

## Tarea 2 – UD3

### CARRITO DE COMPRA con AUTENTICACIÓN

**\*\*IMPORTANTE:**

**HAY DOS PRÁCTICAS, UNA SIN BBDD EN LA CARPETA *CarroCompraAUTH*, Y OTRA CON LA BBDD EN LA CARPETA *CarroCompraAUTH\_BBDD*.**

**Descripción de la Práctica:**

La práctica consiste en la implementación de un sistema de autenticación de usuario mediante un formulario propio en PHP, con contraseñas almacenadas de manera segura utilizando la función `password_hash()` y verificación mediante `password_verify()`.

Con esta autenticación, se busca restringir el acceso a la aplicación solo a usuarios autenticados, y algunas partes específicas como la gestión de stock, solo a usuarios específicos.

Se utiliza `session_start()` para manejar las variables de sesión y rastrear la autenticación del usuario.

El código se estructura en archivos separados para realizar tareas específicas, como mostrar formularios, cerrar sesiones y verificar la autenticación.

Y luego una segunda opción (después de las capturas), teniendo guardada la información de usuarios y resúmenes (hash) de contraseñas en una tabla de base de datos.

***\*\*Indicar que las explicaciones de este documento están un poco escuetas, sobre todo las cosas técnicas, porque está todo explicado/comentado en el código de todos los archivos php, todos los pasos y líneas del código están explicados con comentarios.***

**Pasos Realizados y Estructura del código:****1. Creación usuarios y contraseñas:**

- **Stock.php**
- Creo array usuarios para almacenar los usuarios y sus contraseñas, a modo de clave-valor.
- Como tenemos que crear contraseñas seguras, usamos la función `password_hash()`, y ese código hash es el que almacenamos en el array. *(las líneas de generación de hash las comentamos para que se puedan visualizar pero no deberían de estar ahí como protección).*

**2. Implementación de la Autenticación:**

- **index.php**
  1. Se creó un formulario sencillo de autenticación en `index.php`, para que sea lo que aparezca en el navegador al querer ir a cualquiera de las páginas de mi aplicación.
  2. Include de `stock.php`.
  3. Comprueba si existe un indicador de redirección (`redir`) en la solicitud. Si existe, muestra un mensaje de error de *Nombre o la contraseña son incorrectos..*
  4. Aquí creamos una función `muestraFormulario` al que, al llamarla, paso una url para que se redirija la página, en este caso le pasamos la url `auth.php`.
- **Auth.php**
  1. Se inicia la sesión y carga el archivo `stock.php` con include.

2. Creo variables para obtener el nombre de usuario y la contraseña de la solicitud.
3. El código de `auth.php` verifica las credenciales. Para comprobar el nombre se usa `array_key_exists` para verificar si el `nombreUsuario` está en la clave del array y para la contraseña, se usa `password_verify`, para verificar que la contraseña introducida coincide con la almacenada en el array `usuarios`. Y se redirige según el tipo de usuario.
4. Si es admin, puede acceder a todas las páginas de la aplicación, por lo que se le muestra enlaces al carrito de compras, a la gestión de stock y a la info del stock. Además de un mensaje de bienvenida y otro enlace para cerrar sesión.
5. Si no es admin, solo puede acceder a la página de carrito de compras, por lo que se muestra enlace a esta página además de un mensaje de bienvenida y enlace de cierre de sesión.
6. Si la autenticación es correcta, en la variable de sesión `"authok"` almacenamos el valor 1, indicando así que el usuario está autenticado..
7. Si la autenticación falla, se redirige a la página inicial (`index.php`) con un header y con `Location`, y parámetro `redir`, haciendo que ésta muestre el mensaje de error correspondiente al tener el parámetro `redir`.

### 3. Restricción de Acceso:

- **check-auth.php**
- Se crea `check-auth.php` para verificar si el usuario ha iniciado sesión y si no ha iniciado sesión, se redirige a la página de inicio.
- Si la variable de sesión `authok` no está establecida o su valor no es igual a 1, significa que el usuario no está autenticado.
- Este código asegura que el usuario solo tenga acceso a las partes protegidas de la aplicación si ha iniciado sesión correctamente. Si la variable de sesión `'authok'` no está establecida o su valor no es igual a 1, el usuario es redirigido a la página de inicio con un indicador de redirección. En resumen, garantiza la autenticación del usuario antes de permitir el acceso a ciertas áreas de la aplicación.

### 4. Cierre de Sesión:

- **logout.php**
- Creamos un `logout.php` para cerrar la sesión con `sesión_destroy()` y redirigir al usuario a la página de inicio con un `header(Location)`.
- Los enlaces a esta página, los hemos implementado en las páginas `carritoCompras`, `gestiónStock`, `infoStock` y en `auth`, para que el usuario siempre tenga la opción de cerrar sesión.

## Desafíos y Soluciones:


Lo más complicado tal vez haya sido la creación de código hash y su autenticación, pero una vez entiendes el fin y las funciones `password_hash` y `password_verify`, es fácil implementarlo.

Esta práctica no me ha resultado difícil, pues me he fijado bien en el ejemplo del profesor. Lo más complicado ha sido entender cada código y sobretodo la unión y función de cada archivo php, para qué servía cada uno. Pero no me ha sido fácil entenderlo y luego adaptarlo al código de mi aplicación.

También he visto y entendido la necesidad de repartir todo el código y funcionalidad en varios archivos diferentes para una aplicación ordenada y funcional.

