

**PROYECTO DE MEMORIA DE ESTADÍA**

**PROFESIONAL:**

Metodología para el Desarrollo Web Dinámico enfocado en Sistemas de Gestión y Administración Empresarial

**REALIZADO EN:**

Arkebit S.A. de C.V.

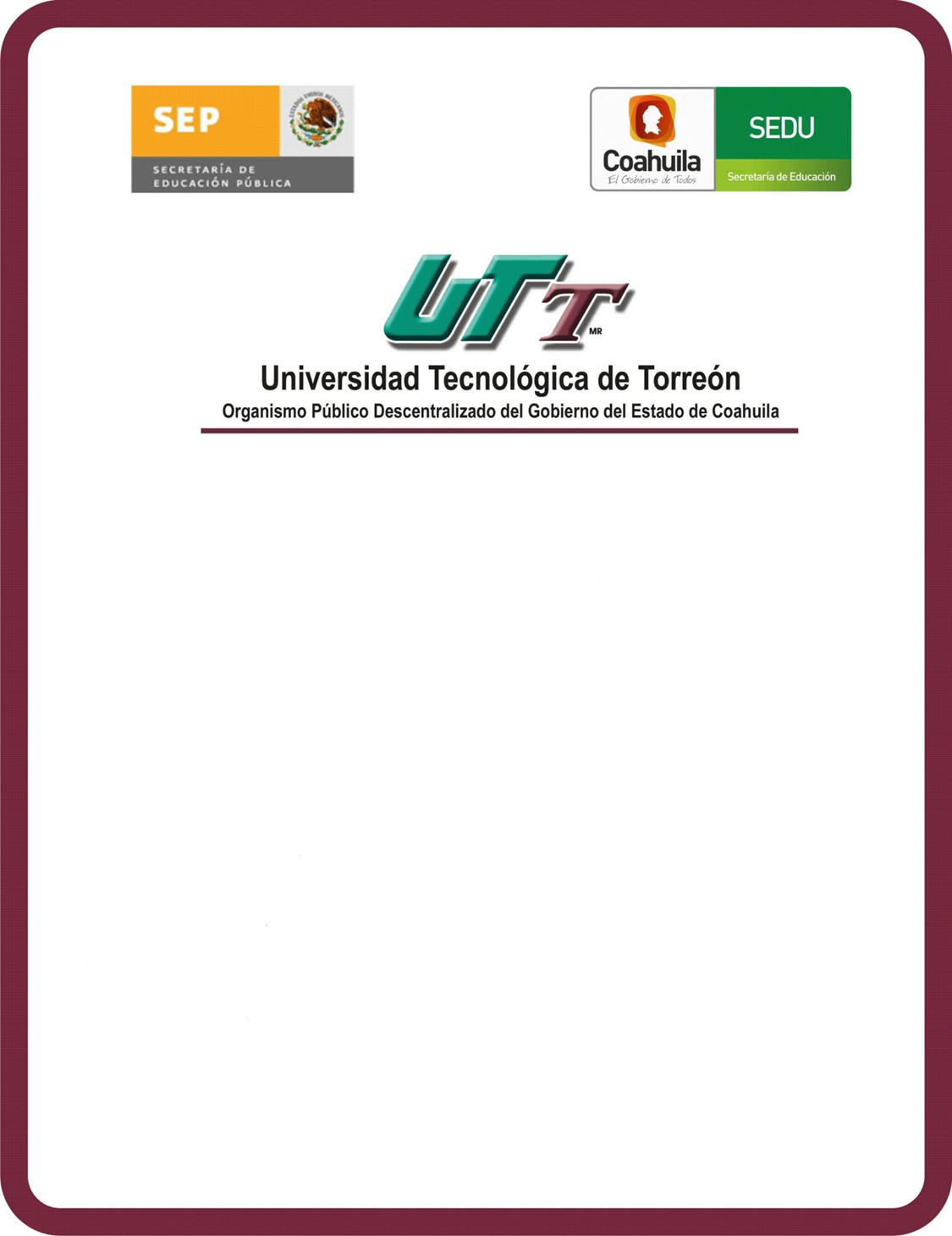
**PRESENTA:**

Luis Miguel Guzmán Ruiz

**PARA OBTENER EL TITULO DE:**

Ingeniería en Tecnologías de la Información

**TORREÓN, COAHUILA ABRIL 2016**



**PROYECTO DE MEMORIA DE ESTADÍA**

**PROFESIONAL:**

Metodología para el Desarrollo Web Dinámico enfocado en Sistemas de Gestión y Administración Empresarial

**REALIZADO EN:**

Arkebit S.A. de C.V.

**ASESOR DE LA ESTADIA:**

MSC. Daniel Rosales Díaz Mirón

**PRESENTA:**

Luis Miguel Guzmán Ruiz

**TORREÓN, COAHUILA ABRIL 2016**

**Agradecimientos**

**A MI MADRE:**

A ella que siempre me alentó para seguir adelante, la que se preocupó cada instante porque yo estuviera bien, la que me daba consejos cuando lo necesitaba, la que me regañaba para que fuera por el camino correcto en la vida

**A MIS ABUELOS:**

Que han sido parte importante en toda mi formación como persona y como profesionista por darme el cariño, comprensión y apoyo incondicional en toda mi travesía por esta carrera tan importante para mis muchas gracias a ellos que han sido mis segundos padres.

**A MIS HERMANOS:**

Que gracias a ellos me doy cuenta que encuentran en mí el ejemplo de terminar una carrera profesional y lo que cuesta terminar los estudios por ayudarme cuando necesitaba alguna ayuda ahí estuvieron.

**A MIS MAESTROS:**

Que con su experiencia, sabiduría y calidad humana supieron transmitir conocimientos y competencia para lograr mi formación académica y humana para la culminación de mi carrera profesional.

**AL PERSONAL DE ARKEBIT S.A. DE C.V.**

Que hicieron posible la realización de mis prácticas profesionales y brindarme un poco de sus habilidades, conocimientos y experiencias trasmitiendo un mundo

nuevo en el sector productivo.

*Tabla de contenido*

Portada

*Contraportada*

*Hoja de liberación*

*Agradecimientos*

*Introducción*

Tabla de contenido

[Introducción 1](#_Toc447845356)

[CAPITULO I 2](#_Toc447845357)

[1.1 Justificación 3](#_Toc447845358)

[Objetivo General 3](#_Toc447845359)

[Objetivos Específicos 3](#_Toc447845360)

[1.2 Datos generales de la empresa 4](#_Toc447845361)

[Nombre de la Empresa 4](#_Toc447845362)

[Sector y Giro de la Empresa 4](#_Toc447845363)

[Servicios 4](#_Toc447845364)

[Dirección 4](#_Toc447845365)

[Proyecto 5](#_Toc447845366)

[Asesor de la Empresa 5](#_Toc447845367)

[Cargo del Asesor 5](#_Toc447845368)

[1.3 Antecedentes de la empresa 5](#_Toc447845369)

[Historia 5](#_Toc447845370)

[1.4 Descripción de la empresa 6](#_Toc447845371)

[Quiénes son 6](#_Toc447845372)

[Visión 6](#_Toc447845373)

[Misión 6](#_Toc447845374)

[Valores 6](#_Toc447845375)

[1.5 Descripción del área donde se realizó la estancia 7](#_Toc447845376)

[Objetivo 7](#_Toc447845377)

[Organigrama 7](#_Toc447845378)

[CAPITULO II: 8](#_Toc447845379)

[Desarrollo de la Estadía. 8](#_Toc447845380)

[2.1 Análisis del problema. 9](#_Toc447845381)

[*2.2* Objetivo del proyecto. 9](#_Toc447845382)

[*2.3* Marco Conceptual 9](#_Toc447845383)

[Programación Orientada a Objetos 9](#_Toc447845384)

[Clase (Programación) 10](#_Toc447845385)

[Objeto (Programación) 10](#_Toc447845386)

[HMTL - HTML5 11](#_Toc447845387)

[CSS - CSS3 12](#_Toc447845388)

[JavaScript 13](#_Toc447845389)

[Twitter Bootstrap 15](#_Toc447845390)

[JQuery 16](#_Toc447845391)

[Animate CSS 18](#_Toc447845392)

[Bootstrap Modal 19](#_Toc447845393)

[Select Bootstrap 20](#_Toc447845394)

[PNotify 20](#_Toc447845395)

[JQuery Validate 21](#_Toc447845396)

[PHP 22](#_Toc447845397)

[MySQL 24](#_Toc447845398)

[PDO 25](#_Toc447845399)

[ERP 26](#_Toc447845400)

[2.4 Desarrollo del proyecto Metodología para el Desarrollo Web Dinámico enfocado en Sistemas de Gestión y Administración Empresarial 27](#_Toc447845401)

[Login 28](#_Toc447845402)

[Menú y Panel Dinámicos 36](#_Toc447845403)

[Altas, Bajas y Cambios con PDO, JQuery y PHP 49](#_Toc447845404)

[Pagos Online (Conekta) 49](#_Toc447845405)

[Suscripciones (Conekta) 58](#_Toc447845406)

[Validaciones 68](#_Toc447845407)

[Notificaciones 68](#_Toc447845408)

[Consultas en PHP usando Transact-SQL para elaboración de Reportes Ejecutivos 68](#_Toc447845409)

[Conclusiones 81](#_Toc447845410)

[Bibliografía 81](#_Toc447845411)

[Anexos 81](#_Toc447845412)

**Índice de Figuras**

Figura 1. Logo HTML5 12

Figura 2. Logo CSS3 13

Figura 3. Logo JavaScript 15

Figura 4. Logo Twitter Bootstrap 16

Figura 5. Rendimiento de Jquery en los navegadores 17

Figura 6. Logo Jquery 17

# Introducción

En función de las nuevas necesidades en nuestro mundo globalizado, y de los requerimentos de las empresas que nacen día con día; las tecnologías, las tecnologías emergentes y todas sus potencialidades han tenido la necesidad de migrar y escalar sus posibilidades, funciones y funcionalidades para que se puedan desarrollar nuevos sistemas que cumplan con las necesidades crecientes actuales.

La finalidad de este proyecto es elaborar una metodología estándar para realizar el desarrollo de aplicaciones web dinámicas con enfoque en el sector productivo empresarial, que consecuentemente ayudará al personal de la empresa a tener como patrón representativo y modelo a seguir dicho dogma, lo cual podrá beneficiar a la institución gracias a que se mantendría un estándar de calidad y el tiempo de desarrollo de sistemas se recortaría en gran medida.

# CAPITULO I

**CAPITULO I**

## 1.1 Justificación

La razon que llevó a tomar la decisión de escoger este tipo de proyecto, fue en gran medida, a la necesidad de desarrollar las mejores prácticas en el desarrollo de software y en todos sus procesos, así como la implementación de un estándard interno en la empresa y un manual de procedimientos para todo el personal nuevo en la empresa, que mejorará la eficiencia en tiempos, costos y diseño.

### Objetivo General

La creación de un estándar para el óptimo desarrollo de sistemas de gestión empresarial en ambientes web, manual de procedimientos para el personal del área de sistemas; así como una herramienta que ayudará a eficientar tiempos, costos y diseño.

### Objetivos Específicos

* Analisis de requerimentos por parte del sistema
* Diseño adecuado de la base de datos de manera en que se pueda migrar el sistema en base a las necesidades emergentes del mismo.
* Implementación de las mejores prácticas en programación Front-End
* Implementación de las mejores prácticas en programación Back-End
* Utilización de las tecnologías más populares en el desarrollo
* Elaboración de un documento de apoyo para el personal interno en el desarrollo de sistemas.
* Eficientar los costos de producción como resultado de las tareas idoneamente distribuidas.
* Eficientar el tiempo de producción como resultado de las tareas idoneamente distribuidas.

## 1.2 Datos generales de la empresa

### Nombre de la Empresa

Arkebit S.A de C.V.

### Sector y Giro de la Empresa

Tecnologías de la Información con enfoque en Desarrollo de Software y Seguridad Informática.

### Servicios

* Asesoría en Servidores GNU/Linux.
* Reconstrucción y reparación de Base de Datos Sybase.
* Capacitación en desarrollo de sistemas de seguridad informática en Python.
* Asesoría en Bases de Datos MS-SQL.
* Instalación y Configuración de servidores en Windows y Linux.
* Instalación y Configuración de IDS, firewalls y sistemas de monitoreo de red.
* Desarrollo de agentes inteligentes de seguridad informática.
* Asesorías en seguridad informática y malware criptográfico.
* Capacitación en diferentes ámbitos de tecnologías de la información.
* Desarrollo de Sistemas web para diferentes rubros.
* Desarrollo de Sistemas en ambientes de escritorio para diferentes rubros.
* Auditoría de vulnerabilidades y de pentesting.

Dirección

**Torreón**  
Saltillo 400 No. 649-Bis  
Ampliación La Rosita  
CP. 27000

Telefono. 87-11-00-24-14

Proyecto

Metodología para el Desarrollo Web Dinámico enfocado en Sistemas de Gestión y Administración Empresarial

### Asesor de la Empresa

Ing. Ernesto Emmanuel Gutierrez

Cargo del Asesor**:**

Administrador General Arkebit

## 1.3 Antecedentes de la empresa

Historia

**Arkebit 2013**

Arkebit nace a partir de una idea de 4 socios de ser una empresa líder dentro del campo de las TI y crear concientización sobre la seguridad informática.

**Arkebit 2014**

Después de 1 año, Arkebit empieza a ser reconocida entre empresas de TI laguneras y universidades gracias a la satisfacción de sus clientes y a múltiples conferencias con alumnos de diferentes planteles universitarios generando conciencia sobre la seguridad informática. La empresa comienza a capacitar personal tanto de universidad como servidores públicos.

**Arkebit 2015**

2 años después de ser fundada, Arkebit es reconocida por grandes empresas tanto públicas como privadas. Reconocidas gracias a sus labores de seguridad informática y a capacitación a clientes entre los cuales están elementos de las fuerzas armadas y personal universitario.

## 1.4 Descripción de la empresa

### Quiénes son

Es una empresa dedicada y apasionada por las tecnologías de la información cuya misión es brindar el mejor servicio sobre tecnologías de vanguardia que satisfagan las necesidades empresariales fomentando la seguridad informática.

### Visión

Llegar a ser un referente como una de las casas desarrolladoras de software y de seguridad informática más importantes del país, sirviendo como ejemplo de la pasión y dedicación a las tecnologías de la información.

### Misión

Ofrecer soluciones tecnológicas innovadoras que generan un valor agregado a nuestros clientes, mediante un equipo de profesionales en tecnologías de la información altamente competitivo.

### Valores

* Calidad
* Formalidad
* Satisfacción del usuario
* Integridad
* Liderazgo
* Creatividad
* Innovación Tecnológica
* Respeto por los individuos
* Derecho de propiedad

## 

## 1.5 Descripción del área donde se realizó la estancia

### Objetivo

Realizar proyectos de corrección en el ámbito de tecnologías de la información que permitan el correcto funcionamiento de los sistemas informáticos para asegurar así un trabajo de calidad para los clientes en el momento que se requiera.

### Organigrama

Administrador y Analista

Ing. Ernesto Emmanuel Gutierrez

Ethical Hacking

Lic. Antonio Gurza Acuña

Arquitecto de Software

Ing. Hermes Uriel Lugo Hernandez

Programador

TSU. Angel Camacho

Programador

TSU. Francisco Martel

Técnico Pentesting

TSU. Andrés Castañón

Programador

TSU. Oscar Garza

Programador

TSU.Luis Guzán

# 

# CAPITULO II:

## Desarrollo de la Estadía.

## 

## 2.1 Análisis del problema.

­­­La idea de este proyecto, surgió en base a la necesidad de la empresa por implementar una metodología que ayudaría en gran medida al desarrollo de software ERP. En el desarrollo de la misma se comprobó de la funcionalidad de esta metodología que posteriormente se explicará a detalle.

## *2.2* Objetivo del proyecto.

Realizar una completa restructuración del sistema de cableado estructurado de la institución, creando nuevas rutas para el cableado tanto te voz como el de datos para así poder garantizar una red convergente que permita a los usuarios y a los beneficiarios realizar todas las actividades que requieran de una manera sencilla y rápida.

## *2.3* Marco Conceptual

### Programación Orientada a Objetos

La programación orientada a Objetos básicamente define una serie de conceptos y técnicas de programación para representar acciones o cosas de la vida real basada en objetos, a diferencia de otras formas de programación como por ejemplo la estructurada, con la POO trabajamos de manera distinta vinculando diferentes conceptos tales como clases, objetos, métodos, propiedades, estados, herencia, encapsulación entre otros, generando cada vez interrelaciones en nuestro desarrollo en pro del funcionamiento del sistema principal, definiendo el programa como un conjunto de estos objetos relacionados entre si.

### Clase (Programación)

Las clases son uno de los principales componentes de un lenguaje de programación, pues en ellas ocurren todos los procesos lógicos requeridos para un sistema, en si podemos definirlas como estructuras que representan objetos del mundo real, tomando como objetos a personas, lugares o cosas, en general las clases poseen propiedades, comportamientos y relaciones con otras clases del sistema.

