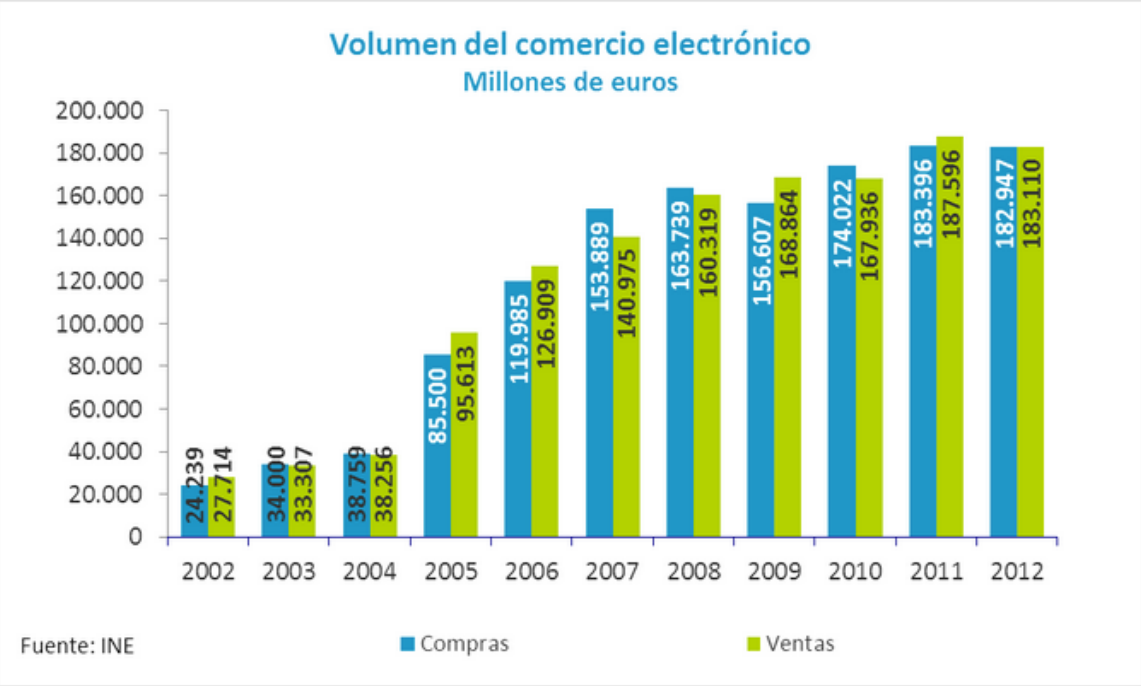


Ilustración 1: Volumen del comercio electrónico en España. Fuente [41].

En 2012 el volumen de las ventas por Internet ha disminuido un 2,4% respecto al año anterior, alcanzándose los 183.110 millones de euros (Ilustración 1).

En cuanto a las compras por Internet, éstas han disminuido en un 0,3% en 2012 respecto de 2011, situándose en los 182.947 millones de euros (Ilustración 1).

Cifra del comercio electrónico sobre el volumen total de negocio de la empresa:

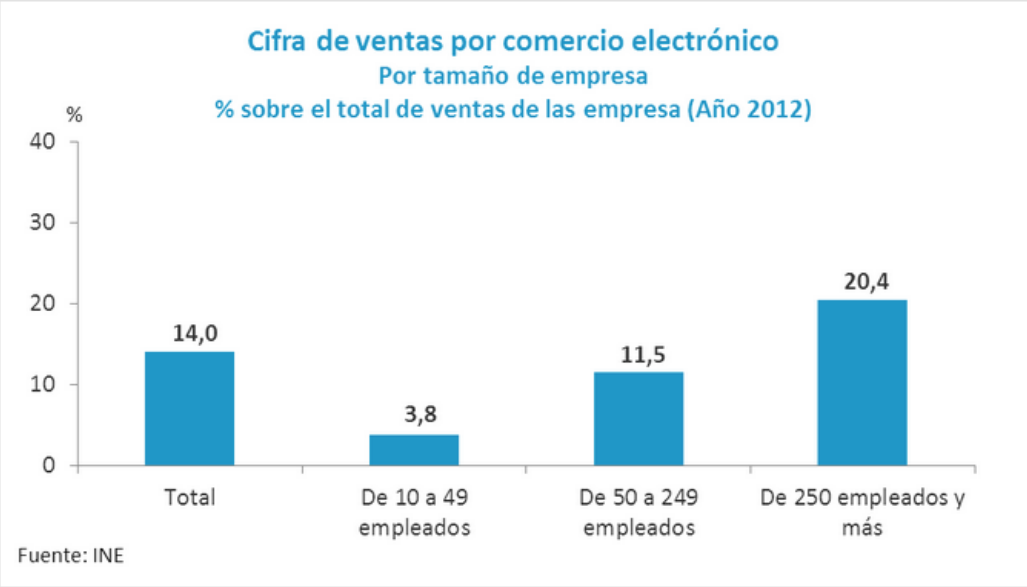


Ilustración 2: Cifra de ventas por comercio electrónico. Fuente [42].

Adicionalmente también en 2012 el 14% de las ventas de las empresas se realizaron a través de comercio electrónico, creciendo 0,3 puntos porcentuales respecto al año anterior (Ilustración 2).

Productos y servicios adquiridos por Internet:

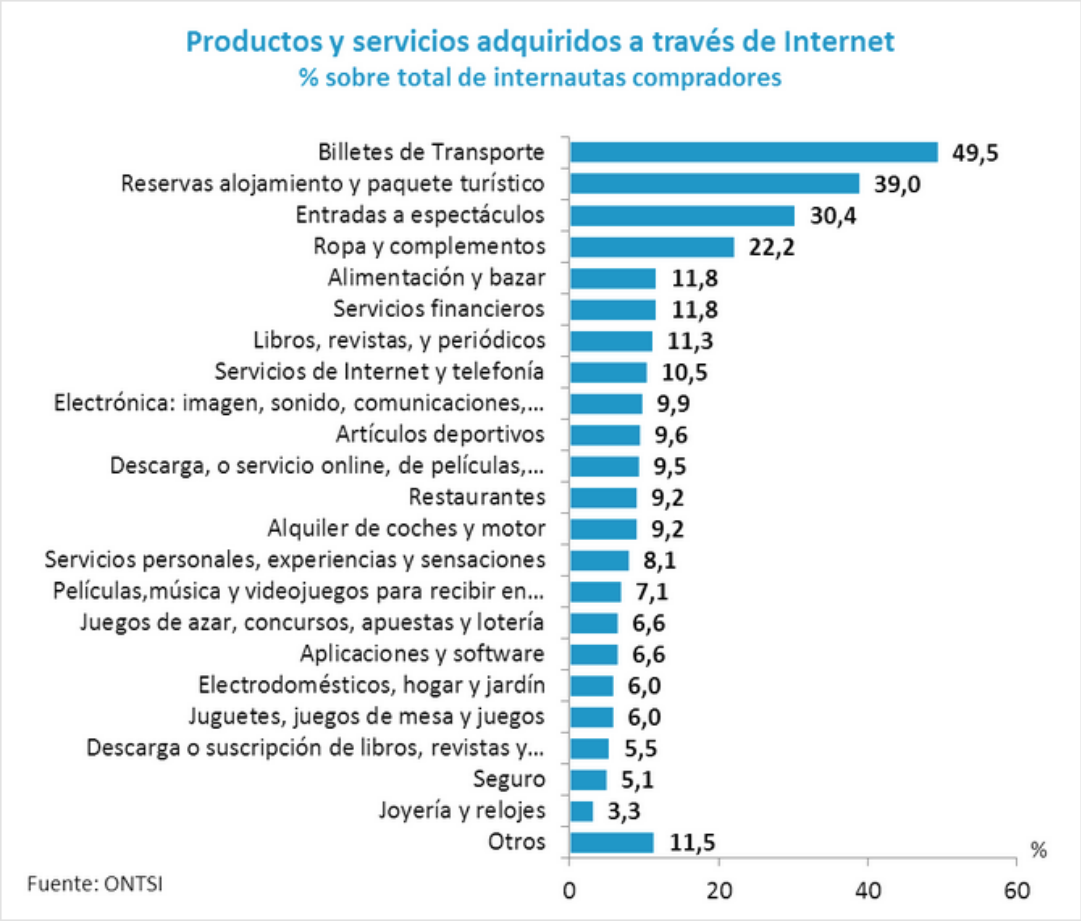


Ilustración 3: Productos y servicios adquiridos a través de Internet. Fuente [40].

Los productos y servicios más adquiridos por Internet son las actividades de ocio y el sector turístico, que siguen siendo claves en las compras en Internet. Cabe destacar la compra de ropa y complementos en el ranking (Ilustración 3).

Empresas que realizan comercio electrónico:

En la actualidad más del 50% de las empresas españolas comercializa mediante comercio electrónico (51,4%), y un 10,9% prevé recurrir a ello en los próximos 2 ó 3 años (Ilustración 4).

Sobre las empresas más pequeñas (entre 2 y 49 trabajadores), es destacable mencionar que la adopción de estas nuevas tecnologías ha tenido una evolución creciente en los

últimos 3-4 años de más del 10%, pudiendo afirmar que el 50% de las empresas hacen uso del comercio electrónico en 2011 (Ilustración 5).

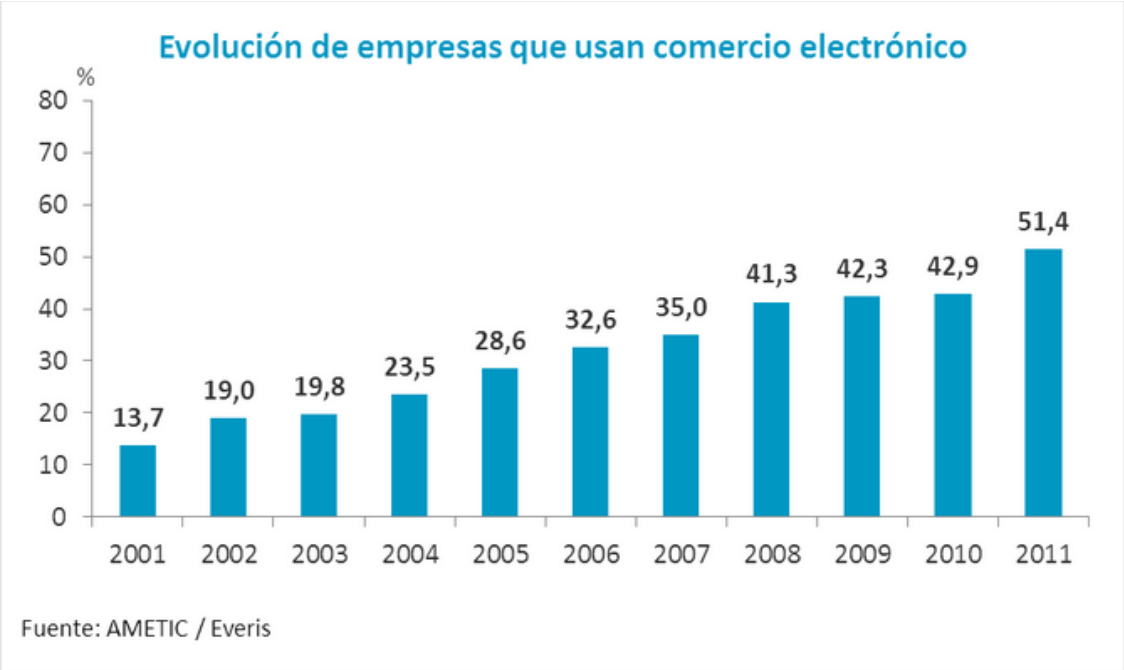


Ilustración 4: Evolución de empresas que usan el comercio electrónico Fuente [39].

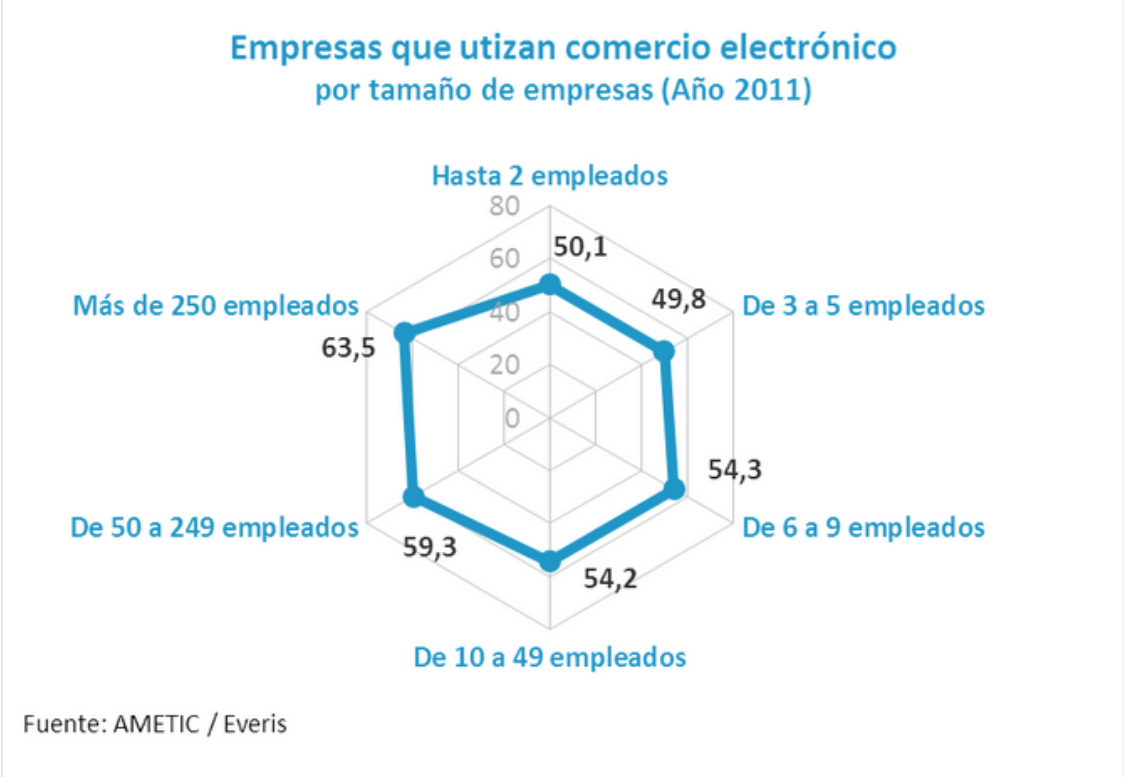


Ilustración 5: Empresas que utilizan comercio electrónico por tamaño. Fuente [39].

1.2. Competidores

En la actualidad existen diversas opciones para crear y personalizar una tienda online sin tener que desarrollar un proyecto propio para la empresa. Un ejemplo es Shopify [4], esta plataforma permite crear una tienda online en pocos pasos y totalmente personalizable mediante plantillas. Sus planes de precios van desde los 14$/mes hasta los 179$/mes. Además, Shopify proporciona herramientas cómo la integración con Facebook, cupones de descuento y herramientas de marketing para las tiendas creadas con la plataforma. Existen una gran variedad de productos y plataformas qué cubren la creación de una tienda online, por citar algunas: Tiendy.com [5], Webnode [6], Urbecom [7], etc.

Además de las opciones comentadas, la plataforma con la que YouMarket tiene más parecido y comparte en gran parte la filosofía es StoreEnvy [8]. Se trata de una plataforma bastante desconocida en España y se basa en la idea de unir las tiendas con usuarios mediante valoraciones, colecciones e interacciones sociales. Los usuarios pueden seguir la actividad de las tiendas y la actividad de otros usuarios con gustos similares. Además, cualquier persona o empresa puede crear su tienda en StoreEnvy en pocos minutos y empezar a vender sus productos. El único coste para el vendedor es un pequeño porcentaje por cada venta realizada o por disponer de varias opciones "premium" para ganar visibilidad dentro de la plataforma.

1.3. Motivación

El proyecto YouMarket surge principalmente por tres motivos:

1. La necesidad de crear una plataforma de e-commerce de proximidad para proporcionar un nuevo canal de venta a los pequeños comerciantes de una zona geográfica en particular. La idea tras el proyecto es proporcionar a los pequeños comercios una herramienta para ofrecer sus productos en un canal de venta directa en Internet con un coste lo más bajo posible.
2. Otro factor importante que me ha llevado a desarrollar este proyecto es mi

curiosidad por aprender un framework PHP moderno como Symfony2 y aplicarlo a un proyecto lo más real posible.

3. Por último y no menos importante, es el hecho de querer aplicar mis conocimientos sobre desarrollo de aplicaciones Web a un entorno más grande de lo que he podido aplicar hasta la fecha y además sembrar la primera semilla de un proyecto real a largo plazo.

1.4. Objetivos

Creación de una plataforma de e-commerce para potenciar el comercio de proximidad. Cualquier comercio de la zona geográfica delimitada por YouMarket podrá crear y mantener un catálogo de productos. Cada comercio dispondrá de una sección propia (a modo de perfil de comercio) con su catálogo de productos disponibles; además, sus productos aparecerán catalogados dentro de las categorías de la plataforma.

El comercio podrá gestionar su perfil añadiendo una descripción, la ubicación física, una foto de perfil y enlaces externos a redes sociales donde tenga presencia. El usuario final, el consumidor, podrá adquirir los productos mediante una plataforma de pago integrada en YouMarket (queda pendiente la plataforma final de pago, se está explorando la posibilidad de integrar PayPal [9] junto con otras opciones más tradicionales de pago online como es el sistema CyberPac [10]). El pago se realiza a YouMarket, quien será el encargado de pagar al proveedor del artículo una vez se realice la entrega del producto comprado por el usuario final (consumidor).

Además, el usuario final (consumidor) podrá interactuar con los productos, perfiles de comercios y de otros usuarios generando comentarios, valoraciones y recomendaciones tanto dentro de YouMarket como en las redes sociales.

Por último, cabe destacar el objetivo de la creación de una plataforma base para que pueda adaptarse a múltiples usos; es decir, se trata de obtener una base para poder

orientar la plataforma a distintos mercados y tipos de productos. Por ejemplo, se podría adaptar YouMarket fácilmente en una plataforma de venta de frutas y verduras para una cooperativa o unión de agricultores.

1.5. Organización de la memoria

En esta memoria se incluyen al detalle los siguientes puntos del proyecto:

- Capítulo 2. Aquí se detalla el análisis de requisitos de la plataforma.
- Capítulo 3. Se detalla el modelo de datos.
- Capítulo 4. Se realiza un resumen de todas las tecnologías en las que se basa el proyecto YouMarket.
- Capítulo 5. Se resumen las características más relevantes de Symfony2.
- Capítulo 6. Se detalla cómo está construido YouMarket.
- Capítulo 7. Se explica la planificación del proyecto.
- Capítulo 8. En este capitulo se explica las diferentes pruebas realizadas sobre el trabajo desarrollado.
- Capítulo 9. Se detallan las opciones y técnicas para escalar YouMarket ante grandes cantidades de tráfico y usuarios.
- Capítulo 10. Se detallan las conclusiones del trabajo realizado además de proponer mejoras y extensiones de futuro.

2. Análisis de requerimientos

2.1. Análisis de requerimientos y funcionalidades

Para el desarrollo del proyecto se analizó previamente una serie de requisitos que debía cumplir y unas funcionalidades que debía aportar. Muchas de las funcionalidades y requisitos se han ido construyendo a medida que avanzaba el proyecto.

El requisito fundamental de la plataforma es poder mantener un catálogo de productos de diferentes comercios y que los usuarios puedan adquirirlos. Todo debe hacerse de la forma más intuitiva y cómoda posible, tanto desde el lado del comprador como para los vendedores de los productos.

2.2. Requisitos funcionales

Se han separado en 4 grandes grupos los requisitos de la plataforma en función del tipo de usuario que se van a necesitar:

- Usuario anónimo: Usuario sin identificar dentro de la plataforma.
- Usuario registrado: Usuario identificado como cliente de YouMarket (comprador).
- Usuario tienda: Usuario identificado como comercio de YouMarket (vendedor).
- Usuario administrador.

A continuación se detallan los casos de uso organizados por tipo de usuario. En esta sección se incluye una breve descripción por cada uno. En el anexo del presente documento se puede consultar con más detalle cada uno de los casos de uso descritos a continuación.

2.2.1. Casos de uso para el usuario anónimo

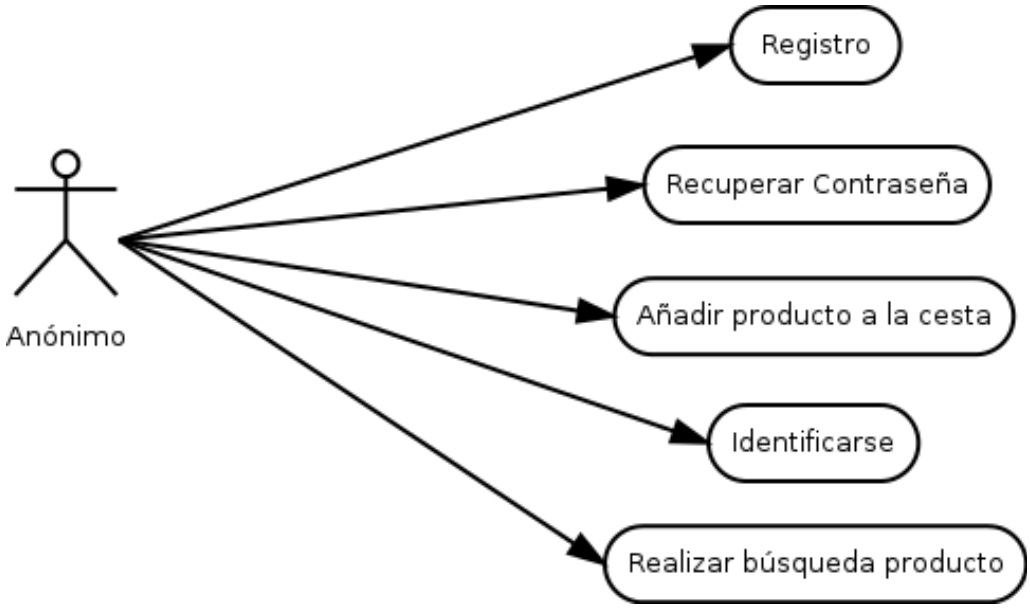


Ilustración 6: Diagrama casos de uso usuario anónimo

Este tipo de usuario puede registrarse, identificarse y navegar por el catálogo de productos y añadir productos a la cesta de la compra. La ilustración 6 muestra los casos de uso disponibles para este usuario.

2.2.2. Casos de uso para el usuario registrado

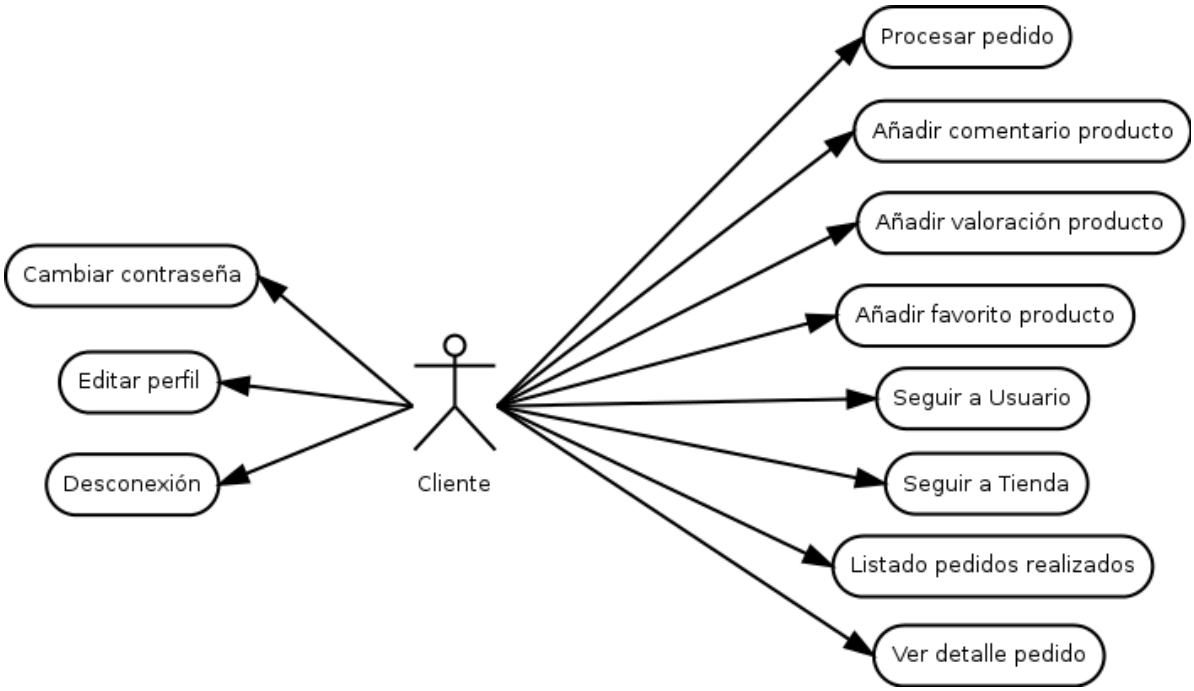


Ilustración 7: Diagrama casos de uso del usuario registrado

El usuario registrado puede realizar las siguientes acciones sobre su cuenta:

- Editar los datos de su perfil.
- Cambiar la contraseña de su cuenta.
- Ver los pedidos realizados.
- Cerrar sesión (Desconexión).

El resto de acciones disponibles para este tipo de usuario son las relacionadas con la compra de productos y de interacción con el resto de usuarios de la plataforma. La ilustración 7 muestra los casos de uso disponibles para este tipo de usuario.