**Capturas de pruebas:**

-Página nada más acceder a la carpeta de la aplicación: index.php:

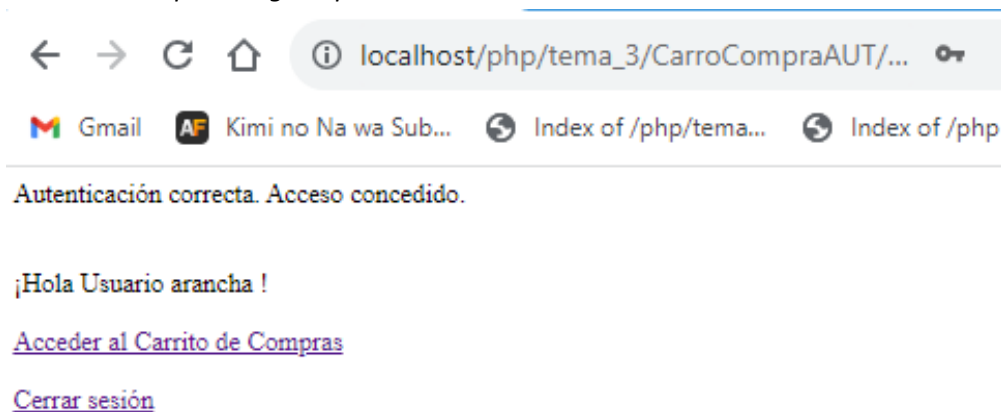


-Probamos con el usuario “arancha” (contraseña la misma):



- y damos a enviar y nos redirige a auth.php. Vemos que se muestra mensaje de autenticación correcta (que se imprime en el if de la verificación) y enlace para acceder al carrito de compra, ya que se ha detectado que no es admin y por lo tanto solo puede acceder ahí, también muestra enlace de cerrar sesión:

*Al iniciar sesión, aparece advertencia del navegador avisando de contraseña poco segura, pues en nuestro caso la hemos puesto igual que el nombre.*



- Accedemos al carrito de compras, y efectivamente nos redirige al carritoCompras.php, donde también se muestra enlace de cerrar sesión:

**CROQUETERÍA CHICHARRO**

**\*\*Todos los packs incluyen dos croquetas\*\***

*Todas las croquetas llevan queso manchego y cebolla.*

**CARRO DE LA COMPRA**

Producto	Añade Uds	Quitar Uds	Total unidades	Total precio
jamón (5 €/pack)	0	0	0	0 €
calabaza (4 €/pack)	0	0	0	0 €
pollo (3 €/pack)	0	0	0	0 €
bacalao (5 €/pack)	0	0	0	0 €
gambas (4 €/pack)	0	0	0	0 €
setas (3 €/pack)	0	0	0	0 €
espinacas (2 €/pack)	0	0	0	0 €

Total de unidades en el carrito: 0

**Precio total a pagar: 0 €**

[Cerrar sesión](#)

- Probamos a cerrar sesión, y nos redirige nuevamente al formulario, ya que este enlace dirige a logout.php y este manda un Location con header a la página del formulario, index.php:

**CROQUETERÍA CHICHARRO**

**Acceso:**

Nombre:

Contraseña:

- Ahora probamos con el usuario “eva”, misma contraseña, y también funciona:

The screenshot shows a web browser at the URL `localhost/php/tema_3/CarroCompraAUT/...`. The page title is "CROQUETERÍA CHICHARRO". Under the heading "Acceso:", there is a login form with the following fields and values:

- Nombre:
- Contraseña:
- Enviar:

Below the form, the message "Autenticación correcta. Acceso concedido." is displayed. Further down, the text "¡Hola Usuario eva !" is shown, followed by two links: [Acceder al Carrito de Compras](#) and [Cerrar sesión](#).

-Y con el usuario rodri:

This screenshot is identical in layout to the previous one, showing the same login page for "CROQUETERÍA CHICHARRO". The login form fields are filled with the username "rodri" and a masked password "....". The "Enviar" button is present. The success message "Autenticación correcta. Acceso concedido." is shown, followed by the greeting "¡Hola Usuario rodri !" and the links [Acceder al Carrito de Compras](#) and [Cerrar sesión](#).

-Y por último probamos con el usuario “admin”, que tiene privilegios para acceder a todas las páginas de la aplicación:

The screenshot shows a web browser at the URL `localhost/php/tema_3/CarroCompraAUT/...`. The page title is "CROQUETERÍA CHICHARRO". Below the title, there is a section labeled "Acceso:". It contains a form with two input fields: "Nombre:" with the value "admin" and "Contraseña:" with masked characters "\*\*\*\*". Below these fields is a button labeled "Enviar".

-Vemos que se muestran los enlaces de acceso a todas las páginas, además de enlace para cerrar sesión.

The screenshot shows the browser at `localhost/php/tema_3/CarroCompraAUT/auth.php`. The page displays the message "Autenticación correcta. Acceso concedido." followed by a greeting "¡Hola administrador!". Below this, it asks "¿Dónde quieres ir?" and lists several links: "Acceso al carrito", "Acceso a la gestión de Stock", "Acceso para ver el stock", and "Cerrar sesión".

