Centro Industrial y Desarrollo
Empresarial Soacha
Regional Cundinamarca

GUIA DE DESARROLLO WEB PAUTAS BASICAS

GESTION DE USUARIOS

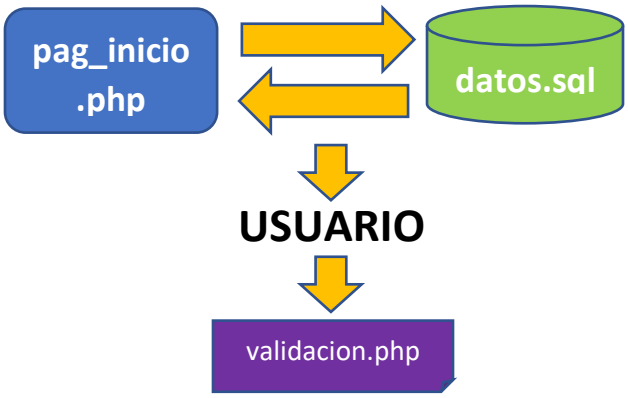
Gestionar recordación de usuario usando cookies

Para este ejercicio realizaremos un formulario de acceso, pero este debe permitirle al usuario que elija si desea o no generar la recordación de sus datos de usuario en el equipo desde el cual está accediendo al sistema.

- Si el usuario no valida la opción recordar credenciales de usuario, entonces cada vez que necesite acceder al sistema debe incluir sus datos.
- Si el usuario valida la opción recordar credenciales de usuario, entonces cada vez que necesite acceder al sistema no debe incluir sus datos ya que sistema a través de la cookie creada guardara sus datos, según el tiempo que se le establezca a la misma.

CREACION DE LOGIN CON PDO

Despliegue



En la base de datos ***datos.sql***, cree una tabla llamada ***datos_usuario*** que tenga los siguientes atributos un identificador único (primary key) para cada usuario, se debe autoincrementar cada vez que un nuevo usuario se registre, usuario, contraseña, recuerde que todos los datos son obligatorios, por tanto valida que ningún dato quede vacio(NULL)

CREATE TABLE datos_usuario (id_usuario INT NOT NULL AUTO_INCREMENT PRIMARY KEY, usuario VARCHAR(20) NOT NULL, password VARCHAR(20) NOT NULL)

inserte por lo menos un usuario de prueba

INSERT INTO `datos_usuario`(`id_usuario`, `usuario`, `password`) VALUES ('1','adrianae@gmail.com','12345')

Dentro de desarrollo web cree una carpeta llamada parte 3 y en ella dos carpetas una llamada sesiones y la otras llamada cookie_usuario

En la carpeta sesiones copie el archivo pag_español.php usado en la parte 2 de este workshop para generar la cookie de idioma y renombrelo como pag_inicio.php

cree un formulario de acceso al sistema llamado form_acceso.php este debe tener una casilla para ingresar el usuario que será de type email y una contraseña que será de type password para que oculte los caracteres ingresados.

ACCESO AL SISTEMA

Usuario:

Contraseña:

ENVIAR

aplique un estilo básico dentro del mismo formulario

```
<style>

    form input[type="submit"] {
        margin-left: 44px;
        text-decoration: none;

        width: 100px;
        height: 30px;
        border: 1px solid #d7d7d7;
        background: #8F8E8C  ;
    }
    form input[type="password"],form input[type="email"]
    {
        display: table-cell;
        width: 300px;
        height: 20px;
        border: 1px solid #d7d7d7;
        border-radius: 3px;
    }
</style>
```

Recuerde en el action de la etiqueta form especificar que nos debe redirigir a la misma página con la función $_SERVER para que cuando el usuario pulse el botón de enviar, la acción del formulario realizase una petición al servidor de recargar la página en la que nos encontramos.

```
<br><br><br> <h4> ACCESO AL SISTEMA</h4>
<form action="<?php echo $_SERVER['PHP_SELF']; ?>" method="post">
    <div>
        <label>Usuario: </label>
        <input type="email" name="correo"><br>

        <label>Contraseña: </label>
        <input type="password" name="password"><br><br>

        <input type="submit" name="enviar" value="ENVIAR" >
    </div>
</form>
```

