Interacción Completa: PHP, HTML, JSON y JavaScript

🔍 Ejemplo Práctico: Sistema de Registro de Usuarios

1. Estructura de Archivos

```
proyecto/
├── index.html
├── registro.php
└── procesar.php
```

2. Archivo HTML (index.html)

```html
<!DOCTYPE html>
<html>
<head>
    <title>Registro de Usuario</title>
</head>
<body>
    <!-- Formulario de Registro -->
    <form id="registroForm">
        <input type="text" name="nombre" placeholder="Nombre" required>
        <input type="email" name="email" placeholder="Email" required>
        <input type="number" name="edad" placeholder="Edad" required>
        <button type="submit">Registrar</button>
    </form>

    <!-- Área para mostrar respuesta -->
    <div id="resultado"></div>
```

```
    <script>
    document.getElementById('registroForm').addEventListener('submit', function(e) {
        // Prevenir envío tradicional del formulario
        e.preventDefault();

        // Obtener datos del formulario
        const formData = new FormData(this);

        // Enviar datos con Fetch API
        fetch('procesar.php', {
            method: 'POST',
            body: formData
        })
        .then(response => response.json())
        .then(data => {
            // Mostrar respuesta del servidor
            const resultadoDiv = document.getElementById('resultado');

            if(data.success) {
                resultadoDiv.innerHTML = `
                    <h3>Registro Exitoso!</h3>
                    <p>Nombre: ${data.nombre}</p>
                    <p>Email: ${data.email}</p>
                    <p>Edad: ${data.edad}</p>
                `;
            } else {
                resultadoDiv.innerHTML = `<p>Error: ${data.mensaje}</p>`;
            }
        })
        .catch(error => {
            console.error('Error:', error);
        });
    });
    </script>
</body>
</html>
```

3. Archivo PHP de Procesamiento (procesar.php)

```php
<?php
// Configuración de cabeceras para JSON
header('Content-Type: application/json');

// Verificar método de solicitud
if ($_SERVER['REQUEST_METHOD'] === 'POST') {
    // Validaciones de datos
    $nombre = filter_input(INPUT_POST, 'nombre', FILTER_SANITIZE_STRING);
    $email = filter_input(INPUT_POST, 'email', FILTER_VALIDATE_EMAIL);
    $edad = filter_input(INPUT_POST, 'edad', FILTER_VALIDATE_INT);

    // Array de respuesta
    $respuesta = [
        'success' => false,
        'mensaje' => ''
    ];

    // Validaciones
    if (empty($nombre)) {
        $respuesta['mensaje'] = 'Nombre inválido';
    } elseif (!$email) {
        $respuesta['mensaje'] = 'Email inválido';
    } elseif (!$edad || $edad < 18) {
        $respuesta['mensaje'] = 'Edad inválida. Debe ser mayor de 18';
    } else {
        // Datos válidos
        $respuesta = [
            'success' => true,
            'nombre' => $nombre,
            'email' => $email,
            'edad' => $edad
        ];

        // Aquí podrías guardar en base de datos
```

```
        // Por ejemplo: guardarUsuario($nombre, $email, $edad);
    }

    // Convertir a JSON y enviar
    echo json_encode($respuesta);
    exit();
} else {
    // Método no permitido
    echo json_encode([
        'success' => false,
        'mensaje' => 'Método no permitido'
    ]);
    exit();
}
?>
```

🔑 Elementos Clave Explicados

HTML

- `<form id="registroForm">` : Formulario de captura de datos
- Atributos en inputs: Identifican datos para PHP `name`
- `addEventListener('submit')` : Intercepta envío del formulario
- `FormData` : Obtiene datos del formulario
- `fetch()` : Envía datos asíncronamente a PHP

PHP (en inglés)

- `$_SERVER['REQUEST_METHOD']` : Verifica método HTTP
- `filter_input()` : Valida y sanitiza entradas
- `header('Content-Type: application/json')` : Indica respuesta JSON

- `json_encode()` : Convierte array a JSON
- `exit()` : Termina ejecución del script

JavaScript

- `fetch()` : Envía solicitud asíncrona
- `.then()` : Maneja respuestas
- `response.json()` : Parsea respuesta JSON
- Manipulación del DOM para mostrar resultados

🚀 Flujo Completo

1. Usuario llena formulario
2. JavaScript captura datos
3. Datos enviados a PHP
4. PHP valida datos
5. PHP genera respuesta JSON
6. JavaScript recibe y muestra respuesta

💡 Buenas Prácticas

- Siempre validar datos
- Usar `filter_input()`
- Sanitizar entradas
- Manejar errores
- Usar JSON para comunicación

⚠️ Consideraciones de Seguridad

- Validar TODAS las entradas
- Usar HTTPS
- Implementar tokens CSRF
- Sanitizar datos antes de procesar
- Limitar tamaño de inputs

¡Domina la interacción PHP-HTML-JSON! 🎉