# UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

## FACULTAD DE INGENIERÍA

# Portafolio de soluciones para una PYME

**INFORME DE ACTIVIDADES PROFESIONALES**

Que para obtener el título de

**Ingeniero en computación**

**P R E S E N T A**

Guillermo Antonio de la Cueva Escandón

**ASESOR DE INFORME**

M.I. Ángel César Govantes Saldivar

**Ciudad Universitaria, Cd. Mx., 2017**

# Agradecimientos

Agradezco a mi madre, Elsa, su fe ciega y su incondicionalidad, que fueron el combustible de mi vida académica. Su creencia en mi potencial (infinitamente superior a la mía propia) me inspiró a trabajar arduamente y a saltar los obstáculos frente a los que, más de una vez, me sentí tentado a detenerme.

Mi gratitud a Patricia, por su sabiduría y su guía para encontrar y mantenerme en la senda cuando me sentía perdido. Sus consejos tuvieron la visión periférica que, de vez en vez, me hacía falta; sus enseñanzas me hicieron ver las cosas de una manera más madura y serena.

Va otro reconocimiento a mi tía Lourdes por el cariño y dedicación, por su vigilante conducción y apoyo en los trances difíciles. Sin su solicita oferta de un sinfín de herramientas, sin su interés en mi tranquilidad y bienestar, no habría podido sobreponerme.

Finalmente, y con la emoción que me embarga frente a uno de los momentos más importantes de mi vida, van unas palabras para la memoria de alguien especial: Araceli (1956-2016). Siempre fiel a sí misma, fuerte ante la adversidad y dotada de enorme corazón, fue generosa con sus risas y afecto. Mi sentido abrazo filial para ella, por sus cuidados y sus consejos. Gracias Chely.

# Índice

Introducción y objetivos ..... 1
Marco teórico ..... 4
MS Consulting y desarrollo de actividades personales en ella ..... 7
I. Desarrollo de aplicación web “Simulador conductual” ..... 9
Objetivo ..... 9
Antecedentes ..... 9
Descripción del problema o contexto de la participación profesional ..... 10
Metodología utilizada y participación profesional ..... 10
Resultados ..... 19
Conclusiones ..... 19
II. Desarrollo de sitio web “DePapel” ..... 21
Objetivo ..... 21
Antecedentes ..... 21
Descripción del problema o contexto de la participación profesional ..... 21
Metodología utilizada y participación profesional ..... 21
Resultados ..... 29
Conclusiones ..... 29
III. Desarrollo de sitio web “Garani S.C.” ..... 31
Objetivo ..... 31
Antecedentes ..... 31
Descripción del problema o contexto de la participación profesional ..... 31
Metodología utilizada y participación profesional ..... 32
Resultados ..... 43
Conclusiones ..... 44
IV. Desarrollo de sitio web “CoViBE” ..... 45
Objetivo ..... 45
Antecedentes ..... 45
Descripción del problema o contexto de la participación profesional ..... 45
Metodología utilizada y participación profesional ..... 46
Resultados ..... 48
Conclusiones ..... 49

V. Diagnóstico y corrección en la aplicación web “Soluciones Virtuales PVI” ..... 51
Objetivo ..... 51
Antecedentes ..... 51
Descripción del problema o contexto de la participación profesional ..... 51
Metodología utilizada y participación profesional ..... 52
Resultados ..... 54
Conclusiones ..... 54

VI. Desarrollo de sitio web “Innova y Renueva. Asociación para el desarrollo, A.C.” ..... 55
Objetivo ..... 55
Antecedentes ..... 55
Descripción del problema o contexto de la participación profesional ..... 55
Metodología utilizada y participación profesional ..... 56
Resultados ..... 88
Conclusiones ..... 88

VII. Desarrollo de sitio web “Restatools” ..... 91
Objetivo ..... 91
Antecedentes ..... 91
Descripción del problema o contexto de la participación profesional ..... 91
Metodología utilizada y participación profesional ..... 91
Resultados ..... 128
Conclusiones ..... 129

Conclusiones generales ..... 131
Fuentes de información ..... 133
Glosario ..... 135

# Introducción y objetivos

El presente trabajo describe las actividades profesionales que desarrollé en el área de Ingeniería en Computación, en la empresa MS Consulting, a cuyo director, Ing. Mauricio Gaona Olivo, agradezco las facilidades y el apoyo concedidos.

MS Consulting es una empresa de servicios dividida en dos ramales: una de administración y contaduría y otra de consultoría de sistemas computacionales que desarrolla e implementa sitios web y/o sistemas sobre Internet y además distribuye productos de cómputo para instituciones privadas. Su equipo de trabajo es multidisciplinario y aunque mi puesto, en lo particular, se ubicaba en el departamento de sitios web e implementación de redes, laboré conjuntamente con las demás áreas desarrollando sistemas.

El texto que presento a continuación refleja mi participación, a lo largo de un año, en diferentes proyectos enfocados a la realización de sitios web para varias PYMES (pequeñas y medianas empresas) y algunas otras entidades que forman parte de la clientela de MS Consulting. Mi labor y mi responsabilidad fue generar un portafolio de negocios para la recopilación de datos, piezas o componentes y planes de seguimiento orientados a llevar un control y mejorar la calidad en tiempos de producción y servicios.

Un portafolio de negocios en una empresa se define como “una herramienta para la alineación estratégica de negocios”, es decir, un conjunto de medios o recursos empleados para sobresalir o ganar ventajas por encima de la competencia –ya sea en el momento o a mediano o largo plazo— y que adicionalmente permita monitorear o supervisar el grado de coherencia que guarda la empresa con sus objetivos estratégicos de origen. Dicho de otra forma, un portafolio resulta ser una agrupación de servicios, productos, datos, programas, etcétera, que se generaron en un momento dado y que se han “guardado” en el portafolio para ser recuperados siempre que se le requiera a fin de optimizar el rendimiento de una PYME.[1]

Por su parte, las PYMES son pequeñas o medianas empresas que desempeñan una actividad de comercio, producción o servicios, cuyo número de empleados es reducido

[1] Llanos Cuenca, Andrés Boza, Ángel Ortiz, “El portafolio de aplicaciones como una herramienta para la alineación estratégica de negocio y SI/TI. Un enfoque de ingeniería empresarial” en “4th International Conference on Industrial Engineering and Industrial Management” [s.f.]. Recuperado el 28 octubre 2016.
http://adingor.es/congresos/web/uploads/cio/cio2010/INFORMATION_SYSTEMS_AND_ICT_INFORMATION_COMMUNICATIONS_AND_TECHNOLOGY/680-689.pdf

y sus montos de facturación no logran ser demasiado elevados. Su tipificación y clasificación en México se muestra en la tabla (ver tabla 0.1):

**Estratificación de las empresas**

| Tamaño | Sector | Rango de número de trabajadores | Rango de monto de ventas anuales (mdp) | Tope máximo combinado |
|---|---|---|---|---|
| Micro | Todos | Hasta 10 | Hasta $4 | 4.6 |
| Pequeña | Comercio | De 10 a 30 | De $4.01 hasta $100 | 93 |
| | Industria y Servicios | De 11 50 | | 95 |
| Mediana | Comercio | De 31 a 100 | De $100.01 hasta $250 | 235 |
| | Servicios | De 51 a 100 | | |
| | Industrias | De 51 a 250 | De 100.01 | 250 |

Tope Máximo combinado se calcula de la suma del 10% de rango de trabajadores y el 90% de rango de monto de ventas anuales. [ TMC = (RT x 0.1) + (RVA x 0.9) ]. Fuente: INEGI, *Boletín de Prensa*, núm. 285/16 en: http://www.inegi.org.mx/saladeprensa/boletines/2016/especiales/especiales2016_07_02.pdf

0.1 Tabla estratificación de las empresas

Actualmente, en diversos países, las PYMES representan la principal fuente de sustento económico, incluso por arriba del 90% como en España y México. En nuestro país se contabilizan aproximadamente 4,015,000 empresas, de las cuales el 99.8% son PYMES que generan el 52% del Producto Interno Bruto (PIB) y conforman un 72% de la tasa de empleo del país, según datos del INEGI.[2]

Al igual que el director de una gran empresa, el propietario o administrador de una PYME tiene que encarar diversos retos en varios frentes, como, por ejemplo, la capacitación de sus empleados, el control de su producción y servicios, el mantenimiento de sus equipos e inmuebles, la administración de sus recursos, el control de ingresos y egresos, la planificación y las estrategias de la empresa, la seguridad, el transporte, la contabilidad, etcétera.

Independientemente del giro del negocio, el responsable debe valerse de herramientas que lo ayuden a eficientar, optimizar y controlar las diferentes áreas de la empresa para mantener buenos niveles de flujo en la producción, comercialización o prestación de servicios, sin incrementar sus costos. En este sentido, y ya desde hace algunas décadas, los recursos informáticos han sido la solución ideal; por este motivo, la participación en el mundo empresarial de profesionales en Ingeniería de la Computación

[2] Se difunden estadísticas detalladas sobre las micro, pequeñas y medianas empresas del país" , en *Boletín de Prensa*, núm. 285/16. Recuperado el 16 de octubre de 2016, en: http://www.inegi.org.mx/saladeprensa/boletines/2016/especiales/especiales2016_07_02.pdf

resulta crucial. Sin expertos capaces de desarrollar, instrumentar y dar soporte a sistemas que optimicen y aceleren los flujos de intercambios comerciales, de servicios y de otro tipo de actividades vinculadas al área comercial, no sería posible incrementar la productividad ni permitir el crecimiento de la economía global.

Hoy en día una de las herramientas más socorridas por empresas grandes y pequeñas son los equipos electrónicos como servidores de datos, web, respaldos, CCTV, equipos personales o terminales y programas especializados en la contaduría, control, administración, comunicación, protocolos, seguridad, etcétera que, cubren las distintas necesidades empresariales. Esta dependencia de la computación y las redes de datos hace indispensable no sólo contar con la infraestructura requerida, sino también disponer de los servicios de un profesional en el área capaz de planear, dirigir y prever soluciones a los problemas empresariales vinculados a los sistemas de cómputo.

Así, el objetivo del presente trabajo es presentar –a manera de informe— la problemática real de varias PYMES en materia de los referidos sistemas y las soluciones que he dado a sus necesidades y dilemas a partir de mi formación profesional.

Vale la pena señalar que no me habría sido posible resolver este variado y amplio conjunto de problemas si a lo largo de la carrera de Ingeniería en computación no me hubiera familiarizado con la naturaleza y el funcionamiento de los servidores, los lenguajes y paradigmas de programación, la operación e intercomunicación de la red global y las locales, las formas de almacenamiento de la información, los riesgos y los mecanismos de seguridad. Muy útiles me fueron también los conocimientos de administración de proyectos, que pude probar en la práctica para resolver situaciones reales. Sin todas estas herramientas teórico-prácticas adquiridas en la Universidad, sencillamente no habría podido desempeñar mi trabajo a cabalidad.

## Marco teórico

Siendo yo el responsable del diseño e implementación de los proyectos que aquí presento, gran parte de mi trabajo fue definir planes útiles para llevarlos a ejecución y en virtud de la gran diversidad de necesidades y especificaciones que me planteó cada cliente hay una natural variación en todos. De ahí que, en el caso de mi labor, no me haya podido servir de un método único o de teorías de validez universal para la resolución de problemas tan heterogéneos.

Sin embargo, sí empleé herramientas teóricas más o menos estandarizadas que fueron útiles para el desarrollo de cada proyecto en particular. Por ejemplo, el "Triángulo de triple restricción",[3] un planteamiento acuñado por expertos en el desarrollo de proyectos, reconocido mundialmente y cuyo cimiento es el modelo de Datos, Información, Conocimiento y Sabiduría (DICS). Según el referido "Triángulo", la calidad de cualquier producto y servicio se condiciona o supedita a tres conceptos básicos: tiempo, costo y alcance. Siempre será posible limitar o disminuir uno o dos de dichos elementos pero jamás los tres si se desean mantener los niveles de calidad. Así, podríamos aumentar los costos y reducir el alcance para que se pueda completar el ciclo de vida del proyecto en menor tiempo, o se puede aumentar el tiempo y reducir el alcance con miras a reducir costos, etcétera.

Este enfoque fue bastante provechoso al momento de prestar asesoría a los clientes para la planeación y desarrollo de las páginas web, e igualmente, pude desarrollar planes más precisos en tiempo y calidad antes de presentar mis propuestas.

[3] Project Management Institute, *Guía de los fundamentos Para la Dirección de Proyectos (Guía del PMBOK*) 5ª ed., Pennsylvania, PMI Publications, 2013.

## Metodología

La metodología empleada para los proyectos de que hablaré más adelante, se basa en un modelo de proceso de cascada o ciclo de vida clásico, pues en estos proyectos los objetivos son relativamente sencillos de entender y, al final, todo se reduce a obtener un producto con ciertas características definidas por el cliente a partir de objetivos predefinidos. En consecuencia, este modelo se ajusta bien para ambas características; además, me brinda la ventaja de permitirme optar entre una relación secuencial o una de superposición. Esto también me ofrece la posibilidad de acelerar ligeramente la vida del proyecto, en función de su progreso y sin perder el orden del avance. Mediante el uso de este modelo y aprovechando igualmente los consejos de mis compañeros de trabajo y mi propia experiencia, fui modificándolo con algunas adiciones y alteraciones, según hicieran falta, ya para facilitar el avance, ya para corregir errores, etcétera.

En lo personal, me resultó muy útil trabajar con una combinación entre dos tipos de ciclos de vida. El primero es el predictivo, en el que el flujo de trabajo es lineal, facilita el orden de ejecución de cada fase y permite enfocarse en cada una de ellas. El segundo modelo que empleé como adición al primero, fue el adaptativo. Me decidí por él en razón de que es natural que durante la vida del proyecto haya necesidad de generar cambios a lo originalmente planificado, en buena medida por economizar gastos y tiempo, pero también por evitar repetir pasos y reajustar otras partes que resultaran afectadas por las modificaciones, sobre todo si ya fueron desarrolladas o si el proyecto está en fase más avanzada. Combinar las iteraciones de métodos ágiles del modelo adaptativo al modelo predictivo me permitió generar un flujo de trabajo formado de varias fases con iteraciones en cada una, lo que garantizó reducir problemas potenciales y evitar cambios innecesarios antes de proceder a otra fase.

Adicionalmente, dentro de mi propio flujo de proceso me basé en el esquema: inicio, planificación, ejecución, monitoreo, control y proceso de cierre. En cuanto a los flujos de proceso lineal, me orienté también a lo propuesto por Rogers S. Pressman[4] en el sentido de que estos constan de: comunicación, planeación, modelado, construcción y despliegue. El proceso de inicio, comunicación y preparación, lo agrupé en una fase única para el análisis, es decir el establecimiento de la comunicación con el cliente, el intento de generar

[4] Roger S. Pressman, *Ingeniería del software, un enfoque práctico*, 7ª ed., New York, Mc Graw Hill, 2010.

un panorama claro de las necesidades, requisitos, etcétera y el planteamiento de las posibles soluciones. Los procesos de ejecución y construcción, los agrupé en la fase de desarrollo, es decir, la materialización de lo planteado en la fase anterior. A los procesos de monitoreo y despliegue los agrupé en la fase de laboratorio y pruebas, momento de generación de un prototipo que se somete a pruebas para producir retroalimentación y realizar depuración y, de ser necesario, incluso regresar a la fase de desarrollo.

Finalmente, respeté el proceso de cierre como una fase independiente, pues son las operaciones preliminares para concluir el proyecto: una segunda prueba y la liberación del producto para su consumo.

Todos los proyectos aquí descritos se sometieron y acataron estos lineamientos teóricos.

## MS Consulting y desarrollo de actividades personales en ella

MS Consulting es una compañía que consta de dos divisiones: una orientada a la administración y a los servicios contables y una segunda dedicada a la consultoría de sistemas computacionales; esta última abarca soporte técnico, redes, desarrollo web, venta e importación de equipo de cómputo. La empresa tiene dos sedes: una en la Ciudad de México y otra en la de Monterrey, Nuevo León.

Creada su división de cómputo en 2006 y sólo dedicada a la reparación de equipos, paulatinamente extendió su actividad a otros ramos. Empezó con la distribución de dominios y hosting en México para una compañía norteamericana; más adelante comenzó la importación y venta de equipos y dispositivos de Apple (iPod, iPhone, iPad) en la Ciudad de México; posteriormente abrió una oficina en Monterrey para ampliar su mercado y satisfacer la creciente demanda.

Me incorporé formalmente a MS Consulting en 2009. Mi tarea era brindar soporte técnico a una clientela específica en la Ciudad de México. Al paso del tiempo, con el aprendizaje y desarrollo de nuevas habilidades, se me encomendaron otras responsabilidades, como la instalación y configuración de redes para medianas empresas, el desarrollo, diseño y depuración de sitios web, así como desarrollo de aplicaciones “a la medida” de cada cliente. A la fecha estoy a cargo de la atención de una empresa grande, tres de tamaño medio y doce negocios pequeños en la Ciudad de México.

En las páginas subsecuentes, daré cuenta del tipo de actividad que he realizado para algunas de ellas y también para otras entidades y agrupaciones, haciendo especial hincapié en los problemas técnicos o empresariales y en sus soluciones.

# I. Desarrollo de aplicación web “Simulador conductual”

## Objetivo

Contar con una herramienta que, mediante una aplicación web, permita la realización de una entrevista conductual de manera remota y simulada a través de internet en apoyo a la formación de los estudiantes de la Facultad de Psicología de la UNAM. El promotor del proyecto es LEVyC (Laboratorio de enseñanza virtual y ciberpsicología), y pretende que los alumnos puedan desarrollar y ejercitar sus habilidades durante la realización de entrevistas a pacientes.

## Antecedentes

El laboratorio de LEVyC es una división de estudios de posgrado e investigación que pertenece a la Facultad de Psicología de la Universidad Nacional Autónoma de México. Sus responsables pretenden servirse de la tecnología para ofrecer apoyo, soluciones y recursos de trabajo a pacientes y estudiantes de esta facultad.

Uno de sus proyectos centrales era la creación de una herramienta para que los alumnos de psicología pudieran desarrollar habilidades en las entrevistas conductuales, garantizándoles un espacio seguro y una práctica en simulador, a fin de que al momento de realizar las entrevistas reales con los pacientes, pudieran desenvolverse mejor y obtener los resultados esperados. Cabe señalar que el propósito de una entrevista conductual es indagar el comportamiento del paciente en distintas situaciones y en interacción con sus relaciones. A partir de estos datos se realiza un análisis que permite explorar la conducta-problema, de lo general a lo particular.

Para diseñar y hacer accesible el simulador a todos los estudiantes de psicología, se le planeó como una aplicación bajo una plataforma web, relacionada con la entrevista conductual que fuese instalada en el sitio web del departamento. Para tales efectos, se requirió disponer de un servidor web con una buena seguridad y disponibilidad en línea, a fin de que los usuarios del proyecto pudieran recurrir a él en todo momento; es importante señalar que este proyecto ya había sido iniciado por otro equipo de ingenieros, aunque no lo concluyeron.

## Descripción del problema o contexto de la participación profesional

El sistema consistió en un sitio web al que los estudiantes tienen acceso; aquí pueden efectuar entrevistas con una secuencia definida a través de videos, que incluyen preguntas de opción múltiple, formularios, documentos y retroalimentación. Todo esto pensado para que los alumnos practiquen y refuercen el aprendizaje en temas de entrevista conductual.

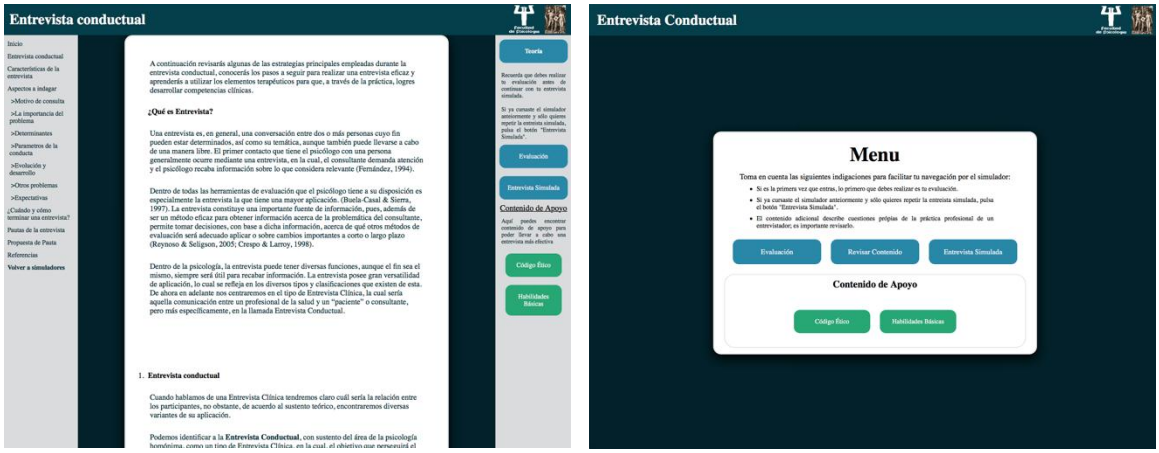
El desarrollo de esta plataforma suponía cubrir ciertas características durante la planeación del proyecto, tales como la compatibilidad de navegadores, la seguridad, la condición intuitiva y la estabilidad, a fin de hacer del simulador un programa sencillo y de manejo amigable, al que los estudiantes y docentes de la Facultad de Psicología pudieran recurrir regularmente.

Como se señaló antes, el proyecto lo avanzaron otros desarrolladores pero no lo concluyeron. Si bien, los videos ya habían sido generados y subidos a YouTube, algunas páginas del sitio web ya tenían insertados los videos, utilizando HTML 5, CSS, JS (JavaScript) y el API de YouTube. Yo retomé el trabajo inconcluso y se me encargó avanzarlo y terminar el proyecto.

## Metodología utilizada y participación profesional

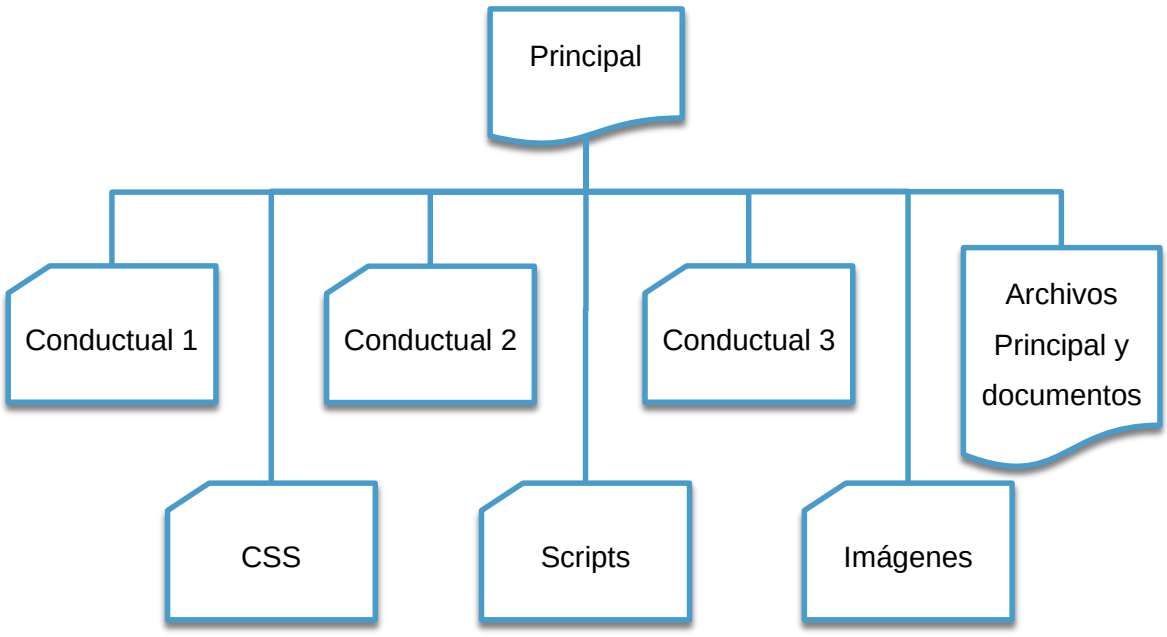
Para lograr un proyecto intuitivo, simple y eficaz, fue necesario planear y crear un flujo de trabajo determinado para cada caso. Cada flujo de trabajo dispone de una cierta secuencia en la que el usuario debe ir cubriendo distintos puntos; por otro lado en algunos de estos puntos hay opciones adicionales al flujo principal para recurrir a ciertos elementos que pudieran resultar útiles en cada etapa. De este modo se evita incurrir en confusiones durante el proceso de aprendizaje.

El diseño también era una parte fundamental del proyecto para hacer de esta plataforma una pieza de software intuitiva y agradable a la vista, en consecuencia, optamos por ventanas simples, colores oscuros, diseño responsivo y menús con botones de preferencia en los laterales para dar orden y estructurar a los elementos de la plataforma (ver figura 1.1).




1.1 Captura de pantalla del diseño visual

Lo primero que hice fue retomar el proyecto que habían realizado los otros desarrolladores; inicialmente diseñé la plataforma por módulos con el objeto de facilitar y darle estructura, de forma tal que fuera más organizada y se pudiera actualizar fácilmente –cuando fuera necesario– con nuevos contenidos o cambios en el programa. Para eso organicé los materiales y documentos que fueron utilizados para el sistema como se muestra en el diagrama de abajo figura 1.2.




1.2 Mapa de y estructura de la plataforma

En el directorio principal se contienen los archivos principales del sistema, en donde está la sección introductoria y el menú, las carpetas que contienen los scripts, hojas de

estilo, e imágenes utilizadas y documentos de texto. En la carpeta principal los archivos del directorio ofrecen la posibilidad de seleccionar de entre alguno de los tres modelos diferentes de Sistema conductual (Conductual 1, Conductual 2 y Conductual 3). A fin de averiguar cuál quedaría se hicieron pruebas con una pequeña muestra de alumnos de la facultad, y se calificó por medio de retroalimentación. Así, la estructura empezó por estable tres directorios separados dentro del directorio principal, que se corresponden con los tres modelos del simulador, utilizando los mismos archivos de las hojas de estilo, scripts y documentos que están en el principal. Tal cosa evita la redundancia y repetición de datos, además de dar una breve introducción de la plataforma por medio de un video introductorio que explica en qué consiste y cómo se llevara a cabo el ejercicio.

Estas carpetas contienen el API de YouTube con los scripts para el control de los videos que se utilizan en el sistema; yo mismo generé algunos de ellos para el control de los elementos dentro del sistema y la navegación. En la carpeta de CSS están contenidas todas las hojas de estilo para darle presentación y un diseño intuitivo a las páginas del simulador.

Mi primer paso fue corregir los errores en los videos para que se cargaran correctamente las preguntas de opción múltiple en el momento que se requirieran y conforme avanzara el video, controlando así el estado del video y deshabilitando las opciones de los controles de video de YouTube. Para eso restructuré el código en los archivos, agregué funciones propias para un mejor control, generé un archivo que cargaba el API de YouTube, controlando y redimensionando así el tamaño de video según el tamaño de la ventana para hacerla responsiva, generar las capas de overlay para las preguntas, los controles del video, datos de depuración, etcétera. Todas o parte de estas funciones eran cargadas en cada video directamente. Un ejemplo de este código se puede apreciar en las figuras 1.3, 1.4, 1.5 y 1.6

```
videos.innerHTML+='\<div id="player" style="position:relative;"></div>\
 <div style="width:100px; height:50px; position:absolute; bottom:0px; right:0px; z-index:2;"></div>\';
var tag = document.createElement('script');
tag.src = "https://www.youtube.com/iframe_api";
var firstScriptTag = document.getElementsByTagName('script')[0];
firstScriptTag.parentNode.insertBefore(tag, firstScriptTag);
var player;
function onYouTubeIframeAPIReady() {
 player = new YT.Player('player', {
  height: height,
  width: width,
  videoId: url,
  playerVars: {
   'modestbranding': 1,
   'controls': 0,
   …
  },
  events: {
   'onReady': onPlayerReady,
   'onStateChange': onPlayerStateChange,
  }
 });
}
var formsInside = 0;
function threeNodes(Nodo){
 if(Nodo.children.length == 0) return false;
 for(i=0;Nodo.children.length>i;i++) if(Nodo.children[i].tagName=="FORM"){
   formsInside++;
   return true;
  }
  threeNodes(Nodo.children[i]);
}
function onPlayerReady(event) {event.target.playVideo();}
```

1.3 API de YouTube para la carga e inserción de videos de YouTube a la plataforma

```
function overlay(){
 var forma = document.getElementsByClassName('caja');
 var lista = forma.length-1;
 while(0<=lista){
  forma[lista].style.width = width+'px';
  forma[lista].style.height = height+'px';
  if(forma[lista].hasAttribute('left'))
   forma[lista].style.left = '0px';
  else
   forma[lista].style.right = '0px';
  lista--;
 }
}

function redimencionar(){
 videos = document.getElementById('videos');
 windowPlayer = document.getElementById('player');
 windowHeight = window.innerHeight;
 height = windowHeight-headerHeight;

 videos.style.height = height+'px';
 windowPlayer.setAttribute("height",height);

 var windowWidth = window.innerWidth;
 if(document.getElementsByTagName('aside').length>0){
  var asideWidth = document.getElementsByTagName('aside')[0].clientWidth;
  videos.style.width = windowWidth-asideWidth+'px';
 }
 else{
  videos.style.width = windowWidth+'px';
 }
 width = document.getElementById('videos').clientWidth;
 windowPlayer.setAttribute("width",width);

 if(document.getElementsByClassName('caja').length>0){
  overlay();
 }
}
```

1.4 Código JS para control de dimensiones del video y control de la capa de preguntas de selección múltiple

Para el caso de los videos que incluyen las preguntas de opción múltiple, generé un par de scripts adicionales que llevan: control del número de aciertos, parar y reanudar el video en el tiempo indicado, control de los elementos que debe aparecer y desaparecer, cargar la lista de preguntas y respuestas para comparar con lo que el usuario ingrese, entre otras.

```
var
t=[64,78,92,113,114,146,162,166,168,188,214,243,259,260,285,313,333,339,340,356,381,400,416,418,435,437,454];

var answer = ['¿Cuál es el motivo por el que acude?',
        'Buenas tardes, ¿Cuál es el motivo por el que acude?',
        '¿Podría contarme cuál es el motivo por el que acude?',
        'Dígame ¿qué problema le aqueja actualmente?',
        '¿Esta dificultad le ocurre con todas las personas?',
        '¿Podría ser más específico respecto a esta dificultad para comunicarse?',
        '¿Me podrías dar más detalles acerca de esto que me comenta?',
        …
          ];

var score = '<h3>¡Enhorabuena!</h3> … ;

function loadAnswer(number){ answer[number]; }
```

1.5 Código JS para asignación de preguntas en la capa de selección múltiple

```
 function onPlayerStateChange(event) {
  if (event.data == YT.PlayerState.PLAYING) {setInterval(stopVideo,100); }
  if (event.data == YT.PlayerState.ENDED ) {ShowScore(score);}
}

…

function stopVideo() {
  var a = Math.round(player.getCurrentTime());
  if(a==t[count]){
    player.pauseVideo();
    idoverlay = 'overlay'+(count+1);
    document.getElementById(idoverlay).style.display='block';
    count++;
  }
}

function verificar(noverlay, np, nrespok) { … }

function continuar(resp, nov){
 if(resp==1) { … }
 if(resp==0){ … }
}

function avanzar(nov) { … }

function ShowScore(texto) { … }
```

1.6 Código JS para el control de los videos de YouTube insertados

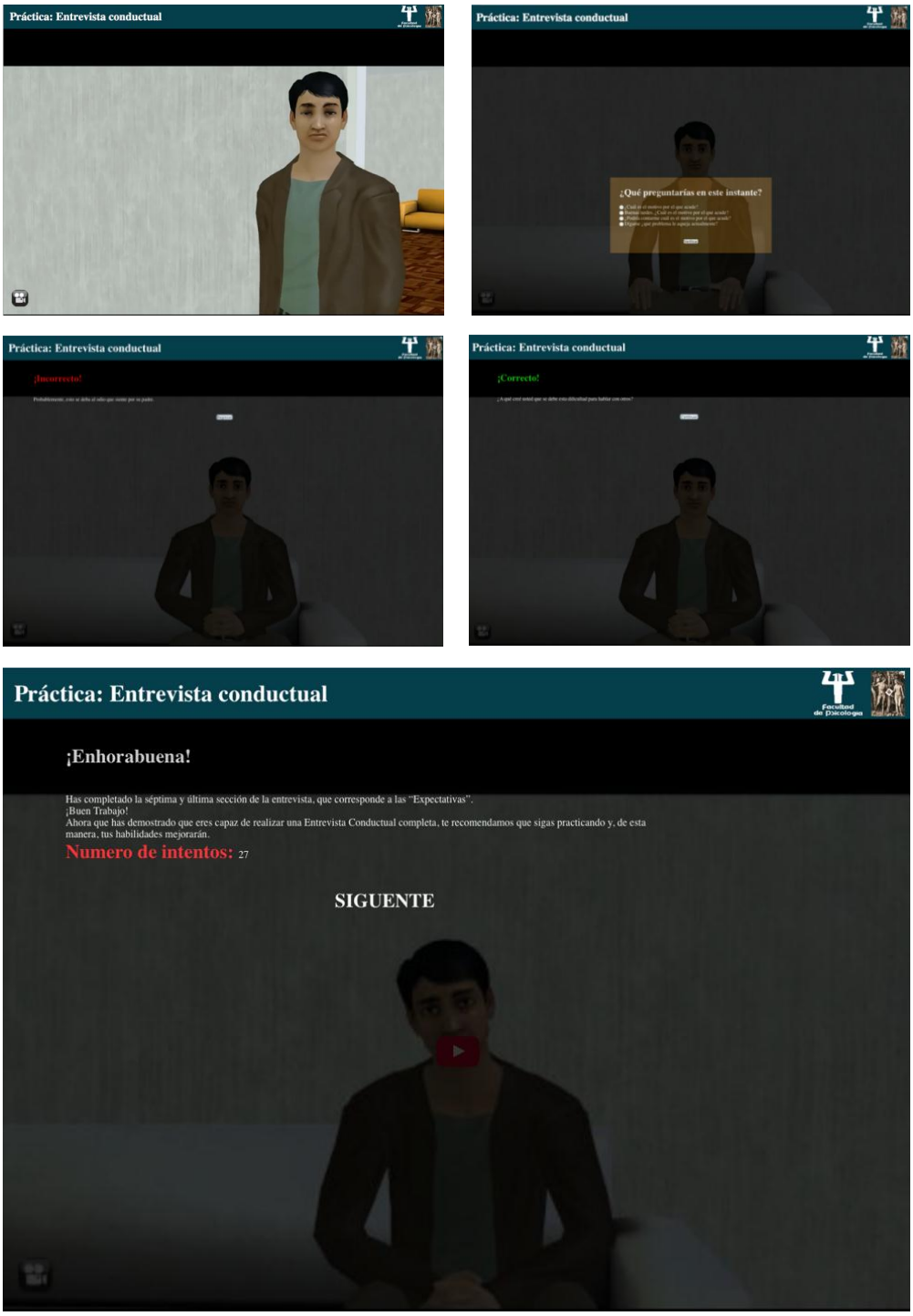
En mi diseño solamente fue necesario llamar tres scripts para poder agregar un video con las preguntas, aquí la función de redimensionar asegura que el video cubra toda la ventana al finalizar de la carga de la página y también, que al mismo tiempo se redimensione la ventana del navegador. El elemento DIV con el atributo ID “Videos” cargará los elementos de YouTube con el atributo “dirURL”, se deberá poner el código que identifica el video que se quiere cargar, eso puede encontrarse en el URL que genera en la página de YouTube cuando se accede al video.

El cuadro con el ID de “Validar”, sirvió para desplegar el resultado de su respuesta, y es necesario agregar este elemento ya que en él se escribe el mensaje deseado que saldrá, ya como acierto, ya como falla. Finalmente, todos los elementos que lleven el ID “overlay#” (donde el “#” es la numeración) son las capas que cargan las preguntas de manera automática con el orden de la numeración que lleven dichas capas, sin importar que estén dispuestas aleatoriamente en el orden de la estructura.

