Proyecto de final de curso

Datos del Alumno:

* Nombre Completo: Antonio Contreras Román
* Curso: 2 DAW (Diseño de Aplicaciones Web)
* Tutor: Luís Sanchez
* Centro: IES Campanillas
* Github: https://github.com/antonio06/ProyectoBin

Introducción:

Este proyecto corresponde a la asignatura de proyecto integrado del Segundo año del ciclo superior de DAW, cuya finalidad es la de diseñar un gestor de actividades y participantes para la asociación sin ánimo de lucro BIN. Cuyo intermediario ha sido el mi profesor y tutor Luís y en última instancia la profesora Cristina en representación de la propia asociación. Este proyecto está íntegramente realizado en tecnologías web como PHP, CSS3 y JavaScript ayudándome también de framework como es materialize y el gestor de iconos Font Awesome y el gestor de plantillas Twig. Todo este proyecto, además de la documentación se encuentran íntegramente en mi cuenta de github que se encuentran en los datos del alumno.

Como ya mencione es un proyecto web y como tal he utilizado el gestor de bases de datos que proporciona phpMyadmin.

**Explicación detallada de la aplicación**

Para empezar vamos a hablar como es lógico de los cimientos bases para construir esta aplicación. Lo más importante para que todo funcione bien es una buena base de datos la cual almacene los datos sensibles, y de sentido a la aplicación.

Este proyecto cuenta con dos tablas base muy importantes como son la tabla **Actividades** y **Personas.**  Ya he comentado que esta aplicación web va destinada a la gestión de actividades y la participación en ellas de personas. Estas actividades cuentan como datos con el propio nombre que tendrá la actividad el estado en el que se encuentra dicha actividad (contamos con varios estados ya predefinidos como son “En proyecto”, “Abierto de solicitud”, “Finalizado plazo de solicitud” etc), El nombre del coordinador, el del ponente, la ubicación en la que se realizará dicha actividad tendrán una fecha de inicio y de fin, un horario de inicio y de fin, un total de horas a realizar en la actividad, el precio en euros si tendrá o no IVA un descriptor (también cuentan con datos ya predefinidos “Infantil”, “Primaria”, “Profesores” etc) y una observación. La siguiente tabla principal de esta aplicación es la de personas que cuentan con un DNI, nombre, 1º y 2º apellido, un perfil(para este campo es necesario crear una tabla aparte ya que una persona puede tener uno o varios perfiles aunque para esta tabla serán un número), una foto o logo, un sexo (hombre o mujer), una fecha de nacimiento, la dirección de dicha persona, el municipio y la provincia donde vive, el país una fecha de alta y otra de baja, el número de la seguridad social, número de cuenta bancaria el email y la contraseña que tendrán para entrar en nuestra aplicación y gestionar su perfil y sus actividades, también contarán con un perfil de usuario en los cuales se definen:

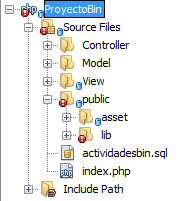
* Administrador: Perfil con permisos totales al panel de gestión.
* Limitado: Perfil con permisos solo de lectura al panel de gestión.
* Usuario: Perfil que solo podrá gestionar las actividades en las que participe y su perfil.

Como último cuenta con una observación. Complementariamente a esta tabla fue necesario crear una tabla secundaria llamada **Perfil** ya que las personas que participan en las actividades pueden ingresar a dicha actividad con varios perfiles NOTA(una persona puede tener muchos perfiles pero sólo puede participar en una actividad con un único perfil). Los perfiles creados son “Socio”, “Ponente” , “Monitor”, “Participante” y “Colaborador”. Ya contamos con unas actividades y unas personas que pueden realizarlas con un perfil distinto. Deducimos que una persona puede realizar muchas actividades y que una actividad puede contener muchas personas, para esto como podemos deducir tenemos que crear una cuarta tabla que la llamaremos **Participa**, que contendrá el código de una persona que participa en una actividad X con un perfil determinador.

Como ya dijimos una persona puede participar en tantas actividades como quiera pero no puede participar con un mismo perfil en una misma actividad. El problema surge ya que en la base de datos podemos decir que no acepte que un participante esté en una actividad con un perfil se repita más de una vez pero no que se repita el perfilen la misma actividad, aunque eso lo solucionamos con la programación en PHP.

