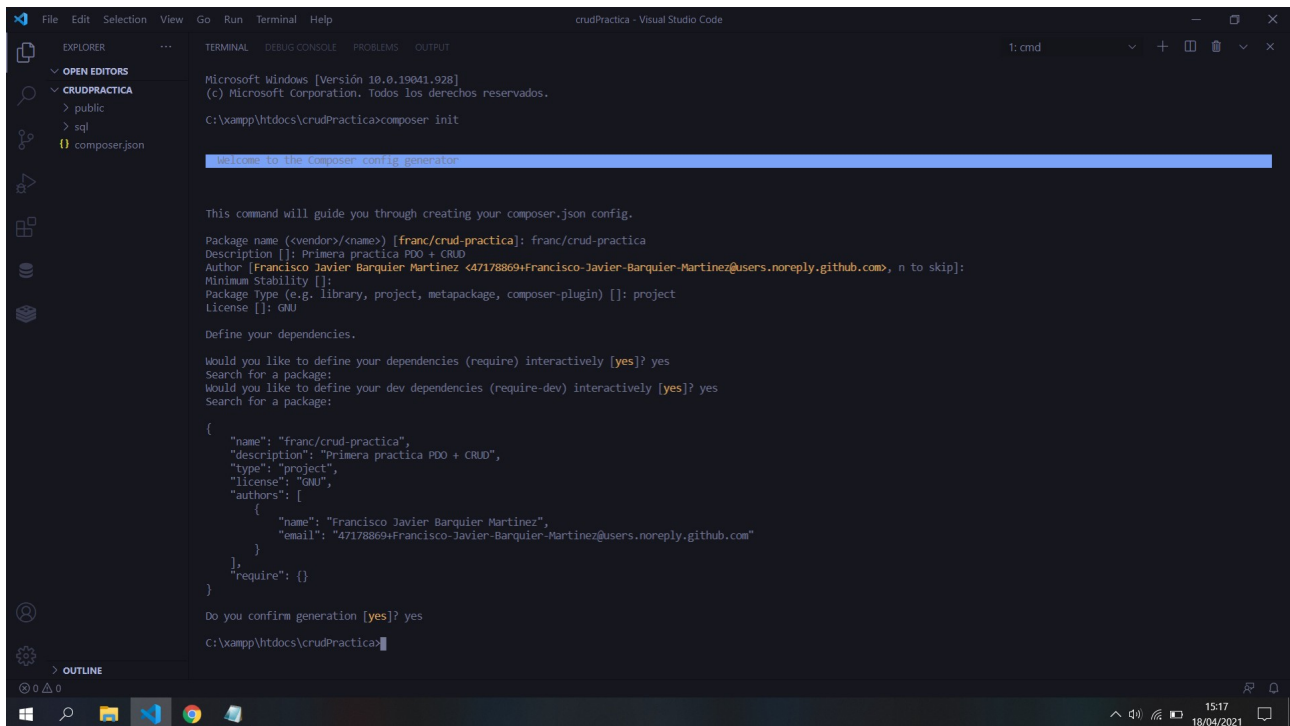


PDO + CRUD1

Creamos nuestra carpeta para el proyecto, en la cual crearemos a su vez la carpeta sql y public. A continuación abrimos una terminal dentro del proyecto y usamos el comando `composer init`.

Ahora rellenamos los campos que nos saldrán a continuación.



```
Microsoft Windows [Versión 10.0.19041.928]
(c) Microsoft Corporation. Todos los derechos reservados.

C:\xampp\htdocs\crudPractica>composer init

Welcome to the Composer config generator

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

Package name (vendor/<name>) [franc/crud-practica]: franc/crud-practica
Description []: Primera practica PDO + CRUD
Author [Francisco Javier Barquier Martinez <47178869+Francisco-Javier-Barquier-Martinez@users.noreply.github.com>, n to skip]:
Minimum stability []:
Package type (e.g. library, project, metapackage, composer-plugin) []: project
License []: GNU

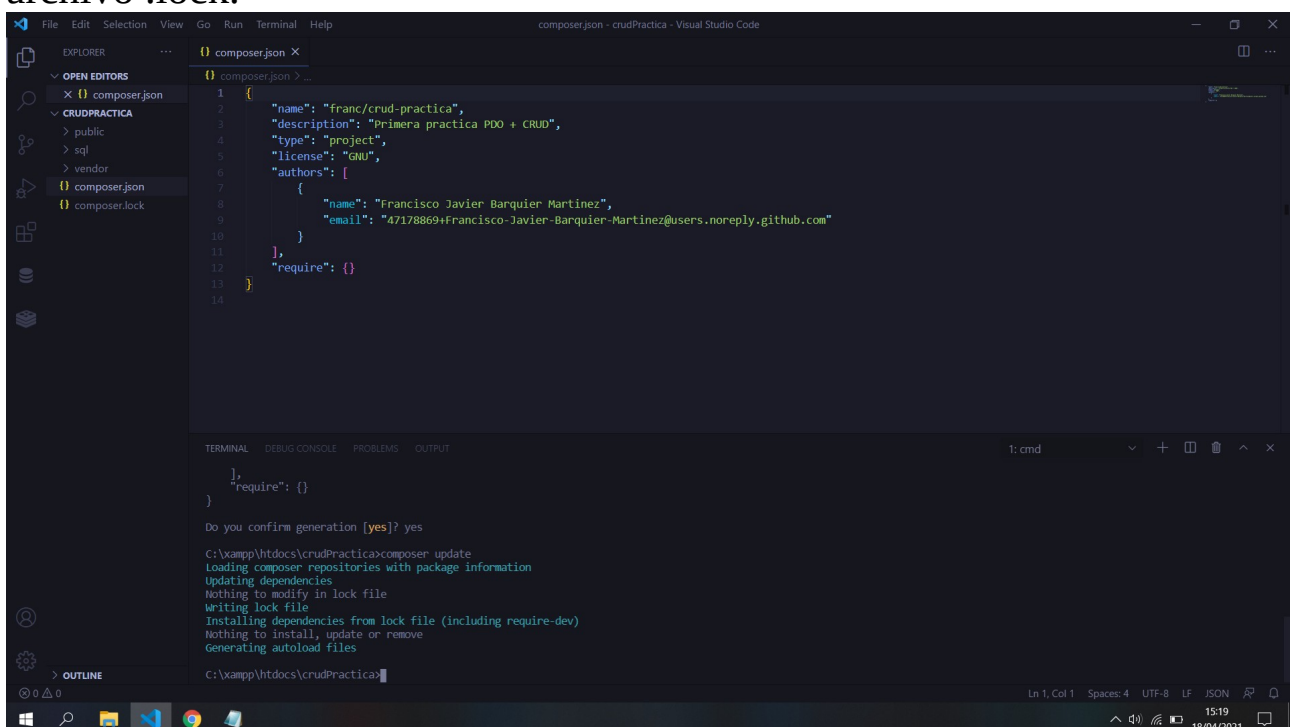
Define your dependencies.

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

{
    "name": "franc/crud-practica",
    "description": "Primera practica PDO + CRUD",
    "type": "project",
    "license": "GNU",
    "authors": [
        {
            "name": "Francisco Javier Barquier Martinez",
            "email": "47178869+Francisco-Javier-Barquier-Martinez@users.noreply.github.com"
        }
    ],
    "require": {}
}

Do you confirm generation [yes]? yes
C:\xampp\htdocs\crudPractica>
```

