UNIDAD6.ACTIVIDAD2

DESARROLLO

Francisco Javier Signes Costa 2º DAW online

Una vez que hemos visto las ventajas que nos proporcionan librerías como Bootstrap, Foundation o Materialize, vamos a hacer que nuestro diseño tenga una apariencia profesional y lo dotaremos de funcionalidades interactivas para mejorar la experiencia de uso.

Sobre el proyecto que estamos haciendo, implementa estas librerías o sustituye nuestros diseños actuales por diseños ya hechos y utiliza los conocimientos adquiridos en la unidad según los requisitos mínimos que se detallan abajo, aunque puedes añadir todas las mejoras que consideres.

El proyecto deberá de tener cómo mínimo los siguientes requisitos:

- Una página de login con apariencia profesional, que valide los datos del formulario con JavaScript o alguna librería, que envíe los datos del usuario al servidor mediante JavaScript o alguna librería, que sean validados en el servidor y contrastados con la base de datos para ver si existe el usuario. El servidor deberá devolver una respuesta al cliente y en caso de ser correcto, redirigir a nuestra página de administración del sitio web, a nuestra página “dashboard”.
- Una página para crear nuevos usuarios con apariencia profesional y que cumpla igualmente con los requisitos de la página de login, pero una vez guardado el usuario, en vez de redirigirte a la página “dashboard”, le redirigirá a la página de login.
- Nuestra página dashboard debe tener una apariencia profesional y deberá de tener un botón que permita cerrar la sesión, destruir la cookie de sesión, y devolvernos a la página principal de nuestro sitio web.
- En la página dashboard, tendremos dentro del menú una opción llamada usuarios, en la que tendremos una lista con los usuarios que hay en la base de datos. Y tenemos que crear dos botones. Un botón que nos permita eliminar a un usuario, para lo que tendrá que mostrarnos un mensaje de alerta antes de eliminarlo. Otro botón que al pulsarlo nos permita modificar la contraseña del usuario. Ambos botones deberán de funcionar a través de JavaScript, VueJS o alguna librería.

Introducción

Esta serie de tres prácticas termina en esta que es donde le damos un poco de estilo con Bootstrap y añadimos las últimas funcionalidades. He reutilizado todo el código que he podido de las prácticas anteriores añadiendo solamente dos archivos nuevos. Uno que elimina el usuario y el otro que modifica la contraseña. Ambos archivos se ejecutan mediante unos botones en la tabla de panel.php.

Como siempre, te dejo el código al completo en el repositorio.

Eliminar_usuario.php

```php
<?php
session_start();
include('db.php');
include('UserModel.php');

// Verificar si el ID del usuario es pasado por URL
if (isset($_GET['id'])) {
    $usuarioId = $_GET['id'];

    // Crear una instancia de UserModel
    $userModel = new UserModel($conn);

    // SQL para eliminar el usuario
    $sql = "DELETE FROM usuarios WHERE id = ?";
    $stmt = $conn->prepare($sql);
    $stmt->bind_param("i", $usuarioId);

    if ($stmt->execute()) {
        $_SESSION['mensaje'] = 'Usuario eliminado correctamente.';
    } else {
        $_SESSION['error'] = 'Hubo un problema al eliminar el usuario.';
    }

    // Redirigir al panel de administración
    header('Location: panel.php');
    exit;
} else {
    // Si no se pasa el ID, redirigir a panel.php
    header('Location: panel.php');
    exit;
}

```

Iniciamos sesión e incluimos db.php con la conexión a la base de datos y UserModel.php con funciones de manejo de usuarios.

Verificamos con un condicional si en la URL se ha pasado el parámetro id y la almacenamos en la variable $usuarioId. Pillamos el id porque es un identificador único en la base de datos. De esta manera podemos eliminar el usuario correcto por su id.

Creamos un objeto UserModel y le pasamos como parámetro la conexión a la base de datos.

Preparamos la consulta de eliminación de la base de datos con unas sentencias SQL. No vamos a insertar directamente el id dentro de la consulta para evitar inyecciones sql, sino que usamos (?) como marcador de posición. Se agrega más tarde.

Vamos preparando el statement usando prepare(), sin ejecutarla todavía.

Vinculamos el parámetro con bind_param(), donde la (i) es un integer y el segundo parámetro es el valor que queremos usar en lugar de (?).

Si se ejecuta $stmt significa que se ha encontrado el id (el usuario existe) por lo tanto se puede eliminar, mostrando el mensaje de eliminación. De lo contrario nos da un mensaje de error.

Una vez que termina el condicional, redireccionamos al usuario al panel.php, independientemente que se haya eliminado o no el usuario.

Si no se dispone de id, directamente obviamos el if anidado y se sale por el último else que nos redirecciona a panel.php.

Modificar_contraseña.php

Lo primero que hacemos es verificar que el usuario está autenticado para acceder al cambio de contraseña. Se establece al revés y es un poco lioso, es decir, si el usuario no ha iniciado sesión, redirigimos a login.php, no dejándole entrar.

En la variable $usuario_id almacenamos la Id del usuario para poderla modificar posteriormente.

Creamos nuevamente una instancia de UserModel (un objeto de la clase que definimos en UserModel.php) y le pasamos como parámetro la conexión a la base de datos. En UserModel tenemos una función para cambiar la contraseña (actualizarContraseña()).

Verificamos que el formulario de cambio de contraseña se ha enviado correctamente mediante POST. De esta manera nos aseguramos antes de procesar el cambio.

Seguidamente obtenemos el id del usuario autenticado y la nueva contraseña que la almacenamos en $nueva_contraseña (ya la tenemos hasheada desde la función actualizarContraseña).

Llamamos al método actualizarContraseña dentro de la clase UserModel y le pasamos dos parámetros ($id y $nueva_contraseña). Guardamos el resultado en la variable $resultado. De esta forma actualizamos la información en la base de datos con un UPDATE (lo tenemos definido en la función).

Verificamos que el cambio ha salido bien y nos redirige a panel.php.

```php
<?php
session_start();
include('db.php');
include('UserModel.php');

// Nos aseguramos de que el usuario esté autenticado
if (!isset($_SESSION['usuario_id'])) {
    header('Location: login.php');
    exit;
}

$usuario_id = $_SESSION['usuario_id'];

// Creamos el objeto del modelo de usuario
$userModel = new UserModel($conn);

// Verificamos si el formulario de cambio de contraseña ha sido enviado
if ($_SERVER['REQUEST_METHOD'] === 'POST') {
    $id = $_SESSION['usuario_id'];
    $nueva_contraseña = $_POST['nueva_contraseña'];

    // Actualizar la contraseña en la base de datos
    $resultado = $userModel->actualizarContraseña($id, $nueva_contraseña);

    if ($resultado) {
        $_SESSION['mensaje'] = 'Contraseña actualizada';
        header('Location: panel.php');
        exit;
    } else {
        $_SESSION['error'] = 'Hubo un problema al actualizar la contraseña.';
    }
}

// Obtenemos la ID del usuario desde la sesión
$userId = $_SESSION['usuario_id'];

?>
```

Código completo en GitHub

De dejo en el enlace el código.

https://github.com/Frank512-lab/2T_EJERCICIOS_SERVIDOR/tree/main/UNIDAD%206/ACTIVIDAD%202