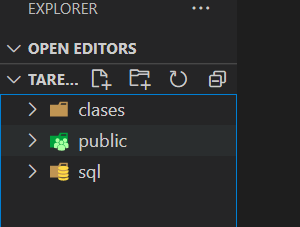
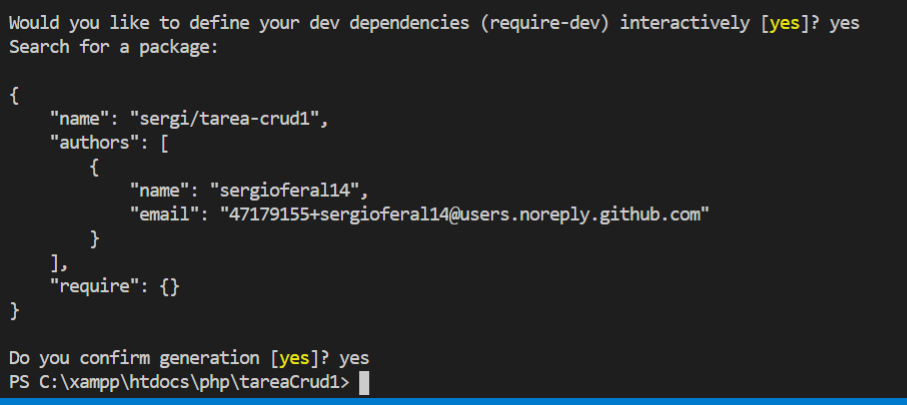
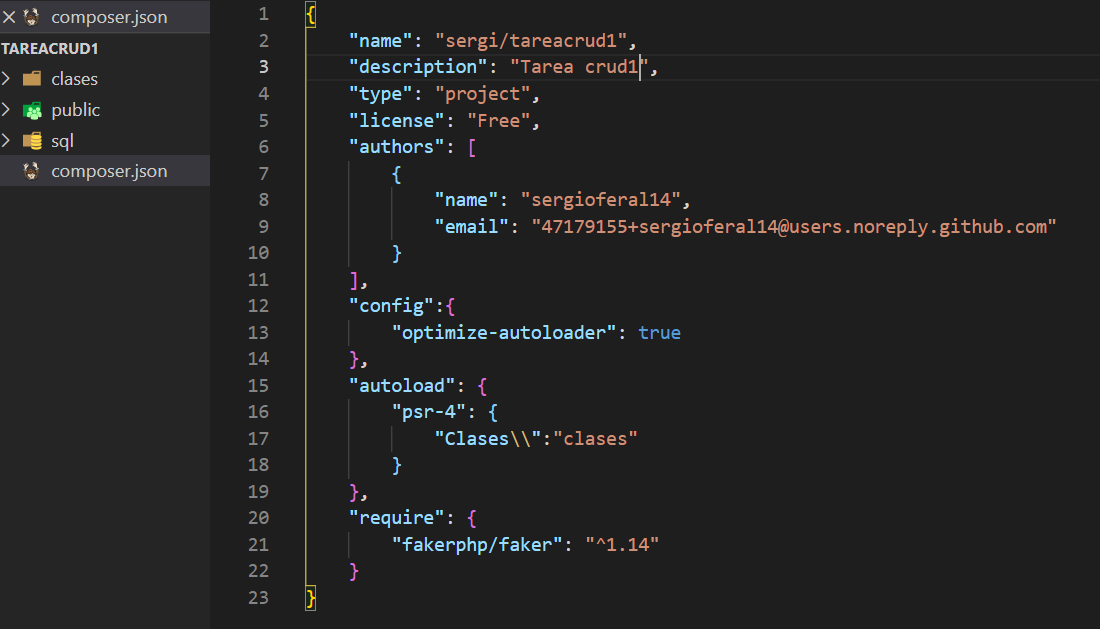
Lo primero que haremos será crear nuestra carpeta del proyecto en la que crearemos una estructura de proyecto con la carpetas usuales (public, src o clases, sql):



A continuación crearemos nuestro *composer.json* por medio de este comando en nuestra terminar del proyecto siguiendo las pautas y preguntas de la creación y le añadiremos el autoload y require para dar uso al faker:

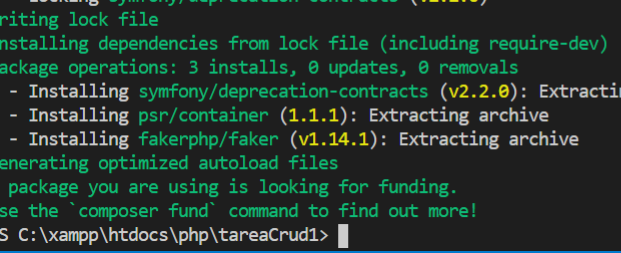




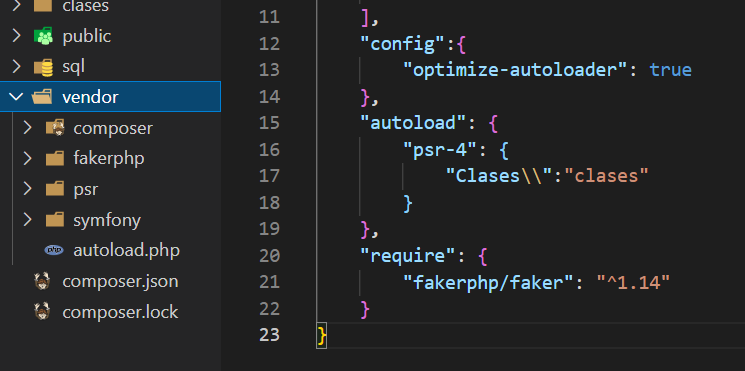


Teniendo ya creado el .json en la línea de comandos instalaremos la carpeta vendor correspondiente con la siguiente orden:

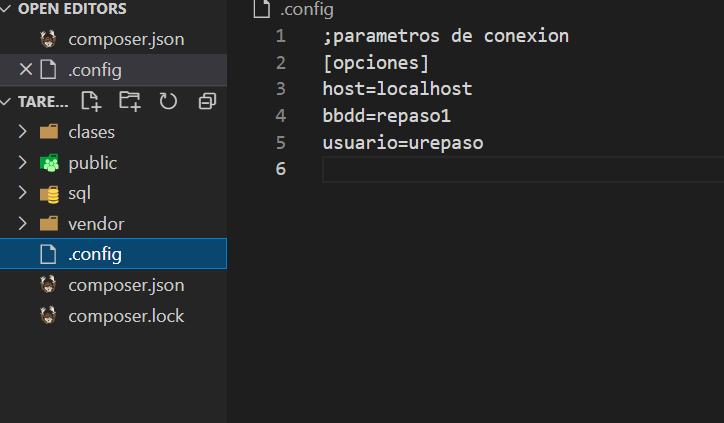
PS C:\xampp\htdocs\php\tareaCrud1> composer update



Y ya tendremos creada nuestra carpeta vendor con el paquete faker:



A continuación crearemos el archivo ***.config*** con el que indicaremos la base de datos que vamos a utilizar y el usuario:



Ahora crearemos la clase ***Conexion.php*** para poder conectarnos con la base de datos indicada del archivo ***.config:***

<?php

namespace Clases;

use PDO;

use PDOException;

//creo la clase, tiene que tener el mismo

//nombre que el archivo,en este caso Conexion

class Conexion{

    protected static $conexion;

    public function \_\_construct()

    {

        if(self::$conexion==null){

            self::crearConexion();

        }

    }

    public static function crearConexion(){

        //1.- leemos el archivo de conficuracion (.config)

        $opciones=parse\_ini\_file('../.config');

        $mibase=$opciones["bbdd"];

        $miUser=$opciones["usuario"];

        //$miPass=$opciones["pass"];

        $miHost=$opciones["host"];

        //2.- creo el dsn (descriptor de servicio) con estos parametros

        $dns="mysql:host=$miHost;dbname=$mibase;charset=utf8mb4";

        //3.- Creo la conexion

        try{

            self::$conexion=new PDO($dns,$miUser);

            //lo siguiente solo para depurar errores

            self::$conexion->setAttribute(PDO::ATTR\_ERRMODE, PDO::ERRMODE\_EXCEPTION);

        }catch(PDOException $ex){

            die("Error al conectar a la BBDD, mansaje: ".$ex->getMessage());

        }

    }

}

Donde podemos encontrar excepciones y métodos para controlar los posibles errores de conexión con la base de datos y otros como el de borrar los creados anterirormente que lo desarrollaremos en otro archivo.

