En esta práctica se trata de realizar una pequeña aplicación como prueba de concepto de todo lo aprendido en este tema. Te sugiero seguir los siguientes pasos:

1. Elije un tema que te guste para hacer la aplicación que mantendrá vía web el stock de una serie de productos de tu elección.

   El tema que voy a escoger es una base de datos sobre una tienda de juegos de mesa. Para ellos me voy a centrar en diferentes títulos de juegos de mesa reales. Voy a realizar una base de datos básica y después me plantearé ampliarla con más campos cuando sea funcional.

2. Diseña la base de datos: puede ser una sola tabla (suficiente) Crea un archivo de texto con los comandos para la creación de la base de datos, usuario, tablas y datos iniciales, de manera que con un comando

   sudo mysql < base_de_datos.txt

   se cree todo lo necesario.

   Siguiendo la teoría del año pasado use lo aprendido en BBDD para crear una base de datos que contuviera un tabla en la que estuvieran los siguientes campos:

   Nombre del juego, descripción y stock. Primero la hice en un Word donde introduje los parámetros por si tenia que hacer alguna modificación. En esta modificación tuve que quitar las comillas en la descripción ya que en SQL las comillas me suponían un problema.

Listado de Juego de Mesa

Juego	**Descripción**	**Stock**
Virus	Tu misión consiste en enfrentarte con arrojo a la pandemia y competir por ser el primero en erradicar los **virus** logrando aislar un cuerpo sano para evitar la propagación de las terribles enfermedades. Éticos o no, todos los medios a tu alcance valen para ganar	25
Monopoly	Monopoly es un juego de mesa basado en el intercambio y la compraventa de bienes raíces	38
Catan	**Catán** es un **juego** de mecánica sencilla. Sobre un tablero que cambia su disposición en cada partida, cada jugador debe construir caminos, pueblos y ciudades y acumular diferentes cartas. Cada uno de estos elementos otorga ciertos puntos y el que llegue a 10 gana	15

Una vez realizado esto introduje los comandos para crear la tabla y la volqué sobre sql para comprobar que todo iba bien. Y en efecto fue así. Adjunto archivo donde se puede ver ambos procesos.

```sql
//Creación de base de datos y borrar si hay otra

DROP DATABASE IF EXISTS juegodemesa;

CREATE DATABASE juegosdemesa;

USE juegosdemesa;

CREATE TABLE juegos(
nombre VARCHAR(100) PRIMARY KEY,
descripcion VARCHAR(1000),
stock int
);

INSERT INTO juegos(nombre,descripcion,stock)
VALUES("Virus","Tu misión consiste en enfrentarte
 con arrojo a la pandemia y competir por ser el primero
 en erradicar los virus logrando aislar un cuerpo sano para evitar
 la propagación de las terribles enfermedades. Éticos o no, todos
 los medios a tu alcance valen para ganar","25");

 INSERT INTO juegos(nombre,descripcion,stock)
VALUES("Monopoly","Monopoly es un juego de mesa basado en el intercambio y la compraventa de bienes raíces","38");

INSERT INTO juegos(nombre,descripcion,stock)
VALUES("Catan","Catán es un juego de mecánica sencilla. Sobre un tablero que cambia su disposición en cada partida, cada jugador debe construir caminos, pueblos y ciudades y acumular diferentes cartas. Cada uno de estos elementos otorga ciertos puntos y el que llegue a 10 gana","15");

INSERT INTO juegos(nombre,descripcion,stock)
VALUES("Trivial","Trivial Pursuit es un juego de mesa muy popular cuyo objetivo es contestar correctamente la mayor cantidad de preguntas para juntar los seis “quesitos” y llegar a la pregunta final para ser el vencedor absoluto de la partida","2");
```

