



Centro Profesional
Universidad Europea Madrid
LAUREATE INTERNATIONAL UNIVERSITIES

**ESCUELA
POLITÉCNICA**

UNIVERSIDAD EUROPEA DE MADRID



ESCUELA POLITÉCNICA

**CICLO FORMATIVO DE GRADO SUPERIOR DESARROLLO DE
APLICACIONES MULTIPLATAFORMA**

PROYECTO FIN DE CICLO

Proyecto Craftandbudget

Autores

Nicolas Alphonse Bertoncello

Issam Natour Muñoz

CURSO 2014-15

TÍTULO: Proyecto Craftandbudget

AUTORES: ISSAM NATOUR MUÑOZ Y NICOLAS ALPHONSE
BERTONCELLO

TUTOR DEL PROYECTO: NOMBRE APELLIDO1 APELLIDO2

FECHA DE LECTURA: 8 de Junio de 2013

CALIFICACIÓN:

Fdo: NOMBRE APELLIDO1 APELLIDO2

Tutor/a del Proyecto

Resumen

Craft&budget es una web app destinada para que aquellos artesanos que tengan una tienda online en el CMS prestashop puedan manejar el stock de su tienda. Podrán tanto manejar el stock de los productos que tengan en venta, como el stock de los materiales que usan para crear sus productos. Con craft and budget podrán además gestionar la creación de sus productos sabiendo cuantos materiales necesitan para crear sus productos y cuántos productos pueden crear con los materiales de los que disponen. Finalmente también dispondrán de una libreta de dirección para almacenar todos sus proveedores.

El stock de los materiales no se encuentra en la base de datos de prestashop, sino que los artesanos tendrán que crearlo con los materiales de que dispongan materiales.

La primera versión se conectará con prestashop únicamente, pero en un futuro podrá conectarse con otras tiendas online. Para conectarse, craft and budget se comunica con prestashop mediante una api rest por la cual se descargará todos los productos que el artesano tiene en la tienda. Usaremos una base de datos mysql para guardar todos los productos y los materiales.

Abstract

Agradecimientos

Issam Natour: Le doy las gracias a mi novia Ineta Mikelionyte, Artesana y propietaria de la tienda artesanal online buhoplace.es, por darnos la idea que nos empujó a crear este proyecto.

Nicolas Alphonse: Mucho trabajo duro detrás de este documento. Se lo dedico a mi familia y amigos, sin ellos no habría sido capaz.



Esta obra se distribuye bajo una licencia Creative Commons.

Se permite la copia, distribución, uso y comunicación de la obra si se respetan las

siguientes condiciones:

- Se debe reconocer explícitamente la autoría de la obra incluyendo esta nota y su
- enlace.
- La copia será literal y completa
- No se podrá hacer uso de los derechos permitidos con fines comerciales, salvo permiso expreso de los autores.

El texto precedente no es la licencia completa sino una nota orientativa de la licencia original completa(jurídicamente válida) que puede encontrarse en:
<http://creativecommons.org/licenses/by-nc-nd/3.0/deed.es>

<http://creativecommons.org/licenses/by-nc-nd/3.0/deed.es>

Contenido

Resumen.....	4
Abstract	4
Agradecimientos	6
Introducción	10
Objetivos	12
Entender la importancia de escoger una tecnología back-end apropiada.....	14
Como realizar un página web responsive	16
Alternativas de web hosting, hacer una aplicación SEO friendly...	16
Motivación.....	16
Antecedentes	17
Desarrollo de la práctica.....	18
Material.....	18
PHP y MySQL.....	18
Javascript y JQuery.	19
Ajax.....	20
Framework codeigniter php	21
Twitter bootstrap.....	24
Api rest	24
Web api Prestashop	25
Planificación	27
Metodología de gestión usada.....	28
Software de gestión usado	29
Descripción del trabajo realizado.....	30
Casos de Uso	30
Codeigniter	32
Web Api Prestashop.....	34
Base de datos MySql.....	37
Resultados y validación (W3Techs - World Wide Web Technology Surveys)	43

Conclusiones	43
Aportaciones	43
Trabajo futuro	44
Apéndices	44
Apéndice A: Product Backlog	45
Apéndice B: Sprint Semanales	46
Bibliografía y web gráfica	50

Introducción

Este proyecto surge en primer lugar y de alguna manera, de la necesidad de una solución web multiplataforma para aquellos artesanos que venden sus productos a través de Prestashop. La intención, es facilitar el manejo de su

stock de materiales de una manera cómoda y sencilla, y que además, esté sincronizada con su tienda eCommerce.

En nuestra aplicación web, los artesanos podrían tanto añadir como quitar materiales de su inventario, dependiendo de si fabrican o compran materiales para sus productos. También, podrán crear una lista de contactos de sus proveedores para poder tenerlos a mano, cuando los necesiten.

Prestashop es un CRM Open Source, de fácil uso, para la creación de tiendas online. Actualmente se estima que hay en la red alrededor de 125000 tiendas online que usan Prestashop. Además, se prevé una tendencia al crecimiento de esta plataforma debido a su popularidad.

Por otro lado, esta plataforma tiene un funcionamiento sencillo, y no requiere de conocimientos demasiado avanzados. Asimismo, tiene un target principal de tiendas eCommerce con estructura y tamaño de PYMES, siendo así, su disposición de productos en 500/1000 unidades a la venta.

Sin lugar a dudas, el terreno del comercio electrónico descrito como área de estudio, son los motivos por los que en un principio nos decidimos a crear un módulo para Prestashop.

Iniciando el desarrollo de un módulo, el equipo de trabajo se encontró con ciertas desventajas. El principal problema que nos encontramos durante el desarrollo de dicho módulo es la escasa documentación existente sobre cómo hacerlo, es decir, ciertas pautas para desarrolladores de manera autónoma o autodidacta. Ya que, aunque Prestashop es Open Source, la manera de generar ingresos de esta plataforma se fracciona de dos formas: mediante servicios de hosting y mediante cursos para que los usuarios aprendan cómo usarlo y que los desarrolladores aprendan cómo funciona.

Fue en ese instante, cuando **decidimos que en vez de crear un módulo deberíamos crear una web app** que se comunicará con Prestashop mediante su Webservice. Al final resultaba ser una buena idea, ya que haciendo una página web ajena a Prestashop, no solo implicaba no tener que usar sus

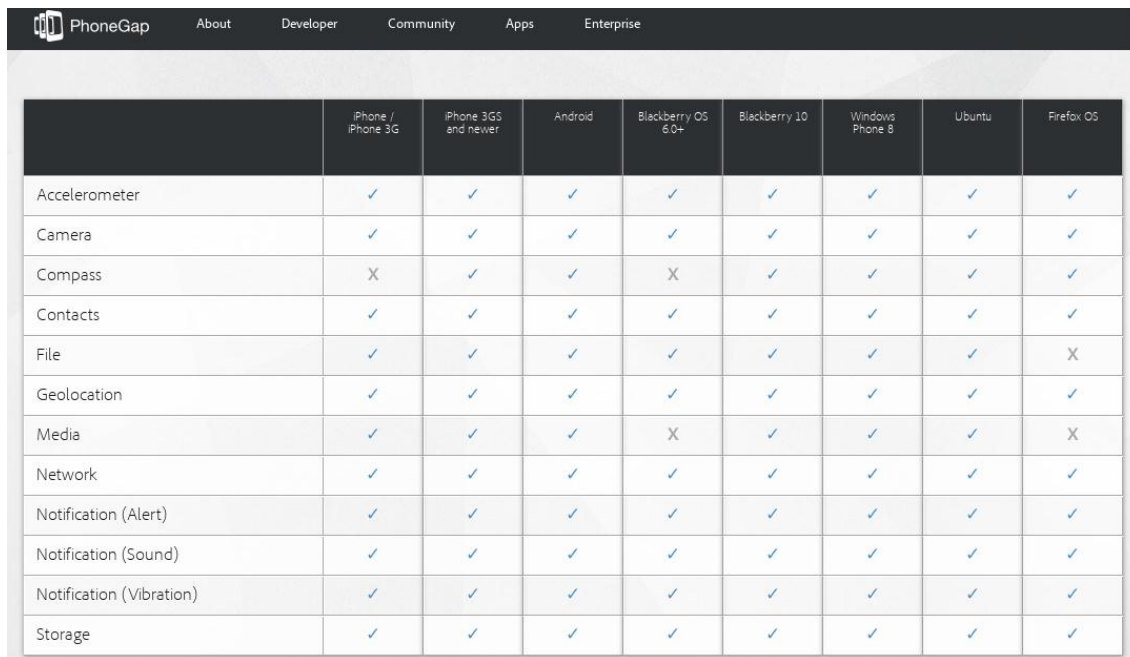
librerías PHP, las cuales, como ya hemos mencionado, tienen una documentación limitada.

También, derivaba en otras posibilidades a contemplar. Por ejemplo, nuestra app no solo se podría comunicar con Prestashop, sino con otros CRM también destinados a la creación de tiendas online. De esta manera, multiplicar los usuarios potenciales era una realidad muy factible.

Asimismo, si nos hubiésemos ceñido a la primera idea, la cual era crear el módulo, tan sólo habríamos usado PHP, HTML y CSS. Por lo que, creando nuestra propia web desde cero no solo íbamos a aprender esos tres lenguajes de programación, sino que también nos veríamos obligados a usar tecnologías que están en boga, tales como javascript y jquery, y lo más importante, aprender a crear páginas web para que se puedan visualizar bien desde cualquier dispositivo.

Objetivos

Los perfiles de programadores web están muy solicitados. De hecho, a la hora de crear aplicaciones para dispositivos móviles, muchas empresas optan por realizarlas en html5 y de ahí exportarlas utilizando algún framework como phonegap. Estas tecnologías se están desarrollando muy rápido, y cada vez se puede acceder a más funcionalidades de los dispositivos móviles, algunas de las cuales, se consideraban en el pasado inviables.