Ya tenemos un pilar importante resuelto, pero solo con la base de datos no podemos hacer mucho, el segundo pilar fundamental de nuestra aplicación será y ya entramos en la parte de PHP de **Clases de objetos .**

Para dicho proyecto hemos utilizado la versión 5.6.15 de php y utilizando el paradigma de orientación a objetos, al igual que para la BBDD definimos dos clases que son fundamentales como son la clases de personas y de actividades y como secundaria una clase para la conexión a la base de datos. Además de el paradigma de orientación a objetos utilizamos la gestión de archivos del MVC (modelo vista controlador).



Este es la estructura básica del proyecto la cual detallamos a continuación:

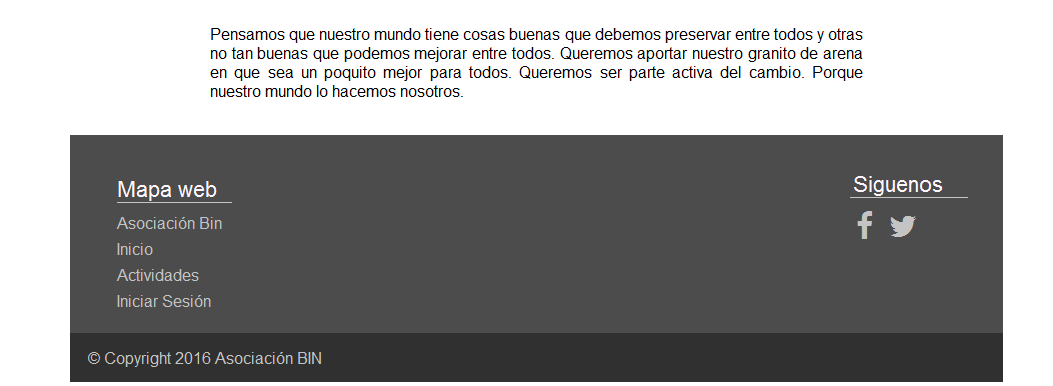
* Model: Dentro de ella encontramos las tres clases principales de nuestro proyecto que son Actividad, Persona y BinDb la gestión de los participantes la realizamos dentro de la clase de Actividades.
* Controller: Aquí encontramos los distintos controladores que usamos. En este fichero como en el View tenemos que mencionar otro aspecto importante dicha actividad la he separado en dos partes, **Parte Pública** y **Parte Privada.** En la parte privada gestionaremos la parte del panel de administración de la aplicación y la gestión de los usuarios con permisos de Administrador y limitados, En la parte pública gestionaremos la parte visible para todo usuario que entre de nuevas a nuestra aplicación, y la de la gestión tanto de su perfil como la de las actividades de las personas con perfil Usuario.
* View: Aquí tenemos todas nuestras vistas tanto de la parte privada como pública de nuestra aplicación.
* Public: Dentro de la cual distinguimos asset en la que se encuentran el archivo de css y los archivos javascripts creados, y la de lib que contiene las distintas librerías que hemos utilizado que son materialize para el css Font Awesome para insertar los iconos, jquery y timepicki una librería externa que utilizamos para visualizar las horas.

Para explicar con más detalle la gestión de la aplicación vamos a ir paso a paso navegando por ella.

Inicio

Nuestro home cuenta con una estructura básica un nav (o menú) con distintos enlaces a las demás páginas de la asociación, el logo principal de la página, un texto explicativo de que es bin y lo que hace y un footer o pie de página con un mapa web y enlaces a las redes sociales



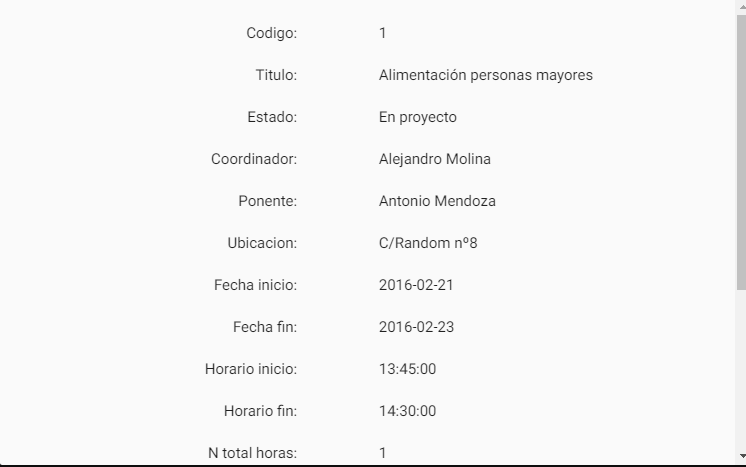


Actividades



Esta es la tabla actividades la cual nos muestra los datos más relevantes de la actividad.