Para que cada capa que contiene la pregunta de opción múltiple, reconozca el número de respuestas que le corresponden, cada respuesta está asociada a un elemento de entrada (INPUT) con ID que va desde el uno hasta el cuatro. Así, una función con el nombre de “verificar” puede identificar la respuesta correcta para cada pregunta; esta función recibe tres parámetros con un dato de tipo entero, el primer parámetro recibe un número que es el mismo valor que el que le corresponde a la capa en la ID de “overlay#”; esto se dispuso así, a fin de avisar a la función qué pregunta se está procesando. El segundo parámetro es el número que define a qué grupo de respuestas pertenece la pregunta y el tercer parámetro es un número que corresponde a la respuesta correcta dentro de ese grupo. A continuación se muestra un segmento de dicho código y capturas de pantalla en la figura 1.7 y 1.8.

```
…
<script src="http://code.jquery.com/jquery-latest.min.js" type="text/javascript"></script>
<script type="text/javascript" src="cargaRespuestas.js"></script>
</head>
<body onResize="redimencionar()">
<div id="header">
 <h1>Práctica: Entrevista conductual</h1>
</div>

<div id="cuerpo">
 <div id="videos" right dirURL="Kpx5xJhdiSk"></div>
 <div id="validar" class="caja"></div>
 <div id="avanzar"></div>
 <div id="overlay1" class="caja">
  <div class="formulario">
   <h3>¿Qué preguntarías en este instante?</h3>
   <form>
    <input id="uno" type="radio" name="p1" value="1" required>
    <label for="uno"> ¿Cuál es el motivo por el que acude?</label>
    <br/>
    <input id="dos" type="radio" name="p1" value="2" required>
    <label for="dos"> Buenas tardes, ¿Cuál es el motivo por el que acude?</label>
    <br/>
    <input id="tres" type="radio" name="p1" value="3" required>
    <label for="tres"> ¿Podría contarme cuál es el motivo por el que acude?</label>
    <br/>
    <input id="cuatro" type="radio" name="p1" value="4" required>
    <label for="cuatro"> Dígame ¿qué problema le aqueja actualmente?</label>
    <br/>
    <button id="btn" type="button" onclick="verificar(1,1,1)">Verificar</button>
   </form>
  </div>
 </div>
 <div id="overlay2" class="caja" >
  <div class="formulario">
   <h3>¿Qué preguntarías en este instante?</h3>
   <form>
    <label><input id="uno" type="radio" name="p2" value="1" required>
     ¿Esta dificultad le ocurre con todas las personas?</label>
    <br/>
    <input id="dos" type="radio" name="p2" value="2" required>
    <label for="dos"> ¿Podría ser más específico respecto a esta dificultad para comunicarse?</label>
    <br/>
    <input id="tres" type="radio" name="p2" value="3" required>
    <label for="tres"> ¿Me podrías dar más detalles acerca de esto que me comenta?</label>
    <br/>
    <input id="cuatro" type="radio" name="p2" value="4" required>
    <label for="cuatro"> ¿Qué es lo que siente al hablar con personas?</label>
    <br/>
    <button id="btn" type="button" onclick="verificar(2,2,4)">Verificar</button>
   </form>
  </div>
 </div>
  …
</div>
<script type="text/javascript" src="../scripts/YTscrip.js"></script>
<script> overlay(); </script>
<script type="text/javascript" src="simulador.js"></script>
</body>
</html>
```

1.7 Código HTML del archivo de “Práctica”




1.8 Capturas de pantalla de la página "Practicas"

Ya he reiterado que en este proyecto colaboré con personas de la división de LEVyC de Psicología, lo que exigió una interlocución, comunicación clara y entendimiento

interdisciplinario, a fin de generar satisfactoriamente un producto de ingeniería que cumpliera con las especificaciones y requisitos del área de psicología.

## Resultados

Los resultados del trabajo fueron mejores de lo que se esperaba: la interfaz gráfica logró ser intuitiva y de fácil manejo, se descartó el problema de compatibilidad de los navegadores recientes y más comunes y el proyecto se logró completar a tiempo. Como único desarrollador del proyecto, logré exitosamente generar un plan de proyecto, aterrizarlo y ejecutarlo, con resultados finales que mostraron buen juicio y adecuada toma de decisiones a la hora de enfrentar los retos y problemáticas. Tuve la disposición y la responsabilidad de cubrir cada requerimiento escrupulosamente y realicé las actividades necesarias para completar el objetivo.

Como responsable dentro de este proyecto, mi aportación fue global, integral y completa, debido a que desde mi llegada, se me encomendó el análisis, rediseño y depuración del proyecto. Hice una investigación de librerías externas buscando que fueran útiles, flexibles, seguras y modernas para su uso. Igualmente, en el desarrollo de la plataforma hice uso de los conocimientos adquiridos a lo largo de la carrera y de los cursos adicionales que tomé en la Facultad de Ingeniería para la elaboración de piezas de ingeniería; recurrí a lenguajes de programación, arquitectura cliente y servidor, lenguaje de etiquetas y estilo. Finalmente, la coordinación que tuve como desarrollador con los integrantes de la división de LEVyC para generar los temas de la entrevista conductual dentro del proyecto fue óptima, organizada y clara. Este proyecto tuvo un periodo de realización aproximadamente de 3 semanas a partir de agosto del 2015.

## Conclusiones

La plataforma se terminó bajo los requisitos solicitados y fue expuesta ante una muestra de alumnos de la Facultad de Psicología, a fin de probarla y obtener de los estudiantes una retroalimentación acerca de la interactividad, la intuitividad, la predictibilidad, la vista, el diseño y la utilidad de dicha plataforma. Todo esto con miras a recabar observaciones para depurar errores o agujeros en el sistema y detectar la facilidad del aprendizaje de los alumnos que utilizaron esta herramienta con los tres modelos de entrevista conductual.

El proyecto se completó de forma exitosa y quedó listo para que en un fututo se pueda empezar un nuevo proyecto o ampliar el ya realizado aunque a la fecha no he vuelto a saber acerca de cómo se ha utilizado el producto.

# II. Desarrollo de sitio web “DePapel”

## Objetivo

Analizar, planear, diseñar y desarrollar un sitio web para publicidad y venta de artículos hechos de papel. Generar un espacio intuitivo, fácil de usar y agradable para los visitantes con la intención de que la empresa se dé a conocer y generé contactos de manera fácil y automática.

## Antecedentes

DePapel es un negocio enfocado a la comercialización de artículos confeccionados de papel. Hasta que establecí contacto con ella, la dueña de este negocio no disponía de ningún medio publicitario electrónico para promover sus productos o servicios y tampoco de un vehículo formal de comunicación con sus potenciales clientes.

## Descripción del problema o contexto de la participación profesional

El proyecto empezó de cero, pues no había ningún plan o idea prestablecidos; la dueña de la empresa sólo tenía una vaga noción sobre cómo debían lucir las imágenes de sus productos. Sus requisitos eran que el diseño fuera agradable y el sistema intuitivo, amén de que incluyera un medio de comunicación; asimismo requería una cuenta de correo personalizado para la empresa y, finalmente, esperaba que todo esto se hiciera al menor costo posible.

Así que, luego de discutir con ella sus necesidades y de tener una idea mejor de los requerimientos, desde el principio, se me encargó investigar, analizar y desarrollar todo el proyecto y quedé como responsable de la dirección, administración de tiempo de vida, y financiación del mismo.

## Metodología utilizada y participación profesional

Como el negocio carecía de antecedentes de proyecto, lo inicié desde los cimientos. Generé un plan de seguimiento, cuya primera fase consistió en el análisis de los requisitos a fin de dar con soluciones óptimas y con un diseño que proporcionara al visitante una experiencia de navegación agradable, intuitiva y sencilla que, conjuntamente, le permitiera encontrar de manera rápida y fácil los datos más generales acerca de los productos y de la empresa.

Los productos de papel no sólo los consumen los adultos, sino que también tienen un gran atractivo para el segmento infantil. Así, determiné como necesidad primordial generar un espacio donde los clientes de todas edades pudieran ingresar indistintamente, disfrutar del contenido y navegar de manera fácil e intuitiva. Sin embargo, debido a que son sólo los adultos los que tienden a indagar cuestiones relativas a costos y entregas de paquetes de productos, decidí que tales aspectos no debían ser añadidos al sitio web en sí, que sólo incluiría imágenes de los productos y sus diseños. Y, en cambio, para facilitar la comunicación de la empresaria con los clientes, opté por añadir una sección de contacto, con un formulario para registrar nombre, teléfono, correo electrónico y un recuadro de texto para comentarios y pedidos. De esta manera la empresaria podría establecer contacto con la clientela real y comunicarse directamente con ella, por tanto, el sitio web quedó únicamente como una atractiva sección publicitaria.

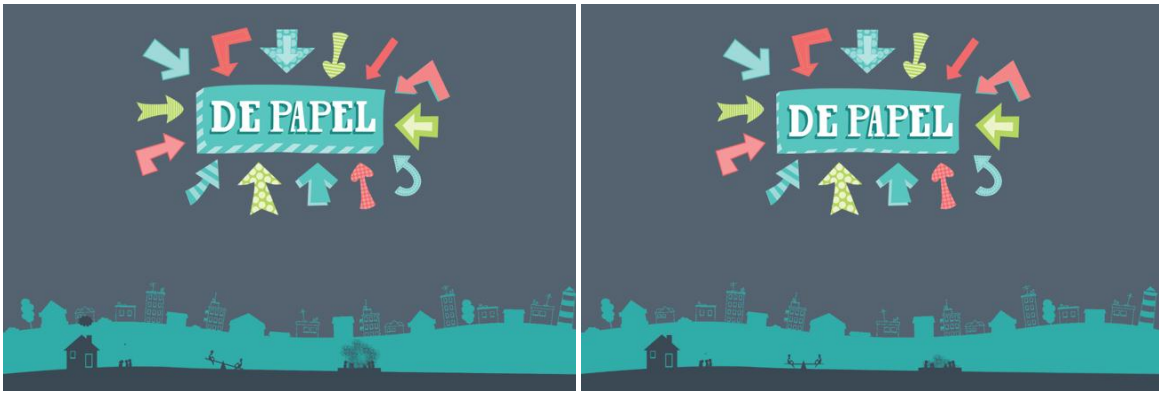
Considerando que el diseño se dirigía tanto a un público adulto como a uno infantil, se asignó a este espacio información elemental, con muchas imágenes iconos y pocos hipervínculos. La idea, insisto, era que esto sirviera sólo como un contacto inicial de la empresa con los clientes potenciales, dejando que los niños disfruten al encontrar la página y navegar por ella para ver los materiales que les ofrece específicamente la empresa, al tiempo que gozan de un entorno visualmente agradable y divertido. Para los adultos –independientemente de que tuvieran o no experiencia en el uso de computadoras— se pensó en un dispositivo que les permitiera encontrar de manera sencilla, a través de unos cuantos clics aquello que les interesara adquirir y ponerse en contacto fácilmente con el proveedor.

La segunda fase consistió en el desarrollo y generación de código con el uso de lenguaje de etiquetas HTML, hoja de estilo CSS, lenguaje de programación PHP e igualmente fue necesario utilizar lenguaje de programación JS. Desarrollé el sitio web con un diseño de interfaz que se atuvo a los colores específicos de la empresa y a los diseños, imágenes, textos iconos, figuras, etcétera, sugeridos por el cliente. En cuestiones de seguridad, solo revisé el código de programación PHP para el envío de formularios, a fin de evitar vulnerabilidades

Para el desarrollo utilicé lenguaje HTML a fin de generar los archivos como el índex y los archivos de error SHTML, como el 400, 401, 500, etcétera a fin de que tuviera un diseño acorde con el sitio en general. Para darle calidad visual generé un único archivo de hoja de estilo al que hacía referencia en cada archivo de HTML, evitando de esta forma redundancia de código y que fuera general para todas las páginas. Solamente en los casos

donde cierto código de estilo fuera aplicado sólo a una página en particular se agregó sobre el archivo HTML una etiqueta “style” para dar un poco de orden y evitar posibles errores al ejecutarse en lugares no deseados por la similitud de las condiciones.

Generé la primera página del sitio web como una portada de presentación, esta página está compuesta por un botón elaborado con imágenes y dotado de una animación por medio de propiedades de CSS que se pueden visualizar al colocar el cursor sobre él. Después, para el fondo utilicé un color gris y varias imágenes GIF sobrepuestas como capas a diferentes niveles para construir un fondo con movimiento (ver figura 2.1 y 2.2).




2.1 Captura de pantalla de la página de inicio

```
…
<style type="text/css">
 body{background:#556371;}

 #framework a{
  position:absolute;
  top:180px;
  left:285px;
  width:435px;
  height:160px;
  background:url('img/index.png') 0 0;
 }
 #framework img{width:800px;}
 #framework a:hover{background:url('img/index.png') 0 -173px;}
…
<div id="framework" align="center">
 <a href="que_es.html"></a>
 <img src="img/arrows.png" alt="arrows"/>
</div>
…
```

2.2 Código HTML del archivo Index

La primera página titulada “¿Qué es?”, habla del carácter de la empresa y su giro específico. Esta página, como el resto, utilicé un elemento “DIV” que conserva centrado el contenido para mantener un orden a la hora de redimensionar la ventana del navegador del

cliente, y utilicé los mismos elementos que en la página pasada para las imágenes animadas de fondo y agregué otras adicionales para adornar.

El cliente pidió específicamente que el contenido se adornara, simulando estar dentro de un bloc de notas en la página web. Con el objetivo de facilitar el trabajo y evitar posibles errores de distribución, opté por dejar el texto –en alta resolución– como parte de la imagen de la libreta. Así, y con la restricción que agregué a la página al redimensionarla, aseguré el correcto despliegue de la información.

En el menú de navegación utilicé un elemento TABLE para la distribución del logotipo y los hipervínculos. En cuanto a la selección de éstos, empleé una etiqueta de lista “UI”, modificando su aspecto mediante el código CSS, le di una apariencia más agradable echando mano de las imágenes que me proporcionó mi cliente y usé la condición del selector “encima” (hover), “activo” (active) y desplazamiento de la imagen dentro de la propiedad de fondo (background) (ver figura 2.3, 2.4 y 2.5). Con ello evité la inclusión de varios archivos que, finalmente, forman parte de un mismo concepto.

2.3 Captura de pantalla de la página “Que es?”

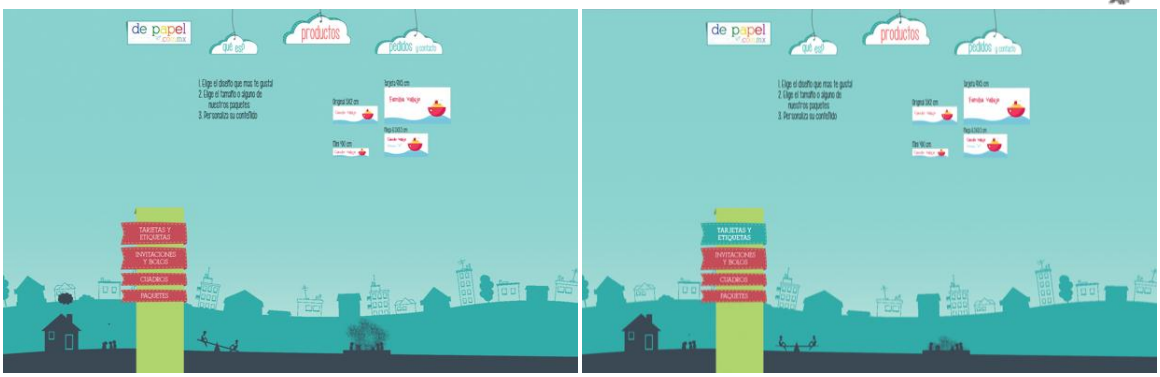
```
…
<link href="stylesheet.css" rel="stylesheet" type="text/css" />
…
<table id="tablenavlist">
 <tr>
  <td width="250"><img src="img/logo.png" alt="logo"/></td>
  <td width="750">
   <ul id="navlist">
    <li id="que"><img src="img/nube1.png" alt="que_es"/></li>
    <li id="prod"><a href="productos.html"></a></li>
    <li id="ped"><a href="contacto.html"></a></li
   </ul>
  </td>
 </tr>
</table>
<img src="img/libreta.png" style="position:absolute; bottom:0px; right:50px; width:700px;" alt="libreta"/>
…
```

2.4 Código HTML del archivo “Que es?”

```
…
#que{
 left:0px;
 width:147px;
}

#prod{
 left:223px;
 width:203px;
}

#ped{
 left:500px;
 width:204px;
}

#que a{background: url('img/sprite1.png') 0px 0px;}
#que a:hover{background: url('img/sprite1.png') -147px 0px;}
#que a:active{background: url('img/nube1.png');}

#prod a{background: url('img/sprite2.png') 0px -30px;}
#prod a:hover{background: url('img/sprite2.png') -204px -30px;}
#prod a:active{background: url('img/nube2.png') 0 -30px;}

#ped a{background: url('img/sprite3.png') 0px 0;}
#ped a:hover{background: url('img/sprite3.png') -204px 0;}
#ped a:active{background: url('img/nube3.png') 0 0;}

#que a,#que a:hover,#que a:active,#prod a,#prod a:hover,#prod a:active,#ped a,#ped a:hover, #ped ↩
a:active{background-repeat:no-repeat;}
```

2.5 Código CSS de los elementos del menú de navegación

Generé una página llamada “Productos” en la que la empresaria puede mostrar algunos de los artículos o diseños a sus clientes potenciales; en ésta agregué un submenú catalogado por tipo de productos, con animación en los hipervínculos para resaltar el sitio donde el cursor estuviera posicionado. Para la generación de este submenú utilicé imágenes puestas sobre elementos DIV con propiedades de CSS y de igual manera con desplazamiento de la imagen para evitar la generación de archivos (ver figura 2.6).



2.6 Capturas de pantalla de la página “Productos”

En esta página se muestran los diferentes productos en grupos separados que al aparecer; alteran el código HTML; para lograr esto utilicé código de JS a fin de manipular un elemento DIV con un ID que es ejecutado en un archivo externo como una función en cada hipervínculo con el atributo “OnClick” que reemplaza con código HTML ya previamente establecido (ver figura 2.7 y 2.8).

```
…
<style type="text/css">
#frame{
 font-family:KGCallMeMaybe;
 position:absolute;
 right:0px;
 top:0px;
 height:100%;
 width:775px;
 border:0px;
 overflow:scroll;
 z-index:0;
}
#tarjeta{
 bottom:400px;
 left:-10px;
 z-index:2;
}
#tarjeta a{background: url('img/tarjetas.png') 0px 0px;}
#tarjeta a:hover{background: url('img/tarjetas.png') -200px 0px;}
#tarjeta a:active{background: url('img/tarjetas.png') -200px 0px;}
#tarjeta, #tarjeta a, #invitaciones, #invitaciones a,#cuadros, ↩
#cuadros a,#paquetes, #paquetes a{
 position:absolute;
 width:200px;
}
#tarjeta a, #invitaciones a, #cuadros a, #paquetes ↩
a{background-repeat:no-repeat;}
#tarjeta, #tarjeta a, #invitaciones, #invitaciones a{height:69px;}
#cuadros, #cuadros a, #paquetes, #paquetes a{height:41px;}

</style>
<script src="data.js"></script>
…
  <img src="img/liston.png" width="150" style="position:absolute; ↩
bottom:-12px; left:30px; z-index:2;" alt="liston"/>
  <div id="tarjeta"><a onClick="tarjeta()"></a></div>
  <div id="invitaciones"><a onClick="invitaciones();"></a></div>
  <div id="cuadros"><a onClick="cuadros()"></a></div>
  <div id="paquetes"><a onClick="paquetes()"></a></div>

  <div id="frame">
    …
  </div>
```

2.7 Código HTML y CSS del archivo “Productos”

```
function inicio(){
 document.getElementById("frame").innerHTML=
 …
}
 Fnction tarjeta(){
 document.getElementById("frame").innerHTML=
 …
}
function invitaciones(){
 document.getElementById("frame").innerHTML=
 …
}
function cuadros(){
 document.getElementById("frame").innerHTML=
 …
}
function paquetes(){
  document.getElementById("frame").innerHTML=
 …
}
```

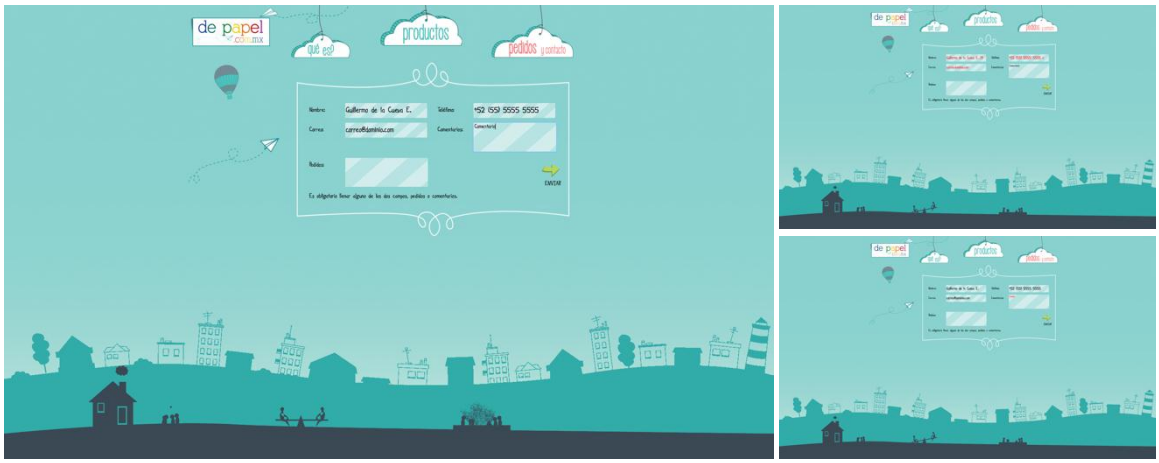
2.8 Código JS para la inserción de código HTML en la página “Productos”

Algunos de estos grupos de productos eran demasiado grandes como para que cupieran en el tamaño de la ventana, por tanto fue necesario hacer que el elemento con ID “frame” que contenía estos grupos fuera desplazable. Así, el visitante podría moverse verticalmente de forma parecida a la de cualquier ventana para visualizar todos los productos. Por esta misma razón tuve que generar un efecto visual de desvanecimiento en la parte superior; y para ello agregué una imagen PNG con canal alfa, que un extremo es del mismo color que el del fondo de la página y en el otro extremo es transparente; lo que produce un degradado de un extremo al otro. Así logré que, en su desplazamiento hacia arriba, los productos fueran desapareciendo suavemente como se puede ver en la figura 2.9, al hacerlos pasar debajo de la imagen GIF; también modifiqué el nivel de las capas de la imagen GIF y del elemento que contiene los productos utilizando las propiedades de profundidad (z-index) y posición (position) en CSS; este último es indispensable para que funcione el primero. Igualmente utilicé la propiedad de desbordamiento (overflow) con el valor de scroll para el elemento que contiene los productos, pues de otra forma se corría el riesgo de que no funcionara la capacidad de desplazar los productos en algunos navegadores y dispositivos.

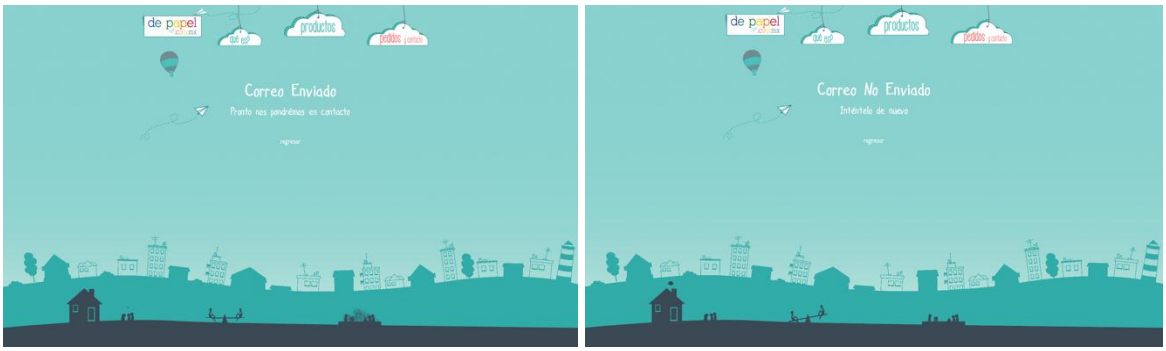
2.9 Captura de pantalla de la página “productos” con el código JS ejecutado para la carga de nuevos contenidos

Finalmente, generé una página para el envío de formularios a la empresaria, a través de los cuales los clientes puedan emitir mensajes de pedidos o comentarios; como se ha dicho, estos incluyen nombre y medio de contacto (teléfono y correo electrónico). Para el envío de información, la página cuenta con código que generé en PHP para revisar que se cumplieran con los campos obligatorios y que cumpliera con patrones determinados. Una vez que el formulario se reenvía sobre misma con la información por el método POST, captura la información, la revisa, la procesa y si cumple con todos los requerimientos, la

envía con la función mail para generar un correo a la cuenta de correo asignada, en caso de que el visitante incurra en algún error o que no cumpla los requisitos, lo regresa y marca en letras rojas el campo que no lo cumple y en caso de algún error o cuestión de seguridad como se muestra en la figura 2.10, el algoritmo primero intentará de bloquear las inyecciones de datos y en caso de no resolverlo, mandará un mensaje de error anulando el envío (ver figura 2.11).




2.10 Captura de pantalla de la página “Contacto”




2.11 Captura de pantalla del resultado generado del envío del formulario en la página “Contacto”

La tercera fase del proyecto consistió en hacer pruebas, recibir retroalimentación y proceder a la depuración. El desarrollo de todo el sitio web lo validé ante la W3C.org por medio de su herramienta para verificar los archivos HTML y CSS en su sitio web y lo probé un con un servidor XAMPP instalado para visualizar el comportamiento del sitio web con el código PHP y hacer su depuración. Una vez terminado el sitio web, se contrató el servicio de hosting y la renta del dominio a una empresa mexicana conocida como Neubox.com. Una vez contratado los servicios, por medio del protocolo FTP con una cuenta de usuario y

una contraseña dada por los proveedores del hosting se hizo la conexión y se subieron los archivos a la ruta “/public-html”, hubo una espera de 24 horas para que el hosting se terminara de configurar y publicara el dominio al resto de la web.

## Resultados

Planeé, elaboré y desarrollé un sitio web accesible a través de internet para la publicidad, venta de productos y contacto con la administradora. Desarrollé un sitio web con un diseño intuitivo y sencillo de navegar para visitantes de cualquier edad y nivel de habilidad en el manejo de computadoras y dispositivos que se conectan a internet que, además, tiene una amplia compatibilidad con diferentes navegadores para asegurar al máximo el correcto despliegue de la información. Para el envío de formularios, utilicé código PHP y lo desarrollé a prueba de inyecciones dándole mayor seguridad y evitando ciertos ataques por este medio.

El avance del proyecto fue ordenado y casi sin retrasos salvo que se tuvo que suspender el proyecto por un largo periodo por parte de la clienta; hubo buena comunicación con la empresaria, ella misma proporcionó los elementos visuales para su página y la interacción entre ella y yo fue buena, así que el proyecto resultó satisfactorio para ambos, el proyecto se empezó en Agosto 2012 y se finalizó por Diciembre 2013. Ya concluido el trabajo, le seguí dando soporte por algunos días para el ajuste de detalles mínimos y actualizaciones de contenido.

## Conclusiones

Cuando el empresario contratante de servicios informáticos no tiene una idea muy precisa sobre el tipo de producto o página web que requiere, es indispensable que el desarrollador pregunte por sus necesidades concretas, que proponga posibilidades y que mantenga abierta la comunicación para los posibles ajustes y cambios. La planeación ha de ser cuidadosa y someterse, en lo general, a consideración del contratante por si tiene sugerencias adicionales. Además, es vital que en el proyecto se desarrollen con suma claridad todas las fases y todos los puntos durante el tiempo de vida de él. De esta suerte se obtienen beneficios como las estimaciones precisas de tiempos y costos e igualmente respuestas rápidas ante situaciones o conflictos no previstos.

El producto final lo entregue oportunamente, terminado y funcional para cumplir con su objetivo.

# III. Desarrollo de sitio web “Garani S.C.”

## Objetivo

Planear, diseñar y desarrollar un sitio web para publicidad, venta, pagos y administración de artículos y servicios de sustentabilidad energética. Generar un espacio estructurado, ordenado, sencillo, agradable a la vista y fácil de usar; por lo que toca a los visitantes, esto va enfocado a darles información acerca del giro del negocio, de sus productos y servicios, en tanto que, para el empresario es un medio idóneo para administrar información e ingresos, además de conectar a sus potenciales clientes de manera fácil y automática.

## Antecedentes

Garani S.C. es una compañía dedicada a la venta de productos y servicios de sustentabilidad energética. Hasta el momento de mi primera entrevista con él, su propietario carecía de sistemas electrónicos para publicidad o ventas que le permitieran llegar a más personas, pero tampoco los tenía para manejar de manera más eficiente la información interna de su empresa.

## Descripción del problema o contexto de la participación profesional

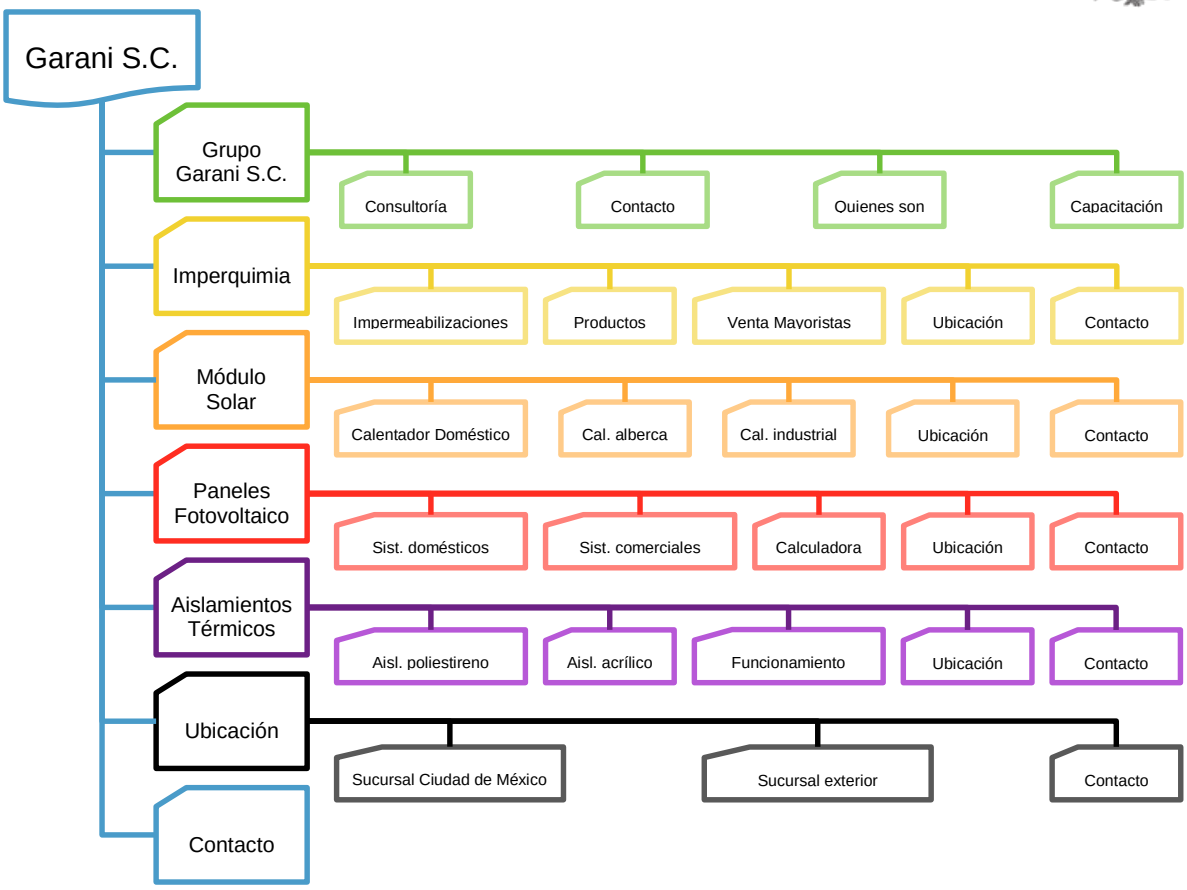
Como en el caso anterior, este empresario apenas si tenía nociones generales de que podía cubrir las necesidades de su negocio a través de medios computarizados, pero no sabía determinar con precisión cuáles podían ser estos. Su problema inmediato consistía en disponer de un recurso que, al tiempo que fuese atractivo, pudiera incluir una gran cantidad de información sobre diferentes productos y servicios para difundirlos. Ésta, naturalmente, debía presentarse ordenadamente, sin que ello resultara en un listado tedioso o cansado para la vista. Mi sugerencia fue elaborar una página web y en vista de que el empresario tenía ya imágenes y había decidido colores particulares, mi reto era lograr con ellos un vehículo de comunicación simple y amigable entre empresario y clientes. Por otro lado, recomendé la implementación de un sistema para administrar la gama de productos ofrecidos y poder recibir y controlar pagos, no únicamente en una sola exhibición, sino también a plazos mensuales. La única restricción que puso el empresario a este respecto fue que no se contratara ningún servicio bancario de pago en línea.

Habiendo llegado a un acuerdo, se me encomendó investigar, analizar y desarrollar el proyecto; se me hizo responsable de la dirección, administración de tiempo de vida, y cálculo financiero del mismo.

## Metodología utilizada y participación profesional

Como la empresa no poseía ningún sitio web, empecé desde el inicio. Generé un proyecto y un plan de seguimiento cuya primera fase consistió en un análisis, que buscaba el cumplimiento de los objetivos, las oportunidades y la manera óptima de conseguirlos. El diseño visual lo ideé como una forma de navegar agradable, intuitiva y sencilla y, además, que permitiera el acceso expedito a los datos relevantes de y para la empresa. Durante este proceso desarrollé ideas, planes y diseños, establecí costos y los acordé con el empresario.

Discutimos de conjunto el ordenamiento de la información sobre productos y servicios que se tendrían que divulgar y para el efecto hice una fragmentación de ellos, separando grandes bloques de acuerdo con el producto o actividad en cuestión. Cada grupo fue, a su vez, subdividido en diferentes modelos de productos y servicios para facilitar la navegación, la búsqueda y localización. A la postre, la jerarquización quedó como sigue (ver figura 3.1):




3.1 Mapa jerárquico del sitio web Garani S.C.

El diseño lo elaboré con base en la estructura antes descrita, en un espacio austero, poco cargado y en todas las páginas apoyado con elementos visuales como imágenes fijas, carruseles y videos. La meta era que fuera intuitivo y fácil de navegar en cada página para que los visitantes ubicaran rápidamente la información de su interés. Para simplificar más las cosas y resolver las dudas iniciales de los visitantes, en la mayoría de las páginas se agregó una sección de preguntas frecuentes que responden las preguntas más comunes para cada grupo y subgrupo. Para evitar la redundancia y aprovechar recursos al máximo, la página “ubicación” en cada grupo apunta a un único archivo HTML que incluye a los subgrupos referidos en la tabla anterior. De igual manera, la sección de contactos de cada grupo conduce una misma y única página. En la página de contactos existía un formulario dirigido a que el visitante pudiera contactar a la empresa; y para hacer posible la identificación de la página y tema en la que estaba interesado el cliente, se agregó un elemento de selección con el título de asunto. En esta se encontraban todos los temas de

interés que el sitio web tenía a disposición; así generé una sección de contactos para todo el sitio web y evité crear más páginas con código repetido para cada sección de la empresa.

La segunda fase consistió en el desarrollo. Generé el código y programé con el uso de lenguaje de etiquetas HTML, hoja de estilo CSS, lenguaje de programación lado de servidor PHP y librerías para la construcción de carruseles y envío de correos HTML.