Una clase se compone por tres partes fundamentales:

* **Nombre**: Contiene el Nombre de la Clase.
* **Atributos**: Representan las propiedades que caracterizan la clase.
* **Métodos**: Representan el comportamiento u operaciones, la forma como interactúa la clase con su entorno.

### Objeto (Programación)

Los objetos representan una entidad concreta o abstracta del mundo real, en programación básicamente se le conoce como la instancia de una clase en si es lo que da el sentido a estas.

Al igual que las clases se componen de tres partes fundamentales:

* **Estado**: Representa los atributos o características con valores concretos del objeto.
* **Comportamiento**: Se define por los métodos u operaciones que se pueden realizar con él.
* **Identidad**: Es la propiedad única que representa al objeto y lo diferencia del resto.

### HMTL - HTML5

Es el lenguaje de marcado predominante para la elaboración de páginas web. .Es usado para describir la estructura y el contenido en forma de texto, así como para complementar el texto con objetos tales como imágenes. El HTML se escribe en forma de «etiquetas», rodeadas por corchetes angulares (<,>). HTML también puede describir, hasta un cierto punto, la apariencia de un documento, y puede incluir un script (por ejemplo JavaScript), el cual puede afectar el comportamiento de navegadores web y otros procesadores de HTML.

La primera descripción pública disponible de HTML fue un documento llamado "HTML Tags", mencionado por primera vez en internet por Berners-Lee en los finales de 1991. A partir de entonces, la especificación de HTML ha estado en continua evolución. Desde 1996, las especificaciones de HTML se han mantenido, con la participación de proveedores de software comercial, por el W3C. Por otra parte, en el año 2000, HTML también se comvirtió en un estándar internacional (ISO/IEC 15445:2000).

**HTML5** es la última versión de [HTML](https://developer.mozilla.org/en-US/docs/HTML). El término representa dos conceptos diferentes:

* Se trata de una nueva versión de HTML, con nuevos elementos, atributos y comportamientos.
* 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 .

Esta es la primera vez que HTML y XHTML se han desarrollado en paralelo. La versión definitiva de la quinta revisión del estándar se publicó en octubre de 2014. Diseñado para ser utilizable por todos los desarrolladores de Open Web, esta página referencía a numerosos recursos sobre las tecnologías de HTML5, que se clasifican en varios grupos según su función.

* **Semántica:** Permite describir con mayor precisión cual es su contenido.
* **Conectividad:** Permite comunicarse con el servidor de formas nuevas e innovadoras.
* **Sin conexión y almacenamiento:** Permite a las páginas web almacenar datos localmente en el lado del cliente y operar sin conexión de manera más eficiente.
* **Multimedia:** Nos otorga un excelente soporte para utilizar contenido multimedia como lo son audio y video nativamente.
* **Gráficos y efectos 2D/3D:** Proporciona una amplia gama de nuevas características que se ocupan de los gráficos en la web como lo son canvas 2D, WebGL, SVG, etc.
* **Rendimiento e Integración:** Proporciona una mayor optimización de la velocidad y un mejor uso del hardware.
* **Acceso al dispositivo:** Proporciona APIs para el uso de varios compomentes internos de entrada y salida de nuestro dispositivo.



**Figura 1. Logo HTML5**

### CSS - CSS3

Hojas de Estilo en Cascada (CSS) es un lenguaje de estilo de hojas usado para describir la presentación de las páginas web. CSS permite la separación del contenido del documento de la presentación del documento (disposición, colores, fuentes, etcétera).

La Especificación de Hojas de Estilo en Cascada nivel 2 Revisión 1 (CSS 2.1) fue publicada en junio de 2011. Según esta especificación:

*“”CSS 2.1 es un lenguaje de estilo de hojas que permite al autor y a los usuarios adjuntar estilo (ej., fuentes y espaciado) a documentos estructurados (ej., documentos HTML y aplicaciones XML). Separando el estilo de presentación del documento del contenido del documento, CSS 2.1 simplifica la creación de páginas web y el mantenimiento del sitio.””*

Algunas de las nuevas propiedades de CSS3 pueden ser usadas ahora mismo, ya que, están casi soportadas por todos los navegadores modernos:

* @font-face
* background-size
* border-image
* border-radius
* box-shadow
* column-count
* column-width
* text-overflow
* text-shadow



**Figura 2. Logo CSS3**

### JavaScript

JavaScript es un lenguaje de scripting multiplataforma y orientado a objetos. Es un lenguaje pequeño y liviano. Dentro de un ambiente de host, javaScript puede conectarse a los objetos de su ambiente y proporcionar control programático sobre ellos.

JavaScript contiene una librería estándar de objetos, tales como Array, Date, y Math, y un conjunto central de elementos del lenguaje, tales como operadores, estructuras de control, y sentencias. El núcleo de JavaScript puede extenderse para varios propósitos, complementándolo con objetos adicionales, por ejemplo:

* **Client-side JavaScript** extiende el núcleo del lenguaje proporcionando objetos para controlar un navegador y su modelo de objetos (o DOM, por las iniciales de Document Object Model). Por ejemplo, las extensiones del lado del cliente permiten que una aplicación coloque elementos en un formulario HTML y responda a eventos del usuario, tales como clicks del ratón, ingreso de datos al formulario y navegación de páginas.
* **Server-side JavaScript** extiende el núcleo del lenguaje proporcionando objetos relevantes a la ejecución de javaScript en un servidor. Por ejemplo, las extensiones del lado del servidor permiten que una aplicación se comunique con una base de datos, proporcionar continuidad de la información de una invocación de la aplicación a otra, o efectuar manipulación de archivos en un servidor.

**Características**

* Es simple, no hace falta tener conocimientos de programación para poder hacer un programa en JavaScript.
* Maneja objetos dentro de nuestra página Web y sobre ese objeto podemos definir diferentes eventos. Dichos objetos facilitan la programacion de paginas interactivas, a la vez que se evita la posibilidad de ejecutar comandos que puedan ser peligrosos para la maquina del usuario, tales como formateo de unidades, modificar archivos etc.
* Es dinámico, responde a eventos en tiempo real. Eventos como presionar un botón, pasar el puntero del mouse sobre un determinado texto o el simple hecho de cargar la página o caducar un tiempo. Con esto podemos cambiar totalmente el aspecto de nuestra página al gusto del usuario, evitándonos tener en el servidor un página para cada gusto, hacer calculos en base a variables cuyo valor es determinado por el usuario, etc.



**Figura 3. Logo JavaScript**

### Twitter Bootstrap

Bootstrap es un framework que simplifica el proceso de creación de diseños web combinando CSS y JavaScript. Ha sido desarrollado por Twitter que recientemente liberó su versión 2.0. La mayor ventaja es que podemos crear interfaces que se adapten a los distintos navegadores (responsive design) apoyándonos en un framework potente con numerosos componentes webs que nos ahorrarán mucho esfuerzo y tiempo.

Bootstrap ofrece una serie de plantillas CSS y ficheros Javascript que nos permiten integrar el framework de forma sencilla y potente en nuestros proyectos webs.

Entre sus principales características encontramos que:

* Permite crear interfaces que se adapten a los diferentes navegadores, tanto de escritorio como tablets y móviles a distintas escalas y resoluciones.
* Se integra perfectamente con las principales librerías Javascript, por ejemplo JQuery.
* Ofrece un diseño sólido usando LESS y estándares como CSS3/HTML5.
* Es un framework ligero que se integra de forma limpia en nuestro proyecto actual.
* Funciona con todos los navegadores, incluido Internet Explorer usando HTML Shim para que reconozca los tags HTML5.
* Dispone de distintos layout predefinidos con estructuras fijas a 940 píxeles de distintas columnas o diseños fluidos.



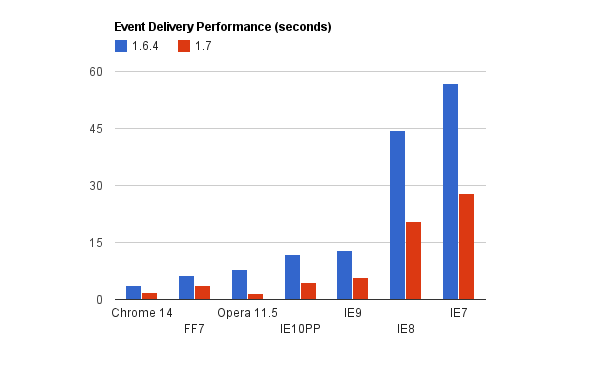
**Figura 4. Logo Twitter Bootstrap**

### JQuery

jQuery es una biblioteca de JavaScript, fue creada por John Resig, permite simplificar la manera de interactuar con los documentos HTML, manipular el árbol DOM, manejo de eventos, desarrollar animaciones y agregar interacción con la técnica AJAX en páginas web.

**Evolución de jQuery**

JQuery es una libreria en constante cambio, es recomendable usar la libreria estable más reciente, ya que en cada nueva versión se integran caracteristicas nuevas al tiempo que se mejoran las anteriores, la imagen siguiente es una compativa en el rendimiento de jQuery 1.6.4 vs jQuery 1.7 en distintos navegadores:



**Figura 5. Rendimiento de Jquery en los Navegadores**

**Cambios entre 1.7 y 1.8**

Cuando **jQuery** migro a la versión **1.8** se hicieron algunos cambios importantes, por ejemplo la versión **1.8** es más pequeña que **1.7**, las funciones css() y animate() libres de prefijos(-webkit, -moz, -ms, -o), se repararon 160 errores, mayor flexibilidad en animaciones y se mejoraron las consultas.

Actualmente Jquery está en la versión 2.0 y ofrece una gran cantidad de herramientas para los desarrolladores Front-End



**Figura 6. Logo Jquery**

### ****Animate CSS****

Animate CSS es un framework de código abierto basado en CSS3 el cual permite animar componentes dentro del DOM de algún proyecto escrito con HTML, para poder utilizarlo se debe de agregar un CDN o el archivo descargado desde la página oficial (<https://daneden.github.io/animate.css/>) a nuestros documentos, luego basta con agregar una o más clases predefinidas en el framework a uno o más elementos de nuestro proyecto para que se produzca la animación.

Entre las clases predefinidas en el framework se encuentran las siguientes:

* bounce
* flash
* pulse
* rubberBand
* shake
* headShake
* swing
* tada
* wobble
* jello
* bounceIn
* bounceInDown
* bounceInLeft
* bounceInRight
* bounceInUp
* bounceOut
* bounceOutDown
* bounceOutLeft
* bounceOutRight
* bounceOutUp
* fadeIn
* fadeInDown
* fadeInDownBig
* fadeInLeft
* fadeInLeftBig
* fadeInRight
* fadeInRightBig
* fadeInUp
* fadeInUpBig
* fadeOut
* fadeOutDown
* fadeOutDownBig
* fadeOutLeft
* fadeOutLeftBig
* fadeOutRight

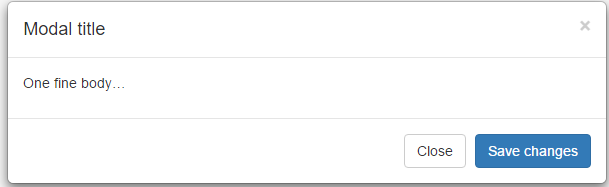


**Figura 7. Logo Animate CSS**

### Bootstrap Modal

Los modales o ventanas modales son simples pero flexibles cuadros de dialogo creados con clases de CSS3 y eventos de JavaScript que permiten desplegar contenido dentro de una página simulando una ventana flotante.

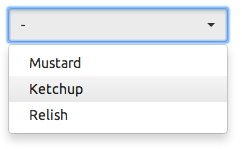
Este tipo de ventanas son muy útiles ya que evita el uso desmedido de formularios y permite obtener una gran experiencia visual a los sistemas que la utilizan, además le dan a los sistemas una gran herramienta de interactividad para el usuario.



**Figura 8. Estructura Básica de un Bootstrap Modal**

### ****Select Bootstrap****

Esta herramienta es un plugin que utiliza la librería dropdown.js de bootstrap para poder añadirle un estilo más formal y profesional a las cajas desplegables de los elementos HTML, asi como una mayor funcionalidad a los mismos, agregandole la posibilidad de hacer una búsqueda interna entre sus elementos.

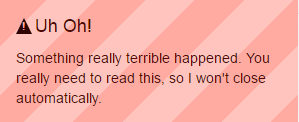


**Figura 9. Apariencia del Select-Bootstrap**

### PNotify

Es de suma importancia en el desarrollo de un sistema, poder mostrar alertas cuando cierta acción o evento se cumple en determinadas situaciones o condiciones. Por defecto los navegadores web, tienen sus propias alertas, las cuales el lenguaje de programación Javascript o en su defecto el Framework Javascript JQuery accionan. Pero, es de importancia darle mas elegancia y diseño a este tipo de alertas y notificaciones.

Existe una librería que nos puede proporcionar lo anteriormente mencionado, la cual es PNotify



**Figura 10. Apariencia de las notificaciones PNotify**

### JQuery Validate

Esta herramienta permite desarrollar una validación de campos y datos de nuestro sistema del lado del cliente, de manera en que podamos recibir los datos que nosotros necesitamos, a manera de evitar errores y falta de coherencia en la base de datos. Esta herramienta viene con un montón de métodos que nos proveen una validación rápida y eficaz en nuestra información.

**Selectores:**

* :blank:– Selecciona todos los elementos con un valor en blanco.
* :filled:– Selecciona todos los elementos con un valor lleno
* :unchecked:– Selecciona todos los elementos con seleccionado.

**Validator:**

La clase validator, regresa un solo valor del tipo booleano al realizar todas las validaciones sobre el formulario que nosotros apuntemos y que obviamente se desea validar: True si todo está correcto y False si existe algún campo que no cumple con los requisitos de la validación.

**Métodos de Validador:**

* Validator.form().Valida el formulario.
* Validator.element().Valida un solo elemento.
* [Validator.resetForm()](http://jqueryvalidation.org/Validator.resetForm)
* [Validator.showErrors() – Show the specified messages.](http://jqueryvalidation.org/Validator.showErrors)
* [Validator.numberOfInvalids() – Returns the number of invalid fields.](http://jqueryvalidation.org/Validator.numberOfInvalids)

**Lista de métodos auxiliares para la validación:**

* Required. Hace que el elemento sea requerido.
* Minlength. Establece la longitud minima del control.
* Maxlength. Establece la longitud máxima del control
* Min. Establece el valor minimo de un elemento.
* Max. Establece el valor máximo de un elemento.
* Email. Exige que el elemento tenga un correo valido.
* url.Exige que el elemento tenga una URL valida
* number. Exige que el elemento contenga un valor númerico.
* equalTo. Valida que el valor de un elemento sea igual a otro elemento.

### PHP

Es un lenguaje de programación [de uso general](https://es.wikipedia.org/wiki/Lenguaje_de_programaci%C3%B3n_de_prop%C3%B3sito_general) de [código del lado del servidor](https://es.wikipedia.org/wiki/Script_del_lado_del_servidor) originalmente diseñado para el [desarrollo web](https://es.wikipedia.org/wiki/Desarrollo_web) de [contenido dinámico](https://es.wikipedia.org/wiki/Contenido_din%C3%A1mico). Fue uno de los primeros lenguajes de programación del lado del servidor que se podían incorporar directamente en el documento [HTML](https://es.wikipedia.org/wiki/HTML) en lugar de llamar a un archivo externo que procese los datos. El código es interpretado por un servidor web con un módulo de procesador de PHP que genera la página Web resultante. PHP ha evolucionado por lo que ahora incluye también una interfaz de [línea de comandos](https://es.wikipedia.org/wiki/L%C3%ADnea_de_comandos) que puede ser usada en [aplicaciones gráficas](https://es.wikipedia.org/wiki/Interfaz_gr%C3%A1fica_de_usuario) independientes. Puede ser usado en la mayoría de los servidores web al igual que en casi todos los sistemas operativos y plataformas sin ningún costo.

PHP se considera uno de los lenguajes más flexibles, potentes y de alto rendimiento conocidos hasta el día de hoy, lo que ha atraído el interés de múltiples sitios con gran demanda de tráfico, como [Facebook](https://es.wikipedia.org/wiki/Facebook), para optar por el mismo como tecnología de servidor.

**Historia**

Fue originalmente diseñado en [Perl](https://es.wikipedia.org/wiki/Perl), con base en la escritura de un grupo de [CGI](https://es.wikipedia.org/wiki/Interfaz_de_entrada_com%C3%BAn) binarios escritos en el [lenguaje C](https://es.wikipedia.org/wiki/C_(lenguaje_de_programaci%C3%B3n)) por el [programador](https://es.wikipedia.org/wiki/Programador) danés-canadiense [Rasmus Lerdorf](https://es.wikipedia.org/wiki/Rasmus_Lerdorf) en el año 1994 para mostrar su [currículum vítae](https://es.wikipedia.org/wiki/Curr%C3%ADculum_v%C3%ADtae) y guardar ciertos datos, como la cantidad de tráfico que su página web recibía. El 8 de junio de 1995 fue publicado "Personal Home Page Tools" después de que Lerdorf lo combinara con su propio *Form Interpreter* para crear PHP/FI.

