Index.php

Tendrá en cuenta 3 formas de acceder:

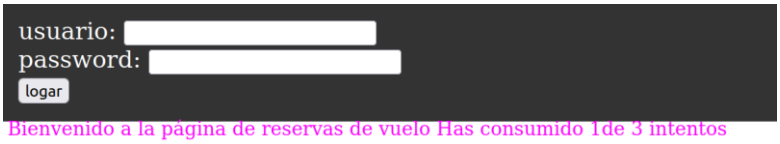
1. No estás logado y es la primera vez que lo intentas: Muestra formulario de logado.

Bienvenido a la página de reservas de vuelo

2. Te has logado. Accede a la aplicación mostrando el perfil y enlace desconexión.

Aquí aparecerá la tabla con tus vuelos contratados

3. Has fallado en el logado indicará el numero de intentos.



Bienvenido a la página de reservas de vuelo Has consumido 1de 3 intentos

4. Has consumido el número de intentos, el acceso a la página queda bloqueada durante x tiempo. En nuestro caso 20 segundos desde el último intento

Index.php debe controlar:

Bienvenido a la página de reservas de vuelo !!!!Bloqueado!!!!!!>/b> Recargue la página después de 20 segundos

- No se está logado: Debemos controlar que no se este bloqueado. Mostrará el formulario de logado en la parte de cabecera y un mensaje de entrada a la aplicación en la parte de contenido
- Está logado: Mostrará la cabecera de perfil y cierto contenido visible por usuario logado. (No implementado).

```
//Contenido de Header
 <?php
      if (!Logar::isLogado()) //No está logado
      {
         if(Logar::desbloquado()>=0)//No está bloqueado
           require_once("VISTA/logado.php"); //Formulario de Logado
      }
      else //Está logado, por lo que recupero el objeto tipo User guardado en Sesión

      {   $user= unserialize($_SESSION["usuario"]);
```

```
        require_once("VISTA/perfil.php");//Capa de Perfil
      }
      ?>
   ///Contenido del content
     <?php
     if (!Logar::isLogado()) //No está logado muestro contenido general
     {
        echo "<span>Bienvenido a la página de reservas de vuelo</span>";
        if (($intentos=Logar::desbloquado())>=0) //No está logado y no está bloqueado. Recupero
intentos para mostralos
        {
                echo "<span> Has consumido ". $intentos .  "de 3 intentos</span>";
        }
         else// No está logado pero si bloqueado
        {
        echo "<span> <b>!!!!Bloqueado!!!!!!>/b> Recargue la página después de 20 segundos
        </span>";
        }
     }
      else //Está logado
        echo "Aquí aparecerá la tabla con tus vuelos contratados";
     ?>
```

CONTROL /class

User.php: Representación del usuario. Id, nombre y password con su constructor, getter y setter.
Un usuario con nombre y password===> Proceso de Logado.
Un usuario con nombre, passwor e id======> Logado ok

Logar.php: Clase estática, sin propiedades, implementa todas las funciones para el proceso de Logado.

```
class Logar {

   static public function logar($nombre, $passwd)
```

Método encargado de comprobar en la BBDD las credenciales de logado, crea un objeto tipo usuario con el id a null, invoca al modelo accediendo a la BD a través del método consultar el cual devolverá un objeto User con id si son correctas las credenciales, sino, devulelve null.(No debe haber logado)
Si ha devuelto un objeto User es que hay ido todo bien por lo que almaceno dicho objeto como variable de Sesión (fijaros que lo serializo)

```
   {
     $log=new Logado(); //Instancio el Modelo
     $usr=$log->consultar(new User($nombre, $passwd, null)); //Invoco al método consultar

     if(!is_null($usr))//Logado ok
     {
        $_SESSION["usuario"]= serialize($usr);
    }
     else// Logado no OK
     {  unset($_SESSION["usuario"]);
```

```
        session_destroy(); //Acabo con la sesión
        if(array_key_exists("intentos", $_COOKIE)) //Activar y actualizar cookie de cuenteo
intentos
        {
            setcookie ("intentos",$_COOKIE["intentos"]+1 , time()+ 10, "/");
        }
        else
        {
            setcookie ("intentos",1 , time()+ 10, "/");
        }
      }
   }
```

Nota: isset($_COOKIE) suele lanzar un warnig o en versiones posteriores a PHP 7 un error, por lo que la función que utiliza es **array_key_exists("intentos", $_COOKIE)),** es decir si está definido la clave "intentos" en el array $_COOKIE.

static public function isLogado()

Devuelve si está definida la variable de sesión usuario, es decir si el usuario está logado

```
   {    if(isset($_SESSION["usuario"]))
          return true;
        else
          return false;

   }
```

static public function desLogar()

Termina con la sesión

```
   {
      unset($_SESSION["usuario"]);
      session_destroy();
   }
```

static public function desbloquado()

Comprueba si ha habido mas de cuatro intentos de logado dentro del tiempo acordado en nuestro caso son 20 segundos. Este tiempo se le ha indicado en el método logar cuando hemos creado la cookie.

Devuelve el número de intentos. Si el número de intentos es -1 es que ya ha habido mas de 3 intentos e indica que está bloqueado.

```
   {
      if(!array_key_exists("intentos", $_COOKIE))
            $int=0;
      else if (array_key_exists("intentos", $_COOKIE) && $_COOKIE["intentos"]<4)
            $int=$_COOKIE["intentos"];
      else
            $int=-1;
      return $int;
   }
}
```

CONTROL/ControlLogar.php es el controlador encargado de Logarse o deslogarse. Hace uso de la clase anterior Logar.php.

Este controlador puede ser invocado por:

- el método GET==> Desde el enlace de cerrar sesión, y realiza el deslogado.
- El método POST==>Desde el formulario y reliza el logado.

```
session_start();
if($_SERVER['REQUEST_METHOD']=="GET")//Cierre de sesión redirige a index.
{
   Logar::desLogar ();
    header("Location: http://localhost:8080/cookies2/index.php");
}
else //Proceso de Logado
{    if(!Logar::isLogado())
   {
      Logar::logar($_POST["nombre"], $_POST["password"]);//Llama al método logar de la clase Logar paśandole los parámetros recibidos.
   }
   if(Logar::desbloquado()!=-1)  //Comprueba que puede intentar seguir logándose
         header("Location: http://localhost:8080/cookies2/index.php");
   else //Destruye la sesión por eso llama a deslogar
      Logar::desLogar ();
      echo "bloqueado";
}
```