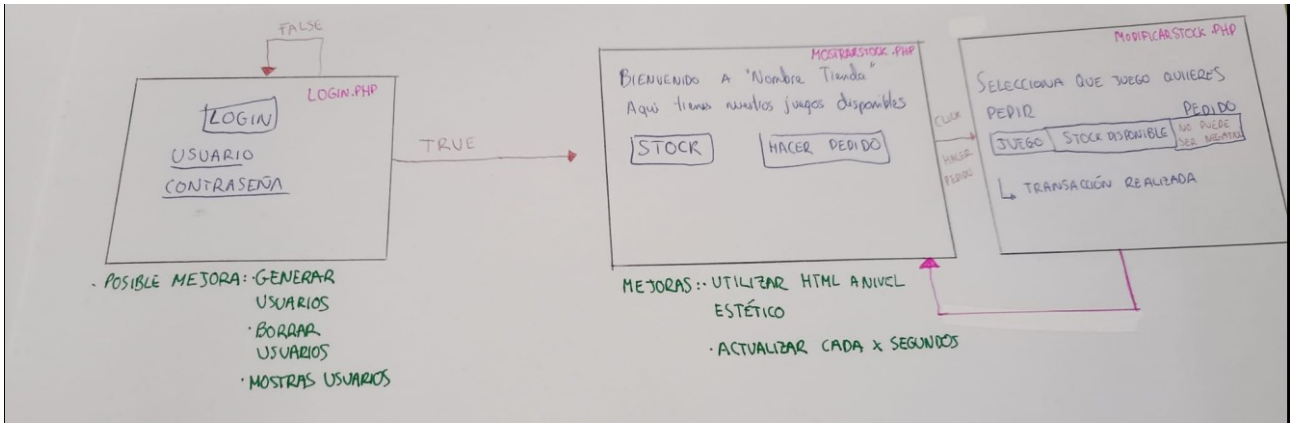
```
+-----------------------+
| Tables_in_juegosdemesa |
+-----------------------+
| juegos                |
+-----------------------+
1 row in set (0,01 sec)

mysql> select * from juegos;
+---------------+------------------------------------------------------------------------------------------------------------------
------------------------------------------------------------------------------------------------------------------------------------
---------------------------------------------------------------------------------+-------+
| nombre        | descripcion

                                                                                 | stock |
+---------------+------------------------------------------------------------------------------------------------------------------
------------------------------------------------------------------------------------------------------------------------------------
---------------------------------------------------------------------------------+-------+
| Bang          | Es un juego de disparos, al estilo Wild West, entre un grupo de Forajidos (Outlaws) y el Sheriff, su objetivo. Están los Alguaciles (Deputies) del
Sheriff, pero también hay un Renegado que solo persigue sus propios intereses!
                                                                                 |    18 |
| Blokus        | Blokus es un juego de estrategia abstracto que se juega como una extraña versión inversa del Tetris. Los jugadores compiten para colocar tantas pie
zas en el tablero como sea posible y cubrir la mayor parte del área. Las piezas del mismo color no pueden tocarse en los bordes exteriores, sino que deben estar conec
tadas por esquinas. Las piezas de diferentes colores pueden tocarse de cualquier manera.    |   215 |
| Catan         | Catán es un juego de mecánica sencilla. Sobre un tablero que cambia su disposición en cada partida, cada jugador debe construir caminos, pueblos y
ciudades y acumular diferentes cartas. Cada uno de estos elementos otorga ciertos puntos y el que llegue a 10 gana
                                                                                 |    15 |
| Ciudadelas    | Ciudadelas es un juego de cartas de estrategia en el cual tendremos que construir una ciudad con sus diferentes distritos y conseguir que sea la de
 mayor esplendor.

                                                                   |    65 |
| Cluedo        | El Cluedo es un juego de tablero donde los participantes deben resolver un crimen ficticio usando su capacidad de deducción.
```

3. Diseña el mapa de pantallas, como guía te propongo:

   1. Pantalla de autenticación inicial.
   2. Pantalla de visualización de stock y enlace a modificación del mismo.
   3. Pantalla para modificar el stock (puede mezclarse con la anterior) que solo permita aumentar el stock

      y añadir o borrar productos.
   4. Pantalla para realizar un pedido.
   5. Pantalla de confirmación de pedido, que obviamente actualizará el stock.

Utilizando la información dada diseñe la siguiente distribución de páginas:



BASE DE DATOS Y PANTALLA DE LOGIN

En la primera pantalla (login.php) nos mostrará uno botones para introducir un usuario y una contraseña. Al completar estos datos se comprobará en la base de datos si este usuario y contraseña son válidos. Si es así nos llevará a la siguiente página, si no volverá a cargar la página de login con un mensaje de error.

