ANÁLISIS, DISEÑO Y PROTOTIPO DE UN GESTOR DE CONTENIDOS PARA PÁGINAS WEB

CARLOS ALBERTO LONDOÑO LOAIZA
JOHN ALEXÁNDER CALDERÓN HERNÁNDEZ

UNIVERSIDAD TECNOLÓGICA DE PEREIRA
FACULTAD DE INGENIERÍAS: ELÉCTRICA, ELECTRÓNICA, FÍSICA Y CIENCIAS DE LA COMPUTACIÓN
INGENIERÍA DE SISTEMAS Y COMPUTACIÓN
2015

ANÁLISIS, DISEÑO Y PROTOTIPO DE UN GESTOR DE CONTENIDOS PARA PÁGINAS WEB

CARLOS ALBERTO LONDOÑO LOAIZA
JOHN ALEXÁNDER CALDERÓN HERNÁNDEZ

DIRECTOR DE PROYECTO:
JUAN DE JESÚS VELOZA MORA

UNIVERSIDAD TECNOLÓGICA DE PEREIRA
FACULTAD DE INGENIERÍAS: ELÉCTRICA, ELECTRÓNICA, FÍSICA Y CIENCIAS DE LA COMPUTACIÓN
INGENIERÍA DE SISTEMAS Y COMPUTACIÓN
2015

TABLA DE CONTENIDO

1. TITULO....9
2. FORMULACIÓN DEL PROBLEMA....10
3. JUSTIFICACIÓN....12
4. OBJETIVOS....13
   4.1. OBJETIVO GENERAL....13
   4.2. OBJETIVOS ESPECÍFICOS....13
5. MARCO REFERENCIAL....14
   5.1. MARCO HISTÓRICO....14
   5.2. MARCO LEGAL....15
   6. ANÁLISIS....16
   6.1. DISEÑO RESPONSIVO Y ELÁSTICO....17
   6.2. FACILIDAD DE USO....27
   6.3. SEGURIDAD:....37
7. DISEÑO....46
   7.1. REQUISITOS FUNCIONALES....47
   7.2. REQUERIMIENTOS NO FUNCIONALES....49
   7.3. CASO DE USO....52
   7.4. HISTORIAS DE USUARIO....52
   7.5. DISEÑO DE CASO DE PRUEBA....59
   7.6. DESCRIPCIÓN CASO DE USO....59
   7.7. DIAGRAMA DE SECUENCIA....66
   7.8. VISTAS DEL HARDWARE....71

7.9. VISTA DEL SOFTWARE 71
7.10. MODELO DE BASE DE DATOS 72
7.11. EJECUCIÓN DE CASOS DE PRUEBAS 73
8. PROTOTIPO 75
8.1. PÁGINA PRINCIPAL 75
8.2. INICIO DE SESIÓN 75
8.3. VISTA DENTRO DEL SISTEMA 75
9. DISEÑO METODOLOGICO PRELIMINAR 77
9.1. HIPÓTESIS 77
9.2. TIPO DE INVESTIGACIÓN 77
9.3. POBLACIÓN 77
10. VARIABLES 77
10.1. DISEÑO DE INSTRUMENTOS PARA TOMA DE INFORMACIÓN 77
11. PARTICIPANTES DEL PROYECTO 78
12. RECURSOS DISPONIBLES 79
13. CRONOGRAMA 80
14. CONCLUSIONES 81
15. RECOMENDACIONES 82
16. GLOSARIO 83
17. BIBLIOGRAFÍA 87

LISTA DE TABLAS

Tabla 1: Requisitos Funcionales - Base de datos 47
Tabla 2: Requisitos Funcionales - Interfaz de Registro de Datos 47
Tabla 3: Requisitos Funcionales - Sistema Adaptable 48
Tabla 4: Requisitos Funcionales - Plantilla de Interfaz 48
Tabla 5: Requisitos Funcionales - Compatibilidad con los Navegadores 48
Tabla 6: Requisitos No Funcionales - Ataques de Seguridad 49
Tabla 7: Requisitos No Funcionales - Operatividad 49
Tabla 8: Requisitos No Funcionales - Acceso a las Cuentas 50
Tabla 9: Requisitos No Funcionales - Visualización con los Navegadores 50
Tabla 10: Requisitos No Funcionales - Tiempo en Visualización 50
Tabla 11: Requisitos No Funcionales - Usabilidad 51
Tabla 12: Requisitos No Funcionales - Limitación de Plantillas 51
Tabla 13: Historia de Usuario – Registro 53
Tabla 14: Historia de Usuario - Compatibilidad con Dispositivos 53
Tabla 15: Historia de Usuario - Compatibilidad con Navegadores 54
Tabla 16: Historia de Usuario - Desempeño 54
Tabla 17: Historia de Usuario - Seguridad 55
Tabla 18: Historia de Usuario - Simplicidad de Diseño 55
Tabla 19: Historia de Usuario - Plantilla 56
Tabla 20: Historia de Usuario - Modificar Plantilla 56
Tabla 21: Historia de Usuario - Conexión a Internet 57
Tabla 22: Historia de Usuario - Sin Conexión a Internet 57
Tabla 23: Historia de Usuario - Base de Datos 57
Tabla 24: Historia de Usuario - Exportar Datos 58
Tabla 25: Descripción Caso de Uso - Registro 60
Tabla 26: Descripción Caso de Uso - Compatibilidad con Dispositivos 63
Tabla 27: Descripción Caso de Uso - Plantilla 63

Tabla 28: Descripción Caso de Uso - Modificar Plantilla 65
Tabla 29: Ejecución Caso de Pruebas.....74
Tabla 30: Cronograma de Actividades.....80

LISTA DE FIGURAS

Figura 1: Código HTML de ejemplo 18
Figura 2: Código de ejemplo CSS.....19
Figura 3: Código de ejemplo Javascript.....20
Figura 4: Código CSS con media queries.....21
Figura 5: Código Bootstrap para el diseño responsivo 22
Figura 6: Foto de los diferentes navegadores (InternetLab, 2012) 23
Figura 7: Fragmento código normalize (Gallagher, Normalize.css, 2011) 24
Figura 8: Fragmento de código modernizr, (Faruk, Paul, Alex, Ryan, Patrick, Stu, and Richard., 2009) 25
Figura 9: Ejemplo de paquetes de iconos incluidos, (Gandy, 2011) 26
Figura 10: Fragmento de código Jquery, (The Jquery Foundation, 2006) 26
Figura 11: Tiempo de carga del prototipo de gestor de contenidos para páginas web 28
Figura 12: Botón compartir para las redes sociales, (AS Amor Sevillista, 2013) ...30
Figura 13: Botón exportar a Excel, (Palacios, Styde.Net, 2015) 30
Figura 14: Botón Actualización Automática, (Palacios, Styde.Net, 2015) 31
Figura 15: Sistema de carga de archivos con dropzone, (Palacios, Styde.Net, 2015).....32
Figura 16: Autenticación en redes sociales, (Palacios, Styde.Net, 2015) 32
Figura 17: Interfaz gráfica de CKeditor, (Palacios, Styde.Net, 2015).....34
Figura 18: Fragmento de código Blade.....34
Figura 19: Archivo de Rutas Laravel.....38
Figura 20: Archivo de Validaciones en Laravel.....41
Figura 21: Código Captcha, (Palacios, Styde.Net, 2015).....42
Figura 22: Caso de uso general.....52
Figura 23: Diagrama de secuencias registró 66
Figura 24: Diagrama de secuencias compatibilidad con el dispositivo 66

Figura 25: Diagrama de secuencias compatibilidad con el navegador 67
Figura 26: Diagrama de secuencias plantilla 67
Figura 27: Diagrama de secuencias exportar datos.....68
Figura 28: Diagrama de actividades registró 68
Figura 29: Diagrama de actividades compatibilidad con el dispositivo 69
Figura 30: Diagrama de actividades modificar plantilla.....69
Figura 31:Diagrama de actividades seguridad.....70
Figura 32: Vista del hardware 71
Figura 33: Vista del software 71
Figura 34: Modelo Entidad – Relación de la base de datos.....72
Figura 35: Página de Inicio Prototipo de Sistema Gestor de Contenidos 75
Figura 36: Página de Inicio de Sesión 75
Figura 37: Vista dentro del Prototipo.....76

1. TITULO

Análisis, diseño y prototipo de un gestor de contenidos para páginas web

2. FORMULACIÓN DEL PROBLEMA

Después de realizar múltiples búsquedas en internet, participar en varios cursos virtuales y programar varias páginas web, se detectó la necesidad de construir un prototipo de un gestor de contenidos para páginas web, que satisfaga las necesidades de facilidad de uso, seguridad y economía, logradas a través de un diseño responsivo y elástico.

Entre los usuarios de las herramientas tecnológicas, sobre todo aquellos que no tienen conocimientos básicos en tecnología e informática, existen ciertos prejuicios, preconceptos o tabúes que los llevan a formular expresiones como “la tecnología me atropella” o “estoy muy viejo para esas cosas”. La tecnología avanza de una manera tan vertiginosa y atropellada que desconcierta a los usuarios y les hace ver como complicadas cosas que realmente son muy elementales. El usuario de las herramientas tecnológicas de un gestor de contenidos para páginas web debe sentirse tranquilo y confiado de que lo que va a usar es muy fácil y práctico, y que no necesita conocimientos de ingeniería para implementar y diseñar todo lo relacionado con páginas web. Muchos usuarios creen que por no tener recursos económicos ni conocimientos avanzados no pueden promocionar sus negocios por la web. Se espera darle solución a este inconveniente con lo propuesto en este proyecto.

De otra parte, hay también algunos elementos relacionados con la presunta inseguridad de la información que se sube a una página web. En efecto, muchas personas se abstienen de hacerlo por el temor de que la información le sea sustraída, plagiada o destinada a un propósito distinto. Grandes ideas de buenos negocios se echan a perder por el temor de que sea explotado por otras personas mediante la sustracción y plagio de la información que sube a través de las páginas web. La persona que haga uso de este proyecto, tendrá un alto porcentaje de confiabilidad de que su información no será alterada.

Otro prejuicio que existe con el diseño e implementación de las páginas web es que son muy costosas debido a que siempre se debe acudir a personal especializado en el desarrollo de software. Esta limitante trae también como consecuencia que muchas personas no participen en la web porque la rentabilidad del negocio no les daría para pagar una página. Por lo tanto, se pretende superar este inconveniente ofreciendo la posibilidad de que sea gratis siempre y cuando se dé el crédito al autor.

Finalmente, existen muchas herramientas para elaborar páginas web pero que se encuentran dispersas y se requiere de mucho tiempo para concentrarlas en un solo sitio. Muchas personas desisten de acometer la tarea de diseñar una página porque considera que acceder a la información es adentrarse en un laberinto del cual difícilmente encontraran salida. Se busca que quien quiera hacer uso de estos recursos los pueda encontrar en un solo sitio.

3. JUSTIFICACIÓN

Un gestor de contenido es un software que permite la administración total de una página web, de manera que el usuario que interactúe con él no necesite conocimientos en desarrollo de software; solo requieren unos conocimientos básicos de la estructura funcional de una página web y de sus componentes.

Se entiende como presencia en la web el hecho de que una persona disponga de un espacio virtual como una página; sin embargo, son muchas las personas que no pueden acceder a dicho servicio porque no tienen los conocimientos necesarios para construirla, lo que las margina de poder disfrutar de estos beneficios tecnológicos.

De otra parte, existe cierta desconfianza de los usuarios hacia los gestores de contenido debido a que no ofrecen un alto porcentaje de seguridad. Si bien es cierto, ningún aplicativo es ciento por ciento seguro, se busca ofrecer una herramienta con un nivel avanzado de seguridad que le garantice confianza al usuario. Igualmente, sería un gran beneficio para las personas si no tienen que pagar para desarrollar estos instrumentos, recursos económicos que podrían utilizar para su capacitación o para el crecimiento de su negocio.

Por todo lo expuesto, se busca darles una solución integral a las dificultades que reportan los usuarios en el diseño y uso de las páginas web. Si bien es cierto, no es una solución última y definitiva, sí contribuye a que se vaya cambiando la mentalidad de las personas en cuanto al tema y se masifique la facilidad y la gratuidad de los recursos web.

Cualquier usuario, indistintamente de sus conocimientos y de sus recursos económicos, encontrara en este gestor una solución para la promoción y posicionamiento de sus productos en el mercado.

4. OBJETIVOS

4.1. OBJETIVO GENERAL

Construir un prototipo de gestor de contenidos para páginas web, usando tecnologías modernas para la construcción de sitios web y que soporte algunas de las condiciones de diseño responsivo y elástico.

4.2. OBJETIVOS ESPECÍFICOS

- Acopiar información documentada y empírica acerca de los gestores de contenido que hay en el mercado digital.
- Analizar sus condiciones de seguridad y de vulnerabilidad a la luz de la guía OWASP.
- Diseñar un prototipo de un gestor de contenidos para páginas web que corrija las falencias encontradas.

5. MARCO REFERENCIAL

5.1. MARCO HISTÓRICO

Según Xavier García Cuerda, “a principios de los años noventa, el concepto de sistemas de gestión de contenidos era desconocido. Algunas de sus funciones se realizaban con aplicaciones independientes: editores de texto y de imágenes, bases de datos y programación a medida.

Ya el año 1994 *Illustra Information Technology* utilizaba una base de datos de objetos como repositorio de los contenidos de una web, con el objetivo de poder reutilizar los objetos y ofrecía a los autores un entorno para la creación basado en patrones. La idea no cuajó entre el público y la parte de la empresa enfocada a la Web fue comprada por AOL, mientras que *Informix* adquirió la parte de bases de datos.

RedDot es una de las empresas pioneras que empezó el desarrollo de un gestor de contenidos el año 1994. No fue hasta a finales del año siguiente que presentaron su CMS basado en una base de datos.

Entre los CMS de código abierto uno de los primeros fue *Typo 3*, que empezó su desarrollo el año 1997, en palabras de su autor, *Kasper Skårhøj*, “antes de que el término gestión de contenidos fuera conocido sobradamente”.

PHPNuke, la herramienta que popularizó el uso de estos sistemas para las comunidades de usuarios en Internet, se empezó a desarrollar el año 2000. La primera versión supuso tres semanas de trabajo al creador, rescribiendo el código de otra herramienta, *Thatware*.” (Xavier García Cuerda, 2004).

A partir del 2004, nace Wordpress, un gestor de contenidos con grandes características, que rápidamente se toma una cuenta de mercado alta, que en el

2015 llega al 23%. Estos gestores de contenido, se fortalecen primero al desarrollo que tuvieron los anteriores gestores, y segundo a la facilidad con la que se podría implementar en servidor.

Otro gestor de contenidos muy mencionado, y a su vez criticado es Joomla, y se dice criticado por su nivel de dificultad para poderse usar. Sin embargo, todos los gestores de contenido creados a partir del 2012 en adelante toman como su base o referente algunos de los dos grandes de la web, es decir Wordpress o Joomla.

5.2. MARCO LEGAL

Ley 1273 de 2009: Por medio de la cual se modifica el Código Penal, se crea un nuevo bien jurídico tutelado - denominado "de la protección de la información y de los datos"- y se preservan integralmente los sistemas que utilicen las tecnologías de la información y las comunicaciones, entre otras disposiciones. (Congreso de la República de Colombia, 2009)

Esta ley evidencia de manera muy clara cada uno de los puntos tenidos en cuenta en el sector colombiano para los delitos informáticos, lo que permite a la hora del desarrollo del prototipo de gestor de contenidos para páginas web tomarla como base en el componente de seguridad, diseño y facilidad de uso.

Artículo 6 de la Ley 23 de 1982: Los inventos o descubrimientos científicos con aplicación práctica explotable en la industria, y los escritos que los describen, solo son materia de privilegio temporal, con arreglo al artículo 120, numeral 18, de la Constitución.