Con la intención de darle calidad visual, generé una única hoja de estilo con los diseños generales que compondría todo el sitio web. En cada página del sitio web solo se necesitaba hacer la referencia a la hoja de estilo mencionada a través de un enlace simplificado; con ello evité repeticiones del mismo código en cada página. En aquellos casos en que una página web requiriera un código adicional y sólo utilizado para esa página, mi decisión fue agregarlo directamente sobre el archivo HTML que lo requiriera con una etiqueta “style”, así resultó fácil mantener cierto orden y evitar comportamientos, inesperados por redundancias, además se le dio prioridad de aplicación a este código adicional sobre los códigos referenciados. En este diseño generé código necesario para mantener en todas las páginas del sitio centradas con un ancho de 1020px tomando en cuenta que, según estadísticas del sitio web “w3schools”, hay registro de que el 96.7 % de sus visitantes poseen resoluciones de pantalla igual o mayores de 1024px en enero del 2016. En el sitio web es muy recurrente el uso de columnas para dar información de manera ordenada por lo que utilicé ID con el nombre de “col1” y “col2” para poder definir las columnas a la altura que fuera necesaria. Para que funcionara, utilicé la propiedad de flotante (float) con un ancho definido de 490px puesto que el ancho de la página no supera el tamaño 1020px y posee también un tamaño de relleno en cada columna como se puede visualizar en la figura 3.2.

```
#col1 h1, #col2 h1{text-align:center;}
#página{
 margin:auto;
 margin-bottom:0px;
 width:1020px;
}
#col1, #col2{
 float:left;
 width:490px;
 min-height:500px;
 padding:10px 10px;
}
#col1{
 clear:both;
 background-color:#E6CDAC;
}
#col2{
 color:#CCC;
 background-color:#8C0A0A;
}
```

3.2 Código CSS

Para todas las páginas de este sitio generé y utilicé un mismo código para la cabecera, el menú principal de navegación y el pie de página que inserté en cada página HTML. Para la cabecera asigné propiedades de estilo del archivo CSS a un elemento DIV con el ID “cabecera” y elementos hijos. El primer elemento hijo es una imagen del logotipo de la empresa que funciona como un hipervínculo hacia la página principal del sitio y los otros dos elementos hijos son otras dos imágenes que funcionan como hipervínculo a las redes sociales de la empresa en Facebook y Twitter; estos elementos utilizan la propiedad de flotante para reacomodarse dentro del elemento padre. Para desarrollar el menú de navegación, generé un elemento DIV que contiene solo elementos de anclaje (A) que funcionan como hipervínculos para redirigir hacia alguna página del sitio web. Dichos elementos de anclaje (A) tienen la propiedad de cambiar su color de fondo cuando el puntero se posiciona encima de ellos. El texto que contienen algunos de estos elementos antes mencionados ocupaba más de un renglón, lo que constituía un problema, pues los elementos tenían un tamaño determinado para contener solo uno, lo que ocasionaba un desacomodamiento de los demás elementos. A fin de resolver esto, asigné a todos aquellos que así lo requirieran una clase que redimensiona el elemento para que se ajuste con los demás. Para el pie de página utilicé un elemento DIV con un ID que contiene elementos SPAN e imágenes, y al igual que los anteriores este elemento padre tomaba las propiedades de estilo para aplicarlo a sí mismo y a los elementos internos a él; utilizando la mencionada hoja de estilo referenciado (ver figura 3.3, 3.4 y 3.5).

```
…
 <div id="cabecera">
  <a href=https://twitter.com/...” class="icons” ↩
id="twitter"><img src="img/twitter.png" ↩
alt="twitter" /></a>
  <a href="https://www.facebook.com/..." ↩
id="facebook"><img src="img/facebook.png" ↩
alt="facebook" /></a>
  <a href="index.html"><img ↩
src="img/cabezera.png" alt="logo" /></a>
 </div>
 <div id="menu">
  <a href="grupogarani/index.html">Grupo ↩
Garani</a>
  <a ↩
href="imperquimia/index.html">Imperquimia</a>
  <a href="modulosolar/index.html">Módulo ↩
Solar</a>
  <a href="panelesfotovoltaicos/index.html" ↩
class="line">Páneles<br />Fotovoltáicos</a>
  <a href="aislamientorermico/index.html" ↩
class="line">Aislamiento<br />Térmico</a>
  <a ↩
href="ubicacion/index.html">Ubicación</a>
  <a href="contacto.php" ↩
class="fill">Contacto</a>
 </div>
 …
 <div id="piepágina">
  <img src="img/garani2.png" alt="garani" />
  <img src="img/imperquimia2.png" ↩
alt="imperquimia" />
  <img src="img/modulosolar2.png" ↩
alt="modulo" />
  <div>
   <span>Grupo Garani; Garani Asesores en ↩
Sustentabilidad S.C.</span><br/>
   <span>Eje Norte Sur 16-17, local 3; ↩
CIVAC; Col. Otilio Montaño; Jiutepec ↩
Morelos.</span><br />
   <span>consultas@garanisc.com.mx (777) ↩
309 35 95, (777) 242 35 81, 01 800 839 ↩
4482</span>
  </div>
 </div>
…
```

3.3 Código HTML

```
…
#cabecera{
 padding:20px 20px;
 background:#FFF;
 position:relative;
}
#menu{background:#000;}
#menu a{
 display:block;
 font-family:Arial, Helvetica, sans-serif;
 padding:31px 0px 30px 0px;
 text-align:center;
 color:#FFF;
 background: url(img/btnmenu.png) repeat-x #000;
 width:145px;
 float:left;
}
#menu a:hover{background:#000;}
#menu .line{padding:21px 0px;}
#menu .fill{width:150px;}
#cuerpo > img{
 margin:0px 0px;
 float:left;
 width:875px;
}
#piepágina{
 clear:both;
 margin-bottom:0px;
 background: url(img/btnmenu.png) repeat-x #000;
 background-size:contain;
 height:60px;
 width:100%;
 color:#FFF;
}
#piepágina img{
 height:40px;
 position:relative;
 top:10px;
 margin:0px 20px 0px 20px;
}
#piepágina div{
 font-size:10px;
 color:#CCC;
 text-align:center;
 width:500px;
 margin:10px 20px 0px 20px;
float:right;
}
.icons{
 position:absolute;
 bottom:10px;
}
#twitter{right:70px;}
#facebook{right:10px;}
#cabecera img[alt=twitter], #cabecera ↩
img[alt=facebook]{width:50px;}
#cabecera img[alt=logo]{width:500px;}
.align-center{text-align:center;}
```

3.4 Código CSS

3.5 Captura de imagen de la cabecera, menú principal y pie de página

Para la página principal –con ayuda de una librería llamada “Flex Slider” de WOO Commerce– agregué un carrusel con imágenes proporcionadas por el cliente para la publicidad y anuncios de todos los servicios y productos que ofrece la empresa. Además, en cada diapositiva asigné un hipervínculo hacia la página correspondiente de la que habla el anuncio. Para el funcionamiento de esta librería los elementos mínimos necesarios son un elemento DIV con un nombre de clase predefinido con el nombre “flexslider”, en su interior un elemento de lista (UL) con un nombre de clase “slides” y finalmente dentro de éste, los elementos de objetos de lista (LI) deseados por cada diapositiva. Finalmente para ejecutarlo y generar de manera dinámica el carrusel, fue necesario darle propiedades específicas, utilizando para ello código JS con jQuery (ver figura 3.6). En esta página, al igual que en muchas otras, agregué dos elementos DIV con los ID Col1 y Col2 respectivamente para crear secciones con 2 columnas; la primera columna incluye texto acerca de la empresa, sus metas y objetivos, además de una imagen; en la segunda columna se agregó información acerca de los servicios que ofrece la compañía y los datos de contacto (ver figura 3.7).

```
…
<script src="http://code.jquery.com/jquery-1.10.1.min.js"></script>
<link rel="stylesheet" href="flexslider/flexslider.css" type="text/css">
<script src="flexslider/jquery.flexslider.js"></script>
<script type="text/javascript" charset="utf-8">
$(window).load(function() {
 $('.flexslider').flexslider({
  controlNav: false,
  directionNav: true,
  prevText: "Anterior",
  nextText: "Siguiente",
  slideshowSpeed: 3000
});});
</script>
<style type="text/css">
.flexslider{margin:-15px 0px -15px 0px;}
.slides li{margin:0px;}
…
</style>
…
 <div id="cuerpo">
  <div class="flexslider">
   <ul class="slides">
    <li><a href="grupogarani/index.html"><img src="img/banner/garanisc.png" alt="garani"/></a></li>
    <li><a href="imperquimia/index.html"><img src="img/banner/imperquimia.png" alt="imperquimia"/></a></li>
    <li><a href="modulosolar/index.html"><img src="img/banner/modulosolar.png" alt="modulo"/></a></li>
    <li><a href="panelesfotovoltaicos/index.html"><img src="img/banner/panelesfotovoltaicos.png" ↩
alt="panel"/></a></li>
   </ul>
  </div>
  <div id="cuerpo">
…
  </div>
```

3.6 Código Flex Slider

3.7 Captura de pantalla del carrusel con Flex Slider

Para la elaboración del formulario de contacto, empleé una página compuesta por una parte de contenido con código HTML y CSS, y otro código en PHP; esta última es lo que utilizará el servidor para generar código HTML de manera dinámica. Empleé los códigos HTML y PHP en un mismo archivo por varias razones. La primera: evitar la generación de archivos adicionales (uno en HTML y otro en PHP) a fin de impedir el salto entre estas páginas. En segundo lugar, ahorrarme el uso de variables de sesión, puesto que si separáramos en archivos diferentes este sistema, sería indispensable el uso de estas variables para el paso de datos entre distintos archivos, por ejemplo, cuando el servidor encuentre un error en los datos ingresados por el visitante para enviar el formulario, o siempre que el servidor necesite notificar si algún campo no cumple con los requisitos. Aunque quizá la razón más importante haya sido evitar el uso de encabezados que se utilizan comúnmente para redireccionar a otra página en el código PHP. Si por alguna razón surge un error o falla en el código PHP, el servidor ya no tiene manera de regresar el visitante a la página en donde empieza (ver figura 3.8).

En el código PHP utilicé una librería llamada PHPMailer que sirve para la construcción de correos electrónicos con formato HTML para el formulario. Este código contiene un fragmento de comprobación que sirven en primera instancia, para verificar que los datos ingresados cumplan con un patrón determinado previamente, y en segunda, para que los campos obligatorios tengan que ser llenados forzosamente. Para lograr esto, utilicé sentencias IF y con la generación de notas que se desplegarán en los campos que no cumplen con los requisitos antes mencionados. Además, otro fragmento revisa que los formularios que se quieran enviar no tengan inyecciones de datos, y en caso de detectar alguno, el sistema pueda tomar acciones como eliminarlos o bien, abortar el proceso.

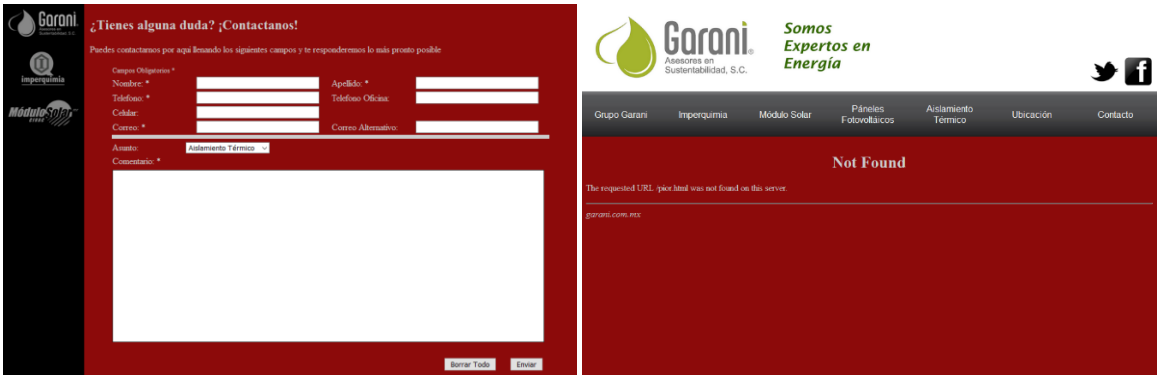
```
<?php
[Código PHP]
?>
…
<style>
#col2{
 width:855px;
 height:670px;
}
form table{
 width:780px;
 border:none;
 margin:auto;
}
form table tr td input[type=text]{width:90%;}
</style>
…
 </div>
 <div id="cuerpo">
  …
  <div id="col2">
   <h2>¿Tienes alguna duda? ¡Contactanos!</h2>
   <p>Puedes contactarnos por aqui llenando los siguientes campos y te responderemos lo más pronto posible</p>
   <form action="<?php echo htmlspecialchars($_SERVER["PHP_SELF"]);?>" method="post" name="contacto" ↩
id="contacto">
    <table>
     <tr><td colspan="4" style="font-size:12px">Campos Obligatorios *</td><tr>
     <tr>
      <td style="width:150px;"><label class="text" for="nombre">Nombre: *</label></td>
      <td style="width:250px;"><input name="fname" type="text" id="nombre" value="<?php echo ↩
[VarNombre];?>"/></td>
      <td style="width:150px;"><label class="text" for="apellido">Apellido: *</label></td>
      <td style="width:250px;"><input name="lname" type="text" id="apellido" value="<?php echo ↩
[VarApellido];?>"/></td>
     </tr>
     …
     <tr><td colspan="4">
       <label id="text">Asunto:</label>
       <select name="asunto" style="margin-left:80px;">
        <option value="Aislamien Termico">Aislamiento Térmico</option>
        <option value="Contacto">Contacto</option>
        …
       </select>
      </td></tr>
     <tr><td colspan="4"><label class="text" for="comentario">Comentario: *</label></td></tr>
     <tr><td colspan="4"><textarea name="comentario" rows="20" wrap="hard" id="comentario" ↩
style="width:760px; height:300px; resize:none;">
      <?php [VarComentario] ?>
     </textarea></td></tr>
     <tr style="height:70px;"><td colspan="2" class="align-center">
      <?php
      if($_SERVER["REQUEST_METHOD"] == "POST")
      if(strlen([ErrorNombre])!=0)
        echo "<h1>".[ErrorNombre]."</h1>";
      else if(strlen([ErrorApellido])!=0)
        echo "<h1>".[ErrorApellido]."</h1>";
      else if(strlen([ErrorCorreo])!=0)
        echo "<h1>".[ErrorCorreo]."</h1>";
      …
      else if(strlen([ErrorComentario])!=0)
        echo "<h1>".[ErrorComentario]."</h1>";
      else
        [Funcion resultados al enviar];
      ?>
```

```
      </td><td colspan="2" style="width:180px; text-align:right;">
       <input type="reset" value="Borrar Todo" style="margin:10px"/>
       <input type="submit" value="Enviar" style="margin:10px"/>
      </td></tr>
    </table>
   </form>
  </div>
 </div>
…
```

3.8 Código general

El formulario está compuesto de campos de texto, un elemento de selección y un elemento de área de texto. Los campos de texto obtienen los datos del visitante: nombre, apellido, teléfono local, teléfono de oficina, celular, correo principal y correo alternativo. En el elemento de selección, el visitante puede elegir el tema sobre el que quiere pedir o enviar información; finalmente, en el área de texto el visitante puede ingresar la información por la que está interesado. Cada campo de texto posee un elemento de etiqueta (LABEL) que precisa qué información debe ingresar el visitante; cada uno de estos pares está relacionado al compartir el mismo valor; en cuanto al atributo FOR, en el elemento de etiqueta y en cuanto al atributo ID, en el campo de texto. Con ello, siempre que el visitante haga clic sobre una de las etiquetas, gracias al campo que está asociado al ésta se activará para empezar a escribir. Finalmente solo se agrega en las etiquetas un símbolo “ * ” y una nota por convención para avisar al visitante de que esos son campos obligatorios de llenar y que la programación de lado del servidor verificará que los elementos obligatorios hayan sido llenados.

También generé archivos SHTML con los errores de petición al servidor 400, 401, 500, etcétera para que tuviera un diseño de acuerdo con el sitio en general (ver figura 3.9).




3.9 Captura de pantalla de la página “Contacto”

Para la sección de ubicación, generé dos páginas en el directorio ubicación, una es la principal y otra con el nombre de “distrito.html” la primera es un mapa que muestra el establecimiento que está en Morelos y el segundo es un mapa que muestra el establecimiento en la Ciudad de México. Para la muestra del mapa utilicé un elemento de contexto de navegación animada (iframe) que la empresa de Google pone a disposición a través de “Maps” para proporcionar a sus visitantes la posibilidad de insertar regiones determinadas del mapa. Para este caso particular se agregó código CSS en los archivos HTML directamente para modificar y adaptar las propiedades en estas páginas de manera particular (ver figura 3.10).

```
...
<div id="col2">
 <h1>Sucursal en el D.F.</h1>
 <p>GARANI Asesores en Sustentabilidad, S.C. ↩
está ubicado en José María Rico #57, Colonia ↩
Actipan, Delegacion Benito Juarez, CP 03230, ↩
México D.F. México.</p>
 <p><iframe ↩
src="https://www.google.com/maps/embed?..." >
</iframe><br><small>
   <a href="https://goo.gl/maps/1PQzm"> ↩
Ver mapa más grande</a></small></p>
</div>
```

3.10 Código y Captura de pantalla de la página “Ubicación”

El resto de las páginas del sitio web mantiene una estructura similar al de la página de principal; adicionalmente cada página contiene un submenú lateral para navegar en las páginas que pertenecen al grupo en el que está el visitante; cabe señalar que el submenú (al igual que el menú) contiene más elementos dentro de él y que fue posible utilizar las mismas propiedades de estilo que las del menú. El resto de estas páginas tendrá información y algunos elementos diferentes en cada página como videos e imágenes con relación a la sección de información a la que se esté recurriendo, pero el diseño visual y la estructura mantendrán una forma similar con pequeños cambios y adaptaciones por medio de código CSS dentro de cada archivo HTML (ver figura 3.11).

3.11 Capturas de pantalla del diseño de las páginas en general

La tercera fase consistió en la depuración, revisión de seguridad y retroalimentación. El sitio web lo validé ante la W3C.org, también generé una simulación y lo probé un con el servidor XAMPP con propósitos de obtener una retroalimentación del comportamiento con el código PHP en los formularios y hacer la depuración necesaria. La cuarta fase, una vez terminado el sitio web, consistió en contratar el servicio de hosting y dominio. En el hosting, por medio del servicio FTP, almacené todos los archivos que componen el sitio web y el dominio lo utilicé para asociar un nombre que se difundirá a través de internet con la dirección IP del servidor que contiene el hosting. Para estos dos servicios contraté a la empresa de Neubox.com por la experiencia previa que tengo con sus servicios.

## Resultados

Planeé, elaboré y desarrollé un sitio web, accesible a través de internet, para la publicidad, información, venta de productos y contacto con la empresa. Diseñé y desarrollé el sitio web con una estructura organizada de forma jerárquica para la distribución de información, además de un diseño intuitivo y sencillo de navegar donde los clientes pudieran encontrar fácilmente la información en que estén interesados. Tiene una amplia compatibilidad con diferentes navegadores para desplegar la información de manera correcta. El código PHP lo diseñé a prueba de inyecciones para una mayor seguridad y evitar ataques o spams. El avance de todo el proyecto fue bien coordinado entre el cliente y yo, los retrasos fueron mínimos y en su mayoría se dieron por ajustes o reordenamientos a la hora de agregar el

texto o material proporcionado por el empresario, el desarrollo de este proyecto tuvo una duración de 1 mes y se inició aproximadamente en junio del 2015. Ciertamente hubo muchos cambios con respecto al plan original y estos fueron dándose sobre la marcha de acuerdo con puntos imprevistos o no considerados inicialmente. El empresario quedó satisfecho en mi trabajo y actualmente, ya como proyectos independientes, sigo dando soporte y actualizando el contenido.

## Conclusiones

Aunque en apariencia una página web es similar a otra u otras, lo cierto es que el proyecto de desarrollo particular de cada una es una especie de huella digital, que pertenecerá exclusivamente a una única institución o empresa con necesidades y rasgos singulares. Por ello, aunque hay muchas maneras de emprender un proyecto y diferentes técnicas, planes y organizaciones para lograrlo, al final, lo más importante es elegir precisamente lo más funcional en cada caso. Por parte del desarrollador, cuenta mucho la disposición para ser flexible y adaptable a las situaciones, sin que ello obste en adoptar finalmente una decisión y una resolución que conduzca a materializar el proyecto. En la ejecución han de preverse situaciones inesperadas que demanden un cambio de táctica y la aplicación de acciones inmediatas para solucionar los inconvenientes. Como responsable uno tiene que ser lo más asertivo posible y aprender de la experiencia para reducir o subsanar los problemas, porque la teoría no siempre se ajusta a las circunstancias que presenta la realidad. Este proyecto, en particular, puso a prueba mi capacidad de adaptar mis conocimientos y habilidades a la consecución de un fin, y lo logré exitosamente.

# IV. Desarrollo de sitio web “CoViBE”

## Objetivo

Analizar, diseñar y desarrollar un sitio web para la divulgación de información, publicidad y contacto con clientes potenciales a fin de brindar servicios de bienestar emocional y consejería a mexicanos que viven en el extranjero.

## Antecedentes

CoViBE es un proyecto que promueve el bienestar emocional de las personas, vía telefónica y por internet. Diseñado específicamente para mexicanos que viven en los Estados Unidos, funciona a través de dispositivos como teléfonos, teléfonos inteligentes, tabletas o computadoras. Sus consejeros atienden a personas que han sufrido o sufren violencia, depresión o soledad y lo hacen desde el supuesto de que una persona en un país ajeno preferirá siempre recibir apoyo de un connacional que de alguien extraño a su propia cultura.

## Descripción del problema o contexto de la participación profesional

Inicialmente se había contratado a un terciario para el desarrollo del sitio web de CoVibe, donde se pudiera publicar información de contenido y servicios; sin embargo, por algunas dificultades el encargado fue despedido y sólo entregó avances del proyecto. Tiempo después, nos contactaron y contrataron para terminar el proyecto. Cuando me hice cargo, se me entregaron los datos de acceso para el servicio de hosting que ya se había pagado previamente, con su nombre de dominio apuntando al servidor. Lo que me encontré fue un sitio web incompleto que tenía redundancias innecesarias, así que lo primero que hice fue revisar, limpiar, corregir y restructurar la parte del proyecto que había sido desarrollada por el terciario que inició el proyecto.

Una vez concluida esta operación preliminar, sostuve entrevistas con la responsable para conocer sus necesidades concretas. Ella tenía ideas claras sobre el diseño y la manera de exponer la información a sus clientes, así que lo demás fue aterrizar y dar forma al plan.

Cuando concluí la planeación, empecé el desarrollo del contenido faltante para completar el sitio web, una vez terminado, depurado y con las pruebas realizadas, procedí a subir todos los datos del proyecto al servidor previamente contratado para la publicación

oficial del programa y la finalización de proyecto. Respecto de éste fui el encargado de analizar, revisar, limpiar y desarrollar el sitio web; personalmente me hice cargo de la dirección, administración de tiempo de vida para publicar el sitio web a través de internet y finalización del proyecto.

## Metodología utilizada y participación profesional

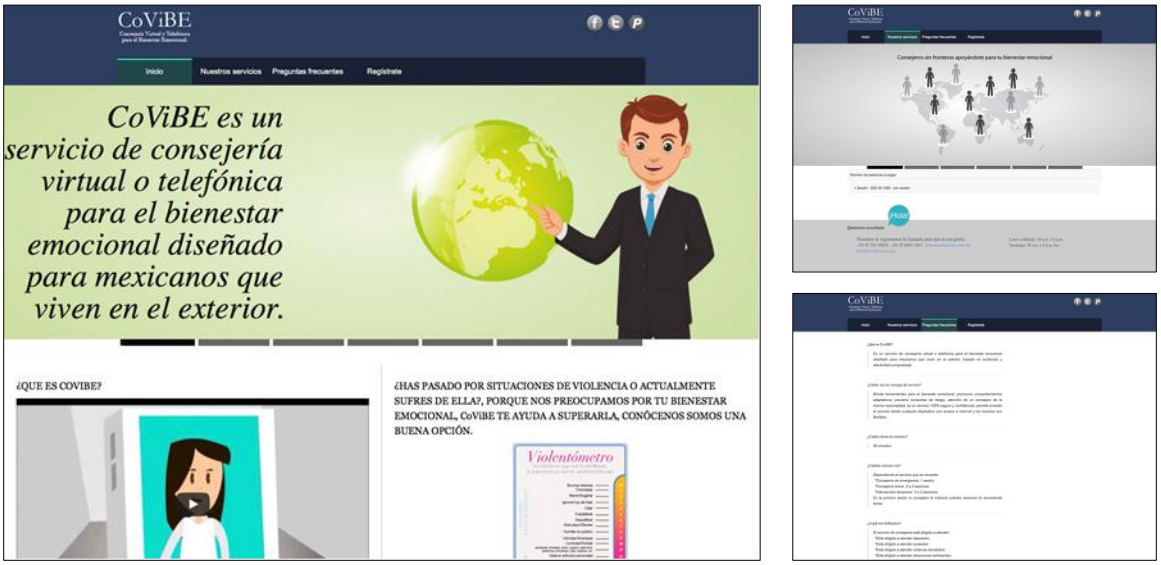
Como en este caso mi papel fue dar continuidad y culminar un proyecto previamente desarrollado, la fase uno consistió, en principio, en recopilar toda la información y el material, revisar el avance del proyecto de mi antecesor para informarme de lo que había hecho y los avances realizados, precisar y redefinir los objetivos y la misión de este sitio web y, en general, aclarar los puntos previamente acordados entre la responsable y el anterior desarrollador.

Le entrevista con la responsable me puso al tanto de sus propósitos y de los ofrecimientos del otro desarrollador, que básicamente consistían en hacer del sitio web un medio publicitario de los servicios de asistencia emocional para un determinado sector de población nacional en el extranjero; debía incluir datos precisos sobre la oferta de atención y una forma sencilla de establecer contacto con los interesados. El proyecto de desarrollo tenía un ciclo de vida definido, pues tenían una fecha precisa para la apertura del servicio al público; por dicho imperativo el desarrollador anterior empezó a diseñar y armar la estructura general del sitio web, pero –como se ha señalado antes– se presentaron dificultades y ahora el tiempo apremiaba.

Fue necesario que entrevistara también al desarrollador anterior para enterarme de detalles técnicos y del estado real de avance del trabajo, indagué por los elementos que empleó en el diseño (por ejemplo, librerías), y pedí pormenores sobre el servicio de hosting y de dominio registrado. Mi antecesor había recurrido a una librería llamada SKEL para dar al sitio web la capacidad responsiva con múltiples dispositivos de diferentes proporciones de pantalla. También supe que alcanzó a desarrollar el menú principal, la cabecera y el pie de página de la primera página web. Tenía elaborado un formulario para el contacto de la página con código HTML y la programación del servidor los elementos básicos de un envío de formulario en PHP. Se había contratado ya un servicio de hosting y registrado un dominio a la par que se generaba el sitio web con la dirección www.covibe.com.mx por lo que se me proporcionaron los datos de accesos para el servicio de hosting.

Ya con toda esta información en mi poder, el siguiente paso fue hacer una revisión general del código que ya se había desarrollado, lo revisé de manera detallada y corregí todo aquello que consideré necesario. Después hice pruebas de compatibilidad con los navegadores, de manera que el sitio web se desplegaba de manera correcta con los navegadores más comunes, como Google Chrome, Mozilla Firefox e Internet Explorer. Y finalmente, hice una consolidación del sitio web limpiando código obsoleto, eliminando redundancias superfluas, simplificando código, etcétera.

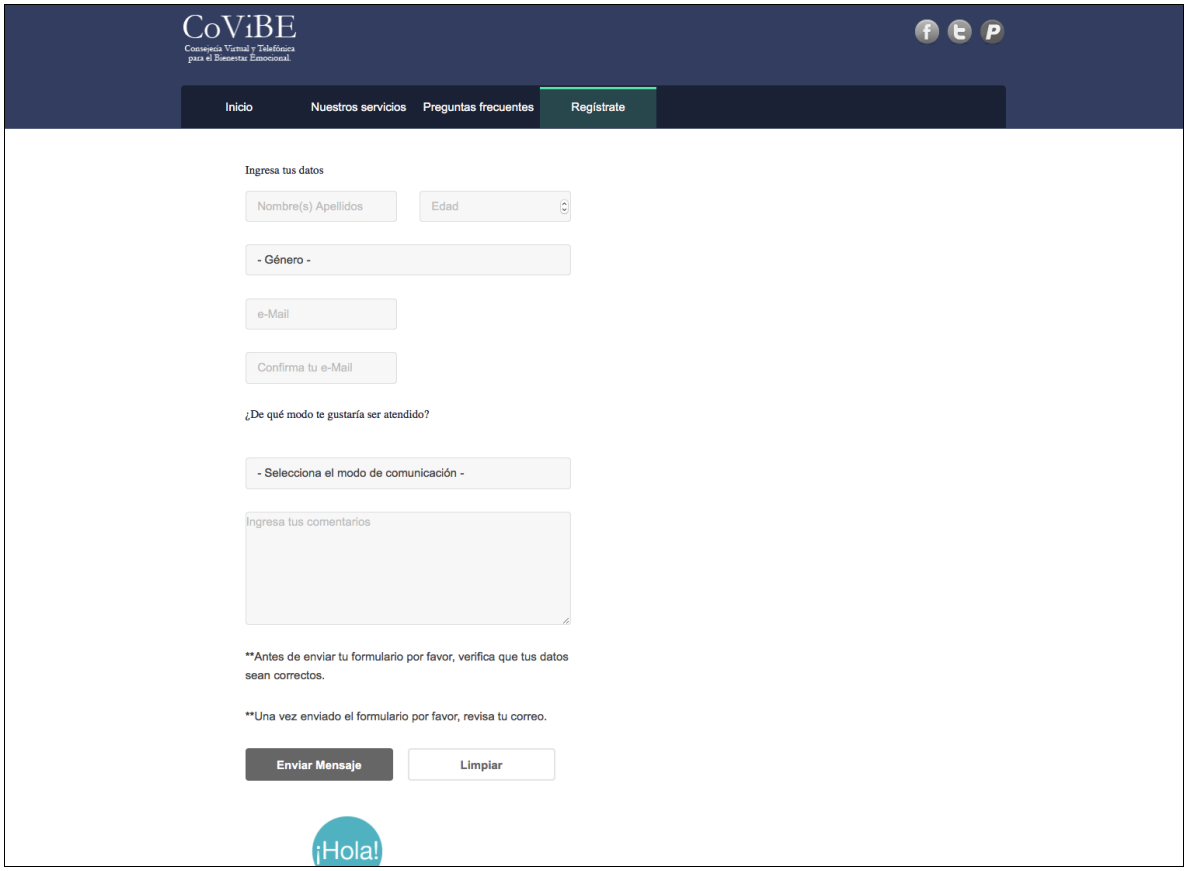
Concluida la corrección integral, lo siguiente fue tomar esta página como plantilla del sitio web y generar las páginas restantes con base en ella, haciendo sólo las alteraciones necesarias para integrar a las nuevas páginas el contenido deseado; es decir, utilicé los mismos elementos que ya estaban en la primera página (ver figura 4.1).




4.1 Captura de pantalla, el lado izquierdo en la página del antiguo desarrollador, las otras dos las creé a partir de ella

Finalmente corregí la página que contiene el formulario (originalmente elaborado por el desarrollador anterior) para la inscripción de las personas interesadas con el servicio. A esta página anexé un código en JS que tenía dos objetivos: la primera controlar los elementos que debían desplegarse u ocultarse de manera dinámica, función que se ejecutaría cuando el cliente seleccionara el medio por el que deseaba ser contactado, y la segunda verificar los datos ingresados sobre la misma página para una mayor comodidad y facilidad del usuario. Además, generé un archivo PHP que contiene el código que hace una segunda validación, elimina código malintencionado o inyecciones de información y

guarda la información del usuario enviando un correo electrónico a la organización que brinda el servicio (ver figura 4.2).




4.2 Captura de pantalla del formulario de la página

## Resultados

Di continuidad, restructuré y desarrollé un sitio web accesible a través de internet, con el objetivo de informar y publicar los servicios que ofrece este proyecto a un sector especial del público. Desarrollé, corregí y optimicé el sitio web sobre el código directamente, limpiando redundancias innecesarias que provocan procesos inútiles que al final no se reflejan en lo que el cliente visualizará. Además, por petición expresa de la responsable, generé un de un diseño sencillo con elementos visuales como imágenes, videos y carruseles para facilitar la navegación de los visitantes y la fácil localización de la información en la que están interesados. Al ser responsivo, el sitio web tiene una amplia compatibilidad con múltiples dispositivos de variadas proporciones de pantalla. Su adaptabilidad permite al cliente desplegar la información de manera correcta y expedita.

Modifiqué el código PHP con el diseño original para hacerlo a prueba de inyecciones y ofrecer una mayor seguridad, además de corregir errores.

En este proyecto obtuve mucho apoyo por parte de los integrantes del proyecto, quienes pusieron a mi disposición toda la información requerida y localizaron elementos extraviados. La buena coordinación general con ellos y la comunicación constante y expedita evitó retrasos. A la postre, la responsable quedó satisfecha con mi trabajo, pues culminamos correctamente el proyecto en el plazo previsto con una duración de 2 semanas aproximadamente empezando el mes de octubre del 2015. Finalmente el sitio web se abrió al público en la fecha fijada.

## Conclusiones

Retomar un proyecto inconcluso realizado por otro u otros desarrolladores tiene sus ventajas y sus aristas. Por un lado, el contar con algunos avances puede traducirse en un ahorro en el tiempo de trabajo, pues algunas etapas del proyecto han sido ya cubiertas; aunque, por el otro, esto no garantiza que las cosas sean correctas o estén bien hechas. La ventaja en el caso del proyecto sobre el que informo fue que pude tener contacto con el desarrollador anterior y empaparme de los detalles de su trabajo, del estado del proyecto de desarrollo y de otras dificultades; con ello, fue posible detectar con mayor rapidez los puntos críticos y avanzar con certeza. Así, terminar el sitio web, generando las páginas faltantes y depurando las existentes, antes de subirlas al servidor previamente contratado para la publicación en línea, no ofreció demasiados obstáculos. Otro aspecto vital fue la buena comunicación que establecí con la responsable y el personal, lo que hizo posible culminar el proyecto en tiempo y forma.

# V. Diagnóstico y corrección en la aplicación web “Soluciones Virtuales PVI”

## Objetivo

Revisar y analizar una plataforma preexistente en busca de errores y generar las correcciones para su buen funcionamiento, además de limpiar la base de datos de información redundante y/o inservible generada por los errores de la plataforma.

## Antecedentes

Soluciones Virtuales PVI es una empresa que ofrece, al público en general, apoyo y servicios psicológicos a través de una plataforma virtual desarrollada como una Web App. Ésta permite la comunicación entre terapeuta y paciente mediante video llamadas, chats en vivo o mensajes; es, además, una herramienta para que los terapeutas pueden llevar un registro de las sesiones y del historial médico de cada paciente y un espacio por cuya mediación el terapeuta puede enviar y recibir archivos formularios que el paciente ha de llenar previamente. Por otro lado, la plataforma administra y distribuye a los pacientes con los terapeutas asignados, a través del control que lleva un administrador.

La base de esta plataforma fue desarrollada por ingenieros de la UNAM; posteriormente fue creciendo con varios proyectos que realizaron diferentes ingenieros en varios momentos a fin de aumentar capacidades y mejorar la calidad del servicio.

## Descripción del problema o contexto de la participación profesional

La plataforma de servicio de Soluciones Virtuales PVI se desarrolló tiempo atrás con tecnologías que ya no son muy comunes o que están francamente en desuso, como Adobe ShockWave para el chat en vivo de las sesiones entre paciente y terapeuta. Por otro lado, utiliza igualmente funciones muy antiguas o en desuso como SQL en PHP, que actualmente y por razones de seguridad, son sustituidas por SQLI. Finalmente, tratándose de un proyecto desarrollaron varios ingenieros en distintos tiempos, se han generado errores dentro del código, lo que se traduce en procesos incompletos o con errores.

Cabe decir que para las video llamadas se utilizaban cuentas de Skype que se agregaban las ventanas sobre la página de la plataforma, lo que en algún momento se deshabilitó por cuestiones prácticas y sólo se conservó el servicio de chat en vivo y de

llamadas telefónicas. Las bitácoras en las que los terapeutas editaban información acerca de sus pacientes y sesiones, no se guardaban correctamente y a la hora de subir los datos, la plataforma tenía comportamientos impredecibles.