Dos programadores [israelíes](https://es.wikipedia.org/wiki/Israel) del [Technion](https://es.wikipedia.org/wiki/Technion), [Zeev Suraski](https://es.wikipedia.org/wiki/Zeev_Suraski) y [Andi Gutmans](https://es.wikipedia.org/wiki/Andi_Gutmans), reescribieron el [analizador sintáctico](https://es.wikipedia.org/wiki/Analizador_sint%C3%A1ctico) (*parser* en inglés) en el año 1997 y crearon la base del PHP3, cambiando el nombre del lenguaje por *PHP: Hypertext Preprocessor*. Inmediatamente comenzaron experimentaciones públicas de PHP3 y fue publicado oficialmente en junio de 1998. Para 1999, Suraski y Gutmans reescribieron el código de PHP, produciendo lo que hoy se conoce como [motor Zend](https://es.wikipedia.org/wiki/Motor_Zend). También fundaron Zend Technologies en [Ramat Gan](https://es.wikipedia.org/wiki/Ramat_Gan), [Israel](https://es.wikipedia.org/wiki/Israel).

En mayo de 2000 PHP 4 fue lanzado bajo el poder del [motor Zend](https://es.wikipedia.org/wiki/Motor_Zend) 1.0. El día 13 de julio de 2007 se anunció la suspensión del soporte y desarrollo de la versión 4 de PHP, a pesar de lo anunciado se ha liberado una nueva versión con mejoras de seguridad, la 4.4.8 publicada el 13 de enero de 2008 y posteriormente la versión 4.4.9 publicada el 7 de agosto de 2008. Según esta noticia se le dio soporte a fallos críticos hasta el 9 de agosto de 2008.

**Caracteristicas**

* Orientado al desarrollo de aplicaciones web dinámicas con acceso a información almacenada en una [base de datos](https://es.wikipedia.org/wiki/Base_de_datos).
* Es considerado un lenguaje fácil de aprender, ya que en su desarrollo se simplificaron distintas especificaciones, como es el caso de la definición de las variables primitivas, ejemplo que se hace evidente en el uso de [php arrays](https://es.wikipedia.org/wiki/Php_arrays).
* El código fuente escrito en PHP es invisible al [navegador web](https://es.wikipedia.org/wiki/Navegador_web) y al cliente, ya que es el servidor el que se encarga de ejecutar el código y enviar su resultado HTML al navegador.
* Capacidad de conexión con la mayoría de los motores de base de datos que se utilizan en la actualidad, destaca su conectividad con [MySQL](https://es.wikipedia.org/wiki/MySQL) y [PostgreSQL](https://es.wikipedia.org/wiki/PostgreSQL).
* Capacidad de expandir su potencial utilizando módulos (llamados *ext's* o extensiones).
* Posee una amplia documentación en su sitio web oficial, entre la cual se destaca que todas las funciones del sistema están explicadas y ejemplificadas en un único archivo de ayuda.
* Es [libre](https://es.wikipedia.org/wiki/Software_libre), por lo que se presenta como una alternativa de fácil acceso para todos.
* Permite aplicar técnicas de [programación orientada a objetos](https://es.wikipedia.org/wiki/Programaci%C3%B3n_orientada_a_objetos).
* No requiere definición de tipos de variables aunque sus variables se pueden evaluar también por el tipo que estén manejando en tiempo de ejecución.
* Tiene manejo de excepciones (desde PHP5).
* Si bien PHP no obliga a quien lo usa a seguir una determinada metodología a la hora de programar, aún haciéndolo, el programador puede aplicar en su trabajo cualquier técnica de programación o de desarrollo que le permita escribir código ordenado, estructurado y manejable. Un ejemplo de esto son los desarrollos que en PHP se han hecho del [patrón de diseño](https://es.wikipedia.org/wiki/Patr%C3%B3n_de_dise%C3%B1o) [Modelo Vista Controlador](https://es.wikipedia.org/wiki/Modelo_Vista_Controlador) (MVC), que permiten separar el tratamiento y acceso a los datos, la [lógica de control](https://es.wikipedia.org/wiki/L%C3%B3gica_de_control) y la [interfaz de usuario](https://es.wikipedia.org/wiki/Interfaz_de_usuario) en tres componentes independientes.
* Debido a su flexibilidad ha tenido una gran acogida como lenguaje base para las aplicaciones WEB de manejo de contenido, y es su uso principal.



**Figura 11. Logo de PHP**

### MySQL

Es un [sistema de gestión de bases de datos](https://es.wikipedia.org/wiki/Sistema_de_gesti%C3%B3n_de_bases_de_datos) [relacional](https://es.wikipedia.org/wiki/Modelo_relacional), [multihilo](https://es.wikipedia.org/wiki/Hilo_de_ejecuci%C3%B3n) y [multiusuario](https://es.wikipedia.org/wiki/Multiusuario) con más de seis millones de instalaciones.

Por un lado se ofrece bajo la [GNU GPL](https://es.wikipedia.org/wiki/Licencia_p%C3%BAblica_general_de_GNU) para cualquier uso compatible con esta licencia, pero para aquellas empresas que quieran incorporarlo en productos [privativos](https://es.wikipedia.org/wiki/Software_propietario) deben comprar a la empresa una licencia específica que les permita este uso. Está desarrollado en su mayor parte en [ANSI C](https://es.wikipedia.org/wiki/ANSI_C).

Al contrario de proyectos como [Apache](https://es.wikipedia.org/wiki/Servidor_HTTP_Apache), donde el software es desarrollado por una comunidad pública y los [derechos de autor](https://es.wikipedia.org/wiki/Derechos_de_autor)del código están en poder del autor individual, MySQL es patrocinado por una empresa privada, que posee el copyright de la mayor parte del código. Esto es lo que posibilita el esquema de licenciamiento anteriormente mencionado. Además de la venta de licencias privativas, la compañía ofrece soporte y servicios. Para sus operaciones contratan trabajadores alrededor del mundo que colaboran vía [Internet](https://es.wikipedia.org/wiki/Internet). MySQL AB fue fundado por [David Axmark](https://es.wikipedia.org/w/index.php?title=David_Axmark&action=edit&redlink=1), [Allan Larsson](https://es.wikipedia.org/w/index.php?title=Allan_Larsson&action=edit&redlink=1) y [Michael Widenius](https://es.wikipedia.org/wiki/Michael_Widenius).

**Caracteristicas:**

* Amplio subconjunto del lenguaje [SQL](https://es.wikipedia.org/wiki/SQL). Algunas extensiones son incluidas igualmente.
* Disponibilidad en gran cantidad de plataformas y sistemas.
* Posibilidad de selección de [mecanismos de almacenamiento](https://es.wikipedia.org/wiki/Mecanismos_de_almacenamiento_(MySQL)) que ofrecen diferentes velocidades de operación, soporte físico, capacidad, distribución geográfica, transacciones.
* Transacciones y [claves foráneas](https://es.wikipedia.org/wiki/Clave_for%C3%A1nea).
* Conectividad segura.
* [Replicación](https://es.wikipedia.org/wiki/Replicaci%C3%B3n_(Inform%C3%A1tica)).
* Búsqueda e [indexación](https://es.wikipedia.org/w/index.php?title=Indexar&action=edit&redlink=1) de campos de texto.

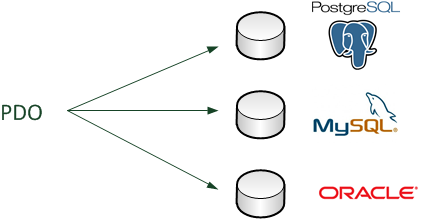


**Figura 12. Logo MYSQL**

### PDO

La extensión *Objetos de Datos de PHP* (PDO por sus siglás en inglés) define una interfaz ligera para poder acceder a bases de datos en PHP. Cada controlador de bases de datos que implemente la interfaz PDO puede exponer características específicas de la base de datos, como las funciones habituales de la extensión. Se ha de observar que no se puede realizar ninguna de las funciones de las bases de datos utilizando la extensión PDO por sí misma; se debe utilizar un [controlador de PDO específico de la base de datos](http://php.net/manual/es/pdo.drivers.php) para tener acceso a un servidor de bases de datos.

PDO proporciona una capa de abstracción de *acceso a datos*, lo que significa que, independientemente de la base de datos que se esté utilizando, se emplean las mismas funciones para realizar consultas y obtener datos. PDO *no* proporciona una abstracción de *bases de datos*; no reescribe SQL ni emula características ausentes. Se debería usar una capa de abstracción totalmente desarrollada si fuera necesaria tal capacidad.



**Figura 13. Abstracción de Datos PDO**

### ERP

Es un sistema de gestión y de datos único, donde converge toda la información de una empresa y ésta es utilizada para tomar decisiones. Decir esto es referirnos a algo muy amplio y sin límites claramente definidos: pues eso es un ERP, algo que puede ser tan variable en tamaño y funcionalidad como variables pueden ser el tamaño y fines de las millones de empresas que funcionan en los distintos países.

La utilidad básica de un software de estas características, es la de ayudar a administrar empresas de cualquier tipo, automatizando todos sus procesos. También ayuda a controlar lo que una organización tiene (stock e inventario) o hace (flujos de trabajo).

Una empresa con un ERP posiblemente esté en ventaja respecto a otra que no disponga de un sistema de este tipo. ¿Por qué? Porque gracias a un ERP la empresa automatiza su gestión: tiene un mayor control de lo que hace y un ahorro de costes (eficiencia y eficacia). Esto hace a las empresas más competitivas: cuesta menos rellenar un formulario en la pantalla y pulsar aceptar (con lo que automáticamente llegará a toda la empresa) que rellenar un formulario en papel, enviarlo por mensajería, y que llegue a su destino. Además un ERP integra todas las actividades de todos los departamentos de una compañía en una sola aplicación, permitiendo modificar datos, realizar consultas y generar informes rápidamente.

Existen básicamente dos tipos de ERP, los generalistas y los especializados. Los ERP generalistas, también llamados horizontales, sirven para cualquier empresa. Los ERP especializados, también llamados sectoriales o verticales, intentan dar soluciones a sectores específicos como la sanidad, la distribución, los medios audiovisuales o la construcción por indicar varios ejemplos.



**Figura 14. Logo ERP**

## 2.4 Desarrollo del proyecto Metodología para el Desarrollo Web Dinámico enfocado en Sistemas de Gestión y Administración Empresarial

Para la realización de dicho proyecto se optó por dividirlo en diferentes fragmentos para llevar un mejor control en la secuencia de la metodología, dichos fragmentos son los siguientes:

* Login
* Menú y Panel Dinámicos
* Altas, Bajas y Cambios con PDO , JQuery y PHP
* Pagos Online (Conekta)
* Suscripciones (Conekta)
* Validaciones
* Notificaciones
* Consultas en PHP usando Transact-SQL para elaboración de Reportes Ejecutivos

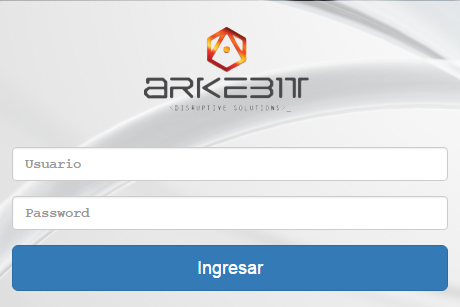
Para la realización de los siguientes modulos del sistema se necesitará lo siguiente:

* Servidor web (Apache, IIS, Nginx, etc.)
* Servidor de Bases de Datos (MYSQL, MariaDB, Oracle , etc.)
* PHP
* Editor de Textos
* Jquery
* Bootstrap V3

### Login

En la elaboración de cualquier aplicación, ya sea web o de escritorio, y especificamente si hablamos de un modulo de recursos empresariales como lo es un ERP, es de suma importancia, tener el control de que usuarios ingresan al sistema y bajo que rol lo hacen, por lo que se explicará a continuación el modulo de “Login” de un sistema.

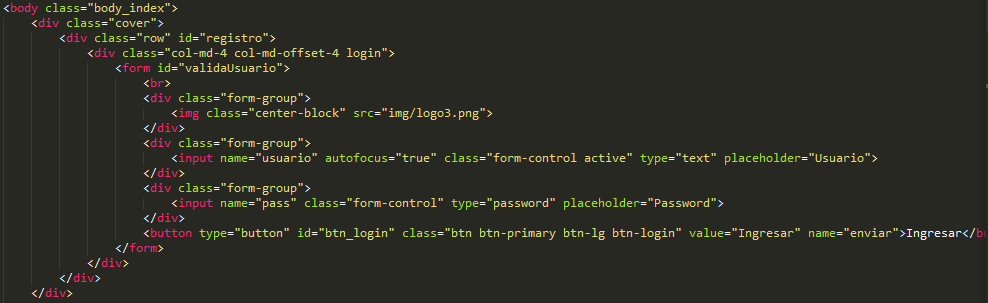
#### Front-End (Diseño)

El diseño del login es muy sencillo, consta basicamente de 2 cajas de texto y un botón, el primer textbox será para el nombre de usuario y el segundo es para la contraseña, mientras que el botón azul “Ingresar” es el que disparará la acción de si existe el usuario y si corresponde su contraseña.

**Figura. Apariencia del Login**

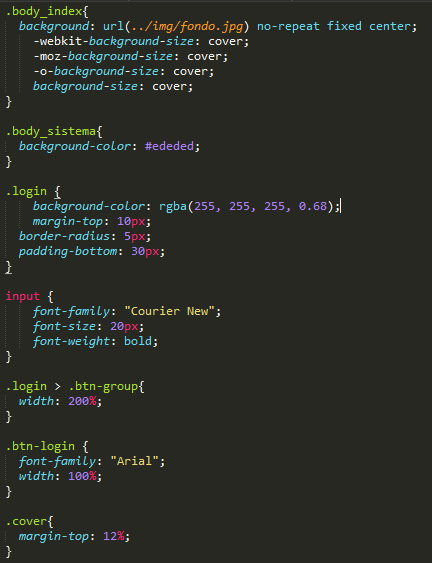
A continuación se muestra el código HTML, en donde se muestran 2 inputs de tipo texto acompañados de un button con type button, de manera en que pueda accionar en el sistema, las dos variables a buscar, en este caso el nombre y la contraseña.

Todo lo anterior dentro de la etiqueta <form> ya que es un pequeño formulario.



**Figura. Código HTML de un login**

En cuanto al código CSS, le añadimos unos estilos básicos para poder darle posición y estilo a los elementos que contienen el login, de manera en que contengan sobriedad y ergonomía.

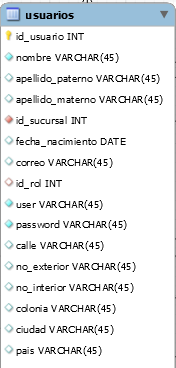


**Figura. Código CSS del login**

#### Back-end

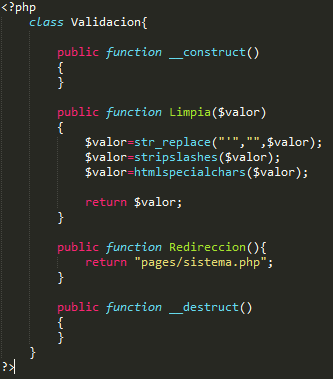
Para realizar el back-end se utilizará PHP como lenguaje ya que ofrece una gran variedad de funciones utiles para estos ejemplos.

En esta ocasión utilizamos solo una tabla llamada “usuarios” de manera en que haremos nuestras peticiones de la base de datos a través de esa tabla.



**Figura: Tabla usuarios requerida para login**

Para que funcione el login, es necesario el uso de varios archivos php, uno que validara lo que se inserta en los textbox, otro que hará la respuesta del login y el final, que tiene los métodos necesarios para la conexión y los métodos de PDO.



**Figura. Clase Validacion**

Esa clase tiene 2 funciones primarias, la función Limpia() que recibe un parametro, (en este caso el usuario y luego la contraseña) y le quita los caracteres en blanco y caracteres especiales, y otra función llamada Redirección que regresa la ruta principal del sistema “pages/sistema.php”.



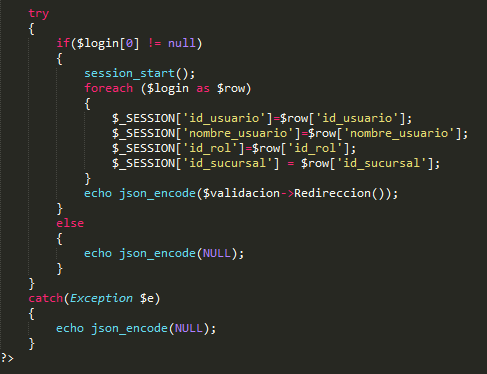
**Figura. LogRespuesta.php**

Este script php, es la parte medular del login por parte del backend ya que hace básicamente lo siguiente:

Al principio, se especifica que contendrá código AJAX y por ende recibirá una aplicación del tipo JSON.

