Universidad de Sonora
División de ingeniería
Departamento de ingeniería industrial

High Def Digest

Memoria de prácticas profesionales

Que como requisito parcial para la obtención del título de:

Ingeniero En Sistemas De Información

Por: Gabriel Gonzalez Quijada

Director: Dr. José Luis Ochoa Hernández

Hermosillo, Sonora Mayo 2016

Índice

1. Introducción

1.1 Breve explicación del proyecto

Fundado en Abril del 2006, High Def Digest (highdefdigest.com) es una guía para los amantes del entretenimiento en el hogar que exigen el mejor contenido y equipo de alta definición, se actualiza diariamente por un equipo de 15 editores y su nicho son las plataformas de alta definición, los principales temas del sitio son los siguientes:

- Blu-ray
- Disco digital versátil de alta densidad (HD DVD)
- Equipo de alta definición
- Juegos
    - 3DS
    - Android
    - iOS
    - PC
    - PS Vita
    - PS3
    - PS4
    - Wii U
    - Xbox 360
    - Xbox One
- Ultra alta definición
    - SH
    - FHD
    - 4K UHD
    - 8K UHD

Fue adquirido por Internet Brands desde Enero del 2008, una compañía que se dedica a la compra de sitios web.

El sitio actualmente cuenta con la misma tecnología e infraestructura desde sus inicios que en aquel entonces eran óptimos porque el sitio era pequeño, pero desde que fue adquirido por Internet Brands ha crecido bastante, más que nada por la buena inversión de contenido que es publicado diariamente. Las visitas han pasado de miles a millones diarias, durante el último año estas visitas han estado

disminuyendo porque el sitio deja de funcionar correctamente debido a la mala infraestructura de servidores y el código con el que fue desarrollado, esta es la razón por la cual el sitio será migrado desde cero a una estructura más limpia, organizada y escalable.

1.2 Objetivo general

Realizar la migración completa del sitio web actual que se encuentra basado en código espagueti al patrón de desarrollo MVC *"Modelo Vista Controlador",* mejorando el Incremento del tráfico, el posicionamiento en buscadores y reducir el tiempo de carga del sitio.

1.3 Sub-objetivos específicos

Para poder llevar a cabo esta migración se realizarán las siguientes actividades:

- Análisis de la estructura del sitio actual
- Análisis y diseño de la nueva base de datos
- Migración de datos
- Desarrollo del frontend para el sitio público
- Desarrollo del backend para el sitio público
- Desarrollo del frontend para el administrador
- Desarrollo del backend para el administrador
- Control de calidad
- Lanzamiento a producción

1.4 Línea metodológica

En general se necesita la migración de un sitio web basado en código espagueti a un patrón MVC *"Modelo Vista Controlador",* el código espagueti está basado en Ruby y será migrado a PHP usando el framework CodeIgniter.

1.5 Cronograma de actividades

Actividades	**Fecha inicio**	**Fecha final**	**Duración**
Análisis de la estructura del sitio actual	01/02/16	05/02/16	5 días
Análisis y diseño de la nueva base de datos	08/02/16	12/02/16	5 días

Migración de datos	15/02/16	19/02/16	5 días
Desarrollo del frontend para el sitio público	22/02/16	26/02/16	5 días
Desarrollo del backend para el sitio público	29/02/16	04/03/16	5 días
Desarrollo del frontend para el administrador	07/03/16	11/03/16	5 días
Desarrollo del backend para el administrador	14/03/16	18/03/16	5 días
Control de calidad	21/03/16	29/03/16	7 días
Lanzamiento a producción	30/03/16	30/03/16	1 día

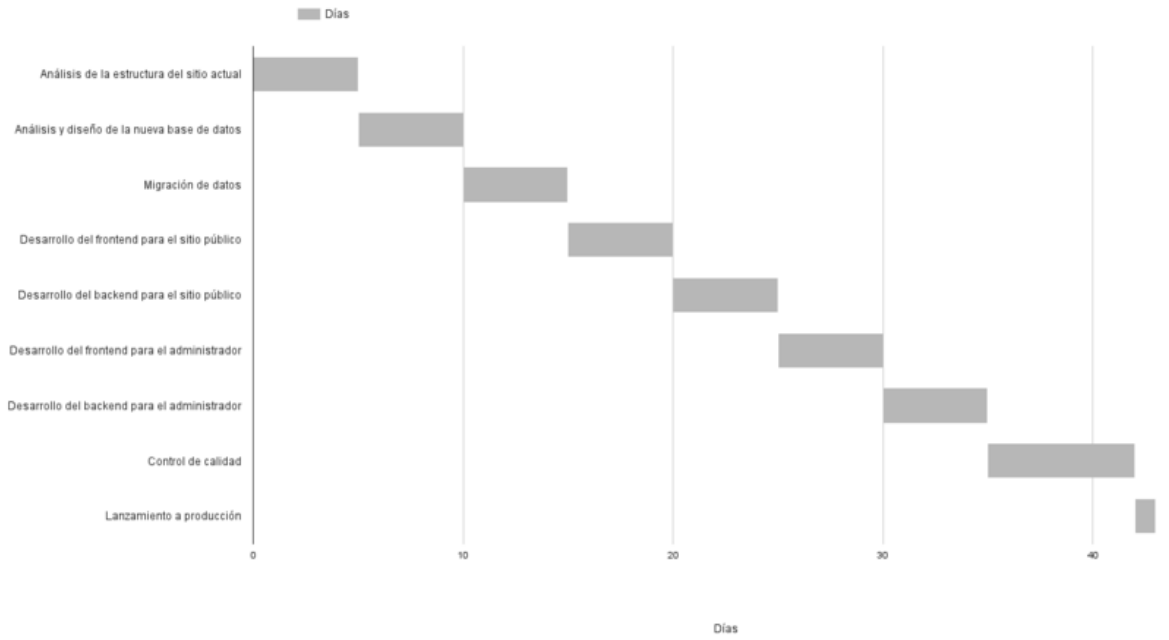

Figura 1. Grafica de Gantt

2. Descripción del contexto

2.1 Equipamiento e instalaciones donde se desarrollaron las actividades que integra el programa de prácticas profesionales

Estas prácticas profesionales fueron desarrolladas en Vangtel, la cual es una compañía que se dedica a prestar servicios de operaciones, recursos humanos y reclutamiento en México para empresas extranjeras, el equipamiento usado en el desarrollo de estas prácticas profesionales fue de:

- 1 computadora
- 2 equilibradores de carga
- 8 servidores web

2.2 Descripción de la normatividad o reglas de operación del programa o unidad receptora

A continuación una lista con los aspectos más importantes del reglamento interno de trabajo de Vangtel:

- Las jornadas de trabajo diurno será de 40 horas por semana, 35 horas en la jornada mixta y 32 horas en la jornada nocturna.
- La empresa concederá en cada jornada un periodo de ½ hora durante el cual los trabajadores tomaran sus alimentos en el comedor de la empresa o en sus domicilios particulares.
- Cuando el trabajador esté imposibilitado para presentarse a trabajar deberá notificarlo a la empresa antes de que comience la jornada de trabajo o a más tardar 3 horas antes después de iniciada la misma.
- Es obligación del trabajador presentarse puntualmente a la hora de entrada y salida de su trabajo.
- Todo solicitante de empleo, así como los trabajadores contratados por la empresa, están obligados a someterse a un examen médico, así como a un análisis de laboratorio antes de ser contratados.
- La empresa podrá rescindir de trabajo con cualquiera de sus trabajadores que cometa uno o más actos estipulados en el artículo 47 de la ley.
- La empresa solo pagará a los trabajadores el salario que estrictamente les corresponde por las horas efectivamente trabajadas en cada jornada consecuente.
- En todo lo previsto en este reglamento interior de trabajo, las partes se arreglaran conforme a las disposiciones de la ley.

2.3 Entorno donde se ubica la unidad receptora

Actualmente Vangtel tiene oficinas en: Obregón Sonora, Hermosillo Sonora Y Guadalajara Jalisco, estas prácticas fueron desarrolladas con el cliente “Internet Brands” en las oficinas de Hermosillo Sonora con domicilio en boulevard Solidaridad #56 colonia Sahuaro.

3. Fundamento teórico de las herramientas y conocimientos aplicados

3.1. Apache

Apache es un servidor web, cuyo nombre proviene de la frase inglesa *"a patchy server"* y es completamente libre, ya que es un software Open Source y con licencia GPL. Una de las ventajas más grandes de Apache, es que es un servidor web multiplataforma, es decir, puede trabajar con diferentes sistemas operativos y mantener su excelente rendimiento.

3.2. Base de datos

Una base de datos es el conjunto de datos informativos organizados en un mismo contexto para su uso y vinculación. Se le llama base de datos a los bancos de información que contienen datos relativos a diversas temáticas y categorizados de distinta manera, pero que comparten entre sí algún tipo de vínculo o relación que busca ordenarlos y clasificarlos en conjunto.

3.3. CDN

CDN significa *"Content Delivery Network"*, y es una red de servidores diseñados para entregar en caché (de forma estática) el contenido de los sitios web a los visitantes. La entrega está determinada por la ubicación de cada visitante de la web.

3.4. CSS

CSS significa *"Cascading Style Sheets"* y es un lenguaje utilizado en la presentación de documentos HTML. Entonces podemos decir que el lenguaje CSS sirve para organizar la presentación y aspecto de una página web.