En mostrar php únicamente se mostrará la base de datos y todos sus stock. Así comprobaremos el stock actual. Diseñando esta página me encontré con la complicación de si otros usuarios realizan acciones sobre la base de datos este stock no seria real. Me planteé inicialmente remostrar la tabla cada x segundos así al menos el error no seria permanente.

Esta página no hará modificaciones y para realizarlas mostrará un botón que será hacer pedido donde trascurrirán las diferentes operaciones.

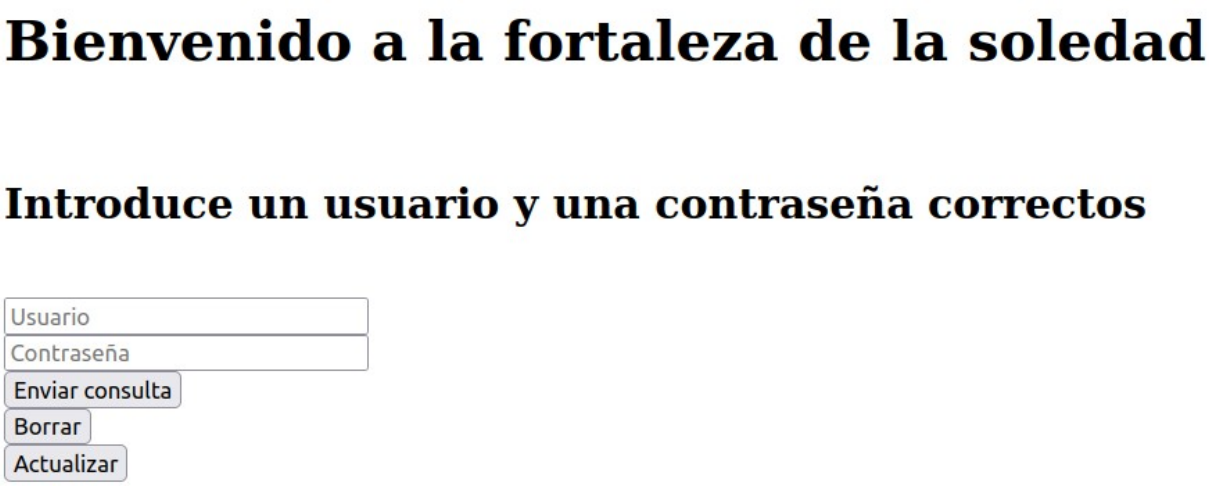
En la página modificarstock .php se nos mostrará un desplegable con los diferentes artículos. Al seleccionar estos nos mostrará el stock disponible y además tendremos un botón de añadir. Almacenaremos todos los pedidos que queramos hacer y tendremos otro botón de enviar. Al realizar esta acción se nos restará el stock del pedido de la base de datos y se nos mostrará de nuevo el stock disponible.

Me cree un nuevo usuario en mysql y le día permisos de administración sobre la tablas en cuestión. Con ello el usuario podría acceder a la base de datos.

CREATE USER 'valen'@'localhost' IDENTIFIED BY '1234Abc@';

Ahora tendría que desarrollar la página de login. Para ello haría un proyecto parecido al del carrito de la compra donde se tendría que validar un usuario y una contraseña para acceder al portal de mostrar stock.

Para ello el comando sesión start era imprescindible y lo coloque nada más empezar el documento. Desarrolle el html donde insertaría el formulario. Como ya aprendí en la anterior práctica necesitaba poner los valores del post de usuario y contraseña vacíos si no había nada para evitar un error. Una vez desarrollado el formulario se veia de la siguiente manera:



Ahora había que desarrollar la lógica en php que comprobará tanto el usuario y la contraseña. Si esto era así directamente nos llevaría a la página de mostrar stock. Y si no era así nos volvería a cargar la interfaz de login. El resultado de esta lógica fue esta:

```php
if($_POST["usuario"]=="valen" && $_POST["password"] == "1234Abc@")
{
    $_SESSION["usuario"]= $_POST["usuario"];

    $_SESSION["password"] = $_POST["password"];

    echo "Credenciales correctas <br>";
    echo "Elija dónde quiera acceder <br>";


    header('Location: http://localhost/JuegosMesa/mostrarstock.php');



}else{

    echo "<a href='http://localhost/JuegosMesa/login2.php'>";
}



?>
```

PANTALLA MOSTRAR STOCK

Ahora tocaba desarrollar la página de mostrar el stock. Para ello lo primero era comprobar que el usuario se había logeado bien. Para ello utilizamos el comando session_start(). Añadimos (como en el carrito de la compra) una condición por la que si el usuario o la contraseña no era los correcto redireccionara a la página de login.