-Probamos con acceso a la gestión de stock, y nos redirige bien, mostrando además enlace de cerrar sesión:

The screenshot shows the browser at `localhost/php/tema_3/CarroCompraAUT/gestionStock.php`. The page title is "CROQUETERÍA CHICHARRO". Below the title, there is a section labeled "Gestión de Stock". A link "Cerrar sesión" is visible. Below the link is a table with 4 columns: "Producto", "Añadir Uds", "Quitar Uds", and "Total uds en Stock". The table contains 8 rows of product data. At the bottom of the page, there is a button labeled "\*\*\*\*\* ACTUALIZAR STOCK \*\*\*\*\*".

Producto	Añadir Uds	Quitar Uds	Total uds en Stock
jamón (5 €/pack)	0	0	20
calabaza (4 €/pack)	0	0	20
pollo (3 €/pack)	0	0	20
bacalao (5 €/pack)	0	0	25
gambas (4 €/pack)	0	0	30
setas (3 €/pack)	0	0	35
espinacas (2 €/pack)	0	0	40

## **PARTE OPCIONAL: usuarios y contraseñas hash en tabla de BBDD**

Para no borrar código de la aplicación, he copiado los archivos para modificar código sin borrar el de la aplicación inicial. De esta manera ahora hay dos aplicaciones, una con autenticación php y otra con autenticación con BBDD.

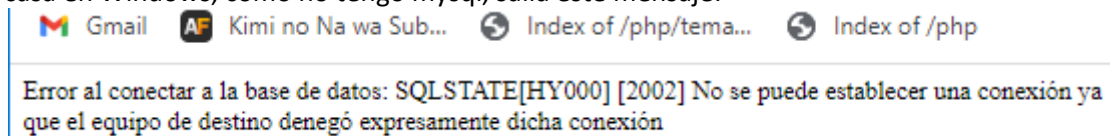
### **Pasos Realizados y Estructura del código:**

#### **1. Modificación de croqueteria.sql:**

- Añadimos la creación de una base de datos llamada 'croqueteria', y un usuario 'administrador' dándole todos los privilegios.
- Y creamos una tabla llamada 'usuarios' para almacenar los nombres de usuario en columna *usuario* y las contraseñas hash en la columna *pass*.

#### **2. Modificación de stock.php:**

- Primero tenemos que hacer la conexión con la base de datos y lo hacemos con PDO y un bloque try-catch para capturar posibles errores con getMessage(). Lo tuve que implementar así pues me surgieron errores y de esta manera se mostraba el tipo de error. Me sirvió de ayuda para solucionar algunos problemas, sobre todo al hacer la prueba en casa en Windows, como no tengo mysql, salía este mensaje:



Entonces este mensaje no dice que haya problema con el código o al conectar a la bbdd como tal, si no que el problema está en el equipo de destino.

Luego hice la prueba en la máquina virtual Ubuntu y el mensaje que salió era que no reconocía la bbdd 'croqueteria', por lo que por ello cargué el script sql manualmente en mysql.

- En la línea de conexión pdo, mandamos los datos de nuestra BBDD con el usuario y contraseña correspondiente.
  - Y para obtener los datos, en nuestro caso los usuarios y contraseñas, realizamos una consulta a la tabla de usuarios en la base de datos con la función *query()* para obtener la información de cada registro, es decir, de cada fila de valores, usando la función *fetch()* en un while. Con este función accedemos a una fila siempre que haya.
  - Y estos valores los almacenamos en nuestro array usuarios, el nombre como la clave y la contraseña como valor.
  - En el caso de que la consulta nos de algún error, sale mensaje de error con la función *die*, que también detiene el código.
3. Los demás archivos no tienen ningún cambio y están igual que en la aplicación anterior, exceptuando la eliminación de algunas líneas de comentarios.

## Desafíos y Soluciones:

Al crear el archivo sql, fijándome en el ejemplo del profesor me fue fácil adaptarlo a mi código y a los valores que yo quería almacenar. Pero en esta parte de la práctica tuve problemas con las contraseñas y usuarios. Me conectaba a la bbdd correctamente pero al poner el usuario y contraseña siempre salía mensaje de datos incorrectos.