Si queremos ver más información extra sobre dicha actividad podemos clicar sobre el icono con forma de lupa la cual nos abre un modal (que está ya predefinido por materialize) con los detalles



El paginado de esta y todas las páginas está realizado por medio de ajax con el ahorro interno correspondiente al cargado de toda la página ya que mediante este podemos solo cargar la parte que nos interesa en nuestro caso la tabla.

Iniciar Sesión



En esta parte contamos con el formulario para ingresar con la contraseña y el usuario.

Internamente lo que ocurre es que el usuario y la contraseña son enviados como métodos Post al fichero login.php en dicho fichero hay dos métodos de clases con los cuales obtenemos tanto el usuario como la contraseña y mediante la función propia de php password\_verify comprobamos que la contraseña introducida por el usuario coincide con la que sacamos mediante el método y además comprobamos que el usuario es el mismo al sacado por el método. Si estas dos condiciones se cumplen la siguiente condición a cumplir será si dicho usuario tiene un perfil administrador o limitado o es un usuario ya que cada uno envía a una parte distinta de la aplicación.

Ya hemos introducido nuestro usuario y nuestra contraseña imaginemos que somos administradores, la aplicación nos enviará al panel de gestión



En ella tenemos un titulo que nos indica donde estamos el menú principal y un mensaje de bienvenida. Tanto para esta parte, como para la parte pública de la aplicación, hemos utilizado la herencia de twig la que nos permite crear un fichero padre donde tenemos el menú y los meta y links necesarios para las librerías y css y desde ahí llamar a ficheros hijos .

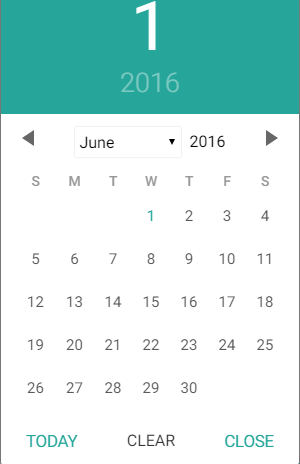
La primera pestaña de nuestro menú es la de **Gestión Actividades**.



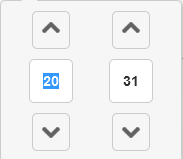
En dicha pestaña contamos con una tabla de las actividades con su paginación, si vamos a la parte derecha veremos una serie de botones (todos ellos proporcionados por Font awesome) al estar como administradores de dicha aplicación tenemos permisos totales y podremos crear modificar ver y borrar todos los datos.

Si pulsamos sobre el botón de añadir (icono de la cruz) se nos abrirá un modal con el formulario de las actividades.

Hay que destacar que dicho formulario está validado por javascript (solo ciertas partes) y el resto por php por lo tanto una actividad no podrá ingresarse si no tiene como mínimo un titulo un estado una ubicación una fecha de inicio de fin, un horario de inicio y de fin un total de horas, un iva y un descriptor si todos los datos han sido añadidos correctamente realizará el insert correspondiente en caso de que el servidor devuelva un error esté será reflejado en el modal. Para los datos que contienen fecha materialize cuenta con un datepicker propio y prediseñado, el cual nos permite ver de manera fácil y sencilla los días y el mes en el que estamos, aunque tiene un inconveniente lo que inserta en el input no es un formato fecha admitido dentro de los type date (formato 00/00/0000) si no que pone el día el mes (en texto) y el año por lo tanto en chrome tenemos un problema.



Para el horario nos ayudamos de una librería externa y gratuita encontrada por internet, llamada timepicki



Hay que reseñar que en firefox funciona perfectamente pero en chrome no inserta la hora por el tipo time la solución para que en ambos navegadores funcione es ponerlo en tipo text.

Para ver más información y la licencia del producto podemos ir al perfil de github https://github.com/senthilraj/TimePicki/blob/master/LICENSE.txt

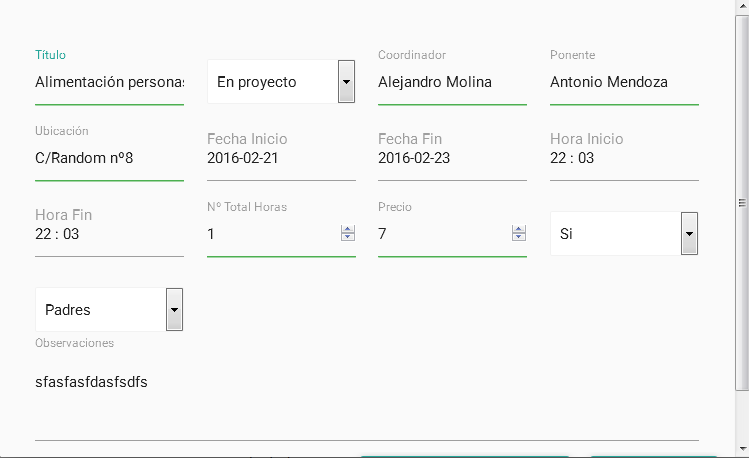
En cada línea de registro contamos con tres botones.

