Generalidades del sistema JCFW

El sistema de entorno de trabajo se ha desarrollado con base al historial de diferentes proyectos desarrollados en los últimos años, y a los proyectos iniciales que se han desarrollado en el trabajo del año 2012.

Se ha tomado los conocimientos de manejo de FrameWorks externos como Cakephp de la forma más sencilla y haciendo lo necesario para garantizar un nivel de seguridad aceptable.

Se han creado un entorno basicosistema que maneja una sola base de datos, con n perfiles de usuario, y n idiomas, para un solo país, o sin importar el país.

Arquitectura

Se ha manejado la arquitectura MVC (modelo, vista, controlador), en cuanto a la programación en el servidor, también se ha tenido en cuenta la visualización en el cliente utilizando HTML5, CS3 y Java Script. Apartando los conceptos de HTML5 y CS3, se han tenido en cuenta los siguientes FrameWorks de tipo cliente como son Font Awesome y Bootstraps 4, que es un framework bastante completo para asumir fácil, profesional y profundamente los estilos de forma general para todos los elementos que se pueden llegar a necesitar en cualquier proyecto, además que se integra perfectamente al FrameWork JQuery que es también un entorno de trabajo del lenguaje Java Script, logrando una integración general de todos los elementos posibles, que manejen la mayor cantidad de estándares para que funcionen en forma correcta en la mayor cantidad de dispositivos. Tambien se le adapto una db con todas las palabras claves que trae schemas.org y genera unos códigos básicos de microdatos.

Semántica Generalidades

Teniendo en cuenta los estándares se ha tomado una filosofía asumida en el FrameWork Cakephp, que tiene varias filosofías a la hora de asumir los temas a elección de los desarrolladores como son los nombres de tablas y campos.

Como empezar

Lo primero que hay que hacer en crear la base de datos, la semántica utilizada es la misma de Cake, la idea es que las tablas siempre sean en plural por ejemplo usuarios, productos.

Los campos en cada tabla, siempre debe tener un id que se llame id, de ninguna manera será id\_usuarios, ni nada por el estilo, solo id.

Se ha dispuesto de algunos campos fijos que deberían ir en todas las tablas aparte del id, estos campos son creado y editado. Que son de tipo datetime en formato “AAAA-MM-DD HH:MM:SS”, creado es NOT NULL y editado es NULL, allí solo se guardan las fechas y horas de su creación y última modificación, estos campos se alimentan de forma automática.

Se hace un manejo de algunos campos especiales, los campos son terminados en una palabra clave que ayuda mucho en la creación de archivos.

Para los campos que son relacionales se debe manejar con el nombre de la tabla en singular y adicionando al final \_id, por ejemplo usuario\_id producto\_id, para que el sistema asuma la relación es necesario que este campo sea índice en la tabla.

Para los campos que se quiera incluir archivos en la base de datos, se manejan 4 campos, estos campos se definen con el nombre del campo en singular y las siguientes terminaciones \_file de tipo longblob, que contendrá el archivo en sí, el mismo nombre del campo que termine en \_file\_name de tipo varchar en donde va contener el nombre original del archivo, el mismo nombre del campo que termine en \_file\_type de tipo varchar que contendrá el MIME-Type del archivo, el mismo nombre del archivo que termine en \_file\_size de tipo bigint que contendrá el tamaño del archivo en bytes.

Los campos deben ser en singular de la forma más corta y descriptiva tratando de no dejar a la medida de lo posible mas de 2 palabras, ejemplo código\_respuesta, tipo\_pago.

Los campos en que se planee manejar información de tipo JSON, serán campos en plural, ej. parámetros, etc.

Los campos de tipos date, se manejara con un datepicker automáticamente en los formularios generados.

Los campos de tipos tinyint, se asumirán que son de un tipo de opción de un máximo de 5 opciones, por ejemplo, tipo de documento, en el que van las opciones, cc, ce, pasaporte, nit. Estos datos serán iguales e inmodificables en todo lugar, y no vale la pena manejar una tabla que se llame tipo de documentos porque no valdría la pena, además que el manejo de la información incrementaría el proceso adicional innecesario. Otro ejemplo puede ser tipo de entorno, si son máximo 5 y no se piense alterar, por ejemplo: estudiante, profesor, administrativo, y académico. Estos entornos o perfiles de una institución educativa, es casi imposible que cambien.

Hay un campo manejado de forma automática y este sería el campo sexo, que lo toma automáticamente como hombre y mujer.

Configuración

La configuración se maneja en la carpeta admin/inc/, y los archivos de configuración son: config.php, config\_raiz.php, config\_perfil.php, y el archivo db.php

El proyecto base esta generalmente montado con un perfil de usuarios que es una tabla que viene en la db ejemplo, este usuario es el que se registra, y el perfil del administrador que es manejado de una tabla especial admins.

En los archivos config.php, config\_raiz.php y config\_perfil.php se manejan el mismo tipo de variables, básicamente se configura el dominio y las carpetas donde va montado el proyecto.