Las ideas o contenido conceptual de las obras literarias, artísticas y científicas no son objeto de apropiación. Esta Ley protege exclusivamente la forma literaria, plástica o sonora, como las ideas del autor son descritas, explicadas, ilustradas o incorporadas en las obras literarias, científicas y artísticas.

Las obras de arte aplicadas a la industria solo son protegidas en la medida en que su valor artístico pueda ser separado del carácter industrial del objeto u objetos en las que ellas puedan ser aplicadas. (Congreso de la República de Colombia, 1982), (DIRECCIÓN NACIONAL DE DERECHOS DE AUTOR, s.f.)

El prototipo de gestor de contenido para páginas web, debe de contar con algún tipo de licencia, que les permita a los autores conservar sus derechos, esto se evidencia en la ley, sin embargo, el proyecto será registrado bajo la licencia GNU (Stallman, 1999), que le permite a cualquier persona poderlo usar.

6. ANÁLISIS

Tal como está planteado en el objetivo general, este proyecto se orienta hacia la construcción de un prototipo de un gestor de contenidos para páginas web a partir de tres puntos básicos: diseño responsivo y elástico, facilidad de uso y seguridad, los cuales se analizan en detalle.

En el responsive web design el diseño y el contenido se adaptan a cada pantalla, entregando una experiencia de usuario muy similar en resoluciones bajas, altas o en formatos de distintas pulgadas.

Para lograr esto, los contenidos se ordenan en bloques que se reorganizan según las características de la pantalla y el navegador que se utiliza. Las partes y la jerarquía de los elementos se definen según una serie de parámetros, entre ellos:

- Ancho y alto de la ventana del navegador.
- Orientación del dispositivo.
- Proporción entre el alto y ancho de la pantalla.
- Resolución del dispositivo, es decir, la precisión del detalle en las imágenes de mapa de bits.

De esta forma el usuario ve una diagramación distinta en cada dispositivo, pero aun así puede acceder a todos los contenidos. La información se mantiene igual, pero su presentación se optimiza según el aparato y el navegador usado. (Ideas Digitales Aplicadas, 2015).

6.1. DISEÑO RESPONSIVO Y ELÁSTICO

Cuando se habla de diseño, nuestra mirada se centra en la parte gráfica o visible del proyecto y el componente de este proyecto con el cual el usuario va a ver y a interactuar, lo que hace que sea un elemento muy importante. Sin embargo, para los ingenieros de sistemas la parte gráfica es compleja, ya que nunca se preocupan por capacitarse en diseño gráfico. Pero en este proyecto el usuario final solo desea interactuar con el software y no tener que pensar en nada más que usarlo.

Se realizó una búsqueda en internet sobre el proceso de diseño gráfico para páginas web, y se encontraron muchas herramientas que permiten crear dicho diseño de manera muy gráfica, pero todas ellas son incompletas ya que no permiten que el usuario exporte su proyecto a un gestor de contenidos como tal, es decir, si el usuario crea una interfaz gráfica muy agradable tendría que conocer de lenguajes de programación web para poder administrar estos esquemas gráficos y para que varíen en el tiempo. Lo que se pretende con este proyecto es que el usuario utilice un diseño básico y agradable y que a la vez se centre en la funcionalidad y administración del contenido de su sitio web.

Cuando se elabora un diseño para páginas web, se usan herramientas como generadores de código, los cuales, desde nuestra experiencia como programadores, no son aconsejables porque generan código innecesario o código no estandarizado para el navegador, lo cual influye negativamente en el posicionamiento en las primeras listas de los buscadores. Este factor es muy

importante cuando se quiere que nuestra página o proyecto sea visto en las primeras páginas de buscadores como Google.

Todo esto quiere decir que es importante a la hora de realizar un diseño para la web tener un conocimiento en lenguajes como HTML (lenguaje de marcas de hipertexto), CSS (hoja de estilos en cascada), Javascript, cada uno de los cuales cada uno realiza un aporte importante y significativo.

El HTML se encarga de la estructura del sitio web, es decir, así como cuando se crea un cuento, donde se dice que el cuento tiene inicio de la historia, nudo de la historia y desenlace de la historia, las páginas web también se dividen en varias partes, como lo son el encabezado, donde se ubica el menú, el cuerpo donde se ubica todo el contenido de la página web, y el pie de página, que es la parte final de la página web. Cabe la pena rescatar que cada elemento en HTML es una etiqueta y HTML posee una etiqueta para cada elemento que se quiere representar, por ejemplo, un párrafo, una lista, un enlace (Developer Mozilla, 2005). A continuación, se muestra un ejemplo del código HTML estructura:

proyecto.html

```
1  <!DOCTYPE html>
2  <html lang="es">
3  <head>
4      <meta charset="UTF-8">
5      <title>Documento</title>
6  </head>
7  <body>
8      <h1>Hola mundo desde el gestor de contenidos</h1>
9  </body>
10 </html>
11
```

Figura 1: Código HTML de ejemplo

Esto es básicamente el HTML, pero si se observa en el texto, solo se habla de estructura, no de colores ni formas, ni alineación, etc. Para ello fue creado el CSS, el cual permite darle alguna forma, color o posición a cada una de las etiquetas creadas en el HTML.

El CSS o la hoja de estilos en cascada (Developer Mozilla, 2015), es el encargado de volver bonito el cuerpo del documento, y en su última actualización llamada CSS3 nos permite hacer uso de un término muy usado actualmente llamado *media queries* y del cual se hablará más adelante.

A continuación, se muestra un ejemplo de código CSS:

proyecto.html estilo.css

```
1  body{
2      background: red;
3      color: white;
4      width: 100%
5  }
6  h1{
7      color: black;
8      font-family: Arial;
9      font-size: 16;
10 }
```

Figura 2: Código de ejemplo CSS

Hasta ahora se ha dicho que el prototipo tiene una forma, un cuerpo y que adicionalmente es bonito porque se logró combinar cada una de las etiquetas del HTML con los estilos del CSS, pero qué pasa con el sitio que es estático, no tiene movimiento, entonces es donde aparece el mágico lenguaje de programación llamado Javascript, el cual se va a encargar de que el aplicativo sea dinámico, que tenga elementos como un slider de imágenes, que las imágenes se vuelvan grandes cuando se pasa el mouse por encima de ellas, que la página se actualice de manera

automática sin necesidad de recargarse. Todo esto lo hace posible Javascript, el cual va a tomar un lugar importante en el prototipo de gestor de contenidos.

Ejemplos de código Javascript:

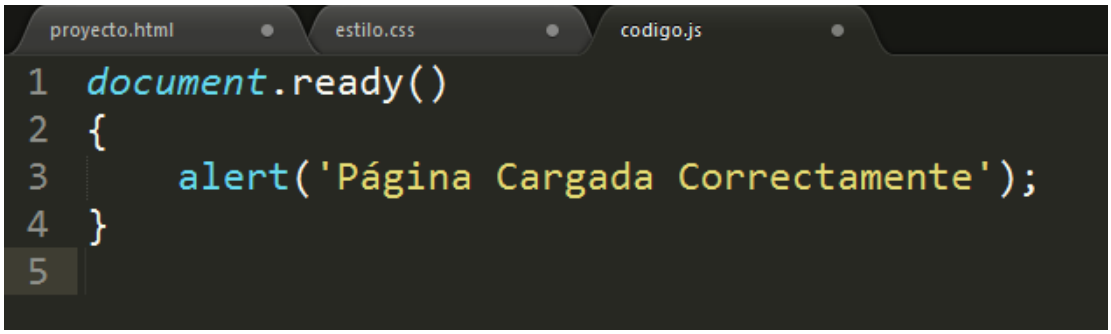


Figura 3: Código de ejemplo Javascript

Como aprender estos tres lenguajes al mismo tiempo se hace difícil para alguien que no sea profesional en ingeniería de sistemas, existe un gestor de herramientas digitales conocido como *framework* (Actualidad Geek, 2014), que facilita el proceso de desarrollo de software para gestor de contenidos. El *framework* más popular se llama Bootstratp

En la actualidad, los celulares, tabletas y computadoras no poseen un estándar para las pantallas; por lo tanto, se hace necesario crear sitios web que se adapten a dichos dispositivos y que su diseño continúe siendo tan agradable como lo son originalmente, y más que eso es que el gran motor de la web llamado Google, en su nuevo sistema de posicionamiento de las páginas en los primeros resultados, colocó como requisito que cualquier página web que quiere ser pionera en el sistema de búsquedas, deberá ser adaptable al cualquier dispositivo electrónico.

Es importante tener en cuenta el principio de reutilización de un código, el cual en las condiciones anotadas es muy difícil de cumplir debido a que se debe crear la misma página para cada dispositivo electrónico, lo que equivale a ejecutar tres

veces la misma opción. Sin embargo, la solución se encuentra en el nuevo CSS3, cuya última actualización incluye los llamados *media queries*, o elementos que permiten enlazar una misma página a distintos dispositivos, tal como se muestra a continuación.

Figura 4: Código CSS con media queries

El prototipo propuesto usa dos elementos muy potentes en la fase de diseño como son diseño responsivo y diseño elástico. A continuación, se detalla que es cada uno de ellos.

El diseño responsivo quiere decir que cuando la pantalla cambie de tamaño por los dispositivos que la usan, algunos elementos desaparezcan de la página, para que el contenido restante se visualice de mejor calidad. Y es allí donde Bootstrap hace un gran aporte ya que permite a través de las *medias queries* (Developer Mozilla, 2015) que se habían visto en el paso anterior, cuando captura el tamaño de la pantalla cambia el comportamiento de ciertos elementos, tales como ocultar objetos, cambiarle el tamaño, el color, la forma, etc., según se haya decidido en el diseño.

El diseño responsivo proviene de una filosofía llamada móviles primeros (*Mobile first*), donde se dice que todo diseño que se realice primero se debe desarrollar para móviles, luego para tabletas y por ultimo para pantallas de escritorio.

De otra parte, los navegadores permiten a través de una de sus opciones llamadas diseños elásticos, que algunas etiquetas HTML reajusten su tamaño automáticamente según el tamaño del dispositivo. Es por ello que no se habla de solo diseño responsivo, porque no es bien visto ocultar todos los elementos, en cambio sí es bien visto que se conserven algunas partes u opciones del diseño original para el diseño más pequeño. Es aquí donde las etiquetas elásticas ayudan a complementar un diseño diciendo que se compone de diseños elásticos y diseño responsivo.

Pero de nuevo se acaba de ver dos conceptos nuevos, pero no se sabe cómo aplicarse, entonces es aquí donde se vuelve a retomar a Bootstrap, el cual dentro de sus herramientas ya elaboradas, dispone de una grilla o de un sistema de grillas, o en algunos casos llamado rejillas, para ayudar a diseñar el prototipo y hacerlo que se adapte a cualquier dispositivo, y es que a través de esta grilla el protititpo se adapta a cada dispositivo dividiendo la pantalla en doce partes iguales, e indicando en cada pantalla, cuantas columnas se van a ocupar. Y para la parte elástica, Bootstrap automáticamente toma del navegador un listado de etiquetas que se pueden permitir para los usuarios.

proyecto.html | estilo.css | index.blade.php | codigo.js

```
@extends('layouts.principal')
@section('contenido')
    <div class="col-sm-9 col-sm-offset-3 col-md-10 col-md-offset-2 main">
        <h3 class="page-header">Dashboard</h3>
        <div class="col-xs-12 col-sm-12">
            <div class="thumbnail">
                <div class="caption">
                    <img class="img-responsive center-block" src="{{asset('images/fondo.jpg'
                    <p class="alert alert-info text-center"><b>Dirección</b> <br> <b>Teléfon
                </div>
            </div>
        </div>
    </div>
@endsection
```

Figura 5: Código Bootstrap para el diseño responsivo

En esta instancia del proyecto, se es necesario estandarizar el código, es decir que sea lenguaje natural para todos los navegadores, ya que el prototipo es multiusuario, es decir que muchos usuarios lo pueden usar al mismo tiempo, multiplataforma, es decir que es compatible con Windows, MAC, Linux, etc., y múltiples navegadores, es decir compatible con Mozilla, Google Chrome, Opera, Safari, Internet Explorer.

Figura 6: Foto de los diferentes navegadores (InternetLab, 2012)

Aparentemente esta última característica para ser fácil a simple vista, pero hay una dificultad y es que cada navegador utiliza las etiquetas HTML y CSS de forma distinta, lo cual genera que en el diseño en cada navegador se vea de forma distinta, pero sin embargo a cada uno de los problemas planteados en este proyecto se le dado una solución, y este no va a ser la excepción, para hacer que todos los navegadores se comporten igual se utiliza un archivo llamado Normalize (Gallagher, Normalize.css, 2011), el cual es un archivo CSS que resetea todos los estilos a cero, haciendo que todos los navegadores, interpreten los mismo valores para cada etiqueta.

```css
/*! normalize.css v3.0.2 | MIT License | git.io/normalize */

/**
 * 1. Set default font family to sans-serif.
 * 2. Prevent iOS text size adjust after orientation change, without disabling
 *    user zoom.
 */

html {
  font-family: sans-serif; /* 1 */
  -ms-text-size-adjust: 100%; /* 2 */
  -webkit-text-size-adjust: 100%; /* 2 */
}

/**
 * Remove default margin.
 */

body {
  margin: 0;
}
```

Figura 7: Fragmento código normalize (Gallagher, Normalize.css, 2011)

Ya se ha creado un diseño ajustable a cualquier dispositivo a través del diseño elástico y el diseño responsivo, ya se puede visualizar en cualquier navegador, pero ahora falta una característica importante, y es que en lenguaje HTML va en su versión HTML5, donde incluye muchas mejoras, pero de nuevo hay un problema de compatibilidad de algunas etiquetas con los navegadores, pero para ello se hara uso de una nueva herramienta llamada *Modernizr*, que lo que hace es adaptar las nuevas etiquetas HTML5 para los navegadores viejos que no las soportan, entonces al código se le añade un nuevo archivo llamado *Modernizr* (Faruk, Paul, Alex, Ryan, Patrick, Stu, and Richard., 2009).

```
1  Modernizr.on('flash', function( result ) {
2    if (result) {
3      // the browser has flash
4    } else {
5      // the browser does not have flash
6    }
7  });
```

Figura 8: Fragmento de código modernizr, (Faruk, Paul, Alex, Ryan, Patrick, Stu, and Richard., 2009)

Para culminar la etapa de diseño, se hará uso de una herramienta adicional llamada fuente de iconos vectoriales es decir que no se pixelan, y sirven para hacer mucho más grafica la aplicación y no caer en el error de llenar de texto todo el contenido. Cuando se añade la fuente de iconos al proyecto, se dispone de un paquete de 605 iconos, con los cuales se puede representar ciertas sesiones del prototipo de gestor de contenidos, y esto se hace solo añadiendo el código de cada icono donde se quiere que aparezca en la vista.