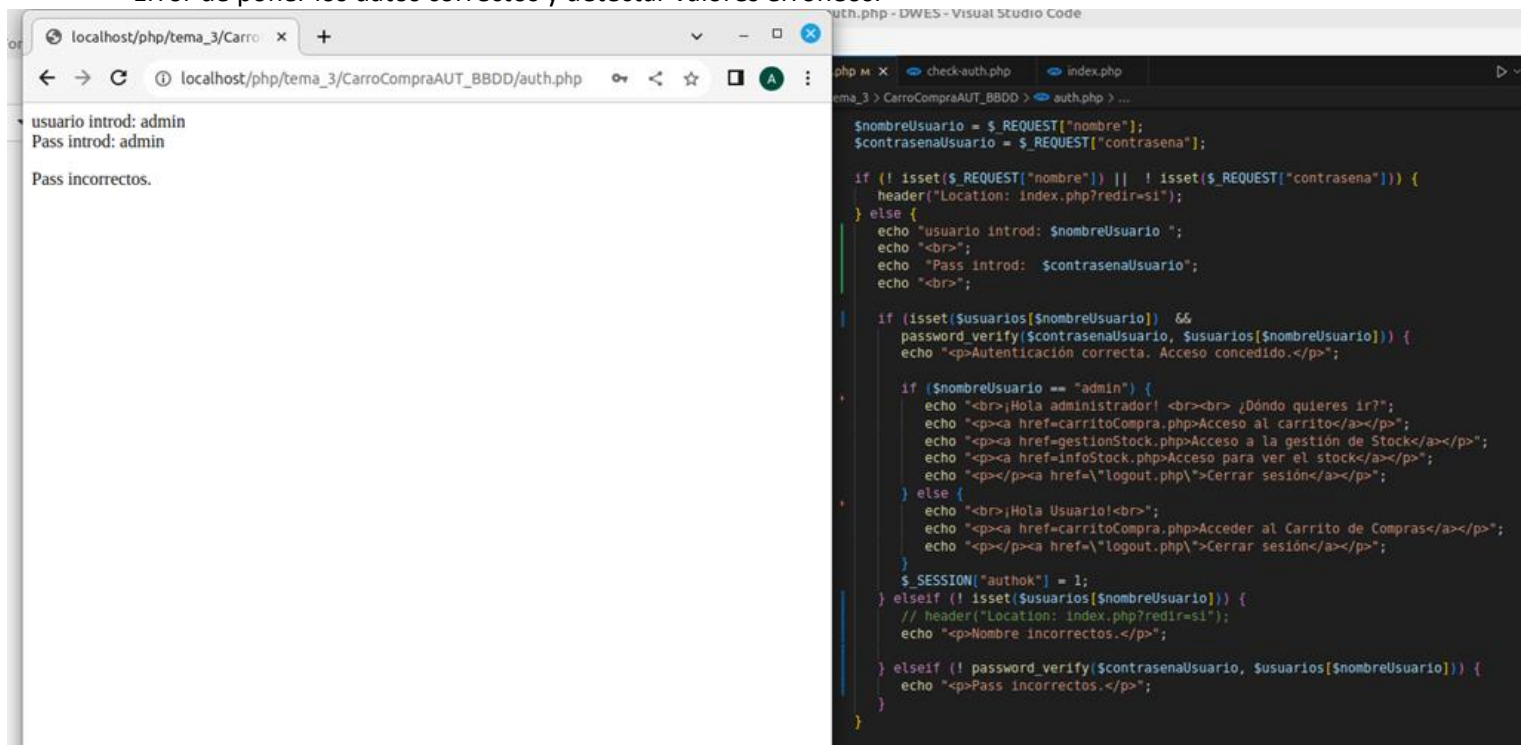
Por esta razón, en auth.php tuve que hacer varios echos para ver dónde estaba el fallo, si no cogía bien los valores del array, por si no estaba haciendo bien la asignación de valores de la bbdd en el array. Con estos echos, mostraba los valores introducidos y los valores almacenados, así pude comparar los valores y conseguí ver que el fallo estaba en que almacené mal las contraseñas hash.

Creé hashes nuevos y los puse en la bbdd, pero seguía saliendo el mismo error, y en el echo de contraseña almacenada, seguía saliendo la anterior, por lo que hice echo en stock.php de todos los usuarios y estas contraseñas las puse de nuevo en la bbdd, relacionadas correctamente. Pero no entiendo este error, pues en el script del sql, está puesto un drop database, por lo que cada vez que se inicie la aplicación se debería de borrar y crear nuevamente.

Por lo tanto envié el script sql manualmente a mysql, y luego la aplicación por fin funcionó.

## Capturas de pruebas:

- Error de poner los datos correctos y detectar valores erróneos:



-Accedo a mysql y veo que los hashes almacenados los introduje en usuarios que no son el suyo. En la pantalla de abajo-derecha vemos los hashes correctos con su usuario:



```

3
4 $frutas = [];
5
6 $pdo = new PDO("mysql:db=D90A.2158.HHZc." WHERE usuario = 'rodri';
7 Query OK, 1 row affected (0,01 sec)
8 if ($consulta = $pdo->query("SELECT * FROM usuarios WHERE usuario = 'rodri'")) {
9     while ($registro = $consulta->fetch()) {
10         $frutas = $registro;
11         $stock = $registro;
12         $almacen = $registro;
13         array_push($frutas, $registro);
14     }
15 }
16 else {
17     die("Problema accediendo a la base de datos");
18 }
19
20 foreach ($frutas as $fruta) {
21     if (!isset($_SESSION['usuario'])) {
22         if (isset($_REQUEST['enviar'])) {
23             foreach ($frutas as $fruta) {
24                 if ($REQUEST['usuario'] == $fruta['usuario'] && $REQUEST['password'] == $fruta['password']) {
25                     $_SESSION['usuario'] = $fruta['usuario'];
26                     $_SESSION['password'] = $fruta['password'];
27                     header("Location: index.php");
28                     exit();
29                 }
30             }
31         }
32     }
33 }
34
35 $frutas.sql
36 croquetaeria.sql
37
38 php > tema_3 > CarroCompraAUT_BBDD > croquetaeria.sql
39 CREATE USER administrador IDENTIFIED BY 'administrador';
40
41 USE croquetaeria;
42 GRANT ALL ON * TO administrador;
43
44 CREATE TABLE usuarios (
45     usuario varchar(20) NOT NULL,
46     pass varchar(70) DEFAULT NULL,
47     PRIMARY KEY (usuario)
48 ) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;
49
50 INSERT INTO usuarios VALUES
51 (
52     'admin', '$2y$10$/G6E0U2aoxM8EF/1vUu2BbiUY/BZhqaGivU9eMQsdLLYdxF1dC',
53     'arancha', '$2y$10$/tx0Af5xLmpk8w40pbHSA.VI1bM3ION.XUjhYUxNZimTnRakZH8YC',
54     'eva', '$2y$10$/RDJlk/wLwYkg4LEHfg0eS0ZWGEjoxDdGGApMdi8lJmabVVNPLeK91',
55     'rodri', '$2y$10$/orwHE8cTZH4JGDx/n4B0B05ZvokauYVqkVTRQfD9oA.2158.HHZc.'
56 );
57
58 stock.php
59 frutas.sql
60
61 php > tema_3 > CarroCompraAUT_BBDD > stock.php
62 // $admin = password_hash("admin", PASSWORD_DEFAULT);
63
64 // echo "Saran <br>";
65 // echo "Eva <br>";
66 // echo "Rodri <br>";
67 // echo "Admin <br>";
68 // Creo un array con usuarios y sus contraseñas hash
69 $usuarios = array(
70     "arancha" => "$2y$10$/tx0Af5xLmpk8w40pbHSA.VI1bM3ION.XUjhYUxNZimTnRakZH8YC",
71     "eva" => "$2y$10$/RDJlk/wLwYkg4LEHfg0eS0ZWGEjoxDdGGApMdi8lJmabVVNPLeK91",
72     "rodri" => "$2y$10$/orwHE8cTZH4JGDx/n4B0B05ZvokauYVqkVTRQfD9oA.2158.HHZc.",
73     "admin" => "$2y$10$/G6E0U2aoxM8EF/1vUu2BbiUY/BZhqaGivU9eMQsdLLYdxF1dC",
74 );
75
76 //array de productos con precio para comprar:
77 $productos = array(
78     "manzana" => 1.5,
79     "naranja" => 1.2,
80     "plum" => 1.8,
81     "uva" => 2.0,
82     "kiwi" => 1.5,
83     "melocoton" => 1.2,
84     "fresa" => 1.5,
85     "mandarina" => 1.2,
86     "limon" => 1.5,
87     "naranja" => 1.2,
88     "plum" => 1.8,
89     "uva" => 2.0,
90     "kiwi" => 1.5,
91     "melocoton" => 1.2,
92     "fresa" => 1.5,
93     "mandarina" => 1.2,
94     "limon" => 1.5,
95 );
96
97 You, hace 22 horas - clase 18