The screenshot shows the PhoneGap website with a navigation bar and a table of device compatibility. The table lists features and their support across different mobile operating systems.

	iPhone / iPhone 3G	iPhone 3GS and newer	Android	Blackberry OS 6.0+	Blackberry 10	Windows Phone 8	Ubuntu	Firefox OS
Accelerometer	✓	✓	✓	✓	✓	✓	✓	✓
Camera	✓	✓	✓	✓	✓	✓	✓	✓
Compass	X	✓	✓	X	✓	✓	✓	✓
Contacts	✓	✓	✓	✓	✓	✓	✓	✓
File	✓	✓	✓	✓	✓	✓	✓	X
Geolocation	✓	✓	✓	✓	✓	✓	✓	✓
Media	✓	✓	✓	X	✓	✓	✓	X
Network	✓	✓	✓	✓	✓	✓	✓	✓
Notification (Alert)	✓	✓	✓	✓	✓	✓	✓	✓
Notification (Sound)	✓	✓	✓	✓	✓	✓	✓	✓
Notification (Vibration)	✓	✓	✓	✓	✓	✓	✓	✓
Storage	✓	✓	✓	✓	✓	✓	✓	✓

Durante los dos años del Ciclo Superior de Desarrollo de Aplicaciones Multiplataforma, nos hemos centrado en el aprendizaje de la programación java. Por lo que, consideramos interesante y de un valor añadido el aprender a realizar aplicaciones web multi-dispositivo.

El objetivo principal que perseguimos al empezar este proyecto era poseer una visión general del proceso de creación de páginas web, así como, de las tecnologías más usadas y las ventajas de unas respecto a otras. También, las ventajas que ofrece el encapsular tu aplicación desde html5 en vez de crearla directamente con tecnologías nativas.

- Investigación sobre las diferentes tecnologías que hay para realizar el back-end de un página web, cuales son las ventajas y desventajas de usar unos frente a otros. Porque es ampliamente usado PHP
- Como maquetar para que la página web pueda verse correctamente en diferentes tamaños de pantallas.
- Las diferentes plataformas de web hosting que hay disponibles y cuáles son los puntos en los que tienes que centrar para escoger uno u otro.

- Que tecnologías existen hay para exportar tu página web a aplicaciones nativas de ANDROID e IOS.
- XAMPP

Entender la importancia de escoger una tecnología back-end apropiada

Una de las decisiones más importantes a la hora de crear un producto, es escoger la pila de tecnologías en la que te vas a basar. El back-end de una aplicación es la capa de software que el usuario final no ve, pero no por ello es menos importante.

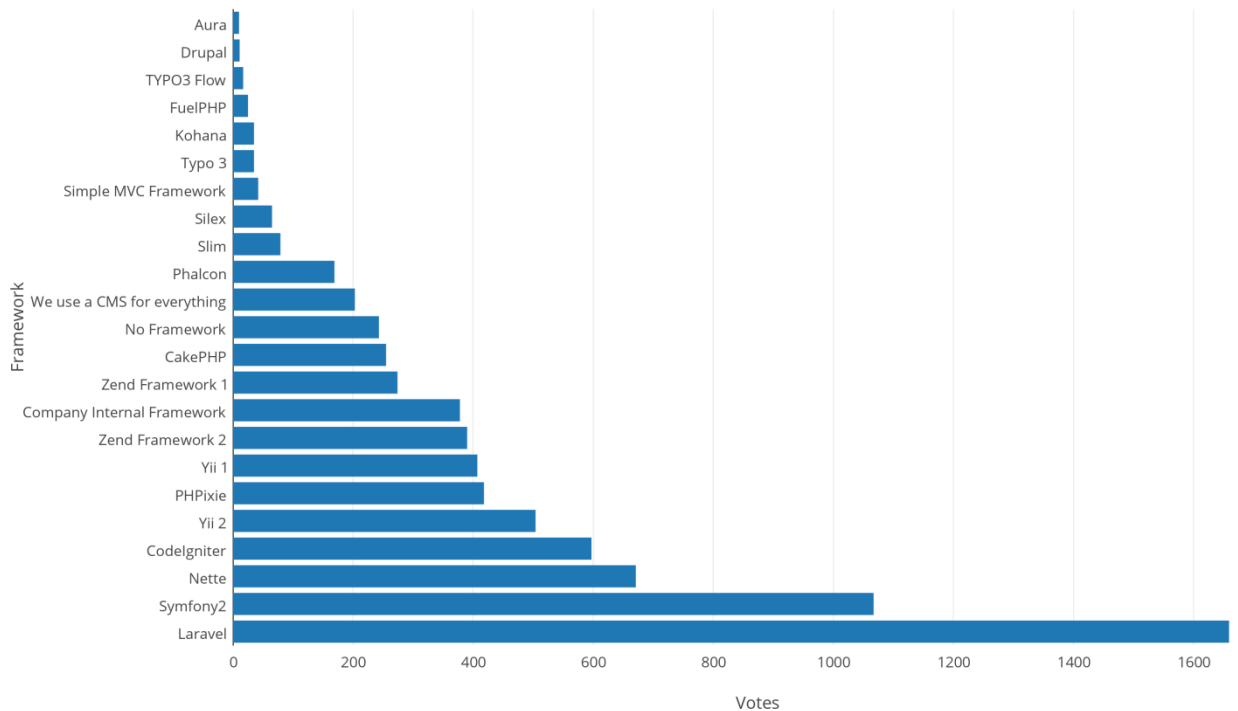
El back-end de la aplicación es la parte del código de la página web que se encarga de la manipulación y flujo de los datos. Generalmente está compuesta de tres elementos: un servidor, una aplicación y una base de datos. Para enlazar estos tres elementos se usan lenguajes especializados que se ejecutan desde el lado del servidor.

A la hora de escoger que tecnología back-end es importante escoger una que te permita escalar y operar a un precio bajo.

Para realizar nuestra aplicación, nos hemos decantado por el lenguaje de script PHP. Actualmente, según W3tech⁶, PHP es usado por el 81.9% de páginas web del planeta.

En el mundo profesional generalmente se usa PHP mediante un framework para facilitar el proceso de creación de software. Por ello decidimos usar uno para experimentar de primera mano la ventajas que ofrece. Finalmente elegimos el framework Codeigniter el cual está basado en MVC. A pesar de que hayan aparecido nuevos frameworks muy potentes como laravel, hemos decidido realizar el proyecto en codeigniter porque es rápido, fácil de instalar y configurar, está bien documentado y nos aporta una buena introducción a la convenciones de MVC en PHP. Además tiene un sistema de ruteo fácil, es fácil de extender y existentes una amplia base de librerías.

PHP Framework Popularity at Work - SitePoint, 2015



En cuanto a nuestra base de datos, nuestra aplicación correrá en una base de datos MySQL. En un principio habíamos decidido utilizar MongoDB, pero existía el problema que casi ninguno de los proveedores que barajábamos para albergar nuestra aplicación, ofrecía MongoDB como una solución por defecto, sino que era de pago. MongoDB nos parecía una buena solución ya que nuestra app necesitaba una base de datos dinámica, en la que no era necesario seguir ningún esquema. Por ejemplo, los productos que los artesanos venden en la tienda, pueden tener una gran variedad de materiales. Y además un mismo material puede provenir de distintos proveedores. Usando MongoDB podríamos haber conseguido una base de datos dinámica y menos rígida más apropiada para la gestión de materiales de artesanos a pequeña escala.

Finalmente nos decantamos por MySQL, ya que esta está disponible en la mayoría de los servidores de hosting, PHP se integra bien con éste, y es rápido. Además es la segunda de bases de datos más usada hasta el

momento, lo que quiere decir que está muy demandada en el mundo profesional.

Como realizar un página web responsive

A la hora de realizar la maquetación web, deberías de ser capaz de diseñar, maquetar, cambiar colores etc. con relativa facilidad. Para ello es importante escoger un framework que te permita crear una aplicación agradable y fácil de usar sin mucha dificultad. Además, nosotros hemos decidido que nuestra aplicación debe ser responsive, es decir, que pueda ser utilizada en diferentes dispositivos sin que la usabilidad se vea perjudicada. En nuestro proyecto es más importante si cabe, ya que está enfocada a artesanos, y estos no siempre tendrán a mano un ordenador para gestionar sus materiales cuando estén fabricando sus productos, pero un móvil si que es posible que lo tenga a mano.

Para la maquetación usaremos el lenguaje de marcas HTML junto con css. De igual modo, que para el back-end, para facilitar el proceso de diseño de esta usaremos el framework Twitter bootstrap.

Alternativas de web hosting, hacer una aplicación SEO friendly.

A la hora de sacar al mercado una página web, sino puedes usar tus propios servidores, seleccionar un servicio hosting que se ajuste a tus expectativas es vital. Con este proyecto queremos investigar cuales son las alternativas existentes en el mercado actual en lo referente a estos servicios, así como aprender a distinguir a servicios de mala calidad de los que ofrecen una buena.

Para que tu aplicación web tenga éxito, también es importante el tener en cuenta como incrementar tu SEO. Nuestro objetivo es también por tanto, el crear una aplicación lo más 'SEO friendly' posible. Para ello, vamos a realizar estudiar cuales son los puntos más importantes que usan los buscadores para

Motivación

Cada vez son más las personas que se animan a crear artesanías y venderlas en internet. Existen varias tiendas online en las que los artesanos pueden

publicar sus productos, el problema de estás es que cobran un porcentaje por cada venta que hagas. Por ello, muchos artesanos optan por crearse sus propias tiendas online para así evitar pérdidas.

Existen muchos CMS especializados en la creación de tiendas online. Shopify, Bigcommerce, 3dcart. Todas estas tiene el inconveniente que son de pago. Entre todas ellas están teniendo mucho éxito Prestashop, la cual es gratuita y Open Source. Es de fácil uso y cada vez es más usada, sobre todo por gente que no tiene conocimientos de programación. Está basado en módulos, los hay gratuitos y otros de pago, por los cuales puedes extender la funcionalidad de tu tienda.

