***TEMA 1. FUNDAMENTOS DE INTERNET***

1. **QUÉ ES HTTP Y SUS DIFERENTES VERSIONES**
2. **SOLICITUD HTTP**
3. **RESPUESTA HTTP**
4. **IP & TCP**
5. **EL DNS Y SU FUNCIONAMIENTO**
6. **QUÉ ES EL NOMBRE DE DOMINIO**
7. **QUÉ ES EL ALOJAMIENTO WEB**

***1.*QUÉ ES HTTP Y SUS DIFERENTES VERSIONES**

**HTTP (Hypertext Transfer Protocol)**

* **¿Qué es?** El protocolo que permite la comunicación entre clientes (navegadores) y servidores web.
* **¿Cómo funciona?** Se basa en peticiones y respuestas. El cliente solicita recursos (páginas web, imágenes, etc.) y el servidor responde con la información solicitada.

**Versiones de HTTP**

* **HTTP/0.9 (1991)**: La versión inicial, muy básica. Solo permitía solicitudes GET y no incluía encabezados.
* **HTTP/1.0 (1996)**: Introdujo los encabezados, lo que permitió una mayor flexibilidad. Sin embargo, cada conexión requería una nueva conexión TCP, lo que generaba sobrecarga.
* **HTTP/1.1 (1997)**: La versión más utilizada durante mucho tiempo. Permitió conexiones persistentes (múltiples solicitudes en la misma conexión TCP), mejorando el rendimiento.
* **HTTP/2 (2015)**: Una revisión importante que introdujo multiplexación (múltiples solicitudes y respuestas simultáneas en la misma conexión), compresión de encabezados y otras mejoras para reducir la latencia y mejorar la velocidad.
* **HTTP/3 (2020)**: La versión más reciente, basada en el protocolo QUIC sobre UDP. Ofrece mejoras en velocidad, seguridad y resistencia a la pérdida de paquetes.

**En resumen:** HTTP ha evolucionado para mejorar la eficiencia y la velocidad de la comunicación web. Cada nueva versión ha introducido mejoras significativas en comparación con la anterior.

**COOKIES**

**Qué son las cookies?**

\* Pequeños archivos de texto que los sitios web almacenan en tu navegador.

\* Contienen información sobre tu actividad en el sitio.

¿**Para qué sirven?**

\* Recordar tus preferencias:

\* Datos de inicio de sesión

\* Idioma preferido

\* Configuración del carrito de compras

\* Personalizar tu experiencia:

\* Contenido y publicidad personalizada

\* Analizar el uso de los sitios web:

\* Páginas visitadas

\* Enlaces en los que haces clic

\* Tiempo en cada página

\* Realizar un seguimiento de tu actividad en línea:

\* Empresas de publicidad rastrean tu actividad en diferentes sitios web

**Tipos de cookies**

\* Técnicas: Esenciales para el funcionamiento básico del sitio web.

\* De personalización: Recuerdan tus preferencias.

\* De análisis: Recopilan datos sobre cómo utilizas el sitio web.

\* Publicitarias: Se utilizan para mostrarte anuncios personalizados.

**HTTPS**

¿Por qué usar HTTPS en Python?

**Seguridad**

\* Cifrado de datos: Protege la información sensible (contraseñas, datos personales, etc.) de miradas indiscretas.

\* Autenticación: Verifica la identidad del servidor y evita suplantaciones.

\* Integridad: Asegura que los datos no sean modificados durante la transmisión.

**SEO**

\* Google prioriza sitios web con HTTPS, lo que mejora el posicionamiento en buscadores.

**Confianza del usuario**

\* El candado verde en la barra de direcciones indica un sitio seguro, lo que aumenta la confianza del usuario.

**Prevención de ataques**

\* Evita ataques de intermediario (Man-in-the-Middle) donde un atacante intercepta y modifica la comunicación.