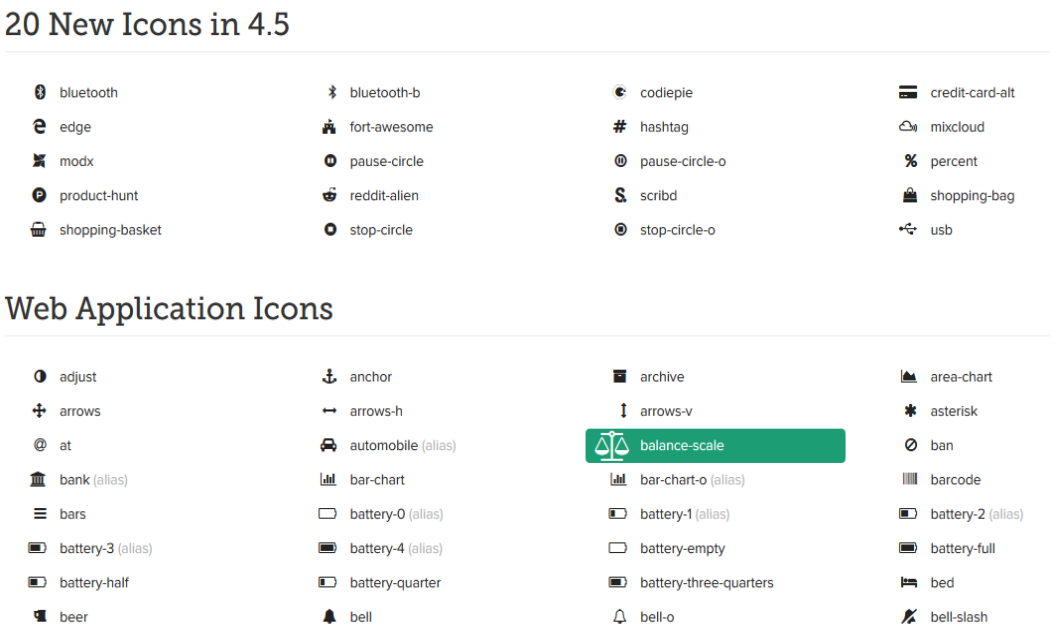


Figura 9: Ejemplo de paquetes de iconos incluidos, (Gandy, 2011)

Bueno todo el tiempo se habla de HTML y CSS, y no se ha empleado el termino Javascript, pero la realidad en este momento es que escribir código puro en Javascript para iniciar en el mundo de la web es un complejo, y es el mensaje que trae Bootstrap, ya que él tampoco lo hace y es que se puede hacer uso de librerías como Jquery, la cual es un conjunto de código Javascript, que facilita la interacción con el DOM, pero que es el DOM (document object model), esto quiere decir que Jquery puede interactuar con cada elemento de la estructura del proyecto que está escrita en HTML, entonces lo que hace Jquery es tomar un elemento y realizar una acción al pasar el mouse por encima de un título cambiar el color, etc. Si observan se enfoca en animar cada espacio del proyecto, evitando que quede estático.

Dentro de las muchas funcionalidades de Jquery, hay algunas resaltables como capturar los campos de los formularios, hacer validaciones en el navegador, para evitar que se envíen datos erróneos, cambiar la hoja de estilos, capturar el mouse, capturar el teclado, invocar funciones, ejecutar estructuras repetitivas y condicionales. Vale la pena aclarar todas estas funcionalidades son las que el usuario va a interactuar.

```
1  /*! jQuery v1.11.3 | (c) 2005, 2015 jQuery Foundation, Inc. | jquery.org/license */
2  !function(a,b){"object"==typeof module&&"object"==typeof module.exports?module.exports=a.
       document?b(a,!0):function(a){if(!a.document)throw new Error("jQuery requires a window
       with a document");return b(a)}:b(a)}("undefined"!=typeof window?window:this,function(a,b)
       {var c=[],d=c.slice,e=c.concat,f=c.push,g=c.indexOf,h={},i=h.toString,j=h.hasOwnProperty,
       k={},l="1.11.3",m=function(a,b){return new m.fn.init(a,b)},n=/^[\s\uFEFF\xA0]+|[\s\uFEFF
       \xA0]+$/g,o=/^-ms-/,p=/-([\da-z])/gi,q=function(a,b){return b.toUpperCase()};m.fn=m.
       prototype={jquery:l,constructor:m,selector:"",length:0,toArray:function(){return d.call(
       this)},get:function(a){return null!=a?0>a?this[a+this.length]:this[a]:d.call(this)},
       pushStack:function(a){var b=m.merge(this.constructor(),a);return b.prevObject=this,b.
       context=this.context,b},each:function(a,b){return m.each(this,a,b)},map:function(a){
       return this.pushStack(m.map(this,function(b,c){return a.call(b,c,b)}))},slice:function(){
       return this.pushStack(d.apply(this,arguments))},first:function(){return this.eq(0)},last:
       function(){return this.eq(-1)},eq:function(a){var b=this.length,c=+a+(0>a?b:0);return
       this.pushStack(c>=0&&b>c?[this[c]]:[])},end:function(){return this.prevObject||this.con
```

Figura 10: Fragmento de código Jquery, (The Jquery Foundation, 2006)

Pero hay que tener en cuenta, que el proyecto va a ser posible descargarlo al computador y ser modificado por aquellos usuarios que dominan más es código, por eso es necesario que todo sea muy claro y no haya librerías complejas.

De esta manera se creó un breve resumen de todas las herramientas usadas para crear la parte gráfica.

6.2. FACILIDAD DE USO

Facilidad de uso o la usabilidad es la medida de la calidad de la experiencia que tiene un usuario cuando interactúa con un producto o sistema. Esto se mide a través del estudio de la relación que se produce entre las herramientas (entendidas en un Sitio Web el conjunto integrado por el sistema de navegación, las funcionalidades y los contenidos ofrecidos) y quienes las utilizan, para determinar la eficiencia en el uso de los diferentes elementos ofrecidos en las pantallas y la efectividad en el cumplimiento de las tareas que se pueden llevar a cabo a través de ellas, (Gobierno de Chile, 2013)

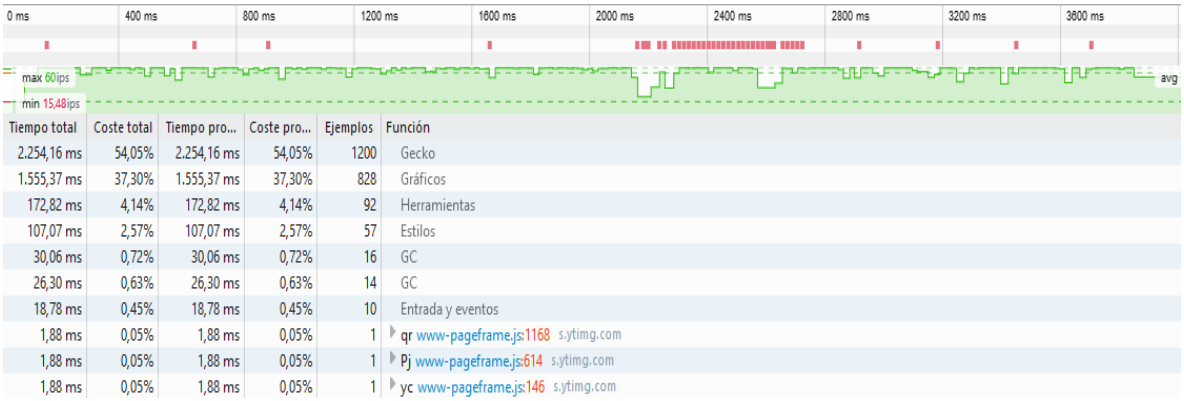
Los programas complejos de usar, o que necesiten manuales de muchas páginas están diseñados para otros tipos de usuarios avanzados. El usuario que usa el prototipo de gestor de contenido es una persona con una necesidad inicial y es crear un espacio en la web y que pueda administrarlo de manera autónoma sin necesidad de terceros, y es lo que busca promover, que todo sea fácil de usar, y que no se necesite otras personas para realizar una tarea.

Todo sitio para ser usable se puede medir de acuerdo a las siguientes características:

- **Facilidad de aprendizaje**: define en cuánto tiempo un usuario, que nunca ha visto una interfaz, puede aprender a usarla bien y realizar operaciones básicas.
- **Facilidad y Eficiencia de uso**: determina la rapidez con que se pueden desarrollar las tareas, una vez que se ha aprendido a usar el sistema.

- **Facilidad de recordar cómo funciona**: se refiere a la capacidad de recordar las características y forma de uso de un sistema para volver a utilizarlo a futuro.
- **Frecuencia y gravedad de errores**: plantea la ayuda que se le entrega a los usuarios para apoyarlos cuando deban enfrentar los errores que cometen al usar el sistema.
- **Satisfacción subjetiva**: indica lo satisfechos que quedan los usuarios cuando han empleado el sistema, gracias a la facilidad y simplicidad de uso de sus pantallas, (Gobierno de Chile, 2013).

Si se realiza un análisis por cada paso del proyecto, se obtiene que es realmente rápido, ya que toma 1224 ms en estar cargada en condiciones ideals, y a su vez el hecho de utilizar librerías optimizadas como Bootstrap y Jquery, hace que la carga se realice de manera más eficiente. También se resalta que uno de los requisitos de google para darle un ranking de posicionamiento a una página o sitio web depende de su rapidez de carga, y esta depende de todos estos factores y también de la minificación de archivos CSS y archivos Javascript, recordando que los archivos CSS deben estar compilados en uno solo. A continuación, se observa el cuadro de carga tomado por el navegador Mozilla.



Tiempo total	Coste total	Tiempo pro...	Coste pro...	Ejemplos	Función
2.254,16 ms	54,05%	2.254,16 ms	54,05%	1200	Gecko
1.555,37 ms	37,30%	1.555,37 ms	37,30%	828	Gráficos
172,82 ms	4,14%	172,82 ms	4,14%	92	Herramientas
107,07 ms	2,57%	107,07 ms	2,57%	57	Estilos
30,06 ms	0,72%	30,06 ms	0,72%	16	GC
26,30 ms	0,63%	26,30 ms	0,63%	14	GC
18,78 ms	0,45%	18,78 ms	0,45%	10	Entrada y eventos
1,88 ms	0,05%	1,88 ms	0,05%	1	qr www-pageframe.js:1168 s.ytimg.com
1,88 ms	0,05%	1,88 ms	0,05%	1	Pj www-pageframe.js:614 s.ytimg.com
1,88 ms	0,05%	1,88 ms	0,05%	1	yc www-pageframe.js:146 s.ytimg.com

Figura 11: Tiempo de carga del prototipo de gestor de contenidos para páginas web

La navegación por el software se hace a través de una sola barra de menú, lo que hace que sea simple, porque allí se encuentran todas las opciones con las que el usuario puede interactuar, no se plantean rutas alternas, para evitar que el usuario se pierda en su uso.

Debido a la naturaleza del prototipo, el texto que indexa Google, se encarga de escribirlo directamente el usuario final. Sin embargo, el contenido con el cual se desarrolla el gestor de contenidos, lleva texto muy preciso sobre su funcionalidad.

La naturaleza aplicada en el ítem anterior es decir diseño, hace que el prototipo de gestor de contenidos sea adaptado a cualquier dispositivo móvil, Tablet o de escritorio, lo que crea un punto a favor en la parte de usabilidad, recordando que también es compatible con múltiples navegadores, de acuerdo al código que se usa.

El componente de actualización es claro en el prototipo de gestor de contenidos, ya que debido al sistema de control de versiones GitHub que se usa permite todo el tiempo actualizar el proyecto sin realizar tareas adicionales, y aquí se aclara que GIT es el lenguaje de control de versiones más popular que existe, y que en el prototipo se implementa para poderse actualizar de manera inmediata.

Ya se observa como proyecto cumple los 5 pasos básicos de la usabilidad, a continuación, se hará una breve descripción de una serie de herramientas que primero facilitan la interacción del software con el usuario.

a) BOTÓN DE COMPARTIR EN FACEBOOK Y TWITTER:

Este botón le permite al usuario que interactúa con el aplicativo con un solo clic compartir el contenido que visualiza con un solo clic, cabe resaltar, que esto hace que el sistema se integre automáticamente a las redes sociales.

Esto se logra a través de la API de Facebook y de Twitter, donde cada usuario se registra en las cuentas de las redes sociales para poderlas usar.

Figura 12: Botón compartir para las redes sociales, (AS Amor Sevillista, 2013)

b) EXPORTAR EXCEL:

El usuario puede a través de un solo clic exportar a Excel el contenido de las tablas de entradas o páginas creadas, lo cual le permite tener control e inventario detallado de las actividades que realiza en el aplicativo.

Debido a que el aplicativo como se evidencia en otros sectores del texto está construido en PHP con el *Framework* Laravel, se hace posible usar elementos ya creados de la tienda de Composer, uno de ellos es este complemento que permite exportar a Excel el contenido.

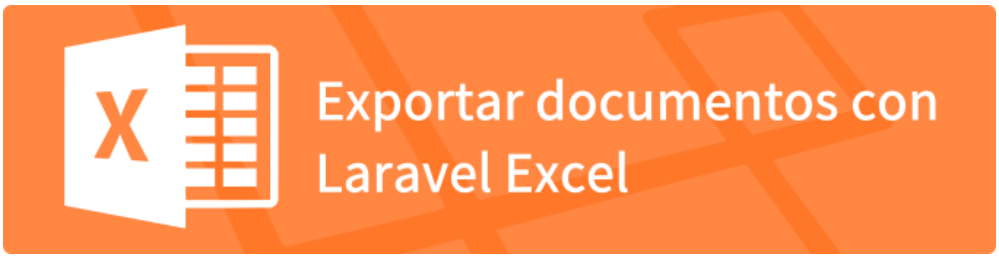


Figura 13: Botón exportar a Excel, (Palacios, Styde.Net, 2015)

c) ACTUALIZACIÓN DE SECTORES AUTOMÁTICOS:

Otra de las preocupaciones sin duda de los usuarios es que no se da cuenta de los datos cambiados y quisiera sin recargar la página que los datos se visualizarán automáticamente. Para esta preocupación del usuario se añade una porción de código en JQUERY a través de una petición AJAX que se encarga de enviar peticiones al servidor, cada que algo cambia en la base de datos de manera automática, esta respuesta llega en formato JSON y es devuelta al usuario final.

Figura 14: Botón Actualización Automática, (Palacios, Styde.Net, 2015)

d) CARGAR ARCHIVOS CON DROPZONE:

Dropzone es un complemento escrito en JAVASCRIPT que permite cargar archivos al software tales como fotos, de una manera automática, que solo sea arrastrar la imagen al área de la carga y soltar. Función que le permite al software ser más fácil de usar.

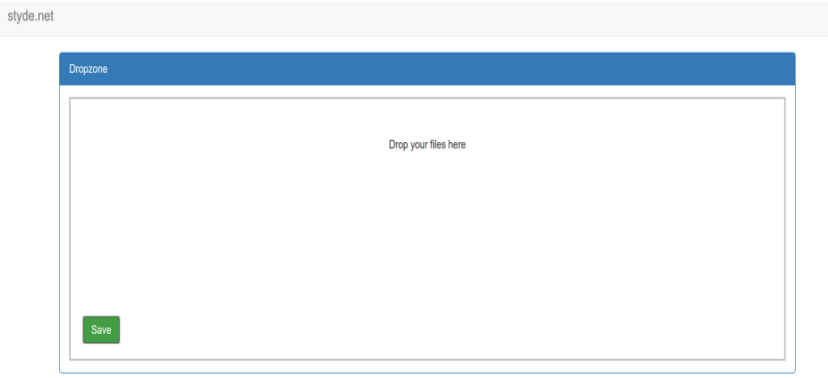


Figura 15: Sistema de carga de archivos con dropzone, (Palacios, Styde.Net, 2015)

e) AUTENTICACIÓN CON REDES SOCIALES:

Dado a que un gran porcentaje de los usuarios que usan el prototipo de gestor de contenidos, tiene una cuenta en la red social Facebook, seria para toda una facilidad que pueda iniciar sesión con su cuenta de Facebook. Esto se logra creando una aplicación en Facebook e integrándola al código.