Seguimos con el comando “`composer update`” para generar el archivo `.lock`.



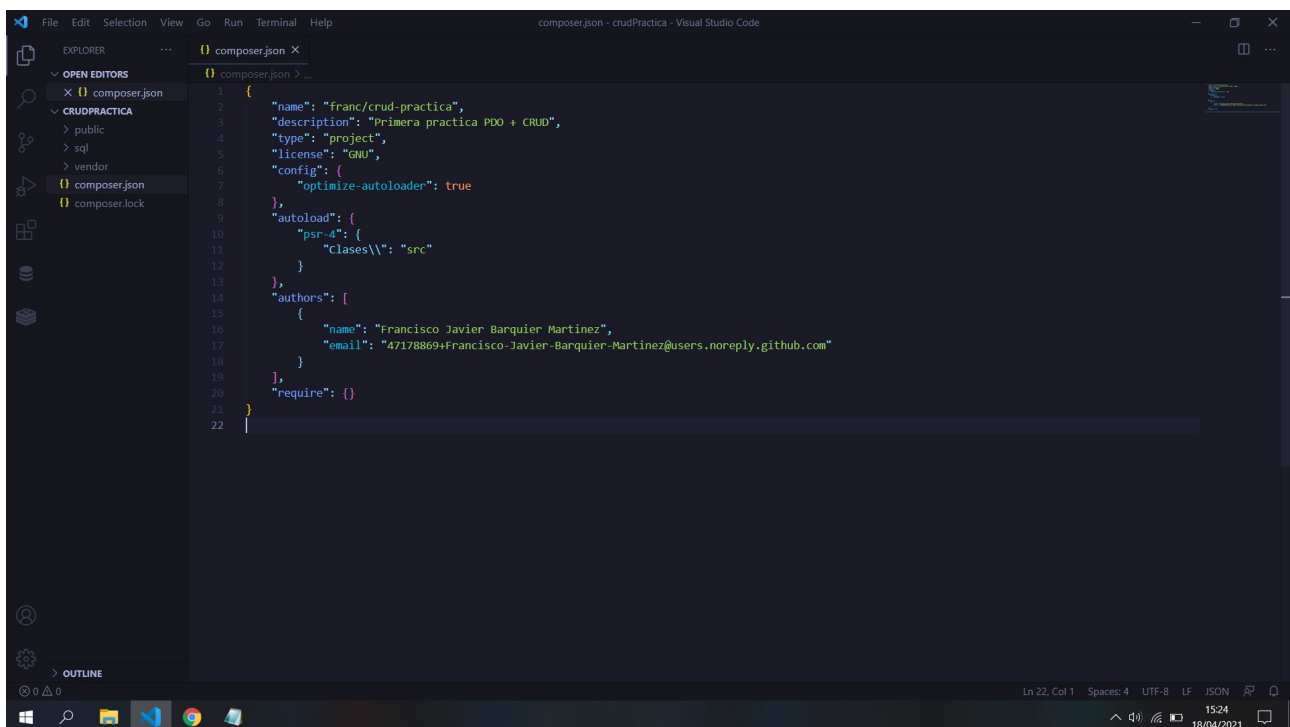
```
1 {
2     "name": "franc/crud-practica",
3     "description": "Primera practica PDO + CRUD",
4     "type": "project",
5     "license": "GNU",
6     "authors": [
7         {
8             "name": "Francisco Javier Barquier Martinez",
9             "email": "47178869+Francisco-Javier-Barquier-Martinez@users.noreply.github.com"
10        }
11    ],
12    "require": {}
13 }
14

Do you confirm generation [yes]? yes

C:\xampp\htdocs\crudPractica>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 autoload files
C:\xampp\htdocs\crudPractica>
```

