Docente: Germán Alberto Angarita Henao

API Y API REST

¿Qué es una API?

Una **API** (*Application Programming Interface* o Interfaz de Programación de Aplicaciones) es un conjunto de reglas y definiciones que permiten que dos sistemas o aplicaciones se comuniquen entre sí. Funciona como un intermediario que facilita el intercambio de datos y funcionalidades sin que los sistemas necesiten conocer los detalles internos del otro.

Ejemplo:
Un sistema de pago como PayPal ofrece una API para que otros sitios web puedan integrar pagos sin necesidad de gestionar toda la infraestructura de transacciones.

¿Qué es una API REST?

Una **API REST** (*Representational State Transfer o Transferencia de estado representacional*) es un tipo de API que sigue los principios de **REST**, un estilo de arquitectura basado en el protocolo HTTP. Estas APIs permiten la comunicación entre clientes y servidores de una manera eficiente y escalable.

Características principales de una API REST:

1. **Cliente-Servidor**: El cliente (ejemplo: una aplicación web) y el servidor (ejemplo: una base de datos) están separados.
2. **Sin estado**: Cada solicitud enviada al servidor debe contener toda la información necesaria para procesarla, sin depender de solicitudes anteriores.
3. **Uso de métodos HTTP**: Se utilizan los métodos estándar de HTTP para operar con los recursos:
    - GET → Obtener datos
    - POST → Crear un nuevo recurso
    - PUT → Actualizar un recurso existente
    - DELETE → Eliminar un recurso
4. **Representación de los recursos**: Los recursos se identifican mediante **URLs**(Uniform Resource Locator) y los datos pueden ser representados en formatos como JSON o XML.
5. **Uso de códigos de estado HTTP**: Indican el resultado de una solicitud (200 OK, 404 Not Found, 500 Internal Server Error, etc.).

Contacto: gangaritah@gmail.com

Docente: Germán Alberto Angarita Henao

Ejemplos de Uso de APIs REST

Las APIs REST se utilizan en múltiples ámbitos del desarrollo de software para permitir la comunicación entre sistemas. Aquí te dejo algunos ejemplos comunes:

1. Autenticación de Usuarios

Muchas aplicaciones utilizan APIs REST para autenticación y autorización de usuarios.

2. Consumo de APIs de Terceros (Ejemplo: API de Clima)

Las aplicaciones pueden consumir APIs REST públicas para obtener información en tiempo real.

3. Integración con Pagos (Ejemplo: API de PayPal o Stripe)

Las APIs REST permiten procesar pagos en tiendas en línea.

4. Consumo de APIs de Redes Sociales (Ejemplo: Twitter API)

Directorios de APIS
Existen directorios de APIS que nos facilitan el trabajo para encontrar las que tienen las funcionalidades que requerimos para nuestros proyectos. Un sitio muy famoso es RapidApi:
https://rapidapi.com/categories

JSON

JSON (JavaScript Object Notation) es un formato ligero para el intercambio de datos, fácil de leer y escribir para los humanos, y sencillo de procesar para las máquinas. Se basa en pares clave-valor y estructuras de datos como objetos y listas.

Estructura básica

Objetos

Un objeto en JSON se representa con llaves {} y contiene pares **clave-valor**, donde la clave siempre es un **string** y el valor puede ser de varios tipos.

Contacto: gangaritah@gmail.com


```
                    {
                      "nombre": "Juan",
  "edad": 30,
  "esEstudiante": false
}
```

Listas (Arrays)

Las listas se representan con corchetes [] y pueden contener valores de cualquier tipo, incluso otros objetos o listas.

```
{
  "nombres": ["Ana", "Luis", "María"]
}
```

También se pueden combinar objetos dentro de listas:

```
{
  "personas": [
    { "nombre": "Ana", "edad": 25 },
    { "nombre": "Luis", "edad": 30 }
  ]
}
```

Tipos de datos en JSON

1. **Strings (cadenas de texto)**: "Hola mundo"
2. **Números**: 10, 3.14
3. **Booleanos**: true, false
4. **Nulos**: null
5. **Objetos**: { "clave": "valor" }
6. **Listas**: [1, 2, 3]

Docente: Germán Alberto Angarita Henao

Estructura de una petición HTTP

Una petición HTTP tiene tres secciones principales:

1. **Línea de petición**
2. **Cabeceras (Headers)**
3. **Cuerpo (Body) [Opcional]**

1. Línea de petición

Es la primera línea de la petición y define tres elementos clave:

- **Método HTTP**: Indica la acción que se desea realizar. Ejemplos: GET, POST, PUT, DELETE.
- **URI (Uniform Resource Identifier)**: Es el recurso al que se está accediendo (ruta).
- **Versión del protocolo HTTP**: Especifica la versión del protocolo utilizada (HTTP/1.1, HTTP/2).

Ejemplo de línea de petición:

GET /productos HTTP/1.1

Esto significa que se está solicitando la lista de productos mediante GET en la versión HTTP/1.1.

2. Cabeceras (Headers)

Las cabeceras proporcionan información adicional sobre la petición, como el formato de los datos, la autenticación o el agente de usuario.

Ejemplo de cabeceras comunes:

Host: www.ejemplo.com
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64)
Content-Type: application/json
Authorization: Bearer token_aqui

Contacto: gangaritah@gmail.com

Algunas cabeceras importantes:

- **Host**: Indica el dominio del servidor.
- **User-Agent**: Identifica el cliente que realiza la petición.
- **Content-Type**: Especifica el tipo de contenido en el cuerpo de la petición.
- **Authorization**: Usado para enviar credenciales o tokens de acceso.

3. Cuerpo (Body) [Opcional]

El cuerpo de la petición se usa en métodos como POST y PUT para enviar datos al servidor.

Ejemplo de un cuerpo en JSON:

```
{
  "nombre": "Laptop",
  "precio": 1200,
  "stock": 5
}
```

Esto se enviaría en una petición POST o PUT para agregar o actualizar un producto.

Ejemplo completo de una petición HTTP

POST /productos HTTP/1.1
Host: www.ejemplo.com
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64)
Content-Type: application/json
Authorization: Bearer token_aqui
Content-Length: 57

{
  "nombre": "Laptop",
  "precio": 1200,
  "stock": 5
}