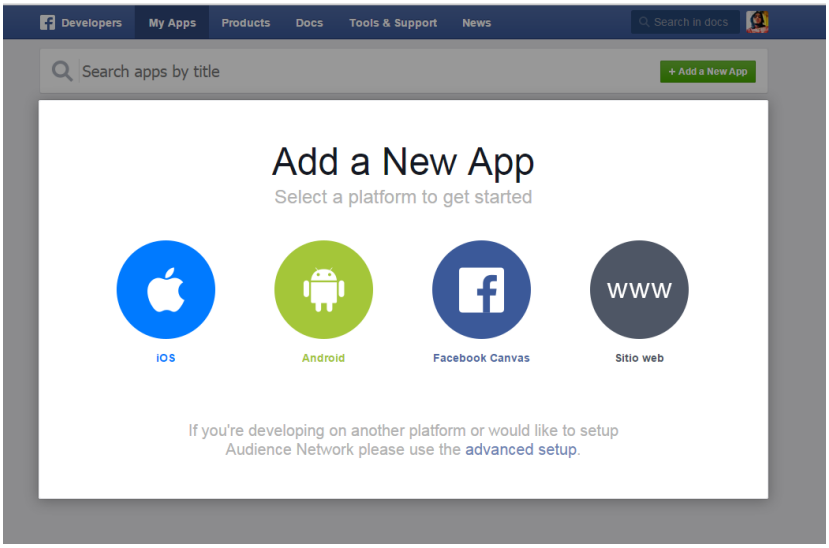


Figura 16: Autenticación en redes sociales, (Palacios, Styde.Net, 2015)

f) FILTROS Y BÚSQUEDAS:

Utilizando el complemento *Datatables* (SpryMedia, 2007), creado con Jquery, permite visualizar el contenido de la base de datos en una tabla, lo cual ayuda a conservar un orden y a realizar búsquedas por cualquier campo de la base de datos. Cabe resaltar que es el complemento programado en Jquery el que se encarga de buscar el contenido.

g) INTEGRAR GOOGLE MAPS:

Google Maps, es quizás una de las herramientas más importantes a la hora de hablar de posicionamiento y ubicación, es por ello que es importante integrarlo en esta versión del prototipo de gestor de contenidos de modo que el usuario, pueda integrarlo en el momento que lo crea necesario.

Para integrarlo en el proyecto se usa la API de google, la cual es compatible con Laravel.

h) INTEGRAR YOUTUBE

El canal de Youtube almacena millones de videos, y a su vez YouTube permite a través de composer obtener una API, para enlazarla al prototipo de gestor de contenidos, de modo, que el usuario pueda publicar videos en la página.

i) CKEDITOR

Quizás el complemento más importante para un usuario final a la hora de crear una entrada o noticia, ya que es la forma más grafica para hacerlo. Este se conoce como editor de textos sencillo que cuenta con las herramientas necesarias para darle formato a un texto.

Para integrarlo al proyecto es necesario descargarlo de la página del autor.

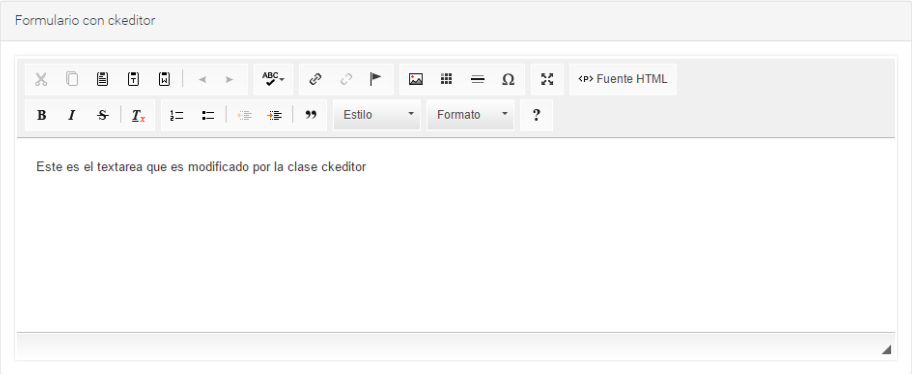


Figura 17: Interfaz gráfica de CKeditor, (Palacios, Styde.Net, 2015)

j) MOTOR DE PLANTILLAS BLADE

Blade como se define al comienzo es el motor de plantillas de laravel, esto que quiere decir, es el HTML dinámico; en el HTML no se puede colocar otros datos que no sean las etiquetas nativas, pero a través de este sistema de plantillas, se puede agregar diferentes sentencias propias de la programación como lo son el foreach, if, else, el único requisito es anteponerle un símbolo @.

```
1  @for ($i = 0; $i < 10; $i++)
2      The current value is {{ $i }}
3  @endfor
4
5  @foreach ($users as $user)
6      <p>This is user {{ $user->id }}</p>
7  @endforeach
8
9  @forelse ($users as $user)
10     <li>{{ $user->name }}</li>
11 @empty
12     <p>No users</p>
13 @endforelse
14
15 @while (true)
16     <p>I'm looping forever.</p>
17 @endwhile
```

Figura 18: Fragmento de código Blade

k) SOPORTE

Nada más confiable para el usuario que disponer de un espacio para preguntar sus inquietudes y reportar los errores que ha tenido, a través de un botón llamado soporte, los usuarios reportan los daños obtenidos, y en un tiempo corto, recibe solución a su problema. Lo cual va generando una confianza entre el usuario y el software.

l) RECUPERAR CONTRASEÑAS

Comúnmente el usuario olvida su contraseña de acceso a un sistema y más cuando lleva varios días sin usarlo, la propuesta es muy clara respecto a la recuperación de contraseñas y es permitirle al usuario a través de un campo de olvide contraseña.

Pero como funciona esto, prácticamente la magia de esta funcionalidad la realiza Laravel, y eso radica en que Laravel trae por defecto un módulo de recuperación de contraseñas, lo único que se hace activarlo y configurar el servicio de correo.

m) RECORDAR CAMPOS CUANDO HAY FORMULARIOS CON ERRORES

Es común que el usuario cometa errores al intentar agregar un registro a la base de datos, pero sería complejo cuando el formulario no recuerda lo que el usuario a escrito, y más si era algo extenso. Para ello Laravel trae un campo llamado old:value, que retorna el valor agregado anteriormente, así de esta forma si hay errores, el usuario no pierde lo que está escribiendo, solo los corrige y continua.

n) IMPRIMIR ERRORES MÁS CLAROS:

Como estrategia de confirmación, es importante que cada que el usuario realice una transacción donde se involucre la base de datos, se le devuelva un mensaje como notificación indicándole si fue exitosa o no, y en caso contrario el mensaje con los

errores que obtuvo. Esto se obtiene a través de Bootstrap, que ya trae los mensajes creados, solo es cuestión de hacerles un llamado donde se requieran usar.

o) MIGAS DE PAN:

Las migas de pan hacen parte de Bootstrap, las cuales equivalen a la ruta en la pantalla donde se encuentra el usuario, ayuda a que el usuario no se pierda, y tenga facilidad de navegación por la página, ya sea para ir adelante o para retroceder.

p) URL AMIGABLES:

Las direcciones que se presentan en la parte superior, donde el usuario navega, es decir la url, le informa al usuario donde se encuentra ubicado, es importante que esta dirección sea clara, y no tenga símbolos raros que le genere al usuario confianza. Esto lo hace Laravel a través de su archivo de rutas, donde permite tener las rutas en el formato que se requiera, y de la manera más clara para el usuario. Para este archivo no es necesario la instalación de ningún complemento.

Si se observa todas las anteriores herramientas incluidas en el proyecto, lo que hacen es mejorar la experiencia de usuario, haciendo que el prototipo de gestor de contenidos tenga una facilidad de uso alta.

Se destaca que muchas de esas herramientas son obtenidas en la tienda de PHP llamada Packagist, a través de su instalador de dependencias, llamado Composer.

6.3. SEGURIDAD:

Quizás el tema más complejo de todo este trabajo se llama seguridad, pero vale la pena trabajarlo, ya que es lo que todas las empresas de tecnologías web le apuestan a diario, pero muy pocas logran conseguir.

Hablar de seguridad web, según OWASP se define como “Ausencia del peligro” (Comunidad OWASP, 2005), en este caso que se habla de un aplicativo web también se habla de ausencia del peligro, sin embargo, se enfocara en la seguridad de solo el aplicativo, ya que el campo seguridad es muy amplio.

A continuación, se harán unas definiciones de los problemas más comunes de seguridad y la forma como el prototipo de gestor de contenidos los va a combatir. A su vez se nombran las formas preventivas de reducir el riesgo.

a. SQL INJECTION:

Consiste en la detención de una vulnerabilidad por parte del atacante y a su vez la inyección de código a través de esa vulnerabilidad.

Se enfoca en encontrar un formulario o un espacio para ingresar código en este caso malicioso, e inclusive a través de la URL.

Para mitigar dichos ataques se tienen varias estrategias, que se define en los puntos a continuación:

- Validación de la información ingresada en los formularios
- Registrar solo las URL necesarias para el proyecto
- Validar que la persona que va a insertar un registro en la base de datos tenga los permisos necesarios.

b. URL SEGURAS:

Las URL seguras, se define como aquellas rutas que no permiten ambigüedades en su información. Es decir, se debe de definir qué tipo de información procesa cada URL según la necesidad y según el método.

Laravel tiene una facultad y es que el archivo de rutas no es posible acceder a él desde la vista, la única forma seria desde el servidor, pero ese ya nos enfocaríamos en otros tipos de protección. Dicho archivo de rutas solo permite cargar la información registrada allí, esto quiere decir que, si yo intento ingresar a una URL no registrada, me arrojara el error 404, y a su vez si la ruta se encuentra creada, se verificada que el usuario tenga permisos para crear registros.

```
//Ruta Pagina Principal
Route::get('index', 'InicioController@index');

//Rutas Inicio de Sesion
Route::get('auth/login', 'Auth\AuthController@getLogin');
Route::post('auth/login', 'Auth\AuthController@postLogin');
Route::get('auth/logout', 'Auth\AuthController@getLogout');

//Rutas de Registro de Usuario
Route::get('auth/register', 'Auth\AuthController@getRegister');
Route::post('auth/register', 'Auth\AuthController@postRegister');

Route::post('crearUsuario', 'UsuarioController@crearUsuario');

Route::group(['middleware' => 'auth'], function()
{
    //Rutas de Pantalla Inicio
    Route::get('/', 'UsuarioController@inicio');
    Route::get('inicio', 'UsuarioController@inicio');
```

Figura 19: Archivo de Rutas Laravel

c. Middleware:

Los middlewares son un término utilizado en laravel para proteger la aplicación y estos radican en unas reglas que se deben de cumplir, de lo contrario arroja un

error. Dichas reglas pueden ser validaciones a la información ingresada, como validaciones a rutas, por ejemplo, el middleware Auth en laravel, validad que usuario para poder usar el software se encuentre con sesión iniciada.

d. ROLES:

Los roles se definen como los permisos de usuario que tiene una persona para usar el software, es importante que un software cuente con diferentes roles, para así asegurar que las personas con poco o mucho conocimiento realicen actividades indeseadas en el software. Como medida preventiva se crea un middleware llamado role, que permita filtrar el role del usuario y solo mostrarle el conjunto de opciones propias de su usuario.

e. ATAQUES XSS:

Los ataques xss, es una forma de introducir código HTML, CSS o Javascript en un espacio la página web vulnerable, es muy similar al inyection, a diferencia que el inyection realiza operaciones en la base de datos, y el xss busca alterar los datos que se muestran en la vista.

Para prevenir el xss hay varias opciones:

- Crear una lista de palabras que no puede recibir un campo de formulario
- Validar los datos ingresados
- Validar la URL de la persona que envía datos

f. ATAQUES CSRF:

Obliga al usuario a realizar una petición al servidor en el cual esta logueado, es decir, en el cual tiene confianza.

Este es un problema un poco complejo de abordar ya que como se menciona en la definición aquí el atacante entro en la cuenta de cualquier usuario y procede a hacer alteraciones en la base de datos.

La forma de mitigar este ataque, es continuar con las mismas validaciones así el usuario este autenticado, y son:

- Validar campos
- Validar URL

g. VALIDACIÓN DE FORMULARIOS:

Consiste en recoger los valores ingresados en un formulario y validar que correspondan a unas reglas indicadas.

Laravel dispone de un excelente modulo para estas validaciones y permite definir el tipo de campo, la extensión, la longitud, si existe, etc. Si es válido continua el proceso, de lo contrario lo re direcciona a la página donde estaba con información de errores.

Es la validación más segura que se puede hacer en laravel, ya que permite validación con base de datos.

También se puede filtrar símbolos que no se quiere que el usuario introduzca por seguridad, tales como “”<>#, etc.

```php
public function crearUsuario(Request $request)
{
    $this->validate($request, [
        'email' => 'required|email',
        'password' => 'required|confirmed|min:6',
        'nombre' => 'required|min:3',
        'apellido' => 'required|min:3',
        'telefono' => 'required|min:5',
        'direccion' => 'required|min:5',
        'tipo_usuario' => 'required'
    ]);
    $usuario = new User();
    $usuario->fill($request->all());
    $usuario->password = bcrypt($request->get('password'));
    $usuario->save();

    return redirect()->back()->with('alert', 'El usuario se creó
}
```

Figura 20: Archivo de Validaciones en Laravel

h. MODELO VISTA CONTROLADOR:

El modelo vista controlador es un patrón de diseño creado para laravel, que permite que se repartan las cargas de la información y que todo sea controlado a través de un archivo de rutas. Este modelo aporta a la seguridad, ya que si se observa no es posible acceder desde las vistas a ningún archivo, solo lo que imprime las rutas.

i. LISTA NEGRA DE PALABRAS:

Consiste en un listado de palabras que el usuario no puede ingresar cuando diligencie un campo de un formulario, tales como:

- DROP
- DELETE
- SELECT
- UPDATE

Debido a que estas palabras si se envían en una sentencia lógica, podría tomarse como una consulta y hacer daño a la base de datos, es importante validar antes de enviar a la base de datos que los formularios no contengan estas palabras.

j. LISTA NEGRA DE DIRECCIONES IP:

Consiste en un listado de direcciones IP, de las cuales se conocen a intentando vulnerar la aplicación, para que no tenga acceso al aplicativo.

k. CÓDIGO CAPTCHA:

Consiste en una cadena de caracteres que debe ser escrita por el usuario y no es vista por los robots, se usa para evitar el registro o inicio de sesión de usuarios que no sean humanos.

Es importante la implementación de este módulo, ya que el hecho de no tenerlo, podría causar la caída del servidor a través de una denegación de servicio.

La implementación en Laravel es muy fácil solo es descargar el paquete e integrarlo al proyecto.

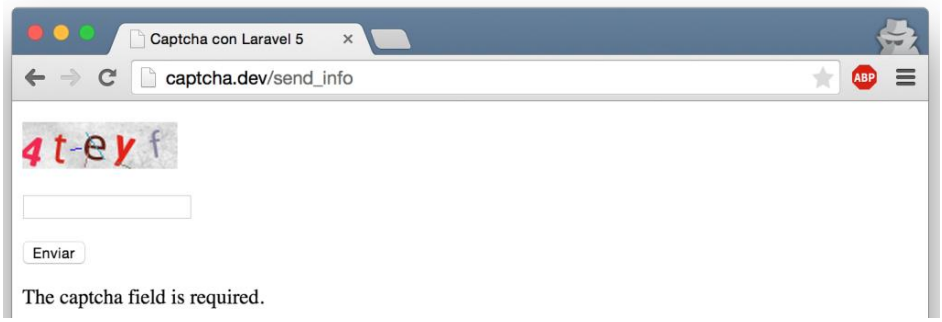


Figura 21: Código Captcha, (Palacios, Styde.Net, 2015)

l. LÍMITE DE INTENTOS DE INICIO DE SESIÓN:

Cuando se evidencia el intento continuo erróneo de inicio de sesión, es importante que el software almacene esta información, y lleve un contador, en este caso lo que haría sería bloquearle por un tiempo limitado para que no pueda iniciar sesión dándole a entender que intento demasiadas veces con un código erróneo y que lo intente de nuevo unos minutos después.

m. CSRF_TOKEN:

Consiste en una cadena codificada que se genera en las cabeceras de los formularios, con el fin de validar que sea un formulario generado por laravel y no por un usuario atacante. Es muy importante esta funcionalidad en el framework, ya que sin ella va a generar error.

n. ENCRIPTACIÓN DE CONTRASEÑAS:

La encriptación consiste en un proceso de cifrado de una clave para que no pueda se desencriptada, le brinda una comunicación segura al usuario cada vez que inicia sesión. El algoritmo de encriptación usado es el md5.

o. BORRADO LÓGICO:

También conocido como soft delete consiste en un proceso donde se busca no borrar el registro de la base de datos, por el contrario, desactivar el campo para que la base de datos no lo siga leyendo y mostrando.

