

- Descripción del proyecto
- Tecnologías usadas
- Instalación y puesta en marcha
- Base de datos y migraciones
- Endpoints de la API
 - Fichas de Despiece
 - Ejemplo de respuesta listado paginado
 - Ejemplo de respuesta del detalle de una ficha
- Autenticación con Sanctum
 - Registro
 - Login
 - Logout
- Testing
 - Configuración del entorno de testing
 - Tests creados
 - FichaDespieceTest.php
 - AuthTest.php
 - Ejemplo de un test
 - Ejecutar los tests
 - Qué aprendí con los tests (reflexión personal)

Backend – API REST Fichas de Despiece

API hecha en Laravel para gestionar fichas de despiece de piezas metálicas. Es el backend del proyecto final de ciclo que consume el frontend en React.

Descripción del proyecto

La idea es que una empresa que trabaja con piezas metálicas (aluminio y acero) pueda guardar las fichas de despiece de cada pieza: nombre, material, medidas, cantidad y cliente al que pertenece. También se puede subir un PDF con la ficha técnica.

El backend es una API REST con Laravel 11 que devuelve JSON y consume el frontend en React+TypeScript.

Tecnologías usadas

- PHP 8.x / Laravel 11
 - SQLite (desarrollo y testing) / MySQL (producción)
 - Laravel Sanctum (autenticación por tokens)
 - PHPUnit (testing)
-

Instalación y puesta en marcha

```
# Clonar el repo e ir a la carpeta backend
cd backend

# Instalar dependencias
composer install

# Copiar el .env de ejemplo y configurarlo
cp .env.example .env
php artisan key:generate

# Crear la base de datos SQLite (o configurar MySQL en .env)
touch database/database.sqlite

# Ejecutar migraciones y seeders
php artisan migrate --seed

# Arrancar el servidor
php artisan serve
```

Si usas MySQL, cambia `DB_CONNECTION`, `DB_DATABASE`, `DB_USERNAME` y `DB_PASSWORD` en el `.env` antes de migrar.

Base de datos y migraciones

El proyecto tiene estas tablas principales:

Tabla	Descripción
<code>users</code>	Usuarios de la app (autenticación)

Tabla

Descripción

fichas_despiece	Fichas de despiece con medidas y datos de cliente
personal_access_tokens	Tokens de Sanctum

Para poblar la base de datos con datos de prueba:

```
php artisan db:seed
```

El seeder **FichaDespieceSeeder** crea varias fichas con materiales y clientes de ejemplo.

Users:

Filas: 4									Filtrar 4 filas...	Actualizar a PRO	U	D
	id	name	email	email_verified_at	password	remember_token	created_at	updated_at				
1	1	Fran	fran@gmail.com	2026-02-18 21:...	\$2y\$12\$Z0mWpuFASS.fWpebd3IleDRibrCatT6tUOTta9...	FjFt3esO6t	2026-02-18 21:...	2026-02-18 21:...				
2	2	f	f@gmail.com	NULL	\$2y\$12\$AWtXPpRA/a36T3lcl3oxuVmwn0YzrCizo0TV3l...	NULL	2026-02-18 21:...	2026-02-18 21:...				
3	3	Alfonso	alfonso@gmail.com	NULL	\$2y\$12\$RTFsVsa90eSx3BH1o7YQNety9GUybXk2RmtEc...	NULL	2026-02-20 12:...	2026-02-20 12:...				
4	4	Fdddadads	asdsd@gmail.com	NULL	\$2y\$12\$bYlt5EDOTXE6.zkl.6kVMod7YkGahZZbgYF/q...	NULL	2026-02-20 12:...	2026-02-20 12:...				

Fichas:

Filas: 10										Filtrar 10 filas...	Actualizar a PRO	U	D
	id	nombre_pieza	material	largo	ancho	grosor	cantidad	cliente	archivo_pdf	created_at	updated_at		
1	1	Placa Base Estructuraa	Aluminio	200	150	10	4	Industrias Martínez S.L.	fichas_pdf/4Orbwf...	2026-02-18 21:...	2026-02-19 20:...		
2	2	Perfil Lateral Izquierdo	Aluminio	350	40	3	2	Metalúrgica Pérez	NULL	2026-02-18 21:...	2026-02-18 21:...		
3	3	Perfil Lateral Derecho	Aluminio	350	40	3	2	Metalúrgica Pérez	NULL	2026-02-18 21:...	2026-02-18 21:...		
4	4	Tapa Superior	Aluminio	500	300	5	1	Construcciones López	NULL	2026-02-18 21:...	2026-02-18 21:...		
5	5	Soporte Refuerzo Central	Acero	120	80	8	6	Industrias Martínez S.L.	NULL	2026-02-18 21:...	2026-02-18 21:...		
6	6	Chapa Frontal	Acero	400	250	6	1	Taller Gómez	NULL	2026-02-18 21:...	2026-02-18 21:...		
7	7	Ángulo Esquinero	Aluminio	100	100	4	8	Construcciones López	NULL	2026-02-18 21:...	2026-02-18 21:...		
8	8	Nervio Transversal	Acero	280	30	5	10	Metalúrgica Pérez	NULL	2026-02-18 21:...	2026-02-18 21:...		
9	9	Bandeja Portacables	Aluminio	600	80	2	3	Taller Gómez	NULL	2026-02-18 21:...	2026-02-18 21:...		
10	11	Placa Base Estructura	Aluminio	100	15	20	60	Industrias Martínez S.L.	fichas_pdf/TiH7qjf...	2026-02-19 20:...	2026-02-19 20:...		

Endpoints de la API

Todos los endpoints van con el prefijo **/api/v1/**.

Fichas de Despiece

Método	Ruta	Descripción	Auth
GET	/fichas-despiece	Listado paginado de fichas	No
GET	/fichas-despiece/{id}	Detalle de una ficha	No
GET	/fichas-despiece/{id}/pdf	Descargar el PDF de la ficha	No
POST	/fichas-despiece	Crear una nueva ficha	Sí
PUT/PATCH	/fichas-despiece/{id}	Actualizar una ficha	Sí
DELETE	/fichas-despiece/{id}	Eliminar una ficha	Sí

Ejemplo de respuesta listaado paginado

#ID	NOMBRE DE PIEZA	MATERIAL	LARGO (MM)	ANCHO (MM)	GROSOR (MM)	CANTIDAD	CLIENTE	ACCIONES
#13	Placa Base Estructura	ACERO	200.00	150.00	10.00	4	Industrias Martínez S.L.	<button>Ver</button> <button>Editar</button> <button>Borrar</button>
#14	Perfil Lateral Izquierdo	ALUMINIO	350.00	40.00	3.00	2	Metalúrgica Pérez	<button>Ver</button> <button>Editar</button> <button>Borrar</button>
#15	Perfil Lateral Derecho	ALUMINIO	350.00	40.00	3.00	2	Metalúrgica Pérez	<button>Ver</button> <button>Editar</button> <button>Borrar</button>
#16	Tapa Superior	ALUMINIO	500.00	300.00	5.00	1	Construcciones López	<button>Ver</button> <button>Editar</button> <button>Borrar</button>
#17	Soporte Refuerzo Central	ACERO	120.00	80.00	8.00	6	Industrias Martínez S.L.	<button>Ver</button> <button>Editar</button> <button>Borrar</button>
#18	Chapa Frontal	ACERO	400.00	250.00	6.00	1	Taller Gómez	<button>Ver</button> <button>Editar</button> <button>Borrar</button>
#19	Ángulo Esquinero	ALUMINIO	100.00	100.00	4.00	8	Construcciones López	<button>Ver</button> <button>Editar</button> <button>Borrar</button>
#20	Nervio Transversal	ACERO	280.00	30.00	5.00	10	Metalúrgica Pérez	<button>Ver</button> <button>Editar</button> <button>Borrar</button>
#21	Bandeja Portacables	ALUMINIO	600.00	80.00	2.00	3	Taller Gómez	<button>Ver</button> <button>Editar</button> <button>Borrar</button>
#22	Placa Anchaje	ACERO	90.00	90.00	12.00	12	Industrias Martínez S.L.	<button>Ver</button> <button>Editar</button> <button>Borrar</button>
#11	Placa Base Estructura	ALUMINIO	100.00	15.00	20.00	60	Industrias Martínez S.L.	<button>Ver</button> <button>Editar</button> <button>PDF</button> <button>Borrar</button>
#1	Placa Base Estructuraaa	ALUMINIO	200.00	150.00	10.00	4	Industrias Martínez S.L.	<button>Ver</button> <button>Editar</button> <button>PDF</button> <button>Borrar</button>
#2	Perfil Lateral Izquierdo	ALUMINIO	350.00	40.00	3.00	2	Metalúrgica Pérez	<button>Ver</button> <button>Editar</button> <button>Borrar</button>

Ejemplo de respuesta del detalle de una ficha

Autenticación con Sanctum

Las rutas de escritura (crear, editar, borrar) están protegidas con `auth: sanctum`. Para usarlas hay que obtener un token primero.