2.2.3. Casos de uso para el usuario tienda

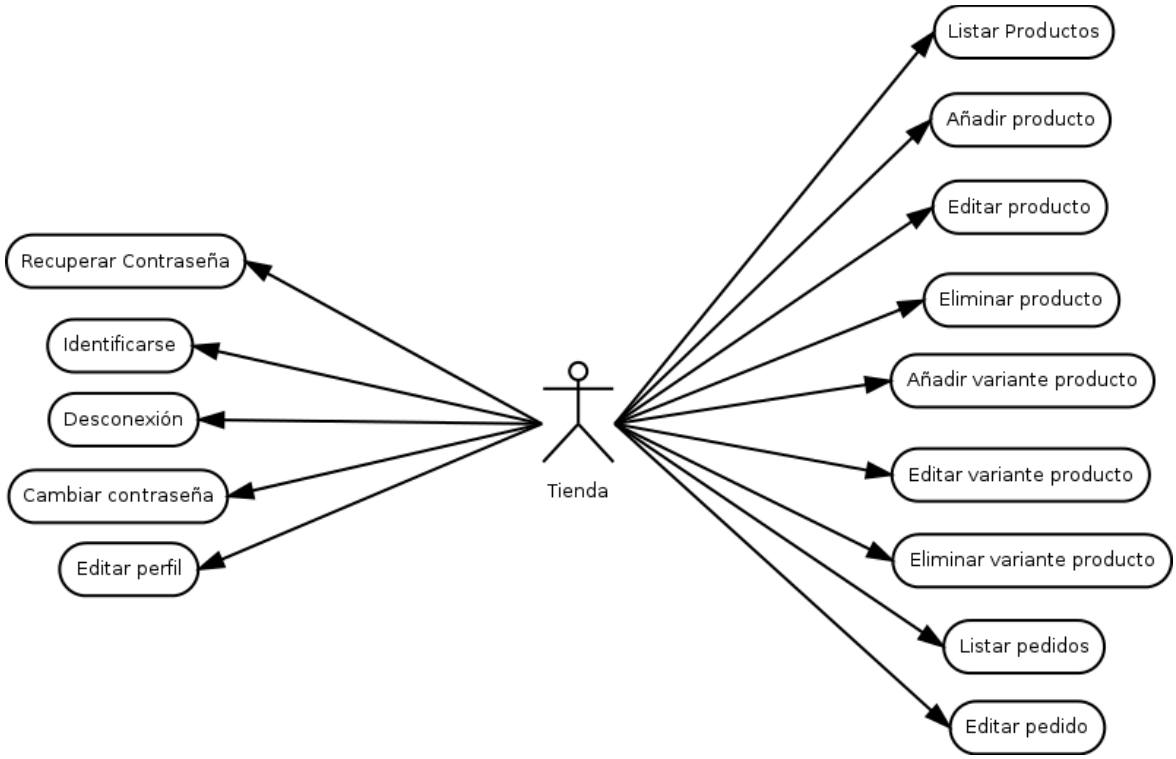


Ilustración 8: Diagrama casos de uso del usuario tipo tienda

Del mismo modo que el usuario registrado, este usuario dispone de unas acciones relacionadas con su cuenta en particular, además de poder gestionar su catálogo de productos y los pedidos que los compradores realizan sobre ellos. La ilustración 8 muestra los casos de uso disponibles para este tipo de usuario.

2.2.4. Casos de uso para el usuario administrador

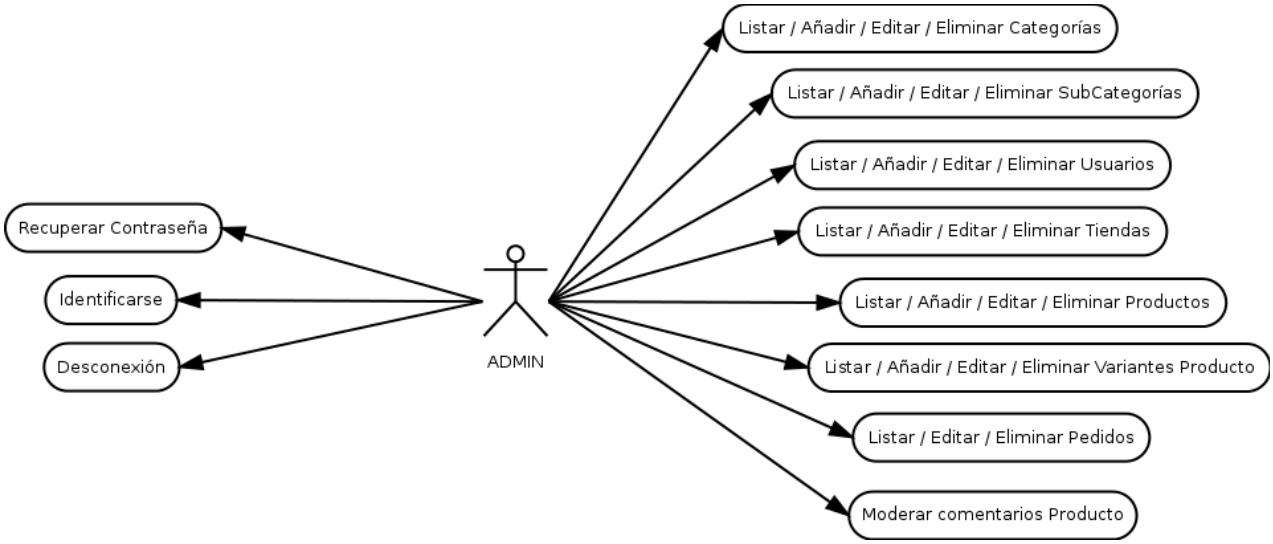


Ilustración 9: Diagrama casos de uso del usuario administrador

El usuario administrador dispone de las acciones tradicionales de listado, edición y eliminación sobre los elementos de la aplicación. La ilustración 9 muestra los casos de uso disponibles para este usuario.

2.3. Requisitos no funcionales

- **Escalabilidad**: Se debe obtener una gran escalabilidad de la plataforma.
- **Accesible**: Debe ser accesible desde diferentes dispositivos y distintos navegadores web.
  - No es necesario dar soporte a versiones anteriores de Internet Explorer 9.
  - Usar un diseño Responsive [11] para evitar crear diferentes versiones específicas para cada dispositivo de navegación.
- **Velocidad:** La plataforma debe ser rápida en cargar, especialmente cuando el dispositivo de navegación es una tablet o smartphone [12].

3. Diseño del modelo de datos

3.1. Modelo Entidad / Relación (ER)

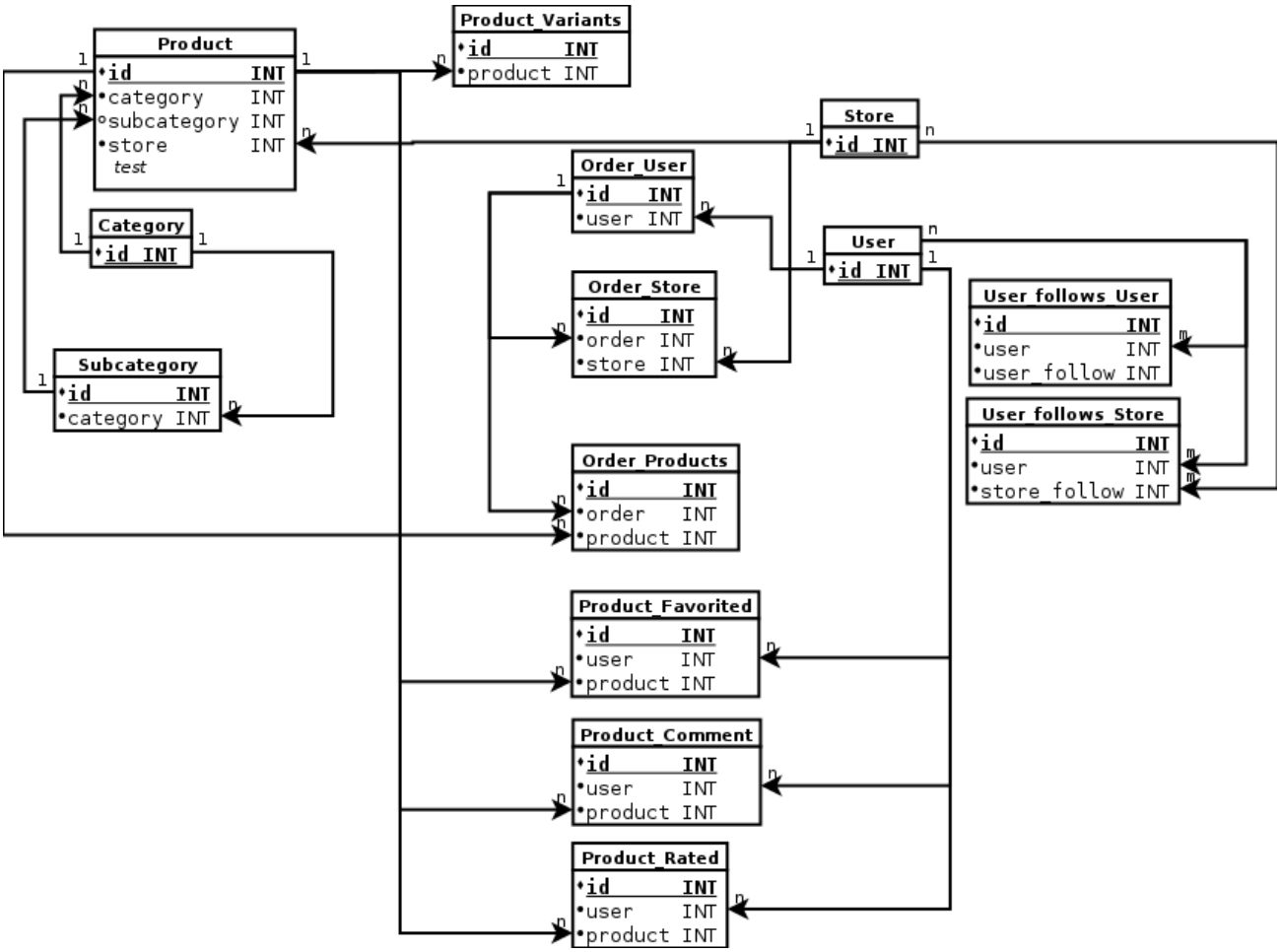


Ilustración 10: Modelo Entidad / Relación

En la ilustración 10 se presentan las relaciones entre las diferentes tablas del modelo de datos.

3.2. Modelo de datos

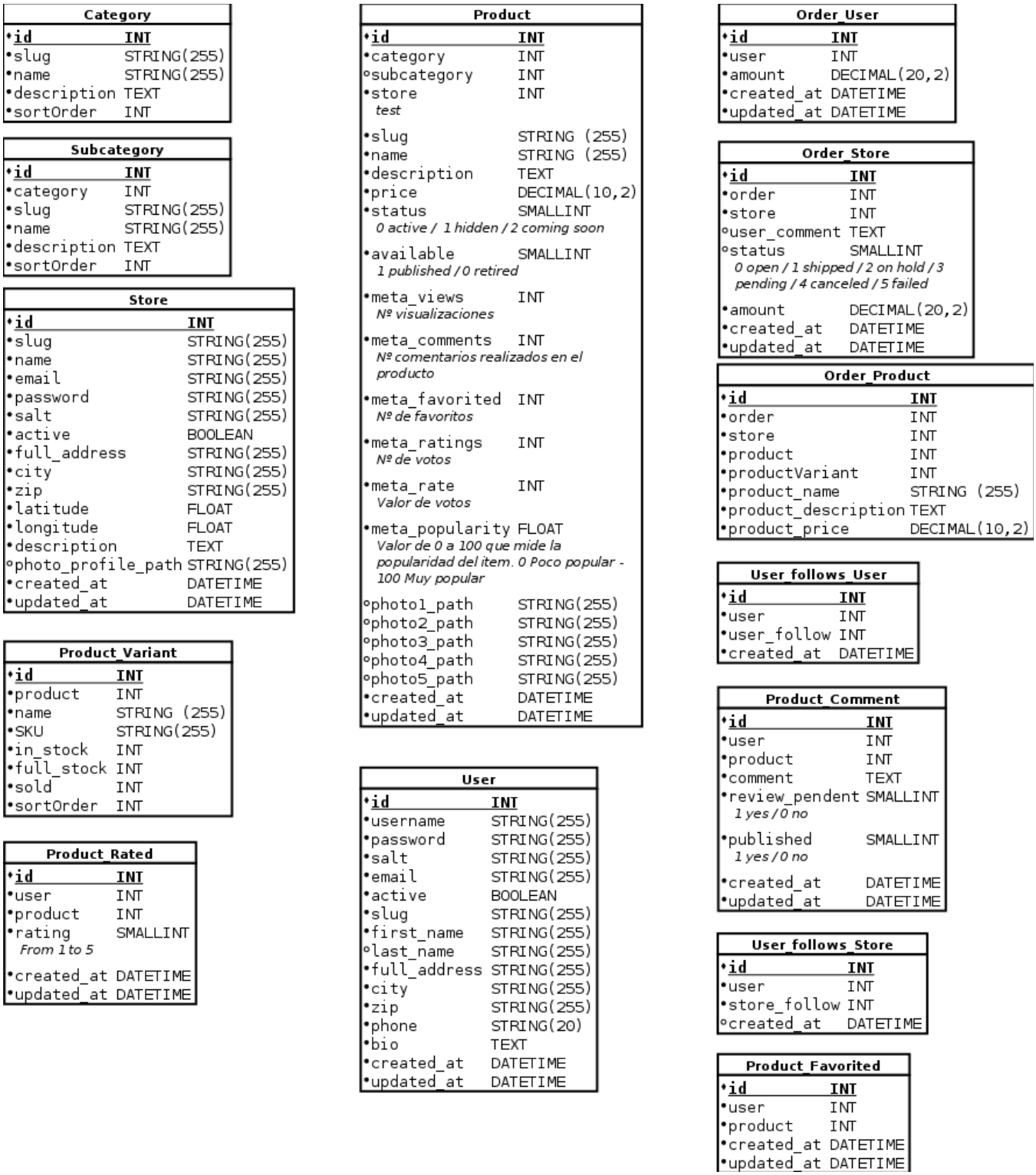


Ilustración 11: Definición del modelo de datos

En la ilustración 11 se presenta la definición de todos los atributos de cada tabla del modelo de datos de YouMarket.

4. Tecnología usada

4.1. Servidor LAMP – Linux / Apache / MySQL / PHP

Se ha desarrollado YouMarket bajo una arquitectura LAMP (Linux, Apache, MySQL y PHP). Para el desarrollo, se ha usado la distribución de Ubuntu Server 12.04.4 LTS [13] publicada en Abril de 2012 y con soporte hasta Abril de 2017. La elección de una distribución Long Term Support (LTS) es fundamental para obtener un soporte a largo plazo y asegurar así un amplio ciclo de vida de los servicios de los que depende la aplicación.

Servicios básicos instalados:

- Apache: versión 2.2.22
- MySQL: versión 5.5.35
- PHP: versión 5.3

En el anexo se encuentran disponibles las configuraciones detalladas de los servicios comentados anteriormente.

Para implementar el servidor de desarrollo se ha usado una máquina virtual de VMWare [14]; esto ha permitido poder replicar exactamente un hipotético escenario real completamente aislado del sistema operativo de usuario. La comunicación con los archivos del servidor se ha realizado gracias a Samba [15], montando una unidad de red de la carpeta del proyecto del servidor virtualizado en la máquina de desarrollo huésped.

4.2. Composer

Composer [16] es una herramienta para gestionar las dependencias en PHP. Permite al desarrollador definir las librerías usadas por la aplicación y asegurar que otros desarrolladores de la aplicación cumplan esas dependencias de forma precisa. Composer no maneja ni gestiona las librerías instaladas globalmente en un sistema operativo, los

gestiona a nivel de proyecto.

El problema que Composer soluciona es:

- Tienes un proyecto que depende de un numero de librerías.
- Algunas de estas librerías dependen de otras librerías.
- Tú eres quien define los elementos de los que dependes.
- Composer encuentra qué versión de los paquetes necesitan ser instalados y los instala (se los descarga y los deja en el proyecto).

4.3. Framework PHP – Symfony2

Symfony2 es la piedra angular del proyecto, es un web framework PHP basado en componentes (librerías) que permite crear aplicaciones y sitios web rápidos y seguros de forma profesional. Actualmente ningún otro framework PHP dispone de todas las características soportadas por Symfony2 al mismo tiempo; dichas características se pueden resumir en:

- Testing: Integra herramientas de test automatizadas por defecto.
- Debugging: Proporciona una potente herramienta de debug para analizar qué ocurre en cada petición y las consultas realizadas a la base de datos de la aplicación.
- Publicación a producción: Se pueden integrar de forma sencilla aplicaciones externas para facilitar el despliegue de las aplicaciones al entorno de producción.
- Separación del código: El código de Symfony2 es completamente desacoplable, se pueden integrar partes de aplicación en otros proyectos rápidamente.
- Soporte de un potente ORM (Object Relational Mapper) como es Doctrine [17]

Symfony2 se explica en más detalle en el capítulo **5. Symfony2 en detalle.**

4.4. HTML & CSS Framework Bootstrap

En la actualidad se interactúa con la Web desde multitud de dispositivos, con diferentes características, distintos tamaños de pantalla y un largo etcétera de variables a tener en

cuenta si el código HTML y CSS se escribe desde cero. Debido a este aumento de variables a considerar por los programadores web, se empezaron a hacer populares y extenderse los llamados HTML & CSS Frameworks. Estos frameworks están construidos con archivos HTML, CSS y javascript, que tienen unos claros objetivos:

- Proporcionar una base para contenido fluido (Responsive Design)
- Adaptar los contenidos a todos los tamaños de pantalla de los dispositivos de navegación (Responsive Design)
- Detección de la navegación desde un dispositivo móvil, para adaptar el sistema de navegación de menús para facilitar su uso con la pantalla de un móvil o tableta.
- Solucionar problemas de renderizado de los diferentes navegadores web.
- Facilitar la representación de formularios accesibles desde cualquier dispositivo.
- Reducir costes en la creación de una página web, al aplicar soluciones genéricas.
- Aumentar la velocidad de creación de páginas web.
- Facilidad de aplicar las soluciones de futuros problemas y en cambios en la especificación de xHTML o CSS.

Existen actualmente una amplia variedad de este tipo de frameworks, por ejemplo Foundation [18] o Bootstrap [19]. La elección de uno u otro dependerá de las necesidades del proyecto, el apoyo que recibe por parte de la comunidad el framework o de la velocidad del framework en solucionar bugs e implementar mejoras.

La base del frontend de YouMarket está construida con Boostrap, que ha facilitado enormemente el desarrollo. Se ha conseguido una navegación adaptable a dispositivos móviles desde el inicio sin tener que pensar en una futura revisión para cumplir con este requisito imprescindible.

4.5. Gestor de código – Github

Se ha usado Github para llevar a cabo dos tareas bien diferenciadas. Por un lado, de forma interna del proyecto para mantener una revisión del código fuente que se ha sido

construido paso a paso.

Por otro lado, Symfony2 (y sus dependencias) se instala y actualiza de forma muy sencilla por GitHub vía Composer. Todos los componentes externos de Symfony2 pueden ser descargados por vía de Composer, que a su vez hace uso de GitHub como fuente para descargar las dependencias.

5. Symfony2 en detalle

5.1. Definiendo Symfony2

Symfony2 es un web framework PHP basado en componentes (librerías). Para ser más exactos, Symfony2 está construido sobre un conjunto de componentes PHP reutilizables, independientes y desacoplados entre sí, pero que en conjunto crean una cohesión que resuelve los problemas comunes del desarrollo web.

Dependiendo del proyecto y necesidades se puede elegir sólo algunos de los componentes Symfony2 o todos (full-stack). El desarrollador tiene la libertad de elegir la forma de atacar el problema que desea resolver.

Los componentes Symfony2 son como las tradicionales librerías de cualquier proyecto o desarrollo, en concreto, en Symfony2 hay 21 disponibles de forma directa:

BrowserKit	EventDispatcher	Routing
ClassLoader	Finder	Security
Config	Form	Serializer
Console	HttpFoundation	Templating
CssSelector	HttpKernel	Translation
DependencyInjector	Locale	Validator
DomCrawler	Process	Yaml

Además, incorpora de serie otros componentes externos:

Assetic	Doctrine	Twig
Monolog	SwiftMailer	

Todos los componentes deben integrarse de alguna manera dentro de Symfony2; para ello, se hace uso de los “bundles” que integran un componente con el resto del framework y de la aplicación. Symfony2 incorpora todos los bundles necesarios para integrar los componentes anteriores. Puede parece un poco confusa esta forma de organizar el código, pero incluso el código de las aplicaciones desarrolladas con

Symfony2 deben ir en uno o más bundles dentro del framework. De esta forma, se mantiene la filosofía de código desacoplabe y reutilizable.

5.2. Symfony2, un framework HTTP

Según su creador, Fabien Potencier, Symfony2 es un framework HTTP al exponer lo siguiente: "*I don't like MVC because that's not how the web works. Symfony2 is an HTTP framework; it is a Request/Response framework. That's the big deal. The fundamental principles of Symfony2 are centered around the HTTP specification.*" *[20]*

En realidad, la mayoría de los frameworks modernos, tanto para web como para otros entornos, implementan el patrón MVC [21] (Modelo, Vista, Controlador), pero Symfony2 no se encarga del Modelo propiamente, sino que lo delega a un componente externo.

Por defecto, en Symfony2, del modelo se encarga el componente llamado Doctrine [17]. Doctrine es un Object Relational Mapper (ORM) y se encarga de convertir el modelo de datos almacenado en tablas a objetos (y viceversa) dentro del código de la aplicación de forma transparente para el programador.

En resumen, Symfony2 se encarga del Controlador y de la Vista; el Modelo lo delega a otro componente que no depende del núcleo de Symfony2.

Esto no significa que no se pueda desarrollar en Symfony2 bajo el patrón MVC, Symfony2 no interfiere en el paradigma de la arquitectura y es el programador el que debe tener en cuenta este hecho. En la mayoría de bibliografía, ponencias, transparencias y otro material didáctico que ha sido consultado para aprender Symfony2, todos hablan de Symfony2 como un framework MVC, aunque en realidad, el propio creador no lo califica de tal forma.

5.3. Ventajas de Symfony2

Symfony2 quiere solucionar los retos de la programación web actual y futura para que el programador pueda centrarse en el problema de verdad y no tener que reinventar la

rueda a cada paso para solucionar problemas genéricos.

Retos de la programación web:

- persistencia de datos
- seguridad
- formularios
- validación
- motor de plantillas
- archivos de log
- rendimiento
- caché
- archivos web
- internacionalización
- tareas programadas
- enrutamiento

5.3.1. Buenas prácticas en Symfony2:

Symfony2 adopta buenas prácticas de otros frameworks, incluso de otros lenguajes de programación y entornos.

- La seguridad en Symfony2 está basada en los mismos principios que la seguridad de Spring (Java).

```
<http>
  <intercept-url pattern='/login.htm*' filters='none'/>
  <intercept-url pattern='/**' access='ROLE_USER' />
  <form-login login-page='/login.htm'
              default-target-url='/home.htm'
              always-use-default-target='true' />
</http>
```

spring

```
firewalls:
  login:
    pattern:    ^/login
    anonymous:  ~
  all:
    pattern:    ^/
    form_login:
      login_path:                   /login
      default_target_path:          /home.htm
      always_use_default_target_path: true
```

Ilustración 12: Configuración de seguridad, Spring vs Symfony2

- La forma en la que se trabaja con el modelo de datos está basada en Hibernate, el ORM de Java.

Ilustración 13: Manipulando el modelo, Hibernate vs Symfony2

- La consola está basada en la de Ruby on Rails.

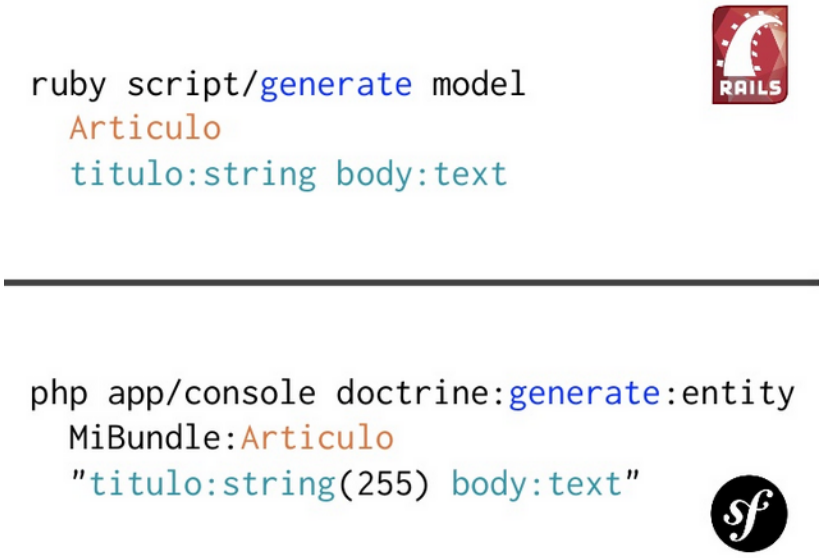


Ilustración 14: Consola de comandos, Ruby vs Symfony2

- El motor de plantillas Twig que incorpora Symfony2 está basado en el motor de plantillas de Django de Python.

```
{% extends "base_generic.html" %}
{% block title %} {{ seccion.titulo }} {% endblock %}
{% block content %}
<h1> {{ seccion.titulo }} </h1>

{% for articulo in articulos %}
  <h2> {{ articulo.titulo|upper }} </h2>
{% endfor %}
{% endblock %}
```

django

```
{% extends "::base_generic.html" %}
{% block title %} {{ seccion.titulo }} {% endblock %}
{% block content %}
<h1> {{ seccion.titulo }} </h1>

{% for articulo in articulos %}
  <h2> {{ articulo.titulo|upper }} </h2>
{% endfor %}
{% endblock %}
```

Ilustración 15: Código de la vista, Django vs Symfony2

5.3.2. Flexibilidad

Se pueden usar archivos de configuración en YAML, XML o PHP.

