**RECETARIO LAROUSSE**



MANUAL TÉCNICO PARA EL PROYECYO WEB DE RECETARIO LAROUSSE

Fecha: Mayo 2013.

|  |  |  |
| --- | --- | --- |
| Contribuyentes | | |
| Aguilar Pardo Gersaín | Desarrollo | Tecnológico de Apatzingán. |
| Cornejo Méndez Josué | Diseño | Tecnológico de Morelia. |
| García Solorio Isidro Adán | Desarrollo | Tecnológico de Morelia. |



INDICE

[**1.- INTRODUCCIÓN** 3](#_Toc357498993)

[**2.- FUNCIONAMIENTO** 4](#_Toc357498994)

[**2.1.- CARPETA PRINCIPAL** 5](#_Toc357498995)

[**2.2.- CARPETAS DEL SISTEMA** 5](#_Toc357498996)

[**2.3.- PRIMER USO** 5](#_Toc357498997)

[**2.4.- 404 FORBIDDEN** 5](#_Toc357498998)

[**2.5.- APLICACIONES** 6](#_Toc357498999)

[**2.5.1- AGREGAR APLICACIONES** 6](#_Toc357499000)

[**2.5.2- VER APLICACIONES** 6](#_Toc357499001)

[**2.5.3.- EDITAR APLICACIONES** 6](#_Toc357499002)

[**2.5.4.- ELIMINAR APLICACIONES** 6](#_Toc357499003)

[**2.6.- RECETAS** 6](#_Toc357499004)

[**2.6.1.- AGREGAR RECETAS** 6](#_Toc357499005)

[**2.6.2.- VER Y EDITAR RECETAS** 7](#_Toc357499006)

[**2.6.3.- ELIMINAR RECETAS** 7](#_Toc357499007)

[**2.7.- CATEGORIAS** 7](#_Toc357499008)

[**2.7.1.- AGREGAR CATEGORIAS** 7](#_Toc357499009)

[**2.7.2.- VER Y EDITAR CATEGORIAS** 7](#_Toc357499010)

[**2.7.4.- ELIMINAR CATEGORIAS** 8](#_Toc357499011)

[**2.8.- GLOSARIO** 8](#_Toc357499012)

[**2.8.1.- AGREGAR GLOSARIO** 8](#_Toc357499013)

[**2.8.2.- VER GLOSARIO** 8](#_Toc357499014)

[**2.8.3.- EDITAR GLOSARIO** 8](#_Toc357499015)

[**2.8.4.- ELIMINAR GLOSARIO** 8](#_Toc357499016)

[**2.9.- VIDEO** 8](#_Toc357499017)

[**2.9.1.- AGREGAR VIDEO** 8](#_Toc357499018)

[**2.9.2.- VER VIDEO** 8](#_Toc357499019)

[**2.9.3.- EDITAR VIDEO** 8](#_Toc357499020)

[**2.9.4.- ELIMINAR VIDEO** 8](#_Toc357499021)

[**2.10.- RECETAS COMPLEMENTARIAS** 8](#_Toc357499022)

[**2.10.1.- AGREGAR COMPLEMENTARIA** 8](#_Toc357499023)

[**2.10.2.- VER COMPLEMENTARIA** 8](#_Toc357499024)

[**2.10.3.- EDITAR COMPLEMENTARIA** 8](#_Toc357499025)

[**2.10.4.- ELIMINAR COMPLEMENTARIA** 8](#_Toc357499026)

[**3.- EXPORTAR** 8](#_Toc357499027)

# **1.- INTRODUCCIÓN**

El recetario de Larousse es un compendio de recetas, las recetas dependen de una aplicación la cual determina el nombre de la aplicación que contendrá un sinfín de recetas, dichas recetas están divididas por categorías, las cuales cuentan con varias características como lo son tiempo de cocción, tiempo de preparación, costo, foto y otras características.