Registro

POST /api/v1/auth/register

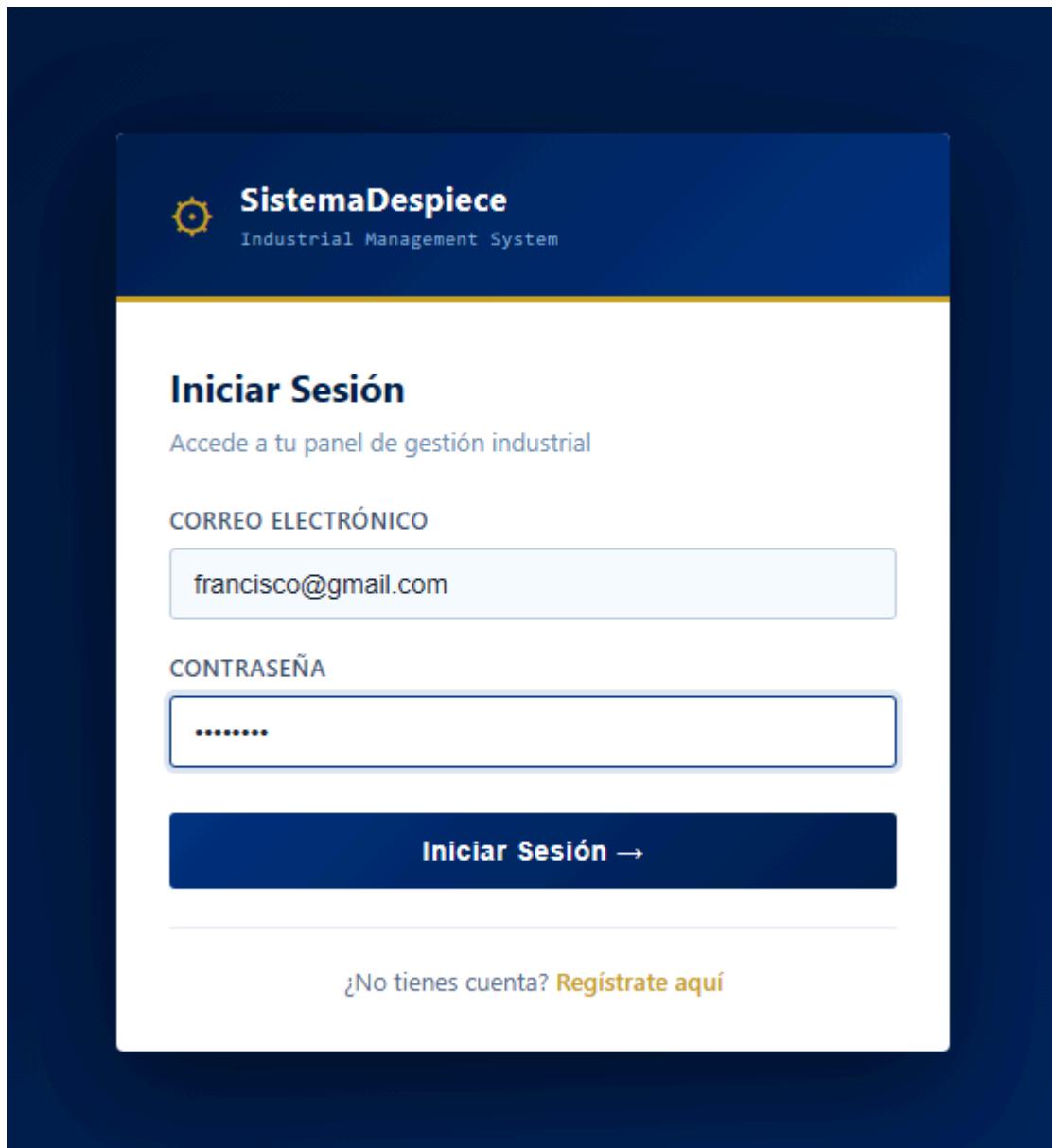
The screenshot shows the 'Crear Cuenta' (Create Account) page of the SistemaDespiece website. The header features a gear icon and the text 'SistemaDespiece' followed by 'Industrial Management System'. The main form fields are labeled 'NOMBRE COMPLETO' (Full Name) containing 'Francisco', 'CORREO ELECTRÓNICO' (Email) containing 'francisco@gmail.com', 'CONTRASEÑA' (Password) containing '.....', and 'CONFIRMAR CONTRASEÑA' (Confirm Password) containing '.....'. A large blue button at the bottom left says 'Crear Cuenta →'. Below the form, a link says '¿Ya tienes cuenta? Inicia sesión aquí'.