El problema con Prestashop, es que no tiene ningún módulo para ayudar a la creación de productos para los artesanos. No tiene ningún módulo para facilitar el proceso de creación de artesanías. En un principio teníamos pensado crear un módulo, pero decidimos sincronizarlo con Prestashop a través de su api web basada en api rest. Tomamos esta decisión para así también poder sincronizar nuestra aplicación web con otras tiendas online que no sean Prestashop. La idea es que en un futuro craft and budget sea una herramienta de gestión para los artesanos en los que pueden exportar e importar sus productos a diferentes tiendas online. De esta manera tendríamos más usuarios potenciales.

Además de sincronizar nuestra app con tiendas personales creadas por los artesanos, también vemos lógico sincronizarla con portales e-Commerce. Estos portales, a diferencia de las tiendas personales donde solo están disponibles productos propios del artesano, son tiendas virtuales masivas donde artesanos pueden subir sus productos para que gente interesada pueda comprarlos. Los dos portales e-Commerce enfocados para artesanos más importantes en estos momentos y en los que teníamos pensado sincronizarlas son Etsy, Dawanda. Ambas disponen de web api basadas en rest api, iguales que Prestashop.

Antecedentes

Después de investigar diferentes tecnologías disponibles, más tarde expondremos cuales, nos decantamos por las siguientes:

Desarrollo de la práctica

En esta sección vamos a ofrecer información sobre la realización del trabajo en sí. En esta parte se describe lo que se ha hecho, cómo se ha llevado a cabo, por qué se ha hecho así y no de otra manera, qué materiales o herramientas han sido necesario utilizar, qué metodología de trabajo y validación se ha utilizado.

Material

A continuación vamos a detallar cada tecnología que ha estado implicada en la realización de nuestro proyecto.

PHP y MySQL

PHP es el lenguaje de lado servidor más extendido en la web. Nacido en 1994, se trata de un lenguaje de creación relativamente reciente, aunque con la rapidez con la que evoluciona Internet parezca que ha existido toda la vida. Es un lenguaje que ha tenido una gran aceptación en la comunidad de desarrolladores, debido a la potencia y simplicidad que lo caracterizan, así como al soporte generalizado en la mayoría de los servidores de hosting.

PHP nos permite embeber su pequeños fragmentos de código dentro de la página HTML y realizar determinadas acciones de una forma fácil y eficaz, combinando lo que ya sabemos del desarrollo HTML. Es decir, con PHP escribimos scripts dentro del código HTML, con el que se supone que ya estamos familiarizados. Por otra parte, y es aquí donde reside su mayor interés, PHP ofrece un sinfín de funciones para la explotación de bases de datos de una manera llana, sin complicaciones.

PHP es el acrónimo de HyperText Preprocessor. Para crear páginas web dinámicas, es necesario recuperar datos y mostrarlos en la página web. Para almacenar los datos hemos escogido MySql, el cual es un sistema gestor de bases de datos gratuito y popular, que se integra perfectamente con PHP con el fin de crear páginas webs dinámica y funcionales. MySQL una base de datos

RDBMS (Relational database management system) rápida, y fácil de usar que se usa en la mayoría de páginas web.

Las principales razones por las que nos hemos decidido a escoger php y MySQL como back-end de nuestra aplicación son:

- Son gratuitas. La mejor opción de coste-eficiencia.
- Son orientadas a la creación de páginas webs. Ambos fueron creados específicamente para ser usados en la programación de páginas webs dinámicas.
- Son fáciles de usar. Ambos fueron creados para crear webs rápidamente.
- Son rápidas. Fueron diseñados para que fuesen rápidos. Juntos proveen una de las maneras más rápidas de crear páginas webs dinámicas.
- Se comunican bien entre si. PHP tiene características integradas para comunicarse con MySql. No es necesario tener un conocimiento de los detalles técnicos, PHP se encarga de ellos.
- Ambos tienen una amplia comunidad de desarrolladores. Tiene una amplia base de soporte técnico, además, como se suelen usar juntos, comparten la misma comunidad de desarrolladores.
- Al ser ambos de código abierto, son customizables. Permiten a los programadores modificar el software PHP y MySQL para que se ajusten a sus necesidades particulares.

Javascript y JQuery.

JavaScript es un lenguaje de programación interpretado, orientado a objetos, basado en prototipos e imperativo con tipado dinámico, se diseñó en 1995 con un lenguaje muy similar a C.

Javascript es utilizado principalmente en el lado del cliente, aunque tiene otros usos actualmente, javascript se implementa como parte del navegador web y permite mejoras en la interfaz de usuario y páginas web dinámicas.

jQuery es una biblioteca JavaScript, rápida, pequeña y rica en funciones. Actúa como el HTML, recorre y manipula documentos, maneja y captura eventos y

animaciones. Implementa un Ajax mucho más simple con una API fácil de usar que funciona a través de una gran multitud de navegadores.

Gracias a su combinación de versatilidad y extensibilidad, jQuery ha cambiado la forma de escribir Javascript.

Se ha decidido el usar Javascript y JQuery debido por los siguientes motivos:

- Permite crear páginas webs interactivas y dinámicas sin tener que realizar peticiones al servidor constantemente, ya que actúa en el navegador del usuario.
- Tanto javascript y JQuery tienen un gran soporte y una extensa documentación a lo largo de la web

Ajax

AJAX es acrónimo de “Asynchronous JavaScript And XML”, diseñado y creado en 2005. Permite usar una técnica de desarrollo web para crear aplicaciones interactivas o RIA (Rich Internet Applications). Estas aplicaciones son ejecutadas en el cliente, es decir, en el navegador mientras se mantiene la comunicación asíncrona con el servidor en segundo plano. Gracias a esto es posible realizar cambios sobre las páginas sin necesidad de recargarlas, mejorando la usabilidad, velocidad e interactividad.

Ajax es una combinación de cuatro tecnologías ya existentes: HTML y hojas de estilos en cascada (CSS) para el diseño que acompaña a la información. XML, que es el formato usado generalmente para la transferencia de datos solicitados al servidor y el objeto XMLHttpRequest que sirve para intercambiar datos de forma asíncrona con el servidor

Por lo general Ajax se define de la siguiente forma:

“Ajax no es una tecnología en sí mismo. En realidad, se trata de varias tecnologías independientes que se unen de formas nuevas y sorprendentes.”

Framework codeigniter php

Para la parte del servidor, decidimos usar el framework de PHP Codeigniter. Queríamos asegurarnos de que desarrolláramos una página web estructurada, de fácil mantenimiento y legibilidad. Usar un framework te garantiza todo eso.

Codeigniter te permite ahorrar tiempo, ya que este se encarga de la sanitación de los datos, manejos de errores, procesos de logeo, proceso de registro, y manejo de las sesiones. También cuenta con multitud de librerías que te facilitan tareas como enviar e-mails, validación de informes, subida de ficheros a la base de datos. Sin todas estas librerías el desarrollo de la página web habría sido mucho más lento, tedioso e inseguro.

Además al usar un framework, es difícil que acabes con una estructura de directorio poco organizada, ya que la mayoría de estos viene con una estructura predefinida para usar. En concreto, Codeigniter usa el modelo MVC (modelo vista controlador), el cual te permite separar la lógica de las interfaces de usuario de manera que las páginas web contienen pocos scripts. En el caso de Codeigniter, el modelo representa la estructura de datos. Es donde se realizan las conexiones con las bases de datos y donde se encuentran las funciones que permiten insertar, coger, y actualizar los datos. La vista es donde se presenta la información al usuario. Generalmente, en Codeigniter se suelen separar en footer, body y el header para no repetir código. Por último el controlador es un intermediario entre la vista y el modelo. Además es el encargado de procesar las request de http como los get y post de los formularios. También es el encargado de generar las vistas.

Codeigniter ofrece también flexibilidad a la hora de cambiar la plataforma de la base de datos. Tan solo tienes que cambiar algunos archivos de configuración y ya está lista para cambiar la plataforma en la que corre tu aplicación.

Codeigniter ofrece por defecto buenas medidas de seguridad. Por ejemplo, Codeigniter ofrece las siguientes funcionalidades por defecto:

- Cada valor que a un objeto de la base de datos es filtrado contra ataques SQL de inyección.
- Todas las funciones generadoras de HTML, como las de formularios y URL filtran los datos de salida automáticamente.
- Todo dato ingresado por el usuario puede ser filtrado contra XSS.
- Posibilidad de encriptar cookies automáticamente tan solo cambiando opciones de configuración.

Optimización de seo por defecto. Las URLs generados por Codeigniter son limpias y amigables con los sistemas de búsqueda online. Lo consigue porque en vez de usar direccionamiento de URL estándar, por ejemplo:

"http://www.example.com/catalog.asp?itemid=232&template=fresh&crct=ppc&crsource=google&crkw=buy-a-lot"

Codeigniter usa un sistema personalizado en el que es mucho más legible, y es comprensible lo que hace la url, ya que como ya se detallará más adelante, codeigniter usa query string del tipo:

"example.com/news/article/345"

En donde:

- *example.com*: Dominio de la web.
- *news*: controlador que se esta utilizando
- *article*: método dentro del controlador que se esta llamando.
- *345*: parámetro que se le esta pasando al controlador.

Codeigniter permite crear librerías propias, que no son más que la forma que tienen para llamar a las clases guardadas en el directorio "application/libraries", sirven para comunicar el modelo con el controlador y tratar la información entre estas, también separa los recursos locales de los recursos del framework.

Además Codeigniter permite extender las clases nativas para agregar alguna funcionalidad a alguna librería existente o incluso sustituir bibliotecas nativas.

En codeigniter también encontramos modelos, que son clases PHP que están diseñadas para trabajar con la información de la base de datos. En los modelos haríamos todas las peticiones SELECT, UPDATE, DELETE e INSERT a la base de datos.

