|  |
| --- |
| Logo  **UNIVERSIDAD EUROPEA DE MADRID**    ESCUELA POLITÉCNICA  CICLO FORMATIVO DE GRADO SUPERIOR DESARROLLO DE APLICACIONES MULTIPLATAFORMA |
| PROYECTO FIN DE CICLO |
| **[Escribir el título del proyecto]** |
|  |
| **Autores** |
| **CURSO 2014-15** |

**TÍTULO**: -----------------------------------------------------------

**AUTOR**: NOMBRE APELLIDO1 APELLIDO2

**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 descargá 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 agradece el presente documente a su dueño y señor Nicolas Alphonse. Sin el este proyecto no habría salido adelante.

Nicolas Alphonse: Mucho trabajo duro detrás de este documente. 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

Contenido

[RESUMEN 5](#_Toc421008219)

[ABSTRACT 5](#_Toc421008220)

[Agradecimientos 7](#_Toc421008221)

[Introducción 13](#_Toc421008222)

[Objetivos 14](#_Toc421008223)

[Entender la importancia de escoger una tecnología backend apropiada 15](#_Toc421008224)

[Como realizar un página web responsive 16](#_Toc421008225)

[Alternativas de web hosting, hacer una aplicación SEO friendly. 17](#_Toc421008226)

[Motivación 17](#_Toc421008227)

[Antecedentes 18](#_Toc421008228)

[Desarrollo de la práctica 19](#_Toc421008229)

[Material 19](#_Toc421008230)

[PHP y MySQL 19](#_Toc421008231)

[Framework codeigniter php 20](#_Toc421008232)

[twitter bootstrap 21](#_Toc421008233)

[api rest 21](#_Toc421008234)

[Base de datos en prestashop 21](#_Toc421008235)

[Comunicarse con prestashop mediante su web api 21](#_Toc421008236)

[Planificación 21](#_Toc421008237)

[Descripción del trabajo realizado 22](#_Toc421008238)

[Resultados y validación 22](#_Toc421008239)

[Conclusiones 22](#_Toc421008240)

[Aportaciones 22](#_Toc421008241)

[Trabajo futuro 22](#_Toc421008242)

[Bibliografía y webgrafía 22](#_Toc421008243)

# Introducción

Nuestro proyecto de fin de ciclo trata sobre una solución web multiplataforma para que aquellos artesanos que venden sus productos a través de prestashop puedan manejar su stock de materiales de una manera cómoda y sencilla y que además esté sincronizada con su tienda. En nuestra aplicación web, los artesanos podrán 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 onlines que usan prestashop y cada vez se está haciendo más popular.

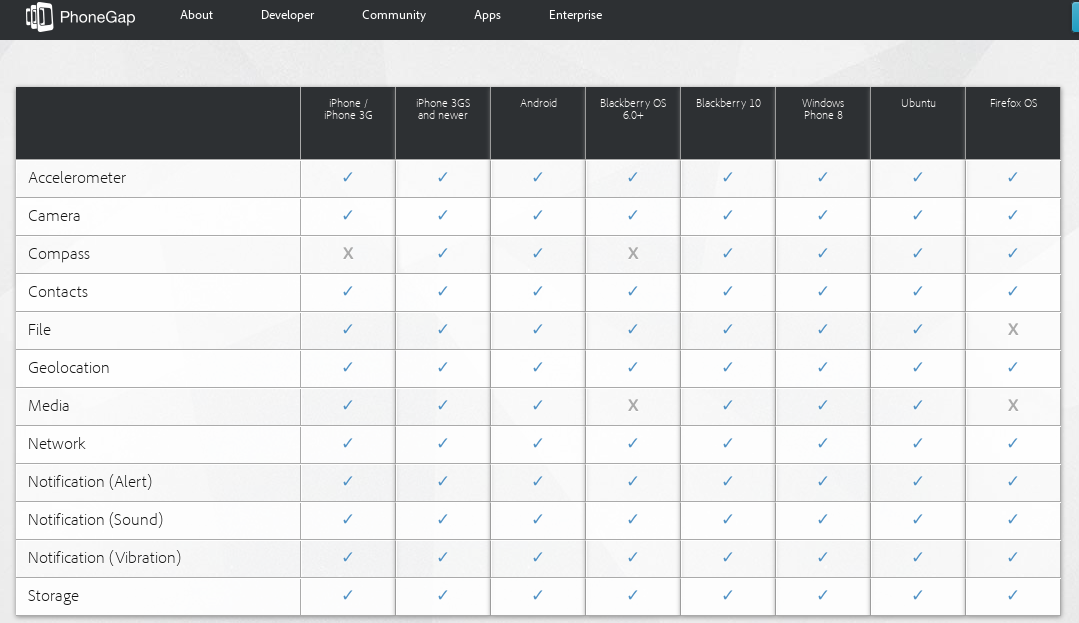
Además es de fácil uso y está destinado a personas que tiene tiendas de tamaño pequeño-medio, de entre 500/1000 unidades a la venta. Estas fueron las principales razones por las que en un principio nos decidimos a crear un módulo para este. Es open source, es de fácil uso y está destinado a pequeñas tiendas. El principal problema que nos encontramos cuando estábamos creando el módulo es la escasa documentación que hay sobre como hacerlo, ya que aunque prestashop es open source, gana dinero de dos maneras: mediante servicios de hosting, y mediante cursos para que los usuarios aprendan cómo usarlo y que los desarrolladores aprendan cómo funciona.