Es vital este campo porque permite la recuperación de la información posteriormente.

p. REGISTRO CON VALIDACIÓN DE EMAIL:

Cuando un usuario crea una cuenta en algún software es importante validar el correo electrónico para saber que se habla con una persona interesado en el prototipo de gestor de contenidos.

Por eso es de vital importancia validar cada que un usuario se registre, para evitar información falsa, y también cuando se vaya a recuperar la contraseña.

q. PRUEBAS DE INTEGRACIÓN:

Las pruebas unitarias permiten detectar errores que se pudieron cometer a la hora de la programación del código.

Consiste prácticamente en volver a escribir el código validando con una base de datos alterna, cada uno de las rutas, vistas, etc.

r. ORM ELOQUENT

El mapeador de objetos relacional Eloquent, es una potente herramienta que permite realizar un sinfín de consultas a la base datos de manera ordenada y confiable, sin temor a que se vaya a obtener información erronea.

Aporta a la seguridad del programa, porque si las consultas y los resultados que se obtiene son confiables, permite dar información de calidad.

s. MÉTODO POST Y MÉTODO GET:

Los métodos POST y GET sirven para enviar información al servidor con la única diferencia es que POST oculta las variables que va a enviar, y GET las envía por la URL, es decir visibles al usuario y al atacante, y es aquí donde está la clave, cuando la información que se requiere enviar es importante y contiene información sensible,

se debe de enviar por el método POST de lo contrario se puede enviar por el método GET.

7. DISEÑO

Requerimientos funcionales y no funcionales

Descripción general

Perspectiva del prototipo

El prototipo consiste en un gestor de contenido de facilidad de uso con muy buena seguridad y un diseño adaptativo que cumpla con los requerimientos de responsivo y diseño elástico.

Funcionalidad del prototipo

El prototipo permite al usuario crear una página web con solo seguir unos pasos previamente estudiados para el diseño de sitios web, ofreciendo al usuario que no conoce del diseño de páginas web un entorno agradable, de fácil manejo y seguro, permitiendo el desarrollo de sus proyectos, así el usuario desconoce del desarrollo (código) de las mismas páginas.

Suposiciones y dependencias

El prototipo se va a diseñar para multiplataforma Windows, Mac, Linux, Tablet y dispositivos móviles que tengan la capacidad de correr el software, los equipos cómputo deben de cumplir los requerimientos mínimos de un procesador 1.0 GHz, RAM 256 MB y capacidad de almacenamiento de 50 MB y los dispositivos móviles con RAM de 512 MB y que cuenten con sistema operativo 4.0 en adelante para dispositivos Android y IOS versión 5.

7.1. REQUISITOS FUNCIONALES

Nombre	Base datos
Tipo	Requisito
Prioridad	Alta
Descripción	Cada vez que se cree una nueva cuenta con los datos de cada usuario se debe almacenar en la base de datos (Sean en unidad de almacenamiento del pc o servidor)
Entrada	Datos ingresados por el usuario
Proceso	Guardar datos del usuario
Salidas	Registro de datos

Tabla 1: Requisitos Funcionales - Base de datos

Nombre	Interfaz de Registro de datos
Tipo	Requisito
Prioridad	Alta
Descripción	El sistema contara con una interfaz amigable e intuitiva para que el usuario diligencie de manera fácil los datos de su registro.
Entrada	Información del usuario
Proceso	Interfaz de registro
Salidas	Datos registrados

Tabla 2: Requisitos Funcionales - Interfaz de Registro de Datos

Nombre	Sistema adaptable
Tipo	Requisito
Prioridad	Alta
Descripción	El sistema debe de tener la capacidad de adaptarse a cualquier dispositivo (computadores, celulares, Tablet).

Entrada	Gestor de contenido
Proceso	Adaptación al dispositivo huésped
Salidas	Utilizar el gestor de contenido en dispositivo huésped

Tabla 3: Requisitos Funcionales - Sistema Adaptable

Nombre	Plantilla de interfaz
Tipo	Requisito
Prioridad	Alta
Descripción	El sistema contara con una plantilla, que permiten al usuario hacer modificaciones y adaptarlas a su gusto
Entrada	Plantilla del sistema
Proceso	Modelar plantilla por usuario
Salidas	Adaptación de la plantilla al gusto del usuario

Tabla 4: Requisitos Funcionales - Plantilla de Interfaz

Nombre	Compatibilidad con los navegadores
Tipo	Requisito
Prioridad	Alta
Descripción	El sistema debe ser compatible con los navegadores de internet actuales (Internet Explorer, Mozilla Firefox, Google Chrome).
Entrada	Gestor de datos
Proceso	Compatibilidad con el navegador
Salidas	Visualización en el navegador

Tabla 5: Requisitos Funcionales - Compatibilidad con los Navegadores

7.2. REQUERIMIENTOS NO FUNCIONALES

Nombre	Ataques de seguridad
Tipo	Requisito
Prioridad	Alta
Descripción	El sistema debe de estar en la capacidad de evitar ataques tales como SQL inyection, Ataques xss, Middleware, Ataques CSRF, Validación de formularios.
Entrada	Ataques SQL inyection, Ataques xss, Middleware, Ataques CSRF, Validación de formularios.
Proceso	Detectar ataque
Salidas	Detener ataque

Tabla 6: Requisitos No Funcionales - Ataques de Seguridad

Nombre	Operatividad
Tipo	Requisito
Prioridad	Alta
Descripción	El gestor de contenido debe de ser de fácil manejo para los usuarios que no tiene en conocimiento en el desarrollo de páginas web
Entrada	Idea de la página web
Proceso	Diseño y configuración de la página web por parte del usuario
Salidas	Página web terminada por el usuario

Tabla 7: Requisitos No Funcionales - Operatividad

Nombre	Acceso a cuentas
Tipo	Requisito
Prioridad	Alta

Descripción	El acceso a las cuentas debe de ser a través de claves de mínimo 6 dígitos
Entrada	Inicio de sección
Proceso	Solicitar clave de acceso
Salidas	Iniciar sección

Tabla 8: Requisitos No Funcionales - Acceso a las Cuentas

Nombre	Visualización con los navegadores
Tipo	Requisito
Prioridad	Alta
Descripción	El sistema debe poder visualizarse en los navegadores actuales (Internet Explorer, Mozilla Firefox, Google Chrome).
Entrada	Proyecto web en ejecución
Proceso	Adaptación al navegador
Salidas	Visualizar el proyecto actual

Tabla 9: Requisitos No Funcionales - Visualización con los Navegadores

Nombre	Tiempo en visualización
Tipo	Requisito
Prioridad	Media
Descripción	El sistema no debe de tardar más 6 segundos en visualizar el contenido modificado en el gestor de contenidos.
Entrada	Proyecto web
Proceso	Modificar contenido
Salidas	Cambios realizados en el gestor de contenidos

Tabla 10: Requisitos No Funcionales - Tiempo en Visualización

Nombre	Usabilidad
Tipo	Requisito

Prioridad	Alta
Descripción	El sistema debe presentar mensajes de error que permitan al usuario identificar el tipo de error
Entrada	Problema presentado al usuario en alguna parte del proyecto.
Proceso	Analizar el problema
Salidas	Determinar qué tipo de problema es y visualizar en pantalla

Tabla 11: Requisitos No Funcionales - Usabilidad

Nombre	Limitación de plantillas
Tipo	Requisito
Prioridad	Media
Descripción	El sistema solo contara con una plantilla base la cual puede ser modificada por el usuario
Entrada	Plantilla base
Proceso	Modificar plantilla base del gestor de contenidos
Salidas	Plantilla modificada al gusto del usuario

Tabla 12: Requisitos No Funcionales - Limitación de Plantillas

7.3. CASO DE USO

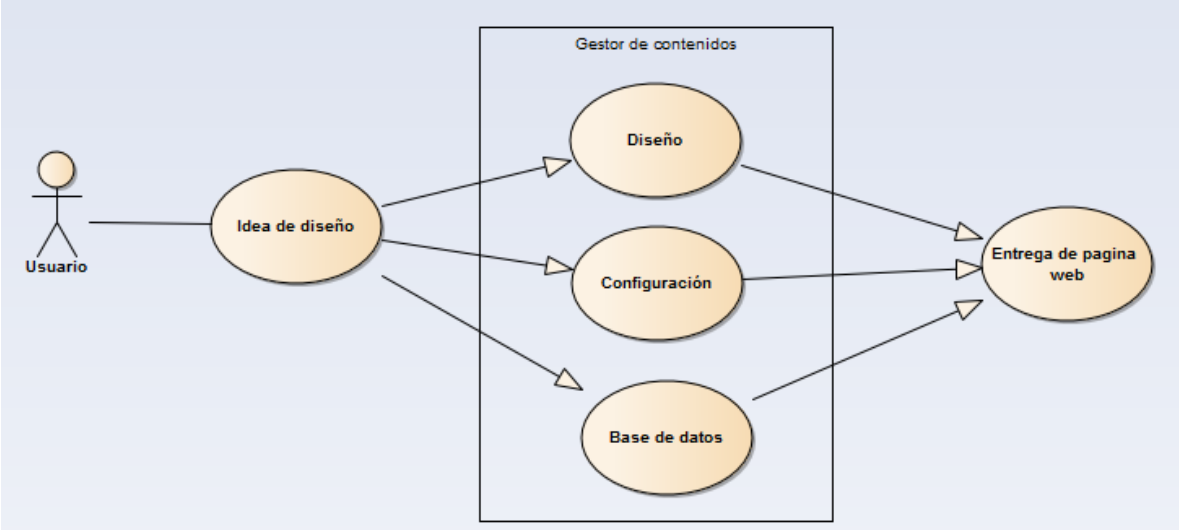


Figura 22: Caso de uso general

Configuración

- Registro
- Inicio de sesión
- Seguridad
- Por en línea (Subir a servidores)

Diseño

- Plantilla
- Estilo encabezado, cuerpo y pie de página.
- Vistas
- Editar contenido

Base de datos

- Almacenamiento de información

7.4. HISTORIAS DE USUARIO

Historia de usuario	
Numero: 1	**Nombre:** Registro
Iteración:	**Prioridad del negocio:** Alta
Descripción:	Yo como usuario que no cuento con una cuenta espero se me pidan los siguientes datos: • Nombre y apellidos • Correo electrónico • Nombre de la cuenta • contraseña
Observaciones:	El proceso de registro solo se lleva acabo, cuando el usuario apruebe el registro

Tabla 13: Historia de Usuario – Registro

Historia de usuario	
Numero: 2	**Nombre:** compatibilidad con dispositivos
Iteración:	**Prioridad del negocio:** Muy alta
Descripción:	Todos los usuarios esperan que el software sea totalmente compatible con los dispositivos actuales del mercado. • Tablet • Celulares • Computadores
Observaciones:	El gestor de contenidos se adaptará de acuerdo a cada dispositivo.

Tabla 14: Historia de Usuario - Compatibilidad con Dispositivos

Historia de usuario	
Numero: 3	**Nombre:** compatibilidad con navegadores
Iteración:	**Prioridad del negocio:** Alta
Descripción:	Todos los usuarios esperan que el software sea totalmente compatible con los siguientes navegadores. • Internet Explorer 7 o superior • Firefox 23 o superior. • Google chrome.
Observaciones:	Ninguna

Tabla 15: Historia de Usuario - Compatibilidad con Navegadores

Historia de usuario	
Numero: 4	**Nombre:** Desempeño
Iteración:	**Prioridad del negocio:** Muy Alta
Descripción:	Todos los usuarios esperan que el software responda de manera oportuna, para este caso se espera que: • El software facilite las herramientas para el diseño de las páginas web • El software soporte varios usuarios al tiempo trabajando en el mismo proyecto. • El software sea rápido al momento de subir y modificar plantillas de las páginas.
Observaciones:	Ninguna

Tabla 16: Historia de Usuario - Desempeño

Historia de usuario	
Numero: 5	**Nombre:** Seguridad
Iteración:	**Prioridad del negocio:** Muy Alta
Descripción:	Todos los usuarios esperan que el software cuente con una seguridad para ataques xss, SQL inyection, validación de cuentas, validación de email, que se realicen cambios de contraseñas periódicamente, numero de intentos de sección limitados, validación de rutas.
Observaciones:	Ninguna

Tabla 17: Historia de Usuario - Seguridad

Historia de usuario	
Numero: 6	**Nombre:** simplicidad de diseño
Iteración:	**Prioridad del negocio:** Muy Alta
Descripción:	Todos los usuarios esperan que el software se lo mas fácil de manejar, que un usuario que nunca ha diseñado una página web, se sienta en la capacidad de construir un sitio web con solo seguir unos pequeños pasos descriptos en el gestor.
Observaciones:	El usuario debe de saber manejar el computador en los aspectos más básicos.

Tabla 18: Historia de Usuario - Simplicidad de Diseño

Historia de usuario	
Numero: 7	**Nombre:** plantilla
Iteración:	**Prioridad del negocio:** Alta
Descripción:	Todos los usuarios esperan que el software contenga una serie de plantillas la cual permitan al usuario empezar a desarrollar su sitio web en base a estas plantillas.
Observaciones:	Se posea un numero límite de plantillas

Tabla 19: Historia de Usuario - Plantilla

Historia de usuario	
Numero: 8	**Nombre:** Modificar plantillas
Iteración:	**Prioridad del negocio:** Media
Descripción:	Todos los usuarios esperan que el software permita modificar las plantillas predefinidas en el gestor de contenidos.
Observaciones:	Ninguna

Tabla 20: Historia de Usuario - Modificar Plantilla

Historia de usuario	
Numero: 9	**Nombre:** conexión a internet
Iteración:	**Prioridad del negocio:** Alta
Descripción:	Todos los usuarios esperan que el software permita exportar su proyecto a un sitio en la web de manera fácil y rápida.

Observaciones:	Siempre se debe de contar con una conexión a internet, la subida del sitio web depende del ancho de banda de cada usuario.

Tabla 21: Historia de Usuario - Conexión a Internet

Historia de usuario	
Numero: 10	**Nombre:** Sin conexión a internet
Iteración:	**Prioridad del negocio:** Media
Descripción:	Todos los usuarios esperan que el software permita trabajar en los sitios web sin necesidad de contar con una conexión a internet y esta posteriormente pueda ser subida a la web.
Observaciones:	Tener espacio de almacenamiento en el computador que se encuentre trabajando.

Tabla 22: Historia de Usuario - Sin Conexión a Internet

Historia de usuario	
Numero: 11	**Nombre:** Base de datos
Iteración:	**Prioridad del negocio:** Alta
Descripción:	Todos los usuarios esperan que el software tenga una base de datos en la que se pueda consultar todos los proyectos realizados ya sea con conexión o sin conexión a internet.
Observaciones:	

Tabla 23: Historia de Usuario - Base de Datos

Historia de usuario	
Numero: 12	**Nombre:** Exportar datos
Iteración:	**Prioridad del negocio:** Media
Descripción:	Los usuarios esperan que el software permita exportar los proyectos realizados en un tipo de dato compatible con otros gestores.
Observaciones:	

Tabla 24: Historia de Usuario - Exportar Datos

7.5. DISEÑO DE CASO DE PRUEBA

La planificación de pruebas se realizó con el fin de comprobar y autenticar que los requisitos funcionales y no funcionales del gestor de contenidos permitirán revisar el avance, detectar errores y garantizar la trazabilidad de los requerimientos.

El oportuno desarrollo de pruebas y aplicación de estas permite encontrar errores, defectos y fallas en la etapa de desarrollo para poder realizar las correcciones necesarias y evitar problemas cuando se ponga en marcha el proyecto, lo cual asegura la calidad del producto.