Para el motor de plantillas es posible usar Twig o directamente PHP.

Para el almacenamiento del modelo se puede utilizar SQL (MySQL, Oracle, SQLServer...) o NoSQL (MongoDB...).

5.3.3. Rendimiento

Todo en Symfony2 se transforma a PHP, es decir, los archivos de configuración en YAML o XML, las plantillas Twig, las anotaciones en el código, todo, se transforma a PHP y se almacena el bytecode en la caché alternativa de PHP (APC [22], por sus siglas en inglés, Alternative PHP Cache).

Por defecto, además, Symfony2 incorpora un proxy inverso escrito en PHP para evitar tener que acceder a la base de datos en cada nueva petición. El proxy se encarga de cachear por un tiempo la respuesta de un usuario y servirla a distintos usuarios si hay que servir los mismos datos. Este proxy por defecto está bien para la fase de desarrollo y las pruebas de integración. En producción, lo recomendable es delegar

este trabajo a proxy's inversos más avanzados como es Varnish [23] o Squid [24].

Symfony2 está preparado para el uso de estos tipos de proxy's avanzados y es relativamente sencillo sustituir el proxy por defecto de Symfony2 por uno más profesional y robusto.

5.4. Componentes de Symfony2

5.4.1. Routing

El componente de Routing es el encargado de asociar una petición (Request) con el código de la aplicación para generar una respuesta (Response).

```
use Symfony\Component\HttpFoundation\Request;
use Symfony\Component\Routing\Matcher\UrlMatcher;
use Symfony\Component\Routing\RequestContext;
use Symfony\Component\Routing\RouteCollection;
use Symfony\Component\Routing\Route;

$routes = new RouteCollection();
$routes->add('hello', new Route('/hello', array('controller' => 'foo')));

$context = new RequestContext();

// this is optional and can be done without a Request instance
$context->fromRequest(Request::createFromGlobals());

$matcher = new UrlMatcher($routes, $context);

$parameters = $matcher->match('/hello');
```

Ilustración 16: Ejemplo del componente Routing.

5.4.2. ClassLoader

El componente ClassLoader proporciona un autocargador de espacios de nombres (disponible desde PHP 5.3). Realmente es muy flexible y permite cargar clases de diferentes directorios basados por sub-espacios.

```php
require_once __DIR__.'/src/Symfony/Component/ClassLoader/UniversalClassLoader.php';

use Symfony\Component\ClassLoader\UniversalClassLoader;

$loader = new UniversalClassLoader();
$loader->registerNamespaces(array(
    'Symfony'          => array(__DIR__.'/src', __DIR__.'/symfony/src'),
    'Doctrine\\Common' => __DIR__.'/vendor/doctrine-common/lib',
    'Doctrine\\DBAL'   => __DIR__.'/vendor/doctrine-dbal/lib',
    'Doctrine'         => __DIR__.'/vendor/doctrine/lib',
    'Monolog'          => __DIR__.'/vendor/monolog/src',
));
$loader->registerPrefixes(array(
    'Twig_' => __DIR__.'/vendor/twig/lib',
));
$loader->register();
```

Ilustración 17: Ejemplo del componente ClassLoader

5.4.3. Console

Aunque Symfony2 sea un web framework, es necesaria una consola para poder generar comandos: limpiar la caché, actualizar la estructura de la base de datos, generar comprobaciones de las rutas, etc. El componente Console permite crear aplicaciones basadas en consola con muy pocas líneas de código.

```php
use Symfony\Component\Console\Application;
use Symfony\Component\Console\Input\InputInterface;
use Symfony\Component\Console\Input\InputArgument;
use Symfony\Component\Console\Input\InputOption;
use Symfony\Component\Console\Output\OutputInterface;

$console = new Application();
$console
    ->register('ls')
    ->setDefinition(array(
        new InputArgument('dir', InputArgument::REQUIRED, 'Directory name'),
    ))
    ->setDescription('Displays the files in the given directory')
    ->setCode(function (InputInterface $input, OutputInterface $output) {
        $dir = $input->getArgument('dir');

        $output->writeln(sprintf('Dir listing for <info>%s</info>', $dir));
    })
;
$console->run();
```

Ilustración 18: Ejemplo del componente Console.

5.4.4. YAML

El formato YAML es fantástico para generar y mantener archivos de configuración. El componente YAML es el más popular de Symfony2, debido a que es la única librería PHP que implementa la especificación 1.2 de YAML.

```
use Symfony\Component\Yaml\Yaml;

$array = Yaml::parse($file);

print Yaml::dump($array);
```

Ilustración 19: Ejemplo del componente YAML.

5.4.5. Finder

El componente Finder nos permite hacer búsquedas de archivos en el sistema de ficheros basándonos en un sistema de consultas (Query's). Además, es capaz de conectarse con sistemas de archivos remotos, como por ejemplo el S3 de Amazon.

```
use Symfony\Component\Finder\Finder;

$finder = new Finder();

$iterator = $finder
  ->files()
  ->name('*.php')
  ->depth(0)
  ->size('>= 1K')
  ->in(__DIR__);

foreach ($iterator as $file) {
    print $file->getRealpath()."\n";
}
```

Ilustración 20: Ejemplo del componente Finder.

```php
$s3 = new \Zend_Service_Amazon_S3($key, $secret);
$s3->registerStreamWrapper("s3");

$finder = new Finder();
$finder->name('photos*')->size('< 100K')->date('since 1 hour ago');
foreach ($finder->in('s3://bucket-name') as $file) {
    print $file->getFilename()."\n";
}
```

Ilustración 21: Ejemplo del componente Finder sobre S3 de Amazon.

5.4.6. Process

Este componente nos permite ejecutar comandos en un sub-proceso, es decir, en segundo plano. Además, soporta callbacks para notificaciones en tiempo real. Por ejemplo, listar un directorio y devolver el resultado es tan fácil como:

```php
use Symfony\Component\Process\Process;

$process = new Process('ls -lsa');
$process->setTimeout(3600);
$process->run();
if (!$process->isSuccessful()) {
    throw new RuntimeException($process->getErrorOutput());
}

print $process->getOutput();
```

Ilustración 22: Ejemplo del componente Process.

5.4.7. HttpFoundation

El componente HttpFoundation añade una capa orientada a objetos por encima de PHP para todo lo relacionado con la Web: Request (Petición), Response (Respuesta), subida de archivos, cookies y sesiones de usuario.

5.4.8. DependencyInjection

El componente DependencyInjection permite centralizar y estandarizar la forma en la que se crean los objetos en la aplicación.

5.4.9. HttpKernel

El componente HttpKernel proporciona un proceso estructurado para convertir una Request (Petición) a una Response (Respuesta) haciendo uso del EventDispatcher.

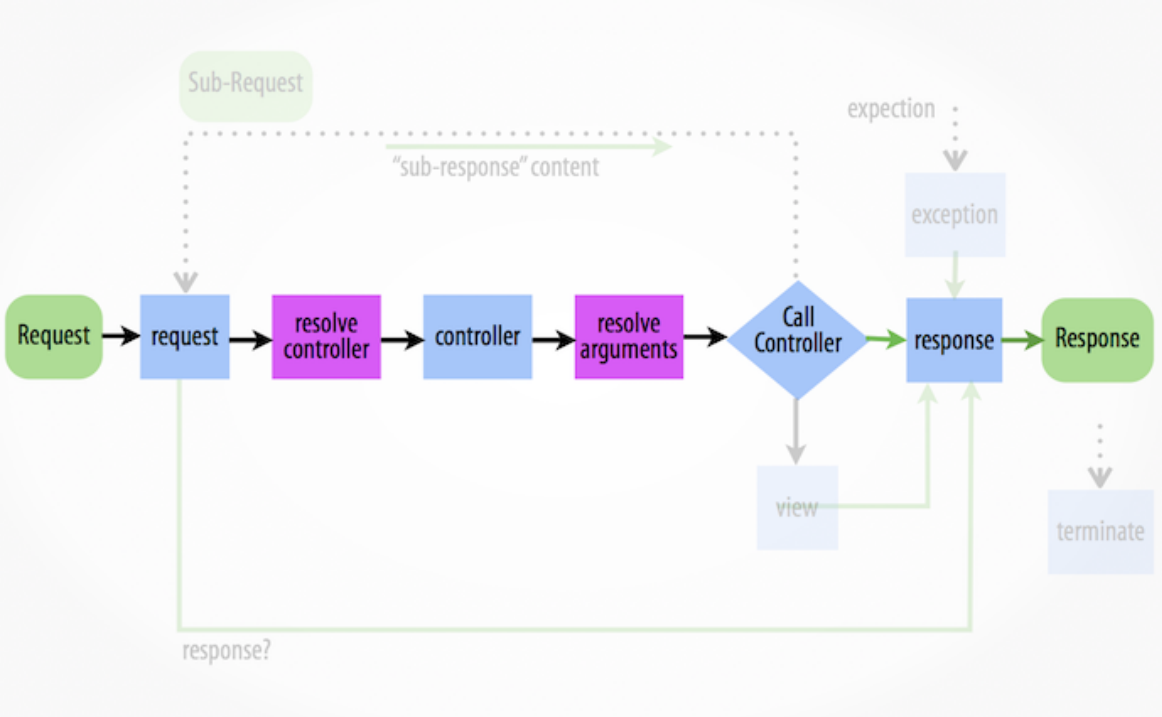


Ilustración 23: Proceso Request -> Response completo

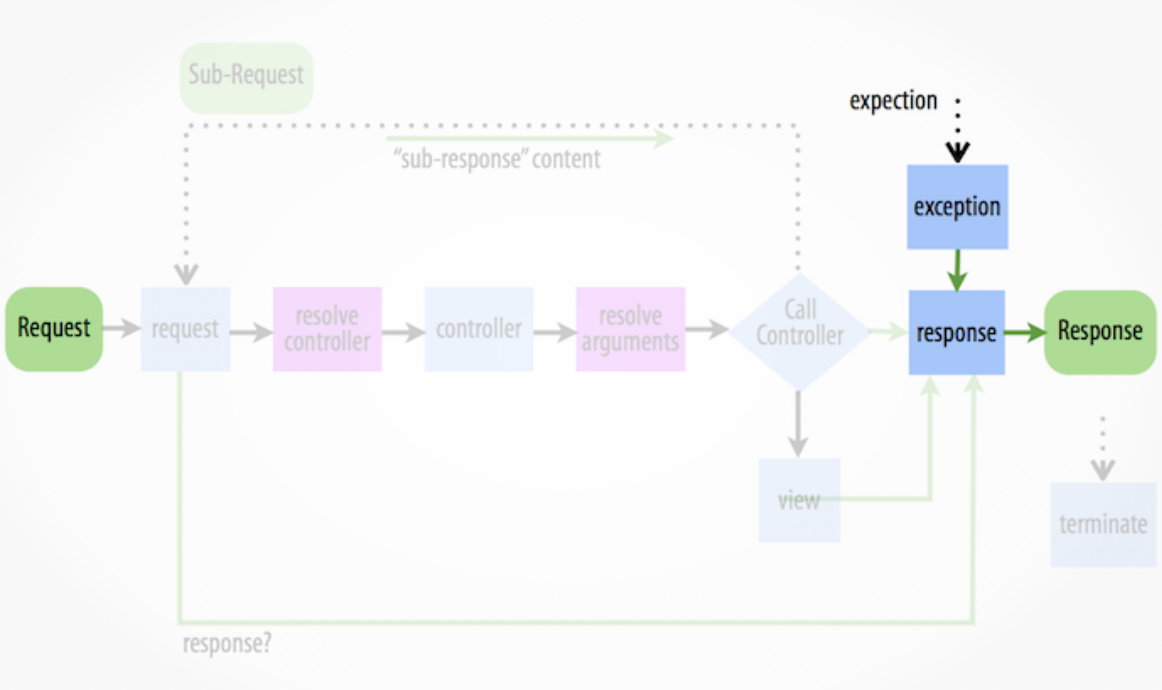


Ilustración 24: Ejemplo del componente HttpKernel y el EventDispacher "capturando" una excepción.

5.4.10. Motor de plantillas Twig

Twig no es un componente interno de Symfony2, es externo, aunque en la instalación normal de Symfony2 viene incluido por defecto.

Una plantilla es simplemente un archivo de texto que puede generar cualquier formato basado en texto (HTML, XML, CSV, RSS...)

El formato más común de plantillas en PHP es un archivo de texto que contiene una mezcla de código HTML y PHP (Ilustración 25).

```
8
9▼      <article id="post-<?php the_ID(); ?>" <?php post_class(); ?>>
10▼         <div class="gallery-articles row">
11
12▼             <div class="twelve columns">
13
14                  <div class="entry-content">
15                      <?php the_content(__('Read more...', 'leaf')); ?>
16                  </div><!-- .entry-summary -->
17
18▼                 <header class="entry-header">
19                      <?php if( get_the_title() != "Untitled") { ?>
20                      <h1 class="entry-title">
21                          <a href="<?php the_permalink(); ?>" title="<?php echo esc_att
22                      </h1>
23                      <?php }else{ ?>
24                      <h1 class="entry-title">
25                          <a href="<?php the_permalink(); ?>" title="" rel="bookmark">S
26                      </h1>
27                      <?php } ?>
28                  </header><!-- .entry-header -->
29
30              </div><!-- .twelve .columns -->
31
32          </div><!-- .gallery-articles .row -->
```

Ilustración 25: Extracto de código de plantilla de Wordpress

Pero Symfony2 incorpora un motor de plantillas mucho más poderoso llamado Twig. Twig permite escribir de forma concisa, entendible y leíble las plantillas que son fácilmente amigables con los diseñadores y además mucho más potentes que una plantilla tradicional en PHP.

```
<section class="user-header">
    <div class="container">
        <div class="row-fluid">
            <div class="span2">
                <div class="user-profile-image">
                    {% if perfil.getWebPathPhotoProfile == null %}
                    <img src="{{ (assets_path ~ '/images/nologo_user.jpg')
                    | imagine_filter('avatar_user_thumb') }}" class="img-
                    circle" alt="" />
                    {% else %}
                    <img src="{{ perfil.getWebPathPhotoProfile |
                    imagine_filter('avatar_user_thumb') }} " class="img-
                    circle" alt="" />
                    {% endif %}
                </div>
            </div>
            <div class="span10">
                <div class="user-profile-name">
                    <h1>{{ perfil.firstname ~ ' ' ~ perfil.lastname }}</h1>

                    <a href="#" class="btn btn-primary btn-small" data-
                    toggle="modal">
                        Seguir
                    </a>
                </div>
            </div>
        </div>
    </div>
</section>
```

Ilustración 26: Código de plantilla Twig de YouMarket

En la ilustración 26 se muestra un extracto del código de la plantilla de la cabecera de YouMarket donde se muestra información del usuario en su perfil público.

5.4.11. Ventajas del motor de plantillas Twig

El motor de plantillas Twig ha sido concebido para ser simple y no procesa ninguna etiqueta PHP.

Twig además puede hacer cosas que PHP no puede: controlar los espacios en blanco, escapar el contenido, incluir funciones y filtros propios que sólo afectan a las plantillas. Twig contiene esas pequeñas características que hacen el trabajo de escribir plantillas un proceso fácil y de mayor precisión.

Algunas características relevantes:

- Filtros: se pueden aplicar filtros a las variables Twig.

- Caché: Incorpora un potente sistema de caché.
- Herencia en plantillas: las plantillas pueden heredar de otras plantillas.
- Escapado de las variables de salida.
- Debugging de plantillas.
- Múltiples formatos de salida.

5.4.12. Assetic

Assetic combina dos grandes conceptos: archivos y filtros.

Los archivos son por ejemplo: CSS, javascripts e imágenes. Los filtros son acciones que pueden ser aplicadas sobre estos archivos antes de ser servidos al navegador. Esto permite una separación entre los archivos almacenados en la aplicación y los archivos presentados al usuario final.

5.4.13. Componente de formularios

El componente de formularios permite fácilmente crear, procesar y reutilizar formularios HTML.

Valor añadido del componente de formularios de Symfony2:

Request Handling: se encarga de manipular la Request en las peticiones (gestión envío del formulario HTML).

Protección a CSRF: protección contra ataques Cross-Site-Request-Forgery [25] (CSRF)

Plantillas: se integra con la capa de plantillas Twig, que permite la reutilización de HTML cuando se muestra el formulario.

Traducción: soporte para la traducción de los mensajes de error, etiquetas de campo y otros textos relacionados con formularios.

Validación: formularios integrados con una librería de validación para generar mensajes de error para los datos enviados por el usuario.

5.5. Doctrine y Symfony2

Doctrine es un ORM [17] (Object Relational Mapping). Symfony2 soporta por defecto Doctrine, pero existen otras soluciones para gestionar el modelo de una aplicación Symfony2.

Doctrine, además de soportar bases de datos relacionales (MySQL por ejemplo), también soporta motores no relacionales como por ejemplo MongoDB.

5.5.1. Definición de ORM

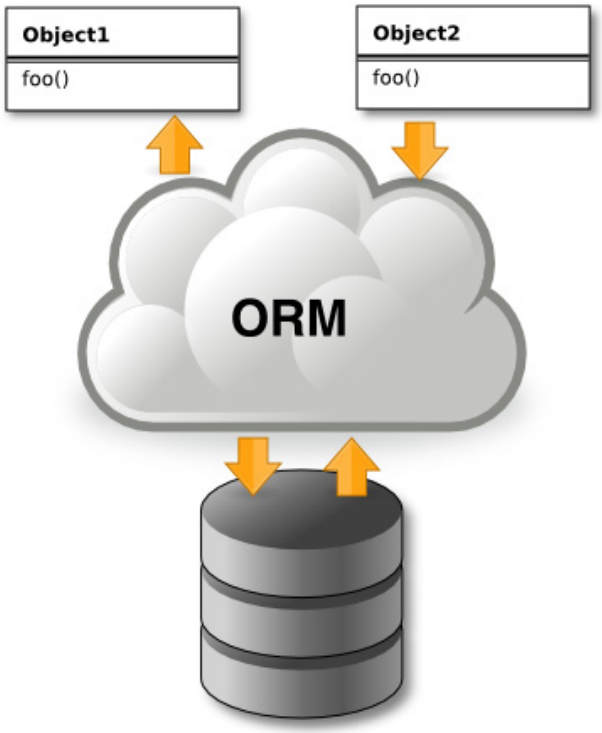


Ilustración 27: Concepto de ORM

En general, se guardan los datos de las aplicaciones en filas dentro de tablas en una base de datos. Pero como programadores, queremos trabajar con clases y objetos dentro de la aplicación y es aquí donde un ORM realiza su trabajo: se encarga de traducir las tablas en objetos y los objetos a tablas de manera transparente para el programador.

Capas de Doctrine:

- Doctrine ORM: puente entre el modelo relacional y los objetos.
- Doctrine DBAL: API de abstracción de la base de datos. En Symfony2, DBAL se encarga de realizar la conexión con la base de datos.

```
51  # Doctrine Configuration
52  doctrine:
53      dbal:
54          driver:   "%database_driver%"
55          host:     "%database_host%"
56          port:     "%database_port%"
57          dbname:   "%database_name%"
58          user:     "%database_user%"
59          password: "%database_password%"
60          charset:  UTF8
```

Ilustración 28: Configuración dbal en Symfony2 (config.yml)

- PDO: API para distintos drivers de base de datos. (No es una capa de Doctrine, pero hace uso de ella.)

5.5.2. Entidades

Cada elemento del modelo, es decir, "las tablas" de la base de datos son generadas por clases PHP llamadas "entidades". Hay una entidad por cada elemento del modelo de datos. El programador debe preocuparse de crear las entidades y Doctrine se encargará de generar la base de datos y las tablas necesarias para unir el modelo lógico que está implementando el desarrollador en la base de datos.

La unión entre la entidad y Doctrine (mapping) se puede realizar por dos vías: una forma de hacerlo es usando anotaciones y la otra es usando un archivo de configuración. La mejor forma de hacerlo es por vía de las anotaciones. Este método se realiza escribiendo anotaciones dentro de la clase que define la entidad. La ilustración 29 muestra la definición de la entidad Category de YouMarket y el mapping

de Doctrine usando anotaciones.

```php
<?php
// src/DCSYS/Bundle/YouMarketBundle/Entity/Category.php
namespace DCSYS\Bundle\YouMarketBundle\Entity;

use Doctrine\ORM\Mapping as ORM;
use DCSYS\Bundle\YouMarketBundle\Util\Util;

/**
 * @ORM\Entity(repositoryClass="DCSYS\Bundle\YouMarketBundle\Entity\Repository\CategoryRepository")
 * @ORM\Table(name="category")
 */
class Category
{
    /**
     * @ORM\Id
     * @ORM\Column(type="integer")
     * @ORM\GeneratedValue(strategy="AUTO")
     */
    protected $id;

    /**
     * @ORM\Column(type="string", length=255, unique=true)
     */
    protected $slug;

    /**
     * @ORM\Column(type="string", length=255)
     */
    protected $name;

    /**
     * @ORM\Column(type="text")
     */
    protected $description;
```


Ilustración 29: Entidad Category de YouMarket

En las anotaciones también se define el nombre de la tabla que usa la entidad, la definición del tipo de dato de cada atributo de la entidad y además las relaciones de integridad con las demás entidades. En la documentación [17] de Doctrine se explica con detalle cada configuración posible.

5.5.3. Obteniendo datos con Doctrine

No es posible realizar directamente consultas contra la base de datos sin antes pasar por el Entity Manager (em). El Entity Manager es un objeto de Doctrine que se encarga de gestionar la conexión con la base de datos y realizar la interacción con las entidades.

```
$em = $this->getDoctrine()->getManager();
$categoria = $em->getRepository('DCSYSYouMarketBundle:Category')->findOneBySlug($category);
```

Ilustración 30: Obteniendo el EntityManager y consultando las categorías

En la ilustración 30 se obtiene el Entity Manager y se realiza una consulta sobre la entidad Category.

Para obtener datos del modelo de la aplicación con Doctrine disponemos de unos métodos generados automáticamente por Doctrine y por cada entidad de la aplicación sin necesidad de generar ningún código por nuestra parte como desarrolladores. Los métodos con los que consultar la base de datos se definen en clases que toman el nombre de "repositorios". Los métodos genéricos por entidad que proporciona Doctrine se encuentran definidos en "Doctrine\ORM\EntityRepository" y por defecto incluyen los siguientes:

- **findAll():** obtiene todos los registros de la tabla. Retorna un array.
- **find():** obtiene un registro a partir de la clave primaria de la tabla.
- **findBy()**: obtiene los registros encontrados pudiendo pasar como argumentos los valores que irían dentro del WHERE.
- **findOneBy()**: obtiene un registro pudiendo pasar como argumentos los valores que irían dentro del WHERE.

Los métodos anteriores son muy simples y en la mayoría de situaciones es necesario hacer consultas más complejas, en estos casos, es necesario crear clases de repositorio propias para cada entidad que necesiten consultas más potentes. Dichos repositorios incluyen consultas SQL que pueden ser escritas usando el Doctrine Query Language [26] (DQL) o en SQL nativo. Básicamente se trata de extender las funciones proporcionadas por el repositorio por defecto de Doctrine para cada entidad.

```php
<?php
// src/DCSYS/Bundle/YouMarketBundle/Entity/Repository/CategoryRepository.php
namespace DCSYS\Bundle\YouMarketBundle\Entity\Repository;

use Doctrine\ORM\EntityRepository;

/**
 * CategoryRepository
 *
 * This class was generated by the Doctrine ORM. Add your own custom
 * repository methods below.
 */
class CategoryRepository extends EntityRepository
{
    public function findAllBySortOrder() {
        return $this->createQueryBuilder('t')
        ->addOrderBy('t.sortOrder', 'ASC')
        ->getQuery()
        ->getResult();
    }
}

```

Ilustración 31: Repositorio de Category

En la ilustración 31 se muestra la definición del repositorio con una consulta escrita en DQL. La asociación entre la clase que define el repositorio y la entidad se realiza vía anotaciones; tomando el ejemplo de la ilustración 29, se realiza dicha asociación en la línea 9 del código de ejemplo.

5.5.4. Data Fixtures

Las Data Fixtures son datos de prueba y test que se usan para rellenar el modelo de datos para realizar los test y ver el comportamiento de la aplicación con datos lo más real posibles. Doctrine incluye un componente para facilitar la creación de estos datos de prueba llamado DoctrineFixturesBundle.

Las fixtures de Doctrine son clases PHP dentro de un bundle de la aplicación. El desarrollador puede escribir tantas fixtures como entidades tiene la aplicación y llenar la base de datos de estos datos de prueba que él mismo ha preparado.

Esta función es realmente útil durante la implementación, pues permite jugar con datos de prueba muy fieles al entorno real.

5.6. Seguridad en Symfony2

La seguridad siempre es la parte más difícil de resolver correctamente, pero el componente de seguridad de Symfony2 se encarga de realizar la mayor parte del trabajo.

El objetivo de la seguridad es evitar que un usuario acceda a un recurso al cual no debería tener acceso. En Symfony2 la seguridad está basada en dos procesos: autenticación y autorización.

La autenticación es el proceso por el cual la aplicación trata de averiguar quién eres. Una vez eres conocido, el siguiente paso es decidir si deberías tener acceso a un determinado recurso.

