# Informe Técnico - Desarrollo API Proyectos (EVU1)

Fecha: 2025-09-07

Autor: Equipo de Desarrollo (Tech Solutions)

## Resumen ejecutivo

Se implementó un API REST protegido por JWT para la gestión de proyectos junto con pruebas automáticas que validan los requisitos funcionales solicitados. Este informe documenta los cambios, la evidencia (salidas de comandos y resultados de pruebas) y conclusiones.

## Requerimientos y cobertura

* POST /api/proyectos: crear proyecto, todos los campos requeridos, responde 201. — CUMPLIDO
* GET /api/proyectos: listar todos, 200, devuelve arreglo vacío si no hay registros. — CUMPLIDO
* GET /api/proyectos/{id}: devuelve 200 con datos o 404 si no existe. — CUMPLIDO
* PUT/PATCH /api/proyectos/{id}: actualizar, devuelve 200 y datos actualizados (404 si no existe). — CUMPLIDO
* DELETE /api/proyectos/{id}: elimina, devuelve 204 y cuerpo vacío (404 si no existe). — CUMPLIDO

## Evidencias incluidas

1. Archivo de colección Postman: postman/EVU1-api-collection.json (requests de register/login y endpoints de proyectos).
2. Tests automatizados: tests/Feature/ProyectoApiTest.php (Pest / PHPUnit).
3. Salida de php artisan route:list --path=api.
4. Salida de ejecución de pruebas (Pest).
5. Archivos modificados/creados relevantes.

## Archivos creados / modificados

* app/Http/Controllers/Api/ProyectoApiController.php — Controlador API con index, store, show, update, destroy.
* routes/api.php — Rutas API actualizadas (uso directo de middleware JwtMiddleware::class).
* app/Http/Middleware/JwtMiddleware.php — Middleware existente que valida token JWT.
* app/Models/Proyectos.php — Modelo con $fillable apropiado (incluye created\_by).
* postman/EVU1-api-collection.json — Colección Postman para pruebas.
* tests/Feature/ProyectoApiTest.php — Tests Feature con RefreshDatabase y JWT.

## 1) Salida de php artisan route:list --path=api

( POST api/login ............................................................................................. AuthController@login  
 GET|HEAD api/proyectos .............................................................................. Api\ProyectoApiController@index   
 POST api/proyectos .............................................................................. Api\ProyectoApiController@store   
 GET|HEAD api/proyectos/{id} .......................................................................... Api\ProyectoApiController@show   
 PUT|PATCH api/proyectos/{id} ........................................................................ Api\ProyectoApiController@update   
 DELETE api/proyectos/{id} ....................................................................... Api\ProyectoApiController@destroy   
 POST api/register ....................................................................................... AuthController@register   
  
 Showing [7] routes)

![Captura - rutas API](./images/route\_list.png)

## 2) Salida de ejecución de pruebas (Pest)

(PHPUnit 11.5.15 by Sebastian Bergmann and contributors.  
  
Runtime: PHP 8.2.12  
Configuration: C:\laragon\www\EVU1\phpunit.xml  
  
  
 PASS Tests\Feature\ProyectoApiTest  
 ✓ list projects returns empty array when none 0.46s   
 ✓ create project returns 201 and stores 0.03s   
 ✓ show returns 200 or 404 0.02s   
 ✓ update project returns 200 and updates 0.02s   
 ✓ delete project returns 204 and removes 0.02s   
  
 Tests: 5 passed (16 assertions)  
 Duration: 0.73s  
  
Time: 00:00.580, Memory: 44.00 MB  
  
Proyecto Api (Tests\Feature\ProyectoApi)  
 ✔ List projects returns empty array when none  
 ✔ Create project returns 201 and stores  
 ✔ Show returns 200 or 404  
 ✔ Update project returns 200 and updates  
 ✔ Delete project returns 204 and removes  
  
OK (5 tests, 16 assertions))

|  |
| --- |
| Captura - pruebas Pest |

Captura - pruebas Pest

## 3) Fragmentos de código relevantes

* Método store en ProyectoApiController (validación y creación, asigna created\_by):

// ... (ver archivo app/Http/Controllers/Api/ProyectoApiController.php)

* Middleware utilizado en rutas:

Route::middleware([\App\Http\Middleware\JwtMiddleware::class])->group(function () {  
 // rutas proyectos  
});

## 4) Pruebas realizadas

Se ejecutaron las pruebas automáticas unitarias/feature con Pest (PHPUnit). Las pruebas cubren: - Listado vacío - Creación y comprobación en BD - Recuperación por id y 404 - Actualización y comprobación en BD - Eliminación y verificación

## 5) Conclusiones y próximos pasos

* Todos los requisitos funcionales solicitados están implementados y verificados con tests automatizados.
* Recomendado: generar colección Postman con variables de entorno en el entorno de pruebas para facilitar la reproducción (ya existe en postman/EVU1-api-collection.json).
* Si deseas, genero el PDF del informe y lo adjunto al repositorio; si la conversión automática falla en este entorno, dejaré el Markdown y las instrucciones para convertirlo localmente.

Anexo: evidencia detallada

### A. php artisan route:list --path=api output

(ROUTE\_LIST\_OUTPUT)

### B. Ejecución de pruebas (Pest)

(PEST\_OUTPUT)