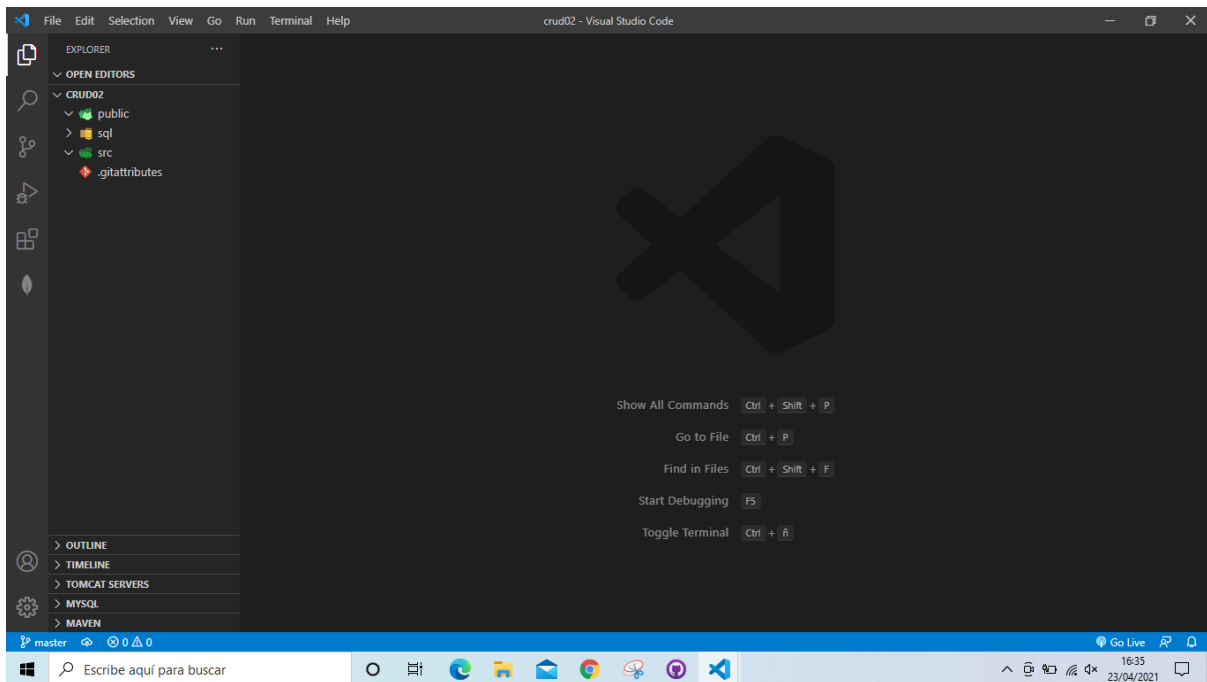


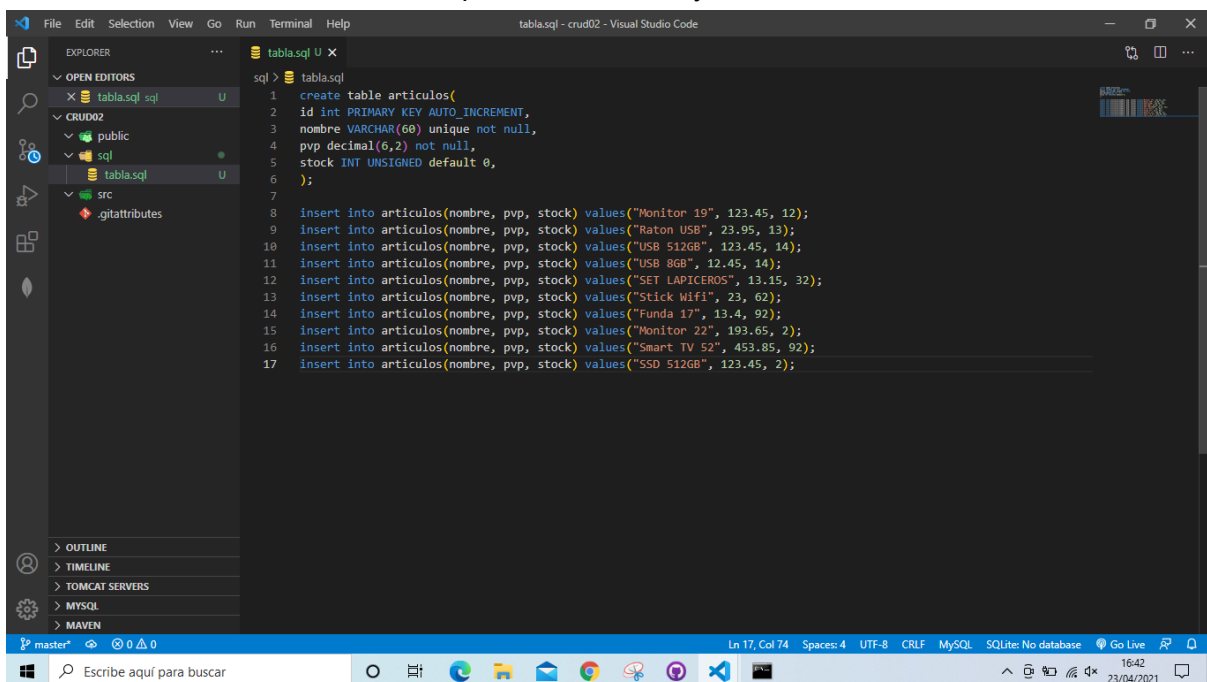
# Haciendo Crud desde cero

## 1-Crear estructura de clases y composer

Creamos las carpetas donde se crearán los archivos correspondientes. En un crud con base sql debe tener public (lo que vemos como usuario), src (los programas y el tratamiento de información) y sql (base de datos).



Observamos la tabla con los correspondientes datos y sus atributos.



Empezamos a aplicar el comando composer para iniciar la descarga de las librerías correspondientes y los ajustes necesarios para que funcione desde nuestro localhost de windows (linux lo más recomendado).

En MySQL debemos crear la tabla y las inserciones en MariaDB para tener una base de datos para trabajar en php.

Entrar en mysql

Crear usuario

```
mysql -u root
```

```
create user user@localhost identified by secret0;
```

```
sudo mysql -u root
```

Crear base de datos

Otorgar privilegios al usuario en la base de datos

```
create database crud02;
```

```
grant all privileges on crud01.* to user@localhost;
```

Usar la base de datos

```
use crud02
```

```
Simbolo del sistema - mysql -u root -p
Microsoft Windows [Versión 10.0.18363.1500]
(c) 2019 Microsoft Corporation. Todos los derechos reservados.

C:\Users\Jose>mysql -u root -p
Enter password: ****
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 3
Server version: 10.5.8-MariaDB mariadb.org binary distribution

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MariaDB [(none)]> create database crud02;
Query OK, 1 row affected (0.011 sec)

MariaDB [(none)]> use databases;
ERROR 1049 (42000): Unknown database 'databases'
MariaDB [(none)]> show databases;
+-----+
| Database |
+-----+
| base1    |
| crud01   |
| crud02   |
| examen_1 |
| information_schema |
| mysql    |
| performance_schema |
| test     |
+-----+
8 rows in set (0.048 sec)

MariaDB [(none)]> grant all privileges on crud02.* to user@localhost;
Query OK, 0 rows affected (0.022 sec)

MariaDB [(none)]> use crud02;
Database changed
MariaDB [crud02]> 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 (0.068 sec)
```

```
Simbolo del sistema - mysql -u root -p
MariaDB [crud02]> insert into articulos(nombre, pvp, stock) values("Smart TV 52", 453.85, 92);
ERROR 1146 (42S02): Table 'crud02.articulos' doesn't exist
MariaDB [crud02]> insert into articulos(nombre, pvp, stock) values("SSD 512GB", 123.45, 2);
ERROR 1146 (42S02): Table 'crud02.articulos' doesn't exist
MariaDB [crud02]> 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 (0.068 sec)

MariaDB [crud02]>
MariaDB [crud02]> insert into articulos(nombre, pvp, stock) values("Monitor 19", 123.45, 12);
Query OK, 1 row affected (0.142 sec)

MariaDB [crud02]> insert into articulos(nombre, pvp, stock) values("Raton USB", 23.95, 13);
Query OK, 1 row affected (0.003 sec)

MariaDB [crud02]> insert into articulos(nombre, pvp, stock) values("USB 512GB", 123.45, 14);
Query OK, 1 row affected (0.002 sec)

MariaDB [crud02]> insert into articulos(nombre, pvp, stock) values("USB 8GB", 12.45, 14);
Query OK, 1 row affected (0.002 sec)

MariaDB [crud02]> insert into articulos(nombre, pvp, stock) values("SET LAPICEROS", 13.15, 32);
Query OK, 1 row affected (0.001 sec)

MariaDB [crud02]> insert into articulos(nombre, pvp, stock) values("Stick Wifi", 23, 62);
Query OK, 1 row affected (0.002 sec)

MariaDB [crud02]> insert into articulos(nombre, pvp, stock) values("Funda 17", 13.4, 92);
Query OK, 1 row affected (0.002 sec)

MariaDB [crud02]> insert into articulos(nombre, pvp, stock) values("Monitor 22", 193.65, 2);
Query OK, 1 row affected (0.002 sec)

MariaDB [crud02]> insert into articulos(nombre, pvp, stock) values("Smart TV 52", 453.85, 92);
Query OK, 1 row affected (0.002 sec)

MariaDB [crud02]> insert into articulos(nombre, pvp, stock) values("SSD 512GB", 123.45, 2);
Query OK, 1 row affected (0.003 sec)

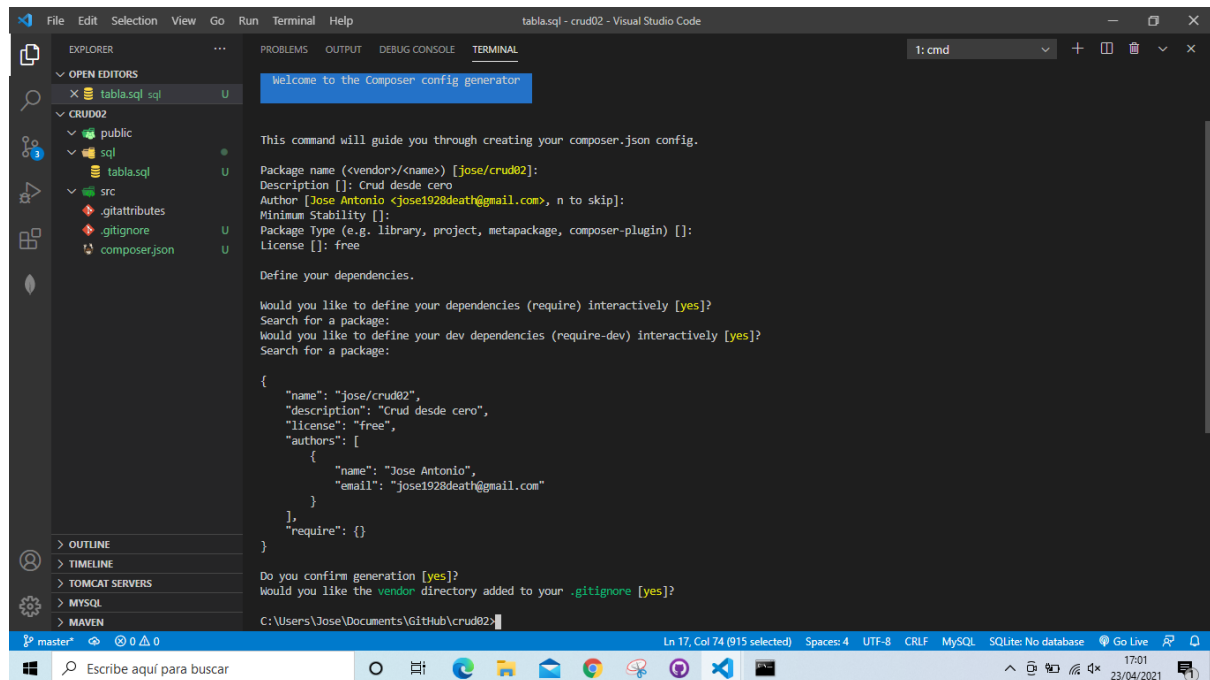
MariaDB [crud02]>
```

```
Simbolo del sistema - mysql -u root -p
Query OK, 1 row affected (0.003 sec)

MariaDB [crud02]> 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 (0.005 sec)

MariaDB [crud02]>
```

