**2020. “Año de Laura Méndez de Cuenca; emblema de la mujer Mexiquense”**

UNIVERSIDAD TECNOLÓGICA DEL VALLE DE TOLUCA

Dirección de Carrera de Tecnologías de la Información y Comunicación

Técnico Superior Universitario en Tecnologías de la Información Área

Desarrollo de Software Multiplataforma

**Nombre del Trabajo**

“Tareas”

**Nombre dela Materia**

Aplicaciones Web Orientadas A Servicios

**Nombre del Profesor**

Fernando Bonifacio Morales

**Nombre del Estudiante**

Dulce María Sevilla Ortiz

**Grupo**

TI DSM – 42

**Cuatrimestre**

4º

**Lugar**

Santa María Atarasquillo, Lerma, México.

**SECRETARÍA DE EDUCACIÓN**

SUBSECRETARÍA DE EDUCACIÓN MEDIA SUPERIOR Y SUPERIOR

DIRECCIÓN GENERAL DE EDUCACIÓN SUPERIOR

Carretera del Departamento del D.F. km 7.5, Santa María A tarasquillo, C.P. 52044, Lerma, Estado de México.

Tels.: (728) 285 95 52, 285 99 69, 282 22 47. utvtol.edu.mx



Contenido

[Significado De Los Errores HTTP 4](#_Toc51796853)

[Api Aplicaciones Web 9](#_Toc51796854)

[Que es el formato Json y xml. 13](#_Toc51796855)

[JSON 13](#_Toc51796856)

[XML 16](#_Toc51796857)

[Investigar que es un endpoint en un api. 2](#_Toc51796858)

[Infografía Arquitectura MCV. 7](#_Toc51796859)

# Significado De Los Errores HTTP

**Error 500:** **Internal Server Error (Error de servidor interno)**

Este Indica un error interno, lo que se puede traducir que la página web tiene algún error en el código, por lo que el servidor no puede generar el código HTML para devolver al usuario.

El mensaje error se puede ver de varias maneras porque cada sitio web tiene permiso para personalizar el mensaje.

**Formas comunes de ver el error HTTP 500:**

* 500 Internal Server Error.
* HTTP 500 - Internal Server Error.
* Temporary Error (500).
* HTTP 500 Internal Error.
* 500 Error.
* HTTP Error 500.

Este tipo de error HTTP puede darse por diversas causas a nivel interno y por lo tanto puede resolverse de distintas maneras algunas de las soluciones son:

* **No tener los permisos necesarios:** Al no tener los permisos para acceder a una carpeta o archivo el servidor restringe el acceso.
* **Tiempo de espera de PHP:** Si en la conexión con algún tipo de recurso sobre pasa el tiempo de espera dará como consecuencia un error 500. Por lo que una solución sería ejecutar tiempos de espera correctos.

Este error sucede debido a algún fallo en la programación, o se ha llevado a cabo algún cambio en la plantilla o tema del sitio web.

**Error 502: Bad Gateway (Puerta de enlace incorrecta)**

Indica un error de comunicación entre dos servidores. Sucede cuando el cliente se conecta a un servidor que actúa como un proxy o un portal que necesita acceder a un servidor ascendente, el cual proporciona un servicio adicional al servidor inicial. En estos casos, usualmente el otro servidor se localiza en un nivel superior en la jerarquía de servidores.

En otras palabras se produce cuando no hay una respuesta válida por parte del servidor ascendente. No quiere decir necesariamente que este servidor esté caído, sino que los dos servidores no coinciden en el protocolo a usarse para intercambiar información. Sucede usualmente cuando uno de las máquinas no se encuentra correctamente configurada.

**Principales causas del Error 502:**

* Servidor caído o sobrecargado.
* Error en la comunicación entre los dos servidores no se han puesto de acuerdo en el protocolo que se debe usar para el intercambio de información.
* Uno de los servidores no está correctamente configurado o programado.
* Error navegador.
* CDN (en el caso que utilices una en tu página web).

**Principales soluciones ante el Error 502:**

* Ponerte en contacto con tu proveedor de hosting.
* Refrescar el navegador web pulsando CTRL+F5.
* Eliminar la cache y las cookies del navegador.
* Reiniciar o reinstalar tu navegador.
* Probar con otro navegador y desde otro dispositivo.

**Error 402: Payment Required (Problema de autorización)**

Es un código de respuesta de estado de error de cliente no estándar que está reservado para uso futuro.

A veces, este código indica que la solicitud no se puede procesar hasta que el cliente realice un pago. Originalmente fue creado para habilitar el efectivo digital o los sistemas de (micro) pago e indicaría que el contenido solicitado no está disponible hasta que el cliente realice un pago.

**Error 403: Forbidden (Acceso denegado o prohibido)**

Es un código de estado HTTP, el cual significa que no tiene permiso para acceder al archivo/carpeta al que querías llegar, de modo que se te ha denegado la entrada.

Este es una respuesta de error por parte del cliente, debido al navegador o del dispositivo que se están utilizando para tratar de acceder a la página o por el contrario son códigos por parte del servidor.

Una posible razón que explica este tipo de error es que el archivo específico que ha solicitado el cliente no tiene permisos para ser visto desde la web. Todas estas medidas de seguridad son tomadas por el administrador web para proteger archivos que tienen información clasificada.

**Posibles causas:**

* Permisos de archivos o carpetas incorrectos.
* Ajustes incorrectos en el archivo .htaccess.
* Un problema con el navegador web.
* Contenido restringido.