```

- Hago el cambio en la bbdd (abajo-derecha) y en auth pongo más echos para mostrar la contraseña almacenada y la introducida, y así ver que está correcto. Y por fin se hace la conexión, probando con admin:

```

localhost/php/tema_3/CarroCompr...
localhost/php/tema_3/CarroCompr...
usuario introd: admin
Pass introd: admin
Contraseña almacenada:
$2y$10$/G6E0U2aoxM8EF/1vUu2BbiUY/BZhqaGivU9eMQsdLLYdxF1dC
Contraseña introducida: admin
Autenticación correcta. Acceso concedido.
¡Hola administrador!
¿Dónde quieres ir?
Acceso al carrito
Acceso a la gestión de Stock
Acceso para ver el stock
Cerrar sesión

```

```

8
9
10 if (!isset($_REQUEST['nombre'])) || !isset($_REQUEST['contrasena']) {
11     header("Location: index.php?redir=1");
12 } else {
13     echo "usuario introd: $nombreUsuario ";
14     echo "<br>";
15     echo "Pass introd: $contrasenaUsuario";
16     echo "<br>";
17     echo "Contraseña almacenada: " . $usuarios[$nombreUsuario] . "<br>";
18     echo "Contraseña introducida: " . $contrasenaUsuario . "<br>";
19
20     if (isset($usuarios[$nombreUsuario]) &&
21         password_verify($contrasenaUsuario, $usuarios[$nombreUsuario])) {
22         echo "<p>Autenticación correcta. Acceso concedido.</p>";
23
24         if ($nombreUsuario == "admin") {
25             echo "<br>¡Hola administrador! <br><br> ¿Dónde quieres ir?";
26             echo "<p><a href='carritoCompra.php'>Acceso al carrito</a></p>";
27         }
28     }
29 }
30
31 croquetaeria.sql
32 stock.php
33 frutas.sql
34
35 php > tema_3 > CarroCompraAUT_BBDD > croquetaeria.sql
36 CREATE USER administrador IDENTIFIED BY 'administrador';
37
38 USE croquetaeria;
39 GRANT ALL ON * TO administrador;
40
41 CREATE TABLE usuarios (
42     usuario varchar(20) NOT NULL,
43     pass varchar(70) DEFAULT NULL,
44     PRIMARY KEY (usuario)
45 ) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;
46
47 INSERT INTO usuarios VALUES
48 (
49     'admin', '$2y$10$/G6E0U2aoxM8EF/1vUu2BbiUY/BZhqaGivU9eMQsdLLYdxF1dC',
50     'arancha', '$2y$10$/tx0Af5xLmpk8w40pbHSA.VI1bM3ION.XUjhYUxNZimTnRakZH8YC',
51     'eva', '$2y$10$/RDJlk/wLwYkg4LEHfg0eS0ZWGEjoxDdGGApMdi8lJmabVVNPLeK91',
52     'rodri', '$2y$10$/orwHE8cTZH4JGDx/n4B0B05ZvokauYVqkVTRQfD9oA.2158.HHZc.'
53 );
54
55

```

-Hago la prueba también con usuario “eva” sin privilegios de administrador, y también se hace conexión, comprobamos también que la contraseña y hash coincide:

The screenshot displays a web browser window on the left and a terminal window on the right. The browser shows the login page for 'CarroCompra' at localhost. The user 'eva' has successfully logged in, and the page displays a welcome message and links to the shopping cart and session management. The terminal window shows the PHP code for the login process and the SQL code for the database setup.