Con todas estas limitaciones, la plataforma siguió funcionando y brindando servicios al público, aunque la herramienta estaba severamente afectada y sus responsables buscaban alternativas y soluciones.

## Metodología utilizada y participación profesional

Dado que la plataforma de Soluciones Virtuales PVI había sido modificada varias veces, la primera medida fue familiarizarse y conocer su funcionamiento, de ahí que la fase uno consistió en hacer una investigación y un análisis de los cambios operados, de sus autores y de las razones que los determinaron. Hubo que sacar en claro qué fragmentos se agregaron y adaptaron a la pieza original y también ubicar aquellos que se modificaron. Procedí a hacer un cotejo entre la plataforma original y la versión modificada para hacerme una idea más nítida de la estructura y evitar introducir alteraciones que afectaran sus partes. Realicé igualmente pruebas de funcionamiento para detectar las partes que fallaban y tomé nota de los errores que se habían registrado en el código.

En la segunda fase del proyecto alteré o generé, según fuera el caso, nuevo código para hacer funcionar los módulos que fallaban en la plataforma; asimismo limpié la base de datos que tenía información con datos erróneos. Para tales fines y para disponer de un respaldo de seguridad, previamente hice una copia de los archivos que había alterado. Siempre que encontraba un fragmento de código inservible lo aislaba, comentándolo en bloque, e inmediatamente generaba abajo el nuevo código que debía sustituirlo. Si, en cambio, el fragmento servía y sólo hacía falta realizar algunas alteraciones, lo hacía así sobre el código generado anteriormente. Esto me permitió disponer del código inservible para fines de comparación y también para conseguir que se integrara bien a la plataforma (ver figura 5.1).

```
<script language="javascript" type="text/javascript">
    function videochat(str)
    {
        pagina = "videoPac.php?idVar="+str;
        var opciones="toolbar=no, location=no, directories=no, status=no, menubar=no, scrollbars=yes, resizable=yes, width=508, height=365, top=85, left=140";
        window.open(pagina,"",opciones);
    }
    function AbrirVideochat(){
        console.log(navigator.userAgent);
        if(navigator.userAgent.toLowerCase().indexOf('chrome') > -1 || navigator.userAgent.toLowerCase().indexOf('firefox') > -1){
            var code = document.getElementById("inputURL").value;
            if(/^[0-9]{9}$/.test(code)){
                /*document.getElementById("inputURL").style.display="none";
                document.getElementById("BotonAbrir").style.display="none";
                var win = document.getElementById("videoWindow");
                win.setAttribute("src","https://vline.com/+"+code);
                win.style.display="inline";
                document.getElementById("BotonCerrar").style.display="inline";*/

                var winVideochat = document.getElementById("videochatwindow");
                winVideochat.innerHTML = '\
                    <iframe id="videoWindow" src="https://appr.tc/r/'+code+'" style="width:460px; height:400px;"></iframe>\
                    <button id="BotonCerrar" onclick="CerrarVideochat()" style="margin-top:10px;">Cerrar Chat</button>\
                ';
                //if(/^[a-zA-Z0-9]{16}$/.test(code)){
                //<iframe id="videoWindow" src="https://vline.com/+'+code+'" style="width:400px; height:300px;"></iframe>\
            }
            else{
                alert("La clave debe ser de 9 caracteres");
            }
        }
        else{
            var winVideochat = document.getElementById("videochatwindow");
            winVideochat.style.background = "#FFF";
            winVideochat.style.padding = "20px 0px 20px 0px";
            winVideochat.innerHTML = "\
                <p>El navegador no es compatible para inicializar el Videochat, porfavor utilice Google Chrome. Si no lo tiene instalado puede descargarlo de aqui</p>\
                <a href='https://www.google.com/chrome' target='_blank'>Google Chrome</a>\
            ";
        }
    }
    function CerrarVideochat(){
            /*document.getElementById("inputURL").style.display="none";
            document.getElementById("BotonAbrir").style.display="none";
            var win = document.getElementById("videoWindow");
            win.setAttribute("src","");
            win.style.display="inline";
            document.getElementById("BotonCerrar").style.display="inline";*/
            var winVideochat = document.getElementById("videochatwindow");
            winVideochat.innerHTML = '\
            <p style="font-size:30px;">Clave Videochat</p>\
            <input type="text" id="inputURL" style="margin-left:40px;" />\
```

5.1 Captura de imagen, segmentos alterados y comentados

Inicialmente, generé un respaldo completo de la base de datos como archivo SQL y a continuación hice una réplica de ella con un nombre diferente. La intención de tener la segunda fue estar en posibilidades de truncar la base de datos original y de cambiar la configuración de la plataforma, a fin de que ésta se conectase a la nueva base de datos para pruebas, antes de pasar a la fase tres.

En esta fase realicé igualmente las pruebas de depuración de la plataforma; como la base de datos a la que conecté de momento estaba en blanco, toda la prueba se realizó desde el principio, empezando por la creación de usuarios, la creación de políticas de acceso, las bitácoras y las pruebas de conversación en línea, el seguimiento de casos, el envío de documentos etcétera. Con la retroalimentación obtenida, fui afinando el código de la plataforma y corrigiendo errores imprevistos. Una vez realizadas las correcciones y terminado el proceso completo, se volvió a configurar la plataforma para conectarse a la base de datos original y limpié la información de la base de datos corrupta o inservible.

La fase final consistió en subir los archivos corregidos de la plataforma al servidor, eliminar la antigua base de datos en el servidor y cargar la nueva con los datos corregidos. Después realicé nuevamente una revisión profunda para cerciorarme de que todo funcionara según lo planeado, porque de vez en vez, entre las diferentes versiones de PHP

y sus configuraciones en cada servidor, pueden darse procesos que generen errores o que funcionen de manera diferente (ver figura 5.2).

5.2 Captura de pantalla de la plataforma

## Resultados

Entregué la plataforma a la empresa con las correcciones realizadas y probadas, la duración de este fue de 3 semanas en noviembre del 2015; también remplacé los archivos modificados con la versión que estaba dentro del servidor con el que la empresa da servicio. Pedí al personal que sometiera la plataforma a un periodo de pruebas a fin de eliminar los errores que pudieran haber quedado. Además, ante cualquier eventualidad futura, conservé un respaldo de la última versión de la plataforma. Al finalizar el proyecto la plataforma quedó lista para la reanudación de los servicios, la base de datos lista y depurada y el acceso y sus funcionalidades operando como se deseaba.

## Conclusiones

Revisar, dar servicio y actualizar plataformas que se han elaborado colectivamente en épocas distintas plantea sus complejidades. La parte más difícil y laboriosa no es hacerse una idea de la estructura original y su funcionamiento, sino ubicar las partes añadidas o modificadas y rastrear y detectar los problemas que éstas ocasionan en la operación general. Arreglar el conjunto supone hacer una auténtica reingeniería de él, volverlo a concebir en su totalidad y asegurarse de que funcione perfecta y armoniosamente.

# VI. Desarrollo de sitio web “Innova y Renueva. Asociación para el desarrollo, A.C.”

## Objetivo

Generar un espacio para la divulgación de información acerca de una asociación civil sin fines de lucro, sus objetivos, actividades y proyectos. Además, publicitar la venta de productos elaborados por las comunidades indígenas y campesinas apoyadas por ella, así como difundir la organización de eventos cuyo fin es allegarse recursos y, finalmente, lograr una cobertura pública mayor a través de las redes sociales.

## Antecedentes

Innova y Renueva. Asociación para el Desarrollo, A.C. es una organización enfocada a brindar apoyo a las comunidades indígenas o campesinas más pobres o marginadas. Esto lo canalizan a través del fomento a la producción de aceites esenciales y de la realización de otros proyectos. En el momento en que trabé contacto con ellos, sólo recurrían a la publicidad a través de redes sociales: en una cuenta de Facebook subían información y elementos visuales para difundir sus actividades.

## Descripción del problema o contexto de la participación profesional

Como se ha señalado, Facebook era el único medio que tenía la asociación para contactar al público, pero no disponía de otros recursos o medios de alcance mayor ni de comunicación con los posibles interesados en sus proyectos y actividades.

El responsable o representante no tenía mucha experiencia ni desplegaba una actividad significativa en su red social. Así que, luego de la entrevista con él, sugerí la elaboración de un sitio web, visualmente grato, intuitivo y fácil de navegar. En materia de imágenes, el responsable sólo disponía de las incluidas en un Power Point, que fueron las que aproveché para el diseño visual. Igualmente, le recomendé ampliar sus redes sociales a través de la creación de una cuenta en el sitio web de Twitter, donde pudiera incluir información y fotografías de sus actividades.

Una vez que llegamos a un acuerdo fui el encargado de analizar, investigar y desarrollar todo el proyecto; asimismo fui responsable de la dirección, administración de tiempo de vida y cálculo de costos.

## Metodología utilizada y participación profesional

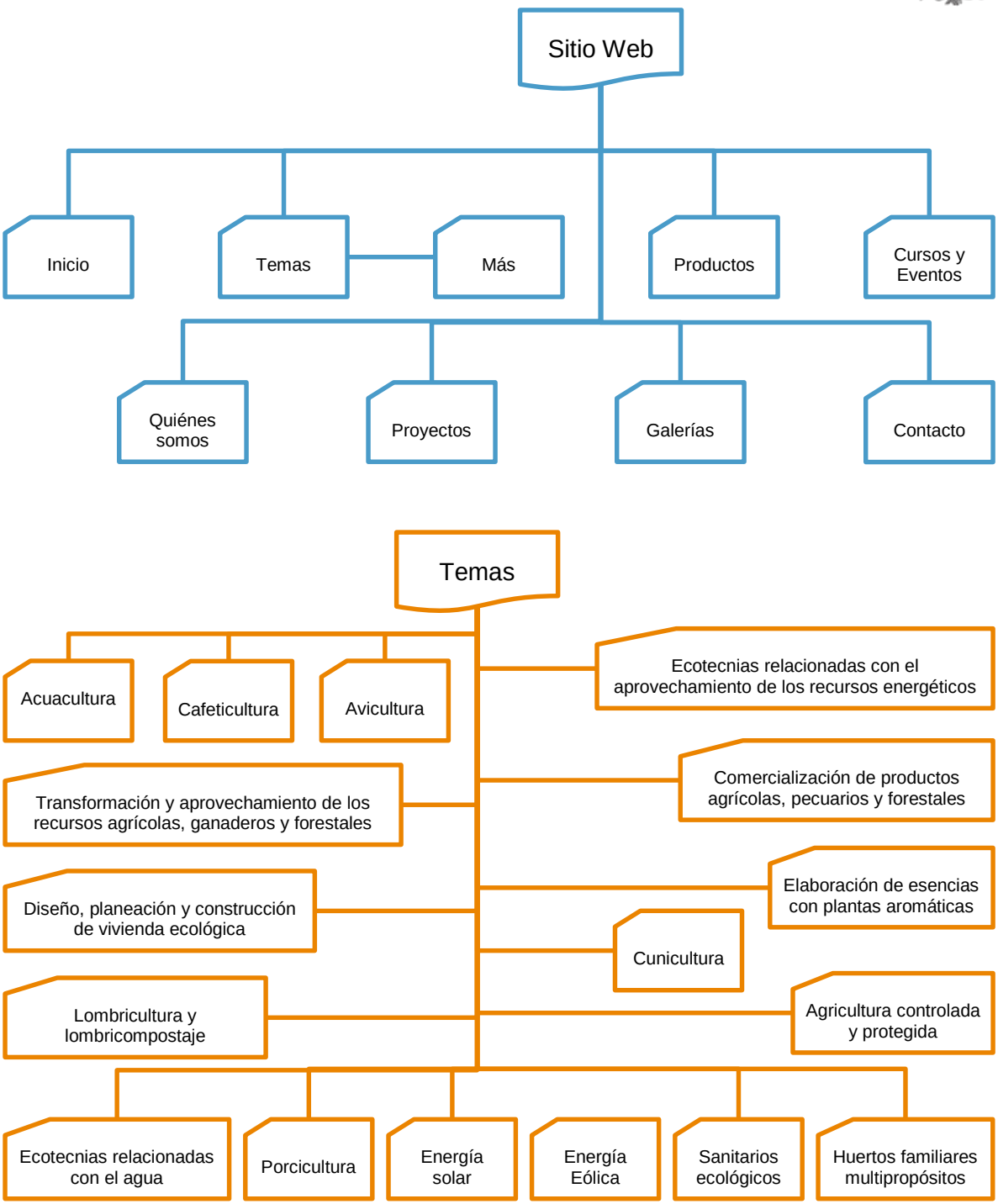
La asociación no tenía ningún servicio de hosting contratado, dominio registrado ni elementos para un sitio web. Por ello mi primer paso fue idear y estructurar un plan para la creación y desarrollo de un sitio web personalizado que cubriera las expectativas y requerimientos del representante de la asociación en materia de publicidad y difusión.

En la primera fase –la de planeación— realicé el análisis general, considerando los requisitos que debían ser cubiertos, determiné las oportunidades y exploré las perspectivas.

Planteé el diseño para que el visitante del sitio tuviera una experiencia de navegación intuitiva y sencilla; desarrollé e interpreté la estructura con estricto apego a las necesidades de mi cliente y con respeto a sus ideas e iniciativas, según su viabilidad. Finiquitada la fase de generación de ideas, planes y diseños, hice los cálculos de costos para cada fase del proyecto, que el cliente aceptó.

Una vez teniendo a mano la información que habría de incluirse en el sitio web, hice fragmentación de temas, cada uno de los cuales figuraría en una página del sitio; así quedaron varios bloques, cuya función era describir en lo particular las finalidades, actividades, proyectos y productos de la asociación. Este proceso de separación permitiría al visitante acceder con facilidad y sencillez a la información en la que estuviera interesado.

Al final, la estructura quedó con la jerarquía mostrada a continuación en la figura 6.1:




6.1 Mapa jerárquico del contenido de Innova y Renueva

Elaboré el diseño con un grupo de colores conformados por café, beige y verde; para dar un ambiente más familiar u hogareño, utilicé imágenes de trabajos desarrollados por las comunidades a las que apoya la asociación. En la parte visual

opté por generar un encabezado que abarcara todo el ancho de la ventana y pudiera ser redimensionado, manteniendo siempre centrados y ordenados sus elementos. Dentro de dicho encabezado se incluye el menú de navegación que igualmente abarca todo el ancho de la ventana. Éste se diseñó con propiedades, como el despliegue de submenús al momento en que el cursor se posa sobre ciertos temas.

El cuerpo se dotó de un tamaño determinado cuyo ancho es invariable, esto en consideración a que podría resultar incómodo para el visitante reordenar los textos internos según el tamaño de la ventana. Y como esta sección de la página es fija, lo primero era mantenerlo centrado, antes de poner un fondo que complementara las áreas sobrantes a los costados del cuerpo. Por último, el pie de página, al igual que la cabecera, abarcó todo el ancho de la ventana y la información que despliega esta sección está igualmente centrada.

En este sitio web utilicé lenguaje de estilo y de programación de lado del cliente para darle calidad visual y simplificar un poco el código generado que se desplegará para el cliente. Creé un único archivo CSS a fin de reducir al máximo la repetición de código para todos los elementos que se reiteran en todas las páginas, como el menú de navegación y otros elementos. De esta manera, solo es necesario enlazar la página HTML con el archivo de diseño para cargar las propiedades de los elementos que se representarán en la página.

En la segunda fase –de desarrollo– generé el código y programé, para lo que empleé lenguaje de etiquetas HTML, hoja de estilo CSS, lenguaje de programación lado de servidor PHP, lenguaje de programación lado del cliente JS y librerías para la construcción la carga de información de redes sociales y envío de correos HTML. Creé asimismo un directorio para empezar con el nombre del dominio que llevaría como primera opción (innovayrenueva) y guardé en este directorio todos los archivos que fui generando, tanto por un principio de orden, como para evitar que se mezclaran con otros archivos.

En cuanto a la calidad visual, generé un único archivo de hoja de estilo con los diseños generales que compondría todo el sitio web y así, en cada página de éste solamente se requeriría agregar la referencia a la hoja de estilo por medio de un enlace para cargar todas las propiedades. Esto fue una forma de simplificar y de evitar repetición de código que se va a utilizar en más páginas; en cuanto a los elementos cuyas condiciones particulares solo se aplicarían en una página determinada, agregué sobre el mismo archivo HTML en una etiqueta “style” y puse las propiedades que necesitaba utilizando atributos de

ID, clase o tipo de elemento. Estas acciones facilitaron el mantenimiento de un orden durante el desarrollo e impidieron los comportamientos inesperados de códigos de diferentes partes.

Con miras a aprovechar al máximo los recursos y a reducir los errores y redundancias en las páginas, en este sitio web diseñé pequeños scripts para la generación de código HTML que se despliegan en múltiples páginas del sitio, así, resultaría sencillo y rápido agregar o modificar información que va a afectar a todas las páginas. Además, esta opción que me ahorró tiempo a la hora de escribir y modificar la información, pues de otra forma me habría visto precisado a hacerlo página por página, aparte de la posibilidad de causar posibles errores en alguna o algunas de ellas, que después habría que probar en la fase de depuración. Solo hay que generar el código script y las etiquetas en cada página para ejecutarlo; tratándose de un código HTML que se va a insertar en alguna parte del sitio web, es importante que el script se ejecute después de que todo el sitio web ha sido cargado, porque sólo así el script puede encontrar el elemento en el que va insertar la información necesaria. Y para asegurar que tal condición se cumpla opté por llamar el archivo al final del contenido dentro de la etiqueta “BODY” al igual que la función; esta parte la detallaré más adelante.

En todas las páginas del sitio utilicé un ancho de mínimo de 1028px echando mano de la propiedad de ancho mínima (“min-width”) para el elemento “BODY”, para cuyos fines me basé en los datos estadísticos del sitio web “w3schools”, que trata sobre las resoluciones de pantallas que manejan los visitantes.

La totalidad de las páginas contiene una única estructura interna conformada por un elemento DIV que tiene un ID llamado “envoltura”. Éste sería de suma utilidad al momento de poner un pie de página al final de cada página del sitio y nos daría control del despliegue de los objetos con relación a la proporción de la ventana del navegador. Dentro de ésta tendremos una cabecera y el cuerpo de la página; la cabecera es un elemento “DIV” con su ID para asignarle las propiedades de estilo del archivo CSS al elemento. En el interior de este elemento hay otros dos elementos hijos, uno de ellos es un elemento “DIV” con ID de “logo” que contiene en su seno un elemento de imagen del logotipo de la asociación, el otro es otro elemento DIV que tiene un ID de “nav” vacía hasta que se ejecute el script que insertará la información. Este elemento forma el menú de navegación para que el visitante pueda pasar de una página a otra. A continuación hay un elemento DIV con ID llamado “cuerpo”, que contiene toda la información referente a la página que desplegará el visitante;

dentro de éste, cada página tiene elementos diferentes que se mostrarán en pantalla. Le sigue otro elemento DIV con un ID llamado “pie” que, al ejecutarse el script, sobrescribirá con información acerca de la asociación. Finalmente, los últimos dos son dos elementos script: el primero cargará el archivo y sobrescribirá en los elementos antes mencionados para su correcto funcionamiento y el segundo está destinado a la ejecución de una función que detectará y modificará el hipervínculo correspondiente a la página en la cual se encuentre el cliente, deshabilitándolo y cambiando su aspecto para resaltar en donde está (ver figura 6.2).

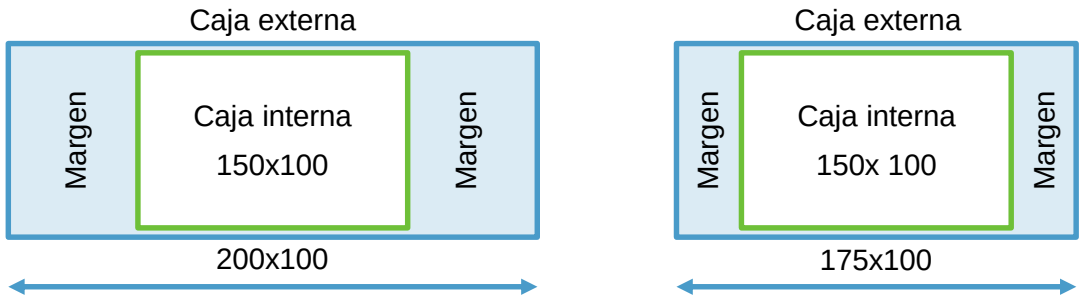
```
<body>
  <div =”envoltura”
   <div id=”cabecera”>
     <div id=”logo”><img …/></div>
     <div id=”nav”></div>
   </div>
   <div id=”cuerpo”>
     …
   </div>
  </div>
  <div id=”pie”>
   …
  </div>
  <script elements.js></script>
  <script>
   active(“inicio”);
  </script>
</body>
```

6.2 Código HTML del archivo Index

Una vez descrita la estructura general de las páginas del sitio web, pasaré a detallar el funcionamiento particular de cada una, igualmente explicaré a qué objetivos obedecen y daré las razones de haber tomado determinadas opciones en concreto.

Que la cabecera sea redimensionable significa que los objetos pueden cambiar su posición, dependiendo de sus propiedades. En este caso y aunque la imagen que forma el logotipo de la organización podía moverse, era importante que no se despegara mucho del resto del contenido; también había que prever que el menú de navegación no se extendiera a todo lo largo de la ventana. Para lograr que la cabecera se extendiera a lo largo del navegador pero su contenido se mantuviera centrado, elaboré un juego de cajas con márgenes automáticos. Para mayor precisión, la idea era generar una caja que adoptara el tamaño de la ventana y que llevara en su interior otra caja, pero el ancho de ésta es estático y cuenta con la propiedad de márgenes automáticos. El resultado será que la caja interna se mantendrá centrada, generando los márgenes laterales de tamaño variable que

rellenarán el espacio que medie entre la caja interna y la externa; así, el elemento será redimensionable pero su contenido se mantendrá centrado (ver figura 6.3).




6.3 Ilustración del uso de cajas combinadas redimensionable y estática para centrar elementos

Las ventajas de utilizar esta técnica son:

- Su construcción es sencilla
- Es exacta
- Se le puede dar toda la altura deseada
- Se puede tener más control y hacerlo encajar con otros elementos de dimensiones determinadas

Sus desventajas son:

- No se pueden utilizar propiedades como flotante (“float”) o posición (“position”) en la caja interior ni exterior
- Los elementos internos pueden salirse del contenedor interno
- Las cajas interna y externa tienen que tener la propiedad de tipo de caja (“display”) como bloque

Ahora bien, en la creación de estos elementos lo primero que hice fue generar un elemento DIV que llamaré “padre” y que tiene un ID con el nombre de “cabecera”. A éste le asigné en un archivo de estilo, un fondo de color y un efecto de sombreado para darle un diseño vistoso; por ser un elemento DIV por defecto su ancho es del 100% y se muestra en forma de bloque, por lo que no es necesario hacer ninguna nueva asignación. Esto genera el cuadro que se redimensionará respecto de la ventana en su interior. En el seno de dicho cuadro generé otro elemento DIV con un ID llamado “logo”; éste recibe las propiedades del archivo de estilo: un ancho de 1280px (el mismo que el del cuerpo de la página) y en los

márgenes el valor de auto para que generé espacios iguales y se mantenga centrado. Dentro de este elemento hijo agregué el elemento IMG que mostrará la imagen del logotipo de la asociación cargado a la derecha, lo que impide que sobresalga del ancho del resto de la página. Aunque no fue un recurso que hubiera empleado aquí, no está de más mencionar que, en caso de que se quisiera agregar un relleno (“padding”) al elemento hijo para agregar texto, por ejemplo, habría que efectuar algunas operaciones adicionales, ya que esta propiedad alterará las dimensiones de la caja hijo. La primera opción es agregar a la caja hijo la propiedad de “box-sizing” con el valor de “border-box” para que el relleno agregado al hijo sea interno y no modifique el tamaño. El problema con esta opción es que la propiedad mencionada pertenece a la versión de CSS 3 y no todos los navegadores pueden interpretarla. La segunda posibilidad es hacer el cálculo de la dimensión de la caja hijo con su relleno. Toda vez que esta propiedad aumenta las dimensiones de ambas, hay que hacer una resta de la dimensión con las de un relleno dado. Por ejemplo, si nuestra caja hijo es de 1280px y queremos agregarle un relleno de 10px en todos sus lados, la caja se deberá modificar a un ancho de 1260px (si asignara una altura determinada también se deberán restar los 20px) porque el relleno agregará 10px un lado, 10px de otro, 10px arriba y 10px abajo. Así, los navegadores generarán en este orden: 10px de relleno izquierdo, 1260px de ancho de la caja hijo y otros 10px de relleno derecho, que dan los 1280px. Dicho método es compatible prácticamente con todos los navegadores.

El menú de navegación cuenta con un ID nombrado como “nav”; por cuestiones estéticas éste será del tamaño del ancho de ventana, con un fondo de color, que funcionará como nodo padre. Y aunque no tiene nodos hijos en el código original, con la ayuda de un archivo script, generará el código que se implanta en el elemento; el nodo hijo recibirá de inmediato las propiedades para mantenerse centrado, al igual que en el ejemplo anterior. Como ya mencioné es importante que la ejecución del script sea posterior a la carga de todo el contenido HTML para que pueda encontrar el elemento que va actuar. Para lograrlo, creé un archivo JS, cuya primera operación es localizar el elemento padre; esto se puede hacer utilizando la siguiente instrucción “document.getElementById([Nombre de la ID])”. La instrucción busca localizar el elemento que tiene el nombre escrito –en este caso “nav”– una vez localizado, hay que guardarlo en una variable para generar el código que se va a agregar dentro del elemento. Y para este efecto se emplea la instrucción “innerHTML” que sobrescribirá todo lo que haya dentro del elemento padre. El código queda de esta manera en la figura 6.4:

```
document.getElementById("nav").innerHTML+='\
<ul>\
  <li><a href="index.html">inicio</a></li>\
  <li><a href="nosotros.html">quienes somos</a></li>\
  <li>\
   <ul>\
     <li><a href="temas.html#1">diseño, planeación y construcción de vivienda ecológica</a></li>\
     <li><a href="temas.html#2">ecotecnias relacionadas con el agua</a></li>\
     …
    <li><a href="temas.html#17">comercialización de productos agrícolas, pecuarios y forestales</a></li>\
   </ul>\
   <a href="temas.html">temas</a>\
  </li>
  …
  <li><a href="contacto.php">contacto</a></li>\
</ul>\';
```

6.4 Código JS para la generación automatizada del menú de navegación

Si se desea preservar contenido que ya estaba en el interior de nodo padre, basta con agregar la operación “+=” a la instrucción innerHTML para que sea agregada al final de lo que ya estaba previamente. Una vez generado el archivo script con el código HTML que va a dar como resultado el menú de navegación, lo siguiente fue utilizar las propiedades de estilo para darle propiedades visuales. El menú fue realizado como una lista (UL) y los elementos de la lista (IL); estos últimos pueden poseer anclas (elementos A) para los hipervínculos, o bien, otra lista para los submenús desplegables. A fin de que el menú se desplegase de forma horizontal fue necesario dar propiedades a los elementos de la lista, objetos de la lista y las anclas. En cuanto al elemento de lista hubo que empezar por quitarle todas las propiedades que trae por defecto y que son de la lista; para ello utilicé la propiedad “list-style” que suprime los símbolos de la lista, luego, sobre cada elemento de lista es menester cambiar el tipo de caja que tiene como predeterminado en bloque en la misma línea (“inline”). De esta manera, la lista se presentará de manera horizontal, juntando un elemento tras otro; además, dentro de ella existen dos opciones. La primera, aplicable a los objetos de lista, es agregar el anclaje que, en este caso, se trató de darle el contenido del elemento y el atributo a donde va a transportar al cliente. Las propiedades que considero mínimas son: el relleno (padding) y los márgenes laterales (margin); es conveniente asignar sobre todo el relleno a los elementos de anclaje, pero no a los objetos de lista; de esta manera el relleno generado también tendrá la propiedad del ancla, lo que facilita al visitante el acceso para realizar la acción (ver figura 6.5). Esto independientemente de que hace más sencilla la adición de otras propiedades como el color de fondo y el cambio de propiedades bajo ciertas condiciones.

```
#nav > ul{
  list-style:none;
  …
  padding:0px;
}

#nav > ul > li{
  display:inline;
}

#nav a{
  …
  display:inline-block;
  padding:0 10px 0 10px;
  margin:0 5px;
  …
}
```

6.5 Código CSS para asignar propiedades al menú de navegación

La segunda opción que contiene un objeto de lista es otra lista. En ésta, la siguiente lista funciona como un submenú del objeto, y tanto la segunda lista como la principal poseerán objetos de lista y anclas, y aunque no se hizo así en el caso presente, podría incluir más listas. A fin de que la segunda lista se despliegue solamente cuando el puntero está encima del elemento hay que jugar con la propiedad del tipo de caja (display) y el selector “sobre” (hover), además de cambiar las propiedades, como el tipo de caja para los objetos de lista dentro de la segunda lista, que deberán figurar como bloque a fin de que se acomoden de manera vertical; también las propiedades de posición para que aparezcan acomodados debajo sin que alteren el orden del menú principal, y, finalmente, el nivel de posicionamiento en profundidad (z-index). Para conseguirlo hay que dar las propiedades a las listas secundarias y, a tales efectos, escribí el selector lo escribí de la siguiente manera: “#nav li > ul” con lo que le ordeno que busque todos los elementos de lista (UL) que están contenidos inmediatamente en un objeto de lista (IL) dentro del elemento que tiene un ID con el nombre de “nav”. A estos elementos les asigné las propiedades de tipo de caja (display) como ninguno, posición absoluta, nivel de profundidad (z-index) por encima de los demás; en este caso bastó con el valor de 1, la primera propiedad descrita es para que los elementos no se muestren de manera predeterminada, sino que nosotros lo haremos aparecer cuando una condición se cumpla, como veremos más adelante. La propiedad de posición como absoluto impedirá que altere a la lista principal, dará la apariencia de que es un elemento fuera de la lista principal, que no se verá afectado por las propiedades de dimensión que pudiera generar esta lista sobre la principal. Además esto provocará que las propiedades del padre puedan ser aplicadas también a este elemento, lo que es importante

para que más adelante se puede mostrar y ocultar bajo las condiciones que le pongamos. Y, finalmente, el nivel de profundidad nos asegurara que cuando aparezca este elemento no abrirá un elemento que se muestre por encima, dándonos prioridad para visualizar el submenú sobre el menú principal.

Por otro lado, para que el puntero cause la acción de mostrar las sublistas cuando el cursor este posicionado utilicé el siguiente selector: “#nav li:hover > ul”. Esta forma obliga a que un evento, aparentemente ajeno, afecte a un elemento, aunque en realidad es una forma simplificada para que dos elementos detonen y afecten a un único elemento. Es posible comprobar que el selector “li:hover” se detonará cuando el cursor este encima del objeto de lista (IL) que contendrá a la sublista (UL), lo que hará que aplique las propiedades bajo la condición “encima” que, en este caso, consistirá simplemente en mostrar la sublista cambiando el tipo de caja (display) de ninguno a en línea (inline). Ahora aparentemente si movemos el cursor al submenú se pensaría se salió ya del área del objeto de lista para entrar a la de sublista, lo que, en teoría, debería hacer desaparecer este elemento de nuevo. Sin embargo, aunque visualmente no sea aparente esta sublista, sí está dentro de este objeto de lista, por tanto, todos los elementos dentro de ella deben verse afectados por el selector anterior y como resultado inmediato estas sublista mantendrán estado de “encima” sobre el objeto de lista. Un detalle menor que agregar es que a este objeto de lista le coloqué un ancla inmediatamente después de la sublista, a fin de que en el menú principal se pudiera visualizar un objeto que detone el selector anteriormente mencionado y que el ancla que también está dentro del mismo elemento cumpla con la condición, al igual que la sublista (ver figura 6.6)

```
#nav li > ul{
  display:none;
  padding:20px 0px;
  border-radius:0 0 20px;
  box-shadow:5px 10px -5px #1F190F;
  background:#AB895A;
  position:absolute;
  margin-left:0px;
  margin-top:18px;
  z-index:1;
}

#nav li:hover > ul{
  display:inline;
}

#nav li > ul > li {
  display:block;
  margin:10px;
}
```

6.6 Código CSS para la creación de submenús en el menú de navegación

Los detalles restantes tratan sobre propiedades que le dan a los elementos del menú principal y sus submenús un diseño más intuitivo y de mayor calidad visual. Por ejemplo, utilicé las anclas para incrementar la calidad visual, las propiedades de color, la decoración de texto y la transformación de texto y fuentes. Además con el uso del selector “encima” (hover) en las anclas dentro del menú “#nav a:hover” asigné otras propiedades visuales como un borde inferior que aparecerá cuando el navegador pase por encima del ancla (ver figura 6.7).

6.7 Capturas de pantalla del menú de navegación y submenú

En todas las páginas del sitio el cuerpo está formado por un elemento DIV con un nombre de ID llamado igualmente “cuerpo” y en él agregué toda la información que correspondiera a cada página. Su diseño es muy simple comparado con los otros elementos de la página y lo que busqué aquí fue que resultara estético, de contenido agradable y cómodo a la vista del visitante, pero sin ser complicado a fin de evitar la generación de

errores por cuestiones de despliegue en otros dispositivos, por ciertas condiciones de la ventana que pudieran ocasionar un desacomodo inesperado, o errores en la visualización.

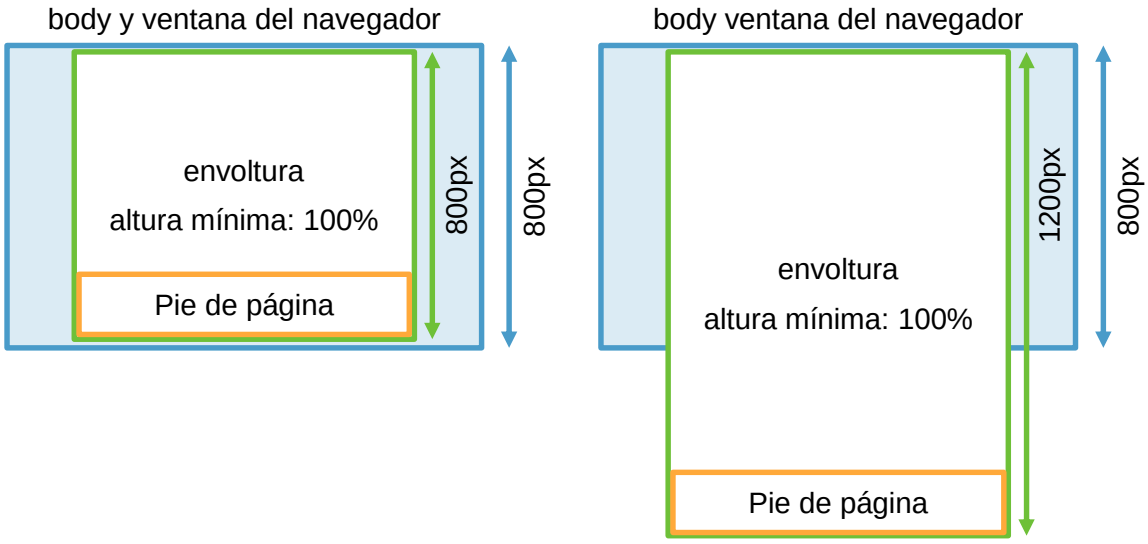
A este respecto, lo primero que consideré fueron las combinaciones de colores, los espacios entre elementos y la facilidad de lectura. Más allá de la parte estética del diseño, el objetivo principal era lograr que los visitantes pudieran obtener, de manera rápida, cómoda y sencilla, toda la información de su interés. Para este elemento agregué las propiedades ancho (width) en 1210px, altura mínima (min-height) al 100%, márgenes (margin) con valor automático, relleno (padding) con los valores 0px 35px 30px y 35px. Este elemento es un ejemplo de lo que había mencionado anteriormente: para que el ancho fuera de 1280px hice la suma de los rellenos laterales con la del ancho de elemento 35 + 1210 + 35; el mínimo de altura lo asigne al 100% para conseguir un pie de página. Con ayuda de un código JS se modifica esta última propiedad, dependiendo del tamaño de la ventana, para lograr que el elemento siempre cubra todo el contenido y que siempre aparezca al final un pie de página. El cálculo del nuevo valor de altura mínima que tendrá el elemento con ID “cuerpo” lo obtendrá del tamaño de la ventana menos las alturas de los elementos “cabecera” y “pie” y siempre que no sea menor a cero. Al tratarse de la propiedad de altura mínima si el contenido causa una altura mayor a la asignada a la propiedad, ésta la respetará (ver figura 6.8).

```
…
document.getElementsByTagName("BODY")[0].onresize = function(){cuerpo()};
cuerpo();

function cuerpo() {
  var contHeight = (document.getElementsByTagName("BODY")[0].clientHeight - 260);
  contHeight = (contHeight < 0) ? 0 : contHeight;
  contHeight += "px";
  document.getElementById("cuerpo").style.minHeight = contHeight;
}
…
```

6.8 Código JS para el cálculo de la altura del elemento “cuerpo”

Aunque hay ciertos detalles que cuidar con el pie de página, pues en un sitio web cada elemento se va acomodando inmediatamente después del que le antecede, así que si la ventana del navegador es mayor que todos los elementos que forman la página, dejará un espacio en blanco en la zona inferior. Y esto es un inconveniente para nuestros fines, toda vez que el pie de página quedaría entonces arriba de esta parte o bien colocarse en algún punto específico de la página, lo que tampoco deseamos. El problema no se resuelve recurriendo a la propiedad posición ya que, dependiendo del tamaño del navegador, tendría

que generar una posición diferente y eso lo hace todo mucho más complicado. Una solución muy sencilla es simplemente generar un elemento que tenga la altura de navegador, a fin de que éste desplace el pie de página lo suficiente para asegurar que siempre estará al final y no dejará ningún un espacio ulterior. Éste es nuestro elemento DIV con ID llamado “envoltura” (ver figura 6.9).