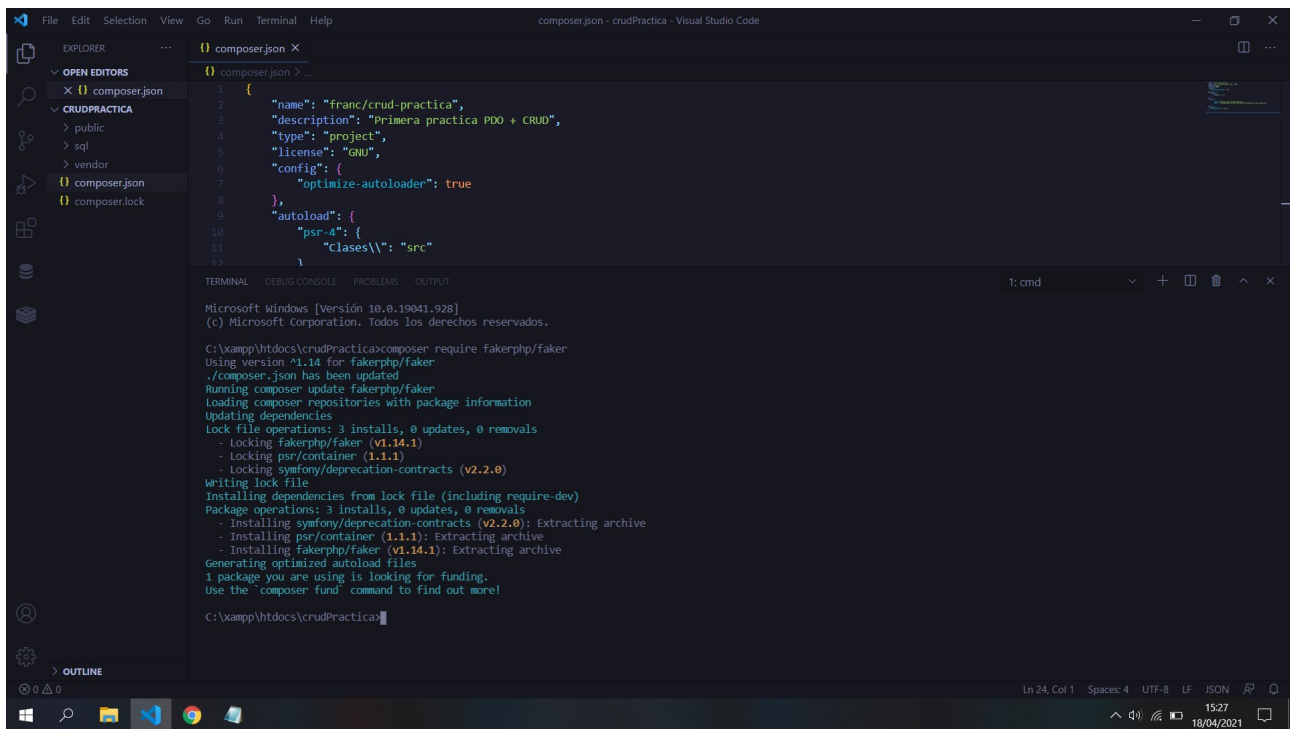
Añadimos las siguientes líneas de código al archivo .lock:

```
"config": {  
    "optimize-autoloader": true  
},  
"autoload": {  
    "psr-4": {  
        "Clases\\": "src"  
    }  
},
```