**Cumplimiento normativo**

\* En algunos sectores, como el financiero o el de salud, es obligatorio usar HTTPS para proteger los datos de los usuarios.

**¿Cómo usar HTTPS en Python?**

\* Librerías: Utiliza bibliotecas como requests para realizar solicitudes HTTPS de forma segura.

\* Certificados SSL: Obtén un certificado SSL para tu servidor y configúralo correctamente.

**2. SOLICITUD HTTP**

Una solicitud HTTP es un mensaje que un cliente (como un navegador web o un programa en Python) envía a un servidor web para solicitar un recurso (como una página web, una imagen o datos).

**COMPONENTES DE UNA SOLICITUD HTTP**

1**: Línea de solicitud**

*a)Método HTTP:*

Indica la acción que se desea realizar sobre el recurso. Los más comunes son:

\* GET: Solicita un recurso.Espera que se le devuelva información.

\* POST: Envía datos al servidor para su procesamiento(ej: nombre de usuario, información de un formulario, etc)

\* PUT: Actualiza un recurso existente.

\* DELETE: Elimina un recurso.

*b) URL:* Identifica el recurso solicitado.

*c) Versión HTTP*: Indica la versión del protocolo HTTP utilizada.

**2:Encabezados**

Proporcionan información adicional sobre la solicitud(ej: el navegador del cliente, el formato de los datos solicitados, etc).Los más comunes son:

\* *Host*: Especifica el servidor al que se dirige la solicitud.

\* *User-Agent*: Identifica el cliente que realiza la solicitud.

\* *Accept*: Indica los tipos de contenido que el cliente puede aceptar.

*\* Content-Type*: Especifica el tipo de contenido del cuerpo de la solicitud (si lo hay).

**3:Cuerpo (opcional):**

\* Contiene datos que se envían al servidor, por ejemplo, en una solicitud POST o PUT.

*Ejemplo de solicitud HTTP*

*GET /index.html HTTP/1.1*

*Host: www.example.com*

*User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/58.0.3029.110 Safari/537.3*