En codeigniter encontramos archivos de ayuda, llamados helpers, que son simplemente una colección de funciones de una categoría partículas. Se encuentran URLS ayudantes, que ayudan a crear vínculos, ayudantes a la creación de los formularios, ayudantes de formateo de texto, ayudantes para la creación y gestión de las cookies, en general, pequeñas funciones simples de procedimiento, cada función auxiliar realizar una tarea independiente y específica. Para cargarlo automáticamente en todos los controladores hay que configurarlo en el autoload de los archivos de configuración.

Los hooks de Codeigniter ayudan a aprovechar el funcionamiento interno de la estructura sin tener que modificar los archivos principales. Cuando se desea algún tipo de acción o evento en una etapa en particular de la aplicación, es necesario llamar a los hooks, como cargar el sistema de verificación de login con las cookies antes de iniciar el controlador de la página, o cargar un script al final de la página antes o después de cargar los controladores.

Codeigniter permite añadir configuraciones a la mayoría de recursos que ofrece, se puede configurar la base de datos, los doctypes de las cabeceras html, las constantes de php, las rutas de la aplicación, como ruta a la pantalla 404 personalizada o urls reescritas. El autoload permite configurar librerías, urls ,drives y lenguajes que se cargan por defecto en todos los controladores

En el archivo de configuración global se definían los sufijos de la url, la página índice, permitir o no algunas configuraciones, encriptación y prefijo de las cookies de la aplicación y formatos de textos y de clases, como date(Y-m-d H:i:s).

Twitter bootstrap

Bootstrap, es un framework originalmente creado por Twitter, que permite crear interfaces web con CSS y JavaScript, cuya particularidad es la de adaptar la interfaz del sitio web al tamaño del dispositivo en que se visualice. Es decir, el sitio web se adapta automáticamente al tamaño de una PC, una Tablet u otro dispositivo.

En este caso, sin experiencia previa en el desarrollo de interfaces, ni en la programación front-end, la decisión fue usar una plantilla de Twitter bootstrap. De esta manera el enfoque de la materia versaba sobre los conocimientos sí poseídos, la programación back-end, sin enfocarse en el diseño de páginas web responsive. Por otro lado, al tratarse una página que tiene como target la manipulación de inventarios por parte de los usuarios, resultaba imprescindible encontrar una plantilla con un back office potente y agradable. Finalmente, la decisión adquirió un nombre final, "INSPINA admin theme". Tras su descarga desde la página <https://wrapbootstrap.com/theme/inspinia-responsive-admin-theme-WB0R5L90S>, donde hay multitud de plantillas para escoger, se prosiguió el desarrollo. Cabe destacar que esa misma plantilla está en cinco versiones: asp.net, ruby on rails, angular js, meteor y HTML/Javascript. La escogida fue esta última y toda la parte dinámica se dejó en manos del lenguaje PHP.

Api rest

Actualmente, todas las empresas que lanzan al mercado Apis webs para que los desarrolladores puedan usar sus servicios, están basadas en APIrest. La web app de Prestashop es una api REST, y para poder sincronizarnos con ella, hemos tenido que comprender en que se basa este tipo de apis.

REST deriva de "Representational State Transfer", que traducido significa "transferencia de representación de estado. La principal característica de REST es que es un servicio que no tiene estado (es stateless), lo que quiere decir que, entre dos llamadas cualesquiera, el servicio pierde todos sus datos.

Esto es, que no se puede llamar a un servicio REST y pasarle unos datos (p. ej. un usuario y una contraseña) y esperar que “nos recuerde” en la siguiente petición. De ahí el nombre: el estado lo mantiene el cliente y por lo tanto es el cliente quien debe pasar el estado en cada llamada. Si quiero que un servicio REST me recuerde, debo pasarle quien soy en cada llamada. Eso puede ser un usuario y una contraseña, un token o cualquier otro tipo de credenciales, pero debo pasarlas en cada llamada. Y lo mismo aplica para el resto de información.

REST se apoya totalmente en el estándar HTTP, y nos permite crear servicios y aplicaciones que pueden ser usadas por cualquier dispositivo o cliente que entienda HTTP, por lo que es increíblemente más simple y convencional que otras alternativas que se han usado en los últimos diez años como SOAP y XML-RPC.

Por su parte REST es simple. REST no quiere dar soluciones para todo y por lo tanto no pagamos con una demasiada complejidad una potencia que quizá no vamos a necesitar.

REST se definió en el 2000 por Roy Fielding, coautor principal también de la especificación HTTP. Podríamos considerar REST como un framework para construir aplicaciones web respetando HTTP.

Por lo tanto REST es el tipo de arquitectura más natural y estándar para crear APIs para servicios orientados a Internet.

Web api Prestashop

La web api de Prestashop usa la arquitectura REST, de esta manera se asegura que está disponible en la mayoría de plataformas, ya que los protocolos HTTP y archivos XML son entendibles por la mayoría de las plataformas, si no todas. Básicamente, lo que la web api ofrece a los desarrolladores es un sistema CRUD para su base de datos. De esta manera

es posible realizar inserts (Create), selects (Retrieve), update (Update), y deletes (Deletes) en la base de datos de Prestashop.

En esta tabla se muestran las equivalencias entre funciones HTTP y sentencias SQL:

HTTP / REST	CRUD	SQL
POST	Create	INSERT
GET	Retrieve	SELECT
PUT	Update	UPDATE
DELETE	Delete	DELETE

Para usar el servicio webservice que nos ofrece prestashop, se ha creado una api-key y habilitado su uso con los permisos necesarios para el uso de los métodos GET, POST, PUT y DELETE.

Mediante la llamada a la api `www.example.com/api/metodo/($1)` GET podemos recoger la información necesaria mediante XML o json y parsearla a nuestro criterio, se puede ampliar o detallar la información obtenida mediante el uso de las opciones de representación facilitadas por Prestashop.

- Display: mediante el uso de este parámetro podemos mostrar el contenido de un campo, o el de todos.
`&display=[id,Price,name] | &display=full`
- Filter: este recurso permite filtrar los campos para mostrar la información del recurso entre dos valores dados, filtrar mediante uno o varios valores específico o que contengan algún valor proporcionado

`&filter[campo]=[1|5] || &filter[campo]=[1] || &filter[campo]=[1,2] ||`

`&filter[campo]=%[1]%`

- Sort: Con sort se ordenan alfanuméricamente los campos que se necesiten de forma ascendente o descendente.

&sort=[campo1_ASC,campo2_DESC]

- Limit: Con limit se puede limitar el máximo de recursos solicitados y el índice del comienzo de la petición

&limit=10 || &limit=9,5

Con esto podemos realizar peticiones muy precisas a la api de Prestashop

?display=[full]&filter[price]=[20]&sort=[price _ASC]&limit=5.

Para realizar un POST o un PUT, hay que enviar un xml con el tipo de método y contenido en el header.

URL: <http://www.site.com/api/products/1>.

Método: "PUT/POST"

Tipo de contenido:"text/xml"

En el servicio web, hay un método para recuperar un XML vacío. Se puede acceder a él formateando una url de la siguiente manera.

Es posible sustituir el valor del parámetro del schema " blank" con "synopsis " para obtener más información sobre los campos de los recursos.

En cambio para un delete simplemente es necesario introducir el parámetro del id del recurso dado

URL: [http:// www.site.com /api/products/1](http://www.site.com/api/products/1)

Método = "DELETE"

Tipo de contenido: "application/x-www-form-urlencoded"

Planificación

En este apartado se va a especificar la forma en que se organizó el equipo a la hora de llevar a cabo el desarrollo de la aplicación. Se pasará a detallar la metodología por la cual se organizó la gestión del proyecto, así como el software que se usó para ello.

Metodología de gestión usada

La metodología que ha sido usada para la realización del proyecto ha sido una metodología basada en scrum. En esta metodología, hay tres diferentes roles en el equipo: el equipo de desarrollo, el 'scrum master' o el gerente de proyecto y por último el 'product owner'.

El product owner representa la voz del cliente. Se asegura de que el equipo Scrum trabaje de forma adecuada desde la perspectiva del negocio. El Product Owner escribe historias de usuario, las prioriza, y las coloca en el Product Backlog. El Scrum Master su trabajo primario es eliminar los obstáculos que impiden que el equipo alcance el objetivo del sprint. El Scrum Master no es el líder del equipo (porque ellos se auto-organizan), sino que actúa como una protección entre el equipo y cualquier influencia que le distraiga. El Scrum Master se asegura de que el proceso Scrum se utiliza como es debido. El Scrum Master es el que hace que las reglas se cumplan. Por último, el equipo tiene la responsabilidad de entregar el producto. Un pequeño equipo de 3 a 9 personas con las habilidades transversales necesarias para realizar el trabajo (análisis, diseño, desarrollo, pruebas, documentación, etc).

Debido a que el proyecto ha sido realizado por un equipo de dos personas, los tres roles han sido compartidos por los dos integrantes del grupo, pero, los tres roles han sido usados.

Al principio de la práctica, realizamos una especificación de requisitos de nuestro software, en el cual se detallaban todas las funcionalidades deseables del software. Este documento es el product back-log de la aplicación, el cuál se trata de un documento de alto nivel en el que se detallan el conjunto de tareas, los requerimientos y las funcionalidades requeridas por el proyecto. En la

metodología scrum, este documento solo puede ser manipulado por el product owner, en nuestro caso, lo hemos realizado los dos integrantes del grupo, y además, como las funcionalidades del proyecto cambiaban, ha estado siendo modificado durante toda la realización del proyecto. El product backlog que usamos se encuentra en el Apéndice A: Product Backlog

Una vez definido el back-log del proyecto, empezamos a definir los sprints. Normalmente, cada sprint comienza con una reunión de planificación de sprint en donde el product owner y el equipo se ponen de acuerdo en que historias del backlog van a ser movidas al sprint back-log. Es responsabilidad del product Owner determinar que trabajo se va a llevar a cabo en cada semana, mientras que el equipo permanece autónomo en lo referente a la toma de decisiones de cómo realizar estos trabajos. Una vez determinado que trabajo el equipo realizará durante el sprint, el product owner no puede añadir o quitar ninguna tarea. En nuestro caso, los dos éramos el product owner y el team, por lo que tomamos la decisión, aceptando el consejo del José Luis Navarro, el CEO de la empresa en la cual estábamos realizando las prácticas; que para tener una visión general del desarrollo del desarrollo producto, realizar una planificación semanal empezando por el final. Es decir, empezar desde la semana en hay que presentar la documentación, y desde ahí ir hacia atrás. Dicho documento está en el Apéndice B: Sprint Semanales. De esta manera, a diferencia con una metodología scrum pura, no nos reuníamos al comienzo de cada sprint (semana) para discutir cuáles eran las historias que debíamos realizar en ese sprint, sino que ya estaban predefinidas desde el comienzo las funcionalidades del sistema que iban a estar listar al final de cada semana. Para lo que si se ejecutó una reunión semanal al comienzo del sprint, era para especificar detalladamente cada una de las tareas que había que realizar a lo largo de la semana y quien las iba a llevar a cabo.

