**TEMA 7: API con PHP.JSON**

**1. Introducción**

Una API es un conjunto de clases, algoritmos y funcionalidades que nos procesan principalmente la información a la base de datos. Controla y gestiona los permisos, accesos y niveles de seguridad.

Existen casos en los que el cliente que accede a nuestra información no es un ser humano, y por no tanto no necesita disponer una visualización de los mismos.

En casos que el cliente requiera un acceso directo a la información se puede establecer una conexión directa con la base de datos, pero esto tiene dos inconvenientes:

* Es muy dependiente del tipo de base de datos utilizado
* Es algo muy peligroso a nivel de seguridad.

**2. API de datos**

Una API sirve de aplicación intermedia que accede a los datos de una base de datos y devuelve la información. Esto soluciona los problemas anteriores:

* La API utiliza un estándar de comunicación como el HTTP, por lo que la forma de obtener los datos de cara al cliente se estandariza.
* La API se encarga de la seguridad.

**3. Construcción de un API sencillo**

Para construir una API será necesario acceder a las tablas de una base de datos, y establecer unos métodos que digan cómo gestionar la información de las mismas.

**3.1 MVC**

En primer lugar se crea una estructura siguiendo el patrón MVC, por lo que se utilizarán:

* Una clase db que realizará la conexión contra la base de datos, y
* Una clase hija que herede de ella y gestione las transacciones de información.

**3.2. CRUD**

En la clase hija desarrollaremos el CRUD, que tendrá los siguientes tipos de funciones:

* Lectura
* Escritura
* Actualización
* Borrado

Uno de los problemas más relevantes dentro de una API es aplicar unos determinados niveles de seguridad.

**3.3. Request METHOD**

El uso de distintos métodos para hacer una request por HTTP gana sentido cuando el usuario es un dispositivo. De esta forma, cada método establecerá el tipo de acción a realizar con nuestra base de datos.

* GET: Para realizar SELECT.
* POST: Para realizar INSERT.
* PUT: Para realizar UPDATE.
* DELETE: Para realizar DELETE.

La superglobal $\_SERVER permite conocer qué método se ha utilizado por parte del cliente.

**3.4. JSON**

JSON (JavaScript Object Notation) es un formato de texto ligero para el intercambio de datos. Pese a ser un suncobjunto de la notación literal de objetos de JavaScript, se ha convertido en un estándar para la comunicación de información y se utiliza especialmente en APIs.

Hoy en día se considera un formato de lenguaje independiente utilizado como alternativa a XML u otros lenguajes de marcas, ya que su nivel de burocracia es mínima, al consistir básicamente en una estructura jerárquica de datos.

En PHP es sencillo codificar y decodificar JSON a partir de arrays asociativos.

**3.5. Estructura final**

Las funciones de la API son las siguientes:

* Recibe las peticiones del cliente.
* Distingue el método solicitado.
* Recoge los datos a través de $\_REQUEST
* Realiza la llamada al método solicitado.
* Devuelve la información en formato JSON mediante un echo del json\_encode.