3.5. GZIP

GZIP es una abreviatura de GNU ZIP y consiste en enviar los códigos de un sitio en formato comprimido, para que ocupen mucho menos espacio y por tanto se transfieran por la red de una manera más rápida.

3.6. HTML

HTML significa *"HyperText Markup Language"* y se emplea para el desarrollo de páginas de internet. Está compuesto por una serie de etiquetas que el navegador interpreta y da forma en la pantalla. HTML dispone de etiquetas para imágenes, hipervínculos que nos permiten dirigirnos a otras páginas, saltos de línea, listas, tablas, etc.

3.7. JavaScript

JavaScript es un lenguaje de programación interpretado y se define como orientado a objetos, basado en prototipos, imperativo, débilmente tipado y dinámico. Se utiliza principalmente en su forma del lado del cliente, implementado como parte de un navegador web permitiendo mejoras en la interfaz de usuario y páginas web dinámicas.

3.8. Memcached

Memcached es un sistema distribuido de propósito general para caché basado en memoria y es empleado para el almacenamiento en caché de datos u objetos en la memoria RAM, reduciendo así las necesidades de acceso a un origen de datos externo.

3.9. MVC

MVC significa *"Model View Controller"* y es un patrón de arquitectura de software que separa los datos y la lógica de negocio de una aplicación de la interfaz de usuario y el módulo encargado de gestionar los eventos y las comunicaciones

3.10. MySQL

MySQL es un sistema de gestión de bases de datos relacional y está considerada como la base datos open source más popular del mundo y una de las más populares en general, sobre todo para entornos de desarrollo web.

3.11. PHP

PHP significa "Hypertext Preprocessor" y es un lenguaje de código abierto muy popular especialmente adecuado para el desarrollo web y que puede ser incrustado en HTML.

3.12. Servidor web

Es un programa especialmente diseñado para transferir datos de hipertexto, es decir, páginas web con todos sus elementos. Estos servidores web utilizan el protocolo HTTP. Los servidores web están alojados en un ordenador que cuenta con conexión a Internet. El servidor web, se encuentra a la espera de que algún navegador le haga alguna petición, como por ejemplo, acceder a una página web y responde a la petición, enviando código HTML, JSON, XML u otros formatos mediante una transferencia de datos en red.

3.13. Ruby

Ruby es un lenguaje de programación interpretado en una sola pasada y su implementación oficial es distribuida bajo una licencia de software libre.

3.14. CodeIgniter

CodeIgniter es un framework para aplicaciones web de código abierto para crear sitios web dinámicos con PHP. Su objetivo es permitir que los desarrolladores puedan realizar proyectos mucho más rápido que creando toda la estructura desde cero, brindando un conjunto de bibliotecas para tareas comunes, así como una interfaz simple y una estructura lógica para acceder esas bibliotecas.

3.15. Equilibrador de carga

El balance o balanceo de carga se refiere a la técnica usada para compartir el trabajo a realizar entre varios procesos, computadoras, discos u otros recursos. Está íntimamente ligado a los sistemas de multiprocesamiento, o que hacen uso de más de una unidad de procesamiento para realizar labores útiles.

3.16. SVN

Apache subversion (abreviado frecuentemente como SVN, por el comando svn) es una herramienta de control de versiones open source basada en un repositorio cuyo funcionamiento se asemeja enormemente al de un sistema de archivos. Es software libre bajo una licencia de tipo Apache/BSD.

3.17. Git

Git es un software de control de versiones diseñado por Linus Torvalds, pensando en la eficiencia y la confiabilidad del mantenimiento de versiones de aplicaciones cuando éstas tienen un gran número de archivos de código fuente.

4. Descripción detallada de las actividades realizadas

4.1. Análisis de la estructura del sitio actual

Las tecnologías que usa el sitio actualmente son las siguientes:

- Apache 1.6
- MySQL 3.0
- RoR 1.1.1

Y sus secciones principales son las siguientes:

- Sitio público
    - Críticas
    - Noticias
    - Fechas de lanzamiento
    - Pre ordenes
    - Páginas estáticas
- Administrador del sitio
    - Críticas
    - Noticias
    - Páginas estáticas
    - Usuarios
    - Imágenes

Después de un análisis detallado hemos decidido usar las siguientes tecnologías en la migración:

- PHP 5.3.23
- CodeIgniter 2.1.3
- Apache 2.2.15
- MySQL 5.5.24

4.2. Carga del sitio

Como mencionamos anteriormente la carga del sitio es bastante lenta, después de un análisis encontramos los siguientes problemas:

- Todos los recursos web cargan del mismo dominio "http://www.highdefdigest.com", bloqueando la carga asíncrona de los recursos
- Imágenes no están optimizadas
- Descarga de múltiples archivos CSS
- Descarga de múltiples archivos JS
- Archivos CSS no están minimizados
- Archivos JS no están minimizados
- Archivos HTML no están minimizados
- No se usa compresión GZIP
- Caché del navegador no está optimizada
- Bastante uso de CSS y JS en línea

Para arreglar estos problemas usaremos las siguientes técnicas y métodos:

- Creación de un subdominio "http://cdn.highdefdigest.com" para descargar los recursos como CSS, JS e imágenes
- Creación de un script PHP para redimensionar imágenes y guardar copias en diferentes tamaños en el servidor con las siguientes dimensiones: 120, 195, 235, 300, 660
- Creación de un script PHP para minimizar y comprimir los archivos CSS
- Creación de un script PHP para minimizar y comprimir los archivos JS
- Creación de un script PHP para minimizar y comprimir los archivos HTML
- Configurar la cache GZIP en el servidor
- Configurar la cache de los recursos en el servidor
- Remover el uso de CSS y JS en línea y moverlo a sus respectivos archivos CSS y JS
- Uso de Memcached

Tenemos planeado que con estos cambios la carga del sitio pase de 9 segundos que está actualmente entre 1 y 2 segundos beneficiando la experiencia del usuario en el sitio.

4.3. Infraestructura de servidores

A continuación, se muestra la estructura de servidores con la que el sitio funciona actualmente:

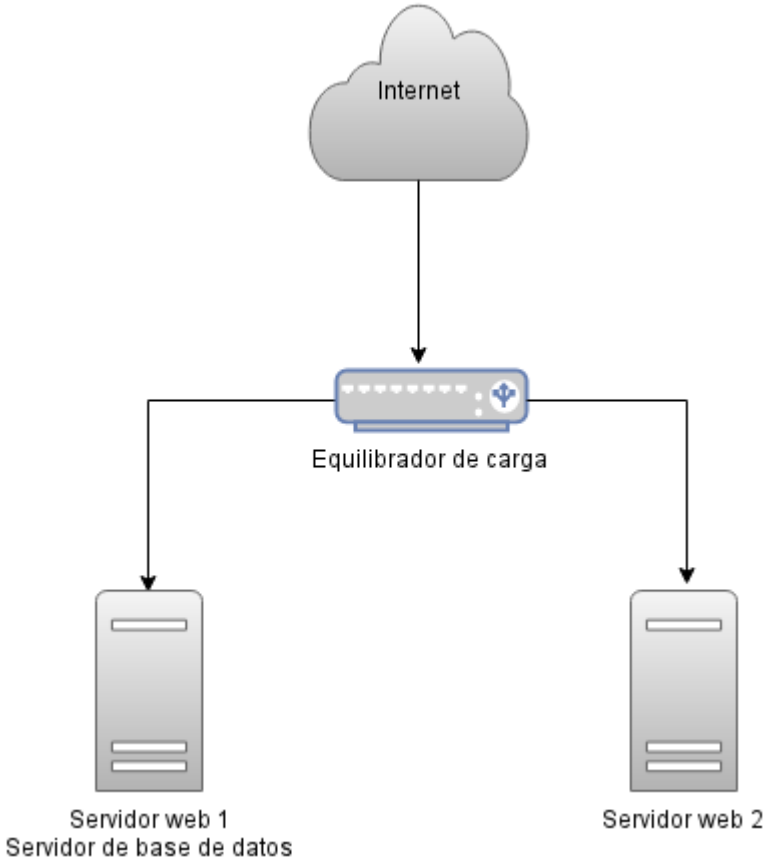

Figura 2. Estructura de servidores actual

Debido al tráfico que el sitio tiene diariamente, el diseño actual de estos servidores Figura 2, no son suficientes, a continuación, se muestra la nueva estructura de servidores en la migración Figura 3:

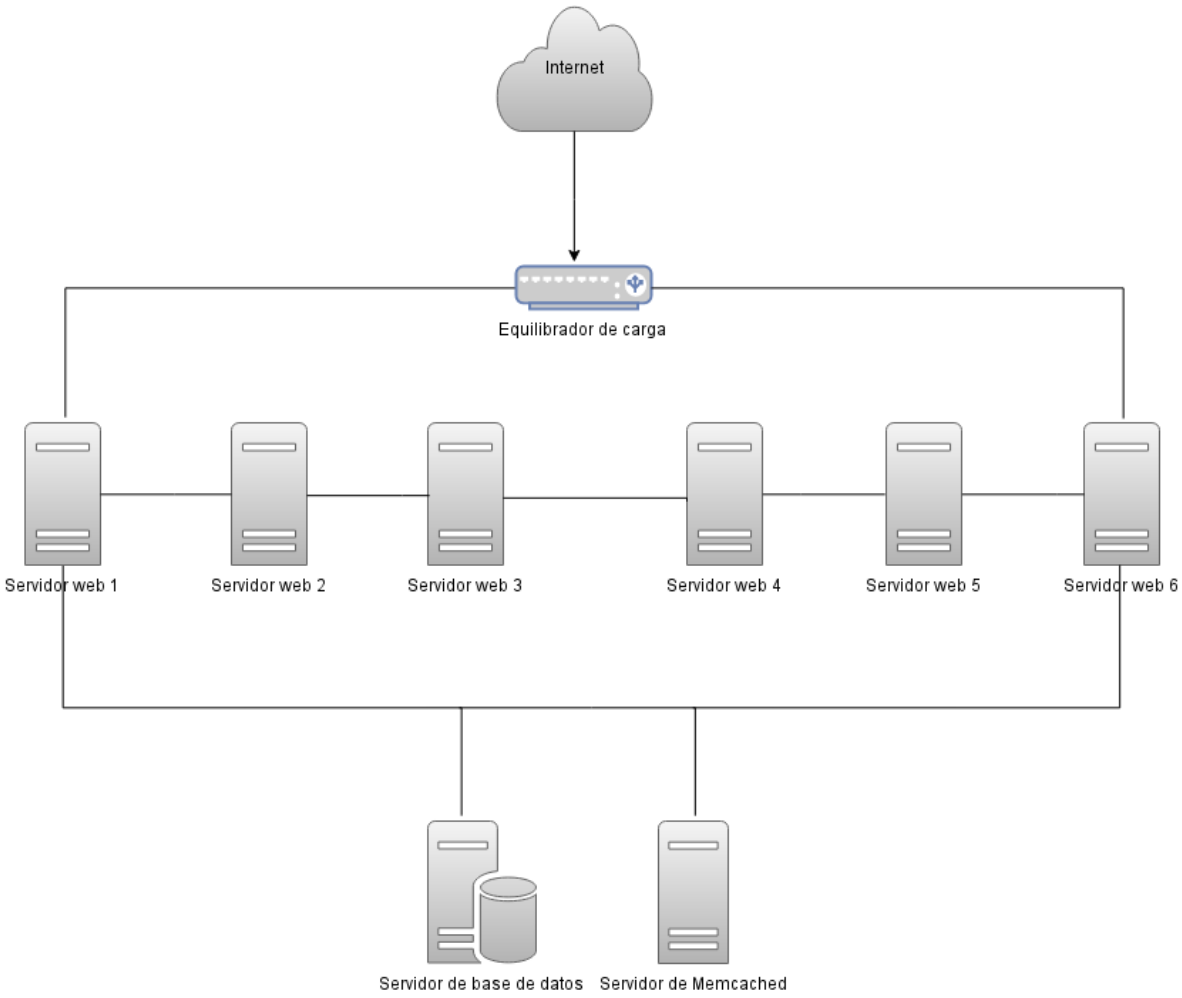

Figura 3. Nueva estructura de servidores

4.4. Base de datos

Actualmente la base de datos está mal diseñada, hacienda que el trabajo con ella sea muy complicado, por ejemplo, si quisiéramos agregar una nueva sección tendríamos que agregar muchas tablas nuevas, afectando el rendimiento del servidor MySQL, adicionalmente el servidor de base de datos deja de funcionar correctamente por la cantidad de consultas y el excesivo tiempo que toma en arrojar resultados, a continuación, se muestra el diagrama actual de la base de datos:

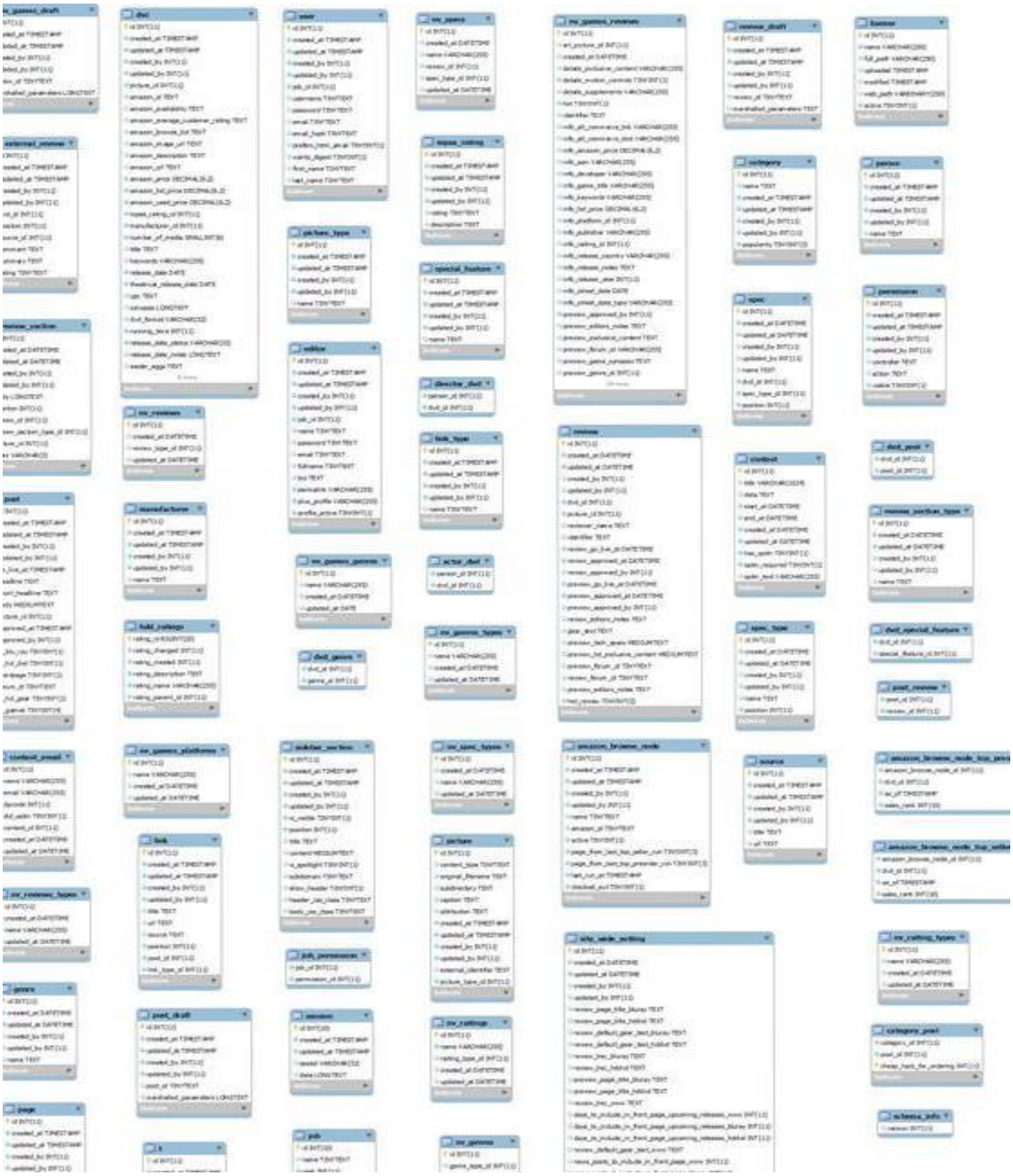

Figura 4. Diseño de base de datos actual

Como se puede observar en la Figura 4, existen muchas tablas y algunas no cuentan con índices ni llaves primarias.

A continuación el nuevo diseño de la base de datos que es más compacto y extensible:

Figura 5. Nuevo diseño de la base de datos

Este nuevo diseño de la Figura 5, hará más fácil el poder trabajar con la base de datos y mejoraremos la rapidez de las consultas, además ésta es mucho más sencilla y compacta, adicionalmente podremos agregar un sin número de tipos de contenido sin necesidad de crear más tablas, a continuación, se muestra el script SQL para crear el esquema de la base de datos:

```
CREATE TABLE `hdd_countries` (
  `country_id` bigint(20) NOT NULL AUTO_INCREMENT,
  `country_code` varchar(2) NOT NULL,
  `country_name` varchar(255) NOT NULL,
  PRIMARY KEY (`country_id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8;

CREATE TABLE `hdd_game_genres` (
  `game_genre_id` bigint(20) NOT NULL AUTO_INCREMENT,
  `game_genre_changed` int(11) NOT NULL,
  `game_genre_created` int(11) NOT NULL,
```

```
  `game_genre_name` varchar(255) CHARACTER SET latin1 NOT NULL,
  PRIMARY KEY (`game_genre_id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8;

CREATE TABLE `hdd_game_platforms` (
  `game_platform_id` bigint(20) NOT NULL AUTO_INCREMENT,
  `game_platform_changed` int(11) NOT NULL,
  `game_platform_created` int(11) NOT NULL,
  `game_platform_name` varchar(255) CHARACTER SET latin1 NOT NULL,
  PRIMARY KEY (`game_platform_id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8;

CREATE TABLE `hdd_genres` (
  `genre_id` bigint(11) NOT NULL AUTO_INCREMENT,
  `genre_changed` int(11) NOT NULL,
  `genre_created` int(11) NOT NULL,
  `genre_name` varchar(255) CHARACTER SET latin1 NOT NULL,
  `user_id` bigint(20) NOT NULL,
  PRIMARY KEY (`genre_id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8;

CREATE TABLE `hdd_groups` (
  `group_id` bigint(20) NOT NULL AUTO_INCREMENT,
  `group_changed` int(11) NOT NULL,
  `group_created` int(11) NOT NULL,
  `group_name` varchar(255) NOT NULL,
  PRIMARY KEY (`group_id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8;

CREATE TABLE `hdd_manufacturers` (
  `manufacturer_id` bigint(20) NOT NULL AUTO_INCREMENT,
  `manufacturer_changed` int(11) NOT NULL,
  `manufacturer_created` int(11) NOT NULL,
  `manufacturer_name` varchar(255) CHARACTER SET latin1 NOT NULL,
  `user_id` bigint(20) NOT NULL,
  PRIMARY KEY (`manufacturer_id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8;

CREATE TABLE `hdd_pages` (
  `page_id` bigint(11) NOT NULL AUTO_INCREMENT,
  `page_active` tinyint(1) NOT NULL DEFAULT '0',
  `page_changed` int(11) NOT NULL,
  `page_created` int(11) NOT NULL,
  `page_description` text CHARACTER SET latin1 NOT NULL,
  `page_slug` varchar(255) CHARACTER SET latin1 NOT NULL,
  `page_title` varchar(255) CHARACTER SET latin1 NOT NULL,
  `user_id` bigint(20) NOT NULL,
  PRIMARY KEY (`page_id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8;

CREATE TABLE `hdd_people` (
  `person_id` bigint(20) NOT NULL AUTO_INCREMENT,
  `person_changed` int(11) NOT NULL,
  `person_created` int(11) NOT NULL,
  `person_name` varchar(255) CHARACTER SET latin1 NOT NULL,
  `user_id` bigint(20) NOT NULL,
  PRIMARY KEY (`person_id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8;

CREATE TABLE `hdd_pictures` (
  `picture_id` bigint(20) NOT NULL AUTO_INCREMENT,
  `picture_alternate` varchar(255) CHARACTER SET latin1 NOT NULL,
  `picture_changed` int(11) NOT NULL,
```

```
  `picture_created` int(11) NOT NULL,
  `picture_name` varchar(255) CHARACTER SET latin1 NOT NULL,
  `picture_title` varchar(255) CHARACTER SET latin1 NOT NULL,
  `user_id` int(11) NOT NULL,
  PRIMARY KEY (`picture_id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8;


CREATE TABLE `hdd_post_reviews` (
  `post_review_id` bigint(20) NOT NULL AUTO_INCREMENT,
  `post_id` bigint(20) NOT NULL,
  `review_id` bigint(20) NOT NULL,
  PRIMARY KEY (`post_review_id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8;


CREATE TABLE `hdd_post_subdomains` (
  `post_subdomain_id` bigint(20) NOT NULL AUTO_INCREMENT,
  `post_id` bigint(20) NOT NULL,
  `subdomain_id` bigint(20) NOT NULL,
  PRIMARY KEY (`post_subdomain_id`),
  KEY `fk_delete_post_subdomains` (`post_id`),
  KEY `post_id` (`post_id`),
  KEY `post_subdomain_id` (`post_subdomain_id`),
  CONSTRAINT `fk_delete_subdomains` FOREIGN KEY (`post_id`) REFERENCES `hdd_posts` (`post_id`) ON
DELETE CASCADE
) ENGINE=InnoDB DEFAULT CHARSET=utf8;


CREATE TABLE `hdd_post_tags` (
  `post_tag_id` bigint(20) NOT NULL AUTO_INCREMENT,
  `post_id` bigint(20) NOT NULL,
  `tag_id` bigint(20) NOT NULL,
  PRIMARY KEY (`post_tag_id`),
  KEY `fk_delete_post_tags` (`post_id`),
  KEY `post_id` (`post_id`),
  KEY `post_tag_id` (`post_tag_id`),
  CONSTRAINT `fk_delete_tags` FOREIGN KEY (`post_id`) REFERENCES `hdd_posts` (`post_id`) ON DELETE
CASCADE
) ENGINE=InnoDB DEFAULT CHARSET=utf8;


CREATE TABLE `hdd_posts` (
  `post_id` bigint(20) NOT NULL AUTO_INCREMENT,
  `post_active` tinyint(1) NOT NULL DEFAULT '0',
  `post_changed` int(11) NOT NULL,
  `post_created` int(11) NOT NULL,
  `post_description` mediumtext CHARACTER SET latin1 NOT NULL,
  `post_publish` int(11) NOT NULL,
  `post_slug` varchar(255) CHARACTER SET latin1 NOT NULL,
  `post_title` varchar(255) CHARACTER SET latin1 NOT NULL,
  `autosave_post_id` bigint(20) DEFAULT NULL,
  `forum_id` bigint(20) NOT NULL,
  `picture_id` bigint(20) NOT NULL,
  `user_id` bigint(20) NOT NULL,
  PRIMARY KEY (`post_id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8;


CREATE TABLE `hdd_ratings` (
  `rating_id` bigint(20) NOT NULL AUTO_INCREMENT,
  `rating_changed` int(11) NOT NULL,
  `rating_created` int(11) NOT NULL,
  `rating_description` text CHARACTER SET latin1 NOT NULL,
  `rating_name` varchar(255) CHARACTER SET latin1 NOT NULL,
  `rating_parent_id` int(11) NOT NULL,
  PRIMARY KEY (`rating_id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8;
```

```
CREATE TABLE `hdd_review_metadata` (
  `review_metadata_id` bigint(20) NOT NULL AUTO_INCREMENT,
  `review_metadata_key` varchar(255) CHARACTER SET latin1 NOT NULL,
  `review_metadata_value` text CHARACTER SET latin1 NOT NULL,
  `review_id` bigint(20) NOT NULL,
  PRIMARY KEY (`review_metadata_id`),
  KEY `review_id` (`review_id`),
  CONSTRAINT `fk_delete_review_metadata` FOREIGN KEY (`review_id`) REFERENCES `hdd_reviews`
(`review_id`) ON DELETE CASCADE
) ENGINE=InnoDB DEFAULT CHARSET=utf8;

CREATE TABLE `hdd_review_types` (
  `review_type_id` bigint(20) NOT NULL AUTO_INCREMENT,
  `review_type_name` varchar(255) CHARACTER SET latin1 NOT NULL,
  PRIMARY KEY (`review_type_id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8;

CREATE TABLE `hdd_reviews` (
  `review_id` bigint(20) NOT NULL AUTO_INCREMENT,
  `review_changed` int(11) NOT NULL,
  `review_created` int(11) NOT NULL,
  `review_hot` tinyint(1) NOT NULL DEFAULT '0',
  `review_preview_publish` int(11) NOT NULL DEFAULT '0',
  `review_release_date` int(11) NOT NULL,
  `review_release_date_status` int(11) NOT NULL,
  `review_review_publish` int(11) NOT NULL DEFAULT '0',
  `review_slug` varchar(255) NOT NULL,
  `review_tags` text NOT NULL,
  `review_title` varchar(255) NOT NULL,
  `review_type_id` bigint(20) NOT NULL,
  `country_id` bigint(20) NOT NULL,
  `manufacturer_id` bigint(20) NOT NULL,
  `picture_id` bigint(20) NOT NULL,
  `rating_id` bigint(20) NOT NULL,
  `user_id` bigint(20) NOT NULL,
  PRIMARY KEY (`review_id`),
  KEY `review_preview_publish` (`review_preview_publish`),
  KEY `review_review_publish` (`review_review_publish`),
  KEY `review_release_date_status` (`review_release_date_status`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8;

CREATE TABLE `hdd_settings` (
  `setting_id` bigint(20) NOT NULL AUTO_INCREMENT,
  `setting_changed` int(11) NOT NULL,
  `setting_created` int(11) NOT NULL,
  `setting_key` varchar(255) CHARACTER SET latin1 NOT NULL,
  `setting_value` text CHARACTER SET latin1 NOT NULL,
  PRIMARY KEY (`setting_id`),
  UNIQUE KEY `setting_key` (`setting_key`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8;

CREATE TABLE `hdd_slides` (
  `slide_id` bigint(20) NOT NULL AUTO_INCREMENT,
  `slide_active` tinyint(1) NOT NULL DEFAULT '0',
  `slide_changed` int(11) NOT NULL,
  `slide_created` int(11) NOT NULL,
  `slide_description` text CHARACTER SET latin1 NOT NULL,
  `slide_picture_alternate` varchar(255) CHARACTER SET latin1 NOT NULL,
  `slide_picture_name` varchar(255) CHARACTER SET latin1 NOT NULL,
  `slide_position` int(11) NOT NULL,
  `slide_title` varchar(255) CHARACTER SET latin1 NOT NULL,
  `slide_uri` text CHARACTER SET latin1 NOT NULL,
  `user_id` bigint(20) NOT NULL,
```

```
  PRIMARY KEY (`slide_id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8;

CREATE TABLE `hdd_subdomains` (
  `subdomain_id` int(11) NOT NULL AUTO_INCREMENT,
  `subdomain_name` varchar(255) CHARACTER SET latin1 NOT NULL,
  `subdomain_url` varchar(255) CHARACTER SET latin1 NOT NULL,
  PRIMARY KEY (`subdomain_id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8;

CREATE TABLE `hdd_tags` (
  `tag_id` bigint(20) NOT NULL AUTO_INCREMENT,
  `tag_changed` int(11) NOT NULL,
  `tag_created` int(11) NOT NULL,
  `tag_name` varchar(255) NOT NULL,
  `tag_slug` varchar(255) NOT NULL,
  PRIMARY KEY (`tag_id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8;

CREATE TABLE `hdd_users` (
  `user_id` bigint(20) NOT NULL AUTO_INCREMENT,
  `user_biography` text NOT NULL,
  `user_changed` int(11) NOT NULL,
  `user_created` int(11) NOT NULL,
  `user_email` varchar(255) NOT NULL,
  `user_google_plus_id` varchar(255) NOT NULL,
  `user_name` varchar(255) NOT NULL,
  `user_password` varchar(255) NOT NULL,
  `user_profile_active` tinyint(1) NOT NULL DEFAULT '0',
  `user_slug` varchar(255) NOT NULL,
  `user_username` varchar(255) NOT NULL,
  `group_id` bigint(20) NOT NULL,
  PRIMARY KEY (`user_id`),
  UNIQUE KEY `user_username` (`user_username`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8;
```

4.5. Migración de datos

Una de las razones por la que se requiere la migración de datos, es que el sistema de base de datos que se tiene actualmente cuenta con datos almacenados de mas de 10 años de antigüedad, son datos que deben de depurarse y actualizarse de ser necesario. Otra de las razones que se consideraron es que la estructura de datos manejada es muy obsoleta y requiere forzosamente una actualización, para ello, se emplearán unos scripts basados en PHP que moverán de una base de datos a otra todos los datos. A continuación, se muestra el listado de los scripts generados:

- countries.php
- game_genres.php
- game_platforms.php
- genres.php
- groups.php

- manufacturers.php
- pages.php
- people.php
- pictures.php
- posts.php
- reviews.php
- reviews_types.php
- settings.php
- tags.php
- users.php

Para realziar la migración de forma controlada, se Tiene un script shell el cual ejecutara todos los scripts PHP que migraran la base de datos.