Ahora tenía que realizar la conexión a la base de datos. Primero declaré las variables de host, dbname, username y password para reutilizarlas dentro del código php si fuera necesario. Asigné los valores que correspondían a la base de datos que había creado y realicé la conexión:

Ahora me enfrentaba con la dificultad de imprimir la base de datos. Para ello necesitaba

```php
$con = new PDO("mysql:host=$host;dbname=$dbname", $username, $password);
```

realizar una consulta y almacenarla en una variable ya que no conocía otro modo de hacerlo. En este caso sólo quería mostrar el resultado por tanto la consulta iba a ser sencilla.

Una vez hecho esto desarrollé una función que una vez conectara prepara la búsqueda (almacenada en una variable). Después de esto se ejecutaría:

```
20   $sql = "SELECT * FROM juegos";
21   $query = $con -> prepare($sql);
22   $query -> execute();
```

Una vez preparada la consulta y ejecutada tocaba imprimirla. Pero claro no quería imprimir sólo la primera línea por tanto busqué en la pagina de php algún método que pudiera servirme. Encontré una función en PDO que permitía devolver un array que contenía todas las filas, pero como había convertido mi búsqueda en un objeto tuve que utilizar el siguiente comando:

```
23   $results = $query -> fetchAll(PDO::FETCH_OBJ);
24
```

Ahora me quedaba imprimir la tabla con una estructura repetitiva. Para ello límite las condición de impresión a cuando el número de fila fuera superior a 0 me imprimiese el búcle. Me decidí por un foreach por que me parecía cómodo. A continuación puse el resultado de cada variable (el nombre, la descripción y el stock) estos mismos irían imprimiendo cada fila hasta el final.

```
if($query -> rowCount() > 0)
    {
        foreach($results as $result)
        {
        echo "<table border='1'><tr>

        <td>".$result -> nombre."</td>
        <br>
        <td>".$result -> descripcion."</td>
        <br>
        <td>".$result -> stock."</td>
        <br>
        </tr></table>";

            }
    }
```

Me dí cuenta de que me aparecía todo junto y por tanto probé a inserta una tabla con bordes y con diferentes filas para que estéticamente se viera mejor. Como tenía el tiempo limitado decidí meterle más tiempo después si acababa el proyecto.

Ahora tenia que crear dos campos que permitieran introducir o modificar elementos del stock. Para ello puse dos enlaces en la parte superior que me llevaran a otro php donde desarrollaría la lógica de cada uno de ellos.

La pantalla de modificar el stock tendría primero en mismo enlace de comprobación de sesiones. Además esta debería incluir un formulario el cual nos permitiría actualizar parámetros. Primero realizaría una conexión a la base de datos. Ahora me encontraba con el problema de que con mi desarrollo actual no podía parametrizar que stock tenia cada uno, ya que no lo había identificado y tras varios intentos me toco modificar la consulta para poder parametrizar el stock.

Necesitaba implementar un valor a la variables dependiendo del valor, por tanto la función que había realizado anteriormente no me valía. Tenía que buscar la manera de dar un atributo. Por tanto busque otra manera de recorrer la consulta.

Con la consulta función flecha y fetch() podría recorrer el objeto, ahora tenia que establecer una condición. Necesitaba una estructura repetitiva. Pensé en un do while, pero realmente no necesitaba comprobar la condición antes, por ello escogí el while.

La condición seria que hiciera esto hasta recorrer la última fila (que era la misma condición del anterior if, pero expresada de otra manera). Ahora solo tenia que concatenar los valores de la tabla. Como ya no tenía la función flecha pude dar a estos valores un identificador el cual podría usar después en la consulta de stock (además arreglé el tema de la tabla y los bordes).

