

Desarrollo Web en entorno servidor

Unidad 4:

Ejercicios

Tabla de contenido

- Ejercicio 1 2
- Ejercicio 2 2
- Ejercicio 3 4
- Ejercicio 4 5
- Ejercicio 5 5
- Ejercicio 6 5
- Ejercicio 7 6
- Ejercicio 8 6
- Ejercicio 9 6
- Ejercicio 10 6

- Ejercicio 1

Al añadir un producto, la aplicación redirige al carrito. Modifícala para que redirija a la tabla de productos, con la misma categoría.

- Ejercicio 2

Crea una nueva página "zonaadmin.php", en la que incluyas diferentes opciones de administración.

A esta página se accederá con un enlace desde la cabecera solo visible para los administradores.



La primera opción a incluir en zonaadmin.php es "Datos de restaurantes", enlace que redirecciona a la página datosusu.php en blanco.



Haz uso del archivo actualizar.sql para incorporar la columna rol a la tabla restaurantes.

Como no tenemos aún definidos roles, vamos a modificar la tabla **restaurantes** para incluir el rol de cada usuario. Los roles son los siguientes:

- 0 para los restaurantes.
- 1 para los administradores.

Usamos actualizar.sql en phpmyadmin para incorporarla columna correspondiente a la tabla **restaurantes** y volcar los datos de acceso de un administrador:

- Usuario: admin@empresa.com
- Clave: 1234

Cuando accedemos a la aplicación hemos de recuperar el rol del usuario que ha accedido, para ello modificamos la función `comprobar_usuario($nombre, $clave)` e incorporamos en la consulta el nombre de la columna correspondiente, **rol**.

```
function comprobar_usuario($nombre, $clave)
{
    $res = leer_config(dirname(__FILE__) . "/configuracion.xml", dirname(__FILE__) .
"/configuracion.xsd");
    $bd = new PDO($res[0], $res[1], $res[2]);
    $ins = "select codRes, correo, rol from restaurantes where correo = '$nombre'
        and clave = '$clave'";
    $resul = $bd->query($ins);
    if ($resul->rowCount() === 1) {
        return $resul->fetch();
    } else {
        return FALSE;
    }
}
```

De esta forma, cuando llamemos a la función `comprobar_usuario($nombre, $clave)` en `login.php`, el array `$usu` contendrá la clave **[rol]**, a mayores de las claves **[correo]** y **[codRes]**.

Además, al guardar dicho array en `$_SESSION($_SESSION['usuario'] = $usu)`, conseguimos que el rol esté disponible en toda la aplicación a través de `$_SESSION['usuario']['rol']`.

Una vez recogido el rol de usuario en `cabecera.php` añadimos las siguientes líneas.

```
<?php
if ($_SESSION['usuario']['rol'] == 1){
    echo "<a href='zonaadmin.php'>Administración</a>";
}
?>
```

Por último, creamos la página `zonaamin.php`

```
<?php

/*comprueba que el usuario haya abierto sesión o redirige*/
require 'sesiones.php';
require_once 'bd.php';
comprobar_sesion();
?>
<!DOCTYPE html>
<html>
    <head>
        <meta charset = "UTF-8">
        <title>Zona administrador</title>
    </head>
    <body>
        <?php require 'cabecera.php';?>
        <h1>Zona de administración</h1>
        <!--lista de vínculos -->
```

```

        <ul>
            <li><a href='datosusu.php'> Datos de restaurantes </a></li>
        </ul>

    </body>
</html>

```

• Ejercicio 3

La página `datosusu.php` muestra una tabla con los datos de los usuarios y, mediante un formulario que redirige a la misma página, permite actualizar los datos modificados.