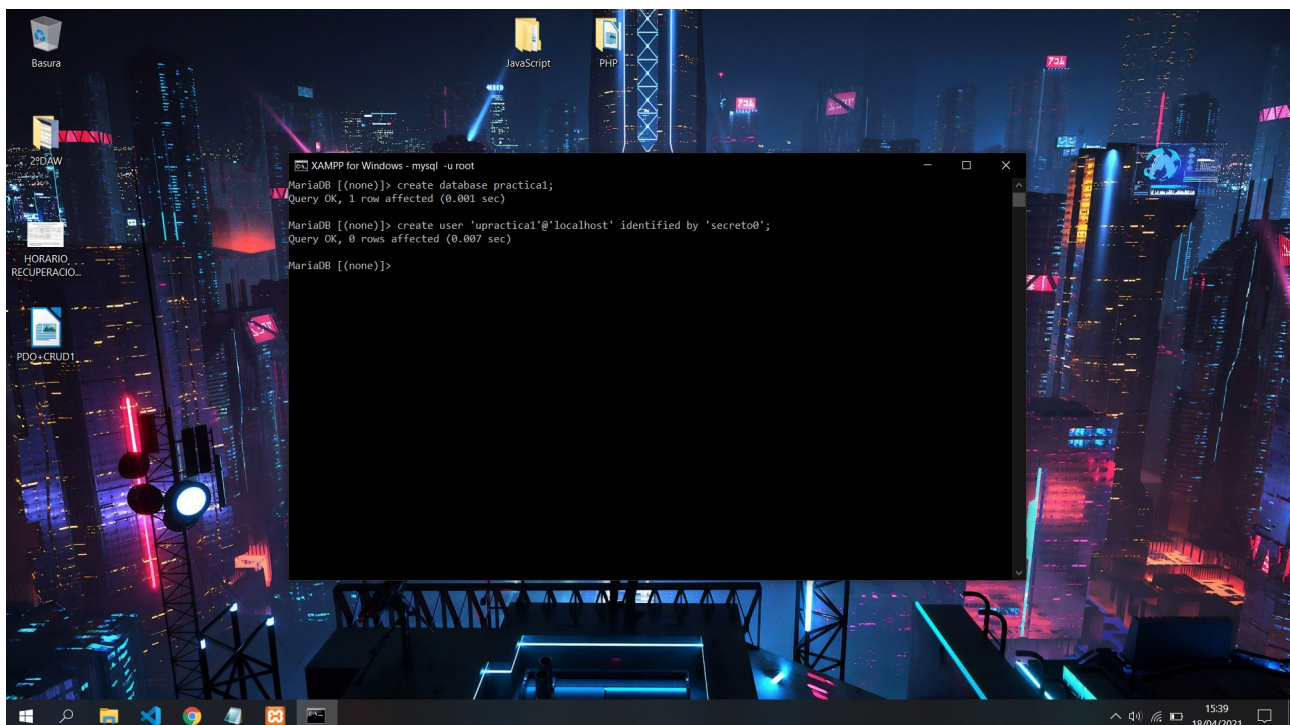


Una vez añadidas esas líneas inserta el comando “composer update”.

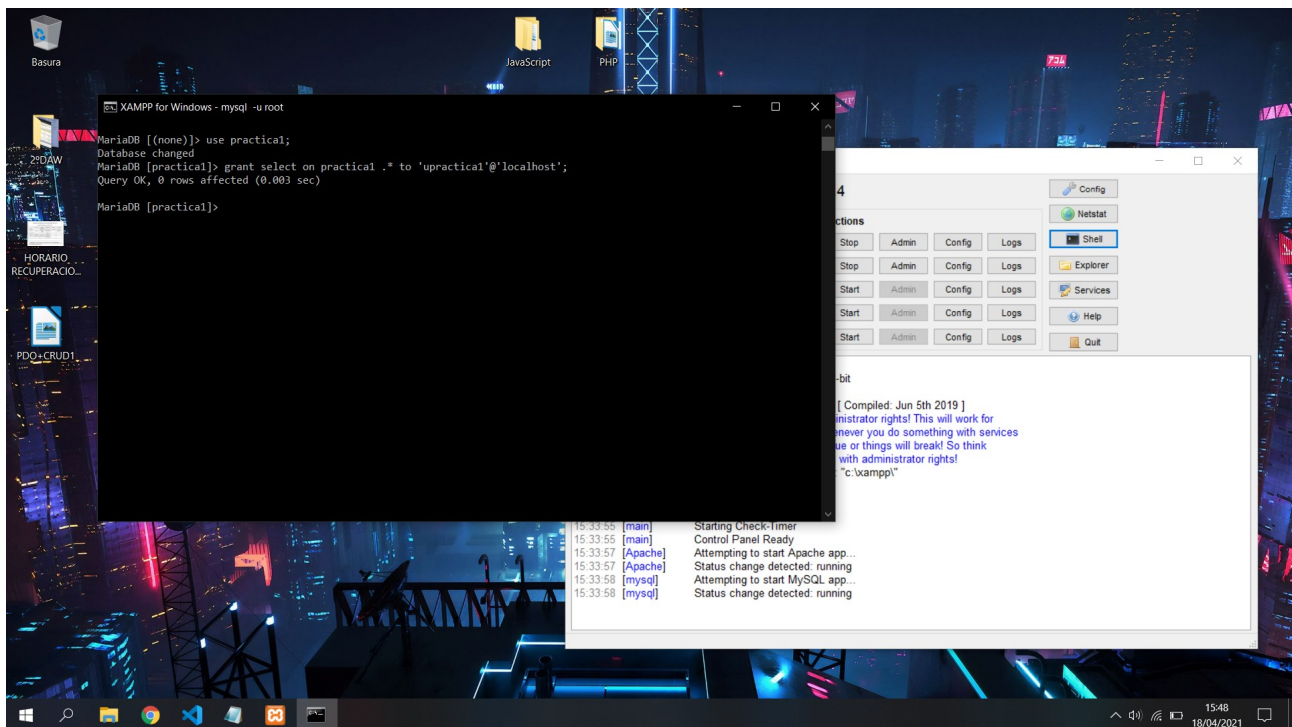
El siguiente paso será instalar las librerías de faker con el comando “composer require fakerphp/faker” en la terminal del proyecto.