El sistema desarrollado de manera web apoya a la captura del contenido de cada aplicación, recetas, categorías, recetas complementarias, glosarios y videos que contendrán cada receta correspondientemente. Cabe mencionar que todos los campos que administra la aplicación web son meramente texto, los cuales solo sirven de referencia a la aplicación móvil, la cual toman esos datos y recopila la información necesaria para crear los recursos de la aplicación.

El sistema web es capaz de generar un archivo de base de datos con el contenido correspondiente a cada aplicación (.db), el cual se complementa con la aplicación móvil, para crear la aplicación correspondiente. Para desarrollar este sistema web se utilizó un framework de desarrollo de PHP llamado CodeIgniter, el cual se basa en el Modelo-Vista-Controlador, lo cual sirve para dividir el desarrollo, en el modelo se encuentra todo lo referente a las consultas de la base de datos, en la vista se encuentra todo el código HTML y en el controlador se encuentra todo lo referente a la lógica del desarrollador y el funcionamiento de la página, CodeIgniter es un framework para desarrollo de aplicaciones - un conjunto de herramientas - para gente que construye sitios web usando PHP. Su objetivo es permitirle desarrollar proyectos mucho más rápido que lo que podría hacer si escribiera el código desde cero, proveyéndole un rico conjunto de bibliotecas para tareas comunes, así como y una interfaz sencilla y una estructura lógica para acceder a esas bibliotecas. CodeIgniter le permite enfocarse creativamente en su proyecto al minimizar la cantidad de código necesaria para una tarea dada.

El framework CodeIgniter utiliza programación orientada a objetos las cuales, los controladores contienen las clases principales y los métodos a los que accedes mediante este sistema en realidad son los que utilizas en la URL.

# **2.- FUNCIONAMIENTO**

El sistema web accede por primea vez a un archivo ubicado en recetario-larousse/iOS/trunk/web/application/config/ llamado routes.php el cual contiene el controlador por default al que llamara la primera vez que este se ejecute, determinado por la sentencia $route['default\_controller'] = apps/index'; lo cual siginifica que accedera a la clase apps dado por el archivo apps.php y dentro de este llamará al método index, el cual cargará archivos como la cabecera del HTML y una página HTML de la carpeta views/pages/index.php, el cual contiene todas las aplicaciones existentes en este sistema, para pasarle los datos de nuestro controlador a la vista se realiza mediante nuestro arreglo $data[] el cual al crearle una posición en este arreglo nos creara una variable que podemos utilizar sin ningún inconveniente en la vista especificada.

## **2.1.- CARPETA PRINCIPAL**

El sistema web funciona mediante una estructura de carpetas que se detallaran a continuación:

**recetario-larousse/iOS/trunk/web/**

## **2.2.- CARPETAS DEL SISTEMA**

Dentro de esta carpeta se encuentran alojados los archivos que sirven para que el sistema web funcione correctamente.

**application**: Esta carpeta la cual contiene todos los archivos principales del proyecto web.

**css:** Esta carpeta contiene todos los archivos referentes al diseño de las páginas web del proyecto.

**js:** Contiene los archivos de scripts como jQuery, jQuery UI y tiniMCE el cual es un editor de texto WISIWYG.

**Manual:** La cual contiene los manuales técnicos y de usuario del sistema.

**resources:** La cual contiene todos los archivos referentes scripts, recursos de bases de datos necesarios para la funcionalidad del sistema web y la base de datos necesaria para el sistema móvil, las carpetas contenidas dentro de esta son:

* **css:** la cual contiene archivos de css para realizar funcionalidades del el editor de colores dinámico.
* **DB:** La cual contiene la base de datos necesaria para el sistema móvil (**larousse.db**).
* **images:** la cual contiene imágenes para el editor de colores de jQueri UI.
* **img:** La cual contiene las imágenes como el logo y el icono que se muestra en las páginas.
* **mysql:** La cual contiene la base de datos necesaria para el funcionamiento del sistema web.
* **sqlite-browser:** La cual contiene un editor y visor de contenido de sqlite para Windows.