Ahora crearemos ***Datos.php*** para declarar los datos que se crearan con el paquete faker y que corresponderán a los tipos declarados de nuestra base de datos creando antes el usuario admin introduciendo sus datos manualmente y seguidamente los demás los cuales se ocupara el Faker:

<?php

namespace Clases;

require '../vendor/autoload.php';

Use Clases\Users;

use Faker\Factory;

class Datos{

    public $faker;

    public function \_\_construct($tabla,$cantidad)

    {

        $this->faker=Factory::create('es\_ES');

        switch($tabla){

            case "users":

                $this->crearUsuarios($cantidad);

                break;

            case "tags":

                $this->crearTags($cantidad);

                break;

        }

    }

    public function crearUsuarios($n){

        //creamos un usuario admin

        $usuario=New Users();

        //Borro todos los anteriores

        $usuario->borrarTodo();

        $usuario->setNombre("Manolo");

        $usuario->setApellidos("Gil Gil");

        $usuario->setUsername("admin");

        $usuario->setMail("admin@gmail.com");

        $pass=hash('sha256',"secret0");

        $usuario->setPass($pass);

        $usuario->create();

        //creamos el resto de usuarios

        for($i=0; $i<$n-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;

    }

    public function crearTags($n){

    }

}

Seguidamente nos crearemos el archivo ***Users.php*** con el que conseguiremos guardar en variables los datos aleatorios del Faker e insertarlos en la base de datos con dichas consultas,además de eso encontraremos el metododo **borrarTodo()** con el que eliminaremos los creados anteiormente:

<?php

    namespace Clases;

    use Clases\Conexion;

    use PDO;

    use PDOException;

    class Users extends Conexion{

        private $id;

        private $nombre;

        private $apellidos;

        private $username;

        private $mail;

        private $pass;

        public function \_\_construct()

        {

            parent::\_\_construct();

        }

        //CRUD---------------------------------------------------------

        public function create(){

            $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'=>$this->pass,

                ]);

            }catch(PDOException $ex){

                die("Error al insertar usuario: ".$ex->getMessage());

            }

        }

        //-------------------------------------------------------

        //metodos para saber si la tabla tiene datos

        public function borrarTodo(){

            $con="delete from users";

            $stmt=parent::$conexion->prepare($con);

            try{

                $stmt->execute();

            }catch(PDOException $ex){

                die("Error al insertar usuario: ".$ex->getMessage());

            }

        }

Tambien se desarrollara el método para validar el nombre y contraseña del usuario en el futuro login:

public function validarUsuario(String $nombre, String $pass): bool{

        $con="select \* from users where username=:u AND pass=:p";

        $stmt=parent::$conexion->prepare($con);

        try{

                $stmt->execute([

                        ':u'=>$nombre,

                        ':p'=>$pass

                ]);

        }catch(PDOException $ex){

           die("Error al validar usuario: ".$ex->getMessage());

        }

        $fila=$stmt->fetch(PDO::FETCH\_OBJ);

        //si fila=null devuelvo true, false en otro caso

        return ($fila!=null)? true: false;

        }

Y seguido de los respectivos getter y setters de cada variable:

/\*\*

         \* Get the value of id

         \*/

        public function getId()

        {

                return $this->id;

        }

        /\*\*

         \* Set the value of id

         \*

         \* @return  self

         \*/

        public function setId($id)

        {

                $this->id = $id;

                return $this;

        }

        /\*\*

         \* Get the value of nombre

         \*/

        public function getNombre()

        {

                return $this->nombre;

        }

        /\*\*

         \* Set the value of nombre

         \*

         \* @return  self

         \*/

        public function setNombre($nombre)

        {

                $this->nombre = $nombre;

                return $this;

        }

        /\*\*

         \* Get the value of apellidos

         \*/

        public function getApellidos()

        {

                return $this->apellidos;

        }

        /\*\*

         \* Set the value of apellidos

         \*

         \* @return  self

         \*/

        public function setApellidos($apellidos)

        {

                $this->apellidos = $apellidos;

                return $this;

        }

        /\*\*

         \* Get the value of userName

         \*/

        public function getUsername()

        {

                return $this->username;

        }

        /\*\*

         \* Set the value of userName

         \*

         \* @return  self

         \*/

        public function setUsername($username)

        {

                $this->username = $username;

                return $this;

        }

        /\*\*

         \* Get the value of mail

         \*/

        public function getMail()

        {

                return $this->mail;

        }

        /\*\*

         \* Set the value of mail

         \*

         \* @return  self

         \*/

        public function setMail($mail)

