MANUAL DE USUARIO

Laravel 12 (API REST) + Vue 3 (Frontend) + Axios + JWT

Este manual documenta **paso a paso** la creación de un proyecto con: - Backend independiente en **Laravel 12** - Frontend independiente en **Vue 3 (Webpack, sin Vite)** - Comunicación vía **Axios** - Autenticación mediante **JWT**

Está pensado como **guía reutilizable** para futuros proyectos.

---

1. REQUISITOS DEL SISTEMA

Backend

- PHP **8.2 o 8.3** (recomendado para Laravel 12)
- Composer
- XAMPP actualizado (Apache + MySQL + PHP 8.2+)

Frontend

- Node.js 18+
- NPM 9+

---

2. CREACIÓN DEL BACKEND (LARAVEL 12 API)

2.1 Crear proyecto Laravel

```
composer create-project laravel/laravel backend
```

Entrar al proyecto:

```
cd backend
```

2.2 Configurar .env

```
APP_NAME=Laravel
APP_ENV=local
APP_URL=http://127.0.0.1:8000

DB_CONNECTION=mysql
DB_HOST=127.0.0.1
DB_PORT=3306
DB_DATABASE=admin_ob
```

```
DB_USERNAME=root
DB_PASSWORD=
```

2.3 Ejecutar servidor

```
php artisan serve
```

Error común: - ❌ `vendor/autoload.php not found` - ✔️Solución: ejecutar `composer install`

---

3. ESTRUCTURA API EN LARAVEL 12

3.1 Archivo routes/api.php

En Laravel 12 **NO viene creado** automáticamente.

Crear manualmente:

```
routes/api.php
```

```php
<?php

use Illuminate\Support\Facades\Route;

Route::get('/ping', function () {
    return response()->json([
        'status' => 'ok',
        'message' => 'Laravel 12 API funcionando 🚀'
    ]);
});
```

Probar en navegador:

```
http://127.0.0.1:8000/api/ping
```

---

4. CORS (ERROR COMÚN)

En Laravel 12 **NO existe config/cors.php por defecto**.

Crear archivo:

```
config/cors.php
```

```php
<?php

return [
    'paths' => ['api/*'],
    'allowed_methods' => ['*'],
    'allowed_origins' => ['*'],
    'allowed_headers' => ['*'],
    'supports_credentials' => false,
];
```

Limpiar caché:

```
php artisan config:clear
```

Error común: - ❌Axios Network Error - ✔️Solución: configurar CORS

---

5. BASE DE DATOS, MIGRACIÓN Y SEEDER

5.1 Migración de usuarios

```
php artisan make:migration create_users_table
```

```
Schema::create('users', function (Blueprint $table) {
    $table->id();
    $table->string('name');
    $table->string('email')->unique();
    $table->string('password');
    $table->timestamps();
});
```

```
php artisan migrate
```

5.2 Seeder de usuario

```
php artisan make:seeder UserSeeder
```

```
User::create([
    'name' => 'Usuario Test',
```

```
    'email' => 'test@test.com',
    'password' => Hash::make('password'),
]);
```

Registrar en DatabaseSeeder:

```
$this->call(UserSeeder::class);
```

Ejecutar:

```
php artisan db:seed
```

---

6. JWT AUTHENTICATION

6.1 Instalar JWT

```
composer require tymon/jwt-auth
```

```
php artisan vendor:publish --
provider="Tymon\JWTAuth\Providers\LaravelServiceProvider"
php artisan jwt:secret
```

6.2 Configurar auth.php

```
'guards' => [
    'api' => [
        'driver' => 'jwt',
        'provider' => 'users',
    ],
],
```

---

7. MIDDLEWARE JWT (LARAVEL 12)

❌**No existe Kernel.php**

Todo se registra en:

```
bootstrap/app.php
```

```
->withMiddleware(function (Middleware $middleware) {
    $middleware->alias([
        'auth.jwt' => \Tymon\JWTAuth\Http\Middleware\Authenticate::class,
    ]);
})
```

---

8. CONTROLADOR AUTH

```
php artisan make:controller AuthController
```

```
public function login(Request $request)
{
    $credentials = $request->only('email', 'password');

    if (!$token = auth('api')->attempt($credentials)) {
        return response()->json(['error' => 'Unauthorized'], 401);
    }

    return response()->json([
        'access_token' => $token,
        'token_type' => 'bearer'
    ]);
}

public function logout()
{
    auth('api')->logout();
    return response()->json(['message' => 'Sesión cerrada']);
}
```

---

9. RUTAS API PROTEGIDAS

```
Route::post('/login', [AuthController::class, 'login']);

Route::middleware('auth.jwt')->group(function () {
    Route::get('/me', fn (Request $r) => $r->user());
    Route::post('/logout', [AuthController::class, 'logout']);
});
```

---

10. POSTMAN (PRUEBAS)

Login

- POST /api/login
- Body JSON

```
{
  "email": "test@test.com",
  "password": "password"
}
```

Rutas protegidas

Authorization → Bearer Token

---

11. FRONTEND (VUE 3 + WEBPACK)

```
vue create frontend
```

Seleccionar: - Vue 3 - Babel - Router - Linter - Webpack

Instalar dependencias:

```
npm install
npm run serve
```

---

12. AXIOS

```
npm install axios
```

```
import axios from 'axios';

axios.get('http://127.0.0.1:8000/api/ping')
  .then(res => console.log(res.data));
```

---

13. ERRORES QUE OCURRIERON (Y SOLUCIÓN)

Error	Causa	Solución
Network Error	CORS	Crear cors.php
404 API	No existe api.php	Crear archivo
Kernel.php no existe	Laravel 12	Usar bootstrap/app.php
vendor/autoload.php	Composer	composer install
Token en login	Concepto incorrecto	Token solo después del login

---

14. CONCLUSIÓN

Este flujo separa correctamente responsabilidades: - Laravel → API segura - Vue → UI - Axios → comunicación - JWT → autenticación

Este manual puede reutilizarse como plantilla base para proyectos futuros.

FIN 🚫