6.9 Ilustración de un “Sticky Foot” usando el elemento envoltura con el pie de página

Importante es señalar la necesidad de agregar la propiedad de altura (height) a los elementos HTML y BODY para que estos tomen la altura del navegador y para que el elemento con ID “envoltura” pueda generar también la altura pero como mínima; así, en el caso de que el contenido sea más amplio que la altura del navegador, ésta podrá ir aumentando. Si no se agrega esta propiedad en los elementos antes dichos, el elemento que desplaza el pie de página generará una altura de 0px o del tamaño del contenido que tenga, ignorando la propiedad declarada.

Después de esto, agregué la propiedad de margen bajo (margin-bottom) con la altura que va a tener el pie de página, pero como valor negativo. Tal arreglo provocará que el pie de página se sobreponga en la parte baja de este elemento, que en realidad será el espacio que más adelante crearemos como un elemento vacío, reservado para el pie de página. Además, es recomendable que al elemento body se declare la propiedad de margen con un valor de 0px, a fin que el elemento con ID “envoltura” encaje perfectamente en las dimensiones de la ventana. De forma contraria, este elemento se verá afectado por el

margen del elemento body y será desplazado las unidades que tiene por defecto, lo que sería causa de que presentara pequeños e innecesarios desfases.

Una vez creados los elementos y asignadas sus propiedades, lo siguiente es crear un elemento vacío inmediatamente al final y dentro del elemento con ID “envoltura” para crear un pie de página que se mueva convenientemente, sobreponiéndose al final del elemento antes dicho. Para ello hay que generar el espacio del mismo tamaño que el del pie de página, a fin de que el contenido que ingresemos en la envoltura no sea tapado por éste. Para hacer lo primero creé el bloque vacío usando el selector después (:after) y dentro de él utilicé las propiedades contenido (content) –que está vacío–, tipo de caja (display) –mostrándolo como bloque con la intención de que sea a todo lo ancho— y altura (height), del mismo tamaño o superior al del pie de página. Lo siguiente era crear el pie de página, para lo que recurrí a un elemento DIV y un ID con el nombre de “pie”. Este elemento recibirá las propiedades de altura, que serán al gusto y en mi caso fueron de 50px y en conjunto, con el margen negativo en el elemento con el ID de “envoltura”, ocasionarán que el pie de página se renderizará sobre el área que creamos como vacío al final del elemento que sirve de envoltura (ver figura 6.10).

```
…
#envoltura{
  min-height:100%;
  overflow: auto;
  margin-bottom: -50px;
}

#envoltura::after{
  content:"";
  display:block;
}
…

#envoltura::after, #pie{
  height:50px;
}
…
```

6.10 Código CSS para el elemento envoltura

El antes descrito es uno de muchos métodos para generar lo que se conoce con el nombre de “Stickyfoot”. Personalmente me gusta emplearlo, puesto que no requiere ninguna propiedad de CSS3, aumentando la compatibilidad con navegadores más antiguos.

Ahora bien, el pie de página, al igual que el menú de navegación, va cargar el contenido desde un archivo script, que buscará el elemento que tenga una ID con el nombre

de “pie” i; con la instrucción innerHTML sobrescribirá sobre ese elemento con un código HTML asignado, de esta manera automatizamos también la información mostrar y evitamos errores como en el menú de navegación.

Había señalado antes que en mi estructura del sitio web tenía dos elementos script; sobre el primero ya describí su función, en cuanto al segundo, es simplemente un llamado a una función con el nombre de “activaNav()”. Su objetivo es localizar la página en la que se encuentre el visitante, deshabilitar ese hipervínculo del menú de navegación y cambiar algunas propiedades visuales para destacar su ubicación. Esto se consigue en un proceso que consta de dos partes: la primera, ejecutada por un script, se hará cargo de barrer todo el contenido del menú de navegación en busca del elemento y la segunda es un conjunto de propiedades especificas en el archivo CSS.

La función activeNav recibe como parámetro una cadena de texto que tiene que ser el contenido del elemento del ancla que hace referencia a alguna de las páginas. Por ejemplo, el hipervínculo de “Cursos y Eventos” es el contenido del ancla que hace referencia a la página “cursosyeventos.html” en el menú de navegación; así, cuando el visitante esté en esa página, la función se ejecutará como activeNav(“Cursos y Eventos”). Lo primero que hace la mencionada función es convertir la cadena de texto que entrará en minúsculas, para evitarnos conflictos de comprobación a la hora de buscar el elemento con el menú de navegación. Lo siguiente es localizar el menú de navegación, para eso utilicé le instrucción “getElementById()” y como parámetro ingresé la cadena “nav”, recordando que tal es el ID que le asignamos a nuestro elemento que contiene el menú de navegación a todas las páginas. Una vez capturado el elemento, con ayuda de la instrucción “getElementByTagName()” y como parámetro la cadena “A”, buscaremos todos elementos de ancla que existen dentro del menú de navegación, lo que nos dará un arreglo de elementos de ancla. Una vez obtenido este arreglo solo restaba recurrir a instrucciones de ciclo y condiciones con la sentencia “mientras” (while) y “si” (if) respectivamente, así fui barriendo uno por uno, comprobando el contenido en cada uno, hasta que empaté el parámetro ingresado con el contenido del elemento. Ya que se cumple la condición, el siguiente paso es agregar al objeto de lista que contiene el ancla el atributo “activo” para que en la segunda parte en el archivo de estilo se le agreguen las propiedades necesarias. En caso de que no encuentre una condición que cumpla, simplemente no asigna a ningún elemento el atributo, como por ejemplo los elementos del submenú de la liga “Temas”.

Ahora para deshabilitar el elemento que a través del script le asigno, agregué en el archivo de estilo el selector de atributo sobre un elemento de objeto de lista, escribiéndolo de la siguiente manera “#nav li[activo]”. Este garantizará que a cualquier objeto de lista (LI) que posea el atributo “activo” y que sea un elemento dentro de este elemento con el ID “nav”, se le asignen las siguientes propiedades.

La primera es eventos del puntero (pointer-event) con en valor de ninguno (none); esta propiedad deshabilita la acción de responder cuando el cursor da clic sobre el elemento; como este elemento contiene el ancla, también lo desactivará, evitando que el usuario pueda hacer la petición de abrir el hipervínculo contenido en el elemento que cumpla con esta sentencia. Por cuestiones de orden visual agregué también la propiedad de cursor (cursor) con el valor de por defecto (default) sobre esta sentencia también, para que el cursor no cambie de icono al pasar sobre el elemento deshabilitado, dando la sensación de que no hay liga en esa sección. Ahora, para destacar en qué página está ubicado el visitante hice uso de un selector muy similar “#nav li[activo] a”, éste solo dice que en lugar de modificar el objeto de lista modificará el elemento de ancla. A dicho elemento solo le asigné las propiedades de tamaño de caja (box-sizing) con todas sus opciones -moz-box-sizing, -webkit-box-sizing y box-sizing con el valor de borde de caja (border-box) para que, al agregar la segunda propiedad, no cause desfases. La segunda propiedad que asigné fue la de borde de abajo (border-bottom) con los valores de 1px sólido y blanco; de esta manera lo que el usuario visualizará, será un botón en el menú de navegación que aparecerá subrayado dentro del cuadro y aun si pone en él el puntero no reaccionará ni visual ni activamente (ver figura 6.11, 6.12 y 6.13).

```
…
function activeNav(string){
  var active = string.toLowerCase();
  var nodoPadre =
document.getElementById("nav");
  var elemento =
nodoPadre.getElementsByTagName("a");
  var cont = elemento.length-1;

  while(0<= cont){
   if(elemento[cont].innerHTML==active){
   elementos = elemento[cont].parentNode;
   elementos.setAttribute("activo","")
   break;
  }
```

6.11 Código JS para la asignación del atributo “activo” en el elemento deseado

```
#nav li[activo]{
  pointer-events:none;
  cursor:default;
}

#nav li[activo] > a{
  -moz-box-sizing: border-box;
  -webkit-box-sizing: border-box;
  box-sizing: border-box;
  border-bottom:1px solid #FFF;
}
```

6.12 Código CSS para desabilitación de liga en elementos con atributo “activo”

6.13 Captura de pantalla de elemento con el atributo “activo”

Ahora, pasando al contenido de las páginas del sitio, la principal contiene información acerca de la asociación y sus diseños fueron tomados del archivo de estilo (como colores, tamaños, posiciones etcétera). En aras de ahorrarme un recuento largo y tedioso, no profundizaré mayormente en el tema, pues actualmente puede encontrarse una gran cantidad de información sobre él. En cambio, pasaré directamente a describir dos elementos particulares que integran esta página; se trata de unos componentes (plug-in) que sirven para agregar contenido de las redes sociales, en este caso de Facebook y Twitter. Estos sirven para cargar cierta información acerca de una cuenta de las redes sociales mencionadas y pueden desplegarlas en algún sitio web para que el visitante sepa qué actividad ha habido, en este caso, en las redes sociales de esta asociación.

6.14 Captura de pantalla de las extensiones para redes sociales

A este respecto, hay empresas que ponen a disposición de los usuarios código para insertar estos contenidos. Para la carga de información en Facebook se pueden encontrar estos segmentos de códigos y otras herramientas útiles en su sitio web https://developers.facebook.com/. No sobra alertar que para poder utilizar estas herramientas hay que tener una cuenta en su sitio web, ya que necesita saber qué cuenta va a generar dichas herramientas. Y a fin de no extenderme demasiado, sólo añadiré que, para agregar el complemento de Facebook utilicé algunos elementos DIV con nombres de ID y CLASES que implican respetar además un código script.

Ahora bien, el código de inicio es general, es decir los primeros componentes son código que no posee ningún dato particular del usuario que Facebook que va a desplegar. Sin profundizar mayormente, diré que los primeros dos bloques son la creación de elementos que necesita Facebook para preparar todo y para las ulteriores alteraciones; lo importante en realidad está en el último bloque, que será del que Facebook tomará atributos preestablecidos por la empresa para que procese información y dé una respuesta respecto de la información que buscamos. A continuación describiré brevemente cada atributo e indicaré para que se utiliza, aunque, como ya mencioné antes, se puede encontrar más información en la página de Facebook “https://developers.facebook.com/docs/plugins/page-plugin”.

- data-href: Indica a Facebook la URL de la página que va a cargar.
- data-with: Indica el ancho de píxel del complemento.
- data-height: Indica la altura del píxel del complemento.
- small-header: Opción de encabezado pequeño.
- data-adapt-container-width: Realiza un ajuste al ancho del contenedor en el que se encuentra.
- data-hide-cover: Opción que sirve para ocultar la foto de portada en el encabezado del complemento,
- data-show-facepile: Muestra las fotos del perfil cuando los amigos indican que les gusta el contenido.
- tabs: Indica qué pestaña va a mostrar: línea de tiempo, mensajes o eventos.

```
…
<div id="fb-root"></div>

<script>(function(d, s, id) {
 var js, fjs = d.getElementsByTagName(s)[0];
 if (d.getElementById(id)) return;
 js = d.createElement(s); js.id = id;
 js.src = "//connect.facebook.net/es_ES/sdk.js#xfbml=1&version=v2.4";
 fjs.parentNode.insertBefore(js, fjs);
}(document, 'script', 'facebook-jssdk'));
</script>
…

<div class="fb-page" data-href="https://www.facebook.com/innovayrenueva" data-width="230" data-height="300" data-
small-header="true" data-adapt-container-width="true" data-hide-cover="false" data-show-facepile="false"
tabs="timeline"><div class="fb-xfbml-parse-ignore"><blockquote cite="https://www.facebook.com/innovayrenueva"><a
href="https://www.facebook.com/innovayrenueva">Innova y Renueva, Asociación para el Desarrollo, A.
C.</a></blockquote></div></div>
…
```

6.15 Código para la implantación de “Facebook” como extensión.

Ciertamente, el complemento de Twitter es mucho más sencillo; la empresa pone a disposición de los usuarios una herramienta que genera widgets embebidos para ser insertados en páginas web. Esto pueden encontrarlo en https://publish.twitter.com/ . Lo único que pide esta herramienta es una URL de la línea de tiempo de Twitter que se desea agregar y unas cuantas opciones. Una vez finalizado, compartirá un código que solicita que se copie y pegue en la página en la que se desea que figure. En realidad, consta sólo de dos elementos: uno de ancla, con una clase que utiliza Twitter para identificar el elemento y un atributo de referencia que Twitter requiere para saber de dónde va cargar toda la información. El otro elemento es un script que toma del servidor de Twitter directamente para hacer todo el proceso. La empresa nos facilita y abrevia mucho el trabajo de insertar todas partes y procesos necesarios; lo que pasa con estas dos partes que forman el complemento es que gracias al script se ejecuta un código cuya tarea es localizar el elemento de anclaje con sus datos de clase y atributo tratarlos con el script de Twitter; después realiza una solicitud a los servidores de Twitter para recibir los datos y reemplazar el elemento de anclaje con un elemento de contexto de navegación animada (iframe) que contiene todos los datos para ser desplegados (ver figura 6.16).

```
…
<a class="twitter-timeline" href="https://twitter.com/innovayrenueva">Tweets by innovayrenueva</a>
<script async src="//platform.twitter.com/widgets.js" charset="utf-8"></script>
 …
```

6.16 Código para la implantación de “Twitter” como extensión

En la página de temas, el contenido está dividido en un menú lateral y el contenido informativo; para este diseño creé dos elementos DIV, uno con ID que tiene el nombre de “menu” y el otro de “texto”. El menú lateral contiene un conjunto de listas con anclas en cada uno de los objetos de todas las listas que apuntan a diferentes secciones de la página por medio de índices que asigné (ver figura 6.17).

```
<div id="menu">
  <h3 style="margin:10px 0; color:#FFF; text-align:center;">Menú</h3>
  <ul>
   <li>
    <a href="#1">· Diseño, planeación y construcción de vivienda ecológica.</a>
    <ul>
     <li><a href="#1.1">· Aprovechamiento del medio físico para el diseño de vivienda.</a></li>
     <li><a href="#1.2">· Revalorización de los diseños tradicionales de vivienda. vivienda vernácula.</a></li>
     <li><a href="#1.3">· Aprovechamiento de los materiales de la región para la construcción de vivienda.</a></li>
    </ul>
   </li>
   <li>
    <a href="#2">· Ecotecnias relacionadas con el agua.</a>
    <ul>
     <li><a href="#2.1">· Ollas de captación de agua de lluvia.</a></li>
     <li><a href="#2.2">· Cisternas de ferrocemento.</a></li>
     <li><a href="#2.3">· Construcción de trincheras.</a></li>
     <li><a href="#2.4">· Cosecha de agua de lluvia.</a></li>
     <li><a href="#2.5">· Lluvia sólida.</a></li>
     <li><a href="#2.6">· Bomba de mecate.</a></li>
     <li><a href="#2.7">· Aprovechamiento de aguas grises.</a></li>
     <li><a href="#2.8">· Filtros.</a></li>
    </ul>
   </li>
   …
</div>
```

6.17 Código HTML para la creación de menú lateral en la página de “Temas”

El otro elemento solo contiene la información con los índices donde va a realizar el salto de cada ancla y está puesto en el menú lateral a cada título del tema; aquí simplemente agregué in ID a cada etiqueta de título (<h1>, <h2>, <h3>, …) con el índice asignado (ver figura 6.18).

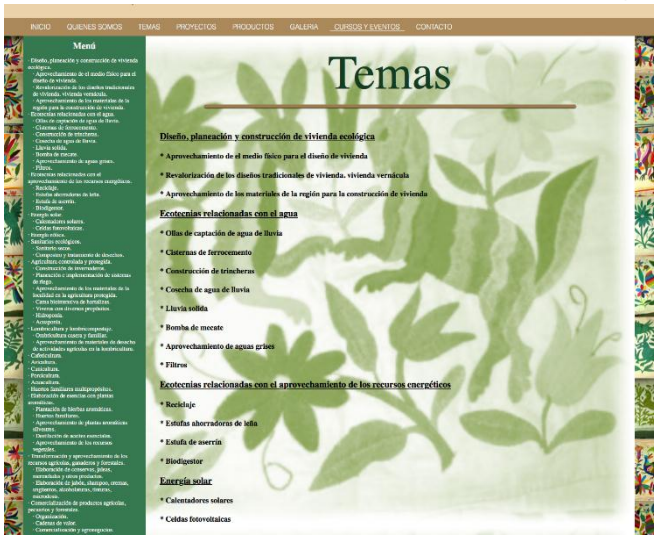
```
<div id="texto">
  <div>
   <h1>Temas</h1>
   <hr style="width:80%">
   <h3 id="1">Diseño, planeación y construcción de vivienda ecológica</h3>
   <h4 id="1.1">Aprovechamiento del medio físico para el diseño de vivienda</h4>
   <h4 id="1.2">Revalorización de los diseños tradicionales de vivienda. vivienda vernácula</h4>
   <h4 id="1.3">Aprovechamiento de los materiales de la región para la construcción de vivienda</h4>

   <h3 id="2">Ecotecnias relacionadas con el agua</h3>
   <h4 id="2.1">Ollas de captación de agua de lluvia</h4>
   <h4 id="2.2">Cisternas de ferrocemento</h4>
   …
   <h3 id="3">Ecotecnias relacionadas con el aprovechamiento de los recursos energéticos</h3>
   <h4 id="3.1">Reciclaje</h4>
   <h4 id="3.2">Estufas ahorradoras de leña</h4>
   …
  </div>
</div>
```

6.18 Código HTML donde se dirigirá por medio de la asignación de “ID”

Y sólo me resta describir las propiedades de estilo que agregué a estos elementos en particular. Para que ambos elementos se acomodaran a los lados a todo lo largo de la página, opté por utilizar la propiedad de posición en los dos; para el menú lateral utilicé el posicionamiento absoluto y con la información utilicé el posicionamiento relativo porque éste determinará la altura de la página y si fuera absoluto dicha altura no se reflejaría en el elemento padre. Lo siguiente fue dar a cada elemento las coordenadas; para el menú lateral (absoluto) utilicé la propiedad izquierda (left) con un valor de 0px y un ancho del 20% respecto del elemento padre (el elemento con ID llamado “cuerpo”). Para el texto asigné la propiedad de izquierda (left) con un valor de 20%, a fin de que se recorriera la distancia del ancho del menú lateral. Al tener éste la propiedad de posición absoluta, el elemento que contiene el texto partirá con las coordenadas de su izquierda desde el limite izquierdo del elemento padre, aun siendo su posición relativa; además, le asigné un ancho (width) del 80% para rellenar el espacio sobrante.

Una nota importante fue que al elemento padre en esta página le agregué particularmente la propiedad de posición relativa para lograr que crezca la altura correlativamente al elemento que contiene el texto. Y como por obvias razones este elemento es más amplio que el menú lateral, tiene que rellenar el espacio que sobre, generado por el elemento con el texto (ver figura 6.19).

```
#cuerpo{
  padding:0px;
  position:relative;
  width:1280px;
}

#menu{
  position:absolute;
  left:0px;
  width:20%;
  height:100%;
  background:#377950;
  overflow:scroll;
}

#texto{
  position:relative;
  width:80%;
  left:20%;
}
```




6.19 Código CSS y Captura de imagen del menú lateral de la página “Temas”

En la página de proyectos, agregué un botón para ventas por medio de PayPal, Naturalmente, lo primero fue crear para la asociación una cuenta de PayPal, en la que ingresé los datos necesarios (tipo de empresa, cuenta de banco, un ID y cuenta de correo, etcétera). Hecho esto, con las herramientas de PayPal creé un botón de donación a través de la URL “https://www.paypal.com/mx/cgi-bin/webscr?cmd=_button-designer&factory_type=donate” que genera un código HTML que hay que copiar y pegar en la página web donde se desea que aparezca. En este caso, se trató de la página de proyectos.

En la página de galerías usé la técnica AJAX para la carga de las imágenes sobre el sitio web; así, basta con subir imágenes con extensiones JPG y PNG dentro de un directorio asignado como galerías. Esto permitirá que cualquiera –yo, como desarrollador o el responsable de la asociación–, y a través de la ayuda de algún programa de trasferencia de archivos FTP pueda subir los archivos de imagen, sin tener que agregar las etiquetas en la página web para visualizarlas directamente. Además, con el uso JS generé código para crear un lightbox simple que permita la visualización de estas imágenes de manera ampliada respecto de su tamaño real. Esto se hace a través de dos funciones que se ejecutan bajo diferentes eventos, cambiando propiedades y datos en elementos que ya están preestablecidos en la página para mostrar la información deseada.

Para esta página primero tuve que cerciorarme de que el contenido del archivo HTML estuviera ya listo antes de empezar con la solicitud para consultar los archivos

existentes para desplegar. Así, inserté mi script dentro del elemento BODY, de modo que cuando se ejecute el script tendré por seguro que los elementos que componen la página ya estarán establecidos y cuando haga la solicitud el resultado lo insertará dentro de un elemento DIV que nombré con la ID de “galería” (ver figura 6.20).

```
<!doctype html>
<html>
…
<body>
<div id="envoltura">
  …
  <div id="cuerpo">
   <h1>Galerías</h1>
   <hr>
   <div id="galeria"></div>
  </div>
  …
</div>
…
<script type="text/javascript">activeNav("galeria");</script>
…
</body>
</html>
```

6.20 Código HTML de la página “Galerías”

Lo primero era crear un objeto que permitiera la comunicación con el servidor de manera transparente o en segundo plano, para eso utilicé las instrucciones XMLHttpRequest() y ActiveXObject(“Microsoft.XMLHTTP”). La segunda de ellas es utilizada para navegadores Internet Explorer con las versiones 5 y 6, toda vez que estos navegadores no funcionan con la primera instrucción. Para saber si el navegador es compatible con la instrucción XMLHttpRequest simplemente puse en una sentencia de condición la instrucción “window.XMLHttpRequest” que le hará buscar si existe el objeto para ser usado; si no es compatible, será necesario emplear el ActiveX para crearlo. Ya creado el objeto, hay que prepararlo para la solicitud, a tales fines me serví del método “open”, cuyos parámetros son: el método, a qué archivo del servidor se va a mandar la solicitud y de qué forma (asíncrona o no). Los parámetros empleados fueron los del método POST y de manera síncrona, aunque lo recomendable es que siempre se utilice de manera asíncrona. Sin embargo, en este caso se requería que la función no avanzara hasta no obtener los resultados de la solicitud, por esa razón me decidí por la otra modalidad.

Teniéndolo todo listo y mediante el método “send” se envía la solicitud, el objeto entonces la manda y aunque también es posible enviarla con parámetros, en este caso se optó por vacío. Cuando el servidor genera la respuesta la envía de regreso y el objeto la

captura. A fin de obtener los resultados e insertarlos en mi página web utilicé el método “responseText” en conjunto con la instrucción innerHTML (ver figura 6.21).

```
…
<script>
  var xhttp = new XMLHttpRequest();
  xhttp.open("POST", "buscararchivos.php", false);
  xhttp.setRequestHeader("Content-type", "application/x-www-form-urlencoded");
  xhttp.send();
  document.getElementById("galeria").innerHTML = xhttp.responseText;
</script>
</body>
</html>
```

6.21 Código JS para la carga de imágenes de manera asíncrona mediante el uso de XMLHttpRequest

El archivo que recibe la petición es uno PHP que, a través de las instrucciones “glob”, busca en un directorio determinado aquellos archivos que cumplan con las condiciones. En el presente caso la condición fue cualquier archivo que tuviera extensión “JPG”, “JPGE” y “PNG”. Como se aprecia, la instrucción permitirá obtener los ficheros y directorios en un arreglo y como en este caso nuestra condición era un conjunto de extensiones fue necesario agregar dentro de esta función la bandera “GLOB_BRANCE”, que indicará que cualquiera de esas tres extensiones puede cumplir con la condición. Una vez obtenidos los archivos simplemente generamos una cadena de texto que contiene código HTML con las referencias a las imágenes. Para darle organización al arreglo de las imágenes e irlas agregando de manera ordenada utilicé también la instrucción “natsort” y aunque éste no es indispensable para su funcionamiento genera el código siguiente que será inyectado dentro del elemento “galería” (ver figura 6.22 y 6.23).

```
<?php
$imagenes = array();
foreach(glob('img/galerias/*.{jpg,jpeg,png}',GLOB_BRACE) as $img){
  array_push($imagenes,$img);
}
natsort($imagenes);
foreach ($imagenes as $temp){
  echo '<div><img onclick="showPic(\''.$temp.'\')" src="'.$temp.'"/></div>';
}
?>
```

6.22 Código PHP para el barrido de todos los archivos de tipo imágenes dentro de un directorio

```
<div><img onclick="showPic('[RUTA DE IMAGEN 1]')" src="[RUTA DE IMAGEN 1]"/></div>
<div><img onclick="showPic('[RUTA DE IMAGEN 2]')" src="[RUTA DE IMAGEN 2]"/></div>
<div><img onclick="showPic('[RUTA DE IMAGEN 3]')" src="[RUTA DE IMAGEN 3]"/></div>
<div><img onclick="showPic('[RUTA DE IMAGEN 4]')" src="[RUTA DE IMAGEN 4]"/></div>
<div><img onclick="showPic('[RUTA DE IMAGEN 5]')" src="[RUTA DE IMAGEN 5]"/></div>
…
```

6.23 Código devuelto por el servidor tras procesar la solicitud del archivo PHP

Finalmente, ya obtenidos todos los elementos de imagen en la página, lo restante fue crear mi propio lightbox. Para tal propósito hice uso de código JS y CSS, además de dejar establecidos unos elementos sobre la página web. La operación inicial es crear los elementos HTML que se mostrarán para que se visualice el lightbox; dicho objeto está compuesto por un elemento DIV que nombré con un ID como “capa”, éste contiene a su vez otros dos elementos DIV que tienen como ID “texto” y “pic” respectivamente. El elemento padre funciona como un fondo negro con transparencia que ocupa toda la ventana del navegador y a tales efectos eso asigné a este elemento las propiedades fondo (background) con la función “rgba()” y los valores 0, 0, 0, 0.5 respectivamente, en el que el primer valor es la cantidad de rojo, el segundo la cantidad de verde, el tercero la cantidad de azul y finalmente el último, el valor alfa, que da la trasparencia. Después agregué las propiedades alto y ancho (height y width) al 100%, con posición arreglada (fixed) para que el objeto se mantenga tal cual, a pesar del desplazamiento de la ventana; añadí el valor de profundidad (z-index) en 2 para asegurarme de que estará por encima de todo y -algo muy importante— incluí el tipo de caja (display) en nada (none). Esto último es fundamental, porque se busca que solo aparezca cuando el visitante haga clic en alguna de las imágenes. El elemento con el ID “texto” solamente servirá para mostrar un mensaje en la parte inferior de la pantalla que indicará la manera de salir del lightbox. A éste le agregué las propiedades alineación de texto (text-align) centrado, fondo blanco, posición absoluta, la propiedad abajo (bottom) con 0px y el ancho del 100%. Tales propiedades nos asegurarán que este mensaje siempre aparecerá como una barra blanca con el mensaje al borde de la ventana de abajo. Por último, el elemento con el ID “pic” contendrá la imagen que se mostrará al visitante cuando haga clic sobre alguna de las imágenes previamente desplegadas. Aunque la imagen se insertará por medio de las funciones que más adelante explicaré, hay que dejar también asignadas las propiedades necesarias para que el cuadro siempre esté en medio de la ventana. Tal cosa la conseguí asignando al elemento las propiedades: posición relativa, un ancho del 100% y la propiedad de arriba (Top) al 50%. Además, agregué

también propiedades al elemento IMG con el ID “pic” para darle un poco de diseño y asegurar que se mantenga centrado. Las propiedades de este último son: tipo de caja (display) en bloque, márgenes automáticos relleno (padding) de 0px, borde (border) de 15px sólido y color blanco y esquinas redondeadas (border-radius) de 15px. Esta última propiedad hará que en el borde sobre la imagen en las esquinas sea redondeado, esta propiedad pertenece al CSS 3 por tanto, si se usa en navegadores más antiguos no será interpretada, pero no afectará la funcionalidad, pues sólo se verá como bordes con puntas (ver figura 6.24).

```
…
#capa{
  background:rgba(0,0,0,0.5);
  width:100%;
  height:100%;
  position:fixed;
  z-index:2;
 display:none;
}

#pic{
  width:100%;
  position:relative;
  top:50%;
}

#pic img{
  display:block;
  margin:auto;
  padding:0px;
  border:15px solid #FFF;
  border-radius:15px;
}
#texto{
  text-align:center;
  background:#FFF;
  font-size:30px;
  position:absolute;
  bottom:0px;
  width:100%;
}
…
<body>
<div id="capa" onClick="hidePic()">
<div id="texto">Haga clic para regresar</div>
<div id="pic"></div>
</div>
…
</body>
</html>
```

6.24 Código CSS y HTML para la creación de un “Light Box”

Descritos y preparados los elementos para su función, pasaré a explicar las funciones que cambiarán algunas propiedades y cargarán las imágenes. La primera que

desarrollé, fue una denominada showPic, que recibe como parámetro una cadena de texto al que nombraré “src”. Su objetivo primordial es cambiar la visualización del elemento “capa” para que sea visible, aunque previamente es necesario localizar el elemento. Para este propósito recurrí a la instrucción getElementById() con el parámetro “capa”, una vez localizado y guardado en una variable llamada capa, cambié la propiedad de tipo de caja, que ya tenía previamente en nada (none), a bloque (block) por medio de la instrucción: “capa.style.display = ‘block’ ”.

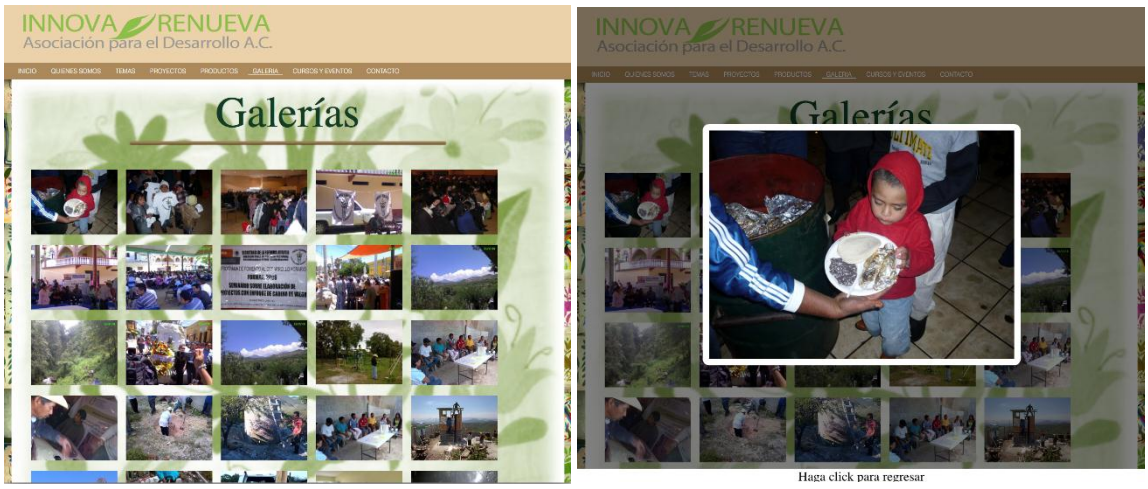
Una vez que la visualización cambió a ser visible, el siguiente paso es inyectar la imagen en la que el visitante da clic. En principio, hay que localizar el elemento que tiene por ID “pic”, lo que hice a través de la instrucción getElementById, pero ahora como parámetro con la cadena “pic”, y agregado a una variable, simplemente con el uso de la instrucción innerHTML, inyecté un texto en conjunto con nuestra variable “scr”, que es un valor que recibió para la función, a fin de que tengamos como resultado algo parecido a “<img src=” [RUTA] ”>. El último detalle para centrar permanentemente la imagen en nuestra pantalla fue obtener la altura de imagen mediante la instrucción “pic.clientHeight”; ese valor lo dividí a la mitad y lo asigné como valor inverso al elemento “pic” como margen superior (top-margin) con el uso de la instrucción “pic.style.marginTop = (heightPic/2)” donde heightPic es el valor que recibió de la instrucción “pic.clientHeight”.

Si se recuerda, en párrafos anteriores habíamos asignado a la propiedad para el elemento “pic” una posición absoluta con la posición arriba (top) en 50% y eso, con el margen superior a la mitad de la imagen, asegurará que el elemento estará centrado sobre la pantalla. Esta última propiedad la agregué sobre la función, ya que las imágenes pueden variar de forma es necesario que cada margen superior se generé individualmente para garantizar que siempre estará centrada. Una nota importante sobre esta función es que el parámetro que recibe de la función ya está previamente establecido por el archivo PHP que antes mencioné; si se revisa se podrá encontrar que el resultado de la petición será un elemento DIV que tiene dentro un elemento IMG, cuyo atributo “onclick” funciona como detonador cuando un el cursor hace clic sobre el elemento, y ejecuta la función showPic con su parámetro particular. De esta manera nos cercioramos de que siempre cargará la imagen sobre la que se haya dado el clic. Ahora, la segunda función que creé revierte los cambios que la función anterior realiza, es decir, vuelve a ocultar el elemento con ID “capa”, vacía, el elemento con ID ”pic” y restablece el margen superior a 0px. Para ello, lo primero es, nuevamente, localizar nuestros dos elementos. Una vez hecho esto y guardados en una

variable volvemos a evocar las mismas instrucciones, pero ahora con valores diferentes, para el margen superior le asigne un valor de 0px; el contenido del elemento con ID “pic” lo asigne como vacío, simplemente colocando doble comilla, y el tipo de caja del mismo elemento como nada (none) de nuevo (ver figura 6.25 y 6.26).

```
function showPic(src){
        capa = document.getElementById("capa");
        pic = document.getElementById("pic");
        capa.style.display = "block";
        pic.innerHTML = '<img src="'+src+'"/>';
        heightPic = pic.clientHeight;
        pic.style.marginTop = -(heightPic/2)+"px";
}

function hidePic(){
        capa = document.getElementById("capa ");
        pic = document.getElementById("pic");
        pic.style.marginTop = "0px";
        pic.innerHTML = "";
        capa.style.display = "none";
}
```

6.25 Código JS para el control del “Light Box”




6.26 Captura de pantalla de la página galerías y del “Light Box”

La página de contacto la elaboré como un archivo PHP que incluye igualmente los elementos HTML y CSS, además del código para el servidor. La página es sencilla, en relación con los elementos HTML, y está formada por un elemento FORM que, a su vez, tiene en su interior elementos INPUT, LABEL y TEXTAREA para la captura de información. Inmediatamente después de generar el elemento FORM agregué una estructura de tabla para disponer de mayor control de los elementos internos del formulario; a cada entrada (INPUT) le asocié un elemento de etiqueta (LABEL) para que al visitante le resultara más

cómoda y fácil de entender. Así, si éste hace clic en el texto, dependiendo de la etiqueta, ésta activaría el cuadro de texto determinado y de igual manera el cuadro de texto (TEXTAREA). El formulario está compuesto por un campo de nombre, apellido, teléfono, correo y comentarios, más los botones de enviar o limpiar. Las primeras cuatro entradas constan de una etiqueta y un elemento de entrada; cada elemento está dentro de una celda de la tabla antes mencionada. Creé los elementos para cada etiqueta le asigné un valor y un atributo FOR con un nombre a fin de poder ligar la etiqueta con la entrada y de que se activara el elemento de entrada al hacer clic; por otro lado, a cada celda en la que está contenida la etiqueta agregué un elemento SPAN que sirve para escribir mensajes de error si no se cumple algún requisito al enviar el formulario; en otra parte explicaré como trabaja. Luego, a cada elemento de entrada le agregué atributos de ID, con el valor que le corresponde de cada etiqueta; puse NAME para que el servidor recuperara los datos ingresados por el visitante y VALUE, que en realidad tiene una variable de PHP que regresará el dato que el visitante haya ingresado al intentar enviar y que, por alguna razón, se encontrase con incumplimiento de la petición, esto a fin de que el usuario no tenga que volver a escribir todo de nuevo. Para la sección de comentarios, fue necesario fusionar las celdas de la tabla vertical mediante con el atributo COLSPAN, para que éste (que era el elemento para los comentarios), pudiera ocupar todo el espacio disponible y no se viera cortado o a la mitad. Asimismo, utilicé un elemento de cuadro de texto (TEXTAREA) a fin de facilitar la escritura de un texto más largo, en un espacio mayor que el de los campos anteriores. De igual manera, agregué los atributos ID y NAME pero aquí, en vez de usar el atributo VALUE para rescatar el dato del visitante, lo inserté entre las etiquetas. Finalmente, los últimos dos elementos son botones que ejecutan la tarea de enviar el formulario o de limpiar el contenido del mismo. Esta última sección la diseñe para que fuera creada de manera dinámica por parte del servidor, toda vez que el proceso de enviar el formulario lo realiza sobre el mismo archivo; en consecuencia es indispensable controlar si va a generar los botones o bien generar algún mensaje de estado una vez que se haya enviado el formulario. Esto evita al visitante la duda de si su formulario fue enviado o no.