Hacemos composer dentro de nuestro proyecto para cargar las librerías y la carpeta vendor  
En el terminal escribimos **composer init;**



The screenshot shows the Visual Studio Code interface with the 'composer init' command being executed in the terminal. The terminal output is as follows:

```
Welcome to the Composer config generator.

This command will guide you through creating your composer.json config.

Package name (<vendor>/<name>) [jose/crud02]:
Description []: Crud desde cero
Author [Jose Antonio <jose1928death@gmail.com>, n to skip]:
Minimum Stability []:
Package Type (e.g. library, project, metapackage, composer-plugin) []:
License []: free

Define your dependencies.

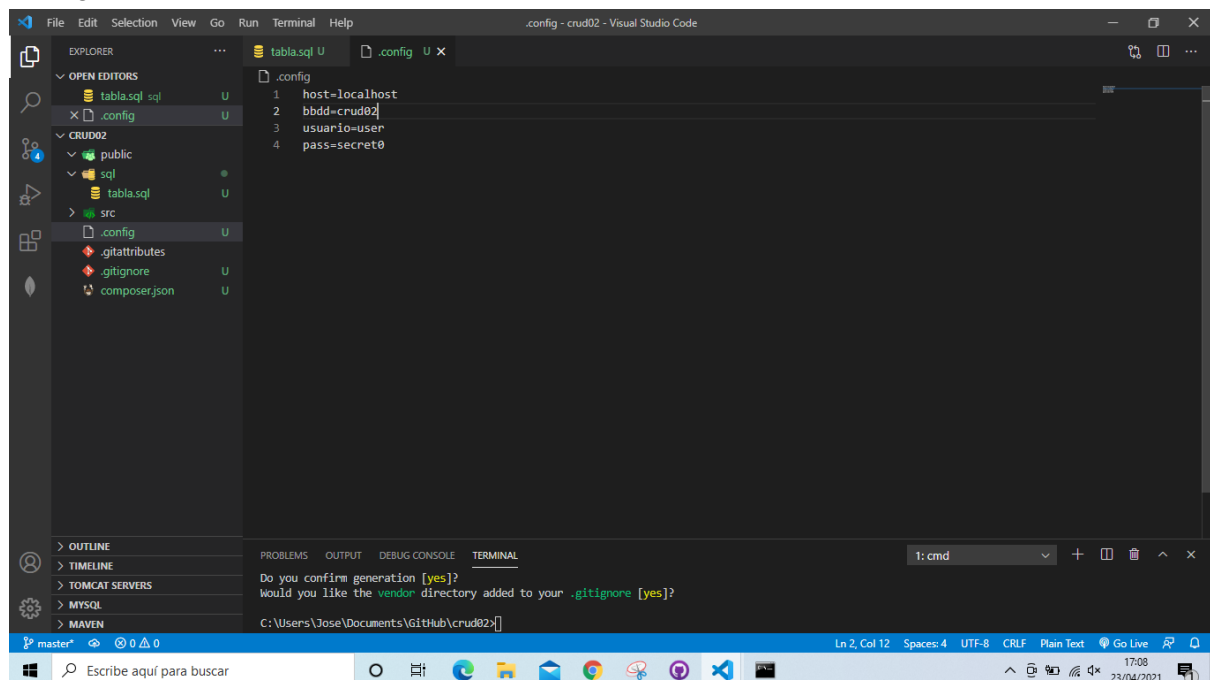
Would you like to define your dependencies (require) interactively [yes]?
Search for a package:
Would you like to define your dev dependencies (require-dev) interactively [yes]?
Search for a package:

{
    "name": "jose/crud02",
    "description": "Crud desde cero",
    "license": "free",
    "authors": [
        {
            "name": "Jose Antonio",
            "email": "jose1928death@gmail.com"
        }
    ],
    "require": {}
}

Do you confirm generation [yes]?
Would you like the vendor directory added to your .gitignore [yes]?

C:\Users\Jose\Documents\Github\crud02
```

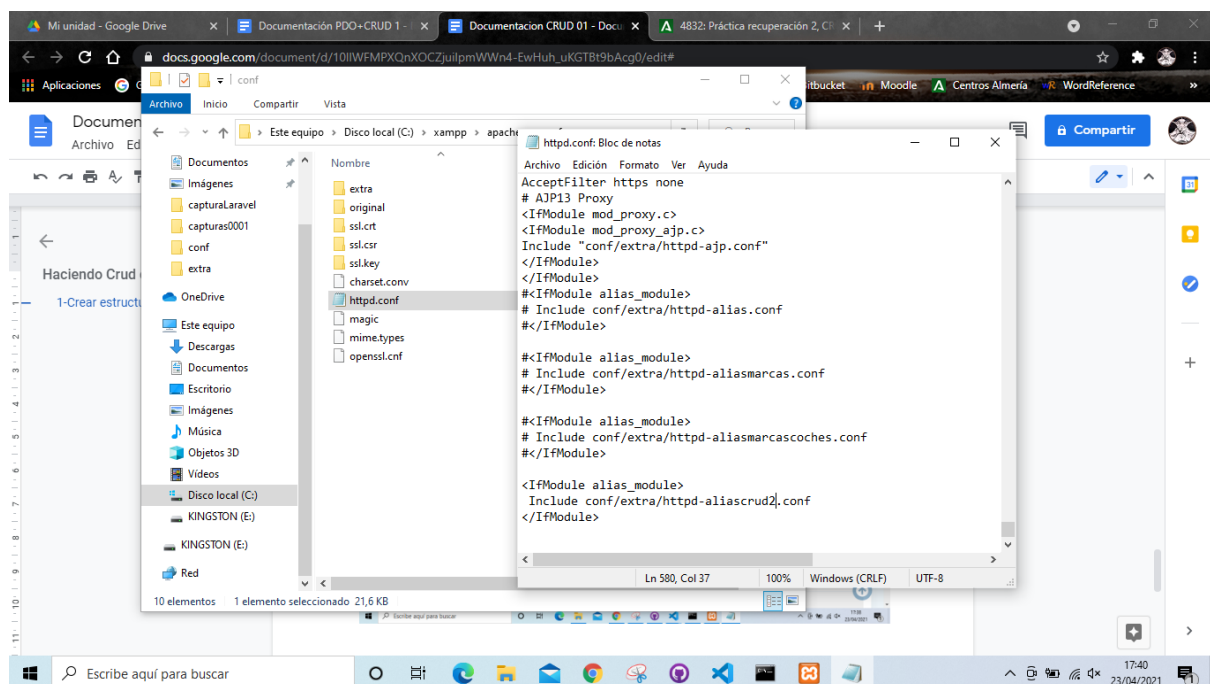
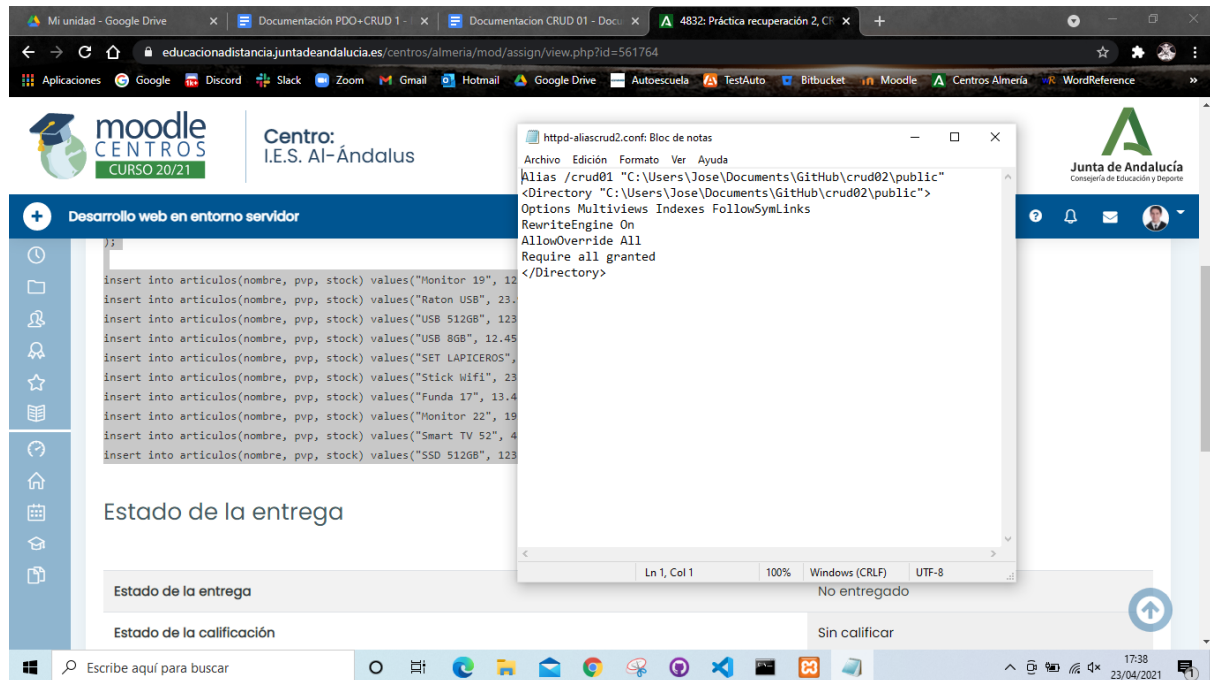
.config es donde pondremos los parametros para conectarnos al pdo de la base de datos.