**Browser Output:**

```

usuario introd: eva
Pass introd: eva
Contraseña almacenada:
$2y$10$RDJlk/wLwYkg4LEHFgOeSOZWGEjoxDdGGApMDi8ljmabVVNPLek9i
Contraseña introducida: eva

Autenticación correcta. Acceso concedido.

¡Hola Usuario!

Acceder al Carrito de Compras
Cerrar sesión

```

**PHP Code (auth.php):**

```

8
9
10 if (!isset($_REQUEST["nombre"]) || !isset($_REQUEST["contrasena"])) {
11     header("Location: index.php?redir=s1");
12 } else {
13     echo "usuario introd: $nombreUsuario ";
14     echo "<br>";
15     echo "Pass introd: $contrasenaUsuario";
16     echo "<br>";
17     echo "Contraseña almacenada: " . $usuarios[$nombreUsuario] . "<br>";
18     echo "Contraseña introducida: " . $_POST["contrasena"] . "<br>";
19
20     if (isset($usuarios[$nombreUsuario]) &&
21         password_verify($_POST["contrasena"], $usuarios[$nombreUsuario])) {
22         echo "<p>Autenticación correcta. Acceso concedido.</p>";
23
24         if ($nombreUsuario == "admin") {
25             echo "<br>¡Hola administrador! <br><br> ¿Dónde quieres ir?";
26             echo "<p><a href='carritoCompra.php'>Acceso al carrito</a></p>";
27         }
28     }
29 }

```

**SQL Code (croqueteria.sql):**

```

8 CREATE USER administrador IDENTIFIED BY 'administrador';
9
10 USE croqueteria;
11 GRANT ALL ON * TO administrador;
12
13 CREATE TABLE usuarios (
14     usuario varchar(20) NOT NULL,
15     pass varchar(70) DEFAULT NULL,
16     PRIMARY KEY (usuario)
17 ) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;
18
19 INSERT INTO usuarios VALUES
20 ('admin', '$2y$10$/G6E0U2aoxM8EF/1v1Uu2Bb1UY/BZhqaGivU9eMQsdLLYdxFidC'),
21 ('arancha', '$2y$10$/tx0Af5xLmpk8w40pbHSA.VI1bM3ION.XUjhyUxNZImTnRakZH8YC'),
22 ('eva', '$2y$10$RDJlk/wLwYkg4LEHFgOeSOZWGEjoxDdGGApMDi8ljmabVVNPLek9i'),
23 ('rodrigo', '$2y$10$/wHE8cTZH4JGDX/n480805ZvokauYVqkVTRQfD9oA.2158.HHZc');

```

### Comentarios y conclusiones:

Con esta práctica, hemos podido entender que la implementación de un sistema de autenticación segura es crucial para proteger áreas de una aplicación que son más sensibles.

El uso de funciones de hashing como `password_hash()` proporciona un extra de seguridad.

La persistencia de sesiones, que aprendimos en prácticas anteriores, y el manejo adecuado de la información de autenticación son esenciales para que el usuario pueda hacer un uso fluido y seguro de la aplicación.

La práctica también resalta la importancia de la gestión segura de contraseñas para proteger la transmisión de datos.

## **OPCIONAL PARTE 5: PRUEBA Y DOCUMENTACIÓN AUTENTICACIÓN CON HTTP BASIC**

He probado ambas opciones, la de hash y la de texto plano, pero voy a comentar la de hash, ya que la función de ambas app son las mismas, solo cambia el tipo de contraseña.

***\*\*He añadido comentarios en el propio código. Mis comentarios están en mayúsculas.***

Contamos con los archivos `auth-con-hash.php`, para la autenticación, y `logout-hash.php`, para el cierre de sesión, que permiten implementar un sistema de autenticación y cierre de sesión utilizando autenticación básica HTTP en PHP.

### **1. Autenticación (auth-con-hash.php):**

- Primero se establece usuario como pepe en `$validUser` y contraseña en formato hash en `$validHash`, y se utiliza un `$random` para evitar problemas de caché.
- Este script verifica si se han proporcionado credenciales mediante la autenticación básica HTTP.
- Si no se han proporcionado credenciales, se envía un header de autenticación básica y sale ventana emergente para introducir claves, y si damos a cancelar se muestra un mensaje de "Acceso denegado".
- Si las credenciales son correctas (usuario y contraseña coinciden con `$validUser` y `$validHash`), se muestra un mensaje de bienvenida.
- Si las credenciales son incorrectas, se muestra un mensaje de "Acceso denegado" junto con un enlace para volver a intentar.

### **2. Cierre de Sesión (logout-hash.php):**

- Cuando un usuario hace clic en el enlace de cierre de sesión (`logout-hash.php`), se redirige a esta página.
- El código de `logout-hash.php` fuerza la solicitud de credenciales al establecer `$_SESSION['auth']`.
- Se envía un encabezado de autenticación básica si no se han establecido credenciales, o si las credenciales no coinciden con las válidas y se muestra un mensaje de "Acceso restringido".
- Se proporciona un enlace para volver a la página anterior (almacenada en `$_SERVER['HTTP_REFERER']`).

## **DATOS:**

Como tenemos las variables de usuario y contraseña en los dos archivos, estas variables `$validUser` y la contraseña en formato hash `$validHash` deben coincidir entre ambos archivos para que la autenticación y el cierre de sesión funcionen correctamente.