**Error 404: Not Found (No encontrada)**

Este código indica que el servidor no ha encontrado nada en la ubicación especificada por el cliente, ya sea porque la URL no se ha ingresado correctamente o la estructura de URLS permanentes (también conocidas como permalinks) ha sido modificada de tal manera que todos los enlaces han sido movidos a diferentes ubicaciones.

Se puede presentar cuando un sitio ha sido trasladado recientemente a otro servido y el DNS apunta a la ubicación antigua.

En ocasiones este error se puede evitar, pues recomendable optimizar esta página en tu sitio web y activar la re-dirección automática en caso hayas modificado los enlaces permanentes del sitio.

**Causas por las cuales sucede este error:**

* El enlace de la página web es incorrecto.
* La página o el artículo al que intentamos acceder han sido eliminados.
* El producto de la tienda online ha sido borrado o ya no se encuentra disponible.
* El buscador integrado de la web no encuentra referencias a los términos de búsqueda.

**Error 301:** **Moved Permanently (Movido Permanentemente)**

Estos códigos indican que se está haciendo una redirección de una página a otra o esta redirección puede ser permanente (código 301) o temporal (código 302).

Se introduce el código 301 estamos ante una redirección de tipo permanente, que se utiliza cuando la URL de una página se ha modificado. Asimismo, esta fórmula también puede resultar útil cuando creamos una nueva página (por ejemplo, un post) con contenido actualizado de otra ya existente. Cuando aplicamos una redirección 301 a una URL que ya está bien posicionada en los buscadores de internet, la próxima vez que el robot de Google visite esa página, entenderá que la URL ya está obsoleta y la sustituirá por la nueva URL a la que apunta la redirección.

**Causas acerca de este error:**

* Cuando debemos hacer un cambio de dominio.
* Cuando hemos de realizar un cambio en las URL para quesean semánticas (o amigables).
* Aprovechar el posicionamiento de un post antiguo ya obsoleto para crear un artículo nuevo con un contenido de mayor calidad y actualizado.

**Error 410:**

Se trata de un código de estado HTTP conocido como 410 Gone o Ya no existe/Ya no disponible. Cuando una URL está marcada con esta cabecera se manda un mensaje a los buscadores donde se les informa que ya no existe y no va a volver a existir.

El error 410 también indica que el servidor web no tiene una dirección de reenvío para la URL, así que no puede brindar un re direccionamiento al nuevo servidor web. Si el servidor web no sabe, o no tiene forma de saber, si la condición es permanente o no, debería usar en su lugar el código de estado 404 - Not found.

**Posibles causas:**

* Un virus pudo afectar al navegador de Internet.
* El sistema operativo presenta algún daño como consecuencia de un reciente cambio de software.
* Un programa no autorizado contiene elementos infectados que afectan al navegador.

# Api Aplicaciones Web

****

**Concepto:**

**Api (Interfaz de programación de aplicaciones):** es un conjunto de definiciones y protocolos que se utiliza para desarrollar e integrar el software de las aplicaciones.

**Otorgan:**

* Proporcionan oportunidades de innovación.
* Ideal al diseñar herramientas.
* Productos nuevos o gestionar actuales.
* Flexibilidad.
* Simplifican el diseño.
* Administración.
* Uso de aplicaciones.

Se consideran como contratos, con documentación que representa un acuerdo entre las partes, si una de las partes envía una solicitud remota con cierta estructura en particular, esa misma estructura determinará cómo responderá el software de la otra parte.

****Las API son un medio simplificado para conectar su propia infraestructura a través del desarrollo de aplicaciones nativas de la nube, pero también le permiten compartir sus datos con clientes y otros usuarios externos.

**Para que sirve una Api:**

Permiten que sus productos y servicios se comuniquen con otros, sin necesidad de saber cómo están implementados. Esto simplifica el desarrollo de las aplicaciones , permite ahorrar tiempo y dinero.

Son útiles para cuando lo único que se quiere es utilizar las funciones de determinado servicio ya sea para ofrecer ventajas a sus usuarios o atraer a los usuarios de ese servicio a que utilicen tu aplicación.

Las API le pueden permitir a los servicios o redes sociales limitar ciertas funciones dentro de las aplicaciones para que terceros puedan utilizar.

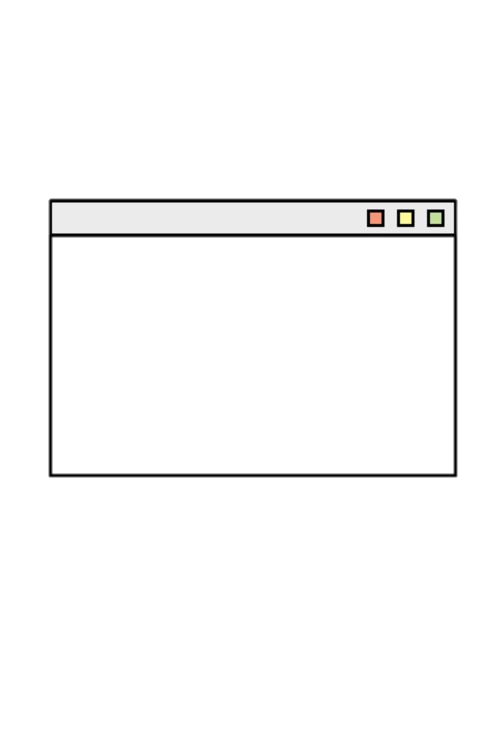
Una Api puede servir también para comunicarse con una base de datos sistema operativo o protocolo de comunicaciones.