*Accept: text/html, application/xhtml+xml, application/xml;q=0.9, \*/\*;q=0.8*

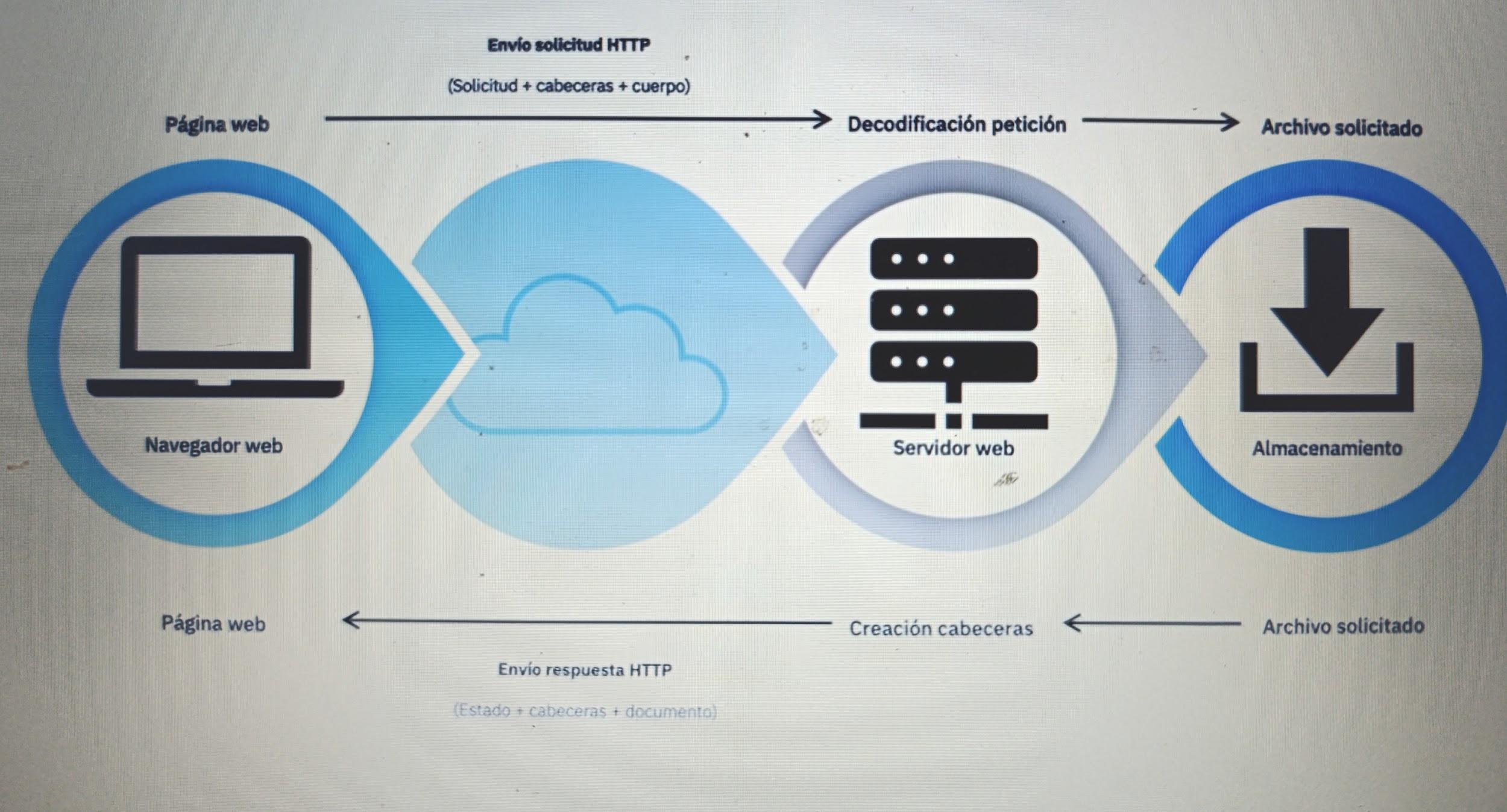
**PROCESO DE UNA SOLICITUD HTTP**

1: El cliente envía la solicitud HTTP al servidor.

2: El servidor recibe la solicitud y la procesa (busca el archivo solicitado del almacenamiento del servidor).

3: El servidor envía una respuesta HTTP al cliente.

4:El cliente recibe la respuesta y la utiliza para mostrar el recurso solicitado o realizar alguna otra acción.



\*\*\*En Python:

Puedes realizar solicitudes HTTP en Python utilizando bibliotecas como requests. Esta biblioteca simplifica la creación y el envío de solicitudes HTTP, así como el procesamiento de las respuestas

**3. RESPUESTA HTTP**

Una respuesta HTTP es un mensaje que un servidor web envía a un cliente (como un navegador web o un programa en Python) en respuesta a una solicitud HTTP.

La respuesta contiene información sobre el resultado de la solicitud y, en algunos casos, los datos solicitados.

**COMPONENTES DE UNA RESPUESTA HTTP**

**1: Línea de estado:**

**\* Versión HTTP**: Indica la versión del protocolo HTTP utilizada.

**\* Código de estado**: Un número de tres dígitos que indica el resultado de la solicitud. Algunos códigos comunes son:

\* 200 OK: La solicitud se ha completado correctamente.

\* 404 Not Found: El recurso solicitado no se ha encontrado.

\* 500 Internal Server Error: Se ha producido un error en el servidor.

**2:Frase de estado:**

Una breve descripción del código de estado.

**3:Encabezados:**

Proporcionan información adicional sobre la respuesta, como el tipo de contenido, la fecha, etc.Algunos encabezados comunes son:

\* *Content-Type*: Especifica el tipo de contenido del cuerpo de la respuesta.