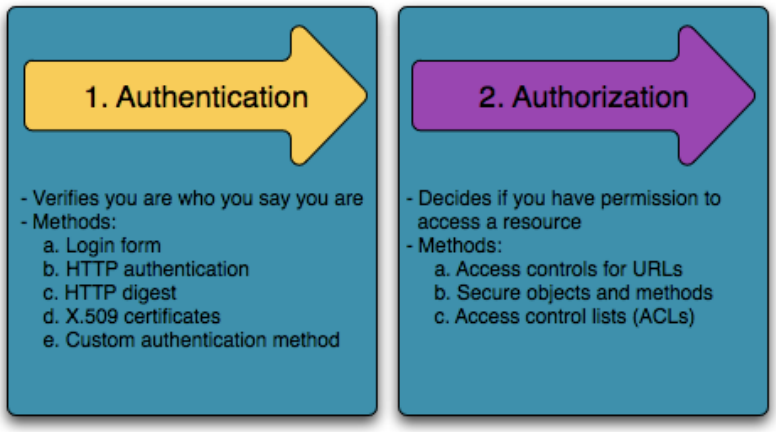


Ilustración 32: Las dos etapas de la seguridad Symfony2

La autenticación, que siempre se aplica en primer lugar, se basa en el uso de un firewall (cortafuegos) cuyo trabajo consiste en determinar la identidad del usuario a través de varios métodos posibles (mediante la autenticación HTTP, un formulario de acceso, etc.)

Una vez que un usuario se autentica, la etapa de autorización determina si el usuario tiene acceso al recurso solicitado. Normalmente se asigna un rol a una URL, clase o método y si el usuario no dispone de ese rol, se le deniega el acceso.

La configuración del sistema de seguridad en Symfony2 se define en el archivo */app/config/security.yml*

5.6.1. Firewalls (autenticación)

El sistema de seguridad de Symfony2 se activa cuando el usuario solicita una URL protegida por un firewall. El trabajo del firewall consiste en determinar si el usuario necesita estar autenticado y en caso de necesitarlo, iniciar el proceso de autenticación.

```
26    firewalls:
27        # Firewall de la extranet a la que se conectan las tiendas
28        admin:
29            pattern:        ^/admin
30            provider:       tiendas
31            anonymous:      ~
32            form_login:
33                login_path: /admin/login
34                check_path: /admin/login_check
35            logout:
36                path:       /admin/logout
37                target:     /admin
38
39        # Firewall del backend (Sonata Admin Bundle)
40        backend:
41            pattern:        ^/backend
42            http_basic:
43                realm: "Acceso Backend"
44                provider: memoria
45
46        # Firewall global utilizado en la parte pública o frontend
47        frontend:
48            pattern:        ^/*
49            provider:       usuarios
50            anonymous:      ~
51            form_login:
52                login_path: dcsys_youmarket_user_login
53                check_path: dcsys_youmarket_user_login_check
54            logout:
55                path:       dcsys_youmarket_user_logout
56
```

Ilustración 33: security.yml - Definición firewalls

Cada firewall se define con una regla de validación (pattern) de actuación del firewall, el proveedor (provider) de los usuarios, si puede ser accesible por usuarios anónimos (no autorizados) y las rutas para iniciar sesión y desconexión (form_login, logout). La ilustración 33 muestra la definición de los firewalls configurados en YouMarket

5.6.2. Control de acceso (Autorización)

Se encarga de verificar si el usuario que ha solicitado la URL tiene los permisos necesarios para obtener el recurso. Si el usuario no cuenta con las credenciales

correctas, el firewall toma el control y actúa redirigiendo al usuario al formulario de login (valores de configuración form_login) o mostrando la caja de autenticación básica de HTTP dependiendo del método de autenticación configurado.

```
62
63    access_control:
64        - { path: ^/usuario/(login|registro|recuperar), roles: IS_AUTHENTICATED_ANONYMOUSLY }
65        - { path: ^/usuario/*, roles: ROLE_USUARIO }
66        - { path: ^/pedido/*, roles: ROLE_USUARIO }
67        - { path: ^/admin/(login|recuperar), roles: IS_AUTHENTICATED_ANONYMOUSLY }
68        - { path: ^/admin/*, roles: ROLE_TIENDA }
69        - { path: ^/backend/*, roles: ROLE_SUPER_ADMIN }
70
71
```

Ilustración 34: security.yml - Definición de la autorización basada en roles

5.6.3. Diagramas del funcionamiento del sistema de seguridad

Independientemente del método de autenticación, el flujo de la petición siempre es el mismo:

1. Un usuario accede a un recurso protegido.
2. La aplicación redirige al usuario al formulario de acceso.
3. El usuario presenta sus credenciales (por ejemplo nombre de usuario/contraseña).
4. El firewall autentica al usuario.
5. El usuario intenta de nuevo la petición original ahora que ya está autenticado.

Escenario 1: Accediendo a una URL no restringida. El firewall detecta al usuario Anónimo y le permite pasar. Una vez dentro del firewall, el control de acceso detecta que no es necesaria ninguna credencial (Ilustración 35).

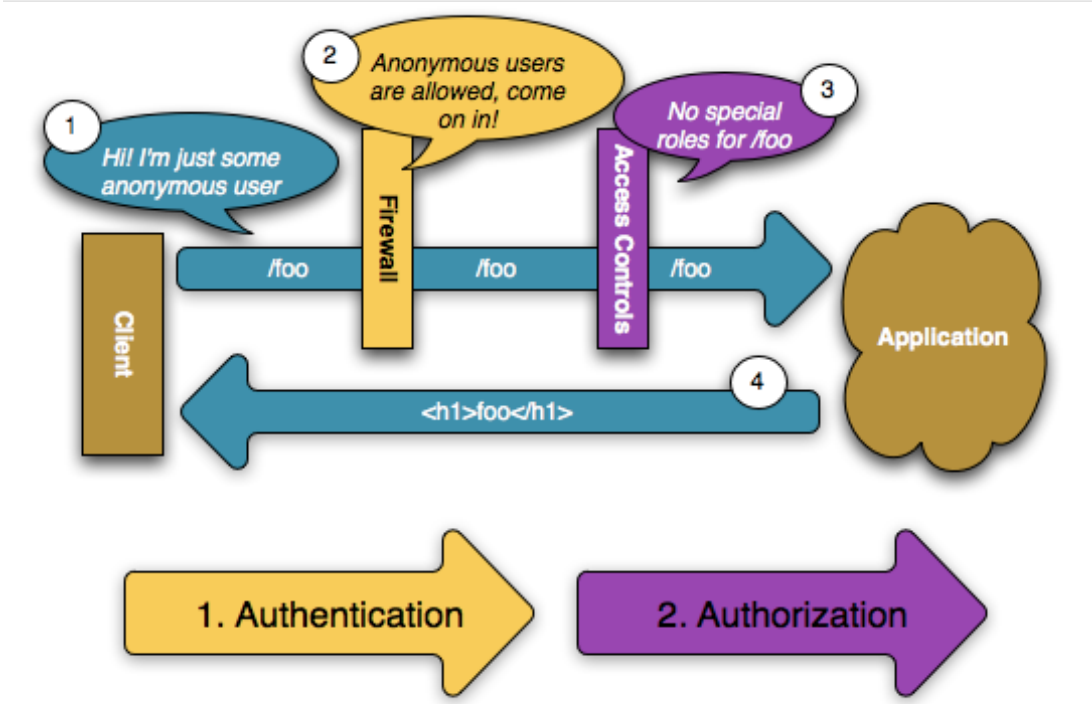


Ilustración 35: Escenario 1

Escenario 2: Denegando el acceso a una URL protegida. El usuario Anónimo intenta acceder al recurso protegido "/admin/foo". El firewall identifica al usuario pero el control de acceso comprueba que no dispone del rol ROLE_ADMIN por lo que el firewall toma el control redirigiendo al usuario a la página de login (Ilustración 36).

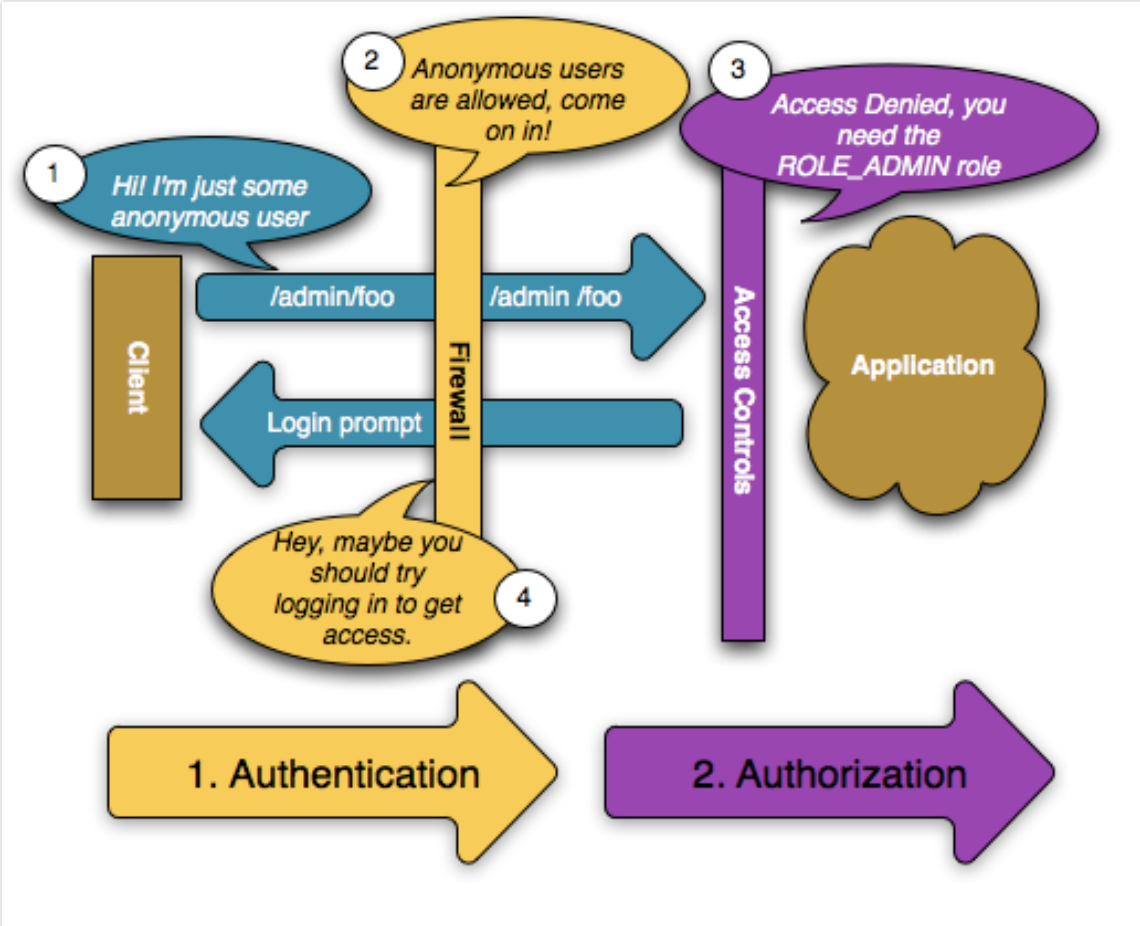


Ilustración 36: Escenario 2

Escenario 3: Denegando el acceso a un usuario sin la autorización adecuada. El usuario “ryan” no es anónimo, ha sido autentificado anteriormente, pero accede a un recurso que no tiene acceso. Symfony2, le devuelve un error 403 (Ilustración 37).

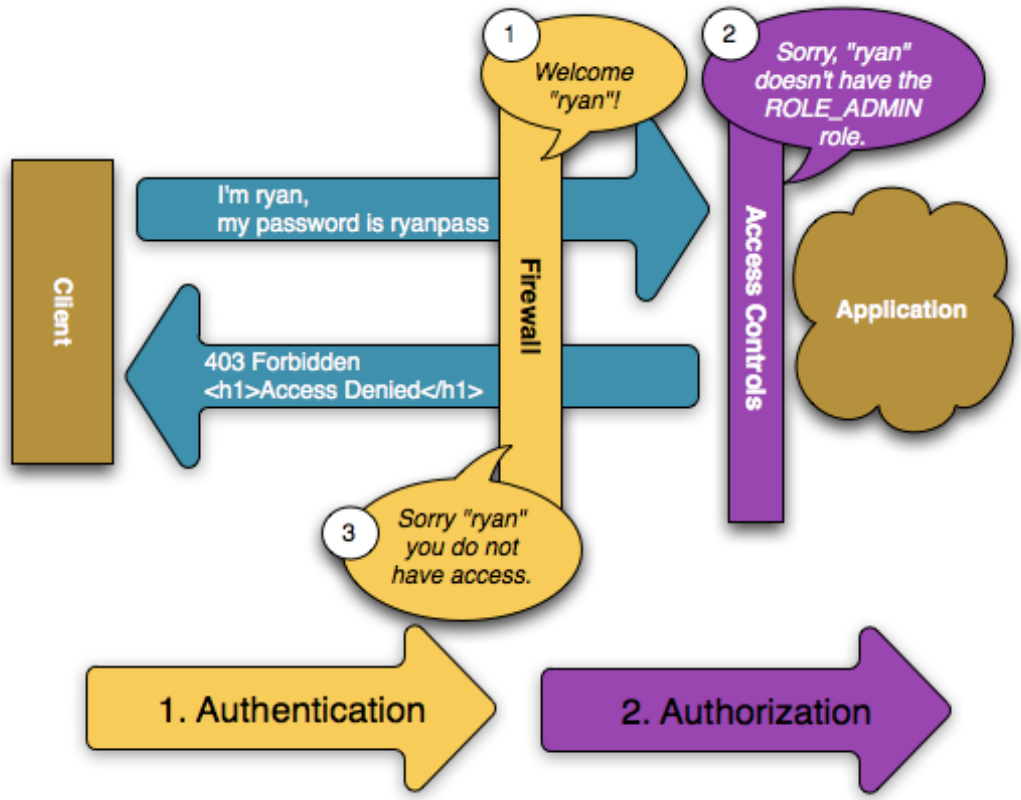


Ilustración 37: Escenario 3

Escenario 4: Accediendo a un recurso protegido. En este ejemplo, el usuario admin accede correctamente al recurso protegido (Ilustración 38).

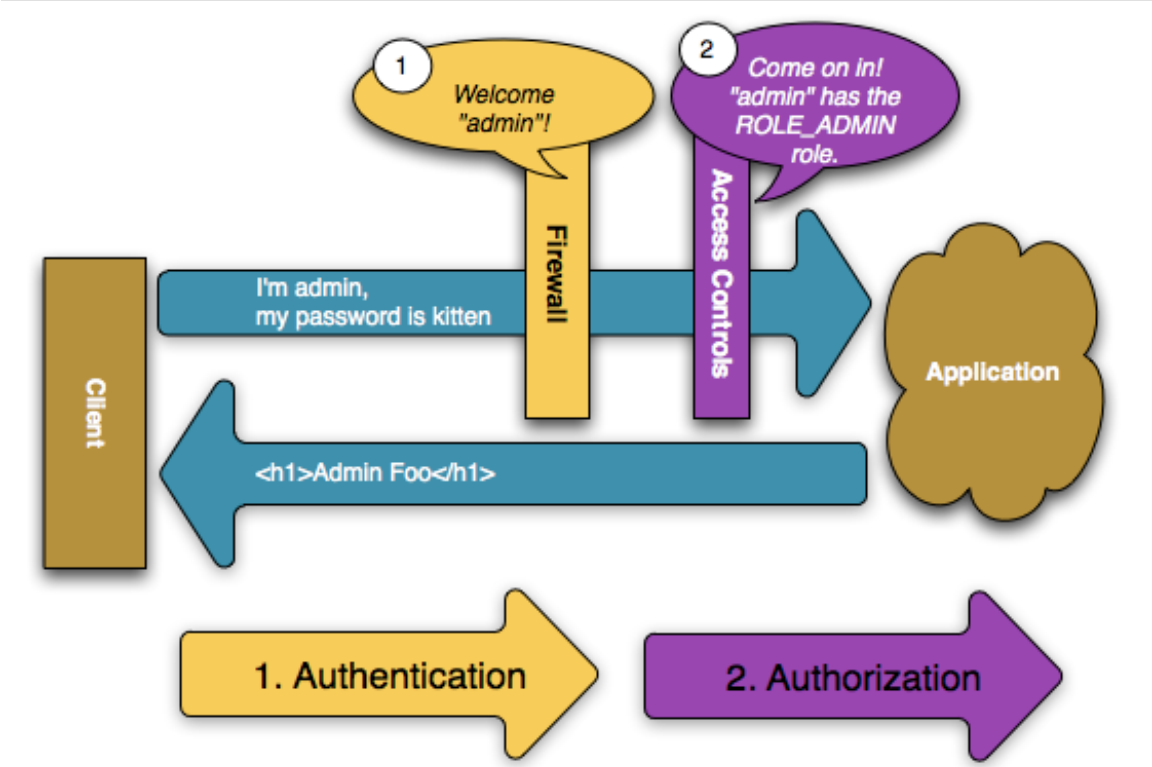


Ilustración 38: Escenario 4

6. Arquitectura de YouMarket

Si bien Symfony2 facilita la creación de una aplicación web, hay que definir correctamente cómo estructurar la aplicación. En principio, basta definir un bundle y escribir todo el código de la aplicación en un único bundle. Pero YouMarket no es una aplicación pequeña y se ha optado por separar la aplicación en tres bundles:

- YouMarketBundle: es la aplicación frontend principal, contiene además las entidades de la aplicación.
- YouMarketAdminBundle: es la parte de administración de las tiendas de la aplicación.
- YouMarketBackendBundle: el backend para la gestión general de todos los elementos de la aplicación.

6.1. Estructura básica de un bundle.

Un bundle debe seguir una estructura de directorios estándar por convención:

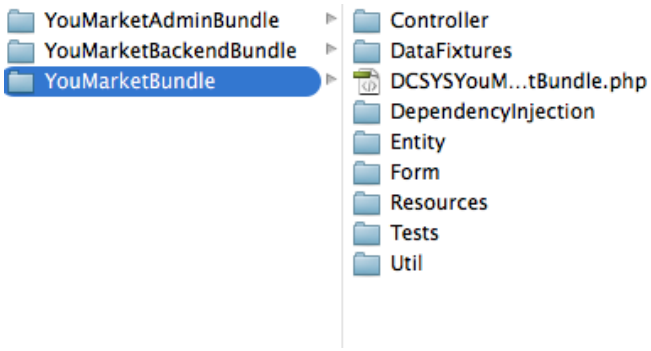


Ilustración 39: Estructura básica de un bundle

- **Controller**: contiene los controladores del bundle.
- **Entity**: contiene las entidades y repositorios (la definición del modelo).
- **Form**: contiene las clases de formularios.
- **Resources**: aquí se encuentran, entre otros recursos, las vistas, las imágenes, archivos javascript, archivos de configuración, etc.
- **Test**: se definen las clases de test del bundle.

6.2. Estructura del bundle YouMarketBundle

Este bundle se encarga de la mayor parte de la plataforma.

- Controladores
  - CategoryContoller: atiende los listados de productos de YouMarket
  - DefaultController: atiende las páginas estáticas y la portada.
  - StoreController: atiende las páginas relacionadas con las tiendas.
  - UserController: atiende las páginas relacionadas con los usuarios.
- Entidades
  - Category
  - Order
  - OrderProduct
  - OrderStore
  - Product
  - ProductVariant
  - Store
  - SubCategory
  - User
- Vista
  - Frontend
  - Mail
- Formularios
  - ContactType
  - PhotoUserType
  - RegisterUserType
  - SettingsUserType
- DataFixtures
  - CategoryFixtures
  - ProductFixtures
  - StoreFixtures

- SubCategoryFixtures
- UserFixtures

6.3. Estructura del bundle YouMarketAdminBundle

El bunde YouMarketAdminBundle gestiona el panel de administración de que disponen las tiendas para gestionar sus productos y pedidos.

- Controladores
  - DefaultController: atiende todas las peticiones de la gestión de las tiendas. Se ha centralizado todo en un sólo controlador.
- Vista
  - Se han generado las plantillas necesarias para cada vista generada.
- Formularios
  - ChangePasswordType
  - OrderStoreType
  - ProductType
  - ProductVariantType
  - SettingsType

6.4. Estructura del bundle YouMarketBackendBundle

Este bundle se encarga del panel de administración, que ha sido generado gracias a SonataAdminBundle. Para trabajar con SonataAdmin simplemente es necesario definir una clase por cada “entidad” que se quiere administrar y definir en ella los campos de búsqueda, los listados y los campos editables de la entidad. Por lo tanto se deben generar tantos controladores de administración como entidades a gestionar. Para ello se han generado los siguientes controladores:

- CategoryAdmin
- ProductAdmin
- StoreAdmin
- SubCategoryAdmin

- UserAdmin

En el capítulo siguiente se detalla con más extensión el bundle SonataAdmin.

6.5. SonataAdmin Bundle

Para generar el backend de la aplicación se ha usado el SonataAdmin [27], un bundle que permite generar paneles de administración basados en las entidades disponibles de la aplicación. Symfony2 incorpora por defecto un generador de backend llamado AdminGenerator [28], pero no está tan avanzado y maduro como SonataAdmin.

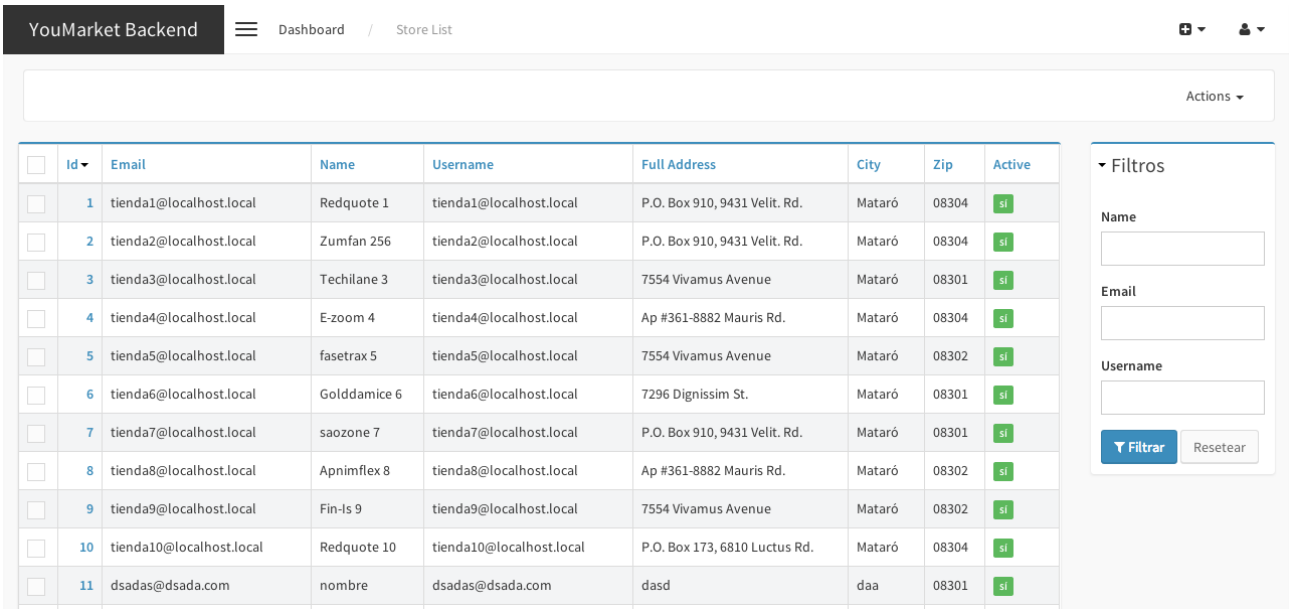


Ilustración 40: Backend de YouMarket

La ventaja de usar SonataAdmin es no tener que generar todo el código de los listados, formularios, ediciones, altas, bajas y etcétera para administrar los datos de la aplicación. Definiendo unos controladores básicos, la aplicación es capaz de generar por sí misma todo lo necesario para administrar las entidades de la aplicación.

6.6. LiipImagine Bundle

LiipImagineBundle[29] es un bundle que ayuda en el manipulado de imágenes bajo PHP haciendo uso de la librería Imagine [30].

Se ha usado este bundle para facilitar la generación de diferentes tamaños de las imágenes de los productos y de las fotos de los perfiles de YouMarket.

La configuración se define en el archivo /app/config/config.yml:

```yaml
#LiipImagineBundle Configuration
liip_imagine:
    resolvers:
       default:
          web_path: ~

    filter_sets:
        avatar_top_thumb:
            quality: 75
            filters:
                thumbnail: { size: [20, 20], mode: outbound }
        product_thumb:
            quality: 100
            filters:
                thumbnail: { size: [268, 268], mode: inset }
        product_detail:
            quality: 100
            filters:
                thumbnail: { size: [670, 670], mode: inset }
        product_detail_thumb:
            quality: 80
            filters:
                thumbnail: { size: [160, 160], mode: outbound }
        avatar_store_thumb:
            quality: 100
            filters:
                thumbnail: { size: [110, 110], mode: outbound }
        avatar_user_thumb:
            quality: 100
            filters:
                thumbnail: { size: [140, 140], mode: outbound }
```

Ilustración 41: Configuración del bundle LiipImagineBundle

En la ilustración 41 se muestra la definición de los diferentes tipos de filtros configurados. Los filtros configurados se aplican sobre las imágenes de las plantillas.

```
<img src="{{ app.user.getWebPathPhotoProfile | imagine_filter('avatar_top_thumb') }} "
class="img-circle" alt="" />
```

Ilustración 42: Ejemplo aplicación filtro de LiipImagineBundle

El ejemplo anterior (ilustración 42) se aplica el filtro definido como "*avatar_top_thumb*" a la imagen de perfil del usuario.

LiipImagineBundle además es capaz de detectar si la imagen original ha cambiado y sólo generar la nueva versión si la imagen ha sufrido algún cambio, por lo que las imágenes de diferentes tamaños generadas por el bundle se almacenan en caché para futuros usos.

7. Planificación

El proyecto se planificó en base a cuatro grandes hitos o fases: fase inicial, fase de diseño, fase de implementación y fase final.