Y como podemos ver se guarda dentro de la base de datos una vez guardamos:

2	2	f	f@gmail.com	NULL	\$2y\$12\$AWtXPpRA/a36T3l1cl3oxuTvmwn0
3	3	Alfonso	alfonso@gmail.com	NULL	\$2y\$12\$RTFsVsa90e5x3BH1o7YQNety9GUj
4	4	Fdddads	asdsd@gmail.com	NULL	\$2y\$12\$bYlpt5ED07XE6.zkL6KVMOd7YkGa
5	5	Francisco	francisco@gmail.com	NULL	\$2y\$12\$YdEmtcOJVh9cRnW8oBoyh.4aYl8h
6					

Login

```
POST /api/v1/auth/login
```



Y ahora nos da paso a la pagina principal ya que se ha iniciado sesion:

Fichas de Despiece								+ Nueva Ficha
#ID	NOMBRE DE PIEZA	MATERIAL	LARGO (MM)	ANCHO (MM)	GROSOR (MM)	CANTIDAD	CLIENTE	ACCIONES
#13	Placa Base Estructura	ACERO	200.00	150.00	10.00	4	Industrias Martínez S.L.	Ver Editar Borrar
#14	Perfil Lateral Izquierdo	ALUMINIO	350.00	40.00	3.00	2	Metalúrgica Pérez	Ver Editar Borrar
#15	Perfil Lateral Derecho	ALUMINIO	350.00	40.00	3.00	2	Metalúrgica Pérez	Ver Editar Borrar
#16	Tapa Superior	ALUMINIO	500.00	300.00	5.00	1	Construcciones López	Ver Editar Borrar
#17	Soporte Refuerzo Central	ACERO	120.00	80.00	8.00	6	Industrias Martínez S.L.	Ver Editar Borrar
#18	Chapa Frontal	ACERO	400.00	250.00	6.00	1	Taller Gómez	Ver Editar Borrar
#19	Ángulo Esquinal	ALUMINIO	100.00	100.00	4.00	8	Construcciones López	Ver Editar Borrar
#20	Nervio Transversal	ACERO	280.00	30.00	5.00	10	Metalúrgica Pérez	Ver Editar Borrar
#21	Bandeja Portacables	ALUMINIO	600.00	80.00	2.00	3	Taller Gómez	Ver Editar Borrar
#22	Placa Anchaje	ACERO	90.00	90.00	12.00	12	Industrias Martínez S.L.	Ver Editar Borrar

Logout

Y para cerrar sección simplemente pulsamos el botón de salir, y esto con la autenticación del bearer token nos saca para la página de inicio:

SistemaDespiece
Industrial Management System

[Iniciar Sesión](#) [Registrarse](#)

SISTEMA INDUSTRIAL V1.0

Gestión de Fichas de Despiece Industrial

Plataforma técnica para la gestión de piezas en fábricas de metal y aluminio.
Precisión, trazabilidad y control en cada proceso.

[Acceder al Sistema →](#) [Crear cuenta](#)

100%
TRAZABILIDAD

[PDF](#)
ADJUNTOS TÉCNICOS

[API](#)
REST VERSIONADA

CARACTERÍSTICAS

Todo lo que necesitas para gestionar tu planta

Esta es la parte que más me ha costado entender al principio pero que tiene mucho sentido una vez te pones con ello.

Configuración del entorno de testing

El archivo `phpunit.xml` ya está configurado para usar SQLite en memoria durante los tests, así no se toca la base de datos real:

```
<env name="DB_CONNECTION" value="sqlite"/>
<env name="DB_DATABASE" value=":memory:"/>
```

Esto significa que cada vez que se ejecutan los tests se crea una BD en memoria, se migran las tablas y cuando acaban desaparece. Así los tests son independientes y no ensucian datos reales.