Este fue el resultado:

```php
$sql = "SELECT * FROM juegos";
$query = $con -> prepare($sql);
$query -> execute();
while($resultado = $query -> fetch())
    {


    echo "
    <table border='1px' style='border-collapse: collapse' style=text-align:center ;>
    <tr>
        <th>Nombre</th>
        <th>Descripcion</th>
        <th>Stock</th>

    </tr>
    <tr>
    <td>".$resultado['nombre']."</td>
    <br>
    <td>".$resultado['descripcion']."</td>
    <br>
    <td>".$resultado['stock']."</td>
    <br>
    </tr>

    </table>";

    }
```

Ya con esto volví a lo que sería la pantalla de modificar stock.

MODIFICAR STOCK

La idea de esta página es que mostrara una tabla más simplificada con los atributos nombre, stock y un botón que permitiera añadir.

Para ello volví a mostrar la consulta y cree los valores anteriormente citados. Todo esto lo hice añadiendo además un botón de enviar y metí todo dentro de un formulario con el método post.

```php
//Botón enviar

echo "<input type='submit' name='enviar' value='Actualizar Stock'>";

//Consulta

$sql = "SELECT * FROM juegos";
$query = $con -> prepare($sql);
$query -> execute();
while($resultado = $query -> fetch())
    {


        echo "
        <table border='1px' style='border-collapse: collapse' style=text-align:center ;>
        <tr>
            <th>Nombre</th>
            <th>Stock</th>
            <th>Añadir producto</th>

        </tr>
        <tr>
        <td>".$resultado['nombre']."</td>
        <td>".$resultado['stock']."</td>
        <br>
        <td><input type='number' name=".$resultado['nombre']." value=".$resultado['stock']." min='0'</td>
        </tr>

        </table>";
    }

    //Consulta para añadir valoresS
```

Ahora tenia que desarrollar una nueva consulta en la que si dábamos al botón de enviar sustituyera el valor que tiene el botón por el stock actual.

Lo primero que hicimos fue añadir una condición por la cual comprobara si hay un actualizar. Si es así entraría dentro. Una vez aquí recorriamos la consulta y le dariamos a la variable stock el valor del parámetro id.

ERRORES encontrados. Al tratar de realizar esto con el parámetro nombre me daba un error. Consultando a varios compañeros (Amado y Miguel) vi que el error y esperaba un valor int en vez de el string que llevaba. Por tanto cambie la base de datos introduciendo el valor ID al comienzo de cada artículo. Esto me llevo a tener que modificar también mostrar stock y esta tabla. Además tuve que pone el valor que cogiera el number al principio fuera el de stock para evitar los errores de nulo.

Una vez hecho esto tocaba preparar la sentencia sql en la que básicamente en los valores de stock de la búsqueda principal se sustituyeran por los que había en el botón number a través de un update. Así quedó el código de actualizar y también remito la siguiente foto de los cambios realizar al otro bloque de código.

```php
if(isset($_POST["actualizar"])){

    if($respuesta = $con -> query("SELECT * from juegos"))
    {
        while($resultado = $respuesta->fetch())
        {
            $variable =['stock' => $_POST[$resultado["id"]]];


            $sql = "UPDATE juegos SET stock=:stock where id =". $resultado["id"];
            $stmt = $con->prepare($sql);
            $stmt -> execute($variable);
        }
    }
}
```

```php
if($consulta=$con -> query("SELECT * from juegos"))
{
    while($resultado = $consulta -> fetch())
    {


        echo "
        <table border='1px' style='border-collapse: collapse' style=text-align:center ;>
        <tr>
            <th>ID</th>
            <th>Nombre</th>
            <th>Stock</th>
            <th>Nuevo Stock</th>

        </tr>
        <tr>
        <td>".$resultado['id']."</td>
        <td>".$resultado['nombre']."</td>
        <td>".$resultado['stock']."</td>
        <br>
        <td><input type='number' min='0' name='".$resultado['id']."' value='".$resultado['stock'] ."'</td>
        </tr>

        </table>";
    }
    }
```

TAREA PENDIENTES

-Me falto tiempo para poder dedicarle a las transacciones.

-Mejorar estéticamente las páginas y mejorar la experiencia de usuario

-Ampliar la base de datos con diferentes campos o más tablas.

-Documentarme más sobre las diferentes opciones que tiene php con respecto a sql y mirar otras variantes como sqli.