- Base de datos en Mysql/MAriaDb y usuario con permiso en ella.

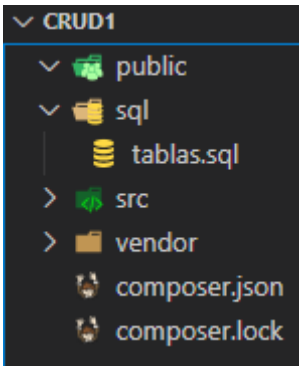
```
mysql> create database crud1;
Query OK, 1 row affected (0.00 sec)

mysql> CREATE USER 'crud_1'@'localhost' IDENTIFIED BY 'crud_1';
Query OK, 0 rows affected (0.00 sec)

mysql> GRANT ALL PRIVILEGES ON * . * TO 'crud_1'@'localhost';
Query OK, 0 rows affected (0.04 sec)
```

Insertando los datos

```
mysql> use crud1;
Database changed
mysql> create table articulos(
    -> id int PRIMARY KEY AUTO_INCREMENT,
    -> nombre VARCHAR(60) unique not null,
    -> pvp decimal(6,2) not null,
    -> stock INT UNSIGNED default 0);
Query OK, 0 rows affected (0.02 sec)

mysql>
mysql> insert into articulos(nombre, pvp, stock) values("Monitor 19", 123.45, 12);
Query OK, 1 row affected (0.00 sec)

mysql> insert into articulos(nombre, pvp, stock) values("Raton USB", 23.95, 13);
Query OK, 1 row affected (0.00 sec)

mysql> insert into articulos(nombre, pvp, stock) values("USB 512GB", 123.45, 14);
Query OK, 1 row affected (0.00 sec)

mysql> insert into articulos(nombre, pvp, stock) values("USB 8GB", 12.45, 14);
Query OK, 1 row affected (0.00 sec)

mysql> insert into articulos(nombre, pvp, stock) values("SET LAPICEROS", 13.15, 32);
Query OK, 1 row affected (0.00 sec)

mysql> insert into articulos(nombre, pvp, stock) values("Stick Wifi", 23, 62);
Query OK, 1 row affected (0.00 sec)

mysql> insert into articulos(nombre, pvp, stock) values("Funda 17", 13.4, 92);
Query OK, 1 row affected (0.00 sec)

mysql> insert into articulos(nombre, pvp, stock) values("Monitor 22", 193.65, 2);
Query OK, 1 row affected (0.00 sec)

mysql> insert into articulos(nombre, pvp, stock) values("Smart TV 52", 453.85, 92);
Query OK, 1 row affected (0.00 sec)

mysql> insert into articulos(nombre, pvp, stock) values("SSD 512GB", 123.45, 2);
Query OK, 1 row affected (0.00 sec)
```

```
mysql> use crud1;
Database changed
mysql> SELECT * FROM articulos;
+----+---------------+--------+-------+
| id | nombre        | pvp    | stock |
+----+---------------+--------+-------+
|  1 | Monitor 19    | 123.45 |    12 |
|  2 | Raton USB     |  23.95 |    13 |
|  3 | USB 512GB     | 123.45 |    14 |
|  4 | USB 8GB       |  12.45 |    14 |
|  5 | SET LAPICEROS |  13.15 |    32 |
|  6 | Stick Wifi    |  23.00 |    62 |
|  7 | Funda 17      |  13.40 |    92 |
|  8 | Monitor 22    | 193.65 |     2 |
|  9 | Smart TV 52   | 453.85 |    92 |
| 10 | SSD 512GB     | 123.45 |     2 |
+----+---------------+--------+-------+
10 rows in set (0.00 sec)
```

- Estructura con composer



- Una vez realizado lo anterior, realizamos la conexión con la base de datos y procedemos a realizar el CRUD.

- En primer lugar, mostraremos todas las filas de la tabla artículos.

Realizo con ayuda de bootstrap una página para mostrar los datos.

En la clase Artículos crearemos el primer método para devolver todos los valores de la tabla artículos y poder mostrarlos posteriormente.

En archivo lista.php donde mostraré la tabla artículos creo un objeto artículos para poder usar la función creada anteriormente y meter todos esos valores en una lista.