Copie el archivo pag_español.php usado en la cookie de idioma y renombrelo como **pagina_inicio.php** en este empezaremos a trabajar para diseñar una pagina de acceso desde donde el usuario podrá logearse, el sistema le dará la bienvenida al usuario

lo primero que debe hacer es agregar el formulario de acceso a la pagina de inicio. Para ello genere un código php antes del header donde debemos usar la librería PDO(extensión Objetos de Datos de PHP), PDO proporciona una capa de abstracción de acceso a datos, lo que significa que, independientemente de la base de datos que se esté utilizando, se emplean las mismas funciones para realizar consultas y obtener datos. Esta librería la debe usar en este caso para generar la conexión y validar los errores de conexión.

crea un **try** esto es un bloque de código que cuando una excepción es lanzada, el código siguiente a la declaración no será ejecutado, y PHP intentará encontrar el primer bloque catch coincidente. código puede estar dentro de un bloque **try** para facilitar la captura de excepciones potenciales. Cada bloque **try** debe tener al menos un bloque **catch** o finally correspondiente. Este try en caso que la conexión a la base de datos no sea exitosa permitirá decirle al sistema que debe hacer. Dentro del catch debemos crear una excepción para el caso que no conecte debe cerrar cualquier proceso que se este ejecutando así:

```
}catch(Exception $e){
 die("Error: " . $e->getMessage());
```

En el try será en caso que todo conecte exitosamente. Primero genere la conexión a la base de datos usando la clase PDO, crea una variable $base para llamar la base de datos y genera la ruta, en este caso con acceso local desde el localhost, recuerde que el nombre de la base de datos es datos.sql. Ahora el try debe validar si la información que el usuario ingreso al formulario se encuentra en la base de datos, para ello configure la validación con PDO usando las función **setAttribute** cree una consulta preparada usando la función **prepare.** Recuerde que en los anteriores ejercicios usábamos la función $_GET, en este caso usa la función $_POST la cual envía los datos a través de un array asociativo para poder vincular los datos. Este código debe ubicarlo en el header antes de la etiqueta **h1** del titulo del blog

```php
try{
    $base=new PDO("mysql:host=localhost; dbname=datos" , "root","");
    $base->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
    $sql="SELECT * FROM datos_usuario WHERE usuario= :login AND PASSWORD= :password";
    $resultado=$base->prepare ($sql);
    $login=htmlentities(addslashes($_POST["correo"]));
    $password=htmlentities(addslashes($_POST["password"]));
    $resultado->bindValue(":login",$login);
   // $password=password_hash($password->getClave(),PASSWORD_DEFAULT);
    $resultado->bindValue(":password",$password);
    $resultado->execute();
    $numero_registro=$resultado->rowCount();
    if($numero_registro!=0){
        //se inicia la sesion si el usuario esta registrado
        session_start();
        $_SESSION["usuario"]=$_POST["correo"];
        header("location:pag_inicio.php");

    }else{
        header("location:form_acceso.php");
    }

}catch(Exception $e){
 die("Error: " . $e->getMessage());
```

conexión a la base de datos

consulta base de datos los datos ingresados y valida si existe o no con los marcadores login y password ingresados en el formulario login.php

creamos un alias en la consulta SELECT para el correo y la contraseña consultada en la base de datos ya que tanto en el usuario= :login como en la contraseña= :password debe usar la función **htmlentities** la cual permite cambiar todos los símbolos ingresados en esos campos en código html, también use la funcion **addslashes** que omite cualquier carácter extraño ingresado en el login.

Luego de ejecutar la consulta debemos generar una estructura de control que permita validar que si la consulta fue exitosa debe activar una sesion para este usuario. esto se hace con el usoa de la funcion $_SESSION la cual pide como parametro el capo que almacena el nombre de usuario al cual se le activara la sesion y este de ser traido desde el formulario, en caso que la sesion se active solo direccionaremos a que la pagina de inicio se recargue y en caso contrario recargara el formulario de acceso para que el usuario ingrese nuevamente sus datos. Hasta aquí solo hemos generado que si el usuario activa una sesion y que ara en caso que sea correcto.

Como sabemos al ejecutarse un programa la lectura del mismo es de arriba hacia abajo por tanto al agregarlo al iniciar el body estaría validando los datos correo y password que el usuario todavía no ha enviado, por tanto el método **POST** no tendría datos que capturar y generara un error.