Durante la fase inicial se trató de analizar y obtener toda la información necesaria para llevar a cabo el proyecto. Además, se realizó una exhaustiva búsqueda de documentación sobre el entorno del e-commerce en España, información sobre la tecnología web actual y las tendencias de futuro.

En la fase de diseño se trabajó en el apartado visual de la aplicación, además dibujando los elementos de interacción y obteniendo el flujo de eventos y de acciones para cumplir con los requisitos establecidos y las funcionalidades deseadas.

Desglose de los hitos y tareas:

Fase inicial: 22 días (27/01/2014 – 25/02/2014)

- Análisis de requisitos – 7 días (27/01/2014 – 04/02/2014)
- Especificación de funcionalidades – 13 días (05/02/2014 – 21/02/2014)
- Planificación – 2 días (24/02/2014 – 25/02/2014)
- Preparación de un entorno básico de desarrollo – 1 día (24/02/2014 – 24/02/2014)
- Documentación acerca de la tecnología web actual – 7 días (27/01/2014 – 21/02/2014)

Fase de diseño: 25 días (25/02/2014 – 31/03/2014)

- Diseño frontend – 10 días (25/02/2014 – 10/03/2014)
- Diseño modelo de datos – 3 días (11/03/2014 – 13/03/2014)
- Diseño arquitectura – 2 días (14/03/2014 – 17/03/2014)
- Maqueta HTML – 10 días (18/03/2014 –31/03/2014)

Fase de implementación: 35 días (01/04/2014 – 19/05/2014)

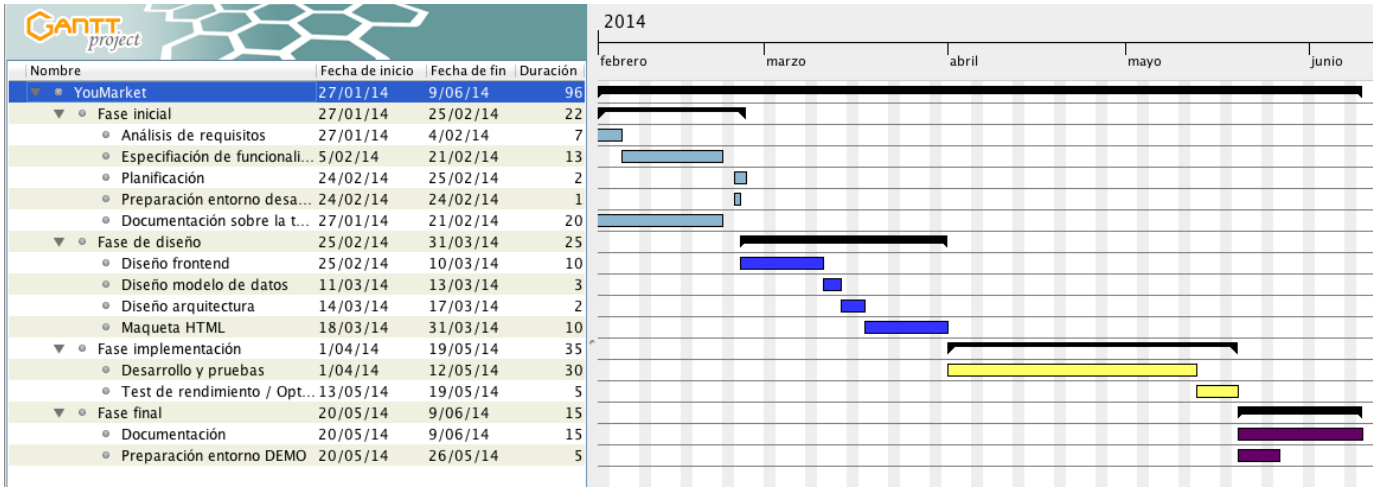
- Desarrollo y pruebas - 30 días (1/04/2014 – 12/05/2014)
- Test de rendimiento / Optimización – 5 días (13/05/2014 – 19/05/2014)

Fase final: 15 días (20/05/2014 – 9/06/2014)

- Documentación – 15 días (20/05/2014 – 09/06/2014)
- Preparación entorno de demo – 5 días - (20/05/2014 – 26/05/2014)

Duración total del proyecto estimado: 96 días.

Diagrama de Gantt de la planificación proyectada:



Nombre	Fecha de inicio	Fecha de fin	Duración
YouMarket	27/01/14	9/06/14	96
Fase inicial	27/01/14	25/02/14	22
Análisis de requisitos	27/01/14	4/02/14	7
Especifiación de funcionali...	5/02/14	21/02/14	13
Planificación	24/02/14	25/02/14	2
Preparación entorno desa...	24/02/14	24/02/14	1
Documentación sobre la t...	27/01/14	21/02/14	20
Fase de diseño	25/02/14	31/03/14	25
Diseño frontend	25/02/14	10/03/14	10
Diseño modelo de datos	11/03/14	13/03/14	3
Diseño arquitectura	14/03/14	17/03/14	2
Maqueta HTML	18/03/14	31/03/14	10
Fase implementación	1/04/14	19/05/14	35
Desarrollo y pruebas	1/04/14	12/05/14	30
Test de rendimiento / Opt...	13/05/14	19/05/14	5
Fase final	20/05/14	9/06/14	15
Documentación	20/05/14	9/06/14	15
Preparación entorno DEMO	20/05/14	26/05/14	5

Ilustración 43: Diagrama de Gantt de la planificación

8. Pruebas de validación y rendimiento

Durante el desarrollo de la aplicación se han realizado una serie de pruebas y tests para asegurar el correcto desarrollo de la plataforma.

8.1. Validación HTML / CSS

El código HTML de la plataforma ha sido comprobado por el validador online [31] de la World Wide Web Consortium (W3C) para asegurar el cumplimiento estricto del marcado HTML5.

Estas validaciones se han realizado tanto en la primera versión del HTML sin programación y posteriormente durante la fase de desarrollo para asegurar la correcta implementación y evitar posibles errores durante el desarrollo de las plantillas de la vista.

8.2. Pruebas sobre las entidades

Las pruebas se han basado en la creación de Data Fixtures durante el desarrollo y verificar que las entidades de la aplicación se generaban correctamente dentro de la base de datos con los juegos de prueba implementados.

8.3. Test de rendimiento

El rendimiento en un entorno real no ha sido posible analizarlo debido a que el proyecto ha sido desarrollado en un entorno didáctico. Sin embargo, se han realizado unos test de velocidad de navegación en base a las recomendaciones que Google determina sobre este aspecto.

Las normas expuestas por Google en su programa PageSpeed Insights [32] son las siguientes y vienen clasificadas en dos grandes grupos: de velocidad y de usabilidad.

- Reglas de velocidad
  - Evitar los redireccionamientos a páginas de destino.
  - Habilitar compresión.
  - Mejorar el tiempo de respuesta de servidor.

    - Especificar caché de navegador.
    - Minificar recursos.
    - Optimizar imágenes.
    - Optimizar la entrega de CSS.
    - Priorizar el contenido visible.
    - Eliminar el javascript que bloquea la visualización de contenido.
    - Usar scripts javascript asíncronos.
- Reglas de usabilidad
    - Evitar los plugins.
    - Configurar la ventana gráfica.
    - Adaptar el contenido a la ventana gráfica.
    - Aplicar el tamaño adecuado a los botones táctiles.
    - Utilizar tamaños de fuente que se puedan leer.

El test se ha realizado con el complemento de PageSpeed de Google usando el navegador web Firefox 29.0.1 sobre la página inicial de YouMarket.

En la ilustración 44 se visualiza el resultado del test. Se puede apreciar que sólo existe una advertencia sobre el elemento “Optimizar imágenes” que puede afectar a la velocidad de navegación de forma relevante. En concreto el test avisa de que es posible optimizar una serie de imágenes para reducir su peso y así mejorar la velocidad de carga de la web (Ilustración 45).

Tras analizar los resultados, se puede afirmar que la aplicación está cumpliendo todas las reglas que marca Google en sus directrices sobre optimización. Es importante cumplir estas normas básicas propuestas por Google ya que estos parámetros los tiene en cuenta el buscador a la hora de posicionar un sitio web en sus resultados de búsqueda.

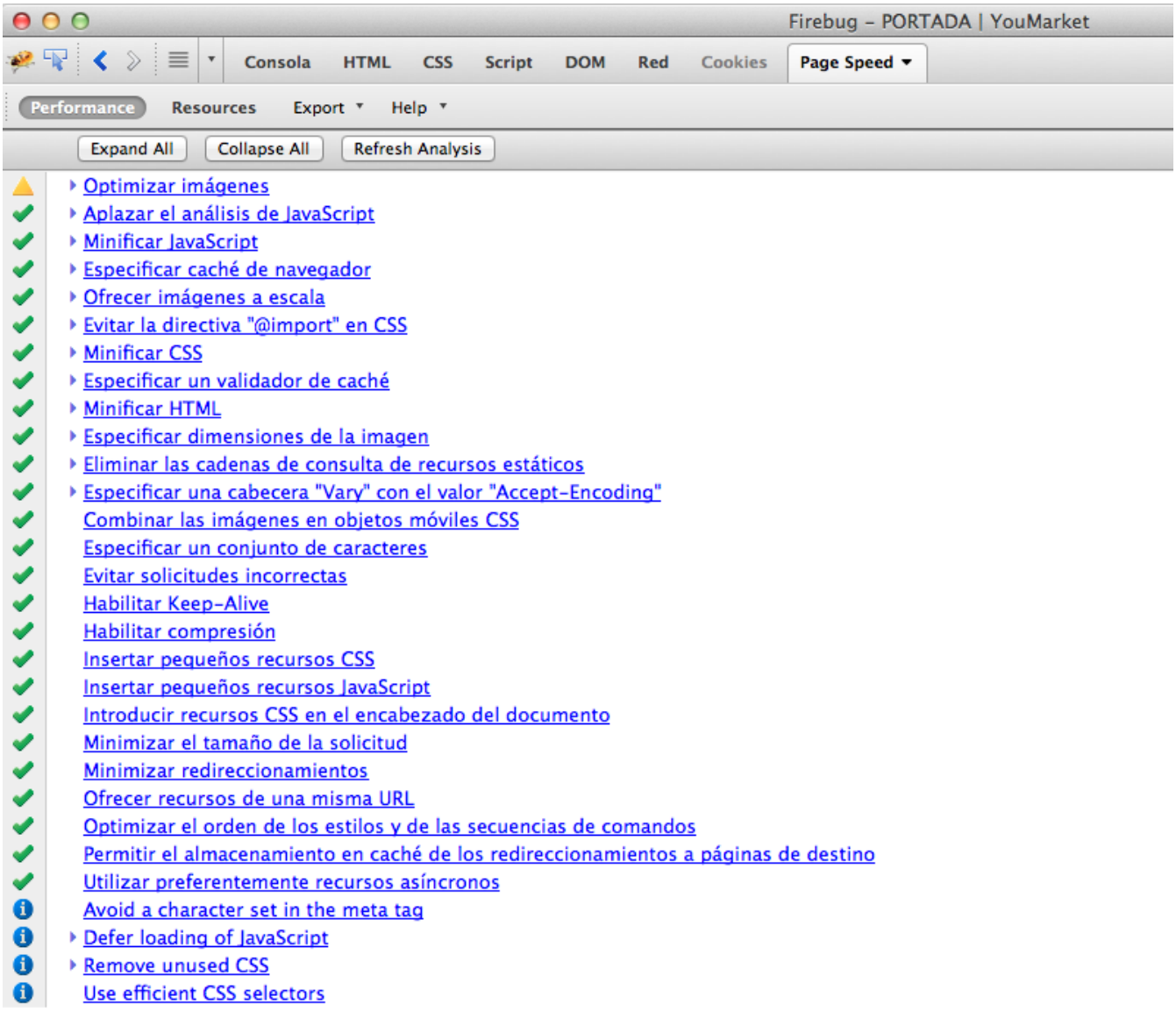


Ilustración 44: Resultado test PageSpeed, porada de YouMarket

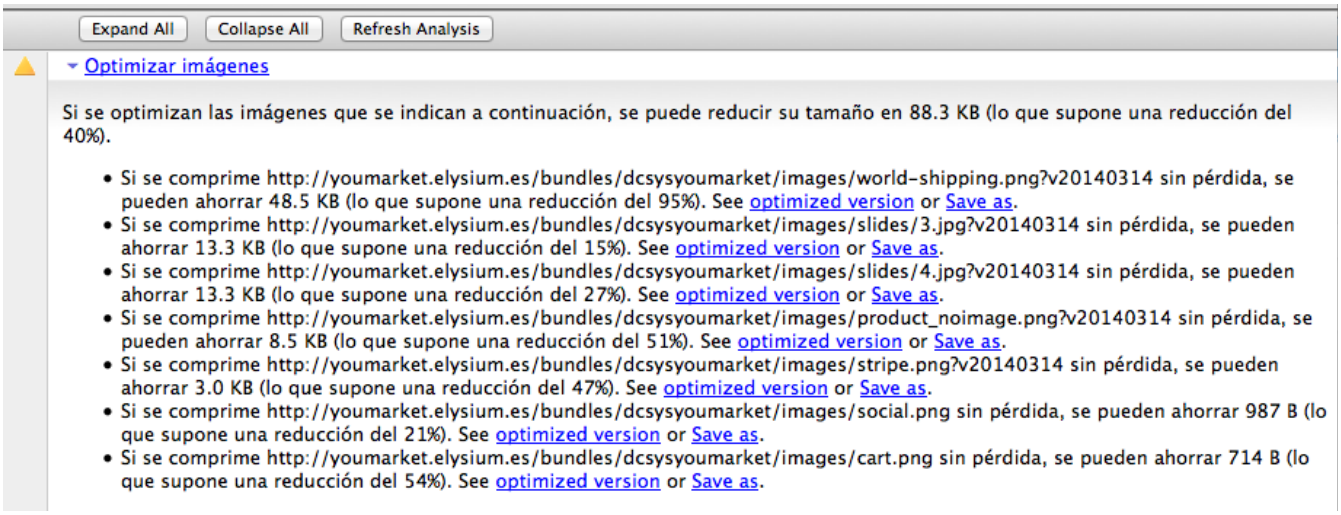


Ilustración 45: Aviso sobre las imágenes a optimizar

9. Escalabilidad del sistema

Cuando se trata de escalar una aplicación PHP se deben considerar diferentes aspectos sobre el entorno:

- Realizar un bytecode caché (APC or Zend Optimizer Plus or eAccelerator).
- Leer archivos desde memoria en lugar del disco duro.
- Cachear y minificar el contenido estático.
- Guardado de las sesiones de usuario.

Con Symfony2, el primer y tercer punto se resuelven fácilmente gracias a Symfony2 que usa por defecto APC [22] para realizar el caché del bytecode. Además, el cacheado y minificación del contenido estático se puede realizar con el componente Assetic. El segundo punto se encuentra parcialmente resuelto también en Symfony2 gracias a su proxy inverso escrito en PHP, aunque para un entorno real se aconseja el uso de Varnish [23] como proxy inverso.

Para solucionar el cuarto punto sobre el guardado de las sesiones de usuarios hay que tener en cuenta que en PHP las sesiones de usuario se guardan por defecto en archivos. Hay diferentes métodos para mejorar el rendimiento, una por ejemplo es usar memcache [33] o mapear el salvado de las sesiones en un ramdisc [34], etc. Pero cuando se desea escalar en diferentes nodos la aplicación (diferentes servidores web detrás de un balanceador de carga) no hay garantía en este tipo de arquitectura distribuida que el mismo usuario sea atendido por el mismo nodo.

Esto implica que las sesiones en memoria deben estar compartidas entre los nodos. Desafortunadamente, si se almacena la sesión de usuario en la memoria de un servidor no se cumple dicha compartición de memoria entre los nodos. La solución es delegar la gestión de las sesiones a un sistema compartido entre los diferentes nodos. Para ello, se puede hacer uso de Redis [35].

9.1. Redis. Gestionar las sesiones de usuario

Redis es un motor de base de datos en memoria basado en tablas hash (clave / valor). Soporta además cadenas, listas, conjuntos (ordenados o no) y hash maps. Los datos deben caber en memoria y pueden ser persistentes o no (configurable). Redis es ideal para escrituras y lecturas intensivas, como lo es controlar las sesiones de usuario de PHP.

Tomando el escenario de la ilustración 46, donde cada servidor web gestiona sus sesiones de usuario, se producen estados inconsistentes debido a que no se puede garantizar que un usuario sea atendido siempre por el mismo nodo.

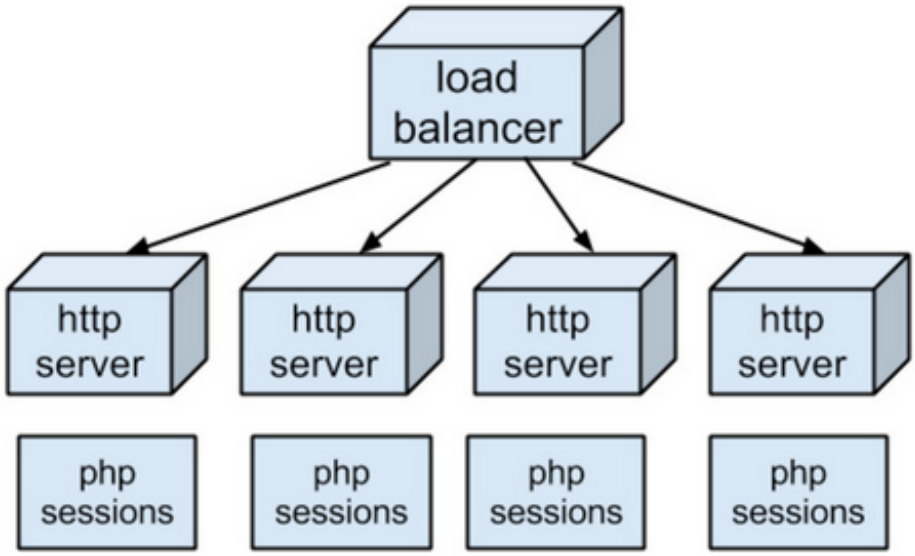


Ilustración 46: Arquitectura no consistente de sesiones de usuario

El ejemplo siguiente es una buena aproximación a una buena solución pero de difícil escalado en escenarios con mucho tráfico: usando un Relational Database Management System (RDBMS), es decir, guardando las sesiones en una base de datos relacional y centralizada a todos los nodos (Ilustración 47).

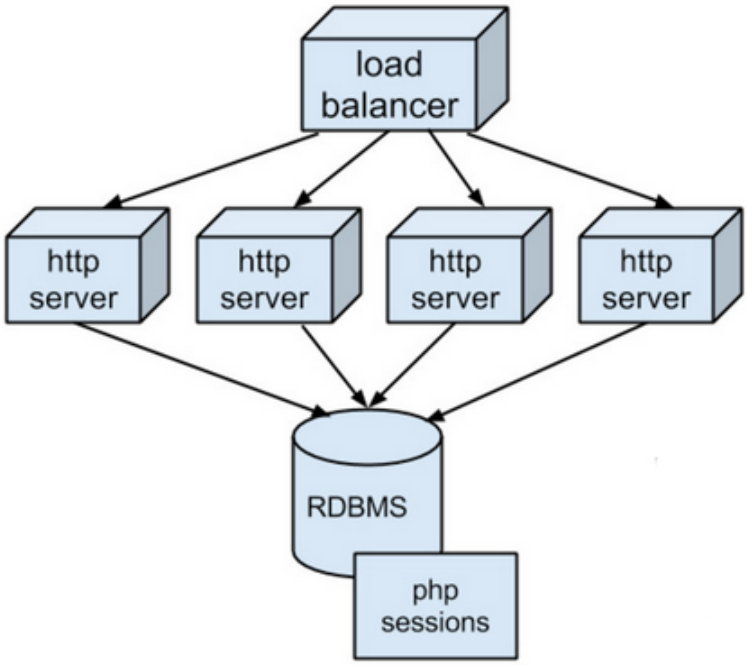


Ilustración 47: Arquitectura basada en un RDBMS

Por último, la solución propuesta al implementar Redis se visualiza en la ilustración 48. Las sesiones se almacenan en memoria gracias a Redis y escala horizontalmente pudiendo añadir tantos servidores Redis como sea necesario en función del tráfico.

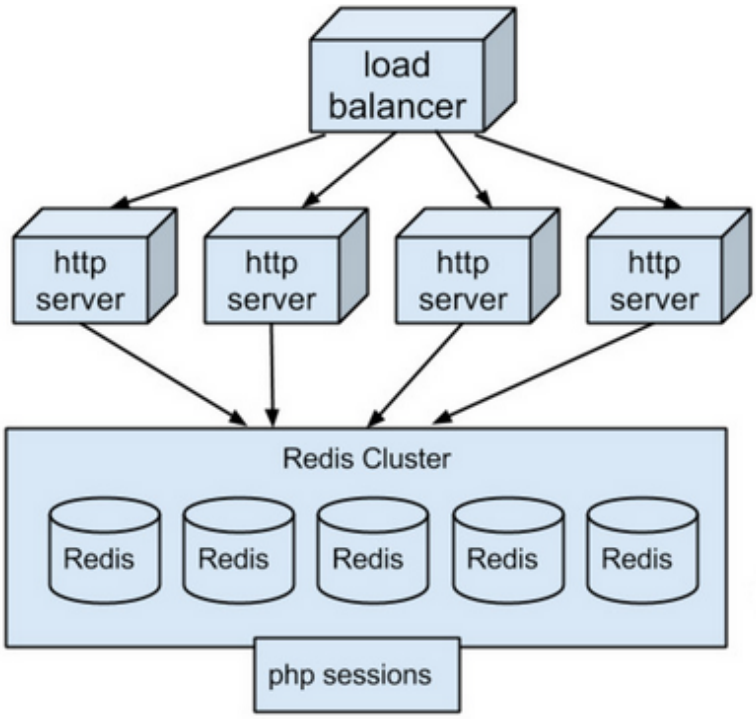


Ilustración 48: Arquitectura basada en Redis

9.2. Observaciones

Por lo general ninguna aplicación web cuenta desde el inicio con una carga de tráfico tan grande como para contar desde el inicio con un balanceador de carga, un cluster Redis y demás aspectos comentados en este punto, pero sí que es fundamental estar preparado ante necesidades futuras para atender una alta carga de tráfico en la aplicación.

10. Conclusiones

Durante el desarrollo de YouMarket se ha tenido desde el inicio una continua iteración del desarrollo. Para ello ha sido imprescindible proporcionar una base robusta para poder seguir añadiendo funcionalidades a medida que el proyecto va creciendo. El uso de un framework PHP de base ha sido la respuesta a esa necesidad de robustez.

Además de apoyarse en un framework para desarrollar YouMarket, cabe destacar el desempeño de llevar a cabo con éxito las etapas de análisis, planificación, diseño y desarrollo de un proyecto web como es YouMarket.

Personalmente, el objetivo del proyecto era realmente descubrir una buena plataforma de desarrollo web en PHP y aplicar tecnologías web actuales y así construir la primera versión de YouMarket de una forma escalable y robusta.

Sin embargo hay una serie de extensiones a tener en cuenta realmente interesantes que se detallan en los siguientes apartados y que no han podido ser abarcados durante la realización del proyecto.

10.1. Mejoras propuestas

10.1.1. Varnish

Usar Varnish [23] como proxy inverso en el entorno de producción para mejorar el tiempo de respuesta y la velocidad de carga de la plataforma.

10.1.2. Monolog

Realizar un logging de eventos dentro de la aplicación con el componente Monolog.

10.1.3. RabbitMQ

RabbitMQ [36] para gestionar las colas de email y otros procesos que pueden gestionarse por colas dentro de la aplicación.

10.1.4. Minimizar el contenido CSS / HTML / Javascript

Activar la unificación y minificación del HTML, CSS y Javascript con Assetic [37]. Dicha acción permite unir todos los archivos CSS en sólo archivo, de igual modo con los archivos javascript. También se trata de eliminar los espacios en blanco y saltos de línea de los archivos tanto HTML, CSS y Javascript de la aplicación.

Con estas acciones se reduce el número de peticiones que debe realizar el navegador web del cliente al servidor web y además el tamaño total de descarga de los archivos.

10.1.5. Content Delivery Network (CDN)

Subida de imágenes a S3 de Amazon o un Content Delivery Network (CDN) equivalente para mejorar el tiempo de respuesta de carga de las imágenes.

10.1.6. Redis

Gestionar sesiones de usuario con Redis, comentado en profundidad en el capítulo 9 de esta documentación.

10.1.7. Inicio de sesión con perfiles sociales

Posibilidad de registrase en YouMarket e iniciar sesión vía una cuenta de Facebook o Twitter.

10.1.8. Tour inicial

Obtener datos de los nuevos usuarios vía un “Tour” sobre sus preferencias y recomendar otros perfiles a seguir basados en esas preferencias.

10.1.9. Subida múltiple de imágenes por “drag & drop”

Implementar el sistema de “drag & drop” para la subida de imágenes de los productos de los comercios.

10.1.10. Optimización del posicionamiento en buscadores

Trabajar en la generación de los títulos, descripciones y palabras clave de forma

automática en función del contenido de cada página generada en YouMarket para mejorar el posicionamiento en buscadores, además de generar esta información para las páginas estáticas de la plataforma.

11. Bibliografía

[1] "RAKUTEN.ES - Comprar y vender electrónica, gourmet, productos de cosmética y más." [Online]. Available: http://www.rakuten.es/. [Accessed: 03-May-2014].

[2] "Manufacturers, Suppliers, Exporters & Importers from the world's largest online B2B marketplace-Alibaba.com." [Online]. Available: http://www.alibaba.com/. [Accessed: 03-May-2014].

[3] "ONTSI | Observatorio Nacional de las Telecomunicaciones y de la SI." [Online]. Available: http://www.ontsi.red.es/ontsi/. [Accessed: 03-May-2014].