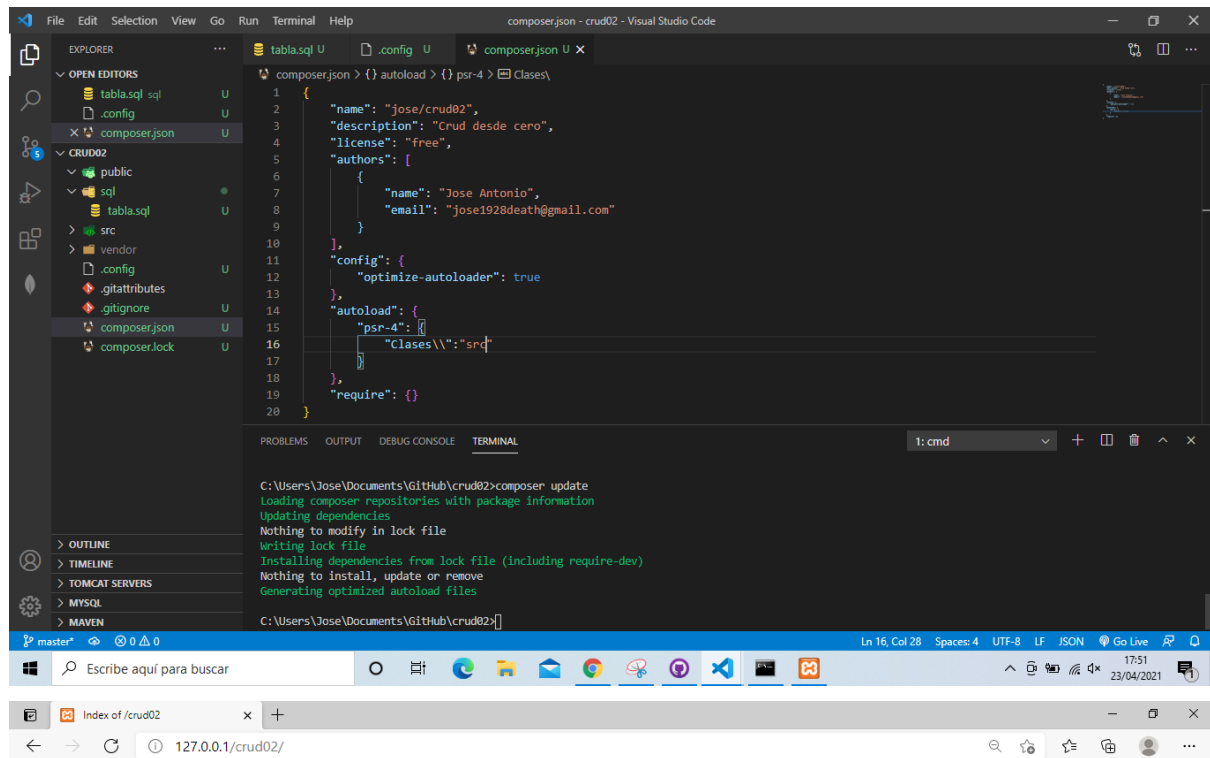
The screenshot shows the Visual Studio Code interface with the '.config' file open in the editor. The file content is as follows:

```
.config
1 host=localhost
2 bdd=crud02
3 usuario=user
4 pass=secret0
```

Ya que estamos en Windows debemos configurar httdocs y httpd-alias para acceder al localhost a partir del proyecto crud. Accedemos a xampp/apache/conf/extra/httpd-alias... para que acceda al localhost con xampp



Debemos escribir en `.config` las siguientes líneas para que tenga el formato psr4 y que el autoload se cargue a partir de `src`.



The screenshot shows the Visual Studio Code interface with the `composer.json` file open. The file contains the following JSON configuration:

```
1 {
2   "name": "jose/crud02",
3   "description": "Crud desde cero",
4   "license": "free",
5   "authors": [
6     {
7       "name": "Jose Antonio",
8       "email": "jose1928death@gmail.com"
9     }
10  ],
11  "config": {
12    "optimize-autoloader": true
13  },
14  "autoload": {
15    "psr-4": {
16      "Classes\\": "src/"
17    }
18  },
19  "require": {}
20 }
```

The terminal at the bottom shows the output of the `composer update` command:

```
C:\Users\Jose\Documents\Github\crud02>composer update
Loading composer repositories with package information
Updating dependencies
Nothing to modify in lock file
Writing lock file
Installing dependencies from lock file (including require-dev)
Nothing to install, update or remove
Generating optimized autoload files
C:\Users\Jose\Documents\Github\crud02>
```

## Index of /crud02

Name	Last modified	Size	Description
------	---------------	------	-------------

<a href="#">Parent Directory</a>	-		
----------------------------------	---	--	--

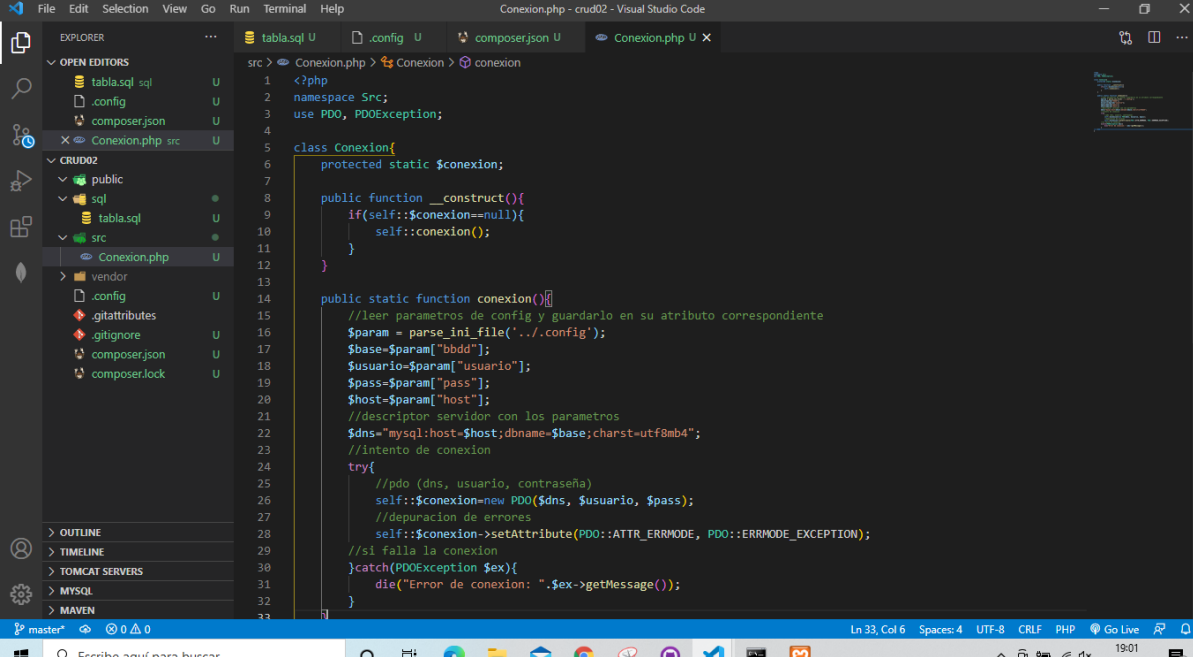
Apache/2.4.46 (Ubuntu) OpenSSL/1.1.1h PHP/8.0.0 Server at 127.0.0.1 Port 80

Si falla el xampp por sql debemos eliminar la tarea en progreso desde administrador de tareas y eliminar el proceso. Una vez hecho, se activará de nuevo el servicio sql desde xampp.

Estos problemas de xampp son subsanados en linux y se corre a tiempo real sin programas secundarios por lo que es recomendable tratarlo con linux.

## 2-Crear conexión

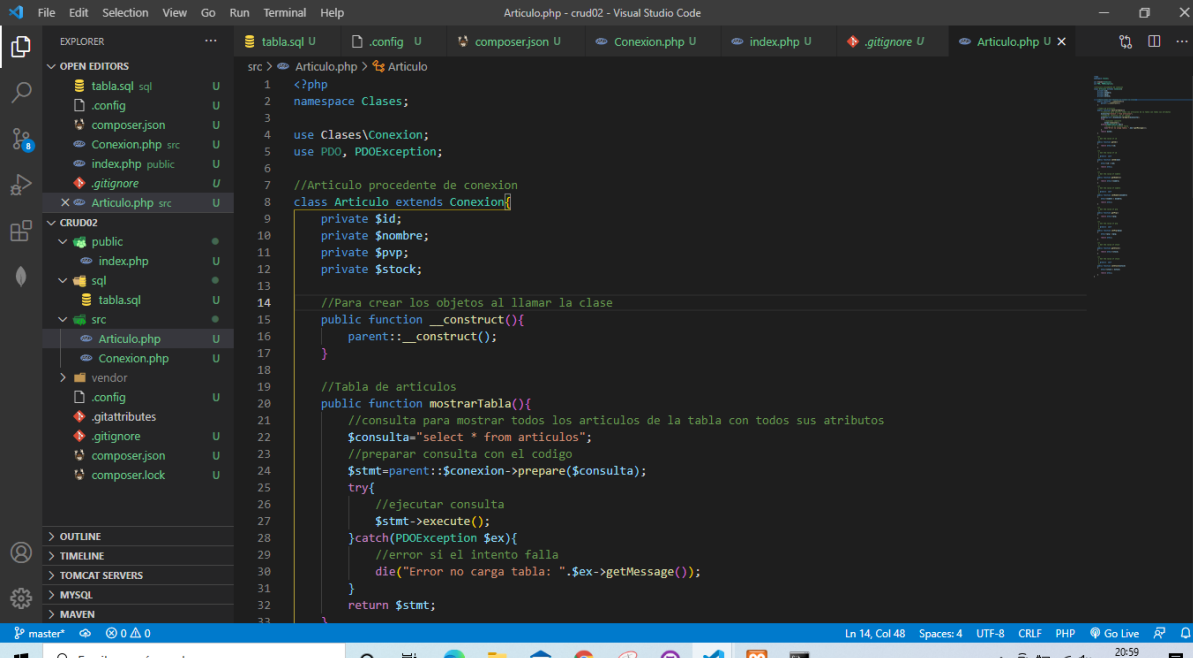
Se crea esta clase para establecer la conexión entre el pdo y mysql para que interactúen entre ellos. Para ello se necesita una base de datos creada, un usuario con todos los privilegios y por último su contraseña. Con ellos podemos establecer operaciones y cambios a la base de datos sql desde php.



```
1 <?php
2 namespace Src;
3 use PDO, PDOException;
4
5 class Conexion{
6     protected static $conexion;
7
8     public function __construct(){
9         if(self::$conexion==null){
10             self::conexion();
11         }
12     }
13
14     public static function conexion(){
15         //leer parametros de config y guardarlo en su atributo correspondiente
16         $param = parse_ini_file("../.config");
17         $base=$param["bdd"];
18         $usuario=$param["usuario"];
19         $pass=$param["pass"];
20         $host=$param["host"];
21         //descriptor servidor con los parametros
22         $dns="mysql:host=$host;dbname=$base;charset=utf8mb4";
23         //intento de conexion
24         try{
25             //pdo (dns, usuario, contraseña)
26             self::$conexion=new PDO($dns, $usuario, $pass);
27             //depuracion de errores
28             self::$conexion->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
29             //si falla la conexion
30         }catch(PDOException $ex){
31             die("Error de conexion: ".$ex->getMessage());
32         }
33     }
34 }
```