**Ejemplos más conocidos de algunas plataformas online:**

**Paginas Online**

WordPress

Google Maps



**Redes sociales**

Facebook

Twitter

****

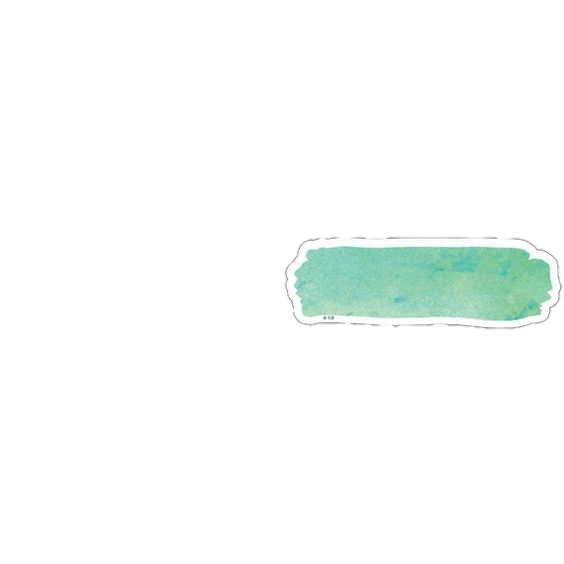
**Beneficios de API:**

Las APIS consisten con códigos predefinidos en otro software, estos ahorran trabajo y son mucho más rentables, ya que no es necesario inventar lo que ya está inventado. Además, son códigos probados que serán funcionales y seguros.

**Para generar ingresos a partir del desarrollo de las API, las empresas utilizan los siguientes métodos**

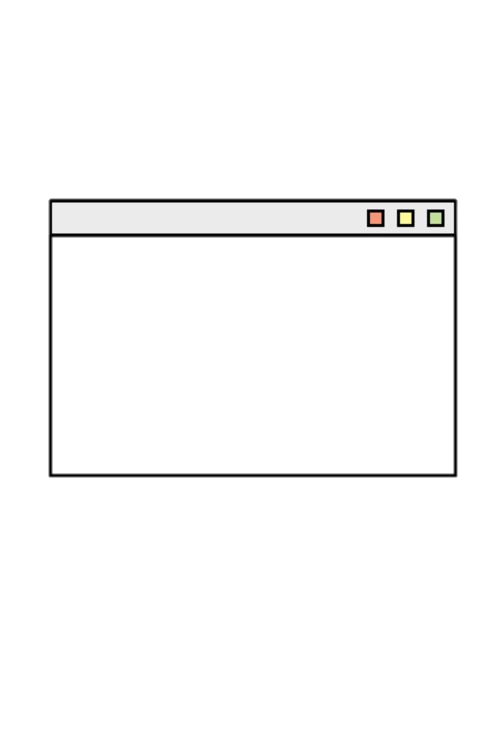
* **Pago por uso:** Es cuando el usuario paga una licencia por uso, que sería como un alquiler.
* **Productos API:** Es cuando el usuario compra la micro-aplicación.

**Ventajas y desventajas**



**Ventajas:**

* **Automatización.**
* **Aplicación.**
* **Nuevos datos disponibles.**
* **Eficiencia.**

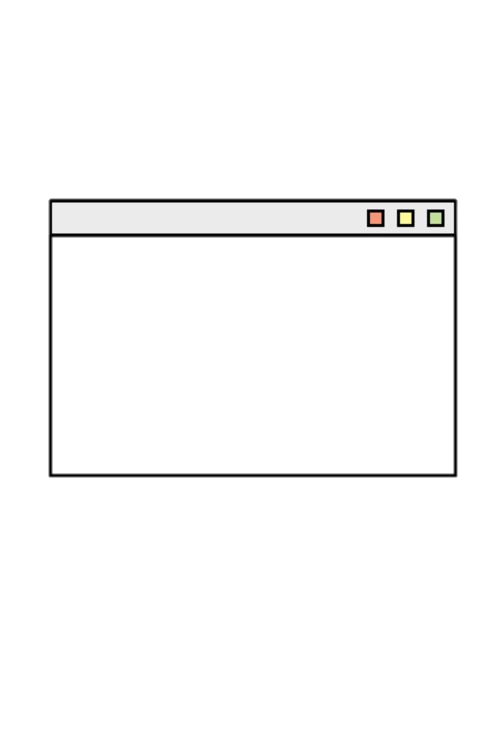


**Desventajas:**

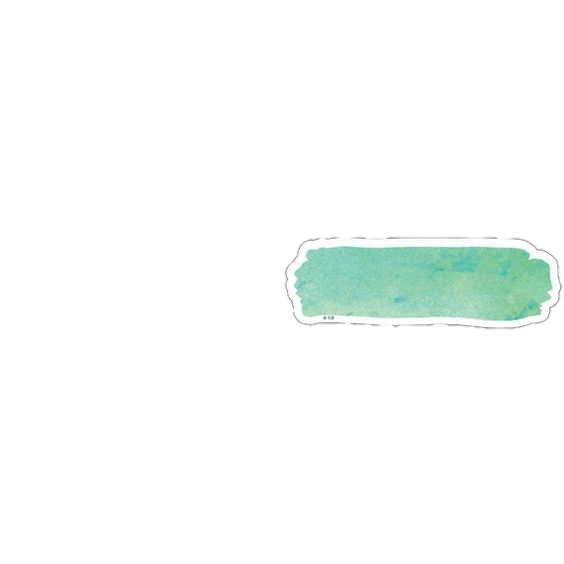
* Cambios
* Control de disponibilidad.
* Conocimiento.