\* *Content-Length*: Indica la longitud del cuerpo de la respuesta.

\* *Date*: Indica la fecha y hora en que se generó la respuesta.

**4: Cuerpo (opcional):**

Contiene los datos solicitados, por ejemplo, el código HTML de una página web, una imagen, etc.

*Ejemplo de una respuesta HTTP:*

*HTTP/1.1 200 OK*

*Content-Type: text/html*

*Content-Length: 1234*

*Date: Tue, 21 May 2024 12:00:00 GMT*

*<!DOCTYPE html>*

*<html>*

*<head>*

*<title>Página de ejemplo</title>*

*</head>*

*<body>*

*<h1>Hola, mundo</h1>*

*</body>*

*</html>*

**4. IP y TCP**

**El Protocolo de Internet** **(IP)**

Es el protocolo encargado de enrutar los paquetes de datos a través de Internet.

\* Asigna direcciones IP únicas a cada dispositivo conectado a la red, tanto origen como destino.

\* Divide los datos en paquetes y los envía a través de diferentes rutas.

\* No garantiza la entrega de los paquetes, solo se asegura de que lleguen a su destino.

**Protocolo TCP (Transmission Control Protocol)**

Es el protocolo encargado de establecer una conexión entre dos dispositivos.

\* Divide los datos en paquetes y los numera para asegurar su orden.

\* Verifica la entrega de los paquetes y los reenvía si es necesario.

\* Asegura la entrega de los datos de forma confiable y en orden.

*En resumen, el protocolo IP se encarga de enviar los datos a su destino, mientras que el protocolo TCP se encarga de que los datos lleguen de forma confiable y en orden. Ambos protocolos trabajan juntos para permitir la comunicación a través de Internet.*

**5.EL DNS Y SU FUNCIONAMIENTO**

DNS (Domain Name System) es un registro que contiene nombres de sitios web y direcciones IP asociadas.

Las DNS son un sistema jerárquico que se ocupa de administrar los nombres de dominio.

Una especie de base de datos y traductor para que, en vez de leer la IP 172.11.122.45 -un número asociado a la máquina donde está alojado el dominio-, leamos miweb.com, algo legible y fácil de recordar.