Software de gestión usado

Para la realización de dichas tareas, se utilizó un software de gestión de proyectos específicamente diseñado para metodologías scrum. En concreto se

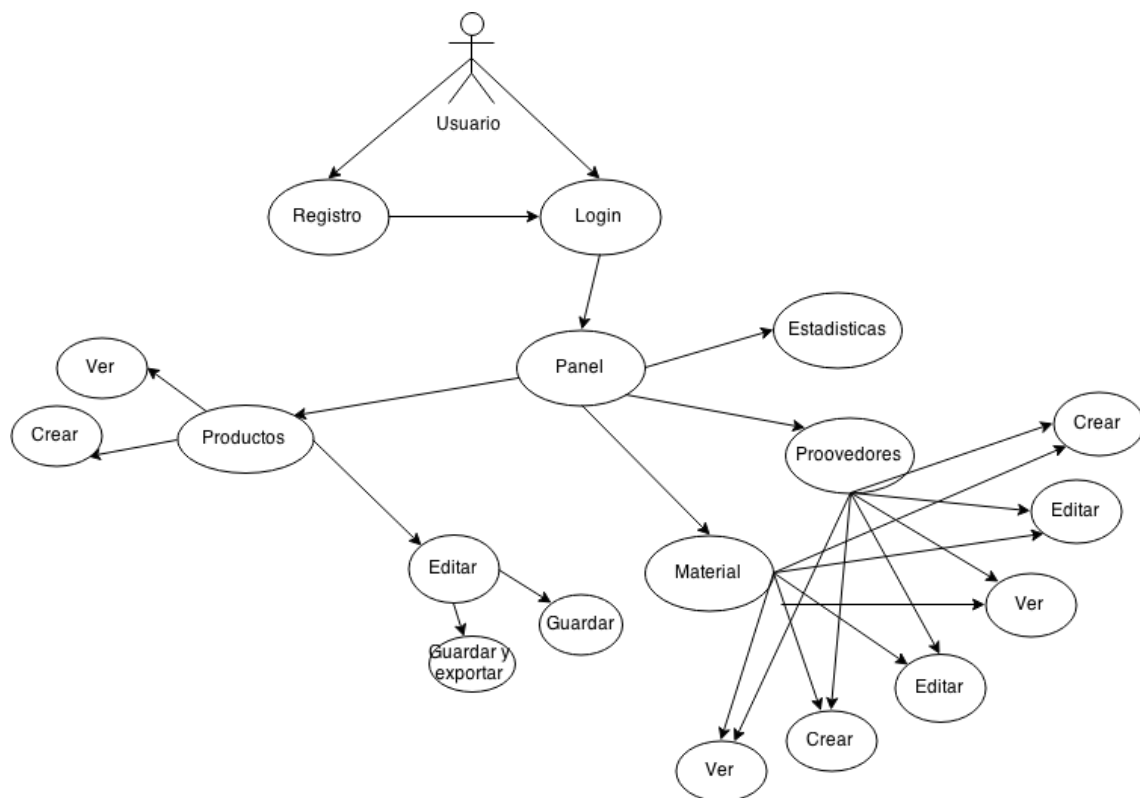
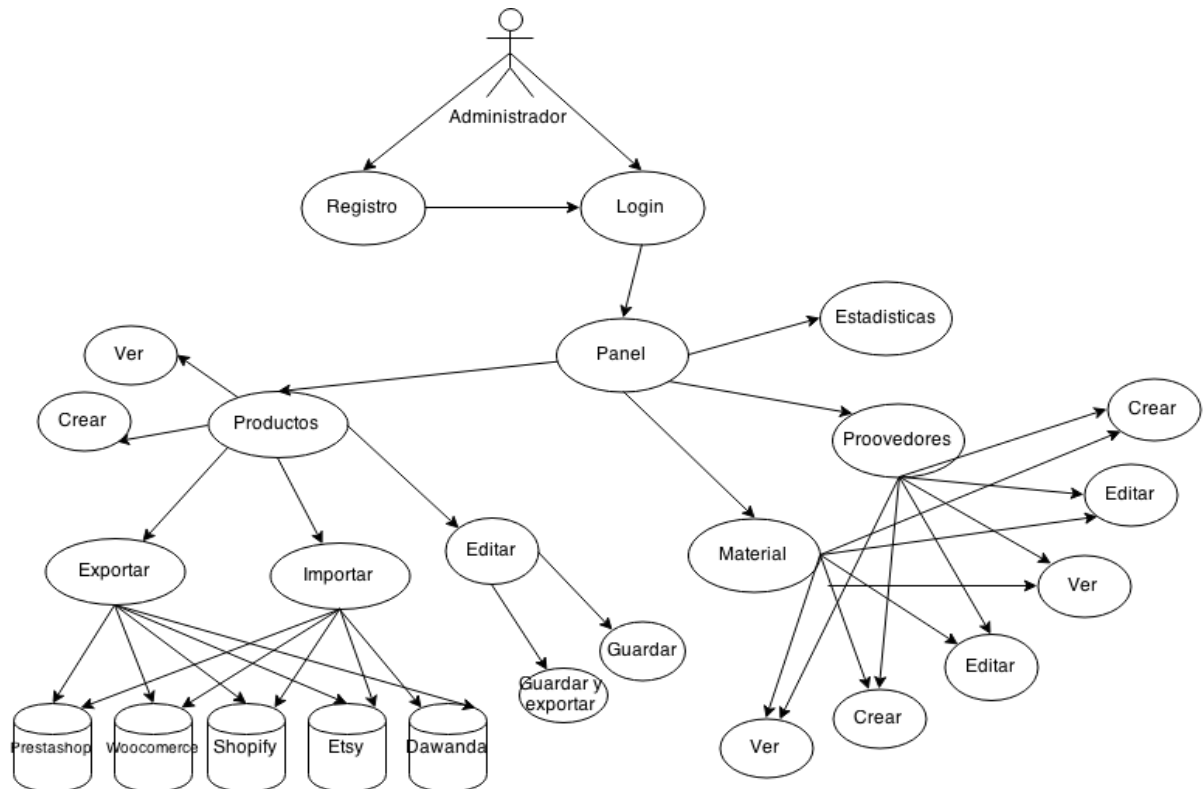
utilizó Trello, en el cual hay cuatro columnas. En la primera se encuentra el product back-log, en esta columna es donde se encuentran todas las historias que hay que realizar del producto. En la segunda columna se encuentra la columna de tareas por hacer, en esta columna se ponen las tareas específicas necesarias para la correcta realización de cada historia. En la tercera columna es donde se encuentran las tareas que se están realizando en el momento. En esta columna además, el Trello permite añadir que persona está realizando dicha tarea. Por último está la columna de tareas finalizadas.

Por último, para la correcta gestión del código que se estaba desarrollando ha sido usado el sistema de control de versiones distribuidas git. Y para hospedar ese repositorio en un servicio web de manera de que fuese accesible por ambos integrantes del grupo, ha sido usado github.

Descripción del trabajo realizado

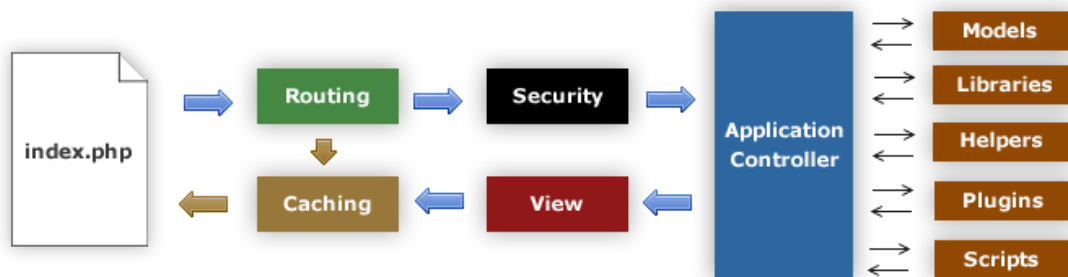
Casos de Uso

Como se puede observar en los diferentes casos de uso de la aplicación, se dispone de dos tipos de roles de usuario, administrador y usuario, cuya diferencia es que un usuario no puede importar y exportar productos a tiendas externas.



Codeigniter

Para la realización del back-end se usa el framework de PHP Codeigniter. Es un framework basado en el modelo vista controlador o MVC.



Como se puede apreciar en el diagrama, el archivo index.php actúa como el controlador frontal, encargándose de inicializar todos los recursos fundamentales para que funcione el framework. Esto conlleva el problema de que todas las URLs de la web app tienen que tener el “index.php” después del nombre del dominio, lo que provoca que las URLs no sean amigables y disminuya el factor SEO. Sin embargo esto puede arreglarse fácilmente añadiendo las siguientes líneas de código en el archivo de configuración de apache “.htaccess”:

```
RewriteEngine On
RewriteCond %{REQUEST_FILENAME} !-f

RewriteCond %{REQUEST_FILENAME} !-d

RewriteRule ^(.*)$ index.php?/$1 [L]
```

Lo que se consigue con estas líneas es redirigir todo el tráfico que reciba nuestro server que no contenga un “index.php”, a otra dirección añadiendo el “index.php”. De esta manera, a todas las URLs que reciba el server se le añadirá un “index.php” después del dominio.