En los tests uso el trait `RefreshDatabase` que aplica las migraciones automáticamente antes de cada test:

```
class FichaDespieceTest extends TestCase
{
    use RefreshDatabase;
```

Tests creados

He creado dos archivos de test en `tests/Feature/`:

`FichaDespieceTest.php`

Prueba los endpoints de las fichas de despiece:

Test	Qué comprueba
<code>test_get_fichas_despiece_list</code>	El listado devuelve 200 y la estructura JSON correcta
<code>test_get_ficha_despiece_detail</code>	El detalle de una ficha devuelve 200 con los datos correspondientes

Test	Qué comprueba
<code>test_get_non_existing_ficha_returns_404</code>	Pedir una ficha con ID inexistente devuelve 404
<code>test_unauthenticated_user_cannot_create_ficha</code>	Sin token, el POST devuelve 401
<code>test_authenticated_user_can_create_ficha</code>	Con token, se crea la ficha y devuelve 201
<code>test_authenticated_user_can_delete_ficha</code>	Con token, se borra la ficha y devuelve 204
<code>test_unauthenticated_user_cannot_delete_ficha</code>	Sin token, el DELETE devuelve 401

AuthTest.php

Prueba el sistema de autenticación:

Test	Qué comprueba
<code>test_user_can_register</code>	Un usuario nuevo se puede registrar y recibe token
<code>test_user_can_login</code>	Login con credenciales correctas devuelve 200 y token
<code>test_authenticated_user_can_logout</code>	Logout con token válido devuelve 200
<code>test_invalid_credentials_return_error</code>	Contraseña incorrecta devuelve 422
<code>test_cannot_register_with_duplicate_email</code>	Email duplicado devuelve 422 con error de validación
<code>test_unauthenticated_request_returns_401</code>	Logout sin token devuelve 401

Ejemplo de un test

```

/**
 * Comprueba que el listado de fichas devuelve 200 y la estructura correcta.
 */
public function test_get_fichas_despiece_list(): void
{
    $response = $this->getJson('/api/v1/fichas-despiece');

    // Primero comprobamos que el status sea 200
    $response->assertStatus(200);

    // Comprobamos que la estructura JSON sea la esperada (paginada)
    $response->assertJsonStructure([
        'data' => [
            '*' => [
                'id',
                'nombre_pieza',
                'material',
                'largo',
                'ancho',
                'grosor',
                'cantidad',
                'cliente',
                'archivo_pdf_url',
                'archivo_pdf_nombre',
                'created_at',
                'updated_at',
            ],
        ],
        'current_page',
        'last_page',
        'per_page',
        'total',
    ]);

    // Comprobamos que entre los datos viene la ficha que sabemos que existe
    $response->assertJsonFragment([
        'nombre_pieza' => 'Placa Base Estructura',
    ]);

    // Sabemos que hay 3 fichas, el total debe ser 3
    $response->assertJsonFragment(['total' => 3]);
}

```

Ejecutar los tests

```
php artisan test
```

```
C:\Users\franp\OneDrive\Escritorio\Ciclo Superior 3\Proyecto React+Laravel\backend>php artisan test

PASS Tests\Unit\ExampleTest
✓ that true is true

PASS Tests\Feature\AuthTest
✓ user can register 0.24s
✓ user can login 0.03s
✓ authenticated user can logout 0.02s
✓ invalid credentials return error 0.01s
✓ cannot register with duplicate email 0.01s
✓ unauthenticated request returns 401 0.01s

PASS Tests\Feature\ExampleTest
✓ the application returns a successful response 0.02s

PASS Tests\Feature\FichaDespieceTest
✓ get fichas despiece list 0.02s
✓ get ficha despiece detail 0.01s
✓ get non existing ficha returns 404 0.01s
✓ unauthenticated user cannot create ficha 0.01s
✓ authenticated user can create ficha 0.01s
✓ authenticated user can delete ficha 0.01s
✓ unauthenticated user cannot delete ficha 0.01s

Tests: 15 passed (93 assertions)
Duration: 0.58s
```

Qué aprendí con los tests (reflexión personal)

En la UD7 explica que el formato TDD consiste en escribir los tests ANTES de hacer el código. Al principio me parece una locura porque ¿cómo escribes un test de algo que no existe? Pero tiene sentido: primero defines cómo debe funcionar y luego codificas para satisfacer esos tests.

Yo lo apliqué: primero diseñé los tests pensando en cómo debe funcionar la API, y luego fui ajustando el código para que pasasen. Por ejemplo, descubrí que el método `store()` devolvía 200 cuando debería devolver 201 al crear un recurso. Los tests me lo dijeron directamente, sin tener que probarlo a mano.

Sin tests probablemente no lo habría notado hasta que alguien del front me dijese que algo iba raro.