7.6. DESCRIPCIÓN CASO DE USO

Caso de uso:	**Registro**
Actores:	Usuario, Administrador
Propósito:	Este caso de uso permite al usuario registrarse al sistema.
Resumen:	El usuario hace la solicitud en gestor de contenido de página web, para crear una cuenta, ingresa los datos personales como nombre, apellidos, correo electrónico y una contraseña.
Tipo:	Esencial
Referencia:	
Precondición:	Cualquier usuario que quiera probar el gestor.
Curso normal de los eventos	
Acción de los actores	**Respuesta del sistema**

1 El usuario solicita crear una cuenta en la opción de registrarse	**2** El sistema despliega un formulario con los siguientes datos: nombre, apellido, correo electrónico y contraseña.
3 El usuario ingresa los datos.	**4** El sistema verifica que los datos sean consistentes.
	5 el sistema verifica que el correo sea válido.
	6 El sistema envía un correo de aprobación del registro a la cuenta del usuario con un link de aprobación.
7 el usuario se logue en el link enviado al correo.	
Pos condición:	
Curso alterno	
Acción 4:	Si algunos de los datos no cumplen con el formato o ya existe el mismo correo, muestra un mensaje donde se indica que se debe ingresar el dato o si el correo ya está registrado.
Frecuencia esperada:	baja

Tabla 25: Descripción Caso de Uso - Registro

Caso de uso:	**Compatibilidad con dispositivos**
Actores:	Sistema
Propósito:	Este caso de uso permite al gestor de páginas web interactuar con los diferentes dispositivos del entorno.

Resumen:	El sistema interactúa con los distintos dispositivos presentes en el mercado, observado el desempeño y adaptabilidad para llevar a cabo cada uno de los proyectos web.
Tipo:	Esencial
Referencia:	
Precondición:	• Cualquier dispositivo que cumpla con los requerimientos del gestor de contenidos. • Tener instalado el gestor de contenido en el dispositivo.
Curso normal de los eventos	
Acción de los actores	**Respuesta del sistema**
1 inicio de sección del gestor de contenidos en el dispositivo.	**2** el sistema envía una solicitud al dispositivo huésped
3 El dispositivo responde de que tipo es: Computo, Tablet o móvil	**4** El sistema toma la solicitud enviada por el dispositivo y es procesada
	5 El sistema se acopla al dispositivo huésped.
5 El usuario empieza a utilizar el gestor de contenidos en su dispositivo.	
Pos condición:	
Curso alterno	
Acción 3:	Si el dispositivo no responde de qué tipo es, el gestor muestra un mensaje de error de compatibilidad.
Frecuencia esperada:	baja

Caso de uso:	**Compatibilidad con navegadores**
Actores:	Usuario
Propósito:	En este caso de uso se quiere evidenciar la compatibilidad que tiene el gestor de contenidos con los navegadores.
Resumen:	Observar que desempeño tiene el gestor de contenidos al momento de interactuar con los distintos navegadores de la web.
Tipo:	Esencial
Referencia:	
Precondición:	Tener un proyecto web en el gestor de contenido y querer visualizar en el navegador.
Curso normal de los eventos	
Acción de los actores	**Respuesta del sistema**
1 El usuario solicita visualizar el proyecto que se tiene en el gestor de contenido.	**2** El gestor de contenidos procesa la solicitud y envía la solicitud al navegador predeterminado.
	3 El navegador responde la solicitud del sistema, enviado información sobre el navegador.
	4 El gestor de contenidos adapta el proyecto al navegador.
5 El usuario puede visualizar en el navegador el proyecto web solicitado.	
Pos condición:	

Curso alterno	
Acción 4:	Si el gestor de contenido no reconoce el navegador, envía un mensaje informando que no es compatible con el navegador.
Frecuencia esperada:	baja

Tabla 26: Descripción Caso de Uso - Compatibilidad con Dispositivos

Caso de uso:	**Plantilla**
Actores:	Usuario, Administrador
Propósito:	Este caso de uso permite al usuario utilizar la plantilla predeterminada en el gestor de contenidos.
Resumen:	El usuario hace clic en el botón de plantillas, inmediatamente se despliega un diseño de plantilla predeterminado por el sistema, la cual pueden ser utilizada y modificado en los proyectos web.
Tipo:	Esencial
Referencia:	
Precondición:	Cualquier usuario que tenga un proyecto en el gestor de contenidos y quiera probar la plantilla.
Curso normal de los eventos	
Acción de los actores	**Respuesta del sistema**
1 El usuario solicita dando clic en el botón de plantilla	**2** El sistema despliega la plantilla predeterminada del sistema.
3 El usuario visualiza la plantilla	

Tabla 27: Descripción Caso de Uso - Plantilla

Caso de uso:	**Modificar plantilla**
Actores:	Usuario, Administrador
Propósito:	Este caso de uso permite al usuario modificar la plantilla predefinida por el sistema.
Resumen:	El usuario después de tener cargada la plantilla en el gestor de contenidos, puede modificar su encabezado, su cuerpo y pie de página.
Tipo:	Esencial
Referencia:	
Precondición:	Tener un proyecto web abierto en el gestor de contenidos.
Curso normal de los eventos	
Acción de los actores	**Respuesta del sistema**
1 El usuario selecciona que parte de la plantilla desea modificar (Cabeza, cuerpo o pie de la página).	**2** El sistema despliega un conjunto de herramientas según la parte de la plantilla que se quiere modificar.
3 El usuario selecciona los elementos que quiere modificar y da en el botón guardar	**4** El sistema carga la plantilla modificada.
5 El usuario visualiza los cambios realizados a la plantilla	
Pos condición:	
Curso alterno	
Acción 4:	El sistema muestra un mensaje si se presenta problemas para cargar la plantilla
Frecuencia esperada:	Baja

Caso de uso:	**Exportar datos**
Actores:	Usuario, Administrador
Propósito:	Este caso de uso permite al usuario exportar los proyectos realizados en el gestor de contenido.
Resumen:	El usuario después de tener cargado su proyecto puede exportar su diseño para que sea compatible con otros editores
Tipo:	Esencial
Referencia:	
Precondición:	Tener un proyecto web.
Curso normal de los eventos	
Acción de los actores	**Respuesta del sistema**
1 El usuario selecciona en el botón exportar.	**2** El sistema procesa la solicitud realizada.
	3 El sistema convierte la información del diseño en un paquete exportable.
4 El usuario obtiene un archivo compatible con los demás editores	
Pos condición:	