## 3-Mostrar tabla

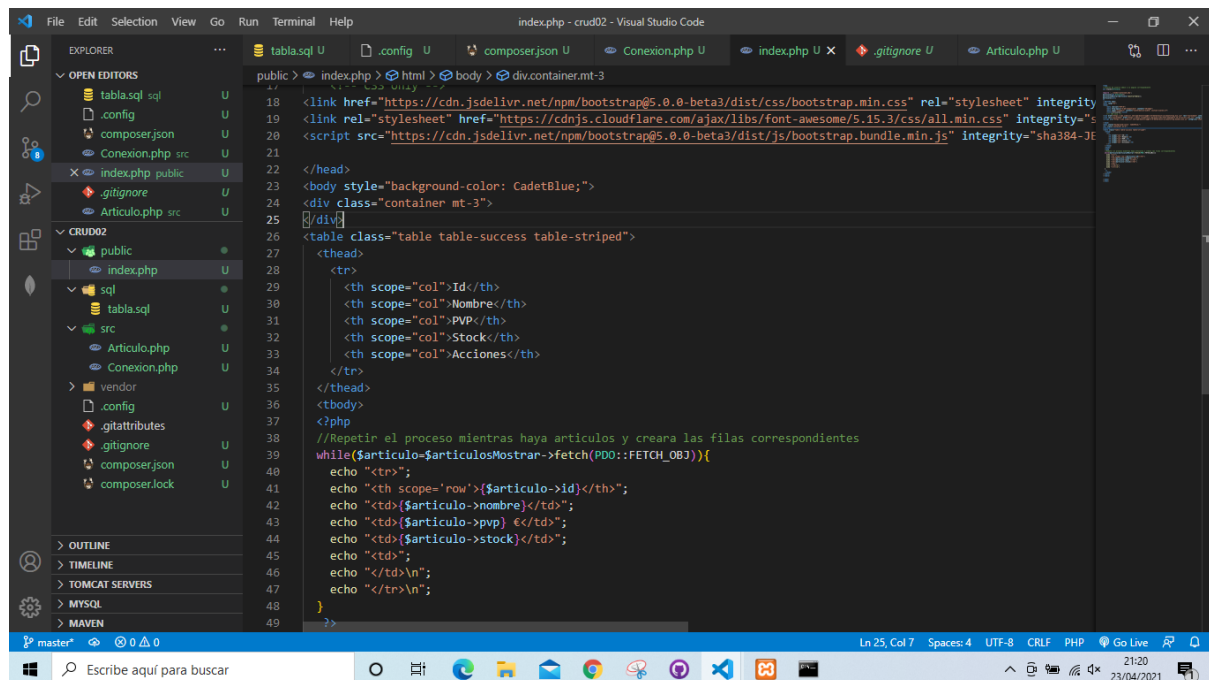
Creamos el objeto articulo con sus atributos correspondientes (el que se señala en la tabla mysql proporcionada) creamos el método \_\_construct para crear el artículo por defecto si llamamos a la clase. El metodo mostrarTabla() para enseñar todos los artículos y sus setters



```
1 <?php
2 namespace Clases;
3
4 use Clases\Conexion;
5 use PDO, PDOException;
6
7 //Articulo procedente de conexion
8 class Articulo extends Conexion{
9     private $id;
10     private $nombre;
11     private $pvp;
12     private $stock;
13
14     //Para crear los objetos al llamar la clase
15     public function __construct(){
16         parent::__construct();
17     }
18
19     //Tabla de articulos
20     public function mostrarTabla(){
21         //consulta para mostrar todos los articulos de la tabla con todos sus atributos
22         $consulta="select * from articulos";
23         //preparar consulta con el codigo
24         $stmt=parent::$conexion->prepare($consulta);
25         try{
26             //ejecutar consulta
27             $stmt->execute();
28         }catch(PDOException $ex){
29             //error si el intento falla
30             die("Error no carga tabla: ".$ex->getMessage());
31         }
32         return $stmt;
33     }
34 }
```

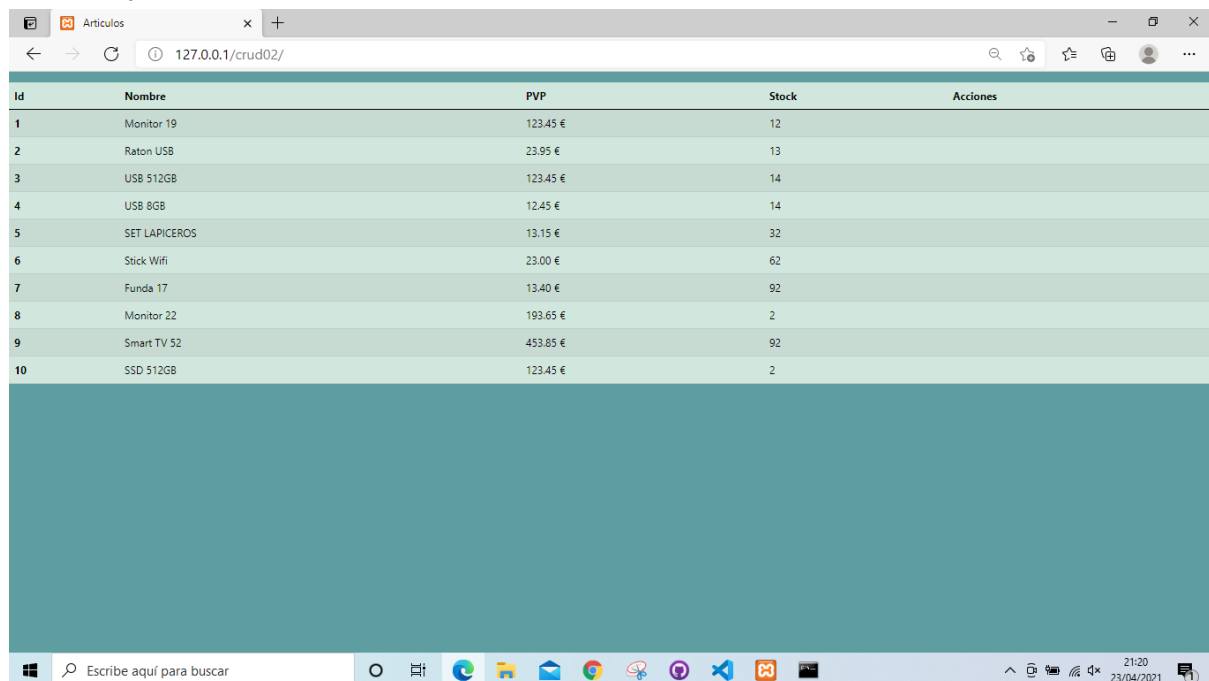
Con una tabla que queramos podemos enseñar los datos de cada artículo con su fila correspondiente.

<https://getbootstrap.com/docs/5.0/content/tables/>



```
18 <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.0.0-beta3/dist/css/bootstrap.min.css" rel="stylesheet" integrity="sha384-zv6Rp/Whlftuq6IHPrw3A5RhnzlrSkhN8Zc3OSVxSM49ESV7SM4X9QvZqwzK60sSx" crossorigin="anonymous">
19 <link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/bootstrap@5.0.0-beta3/dist/css/bootstrap.min.css" integrity="sha384-zv6Rp/Whlftuq6IHPrw3A5RhnzlrSkhN8Zc3OSVxSM49ESV7SM4X9QvZqwzK60sSx" crossorigin="anonymous">
20 <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.0.0-beta3/dist/js/bootstrap.bundle.min.js" integrity="sha384-JmXCpMvWHzQGXnZKvpNHziIV7AFepkgiHMZoWGsbfZ/sC1q8ZQ3Dk647Pgs1d/ZqN" crossorigin="anonymous">
21
22 </head>
23 <body style="background-color: CadetBlue;">
24 <div class="container mt-3">
25 <div>
26 <table class="table table-success table-striped">
27 <thead>
28 <tr>
29 <th scope="col">Id</th>
30 <th scope="col">Nombre</th>
31 <th scope="col">PVP</th>
32 <th scope="col">Stock</th>
33 <th scope="col">Acciones</th>
34 </tr>
35 </thead>
36 <tbody>
37 <?php
38 //Repetir el proceso mientras haya articulos y creara las filas correspondientes
39 while($articulo=$articulosMostrar->fetch(PDO::FETCH_OBJ)){
40 echo "<tr>";
41 echo "<th scope='row'>{$articulo->id}</th>";
42 echo "<td>{$articulo->nombre}</td>";
43 echo "<td>{$articulo->pvp} €</td>";
44 echo "<td>{$articulo->stock}</td>";
45 echo "<td>";
46 echo "</td>\n";
47 echo "</tr>\n";
48 }
49 >>
```

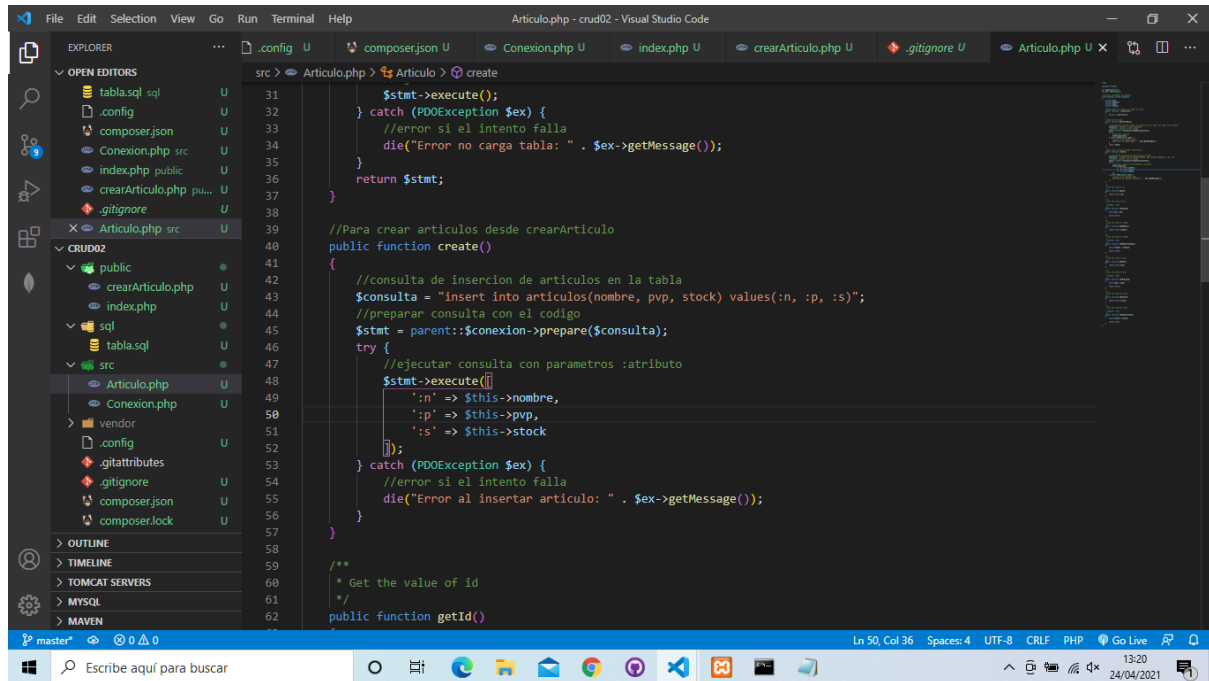
Vista de la tabla con los datos correspondientes (nombre, pvp, stock y acciones), podemos asignarle un botón para crear un articulo nuevo y cada fila con su correspondiente acciones de editar y borrar.