En las pruebas realizadas se ha estado cronometrando el tiempo en el que la información es migrada a la nueva base de datos obteniendo un promedio de 6 horas, cabe recalcar que estos scripts ya han sido optimizados y este es el mejor tiempo que hemos logrado, como el tiempo es bastante grande y el sitio no puede estar en mantenimiento tanto tiempo, se tiene planeado notificar a los editores para que dejen de publicar contenido en ese tiempo, este script deberá ser ejecutado el día y hora que el sitio presenta menos tráfico los cuales son los días Martes y Miércoles entre 1:00 AM y 3:00 AM

4.6. Backend

Decidimos usar CodeIgniter porque el framework es bastante sencillo y el rendimiento es muy bueno, de esta manera si futuros programadores tendrían que agregar o modificar secciones será bastante entendible. Este framework usa el modelo MVC, el cual es basado en capas: Modelo, vista y controlador.

A continuación, se muestra la estructura de los folders y archivos con más importancia en el sitio:

```
1.- ./application
2.- ./system
3.- ./web
4.- ./web/assets
5.- ./web/index.php
```

1. En la carpeta ./application se encuentra el núcleo del framework “CodeIgniter”

2. En la carpeta ./system se encuentra todo el código PHP personalizado del sitio web
3. En la carpeta ./web se encuentra el “DocumentRoot” del sitio web
4. En la carpeta ./web/assets se encuentran todos los archivos estáticos como imágenes, CSS y JS
5. En la carpeta ./web/index.php se encuentra el “Front controller” del sitio web

A continuación, se muestra el VirtualHost del sitio:

```
 <VirtualHost *:80>
        DocumentRoot /var/www/webistrano/highdefdigest.com/current/web
        ServerName www.highdefdigest.com
        ServerAlias highdefdigest.com
        ServerAlias www.admin.highdefdigest.com
        ServerAlias admin.highdefdigest.com
        ServerAlias www.bluray.highdefdigest.com
        ServerAlias bluray.highdefdigest.com
        ServerAlias www.games.highdefdigest.com
        ServerAlias games.highdefdigest.com
        ServerAlias www.hddvd.highdefdigest.com
        ServerAlias hddvd.highdefdigest.com
        ServerAlias www.hdgear.highdefdigest.com
        ServerAlias hdgear.highdefdigest.com
        SetEnv ENVIRONMENT "production"
        CustomLog    "|/usr/sbin/cronolog    /var/log/httpd/highdefdigest.com/%Y-%m-%d-access.log"
combined env=!dontlog
        ErrorLog "|/usr/sbin/cronolog  /var/log/httpd/highdefdigest.com/%Y-%m-%d-error.log"
        <Directory /var/www/webistrano/highdefdigest.com/current/web>
                AllowOverride All
        </Directory>
        AllowEncodedSlashes On
</VirtualHost>
```

Básicamente, lo que este VirtualHost hace es mandar todas las peticiones que no son archivos estáticos al “Front controller” del sitio para ser procesado por PHP y generar una vista en HTML para el usuario.

4.6.1. Modelos

A continuación, se muestra una lista de los modelos que se usan en todo el sitio:

- amazon_model.php
- bonus_view_model.php
- country_model.php
- forums_model.php
- game_genre_model.php
- game_platform_model.php
- genre_model.php

- group_model.php
- manufacturer_model.php
- page_model.php
- person_model.php
- picture_model.php
- post_model.php
- post_review_model.php
- post_subdomain_model.php
- post_tag_model.php
- rating_model.php
- review_actor_model.php
- review_director_model.php
- review_genre_model.php
- review_metadata_model.php
- review_model.php
- review_tag_model.php
- review_type_model.php
- slide_model.php
- subdomain_model.php
- tag_model.php
- user_model.php

Un ejemplo del contenido de los modelos se puede observar en el siguiente script referente al modelo “user_model.php”:

```php
<?php

if ( ! defined('BASEPATH')) exit( 'No direct script access allowed' );

class User_model extends CI_Model
{
    public function __construct()
    {
        parent::__construct();
    }
    /**
     * Delete a user by user_id
     * @param   int     $iUserId
     * @return  boolean
     */
    public function deleteUser( $iUserId = NULL  )
    {
        if( $iUserId === NULL )
        {
            return FALSE;
        }
```

```
        $this->db->where( 'hdd_users.user_id', $iUserId );
        return $this->db->delete( 'hdd_users' );
    }
    /**
     * Get a user by user_id
     * @param   int     $iUserId
     * @return  array
     */
    public function getUserById( $iUserId = NULL )
    {
        if( $iUserId === NULL )
        {
            return FALSE;
        }

        $this->db->select( 'hdd_users.*, hdd_groups.*' );
        $this->db->from( 'hdd_users' );
        $this->db->join( 'hdd_groups', 'hdd_groups.group_id = hdd_users.group_id', 'inner' );
        $this->db->where( array( 'hdd_users.user_id' => $iUserId ) );
        $oResult = $this->db->get();
        return $oResult->row_array();
    }
    /**
     * Get a user by user_slug
     * @param   string  $sUserSlug
     * @return  string
     */
    public function getUserBySlug( $sUserSlug = NULL )
    {
        if( ( $sUserSlug === NULL ) ||
            ( ! is_string( $sUserSlug ) ) ||
            ( $sUserSlug === '' ) )
        {
            return FALSE;
        }

        $this->db->select( 'hdd_users.*, hdd_groups.*' );
        $this->db->from( 'hdd_users' );
        $this->db->join( 'hdd_groups', 'hdd_groups.group_id = hdd_users.group_id', 'inner' );
        $this->db->where( array( 'hdd_users.user_slug' => $sUserSlug ) );
        $oResult = $this->db->get();
        return $oResult->row_array();
    }
    /**
     * Get a user by user_username
     * @param   string  $sUsername
     * @return  array
     */
    public function getUserByUsername( $sUsername = NULL )
    {
        if( $sUsername === NULL )
        {
            return FALSE;
        }

        $this->db->select( 'hdd_users.*, hdd_groups.*' );
        $this->db->from( 'hdd_users' );
        $this->db->join( 'hdd_groups', 'hdd_groups.group_id = hdd_users.group_id', 'inner' );
        $this->db->where( array( 'hdd_users.user_username' => $sUsername ) );
        $oResult = $this->db->get();
        return $oResult->row_array();
    }
    /**
     * Get users
     * @param   int     $iLimit
```

```
     * @param   int     $iOffset
     * @param   string  $sLike
     * @return  array   $arUsers
     */
    public function getUsers( $iLimit = NULL, $iOffset = NULL, $sLike = NULL, $sOrderBy = NULL )
    {
        $arUsers = array();
        $arUser = array();

        $this->db->select( 'hdd_users.*, hdd_groups.*' );
        $this->db->from( 'hdd_users' );
        $this->db->join( 'hdd_groups', 'hdd_groups.group_id = hdd_users.group_id', 'inner' );

        if( $sLike )
        {
            $this->db->like( 'hdd_users.user_username', $sLike );
        }

        if( $sOrderBy )
        {
            $this->db->order_by( $sOrderBy );
        }
        else
        {
            $this->db->order_by( 'hdd_users.user_changed', 'desc' );
        }

        if( $iLimit )
        {
            $this->db->limit( $iLimit, $iOffset );
        }

        $oResult = $this->db->get();

        foreach( $oResult->result_array() as $arUser )
        {
            $arUsers[] = $arUser;
        }

        return $arUsers;
    }
    /**
     * Count all the users inserted
     * @param   string  $sLike
     * @return  int
     */
    public function getUsersCount( $sLike = NULL )
    {
        if( $sLike )
        {
            $this->db->like( 'hdd_users.user_username', $sLike );
        }

        return $this->db->count_all_results( 'hdd_users' );
    }
    /**
     * Insert a user and return the inserted id
     * @param   array   $arUser
     * @return  boolean
     */
    public function insertUser( $arUser = array() )
    {
```

```
        if( ! sizeof( $arUser ) )
        {
            return FALSE;
        }

        $this->db->insert( 'hdd_users', $arUser );
        return $this->db->insert_id();
    }
    /**
     * Update a user by user_id
     * @param   array   $arUser
     * @param   int     $iUserId
     * @return  boolean
     */
    public function updateUser( $arUser = NULL, $iUserId = NULL  )
    {
        if( ( $arUser === NULL ) ||
            ( ! is_array( $arUser ) ) ||
            ( $iUserId === NULL ) )
        {
            return FALSE;
        }

        $this->db->where( 'hdd_users.user_id', $iUserId );
        return $this->db->update( 'hdd_users', $arUser );
    }
}

/* End of file user_model.php */
/* Location: ./application/models/user_model.php */
```