Todo lo que obtenga por parte del formulario, lo extraerá del tipo POST, y posteriormente hace mención a dos scripts necesarios para funcionar “validacion.php” que se encarga de realizar todo lo de la clase anteriormente mencionada y el script “database.php”, que es donde se encuentra la clase que hace la conexión y desconexión de la base de datos, así como de la función “SELECCIONAR” que permite realizar consultas.

Se realizan las instancias pertinentes de dichos scripts y posterior a eso, se ejecuta una consulta que seleccionara datos del usuario que se teclearon en los textbox.



**Figura. TRY-CATCH de la CONSULTA a usuarios**

Despues de que se ejecutó una consulta a la base de usuarios, se guardó en la variable de tipo array $login, en donde si su indice 0 (Encontró resultados) es diferente a nulo, pone los datos de la consulta en variables de sesión para utilizarlos en despues dentro de la sesión y aplica una redirección utilizando la clase validación y la función Redireccionar().

#### Front-End (JQuery)

Solo hace falta acciónar la tecla enter o el botón para que se dispare el código de la siguiente figura el cual realizará lo siguiente:

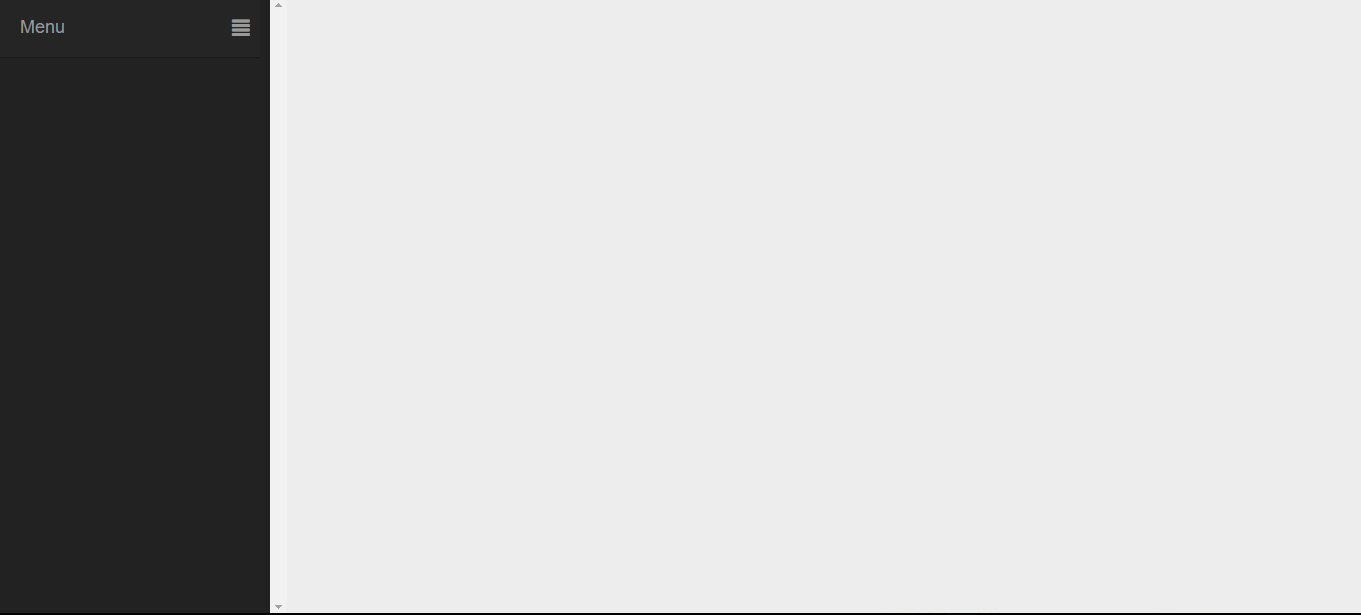
Guarda en una variable llamada login el resultado de un ajax que está apuntado al script LogRespuesta.php, enviandole como parametro el formulario completo del login y declarandole que será del tipo JSON, en la propiedad done de dicho ajax, se declara una función llamada datos que se analizará con un switch, en donde si el case es null o bien no encontró nada, mandará una alerta diciendo que el acceso está denegado y que el usuario y la contraseña no son válidos, en caso contrario, mandará una redirección a datos, y permitirá el acceso al menú principal.

**Figura. Acción disparadora de la tecla enter y del botón Ingresar**

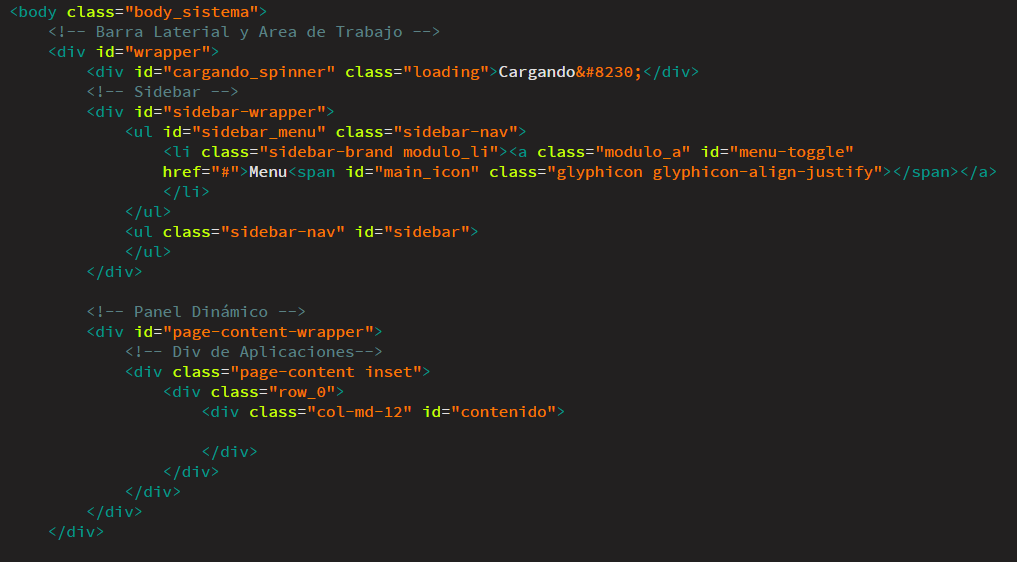
### Menú y Panel Dinámicos

En este apartado se explicará de una manera clara y conciza el proceso necesario para poder realizar un menú dinámico utilizando el método de recursividad y un panel dinámico de trabajo con el cual se optimice cualquier sistema a elaborar ya que con esto se evitarían cambios posteriores en caso de agregar una o más partes al menú.

#### Front-End (Diseño)

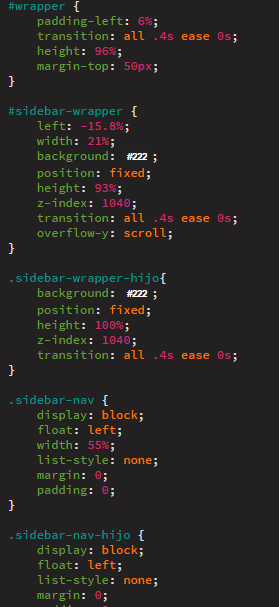
El diseño del menú y del panel dinámico debe ser el siguiente, del lado izquierdo se puede apreciar la estructura del menú base sin ningún módulo y por ende sin ninguna aplicación y del lado derecho el panel dinámico donde posteriormente se mostrará el contenido de cada aplicación.

**Figura. Estructura Básica del menú y del panel**

En cuanto al código HTML para darle forma al menú y al panel dinámico la estructura es muy sencilla, basta con agregar un div contenedor con 2 divs hijos, el primero fungirá como el menú y el segundo como el panel dinámico, ambos divs tienen estilos aplicados para ajustar su tamaño y los colores de fondo. A continuacion podemos ver el código HTML necesario para realizar el diseño.

**Figura. Código HTML Menú y Panel Dinámico**

En cuanto al código CSS podemos ver a continuación una parte de él donde se incluye lo principal para darle el tamaño y la posición a cada uno de los elementos antes citados.



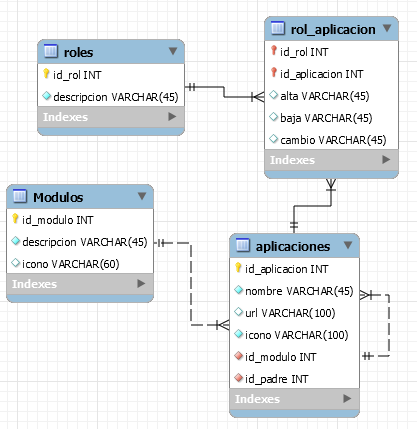
**Figura. Resumen Estilos CSS Menú y Panel Dinámico**

En cuanto a la programación del lado del cliente esto es lo necesario para realizar el panel y el menú de una manera dinámica, a continuación se explicará la programación del lado del servidor (Back-end).

#### Back-end

Para realizar el menú y el panel dinámico es necesario 4 tablas de una base de datos, estas tablas nos servirán para guardar los módulos, las aplicaciones, los roles y las aplicaciones por rol para luego con PHP desplegarlas en el menú.

En la siguiente figura podemos apreciar el modelo de la base de datos necesaria para este ejemplo.

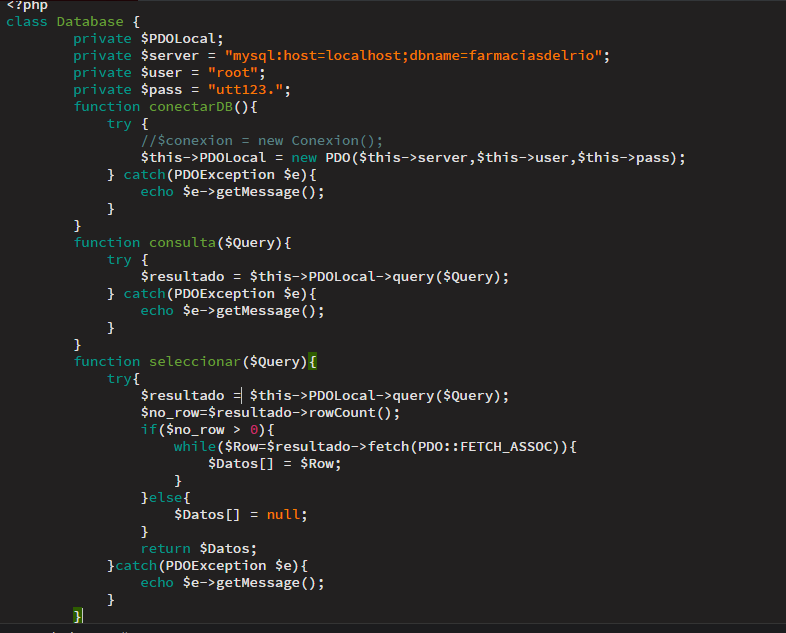


**Figura. Modelo de la Base de Datos para el menú y el panel dinámico**

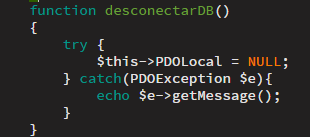
Podemos apreciar que la tabla padre es la de Módulo ya que cada módulo puede contener n cantidad de aplicaciones, en la tabla de aplicaciones podemos ver que incluye los campos de nombre, de url (que luego será utilizado para el panel dinámico), el icono, el id\_modulo para indicar la pertenencia de cada aplicación a el módulo correspondiente y al final una llave compuesta llamada id\_padre que jugará un papel muy importante en el método de recursividad ya que este campo indicará si una o más aplicaciones tienen una aplicación padre. Después vemos las tablas rol y rol\_aplicación donde, dependiendo del rol, el sistema será capaz de mostrar sólo las aplicaciones permitidas por rol.

Para comenzar a trabajar con PHP, es necesario tener una clase utilizando PDO de PHP que será la encargada de procesar todas las interacciones de nuestro sistema con la base de datos, en este caso la clase se llama Database y contiene los métodos para conectar la base de datos, para hacer una query usando Transact-SQL, para consultar algo a la base de datos y por último para desconectar la base de datos.

En las siguientes figuras podemos ver la clase Database con los métodos antes mencionados.



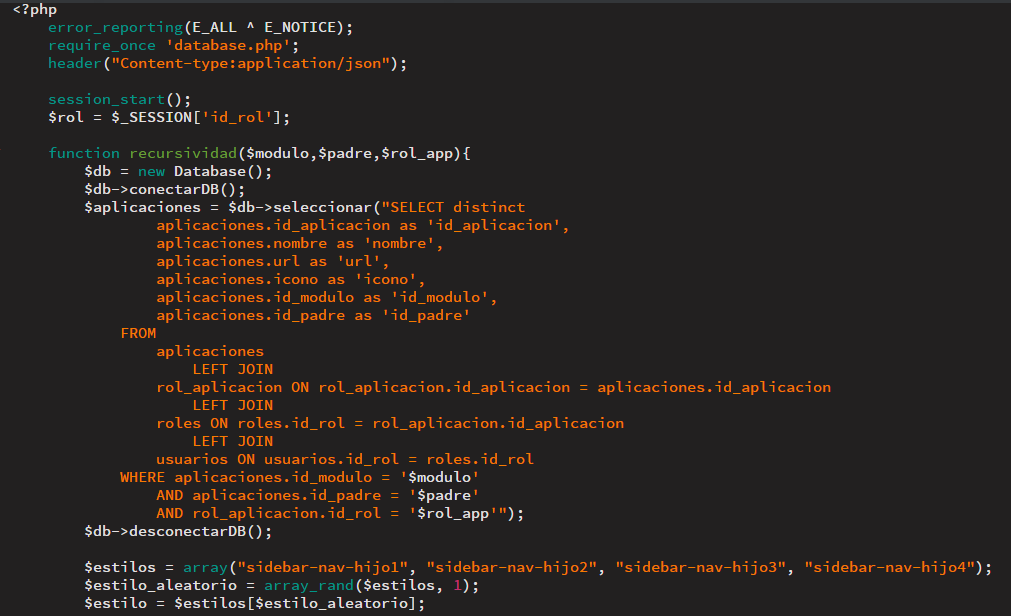
**Figura. Clase Database**



**Figura. Clase Database Metodo para desconectar la Base de Datos**

A continuación se necesita hacer una consulta a la base de datos desde PHP que nos permita saber las aplicaciones existentes en la base de datos, en la siguiente figura podemos apreciar dicha consulta se encuentra dentro de una función llamada recursividad que recibe 3 parámetros, $modulo que en este caso será el id del módulo para que se seleccionen nadamas las aplicaciones del módulo solicitado, $padre para que se seleccionen aplicaciones hijas de aplicaciones padres en caso de que existan sub-acplicaciones y por último $rol\_app para que solo se selecccionen las aplicaciones del rol solicitado, dentro de la consulta se seleccionan los campos aplicaciones.id\_aplicacion as 'id\_aplicacion', aplicaciones.nombre as 'nombre', aplicaciones.url as 'url', aplicaciones.icono as 'icono', aplicaciones.id\_modulo as 'id\_modulo', aplicaciones.id\_padre as 'id\_padre' con sus respectivos alias y por último se indican las tablas de donde se obtendrán los datos y la claúsula WHERE para filtrar los resultados haciendo referencia a los 3 parámetros del método.

Después de ejecutar la consulta se crea un arreglo con varias clases de css predefinidas para luego , ejecutar la función “array\_rand” de PHP para que elija alguna de esas clases a una variable llamada $estilo que se aplicará posteriormente para darle un pequeño color al submenú de aplicaciones.



**Figura. Función recursividad, consulta de aplicaciones**

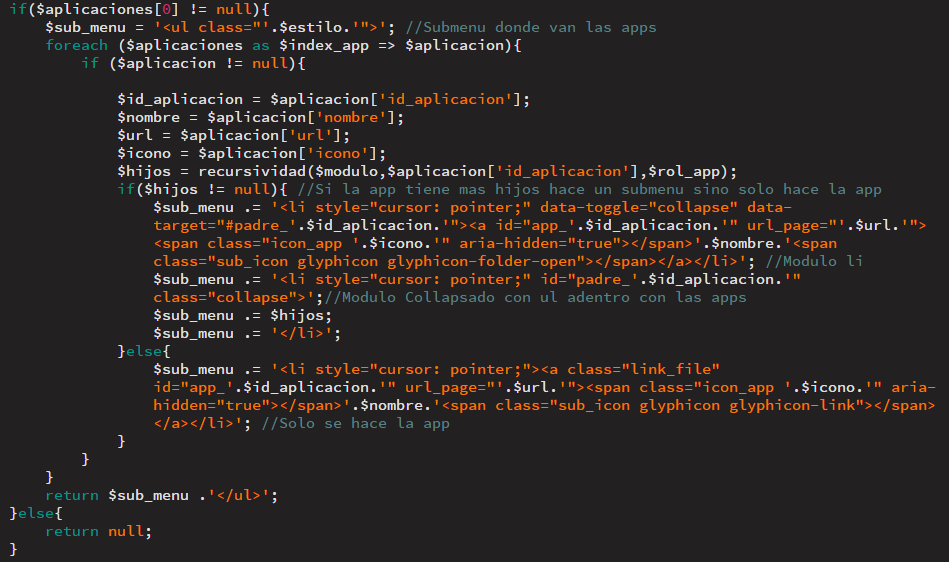
A continuación se comprueba si la consulta tiene datos, si los tiene entra a un foreach para recorrer los datos de la consulta y se declaran las siguientes variables:

* $id\_aplicacion = $aplicacion['id\_aplicacion']
* $nombre = $aplicacion['nombre']
* $url = $aplicacion['url']
* $icono = $aplicacion['icono']
* $hijos = recursividad($modulo,$aplicacion['id\_aplicacion'],$rol\_app)

Estas variables hacen referencia a los campos solicitados en la consulta, sin embargo en la variable $hijos podemos ver que se hace referencia a la función de recursividad, esto para volver a ejecutar la consulta y comprobrar por medio del campo $aplicación[id\_aplicacion] si la aplicación tiene aplicaciones hijas dentro de la tabla.