Después de que el index.php haya sido llamado, el router examina la request del HTTP para determinar qué hacer con la llamada. Si existe algún archivo caché, Codeigniter envía este directamente al navegador web, evitando así el

sistema normal de ejecución. También se puede observar en el diagrama, que antes de que el controlador de la aplicación se cargue, la request de HTTP y cualquier dato enviado por el usuario también es filtrado por el sistema de seguridad de Codeigniter. Después, el controlador carga los modelos necesarios, las librerías, los helpers y cualquier recurso que sea necesario para procesar la petición del cliente. Finalmente, la vista es renderizada y enviada al navegador web para ser visualizada. Si el sistema de cacheo esta activado, la vista es cacheada de tal manera que pueda ser recuperada por siguientes peticiones.

Para este proyecto, se ha creado un controlador por cada recurso, cada controlador determina el recurso usado, por lo que existen los controladores “Usuarios” para gestionar todo lo relacionado a los usuarios, tal como el login, registro y recuperar contraseña, también existen los controladores “Materiales”, “Productos”, “Dashboard”, “Stock” y “Welcome” para la pantalla de la landing page.

Para una mayor usabilidad y limpieza, se ha creado una librería y un modelo de la base de datos por cada controlador.

Para la realización del login se ha recurrido a una librería libre de codeigniter para php hecha por Stéphane Bourzeix. Se ha conseguido a través de esta URL:

"<http://github.com/DaBourz/SimpleLoginSecure>".

Según palabras de los propio autores, "esta librería está diseñada para ofrecer de una manera rápida y simple, una librería de login que te permita poner en marcha un sistema no intrusivo de autorizaciones. No trata de adivinar como quieres estructura tu app, sino que simplemente intenta ofrecer ayuda".

Es una librería muy sencilla, la cual es muy fácil ponerla en marcha. Tan solo tienes que enlazarla con tu modelo para realizar

una consulta a tu base de datos donde guardes la información de tus usuarios. Una vez se le llame a la función login con parámetros "user" y "password", la librería ejecutará un select en la tabla antes seleccionada buscando coincidencias con "user". Si la select devuelve más de una fila, el password que recibió como parámetro la función login se encripta con el algoritmo MD5. Más tarde se verifica con la columna password en la base de datos (esta se encuentra ya encriptada en MD5). Si existe coincidencia, se pasa a almacenar en caché los datos de la sesión que el programador considere oportuno y se devuelve el valor true.

Cabe destacar que el algoritmo MD5 no es seguro para almacenar información confidencial, ya que es posible descifrarla muy fácilmente. Es más recomendable usar algoritmos de cifrados más robustos como bcrypt, scrypt o sha-512.

Web Api Prestashop.

Para realizar este punto, se ha incluido en el proyecto la librería Prestashopwebservice para PHP creada por Prestashop para comunicar más fácilmente la aplicación con una tienda Prestashop. Esta librería permite llamar al xml del recurso solicitado con un código más simple.

"<https://github.com/PrestaShop/PrestaShop-webservice-lib>"

Para llamar al xml de las combinaciones de los productos de prestashop y mostrar todos los campos, solo serían necesarias cuatro líneas.

```
$webService = new PrestaShopWebservice($this->web, $this->ps_api_key,  
$this->debug);
```

```
$opt['resource'] = 'products';  
$opt['display'] = 'full';  
$xml = $webService->get($opt);
```

Una vez realizada la llamada del xml es necesario crear una librería que traduzca el xml a una lista con la estructura que se le ha dado al sistema.

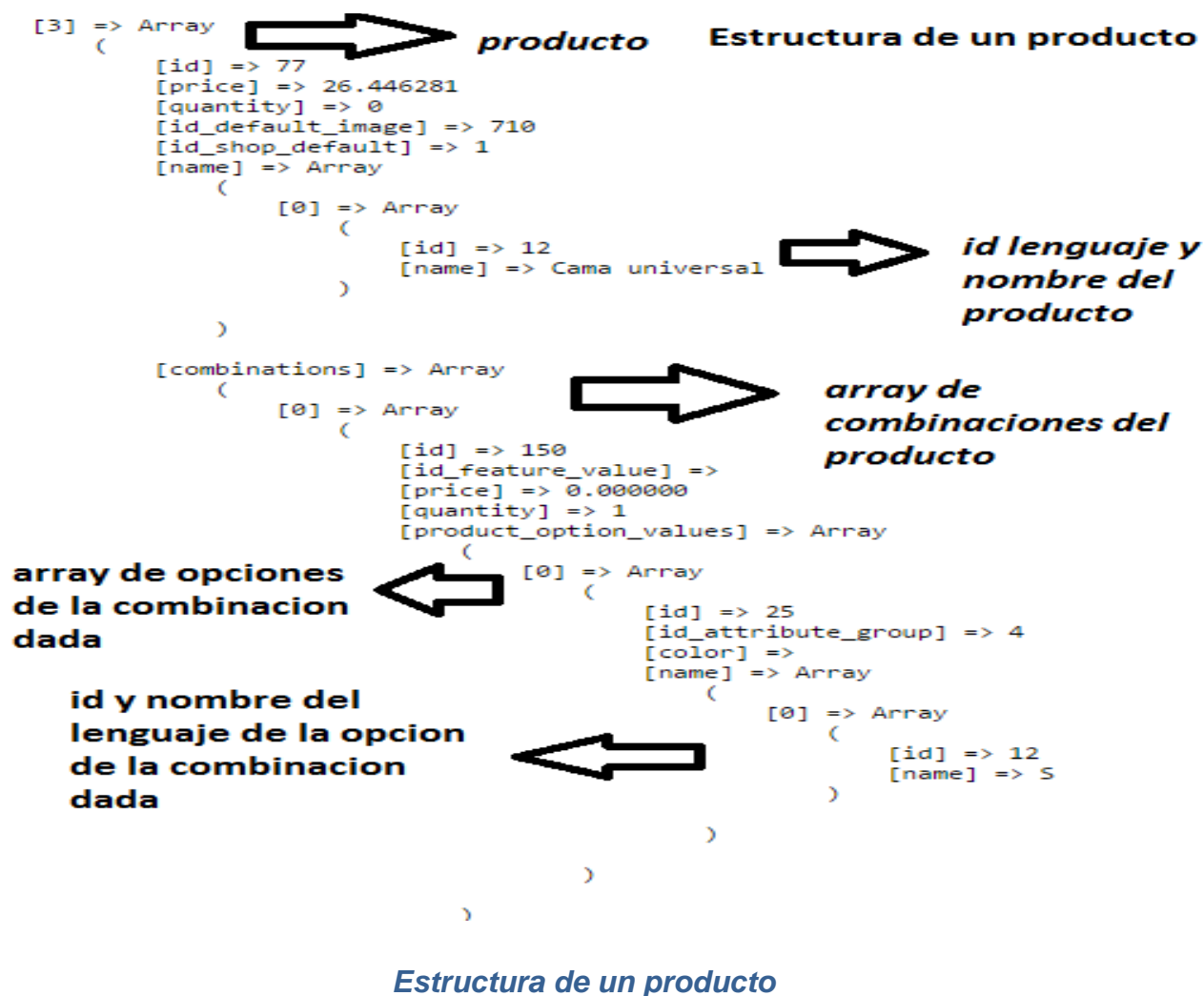
De tal modo que es necesario llamar y guardar mediante la api, los recursos de productos, imágenes, combinaciones, atributos, grupos de atributos y stock, con sus respectivos ids, lenguajes y uniones entre ellas.

Para llamar a todos estos recursos y juntar todas sus referencias usando el mayor rendimiento posible sin realizar demasiadas peticiones al servidor, esto se consigue mediante 4 peticiones mostrando “full” sus campos, parseando y traduciendo la información de estos por separado.

Una vez llamados y guardado en memoria los recursos es necesario unir los atributos en las combinaciones donde coincidan los ids, y las combinaciones en los productos coincidentes. Para esta tarea unimos las peticiones separadas mediante un `array_merge`.

```
foreach ($myproducts as &$miproducto) {  
    if (sizeof($miproducto['combinations'] > 0)) {  
        foreach ($miproducto['combinations'] as &$product_combination) {  
            $product_combination = array_merge($product_combination,  
$mycombinations[$product_combination['id']]);  
            foreach ($product_combination['product_option_values'] as  
&$productoptionvalues) {  
                $productoptionvalues = array_merge($productoptionvalues,  
$product_op_value[$productoptionvalues['id']]);  
            }  
        }  
    }  
}
```

Gracias a esto, se consigue unir los productos, combinaciones, atributos, lenguajes, imágenes con solo cuatro peticiones al servidor, se cambiaría la estructura para adaptarse a las necesidades existentes, y como resultado final se obtiene una lista con una estructura perfectamente formada.



Base de datos MySQL

Se ha tratado de implementar una base de datos lo más completa y escalable posible.

Tiendas:

La aplicación va a ser multitienda. Un artesano puede usar a la misma vez y de forma sincronizada diferentes tiendas online en craft&budget. Para ello, se ha creado la tabla "craftshop" en la que se detallan los datos genéricos que el usuario quiere para todas sus tiendas online. Cada tienda virtual se importará en la tabla "shop". En ella se detalla información de esa tienda en concreto, como el nombre de ésta, o el logo de la tienda. Para conseguir que en una misma aplicación craft&budget haya varias tiendas se crea la tabla de relación n-n "shop_craftshop". En cuanto a la tienda a la que se conecta, como se puede apreciar en la tabla "shop" se necesita almacenar el api key de la web a la que pertenece. Esta api key, permite privilegios CRUD en la base de datos original usando la web api de prestashop o de la tienda en cuestión.