**src:** Carpeta que propone el framework para guardar los recursos del sistema.

**system:** La cual contiene todos los archivos que necesita el framework para funcionar.

## **2.3.- PRIMER USO**

Para utilizar el sistema por primera vez se tiene que crear la base de datos en el gestor de bases de datos de mysql, e importar la base de datos contenida en la carpeta recetario-larousse/iOS/trunk/web/resources/ mySQL/. Llamada recipes.sql, esta base de datos crea la estructura necesaria para que funcione el sistema web.

## **2.4.- 404 FORBIDDEN**

Este error es común si no tienes activado algunos módulos de tu gestos de PHP, para esto debes activar algunos módulos, los cuales son por el lado de PHP se llama php\_gettext, el cual proporciona la posibilidad de omitir el index.php dentro del proyecto, y un módulo de Apache llamado rewrite\_module, el cual provee la posibilidad de utilizar un archivo .htacces para la utilización de gestor de direcciones en nuestro servidor, y la inclusión de URLS mejoradas ya que no contendrán especificado el archivo index.php como dato en la URL.

## **2.5.- APLICACIONES**

### 

### **2.5.1- AGREGAR APLICACIONES**

El sistema agrega aplicaciones las cuales son las necesarias para la aplicación móvil, al agregar aplicaciones accedes la clase apps dado por el archivo **apps.php** y dentro de esta al método créate el cual manda llamar al modelo App\_model dado por el archivo **App\_model.php** y utiliza el método set\_app() de este modelo.

Al tratar de dar de alta un nombre de aplicación que ya existe el sistema bloquea la posibilidad de poder dar de alta una nueva aplicación, esto lo realiza el controlador Apps mediante el método checkExistence().

### **2.5.2- VER APLICACIONES**

Para ver las aplicaciones mostradas en pantalla, se accede al controlador apps.php y al método view( ) el cual muestra las recetas correspondientes a la aplicación seleccionada, este controlador accede al modelo App\_model donde llama al método getCategoryFromAppId($parametro), al cual se le envía como parámetro el número de la aplicación deseada para ver sus recetas.

### **2.5.3.- EDITAR APLICACIONES**

Al tratar de editar una aplicación el sistema nos muestra la posibilidad de editar el nombre de la aplicación seleccionada mediante un pop-up, al realizar esta edición accedes al controlador Apps y a su método edit, el cual recibe un nombre y un id de aplicación este método actualiza los datos de una aplicación mediante el modelo App\_model y su método updateAppName() al cual le enviamos como parámetros el id de la aplicación y el nuevo nombre a editar. Cabe destacar que al momento de tratar de usar un nombre de aplicación que ya existe el sistema no permite utilizarlo y bloquea la posibilidad de hacerlo. Esto lo realiza el Controlador Apps mediante el método updateCheckExistence() el cual es llamado mediante jQuery en tiempo de ejecución.

### **2.5.4.- ELIMINAR APLICACIONES**

Al eliminar una aplicación el sistema accede a el método Apps y mediante el método eliminar del controlador realizas la llamada al modelo App\_model y el método eliminar de este modelo el cual borra de forma definitiva todo lo relacionado con esta aplicación. Como lo son sus glosarios, recetas complementarias y videos.

## **2.6.- RECETAS**

### **2.6.1.- AGREGAR RECETAS**

Para agregar recetas se necesita del controlador recetas.php y el método encargado de mostrar el código html de la nueva receta es nueva(), al completar los campos y crear la receta el sistema se va a el controlador recetas y el método que realiza el siguiente paso para registrar los datos de la receta y dar de alta las relaciones de videos con esta receta, glosarios relacionados en esta receta y recetas complementarias es addComplementarias() del archivo recetas.php. Este método addComplementarias registra todos los campos de la receta y redirecciona al controlador recetas.php y al método relations. Donde nos muestra la interfaz para relacionar videos con recetas, recetas complementarias con nuestra receta recién creada y asignar un glosario a nuestra receta, el controlador encargado de dar de alta todas las relaciones de nuestra receta es complementarias.php y el método encargado de analizar esto se llama adCheck(), este da de alta las relaciones y nos redirige al controlador apps.php -> método view().