**Características**

* Útil
* Fácil de aprender
* Estable frente a mejoras.



* Segura.
* Ofrece buena documentación.



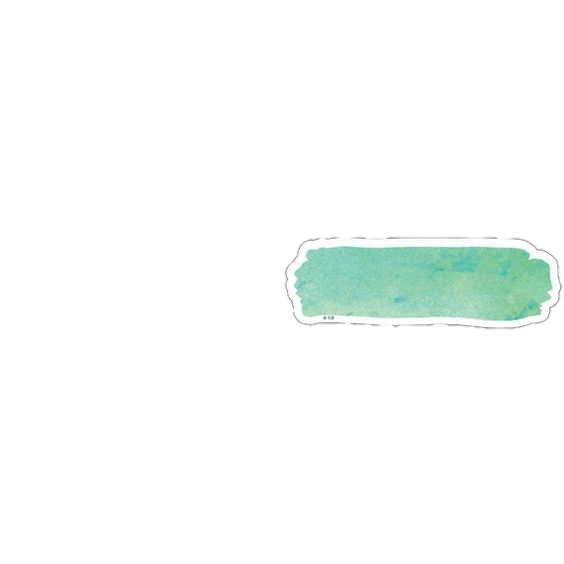


**Aplicaciones web**

Reciben este nombre porque se ejecutan en internet, es decir que los datos o los archivos en los que se trabajan son procesados y almacenados dentro de la web. Estas aplicaciones, por lo general, no necesitan ser instaladas en el computador.

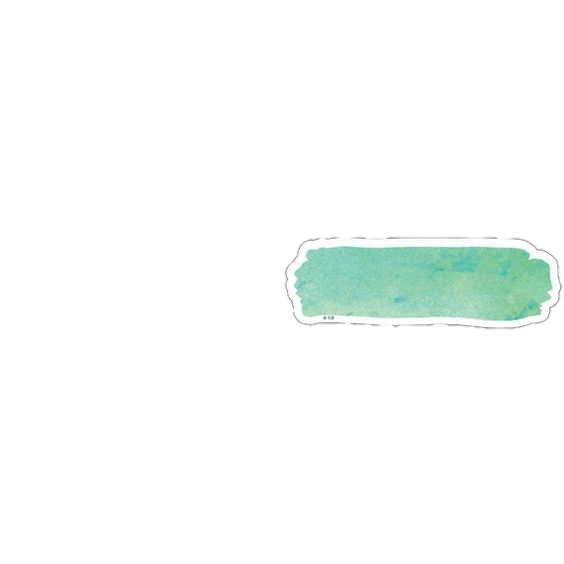
****El concepto está relacionado con el almacenamiento en la nube. Pues toda la información se guarda de forma permanente en grandes servidores de internet y nos envían a nuestros dispositivos o equipos los datos que requerimos en ese momento, quedando una copia temporal dentro de nuestro equipo.

**Características**



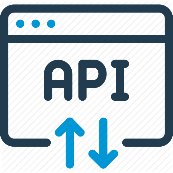
**Compatibilidad multiplataforma**

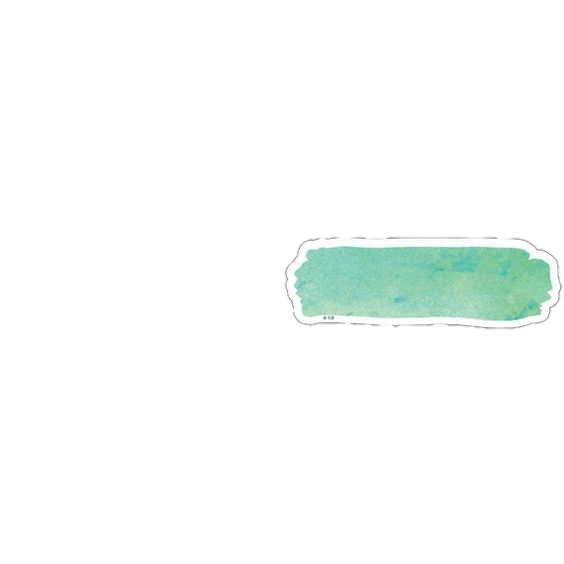
Una misma versión de la aplicación puede correr sin problemas en múltiples plataformas como Windows, Linux, Mac, etc.



**Actualización**

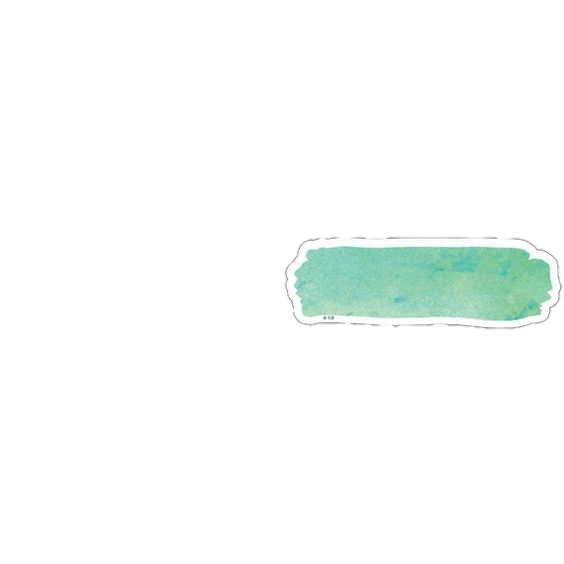
Siempre se mantienen actualizadas y no requieren que el usuario deba descargar actualizaciones y realizar tareas de instalación.