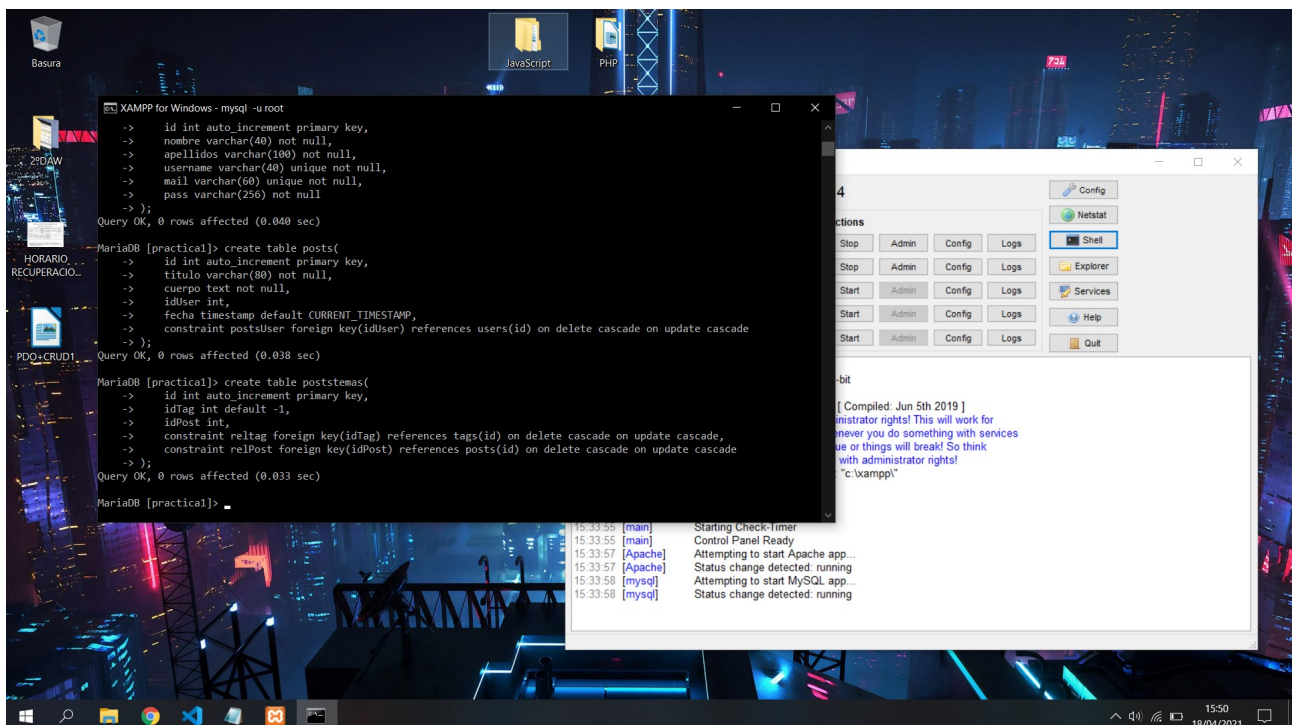
Vamos a proceder a crear nuestra base de datos desde la terminal de xampp en este caso. Llamaré “practica1” a la base de datos con usuario “upractica1” y contraseña secret0.



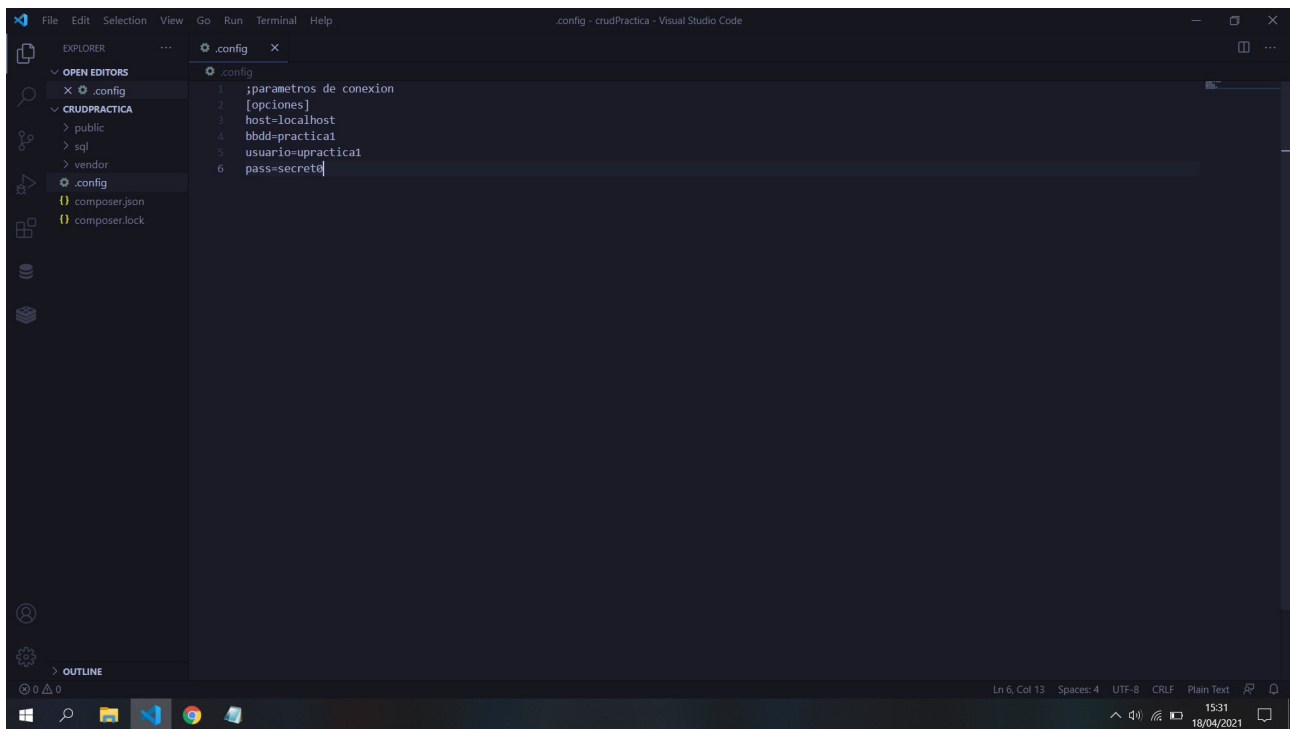
También deberemos dar permisos al usuario para poder acceder a esa tabla



Insertamos la base de datos de prueba en este caso.



Siguiente paso será el archivo .config donde estableceremos nuestro host, base de datos, nombre de usuario y contraseña.