4.6.2. Vistas

Hemos dividido las vistas en dos grupos: administrador y sitio público, a continuación, se muestra una lista de ambas:

- admin/genres
- admin/manufacturers
- admin/pages
- admin/people
- admin/pictures
- admin/posts
- admin/ratings
- admin/reviews
- admin/settings
- admin/slides
- admin/tags
- admin/templates
- admin/users

- public/front-pages
- public/pages
- public/posts
- public/reviews
- public/tags
- public/templates
- public/users

Un ejemplo del código PHP/HTML para la vista “admin/users” se muestra :

```
<?php require_once APPPATH . 'views/admin/templates/header.php'; ?>
<div id="content-header">
    <h1>Users</h1>
</div><!--#/content-header-->
<div id="breadcrumb">
    <a href="/"><i class="icon-home"></i>Home</a>
    <a href="/admin">Administrator</a>
    <a href="/admin/users" class="current">Users</a>
</div><!--#/breadcrumb-->
<div class="container-fluid">
    <?php if( $arAlert ): ?>
    <div class="row-fluid">
        <div class="span12">
            <div class="alert <?php echo $arAlert['class']; ?>">
                <button class="close" data-dismiss="alert">×</button>
                <?php echo $arAlert['message']; ?>
            </div>
        </div>
    </div>
    <?php endif; ?>


    <?php if( sizeof( $arUsers ) ): ?>
    <?php if( $sLike !== '' ): ?>
    <div id="sub-header" class="row-fluid">
        <h2>Showing results for <em><?php echo $sLike; ?></em></h2>
    </div><!--#/sub-header-->
    <?php endif; ?>
    <div class="row-fluid">
        <div id="search" class="pull-left span4">
            <?php echo form_open( '/admin/users', array( 'method' => 'get', 'class' => 'clearfix',
'autocomplete' => 'off' ) ) . "\n"; ?>
                <?php echo form_input( array( 'name' => 's', 'value' => $sLike, 'placeholder' =>
'Search users', 'class' => 'pull-left' ) ) . "\n"; ?>
                <?php  echo  form_button(  array(  'type'  =>  'submit',  'class'  =>  'pull-left',
'content' => '<i class="hide-text icon-search icon-white">Search</i>' ) ) . "\n"; ?>
            <?php echo form_close() . "\n"; ?>
        </div>
        <div class="clearfix pull-right span4">
            <div class="alternate pagination pull-right">
                <div class="counter"><?php echo $iUsers; ?> User(s)</div>
                <?php echo $sPagination; ?>
            </div>
        </div>
    </div>
    <div class="row-fluid">
        <div class="widget-box">
            <div class="widget-content nopadding">
```

```
                <table class="table table-bordered data-table" id="users-buffer">
                    <thead>
                        <tr>
                            <th id="user-username">Username</th>
                            <th id="user-group">Group</th>
                            <th id="user-name">Name</th>
                            <th id="user-email">Email</th>
                            <th id="user-google-plus-id">Google plus id</th>
                            <th id="user-profile-active">Profile page</th>
                            <th id="user-date">Last updated</th>
                        </tr>
                    </thead>
                    <tbody>
                        <?php foreach( $arUsers as $arUser  ): ?>
                        <tr id="user-<?php echo $arUser['user_id']; ?>">
                            <td>
                                <?php if( ( $iLoggedUserId == $arUser['user_id'] ) ||
                                          ( defined( 'ALLOW_USERS_UPDATE' ) ) ): ?>
                                <a  href="/admin/users/user-edit/<?php  echo  $arUser['user_id'];
?>"><?php echo $arUser['user_username']; ?></a>
                                <?php else: ?>
                                <span><?php echo $arUser['user_username']; ?></span>
                                <?php endif;?>
                                <ul class="clearfix buffer-options">
                                    <?php if( ( $iLoggedUserId == $arUser['user_id']) ||
                                                ( defined( 'ALLOW_USERS_UPDATE' ) ) ): ?>
                                    <li><a         href="/admin/users/user-edit/<?php         echo
$arUser['user_id']; ?>" class="btn btn-success btn-mini">Edit</a></li>
                                    <?php endif;?>
                                    <?php if( defined( 'ALLOW_USERS_UPDATE' ) ): ?>
                                    <li><a        href="/admin/users/user-delete/<?php        echo
$arUser['user_id']; ?>" class="btn btn-danger btn-mini">Delete</a></li>
                                    <?php endif;?>
                                    <?php if( $arUser['user_url'] !== '' ): ?>
                                    <li><a    href="<?php    echo    $arUser['user_url'];    ?>"
target="_blank" class="btn btn-primary btn-mini">View</a></li>
                                    <?php endif; ?>
                                </ul>
                            </td>
                            <td><?php echo $arUser['group_name']; ?></td>
                            <td><?php echo $arUser['user_name']; ?></td>
                            <td>
                                <a href="mailto:<?php echo $arUser['user_email']; ?>"><?php echo
$arUser['user_email']; ?></a>
                            </td>
                            <td>
                               <?php if( $arUser['user_google_plus_id'] !== '' ): ?>
                                <a    href="<?php    echo    $arUser['user_google_plus_url'];    ?>"
target="_blank"><?php echo $arUser['user_google_plus_id']; ?></a>
                                <?php endif; ?>
                            </td>
                            <td><?php echo $arUser['user_profile']; ?></td>
                            <td><?php echo $arUser['user_changed']; ?></td>
                        </tr>
                        <?php endforeach; ?>
                    </tbody>
                </table>
            </div>
        </div>
    </div>
    <div class="row-fluid">
        <div class="clearfix pull-right span4">
            <div class="alternate pagination pull-right">
                <div class="counter"><?php echo $iUsers; ?> User(s)</div>
                <?php echo $sPagination; ?>
            </div>
```

```
        </div>
    </div>
    <?php elseif( $sLike !== '' ): ?>
    <div id="sub-header" class="row-fluid">
        <h2>Your search <em><?php echo $sLike; ?></em> didn't match any user</h2>
    </div><!--#/sub-header-->
    <div class="row-fluid">
        <div id="search" class="pull-left span4">
            <?php echo form_open( '/admin/users', array( 'method' => 'get', 'class' => 'clearfix',
'autocomplete' => 'off' ) ) . "\n"; ?>
                <?php echo form_input( array( 'name' => 's', 'value' => '', 'placeholder' =>
'Search users', 'class' => 'pull-left' ) ) . "\n"; ?>
                <?php echo form_button( array( 'type' => 'submit', 'class' => 'pull-left',
'content' => '<i class="hide-text icon-search icon-white">Search</i>' ) ) . "\n"; ?>
            <?php echo form_close() . "\n"; ?>
        </div>
    </div>
    <?php else: ?>
    <div class="row-fluid">
        <div class="widget-box">
            <div class="widget-content">
                <p>There are no users</p>
            </div>
        </div>
    </div>
    <?php endif; ?>
<?php require_once APPPATH . 'views/admin/templates/footer.php'; ?>
```

4.6.3. Controladores

Hemos dividido los controladores en dos grupos: administrador y sitio público, a continuación, se muestra una lista de ambas:

- admin/blu_ray.php
- admin/dashboard.php
- admin/games.php
- admin/genres.php
- admin/hd_gear.php
- admin/manufacturers.php
- admin/pages.php
- admin/people.php
- admin/pictures.php
- admin/posts.php
- admin/ratings.php
- admin/reviews.php
- admin/settings.php
- admin/slides.php
- admin/tags.php
- admin/ultra_hd.php

- admin/uploader.php
- admin/users.php
- public/front_pages.php
- public/pages.php
- public/pictures.php
- public/posts.php
- public/reviews.php
- public/tags.php
- public/users.php