Para evitar que esto pase y podamos ejecutar la validación desde la misma pagina debemos crear un condicionamiento para que verifique si el usuario no ha registrado sus datos continúe y cargue la pagina. Este debe encerrar toda la validación del formulario, por tanto ubíquelo y ábralo antes de try y ciérrelo antes del cierre la etiqueta de php como se observa en la siguiente imagen :

```php
    if(isset($_POST["enviar"])){
    try{
        $base=new PDO("mysql:host=localhost; dbname=datos" , "root","");
        $base->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
        $sql="SELECT * FROM datos_usuario WHERE usuario= :login AND PASSWORD= :password";
        $resultado=$base->prepare ($sql);
        $login=htmlentities(addslashes($_POST["correo"]));
        $password=htmlentities(addslashes($_POST["password"]));
        $resultado->bindValue(":login",$login);
       // $password=password_hash($password->getClave(),PASSWORD_DEFAULT);
        $resultado->bindValue(":password",$password);
        $resultado->execute();
        $numero_registro=$resultado->rowCount();
        if($numero_registro!=0){
            //se inicia la sesion si el usuario esta registrado
            session_start();
            $_SESSION["usuario"]=$_POST["correo"];
            header("location:pag_inicio.php");

        }else{
            header("location:form_acceso.php");
        }

    }catch(Exception $e){
     die("Error: " . $e->getMessage());
    }
}
?>
```

Ahora para identificar que el usuario ya accedió al sistema, la pagina de inicio no se debe mostrar el formulario de acceso, ya que el usuario esta con una sesión activa. Para realizar esto use la etiqueta **include**. La sentencia **include** incluye y evalúa el archivo especificado. Cree también un condicionamiento donde muestre el formulario de acceso solo cuando el usuario no ha validado sus datos o accedido al sistema.

Se debe validar a través de un **if** que si el usuario no ha iniciado sesión por tanto debe cargar el formulario de acceso; si ya valido los datos deje de mostrar el formulario de acceso y darle la bienvenida al usuario para que este sepa que accedió al sistema.

```php
  <?php
//valida si se a iniciado una sesion si no hay una sesion iniciada incluye el formulario de acceso
if(!isset($_SESSION["usuario"])){
include("form_acceso.php");
}else{
    echo"BIENVENIDO AL SISTEMA". $_SESSION["correo"];
}
?>
```

Dentro del código que se estableció para validar los datos con PDO verifique el condicionamiento establecido, ya que este no dará los resultados que necesitamos porque está usando el enrutamiento a pag_inicio.php y nosotros queremos que todo se cargue dentro de la misma página. condicione que si el usuario digita datos los datos correctos se activa la sesión y si los datos con incorrectos le genere un mensaje de error.

```php
<?php

 if(isset($_POST["enviar"])){
  try{
      $base=new PDO("mysql:host=localhost; dbname=datos" , "root","");
      $base->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
      $sql="SELECT * FROM datos_usuario WHERE usuario= :login AND PASSWORD= :password";
      $resultado=$base->prepare ($sql);
      $login=htmlentities(addslashes($_POST["correo"]));
      $password=htmlentities(addslashes($_POST["password"]));
      $resultado->bindValue(":login",$login);
     // $password=password_hash($password->getClave(),PASSWORD_DEFAULT);
      $resultado->bindValue(":password",$password);
      $resultado->execute();
      $numero_registro=$resultado->rowCount();
      if($numero_registro!=0){
          //se inicia la sesion si el usuario esta registrado
          session_start();
          $_SESSION["usuario"]=$_POST["correo"];
         // header("location:pag_inicio.php");

      }else{
         // header("location:form_acceso.php");
         echo "LOS DATOS DE USUARIO SON INCORRECTOS";
      }

   }catch(Exception $e){
    die("Error: " . $e->getMessage());
   }
}
?>
```

Observe los resultados esperados del ejercicio. Si ingresa datos incorrectos valida que los datos son incorrectos

Ahora si ingreso los datos correctos debe desaparecer el formulario de acceso y darme la bienvenida al sistema