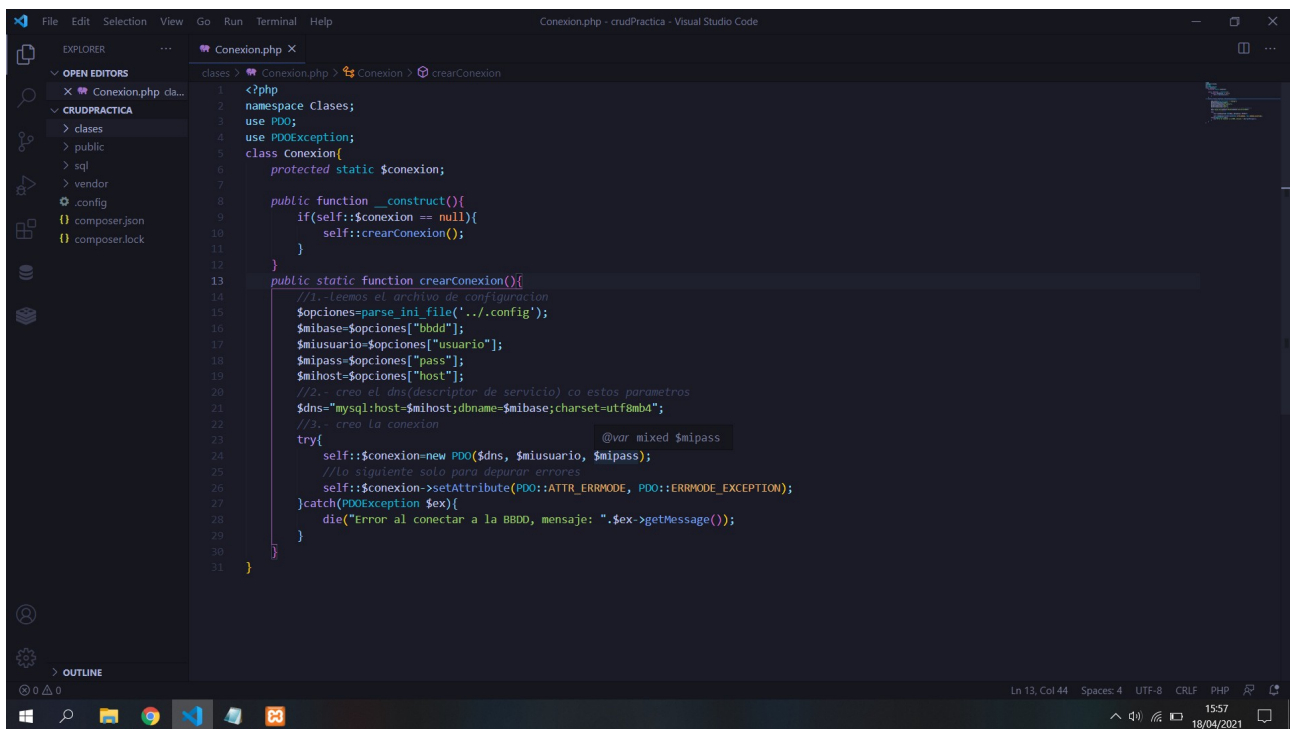


The screenshot shows the Visual Studio Code interface with the Explorer sidebar on the left. The file explorer shows a project structure with folders 'public', 'sql', and 'vendor', and files '.config', 'composer.json', and 'composer.lock'. The main editor window displays the contents of the '.config' file, which contains database connection parameters.

```
.config
1 ;parametros de conexion
2 [opciones]
3 host=localhost
4 bbdd=practical
5 usuario=upractical
6 pass=secret
```

The status bar at the bottom indicates the cursor is at line 6, column 13, with 4 spaces, UTF-8 encoding, CRLF line endings, and Plain Text format. The system clock shows 15:31 on 18/04/2021.

Creamos dentro de clases “Conexion.php” para establecer conexión con la base de datos.