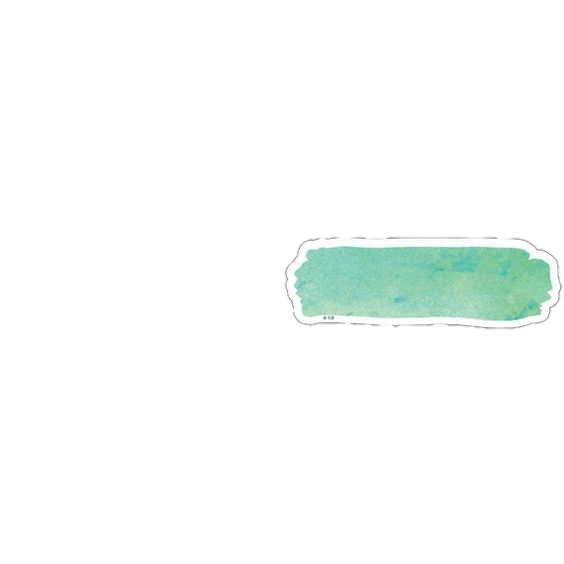
**Acceso inmediato:**

No necesitan ser descargadas, instaladas y configuradas. Además pueden ser accedidas desde cualquier computadora conectada a la red desde donde se accede a la aplicación.



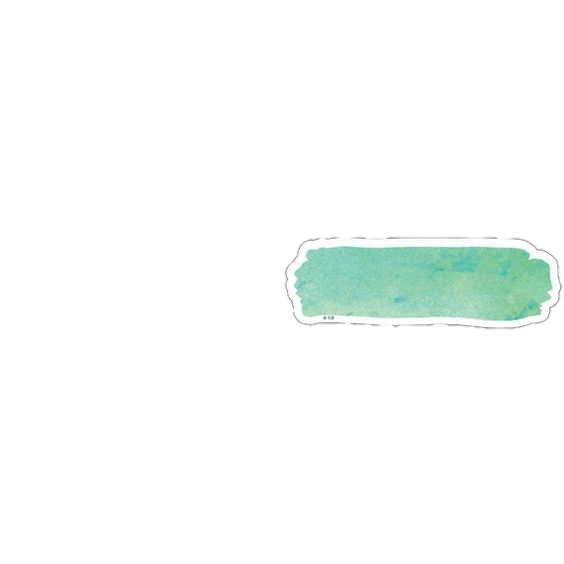
**Menos requerimientos de hardware**

Consumen muy poco espacio en disco y también es mínimo el consumo de memoria RAM en comparación con los programas instalados localmente. Tampoco es necesario disponer de computadoras con poderosos procesadores ya que la mayor parte del trabajo se realiza en el servidor en donde reside la aplicación.



**Menos Bugs (errores)**

Son menos propensas a crear problemas técnicos debido a problemas de software y conflictos de hardware. Los bugs (errores) pueden ser corregidos tan pronto como son descubiertos beneficiando inmediatamente a todos los usuarios de la aplicación.



**Seguridad en los datos**

Los datos se alojan en servidores con sistemas de almacenamiento altamente fiables y se ven libres de problemas que comúnmente sufren los ordenadores de usuarios comunes como virus y/o fallas de disco duro.

# Que es el formato Json y xml.

****

JSON **(JavaScript Object Notation - Notación de Objetos de JavaScript).**

JSON es un formato basado en texto estándar para representar datos estructurados en la sintaxis de objetos de JavaScript. Es comúnmente utilizado para transmitir datos en aplicaciones web (por ejemplo: enviar algunos datos desde el servidor al cliente, así estos datos pueden ser mostrados en páginas web).

****Los JSON son cadenas - útiles cuando se quiere transmitir datos a través de una red. Debe ser convertido a un objeto nativo de JavaScript cuando se requiera acceder a sus datos.

**JSON está constituido por dos estructuras:**

**Una colección de pares de nombre/valor:**

Es conocido como un objeto, registro, estructura, diccionario, tabla hash, lista de claves o un arreglo asociativo.

**Una lista ordenada de valores:**

Esto se implementa como arreglos, vectores, listas o secuencias.

****Una de las características de JSON, al ser un formato que es independiente de cualquier lenguaje de programación, es que los servicios que comparten información por este método no necesitan hablar el mismo idioma, es decir, el emisor puede ser Java y el receptor Python, pues cada uno tiene su propia librería para codificar y decodificar cadenas en este formato.

**Para que sirve:**

Formato de intercambio de datos ideal para usar con API REST o AJAX. A menudo se usa en lugar de XML, debido a su estructura ligera y compacta.

Se usa en muchos sistemas que requieren mostrar o enviar información para ser interpretada por otros sistemas.

**Características de JSON:**

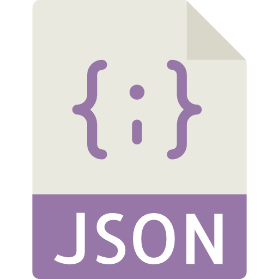
* Es solo un formato de datos.
* Requiere usar comillas dobles para las cadenas y los nombres de propiedades. Las comillas simples no son válidas.
* Una coma o dos puntos mal ubicados pueden producir que un archivo JSON no funcione.
* Puede tomar la forma de cualquier tipo de datos que sea válido para ser incluido en un JSON, no solo arreglos u objetos.
* ****A diferencia del código JavaScript, en el que las propiedades del objeto pueden no estar entre comillas, en JSON solo las cadenas entre comillas pueden ser utilizadas como propiedades.

