

PROYECTO PHP MOUNTAIN-CONNECT

Para empezar, mi proyecto está basado en la página web “Mountain-Connect”, pero trata un tema futbolístico. La página se llama “Football-Connect”, y trata básicamente de que tú, creando un usuario, puedas ver y comentar resultados de partidos, ver las clasificaciones y equipos de las ligas. Además, también te permite crear tus propias ligas con tus amigos, para añadir equipos y resultados.

Link del proyecto: <https://github.com/pasamaar11/mountain-connect.git>

Contenido

index.php	1
header.php	2
footer.php	2
register.php	2
login.php	2
logout.php	3
profile.php	3
editprofile.php	4
auth_check.php	4
leagues.php	4
createLeague.php	5
teams.php	5
matches.php	5
functions.php	6
data.php	6
styles.css	6
config.php	6

index.php

En el index incluyo las páginas de header y footer. Esta página index tiene 4 divs, el primero para insertar un breve título, el segundo para insertar un breve resumen de lo que puedes hacer en la página, en el tercero coloco enlaces a las páginas matches.php,

leagues.php y teams.php, y en el cuarto le recuerdo al usuario de registrarse o iniciar sesión, enlazando a las páginas register.php y login.php

header.php

En el header incluyo un div para colocar el nombre de la página "Football-Connect". Y luego, en una lista desordenada creo un menú de navegación, que cambia dependiendo de si estás con la sesión iniciada o no, esto lo consigo comprobando con `isset($_SESSION)`. Antes de iniciar sesión el menú incluye, Inicio, Login y Registrarse, y al iniciar sesión aparece Inicio, Perfil de usuario y cerrar sesión.

footer.php

En esta página incluyo la parte final que aparece en todas las secciones de la web. El footer contiene un mensaje con el nombre del proyecto, además del año actual. Este archivo se utiliza junto al header para tener una estructura en todas las páginas. De esta manera es todo más ordenado, limpio y tienen todas las páginas algo en común.

register.php

En el register se gestionan nuevos jugadores dentro de la web. Lo primero que incluyo es iniciar sesión, además de incluir un header y un archivo de funciones para poder limpiar los datos.

Si no existe dicho listado de usuarios, se crea un array `$_SESSION['usuarios']`, que es donde se guardan los registros, esto significa que hasta que el usuario no cierre sesión tendrá un usuario en la web, una vez cierre desaparece, esto es porque estamos trabajando si bases de datos.

La página recoge los datos que el usuario establece en el formulario, hay alguna respuesta que está marcada con un select, para que no elija la que quiera, sino las que yo ofrezco. Después de esto se comprueba si hay algún dato no válido, como un email sin "@" o una contraseña con menos de 6 caracteres (impuesto por mi). Si algún dato no es válido o está vacío saltará un mensaje personalizado.

Una vez el usuario rellene el formulario correctamente saldrá un mensaje diciendo que se creó correctamente y se guardará en el array de la sesión activa. Además, la página del registro una vez realizada se recarga para que no se rellene por segunda vez el formulario.

Al final de la página incluye el footer.

login.php

En esta página se gestiona el inicio de sesión de los usuarios registrados. Al principio se incluyen header y funciones necesarias de limpieza y validación de los datos. También

inicia sesión si esta todavía no está iniciada, para poder trabajar con la información almacenada en `$_SESSION`.

Si al enviar el formulario el usuario comete algún error la página muestra mensajes de error, como que el usuario no existe o que la contraseña no es esa.

Para comprobar el usuario y terminar de iniciar sesión la página busca dentro del array de usuarios, donde hemos guardado de forma temporal la información del usuario. Al encontrar un usuario con ese nombre compara las dos contraseñas. La contraseña está cifrada.

Si el usuario consigue iniciar sesión, es decir, que los datos eran correctos, se crea una nueva variable `$_SESSION['usuario_logueado']`, funciona guardando la información del usuario cuando está activo. Esta variable nueva nos permitirá más adelante que otras páginas sepan de quién es la sesión iniciada y cambien, como en el header.

Una vez el usuario inicia sesión se le redirige a la página de su perfil('profile.php').

logout.php

El logout es el encargado de cerrar sesión del usuario, eliminando así el usuario del array 'usuarios'.

Primero comprueba si hay sesión iniciada, sino se inicia para poder trabajar con ella.

Luego se eliminan todos los datos almacenados en `$_SESSION` y se destruye la sesión. Asegura que toda la información del usuario sea eliminada por completo, evitando que otro usuario use esta sesión anterior.

La página al final redirige a la página de login.php

profile.php

En la página del perfil vemos la información del usuario que ha iniciado sesión. Si no está activa la sesión la activa. En esta página están incluidos el header y una página de verificación llamada `auth_check.php`, que asegura que solo los usuarios logueados puedan entrar a este perfil.

La información se obtiene de la variable `$_SESSION['usuario_logueado']`, que es la contenedora de toda la información del usuario logueado. Esta información es mostrada mediante `htmlspecialchars`, que evita problemas con caracteres especiales.

Además, debajo de la información del usuario se muestra un listado de las ligas creadas por el usuario, con la variable `$_SESSION['ligas_usuario']`. Cada liga tiene un botón que redirige a las ligas para verlas.

También tenemos las opciones de editar perfil y de cerrar sesión.

editprofile.php

Se le permite al usuario editar la información de su perfil. Incluye un header y una página externa de verificación de usuario (auth_check.php). La información se obtiene de \$_SESSION[usuario_logueado], que mantiene todos los datos del usuario mientras dure la sesión.