Con la instrucción, htmlspecialchars($_SERVER[”PHP_SELF”]) hago que el envío de los datos se haga al mismo archivo en el que está localizado, y para añadir un poco de seguridad está la función “htmlspecialchars()” con la que protejo a la página para evitar que se pueda inyectar sobre el contenido código dañino o elementos adicionales. Dicha función cambia los caracteres especiales para HTML como “<”, “>”, etcétera a conjunto de

caracteres para que al ser enviados no pasen como código ejecutable dentro del proceso; la instrucción $_SERVER[”PHP_SELF”] es una variable global de PHP que regresa el nombre del archivo en el que se está ejecutando el script, a fin de que cuando el visitante envíe el formulario, lo haga a la a página. Aparte de los fragmentos de código dentro del formulario, la página posee una sección de código PHP donde realiza la verificación, manejo, proceso y respuesta del formulario. Dentro de esa sección –temática en la que no abundaré por cuestiones de seguridad— el formulario es recibido y se extraen los datos para generar una respuesta en la página, como los mensajes de error y el estado, es decir si el envío fue o no exitoso, en cuyo caso también se abre la posibilidad de enviarlo por correo electrónico a las cuentas asignadas (ver figura 6.27).

```
…
<form action="<?php echo htmlspecialchars($_SERVER["PHP_SELF"]);?>" method="post" name="contacto"
id="contacto">
<table>
  <tr><td><p style="font-size:12px; color:#AA0000; margin-bottom:5px;">* Requerido</p></td></tr>
  <tr>
   <td><label id="text" for="nombre">Nombre: *</label><span id="error"> <?php echo [VAR ERROR] ;?></span></td>
   <td><label id="text" for="apellido">Apellido: *</label><span id="error"> <?php echo [VAR ERROR] ;?></span></td>
  </tr>
  <tr>
   <td><input name="fname" type="text" id="nombre" value="<?php echo [VAR] ;?>" /></td>
   <td><input name="lname" type="text" id="apellido" value="<?php echo [VAR] ;?>" /></td>
  </tr>
  <tr>
   <td><label id="text" for="telefono">Telefono: *</label><span id="error"> <?php echo [VAR ERROR] ;?></span></td>
   <td><label id="text" for="correo">Correo: *</label><span id="error"> <?php echo [VAR ERROR] ;?></span></td>
  </tr>
  <tr>
   <td><input name="phone" type="text" id="telefono" value="<?php echo [VAR] ;?>" /></td>
   <td><input name="email" type="text" id="correo" value="<?php echo [VAR] ;?>" /></td>
  </tr>
  <tr>
   <td colspan="2"><label id="text" for="comentario">Comentario: *</label><span id="error"> <?php echo [VAR
ERROR] ;?></span></td>
  </tr>
  <tr>
   <td colspan="2"><textarea name="comentario" rows="20" wrap="hard" id="comentario"><?php echo [VAR]
;?></textarea></td>
  </tr>
<?php
if([CONDICION])
  echo
   '<tr>
     <td colspan="2">
      <input type="reset" value="Borrar Todo"/>
      <input type="submit" value="Enviar"/>
     </td>
   </tr>';
?>
</table>
<p align="center">Lo invitamos a revisar nuestro aviso de <a href="privacidad.html" target="new">Privacidad</a></p>
</form>
<?php echo "<h1>".[CONDICION]."</h1>"; ?>
```

6.27 Código HTML y PHP para el control, generación de respuestas y envío del formulario

Una vez culminada toda esta etapa, en la siguiente fase –la de pruebas— sometí el sitio web a pruebas variadas para comprobar su funcionamiento y comportamiento bajo diferentes escenarios. Recurrí a una plataforma de servidor para simular el comportamiento a la hora de interpretar el lenguaje de programación del lado del servidor y para determinar los resultados que daría al cliente. La plataforma que utilicé para tal propósito fue XAMPP instalada en mi equipo de trabajo, una vez ejecutado y funcionando el sitio, pasé todo el directorio dentro del directorio que XAMPP utiliza para dar el servicio.

El primer paso fue validar íntegramente el sitio web ante la W3C.org por medio de su herramienta para verificar los archivos HTML y CSS. Esto puede hacerse subiendo el archivo, o bien, copiando y pegando todo el contendido como fue mi caso; procedí así con la intención de buscar en cada página web que el archivo estuviera bien formado y validado utilizando el esquema DTD (Document Type Definition). Una vez verificada su validación, lo siguiente fue probar la compatibilidad con los navegadores más usados, que según con los datos estadísticos del sitio web W3schools, durante el mes de agosto de 2016 mostraron los siguientes los porcentajes:

- Google Chrome 72.4 %
- Firefox 16.8 %
- Internet Explorer 5.2 %
- Safari 3.2 %
- Otros 1.1%

Verificar que no haya comportamientos o interpretaciones inesperadas por parte de los navegadores se consigue a través de varios sitios web que ofrecen estos servicios de carga de páginas web en navegadores variados. En este caso y en vista de que aún no se había contratado el servicio de hosting ni el dominio, hice la comprobación de manera manual con los navegadores instalados en mi equipo. Un ejemplo de problema de compatibilidad se da con el navegador Firefox (48.0.2) a la hora de usar la propiedad posición como relativa, lo que deja el espacio del elemento en donde fue insertado a pesar de ser desplazado éste. Tal es un problema común cuando se trata de visualizar los elementos, así ocurre con los Stickyfoots cuando se usa esta técnica con posicionamientos que deja un espacio innecesario debajo del pie de página. El siguiente paso fue revisar que

los procesos ejecutados por el servidor respondieran de manera adecuada y no tuvieran errores de sintaxis o de lógica, además de revisar que la versión de lenguaje de programación (PHP) fuera compatible con las instrucciones utilizadas para la realización de las tareas. Aquí el punto era buscar que no estuviera en desuso o bien, que no fuera una instrucción únicamente aplicable para las últimas versiones de PHP (ver. 7), toda vez que los servidores contratados por lo general siguen trabajando con la versión 5. Lo único que no podía probar a estas alturas era el proceso de enviar el correo por medio de PHP, puesto que necesitaba instalar un servidor de correos para tal propósito; como sea, esta parte es más fácil revisarla ya sobre el servidor en que va a ser hospedado.

Ejecutadas esas tareas, lo siguiente fue recopilar todos los resultados de cada una y realizar el proceso de depurado, para corregir aquellas imperfecciones que pudieran haber pasado por alto en la etapa de desarrollo. Tal cosa se logra con ayuda de herramientas que tienen los navegadores, como los inspectores o depuradores, y recurrí a ellos para llevar un mejor control y verificar el comportamiento al momento del despliegue y proceso de cada página. Además, es un buen recurso para analizar y localizar fallas en las páginas, causadas por errores de lógica o interpretaciones a cargo de los navegadores.

En la última fase (de cierre), contraté y registré el servicio de hosting y el dominio del sitio web. Para el registro del dominio en el sitio web del proveedor del servicio se cuenta con una herramienta que revisa la disponibilidad; en este caso el dominio deseado estaba libre, por lo que quedó como “innovayrenueva.com.mx”. En vista de que el objetivo del sitio era simplemente informar y convertirse en un vehículo publicitario, el plan contratado fue el más simple, esto es aquel que cubría los requisitos mínimos como disponibilidad del 99.9%, inclusión de hasta 300 cuentas de correo con el dominio registrado, almacenamiento mínimo compartido con el servidor de correo, PHP 5, Acceso FTP, respaldos, trasferencias ilimitadas y más.

Una vez contratado y pagado el servicio, con el uso de una herramienta para transferencia FTP como Filezilla, me enlacé con el servidor usando la cuenta y contraseña asignada por el proveedor de servicio de hosting para empezar con el proceso de trasferencia de todo el contenido desarrollado para este sitio web. Ya que la transferencia se realizó correctamente, lo siguiente fue entrar a través del navegador con las mismas cuentas al panel de configuración del hosting (en este servidor CPanel), a fin de crear las cuentas de correo necesarias. La cuenta de correo principal la configuré con capacidad ilimitada, puesto que se función es la captura de todo correo que se envíe. Todo nombre de

nuevo registro de dominio demora aproximadamente unas 24 horas, que es el plazo en el que se propaga el dominio por todo el mundo. Así que, a través de la IP fija del servidor donde está hospedado el sitio web accesé al sito para corroborar que todo funcionara de manera deseada, sometiéndolo a una rápida revisión de pruebas de compatibilidad, comportamiento, etcétera Finalmente, el último paso fue revisar que la página de contacto, después de todo el proceso del lado del servidor, confluyera con el envío de correos a las cuentas asignadas de manera exitosa.

## Resultados

Elaboré, planeé y desarrollé un sitio web accesible a través de internet, para la publicidad, información, venta de los productos y contacto con la asociación Innova y Renueva. El proyecto fue integral, pues abarcó desde su diseño y planeación hasta su ejecución y publicación. Diseñé este sitio web con una estructura organizada de forma jerárquica para la distribución de información con bases y teorías fundamentadas acerca temas de diseño web, cerciorándome de que fuera intuitivo y sencillo de navegar para cualquier visitante y de qué este tuviera un acceso sencillo a la información de su interés. El sitio tiene una amplia compatibilidad con diferentes navegadores y versiones para desplegar la información de manera correcta. Diseñé el código PHP a prueba de inyecciones para una mayor seguridad y para evitar ataques o spams.

Con todo, el avance del proyecto fue lento porque los tiempos y la agenda del representante de la asociación no le dejaban mucho espacio para este trabajo. Así que hubo demoras en la entrega de textos y materiales, el proyecto inicio en julio del 2015 y duro 7 meses causadas mayormente de estas demoras. Igualmente, hubo que introducir cambios al proyecto original sobre la marcha, y estos se dieron sobre todo en el aspecto del diseño visual. A pesar de estas situaciones, el cliente quedó satisfecho con el trabajo que realicé.

## Conclusiones

Una asociación civil no es lo mismo que un negocio. Y aunque en el caso de la asociación Innova y Renueva sí hay venta y promoción de productos elaborados por las comunidades indígenas que apoya, su objetivo final no es económico ni mucho menos de lucro, puesto que no busca prioritariamente la comercialización de bienes.

Por ello, la cuestión central no era publicitar artículos para el mercado, sino dar la mayor difusión posible a las actividades y los proyectos de la organización y alcanzar a un público interesado cada vez más amplio.

Así pues, aunque la recomendación que les hice para cubrir sus necesidades publicitarias fue también la de servirse de un sitio web –como lo hacen las empresas–, les insistí particularmente en el recurso a las redes sociales de Facebook y Twitter. Tener presencia en ellas garantiza a la asociación un enorme universo de personas potencialmente interesadas en sus actividades. Por otro lado, la implementación de herramientas como PayPal en su sitio web permite a la organización disponer de un medio simple y seguro para la recaudación de fondos, a través de la venta de los productos de las comunidades.

Así pues, por estas particulares condiciones y características desarrollé el proyecto de manera integral (página web, Facebook y Twitter) y a la medida. La experiencia con Innova y Renueva me permitió no sólo aplicar los conocimientos técnicos adquiridos en la carrera, sino proyectar un plan global de recursos informáticos conjuntos y ponerlos al servicio de una asociación civil, lo que fue muy ilustrativo.

# VII. Desarrollo de sitio web “Restatools”

## Objetivo

Planear, diseñar, desarrollar y depurar un sitio web para publicitar los servicios de una empresa de mantenimiento integral para restaurantes. Generar un espacio atractivo, intuitivo de difusión, un medio de contratación de nuevos empleados y una vía para generar contactos de manera fácil y automática.

## Antecedentes

Restatools es un negocio que brinda servicio de mantenimiento general a los equipos de cocina de restaurantes, aunque también los ofrece para cocinas domésticas. Cuando me contactó, el dueño apenas iniciaba esta nueva empresa y le urgía darse a conocer por medio de un sitio web.

## Descripción del problema o contexto de la participación profesional

El propietario no tenía ideas claras sobre lo que quería del sitio web, sino apenas algunas nociones muy generales de cómo le gustaría que luciera. Quería, eso sí, un diseño agradable y sencillo, pero de aspecto muy profesional; estaba muy entusiasmado con aportar ideas e iniciativas y acabó por definir que lo que debía figurar como mínimo en su página eran tres elementos: un espacio para describir los servicios que prestaba; un medio de comunicación y una cuenta de correo personalizada, además un apartado donde pudiera recibir currículos de personas interesadas en trabajar en su negocio. Amén de ello, deseaba que el sitio web sólo dispusiera de una página única en la que los visitantes pudieran navegar.

Una vez definido el plan general, se me encomendó la investigación, el análisis y el desarrollo del proyecto, así como la dirección, administración de tiempo de vida y cálculo financiero de él.

## Metodología utilizada y participación profesional

Como Restatools no tenía sitio web hasta el momento, inicié el proyecto desde cero. Generé en principio un plan de seguimiento que constó de 4 fases:

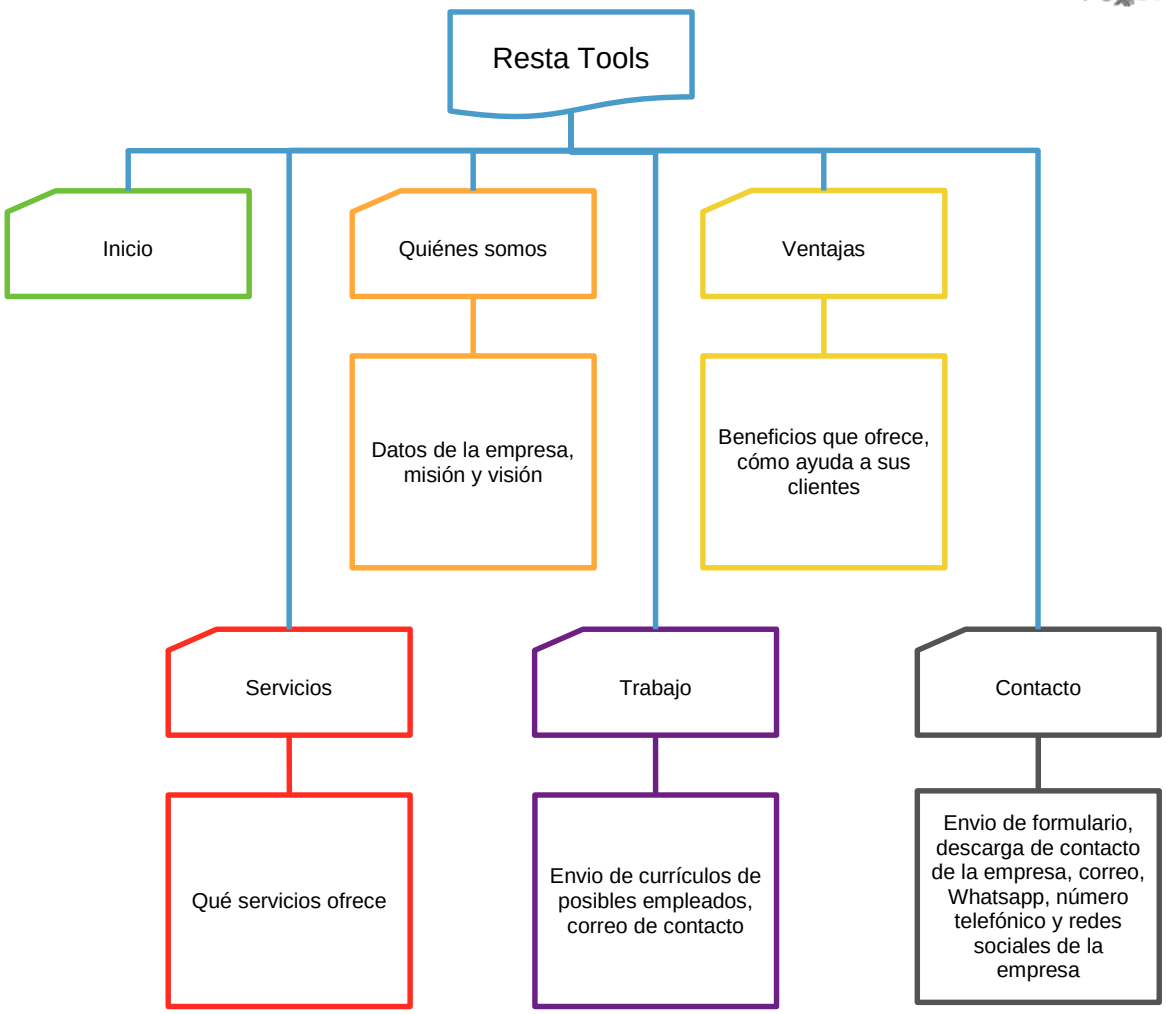
- Fase I o de preparación

- Fase II o de desarrollo
- Fase III o de laboratorio y pruebas
- Fase IV o de cierre

La fase de preparación consistió en el preanálisis, análisis, planeación y diseño del sitio web. Ponderé los requisitos, las oportunidades que podría explotar, la manera óptima de conseguirlo y todo ello canalizado al objetivo central.

Durante el preanálisis entrevisté nuevamente al empresario para reafirmar y definir sus necesidades precisas, es decir, aquellas debían quedar cabalmente satisfechas a través de la construcción de un sitio web. Además de los detalles antes señalados –como que constara de una sola página y que luciera atractivo y muy profesional, que le sirviera de medio de publicidad y de contactos, etcétera— recabé otros datos nuevos, por ejemplo, su interés en el empleo de una fuente de letra en particular y que se utilizaran el amarillo, el negro y el blanco como colores predominantes. En esta parte fui especialmente cuidadoso, pues no sólo se trata de dar al cliente lo que pide, sino, de ser posible, incluso tratar de superar sus expectativas.

Ya teniendo a la mano la información precisa del cliente, empecé el análisis y la planeación para cubrir sus requerimientos. Exploré las opciones que podrían mejorar o facilitar el cumplimiento de metas respecto de las ideas originales, evalué si era viable llevarlas a efecto de una forma determinada o si había alternativa para ensayar otras vertientes. Así, de entrada planeé dividir todo el futuro contenido del sitio web en pequeñas partes, esto con el propósito de ordenar la información que el empresario quería poner a disposición de sus visitantes. Al final, la jerarquización quedó como sigue en la figura 7.1:




7.1 Mapa jerárquico del contenido de Resta Tools

El objetivo de las páginas de un sitio web es tener la capacidad de fragmentar la información en bloques ordenados, ofrecer comodidades a la vista al impedir la carga excesiva o la saturación de elementos por página y facilitar la búsqueda de información a sus visitantes. Y uno de los requisitos que me había impuesto mi cliente era evitar la generación o multiplicación de páginas, así que la opción que propuse fue la actualización del contenido por medio de JS, así se evitaba la multiplicidad, pero se introducía la característica de seccionar la información en partes y de cumplir con conceptos positivos en cuestión del diseño visual.

El diseño lo concebí en un espacio serio, muy ligero, buscando que siempre que fuese intuitivo y fácil de navegar en cada sección (le daré este nombre puesto que no tiene páginas) para le expedita localización de la información. Lo elaboré con un grupo de colores

conformados por amarillo y negro; además, utilicé transparencias en conjunto con las imágenes que me fue proporcionando el empresario.

En cuanto a la estructura y diseño visual del sitio web generé un fondo para la ventana que consistiera en un carrusel de imágenes automatizado y redimensionable, del mismo tamaño de la ventana, sin experimentar deformaciones por la relación entre ancho y alto; esto apoyaría visualmente en general a la página. Los elementos principales del sitio web fueron: un encabezado que abarca todo el ancho de la ventana y que pudiera ser redimensionado, manteniendo siempre los elementos en su lugar. Dentro de él dejé un espacio para el logotipo de la empresa, amén de algunas ligas con otras páginas (como redes sociales y empresariales); incluí un menú de navegación que, al igual que el encabezado, toma todo el ancho de la ventana, aunque sus ligas están centradas.

Para el cuerpo del sitio dispuse un tamaño igual al ancho de la ventana, que alcanzaba la altura necesaria para que todos los elementos de la página ocupasen la totalidad de la pantalla, sin importar las dimensiones de la ventana. Este elemento es el contenedor de las diferentes secciones del sitio web y reemplaza cada texto con el nuevo al que se quiera pasar, todo a fin de evitar el salto de páginas web.

Hay además un elemento que figura un poco arriba del pie de página, tiene determinada anchura y altura y se encarga de desplegar un texto que, dependiendo de la sección, indica al visitante en qué punto preciso del sitio web se encuentran. Por último, hay un pie de página que, al igual que la cabecera, abarca todo el ancho de la ventana. Éste consta de dos partes: la primera hace pasar pequeños textos animados de noticias, moviéndose de izquierda a derecha y la segunda incluye los datos de la empresa. Además, se dispuso una capa que servirá para mostrar una animación introductoria antes de entrar al sitio web.

Utilicé lenguaje de estilo para dar calidad visual al sitio web; usé trasparencias y animaciones a través de keyframes y de simplificar un poco el código generado que desplegará para el cliente y de programación de lado del cliente para el funcionamiento de todo el sitio web. Creé un único archivo CSS para separar entre las propiedades de estilo y el código HTML. Con JS es posible aplicarlas de manera eficiente a los elementos que se sobrescriban o remplacen a la hora de ejecutar instrucciones para el funcionamiento del sitio.

Ya durante la segunda fase, la de desarrollo, generé el código y programé con el uso de lenguaje de etiquetas HTML, hoja de estilo CSS, lenguaje de programación lado de

servidor PHP, lenguaje de programación lado del cliente JS y librerías para la construcción, la carga de información, envío de archivos adjuntos y correos HTML. Creé el directorio con el nombre de “restatools” y todos los archivos que fui generando los guardé en él.

Generé el archivo HTML de inicio con el nombre de “índex” cuyo contenido estará en blanco, aunque tendrá una etiqueta META con los atributos “http-equiv” con el valor de refrescar (refresh) y “content”, con el tiempo y la url separadas por un punto y coma. Lo que hará será redireccionar al visitante a la página “principal” que es la única y donde él puede verse todo su contenido. Las ventajas de esta etiqueta para redireccionar son: que al ser una etiqueta HTML la pueden interpretar todos los navegadores conocidos popularmente y en todas las versiones, que la acción no puede ser bloqueada o suspendida como sí lo sería en el caso de haber usado JS en la instrucción “window.location”; además de que es posible determinar el tiempo en que se ejecuta la acción (ver figura 7.2).

```
<!doctype html>
<html>
<head>
<meta charset="UTF-8">
<meta http-equiv="refresh" content="0; url=principal.html" />
<title>Redireccionando</title>
</head>

<body>
</body>
</html>
```

7.2 Código HTML del archivo Index

Para este sitio web generé dos archivos scripts; el primero de ellos funciona como motor del sitio web para la navegación, el despliegue de información, las animaciones de algunos elementos, los cálculos y procesos internos para el funcionamiento y visualización correcta del sitio, las generaciones de cookies y alteraciones de elemento HTML con la intención de reducir los errores u mantener un orden; con él resulta sencillo agregar o modificar información. Al funcionar como motor o núcleo del sitio web, fue necesario que este script se ejecutara después de que la única página se haya cargado completamente, esto para evitar fallas cuando busque algún elemento que quiera modificar. Por tanto y para asegurar que la condición se cumpla opté por llamar el archivo al final del contenido dentro de la etiqueta “BODY”. El segundo script es un archivo que utilicé solamente para la comprobación y validación en el formulario de empleados potenciales, tema en el que más adelante profundizaré. En vista de que el sitio web todo lo genera en la misma página y por orden, decidí ejecutar este script –al igual que el anterior– al final de la etiqueta “BODY”,

aunque en este último caso no es estrictamente necesario ya que se trata de funciones ejecutadas por elementos HMTL.

Para la página principal, utilicé un ancho de mínimo de 980px empleando la propiedad de ancho mínima (“min-width”) para el fondo, la cabecera, el cuerpo y el pie de página, según los datos estadísticos del sitio web “w3schools”. Usé las fuentes “bitsubishi” y “futura” como fuentes principales para la página.

La estructura está conformada primero por un elemento DIV con el ID “intro” que funciona como una capa del tamaño de la ventana que sirve de carta introductoria o presentación de la empresa; incluye animaciones en CSS3 formadas con imágenes y keyframes para que, una vez que terminada la animación, ésta desaparezca. Le sigue un elemento que funciona como cabecera y que tiene otros dos elementos hijos; uno de ellos es un elemento “DIV” con ID de “logo” que contiene en su interior un elemento de imagen PNG con transparencia del logotipo de la organización y dos elementos de ancla en cuyo interior hay un elemento de imagen. El otro es otro elemento de lista (UL) que tiene un ID de “nav” vacía hasta que en él se ejecute el script que funciona como motor; este elemento forma el menú de navegación, la carga y sobrescribe los contenidos sobre la página. Después sigue un elemento DIV con ID llamado “cuerpo”; en éste se despliega toda la información que el visitante solicite por medio del menú de navegación al generar nuevos nodos y asociarlos como elementos hijos de éste. Sigue otro elemento DIV con un ID llamado “seccion” que solamente indica al visitante en que sección se encuentra en el momento y –en algunas partes– describe la simbología del elemento sobre el que se posiciones el ratón. Queda un elemento DIV con un ID llamado “pie” con una serie de elementos que funcionan como mensajes transitorios y muestran información acerca de la empresa. Finalmente, los últimos dos son elementos script; el primero, que es el que nos interesa por ahora, cargará el archivo y sobrescribirá en los elementos antes mencionados para su correcto funcionamiento (ver figura 7.3).

```
<body onResize="redim()">
 <div id=“intro">
   <img id="anim1" src="img/anim1-250.png" alt=“anim”>
   <img id="anim2" src="img/anim2-250.png" alt=“anim">
 </div>
 <div id="bkgd"></div>
 <div id="cabezera">
  <div id="logo">
   <img src="img/logo3.png" alt="logo" onClick="limpiar()">
   <a href="https://www.facebook.com/restatools" target="_blank">
    <img src="img/fb.png" alt="facebook">
   </a>
   <a href=“https://www.linkedin.com/…” target="_blank">
    <img src="img/ln.png" alt="twitter">
   </a>
  </div>
  <ul id="nav"></ul>
 </div>

 <div id="cuerpo"> </div>
 <div id="seccion">Mantenimiento</div>

 <div id="pie">
  <div id="contactos" style=“display:none;”>
   <img src="img/phone.png" alt="phone" … />
   <img src="img/whatsapp.png" alt="whatsapp" … />
   <img src="img/email.png" alt="email" … />
   <a href="RESTA TOOLS.vcf" … >
    <img src="img/contacto.png" alt=“contacto”/>
   </a>
   <a href="https://www.facebook.com/restatools" target="_blank" … >
    <img src="img/fb-o.png" alt=“facebookdet"/>
   </a>
   <a href=“https://www.linkedin.com/…” …>
    <img src="img/ln-o.png" alt=“linkedindet"/>
   </a>
  </div>

  <div id=“datos">
   Tel. <small>5424 3067</small><br>
   <small>Cuitlahuac 166, Toriello Guerra, Tlalpan, 14050, Ciudad de México</small>
  </div>
  <div class="detalles">Servicios personalizados de mantenimiento a restaurantes</div>
  <div class="detalles">Calidad, rapidez y eficiencia en tus servicios de mantenimiento</div>
  <div class="detalles">Todos tus servicios de mantenimiento en un solo lugar</div>
 </div>

 <script src="script.js" type="text/javascript"></script>
 <script src="model/fileupload.js" type="text/javascript"></script>

</body>
```

7.3 Código HTML del archivo Principal

Descrita la estructura general, paso a detallar el funcionamiento de cada componente. El elemento con el ID “intro” funciona como una capa que cubre toda la ventana para impedir la visualización de la página hasta que se haya terminado de cargar. La capa contiene imágenes que, en conjunto, forman el logotipo de la empresa y que brevemente mostrarán animación, antes de desvanecerse. El logo de la empresa lo integra

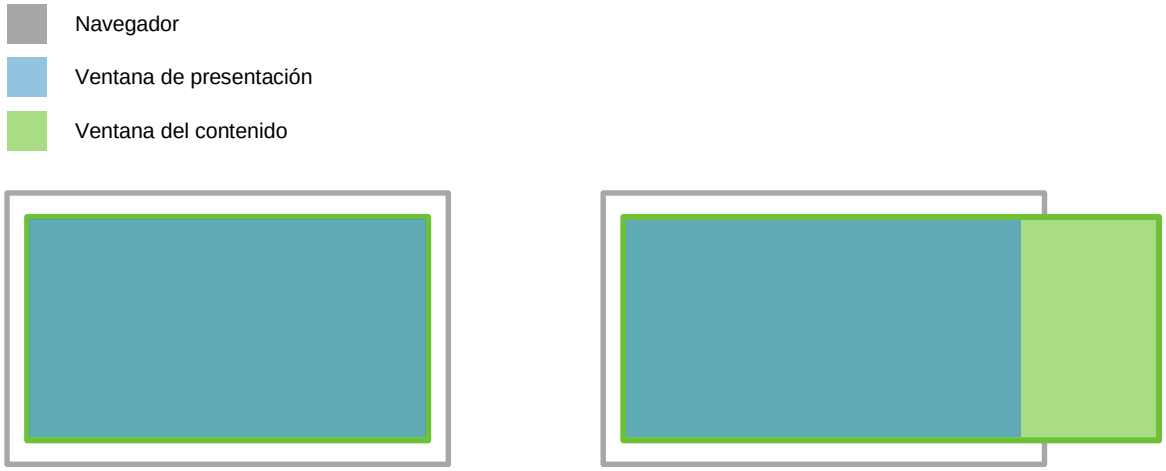
un conjunto de implementos de cocina y mecánicos, rodeados por un engrane; la animación es el giro del engranaje que persistirá lo suficiente para que la página se cargue, lo que da la impresión al visitante de que el sitio está descargando todo su contenido antes de que pueda visualizarlo (ver figura 7.4).

7.4 Página de presentación de Restatools

Incluí esta parte para dar un toque de originalidad e inventiva al sitio y para hacerlo distinto de otros que, simplemente, dan una entrada súbita o directa a las páginas. Adicionalmente, y considerando que el sitio web incluye mucho código de SJ en su funcionamiento, la animación también da tiempo para que, en segundo plano, la página pueda procesar todas las instrucciones, generar los cambios requeridos y reflejarlos, antes de que el visitante entre en el contenido. No dar este intervalo haría que el visitante viera los cambios de propiedades en los elementos, lo que restaría calidad de presentación al sito.

Para esta animación introductoria creé el elemento DIV con el id “intro”, éste fue necesario asegurar que quedara por encima y que cubriera cualquier otro elemento de la página. A tales fines utilicé las propiedades un fondo de color negro, posición absoluta y profundidad (z-index) en 2, después, para que cubriera toda la ventana utilicé las propiedades arriba (top) e izquierda (left), a fin de obligarlo a empezar desde el inicio de

nuestra pantalla en conjunto con la propiedad de posición. Luego, asigné el ancho y alto en 100%; sin embargo, como dije en otra parte, el ancho mínimo del sitio web es de 980px, por tanto, también asigné a esta capa el mismo ancho mínimo. De no tomar esta precaución, el elemento adoptaría la anchura que detecte en la ventana y si ésta es inferior a los 980px quedará al descubierto parte del contenido detrás del sitio web. Dado que el contenido del posterior es más grande que la ventana activaría la barra de desplazamiento horizontal y el espacio que sobresale de la ventana no sería cubierto por la capa con ID “intro”; al desplazarse horizontalmente aparecería la parte posterior antes de acabar la animación (ver figura 7.5).




7.5 Ilustración de ajuste horizontal con la propiedad ***min-width***

Asignando este mínimo de ancho obligamos al elemento a ser siempre del mismo tamaño que el del contenido posterior, sin importar si la ventana es más pequeña. Una vez colocada la capa, lo siguiente fue asignarle propiedades para la finalización de las animaciones. Con este fin primero generé keyframes con el nombre de “capa”, estos modificarán las propiedades que se asignen dentro de ellas. Y como aquí sólo se trataba de pasar de un punto a otro, utilicé las sentencias “from” a “to”. En esta capa el valor que cambié para la animación es la propiedad opacidad (opacity) para que una vez que la animación termine, el fondo vaya disolviéndose suavemente hasta desaparecer (ver figura 7.6).

```
@-webkit-keyframes capa {
  from {opacity:1;}
  to {opacity:0;}
}

@keyframes capa {
  from {opacity:1;}
  to {opacity:0;}
}
```

7.6 Keyframes en CSS para el desvanecimiento de la capa de presentación

Como nota, cabe señalar la importancia de incluir las dos opciones, pues no todos los navegadores en todas las versiónes lo pueden interpretar solamente con “@keyframes” por lo que se recomienda adicionalmente poner “@-webkit-keyframes”, y ambas pueden llevar el mismo nombre “capa”. Como es una característica propia de CSS3, no todos los navegadores podrán visualizar la animación, pero al final no es indispensable que la animación funcione, solo es un detalle que confiere más calidad al sitio.