Fue entonces cuando decidimos que en vez de crear un módulo deberíamos crear una web app que se comunicará con prestashop mediante su web service. Al final fue una buena decisión, ya que haciendo una página web ajena a prestashop no solo implicaba no tener que usar sus librerías php, las cuales como ya hemos dicho tienen una documentación justa, sino que también nos abría las puertas con otras posibilidades que no están contempladas en ellas. Por ejemplo, hacer que nuestra app no solo se comunique con prestashop, sino con otros CRM también destinados a la creación de tiendas online. De ésta manera, multiplicamos nuestros usuarios potenciales.

Además, si nos hubiésemos ceñido a crear el módulo, tan sólo habríamos usado php, html y css. Creando nuestra propia web desde cero no solo íbamos a aprender esas tres, sino que también nos veríamos obligados a usar tecnologías que están en boga tales como javascript y jquery, y lo más importante, aprender cómo crear páginas web para que se puedan ver bien en 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 inviables en un pasado.



Durante los dos años que han durado los estudios nos hemos centrado en la programación java. Consideramos interesante y de un valor añadido el aprender a realizar aplicaciones webs multidispositivo.

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

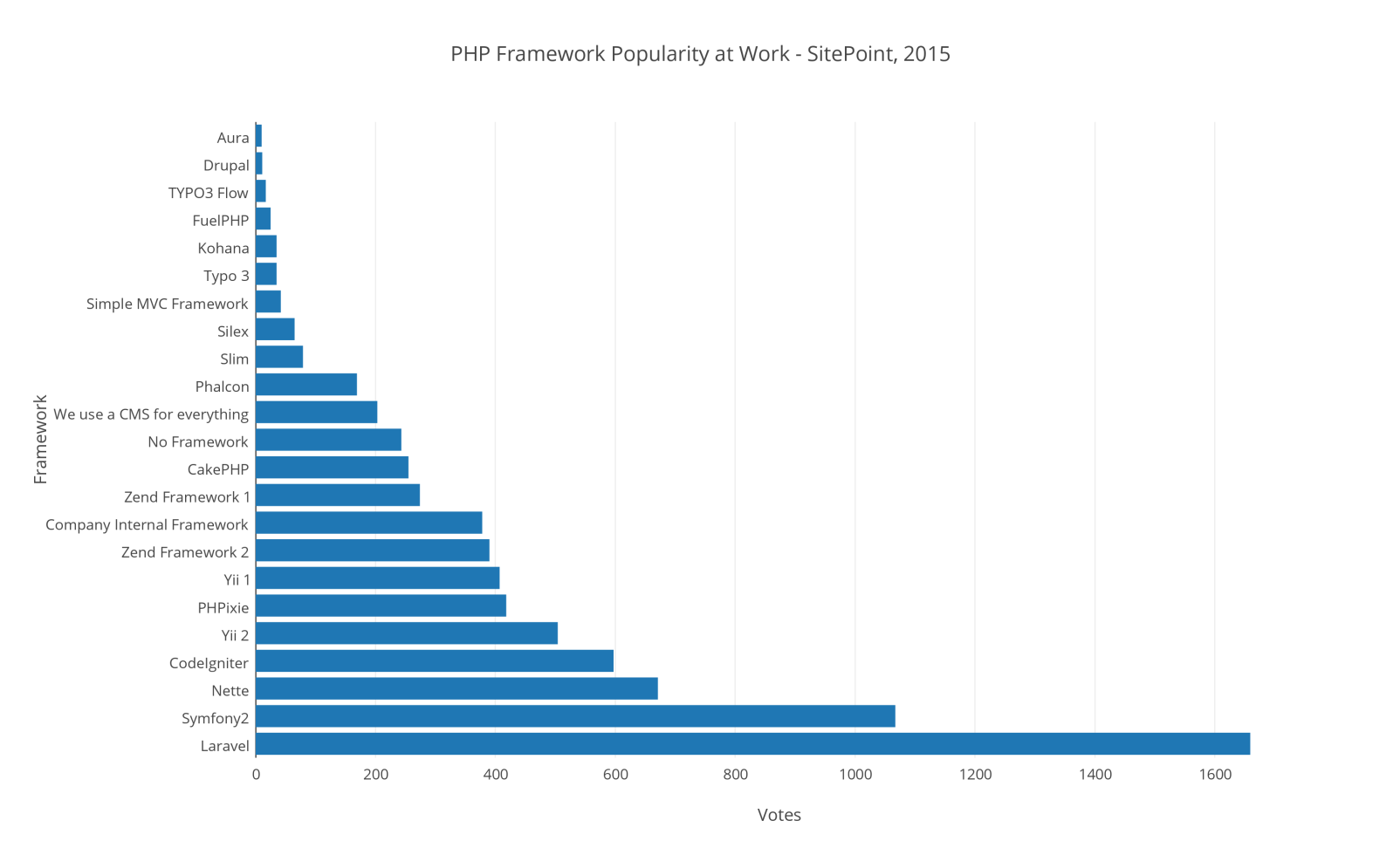
* Investigación sobre las diferentes tecnologías que hay para realizar el back end de un página web, cuales son las ventajas y desventajas de usar unos frente a otros. Porque es ampliamente usado PHP
* Como maquetar para que la págian web pueda verse correctamente en diferentes tamaños de pantallas.
* Las diferentes plataformas de web hosting que hay disponibles y cuales 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 backend apropiada