Se comprueba lógicamente esto dentro de la sentencia if y si se cumple se crea un submenú dentro del menú con las subaplicaciones hijas de la aplicación padre, si no se cumple dicha condición sólamene se crea la aplicación dentro del menú, todos los submenús y aplicaciones se van concatenando a la variable $sub\_menú que al final del foreach es retornada, si la consulta principal no tiene datos simplemente se regresa null.

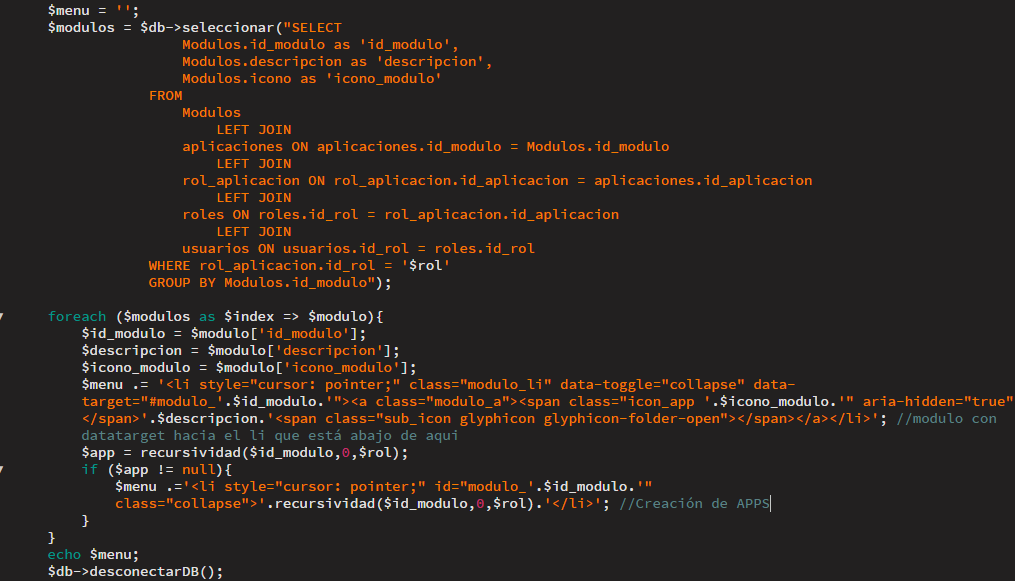
En la siguiente figura se puede apreciar lo antes explicado de una manera más clara ya con el código dentro de un script de PHP totalmente funcional.



**Figura. Recursividad, creación de submenús y aplicaciones**

A continuación es necesario consultar los módulos existentes en la base de datos y crear el menú en forma utilizando el método de recursividad, en la consulta de los módulos es necesari tener los siguientes campos: id\_modulo, descipcion y el icono para luego ejecutar un foreach para recorrer la consulta y crear los módulos dentro del menú y posteriormente, utilizando el método de recursividad anteriormente explicado, las aplicaciones correspondientes a cada módulo y a cada rol en caso de que estas existan.

En la siguiente figura se puede ver la consulta para saber los módulos, la ejecución del método de recursividad y la comprobación para saber si existen aplicaciones del módulo activo en el foreach para la creación de la estructura del menú dentro de la variable $menu para luego ser impresa con la función echo de PHP.



**Figura . Creación del Menú Dinámico**

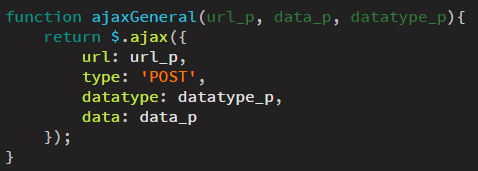
#### Front-End (Jquery)

Luego de imprimir la variable $menu que contiene toda la estructura HTML del menú es necesario inyectar ese código a nuestro menú molde, esto se hace por medio de JQUERY.

Para poder interactuar con el código PHP desde JQUERY es necesario utilizar una función propia llamada AJAX, AJAX nos permite hacer peticiones asícronas al servidor para poder procesar datos y respuestas de nuestro servidor.

En este caso se optó por utilizar AJAX en una función llamada ajaxGeneral dentro de una archivo con extensión .js llamado “apps.js”, está función recibe 3 parámetros (url\_p, data\_p y datatype\_p) estos 3 parámetros son llamados dentro de la función, el primero para indicar la ruta del script de PHP, el segundo para indicarle los datos (en caso de tener que mandar datos) y por último el tipo de datos que puede ser de tipo HTML, JSON, etc.

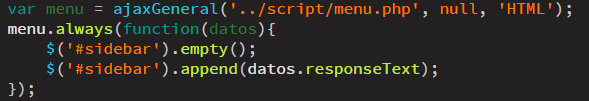
A continuación podemos ver la creación de esta función.



**Figura. Función ajaxGeneral**

Teniendo está función sólo queda mandarla llamar dándole como parámetros la ruta donde creamos la estructura lógica del menú, los datos como nulos ya que no es necesario mandar algún tipo de dato y por último el tipo de datos, en este caso HTML ya que el código de la respuesta es a fin de cuentas código HTML creado con PHP , todo esto se guardará en una variable llamada menú que luego de mandar llamar la función se utiliza el estado “done” nativo de AJAX que hace referencia a cuando la petición hacia el servidor ha sido exitosa y este ha devuelto alguna respuesta, en este caso la variable $menu que contiene nuestro menú dinámico, dentro de este status que funge como función y recibe un parámetro (datos) que es la respuesta del servidor, dentro de esta función se hace referencia al menú base que tiene como id “sidebar” y se limpia con la función empty() para evitar conflictos con el diseño y posteriormente se utiliza la función append() de Jquery para insertar la variable datos que contiene nuestro menú y el método .responseText para convertirlo a string.

En la siguiente figura se muestra a manera de código lo anteriormente explicado.



**Figura. Creación del menú en Front-End con Jquery**

Al tener esto nos basta con irnos a nuestro proyecto y refrescar para ver trabajar nuestro código.

En la siguiente figura se puede apreciar ya el menú creado con sus respectivos módulos y aplicaciones.



**Figura. Menú Dinámico Creado**

A continuación se explicará la manera de crear el panel dinámico, pero ¿Cómo panel dinámico?, el objetivo es utilizar solamente un archivo para mostrar el contenido de las aplicaciones que se encuentran en archivos distintos, esto para darle a nuestro sistema una fluidez óptima.

Para realizar esto es necesario JQUERY y su función ajax, el concepto es bastante sencillo, cuando el usuario de click en alguna de las aplicaciones del menú, Ajax hará una petición a la ruta especificada y traerá el contenido del archivo y lo inyectará en el div de la derecha que fungirá como nuestro panel dinámico.

El código es bastante sencillo, es necesario hacer referencia al menú por medio de su id (#sidebar) y jugar con el evento onClick de la clase .link\_file que está contenida en todas y cada una de las aplicaciones del menú, se utiliza el evento on porque el menú está creado en tiempo de ejecución y sin este evento no es posible realizar nuestro panel.

Dentro de este evento es necesario sacar un atributo de la aplicación a la cual se le ha dado click, este atributo es url\_page y contiene el nombre del archivo que contiene los datos de la aplicación y se guarda en la variable route.

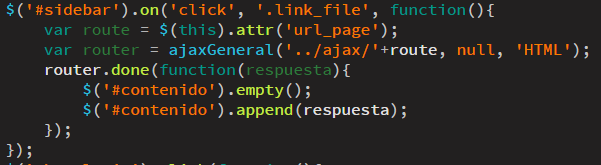
A continuación se puede ver un ejemplo de una aplicación con su atributo url\_page



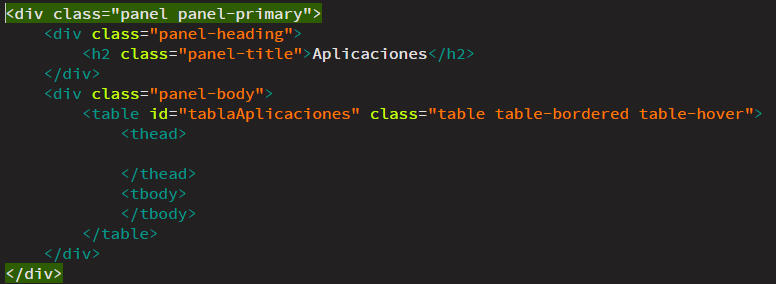
**Figura . Atributo url\_page de una aplicación**

A continuación se hace mención a la función ajaxGeneral y se le da como primer parámetro ‘../ajax/’ + route que hace mención a la ruta donde AJAX buscará el contenido de la aplicación utilizando la variable route que contiene el nombre del archivo .php, los datos se mandan nulos y por último se le dice que trabajará con datos HTML, esta función se guarda en la función router y luego se hace mención al status .done y se hace una función anónima que tiene como parámetro la respuesta del servidor en este caso el contenido de la aplicación y por último se hace mención al div que contendrá el panel dinámico, este se limpia con la función emtpy() y se le inyecta la respuesta con la función append().

En la siguiente figura se puede ver a manera de código lo anteriormente explicado.

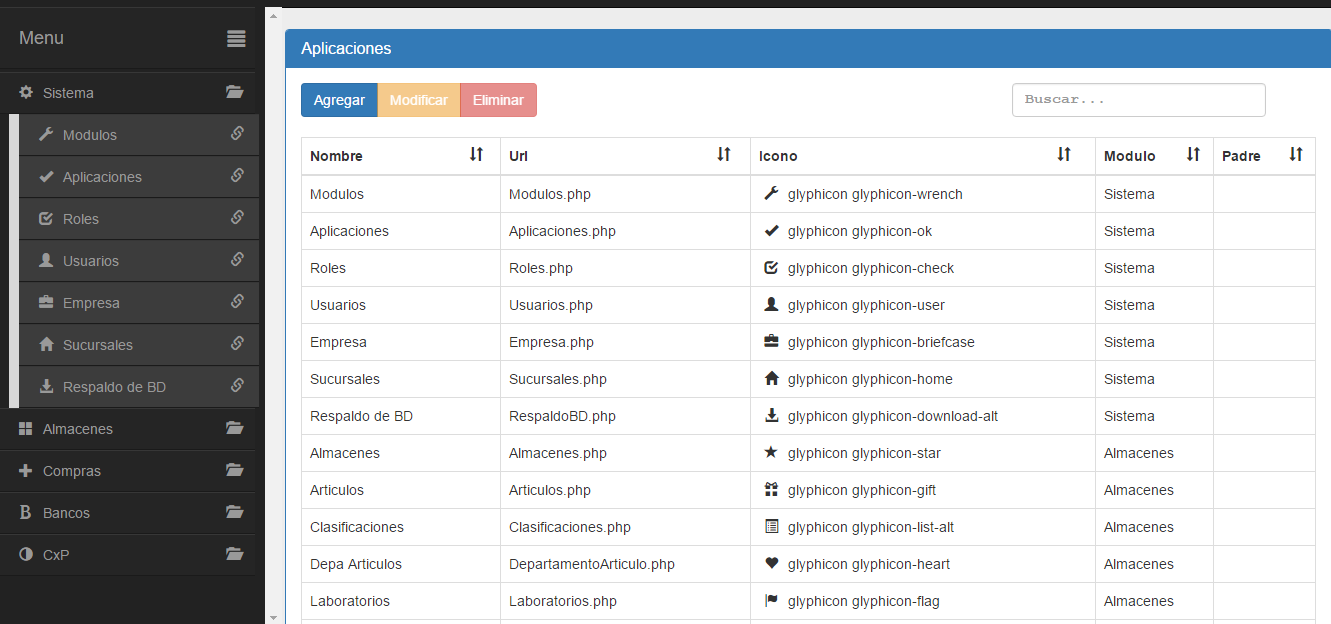


**Figura . Código Jquery para el panel dinámico**

Para prueba de concepto se utilizará la aplicación “Aplicaciones” que contiene el siguiente código HTML.

**Figura. Código HTML Aplicaciones.php**

Para comprobar que nuestro código funcionas se hace click en la aplicación y si todo está correcto el panel dinámico deberá mostrar el contenido de la aplicación como lo muestra la siguiente imagen.



**Figura . Panel Dinámico Finalizado**

### Altas, Bajas y Cambios con PDO, JQuery y PHP

### Pagos Online (Conekta)

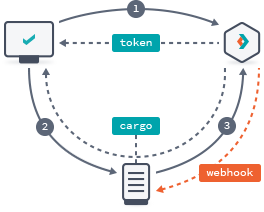
En la actualidad una de las tecnologías emergentes más útiles y más utilizadas por las personas por el ahorro de tiempo que representa es el e-commerce, el e-commerce requiere de pagos online, ya sea con tarjetas de débito o tarjetas de crédito y es de suma importancia que los sistemas que hagan enfasis en está tecnología tengan un sistema eficiente y seguro para realizar pagos online.

Conekta es una empresa que ofrece una API multilenguaje que permite realizar cobros online en cualquier sistema que este utilizando dicha API, pero ¿Cómo lo hace?.

Hay que entender el flujo de la transacción, a continuación se explicará paso a paso este flujo.

1. En primer lugar se capturan los datos del usuario y de la tarjeta (no\_tarjeta, CVC, fecha de expiración, nombre del usuario y el correo).
2. Conekta a tráves de su API en una función manda dichos datos a sus servidores y válida la información enviada para devolver un token a nuestro sistema.
3. Este token es enviado a nuestro servidor para procesar la información detallada del pago (articulo(s), precio, datos fiscales para facturación, etc.)
4. Este token puede ser guardado en nuestra base de datos y al procesar correctamente la transacción en nuestro servidor conekta realiza el cargo a la tarjeta.
5. Al cabo de unos cuantos días (según sea la tarjeta) conekta hace el depósito de la cantidad de la compra menos una pequeña comisión a una tarjeta que el dueño del sistema registre en su panel de administración.

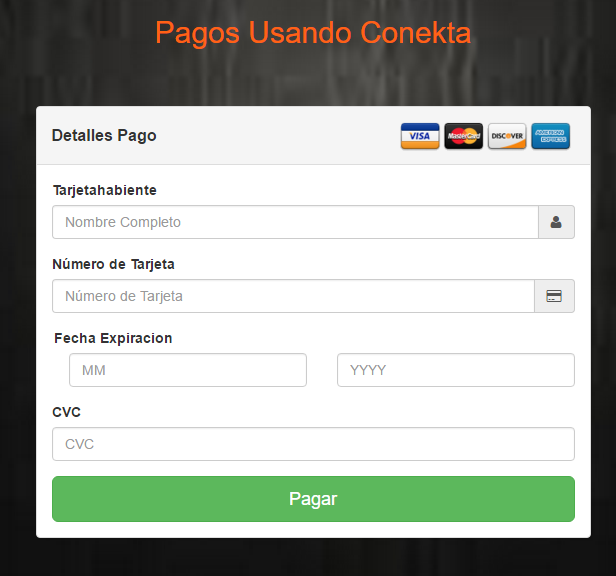
En la siguiente figura podemos ver representado dicho flujo



**Figura . Flujo de datos Conekta**

#### Front-End (Diseño)

Para realizar la prueba de concepto se diseño un formulario para capturar los datos de la tarjeta. En la siguiente figura podemos apreciar el diseño de este formulario.



**Figura . Diseño Formulario**

El formulario es muy sencillo y costa de 5 cajas de texto las cuales contendrán los siguientes datos:

* El nombre del tarjetahabiente, dicho textbox tiene un atributo llamado data-conekta que tiene como valor [name], dicho atributo es usado por la API y por ende no debe ser cambiado.
* El numero de tarjeta, dicho textbox tiene un atributo llamado data-conekta que tiene como valor [numbre], dicho atributo es usado por la API y por ende no debe ser cambiado.
* El mes de expiración, dicho textbox tiene un atributo llamado data-conekta que tiene como valor [exp\_month], dicho atributo es usado por la API y por ende no debe ser cambiado.
* El año de expiración a cuatro digitos, dicho textbox tiene un atributo llamado data-conekta que tiene como valor [exp\_year], dicho atributo es usado por la API y por ende no debe ser cambiado.
* El número CVC, dicho textbox tiene un atributo llamado data-conekta que tiene como valor [cvc], dicho atributo es usado por la API y por ende no debe ser cambiado.