Usuario: [admin@empresa.com](#) [Administración](#) [Home](#) [Ver carrito](#) [Cerrar sesión](#)

Correo	Clave	País	CP	Ciudad	Dirección	Rol	
madrid1@empresa.com	1234	España	28002	Madrid	C/ Padre Claret, 8	0	Modificar
cadiz1@empresa.com	1234	España	11001	Cádiz	C/ Portales, 2	0	Modificar
admin@empresa.com	1234	España	28002	Madrid	C/ Padre Claret, 8	1	Modificar

El ejercicio tiene dos partes.

a) Modifica el bloque de comentarios `--- A modificar ---` de `datosusu.php` para que, si se accede mediante POST a dicha página, se modifiquen los datos correspondientes a través de la función `actualizar_restaurante`, que devuelve TRUE si la modificación se ha ejecutado correctamente.

Si la modificación en la base de datos ha sido correcta, muestra el mensaje "Datos actualizados correctamente". En caso contrario, muestra el mensaje "Error al actualizar los datos".

b) Define la función anterior, la cual tiene por cabecera: `bool: actualizar_restaurante($datos)`. Esta función recibe como argumento un array con los datos del formulario, los cuales hay que modificar en la base de datos. La función ha de modificar la tabla ``restaurantes`` mediante una instrucción SQL preparada.

Devolverá True si la instrucción SQL se ha ejecutada correctamente y False sino.

La sentencia SQL a utilizar es

```

UPDATE restaurantes SET
    Correo = ...,
    Clave = ...,
    Pais = ...,
    Cp = ...,
    Ciudad = ...,
    Direccion = ...,

```

```
Rol = ...,  
Correo = ...  
WHERE CodRes = ..."
```

Incluir la siguiente función en bd.php para mostrar adecuadamente los datos de los restaurantes.

```
function cargar_restaurantes()  
{  
    $res = leer_config(dirname(__FILE__) . "/configuracion.xml", dirname(__FILE__) .  
"/configuracion.xsd");  
    $bd = new PDO($res[0], $res[1], $res[2]);  
    $ins = "select * from restaurantes";  
    $resul = $bd->query($ins);  
    if (!$resul) {  
        return FALSE;  
    }  
    if ($resul->rowCount() === 0) {  
        return FALSE;  
    }  
    //si hay 1 o más  
    return $resul;  
}
```

- **Ejercicio 4**

Modifica el ejercicio anterior para qué al actualizar los datos de los restaurantes, las claves queden encriptadas con el algoritmo bcrypt. Para ello, utiliza la función password_hash(). Modifica la función comprobar_usuario() en bd.php para que valide la clave introducida en login.php mediante la función password_verify().

- **Ejercicio 5**

Modifica la tabla de productos de productos.php para que no muestre los productos sin stock.

- **Ejercicio 6**

Añade el peso total del pedido en el contenido del correo de confirmación.

- **Ejercicio 7**

El fichero `servidor_correo.xml` almacena la información del servidor de correo en un fichero de XML como se hace con el de la base de datos. Modifica las funciones de envío de correo para que utilicen ese fichero. Valida la información recogida en dicho fichero con `servidor_correo.xsd` antes de usarlo.

- **Ejercicio 8**

Las páginas de administración creadas hasta el momento son accesibles por cualquier usuario con el rol 0 si sabe el nombre de la página php a la que quiere acceder.

Por ejemplo, accedemos a la aplicación con `madrid1@empresa.com` y modificamos `localhost/.../categorías.php` por `localhost/.../datosusu.php`

Crea la función `comprobar_admin()` en `sesiones.php` que compruebe si el rol del usuario es 1, en caso de no serlo, se ha de redirigir a `categorias.php`.

- **Ejercicio 9**

Modificar la consulta SQL de `comprobar_usuario()` en `bd.php` por una consulta preparada mediante los dos métodos, nombre y posición(orden)

- **Ejercicio 10**

Crea una página php con un formulario específico para que el administrador pueda modificar las claves. En `zonaadmin.php` crea un vínculo a dicha página con el nombre `Modificación de claves`. Elimina del formulario de `datosusu.php` el campo correspondiente a la clave. Define y modifica de forma adecuada las funciones que sean necesarias en `bd.php`.