[4] "Shopify." [Online]. Available: http://es.shopify.com/.

[5] "Tiendy - Crear Tienda Online." [Online]. Available: http://www.tiendy.com/. [Accessed: 11-Jun-2014].

[6] "Crear tienda online | Tienda online gratis - Webnode." [Online]. Available: http://www.webnode.es/tienda-virtual/. [Accessed: 11-Jun-2014].

[7] "Crear tienda online - TPV online. Crea tu tienda online fácil y rápido." [Online]. Available: http://www.gestiontpv.com/. [Accessed: 11-Jun-2014].

[8] "Free Online Store Builder on Storenvy." [Online]. Available: http://www.storenvy.com/. [Accessed: 11-Jun-2014].

[9] "PayPal España." [Online]. Available: https://www.paypal.com/es/webapps/mpp/accept-payments-online. [Accessed: 05-Jun-2014].

[10] "CyberPac." [Online]. Available: http://www.comerciaglobalpayments.com/tienesunatiendaonline_es.html. [Accessed: 05-Jun-2014].

[11] bradfrostweb.com, "Responsive Strategy | Brad Frost Web," 2014. [Online]. Available: http://bradfrostweb.com/blog/post/responsive-strategy/. [Accessed: 11-Mar-2014].

[12] “Website Goodies - Website speed: How fast should your website be?” [Online]. Available: http://www.websitegoodies.com/website-development/website-speed-how-fast-should-your-website-be. [Accessed: 11-Mar-2014].

[13] “Ubuntu 12.04.4 LTS (Precise Pangolin).” [Online]. Available: http://releases.ubuntu.com/precise/. [Accessed: 03-May-2014].

[14] “VMware Virtualization for Desktop & Server, Application, Public & Hybrid Clouds.” [Online]. Available: http://www.vmware.com/. [Accessed: 23-May-2014].

[15] “Samba - opening windows to a wider world.” [Online]. Available: https://www.samba.org/. [Accessed: 23-May-2014].

[16] “Composer | Dependency Manager for PHP.” [Online]. Available: https://getcomposer.org/. [Accessed: 03-May-2014].

[17] D. P. Team and N. Pacheco, “Doctrine 2 ORM Documentation,” 2011.

[18] “Foundation | The Most Advanced Responsive Front-end Framework from ZURB.” [Online]. Available: http://foundation.zurb.com/. [Accessed: 03-May-2014].

[19] “Bootstrap.” [Online]. Available: http://getbootstrap.com/. [Accessed: 03-May-2014].

[20] “What is Symfony2?,” 2014. [Online]. Available: http://fabien.potencier.org/article/49/what-is-symfony2. [Accessed: 12-Mar-2014].

[21] “The DCI Architecture: A New Vision of Object-Oriented Programming.” [Online]. Available: http://www.artima.com/articles/dci_vision.html. [Accessed: 03-Jun-2014].

[22] “PHP: APC - Manual.” [Online]. Available: http://php.net/manual/es/book.apc.php. [Accessed: 25-Apr-2014].

[23] “Front page | Varnish Community.” [Online]. Available: https://www.varnish-cache.org/. [Accessed: 28-Apr-2014].

[24] “squid : Optimising Web Delivery.” [Online]. Available: http://www.squid-cache.org/. [Accessed:

28-Apr-2014].

[25] "12.12. Protección CSRF (Symfony 2.4, el libro oficial)." [Online]. Available: http://librosweb.es/symfony_2_x/capitulo_12/proteccion_csrf.html. [Accessed: 30-Apr-2014].

[26] "Doctrine Query Language — Doctrine 2 ORM 2 documentation." [Online]. Available: http://doctrine-orm.readthedocs.org/en/latest/reference/dql-doctrine-query-language.html. [Accessed: 30-May-2014].

[27] "Sonata Admin's documentation - Index (2.2)." [Online]. Available: http://sonata-project.org/bundles/admin/2-2/doc/index.html. [Accessed: 03-May-2014].

[28] "Documentation for AdmigeneratorGeneratorBundle." [Online]. Available: http://symfony2admingenerator.org/. [Accessed: 03-May-2014].

[29] "LiipImagineBundle." [Online]. Available: https://github.com/liip/LiipImagineBundle.

[30] "Imagine's documentation!" [Online]. Available: https://imagine.readthedocs.org/en/latest/. [Accessed: 28-May-2014].

[31] "The W3C Markup Validation Service." [Online]. Available: http://validator.w3.org/. [Accessed: 02-Jun-2014].

[32] "Reglas de PageSpeed Insights - PageSpeed Insights — Google Developers." [Online]. Available: https://developers.google.com/speed/docs/insights/rules. [Accessed: 02-Jun-2014].

[33] "PHP: Memcache - Manual." [Online]. Available: http://www.php.net/manual/es/book.memcache.php. [Accessed: 02-Jun-2014].

[34] "Create a RAM disk in Linux | JamesCoyle.net." [Online]. Available: http://www.jamescoyle.net/how-to/943-create-a-ram-disk-in-linux. [Accessed: 02-Jun-2014].

[35] "Redis." [Online]. Available: http://redis.io/. [Accessed: 02-Jun-2014].

[36] "RabbitMQ - Messaging that just works." [Online]. Available: http://www.rabbitmq.com/. [Accessed: 28-May-2014].

[37] “Minificar ficheros CSS y JS con Assetic y Symfony2.” [Online]. Available: http://kiwwito.com/minificar-ficheros-css-y-js-con-assetic-y-symfony2/. [Accessed: 02-Jun-2014].

[38] “Cómo configurar bien Apache para las aplicaciones Symfony2.” [Online]. Available: http://symfony.es/documentacion/como-configurar-bien-apache-para-las-aplicaciones-symfony2/. [Accessed: 23-May-2014].

[39] “Empresas que realizan comercio electrónico | ONTSI.” [Online]. Available: http://www.ontsi.red.es/ontsi/es/indicador/empresas-que-realizan-comercio-electrónico. [Accessed: 03-May-2014].

[40] “Productos y servicios adquiridos a través de Internet | ONTSI.” [Online]. Available: http://www.ontsi.red.es/ontsi/es/indicador/productos-y-servicios-adquiridos-trav%C3%A9s-de-internet. [Accessed: 05-Jun-2014].

[41] “Volumen de negocio de comercio electrónico en España | ONTSI.” [Online]. Available: http://www.ontsi.red.es/ontsi/es/indicador/volumen-de-negocio-de-comercio-electrónico-en-españa. [Accessed: 03-May-2014].

[42] “Cifra de comercio electrónico sobre el total de negocio de la empresa | ONTSI.” [Online]. Available: http://www.ontsi.red.es/ontsi/es/indicador/cifra-de-comercio-electrónico-sobre-el-total-de-negocio-de-la-empresa. [Accessed: 03-May-2014].

12. Anexo

12.1. Archivo de configuración Apache

```
<VirtualHost *:80>
    ServerName youmarket.elysium.es
    DocumentRoot /home/youmarket/public_html/web
    ErrorLog /var/log/virtualmin/youmarket.elysium.es_error_log
    CustomLog /var/log/virtualmin/youmarket.elysium.es_access_log combined
    ScriptAlias /cgi-bin/ /home/youmarket/cgi-bin/
    DirectoryIndex index.html index.htm index.php index.php4 index.php5
        <Directory /home/youmarket/public_html/web>
            Options -Indexes +IncludesNOEXEC +SymLinksIfOwnerMatch
            allow from all
            AllowOverride All Options=ExecCGI,Includes,IncludesNOEXEC,Indexes,M
            ultiViews,SymLinksIfOwnerMatch
        </Directory>
        <Directory /home/youmarket/cgi-bin>
            allow from all
            AllowOverride All Options=ExecCGI,Includes,IncludesNOEXEC,Indexes,M
            ultiViews,SymLinksIfOwnerMatch
        </Directory>
    RewriteEngine on
</VirtualHost>

```

Ilustración 49: Configuración del servidor virtual de Apache2

El ejemplo anterior es una configuración básica para el desarrollo, es recomendable mejorar dicha configuración en entornos de producción se puede seguir por ejemplo la guía descrita en la página web symfony.es [38].

12.2. Configuración de MySQL

Para el desarrollo actual de YouMarket no ha sido necesaria ninguna modificación en particular de la configuración estándar de MySQL.

12.3. Archivo de configuración PHP

Symfony2 requiere de algunos ajustes de la configuración de PHP por defecto. Para comprobar si dicha configuración de PHP cumple con los requisitos marcados por Symfony2, se puede ejecutar el comprobador de configuración que incorpora el propio Symfony2 con el comando siguiente:

$ php app/check.php

La salida de la comprobación es así:

****

** Configuration file used by PHP: /etc/php5/cli/php.ini*

*** Mandatory requirements ***

OK PHP version must be at least 5.3.3 (5.3.10-1ubuntu3.9 installed)

OK PHP version must not be 5.3.16 as Symfony won't work properly withit

OK Vendor libraries must be installed

OK app/cache/ directory must be writable

OK app/logs/ directory must be writable

ERROR date.timezone setting must be set

Set the "date.timezone" setting in php.ini* (like Europe/Paris).

OK Configured default timezone "Europe/Berlin" must be supported by your installation of PHP

OK json_encode() must be available

OK session_start() must be available

OK ctype_alpha() must be available

OK token_get_all() must be available

OK simplexml_import_dom() must be available

OK detect_unicode must be disabled in php.ini

OK PCRE extension must be available

*** Optional recommendations ***

OK Requirements file should be up-to-date

OK You should use at least PHP 5.3.4 due to PHP bug #52083 in earlier versions

OK When using annotations you should have at least PHP 5.3.8 due to PHP bug #55156

OK You should not use PHP 5.4.0 due to the PHP bug #61453

OK PCRE extension should be at least version 8.0 (8.12 installed)

OK PHP-XML module should be installed

OK mb_strlen() should be available

OK iconv() should be available

OK utf8_decode() should be available

OK posix_isatty() should be available

WARNING intl extension should be available

Install and enable the intl extension (used for validators).

WARNING a PHP accelerator should be installed

Install and enable a PHP accelerator like APC (highly recommended).

OK magic_quotes_gpc should be disabled in php.ini

OK register_globals should be disabled in php.ini

OK session.auto_start should be disabled in php.ini

OK PDO should be installed

OK PDO should have some drivers installed (currently available: mysql)

El comprobador informa de los problemas e informa de acciones recomendadas, para el entorno de desarrollo, se han aplicado las siguientes acciones complementarias a la instalación de PHP:

- Editado el archivo php.ini para indicar una zona horaria concreta.
  - Variable: date.timezone = “Europe/Madrid”
- Instalación de la extensión “**intl”.**
  - **Descripción:** intl es la extensión de PHP para la internacionalización. Permite a los programadores realizar un formateo de fecha, hora, moneda y número de en los scripts.
  - **Instalación en el servidor:** $ sudo apt-get install php5-intl
  - Más información: http://es1.php.net/manual/es/intro.intl.php

12.4. Casos de uso usuario Anónimo

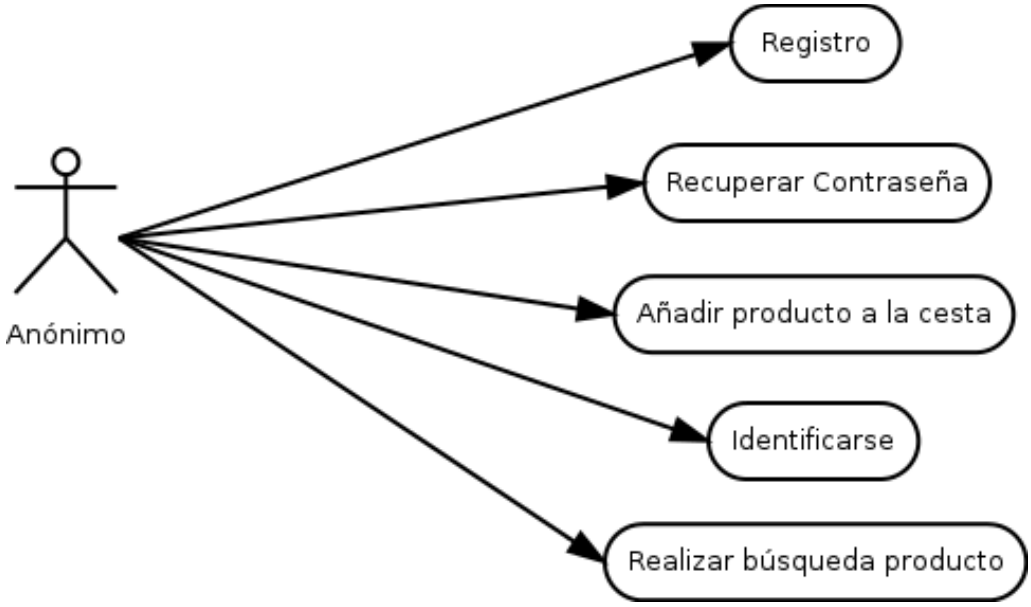


Ilustración 50: Diagrama casos de uso usuario anónimo

UC1. Registro	
Objetivo	Crear una cuenta de usuario en YouMarket
Actor	Usuario anónimo
Precondiciones: -	
Procedimiento: 1. Usuario: Visita YouMarket 2. Sistema: Procesa la petición solicitada 3. Usuario: Accede al formulario de registro de usuarios 4. Sistema: Presenta el formulario de registro 5. Usuario: Rellena los datos y envía el formulario 6. Sistema: Validación de los datos 7. Sistema: Alta del nuevo Usuario. 7.1. Enviar e-mail administrador 7.2. Enviar e-mail al nuevo usuario 8. Sistema: Re-dirige al usuario a la portada	

Extensiones: 6.1. El usuario no introduce datos bien formateados. Volver al paso 4. 6.2. Mostrar mensaje de error en los datos mal introducidos.

UC2. Recuperar Contraseña	
Objetivo	Obtener nueva contraseña en el e-mail usado en el registro.
Actor	Usuario anónimo
Precondiciones: -	
Procedimiento: 1. Usuario: Visita YouMarket 2. Sistema: Procesa la petición solicitada 3. Usuario: Accede al formulario de recuperación de contraseña 4. Sistema: Presenta el formulario de recuperación 5. Usuario: Rellena los datos y envía el formulario 6. Sistema: Validación de los datos 7. Sistema: Genera nueva contraseña del usuario válida 7.1. Enviar e-mail al usuario 8. Sistema: Mostrar mensaje de confirmación	
Extensiones: 6.1. El usuario no introduce datos bien formateados. Volver al paso 4. 6.2. Mostrar mensaje de error en los datos mal introducidos.	

UC3. Identificarse	
Objetivo	Autenticar al usuario.
Actor	Usuario anónimo
Precondiciones: -	

Procedimiento: 1. Usuario: Visita YouMarket 2. Sistema: Procesa la petición solicitada 3. Usuario: Accede al formulario de identificación 4. Sistema: Presenta el formulario de identificación 5. Usuario: Rellena los datos y envía el formulario 6. Sistema: Validación de los datos 7. Sistema: Autentifica al usuario en el sistema 8. Sistema: Re-dirige al panel de usuario
Extensiones: 6.1. El usuario no introduce datos bien formateados. Volver al paso 4. 6.2. Mostrar mensaje de error en los datos mal introducidos.

UC4. Realizar búsqueda producto	
Objetivo	Realizar una búsqueda de productos disponibles en YouMarket
Actor	Usuario anónimo
Precondiciones: -	
Procedimiento: 1. Usuario: Visita YouMarket 2. Sistema: Procesa la petición solicitada 3. Usuario: Introduce una consulta en la caja de texto del buscador 4. Sistema: Validación de los datos 5. Sistema: Realizar la búsqueda 6. Sistema: Mostrar resultados 6.1. Si no hay resultados, mostrar mensaje alternativo.	
Extensiones: 4.1. El usuario no introduce datos bien formateados. Volver al paso 2. 4.2. Mostrar mensaje de error en los datos mal introducidos.	

UC5. Añadir producto a la cesta	
Objetivo	Añadir un producto a la cesta de YouMarket
Actor	Usuario anónimo
Precondiciones: -	
Procedimiento: 1. Usuario: Visita YouMarket 2. Sistema: Procesa la petición solicitada 3. Usuario: Accede al detalle de un producto disponible 4. Sistema: Muestra la información del producto 5. Usuario: Hace click sobre el botón “Añadir al carrito” 6. Sistema: Procesa la petición y añade el producto y la variante elegida a la cesta de compra del usuario. 7. Sistema: Muestra un dialogo para notificar el evento. 8. Usuario: Puede elegir seguir comprando o visitar su cesta de la compra	
Extensiones: 5.1. Si está disponible, el usuario puede elegir la variante del producto disponible.	

12.5. Casos de uso usuario Registrado

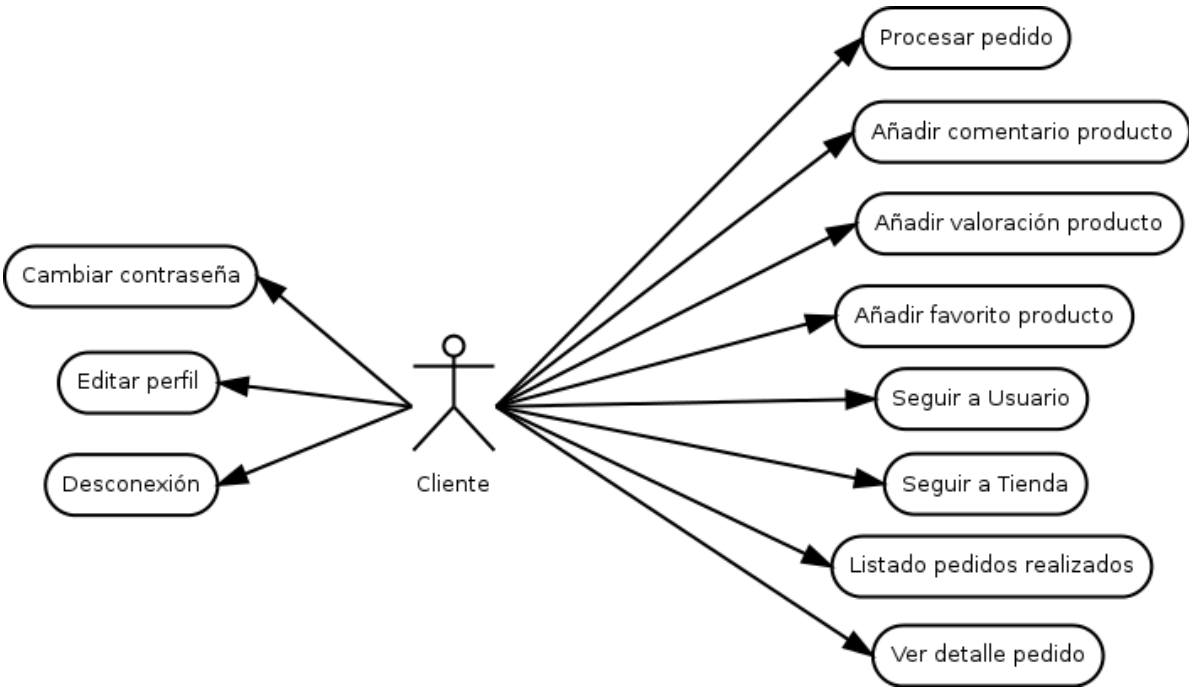


Ilustración 51: Diagrama casos de uso del usuario registrado

UC6. Cambiar contraseña	
Objetivo	Cambiar la contraseña actual de la cuenta por una nueva de elección del usuario que cumpla con los requisitos de seguridad mínimos obligados por YouMarket.
Actor	Usuario Cliente
Precondiciones: El usuario se encuentra correctamente autenticado	
Procedimiento: 1. Usuario: Visita YouMarket 2. Sistema: Procesa la petición solicitada 3. Usuario: Accede a su panel de preferencias → Cambiar contraseña 4. Sistema: Presenta el formulario de cambio de contraseña 5. Usuario: Ingresa su contraseña actual y dos veces la nueva a establecer. Envía los datos. 6. Sistema: Validación de los datos 7. Sistema: Realizar el cambio de contraseña 8. Sistema: Notificar al usuario el estado de la acción (ok / error)	
Extensiones: 5.1. El usuario no introduce datos bien formateados. Volver al paso 4. 5.2. Mostrar mensaje de error en los datos mal introducidos.	

UC7. Editar perfil	
Objetivo	Cambiar las preferencias y datos del usuario.
Actor	Usuario Cliente
Precondiciones: El usuario se encuentra correctamente autenticado	

Procedimiento: 1. Usuario: Visita YouMarket 2. Sistema: Procesa la petición solicitada 3. Usuario: Accede a su panel de preferencias 4. Sistema: Presenta el formulario con los datos editables 5. Usuario: Edita los datos y envía el formulario 6. Sistema: Validación de los datos 7. Sistema: Guardar los datos 8. Sistema: Notificar al usuario el estado de la acción (ok / error)
Extensiones: 5.1. El usuario no introduce datos bien formateados. Volver al paso 4. 5.2. Mostrar mensaje de error en los datos mal introducidos.

UC8. Desconexión	
Objetivo	Desconectar al cliente de la sesión actual.
Actor	Usuario Cliente
Precondiciones: El usuario se encuentra correctamente autenticado	
Procedimiento: 1. Usuario: Visita YouMarket 2. Sistema: Procesa la petición solicitada 3. Usuario: Hace click sobre la opción“ Desconectar” 4. Sistema: Procesa la petición y des-autentifica al usuario. 5. Sistema: Re-dirección a la página principal	

UC9. Añadir comentario producto	
Objetivo	Añadir un comentario personal en el detalle de un producto
Actor	Usuario Cliente
Precondiciones: El usuario se encuentra correctamente autenticado	

Procedimiento:

1. Usuario: Visita YouMarket
2. Sistema: Procesa la petición solicitada
3. Usuario: Visualiza el detalle de un producto
4. Sistema: Presenta los comentarios y el formulario para añadir uno nuevo
5. Usuario: Introduce el contenido del nuevo comentario y envía los datos
6. Sistema: Validación de los datos
7. Sistema: Procesar el mensaje
    - 7.1 Si contiene URLS → marcar para moderación
    - 7.2 Si contiene palabras reservadas → marcar para moderación
    - 7.3 Si el usuario tiene mensajes pendientes de moderación → marcar para moderación.
    - 7.4. Publicar el mensaje si todo es correcto
    - 7.5 Si el mensaje queda en moderación, notificar al administrador (e-mail)
8. Sistema: Notificar al usuario del estado del comentario (pendiente/publicado/error)

Extensiones:

5.1. Si el usuario es anónimo, al intentar introducir el comentario, mostrar un diálogo para invitarlo a identificarse o registrarse como cliente.

6.1. El usuario no introduce datos bien formateados. Volver al paso 4.

6.2. Mostrar mensaje de error en los datos mal introducidos.

UC10. Añadir valoración producto	
Objetivo	Añadir un valor de puntuación (1 al 5) sobre un producto
Actor	Usuario Cliente
Precondiciones: El producto no ha sido valorado por el usuario anteriormente, en ese caso, se modifica el valor en lugar de añadir.	

Procedimiento: 1. Usuario: Visita YouMarket 2. Sistema: Procesa la petición solicitada 3. Usuario: Visualiza el detalle de un producto 4. Sistema: Presenta la valoración actual 5. Usuario: Introduce su valoración 6. Sistema: Validación de los datos 7. Sistema: Procesar la valoración 8. Sistema: Notificar al usuario el estado de la valoración (ok / error)
Extensiones: 5.1. Si el usuario es anónimo, al intentar introducir el comentario, mostrar un diálogo para invitarlo a identificarse o registrarse como cliente. 5.2. Si el usuario dispone de valoración previa para ese producto, la acción será de edición 6.1. El usuario no introduce datos bien formateados. Volver al paso 4. 6.2. Mostrar mensaje de error en los datos mal introducidos.

UC11. Añadir favorito producto	
Objetivo	Añadir un producto al listado personal de favoritos.
Actor	Usuario Cliente
Precondiciones: El producto no ha sido favoritizado por el usuario anteriormente. En ese caso, la misma acción será de eliminar de los favoritos del usuario el producto.	

Procedimiento: 1. Usuario: Visita YouMarket 2. Sistema: Procesa la petición solicitada 3. Usuario: Visualiza el detalle de un producto 4. Sistema: Muestra el icono / botón para añadir a favoritos 5. Usuario: Hace click sobre el icono / botón para realizar la acción. 6. Sistema: Marca como favorito el producto solicitado 7. Sistema: Notificar al usuario el estado de la acción (ok / error)
Extensiones: 5.1. Si el usuario es anónimo, al intentar introducir el comentario, mostrar un diálogo para invitarlo a identificarse o registrarse como cliente. 5.2. Si el producto ya se encuentra en los favoritos del usuario, la acción será de des-marcar el producto cómo tal.

UC12. Seguir a Usuario	
Objetivo	Añadir otro usuario a la lista de “Seguidos” por el usuario
Actor	Usuario Cliente
Precondiciones: El usuario aún no sigue al usuario a seguir. En caso afirmativo, la misma acción será de eliminar de “Seguidos” al usuario visualizado. El usuario se encuentra correctamente autenticado.	
Procedimiento: 1. Usuario: Visita YouMarket 2. Sistema: Procesa la petición solicitada 3. Usuario: Accede al perfil público de un usuario 4. Sistema: Muestra el perfil público del usuario 5. Usuario: Hace click sobre el botón “Seguir” 6. Sistema: Procesa la petición y añade al usuario a la lista de “Seguidos” del usuario. 7. Sistema: Muestra un mensaje de confirmación del evento.	

Extensiones: 5.1. Si el usuario ya es seguidor, se mostrará la opción “Dejar de seguir” y realizará la opción correspondiente.