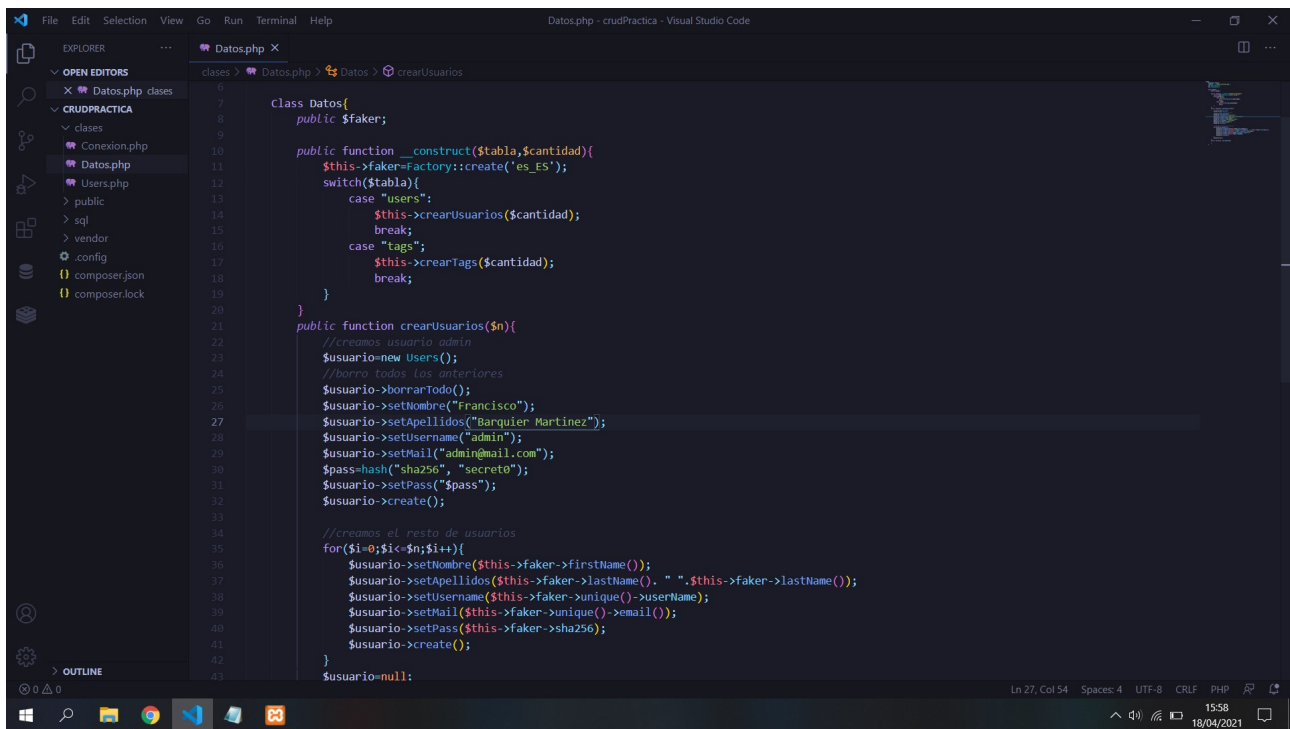


The screenshot shows the Visual Studio Code interface with the Explorer sidebar on the left. The file explorer shows a project structure with folders 'classes', 'public', 'sql', and 'vendor', and files '.config', 'composer.json', and 'composer.lock'. The main editor window displays the contents of the 'Conexion.php' file, which defines a class for database connection.

```
Conexion.php
1 <?php
2 namespace Clases;
3 use PDO;
4 use PDOException;
5 class Conexion{
6     protected static $conexion;
7
8     public function __construct(){
9         if(self::$conexion == null){
10             self::crearConexion();
11         }
12     }
13     public static function crearConexion(){
14         //1.- Leemos el archivo de configuracion
15         $opciones=parse_ini_file('../.config');
16         $mbase=$opciones["bbdd"];
17         $miusuario=$opciones["usuario"];
18         $mipass=$opciones["pass"];
19         $mihost=$opciones["host"];
20         //2.- creo el dns(descriptor de servicio) co estos parametros
21         $dns="mysql:host=$mihost;dbname=$mbase;charset=utf8mb4";
22         //3.- creo la conexion
23         try{
24             self::$conexion=new PDO($dns, $miusuario, $mipass);
25             //Lo siguiente solo para depurar errores
26             self::$conexion->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
27         }catch(PDOException $ex){
28             die("Error al conectar a la BBDD, mensaje: ".$ex->getMessage());
29         }
30     }
31 }
```

The status bar at the bottom indicates the cursor is at line 13, column 44, with 4 spaces, UTF-8 encoding, CRLF line endings, and PHP format. The system clock shows 15:57 on 18/04/2021.

La segunda clase será “Datos.php” para crear el usuario admin y los demas usuarios aleatorios.



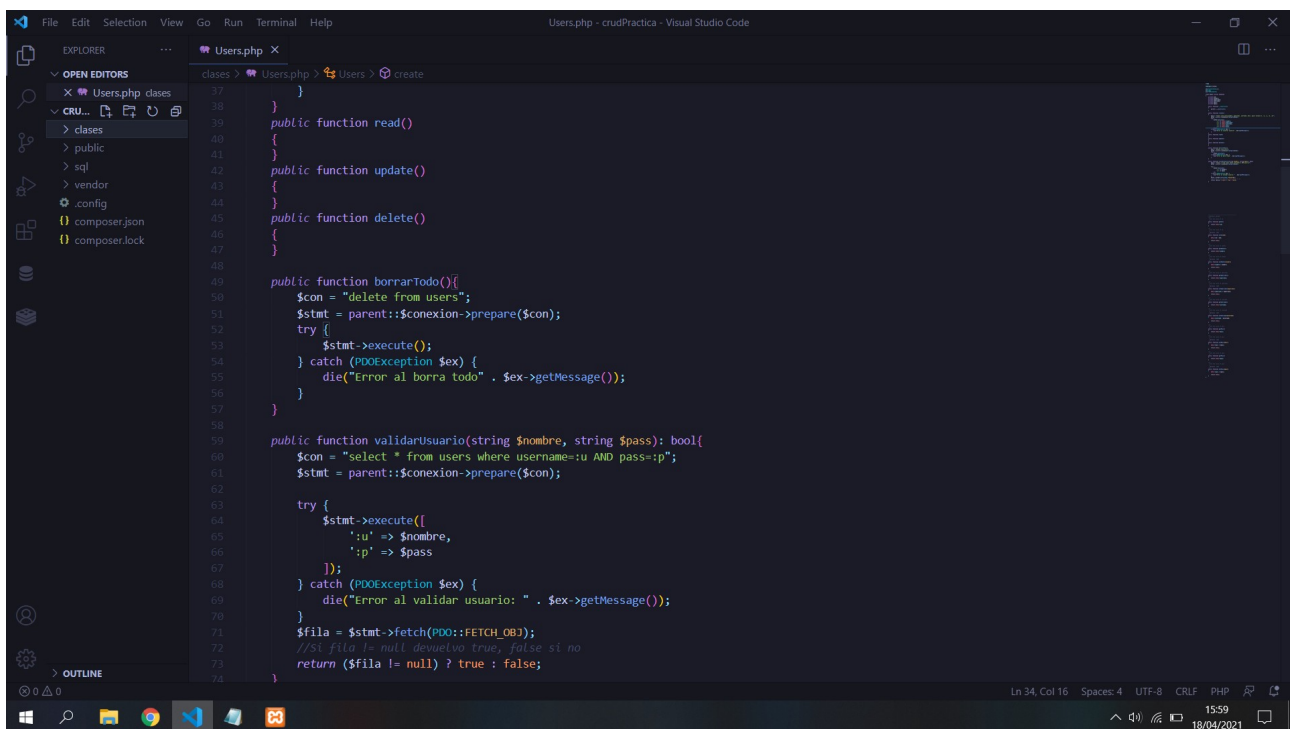
```
Class Datos{
    public $faker;

    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 usuario admin
        $usuario=new Users();
        //borro todos los anteriores
        $usuario->borrarTodo();
        $usuario->setNombre("Francisco");
        $usuario->setApellidos("Barquier Martinez");
        $usuario->setUsername("admin");
        $usuario->setMail("admin@mail.com");
        $pass=hash("sha256", "secret0");
        $usuario->setPass("$pass");
        $usuario->create();

        //creamos el resto de usuarios
        for($i=0;$i<$n;$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;
    }
}
```