**Ventajas:**

* Es auto descriptivo y fácil de entender.
* Su sencillez le ha permitido posicionarse como alternativa a XML.
* Es más rápido en cualquier navegador.
* Es más fácil de leer que XML.
* Es más ligero (bytes) en las transmisiones.
* Velocidad de procesamiento alta.
* ****Puede ser entendido de forma nativa por los analizadores de JavaScript.

**Desventajas:**

* Algunos desarrolladores encuentran su escueta notación algo confusa.
* No cuenta con una característica que posee XML.
* No soporta grandes cargas, solo datos comunes.
* ****Para la seguridad requiere de mecanismos externos como expresiones regulares.

**Sintaxis**

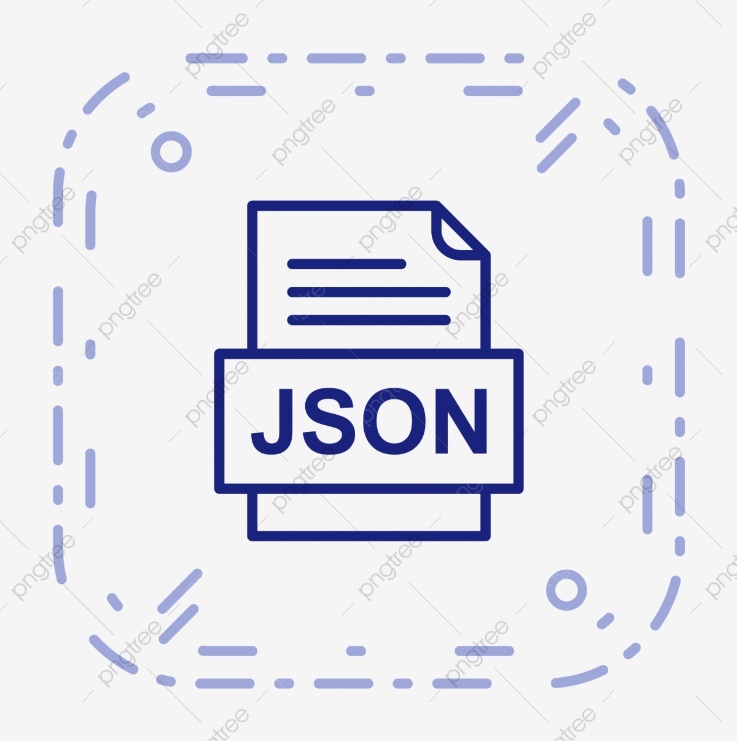
En JSON existen dos tipos de elementos matrices (arrays): Las matrices son listas de valores separados por comas. Las matrices se escriben entre corchetes [ ]

[1, "pepe", 3.14, "Pepito Conejo"]

Objetos (objects). Los objetos son listas de parejas nombre / valor. El nombre y el valor están separados por dos puntos: y las parejas están separadas por comas. Los objetos se escriben entre llaves { } y los nombres de las parejas se escriben siempre entre comillas dobles.

{"nombre": "Pepito Conejo", "edad": 25, "carnet de conducir": true}

Los espacios en blanco y los saltos de línea no son significativos, es decir, puede haber cualquier número de espacios en blanco o saltos de línea separando cualquier elemento o símbolo del documento.

 [

{

"nombre": "Pepito Conejo",

"edad": 25,

"carnet de conducir": true

},

{

"nombre": "Ana Barberá",

"edad": 90,

"carnet de conducir": false

}

****]

**Reglas de sintaxis JSON**

**JSON valores pueden ser:**

* Número (entero o de coma flotante)
* Cadena (entre comillas dobles)
* valor lógico (verdadero o falso)
* Array (entre corchetes)
* Objeto (apoyos)nulo
* Los datos de pares de nombre / valor
* Datos separados por comas
* Guarde los apoyos de objetos
* Guarde los soportes de matriz

****XML **(Extensible Markup Language - Lenguaje de marcado extensible).**

El lenguaje de marcado es un conjunto de códigos que se pueden aplicar en el análisis de datos o la lectura de textos creados por computadoras o personas.

****Proporciona una plataforma para definir elementos para crear un formato y generar un lenguaje personalizado.

**Un archivo XML se divide en dos partes prolog y body:**

* **Prolog:**

Consiste en metadatos administrativos, como declaración XML, instrucción de procesamiento opcional de tipo de documento y comentarios.

* **Body:**

****Se compone de dos partes: estructural y de contenido (presente en los textos simples).

**Para que sirve**

Se centra en la simplicidad, la generalidad y la facilidad de uso por lo tanto, se utiliza para varios servicios web.

Proporciona la estructura lógica del documento, mientras que la estructura del HTML está predefinida, utilizando tags heads e body.

XML simplifica el intercambio de datos y su sintaxis es de gran importancia.

Tanto los sistemas informáticos como las bases de datos contienen información en formatos incompatibles.

Los datos XML se almacenan en formato de texto simple, lo que nos posibilita una forma independiente de almacenar datos. Esto facilita mucho la creación de datos que pueden ser compartidos por diferentes aplicaciones.

**Características**

* Separa datos de HTML.
* Simplifica el intercambio de datos.
* Simplifica el cambio de plataforma.
* Aumenta la disponibilidad de datos.
* Se puede utilizar para crear nuevos idiomas de Internet.