### **2.6.2.- VER Y EDITAR RECETAS**

Para ver las recetas de una manera mucho más detallada se accede al controlador recetas.php -> método ver(), este recibe dos parámetros el primero es el id de la receta el segundo es el id de la aplicación, este método obtiene mediante el modelo recetas\_model lo que se encuentra relacionado los métodos son:

getCategoryFromAppId($id,$id\_app) => obtiene las categorías, primer parámetro id de la receta, segundo parámetro id de la aplicación.

getGlosarioRelacionado($id,$id\_app) => Obtiene el glosario que se encuentra relacionado con nuestra receta, primer parámetro id de la receta, segundo parámetro id de la aplicación.

getVideosRelacionados($id,$id\_app) => Obtiene los videos relacionados con nuestra receta, primer parámetro id de la receta, segundo parámetro id de la aplicación.

getcomplementariasRelacionadas($id, $id\_app) => Obtiene las recetas complementarias que se encuentran relacionadas con nuestra receta, primer parámetro id de la receta, segundo parámetro id de la aplicación.

En esta misma vista se pueden editar las recetas, una vez que te encuentres en recetas/view que significa controlador recetas y el método view, te aparece un botón donde puedes editar los datos de dicha receta, estos datos al completar la edición se dirigen a recetas/edit donde se procesan y actualizan los datos correspondientes.

### **2.6.3.- ELIMINAR RECETAS**

Al eliminar una receta se accede a la URL recetas/eliminar esto a su vez entra al controlador Recetas.php y dentro de este al método eliminar donde recibe el id de la aplicación y el id de la receta, y su vez accede a eliminar todas las relaciones de esta receta con respecto al glosario, los videos relacionados con esta receta y las recetas complementarias de esta receta. El modelo de esta receta se llama recetas\_model.php y el método que elimina las recetas es eliminar.

## **2.7.- CATEGORIAS**

### **2.7.1.- AGREGAR CATEGORIAS**

Al agregar una categoría accedes al controlador categorías.php y el método que da de alta una categoría se llama créate, este método llama al modelo categoría\_model.php y ejecuta su método set\_categoria donde a este método le llega el id de la aplicación, el nombre y un color en código RGB, delimitando cada valor por comas. Por ejemplo 123,34,67.

### **2.7.2.- VER Y EDITAR CATEGORIAS**

Para ver las categorías, es necesario dar click en editar categorías, este muestra la información referente a cada categoría, al modificar algún campo de las categorías, das un click en editar accedes al controlador categorías.php y al método edit, este recibe el id de la aplicación, el id de la categoría, un nombre para la categoría y un color de categoría. El modelo encargado de hacer los cambios en la base es categoría\_model.php y el método que realiza los cambios es update\_categoria este recibe un nombre, un color y un id de la categoria como parámetros.

### **2.7.4.- ELIMINAR CATEGORIAS**

Al tratar de eliminar una categoría, accedes al controlador categorías.php y al método delete, este recibe el id de la categoría a eliminar y mediante el modelo categoría\_model.php ejecuta el método delete\_recipe.

## **2.8.- GLOSARIO**

### **2.8.1.- AGREGAR GLOSARIO**

Al agregar un glosario accedes al controlador glosario.php y al método créate, este ejecuta del modelo glosario\_model.php y ejecuta el método créate de este modelo, dicho método recibe el id de la aplicación, el nombre, una descripción del glosario y el texto de una imagen.

### **2.8.2.- VER Y EDITAR GLOSARIO**

Para ver un glosario das click en el botón de editar donde te muestra los datos correspondientes a cada glosario; muestra la opción para editar sus datos, al hacerlo accedes al controlador glosario.php y este a su método edit, dentro de este método llamas al modelo glosario\_model.php y a su método edit, donde recibe el id del glosario, nombre y el texto de la imagen.