Si pulsamos sobre el icono con forma de lupa nos mostrará un modal con la información completa de dicha actividad.



Cuando mostramos los detalles nos encontramos un botón llamado Suscribirse a la Actividad la cual hace un insert a la tabla participantes, teniendo en cuenta que cuando ingresamos en una actividad lo hacemos con perfil de participante. En caso de que dicha persona ya esté ingresada en la actividad el botón de ingresado no aparecerá.

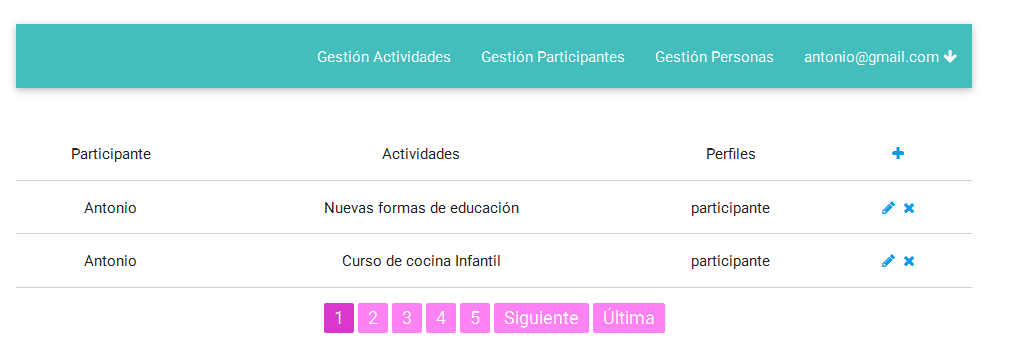
El siguiente botón es el de modificado (icono del lápiz).



Este formulario es el mismo que el de nuevo si el update no es realizado con éxito el servidor nos envía un mensaje de error.

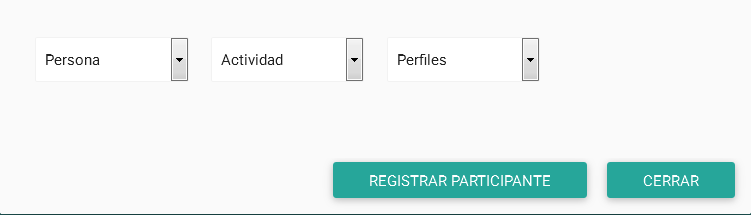
El último botón (el icono con la forma de x) nos borrará la actividad.

El siguiente apartado de nuestro menú es el de **Gestión Participantes**.



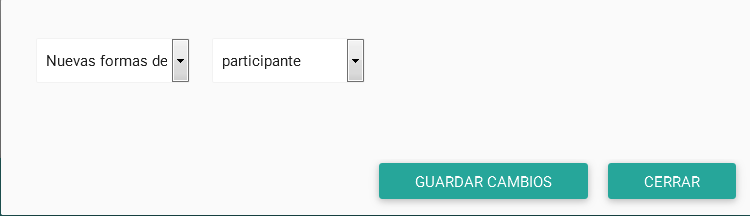
Como en la anterior pestaña tenemos una tabla central con los participantes que en cada actividad y el rol con el que participan. En la parte derecha tenemos nuestros típicos botones la única diferencia es que no tenemos el botón de detalles ya que al ser pocos datos que mostrar es un poco innecesario ponerlo.

El botón de añadir nos muestra el típico modal con tres desplegables con los participantes las actividades y los perfiles.



Para poder insertar un participante es obligatorio tener rellenos los tres campos. Como se comento una persona puede ser registrada en varias actividades distintas pero no en la misma varias veces con distinto perfil. Este caso es necesario controlarlo por php ya que sql no nos permite hacerlo (solo podemos controlar el caso de que los tres datos se repitan en la tabla).

El botón de modificado nos abre el mismo modal pero ya relleno con los datos que vienen de la base de datos para ser modificados (también controlamos en el modificado que el nuevo registro que añadamos no esté ya repetido). La única diferencia es que el nombre no es modificable.