Ahora, volviendo a la propiedad opacidad, ésta solamente hace aumentar o disminuir la transparencia, pero el elemento sigue presente lo que inhabilita al visitante a acceder a los elementos detrás de ella, por tanto es necesario cambiar también propiedad de tipo de caja (display) a nada (none) de modo que, cuando acabe la animación, el elemento no esté desplegado y el visitante pueda interactuar con el contenido de abajo. Para cambiar esta propiedad generé una función llamada intro() en JS que se ejecuta al terminar de cargar la página; ésta prepara primero el contenido posterior, después genera un retraso de una duración igual a la de la animación y una vez terminada, modifica la propiedad tipo de caja a nada con la instrucción document.getElementById(“intro”).style.display = “none”. Así puedo asegurar que, independientemente del navegador que se use, mientras se tenga activado JS, el visitante podrá visualizar la página con todos los elementos cargados. Ya solo resta asignar las propiedades de la animación para determinar su comportamiento, es decir el nombre, el tiempo de duración, el sentido y el retraso (ver figura 7.7 y 7.8).

```
function intro(){
 redim();
 mnsDelay();
 detectCookie();
 setTimeout(function(){
  document.getElementById("intro").style.display = "none";
  navegadorDetect();
 }, 4000);
}
```

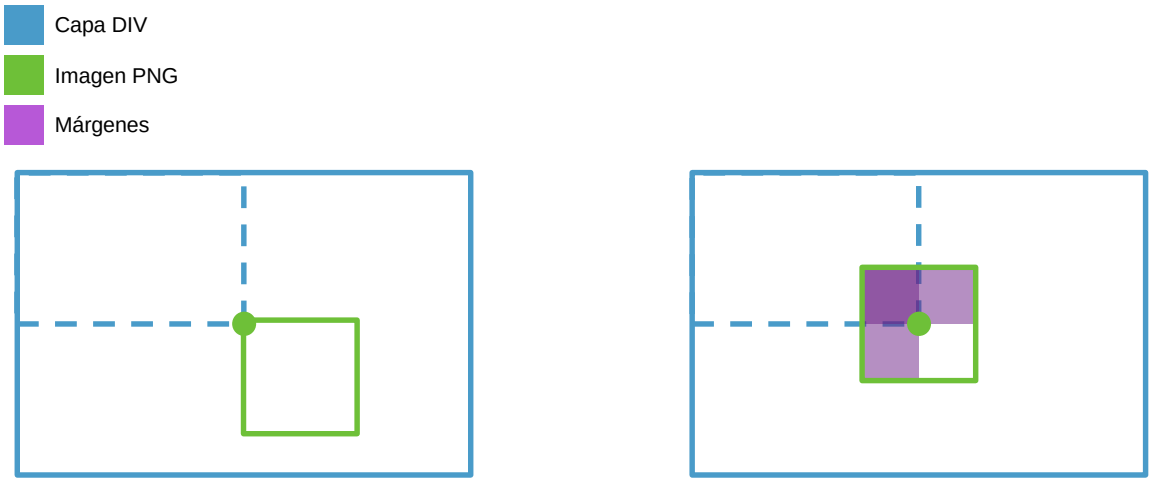
7.7 Función de JS que ejecuta las funciones esenciales del motor

```
#intro, #bkgd, #cabezera, #cuerpo, #pie{
 min-width:980px !important;
}

#intro{
 width:100%;
 height:100%;
 background:#000;
 position:absolute;
 top:0px;
 left:0px;
 z-index:2;

 -webkit-animation-name: capa;
 -webkit-animation-duration: 2s;
 -webkit-animation-delay: 2s;
 -webkit-animation-fill-mode: forwards;

 animation-name: capa;
 animation-duration: 2s;
 animation-delay: 2s;
 animation-fill-mode: forwards;
}
```

7.8 Código CSS para la capa de presentación

Igual que con los keyframes, es recomendable utilizar las propiedades con -webkit-animation para los navegadores que así lo requieran. Hasta ahora está preparado solamente el fondo donde va a estar la animación para que se muestre y se oculte después de ejecutarse, lo siguiente es realizar la animación de los elementos del logotipo de la empresa. Dentro del elemento con ID intro generé los otros dos elementos de imagen que, como había dicho antes, componen el logo cuando se juntan; los archivos de imágenes son PNG con transparencia y tienen el mismo tamaño de 500px, aunque el contenido no abarque todo el espacio. Este punto es importante para que más adelante se puedan centrar y no se distorsionen o se desfasen y para que sea más sencillo alinearlos. Una vez agregados, lo siguiente fue asignarles las propiedades para que se colocaran al centro del elemento padre (intro), y para ello asigné: primero la propiedad de ancho en 500px, esto con el fin de asegurar que la imagen siempre se mantendrá del mismo tamaño, al ancho, y por defecto, al no tener especificada la propiedad de alto. Su valor es en proporción al ancho, posición absoluta con sus valores de desplazamiento de arriba e izquierda en un 50%, esto hará que los elementos estén centrados pero con respecto a su punto de origen que es la esquina superior izquierda, lo que hará que se vean ligeramente descentrados. Para corregir esto les aplicaremos a los elementos un margen negativo del tamaño de la mitad de su ancho y alto en los márgenes izquierdo y superior respectivamente; tal cosa

desplazará su punto de origen al centro del elemento y quedará centrado de manera precisa en la ventana como se muestra en la figura 7.9.




7.9 Cómo centrar una imagen con márgenes

Una vez colocados los objetos en su lugar, agregué la animación al elemento del engrane, que en realidad es el único que se mueve. Al igual que en el caso de su elemento padre, creé otros bloques de keyframes denominados “anim2” para modificar la propiedad transformación (tranform) con el valor de rotar, que va del valor 0 al 359. Es recomendable agregar esta propiedad con las variantes “-webkit-” y ”-moz-” para que los navegadores más populares lo puedan interpretar. Listos los keyframes que utilizaremos para crear la animación, solo resta asignar las propiedades de ejecución. En estos elementos, a diferencia de su elemento contenedor, no necesitamos asignarle propiedades de tipo de caja, puesto que son elementos hijo del elemento “intro”, así que cuando éste cambia su propiedad del tipo de caja, tanto él como sus elementos hijos desaparecerán (ver figura 7.10).

```
@-webkit-keyframes anim2 {
 from {
  -ms-transform: rotate(0deg);
  -webkit-transform: rotate(0deg);
  transform: rotate(0deg);
 }
 to {
  -ms-transform: rotate(359deg);
  -webkit-transform: rotate(359deg);
  transform: rotate(359deg);
 }
}

@keyframes anim2 {
 from {
  -ms-transform: rotate(0deg);
  -webkit-transform: rotate(0deg);
  transform: rotate(0deg);
 }
 to {
  -ms-transform: rotate(359deg);
  -webkit-transform: rotate(359deg);
  transform: rotate(359deg);
 }
}

…

#anim1, #anim2{
 position:absolute;
 top:50%;
 left:50%;
 margin:-250px;
 width:500px;
}

#anim2{
 -webkit-animation-name: anim2;
 -webkit-animation-duration: 2s;
 -webkit-animation-timing-function: ease-in-out;
 -webkit-animation-fill-mode: forwards;
 animation-name: anim2;
 animation-duration: 2s;
 animation-timing-function: ease-in-out;
 animation-fill-mode: forwards;
}
```

7.10 Código CSS para la animación de la imagen dentro de la capa de presentación

El primer elemento del sitio web que creé fue el uno DIV con el ID “bkgd”, un carrusel de imágenes que adopta el tamaño de la ventana sin deformar su relación ancho-alto, con pase de imágenes automatizado y sin controles. Funciona como un fondo animado y el cambio de imágenes es visible para el visitante en elementos con trasparencias: la cabecera y el pie de página, además de los espacios que dentro del elemento cuerpo que se generan, dependiendo de la sección en la que uno se encuentre. Para ello, primero tuve que preparar el elemento colocándolo detrás de todos los restantes pero asegurando que no tapara ninguno a la hora de asignarle los niveles de profundidad, usando el valor de “-1”

pues, por defecto, los elementos creados reciben un valor “0”. Luego le asigné las dimensiones para que cubriera toda la ventana del sitio y para que el carrusel cubriera con las imágenes su totalidad, sin dejar huecos. En ancho y alto le di los mismos valores que los de la ventana, recurriendo a 100% o heredado (inherit); esto se puede hacer porque el elemento es hijo del elemento BODY, por lo que tomará su misma dimensión. Una vez fijo en la posición deseada, hice que el elemento partiera desde el origen, a fin de que, en conjunto con el ancho y alto, pudiera cubrir toda la ventana. Finalmente, le asigné los valores de inicio en lo que respecta a las imágenes; para eso utilicé la propiedad de fondo (background) para poner la primera imagen, solo para asegurarme de que, en el caso de que el navegador no soporte las animaciones de CSS3, cargue ante al visitante una imagen de fondo estática (ver figura 7.11).

```
#bkgd{
 height: inherit;
 width: 100%;
 position: absolute;
 top:0px;
 z-index:-1;

 background-image:url(img/bk1.jpg);
 background-size: cover;
 background-repeat: no-repeat;

 -webkit-animation-name: bkgd;
 -webkit-animation-delay:5s;
 -webkit-animation-duration: 30s;
 -webkit-animation-timing-function: ease-in-out;
 -webkit-animation-fill-mode: both;
 -webkit-animation-iteration-count: infinite;

 animation-name: bkgd;
 animation-delay:5s;
 animation-duration: 30s;
 animation-timing-function: ease-in-out;
 animation-fill-mode: both;
 animation-iteration-count: infinite;
}
```

7.11 Código CSS para la animación del carrusel del fondo de página

Ya establecidas las propiedades iniciales para este elemento, lo siguiente fue agregar las animaciones. Para una animación con paso de imágenes generé los keyframes con nombre “bkgd” usando @keyframes y @-webkit-keyframes. En su interior las propiedades que cambian son las relacionadas con el fondo; como ya había dicho antes, el elemento en realidad ocupa toda la ventana y se ubica hasta atrás, por lo que visualmente no se nota la diferencia. Lo creé así y no con elementos IMG para evitar el uso de librerías externas (como jQuery, Woo u otras); y para ahorrarme el código de programación en JS.

El empleo de librerías, aparte de tedioso, me obligaba a utilizar varias líneas para cubrir casos particulares, asignar valores determinados, realizar las animaciones y generar excepciones para evitar fallas de despliegue. La segunda razón de mi decisión fue que, a través de CSS3, puedo determinar que la imagen cubra todo el espacio de una manera sencilla, sin preocuparme por la posible deformación de la imagen en relación ancho-alto y aunque podría generar más líneas que en JS para incluir más imágenes, es repetitivo y más sencillo. Las propiedades que agregué dentro de los keyframes para generar el cambio de imágenes fueron: el fondo con el valor de la URL del archivo de imagen que adoptará la forma en que va a desplegar la imagen, a la que asigné el valor de cubrir (cover); así, la imagen rellenará todo el elemento, manteniendo, de manera sencilla y rápida, la relación de ancho y alto de la imagen. Por último, solo asigné la propiedad de no repetir la imagen de fondo. Ya generado eso solamente restó repetir el mismo fragmento, cambiando la url de la imagen y el porcentaje, al momento de ejecutarse la siguiente imagen. También fue importante agregar un último keyframe con la misma URL de la imagen que el primero, con la intención de que no se note el final del carrusel. Finalmente, en la asignación de porcentajes de cada keyframe pueden fijarse los valores al gusto; yo opté por poner una cantidad igual al número que resulta de dividir cien entre el número transiciones o bien, en lugar de transiciones, el número de imágenes menos una que hay en el carrusel (ver figura 7.12).

```
@keyframes bkgd {
 0% {
  background:url(img/bk1.jpg);
  background-size: cover;
  background-repeat: no-repeat;
 }
 25% {
  background:url(img/bk2.jpg);
  background-size: cover;
  background-repeat: no-repeat;
 }
 50% {
  background:url(img/bk3.jpg);
  background-size: cover;
  background-repeat: no-repeat;
 }
 75% {
  background:url(img/bk4.jpg);
  background-size: cover;
  background-repeat: no-repeat;
 }
 100% {
  background:url(img/bk1.jpg);
  background-size: cover;
  background-repeat: no-repeat;
 }
}
```

7.12 Keyframes en CSS para la animación del fondo de la página

Al ser redimensionable la cabecera, es común que los objetos dentro de ella puedan cambiar su posición, según sus propiedades. El elemento lleva en su interior un elemento con ID “logo” y otro con “nav” y ambos adoptan el mismo tamaño de ancho que el padre, cubriendo todo el ancho de pantalla. La intención de generar estos tiene que ver con separar una parte de la otra; el primero contiene el logo de la empresa y las ligas a otros sitios web relacionados con ella y el segundo contiene todas las ligas que cargarán el contenido propio del sitio web. El procedimiento fue como sigue: generé un elemento DIV con el ID de “cabecera” y le asigné propiedades con un fondo de color negro, una transparencia y un efecto de sombreado para darle diseño vistoso; tratándose de un elemento DIV por defecto tiene un ancho de 100% y se muestra en forma de bloque, lo que evita asignarlos de nuevo. Esto genera el cuadro que se redimensione respecto de la ventana en su interior. Dentro del cuadro generé primero el elemento DIV con ID “logo” y le asigné las propiedades del archivo de estilo, los márgenes con el valor de 0px y una altura definida de 120px a fin de hacer espacio para los elementos internos; agregué luego el elemento IMG que mostrará la imagen del logo de la empresa cargado a la izquierda; este elemento recibe la propiedad de cursor en pointer que, aparte de mostrar el logo, servirá además para mandar al visitante

a la sección de inicio y a tales fines generé una función en JS que se ejecutará cuando el visitante haga clic sobre la imagen, con ayuda del el atributo “OnClick”.

Dicha función se encuentra en el archivo script que hace las veces de motor de nuestro sitio y lo cuya primera operación es limpiar todo el contenido dentro del elemento DIV con ID “cuerpo” localizando el elemento a través de la DOM (Document Object Model) y escribiendo con un valor vacío. Después, cambia el contenido del elemento con ID “seccion” con el texto de “Mantenimiento” que es el elemento que tenemos abajo y el que nos da más información de en qué parte nos ubicamos dentro del sitio web. Lo siguiente que hace es actualizar el contenido de la etiqueta TITLE con el texto “Resta Tools: Inicio” éste sirve para mostrar la página en que nos localizamos en la ventana del navegador; como en este caso solo se trata de una página, tuve que ir cambiando de forma manual el valor para las distintas cargas que el visitante quisiera hacer de contenido dentro del sitio web. El siguiente paso ejecuta una función llamada “controlMensajes” con el valor de true; esta función activa o desactiva los mensajes noticias que pasan al pie de la página, y más adelante explicaré cómo funciona. Finalmente ejecuta otra función que se llama activeNav que cumple la misma tarea que en el proyecto pasado; cuando se le ingresa la cadena como parámetro la función buscará el elemento que tenga de valor esa cadena y le agregará el atributo “activo” y removerá el atributo en cualquiera de] las (ver página 70)]. En este caso, el parámetro es vacío, por lo que remueve de todos los botones del menú de navegación el atributo activo. Esto está pensado para que el visitante, que ya había cargado alguna otra sección y regrese al inicio la liga que conducía a ella, la encuentre libre nuevamente para reingresar si así lo desea (ver figura 7.13).

```
function limpiar(){
	document.getElementById("cuerpo").innerHTML="";
	document.getElementById("seccion").innerHTML="Mantenimiento";
	document.getElementsByTagName("title")[0].innerHTML="Resta Tools: Inicio";
	controlMensajes(true);
	activeNav("");
}
```

7.13 Función de JS para restaurar el contenido a la sección principal

Después del elemento IMG, generé dos elementos de ancla que coloqué en el extremo derecho de la cabecera, usando la propiedad de flotante (float) hacia a la derecha, a fin de para no cargar muchos elementos del mismo lado del logo y de que el sitio web este balanceado. El primero lleva a la página de Restatools en el sitio web de Facebook en una página nueva con el atributo de (Target) con valor en _blank; dentro de éste hay un

elemento tipo IMG, con una imagen del logo de Facebook para indicar a los visitantes que la página que se puede cargar es de terna del sitio web y proviene de Facebook. El siguiente es también un elemento de ancla que liga con el perfil de la empresa de Restatools en el sitio web de LinkedIn, también le asigné el atributo (Target) con el mismo valor y, de igual manera, contiene un elemento IMG con la imagen de la empresa a la que lleva el ancla.

El menú de navegación es un elemento de lista con un ID nombrado “nav”; por criterios estéticos le di un fondo de color amarillo con degradados (si el navegador puede interpretarlo), con texto negro y sin transparencias, que funcionará como nodo padre que, aunque no tiene nodos hijos en el código original. Al igual que en el anterior y con la ayuda de un archivo script generará el código que se implanta en el elemento; éste centrará los elementos hijos mediante la propiedad de alineamiento de texto (text-align) en conjunto con la propiedad de tipo de caja como bloque y seguido (block-inline) en los elementos hijo que sería los objetos de lista LI. Como en los casos antes descritos, es importante que el script se ejecute después de que el cliente haya recibido todos los elementos para que el script encuentre el elemento en el que va actuar. Por la misma razón, la ejecución del archivo JS se hace al final del elemento BODY (ver figura 7.14 y 7.15).

```
<!doctype html>
<html>
…
<body onResize=“redim()">
 <div id="cabezera">
  <div id=“logo">
   …
  </div>
  <ul id="nav"></ul>
  </div>
…

<script src="script.js" type="text/javascript"></script>
<script src="model/fileupload.js" type="text/javascript"></script>
</body>
</html>
```

7.14 Código HTML del archivo Principal

```
…
document.getElementById(“nav”).innerHTML+='\
 <li onClick="quienes()" >Quienes somos</li>\
 <li onClick="porque()" ><span>¿</span>Porque nosotros<span>?</span></li>\
 <li onClick="servicios()" >Servicios</li>\
 <li onClick="unete()" >Unete al equipo</li>\
 <li onClick="contacto()" >Contacto</li>‘;
…
```

7.15 Código de JS para la generación de enlaces en el menú de navegación

Cada objeto de lista asignado al menú de navegación tiene un atributo que ejecuta una función particular y cada una cargará el contenido específico de esa sección para ser sobrescrita en el cuerpo del sitio web, que se ejecutará cuando el visitante de clic en alguno de estos objetos de lista. La otra forma para que el visitante pueda cargar nuevamente información supone el uso de anclas, pero esto nos obliga a movernos a otras páginas o a algún punto en específico; en este caso, como se buscaba no generar más páginas, era necesario que sólo se desplegara el contenido deseado, así, en el momento en que el visitante quisiera cambiar de sección, había que reemplazar el contenido del cuerpo de la página con otro nuevo. Y la forma más sencilla de hacerlo era a través de scripts que generaran cambios sin moverse de la página.

Como en nuestro menú de navegación específicamente no existen anclas fue indispensable hacer pasar los elementos LI como ligas; para eso usé la propiedad de cursor y lo asigné con el valor de (pointer), de tal modo que cuando el visitante pase el ratón sobre los objetos de lista, el cursor cambiará de forma y tomará el de una liga, y cuando de clic la función se ejecutará cambiando el contenido. También es necesario cambiar esta propiedad para el objeto de lista en el que el visitante esté visualizando el contenido; como había dicho antes, al usar la función “activeNav()” ésta buscará la liga que fue seleccionada y tendrá que desactivarla. Como señalé en otra parte, para lograr esto, el objeto que debe debilitarse es aquel que tenga el atributo “activo”, por tanto, solo es necesario emplear el selector de atributo, para especificar que todo elemento que tenga tal atributo no deberá responder. En mi caso, especifiqué que el elemento LI con tal atributo recibiría esas propiedades escribiéndolo como “#nav li[activo]”, ya que tenemos otras ligas fuera del menú de navegación que cargarán contenido, como es caso del logotipo o bien las ligas que abren páginas externas como las anclas para las redes sociales y empresariales (ver figura 7.16).

```
#nav{
 height:100%;
 padding:0px;
 margin:0px;
 background:#EDC61D;
 background: -webkit-linear-gradient(#EDC61D , #D79508 , #EDC61D); /*EDB745*/
 background: -o-linear-gradient(#EDC61D, #D79508, #EDC61D);
 background: -moz-linear-gradient(#EDC61D, #D79508, #EDC61D);
 background: linear-gradient(#EDC61D , #D79508, #EDC61D);
 list-style:none;
 text-align:center;
}

#nav li{
 display:inline-block;
 …
 text-align:center;
 cursor:pointer;
}

#nav li[activo]{
 pointer-events:none;
 cursor:default;
}
```

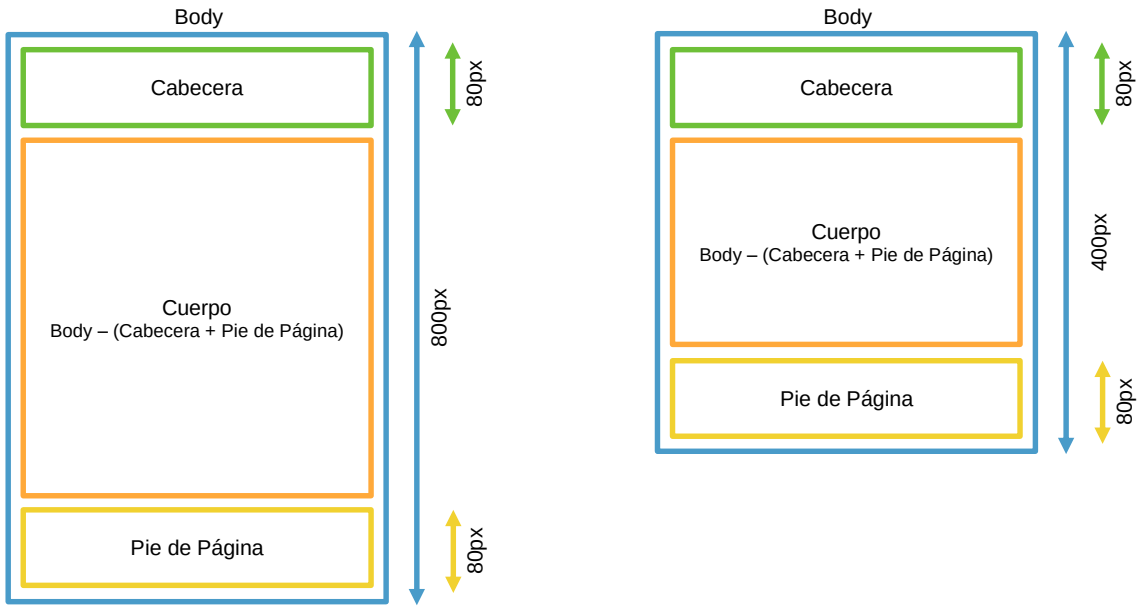
7.16 Código CSS para la asignación de propiedades en el menú de navegación

El cuerpo está formado por un elemento DIV con un nombre de ID igualmente llamado “cuerpo”; éste, con ayuda del motor escrito en JS, inserta toda la información que corresponda a cada sección. Su diseño es muy simple: es sólo un contenedor en cuyo interior recibirá nuevos elementos hijos que informarán la sección que el visitante quiera visualizar. Este elemento recibe la propiedad de altura con un valor calculado de modo que reparta los elementos de cabecera, cuerpo y pie de página sobre toda la ventana del navegador, sin importar las dimensiones, proporciones o dispositivo en el que se le esté visualizando. Para dar al elemento la altura necesaria, generé una función de JS con el nombre de “redim()” dentro del motor donde la función calcula las alturas de la cabecera, el pie de página y la ventana, de tal manera que mantenga siempre todos los elementos repartidos por toda la ventana (ver figura 7.17).

```
function redim(){
 cabezera = document.getElementById("cabezera").clientHeight;
 pie = document.getElementById("pie").clientHeight;
 bodytag.style.minHeight=window.innerHeight+"px";
 cuerpo.style.height = (window.innerHeight-cabezera-pie)+"px";
}
```

7.17 Código JS para la asignación de la altura del elemento “cuerpo”

Generada la función, la programé para que se ejecute bajo las siguientes condiciones: primera, al cargar la página; en este caso, la función se ejecuta dentro de la función intro() antes vista que servía para preparar la página web una vez cargado todo el contenido (ver imagen 7.7). Segunda, cuando la ventana del navegador sufra un cambio en su tamaño o proporción, para esto simplemente ejecuté la función “redim()” con el atributo de al redimensionar sobre la etiqueta body de esta manera “<body onResize=“redim()”>”; así, cuando la ventana cambie la función empezaré de nuevo el cálculo de los elementos y la ventana, redimensionando el elemento con ID “cuerpo” con la nueva altura (ver figura 7.18).




7.18 Ilustración de cómo se calcula la altura del elemento cuerpo

Adicionalmente, agregué propiedades a este elemento para controlar su comportamiento bajo condiciones poco frecuentes que pudieran causar errores a la hora de visualizar el sitio web. Debido a que el elemento cuerpo recibe la propiedad de altura mediante el cálculo entre el tamaño de la ventana y los elementos adyacentes a él, es necesario limitar esta acción a aquellos casos en que la altura de la ventana sea demasiado pequeña, de tal suerte que el elemento cuerpo tenga una altura igual o cercana a 0px, lo que lo haría incómodo de visualizar. En dicho caso es preferible que el elemento cuerpo sea mayor que el tamaño de ventana, por lo que asigné una orden para que se muestre la barra de desplazamiento, a fin de que el visitante pueda moverse a lo alto de la ventana de

manera más cómoda. Con tal objetivo utilicé la propiedad de altura mínima y le asigné el valor de 320px, éste lo calculé más o menos al tamaño del contenido de todo el sitio web. Así, aunque el valor de la propiedad de altura de este elemento cambie con respecto a la altura de la ventana, si queda por debajo del valor de esta propiedad, el elemento cuerpo quedara con altura de 320px.

Habiendo hecho esto, procedí a ocuparme de aquellos elementos internos que fueran más grandes que el elemento en sí, provocando que su salida del cuadro o su superposición a otros, como el pie de página o sección. Para impedirlo, me valí de la propiedad de desborde(overflow), con el valor de oculto; de tal forma que todos los elementos que sobresalgan del cuerpo, sea parcial o completamente, serán recortados y ocultados. Para evitar conflictos de este elemento con el pie de página, agregué un margen inferior del tamaño del pie de página; usando esta propiedad, junto con la de posición absoluta sobre el pie de página, creé un área apartada para evitar superposiciones sobre el elemento cuerpo; independientemente de la altura del último, este espacio apartado siempre será del mismo tamaño que el del pie de página, lo que servirá para que al desplazarse el pie de página siempre aparezca inmediatamente después del elemento cuerpo (ver figura 7.19).

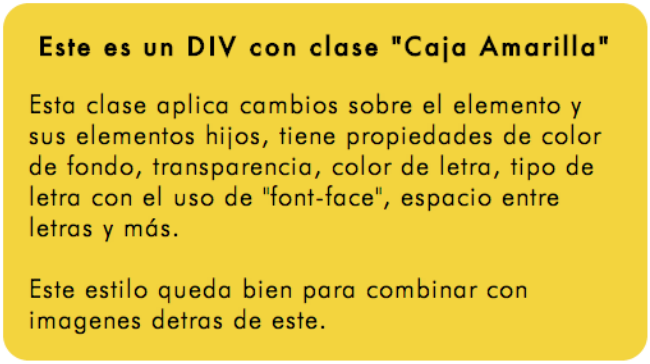
```
#cuerpo{
 min-height:320px;
 margin-bottom:170px;
 overflow:hidden;
}
```

7.19 Código CSS para el elemento “cuerpo”

Este elemento contendrá en su interior lo necesario para desplegar la información que el usuario quiera visualizar con el uso de las anclas en el menú de navegación. Diseñé su contenido de tal modo que se desplegara de dos maneras, dependiendo de la sección seleccionada. La primera es un diseño sencillo que consta de recuadros amarillos con una ligera transparencia, letras negras para resaltar el texto y esquinas redondeadas; este diseño lo utilicé para secciones que contienen casi solo texto y pocas figuras. Su objetivo fue que el fondo del sitio, que es el carrusel de imágenes, hiciera su trabajo al rellenar y dar vista a estas secciones que son ligeras y pueden contener mucho espacio de sobra. Con tal medida, los visitantes pueden leer los contenidos sin experimentar la sensación de tener frente a sí un vacío, además, es un recurso para hacer a este sitio web distinto respecto de otros.

Para que estos elementos adoptaran tal opción, generé elementos DIV y agregué el atributo clase con el nombre de “CajaAmarilla”; en el archivo de hoja de estilo agregué los bloques de código con las características antes descritas, amén de otras adicionales que le confieren mayor calidad visual. Debido a que hay transparencias y a que está ideado para un contenido de poco texto, le agregué propiedades de texto como tamaño y espacio, a fin de que la lectura resultara más sencilla y de atajar las posibles incomodidades derivadas del color y tipo de fuentes con imágenes al fondo. Además, agregué márgenes de 20px para los elementos internos, en combinación con la propiedad de desbordamiento(overflow) que genera márgenes dentro del cuadro, de forma que el contenido tenga siempre un espacio seguro y abarque el fondo del contenedor. El primer y el último elemento dentro del elemento con la clase “CajaAmarilla” utiliza las propiedades de márgenes superior e inferior y todos incluyendo el primero y último tienen márgenes laterales debido a que los márgenes con el valor de 20px (ver figura 7.20).

```
.CajaAmarilla{
 background:rgba(239,198,0,0.75);
 margin:20px;
 border-radius:20px;
 overflow:auto;
}

.CajaAmarilla *{
 font-family:futura !important;
 font-size:20px;
 letter-spacing:1px;
}

.CajaAmarilla > *{
 margin-left: 20px;
 margin-right: 20px;

}

.CajaAmarilla > *:first-child{margin-top:20px;}
.CajaAmarilla > *:last-child{margin-bottom:20px;}
```



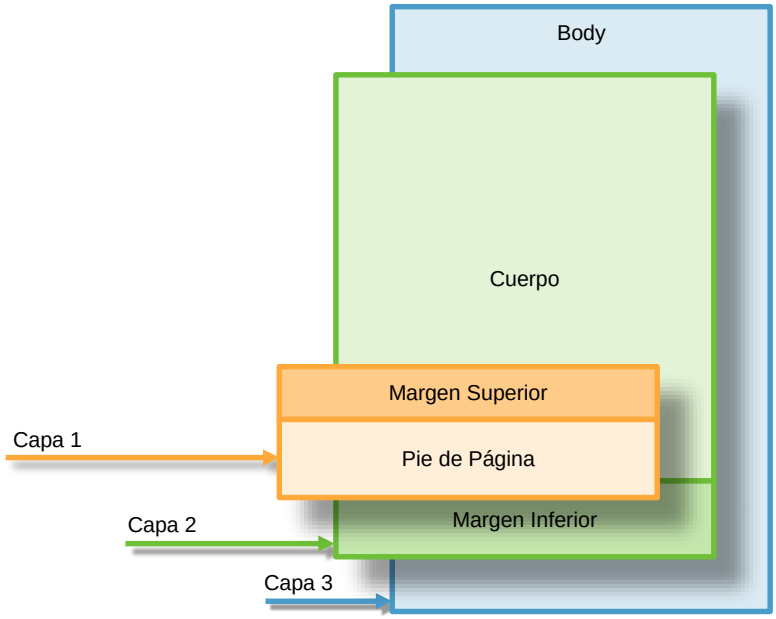

7.20 Código CSS y vista previa para las clases “CajaAmarilla”

En la segunda generé un elemento DIV que rellenará todo el espacio dentro del elemento cuerpo, ya sea de algún color o bien con algún fondo; empleé este diseño para las secciones que llevarán muchos elementos visuales o bastante texto o, bien, para las partes donde queremos evitar que el visitante se distraiga con el fondo del sitio web. A este DIV adicional le asigné las propiedades de fondo, con una imagen que lo cubra por completo o bien con algún color de relleno. Además de agregar las propiedades de margen y relleno con un valor de 0px para ocupar totalmente el espacio, este DIV a su vez puede contener

elementos internos DIV con las propiedades de “Caja Amarilla” como más adelante describiré en algunos ejemplos

Después le sigue un elemento DIV con su ID “seccion” que, como se dijo antes, solo desplegará un texto que indique al visitante en qué sección se halla dentro del sitio web. El texto, al cargarse la página tiene por defecto “Mantenimiento”; no obstante, cuando el usuario presiona una de las anclas del menú de navegación, ejecuta una de las funciones dentro de ella y una de las instrucciones es el cambio del texto, usando la instrucción innerHTML() sobre este elemento.

Finalmente, el pie de página es un elemento DIV con su ID “pie”, con varios elementos en su interior, aunque algunos no están desplegados en un principio. Al igual que al elemento cabecera, a éste le asigné un fondo negro con transparencia a fin de que luciera. Y como el fondo es negro, al color del texto se le asignó el blanco; le di valor de relleno en 0 para aprovechar todo el espacio dentro del elemento; en cuanto a márgenes asigné igualmente 0, salvo el superior, al que di un valor de 25px para de que el texto del elemento cuerpo no estuviera muy cerca al del pie de página y se viera encimado. Además, como este elemento lleva en su seno otros que se moverá dentro de él de un extremo a otro, y como no deseamos que estos sean visibles después de salir del elemento pie, asigné la propiedad de desbordamiento (overflow) en oculto. Por último, este elemento se posiciona al fondo de la página con ayuda de las propiedades de posición y las referencias en las que los valores son absolutos y desde el fondo a 0px respectivamente. Esto nos ayuda a colocar siempre el elemento al final de la página y con el espacio libre generado en el elemento cuerpo, este elemento siempre quedará posicionado después del contenido, además de que con la propiedad de posicionamiento como absoluto servirá para que la posición de los elementos internos a este sea absoluta con referencia a este elemento y no toda la ventana (ver figura 7.21).