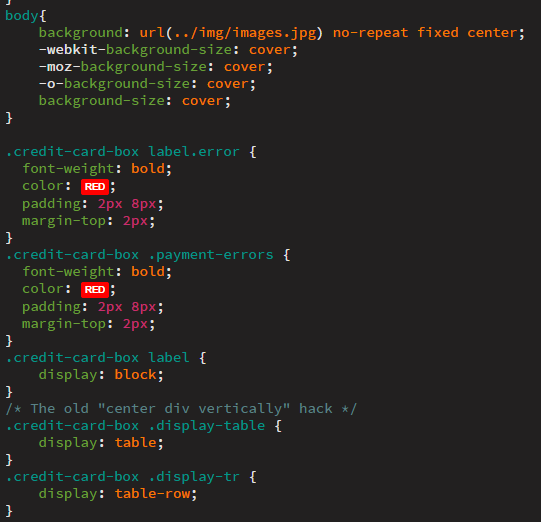
En la siguiente figura se puede apreciar un poco del código HTML donde podemos ver el atributo antes mencionado.



**Figura. Código HTML Formulario Pagos Conekta**

En cuanto al código CSS, se utilizaron clases y ids para darle un acomodo y un tamaño al formulario y su contenido, además de darle un diseño sobrio a la pantalla en general.

En la siguiente figura apreciamos un poco de este código.



**Figura. Código CSS Pagos Conekta**

#### Front-End (JavaScript y Jquery)

Para poder utilizar las funciones de la API de conekta en Front-End necesitamos usar su script el cual podemos adjuntar a nuestro documento como un CDN con la siguiente linea de código.

**Figura . Adjunción del .js de conekta**

En un script hay que adjuntar la llave pública que nos proporciona conecta, dicha llave es necesaria para poder utilizar la API, a continuación podemos ver la llave adjunta al sistema.



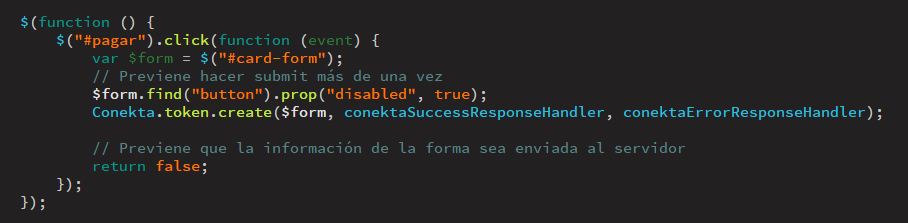
**Figura . Llave Pública Adjunta Pagos Conekta**

A continuación se debe mandar toda la información del formulario a los servidores de conekta por medio de una función en Jquery utilizando la API de conekta, debemos situarnos en el evento click del botón del formulario que dice pagar, al darse click se debe caputar el formulario dentro de una variable.

Luego dentro de ese formulario hay que indicarle que deshabilite el botón de pagar para evitar que se envíe más de una petición a los servidores de conekta.

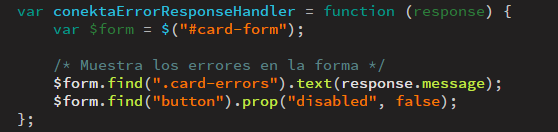
Luego de hacer esto es necesario utilizar una función de la API de conekta, llamada Conekta.token.create que recibe 3 parámetros, el formulario y dos enlazadores de respuesta, uno donde se guardará la respuesta exitosa y otra la respuesta fallida con los errores en caso de existir.

Por ultimo devolvemos false para evitar que estos datos sean enviados a nuestro servidor. En la siguiente figura podemos ver lo anteriormente explicado.



**Figura . Función Conekta Mandar datos Formulario**

Después de esto, conekta procesará y validará los datos enviados y mandará una respuesta que será guardada en alguno de los dos handlers, si conekta nos manda una respuesta fallida entra el conektaErrorResponseHandler y ejecuta una función con un parámetro el cual es la respuesta, dentro de esta función se guarda en una variable el formulario y luego dentro de la clase .card-errors que contiene un div se muestran los errores y se habilita el botón de pagar para que se vuelvan a enviar los datos, está vez de manera correcta. En la siguiente figura podemos apreciar esta función.



**Figura . Función conektaErrorResponseHandler**

Mas sin en cambio, conekta nos envía un success response, entrará en juego el otro handler (conektaSuccessHandler) y ejecutará una función que recibirá un parámetro, en este caso el token creado por conekta, dentro de esta función guardamos el formulario en una variable y le insertamos un nuevo input de tipo oculto con el nombre ‘conektaTokenId’ y le asignamos el id del token que nos envió conekta.

Luego ejecutamos un ajax para mandar dicho token y el nombre del cliente a nuestro servidor por el método POST , no debemos enviar los datos de la tarjeta para evitar problemas con los usuarios y con conekta, con el token y el nombre podemos hacer el cobro de nuestro producto y/o servicio.

Si nuestro servidor ejecutó correctamente el cobro y devolvió una respuesta exitosa en la función success hacemos uso del plugin de notificaciones para mostrar que se ha realizado de manera exitosa la petición y mostramos los datos de la transacción, limpiamos los campos del formulario y por último habilitamos de nuevo el botón para nuevas peticiones.



**Figura. Envio de datos a nuestro servidor pagos Conekta**

#### Back-End

En nuestro servidor debemos hacer uso de la API de Conekta para PHP que podemos descargar desde el repositorio oficial de conekta en Github (<https://github.com/conekta/conekta-php>), luego hay que adjuntarlo a la carpeta raíz de nuestro proyecto y hay que agregar el archivo principal de dicha librería ubicada dentro de conekta-php/lib.

Además hay que hacer uso de la llave privada que nos proporciona conekta para poder trabajar de lado del servidor. En la siguiente figura vemos dicha explicación a manera de código.



**Figura . Adjunción del archivo general de conekta y de la llave privada**

A continuación hay que hacer uso de la función exctract de PHP para extraer los datos enviados en formato JSON desde el AJAX y guardarlos en variables para su posterior uso.

Luego hay que generar un nuevo cargo utilizando los métodos de nuestra API de conekta, para hacer esto declaramos una variable, en este caso llamada $charge y usamos el método create de la clase Conekta\_Charge y le damos los datos de la compra guardados en un array, en este ejemplo le decimos que es un pago de balones de futbol, le damos un id de referencia, la cantidad, el tipo de moneda, en este caso pesos mexicanos, le damos como valor a la tarjeta el token que enviamos por ajax y que fue extraido por la función extract de php y guardado en la variable $conektaTokenId, luego le damos los detalles del comprador, su nombre que es sacado de la variable $nombre recibida por php desde el AJAX que enviamos, su telefono, un email de prueba, datos del cliente en el sistema, los items de la compra con su descripción, su precio unitario, la cantidad, el SKU y su categoria.

En la siguiente figura podemos ver la anterior explicación a manera de código dentro de nuestro ejemplo.



**Figura. Detalles Cobro Conekta 1**

Luego capturamos los datos fiscales del comprador dentro del mismo array del cargo, con la dirección, la ciudad, el estado, el país, el código postal, el id o el RFC en este caso, el nombre de la persona fisica o la persona moral, el teléfono y el correo electrónico.

Por último switcheamos el status del cargo efectuado y si es “paid” o pagado mandamos una respuesta con los datos generales de la compra y un mensaje para decirle al cliente que la compra ha sido exitosa, dicha respuesta es mostrada por JQUERY a tráves de la notificación creada en la función del status success del método AJAX que utilizamos para mandar la información del token y del cliente a nuestro servidor para realizar el cargo de la compra.

En la siguiente figura podemos ver representada dicha explicación

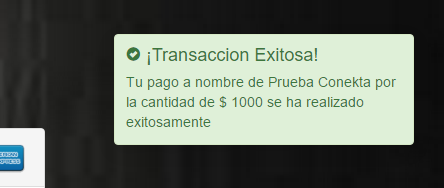


**Figura. Detalles Cobro Conekta 2 y procesamiento de respuesta**

Para realizar la prueba de concepto utilizamos datos de tarjetas de créditos para hacer pruebas que son proporcionados por conekta, dichos datos son los siguientes:

* Número de tarjeta 4242424242424242
* CVC 123
* Mes de Expiración 12
* Año de Expiración 2020
* Tarjetahabiente Conekta Prueba

Al realizar el pago el sistema gestiona todos los datos y al ejecutarse de manera correcta la compra el sistema nos muestra la siguiente alerta indicando que la compra fue exitosa.



**Figura. Notificación Compra Exitosa Conekta.**

### Suscripciones (Conekta)

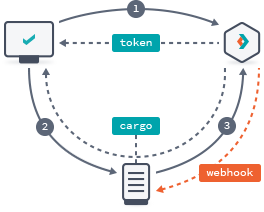
En la actualidad una de las tecnologías emergentes más útiles y más utilizadas por las personas es la suscripción a servicios online, servicios como Netflix, Spotify, Google Play Music utiliza el método de suscripciones para cobrar su servicio cada cierto tiempo a sus clientes sin que estos tengan que estar realizando el pago cada vez que se tenga que pagar, sino que al marcar la fecha de cobro automáticamente se hace el cargo a la tarjeta registrada por el cliente para seguir gozando del servicio al que se encuentra suscrito ,

Conekta es una empresa que ofrece una API multilenguaje que permite realizar cobros online, en este caso a tráves de suscripciones en cualquier sistema que este utilizando dicha API, pero ¿Cómo lo hace?.

Hay que entender el flujo de la transacción, a continuación se explicará paso a paso este flujo.

1. En primer lugar se capturan los datos del usuario y de la tarjeta (no\_tarjeta, CVC, fecha de expiración, nombre del usuario y el correo).
2. Conekta a tráves de su API en una función manda dichos datos a sus servidores y válida la información enviada para devolver un token a nuestro sistema.
3. Este token es enviado a nuestro servidor para procesar la información detallada del pago (articulo(s), precio, datos fiscales para facturación, etc.)
4. Este token puede ser guardado en nuestra base de datos y al procesar correctamente la transacción en nuestro servidor conekta realiza el cargo a la tarjeta.
5. Al cabo de unos cuantos días (según sea la tarjeta) conekta hace el depósito de la cantidad de la compra menos una pequeña comisión a una tarjeta que el dueño del sistema registre en su panel de administración.

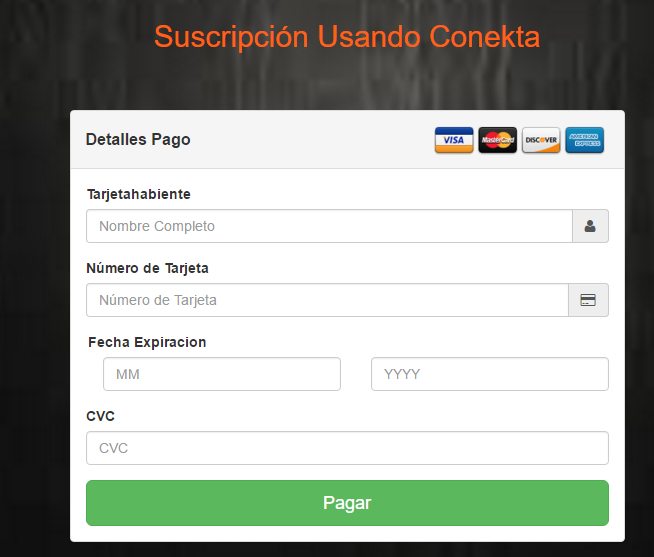
En la siguiente figura podemos ver representado dicho flujo



**Figura . Flujo de datos Conekta**

#### Front-End (Diseño)

Para realizar la prueba de concepto se diseño un formulario para capturar los datos de la tarjeta. En la siguiente figura podemos apreciar el diseño de este formulario.



**Figura . Diseño Formulario**

El formulario es muy sencillo y costa de 5 cajas de texto las cuales contendrán los siguientes datos:

* El nombre del tarjetahabiente, dicho textbox tiene un atributo llamado data-conekta que tiene como valor [name], dicho atributo es usado por la API y por ende no debe ser cambiado.
* El numero de tarjeta, dicho textbox tiene un atributo llamado data-conekta que tiene como valor [numbre], dicho atributo es usado por la API y por ende no debe ser cambiado.
* El mes de expiración, dicho textbox tiene un atributo llamado data-conekta que tiene como valor [exp\_month], dicho atributo es usado por la API y por ende no debe ser cambiado.
* El año de expiración a cuatro digitos, dicho textbox tiene un atributo llamado data-conekta que tiene como valor [exp\_year], dicho atributo es usado por la API y por ende no debe ser cambiado.
* El número CVC, dicho textbox tiene un atributo llamado data-conekta que tiene como valor [cvc], dicho atributo es usado por la API y por ende no debe ser cambiado.

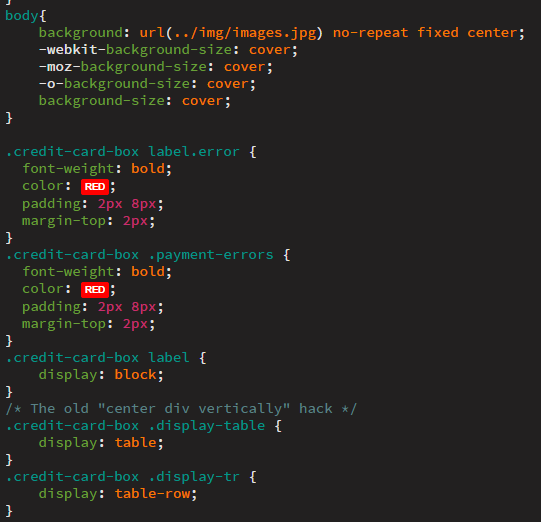
En la siguiente figura se puede apreciar un poco del código HTML donde podemos ver el atributo antes mencionado.



**Figura. Código HTML Formulario Suscripción Conekta**

En cuanto al código CSS, se utilizaron clases y ids para darle un acomodo y un tamaño al formulario y su contenido, además de darle un diseño sobrio a la pantalla en general.

En la siguiente figura apreciamos un poco de este código.



**Figura. Código CSS Suscripción Conekta**

#### Front-End (JavaScript y Jquery)

Para poder utilizar las funciones de la API de conekta en Front-End necesitamos usar su script el cual podemos adjuntar a nuestro documento como un CDN con la siguiente linea de código.

**Figura . Adjunción del .js de conekta**

En un script hay que adjuntar la llave pública que nos proporciona conecta, dicha llave es necesaria para poder utilizar la API, a continuación podemos ver la llave adjunta al sistema.



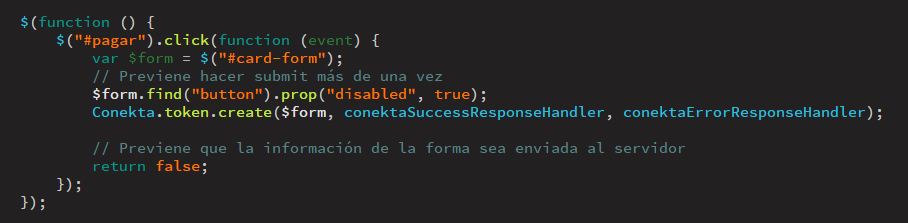
**Figura . Llave Pública Adjunta Suscripción Conekta**

A continuación se debe mandar toda la información del formulario a los servidores de conekta por medio de una función en Jquery utilizando la API de conekta, debemos situarnos en el evento click del botón del formulario que dice pagar, al darse click se debe caputar el formulario dentro de una variable.

Luego dentro de ese formulario hay que indicarle que deshabilite el botón de pagar para evitar que se envíe más de una petición a los servidores de conekta.

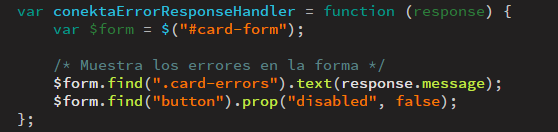
Luego de hacer esto es necesario utilizar una función de la API de conekta, llamada Conekta.token.create que recibe 3 parámetros, el formulario y dos enlazadores de respuesta, uno donde se guardará la respuesta exitosa y otra la respuesta fallida con los errores en caso de existir.

Por ultimo devolvemos false para evitar que estos datos sean enviados a nuestro servidor. En la siguiente figura podemos ver lo anteriormente explicado.



**Figura . Función Conekta Mandar datos Formulario**

Después de esto, conekta procesará y validará los datos enviados y mandará una respuesta que será guardada en alguno de los dos handlers, si conekta nos manda una respuesta fallida entra el conektaErrorResponseHandler y ejecuta una función con un parámetro el cual es la respuesta, dentro de esta función se guarda en una variable el formulario y luego dentro de la clase .card-errors que contiene un div se muestran los errores y se habilita el botón de pagar para que se vuelvan a enviar los datos, está vez de manera correcta. En la siguiente figura podemos apreciar esta función.