Crear cookies para que almacene los datos del usuario

en la carpeta llamada cookie_usuario copie los archivos generados en el ejercicio de sesiones. Edite el archivo form_acceso.php agregue una casilla de verificación para que el usuario seleccione si desea que sus credenciales de usuario sean recordadas.

```
<form action="<?php echo $_SERVER['PHP_SELF']; ?>" method="post">
    <div>
        <label>Usuario: </label>
        <input type="email" name="correo"><br><br>

        <label>Contraseña: </label>
        <input type="password" name="password"><br><br>

        <input type="checkbox" name="recordar">
        <label>Recordarme en este equipo </label><br><br>

        <input type="submit" name="enviar" value="ENVIAR" ><br><br>
    </div>
</form>
```

Desde el archivo **pagina_inicio.php** en este empezaremos a trabajar para diseñar una pagina de acceso desde donde el usuario podrá logearse, elegir si desea almacenar sus datos de usuario en la

terminal también crearemos un área de contenido que solo podrá visualizar si accedió al sistema como un usuario registrado al acceder con el perfil de usuario que le habilitará las opciones de buscar productos que trabajamos en la parte 1 de este workshop y agregar productos que trabajamos en la parte 2 de este workshop

A través de la cookie almacenamos los datos de usuario y si queremos cargar nuevamente la página al digitar la URL esta debe mostrar el contenido de la página del perfil del usuario.

Ahora en la pagina de inicio antes de iniciar la validación del usuario con PDO crea una variable de tipo booleana para guardar la información que el usuario se ha logeado correctamente.

```php
<?php
$autenticar=false;
```

Por que hacemos esto? debido a que cuando inicia el programa el debe hacer dos validaciones:

1. validar si el usuario de logeo correctamente y darle acceso al sistema
   Esta variable almacenara si el usuario de logeo de forma correcta por eso necesitamos que sea booleana ya que si el se logea de forma correcta la variable $autenticar sera verdadera.

```php
if($numero_registro!=0){
 //validar si el usuario se a logeado de forma correcta
 $autenticar=true;
```

2. validar si el usuario selecciono la opción de recordar sus datos de usuario y crear la cookie para ellos usamos la función SETCOOKIE()

```php
//validar si el usuario activo la casilla de verificacion
   if(isset($_POST["recordar"])){
      //crear la cookie por 15 dias
      setcookie("datos_usuario", $_POST["correo"],time()+1296000);
```

para ello modificamos el if que teníamos anteriormente en la validación y estructuramos el código anterior el cual lo que ara es:

- Si un usuario accede por primera vez al sistema y se logea pero no marca la casilla de verificación accede al sistema y cada vez que desee acceder el sistema le pedirá que se logee.

- si aparte de loguearse activa la casilla de verificación pues guardara su datos de usuario y al acceder nuevamente el sistema ya no le pedirá que se logee.

para que puedas continuar trabajando y no tener que crear usuarios diferentes ya que la cookie creada evitara que acceda a la pagina de inicio desde el login crea un archivo para destruir la cookie llamado ***eliminar_cookie.php***

```php
<?php
    setcookie("datos_usuario", "lola@gmail.com", time()-1);
    echo "LA COOKIE FUE ELIMINADA CON EXITO"

?>
```

como lo que estamos guardando en la cookie es el usuario debes agregar el dato el usuario con el que vas a hacer las pruebas.

Ahora debemos ir nuevamente al código de la página de inicio y modificar la inclusión del formulario y cree un código que permita ver si el usuario se a autenticado correctamente y si la cookie se creo

```php
if($autenticar==false){
    if(!isset($_COOKIE["datos_usuario"])){
        include("form_acceso.php");
    }
}
?>
```

con este código lo que se hace es que si un usuario accede a la pagina de inicio por primera vez como no se ha logeado entonces $autenticar=false por tanto no hay cookie creada porque es la primera vez que intenta acceder, por lo que cargara el formulario de acceso.

Ahora debemos personalizar el entorno de usuario que se ha logeado es decir el contenido que solo aparecerá para el usuario cuando se logee correctamente y pertenece al perfil indicado el cual en su menú podrá tener habilitada privilegios para la gestión de la base de datos.