Id	Nombre	PVP	Stock	Acciones
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	

## 4-Crear artículo

Creamos el método create para que insertemos los artículos en la tabla con la consulta de sql de inserción. Para ello debemos marcar parámetros con los atributos que nos piden los artículos.

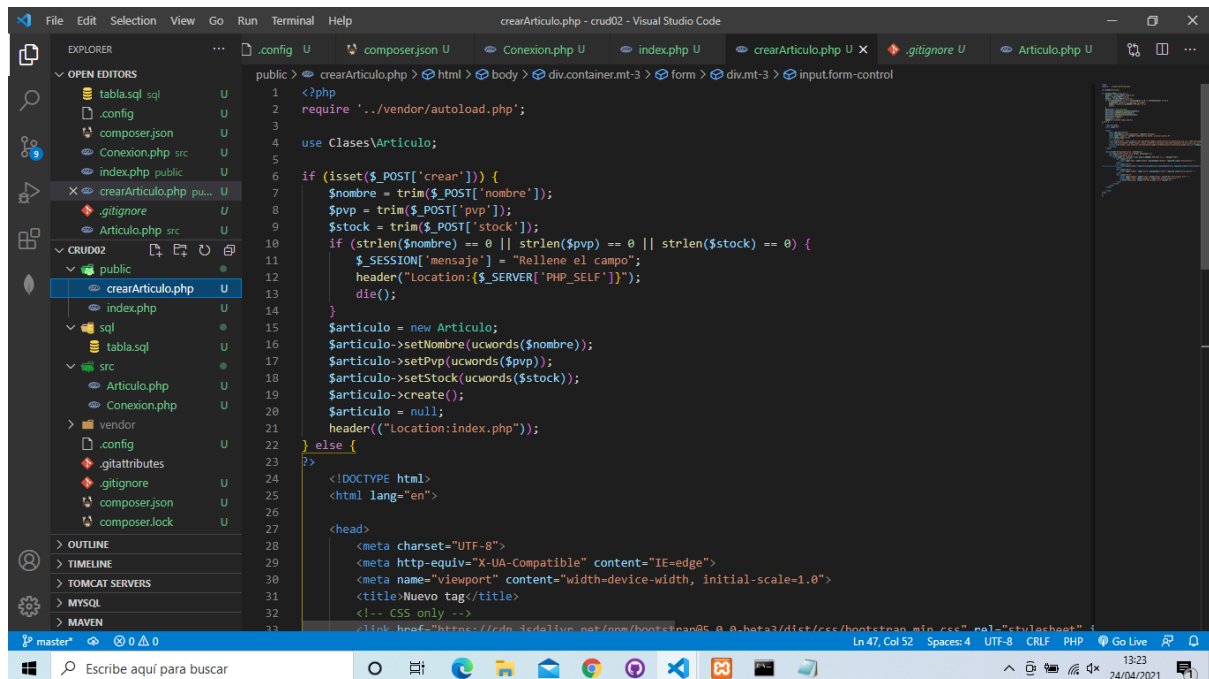


```
File Edit Selection View Go Run Terminal Help
Articulo.php - crud02 - Visual Studio Code

EXPLORER
src > Articulo.php > create
  tabla.sql sql U
  .config U
  composer.json U
  Conexion.php src U
  index.php public U
  crearArticulo.php public U
  .gitignore U
  Articulo.php src U
  CRUD02
  public
    crearArticulo.php U
    index.php U
  sql
    tabla.sql U
  src
    Articulo.php U
    Conexion.php U
  vendor
  .config U
  .gitattributes U
  .gitignore U
  composer.json U
  composer.lock U
  OUTLINE
  TIMELINE
  TOMCAT SERVERS
  MYSQL
  MAVEN

src > Articulo.php > create
31 $stmt->execute();
32 } catch (PDOException $ex) {
33     //error si el intento falla
34     die("Error no carga tabla: " . $ex->getMessage());
35 }
36 return $stmt;
37 }
38
39 //Para crear articulos desde crearArticulo
40 public function create()
41 {
42     //consulta de insercion de articulos en la tabla
43     $consulta = "insert into articulos(nombre, pvp, stock) values(:n, :p, :s)";
44     //preparar consulta con el codigo
45     $stmt = parent::$conexion->prepare($consulta);
46     try {
47         //ejecutar consulta con parametros :atributo
48         $stmt->execute([
49             ':n' => $this->nombre,
50             ':p' => $this->pvp,
51             ':s' => $this->stock
52         ]);
53     } catch (PDOException $ex) {
54         //error si el intento falla
55         die("Error al insertar articulo: " . $ex->getMessage());
56     }
57 }
58
59 /**
60  * Get the value of id
61  */
62 public function getId()
```

Con los campos correspondientes recogemos la información con unos requisitos pertinentes (nombre de tipo string, pvp de tipo float con dos decimales y stock de tipo int), los campos no deben estar vacíos para que lleve a cabo el proceso de inserción gracias al método create.

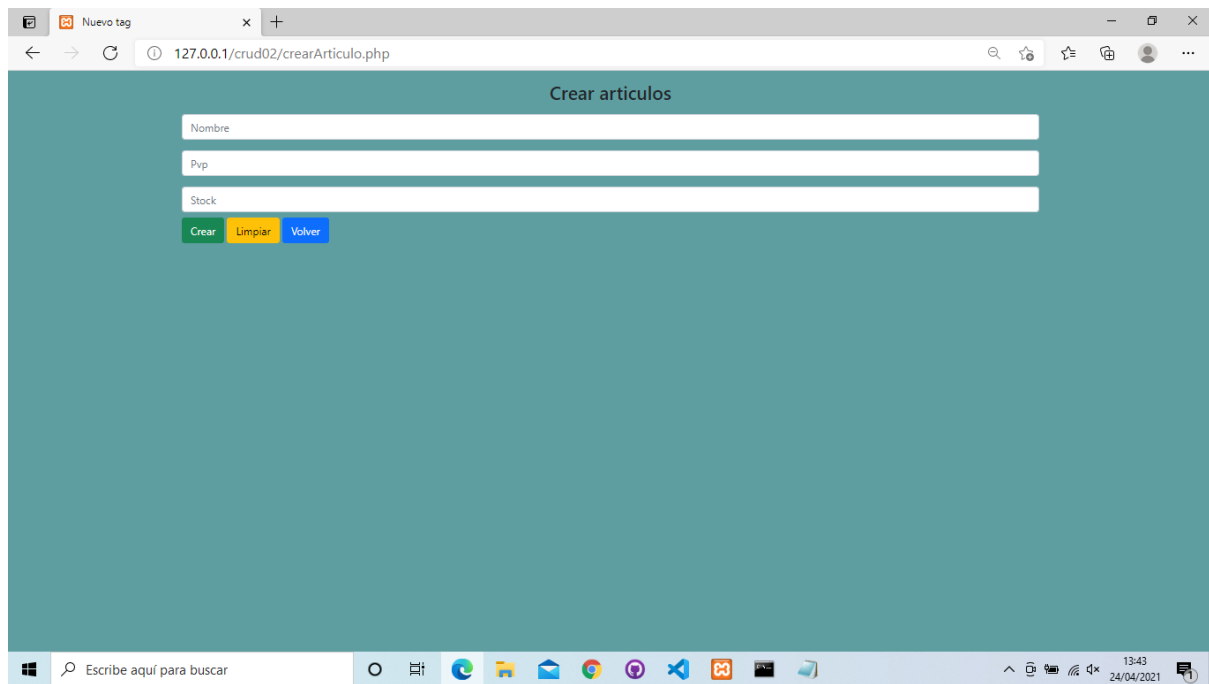


```
File Edit Selection View Go Run Terminal Help
crearArticulo.php - crud02 - Visual Studio Code

EXPLORER
public > crearArticulo.php > html > body > div.container.mt-3 > form > div.mt-3 > input.form-control
  tabla.sql sql U
  .config U
  composer.json U
  Conexion.php src U
  index.php public U
  crearArticulo.php public U
  .gitignore U
  Articulo.php src U
  CRUD02
  public
    crearArticulo.php U
    index.php U
  sql
    tabla.sql U
  src
    Articulo.php U
    Conexion.php U
  vendor
  .config U
  .gitattributes U
  .gitignore U
  composer.json U
  composer.lock U
  OUTLINE
  TIMELINE
  TOMCAT SERVERS
  MYSQL
  MAVEN

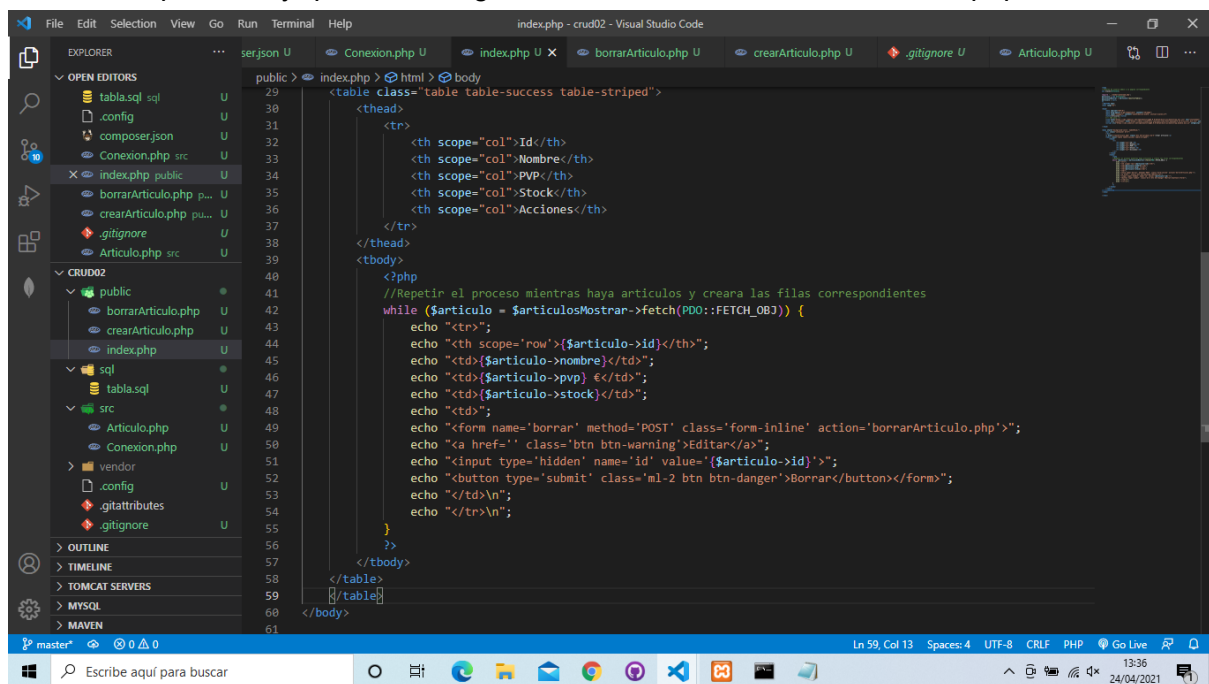
public > crearArticulo.php > html > body > div.container.mt-3 > form > div.mt-3 > input.form-control
1 <?php
2 require '../vendor/autoload.php';
3
4 use Clases\Articulo;
5
6 if (isset($_POST['crear'])) {
7     $nombre = trim($_POST['nombre']);
8     $pvp = trim($_POST['pvp']);
9     $stock = trim($_POST['stock']);
10    if (strlen($nombre) == 0 || strlen($pvp) == 0 || strlen($stock) == 0) {
11        $_SESSION['mensaje'] = "Rellene el campo";
12        header("Location:{$_SERVER['PHP_SELF']}");
13        die();
14    }
15    $articulo = new Articulo();
16    $articulo->setNombre(ucwords($nombre));
17    $articulo->setPvp(ucwords($pvp));
18    $articulo->setStock(ucwords($stock));
19    $articulo->create();
20    $articulo = null;
21    header("Location:index.php");
22 } else {
23 }
24
25 <DOCTYPE html>
26 <html lang="en">
27
28 <head>
29     <meta charset="UTF-8">
30     <meta http-equiv="X-UA-Compatible" content="IE=edge">
31     <meta name="viewport" content="width=device-width, initial-scale=1.0">
32     <title>Nuevo tag</title>
33     <!-- CSS only -->
34     <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.0.0-beta3/dist/css/bootstrap.min.css" rel="stylesheet" >
```