**Figura . Función conektaErrorResponseHandler**

Mas sin en cambio, conekta nos envía un success response, entrará en juego el otro handler (conektaSuccessHandler) y ejecutará una función que recibirá un parámetro, en este caso el token creado por conekta, dentro de esta función guardamos el formulario en una variable y le insertamos un nuevo input de tipo oculto con el nombre ‘conektaTokenId’ y le asignamos el id del token que nos envió conekta.

Luego ejecutamos un ajax para mandar dicho token y el nombre del cliente a nuestro servidor por el método POST , no debemos enviar los datos de la tarjeta para evitar problemas con los usuarios y con conekta, con el token y el nombre podemos hacer el cobro de nuestro producto y/o servicio.

Si nuestro servidor ejecutó correctamente el cobro y devolvió una respuesta exitosa en la función success hacemos uso del plugin de notificaciones para mostrar que se ha realizado de manera exitosa la petición y mostramos los datos de la transacción, limpiamos los campos del formulario y por último habilitamos de nuevo el botón para nuevas peticiones.



**Figura. Envio de datos a nuestro servidor Suscripción Conekta**

#### Back-End

En nuestro servidor debemos hacer uso de la API de Conekta para PHP que podemos descargar desde el repositorio oficial de conekta en Github (<https://github.com/conekta/conekta-php>), luego hay que adjuntarlo a la carpeta raíz de nuestro proyecto y hay que agregar el archivo principal de dicha librería ubicada dentro de conekta-php/lib.

Además hay que hacer uso de la llave privada que nos proporciona conekta para poder trabajar de lado del servidor. En la siguiente figura vemos dicha explicación a manera de código.

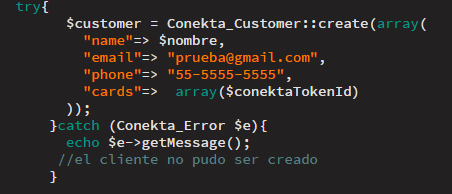


**Figura . Adjunción del archivo general de conekta y de la llave privada**

A continuación hay que hacer uso de la función exctract de PHP para extraer los datos enviados en formato JSON desde el AJAX y guardarlos en variables para su posterior uso.

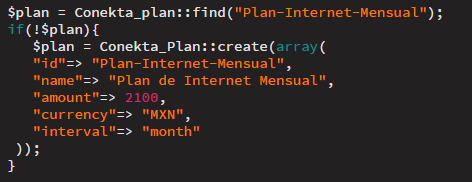
Luego hay que crear un nuevo cliente para guardarlo en nuestra base de datos y poder realizar el cobro de la suscripción, para hacer esto es necesario usar el método create de la clase Conekta\_Customer y dale como datos el nombre del cliente, el correo electrónico, el teléfono y el número de tarjeta (en este caso el token que nos envía conekta).

En la siguiente figura podemos ver la anterior explicación a manera de código dentro de nuestro ejemplo.



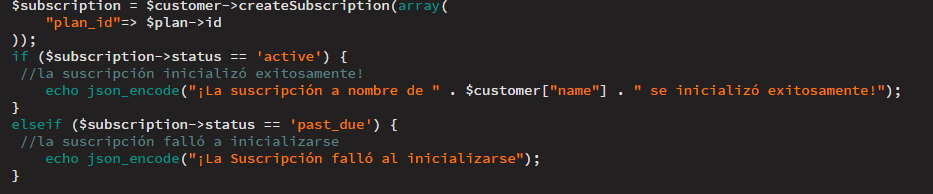
**Figura. Creación Cliente Suscripción**

Después de esto hay que crear un plan de cobro, en este caso se llamará “Plan de Internet Mensual” y tendrá su id, la cantidad a cobrar (2100), el tipo de moneda, en este caso pesos mexicanos y el intervalo de cobro, en este caso se cobrará cada mes. A continuación podemos ver la creción de dicho plan.



**Figura. Creación Plan de Cobro**

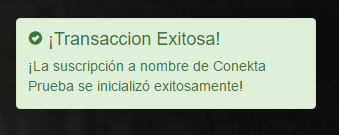
Por último tenemos que crear la suscripción para el cliente que la solicitó, esto lo hacemos con el método createSubscription y le pasamos como parámetro el id del plan para indicar que esta suscrito a ese plan, luego para procesar la respuesta comprobamos la suscripción del cliente y si está activa imprimimos el mensaje de que se la suscripción se ha inicializado correctamente sino mostramos un mensaje de error.



Para realizar la prueba de concepto utilizamos datos de tarjetas de créditos para hacer pruebas que son proporcionados por conekta, dichos datos son los siguientes:

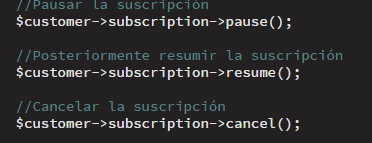
* Número de tarjeta 4242424242424242
* CVC 123
* Mes de Expiración 12
* Año de Expiración 2020
* Tarjetahabiente Conekta Prueba

Al realizar la suscripción el sistema gestiona todos los datos y al ejecutarse de manera correcta la compra el sistema nos muestra la siguiente alerta indicando que la suscripción se ha iniciado correctamente.



**Figura . Suscripción Exitosa**

Podemos pausar, resumir o cancelar la suscripción de algún cliente utilizando los siguientes métodos.



**Figura. Metodos para pausar, resumir o cancelar una suscripción conekta**

### Validaciones

### Notificaciones

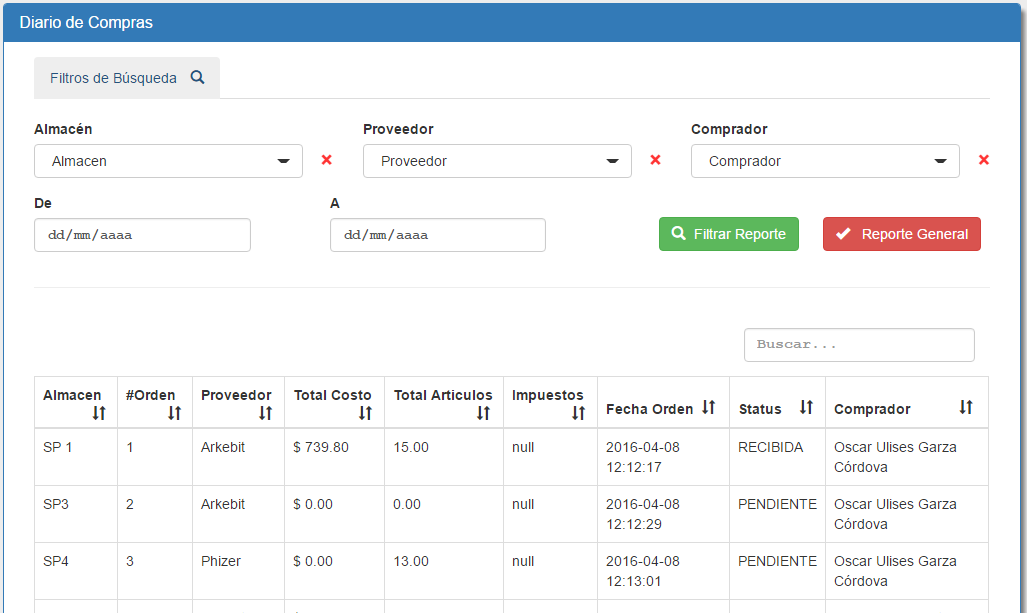
### Consultas en PHP usando Transact-SQL para elaboración de Reportes Ejecutivos

En la actualidad es de suma inportancia dentro de cualquier empresa, sea cual sea su giro el uso de reportes ejecutivos para representar datos estadísticos de la misma de una manera clara y profesional, esto para notar productividad, pérdidas y posibles áreas de oportunidades para las empresas.

Es por eso que cuando se desarrolla un sistema para la gestión empresarial es muy importante incluir módulos para generar reportes ejecutivos de la empresa con datos almacenados en la base de datos, estos reportes pueden ser generados de una manera dinámica, es decir, según el usuario elija el tipo de filtros para el reporte el sistema será capaz de comprobar que filtros fueron seleccionados para crear el reporte solicitado según las necesidades del usuario en turno.

A continuación se hará una breve y clara explicación de como realizar un reporte dinámico usando como ejemplo una aplicación para mostrar el diario de compras de una empresa.

A continuación podemos ver el diseño de dicha aplicación.



**Figura . Diseño Aplicación Diario Compras**

Como podemos ver en la imagen anterior hay un reporte general del diario de compras donde se incluyen datos como el almacén, en número de órden, el proveedor, el costo total, el numero total de articulos, los imputestos, la fecha de la orden, su status y el comprador, además en la parte superior hay una serie de opciones para que el usuario pueda filtrar el reporte por una o más de las opciones, dichas opciones son las siguientes:

* Filtro por Almacén
* Filtro por Proveedor
* Filtro por Comprador
* Filtro por rango de fechas

Como se dijo anteriormente el usuario puede elegir uno o más filtros por ende pueden existir combinaciones en el filtro, el objetivo es hacer una consulta dinámica para realizar dicho reporte.

Para comenzar es necesario llenar las cajas de opciones para los filtros, en este caso la de los almacénes, la de los proveedores y la de los compradores.

Para poder realizar esto necesitaremos usar JQuery y PHP.

En primer lugar tenemos que seleccionar todos los selects de la aplicación con Jquery y ejecutar la función each que nos permitirá ejecutar alguna acción por cada select que encuentre en la aplicación, a continuación sacos los atributos name, nombre y el id y los guardamos en 3 variables, a continuación llenamos un arreglo con 2 objetos, uno con el nombre de acción y el valor de “select” y otro que es el campo y le asignamos la variable etiqueta que tiene como valor el nombre del select, luego hay que ejecutar un ajax para poder llevar a cabo la comunicación con el servidor y poder ejecutar las consultas, le pasamos como parámetros la ruta del script de php, el método de envío, en este caso POST, los datos que en este caso es el arreglo que tiene por nombre datos y por último el tipo de datos, en este caso es JSON.

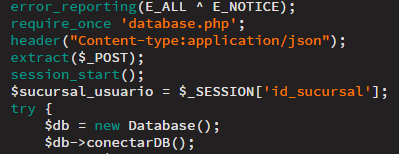
En el status success se comprueba si la respuesta contiene datos y si es verdadero seleccionamos el select por el id y le agregamos en primer lugar el nombre del select como opción nula para que se muestre en primer lugar, luego recorremos la variable donde se guarda la respuesta del servidor y le pasamos los valores value y display para ir diseñando las opciones por cada select, luego hacemos uso de la función selectpicker para usar el diseño de bootstrap-select.

En la siguiente figura podemos ver representada la explicación anterior a manera de código.



**Figura. Llenar las opciones por cada Select Informes**

Ya en el script de PHP indicamos que la aplicación contiene datos de tipo json, usamos la función extract para recibir todos los datos enviados por AJAX, creamos un nuevo objeto de la clase Database que es donde están contenidos los métodos de PDO para ejecutar consultas en Transact-SQL y conectarnos a la base de datos y utilizamos el método conectarDB() para conectar la base de datos y poder ejecutar consultas. En la siguiente figura se representa a manera de código lo anteriormente mencionado.

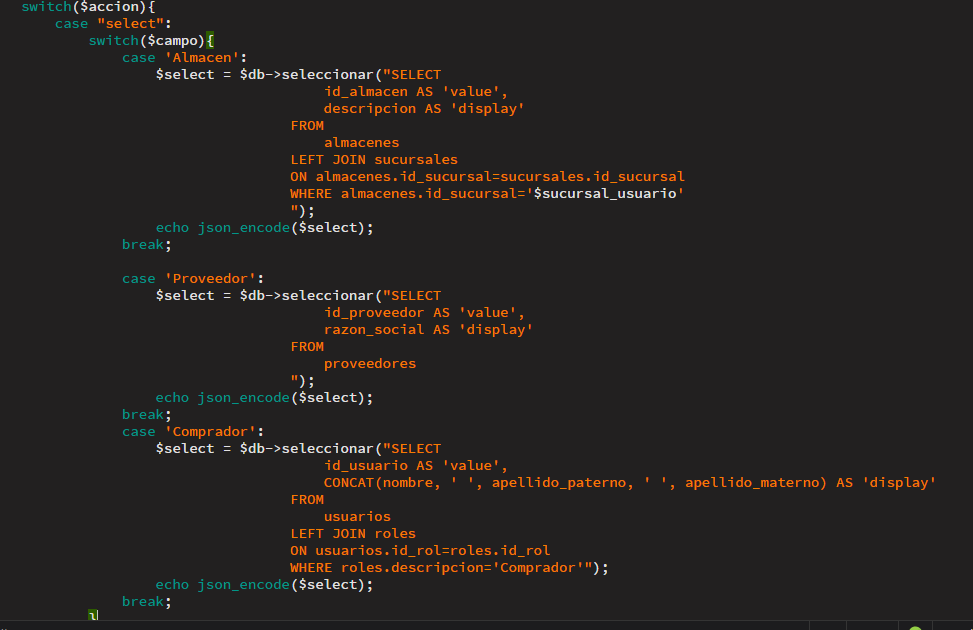


**Figura. Script PHP con la conexión a la BD para reportes ejecutivos**

Si recordamos en nuestro ajax mandamos 2 variables, una llamada acción que tenía como valor “select” y la otra llamada campo que contenia el nombre del campo del select, bueno aquí tenemos que switchear ambas variables, la primera para indicar que entrará al bloque de los selects y no a otros y el segundo para indicar la consulta necesaria para llenar las opciones del select en turno.

En este caso se ejecutan 3 Selects, el primero para seleccionar el id del almacen y su descripción con los alias value y display respectivamente de la sucursal del usuario que este logueado, el segundo para seleccionar el id del proveedor y la razón social del mismo y por último para seleccionar el id y el nombre completo del comprador.

Se hace un echo json\_encode de las 3 consultas para mandar la respuesta y posteriormente en el status success de ajax recorrer dichas respuestas y llenar los selects con las opciones correspondientes. A continuación podemos ver el script con las 3 consultas en Transact-SQL

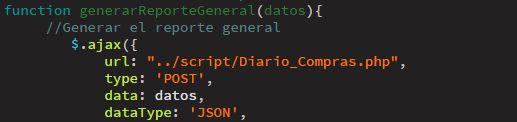


**Consultas para llenar los selects**

A continuación se optó por crear una función para generar los reportes, esto para ahorrar código y simplificarlo, dicha función recibe un parámetro llamado datos. Dentro de la función hay un ajax con los siguientes parámetros:

* url: “../script/Diario\_Compras.php”, este parámetro indica la ruta del script de php a donde se enviará la información
* type: “POST”, este parámetro indica que los datos se enviarán al sevidor por medio del método POST.
* Data: datos, indica que se pasará el parámetro de la función como los datos.
* dataType:”JSON”, este parámetro indica que se trabajaran con datos de tipo JSON.

En la siguiente figura se puede apreciar la función con los parámetros



**Figura. Función generarReporteGeneral**

A continuación hay que situarnos en el evento click del botón que tiene como id generar\_reporte para ejecutar una función al momento de que el usuario de click al botón.

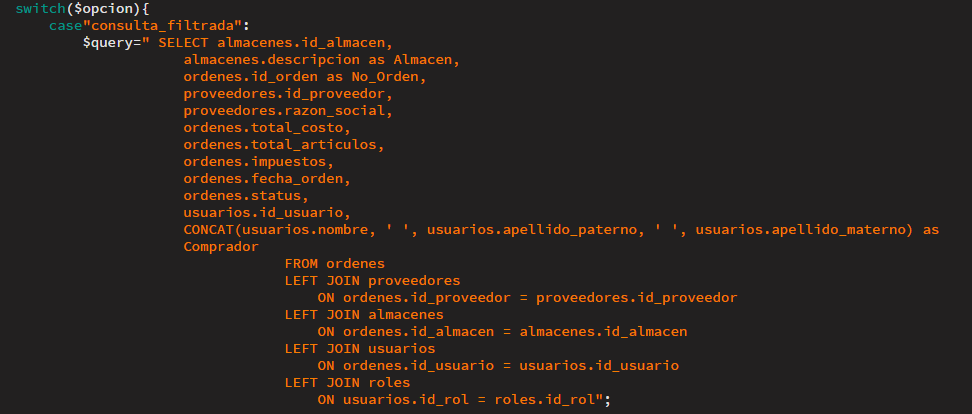
Cuando esto suceda lo primero que se hará es capturar los datos cada uno de los selects y guardarlos en variables representativas para evitar confusiones en PHP, estas variables las guardamos en un arreglo llamado datos a manera de objetos json, además guardamos un objeto con el nombre ‘opcion’ y con el valor ‘consulta\_filtrada’, después hacemos de la función que creamos con anterioridad llamada generarReporteGeneral y le pasamos como parámetro el arreglo con todos los datos para que este a tráves de AJAX mande estos datos al servidor PHP.