Un ejemplo del script PHP para el controlador “admin/users.php” se muestra en las siguientes líneas:

```php
<?php

if ( ! defined('BASEPATH')) exit('No direct script access allowed');

class Users extends Admin_Controller
{

    public function __construct()
    {
        parent::__construct();
        $this->load->helper( 'form' );
        $this->load->model( 'Group_model', 'oGroups' );
        define( 'USERS', '' );
    }
    public function index( $iPageNumber = 1 )
    {
        $this->load->helper( 'text' );
        $this->config->load( 'admin/pagination' );
        $this->load->library( 'pagination' );

        define( 'ALL_USERS', '' );
        $arAlert = array();
        $arGetParams = array();
        $arPagination = array();
        $arUsers = array();

        $iLimit = 0;
        $iOffset = 0;
        $iPageNumber = ( int ) $iPageNumber;
        $iUsers = 0;
        $iTotalPages = 0;

        $sLike = '';
        $sPagination = '';

        $sLike = $this->input->get( 's' );
        $sLike = trim( $sLike );
        $sLike = strip_tags( $sLike );

        $iUsers = $this->oUsers->getUsersCount( $sLike );
```

```
        $arPagination = $this->config->item( 'pagination' );
        $arPagination['base_url'] = '/admin/users';
        $arPagination['total_rows'] = $iUsers;

        if( $sLike )
        {
            $arGetParams['s'] = $sLike;
        }

        if( sizeof( $arGetParams ) )
        {
            $arPagination['first_url'] = '/admin/users?' . http_build_query( $arGetParams );
            $arPagination['suffix'] = '?' . http_build_query( $arGetParams );
        }

        $iLimit = $arPagination['per_page'];
        $iOffset = ( $iLimit )  * ( $iPageNumber - 1 );
        $iTotalPages = ceil( $iUsers / $iLimit );

        if( ( $iPageNumber !== 1 ) &&
            ( $iPageNumber > $iTotalPages ) )
        {
            redirect( '/admin/users' );
        }

        $iLoggedUserId = UsersHelper::getUserId();

        $this->pagination->initialize( $arPagination );
        $sPagination = $this->pagination->create_links();

        $arUsers = $this->oUsers->getUsers( $iLimit, $iOffset, $sLike );

        foreach( $arUsers as $k => $v )
        {
            $arUsers[$k]['user_username'] = strip_tags( $arUsers[$k]['user_username'] );
            $arUsers[$k]['user_name'] = strip_tags( $arUsers[$k]['user_name'] );
            $arUsers[$k]['user_changed']       =       DateHelper::getAdministratorDatetimeFormat(
$arUsers[$k]['user_changed'] );
            $arUsers[$k]['user_google_plus_url']     =     GoogleHelper::getGooglePlusProfileURL(
$arUsers[$k]['user_google_plus_id'] );
            $arUsers[$k]['user_url']      =      UsersHelper::getUserURL(      (      bool      )
$arUsers[$k]['user_profile_active'], $arUsers[$k]['user_slug'] );

            if( $arUsers[$k]['user_profile_active'] === '1' )
            {
                $arUsers[$k]['user_profile'] = 'Active';
            }
            else
            {
                $arUsers[$k]['user_profile'] = 'Inactive';
            }

            if( $sLike !== '' )
            {
                $arUsers[$k]['user_username']  =  highlight_phrase(  $arUsers[$k]['user_username'],
$sLike, '<mark>', '</mark>' );
            }
        }
```

```
        $arAlert = $this->session->flashdata( 'alert' );

        $this->load->view( 'admin/users/index', array(
            'arAlert' => $arAlert,
            'arUsers' => $arUsers,
            'iUsers' => $iUsers,
            'sLike' => $sLike,
            'sPagination' => $sPagination,
            'iLoggedUserId' => $iLoggedUserId
        ) );
    }
    public function user_delete( $iUserId = 0 )
    {
        $arUser = array();
        $iUserId = ( int ) $iUserId;

        $arUser = $this->oUsers->getUserById( $iUserId );

        if( ! sizeof( $arUser ) )
        {
            show_404();
        }

        if( sizeof( $_POST ) )
        {
            $this->oUsers->deleteUser( $iUserId );
            $this->session->set_flashdata( 'alert', array(
                'class' => 'alert-success',
                'message' => 'User has been deleted successfully'
            ) );

            redirect( '/admin/users' );
        }

        $this->load->view( 'admin/users/user-delete', array(
            'arUser' => $arUser,
            'iUserId' => $iUserId
        ) );
    }
    public function user_edit( $iUserId = 0 )
    {
        $arUser = array();
        $iUserId = ( int ) $iUserId;

        $arUser = $this->oUsers->getUserById( $iUserId );

        if( ! sizeof( $arUser ) )
        {
            show_404();
        }

        $this->config->load( 'admin/users/user-edit' );
        $this->load->library( 'form_validation' );
        $this->load->library( 'encrypt' );

        $arGroups = array();
        $arValidations = array();
        $sUserPassword = '';

        $arGroups = $this->oGroups->getGroups();
        $arGroups = UtilsHelper::getOptionsDropdown( $arGroups, 'group_id', 'group_name', array(
```

```
'' => 'Select group' ) );

        $arValidations = $this->config->item( 'validations' );
        $sUserPassword = $this->input->post( 'user_password' );

        $this->form_validation->set_error_delimiters( '<span class="help-block">', '</span>' );



        if( ( sizeof( $_POST ) ) &&
            ( $sUserPassword === '' ) )
        {
            unset( $arValidations[3], $arValidations[4] );
        }

        $this->form_validation->set_rules( $arValidations );

        if( $this->form_validation->run() )
        {
            $arUser = array();
            $arUser['user_biography'] = $this->input->post( 'user_biography' );
            $arUser['user_changed'] = time();
            $arUser['user_email'] = $this->input->post( 'user_email' );
            $arUser['user_google_plus_id'] = $this->input->post( 'user_google_plus_id' );
            $arUser['user_name'] = $this->input->post( 'user_name' );

            if( $sUserPassword !== '' )
            {
                $arUser['user_password'] = $this->encrypt->sha1( $sUserPassword );
            }

            $arUser['user_profile_active'] = ( int ) $this->input->post( 'user_profile_active' );
            $arUser['user_slug'] = UtilsHelper::getPermalink( $arUser['user_name'] );
            $arUser['group_id'] = ( int ) $this->input->post( 'group_id' );

            $this->oUsers->updateUser( $arUser, $iUserId );

            $this->session->set_flashdata( 'alert', array(
                'class' => 'alert-success',
                'message' => 'User has been edited successfully'
            ) );

            redirect( '/admin/users' );
        }

        $this->cache->memcached->delete(        $this->arMemcacheKeys['user_page_reviews']        .
$arUser['user_slug'] );
        $this->cache->memcached->delete( $this->arMemcacheKeys['user'] . $arUser['user_slug'] );

        $this->load->view( 'admin/users/user-edit', array(
            'arGroups' => $arGroups,
            'arUser' => $arUser,
            'iUserId' => $iUserId
        ) );

    }
    public function user_login()
    {
        if( UsersHelper::isLoggedIn() )
        {
```

```
            redirect( '/admin/reviews' );
        }
        $this->arCSS = array();
        $this->arCSS[] = '/css/vendor/bootstrap/bootstrap.min';
        $this->arCSS[] = '/css/vendor/bootstrap/bootstrap-responsive.min';
        $this->arCSS[] = '/css/vendor/bootstrap/unicorn.login';
        $this->arCSS[] = '/css/admin/global';

        $this->config->load( 'admin/users/user-login' );
        $this->load->library( 'form_validation' );
        $this->load->library( 'encrypt' );

        $arUser = array();
        $arValidations = array();
        $sError = '';
        $sPassword = '';
        $sUsername = '';

        $arValidations = $this->config->item( 'validations' );
        $this->form_validation->set_error_delimiters( '<span class="help-block">', '</span>' );
        $this->form_validation->set_rules( $arValidations );



        if( $this->form_validation->run() )
        {
            $sUsername = $this->input->post( 'user_username' );
            $sPassword = $this->encrypt->sha1( $this->input->post( 'user_password' ) );

            $arUser = $this->oUsers->getUserByUsername( $sUsername );

            if( ! sizeof( $arUser ) )
            {
                $sError = 'The username you entered does not belong to any account.';
            }
            elseif( $arUser['user_password'] !== $sPassword )
            {
                $sError = 'The password you entered is incorrect.';
            }
            else
            {
                $this->session->set_userdata( array(
                    'group_id' => $arUser['group_id'],
                    'user_id' => $arUser['user_id']
                ) );
                redirect( '/admin/reviews' );
            }
        }

        $this->load->view( 'admin/users/user-login', array(
            'sError' => $sError
        ) );
    }
    public function user_logout()
    {
        $this->session->sess_destroy();
        redirect( '/admin/login' );
    }
    public function user_new()
    {
        $this->config->load( 'admin/users/user-new' );
        $this->load->library( 'form_validation' );
        $this->load->library( 'encrypt' );
```

```
        define( 'USER_NEW', '' );
        $arGroups = array();
        $arUser = array();
        $arValidations = array();
        $iEpoch = 0;

        $arGroups = $this->oGroups->getGroups();
        $arGroups  = UtilsHelper::getOptionsDropdown( $arGroups, 'group_id', 'group_name', array(
'' => 'Select group' ) );

        $arValidations = $this->config->item( 'validations' );
        $this->form_validation->set_error_delimiters( '<span class="help-block">', '</span>' );
        $this->form_validation->set_rules( $arValidations );

        if( $this->form_validation->run() )
        {
            $iEpoch = time();

            $arUser['user_biography'] = $this->input->post( 'user_biography' );
            $arUser['user_changed'] = $iEpoch;
            $arUser['user_created'] = $iEpoch;
            $arUser['user_email'] = $this->input->post( 'user_email' );
            $arUser['user_google_plus_id'] = $this->input->post( 'user_google_plus_id' );
            $arUser['user_name'] = $this->input->post( 'user_name' );
            $arUser['user_password'] = $this->encrypt->sha1( $this->input->post( 'user_password' )
);
            $arUser['user_profile_active'] = ( int ) $this->input->post( 'user_profile_active' );
            $arUser['user_slug'] = UtilsHelper::getPermalink( $arUser['user_name'] );
            $arUser['user_username'] = $this->input->post( 'user_username' );
            $arUser['group_id'] = ( int ) $this->input->post( 'group_id' );

            $this->oUsers->insertUser( $arUser );

            $this->session->set_flashdata( 'alert', array(
                'class' => 'alert-success',
                'message' => 'User has been created successfully'
            ) );

            redirect( '/admin/users' );
        }

        $this->load->view( 'admin/users/user-new', array(
            'arGroups' => $arGroups
        ) );
    }
}

/* End of file users.php */
/* Location: ./application/controllers/admin/users.php */
```