7.21 Ilustración donde se muestra el espacio reservado para el pie de página con el margen inferior del elemento cuerpo.

El primer elemento dentro del pie de página es un DIV con ID de “contactos”; éste agrupa varios elementos de imágenes y de anclas destinados solamente a la sección de contacto. Por tanto y, por defecto, el tipo de caja no es desplegado hasta que se ejecute la función de “controlMensajes” con el valor como parámetro en false. El papel de esta función es asignar los valores del tipo de caja (display) de los elementos que tienen la clase “detalles” del elemento con ID “contactos”, ya sea que el resultado fuera en bloque y oculto u oculto y bloque respectivamente. En el caso del primero, como es búsqueda por clase, realicé un ciclo con el tamaño del número de elemento con la clase y dentro de ella le asigné el valor dado para cada elemento en el que se encontraba el ciclo, barriendo así todos los elementos existentes de esa clase. De este modo, si es necesario agregar más elementos a las noticias ya no hará falta alterar la función para los nuevos elementos. Ahora bien, para el elemento con el ID de “contactos”, como es único e irrepetible, no hay arreglos y no es necesario generar un ciclo; en consecuencia, a ése directamente le asigné el valor que le correspondiera. Esta función es un fragmento de código que mantiene el control de lo que el pie tiene que desplegar, ya sea las noticias que corren de un lado al otro o bien un pequeño submenú con archivos de contacto, anclas y datos que despliegan sobre el elemento sección cuando el mouse pasa sobre el elemento. El elemento con ID “contactos” solamente sirve para agrupar a este conjunto de elementos que fueron planeados para ser

desplegados en la sección de contactos y reemplazan las noticias que pasan en el pie de página (ver figura 7.22).

```
function controlMensajes(bandera){
 if (bandera){
  mns="block";
  cnt="none";
 }
 else{
  mns="none";
  cnt="block";
 }

 for (i=0; i<document.getElementsByClassName("detalles").length;i++)
  document.getElementsByClassName("detalles")[i].style.display=mns;
 document.getElementById("contactos").style.display=cnt;
}
```

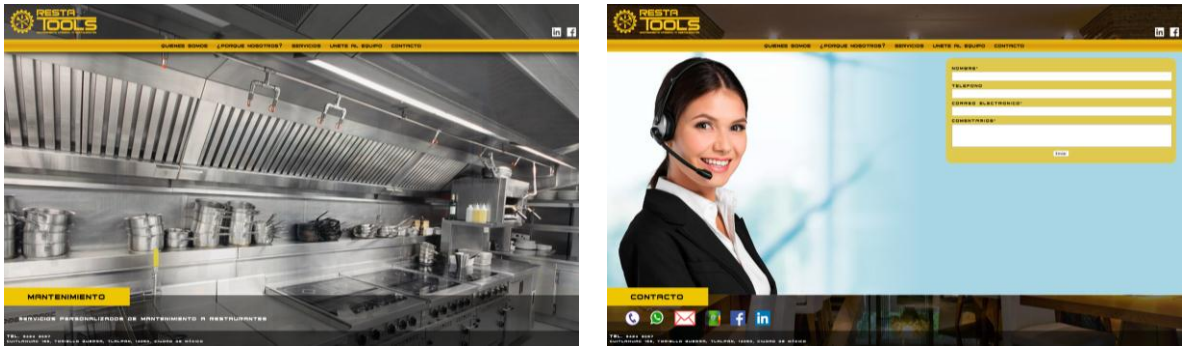
7.22 Código JS para el control de modalidad del elemento “seccion”

Dentro del elemento contactos tenemos primero uno IMG que despliega la imagen de un teléfono, su finalidad es cambiar el texto de la etiqueta seccion por el número telefónico de la empresa al colocar el cursor sobre la imagen y restaurar la etiqueta con el texto “Contactos” al moverse el cursor fuera de ella. A fin de que sobre la sección pudiera escribirse el número telefónico hice una función denominada “reescribir” y ella hará la tarea de alterar el texto que figure en tal elemento; el uso del atributo “onMouseOver” sirve para detonar una acción cuando el cursor este posicionado sobre el elemento con este atributo, por lo tanto con él y con la función antes descrita generé la acción de modificar el texto cuando el cursor se pose sobre el elemento. La cuestión es que, una vez que se ejecuta la función, el valor anterior se pierde y para conseguir que el texto volviera a su estado original una vez que el cursor dejara de estar sobre el elemento, recurrí a otro atributo, “onMouseOut” que detecta el momento en que el cursor se aparta para ejecutar la misma función, pero con el parámetro del texto original. Así, cuando el visitante pase el cursor sobre el elemento, el texto en el elemento sección cambiará al contenido asignado y se restaurará una vez que el cursor salga de él. La función reescribir es simplemente una instrucción que localiza el elemento sección y reescribe el parámetro dado a la función; razón de haberla introducido es simplificar un poco el código, ya que se usa mucho en todos los elementos internos al elemento sección. Y a fin de que en cada elemento no hubiera cadenas de texto muy largas como atributos, utilicé la función para acortar la evocación de esta instrucción y, por tanto, hacer el código HTML más fácil de leer. El siguiente elemento es un IMG con la imagen del logotipo de Whatsapp que, al igual que el pasado, cuando el visitante posa el cursor sobre él ejecuta la función reescribir con el valor del número de

celular de la compañía y cuando el cursor abandona la imagen se restaura, al igual que el primero. Después le sigue un elemento IMG con la imagen de un sobre de correo, este elemento desplegará en el elemento sección el correo de la empresa mediante el cual los visitantes se pueden poner en contacto con la compañía. El que le sigue a diferencia de los primeros, es un elemento de ancla que hace referencia a un archivo VCF que contiene el contacto de la empresa. Este archivo está diseñado y planeado para que los visitantes que accedan al sitio web en su dispositivo móvil, al seleccionar este icono automáticamente descargue y se agregue a su agenda de contactos los datos de la empresa, incluyendo teléfono, correo etcétera. Si el equipo para el acceso es una computadora, la información se descargará e instalará en la agenda de contactos que tenga instalada, tal como Outlook de Microsoft, Agenda de Apple, etcétera. De manera semejante a los anteriores, si el cursor se coloca sobre el ancla el texto del elemento sección, cambiará a un texto que dice "Agéndanos". Falta añadir que en el interior de esta ancla generé un elemento IMG con la imagen de una agenda para que el ancla responda con la imagen para su descarga y en la actualización del texto. Finalmente, los dos últimos elementos son anclas que apuntan a páginas externas, el primero es una que hace referencia la página de la empresa dentro del sitio web de Facebook, en una página en blanco y cuando el mouse se coloca sobre el elemento, su texto cambia a "Facebook" y dentro hay una imagen con el logro de la compañía. La segunda es un ancla que apunta a la página de la empresa dentro del sitio web de LinkedIn y, nuevamente, en una página en blanco, cuando el cursor se coloca sobre el texto cambian el elemento seccion a LinkedIn y contiene una imagen del logotipo de la empresa LinkedIn (ver figura 7.23 y 7.24).

```
…
<div id="pie">
 <div id="contactos" style=“display:none;">
  <img src="img/phone.png" alt="phone" onMouseOver="rescribir('5424 3067')" onMouseOut=“rescribir('Contacto')"/>
  <img src="img/whatsapp.png" alt="whatsapp" onMouseOver="rescribir('(55) 1355 3081')"
onMouseOut=“rescribir(‘Contacto')"/>

  <img src="img/email.png" alt="email" onMouseOver="rescribir('info<span>@</span>restatools.com.mx')"
onMouseOut=“rescribir(‘Contacto')"/>

  <a href="RESTA TOOLS.vcf" onMouseOver="rescribir('Agendanos')" onMouseOut=“rescribir('Contacto')">
   <img src="img/contacto.png" alt=“contacto"/>
  </a>
  <a href="https://www.facebook.com/restatools" target="_blank" onMouseOver="rescribir('Facebook')"
onMouseOut=“rescribir(‘Contacto')">

   <img src="img/fb-o.png" alt=“facebookdet"/>
  </a>
  <a
href="https://www.linkedin.com/company/10234668?trk=tyah&trkInfo=clickedVertical%3Acompany%2CclickedEntityId%3
A10234668%2Cidx%3A1-1-1%2CtarId%3A1445903652787%2Ctas%3Aresta%20tools" target="_blank"
onMouseOver="rescribir('LinkedIn')" onMouseOut=“rescribir(‘Contacto')">

   <img src="img/ln-o.png" alt=“linkedindet"/>
  </a>
 </div>
 <div id=“datos">
  Tel. <small>5424 3067</small><br>
  <small>Cuitlahuac 166, Toriello Guerra, Tlalpan, 14050, Ciudad de México</small>
 </div>
 <div class="detalles">Servicios personalizados de mantenimiento a restaurantes</div>
 <div class="detalles">Calidad, rapidez y eficiencia en tus servicios de mantenimiento</div>
 <div class="detalles">Todos tus servicios de mantenimiento en un solo lugar</div>
</div>
```

7.23 Código HTML en la parte de pie de página del archivo Principal



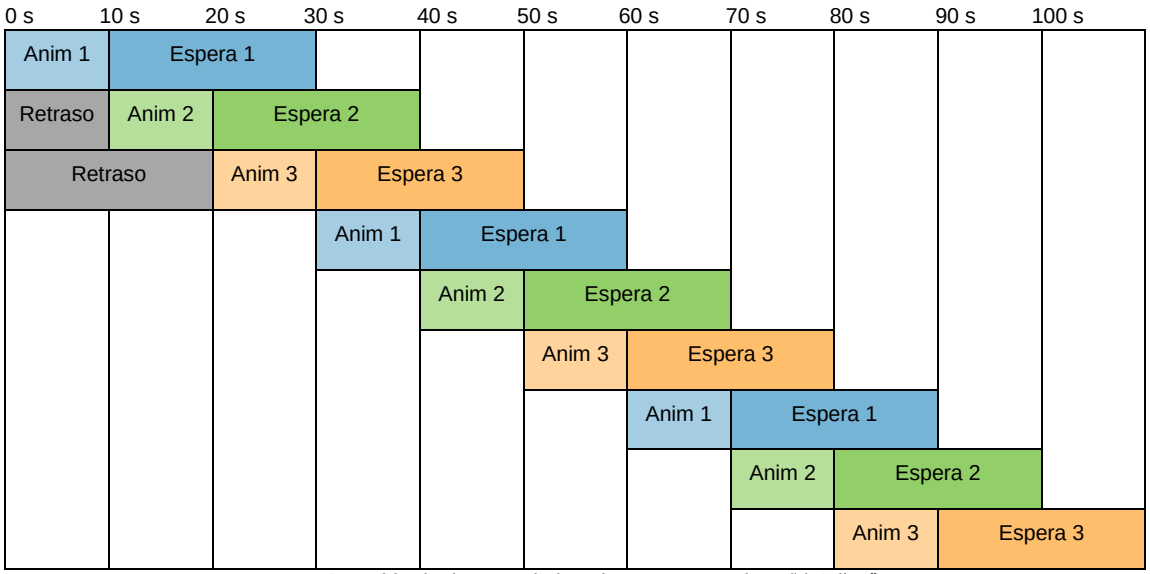

7.24 Captura de Pantalla de las dos modalidades del elemento “seccion”

A los contactos sigue un elemento DIV con ID de “datos” ; éste es estático y solo contiene elementos SMALL para proporcionar la dirección y teléfono de la empresa que pueden ser útiles en cualquier momento. Dentro del pie de página, éste es el último que debe salir a la hora de que el visitante lo visualice, y para que esto se cumpla, se le asignaron algunas propiedades de estilo. Inicialmente, se le asignó el ancho en un 100%

para asegurar que el siempre coincidirá con la anchura del pie de página; suprimí todo margen que pudiera haber y di el valor de 0 para evitar un desfase con respecto al elemento padre y también para impedir comportamientos inesperados, tales como activar la barra horizontal para desplazarse hacia un espacio vacío. Después asigne la colocación vertical dentro, con el uso de la posición en absoluto y tomando como referencia desde abajo a 0px; esto me asegurará que siempre será el último dentro del pie de página; el alto será el valor que necesite para desplegar de manera automática el texto que contiene. Ya colocado el elemento le asigné el tamaño de texto a 16px, un fondo negro con transparencia, un relleno para que el texto no quedara en las orillas y, por último, una envoltura del texto (text-wrap) para evitar el cambio de línea si no cabe todo el texto con el valor de nada (none) y (overflow) con valor de nada. Así, todo texto que salga del cuadro no será mostrado y se evitarán algunos errores de despliegue.

Por último generé tres elementos DIV con la clase “detalles”, al recibir dicha clase, estos elementos, el motor se encarga de prepararlos y asignarles las propiedades con los valores adecuados que se calcularon para cada elemento. Estos heredan propiedades de la clase “detalles” en el archivo CSS con keyframes que los preparan para su animación; a este respecto primero determiné el ancho que deberán tener los elementos y determinar cuánto sería necesario desplazar el elemento fuera del pie de página. Como había explicado antes, al tener el pie de página la propiedad de desbordamiento (overflow) como oculto y posicionamiento como absoluto, los elementos en su interior no se mostrarán si están fuera del elemento, ello independientemente de que el origen de estos elementos será el pie de página. Una vez asignado el ancho, desplaza todos los elementos en una cantidad mayor al del ancho fuera del elemento, usando las propiedades de posición. Por otro lado, en previsión de situaciones en las que el texto pudiera ser de dos renglones o más o se saliera del ancho asignado, utilicé la propiedad de envoltura de texto (text-warp) como nada, lo que impedirá errores. Una vez que estas propiedades son asignadas a los elementos, lo siguiente es generar el keyframe que van a tomar para la animación, es decir, el desplazarse desde un extremo al otro. En todos los casos, el keyframe fue el mismo, sólo que necesité calcular los cuatro estados en los que estaría cada mensaje. La primera transición fue preparación para entrada y el mensaje sale del extremo izquierdo, moviéndose hasta llegar al punto donde asigné el inicio de la segunda transición. Luego de que se ejecutara la animación de entrada, lo siguiente fue generar los keyframes para que el objeto se mantuviera estático por un tiempo determinado.

El tercer paso fue la animación de salida para el mensaje, que se deslazaría a la derecha hasta salir de la página. Finalmente, en el último paso el mensaje permanecerá en espera de que todos los mensajes hayan salido y se reinicie toda la animación. Este ultima transición tiene una duración de ⅔ debido a que hay que pasar tres mensajes y es necesario mantener posicionado el elemento una vez que ya salió, hasta que se acabe la traición del resto de los elementos (ver figura 7.25).




7.25 Tabla de tiempos de los elementos con clase “detalles”

La duración de la animación y el tiempo de retraso para estos elementos los generé en el motor con la función mnsDelay() que se ejecuta al cargar la página y aunque los elementos se oculten cuando entra la sección contacto, se mantendrán los valores de la animación. Esta función localiza todos los elementos que contengan la clase “detalles” y a cada uno le asigna la propiedad animación (animation) con el valor de duración igual al producto del número de elementos encontrados con clase “detalles” por diez y un retraso igual al producto del índice del elemento por diez. De esta manera genera el desfasamiento y la duración de los mensajes que se moverán de manera ordenada. Si quisiera agregar una cuarta, solo sería necesario cambiar los porcentajes en los keyframes, de forma que el número de elementos con la clase “detalles” sea igual al denominador, en este supuesto caso sería ¼ si agrego uno más (ver figura 7.26 y 7.27).

```
@-webkit-keyframes mensaje {
 0% {left:-100%;}
 3% {left:2%;}
 30% {left:2%;}
 33% {left:100%;}
 100% {left:100%;}
}

@keyframes mensaje {
 0% {left:-100%;}
 3% {left:2%;}
 30% {left:2%;}
 33% {left:100%;}
 100% {left:100%;}
}
…
.detalles{
 position:absolute;
 left:-100%;
 top:40%;
 font-size:20px !important;
 color:#FFF;
 text-transform:uppercase;
 width:100%;
 text-wrap:none;
 overflow:hidden;
}
```

7.26 Keyframes en CSS para la animación de las noticias

```
function mnsDelay(){
 for (i=0; i<document.getElementsByClassName("detalles").length;i++)
  document.getElementsByClassName("detalles")[i].style.animation="mensaje “+
  (document.getElementsByClassName(“detalles").length*10)+
  "s ease-in "+(10*i)+"s infinite normal forwards";
}

function intro(){
 redim();
 mnsDelay();
 …
}

window.onload=intro();
```

7.27 Código JS para la asignación de tiempos de animación para cada elemento con clase “detalles”

A continuación explicaré el contenido en cada sección de la página. Todas las secciones en este sitio web son código HMTL que se inyectan sobre el elemento con ID “cuerpo” y hacen algunas alteraciones en otros elementos para el correcto despliegue de la sección. Esto lo hice por medio de funciones en el motor que se ejecutan al hacer clic sobre algún elemento del menú de navegación. Como se requiere un navegador que soporte estas características de CSS3, creé una función navegadorDetect() que se ejecutara al cargar la página. Ésta reconoce el navegador que el visitante utiliza y si detecta si es o no

completamente compatible, en caso que no lo sea, generará una pequeña pestaña que invita al usuario a descargar Google Chrome para disfrutar plenamente de todas las características del sitio (ver figura 7.28).

7.28 Captura de pantalla cuando el sitio web detecta un navegador que no es completamente compatible con el sitio.

Cuando se hace clic en la liga “Quiénes somos”, se ejecuta una función quienes() que genera un elemento DIV con la clase “CajaAmarilla”. Ahí se dan detalles acerca de la empresa, quiénes la forman y a qué se dedican. Esa clase solamente le da al elemento DIV las propiedades para que tome el diseño de una caja amarilla como habíamos visto en la imagen 7.20. Después, localiza el elemento con ID “seccion” y con la instrucción innerHTML cambia su contenido a “Quiénes somos”, de igual manera, cambia el elemento TITLE a “Resta Tools: Quienes Somos” para el correcto despliegue en la ventanas del navegador del visitante y en la sección “Quienes Somos”, para avisar al visitante en que parte está localizado. Después se ejecuta la función controlMensajes() con el valor de verdadero para mantener las noticias avanzando (ver en la página 113) y finalmente se ejecuta la función activeNav() con el parámetro “Quienes Somos” para que el elemento en el menú de navegación se deshabilite, para ver más de la función saltar a la página (ver en la página 108).

En la sección “¿Porque Nosotros?” se ejecuta la función que genera un elemento DIV con caja amarilla; dentro de ella creé un elemento de lista en el que cada objeto de lista contiene un elemento de imagen que muestra una paloma con la intención de reemplazarlas por las marcas que genera por defecto. En estas imágenes, con ayuda del archivo de estilo, removí márgenes y les asigné la altura necesaria para centrarlas lo más posible con el texto. Tal función altera también el elemento con ID “seccion” y el elemento TITLE en ¿Por qué Nosotros? y “Resta Tools: ¿Por qué Nosotros?” respectivamente.

En la siguiente sección, la de “Servicios”, a diferencia de los primeros la función genera un elemento DIV con fondo blanco del tamaño del elemento con ID “cuerpo” que en

su interior tiene dos elementos DIV. El primero tiene un ID con el nombre “serv” y la propiedad de flotante con valor hacia la izquierda; éste, en forma de lista, contiene texto acerca de los planes de mantenimiento. Con auxilio del ID “serv” a éste le asigné dos valores de texto, el primero de los cuales se ejecuta de manera normal con un tamaño de letra de 16 px y el segundo utilizando la condición de “media screen” en CSS; así, si la altura es mayor a 650px, el texto en el seno de este elemento tomará el valor de 2.5vh, donde vh es la unidad de 1% relacionada con la altura del puerto de vista dentro del navegador. Esto último lo asigné para los casos en los que la ventana del navegador sea pequeña, como en los dispositivos móviles, donde prevengo que el texto se desborde del elemento con ID “cuerpo”. El segundo es un elemento DIV en cuyo interior hay un pase de diapositivas que creé con un elemento ID bajo el nombre “carrousel”. Con ayuda de las propiedades y las animaciones en CSS3 generé las diapositivas con imágenes de instrumentos de cocina, de manera muy similar al del fondo del sitio web que tiene por keyframe el nombre de “bkgd” (ver figura 7.29).

```
keyframes carrousel {
 0% {
  background:url(img/carrousel/campana.gif);
  background-size: contain;
  background-repeat: no-repeat;
 }
 10% {
  background:url(img/carrousel/freidora.jpg);
  background-size: contain;
  background-repeat: no-repeat;
 }
 20% {
  background:url(img/carrousel/insectos.jpg);
  background-size: contain;
  background-repeat: no-repeat;
 }
 …
 100% {
  background:url(img/carrousel/campana.gif) #A9D6E4;
  background-size: contain;
  background-repeat: no-repeat;
 }
}

#serv > ul li{
 font-size:16px;
 …
}

#serv small{font-family:futura;}

#carrousel{
 width:100%;
 height:100%;

 background-image:url(img/carrousel/campana.gif);
 background-size: contain;
 background-repeat: no-repeat;

 animation-name: carrousel;
 animation-delay:5s;
 animation-duration: 30s;
 animation-timing-function: ease-in-out;
 animation-fill-mode: both;
 animation-iteration-count: infinite;
 …
}

@media screen and (min-height: 650px){
 #serv *{font-size:2.5vh !important;}
}
```

7.29 Código CSS para la animación de carrusel en la sección de “servicios”

La sección llamada “Únete al equipo” está destinada a los visitantes cuya intención sea trabajar para la empresa; aquí pueden enviar su currículum y la compañía, a su vez, puede contactar posteriormente con los candidatos. En esta sección hay un formulario que hace referencia a un archivo PHP, encargado de cargar el archivo para que sea revisado por algún miembro de la compañía. A fin de que esto fuera posible, primero asigné al formulario

el envío por método POST con la forma de encriptación como multipart/form-data; este último atributo es necesario para indicar el tipo de contenido al enviar el formulario que, en este caso, es un archivo y para que pueda ser reconocido como tal. También asigné el atributo “onSubmit“ con la llamada de una función externa al motor, denominada checksize(). La intención aquí es calcular el tamaño del archivo; cuando éste sobrepasa el límite manda un mensaje de alerta y anula la operación de envío. Aunque dentro del archivo PHP ya se incluye tal comprobación, la intención de añadir esta función es para hacer más eficiente el proceso de envío y respuesta y evitar contratiempos. Así, por ejemplo, si la velocidad de internet de la que dispone el visitante es excesivamente lenta, el aviso aparecerá de inmediato; de no haberse incluido la función, el visitante tendría que terminar el proceso de envío y cargar la página siguiente, sólo para enterarse de que no se pudo remitir el archivo y esto genera molestias.

Ya generado el elemento FORM, creé en su interior un elemento INPUT de tipo archivo con un ID, NAME como “file”, que es donde el formulario captura los datos del archivo que se enviará. Al elemento le asigné la propiedad de tipo de caja como “ninguno”, puesto que, con ayuda de un elemento LABEL que hará las veces de botón y código JS, podremos realizar cambios sobre el elemento que reflejen el archivo adjuntado. Dichos cambios contribuirán a la estética y además, limitarán los tipos de archivos que se pueden enviar. Para el efecto utilicé el atributo “Accept” y agregué los tipos de archivos para PDF, DOC, DOCX; aunque hay que señalar que este método no es muy recomendable para asegurar que el archivo que se envíe sea, en efecto, un archivo de ese tipo, ya que el visitante puede cambiar la extensión del archivo y luego remitirlo. Podemos cerciorarnos de que el archivo sea del tipo requerido más adelante, en el archivo PHP es posible comprobar su contenido y asignar acciones para cualquiera de las dos condiciones, ya sea que se cumpla o no.

Al elemento LABEL le asigné un ID “filelabel” y el atributo FOR con el valor “file” para ligar la acción –al hacer clic sobre éste– con el elemento INPUT que contiene el archivo; ahora bien, para generar los cambios del elemento con el archivo adjunto, utilicé el atributo “onClick” que ejecuta una función uploadfile(). El papel de ésta es obtener el nombre del archivo agregado por el visitante y reemplazarlo por el contenido del elemento LABEL, de tal manera que cuando el visitante adjunta el archivo, el botón cambie el texto predeterminado “Sube tu CV” al nombre del archivo que esta por enviar. Por último, hay un elemento INPUT de tipo submit que realizará la acción de enviarlo (ver figura 7.30 y 7.31).

```
<div style="background:url(img/personal.jpg) #FFF; background-size:contain; … ”>
 <div class="CajaAmarilla" …>
  <p>En RESTA TOOLS estamos buscando nuevos talentos para formar parte de nuestro equipo técnico, …</p>
  <p style="margin-bottom:0px;">Nosotros nos pondremos en contacto contigo.</p>
  <small style="font-size:10px;">Archivos PDF, DOC, DOCX y menores a 5MB</small>
  <form action="upload.php" method="post" enctype="multipart/form-data" onsubmit="return checksize()" …>
   <input type="file" name="file" id="file" class="inputfile" style="display:none;"
accept="application/pdf,application/msword,application/vnd.openxmlformats-
officedocument.wordprocessingml.document">
   <label for="file" style="background: #000; color: #FFF; padding: 10px; …” id="filelabel" onclick="uploadfile()">Sube tu
CV</label><br>
   <input type="submit" name="submit">
  </form>
 </div>
</div>
```

7.30 Código HTML para la sección de “Únete al Equipo”

```
function uploadfile(){
 var input = document.getElementById("file");
 var label = input.nextElementSibling,
 labelVal = label.innerHTML;

 input.addEventListener( 'change', function( e )
 {
  var fileName = '';
  fileName = e.target.value.split( '\\' ).pop();
  if(fileName=='')
   fileName="Sube tu CV";
  document.getElementById("filelabel").innerHTML = fileName;
 });
}
```

7.31 Código JS para la carga de archivos al servidor

Una vez que el visitante manda el archivo, el servidor realiza una serie de comprobaciones antes de enviarlo. Lo primero que hice fue generar el archivo PHP para que, por medio de una cookie, enviara los resultados de regreso a la página del sitio web; la primera parte consiste reconocer si se produjo el envío por método POST como medida de seguridad para proveer posibles fallos o ataques. Una vez que se reconoce lo siguiente es localizar el archivo adjunto y su condición, para lograrlo comprobé el contenido de la variable $_FILES[‘file’][’error’] donde “file” es el nombre del elemento que contiene el archivo en el formulario anterior y error es un valor que genera PHP de manera automática y nos informa si existe algún error, asignando valores diferentes sobre este índice. El segundo paso fue comprobar que el contenido del archivo enviado cumpla con los tipos de archivos que se asignaron. Aquí se utiliza la instrucción info (FILEINFO_MIME_TYPE) sobre el archivo enviado, de ese modo se determina qué tipo de archivo es, por mucho que la extensión del archivo llegara a alterarse de manera manual. Lo siguiente fue comprobar el

tamaño del archivo utilizando –dentro de la variable “file”– el índice “size” de la siguiente manera: $_FILES[‘file’][‘size’], y lo que faltaría es solamente asignarle una condición. Finalmente si se cumplen todas las condiciones, se empieza el proceso de captura y reenvío del archivo adjunto. Una vez realizado el proceso y generada la cookie, el servidor enviará al visitante a la página principal y mediante una función que creé con el nombre de detectaCookie(), ésta buscara la cookie que se generó al enviar el archivo adjunto, agregando el texto indicado con respecto a la condición del archivo adjunto y luego eliminando la cookie (ver figura 7.32).

```
function detectCookie(){
 …
 cookieValue = getCookie("restatoolsCV");
 if(cookieValue!==""){
  switch(cookieValue){
   case "enviado":
    texto='<h1 style="font-size:50px;">Su CV ha sido enviado</h1>';
    break;
   case "maxtamano":
    texto='<h1 style="font-size:50px;">Su CV no se ha podido enviar</h1><p>Solo se puede enviar archivos …</p>';
    break;
   case "noformato":
    texto='<h1 style="font-size:50px;">Su CV no se ha podido enviar</h1><p>Solo se puede enviar archivos …</p>';
    break;
   case "noarchivo":
    texto='<h1 style="font-size:50px;">Su CV no se ha podido enviar</h1><p>No se cargó ningun archivo</p>';
    break;
   default:
    texto='<h1 style="font-size:50px;">Su CV no se ha podido enviar</h1><p>No se pudo enviar, intentelo de …</p>';
    break;
  }
  enviado(texto);
  document.cookie = "restatoolsCV=; expires=Thu, 01 Jan 1970 00:00:00 UTC";
 }
}

function enviado(html){
 document.getElementById("cuerpo").innerHTML='\
  <div class="CajaAmarilla" style="width:90%; text-align:center;">\
   '+html+'\
  </div>';

 document.getElementById("seccion").innerHTML="Mantenimiento";
 document.getElementsByTagName("title")[0].innerHTML="Resta Tools: Enviado";
 controlMensajes(true);
 activeNav("");
}
```

7.32 Código JS para la generación de mensajes en respuesta al formulario “Únete al Equipo” con el uso de cookies

Finalmente la última sección, “contacto”, es una que genera un elemento DIV que contiene un formulario para contactar con la empresa. Éste se conforma con varios elementos diferentes y los datos que pide son: nombre, teléfono, correo electrónico y un texto de comentarios. El formulario, al igual que el antecedente, contiene el atributo

“onSubmit” que ejecuta una función llamada testcontact() al ser enviado; esta función revisa que los campos sean llenados propiamente antes del envío. El formulario manda los datos a un archivo PHP que comprobará los datos ingresados y generará una acción y una respuesta de acuerdo al resultado que se dé a través de una cookie que se mostrará en la pantalla principal –como en el formulario previo–, solamente que el nombre de la cookie cambiará (ver figura 7.33).

```
function detectCookie(){
 cookieValue = getCookie("restatoolsForm");
 if(cookieValue!==""){
  switch(cookieValue){
   case "enviado":
    texto='<h1 style="font-size:50px;">Su mensaje ha sido enviado</h1><p>Nos pondremos en contacto con …</p>';
    break;
   case "noverif":
    texto='<h1 style="font-size:50px;">Su mensaje no pudo ser enviado</h1><p>Resvise que haya ingresado, …</p>';
    break;
   case "comentmin":
    texto='<h1 style="font-size:50px;">Su mensaje no pudo ser enviado</h1><p>El comentario debe ser …</p>';
    break;
   case "telform":
    texto='<h1 style="font-size:50px;">Su mensaje no pudo ser enviado</h1><p>No cumple con un número …</p>';
    break;
   case "noenviado":
    texto='<h1 style="font-size:50px;">Su mensaje no pudo ser enviado</h1><p>Inténtelo de nuevo</p>';
    break;
   default:
    texto='<h1 style="font-size:50px;">Su mensaje no pudo ser enviado</h1><p>Inténtelo de nuevo</p>';
    break;
  }
  enviado(texto);
  document.cookie = "restatoolsForm=; expires=Thu, 01 Jan 1970 00:00:00 UTC";
 }
 …
 function enviado(html){
  …
 }
}
```

7.33 Código JS para la generación de mensajes en respuesta al formulario “Contacto” con el uso de cookies

Además, esta sección tiene la peculiaridad de ejecutar la función controlMensajes() con un valor negativo que, como había explicado antes, deshabilitará las noticias y habilitará los iconos para contacto de la empresa.

## Resultados

Planeé, elaboré y desarrollé un sitio web, accesible a través de internet, para la publicidad, información, venta de servicios, reclutamiento y contacto con la empresa Restatools. Diseñé y desarrollé el sitio web con cambio de contenido dinámico, con una estructura organizada de forma jerárquica para la distribución de información, además de dotarla de un diseño

intuitivo, atractivo y sencillo de navegar, a fin de que los visitantes pudieran encontrar fácilmente la información de si interés, que se presenta de una manera resumida y amena. Tiene una amplia compatibilidad con diferentes dispositivos para desplegar la información de manera correcta. El código PHP lo diseñé a prueba de inyecciones para una mayor seguridad, así como para evitar ataques o spams. El avance del proyecto fue adecuado, fue un trabajo que duro aproximadamente 2 meses empezando en octubre del 2015, y hubo buena coordinación con el empresario. El cliente quedó satisfecho con mi trabajo.

## Conclusiones

El proyecto de este sitio web constituyó un verdadero desafío de desarrollo dadas las restricciones que impuso el cliente. La gama de recursos y de inventiva del desarrollador desempeñan aquí un papel fundamental, pues la construcción de una página web funcional que cumpla las especificaciones que plantea el contratante depende de la habilidad con que el ingeniero sea capaz de desempeñarse en la consecución de los fines planteados.

El seguimiento del proyecto, las demandas del cliente y la aplicación sistemática al trabajo de mi parte fueron cruciales. A pesar de mis conceptos personales sobre lo que constituye un buen sitio web, muchas veces fue necesario mostrar una gran flexibilidad y adaptabilidad para priorizar los gustos de mi cliente y hacer que resultaran funcionales en el sito.

# Conclusiones generales

La coparticipación de un Ingeniero en Computación en el ámbito de la publicidad y la difusión de servicios y productos de empresas, organizaciones y entidades a través de páginas web y redes, en principio no se reduce, ni debe reducirse, a la parte estrictamente técnica de la especialidad. Ciertamente, el desarrollo y aplicación de conocimientos profesionales son indispensables para dar satisfacción a las crecientes necesidades de compañías y organizaciones, pero mi experiencia personal y mi interacción con ellas –que es justamente de lo que he dado cuenta en este informe— me han revelado que hay otros factores y habilidades igualmente importantes que es indispensable poner en juego junto con los saberes técnicos.

La primera de ellas es la capacidad de comunicación, es decir, la facultad de interpretar y traducir a términos técnicos viables y funcionales, las necesidades particulares de cada empresario o responsable de institución. Por competente que sea un ingeniero en el diseño, implementación y ejecución de sistemas o páginas web; por excelso que sea su trabajo en términos técnicos, no obtendrá buenos resultados si carece de las cualidades sociales necesarias para entender los requerimientos de su clientela y darles respuesta satisfactoria desde el punto de vista profesional.

La segunda, la habilidad de canalizar el conjunto de problemas distintos que pueden presentarle sus clientes en una carpeta o portafolio de sugerencias y opciones, de entre las cuales estos pudieran elegir la o las que mejor cumplan sus expectativas y satisfagan sus pretensiones. Un ingeniero que se desempeña en esta área debe estar bien capacitado y disponer de un abanico de recursos o catálogo de posibilidades que permitan la selección y que den a sus clientes la certeza de que han solucionado sus problemas de la mejor manera, puesto que están depositando su confianza en un verdadero profesional del ramo.

La tercera y última detectada, es la necesidad permanente, para el ingeniero, de estar actualizado en materia de avances y adelantos tecnológicos en su ramo, a fin de estar en posibilidades de ofrecer a su clientela las soluciones y recursos más recientes y funcionales.

El cumplimento de esta serie de premisas garantizará que el profesional de la computación desempeñe una labor de alta calidad al servicio del empresariado o las organizaciones que necesitan difusión e interacción social a través de los medios electrónicos y del ciberespacio.

Finalmente, cabe destacar que el alto nivel formativo que recibí en la carrera de Ingeniería en computación no sólo me ha habilitado para insertarme de manera exitosa en el campo laboral del país, sino, sobre todo (y lo que es más de agradecer) me ha moldeado en la práctica autodidacta de investigación y en el permanente desarrollo de capacidades para su aplicación en el ámbito productivo. El germen educativo inicial ahora se ha convertido, para mí, en una forma de vida y en una retroalimentación constante en mi ejercicio profesional.

# Fuentes de información

Browser Display Statistics" [s.f.]. Recuperado el 09 abril 2016, de http://www.w3schools.com/browsers/browsers_display.asp

Browser Statistics" [s.f.]. Recuperado el 08 septiembre 2016, de http://www.w3schools.com/browsers/browsers_stats.asp

CSS | MDN" [s.f.]. Recuperado el 31 octubre 2016, de https://developer.mozilla.org/es/docs/Web/CSS

Duckett, Jon, *HTML & CSS Design and build websites*, Indianapolis, John Wiley & Sons, 2014.

Facebook for developers" [s.f.]. Recuperado el 02 septiembre 2016, de https://developers.facebook.com/

"4th International Conference on Industrial Engineering and Industrial Management" [s.f.]. Recuperado el 28 octubre 2016, de http://adingor.es/congresos/web/uploads/cio/cio2010/INFORMATION_SYSTEMS_AND_ICT_INFORMATION_COMMUNICATIONS_AND_TECHNOLOGY/680-689.pdf

HTML | MDN" [s.f.]. Recuperado el 31 octubre 2016, de https://developer.mozilla.org/es/docs/Web/HTML

HTML 5 - HTML | MDN" [s.f.]. Recuperado el 31 octubre 2016, de https://developer.mozilla.org/es/docs/HTML/HTML5
IBM Knowledge Center" [s.f.]. Recuperado el 31 octubre 2016, de http://www.ibm.com/support/knowledgecenter/es/SSEPGG_9.7.0/com.ibm.db2.luw.sql.ref.doc/doc/c0004100.html

"iframe – HTML | MDN" [s.f.]. Recuperado el 02 septiembre 2016, de https://developers.mozilla.org/es/docs/Web/HTML/Elemento/iframe

Jason, Beaird y James, George, *Principles of beautiful web design*, 3a ed., Melbourne, SitePoint, 2014.

JavaScript | MDN" [s.f.]. Recuperado el 31 octubre 2016, de https://developer.mozilla.org/es/docs/Web/JavaScript

"La Entrevista Conductual" [s.f.]. Recuperado el 28 octubre 2016, de http://online-psicologia.blogspot.mx/2007/11/la-entrevista-conductual.html

"Librosweb" [s.f.]. Recuperado el 29 agosto 2016, de http://librosweb.es

Pressman, Roger S., *Ingeniería del software, un enfoque práctico*, 7ª ed., New York, Mc Graw Hill, 2010.

Proyect Management Institute, *Fundamentos Para la Dirección de Proyectos (Guía del PMBOK)* 5ª ed., Pennsylvania, PMI Publications, 2013.

¿Qué es una API?" [s.f.]. Recuperado el 31 octubre 2016, de https://hipertextual.com/archivo/2014/05/que-es-api/

Twitter Developers" [s.f.]. Recuperado el 02 septiembre 2016, de https://dev.twitter.com/web/embedded-timelines

Twitter Publish" [s.f.]. Recuperado el 02 septiembre 2016, de https://publish.twitter.com/

# Glosario

- HTML: Lenguaje de Marcado para Hipertextos (HyperText Markup Language) es el elemento de construcción más básico de una página web y se usa para crear y representar visualmente una página web. Determina el contenido de la página web, pero no su funcionalidad.
- HTML 5: Es la última versión de HTML. El término representa dos conceptos diferentes: Se trata de una nueva versión de HTML, con nuevos elementos, atributos y comportamientos y contiene un conjunto más amplio de tecnologías que permite a los sitios Web y a las aplicaciones ser más diversas y de gran alcance. A este conjunto se le llama HTML5 y amigos, a menudo reducido a HTML5
- CSS: Hojas de Estilo en Cascada (Cascading Style Sheets) es el lenguaje utilizado para describir la presentación de documentos HTML o XML, esto incluye varios lenguajes basados en XML como son XHTML o SVG. CSS describe como debe ser renderizado el elemento estructurado en pantalla, en papel, hablado o en otros medios.
- JS: JavaScript es un lenguaje ligero e interpretado, orientado a objetos con funciones de primera clase, más conocido como el lenguaje de script para páginas web. Es un lenguaje script multi-paradigma, basado en prototipos, dinámico, soporta estilos de programación funcional, orientada a objetos e imperativa.
- API: Una API es un conjunto de funciones y procedimientos que cumplen una o muchas funciones con el fin de ser utilizadas por otro software. Las siglas API vienen del inglés Application Programming Interface. En español sería Interfaz de Programación de Aplicaciones.
- SQL: Es un lenguaje estandarizado que sirve para definir y manipular los datos de una base de datos relacional. De acuerdo con el modelo relacional de datos, la base de datos se crea como un conjunto de tablas, las relaciones se representan mediante valores en las tablas y los datos se recuperan especificando una tabla de resultados que puede derivarse de una o más tablas base.