Se consigue así que los artesanos puedan utilizar craft&shop para sincronizar todas sus tiendas online, y saber cuántos y qué productos tienen online en cada tienda. Para esta situación cuando los usuarios sepan rápidamente de qué tienda es cada producto, se ha decidido no solo especificar una columna para el nombre de la tienda virtual, sino que también se ha creado una columna para almacenar el logo de la tienda. Así, cuando el usuario esté visualizando todos los productos que tiene a la venta, podrá saber a qué tienda pertenece cada uno solo con ver la imagen.

Para relacionar cada producto del artesano con la tienda a la que

pertenece, se usan la tabla "types". Evitando así la duplicación de datos del nombre de la tienda y del logo de esta. Como se puede ver en la diagrama de la base de datos, la tabla "shop" hace referencia al id de la tabla "types".

Manejo de usuarios:

Como se puede apreciar en el modelo, se ha dado la funcionalidad a la app de que se puedan dar de alta varios usuarios a la vez en una misma aplicación craft&budget. La información básica de los usuarios se encuentra en la tabla "user". Así mismo, los usuarios pueden tener o un rol de administrador, o un rol de usuario normal. Los usuarios con rol de administrador son los únicos usuarios que pueden importar e exportar datos hacia las tiendas online. Los tipos de roles están especificados en la tabla "user_rol". Finalmente, se ha creado una tabla "craftshop_users" en la que se relacionan un usuario de la tabla "user", con un rol de usuario de la tabla "user_rol" y con una tienda de la tabla "shop".

Multilenguaje

Se ha decidido implementar una página multilenguaje para así tener más usuarios potenciales. Para ello, se ha creado una tabla "lang" en donde se especifica cada lenguaje que aceptará craft&shop. En esta tabla se insertarán todos los valores estándar iso 639-2 de todos los lenguajes disponibles. Estos lenguajes están escritos en tres columnas, una para el nombre del lenguaje, otra para código numérico del lenguaje, y la última para el código alfabético. Se relacionará cada usuario con un o varios lenguajes. Para saber que lenguajes tiene disponible un usuario, se usará la web api de prestashop, ya que en su base de datos se almacena los lenguajes que tiene el usuario. La idea de implantar esta funcionalidad es que muchos artesanos traducen sus productos al inglés o a otro idioma para conseguir más clientes. Desde prestashop

no es posible averiguar cuál es su lenguaje principal, así que se optó por descargar todos sus productos en todos los lenguajes que tenga especificado, y que sea el propio usuario quien elija su lenguaje a través de una pantalla de configuración. Se ha considerado que las mejores partes en las incorporar multilenguaje de nuestra aplicación sea los productos, y los datos estáticos de la página, es decir, los menús y mensajes de información. Los materiales, y proveedores solo se almacenarán en la base de datos en un solo lenguaje, debido a que nuestra app será un back office para los artesanos, no un producto final para la vista de los clientes. Tiene más sentido que la información insertada por el usuario se almacene tan solo en un lenguaje, el lenguaje principal del usuario. La tabla que se encargará de almacenar los diferentes lenguajes de cada producto será la tabla "product_language". Esta tabla es la relación n-n entre la tabla "product" y la tabla "lang". Un producto puede estar traducido a muchos lenguajes, y un lenguaje puede estar siendo usado en muchos productos.

Materiales:

Los materiales se almacenarán en una tabla "materiales". Como se ha dicho anteriormente, los materiales son los materias necesarias para poder fabricar los productos de los artesanos. Estos materiales tendrá que ser introducidos por los artesanos, ya que no existen en las tiendas virtuales. Como nuestro sistema va a ser multiusuario para una misma tienda craft&budget, se ha optado por relacionar cada material no con un usuario, pero con la tienda craft&budget que comparten todos los usuarios de una misma tienda. Para que los artesanos puedan tener más organizado el stock de materiales, se ha optado por relacionar a cada material con una categoría. La columna "id_category" de la tabla "material" hace referencia a la tabla donde se guardan todas las categorías de la base de datos, en la tabla "category". De esta manera, los usuarios podrán saber todos los

materiales disponibles de una categoría dada, y podrán buscar de manera eficiente el stock de materiales. Esta última información se encuentra en la tabla "stock_material".

Los materiales no solo tiene la funcionalidad de especificar cuantos materiales tiene un usuario, sino que también sirven para saber cuantos materiales son necesarios para fabricar un producto. La tabla materiales representa los datos de los materiales, no el stock que hay de dicho materia. Para hacer ello se ha decidido crear en la tabla "materiales" un dato "id_measurement", el cual hace referencia a la tabla "measurement". En esta tabla lo que se almacena es el tipo de medidas de los materiales, es decir si están especificados en kg, g, l, metros cuadrados, etc... Para tener costancia de la cantidad de cada material, se ha creado la tabla "stock_material".

Proveedores

En cuanto a los proveedores, estos es almacenan en la tabla "supplier". De estos se almacenará la dirección, y el nombre. Además para saber que materiales tiene que proveedor, se ha creado la tabla "material_supplier". Es una relación n-n, en donde un material puede ser comprado por varios proveedores. Lo hemos creído conveniente hacerlo así, para dar la oportunidad a los artesanos a poder tener en su base de datos varios proveedores para un mismo material, por si en algún futuro alguno de sus proveedores falla.

Productos

En cuanto a los productos, estos se cogen directamente de la tienda online del artesano. Se almacenarán en la tabla "product". Se ha decidido que cada producto solo puede ser de una tienda online. A pesar de que en la realidad se pueda tener un mismo producto publicitado en varias tienda, hemos considerado más adecuado que si son de tiendas separadas, el producto en nuestra bases de datos es diferente. Ya que podría darse el case de que el artesano quisiera

diferentes precios dependiendo de la tienda.

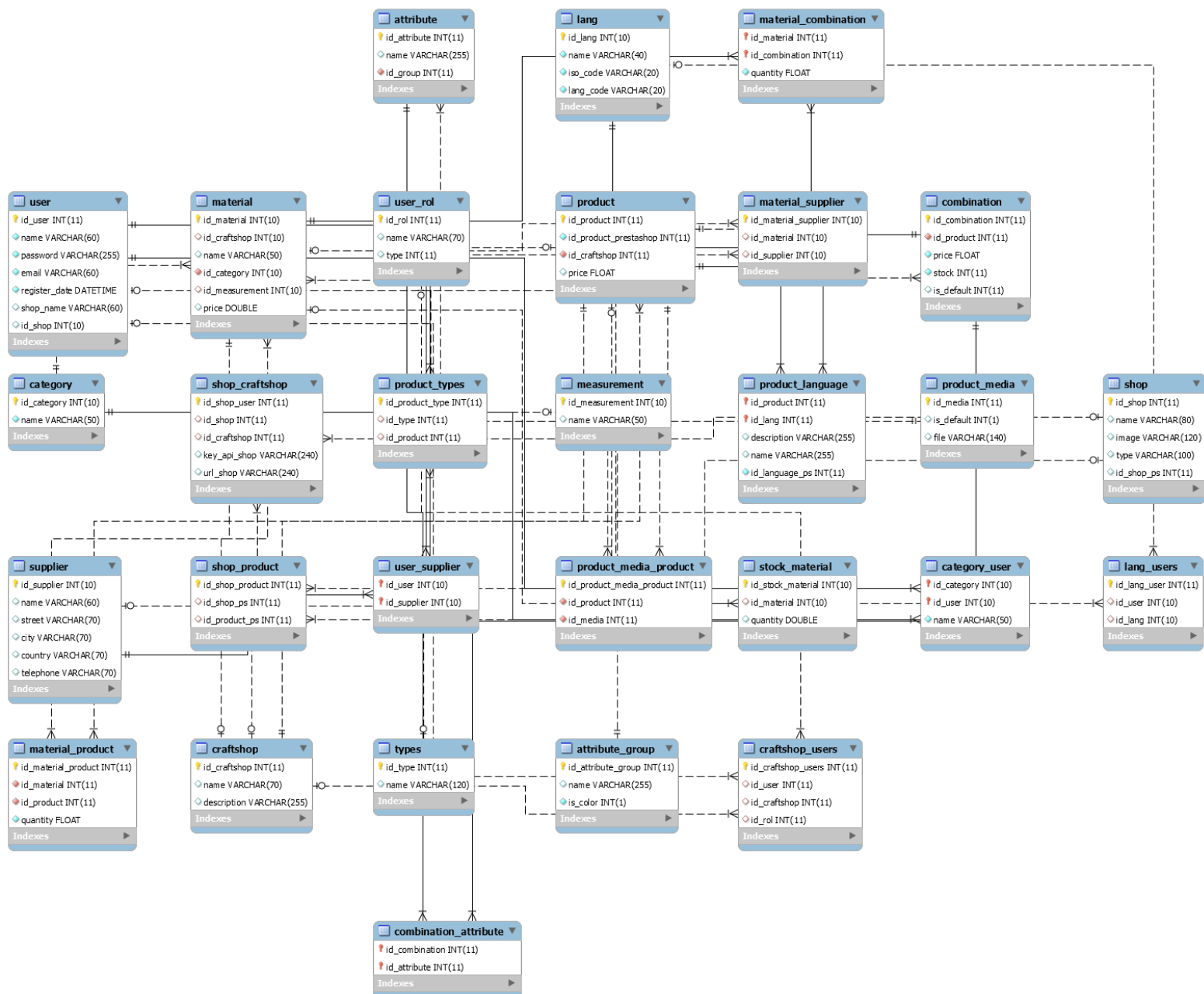
Además también se almacena una imagen de cada material. En realidad, según nuestra base de datos se podría almacenar más de una imagen, pero ya que la imagen solo tiene sirve para facilitar la lectura de productos desde las tablas, no para mostrar la mercancía a clientes, se ha decidido que lo más conveniente es tan solo cargar la imagen principal del producto desde prestashop. Se ha optado por no guardar la imagen en formato binario en la base de datos, sino guardar el nombre de la imagen desde la ruta relativa en nuestro sistema de ficheros. Estas rutas se almacenarán en la tabla "product_media" y todas las imágenes se guardarán en la carpeta assets/images. De esta manera, es mucho más fácil guardar imágenes, pues tan solo hay que llamar a la librería FileUpload de codeigniter, y este se encarga de guardarlos en el sistema de ficheros, tan solo habría que almacenar el string del nombre de la imagen.