### **2.8.3.- ELIMINAR GLOSARIO**

Para eliminar un glosario el controlador es glosario.php y el método al que accedes se llama delete, este llama el modelo glosario\_model.php y dentro de este al método delete, este método recibe como parámetros el id del glosario y elimina los datos de ese glosario de la base de datos.

## **2.9.- VIDEO**

### **2.9.1.- AGREGAR VIDEO**

Al agregar un video, accedes al controlador videos.php y al método créate, este llama al modelo video\_model.php y a su vez al método create(), este recibe como parámetros el id de la aplicación un nombre de video y un título descriptivo para cada situación.

### **2.9.2.- VER Y EDITAR VIDEO**

Para ver un video solo tienes que dar click en la parte de editar un video, este te muestra los datos correspondientes de cada video, y a su vez te muestra la opción para poder editar este video, al hacerlo accedes al controlador videos.php y al método edit ( ), donde mediante el modelo video\_model.php ejecutas el método edit ( ), este recibe como parámetros el id del video a editar, el título y el texto del archivo de video.

### **2.9.4.- ELIMINAR VIDEO**

Para eliminar un video accedes al controlador videos.php y al método delete, este llama al modelo video\_model.php y dentro de este al método delete ( ), este recibe como parámetro el id del video a eliminar.

## **2.10.- RECETAS COMPLEMENTARIAS**

### **2.10.1.- AGREGAR COMPLEMENTARIA**

Para agregar una receta complementaria es necesario acceder al controlador complementarias.php y al método create, este llama al modelo complementarias\_model.php y dentro de este al método create donde este método recibe el id de la aplicación, el título y un contenido correspondiente a una complementaria.

### **2.10.2.- VER Y EDITAR COMPLEMENTARIA**

Para ver una receta complementaria tienes que dar click en editar y obtienes los datos de esa receta en un PopUp, y te muestra la posibilidad de editar esta, al hacerlo accedes al controlador complementarias.php y al método edit, este llama al modelo complementarias\_model.php y dentro de este al método update ( ), este recibe el titulo, el contenido y el id de dicha receta para editarlo en la BD.

### **2.10.4.- ELIMINAR COMPLEMENTARIA**

Al eliminar una receta complementaria, se accede al controlador complementarias.php y al método delete (), este llama al modelo complementarias\_model.php y dentro de este al método delete donde este recibe un id de la receta complementaria y la elimina de la BD.

# **3.- EXPORTAR**

## **3.1.- CONFIGURACION INICIAL**

Para que la manera de exportar los archivos sea ejecutado de manera correcta, es necesario que se active el módulo de php para generar archivos de sqlite llamado php­\_pdo\_mysql, este permitirá generar un objeto PDO el cual es el necesario para ejecutar las consultas de sqlite en un archivo.

## **3.2.- CONTROLADOR EXPORTAR**

Al entrar en la parte de exportar inicialmente accedes al controlador export.php y al método create ( ), este método primero crea arreglos que contienen los datos referentes ala aplicación actual, después eliminamos la base de datos en caso de que exista con el método deleteBDIfExists del controlador, inmediatamente de eliminar la base de datos se vuelve a crear mediante el método createDBSqlite, el método createTablesSQlite genera toda la estructura necesaria para que los datos se puedan agregar de manera exitosa, el método insertIntoTableSQlite inserta los datos contenidos en los arreglos en sus tablas correspondientes mediante un algoritmo de análisis de datos, el cual separa los datos en tokens de acuerdo a el contenido de cada arreglo.

## **3.3.- MODELO EXPORTAR**

Contiene todos los métodos necesarios para generar los arreglos necesarios en el controlador, al momento de ingresar los datos en la base de datos para el sistema móvil.

## **3.4.- VISTA EXPORTAR**

La vista de exportar muestra un preview de los datos insertados en el archivo sqlite. Ademas de mostrar la posibilidad de descargar el archivo Larousse.db