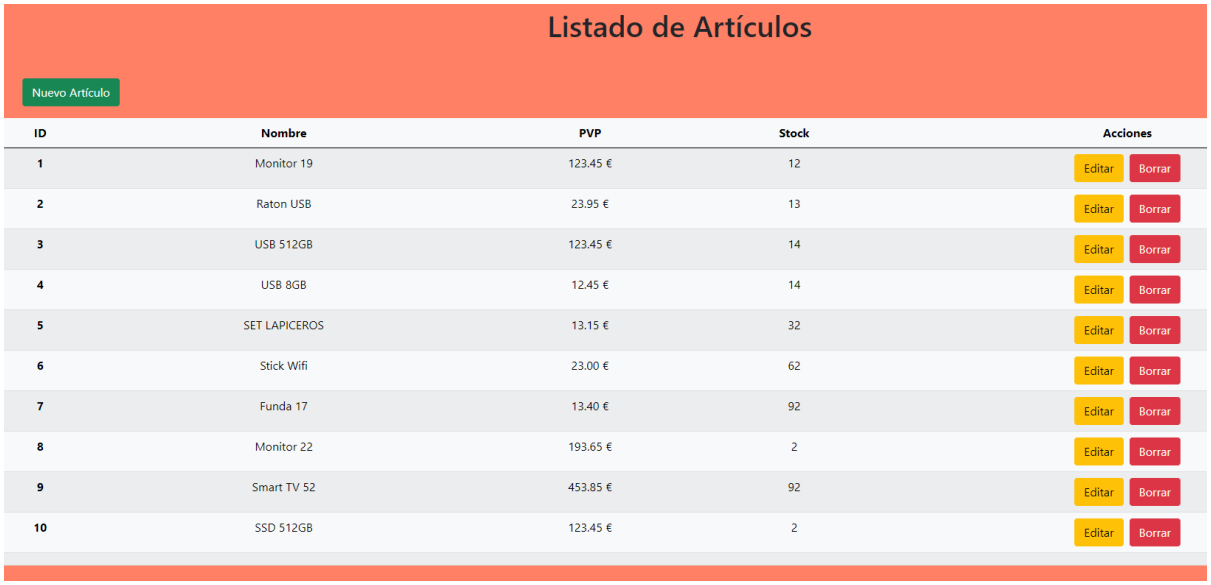
Finalmente, declaro la variable artículos como null, por si posteriormente se quiere usar.

```php
<?php
use Clases\Articulos;
require '../vendor/autoload.php';

$articulos=new Articulos();
$lista=$articulos->devolverArticulos();
$articulos=null;
?>
```

Para mostrar los valores deberemos crear un bucle donde coja una fila y guarde los valores en variables. Y esas variables las introducimos en el código HTML.

```php
<?php
    while($articulo=$lista->fetch(PDO::FETCH_OBJ)){
        echo "<tr>";
        echo "<th scope='row'>{$articulo->id}</th>";
        echo  "<td>{$articulo->nombre}</td>";
        echo  "<td>{$articulo->pvp} €</td>";
        echo  "<td>{$articulo->stock}</td>";
        echo  "<td>";
        echo "<form name='as' method='POST' class='form-inline' action='borrarArt.php'>";
        echo "<a href='editarArt.php?id={$articulo->id}' class='btn btn-warning me-2'>Editar</a>";
        echo "<input type='hidden' name='id' value='{$articulo->id}'>";
        echo "<button type='submit' class='ml-2 btn btn-danger'>Borrar</button></form>";
        echo "</td>\n";
        echo "</tr>\n";
    }
?>
```

Las acciones de los botones son archivo que serán creados posteriormente para su uso.



- Ahora introduciremos la opción para crear un nuevo artículo

Primero realizaré la función en la clase artículos.

```php
public function create(){
    $i="insert into articulos(nombre, pvp, stock) values(:n, :p, :s)";
    $stmt=parent::$conexion->prepare($i);
    try{
        $stmt->execute([
            ':n'=>$this->nombre,
            ':p'=>$this->pvp,
            ':s'=>$this->stock
        ]);
    }catch(PDOException $ex){
        die("Error al insertar un articulo: ".$ex->getMessage());
    }
}
```

A continuación, crearé un formulario con los campos a rellenar con la ayuda de bootsrap.

Además, comprobaré si el nombre del artículo ya está en uso o no.

```php
public function comprobarArticulo($articulo){
    $c="select * from articulos where nombre=:n";
    $stmt=parent::$conexion->prepare($c);
    try{
        $stmt->execute([
            ':n'=>$articulo
        ]);
    }catch(PDOException $ex){
        die("Error al comprobar articulo: ".$ex->getMessage());
    }
    $fila=$stmt->fetch(PDO::FETCH_OBJ);
    return ($fila==null) ? true : false;
}
```

En el método paso el nombre del artículo y lo compruebo si existe con una consulta.

