**Estructura del framework**

**Explicación de las carpetas**

1. **Application:**

Carpeta que contiene los archivos de configuración principal que serán usados desde cualquier parte del proyecto.

A continuación se detallara un poco más sobre los archivos .php dentro de esta carpeta

**-Boostrap.php:**

Contiene la clase “Bootsrap” la cual se encarga de llamar a los controladores dentro de la carpeta “Controllers”, con el fin de llegar al método que está siendo solicitado vía URL.

En pocas palabras procesa las llamadas a los controladores y a los métodos.

**-Config.php:**

En este archivo definimos las constantes que se podrán usar en la totalidad de la estructura del proyecto, por ejemplo constantes tales como “BASE\_URL” que se define como nombre para obtener la URL base a utilizar, en este caso “localhost/mvc\_1”.

Se establece también cual será el layout por defecto del proyecto, podemos también definir en este archivo la cantidad de tiempo activa de una sesión, así como también definir los parámetros de conexión a la base de datos que usaremos.

**-Controller.php:**

Contiene la clase “Controller” padre de la cual heredaran todos los controladores a ser utilizados dentro del proyecto, contiene diversas funciones tanto para el filtrado de strings, integer limpieza de cadenas, carga de modelos, redireccionamientos etc.

**-Database.php:**

Contiene la clase “Database” la cual extiende de la clase “PDO”, el método contructor de la clase Database hace referencia al método contructor de la clase PDO donde se establece el driver de conexión a utilizar, asi como también se pasan las constantes que definimos en “Config.php” con el host, nombre de la base de datos, usuario y contraseña para establecer la conexión.

**-Hash.php**

Contiene la clase “Hash” la cual posee un método estático, que sirve para encriptar las contraseñas, según el algoritmo deseado.

**-Model.php**

Contiene la clase “Model” padre de la cual heredaran todos los modelos a ser utilizados dentro del proyecto, su método constructor es una instancia de la clase “Database” permitiéndonos poder establecer una conexión con la base de datos.

Dentro de esta clase encontramos diversas funciones que nos permitirán el manejo dinámico de la información desde los modelos hijos que hereden de dicha clase.

**-ModelHtml.php**

Contiene la clase “HTML” en la cual tenemos diversas funciones que renderizan algunas de las etiquetas html de más uso para poder tenerlas disponibles desde cualquier sitio del proyecto, dicha clase está pensada más para las vistas, así evitamos la carga excesiva de etiquetas html dentro de las mismas.

**-Request.php**

Contiene la clase “Request” la cual es un support que se encarga de verificar la URL, limpiarla y devolver el controlador, método y (en caso de poseer) los argumentos solicitados por el usuario a través de la URL.

**-Router.php**

Contiene la clase “Router” la cual nos provee de una única función llamada “create\_action\_url” que recibe 2 parametros y nos permitirá armar la ruta hacia una acción en el controlador por ejemplo:

**Router::create\_action\_url(“grupos/view”, [1] ) ;**

En la anterior línea estamos accediendo al método estático “create\_action\_url” de la clase “Router”, y le estamos pasando como primer parámetro un string que contiene el nombre del controlador seguido de un “/” y el nombre del método dentro de dicho controlador, el segundo parámetro (que puede ser nulo) lo pasamos como un array en este caso enviamos un “1”, la función simplemente retornara la ruta hasta el controlador que establecimos, para el ejemplo obtendremos en la barra de navegación:

“index.php?url=grupos/view/1”

**-Session.php**

Contiene la clase “Session” la cual proporciona los métodos necesarios para el manejo de sesiones, tanto como la definición y obtención de variables de sesión, niveles de acceso para los usuarios, tiempo de sesión etc.

**-View.php**

Contiene la clase “View” la cual posee un importante metodo llamado “render”, este nos permitirá renderizar las vistas desde los controladores, este renderiza el layout como las opciones del menú.

1. **controllers:**

Carpeta que contendrá todos los controladores a ser utilizados dentro del proyecto, cual es la esencia del patrón de diseño “MVC” dividir los archivos de tal manera que se trabaje de una forma más organizada, siguiendo el estándar cada controlador debe tener su vista así como también su modelo, y estos controladores deben llamarse de cualquier forma que le sea fácil de identificar seguido de la palabra “Controller”.

Dentro del proyecto un controlador común posee un método constructor, el cual inicializa el método constructor de su padre “Controller.php”, también posee un método index(), nuevo(), editar(), eliminar() métodos para la construcción de un CRUD básico.

1. **models:**

Carpeta que contendrá todos los modelos a ser utilizados dentro del proyecto, siguiendo el patrón de diseño MVC, todos los modelos deben llevar el nombre con el cual lo definirán y al final la palabra “Model”.

Dentro del proyecto un modelo común extiende de la clase padre “Model.php” y debe llevar 2 atributos públicos que corresponden al nombre de la tabla a la cual se está haciendo el modelo, y su llave primaria, así como también a de contar con los métodos get, insert, update, delete los mismos serán los que tendrán manipulación directa con la base de datos tanto para obtener información, como insertar, actualizar o eliminar la misma según sea el caso.

1. **views:**

Carpeta que contendrá todas las vistas a ser utilizadas dentro del proyecto, siguiendo el patrón de diseño MVC, todas las vistas estarán categorizadas dentro de sus respectivas carpetas al mismo tiempo dentro de las carpetas estarán los archivos que corresponden a cada una de las vistas para las acciones que se desean realizar teniendo en cuenta los métodos que se hayan definido en los controladores.

Dentro del proyecto una vista común posee su carpeta con el nombre que se le quiera definir, y dentro de dicha carpeta un archivo llamado index.php, nuevo.php, editar.php, archivos que se corresponden con los métodos del controlador y poseen la estructura básica HTML que se quiere mostrar al usuario

**Archivos adicionales**

**-index.php:**

Archivo principal del proyecto el cual se encarga de requerir los archivos esenciales de la carpeta “application”, y los modelos de la carpeta “models” para poder usarlos desde cualquier parte del proyecto, asi como tambien hace un llamado a la función run() de la clase Bootstrap para incializar las funciones encargadas del procesamiento y llamados a los controladores/métodos por medio de la URL

**.htacces:**

Archivo de configuración de apache para manejar las peticiones de entrada y requerimientos URL del aplicativo.