Tabla 28: Descripción Caso de Uso - Modificar Plantilla

7.7. DIAGRAMA DE SECUENCIA

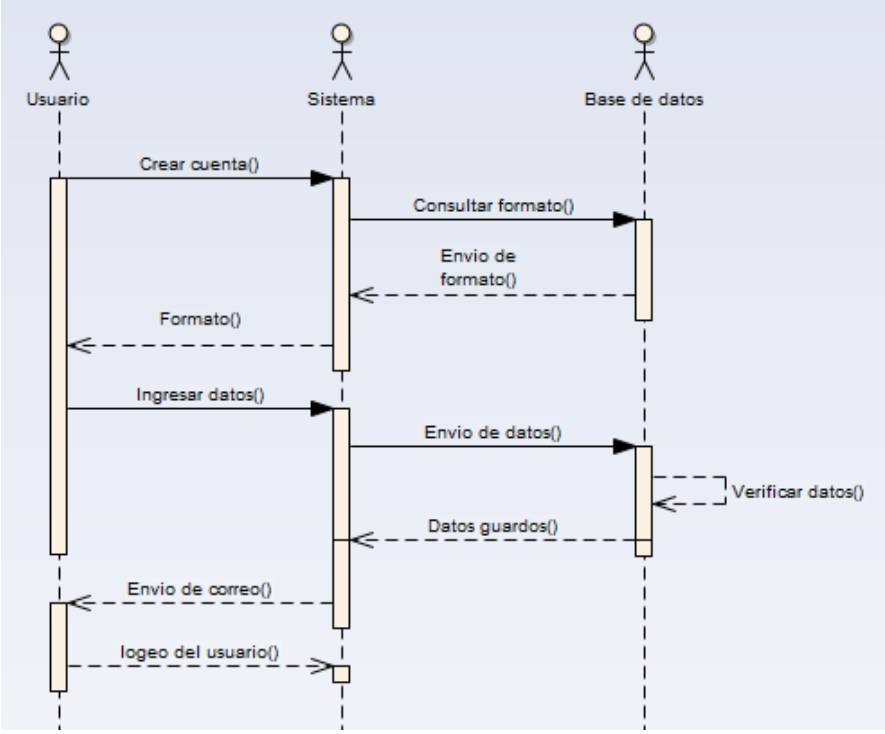


Figura 23: Diagrama de secuencias registró

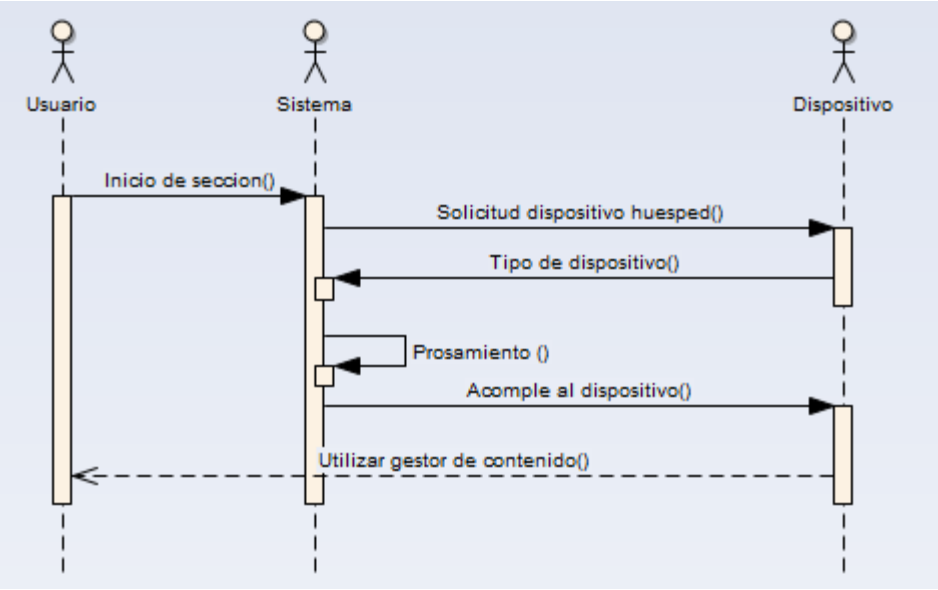


Figura 24: Diagrama de secuencias compatibilidad con el dispositivo

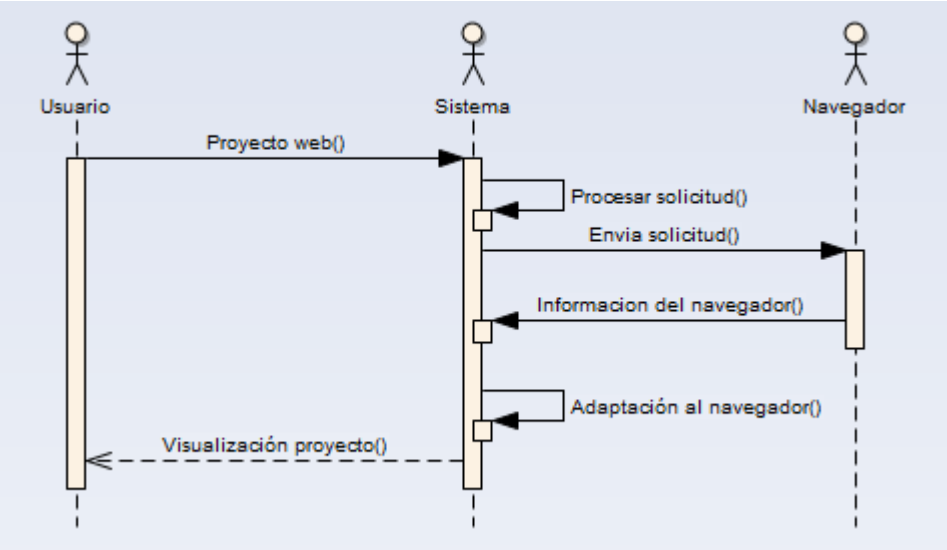


Figura 25: Diagrama de secuencias compatibilidad con el navegador

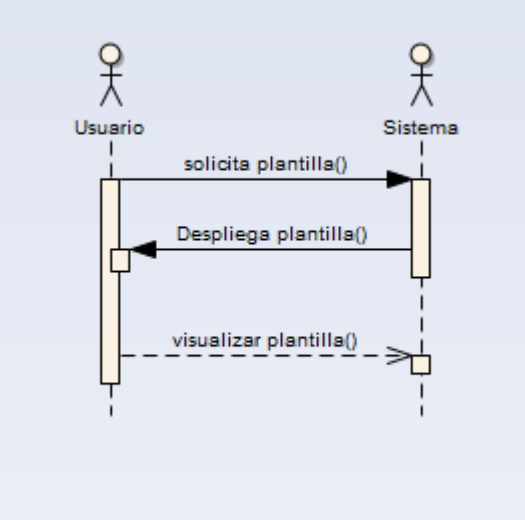


Figura 26: Diagrama de secuencias plantilla

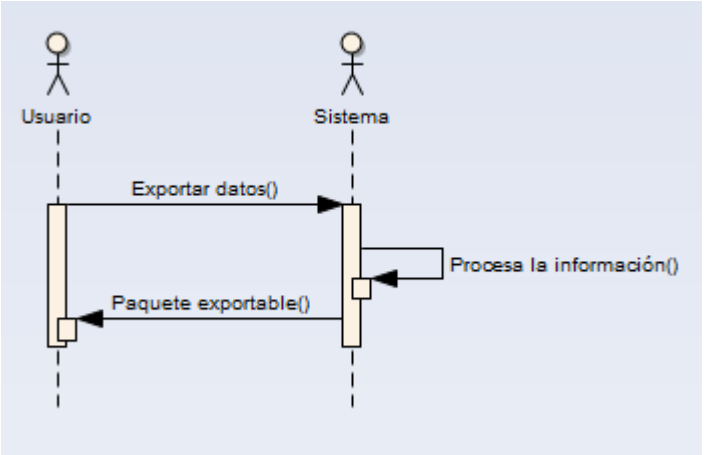


Figura 27: Diagrama de secuencias exportar datos

Diagrama de actividades

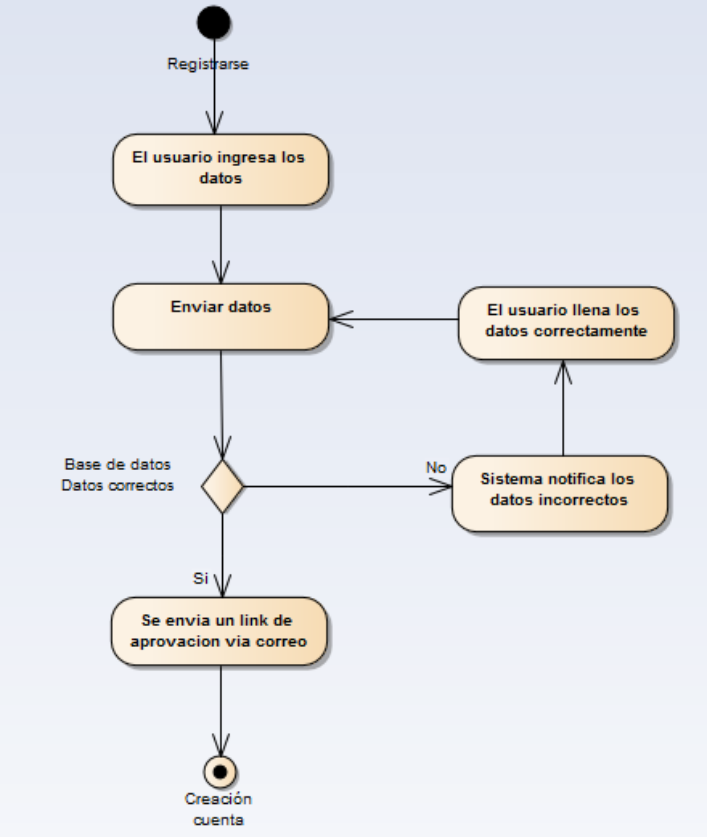


Figura 28: Diagrama de actividades registró

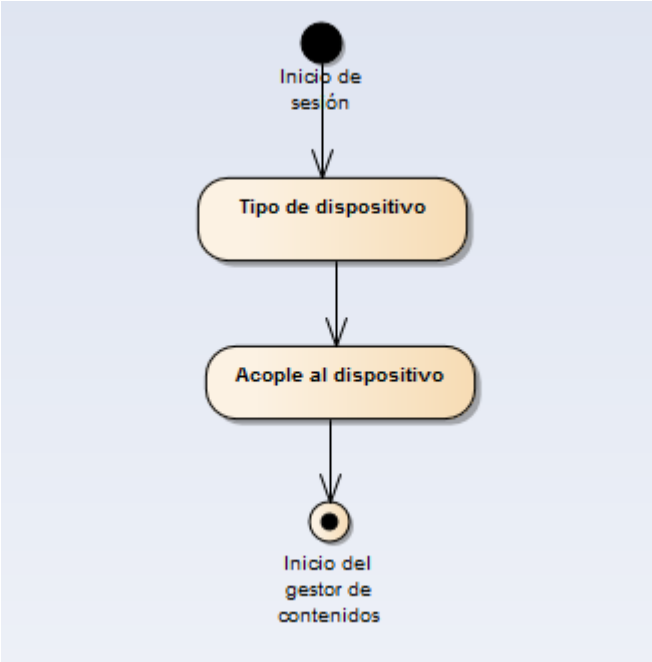


Figura 29: Diagrama de actividades compatibilidad con el dispositivo

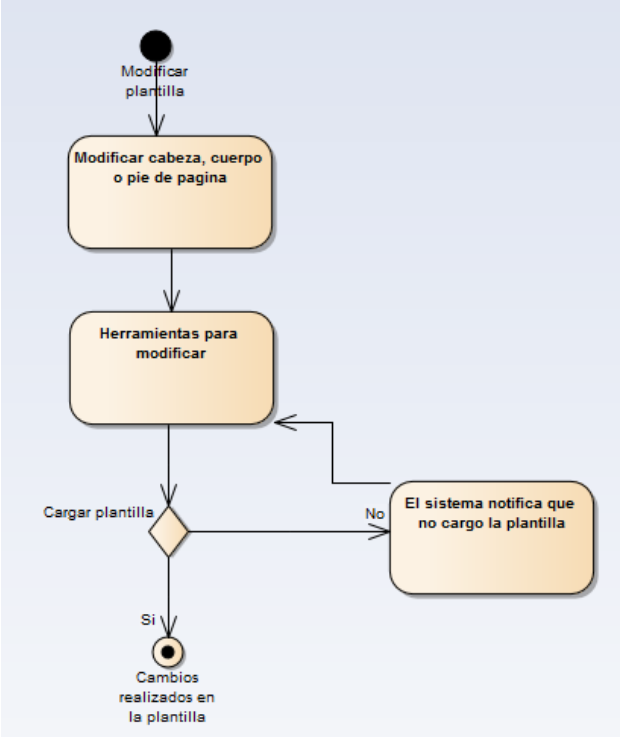


Figura 30: Diagrama de actividades modificar plantilla

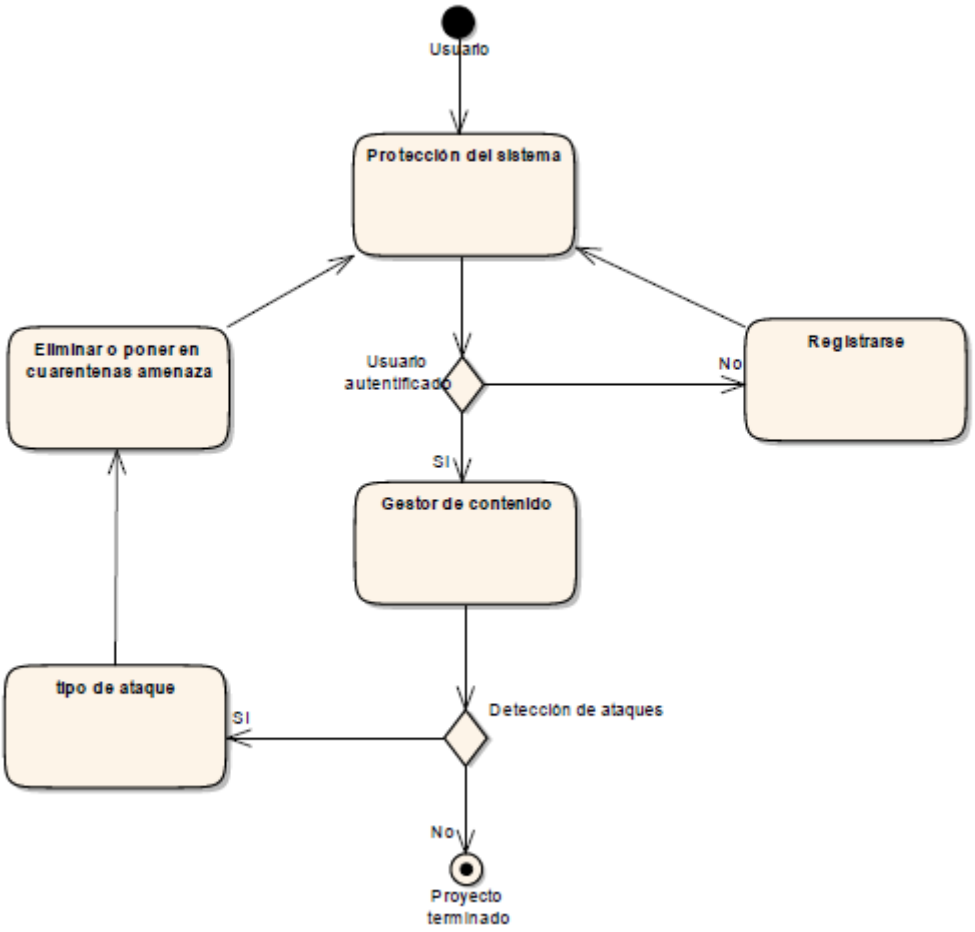


Figura 31:Diagrama de actividades seguridad

7.8. VISTAS DEL HARDWARE

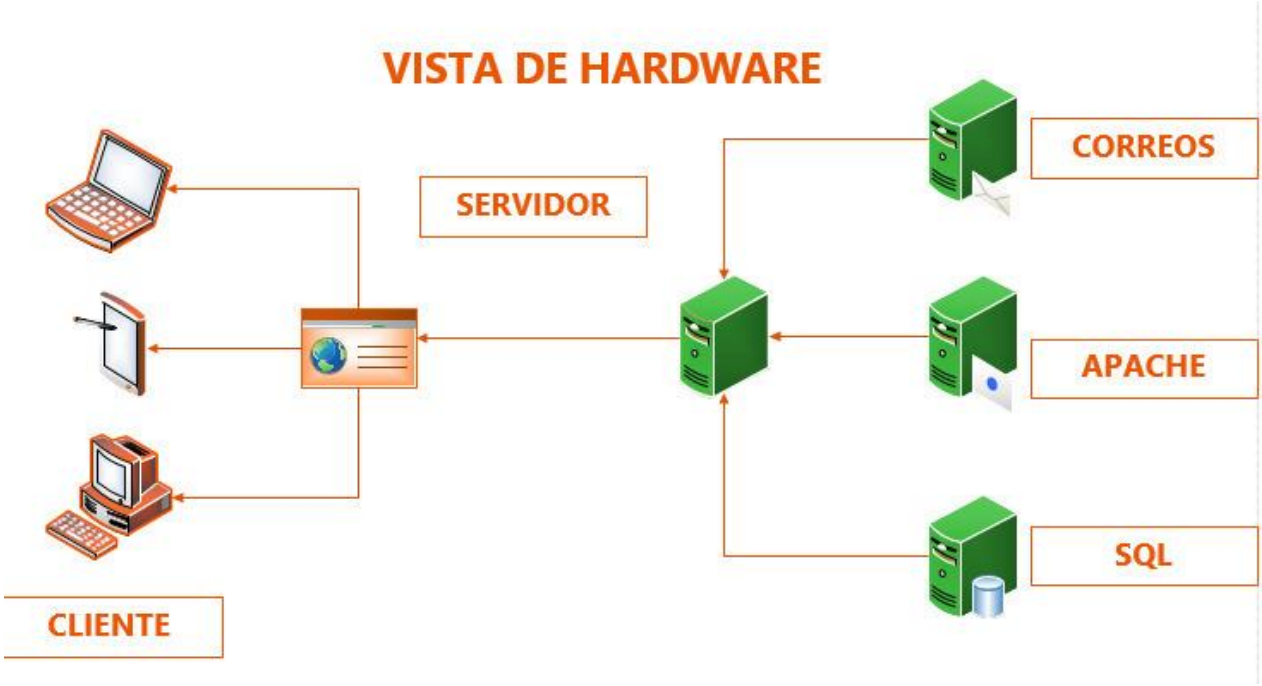


Figura 32: Vista del hardware

7.9. VISTA DEL SOFTWARE

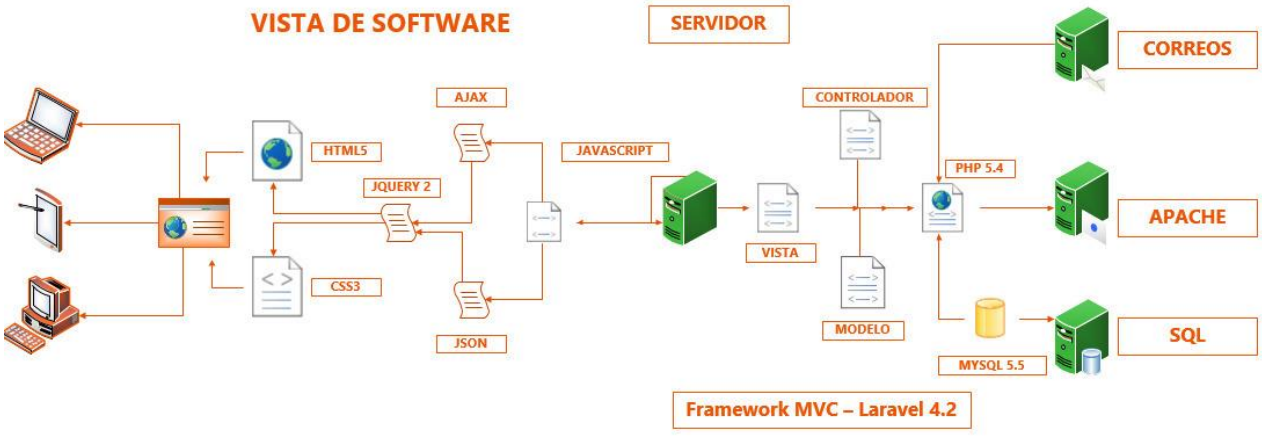


Figura 33: Vista del software

7.10. MODELO DE BASE DE DATOS

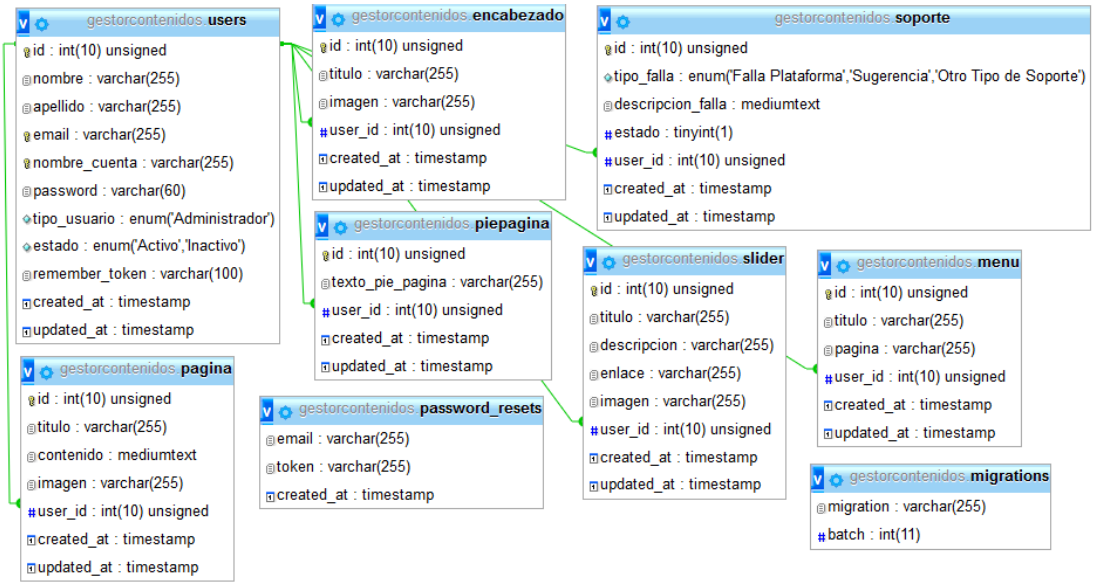


Figura 34: Modelo Entidad – Relación de la base de datos

7.11. EJECUCIÓN DE CASOS DE PRUEBAS

Descripción	Estado	Resultado	Observaciones
Registro del usuario	Ejecutada	El usuario se puede registrar satisfactoriamente y comprueba correctamente los campos	
Inicio de sesión	Ejecutada	El usuario puede registrase correctamente en sistema.	
Compatibilidad de dispositivos	Ejecutada	Se comprobó la compatibilidad del gestor de contenidos arrojando un resultado satisfactorio con los dispositivos de computo, Tablet y móviles móviles)	Recordar que el gestor de contenidos fue probado en un sistema operativo android versión 4.0 o superior y IOS versión 4 en adelante.
Compatibilidad navegadores	ejecutada	Los diseños realizados en el gestor de contenidos pueden visualizarse satisfactoriamente	

		en los siguientes navegadores (Google Chrome, Internet Explorer y Mozilla Firefox).	
Editar Plantillas	Ejecutada	Se comprobó que se cargaran la plantilla predeterminada en el sistema, además de permitir modificar su contenido (cabeza, cuerpo y pie de página).	Se debe tener en cuenta que el gestor de contenidos permite modificar solo algunas características de la plantilla.
Seguridad	Ejecutada	Se realizar pruebas del tipo xss, validación de rutas, ataques SQL inyection, validaciones de email, se cuenta con una lista negra que va recopilando ips maliciosas, lista de palabras reservadas del sistema.	