        {

                $this->mail = $mail;

                return $this;

        }

        /\*\*

         \* Get the value of pass

         \*/

        public function getPass()

        {

                return $this->pass;

        }

        /\*\*

         \* Set the value of pass

         \*

         \* @return  self

         \*/

        public function setPass($pass)

        {

                $this->pass = $pass;

                return $this;

        }

    }

Vamos a hora con ***CrearDatos.php*** que nos servirá como accionador de la creación de dichos usuarios y que será bastante corto por ahora donde indicaremos el numero de usuarioas a crear que serán 50 en este caso y que los pasaremos a la tabala users de nuestra base de datos:

<?php

require '../vendor/autoload.php';

use Clases\Datos;

$usu=new Datos('users', 50);

echo "<h2>Usarios Creados</h2>";

Y por ultimo el login que de primeras utilizaremos una plantilla de formulario login de bootstrap quedando tal que asi junto con un validador de errores al iniciar el archivo en el navegador:

<!DOCTYPE html>

<html lang="es">

<head>

    <meta charset="UTF-8">

    <meta http-equiv="X-UA-Compatible" content="IE=edge">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.0.0-beta3/dist/css/bootstrap.min.css" rel="stylesheet" integrity="sha384-eOJMYsd53ii+scO/bJGFsiCZc+5NDVN2yr8+0RDqr0Ql0h+rP48ckxlpbzKgwra6" crossorigin="anonymous">

    <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/5.15.3/css/all.min.css" integrity="sha512-iBBXm8fW90+nuLcSKlbmrPcLa0OT92xO1BIsZ+ywDWZCvqsWgccV3gFoRBv0z+8dLJgyAHIhR35VZc2oM/gI1w==" crossorigin="anonymous">

    <title>Login</title>

</head>

<body style="background-color: bisque;">

<h3 class="text-center mt-3">Posts Almería S.A.</h3>

<div class="container mt-3 mb-4">

<?php

        if(isset($\_SESSION['error'])){

            echo "<p class='my-3 p-3 bg-dark text-danger font-weight-bold'>{$\_SESSION['error']}</p>";

            unset($\_SESSION['error']);

        }

        ?>

        <div class="card text-white bg-secondary mb-3 m-auto mt-5" style="max-width: 48rem;">

        <div class="card-header text-center">Login</div>

        <div class="card-body">

            <form name="login" action="<?php echo $\_SERVER['PHP\_SELF']?>" method="POST">

                <div class="form-group">

                    <label for="nu">Nombre de Usuario</label>

                    <input type="text" class="form-control" id="nu" placeholder="Ingrese su nombre" name="nombre" required="">

                </div>

                <div class="form-group mt-3">

                    <label for="np">Password</label>

                    <input type="password" class="form-control" id="np" placeholder="Password" name="pass" required="">

                </div>

                <div class="mt-3">

                <button type="submit" class="btn btn-primary" name="login"><i class="fas fa-sign-in-alt mr-2"></i> Login</button>

                </div>

            </form>

        </div>

    </div>

</div>

</body>

En la parte superior del login nos encontramos con trozo de código de php que nos permitirá validar si los datos introducidos coinciden con los existentes en la base de datos.En caso afirmativo nos saltara de pagina y nos mostrara un mensaje de éxito,y en caso erróneo nos avisara con un mensaje de que los datos no coinciden:

<?php

    session\_start();

    require "../vendor/autoload.php";

    use Clases\Users;

    function mostrarError($texto){

        $\_SESSION['error']=$texto;

        header("Location:{$\_SERVER['PHP\_SELF']}");

        die();

    }

    if(isset($\_POST['login'])){

        //procesare el formulario

        $nombre=trim($\_POST['nombre']);

        $pass=trim($\_POST['pass']);

        if(strlen($nombre)==0 || strlen($pass)==0){

            mostrarError("Rellene los campos !!");

        }

        $passBuena=hash('sha256',$pass);

        $usuario=new Users();

        if($usuario->validarUsuario($nombre,$passBuena)){

            $usuario=null;

            die("Usuario logueado correctamente");

        }else{

        $usuario=null;

        mostrarError("Nombre de usuario incorrecto!!!!");

        }

    }else{

   ?>

El **else** que controla que se vuelva a mostrar la misma pagina si los datos son erróneos se cerrara justo después del contenido html abriendo otra llave **<?php:**

</body>

</html><?php } ?>