**Idiomas creados por XML**

RSS (utilizado en noticias).

RDF y OWL.

SMIL.

XHTML.

WSDL.

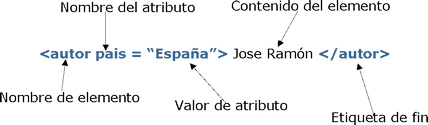
****WAP y WML.

**Sintaxis:**

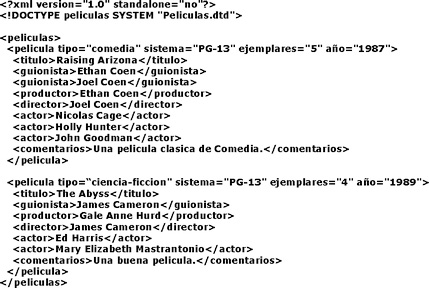
* Todo elemento tiene que tener su correspondiente etiqueta de inicio y de cierre, o una sola etiqueta vacía.
* Todo documento, debe haber un elemento (llamado raíz de documento) que contenga a los demás.
* Todos los elementos deberán estar correctamente anidados.
* ****Todos los valores de los atributos deberán ir entre comillas.

**Estructura**

Un documento XML está formado por datos de caracteres y marcado, el marcado lo forman las etiquetas:

****

**Ejemplo**



# Investigar que es un endpoint en un api.



Es un dispositivo informático remoto que se comunica con una red a la que está conectado. Los endpoints son las URLS de un API o un backend que responden a una petición.

Endpoint: URL de un servicio que utiliza un sitio web para cargar o consumir información.

**Ejemplos de Enpoint:**

Una vez finalizada la definición del modelo y errores que devolveremos, el siguiente paso es definir los endpoints y métodos http que nuestro API expondrá, por ejemplo, un servicio para filtrar productos por código EAN sería algo así:

"paths": {

"/products": {

"get": {

"summary": "Product listing",

"parameters": [

{

"name": "eanCode",

"in": "query",

"description": "EAN Code",

"type": "string",

"required": false

},

{

"name": "description",

"in": "query",

"description": "Description",

"type": "string",

"required": false

}

],

"tags": [

"products"

],

"responses": {

"200": {

"description": "Product listing",

"schema": {

"$ref": "#/definitions/Product"

}

},

"400": {

"description": "Bad request",

"schema": {

"$ref": "#/definitions/Error"

}

},

"401": {

"description": "Unauthorized",

"schema": {

"$ref": "#/definitions/Error"

}

},

"403": {

"description": "Forbidden",

"schema": {

"$ref": "#/definitions/Error"

}

},

"404": {

"description": "Not found",

"schema": {

"$ref": "#/definitions/Error"

}

},

"405": {

"description": "Method not allowed",

"schema": {

"$ref": "#/definitions/Error"

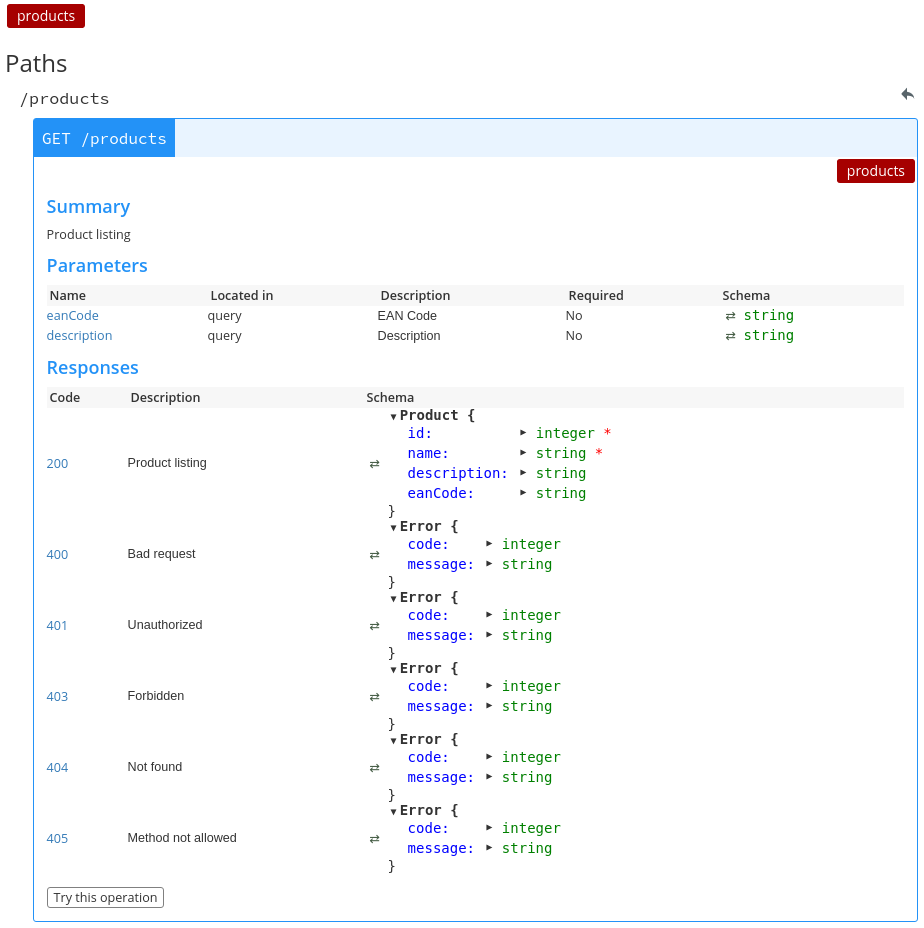
}

}

}

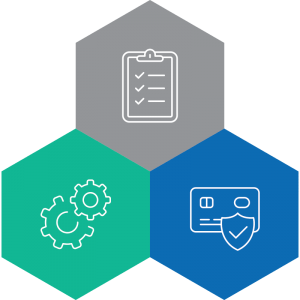
},

Este código generaría el siguiente endpoint que se muestra a continuación:

****

**Para que sirve:**

Se centran en prevenir amenazas a través de diversos enfoques, juntando los tradicionales con los más modernos. Es necesario que las empresas comprendan las amenazas de las que se intentan prevenir y adquieran el mayor número de técnicas posibles.