cree un archivo llamado ***perfil_usuario.php*** y diseñe un menú desplegable para el usuario logeado pueda realizar actividades propias de su perfil como consultar un producto o agregar productos a la base de datos

```html
<nav>
    <ul class="menu">
                    <li><a href="#">Procesos</a>
        <ul class="submenu">
            <li><a href="\desarrollo web1\parte 1\busqueda.php">Buscar productos</a></li>
            <li><a href="/desarrollo%20web1/parte%202/productos/form_registro.php">Agregar productos</a></li>
        </ul>
    </li>
    <li><a href="pag_inicio.php">Inicio</a></li>
    <li><a href="#">Nosotros</a></li>

    <li><a href="#">Contacto</a></li>
    </ul>
</nav>
```

cree un estilo sencillo y guárdelo en la carpeta css como style_perfil.css

```css
* {
    margin: 0;
    padding: 0;
    -webkit-box-sizing: border-box;
    -moz-box-sizing: border-box;
    box-sizing: border-box;
}

header {
    width: 100%;
}

.menu>li {
    position: relative;
    z-index: 1;
    display: inline-block;
}

.menu>li>a {
    display: block;
    padding: 15px 20px;
    color: #eee7e7;
    font-family: 'Open sans';
    text-decoration: none;
    text-size-adjust: 16%;
}

.menu li a:hover {
    color: #CE7D35;
    transition: all .3s;
}
```

```css
/* Submenu*/

.submenu {
    position: absolute;
    z-index: 1;
    background: #333333;
    width: 120%;
    visibility: hidden;
    opacity: 0;
    transition: opacity 1.5s;
}

.submenu li a {
    display: block;
    padding: 15px;
    color: #fff;
    font-family: 'Open sans';
    text-decoration: none;
}

.menu li:hover .submenu {
    z-index: index;
    visibility: visible;
    opacity: 1;
}
```

Ahora diríjase a la página de inicio estructure el código para validar si el usuario se logeea correctamente o existe una cookie creada y le permita acceda al sistema a realizar procesos de su perfil de usuario.

para ello cree un if que valide si el logeo a tenido éxito o existe un cookie creada para ese usuario para ello use el operador lógico O(representado así ||). Ubíquelo después del nav de la pagina de inicio.

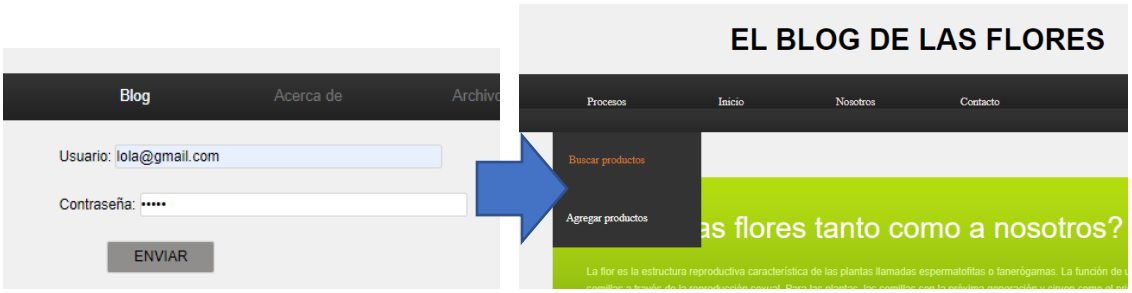
```php
<?php
//insertamos el menu de tareas para el usuario que accedio correctamente
if($autenticar==true || isset($_COOKIE["datos_usuario"])){
    include("perfil_usuario.php");
}
?>
```

ahora el sistema valida si el usuario ingresa sus credenciales de forma correcta lo envía al perfil de usuario o si existe una cookie al poner la url de inicio ya no le pedirá que se logee sino que será redireccionado al perfil de usuario que incluye el menú desplegable para que el usuario pueda buscar o agregar productos

verifiquemos el resultado del ejercicio

Primer caso el usuario se logea con datos erróneos el sistema los debe validar y retornar el formulario de acceso y mostrar el mensaje de error

Segundo caso el usuario accede por primera vez se logea pero no activa la caja de verificación por tanto accede al sistema pero no crea la cookie por lo que si cierra la página y luego más tarde intenta acceder pues deberá loquearse



Tercer caso el usuario accede a la página de inicio se logea y también marca la casilla de verificación accede al sistema y la cookie fue creada por tanto si cierra la página y luego intenta acceder nuevamente digita la URL el programa le mostrara al perfil del usuario.

si el usuario intenta acceder con la URL no le pedirá que se logee e ingresara al perfil de usuario

Para acceder nuevamente con el mismo usuario desde el login puedes eliminar la cookie ejecutando el archivo ***eliminar_cookie.php***