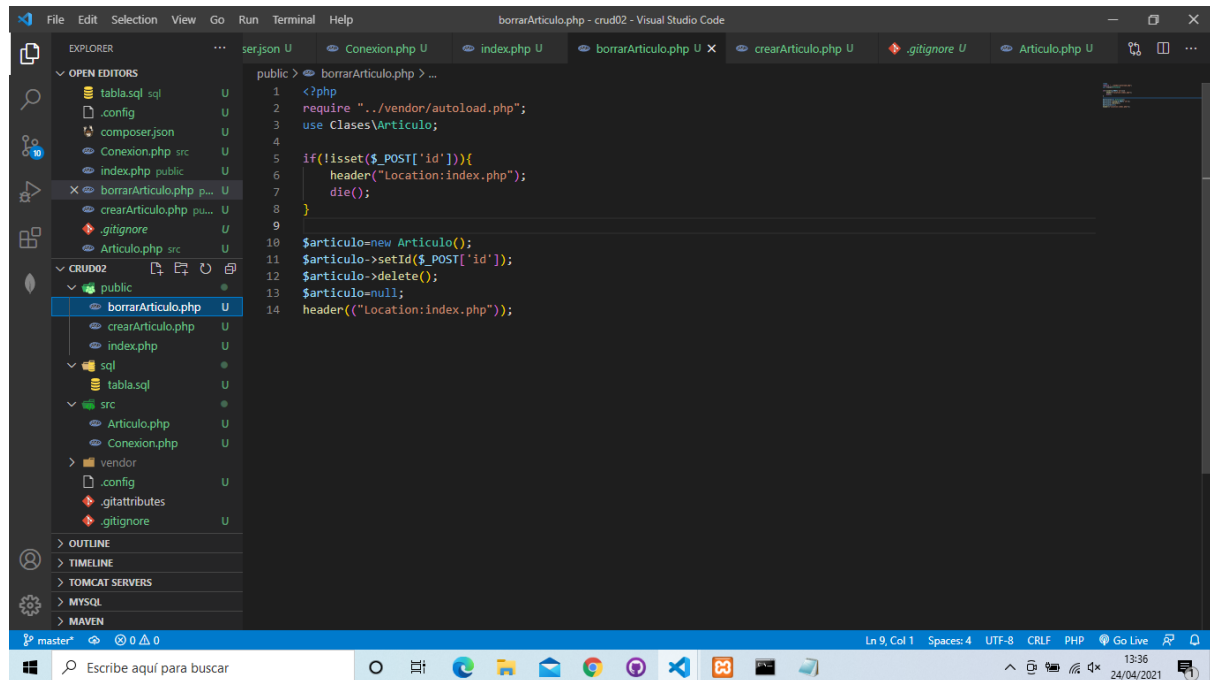


## 4-Borrar artículo

Asignamos al artículo su id correspondiente dentro de index.php para que cada fila sea identificado por su id y que sea recogido con el id hacia el borradoArticulo.php con el id

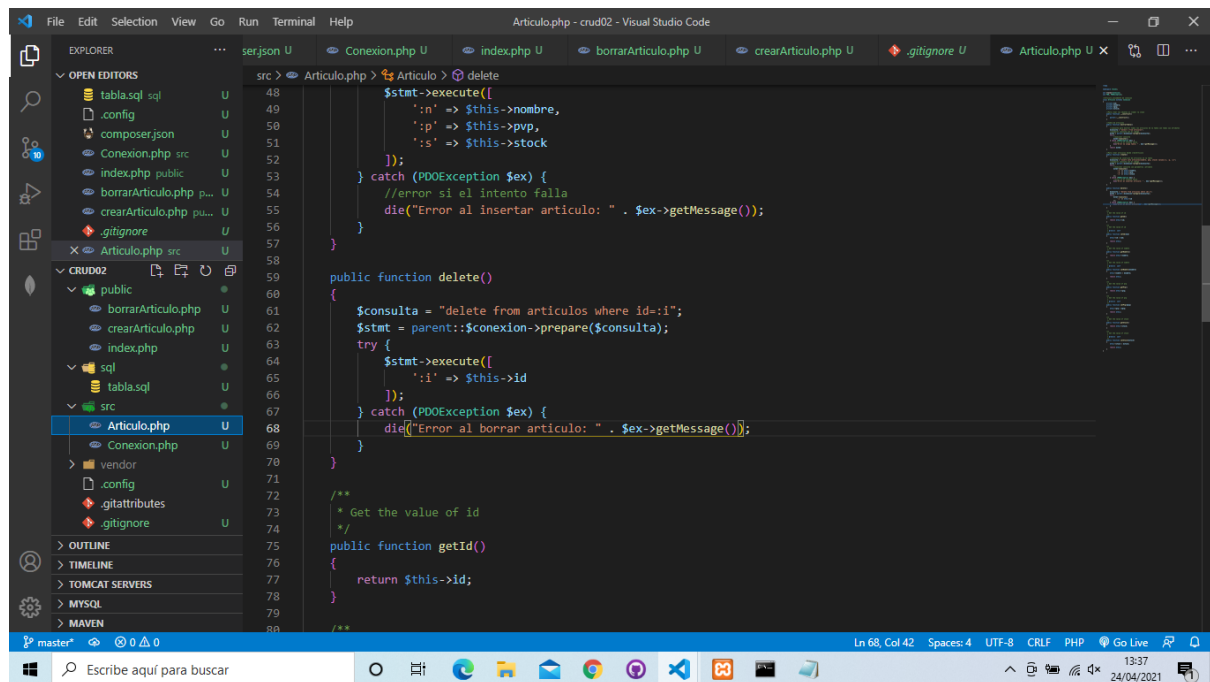


Esta clase es sencilla, solamente llama al método delete y que cargue de nuevo con el artículo cuyo id sea borrado y actualizado de nuevo para observar que el ítem ha sido borrado



```
1 <?php
2 require "../vendor/autoload.php";
3 use Clases\Articulo;
4
5 if(isset($_POST['id'])){
6     header("Location:index.php");
7     die();
8 }
9
10 $articulo=new Articulo();
11 $articulo->setId($_POST['id']);
12 $articulo->delete();
13 $articulo=null;
14 header("Location:index.php");
```

Método de borrar artículos cuyo id coincida con una fila en la tabla index.php



```
48 $stmt->execute([
49     ':n' => $this->nombre,
50     ':p' => $this->pvp,
51     ':s' => $this->stock
52 ]);
53 } catch (PDOException $ex) {
54     //error si el intento falla
55     die("Error al insertar artículo: " . $ex->getMessage());
56 }
57
58 public function delete()
59 {
60     $consulta = "delete from articulos where id=:i";
61     $stmt = parent::$conexion->prepare($consulta);
62     try {
63         $stmt->execute([
64             ':i' => $this->id
65         ]);
66     } catch (PDOException $ex) {
67         die("Error al borrar artículo: " . $ex->getMessage());
68     }
69 }
70
71 /**
72  * Get the value of id
73  */
74 public function getId()
75 {
76     return $this->id;
77 }
78
79 /**
```

Articulos

127.0.0.1/crud02/index.php

Crear articulo

Id	Nombre	PVP	Stock	Acciones
1	Monitor 19	123.45 €	12	Editar Borrar
2	Raton USB	23.95 €	13	Editar Borrar
3	USB 512GB	123.45 €	14	Editar Borrar
4	USB 8GB	12.45 €	14	Editar Borrar
5	SET LAPICEROS	13.15 €	32	Editar Borrar
6	Stick Wifi	23.00 €	62	Editar Borrar
7	Funda 17	13.40 €	92	Editar Borrar
8	Monitor 22	193.65 €	2	Editar Borrar
9	Smart TV 52	453.85 €	92	Editar Borrar
10	SSD 512GB	123.45 €	2	Editar Borrar
11	Tv	12.00 €	12	Editar Borrar
12	Movil	12.12 €	1000	Editar Borrar