La clase “Users.php” tendrá los métodos entre ellos borrarTodo() y para validar los usuarios validarUsuarios().



```
public function read()
{
}

public function update()
{
}

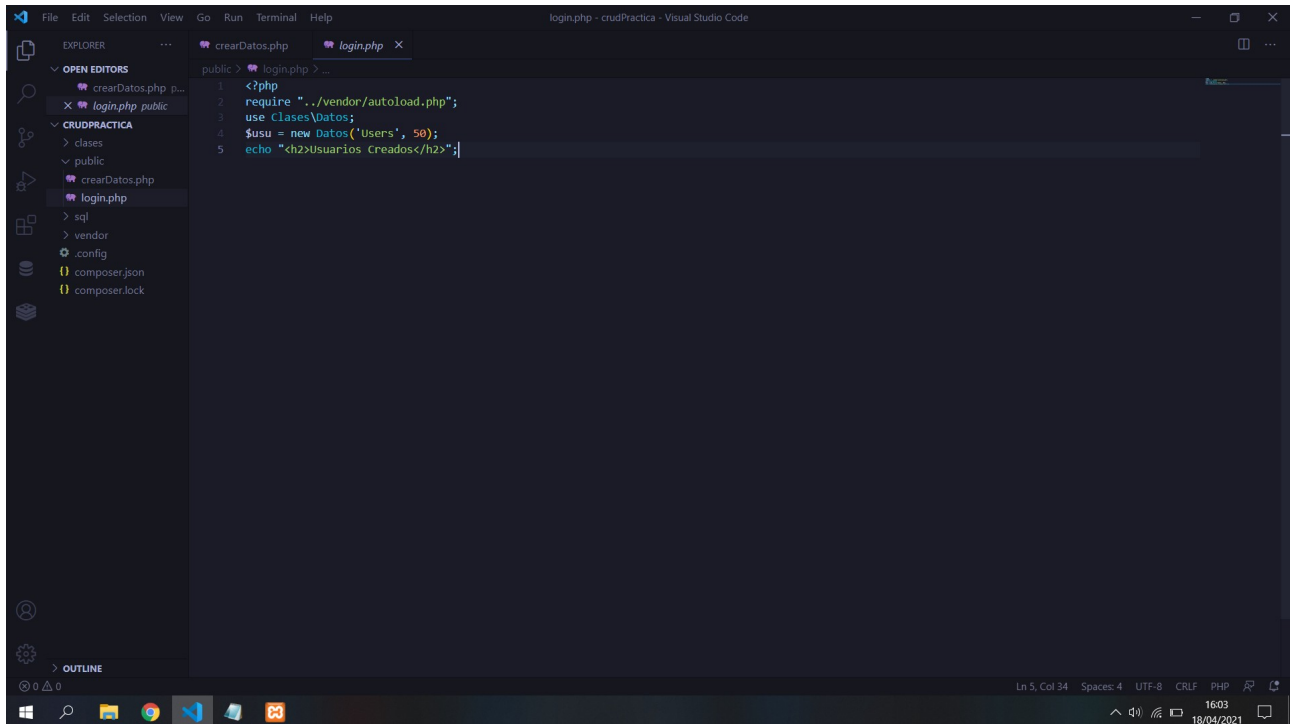
public function delete()
{
}

public function borrarTodo(){
    $con = "delete from users";
    $stmt = parent::$conexion->prepare($con);
    try {
        $stmt->execute();
    } catch (PDOException $ex) {
        die("Error al borra todo" . $ex->getMessage());
    }
}

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 = $stmt->fetch(PDO::FETCH_OBJ);
    //Si fila != null devuelve true, false si no
    return ($fila != null) ? true : false;
}
```

Por ultimo login.php será para generar 50 usuarios.



```
1 <?php
2 require "../vendor/autoload.php";
3 use Clases\Datos;
4 $usu = new Datos('Users', 50);
5 echo "<h2>Usuarios Creados</h2>";
```

Por último tras iniciar crearDatos.php en nuestro xampp iremos a login.php e iniciamos sesión.