UC13. Seguir a Tienda	
Objetivo	Añadir una tienda a lista de “Seguidos” por el usuario
Actor	Usuario Cliente
Precondiciones: El usuario aún no sigue la tienda. En caso afirmativo, la misma acción será de eliminar de “Seguidos” la tienda visualizada. El usuario se encuentra correctamente autenticado	
Procedimiento: 1. Usuario: Visita YouMarket 2. Sistema: Procesa la petición solicitada 3. Usuario: Accede al perfil de una tienda 4. Sistema: Muestra el perfil público de la tienda 5. Usuario: Hace click sobre el botón “Seguir” 6. Sistema: Procesa la petición y añade la tienda a la lista de “Seguidos” del usuario. 7. Sistema: Muestra un mensaje de confirmación del evento.	
Extensiones: 5.1. Si el usuario ya es seguidor, se mostrará la opción “Dejar de seguir” y realizará la opción correspondiente.	

UC14. Listado de pedidos realizados	
Objetivo	El usuario visualiza el histórico de sus pedidos realizados y el estado de los mismos.
Actor	Usuario Cliente
Precondiciones: El usuario se encuentra correctamente autenticado.	

Procedimiento: 1. Usuario: Visita YouMarket 2. Sistema: Procesa la petición solicitada 3. Usuario: Accede a su panel de configuración, menú Pedidos 4. Sistema: Muestra el listado de pedidos realizados.
Extensiones: 4.1. Sistema: En el caso de no existir pedidos, el sistema informará de ello con un mensaje.

UC15. Ver detalle pedido	
Objetivo	El usuario visualiza el detalle de un pedido realizado.
Actor	Usuario Cliente
Precondiciones: El usuario se encuentra correctamente autenticado.	
Procedimiento: 1. Usuario: Visita YouMarket 2. Sistema: Procesa la petición solicitada 3. Usuario: Accede a su panel de configuración, menú Pedidos 4. Sistema: Muestra el listado de pedidos realizados. 5. Usuario: Accede al detalle de un pedido desde el listado. 6. Sistema: Muestra la información del pedido solicitado.	
Extensiones: 4.1. Sistema: En el caso de no existir pedidos, el sistema informará de ello con un mensaje.	

12.6. Casos de uso usuario Tienda

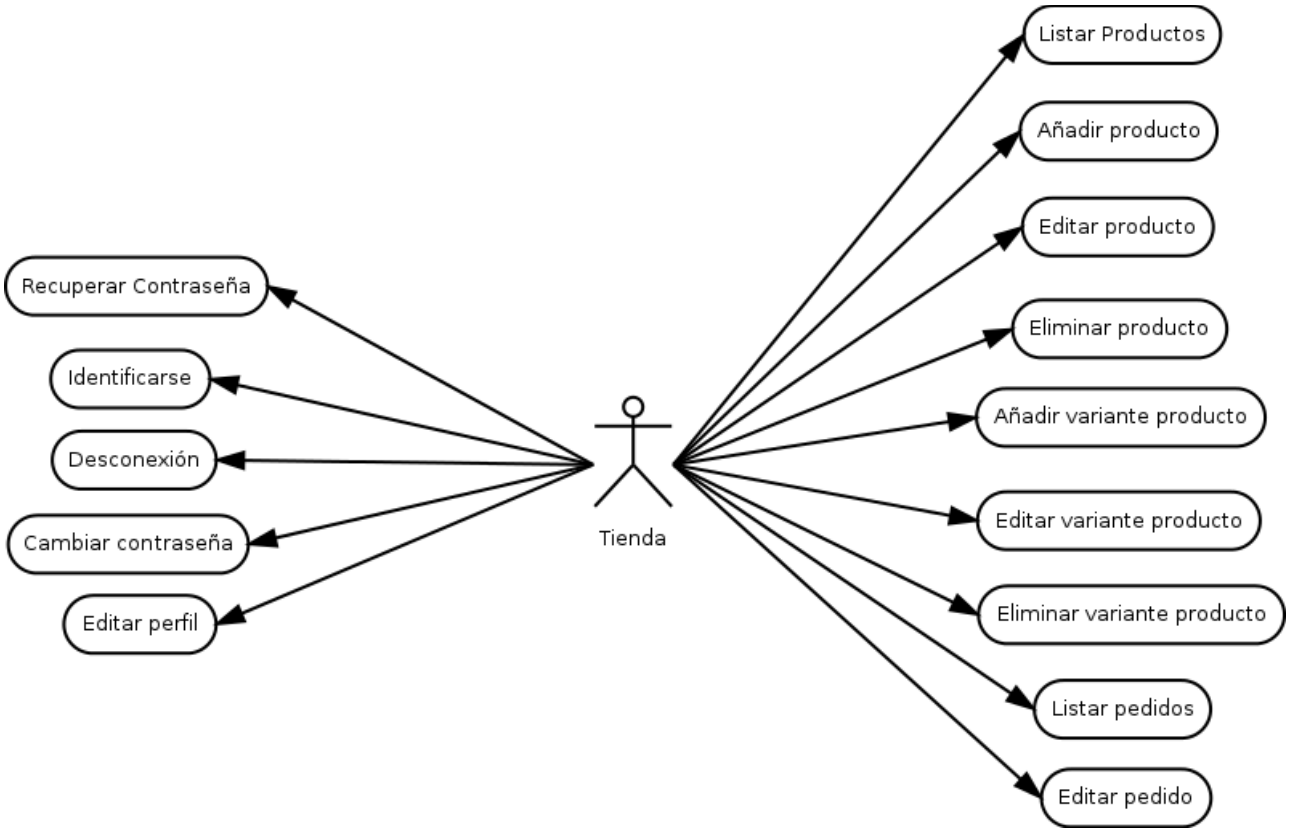


Ilustración 52: Diagrama casos de uso del usuario tipo tienda

UC16. Listar Productos	
Objetivo	Visualizar los productos de los que dispone el usuario dentro de la plataforma.
Actor	Usuario Tienda
Precondiciones: El usuario se encuentra correctamente autenticado.	
Procedimiento: 1. Usuario: Accede al panel de configuración 2. Sistema: Muestra las opciones 3. Usuario: Selecciona la opción Productos 4. Sistema: Muestra los productos del usuario	

Extensiones: 4.1. Sistema: En el caso de no existir productos, el sistema informará de ello con un mensaje.

UC17. Añadir Producto	
Objetivo	Añadir un producto a la tienda
Actor	Usuario Tienda
Precondiciones: El usuario se encuentra correctamente autenticado.	
Procedimiento: 1. Usuario: Accede al panel de configuración 2. Sistema: Muestra las opciones 3. Usuario: Selecciona la opción Productos 4. Sistema: Muestra los productos del usuario 5. Usuario: Pulsa la opción “Añadir” 6. Sistema: Muestra el formulario para añadir un nuevo producto. 7. Usuario: Rellena los datos y envía el formulario 8. Sistema: Validación de los datos 9. Sistema: Añade el producto al sistema 10. Sistema: Muestra el listado de productos y un mensaje de confirmación	
Extensiones: 4.1. Sistema: En el caso de no existir productos, el sistema informará de ello con un mensaje. 8.1. El usuario no introduce datos bien formateados. Volver al paso 6. 8.2. Mostrar mensaje de error en los datos mal introducidos.	

UC18. Editar Producto	
Objetivo	Editar un producto de la tienda
Actor	Usuario Tienda
Precondiciones: El usuario se encuentra correctamente autenticado.	

Procedimiento: 1. Usuario: Accede al panel de configuración 2. Sistema: Muestra las opciones 3. Usuario: Selecciona la opción Productos 4. Sistema: Muestra los productos del usuario 5. Usuario: Pulsa la opción “Editar” del producto deseado 6. Sistema: Muestra el formulario para modificar los datos del producto. 7. Usuario: Rellena los datos y envía el formulario 8. Sistema: Validación de los datos 9. Sistema: Edita el producto del sistema 10. Sistema: Muestra el listado de productos y un mensaje de confirmación.
Extensiones: 4.1. Sistema: En el caso de no existir productos, el sistema informará de ello con un mensaje. 8.1. El usuario no introduce datos bien formateados. Volver al paso 6. 8.2. Mostrar mensaje de error en los datos mal introducidos.

UC19. Eliminar Producto	
Objetivo	Eliminar un producto de la tienda
Actor	Usuario Tienda
Precondiciones: El usuario se encuentra correctamente autenticado.	

Procedimiento: 1. Usuario: Accede al panel de configuración 2. Sistema: Muestra las opciones 3. Usuario: Selecciona la opción Productos 4. Sistema: Muestra los productos del usuario 5. Usuario: Pulsa la opción “Eliminar” del producto deseado 6. Sistema: Muestra un mensaje de confirmación 7. Usuario: Acepta la acción de eliminar 8. Sistema: Elimina el producto del sistema 9. Sistema: Muestra el listado de productos y un mensaje de confirmación.
Extensiones: 4.1. Sistema: En el caso de no existir productos, el sistema informará de ello con un mensaje. 7.1. El usuario no acepta la eliminación, se vuelve al paso 4.

UC20. Añadir variante de productos	
Objetivo	Añadir una referencia al producto.
Actor	Usuario Tienda
Precondiciones: El usuario se encuentra correctamente autenticado.	

Procedimiento: 1. Usuario: Accede al panel de configuración 2. Sistema: Muestra las opciones 3. Usuario: Selecciona la opción Productos 4. Sistema: Muestra los productos del usuario 5. Usuario: Pulsa la opción “Editar Variantes” del producto deseado 6. Sistema: Muestra el listado de variantes del producto asociados. 7. Usuario: Pulsa la opción “Añadir” del producto deseado 8. Sistema: Muestra el formulario para añadir variantes al producto. 9. Usuario: Rellena los datos y envía el formulario 10. Sistema: Validación de los datos 11. Sistema: Añade la variante del producto al sistema. 12. Sistema: Muestra el listado de variantes del producto y un mensaje de confirmación.
Extensiones: 4.1. Sistema: En el caso de no existir productos, el sistema informará de ello con un mensaje. 10.1. El usuario no introduce datos bien formateados. Volver al paso 8. 10.2. Mostrar mensaje de error en los datos mal introducidos.

UC21. Editar variante de producto	
Objetivo	Editar una referencia del producto.
Actor	Usuario Tienda
Precondiciones: El usuario se encuentra correctamente autenticado.	

Procedimiento: 1. Usuario: Accede al panel de configuración 2. Sistema: Muestra las opciones 3. Usuario: Selecciona la opción Productos 4. Sistema: Muestra los productos del usuario 5. Usuario: Pulsa la opción “Editar Variantes” del producto deseado 6. Sistema: Muestra el listado de variantes del producto asociados. 7. Usuario: Pulsa la opción “Editar” de la variante de producto deseada 8. Sistema: Muestra el formulario para editar variantes del producto. 9. Usuario: Rellena los datos y envía el formulario 10. Sistema: Validación de los datos 11. Sistema: Edita la variante del producto del sistema. 12. Sistema: Muestra el listado de variantes del producto y un mensaje de confirmación.
Extensiones: 4.1. Sistema: En el caso de no existir productos, el sistema informará de ello con un mensaje. 10.1. El usuario no introduce datos bien formateados. Volver al paso 8. 10.2. Mostrar mensaje de error en los datos mal introducidos.

UC22. Eliminar variante de producto	
Objetivo	Eliminar una referencia del producto
Actor	Usuario Tienda
Precondiciones: El usuario se encuentra correctamente autenticado.	

Procedimiento: 1. Usuario: Accede al panel de configuración 2. Sistema: Muestra las opciones 3. Usuario: Selecciona la opción Productos 4. Sistema: Muestra los productos del usuario 5. Usuario: Pulsa la opción “Editar Variantes” del producto deseado 6. Sistema: Muestra el listado de variantes del producto asociados. 7. Usuario: Pulsa la opción “Eliminar” de la variante de producto deseada. 8. Sistema: Muestra un mensaje de confirmación 9. Usuario: Acepta la acción de eliminar 10. Sistema: Elimina la variante de producto del sistema 11. Sistema: Muestra el listado de variantes del producto y un mensaje de confirmación.
Extensiones: 4.1. Sistema: En el caso de no existir productos, el sistema informará de ello con un mensaje. 9.1. El usuario no acepta la eliminación, se vuelve al paso 6.

UC23. Listar pedidos	
Objetivo	Obtener un listado de los pedidos que han sido realizados en la tienda.
Actor	Usuario Tienda
Precondiciones: El usuario se encuentra correctamente autenticado.	
Procedimiento: 1. Usuario: Accede al panel de configuración. 2. Sistema: Muestra las opciones. 3. Usuario: Selecciona la opción Pedidos. 4. Sistema: Muestra los pedidos realizados a la tienda del usuario.	
Extensiones: 4.1. Sistema: En el caso de no existir pedidos, el sistema informará de ello con un mensaje.	

UC24. Editar pedido	
Objetivo	Modificar el estado de un pedido.
Actor	Usuario Tienda
Precondiciones: El usuario se encuentra correctamente autenticado.	
Procedimiento: 1. Usuario: Accede al panel de configuración. 2. Sistema: Muestra las opciones. 3. Usuario: Selecciona la opción Pedidos. 4. Sistema: Muestra los pedidos realizados a la tienda del usuario. 5. Usuario: Pulsa la opción “Editar” del pedido deseado 6. Sistema: Muestra el formulario para modificar los datos del pedido. 7. Usuario: Rellena los datos y envía el formulario 8. Sistema: Validación de los datos 9. Sistema: Edita el pedido del sistema 10. Sistema: Muestra el listado de pedidos y un mensaje de confirmación.	
Extensiones: 4.1. Sistema: En el caso de no existir pedidos, el sistema informará de ello con un mensaje. 8.1. El usuario no introduce datos bien formateados. Volver al paso 6. 8.2. Mostrar mensaje de error en los datos mal introducidos.	

12.7. Casos de uso usuario Administrador

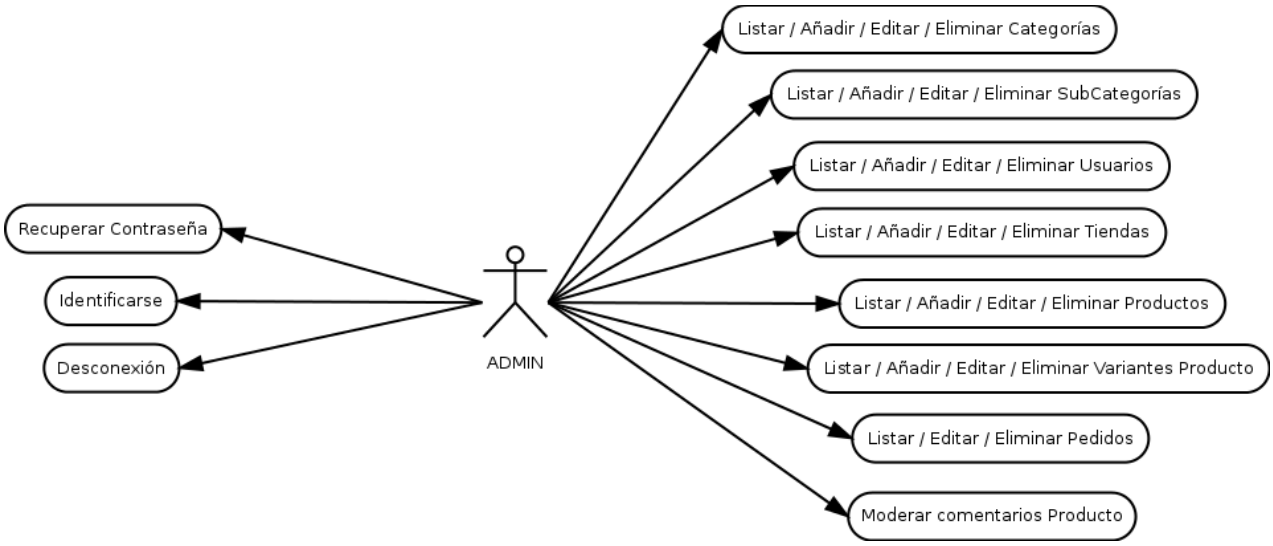


Ilustración 53: Diagrama casos de uso del usuario administrador

Para simplificar la documentación de este apartado, se ha defino ENTIDAD para referir a Categorías, SubCategorías, Usuarios, Tiendas, Productos, Variantes Productos y Pedidos. El funcionamiento de los casos de uso de este Actor es idéntico para cada elemento listado.

UC25. Listar [ENTIDAD]	
Objetivo	Obtener un listado de todos los elementos de la [ENTIDAD]
Actor	Usuario Administrador
Precondiciones: - El usuario se encuentra correctamente autenticado. - El usuario se encuentra en la página principal del Backend.	
Procedimiento: 1. Usuario: Accede al listado de [ENTIDAD] 2. Sistema: Muestra el listado de elementos de [ENTIDAD] disponibles.	

Extensiones: 2.1 Sistema: Si no hay elementos disponibles, se notifica al usuario. 3. Usuario: Realizar una búsqueda por un atributo de [ENTIDAD] 4. Usuario: Realizar una ordenación de elementos por un atributo de [ENTIDAD] mostrada. 5. Usuario: Puede volver al listado, editar un elemento o crear uno nuevo.

UC26. Añadir [ENTIDAD]	
Objetivo	Crear un nuevo elemento del tipo [ENTIDAD]
Actor	Usuario Administrador
Precondiciones: - El usuario se encuentra correctamente autenticado. - El usuario se encuentra en la página principal del Backend.	
Procedimiento: 1. Usuario: Accede al listado de [ENTIDAD] 2. Sistema: Muestra el listado de elementos de [ENTIDAD] disponibles. 3. Usuario: Pulsa la opción “Añadir” 4. Sistema: Muestra el formulario para introducir los datos del nuevo elemento. 5. Usuario: Rellena los datos y envía el formulario 6. Sistema: Validación de los datos 7. Sistema: Muestra el listado de elementos de [ENTIDAD] disponibles. 8. Sistema: Mostrar mensaje de confirmación.	
Extensiones: 2.1. Sistema: Si no hay elementos disponibles, se notifica al usuario. 6.1. El usuario no introduce datos bien formateados. Volver al paso 4. 6.2. Mostrar mensaje de error en los datos mal introducidos. 7.1. El usuario puede elegir volver al listado o crear un nuevo elemento tras la inserción.	

UC27. Editar [ENTIDAD]

Objetivo	Modificar atributos de [ENTIDAD]
Actor	Usuario Administrador
Precondiciones: - El usuario se encuentra correctamente autenticado. - El usuario se encuentra en la página principal del Backend.	
Procedimiento: 1. Usuario: Accede al listado de [ENTIDAD] 2. Sistema: Muestra el listado de elementos de [ENTIDAD] disponibles. 3. Usuario: Pulsa la opción “Editar” 4. Sistema: Muestra el formulario para editar los datos del elemento. 5. Usuario: Rellena los datos y envía el formulario 6. Sistema: Validación de los datos 7. Sistema: Muestra el listado de elementos de [ENTIDAD] disponibles. 8. Sistema: Mostrar mensaje de confirmación.	
Extensiones: 2.1. Sistema: Si no hay elementos disponibles, se notifica al usuario. 6.1. El usuario no introduce datos bien formateados. Volver al paso 4. 6.2. Mostrar mensaje de error en los datos mal introducidos.	

UC28. Eliminar [ENTIDAD]	
Objetivo	Dar de bajo un elemento del tipo [ENTIDAD]
Actor	Usuario Administrador
Precondiciones: - El usuario se encuentra correctamente autenticado. - El usuario se encuentra en la página principal del Backend.	

Procedimiento: 1. Usuario: Accede al listado de [ENTIDAD] 2. Sistema: Muestra el listado de elementos de [ENTIDAD] disponibles. 3. Usuario: Pulsa la opción “Editar” 4. Sistema: Muestra el formulario para editar los datos del elemento. 5. Usuario: Pulsa el botón “Borrar” 6. Sistema: Muestra un mensaje de confirmación de la acción 7. Usuario: Pulsa la opción “Sí, borrar”
Extensiones: 2.1. Sistema: Si no hay elementos disponibles, se notifica al usuario. 7.1 Si el usuario pulsa la opción “Editar”, se vuelve al paso 4.

12.8. Manual de usuario tienda (comercio)

12.8.1. Acceso al panel de tienda

Para acceder a la cuenta de comercio si no se encuentra ya autentificado se debe realizar por vía del menú habilitado para ello en la parte superior de Youmarket.

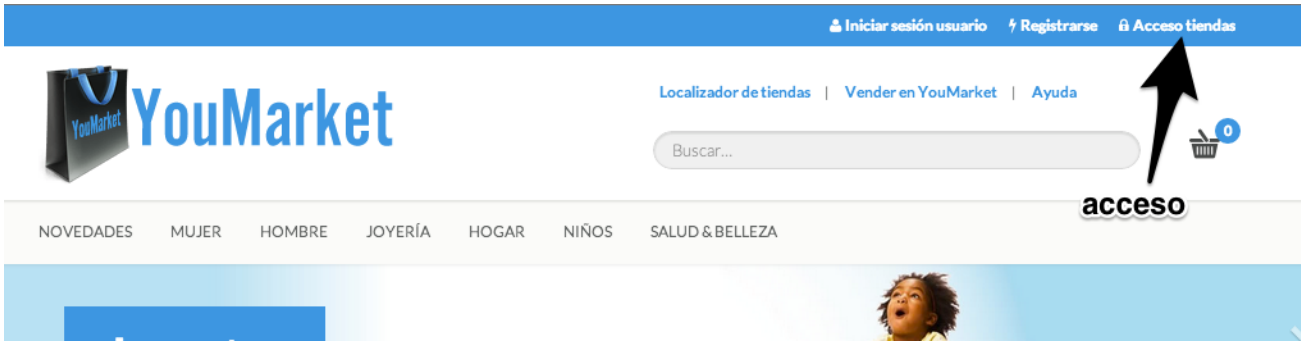


Ilustración 54: Acceso Tiendas

Se presenta el formulario para el inicio de sesión como usuario tienda:

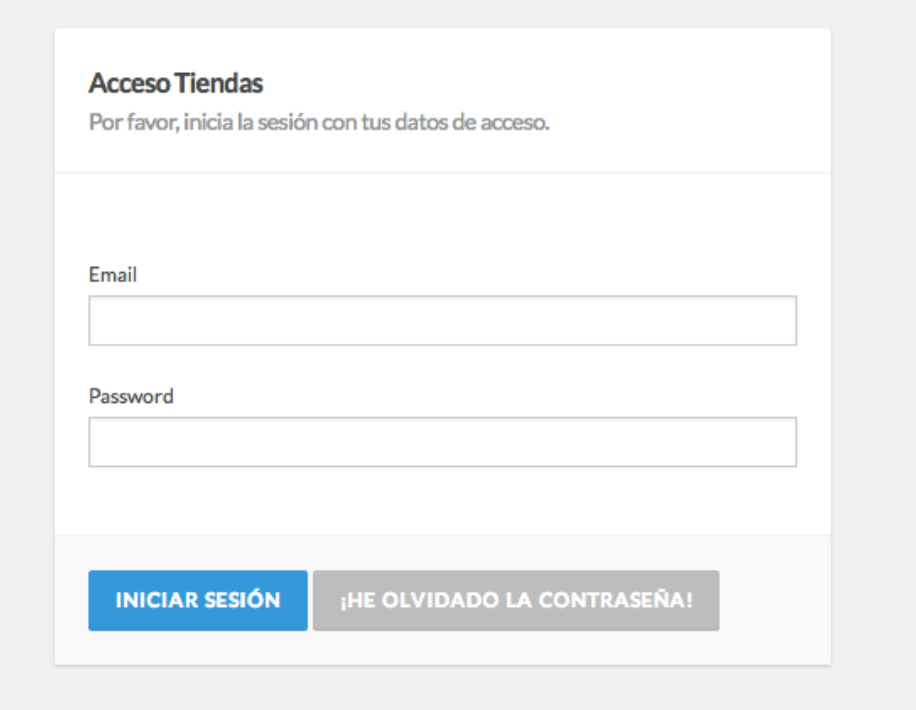


Ilustración 55: Formulario para iniciar sesión. Usuario Tienda

En el caso de no recordar la contraseña de acceso, se puede generar una nueva contraseña de acceso que será enviada al correo electrónico de la cuenta (Ilustración 56) pulsando el botón "¡He olvidado la contraseña!" del formulario de acceso (Ilustración 55).

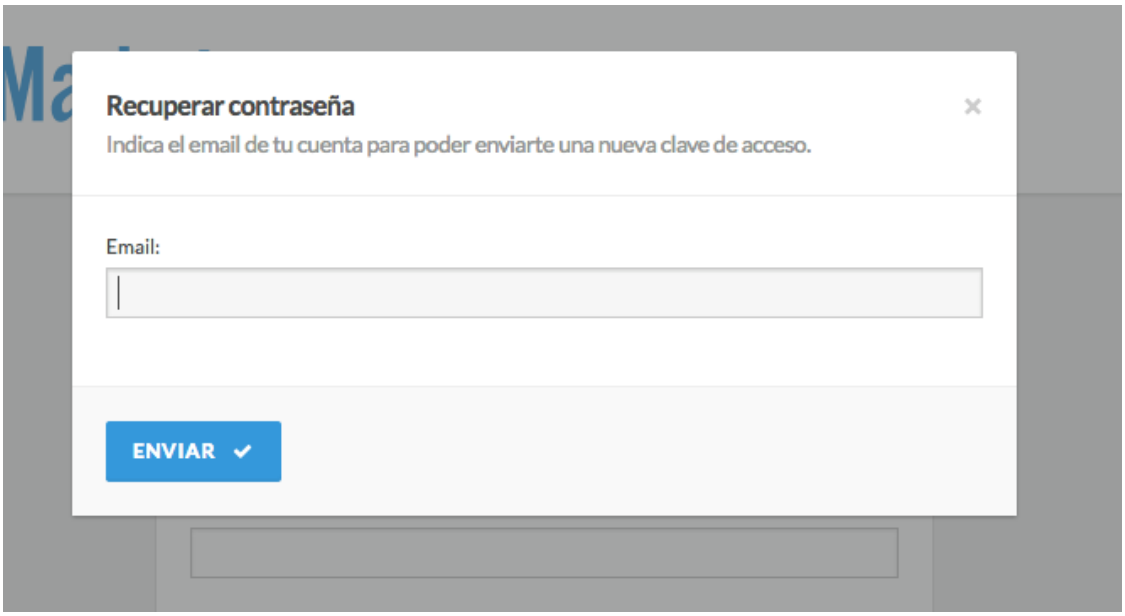


Ilustración 56: Recuperación de contraseña

Si ya se encuentra el usuario autentificado, la parte superior de YouMarket muestra el nombre de la cuenta (haciendo click sobre el nombre se accede a la cuenta de la tienda) y la opción de desconectarse de la sesión actual.