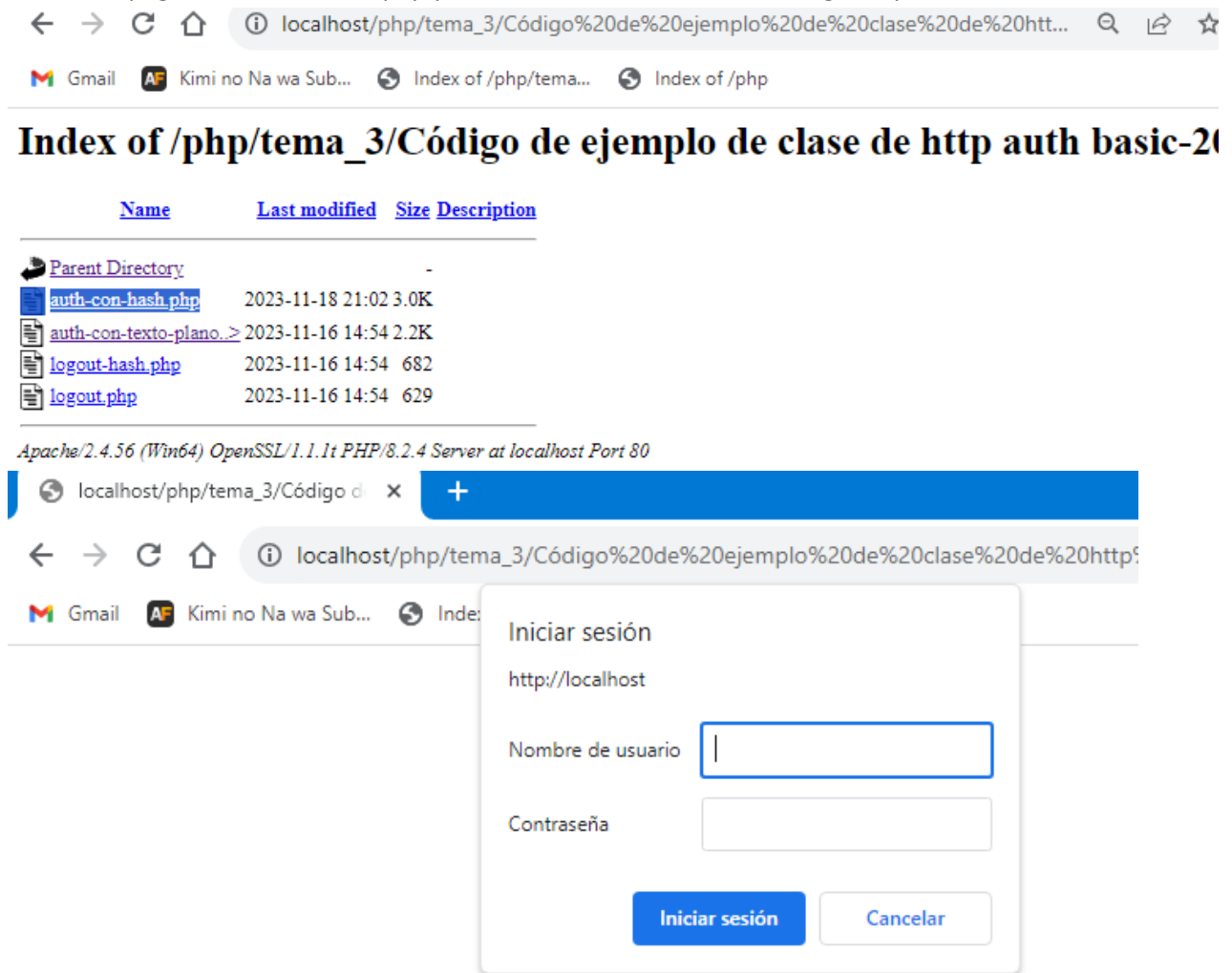
El usuario puede iniciar sesión utilizando `auth-con-hash.php` y, después, cerrar sesión haciendo clic en el enlace proporcionado por `logout-hash.php`.

Los enlaces usados con parámetros `id=$random`, es para evitar problemas de caché y asegurarse de que cada solicitud sea única.

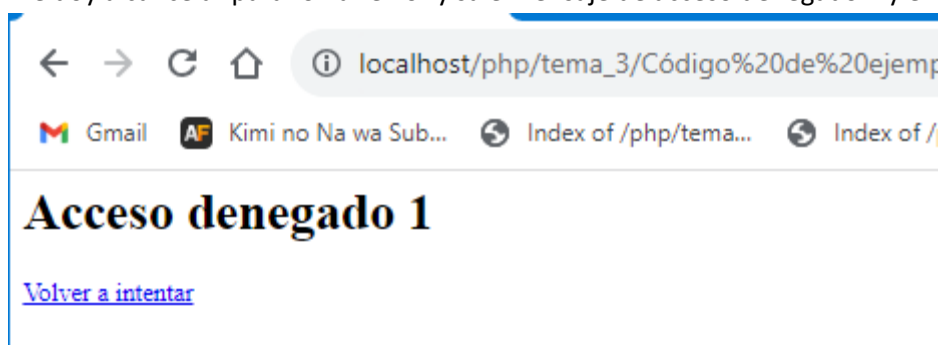
Esta opción de autenticación basic me gusta, de esta manera no tienes que crear un formulario a parte, y es la propia página la que te pide las claves en formato de ventana emergente.