Como es una aplicación destinada vendedores de artesanía, y los productos que venden suelen estar hechos a medida, o al menos están en varios colores, tamaños, materiales, etc... Se ha implementado la base de datos de manera que se pueda permitir a los artesanos poder almacenar sus materiales y productos de forma dinámica. Para ellos, se ha creado la tabla "combination", la que sirve como referencia para determinar una variedad de un determinado producto. Por ejemplo, si una mesa es XXL, si está pintada de color azul, etc.. Esta combinación tendrá un stock y un precio. Para almacenar en la base de datos las características de dicha combinación, hemos creado otra tabla, la tabla "attributes". Finalmente, para relacionar una combinación de un producto con un atributo determinado, se ha creado la tabla "combination_attribute".

Cabe destacar, que los materiales se relacionan con los productos,

Proyecto CraftandBudget

no a través de la tablas "productos" y "materiales", sino a través de la tabla "combinación", "materiales". Esto es así, porque dependiendo de los atributos de un producto, este necesitará un tipo de material u otro. Si tienes una mesa, y esta tiene el atributo rojo; interesa relacionar el atributo "pintura roja para mesas" a la combinación, y no al producto mesa.



Resultados y validación (W3Techs - World Wide Web Technology Surveys)

Conclusiones

En la realización de esta práctica nos hemos dado cuenta de la importancia que tienen las apis a la hora de crear tus propias aplicaciones. No solo de las api de tiendas online, sino también de apis que te ayudan a que tus servicios web sean más completos, por ejemplo, que se cargue por defecto una imagen cuando te das de alta con un correo electrónico mediante gravatar. A través de estas apis, los programadores tienen la posibilidad de crear nuevos servicios web juntando partes de apis.

También resulta interesante conocer las diferentes metodologías a la hora de abordar la creación aplicaciones para dispositivos móviles mediante tecnologías de programación web. Cada vez está siendo más usado tecnologías sustitutivas de las nativas para crear aplicaciones móviles como phonegap/cordova.

Uno de los apartados en los que más hemos encontrado problemas ha sido a la hora de la planificación y de la gestión del tiempo. A menudo, tareas que en principio parecían sencillas de implementar se convertían en tareas difíciles de llevar a cabo que se demoraban en el tiempo. También, en la propia viabilidad del proyecto. Han habido funcionalidades que nos parecieron implementables pero que después de pensarlas mejor nos dábamos cuenta de que eran tremendamente difíciles de llevarlas a cabo.

Aportaciones

Una de las mayores dificultades encontradas a la hora de desarrollar el proyecto, era la carencia de conocimientos en el desarrollo web con php, javascript, jquery y el uso de frameworks php, dado que en el ciclo solo vimos

desarrollo de aplicaciones java y android, también en un primer momento se decidió realizar una aplicación android nativa que conectase a la web mediante una api, pero a lo largo del proyecto se vió que el desarrollo de aplicaciones nativas está mermando, al contrario que las web apps desarrolladas en html, css3,js.

Así mismo con este proyecto se ha aprendido una introducción muy amplia sobre el desarrollo de aplicaciones web responsive, adaptadas a cualquier tipo de dispositivo.

Trabajo futuro

Para un futuro, pensamos que craft and budget debería de tomar un camino en el cual ésta se integrase con la mayoría de las tienda online del mercado. No solo tiendas online personales creadas a partir de un CRM como, lo que hemos hecho con prestashop o otras como Magento. Sino también integrarlas con tiendas online de terceros como Ebay, u otras destinadas a artesanos en concreto como dawanda.

También consideramos, que aparte de que nuestra página web sea multidispositivo mediante el framework twitter bootstrap, sería un plus el conseguir convertirla en una web app. De esta manera podríamos ponerla a la venta en las tiendas de aplicaciones play store o apple store. Además, de esta manera podríamos dar una mejor experiencia de usuario a los clientes, ya que estos no tendría que abrir el navegador web para usar craft and budget, sino que podrían abrirlo como una aplicación más de su dispositivo móvil.

Apéndices

Apéndice A: Product Backlog

Pantalla Productos

Para que el artesano pueda gestionar los materiales que necesite para la creación de sus productos:

- En el menú vertical dentro de la edición del producto añadir una pestaña para añadir materiales (Materiales del producto):
 1. Crear la interfaz con la tabla de Materiales y tabla de materiales_productos_X así añadir materiales a materiales_producto fácilmente.
 2. Desde esta pantalla hay botones “crear”, “modificar” y “quitar” con sus funcionalidades.

Para que el artesano sepa qué productos puede crear dependiendo de los materiales que tenga:

- Crear una columna extra en la pantalla de productos que muestre si se puede crear el producto con los materiales que ya se tienen, y cuantos podría crear en el caso de que haya (Verde, puedes hacer el producto original, naranja hay alguno que no se puede hacer, y rojo no se puede hacer ninguno).

Para que el artesano pueda gestionar las variaciones de su producto (por ejemplo: una mesa “x” con cajones o sin cajones)

- Añadir a las variaciones del producto los materiales del que esta formado este producto, a partir de un modelo del producto original (u otro), así cambiaría el precio del producto dependiendo del precio de estos materiales.
 1. Crear interfaz para añadir materiales a la variación del producto para calcular su valor.

Pantalla Materiales

- Crear tabla materiales para la gestión de los mismos, Desde la pestaña Artesanía->Materiales crearíamos y gestionaríamos los Materiales (creación, modificación y borrado)
 1. 123456 Creación: crear el material con : Nombre, descripción, foto detalle, tamaño, unidad, precio/unidad etc.
 2. Modificación: modificar el material (salvo el id)
 3. Borrar: borrar el material y sus referencias en MATERIALES_PRODUCTOS.

Pantalla Proveedores de materiales

Crear tabla e interfaz gráfica de los proveedores de materiales, partiendo de la ya existente en Prestashop pero modificada y adaptada a nuestras necesidades.

Pantalla Estadísticas

Mostrar en una página del back-office las estadísticas por producto:

- Coste de fabricación
- Coste de la mano de obra
- Coste del transporte de ese producto
- Precio de venta al por mayor (- Beneficio de venta al por mayor)
- Precio de venta al por menor (- Beneficio del por menor)
- Impuestos
- Precio de venta final al comprador:

Extras:

Creación de la app android para comunicar la parte de artesanía con el móvil.

Apéndice B: Sprint Semanales

Planificación semana 1

Fecha:

13 – 17 de abril de 2015

Tareas:

Investigar diseño web responsive y MVC aplicado en Prestashop.

Investigar y estudiar sobre la creación de módulos Prestashop.

Investigar la arquitectura de la base de datos de Prestashop.

Planificación semana 2

Fecha:

20 – 24 de abril de 2015

Tareas:

la estructura de archivos del módulo

Diseñar el logotipo del módulo

Diseñar la base de datos (tabla de nuestro módulo)

Planificación semana 3

Fecha:

27 de abril – 1 de mayo de 2015

Tareas:

Crear la interfaz para la gestión de materiales:

Implementar las funcionalidades "Crear, modificar y borrar materiales"

Planificación semana 4

Fecha:

4 – 8 de mayo de 2015

Tareas:

Implementar el servicio de stock de materiales para que el artesano sea capaz de gestionar sus existencias.

Elaborar la interfaz y funcionalidad de proveedores de materiales.

Planificación semana 5

Fecha

11 – 15 de mayo de 2015

Tareas:

Modificar el módulo productos de Prestashop para añadir Materiales a los productos del artesano y la disponibilidad para crear ese producto dependiendo del inventario de materiales.

Añadir la funcionalidad de gestionar la cantidad de productos que puede fabricar el artesano a partir del stock de materiales.

Planificación semana 6

Fecha

18 – 22 de mayo de 2015

Tareas

Añadir materiales a las variaciones de los productos prestashop para que el artesano pueda modificar la cantidad de material necesario dependiendo de la variación que él establezca

Desarrollar el código para las Estadísticas

Planificación semana 7

Fecha

25 – 29 de mayo de 2015

Tareas

Elaboración pantalla estadísticas

Pulir y corregir errores del módulo

Preparar la presentación del proyecto

Planificación semana 8

Fecha

1 – 5 de junio de 2015

Tareas

Presentación del proyecto

Subida de la página web a hosting seleccionado.

Bibliografía y web grafía

codeigniter.com. (s.f.). Recuperado el 05 de 2015, de http://www.codeigniter.com/user_guide/

db-engines. (s.f.). Recuperado el 05 de 2015, de <http://db-engines.com/en/ranking>

Kuga, J. (s.f.). *smartec.la*. Recuperado el 05 de 2015, de <http://www.smartec.la/blog/por-que-usar-un-framework>

phonegap.com. (s.f.). Obtenido de <http://phonegap.com/about/feature/>

SitePoint. (s.f.). Recuperado el 05 de 2015, de <http://www.sitepoint.com/best-php-framework-2015-sitepoint-survey-results/>

W3Techs - World Wide Web Technology Surveys. (s.f.). Recuperado el 05 de 2015, de http://w3techs.com/technologies/overview/programming_language/all

Wikipedia. (s.f.). Recuperado el 06 de 2015, de <http://es.wikipedia.org/wiki/Scrum>

Wikipedia. (s.f.). *wikipedia*. Recuperado el 05 de 2015, de http://es.wikipedia.org/wiki/Inyecci%C3%B3n_SQL

Zahasman, T. (s.f.). *Oentrepreneur*. Recuperado el 05 de 2015, de <http://web2.0entrepreneur.com/7/why-you-should-use-a-web-application-framework.html>

Author: Nicolas Alphonse

Created: 2015-06-02 Tue 11:03

Emacs 24.5.1 (Org mode 8.2.10)

Validate