La constante más importante es PATO que quiere decir el Path cero, ósea donde va ir el raíz del proyecto, si el lugar donde se monta el proyecto es la carpeta raíz, la constante PATO debe configurarse con un slash / y no más. De lo contrario debe tener el nombre de la carpeta del proyecto con los respectivos slash al principio, en medio y al final, ejemplo /prueba/febrero/.

Otra constante importante es PATU que solo indica la raíz del proyecto sin carpetas de perfil.

Otra que se debe explicar es la de IDIOMA, esta define el idioma default del sistema y hace referencia al campo en la tabla idioma de la base de datos, la idea es manejar 4 letras, las dos primeras es el idioma y las otras dos son del país; si requiere adicionar un idioma debe crear el campo en la tabla idioma en minúsculas preferiblemente, por ej: ptbr, que seria para portugués en el país Brasil, hay que manipular la función para que google traduzca y adicione automáticamente en la base de datos en el archivo admin/inc/funciones.php y para cambiar el idioma del sistema se manipula la variable de sesión llamada: ‘JC\_Idioma’.

La constante PPUBLICO, define el perfil que se va a manejar como perfil público (publico con entrada de usuario y contraseña), por ejemplo usuarios o clientes, esto se utiliza básicamente en la instalación, ya que no se permite crear como perfil adicional, debe ser con el mismo nombre de la tabla de usuarios con los campos mínimos para la entrada, definidos más adelante, en la instalación. En la db de ejemplo viene como la tabla usuarios.

Si el proyecto está bien configurado, el sistema ignora estas subcarpetas donde está el proyecto y funcionan perfectamente las constantes MODULO, PROCESO y todas las variables que vengan separadas con slash.

La constante TEMP es una carpeta en la que guardamos temporalmente todo lo que sube y que por la misma estructura de URLs amigables no es accesible desde la web, a esta carpeta se le deben verificar los permisos para las posibles subidas de archivos.

De resto son constantes que se van a manejar con referencia a sitios como facebook, addthis, paypal, pagosonline y la información de un correo para enviar posibles mails.

Instalación

Para instalar el sistema, primero se debe dejar configurada la base de datos, y configurados los archivos de configuración, y entramos al sitio localhost/proyecto/install.php y se despliega un formulario donde lee toda la base de datos y uno le da indicaciones de lo que debe crear.

Si ha configurado bien el archivo config.php, el sistema tendrá una tabla de perfiles de usuario publico que son los que trabajan en el FRONTEND, si usted quiere crear entornos paralelos como por ejemplo distribuidores, operadores, u otro, debe haber creado ya una tabla con el respectivo nombre, manteniendo la semántica, ósea en plural.

Esta tabla se puede manejar con un entorno paralelo que el sistema crea automáticamente dentro de una carpeta llamada con el mismo nombre de la tabla, y permite el respectivo acceso, es necesario que esta tabla de perfil tenga los campos mínimos como son: id, nombre, apellido, email, clave y estado. Para que se creen estos entornos se chulea en el checkbox que hay al lado del nombre de la tabla.

Los perfiles de usuario que está configurado en el config.php con la constante PPUBLICO y perfil de los administradores no hay que configurarlos, ya que esto se hace automáticamente.

Cada tabla tiene un formato en el que uno le indica que campos se muestran en el index, ósea en el listado, que datos se generan en el pantallazo de ver, que datos se necesitan para el formulario de adición, que datos se muestran en el formulario de edición, si los registros van a poder borrarse, si van a parecer en el filtro de la pagina de listado para mejorar los resultados de búsqueda, dos campos relacionales para que lea de otras tablas, y por último, las posibles opciones de campo que normalmente son cuando se ponen campos de entero pequeño de opciones fijas.

Antes de empezar a llenar el formato de instalación, es buena idea revisar que los campos relacionados estén bien configurados en la base de datos, de lo contrario hay que decirle a la base de datos que el campo debe ser un index y recargar la pagina de instalación.

Se llenan todos los datos que un requiera y se le da la orden de weberiar, esto genera todos los archivos que manejan el sitio completo. Ya solo es copiar y pegar y acomodar posibles diseños.

Las restricciones de datos requeridos, se toman automáticamente de los campos que se hayan definido en la base de datos como NULL o no NULL, la verificación de las posibles casillas de email o de número se verifican mediante HTML5 o mediante Java Script dependiendo de la versión del navegador.

URL amigables

El sistema tiene un manejo de URLs amigable, que funciona de acuerdo a las tablas de la base de datos y se llama en la programación con la constante MODULO, y hay una segunda constante que se llama PROCESO que es el segundo dato del URL, de ahí en adelante todos los datos que llegan serán tomados como un arreglo, si no está definido ni el modulo ni el proceso, el sistema lo toma por defecto como index, ejemplo:

Si la dirección URL es: [www.dominio.com/usuarios/ver/5/](http://www.dominio.com/usuarios/ver/5/)

En el ejemplo se cargara el modulo usuarios en el proceso ver y el id del usuario debería ser el 5.