Tabla 29: Ejecución Caso de Pruebas

8. PROTOTIPO

8.1. PÁGINA PRINCIPAL

Figura 35: Página de Inicio Prototipo de Sistema Gestor de Contenidos

8.2. INICIO DE SESIÓN

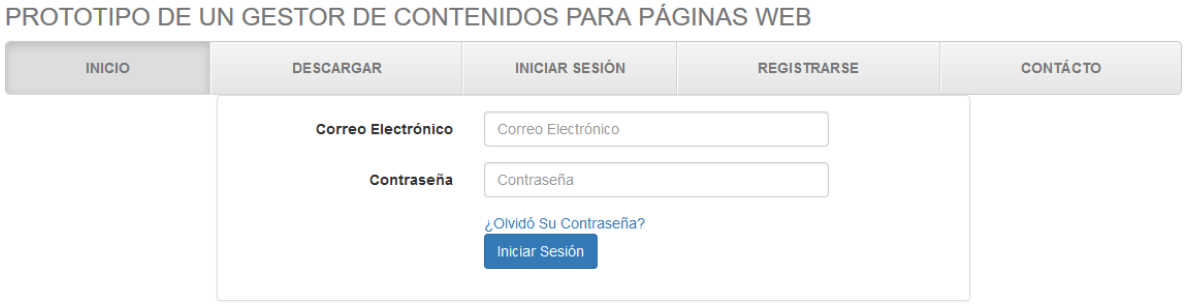


Figura 36: Página de Inicio de Sesión

8.3. VISTA DENTRO DEL SISTEMA

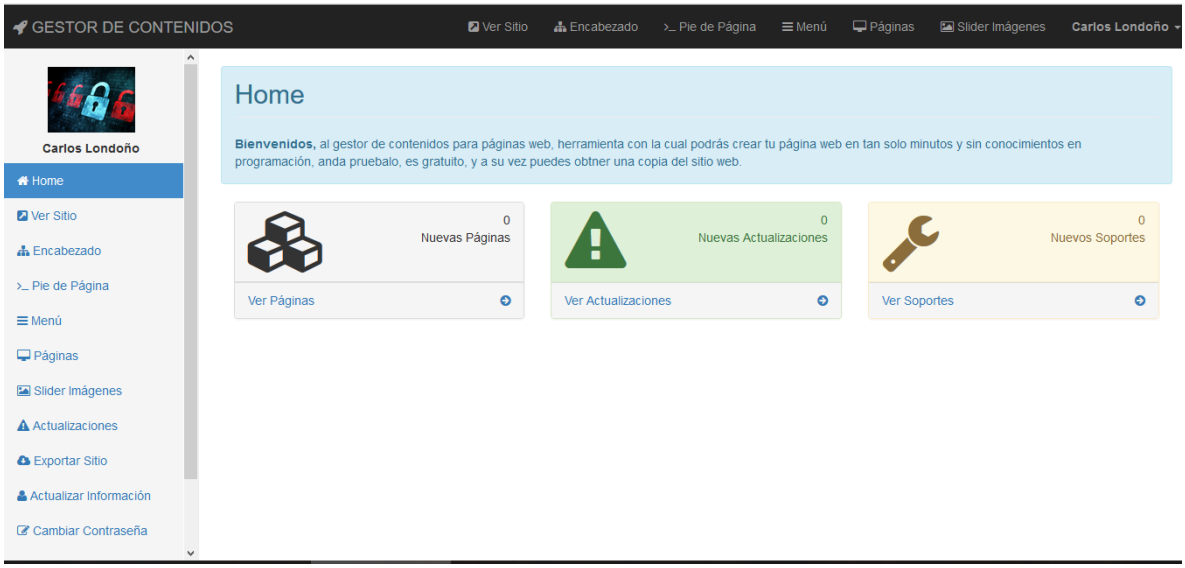


Figura 37: Vista dentro del Prototipo

9. DISEÑO METODOLOGICO PRELIMINAR

9.1. HIPÓTESIS

¿Es posible crear un gestor de contenidos que sea fácil de usar, seguro y que tenga un diseño agradable al usuario?

9.2. TIPO DE INVESTIGACIÓN

Documental y practica

9.3. POBLACIÓN

Con este proyecto se beneficia todas aquellas personas que tengan dificultad y desconocimiento en el uso de herramientas web para la administración de páginas web.

10. VARIABLES

Gestor de contenidos

10.1. DISEÑO DE INSTRUMENTOS PARA TOMA DE INFORMACIÓN

Se utilizará el modelo de investigación web, que permita buscar información en las diferentes páginas web sobre la estructura de una página web, técnicas de seguridad, desarrollo de sitios web adaptivos y diseños agradables al usuario.

11. PARTICIPANTES DEL PROYECTO

Estudiantes:

Carlos Alberto Londoño Loaiza
John Alexánder Calderón Hernández

Docente:

Ingeniero Juan de Jesús Veloza Mora

12. RECURSOS DISPONIBLES

Servidor Web, para realizar las pruebas del funcionamiento correcto del software a la hora de haber finalizado el proyecto.

Computador, realizar la programación de cada una de las etapas del software y hacer las pruebas pertinentes.

Conexión a internet, para realizar las diferentes investigaciones que nos permitan obtener información puntual sobre el proyecto.

Acceso a diferentes gestores de contenido, para tomar como referente su funcionamiento

13. CRONOGRAMA

CONCEPTO	AÑO 2015											
	SEPTIEMBRE				OCTUBRE				NOVIEMBRE			
	1	2	3	4	5	6	7	8	9	10	11	12
DISEÑO DEL PROYECTO												
CREACIÓN Y CONFIGURACION DEL SERVIDOR												
CONFIGURACION DEL DOMINIO												
PUBLICACIÓN DEL CODIGO FUENTE												
CONFIGURACION DEL SERVIDOR DE BASES DE DATOS												
PRUEBAS DE FUNCIONAMIENTO												
PRUEBAS DE SEGURIDAD												
CREACION DE ENTORNO DE USUARIOS DE PRUEBAS												
PUBLICACION FINAL DEL PROYECTO												

Tabla 30: Cronograma de Actividades

14. CONCLUSIONES

Con este proyecto se logró comprender las diferentes etapas que tiene el desarrollo de un prototipo de software en cuanto a su requerimiento, análisis y desarrollo del software.

En el desarrollo del prototipo se logró unir diferentes componentes que ofrece la web para la implementación del software.

A través del desarrollo del proyecto se logró aplicar los conocimientos adquiridos en el transcurso de la carrera, principalmente de las materias de ingeniería del software, laboratorio del software y base de datos.

15. RECOMENDACIONES

Masificar la capacitación y uso del prototipo.

Haciendo uso de la pestaña soporte en el menú principal del prototipo, cualquier usuario pueden reportar dificultades o inconvenientes que se presenten, lo mismo que hace recomendaciones para mejorar su interfaz y funcionalidad.

Si el usuario es una persona con conocimientos avanzados, puede descargar del sitio el código fuente del proyecto y adaptarlo a sus necesidades. No sobra recordar que debe tener en cuenta los derechos de autor.

Si podría implementar en las instituciones educativas con el fin de que los estudiantes, desde muy temprana edad incursionen en el diseño de páginas web.

16. GLOSARIO

- Gestor de contenidos: Los sistemas de gestión de contenidos (Content Management Systems o CMS) es un software que se utiliza principalmente para facilitar la gestión de webs, ya sea en Internet o en una intranet, y por eso también son conocidos como gestores de contenido web (Web Content Management o WCM). Hay que tener en cuenta, sin embargo, que la aplicación de los CMS no se limita sólo a las webs. (Universidad Abierta de Cataluña, 2004)

- Usuario: persona que interactúa con el proyecto.

- Tecnologías web modernas: conjunto de herramientas y prácticas vanguardistas que usan para la construcción de aplicativos webs.

- Seguridad: capacidad de un sistema para no permitir el uso de personas ajenas al sistema.

- Usabilidad: capacidad de un sistema de poseer características que permitan que las personas que no lo conocen lo puedan usar de una manera muy fácil.

- Diseño adaptivo: capacidad de los sistemas de adaptarse a cualquier entorno donde se use.

- Framework: un conjunto estandarizado de conceptos, prácticas y criterios para enfocar un tipo de problemática particular que sirve como referencia, para enfrentar y resolver nuevos problemas de índole similar. (Wikipedia, 2015)

- Laravel: Es un framework escrito en php que implementa el patrón modelo vista controlador, creado por Taylor Otwell en el 2011, sirve para desarrollar

aplicaciones modernas de manera fácil y segura, resuelve situaciones como manejo de eventos, autenticación, etc. (Anton, 2015), (Otwell, 2011)

- HTML: lenguaje de marcado de texto.
- CSS: hoja de estilos en cascada
- JAVASCRITP: lenguaje de programación que funciona en el navegador, para escribir páginas web dinámicas.
- JQUERY: librería escrita en JAVASCRIPT, para interactuar con el DOM, (The Jquery Foundation, 2006)
- Backend: conjunto de códigos de programación que corren en el servidor. (PLATZI, 2013)
- Frontend: conjunto de códigos que corren en el cliente, es decir el navegador. (PLATZI, 2012)
- Bootstrap: Conjunto de herramientas CSS, HTML y Javascript optimizado para ser usado en los proyectos, (Twitter, 2009)
- Font Awesome: Fuente vectorial de iconos libres para usar en proyectos web. (Dave Gandy, 2014)
- JSON: Javascript Objetc Notation, es un formato ligero para el intercambio de información entre las bases de datos y el frontend. (Javascript Foundation, 2008)
- JQUERY: Librería en Javascript que facilita la interacion entre los elementos de Javascript y el HTML. (The Jquery Foundation, 2006)

- AJAX: Asynchronous JavaScript + XML, es el formato utilizado para realizar la actualización inmediata de datos en el cliente, permite hacer peticiones asincrónicas. (Libros Web, 2006)

- Datatables: Conjunto de herramientas escritas en HTML, CSS y Javascript, que facilitan el uso de las tablas, a través de formatos ya realizados. (SpryMedia, 2007)

- Google Maps: es un servidor de aplicaciones de mapas en la web que pertenece a **Google**. Ofrece imágenes de mapas desplazables, así como fotografías por satélite del mundo e incluso la ruta entre diferentes ubicaciones o imágenes a pie de calle con **Google** Street View. (Google, 2008)

- Youtube: Canal oficial de google que permite subir y compartir videos, dispone de una de las bibliotecas de videos más grandes del mundo. (Google, 2006)

- API: Una interfaz hecha para desarrolladores que expone nuestros recursos de una forma entendible y accesible para otros. (Trinidad, 2015)

- Composer: manejador de paquetes para php, permite una fácil instalación de todos los paquetes usados para el lenguaje PHP, (Nils Adermann, 2005)

- Packagist: manejador de paquetes para Laravel, en este sitio los desarrolladores comparten porciones de código hecho, que realizan una función específica. (Toran Proxy, 2012)

- Blade: Motor de plantillas de Laravel, que permite usar HTML dinámico en los proyectos, es decir incluir etiquetas tales como if, for, etc.

- Media Queries: Una **media query** consiste en un tipo de medio y al menos una consulta que limita las hojas de estilo utilizando características del medio como ancho, alto y color. Añadido en CSS3, las media queries dejan que la

presentación del contenido se adapte a un rango específico de dispositivos de salida sin tener que cambiar el contenido en sí. (Mozilla Developer Network, 2015)

17. BIBLIOGRAFÍA

- Actualidad Geek. (10 de 2014). *Actualidad Geek*. Obtenido de http://www.actualidadgeek.org/2014/09/que-son-los-framework-de-desarrollo-y-por-que-usarlos/
- Anton, C. (19 de 08 de 2015). *PLATZI*. Obtenido de https://platzi.com/blog/laravel-framework-php/
- AS Amor Sevillista. (2013). *AS Amor Sevillista*. Obtenido de http://www.amorsevillista.com/2013/09/agregar-botones-para-compartir-en-redes.html
- Comunidad OWASP. (2005). *OWASP*. Obtenido de https://www.owasp.org/index.php/Main_Page
- Congreso de la República de Colombia. (1982). *Alcaldia de Bogota*. Obtenido de http://www.alcaldiabogota.gov.co/sisjur/normas/Norma1.jsp?i=3431
- Congreso de la República de Colombia. (2009). *MINTIC*. Obtenido de http://www.mintic.gov.co/portal/604/w3-article-3705.html
- Dave Gandy. (2014). *Font Awesome*. Obtenido de http://fortawesome.github.io/Font-Awesome/
- Developer Mozilla. (08 de 2005). *Mozilla Developer Network* . Obtenido de https://developer.mozilla.org/es/docs/Web/HTML
- Developer Mozilla. (03 de 2015). *Mozilla Developer Netwok*. Obtenido de https://developer.mozilla.org/es/docs/Web/CSS
- Developer Mozilla. (02 de 2015). *Mozilla Developer Network*. Obtenido de https://developer.mozilla.org/es/docs/CSS/Media_queries
- DIRECCIÓN NACIONAL DE DERECHOS DE AUTOR. (s.f.). *DIRECCIÓN NACIONAL DE DERECHOS DE AUTOR*. Obtenido de http://derechodeautor.gov.co/web/guest/leyes
- Faruk, Paul, Alex, Ryan, Patrick, Stu, and Richard. (2009). *Modernizr*. Obtenido de https://modernizr.com/
- Gallagher, N. (2011). *Normalize.css*. Obtenido de https://necolas.github.io/normalize.css/

- Gallagher, N. (2011). *Normalize.css*. Obtenido de https://necolas.github.io/normalize.css/
- Gandy, D. (2011). *Font Awesome*. Obtenido de http://fortawesome.github.io/Font-Awesome/
- Gobierno de Chile. (2013). *Guía Digital Beta*. Obtenido de http://www.guiadigital.gob.cl/articulo/que-es-la-usabilidad
- Google. (2006). *Youtube*. Obtenido de https://www.youtube.com/
- Google. (2008). *Google Maps*. Obtenido de https://www.google.es/maps
- Grisales, A. R. (23 de Agosto de 2009). *Lectura con Alberto*. Obtenido de http://www.alberto.com
- Ideas Digitales Aplicadas. (15 de 07 de 2015). *Ideas Digitales Aplicadas*. Obtenido de http://www.ida.cl/blog/diseno/diferencias-diseno-web-fluido-adaptativo-responsivo/
- InternetLab. (2012). *InternetLab*. Obtenido de http://www.internetlab.es/post/1560/navegadores-web/
- Javascript Foundation. (2008). *Introducing JSON*. Obtenido de http://www.json.org/
- Libros Web. (2006). *Libros Web*. Obtenido de http://librosweb.es/libro/ajax/capitulo_1.html
- Modernizr. (2012). *Modernizr*. Obtenido de https://modernizr.com/
- Mozilla Developer Network. (23 de 02 de 2015). *Mozilla Developer Network*. Obtenido de https://developer.mozilla.org/es/docs/CSS/Media_queries
- Nils Adermann, J. B. (2005). *Composer*. Obtenido de https://getcomposer.org/
- Otwell, T. (2011). *Laravel*. Obtenido de http://laravel.com/
- Palacios, D. (03 de 06 de 2015). *Styde.Net*. Obtenido de https://styde.net/exportar-hoja-de-calculo-con-eloquent-y-laravel-excel/
- Palacios, D. (14 de 05 de 2015). *Styde.Net*. Obtenido de https://styde.net/uso-del-metodo-rendersections-con-query-ajax-y-formato-json/
- Palacios, D. (17 de 09 de 2015). *Styde.Net*. Obtenido de https://styde.net/subir-archivos-en-laravel-con-dropzone/

- Palacios, D. (06 de 05 de 2015). *Styde.Net*. Obtenido de https://styde.net/autenticacion-en-laravel-5-con-eloquent-oauth-y-facebook/
- Palacios, D. (22 de 05 de 2015). *Styde.Net*. Obtenido de https://styde.net/ckeditor-en-laravel-5/
- Palacios, D. (05 de 07 de 2015). *Styde.Net*. Obtenido de https://styde.net/captcha-en-tus-formularios-con-laravel-5/
- PLATZI. (2012). *PLATZI*. Obtenido de https://platzi.com/frontend/
- PLATZI. (2013). *PLATZI*. Obtenido de https://platzi.com/backend/
- SpryMedia. (2007). *Data Tables*. Obtenido de https://www.datatables.net/
- Stallman, R. (1999). *El sistema GNU*. Obtenido de http://www.gnu.org/licenses/licenses.es.html
- The Jquery Foundation. (26 de 8 de 2006). *Jquery*. Obtenido de https://jquery.com/
- Toran Proxy. (2012). *Packagist*. Obtenido de https://packagist.org/
- Trinidad, P. (11 de 08 de 2015). *PLATZI*. Obtenido de https://platzi.com/blog/como-crear-apis/
- Twitter. (2009). *Bootstrap*. Obtenido de http://getbootstrap.com/
- Universidad Abierta de Cataluña. (29 de 11 de 2004). *MOSAIC*. Obtenido de http://mosaic.uoc.edu/2004/11/29/introduccion-a-los-sistemas-de-gestion-de-contenidos-cms-de-codigo-abierto/
- Wikipedia. (2015). *Wikipedia*. Obtenido de https://es.wikipedia.org/wiki/Framework
- Xavier García Cuerda. (29 de 11 de 2004). *Universidad Abierta de Cataluña*. Obtenido de http://mosaic.uoc.edu/2004/11/29/introduccion-a-los-sistemas-de-gestion-de-contenidos-cms-de-codigo-abierto/