API

1. ¿Qué es una API?

- API: Interfaz de Programación de Aplicaciones.
  Una API (Application Programming Interface) es un conjunto de definiciones y protocolos que permiten la comunicación entre diferentes sistemas de software. Las APIs permiten que una aplicación solicite datos o servicios de otra aplicación, facilitando la integración y la interoperabilidad entre diferentes sistemas.
- Permite la comunicación entre diferentes sistemas de software.

API REST

Podríamos definir API REST como un servicio que nos provee de las funciones que necesitamos para poder obtener información de un cliente externo, como, por ejemplo, una base de datos alojada en cualquier parte del mundo desde dentro de nuestra propia aplicación.

Vamos a pensar en Instagram, una aplicación con millones de usuarios. Es inviable tener la información de cada usuario dentro de la aplicación ¿verdad? Pues para solventar el problema usan servicios API REST. Lo primero que hacemos al entrar en la aplicación es un login, esto sería el primero de los servicios, pues nosotros le mandamos al servidor el usuario y contraseña y este nos devolvería la información que debemos mostrar en la aplicación.

Disponemos de cuatro tipos distintos de peticiones como norma general.

- **Get**: Son las peticiones más sencillas, solo nos devuelven información. Si necesitamos pasarle un parámetro a la petición será a través de la URL. Es decir, si por ejemplo tenemos que hacer una petición que depende de una id (ej. la identificación del usuario) la URL se formaría así *https://ejemplo.com/informacion/1*, siendo 1 el parámetro que le pasamos. El problema de esto es que es poco seguro para pasar información delicada.
  SELECT.

- **Post**: Similar a Get pero los parámetros no se pasan por **url**, por lo que es más seguro para mandar información.
  Se usa para enviar datos al servidor y **crear un nuevo recurso**. Este método puede modificar el estado del servidor.
  INSERT.

- **Put**: Se usa para actualizar un recurso existente en el servidor. Si el recurso no existe, puede crearlo.
  UPDATE/INSERT.

- **Delete**: Sería el último de los cuatro que nos permitiría borrar los registros de la base de datos.
  DELETE.

La información suele venir en dos formatos distintos, XML o JSON. Para no meternos mucho en el tema, solo hablaremos del JSON que es el formato más habitual y con el que nosotros vamos a trabajar.

Json es un formato de texto simple, es el acrónimo de *JavaScript Object Notation*. Se trata de uno de los estándares para el traspaso de información entre plataformas, tiene una forma muy legible que nos permite entender su contenido sin problema. Un ejemplo sencillo sería este:

```
{
  "deportistas": {
    " deportista": [
     {
       "id": "1",
       "nombre": "Rafael",
```

```
    "apellido": "Nadal",
    "foto": "https://jsonformatter.org/img/rafa-nadal.jpg"
   },
   {
    "id": "2",
    "nombre": "Carolina",
    "apellido": "Marín",
    "foto": "https://jsonformatter.org/img/carolina-marin.jpg"
   },
   {
    "id": "3",
    "nombre": " Aitana",
    "apellido": " Bonmatí",
    "foto": "https://jsonformatter.org/img/aitana-bonmatí.jpg"
   }
  ]
 }
}
```


Todo formato Json empieza y termina con llaves y tiene una clave-valor. La clave *deportistas* contiene a su vez una lista de *deportista* (fijaros que en vez de llaves tiene corchetes), que este almacena id, nombre, apellido y foto. Así podemos pasarnos gran cantidad de información de una plataforma a otra con unos estándares que nos ayudan a simplificar el proceso.

Si aun así se os complica la lectura de estos ficheros al principio, podemos hacer uso de multitud de webs que simplifican la forma de verlo, como, por ejemplo JsonEditOnline.

Conversión entre diferentes formatos de documentos.

- **Conversión de XML a JSON**:
  - **Ejemplo**: `xml`

```
<persona>
      <nombre>Juan</nombre>
      <edad>30</edad>
      <ciudad>Madrid</ciudad>
</persona>
```

  - se convierte a: `json`

```
{
      "persona":
```

```
	{
		"nombre": "Juan",
		"edad": 30,
		"ciudad": "Madrid"
	}
}
```