4.7. Frontend

En la figura 6, podemos observar el diseño gráfico que presenta el sitio Hig-Def Digest actualmente:

Figura 6. Diseño gráfico actual del sitio High-Def DVD Digest

En la figura 7, se puede observar el rediseño del sitio después de considerar diversas opciones y empleando las tendencias.:

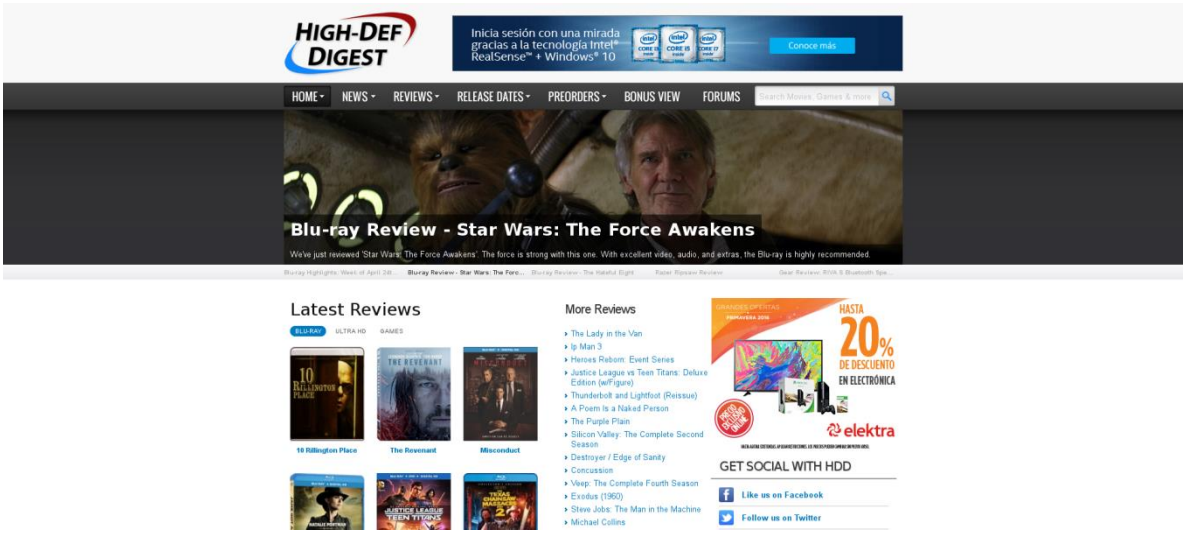

Figura 7. Nuevo diseño gráfico del sitio High-Def Digest

Diseño para el panel de administrador:

Figura 8. Diseño gráfico para el panel de administración

5. Análisis de la experiencia adquirida

5.1 Análisis general del programa, su diseño, desarrollo y organización

Cuando empecé a desarrollar este proyecto puse en práctica todos los conocimientos adquiridos durante mis estudios, y aprendí que las tecnologías es algo muy cambiante y de un día para el otro pasan a hacer obsoletas, por lo que debemos de mantenernos al día con las nuevas tendencias en ellas, he compilado una lista de fortalezas y debilidades que note durante mi estancia profesional:

Fortalezas

- Cimientos necesarios para aprender cualquier lenguaje de programación
- Buena comunicación con el equipo
- Buenas bases para la tecnología de servidores
- Buen manejo de base de datos
- Buen manejo de lenguajes de programación para el servidor
- Responsabilidad
- Ética

Debilidades

- Manejo pobre del idioma Inglés
- Manejo pobre de HTML
- Manejo pobre de CSS

- Manejo pobre de JavaScript
- Manejo pobre de Linux

5.2 Análisis de los objetivos del programa: grado de consecución

Hemos tenido resultados bastante satisfactorios con la migración del sitio web, básicamente hemos logrado reducir el tiempo de carga del sitio, mejorar el posicionamiento en los buscadores web, el tráfico ha incrementado y el sitio ha estado generando más ganancias.

5.3 Análisis de las actividades realizadas

A una semana de que el sitio fue lanzado a producción, hemos estado recopilando información para medir el nivel de aceptación entre los administradores, editores y los usuarios del sitio que es el parámetro más importante. Hemos clasificado estos resultados en positivos y negativos, a continuación, se muestra un desglose de ambos:

Positivos

- La carga del sitio pasó de 9 segundos a: 1.5 a 2 segundos aproximadamente
- Los servidores web han estado trabajado constantemente sin problemas
- El tráfico del sitio ha estado aumentando gradualmente
- Las ganancias del sitio se han incrementado en un 25%
- Se ha estado publicando 50% más de información
- Es más fácil encontrar la información
- Los buscadores web han estado posicionando mejor al sitio
- Se ocupa menos mano de obra por parte de los desarrolladores y el equipo de calidad
- La interacción con el usuario ha incrementado aproximadamente 40% en la sección de comentarios y sugerencias

Negativos

- Alrededor de un 10% de los usuarios están molestos porque no les gusto el rediseño del sitio
- Alrededor de un 5% de los usuarios están molestos porque se muestra menos información en la página de inicio

5.4 Análisis de la metodología utilizada

5.4.1 MVC

Al usar el modelo MVC *"Modelo Vista Controlador",* para el desarrollo de este sitio web hemos notado que el desarrollo se agiliza bastante, en general porque todo el código está dividido en capas y bloques muy pequeños los cuales se puedan reutilizar por toda la aplicación.

5.4.2 Memcached

Con el uso de Memcached se reduce significamente el uso de la base de datos, creando páginas dinámicas en mucho menos tiempo.

5.4.3 Minimización de imágenes, CSS y JS

La carga en el lado del cliente ha sido optimizada bastante, al haber minimizado las imágenes, CSS y JS hemos dejado de hacer muchas peticiones en el servidor reduciendo significativamente la carga del sitio en el lado del cliente.

5.4.4 MySQL

Con la nueva estructura de la base de datos hemos creado una forma bastante sencilla para trabajar con la base de datos, cualquier operación que se necesite hacer, es realizada bastante simple, adicionalmente a esto, el servidor web de la base de datos no ha tenido complicaciones de carga como pasaba anteriormente.

6. Conclusiones y recomendaciones

Como se puede observar, tenemos una gran mayoría de resultados positivos sobre los negativos, los cuales representan solo problemas visuales (de contenido y organización por lo que concluimos que el trabajo realizado cumplió con las expectativas.

Los administradores del sitio han estado ideando nuevos proyectos para lanzarse en los siguientes meses, cosa que nunca hubiera sido posible con el rediseño del sitio, entre ellos se tienen:

- Creación de una versión amigable para dispositivos móviles
- Creación de una nueva sección en el sitio para formatos de disco y cintas antiguas como “VHS”
- Aumentar el equipo editorial

6.1 Recomendaciones

Como cualquier sitio web las optimizaciones son interminables, hay unas de mayor importancia que otras, pero en general, creo que las siguientes deben de ser resueltas lo más pronto posible:

6.1.1 Versión amigable para dispositivos móviles

Hemos estado analizando los dispositivos que los usuarios usan para conectarse a nuestro sitio y hemos notado que más del 40% del tráfico viene de tabletas y celulares, creemos que el tráfico empezará a bajar nuevamente si este problema no es resuelto lo más pronto posible.

6.1.2 Problemas para hacer lanzamientos a producción

Actualmente, el proceso para hacer lanzamientos a producción, consiste en lanzar primero el código a los servidores de pruebas para después lanzarlo a los servidores de producción desde los servidores de pruebas, una vez finalizado este proceso, tenemos que correr una serie de comandos para limpiar memorias directamente en los servidores de producción, lo cual puede causar bastantes errores, se recomienda añadir un script shell el cual se encargue de hacer esto de forma automática cuando un lanzamiento a producción sea finalizado.

6.1.3 Problemas con el control de versiones del código

Actualmente, el control de versiones del código está basado en “SVN” el cual en general es bastante lento, en especial para hacer *commits* y copias del código, cabe recalcar que SVN es una tecnología bastante antigua y no hay mucha documentación, por lo cual se recomienda mover todo el código a GIT.

Es de vital importancia completarlos en la siguiente versión del sitio, en especial el soporte para dispositivos móviles, porque esto repercutirá directamente con el posicionamiento en los buscadores web.

7. Referencias bibliográficas y virtuales

https://es.wikipedia.org/wiki/Servidor_HTTP_Apache
https://es.wikipedia.org/wiki/Base_de_datos
https://es.wikipedia.org/wiki/Red_de_entrega_de_contenidos
https://es.wikipedia.org/wiki/Hoja_de_estilos_en_cascada
https://es.wikipedia.org/wiki/Gzip
https://es.wikipedia.org/wiki/HTML
https://es.wikipedia.org/wiki/JavaScript
https://es.wikipedia.org/wiki/Memcached
https://es.wikipedia.org/wiki/Modelo%E2%80%93vista%E2%80%93controlador
https://es.wikipedia.org/wiki/MySQL
https://es.wikipedia.org/wiki/PHP
https://es.wikipedia.org/wiki/Servidor_web
https://es.wikipedia.org/wiki/Ruby_on_Rails
https://es.wikipedia.org/wiki/Ruby
https://es.wikipedia.org/wiki/EllisLab
https://es.wikipedia.org/wiki/Balance_de_carga
https://es.wikipedia.org/wiki/Subversion_(software)
https://es.wikipedia.org/wiki/Git