12.8.2. Elementos disponibles y dashboard

Tras acceder a la cuenta Tienda, se presenta la página principal de la cuenta (dashboard). En el dashboard se muestra información relevante de la cuenta: estadísticas de visitas a los productos del comercio, número totales de ventas, etc. Sin embargo, actualmente estas funcionalidades no se encuentran aún disponibles.

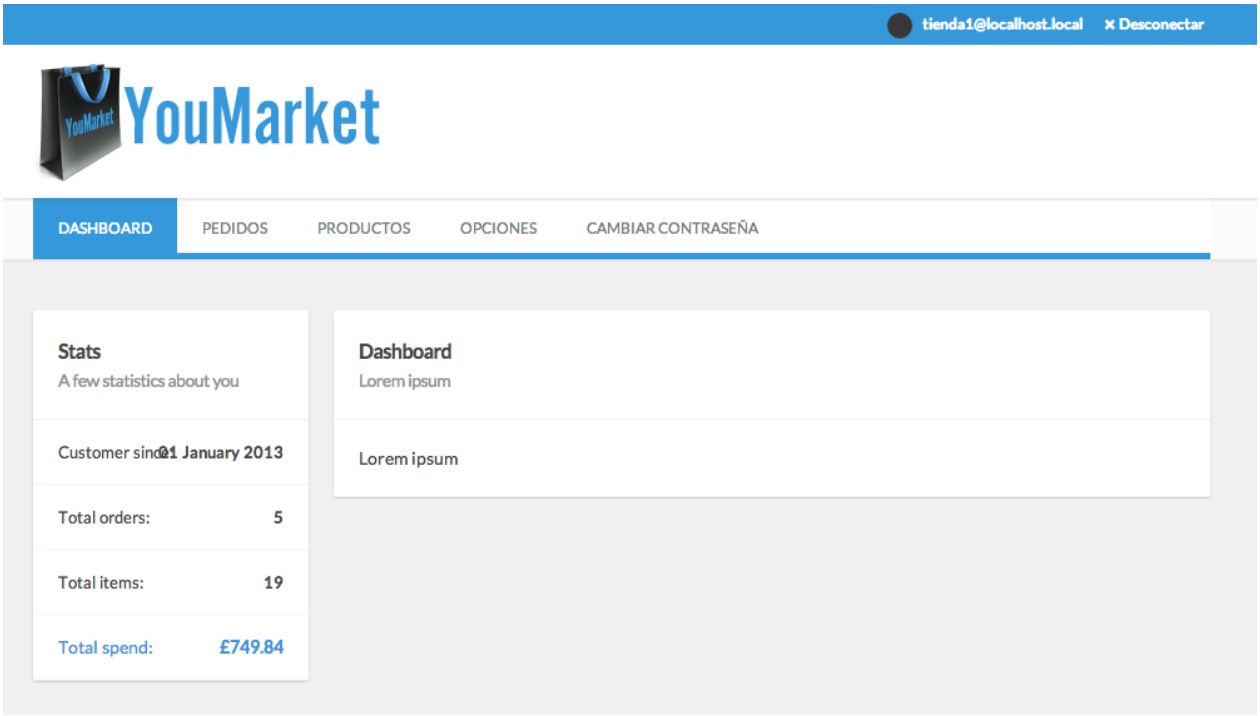


Ilustración 57: Dashboard usuario Tienda

La navegación del panel de usuario Tienda se basa en un menú desde donde se pueden acceder a las diferentes secciones: Dashboard, Pedidos, Productos, Opciones y Cambiar Contraseña.

12.8.3. Cambiar opciones de la tienda y de la cuenta

Se pueden modificar varios parámetros de la cuenta, accediendo para ello vía el menú

en el apartado “Opciones”.

Ilustración 58: Formulario opciones de Tienda

Una función interesante es la posibilidad de geoposicionar la tienda en un mapa. Se puede mover el “pin” arrastrando y soltando sobre la posición deseada y también gracias al botón “Localizar desde dirección” el pin se colocará automáticamente sobre la dirección introducida en los campos de dirección superiores (Ilustración 59).

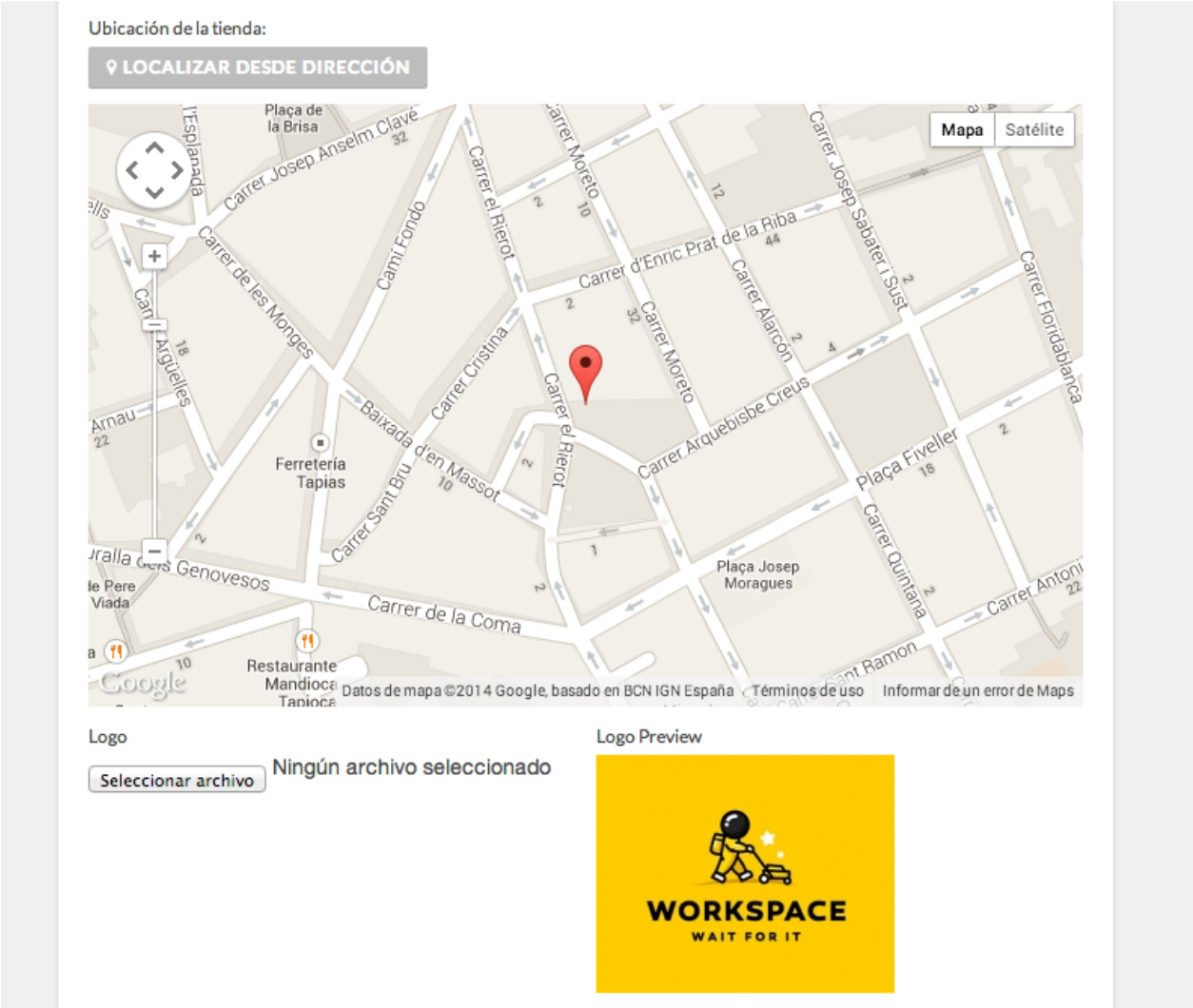


Ilustración 59: Localización de tienda en mapa

12.8.4. Cambiar contraseña de la cuenta

Accediendo a la opción del menú “Cambiar Contraseña” se puede modificar la contraseña actual de la cuenta ingresando previamente la actual antes de introducir la nueva contraseña.

12.8.5. Gestión de productos

Para gestionar los productos de la tienda, se debe acceder a la opción “Productos” del menú de la cuenta.

Inicialmente se presenta un listado de los productos dados de alta en YouMarket (Ilustración 60).

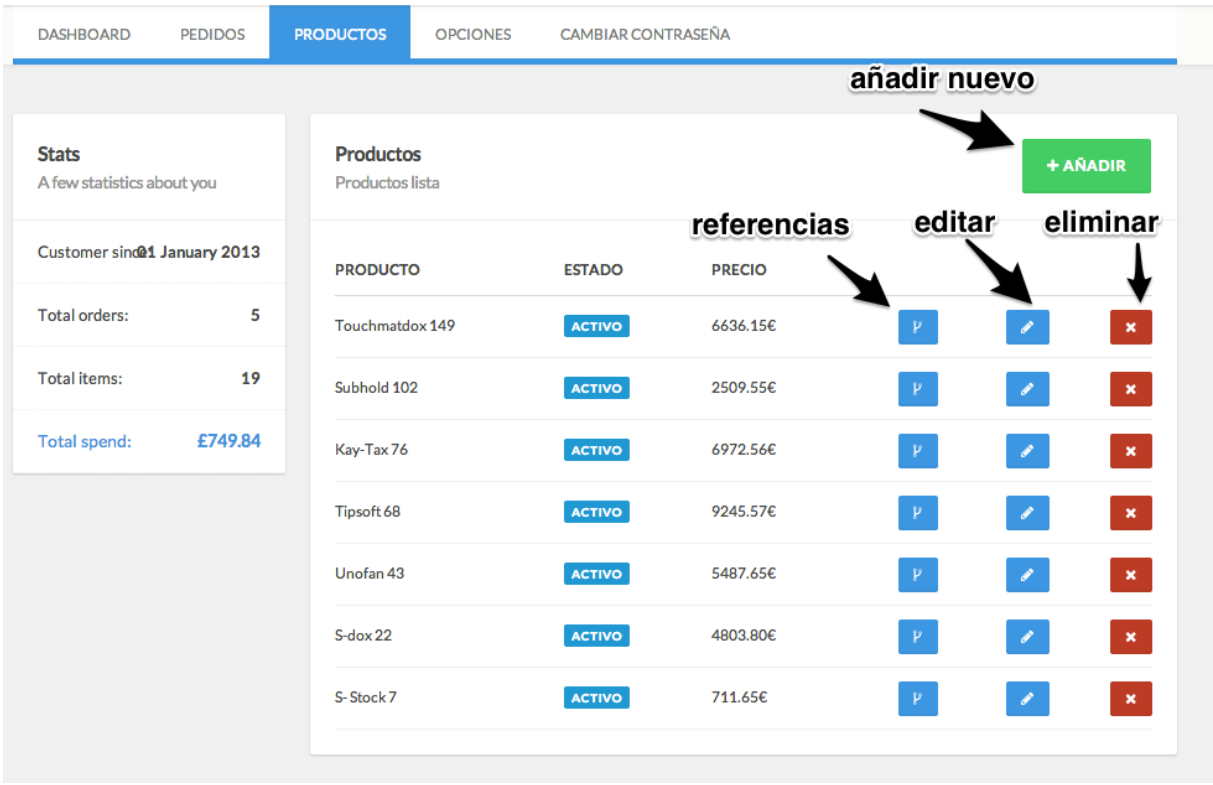


Ilustración 60: Listado de productos

Para añadir un nuevo producto se realiza haciendo click el botón "Añadir". El formulario de alta de nuevo producto nos permite configurar todos los detalles del producto y adjuntar hasta 5 imágenes asociadas a él (Ilustración 61).

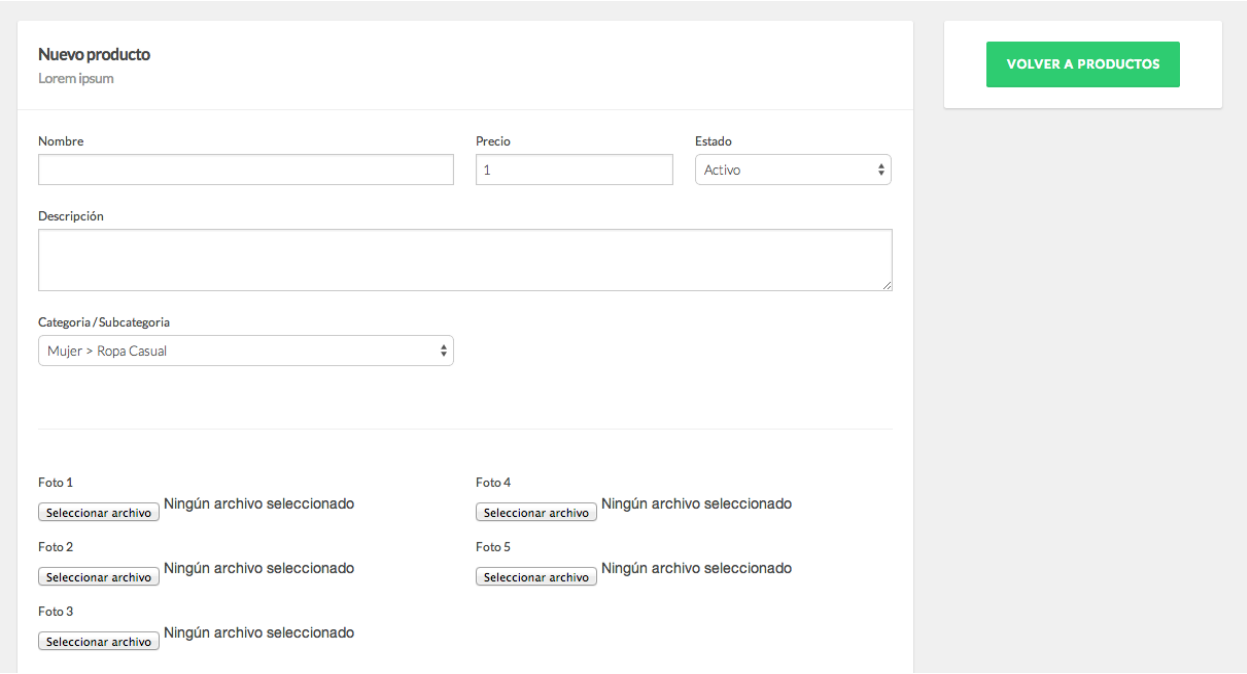


Ilustración 61: Formulario de nuevo producto

La opción “referencias” (variantes) nos permite acceder a las referencias que hemos dado de alta para el producto (Ilustración 60). Cada producto puede contener de una a más referencias asociadas. Esto permite para un mismo producto, poder mostrarse en la tienda con diferentes tallas y colores por ejemplo. Cada variante está definida por un nombre, SKU (código), un stock actual y un stock máximo.

12.8.6. Gestión de pedidos

Desde la opción del menú “Pedidos” se gestionan los pedidos realizados a la tienda. La gestión se basa en un listado y la posibilidad de editar el estado de lo mismos. Cada cambio de estado notificará al comprador de este hecho. En cada pedido también se informa del estado del pago por parte del comprador.

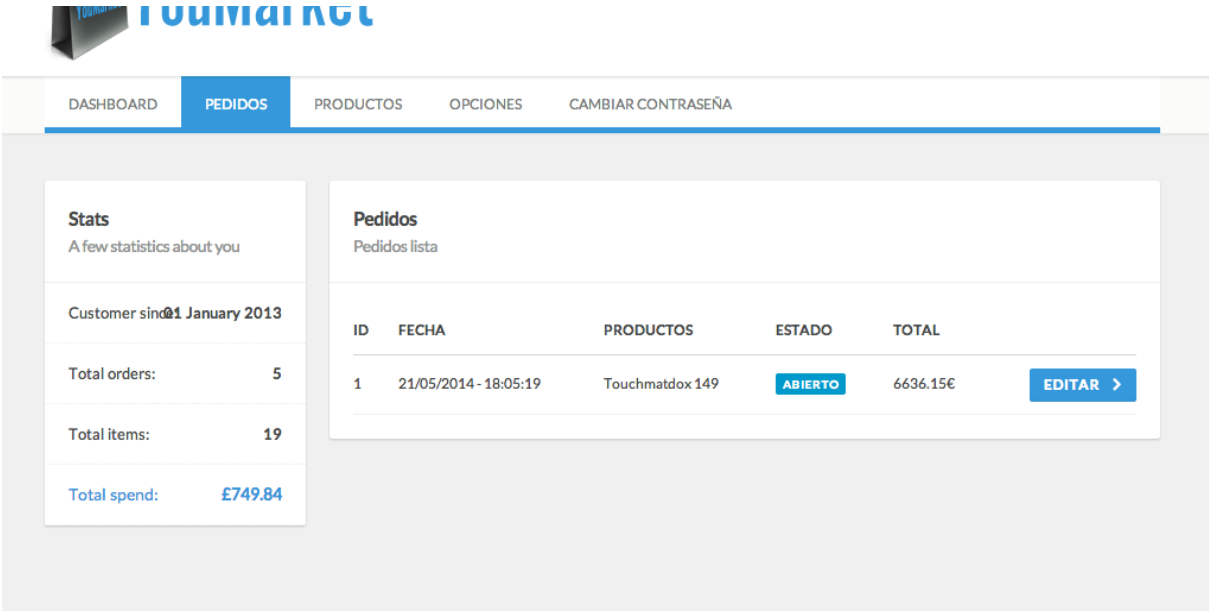


Ilustración 62: Listado de pedidos realizados a la tienda.

12.9. Manual de usuario administrador (backend)

12.9.1. Acceso al backend

El acceso al backend se realiza directamente accediendo a la siguiente URL: http://youmarket.elysium.es/backend/

El sistema nos pide un usuario y contraseña para autenticarnos antes de acceder al

backend (Ilustración 63).

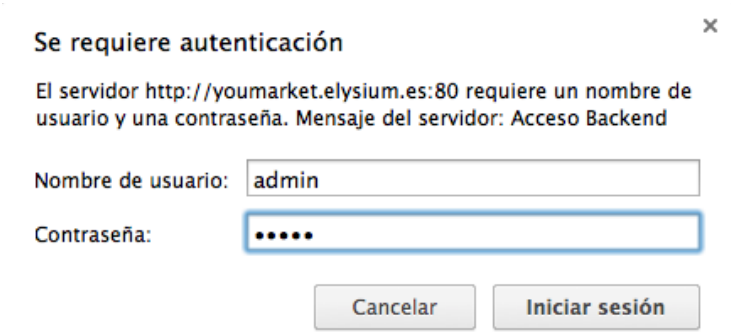


Ilustración 63: Formulario de ingreso usuario / contraseña del backend

Una vez autorizados, se muestra la página principal del backend (Ilustración 64).

Ilustración 64: Página principal backend

12.9.2. Página principal y descripción de los controles

Desde la página principal del backend (Ilustración 64) se accede a todas las entidades editables de la aplicación. Las acciones principales son Listar y Agregar nuevo.

12.9.3. Listar y editar registros

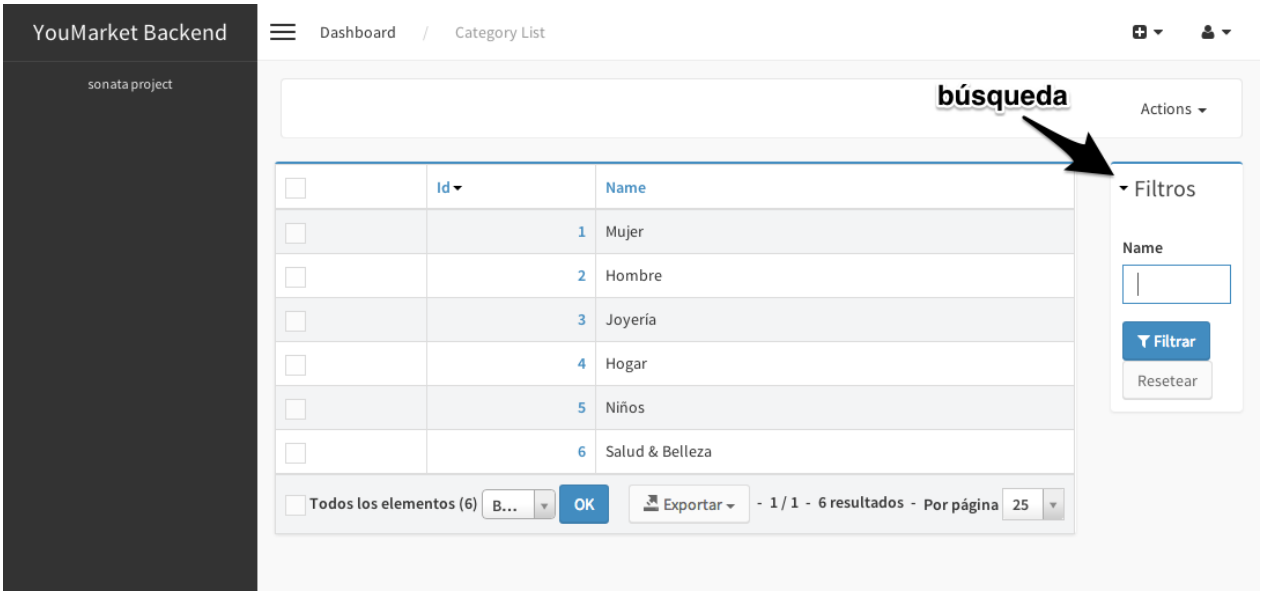


Ilustración 65: Listado de la entidad Category

En la ilustración 65 se muestra un listado del backend. Se puede filtrar y ordenar directamente los registros obtenidos.

Al final de la tabla se puede configurar el número de resultados por página y exportar los datos en diferentes formatos.

Para editar un elemento es necesario hacer click en la fila del registro que se desea editar para visualizar el formulario de edición del registro.

12.9.4. Añadir un nuevo registro

Para agregar un nuevo registro se puede realizar directamente desde la página principal del backend o desde la opción “Actions” de un listado (Ilustración 66).

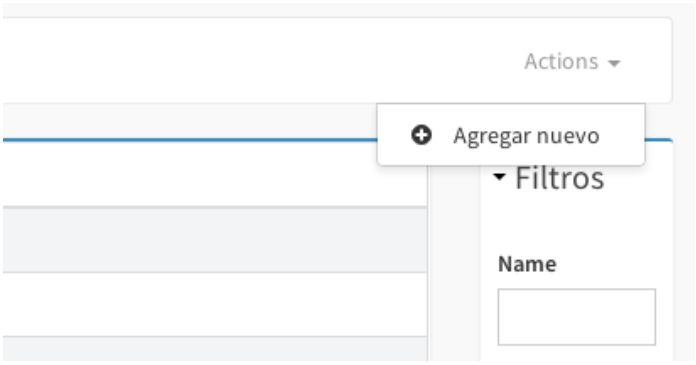


Ilustración 66: Detalle de "Actions" en un listado

El formulario para añadir un nuevo registro se presenta como el ejemplo de la ilustración 67.

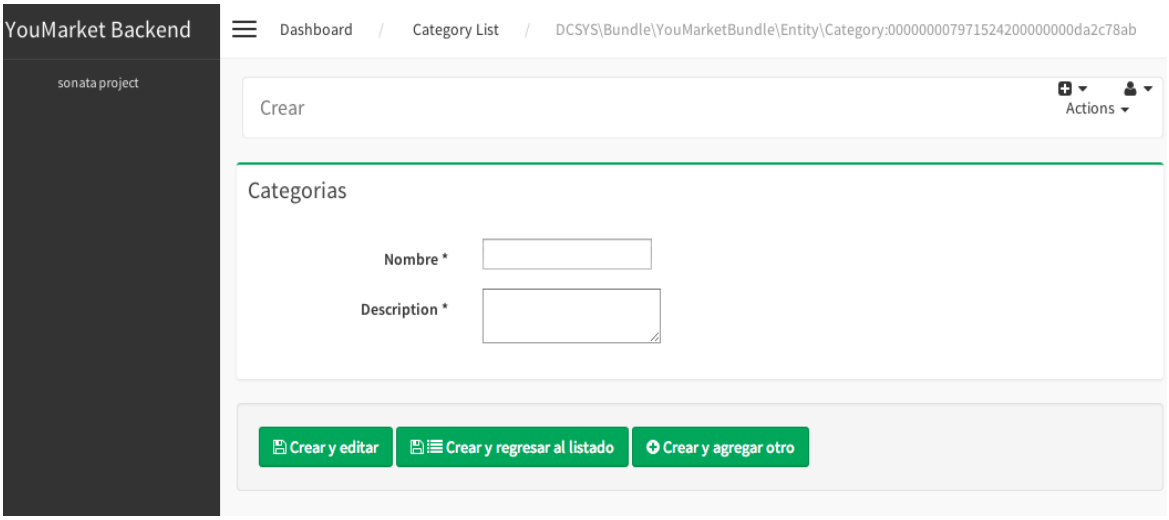


Ilustración 67: Formulario para añadir una categoría

En cada formulario de alta existen tres acciones disponibles: Crear y editar, Crear y regresar al listado, Crear y agregar otro.