**CAPTURAS DE PRUEBAS:**

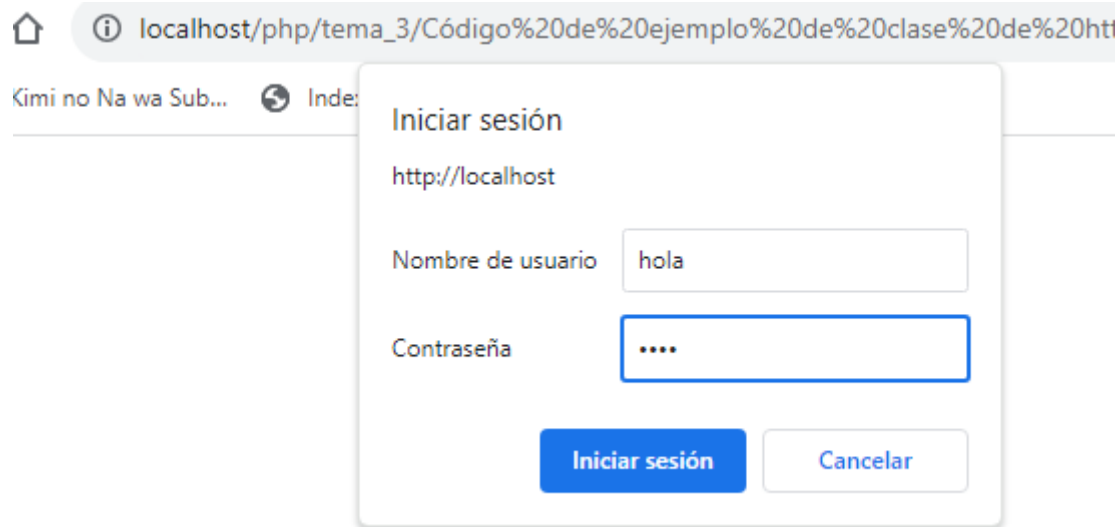
-Accedo la página auth-con-hash.php y directamente sale ventana emergente pidiendo claves:



- Le doy a cancelar para forzar error y sale mensaje de acceso denegado 1 y enlace para volver a intentar:



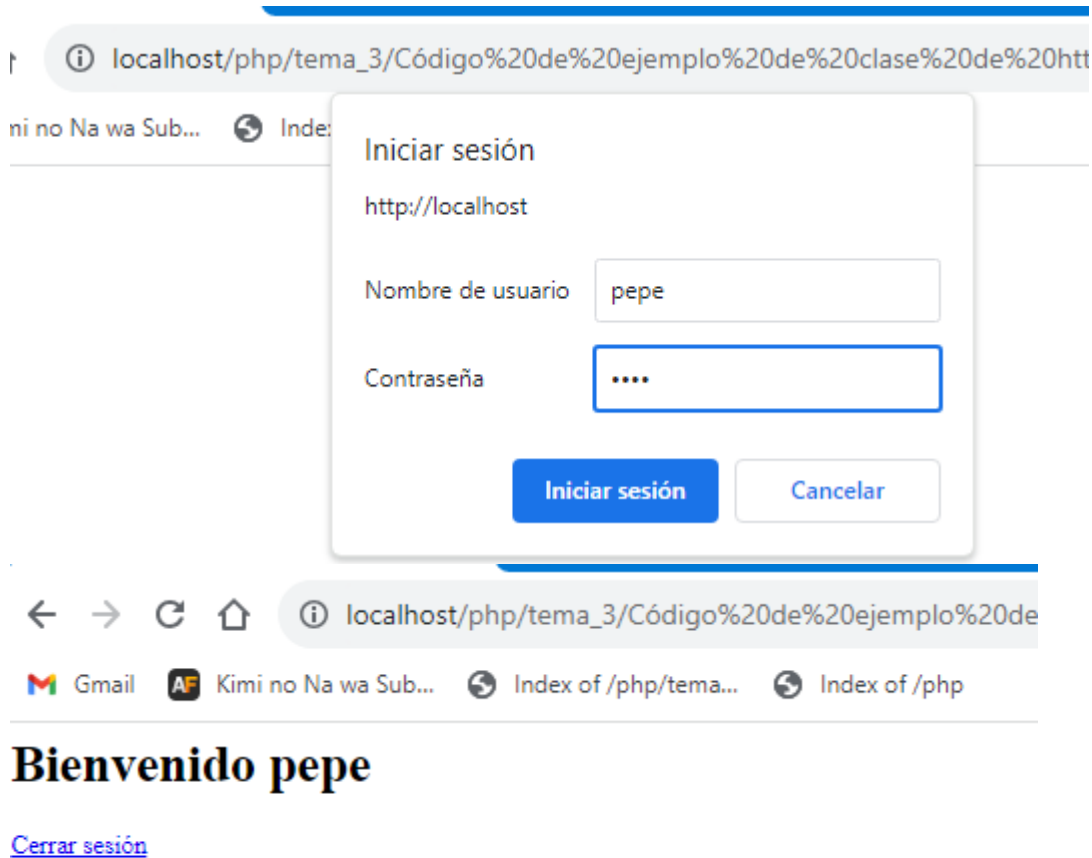
- Aparece otra vez la ventana con las claves, y ahora pruebo con claves erróneas para forzar otro error:



- y al iniciar sesión, sale mensaje de acceso denegado y enlace para volver a intentar:



- Ahora pruebo con las claves correctas e inicio sesión:



- y si cierro sesión, aparece nuevamente la ventana emergente para solicitar claves.