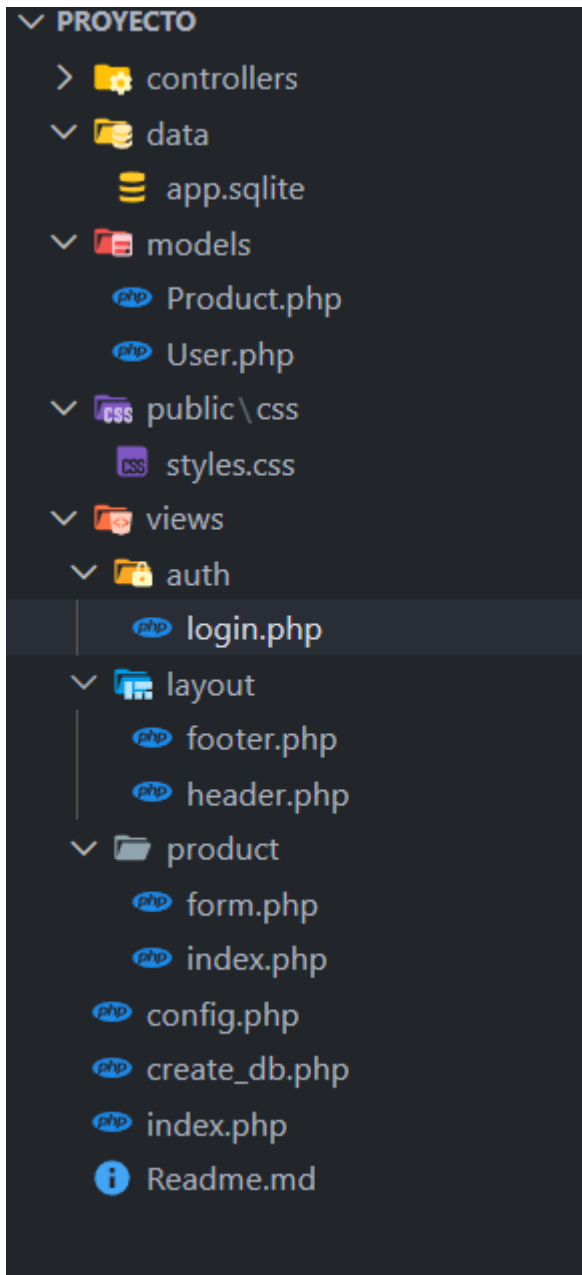


- Proyecto
 - Version 2

Proyecto



Esta es la estructura que he seguido como guion con una de las actividades echas en clase

```

create_db.php
1  <?php
2
3  $db = new PDO(dsn: 'sqlite:data/app.sqlite');
4  $db->setAttribute(attribute: PDO::ATTR_ERRMODE, value: PDO::ERRMODE_EXCEPTION);
5
6  // Tabla de usuarios
7  $db->exec(statement: "
8  CREATE TABLE IF NOT EXISTS users (
9      id INTEGER PRIMARY KEY AUTOINCREMENT,
10     username TEXT UNIQUE NOT NULL,
11     password TEXT NOT NULL,
12     rol TEXT DEFAULT 'user'
13 );
14 ");
15
16 // Usuario admin
17 $adminPass = password_hash(password: "admin", algo: PASSWORD_DEFAULT);
18 $db->exec(statement: "INSERT INTO users (username, password, rol) VALUES ('admin', '$adminPass', 'admin'
19
20
21 // Tabla de productos
22 $db->exec(statement: "
23 CREATE TABLE IF NOT EXISTS products (
24     id INTEGER PRIMARY KEY AUTOINCREMENT,
25     nombre TEXT NOT NULL,
26     descripcion TEXT NOT NULL,
27     precio REAL NOT NULL,
28     stock INTEGER NOT NULL,
29     categoria TEXT NOT NULL
30 );
31 ");
32
33 // Productos de bricolaje
34 $db->exec(statement: "
35 INSERT INTO products (nombre, descripcion, precio, stock, categoria) VALUES
36 ('Taladro percutor 750W', 'Taladro para bricolaje general', 89.99, 12, 'Herramientas'),
37 ('Pintura acrílica blanca 5L', 'Bote de pintura interior', 24.50, 30, 'Pinturas'),

```

Primero he empezado cambiando el archivo para la creacion de la base de datos con los productos que necesito

```
models > Product.php > ...
1  <?php
2  declare(strict_types=1);
3
4  5 references | 0 implementations
5  class Product
6  {
7      0 references
8      public int $id;
9      0 references
10     public string $nombre;
11     0 references
12     public string $descripcion;
13     0 references
14     public float $precio;
15     0 references
16     public int $stock;
17     0 references
18     public string $categoria;
19
20     /**
21      * Obtiene todos los productos.
22      */
23     1 reference | 0 overrides
24     public static function all(PDO $pdo): array
25     {
26         $stmt = $pdo->query(query: 'SELECT * FROM products ORDER BY id DESC');
27         return $stmt->fetchAll(mode: PDO::FETCH_ASSOC);
28     }
29
30     /**
31      * Busca un producto por ID.
32      */
33     1 reference | 0 overrides
34     public static function find(PDO $pdo, int $id): ?array
35     {
36         $stmt = $pdo->prepare(query: 'SELECT * FROM products WHERE id = :id');
37         $stmt->execute(params: [':id' => $id]);
38         $row = $stmt->fetch(mode: PDO::FETCH_ASSOC);
39     }
40 }
```

Despues en Productos he puesto todos los atributos y las funciones de mostrar, eliminar, editar

```

is > User.php > ...
<?php
declare(strict_types=1);

3 references | 0 implementations
v class User
{
    2 references
    public int $id;
    2 references
    public string $username;
    1 reference
    public string $password; // Coincide con la columna de la BD

v /**
 * Busca un usuario por su nombre de usuario.
 */
1 reference | 0 overrides
v public static function findByUsername(PDO $pdo, string $username): ?User
{
    $stmt = $pdo->prepare(query: 'SELECT * FROM users WHERE username = :u LIMIT 1');
    $stmt->execute(params: [':u' => $username]);
    $row = $stmt->fetch(mode: PDO::FETCH_ASSOC);

v if (!$row) {
    return null;
}

    $user = new User();
    $user->id = (int)$row['id'];
    $user->username = $row['username'];
    $user->password = $row['password'];

    return $user;
}
}

```

Y para los usuarios una funcion para ver si existen

Version 2

Cambiar el css y corregir errores