Creación de una API en PHP

TECNOLOGÍAS WEB II

Carlos Rojas Sánchez

Licenciatura en Informática

Universidad del Mar

Agenda

¿Qué es una API?

PHP y APIs

Framework recomendado

Instalación

Estructura del proyecto

Primer endpoint

Controladores

Buenas prácticas

Conclusiones

¿Qué es una API?

Concepto de API

- API: Interfaz de Programación de Aplicaciones.
- Permite que sistemas intercambien datos de forma estructurada.
- Generalmente se usa JSON como formato de intercambio.
- Las APIs REST son las más comunes hoy en día.

¿Qué es un Endpoint?

Un endpoint es una URL específica dentro de una API donde un cliente puede acceder a un recurso o ejecutar una acción.

- Representa un punto de comunicación entre el cliente y el servidor.
- Cada endpoint debe tener un propósito claro.
- Usualmente sigue el estilo REST.
- Ejemplo: `/api/v1/usuarios`

Tipos de Endpoints

- GET: Obtener información.
- POST: Crear recursos.
- PUT/PATCH: Actualizar recursos existentes.
- DELETE: Eliminar recursos.
- Importante: seguir buenas prácticas de REST.

Tipos de Endpoints

- GET: Obtener información.
- POST: Crear recursos.
- PUT/PATCH: Actualizar recursos existentes.
- DELETE: Eliminar recursos.
- Importante: seguir buenas prácticas de REST.

Ejemplos de Endpoints REST

```
GET /api/v1/usuarios
POST /api/v1/usuarios
GET /api/v1/usuarios/{id}
PUT /api/v1/usuarios/{id}
DELETE /api/v1/usuarios/{id}
```

Estos definen operaciones CRUD estándar.

PHP y APIs

¿PHP sirve para APIs?

- Sí, PHP puede crear APIs robustas.
- Aunque se puede hacer con PHP nativo, usar un framework es más eficiente.
- Frameworks recomendados:
  - Laravel
  - Slim Framework
  - Lumen (microframework)

Framework recomendado

¿Por qué Lumen?

- Micro-framework basado en Laravel.
- Muy rápido y óptimo para APIs.
- Soporta routing, middleware, validación, Eloquent ORM.
- Ideal si necesitas rendimiento y estructura limpia.

Instalación

Requisitos

- PHP >= 8.1
- Composer
- Servidor local: Apache, Nginx o PHP built-in server

Instalación de Lumen

```
composer create-project --prefer-dist laravel/lumen api-demo
```

- Esto crea el proyecto base.

Estructura del proyecto

Carpetas principales

- /app → Controladores y lógica
- /routes → Archivo web.php donde definimos rutas
- /bootstrap → Configuración de inicio
- /public → Carpeta pública del servidor

Primer endpoint

Crear una ruta simple

Archivo: `routes/web.php`

```
$router->get('/hola', function () {
    return response()->json(['mensaje' => 'Hola desde la API']);
});
```

Iniciar el servidor

```
php -S localhost:8000 -t public
```

Abrir en navegador:

`http://localhost:8000/hola`

Controladores

Crear un controlador

Archivo:
`app/Http/Controllers/UsuarioController.php`

```
<?php
namespace App\Http\Controllers;
use Laravel\Lumen\Routing\Controller as BaseController;

class UsuarioController extends BaseController {
    public function lista() {
        return [
            ['id' => 1, 'nombre' => 'Carlos'],
            ['id' => 2, 'nombre' => 'Ana']
        ];
    }
}
?>
```

Registrar ruta para el controlador

Archivo: `routes/web.php`

```
$router->get('/usuarios', 'UsuarioController@lista');
```

Buenas prácticas

Recomendaciones

- Usar controladores para mantener organizado el código.
- Implementar validación de datos.
- Usar Eloquent si hay base de datos.
- Implementar versionado de API (/api/v1/...)
- Usar middleware para autenticación.

Conclusiones

Conclusiones

- Lumen es ideal para crear APIs rápidas y estructuradas.
- PHP sigue siendo útil para backend moderno.
- Los frameworks simplifican y estandarizan el desarrollo.