El botón de borrado hace un delete al registro indicado.

La última opción de nuestro menú es la de **Gestión Personas.**



En esta opción tenemos la tabla con los datos más significativos de persona y a la derecha los botones de nuevo, ver detalles, modificado y borrado.

El botón de nuevo nos muestra el modal con el formulario de personas al igual que el de actividades está validado por javascript (también controlamos por parte del servidor que los datos que nos vienen son los correctos).



De esta parte tenemos que destacar dos partes importantes la inserción de la foto como la de la password.

Para la foto utilizamos el tipo de input de file, este nos llega al servidor mediante un $\_FILES[‘foto’][‘tmp\_name’], antes de insertarlo utilizamos la función de php file\_get\_contents con la que conseguimos es pasar a cadena . Una vez pasada por esta función le hacemos un addslashes que añadirá barras espaciadoras “/” entre los caracteres y a eso lo pasamos a base 64 con la función base64\_encode. También tenemos que hacer una pequeña modificación en la base de datos poniendo en el campo que tenemos que usar como longblod.

Para la password la contraseña nos llega a través de un $\_POST antes de insertarla utilizamos la función que nos trae php que es la de password\_hash, esta función admite dos parámetros, uno de ellos es el password y el otro el método de encriptación que en nuestro caso es el PASSWORD\_DEFAULT.

El botón de modificado nos muestra el modal con el formulario relleno.



El último botón es el de borrar.

La última opción que tenemos en nuestro menú del panel de gestión es un submenú desplegable el cual al desplegarlo tenemos las opciones de **Mi perfil, Mis actividades y Salir del perfil**.

Si pulsamos sobre Mi perfil tenemos el formulario con algunos de los datos más relevantes del perfil. Aquí el usuario podrá cambiar ciertos datos de su perfil.



La siguiente opción es Mis Actividades donde tendremos una tabla con las actividades en las que dicha persona está suscrito

En esta tabla no incluimos el nombre ya que entendemos que la persona que está en estas actividades es el que está conectado, además de el botón de borrado.

La última opción de este submenú es la de Salir del perfil que no es más que un borrado de todas las sesiones y header location a las actividades.

Hemos visto lo que puede hacer un usuario con permisos de administrador en nuestra aplicación (ya comentamos que los que tienen permisos limitados solo pueden leer datos pero no crear ni modificar los datos). Ahora veremos que ocurre cuando un usuario con perfil usuario se conecta.



Como vemos un usuario con perfil de usuario es redirigido a la página principal de actividades pero con el mismo submenú de los administradores (con las mismas opciones y funcionalidades).

Para el diseño de la aplicación para las partes de actividades, el login y el panel de administración hemos utilizado el framework css de materialize este nos proporciona elementos ya creados y con un diseño establecido, el home (inicio de la web) el css es propio además de ciertos elementos como los mensajes de error que salen en el panel de administración como al borrar o crear/modificar elementos . En cuanto a colores seguimos un poco la gama de colores del logo corporativo de la asociación (gamas de negro, cyan y rosa).

El siguiente punto a tener en cuenta es el responsive de la página, materialize nos proporciona un sistema de rejillas de 12 columnas, la manera que tiene de dividir la página es la siguiente para pantallas de móvil utiliza una de menor de 600px para tablets de 992px y para pantallas más grandes más de 992px (en el responsive creado por nosotros utilizamos las mismas medidas). Para el responsive de la tabla materialize nos da una idea nueva, gracias a la clase responsive-table al poner en un tamaño de móvil nuestra pantalla y gracias a css podemos hacer que cada tr de la tabla contenga todos los datos de una fila, aquí hay un ejemplo sencillo encontrado en el blog css-strick <https://css-tricks.com/examples/ResponsiveTables/responsive.php> aunque la gran diferencia con respecto a materialize es que este encierra todo en un div y para poder movernos y ver los demás datos nos muestra una barra de scroll.

Links y referencias :

Todas las utilidades frameworks y recursos se encuentran en internet y están disponibles en sus distintas web para ser consultadas y todas son totalmente gratuitas

Materialize: <http://materializecss.com>

Font-Awesome: <http://fontawesome.io/icons/>

Plugin TimePicki: esta es su web oficial <http://senthilraj.github.io/TimePicki/> también proporcionamos el repositorio de github <https://github.com/senthilraj/TimePicki>

Blog donde encontré documentación útil sobre el responsive de tablas: <https://css-tricks.com/responsive-data-tables/>