Escribe aquí para buscar

## 4-Leer artículo para actualizar

Leemos el artículo a partir de su id que es único, con la siguiente consulta sql buscandolo a partir del id del artículo y devolvemos el objeto en cuestión.

```

File Edit Selection View Go Run Terminal Help
Articulo.php - crud02 - Visual Studio Code

EXPLORER
  .config U
  composer.json U
  Conexion.php U
  index.php U
  borrarArticulo.php U
  crearArticulo.php U
  .gitignore U
  Articulo.php U
  actualizarArticulo.php U
  crud... U
  sql U
  tabla.sql U
  src U
  vendor U
  .config U
  gitattributes U
  OUTLINE
  TIMELINE
  TOMCAT SERVERS
  MYSQL
  MAVEN

src > Articulo.php > read
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97

    });
    } catch (PDOException $ex) {
        die("Error al borrar articulo: " . $ex->getMessage());
    }
}

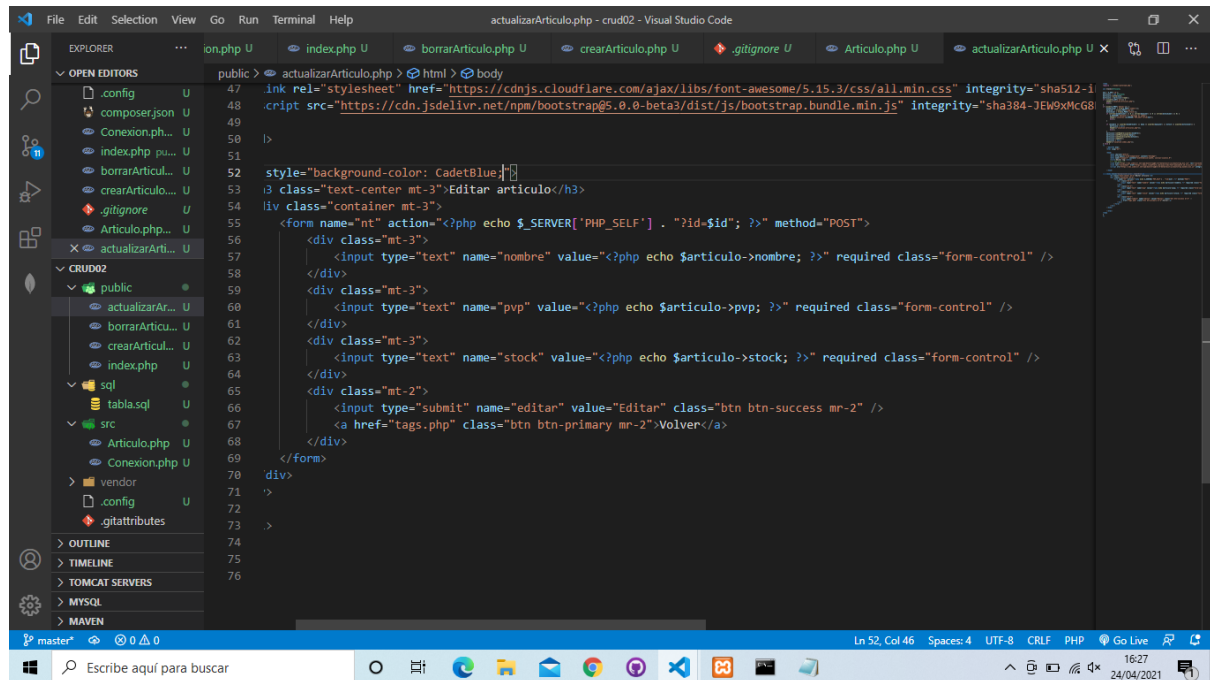
public function read()
{
    $consulta = "select * from articulos where id=:1";
    $stmt = parent::$conexion->prepare($consulta);
    try {
        $stmt->execute([
            ':1' => $this->id
        ]);
    } catch (PDOException $ex) {
        die("Error al leer articulo: " . $ex->getMessage());
    }
    return $stmt->fetch(PDO::FETCH_OBJ);
}

/**
 * Get the value of id
 */
public function getId()
{
    return $this->id;
}

/**
 * Set the value of id
 */
public function setId($id)
{
    $this->id = $id;
}

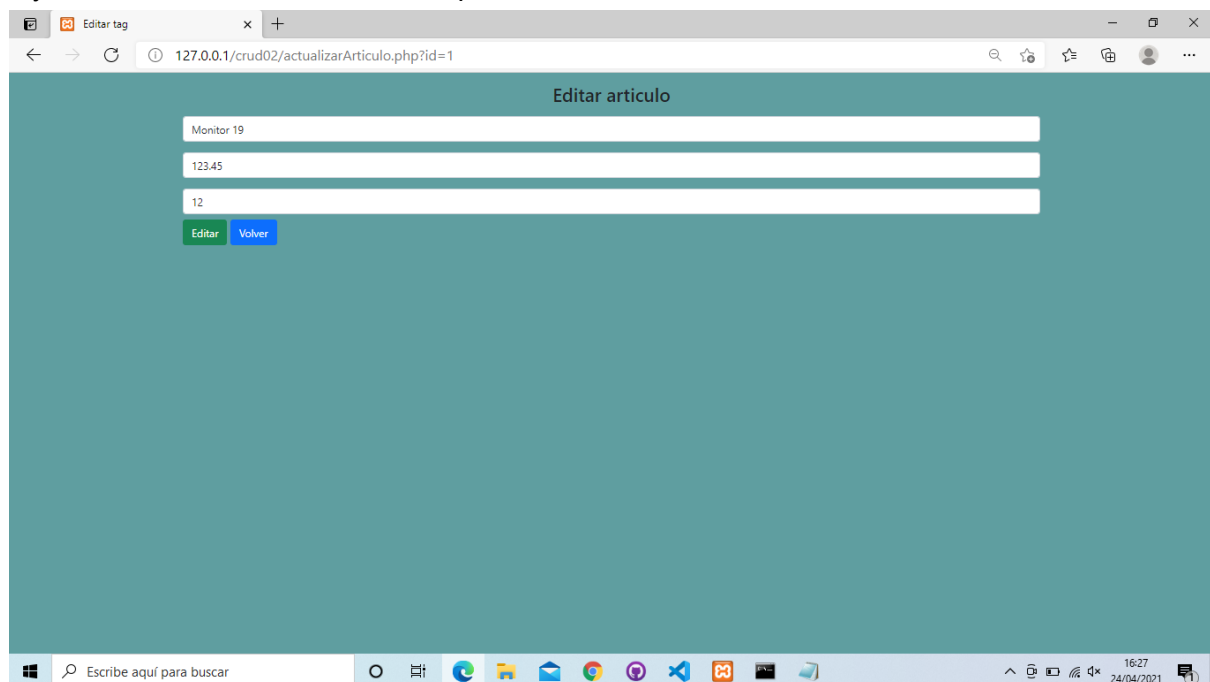
```

Obtener los datos y recibirlos en los inputs, gracias al id obtenido desde index.php obtenemos los atributos correspondientes desde el método read y vamos recogiendo sus atributos.



```
47 <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/5.15.3/css/all.min.css" integrity="sha512-i
48 <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.0.0-beta3/dist/js/bootstrap.bundle.min.js" integrity="sha384-JEW9xMcG8
49
50 >
51
52 <div class="text-center mt-3">
53 <div class="text-center mt-3">
54 <div class="text-center mt-3">
55 <div class="text-center mt-3">
56 <div class="text-center mt-3">
57 <div class="text-center mt-3">
58 <div class="text-center mt-3">
59 <div class="text-center mt-3">
60 <div class="text-center mt-3">
61 <div class="text-center mt-3">
62 <div class="text-center mt-3">
63 <div class="text-center mt-3">
64 <div class="text-center mt-3">
65 <div class="text-center mt-3">
66 <div class="text-center mt-3">
67 <div class="text-center mt-3">
68 <div class="text-center mt-3">
69 <div class="text-center mt-3">
70 <div class="text-center mt-3">
71 <div class="text-center mt-3">
72 <div class="text-center mt-3">
73 <div class="text-center mt-3">
74 <div class="text-center mt-3">
75 <div class="text-center mt-3">
76 <div class="text-center mt-3">
```

Y ya obtendremos los datos correspondientes desde dicho id ofreciendonos sus atributos.



