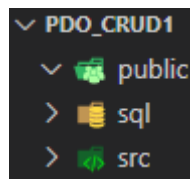


1. Crearemos una estructura de proyecto con las carpetas usuales (public, src, sql)



Utilizando la tabla "user" del proyecto comenzado en clase

```
drop table if exists users;

-----

create table users(
    id int auto_increment primary key,
    nombre varchar(40) not null,
    apellidos varchar(100) not null,
    username varchar(40) unique not null,
    mail varchar(60) unique not null,
    pass varchar(256) not null
);
```

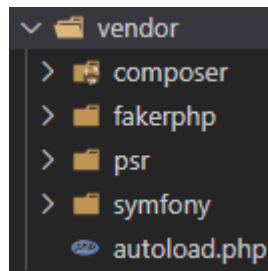
2. Crearemos un archivo de configuración con los parámetros de conexión

```
.config
1  [[opciones]]
2  host=localhost
3  bbdd=pdo_crud1
4  usuario=crud
5  pass=crud
```

3. Utilizaremos autocarga de clases y **Faker** implementado con **composer**

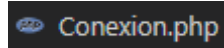
```
PS C:\xampp\htdocs\php\pdo_crud1> composer require fakerphp/faker
Using version ^1.14 for fakerphp/faker
./composer.json has been created
Running composer update fakerphp/faker
Loading composer repositories with package information
Updating dependencies
Lock file operations: 3 installs, 0 updates, 0 removals
- Locking fakerphp/faker (v1.14.1)
- Locking psr/container (1.1.1)
- Locking symfony/deprecation-contracts (v2.2.0)
Writing lock file
Installing dependencies from lock file (including require-dev)
Package operations: 3 installs, 0 updates, 0 removals
- Installing symfony/deprecation-contracts (v2.2.0): Extracting archive
- Installing psr/container (1.1.1): Extracting archive
- Installing fakerphp/faker (v1.14.1): Extracting archive
Generating autoload files
1 package you are using is looking for funding.
Use the `composer fund` command to find out more!
PS C:\xampp\htdocs\php\pdo_crud1>
```

```
composer.json > ...
1  {
2      "name": "yepes/pdo_crud1",
3      "description": "Practica Crud-PDO",
4      "license": "free",
5      "authors": [
6          {
7              "name": "Jose Yepes",
8              "email": "joselitoyepes10@gmail.com"
9          }
10     ],
11     "config": {
12         "optimize-autoloader": true
13     },
14     "autoload": {
15         "psr-4": {
16             "src\\": "src\\classes"
17         }
18     },
19     "require": {
20         "fakerphp/faker": "^1.14"
21     }
22 }
```

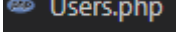


4. Meteremos 50 usuarios de prueba, el primero un usuario **admin** con pass conocida

Antes de nada, creamos la conexión con la base de datos.

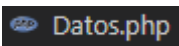


Creamos la clase Usuarios y Datos:

- Usuarios  : En esta clase crearé las funciones del Crud, para ellos necesitaremos crear las variables privadas y sus realizar sus respetivos getters and setters.

En este caso crearé la función create, donde de primeras cifraremos la contraseña introducida.

```
public function create(){
    $pass=hash('sha256', $this->pass);
    $con="insert into users(nombre, apellidos, username, mail, pass) values(:n,:a,:u,:m,:p)";
    $stmt=parent::$conexion->prepare($con);
    try{
        $stmt->execute([
            ':n'=>$this->nombre,
            ':a'=>$this->apellidos,
            ':u'=>$this->username,
            ':m'=>$this->mail,
            ':p'=>$pass
        ]);
    }catch(PDOException $ex){
        die("Error al Crear Usuario: ".$ex->getMessage());
    }
}
```

- Datos  : En esta clase crearé un constructor que determinará la cantidad de usuarios que se quieren añadir, y a su vez el constructor llama a una función callback para realizar crear los usuarios.

```
public function __construct($table, $cantidad){
    $this->faker=Factory::create('es_ES');
    switch($table){
        case "users":
            $this->crearUsuarios($cantidad);
            break;
    }
}
```

En el constructor también indicaremos que tabla queremos modificar, "users".

```

public function crearUsuarios($cantidad){

    $usuario=New Users();
    $usuario->setNombre("Jose");
    $usuario->setApellidos("Yepes");
    $usuario->setUsername("admin");
    $usuario->setMail("jose@gmail.com");
    $pass=hash('sha256', "admin");
    $usuario->setPass($pass);
    $usuario->create();
    $usuario=New Users();

    for($i=0; $i<$cantidad-1;$i++){
        $usuario->setNombre($this->faker->firstName());
        $usuario->setApellidos($this->faker->lastName()." ".$this->faker->lastName());
        $usuario->setUsername($this->faker->unique()->userName);
        $usuario->setMail($this->faker->unique()->email);
        $usuario->setPass($this->faker->sha256);
        $usuario->create();
    }
    $usuario=null;
}

```

En la función “crearUsuarios()” introduciremos un usuario admin y un for donde introduciremos datos mediante faker.

El nombre de usuario y el email debe ser único, y la contraseña la cifraremos mediante “sha256”.

Para poder crear los usuarios he creado en la carpeta public un archivo php, crearUsuarios.

```

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

$usuario=new Datos('users', 50);
echo "Usuarios Creados";

```

localhost/php/pdo_crud1/public/crearUsuarios.php

Ejecutamos el archivo y se crearán los usuarios:


Usuarios Creados

5. Haremos un formulario de "login" plenamente operativo.

Primero crearemos la función para logearnos en la clase “Users”, donde comprobaremos los datos pasados con los de la base de datos

```
public function login(string $username, string $pass){
    $con="select * from users where username=:u AND pass=:p";
    $stmt=parent::$conexion->prepare($con);
    try{
        $stmt->execute([
            ':u'=>$username,
            ':p'=>$pass
        ]);
    }catch(PDOException $ex){
        die("Error al borrar: ".$ex->getMessage());
    }
    $fila=$stmt->fetch(PDO::FETCH_OBJ);
    return ($fila!=null)? true:false;
}
```

Después crearemos un formulario html en un archivo .php llamado login, y le añadimos funcionalidad.



Por último al pulsar el botón de “Login”, se aplicará esta condición donde se recogerán los datos del html, con la peculiaridad de que pasaremos ya formateada la contraseña, y con la función login(), comprobaremos si los datos son o no correctos.

```
if (isset($_POST['login'])) {
    $username = trim($_POST['username']);

    $pass = trim($_POST['pass']);
    $passOK=hash('sha256', $pass);

    $usuario = new Users();

    if($usuario->login($username, $passOK)){
        $usuario=null;
        die("Usuario Logeado");
    }else{
        $usuario=null;
        die("Datos Incorrectos");
    }
}
```