Si no hay ninguna fila que coincida me devuelve true y podremos continuar creando el artículo, y si me devuelve falso saltara un error.

```php
<?php
require '../vendor/autoload.php';

use Clases\Articulos;

if(isset($_POST['crear'])){

    $nombre=trim($_POST['nombre']);
    $pvp=trim($_POST['pvp']);
    $stock=trim($_POST['stock']);
    $articulo= new Articulos();

    if($articulo->comprobarArticulo(ucwords($nombre))){
        $articulo->setNombre(ucwords($nombre));
        $articulo->setPvp(ucwords($pvp));
        $articulo->setStock(ucwords($stock));
        $articulo->create();
        $articulo=null;
        header("Location:lista.php");
    }else{
        $articulo=null;
        header("Location:{$_SERVER['PHP_SELF']}");
        die("Este artículo ya existe");
    }
}
?>
```

En este código php implementado en el archivo crearArticulo.php, realizamos una acción al seleccionar el botón crear.

Obtenemos el texto y lo introducimos en una variable, uso la función ucwords para poner la primera letra en mayúscula.

Después utilizamos la función creada anteriormente para comprobar si el nombre del artículo existe o no.

En caso de que no creamos un nuevo artículo, y si existe nos saltará un mensaje de error.

- Ahora voy a darle funcionalidad al botón borrar.

Como anteriormente crearé una función delete para dicho cometido.

```php
public function delete(){
    $consulta="delete from articulos where id=:i";
    $stmt=parent::$conexion->prepare($consulta);
    try{
        $stmt->execute([
            ':i'=>$this->id
        ]);
    }catch(PDOException $ex){
        die("Error al borrar el articulo: ".$ex->getMessage());
    }
}
```

Le añado utilidad al botón borrar, y lo redirijo a un nuevo archivo php creado.

```php
<?php

require "../vendor/autoload.php";

use Clases\Articulos;

$articulo=new Articulos();
$articulo->setId($_POST['id']);
$articulo->delete();
$articulo=null;

header(("Location:lista.php"));
```

En el archivo ejerzo la función del borrado del articulo pasado por id.

Haré el ejemplo de eliminar el artículo “Mando de Consola”.

- Por último, le daré funcionalidad al botón Editar.

Primero creamos la función read, que nos leerá el artículo que pasemos por su id.

```php
public function read(){
    $consulta="select * from articulos where id=:i";
    $stmt = parent::$conexion->prepare($consulta);
    try {
        $stmt->execute([':i' => $this->id]);
    } catch (PDOException $ex) {
        die("Error al leer un articulo: " . $ex->getMessage());
    }
    return $stmt;
}
```

En segundo lugar, creamos la función update, que nos actualizará los datos del artículo pasado por id.

```php
public function update(){
    $consulta="update articulos set nombre=:n, pvp=:p, stock=:s where id=:i";
    $stmt = parent::$conexion->prepare($consulta);
    try {
        $stmt->execute([
            ':i' => $this->id,
            ':n' => $this->nombre,
            ':p' => $this->pvp,
            ':s' => $this->stock
        ]);
    } catch (PDOException $ex) {
        die("Error al actualizar articulo: " . $ex->getMessage());
    }
}
```

Con las funciones creadas, creo el archivo editarArt.php donde habrá un formulario hecho con bootstrap

Lo primero que haré será rellenar los campos con sus correspondientes valores.

```php
$id=$_GET['id'];

$articulo=new Articulos();
$articulo->setId($id);
$datos=$articulo->read();
$datos1=$datos->fetch(PDO::FETCH_OBJ);
```

Guardo en la variable datos1 una fila de datos, que obtenemos a través de la id pasada.

```php
<form name="nt" action="<?php echo $_SERVER['PHP_SELF']."?id=$id";?>" method="POST">
<div class="mt-2">
<input type="text" name="nombre" value="<?php echo $datos1->nombre; ?>" required class="form-control mb-2"/>
<input type="text" name="pvp" value="<?php echo $datos1->pvp; ?>" required class="form-control mb-2"/>
<input type="text" name="stock" value="<?php echo $datos1->stock; ?>" required class="form-control mb-4"/>
```

Introduzco los datos en el html.

Por último, le doy funcionalidad al botón editar.

```php
if(isset($_POST['editar'])){
    $nombre=trim($_POST['nombre']);
    $pvp=trim($_POST['pvp']);
    $stock=trim($_POST['stock']);

    $articulo->setNombre(ucwords($nombre));
    $articulo->setPvp(ucwords($pvp));
    $articulo->setStock(ucwords($stock));
    $articulo->update();
    $articulo=null;
    header("Location:lista.php");
}
```

Obtengo los valores de los campos y los introduzco en el artículo y lo actualizo.

Ejemplo:

Obtengo los datos en los campos

Y actualizo el articulo