Una de las decisiones mas importantes a la hora de crear un producto, es escoger la pila de tecnologías en la que te vas a basar. El backend de una aplicación es la capa de software que el usario final no ve, pero no por ello es menos importante. El backend 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 backend 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. Acutalmente, según W3tech[6](file:///C:\xampp\htdocs\craftandbudget\proyecto\craftandbudget.html#porcentajewebsphp), PHP es usado por el 81.9% de páginas web del plantea.

En el mundo profesiona generalmente se usa php mediante un framework para facilitar el proceso de creción de software. Por ello decidimos usar uno para experimentar de primera mano la ventajas que ofrece. Finalmente elegimos el framework Codeigniter el cual esta 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. [6](file:///C:\xampp\htdocs\craftandbudget\proyecto\craftandbudget.html#phpframeworkgraph)



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 barajeabamos 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 es 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 proveenir 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 decidimos 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.[6](file:///C:\xampp\htdocs\craftandbudget\proyecto\craftandbudget.html#basesdatosranking)

### 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 posile que lo tenga a mano.

Para la maquetación usaremos el lenguaje de marcas HTML junto con css. De igual modo, que para el backend, 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 espectativas 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 mas '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 varios 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 asi 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 todos ellas esta teniendo mucho éxito prestashop, la cuál 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 otra 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 prodcutor a diferentes tiendas online. De esta manera tendríamos mas usuario 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 appi 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 managment system) rápida, y fácil de usar que se usa en la mayoría de páginas web.

Las principales razones por las que nos hemos decidido ha escoger php y MySQL como backend 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.

### Framework codeigniter php

Para la parte del servidor, decidimos usar el framework de PHP Codeigniter. Queríamos asegurarnos de que desarrollábamos 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 encargar 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 separa la lógica de las interfaces de usuario de manera que la 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 separa 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 defectos 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&crcat=ppc&crsource=google&crkw=buy-a-lot*"

Codeigniter usa un sistema personalizado en el que es mucho mas legible, y es comprensible lo que hace la url, ya que como ya se detallarás 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.

### 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 nuestro caso, ya que no teníamos ninguna experiencia en el desarrollo de interfaces, ni en la programación front end nos decidimos a usar una plantilla de twitter bootstrap. De está manera nos podemos enfocar en lo que realmente sabemos, la programación backend, sin tener que centrarnos en como diseñar páginas web responsive. En nuestro caso, como se trataba de 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 nos decidimos por “INSPINA admin theme”. Nos los descargamos de la página <https://wrapbootstrap.com/theme/inspinia-responsive-admin-theme-WB0R5L90S>, donde hay multitud de plantillas para escoger. Esa misma plantilla está en cinco versiones: asp.net, ruby on rails, angular js, meteor y html/javascript. Nosotros hemos cogido está última y toda la parte dinámica se la dejamos a 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 appi prestashop

La web appi de prestashop usa la arquitectura REST, de está 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 appi ofrece a los desarrolladores es un sistema CRUD para su base de datos. De está 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 |

## Planificación

## Descripción del trabajo realizado

## Resultados y validación

# Conclusiones

## Aportaciones

## Trabajo futuro

# 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.). *wikipedia*. Recuperado el 05 de 2015, de http://es.wikipedia.org/wiki/Inyecci%C3%B3n\_SQL

Zahasman, T. (s.f.). *0entrepreneur*. 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](http://www.gnu.org/software/emacs/) 24.5.1 ([Org](http://orgmode.org) mode 8.2.10)

[Validate](http://validator.w3.org/check?uri=referer)