**Figura. Evento Click Generar\_Reporte**

A continuación hay que switchear la variable opción, en este caso el valor es “consulta\_filtrada”, dentro de este case lo primero que tenemos que hacer es guardar una consulta general para posteriormente de manera dinámica filtrarla con los filtros que haya seleccionado el usuario, esta consulta seleccionará el id del almacen, su descripcion, el id de la orden, el id del proveedor, su razón social, el costo total, el total de articulos de la orden, los impuestos de la orden, la fecha de la orden, el status de la orden, el id del comprador y el nombre del mismo; estos datos se sacarán de las tablas ordenes, proveedores, almacenes, usuarios y roles.

A continuación podemos ver está consulta a manera de código Transact-SQL.

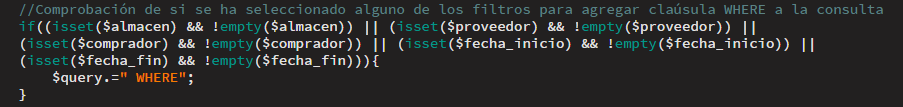


**Figura. Consulta General en Transact-SQL**

A continuación hay que jugar entre las posibles combinaciones de filtros hechos por el cliente, para hacer la explicación un poco más entendible se separará la explicación por campo, es decir, en los siguientes puntos:

* Almacen
* Proveedor
* Comprador
* Fechas

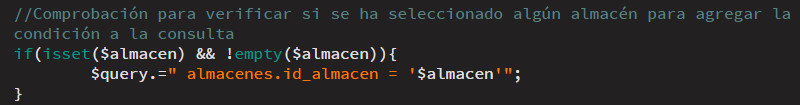
Cabe recalcar que lo primero que debemos comprobar es si se seleccionó alguno de los filtros, para poder realizar dicha comprobación utilizamos las funciones isset para comprobar si se activó el filtro y !empty para indicar si el filtro no está vacio y los operadores lógicos AND para concatenar alguna condición y || para indicar que si una no se cumple puede cumplirse otra condición independiente a la primera, en este caso le decimos que si seleccionó algún almacén o algún proveedor o algún comprador o una de las 2 fechas le vamos a agregar a la consulta general la claúsula WHERE para realizar filtros posteriores por campo o campos. A continuación podemos ver dicha condición.

**Figura Condición para agregar claúsula WHERE al reporte**

#### Filtro Almacén

Dentro de este filtro se tiene que comprobar si se elijió algún almacén, para poder comprobar esto es necesario utilizar la función isset y la función empty negada para saber si se eligió un almacén, si esto se cumple se concatena el string “almacenes.id\_almacen = ‘$almacen’.

A continuación podemos ver la condición necesaria para comprobar esto.



**Figura. Condición Almacén**

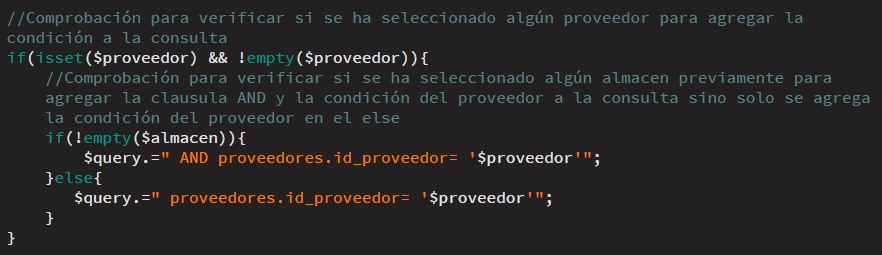
#### Filtro Proveedor

Dentro de este filtro se tiene que comprobar si se elijió algún proveedor, para poder comprobar esto es necesario utilizar la función isset y la función empty negada para saber si se eligió un proveedor.

Si esto se cumple entra a una nueva condición para verificar si se eligió un almacén con anterioridad, si esta condición se cumple se entiende que ya no sólo sería una condición sino 2, una para el almacén y otra para el proveedor y necesitan concatenarse con el operador AND, es por eso que a la consulta se le concatena la condición del proveedor más el operador AND.

Sin embargo si no se seleccionó ningún almacén pero se seleccionó un proveedor simplemente se concatena la el string “proveedorees.id\_proveedor = ‘$proveedor’.

A continuación podemos ver las condiciones necesarias para comprobar esto.



**Figura. Condición Proveedor**

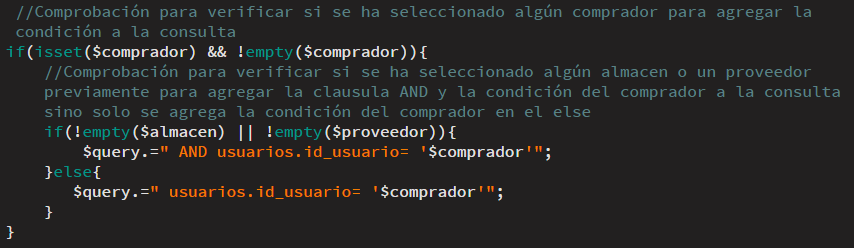
#### Filtro Comprador

Dentro de este filtro se tiene que comprobar si se elijió algún comprador en la aplicación, para poder comprobar esto es necesario utilizar la función isset y la función empty negada para saber si se eligió un comprador.

Si esto se cumple entra a una nueva condición para verificar si se eligió un almacén o un proveedor con anterioridad, si esta condición se cumple se entiende que ya no sólo sería una condición sino 2 o hasta 3, una para el almacén y/o una para el proveedor y otra para el comprador y necesitan concatenarse con el operador AND, es por eso que a la consulta se le concatena la condición del comprador más el operador AND.

Sin embargo, si no se seleccionó ningún almacén o ningún proveedor pero se seleccionó un comprador simplemente se concatena la el string “usuarios.id\_usuario = $comprador’.

A continuación podemos ver las condiciones necesarias para comprobar esto.



**Figura. Filtro Comprador**

#### Filtro Fechas

Es importante recalcar que el filtro por fecha no sólo actua como un intervalo entre la fecha de inicio y la fecha de fin sino que, por ejemplo, si el usuario sólo elige la fecha de inicio se filtrarán todos los registros que se encuentren a partir de la fecha elegida, si sólo se elige la fecha de fin sólo se filtrarán los registros hasta la fecha solicitada y por último si se eligen ambas fechas se hace un filtro de registros para sacar los que se encuentren en el rango de las 2 fechas.

Lo primero que se tiene que comprobar si se elijió alguna de las 2 fechas en la aplicación, para poder comprobar esto es necesario utilizar la función isset y la función empty negada para saber si se eligió alguna de las 2 fechas.

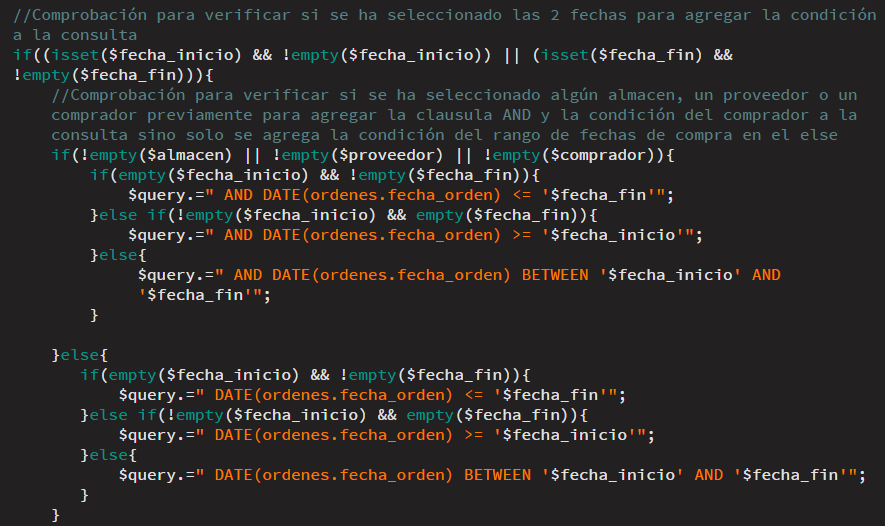
Si esto se cumple entra a una nueva condición para verificar si se eligió un almacén y/o un proveedor y/o un comprador con anterioridad, si esta condición se cumple se entiende que ya no sólo sería una condición sino 2 o hasta 4, una para el almacén y/o una para el proveedor, otra para el comprador y otras para las fechas, y necesitan concatenarse con el operador AND.

Posteriormente si la condición anterior es verdadera se hace una nueva comprobación que pregunta si se eligió solamente la fecha de fin y no se eligió la fecha de inicio, si esta se cumple se agrega la condición utilizando el operador <= para decirle a la consulta que filtre los resultados que se encuentren hasta la fecha solicitada, sino se cumple entra a una nueva condición que comprueba si se eligió solamente la fecha de inicio, si esta se cumple se agrega la condición utilizando el operador >= para decirle a la consulta que filtre los resultados que se encuentren a partir de la fecha de inicio, si ninguna de las anteriores condiciones se cumple por lógica quiere decir que se eligieron ambas fechas y entra al else donde se ejecuta el filtro por rango de fechas utilizando la función BETWEEN.

Cabe recalcar que como se encuentra dentro del primer if que comprueba si se eligió un almacén y/o un proveedor y/o un comprador con anterioridad, a la cosulta se le agrega el operado AND para concatenar la condición en turno con alguna condición previa.

Ahora bien si no se seleccionó ningún almacén o ningún proveedor o ningún provedoor pero se seleccionó alguna fecha entra al else del primer if donde se hacen las mismas comprobaciones para saber cual de las fechas fue seleccionada para concatenar simplementa la condición sin el operador AND.

A continuación podemos ver las condiciones necesarias para comprobar esto.



**Figura . Filtros por Fecha**

Al terminar el bloque de condiciones if para llenar los filtros de la consulta hacemos uso del métdo seleccionar de la clase PDO y le pasamos como parámetro la variable $query que contiene la consulta dinámica con los filtros solicitados y por último mandamos la respuesta a tráves de la función echo json\_encode().



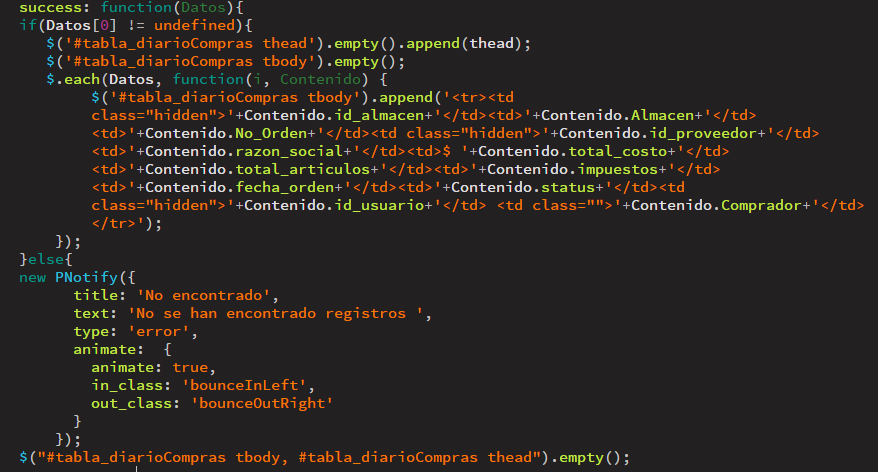
#### Impresión de la tabla con el reporte filtrado

Ahora bien, luego de mandar la respuesta desde el servidor, es necesario volver al código Jquery y centrarnos en el success de la función que utilizamos para enviar los datos de los filtros al servidor, está función es llamada generarReporteGeneral,

Dentro de la función del success tenemos que comprobar que la respuesta contenga algún registro, si es así, limpiamos el thead de la tabla y le agregamos el encabezado, luego limpiamos el tbody de la tabla para luego recorrer la respuesta del servidor que contiene los datos del reporte filtrado, utilizamos el parámetro Contenido para acceder a los items de la consulta que se toman como nombre el alias que se les da en la consulta de Transact-SQL, se van imprimiendo a manera de HTML los renglones y las celdas de la tabla haciendo referencia a todos y cada uno de los campos de la consulta para que con cada iteración se vayan dibujando los registros de la consulta en la tabla.

Si la respuesta no contiene ningún dato entra a la sentencia else donde se muestra una notificación creada con el plug-in Pnotify con el mensaje de “No se han encontrado registros” y se limpia el encabezado y el cuerpo de la tabla.

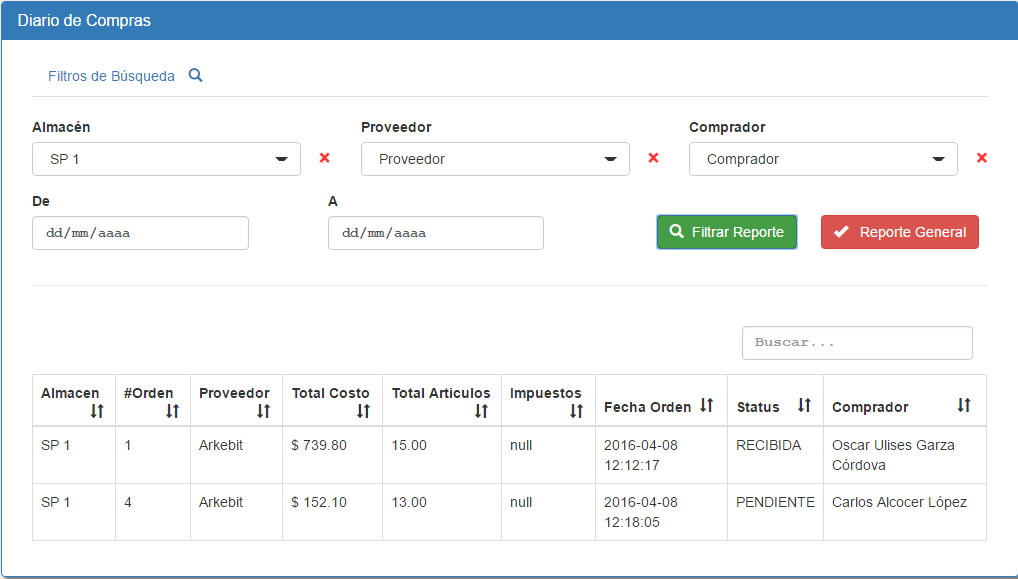
En la siguiente figura podemos ver representada la anterior explicación a manera de código en un editor de textos.



**Figura. Impresión Reporte Ejecutivo Jquery**

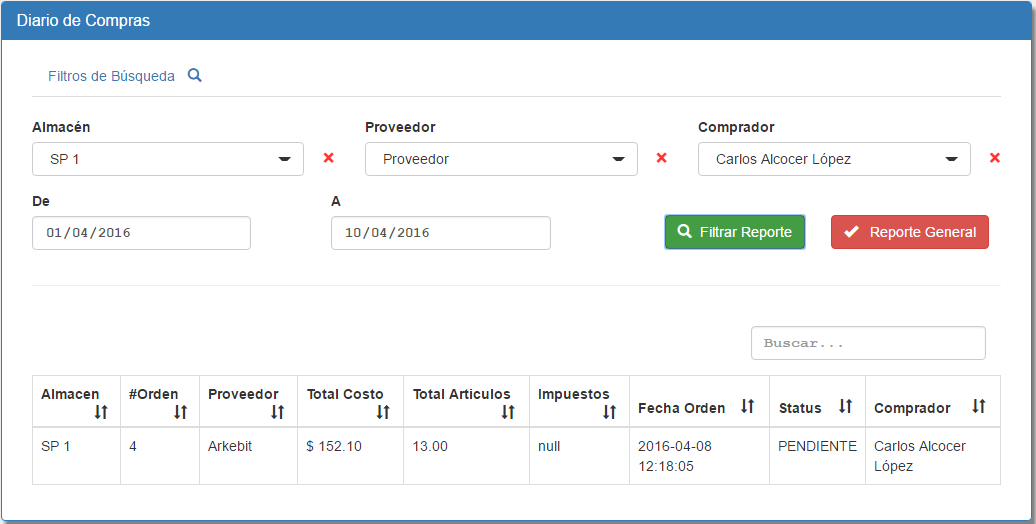
Para realizar la prueba de concepto utilizaremos 2 filtros, uno por almacén y el otro concatenando el almacén con el comprador y las fechas.

Cómo podemos ver en la siguiente figura, se seleccionó el almacén SP 1 para realizar el filtro por almacén y la consulta nos generó la tabla sólo con las ordenes de compra del almacén seleccionado.



**Figura. Prueba de Concepto Filtro Almacén**

Cómo podemos ver en la siguiente figura, se seleccionó el almacén SP 1 para realizar el filtro por almacén, además se seleccionó el comprador Carlos Alcocer López para realizar un filtro por comprador tambíen y se seleccionó un rango de fechas del 1 de Abril del 2016 al 10 de Abril del 2016 para filtrar la consulta en un intervalo de fechas y la consulta nos generó la tabla sólo con las ordenes de compra con los filtros seleccionados.



**Figura. Prueba de Concepto Filtro Múltiple**

## Conclusiones

## Bibliografía

## Anexos