****Dentro de las técnicas más tradicionales nos podemos encontrar con los antivirus convencionales que frenan los malware conocidos, bloqueo de las aplicaciones, monitorización de los comportamientos, sistemas de prevención de intrusión, prevención de la pérdida de datos , la  protección y control web.

**Estructura:**

**Se implementan como peticiones HTTP, en las que:**

La URL representa el recurso

****http://www.formandome.es/api/cursos/1

**El método (HTTP Verbs) representa la operación:**

GET http://www.formandome.es/api/cursos/1

**El código de estado HTTP representa el resultado:**

200 OK HTTP/1.1

404 NOT FOUND HTTP/1.1

**Creación de recursos:**

La URL estará “abierta” (el recurso todavía no existe y por tanto no tiene id)

El método debe ser POST

<http://eventos.com/api/eventos/3/comentarios>

# Infografía Arquitectura MCV.

URL infografía.

<https://www.canva.com/design/DAEIq5ZDoLU/share/preview?token=U8PYtJz54_ljEsBqvgEWkw&role=EDITOR&utm_content=DAEIq5ZDoLU&utm_campaign=designshare&utm_medium=link&utm_source=sharebutton>



# Investigación sobre Blade en Laravel.

**¿Qué es Blade en Laravel?**

Laravel soporta el motor de plantillas Blade fuera de la caja. El motor de plantillas Blade nos permite crear plantillas maestras y plantillas infantiles para cargar contenido de plantillas maestras, podemos tener variables, bucles y declaraciones condicionales dentro del archivo Blade.

Blade es básicamente un sub-lenguaje muy sencillo, que antes de ser usado por nuestra aplicación, es compilado a PHP plano. Esta librería permite realizar todo tipo de operaciones con los datos, además de la sustitución de secciones de las plantillas por otro contenido, herencia entre plantillas, definición de layouts o plantillas base, etc.

Provee ventajas pues además de modularizar nuestras vistas de una forma sorprendente, también nos permite usar estructuras de control y variables de PHP directamente en ellas.

Para poder utilizar el sistema de plantillas Blade se tiene que utilizar la extensión .blade.php para referenciar una vista desde el fichero de rutas o desde el controlador se utilizara view ('home') dependiendo si el fichero se llama home.php o como home. Blade.php.

Básicamente: {{ }} es un sustituto de <?php ?> que por un lado es más corto y fácil de escribir, por otro usar etiquetas PHP (<? ?>) se “mezcla” con las etiquetas del HTML, así que usar motores como Blade es más “limpio” por así decirlo.

Blade sólo reemplaza {{ \* }} por <?php echo \* ?>, es decir, todo el código PHP que quieras escribir es válido dentro de {{ }}, por ejemplo:

<h1>Hello {{ strtoupper($name) }}</h1>

Blade es un motor de plantillas potente que nos permite modularizar y estilizar a un gran nivel nuestro HTML.

**Sentencias de Blade y función.**

**@extends ('nombre\_témplate'):** Esta sentencia nos ayuda a decirle a una vista cual va a ser el témplate que se va a usar.

**@yield ('nombre'):** Esta sentencia nos permite declarar un futuro section de nuestro HTML que se definirá en las vistas que son heredadas y no puede agregarse algún tipo de contenido por defecto, este sólo se usa en archivos que toman el rol de Témplate.

**@section('nombre'):** Esta sentencia tiene dos usos dependiendo de que queremos declarar, el primero es que nos permite declarar una sección dentro del témplate que puede tener un contenido por defecto que si no es redefinido en la vista que herede el témplate entonces aparecerá; el segundo nos permite asignar el contenido en una seccion que fue declarada en nuestro témplate, es decir esta palabra section se usa tanto en el témplate como en las vistas hijas, una diferencia más es que si se usa en el témplate entonces la sección termina con un @show, pero si se usa en una vista hija entonces termina la seccion con un @stop.

**@show:** Esta sentencia se usa para decir donde termina el section definido en el témplate.

**@parent:** Esta sentencia nos ayuda a cargar el contenido por defecto de un section del témplate, esto podemos usarlo cuando queremos agregar más contenido dentro pero sin alterar el contenido por defecto, es decir agregarle más HTML, esta sentencia se usa dentro de un section, podemos hacer un símil con el súper () de Java que sirve para llamar al constructor de la superclase de la que se hereda.

**@stop:** Esta sentencia nos permite decir donde termina un section cuando se usa el section dentro de las vistas hijas.

**@include ('ruta.nombre'):** Esta sentencia nos agrega en el lugar donde sea usada un archivo blade.php que contiene un partial o fragmento parcial de HTML, si ese partial se encuentra en la raíz de las vistas no necesita más que el nombre sin la extensión blade.php.