Editar artículo

Monitor 19

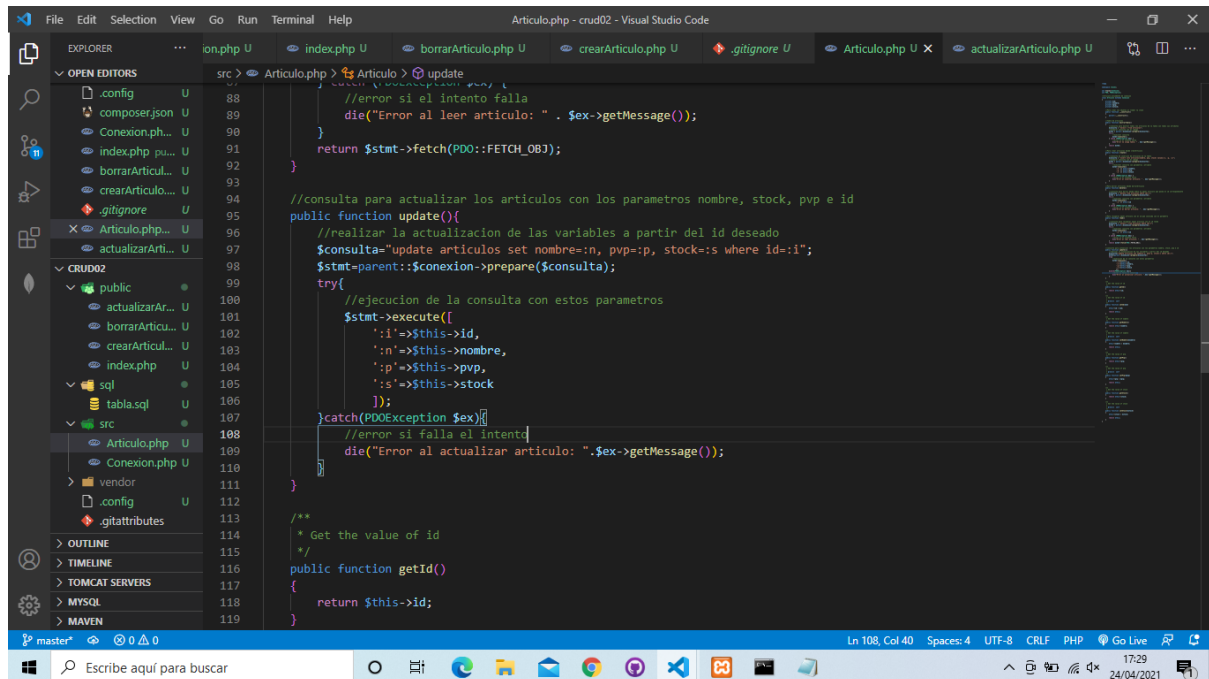
123.45

12

Editar Volver

## 5-Actualizar artículo

Realizamos la consulta para actualizar aquel artículo que ha sido llamado a partir de su id para editar sus atributos (nombre, stock, pvp) y así efectuar la actualización.



```
File Edit Selection View Go Run Terminal Help
Articulo.php - crud02 - Visual Studio Code

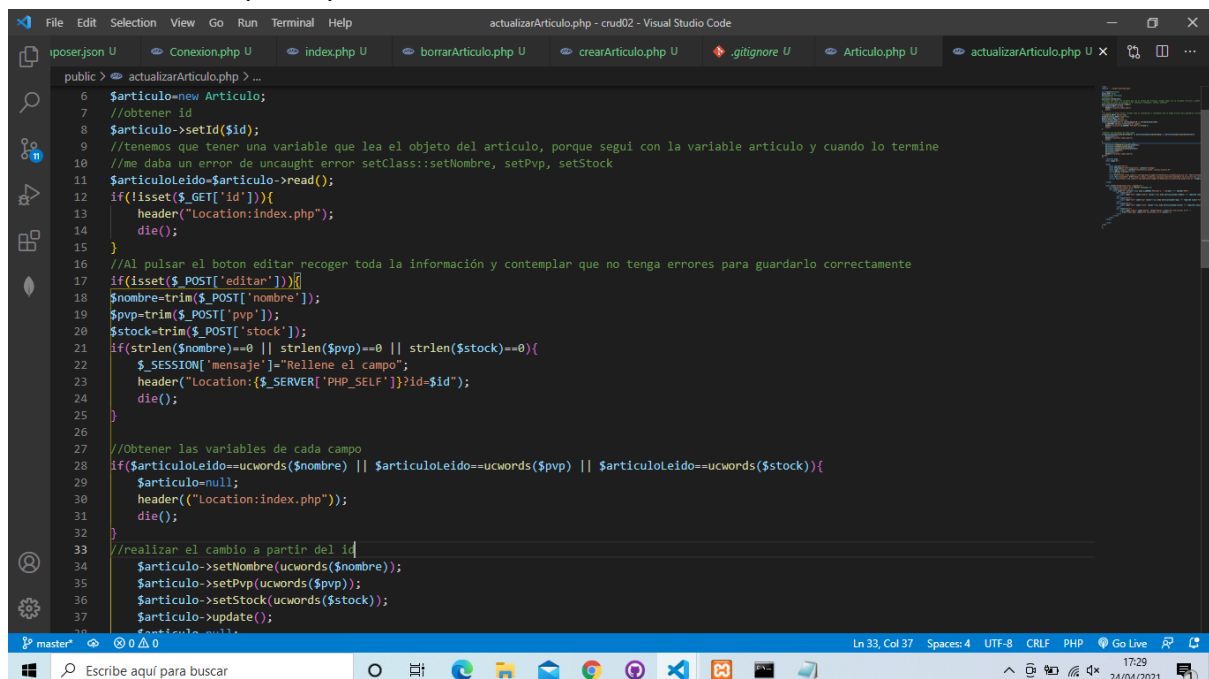
EXPLORER
src > Articulo.php > update
  .config U
  composerjson U
  Conexion.php... U
  index.php pu... U
  borrarArticul... U
  crearArticulo... U
  .gitignore U
  Articulo.php... U
  actualizarArti... U
  CRUD02
  public
    actualizarAr... U
    borrarArticul... U
    crearArticulo... U
    index.php U
  sql
    tabla.sql U
  src
    Articulo.php U
    Conexion.php U
  vendor
  .config U
  .gitattributes

OUTLINE
TIMELINE
TOMCAT SERVERS
MYSQL
MAVEN

master 0 0
Ln 108, Col 40 Spaces: 4 UTF-8 CRLF PHP Go Live
Escribe aquí para buscar
```

```
88 catch(PDOException $ex) {
89     //error si el intento falla
90     die("Error al leer artículo: " . $ex->getMessage());
91 }
92 return $stmt->fetch(PDO::FETCH_OBJ);
93 }
94
95 //consulta para actualizar los artículos con los parámetros nombre, stock, pvp e id
96 public function update(){
97     //realizar la actualización de las variables a partir del id deseado
98     $consulta="update artículos set nombre=:n, pvp=:p, stock=:s where id=:i";
99     $stmt=parent::$conexion->prepare($consulta);
100     try{
101         //ejecución de la consulta con estos parámetros
102         $stmt->execute([
103             ':i'=>$this->id,
104             ':n'=>$this->nombre,
105             ':p'=>$this->pvp,
106             ':s'=>$this->stock
107         ]);
108     }catch(PDOException $ex){
109         //error si falla el intento
110         die("Error al actualizar artículo: " . $ex->getMessage());
111     }
112 }
113
114 /**
115  * Get the value of id
116  */
117 public function getId()
118 {
119     return $this->id;
120 }
```

Recogemos los datos obtenidos de los inputs y los subsanamos por si tiene errores cuando se realice el cambio de datos (espacios en blanco, mayúsculas, etc...) Una vez hecho, realizamos el cambio de información con los setters y una vez obtenidos todos los setters llamar al método update para realizar la consulta.



```
File Edit Selection View Go Run Terminal Help
actualizarArticulo.php - crud02 - Visual Studio Code

composerjson U
Conexion.php U
index.php U
borrarArticulo.php U
crearArticulo.php U
.gitignore U
Articulo.php U
actualizarArticulo.php U

public > actualizarArticulo.php > ...
6 $articulo=new Articulo;
7 //obtener id
8 $articulo->setId($id);
9 //tenemos que tener una variable que lea el objeto del artículo, porque seguimos con la variable artículo y cuando lo termine
10 //me daba un error de uncaught error setClass::setNombre, setPvp, setStock
11 $articuloLeido=$articulo->read();
12 if(!isset($_GET['id'])){
13     header("Location:index.php");
14     die();
15 }
16 //Al pulsar el botón editar recoger toda la información y contemplar que no tenga errores para guardarlo correctamente
17 if(isset($_POST['editar'])){
18     $nombre=trim($_POST['nombre']);
19     $pvp=trim($_POST['pvp']);
20     $stock=trim($_POST['stock']);
21     if(strlen($nombre)==0 || strlen($pvp)==0 || strlen($stock)==0){
22         $_SESSION['mensaje']="Rellene el campo";
23         header("Location:{$_SERVER['PHP_SELF']}?id=$id");
24         die();
25     }
26 }
27 //Obtener las variables de cada campo
28 if($articuloLeido==ucwords($nombre) || $articuloLeido==ucwords($pvp) || $articuloLeido==ucwords($stock)){
29     $articulo=null;
30     header("Location:index.php");
31     die();
32 }
33 //realizar el cambio a partir del id
34 $articulo->setNombre(ucwords($nombre));
35 $articulo->setPvp(ucwords($pvp));
36 $articulo->setStock(ucwords($stock));
37 $articulo->update();
38 $articulo=null;
```

Editar artículo

Monitor 19999

123.46

121111

Editar Volver

Artículos

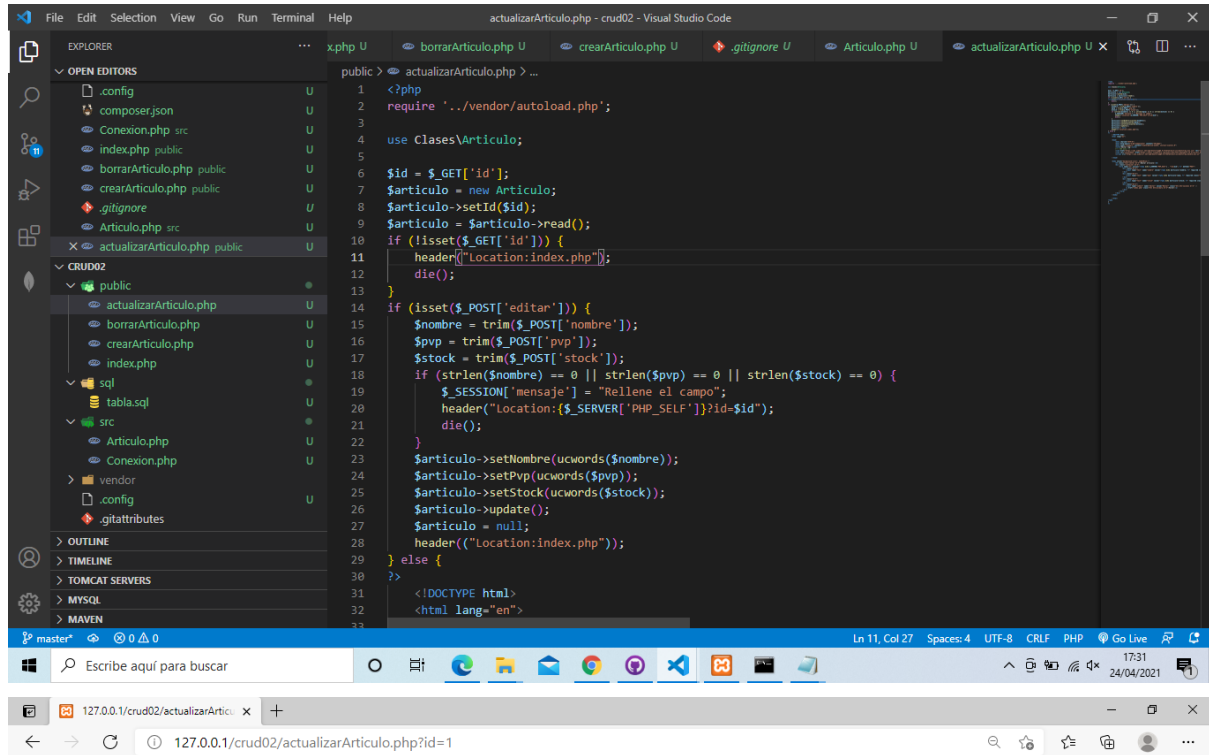
127.0.0.1/crud02/index.php

Crear artículo

Id	Nombre	PVP	Stock	Acciones
1	Monitor 19999	123.46 €	121111	Editar Borrar
2	Raton USB	23.95 €	13	Editar Borrar
3	USB 512GB	123.45 €	14	Editar Borrar
4	USB 8GB	12.45 €	14	Editar Borrar
5	SET LAPICEROS	13.15 €	32	Editar Borrar
6	Stick Wifi	23.00 €	62	Editar Borrar
7	Funda 17	13.40 €	92	Editar Borrar
8	Monitor 22	193.65 €	2	Editar Borrar
9	Smart TV 52	453.85 €	92	Editar Borrar
10	SSD 512GB	123.45 €	2	Editar Borrar

## Errores

No encontraba el por qué me sale este error ya que guardaba los datos correctamente en sus atributos correspondientes, una vez he mirado el proyecto que hicimos en clase encuentre el error. Porque me fallaba en la línea `$articulo = $articulo->read` y entraba en conflicto, tenía que llamarlo con otra variable.



Fatal error: Uncaught Error: Call to undefined method stdClass::setNombre() in C:\Users\Jose\Documents\GitHub\crud02\public\actualizarArticulo.php:23 Stack trace: #0 (main) thrown in C:\Users\Jose\Documents\GitHub\crud02\public\actualizarArticulo.php on line 23