En esta página se vuelve a ver el mismo formulario que en la página register.php, que una vez cambiada la información se actualiza en el perfil y en el array de la sesión.

Una vez actualizados los datos la página nos permite volver a redirigernos al perfil con un enlace.

auth_check.php

Sirve para proteger la información de la web que solo puedan ver los usuarios registrados.

Se verifica si la variable \$_SESSION['usuario_logueado'] existe, eso quiere decir que se ha iniciado sesión. Si la variable no existe quiere decir que no hay nadie logueado y que la página debe redirigirte al login, deteniendo la ejecución.

leagues.php

Dentro de esta página se gestionan las ligas de la web. Primero se inicia sesión si no está activa, y se incluyen header y auth_check.

La página guarda varios arrays en la \$_SESSION para almacenar los datos mientras dure la sesión.

Las variables son:

- \$_SESSION['ligas'] → Almacena todas las ligas existentes con su categoría.
- \$_SESSION['equiposPorLiga'] → Guarda los equipos que pertenecen a cada liga.
- \$_SESSION['partidosPorLiga'] → Guarda los partidos y sus resultados por liga.
- \$_SESSION['ligas_usuario'] → Almacena las ligas creadas por el usuario actual.

Si los arrays no existen se inician con datos de ejemplo para poder utilizar la página desde el principio.

Al usuario se le permite crear nuevas ligas, añadir equipos a una liga, agregar resultados y seleccionar otra liga para ver su información.

Se genera automáticamente la clasificación de cada liga, calculada por puntos, goles en contra, a favor y diferencia de goles, con los resultados guardados en `$_SESSION['partidosPorLiga']`.

La página también incluye enlaces a las páginas de equipos y partidos.

createLeague.php

Con este archivo se consigue que el usuario pueda crear una liga. Incluye el header y el `auth_check`. Además, también se utiliza `data.php` para que solo el usuario pueda ver la información de la página.

El formulario se envía con el método POST. Recoge nombre de la liga y categoría. Se usa `trim()` para evitar espacios innecesarios.

Una vez los datos sean correctos se guarda la información en el array `['ligas']`. Además de registrar la liga también se crean dos arrays `['equiposPorLiga'][$nombre]` y `['partidosPorLiga'][$nombre]`, estos arrays comienzan vacíos y se van rellenando más adelante.

Tras esta acción se muestra un mensaje indicando que se ha realizado la acción correctamente. Al final se carga el footer.

teams.php

Se muestran los equipos de las diferentes ligas de fútbol. En la parte superior, si no está activa la sesión, cómo en todas las páginas que emplea del uso de una sesión activa, se inicia. Luego se incluyen el header y `auth_chekc`.

La página mezcla los equipos creados por mí, desde el código, como equipos de ejemplo, y los equipos creados por el usuario. Esto funciona gracias a guardarlos en el array `[equiposPorLiga]`. Y con el uso de `$_SESSION` podemos leerlo mientras la sesión esté activa.

Para seleccionar las diferentes ligas se utiliza el POST y el GET, nombrando los equipos de dicha liga en una tabla enumerada.

La página incluye enlaces a otras páginas, como a la página de ligas y partidos.

matches.php

En la página se muestran todos los partidos de la liga seleccionada, junto a un área de comentarios para los usuarios registrados (gracias a `auth_check`).

Si todavía no existe el array `[comentarios]` se inicializa como un array vacío. Como todos los arrays de la página, se mantienen “vivos” mientras el usuario esté con la sesión iniciada.

Cuando el usuario escribe un comentario, se guarda el nombre del usuario con el array ['usuario_logueado'], el id del partido, el texto y la fecha. Se guardan dentro del array ['comentario'], con el valor, es decir, ['comentario'][\$idPartido].

La página sigue usando arrays fijos, como he hecho en otras páginas para dar valores por ejemplo. Se utiliza el método array_merge para combinar los partidos creados con los creados anteriormente.

Al final de la página aparecen enlaces a las páginas "leagues.php" y "matches.php".

functions.php

Limpia cualquier dato recibido desde un formulario antes de utilizarlo en el sistema. Evita comportamientos inesperados y protege la aplicación de código no deseado.

Aplica trim() para eliminar espacios innecesarios al inicio y al final del texto. Pasa el resultado por stripslashes(), que es el que elimina las barras invertidas que se añaden solas en algunos entornos.

Se utiliza htmlspecialchars() para convertir los caracteres especiales como ">" en entidades de HTML seguras.

data.php

Se asegura que la sesión esté activa gracias al sesión_status() y comprueba si todavía no se ha iniciado, en ese caso se ejecutará un sesión_start(), permitiendo trabajar con las variables almacenadas en la sesión activa.

El bloque revisa si existen las estructuras necesarias para gestionar las ligas, equipos y partidos del sistema.

Si no existe \$_SESSION['ligas'] se crea un array vacío, al igual que con cualquiera de las otras variables. Esto garantiza que la aplicación siempre tenga listas preparadas para almacenar información.

styles.css

En el archivo css lo que hacemos es personalizar las páginas. Configuramos la fuente, el fondo y los colores básicos del sistema.

Configuramos también el header y la barra de navegación, también personalizamos el formulario de registro para que le sea más sencillo y práctico al usuario. También se configura el footer.

config.php

Nombre: José Javier Pasamar Acón Curso: 2DAWV
Fecha: 16/11/2025

Este ardhivo contiene la url del proyecto y guarda la ruta física del servidor donde se guardarán los ardhivos subidos, como imágenes, documentos, etcétera.

Usa __DIR__ para tomar automáticamente la carpeta actual del archivo y subirlo a (../uploads/).