Adicional

Para quitar funcionalidades de algún perfil, no elimine la funcionalidad del modelo, es mejor borrar la función pública en cada controlador, también, obvio, por presentación, quitarla de las vistas.

Archivo de FUNCIONES.php

Este archivo está ubicado en admin/inc y contiene una serie de funciones básicas que normalmente se requieren en cualquier momento de toda la programación, estas funciones se describirán a continuación:

\_\_() Esta es la función en la que se debería ingresar todos los textos que van en cualquier parte del sitio, esta función recibe una cadena, normalmente alfabética, que se busca en la tabla de idiomas el label de llegada y si no lo encuentra lo agrega a la base de datos, no sin antes haber buscado la respectiva traducción al Ingles, para meter el registro completo.

mayus() Convierte la cadena de llegada en mayúsculas, teniendo en cuenta los caracteres del español.

edad() Esta función recibe una fecha (de nacimiento de una persona) en formato ‘aaaa-mm-dd’ y devuelve la edad actual.

limpia() Esta función limpia cualquier variable de posibles scripts maliciosos.

limpia2() Esta función limpia arrays de posibles scripts maliciosos.

txtmes() Esta función recibe un numero de 1 a 12 y devuelve el mes en español.

iok() Retorna un código html de imagen con la imagen ok.

ierror() Retorna un código html de imagen con la imagen error.

iadmira() Retorna un código html de imagen con la imagen admiración.

iinterro() Retorna un código html de imagen con la imagen interrogación.

veraut() Verifica si estamos logueados de lo contrario se asegura de sacarme del sistema si mandamos el parámetro 2 que es default, pos retorna un script de JavaScript, si mandamos un 1, nos envía un encabezado de redirección.

ifaut() Verifica si estamos logueados y retorna booleano verdadero o falso.

verificaquetengamospais() Es una función para el modelo de portal hecho para saber en cual base de datos vamos a trabajar.

imgresize() Redimensiona una imagen manteniendo proporcionalidad dentro de un tamaño definido y con opción de copiar la imagen generada en una ubicación diferente a la del origen.

Archivo de JCFW.JS

Este archivo está ubicado en /js y contiene una serie de funciones básicas que normalmente se requieren en cualquier momento de toda la programación, estas funciones se describirán a continuación:

jf\_fin() Esta función está escrita para que después de cargar todo ejecute la librería placeholder para asegurarnos que los placeholder funcionen en todos los exploradores, placeholder es la marca de agua que hay en los input antes de escribir algo, también se ejecuta el script que genera los tooltips de bootstrap.

ultimodia() Esta función recibe año y mes y retorna el numero del ultimo día.

MM\_preloadImages() Esta función es la predefinida de Macromedia para hacer precargas de imágenes que posiblemente vamos a utilizar más adelante, como se ejecuta en background es una muy buena ayuda a la hora de llamar aquellas imágenes por que ya se encuentran en el cache.

notxt() Esta función se utiliza para restringir la entrada de texto en cualquier input, esta función se creó para exploradores que no soportaban html5 y se utiliza asi: onKeyPress="return notxt(event)"

numfor() Esta función da formato a los números poniéndoles el punto de mil y millón, etc.

microtime() Genera una cadena de números con el tiempo actual en segundos.

valmail() Valida si lo que recibe es una dirección de correo valida.

abrirpopoup() no se utiliza en si sino se utiliza con las otras funciones.

cerrarpopup() Se puede utilizar en cualquier parte para cerrar el popup del sistema.

olvidoclc() Función del sistema.

centrar() Función del sistema.

popup() Recibe una url que se carga en el popup.

alerta1() Es como un alert de JS pero con diseño del sistema.

confirma1() Es como un comfirm de JS pero aplicado al sistema, recibe 4 parámetros, el primero es el mensaje o pregunta a tener en cuenta, el segundo es el mensaje que sale cuando la respuesta es afirmativa, ej: si, ejecutar, ok, etc. El tercero es la función en JS que debe ejecutar si hace click afirmativo, y el cuarto es un valor que envía a la función de JS del parámetro 3; la opción negativa lo que hace es cerrar el popup de confirmación.

jcpop() Es igual que el alert1 pero sin botón de aceptar.

verimagen() Funciona como un slice de imagen para verla grande, se envía 2 parámetros, la url de la imagen y el titulo del popup.

textarea() Esta función lo que hace es limitar la entrada de caracteres a una entrada de tipo texto, recibe el id del input, lee automáticamente el atributo del textarea llamado maxlength para conocer el máximo, y en un div con el mismo nombre del textarea mas la terminación \_kar se actualiza con el numero de caracteres que se han escrito.

print\_r() Esta función es igual a la de php: print\_r. Y sirve para imprimir los datos de una variable de tipo array o de tipo obj, de las que estemos utilizando en JavaScript, esta función es solo para cuando estamos desarrollando.

http://wsp.sicom.gov.co:83/sicomdata/DepartamentosBasico/

http://wsp.sicom.gov.co:83/sicomdata/DepartamentosBasico('11')/Municipios/