**ESQUEMA DE FUNCIONAMIENTO**

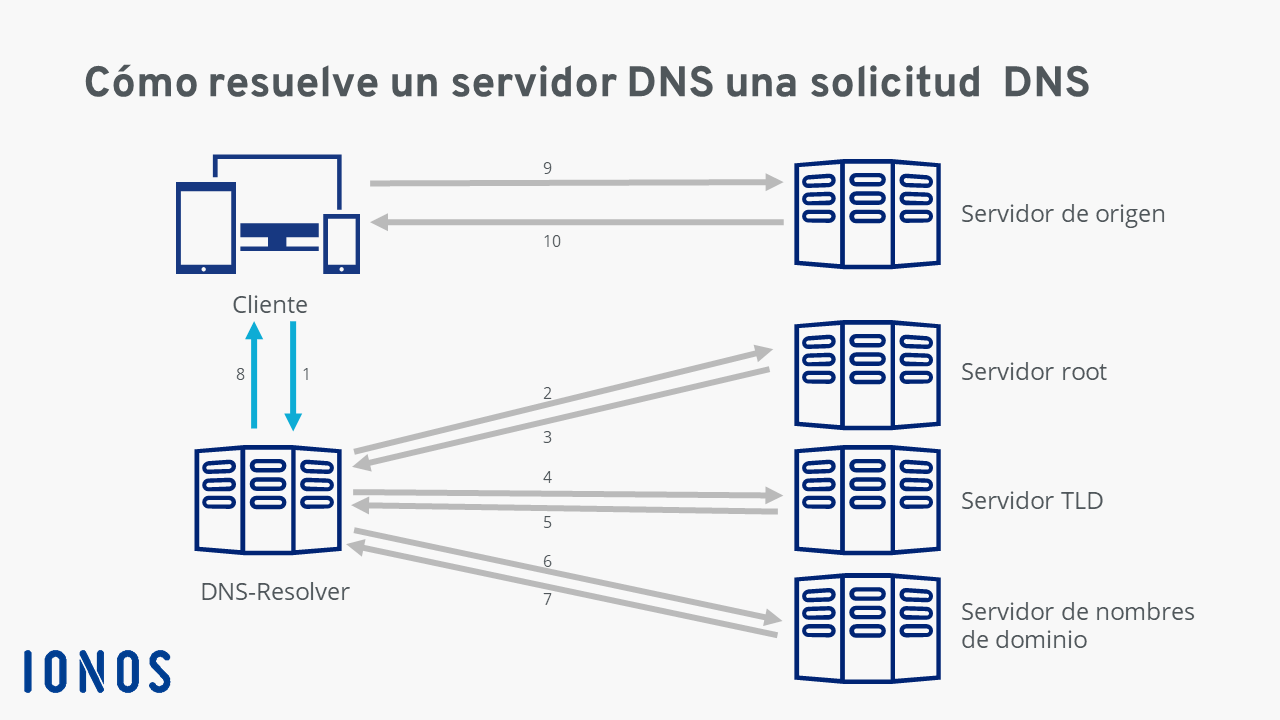
****

1. Cliente DNS: el que realiza peticiones de resolución de nombres.

2. Servidor DNS: los que contestan, resolviendo la petición mediante un sistema de árbol. Cuando configuramos nuestra conexión a internet por primera vez -la mayoría de las ocasiones las DNS vienen asignadas de forma automática-, estamos marcando las direcciones de los servidores DNS.

* **TDL** *Servidor DNS primario:* aquel que guarda la info de distintos nombres de dominio en su propia base de datos.
* **SDL** *Servidor DNS secundario:* si un servidor DNS acude a otro, se convierte en secundario.

3. Zonas de autoridad=servidores Apache: responsables de resolver el dominio que estamos solicitando.



**PARTES DE UN DOMINIO**



Estas son las partes habituales en cualquier dominio:

1. *Protocolo*.Ejemplo: http:// o https://
2. *Subdominio*. Ejemplo: www.
3. *Dominio*. Ejemplo: miweb ó carpinteriadejose
4. *Dominio de primer nivel (TDL*).Ejemplo: .com ó .es ó .fr
5. *Subcarpeta o directorio*. Ejemplo:/carpeta/
6. *Extensiones* (ejemplo: .html o en miweb.com.cat; la parte ".com.cat" sería la extensión)

**6. QUÉ ES EL NOMBRE DE DOMINIO**

Un nombre de dominio es una dirección personalizada y única para tu página web.

Se compone de dos partes: el *nombre* y la *extensión ó TLD.*

Ejemplo: "ibm.com" contiene el *nombre* "ibm" y la *extensión ó TLD*  ".com".

1. **QUÉ ES EL ALOJAMIENTO WEB /HOSTING**

El *hosting* es el espacio donde se aloja un sitio web para que sea visible en internet para todo el mundo.

Ese espacio se almacena físicamente en un *servidor web.*

*Un servidor web es una máquina física, un ordenador muy potente, con una serie de características concretas (procesadores, memoria RAM, disco duro) para almacenar el hosting o <<espacio>> para 1 o varias webs y que sean visibles en Internet.*

**TIPOS DE HOSTING**

1. **Hosting Compartido**: Varios sitios web comparten un mismo servidor.

* Económico, ideal para proyectos pequeños y principiantes.
* Recursos limitados, rendimiento puede verse afectado por otros sitios.

1. **Hosting VPS** (Servidor Virtual Privado):

* Un servidor físico se divide en varios entornos virtuales.
* Más recursos y control que el hosting compartido.
* Recomendado para sitios web con tráfico moderado y necesidades específicas.

1. **Hosting Dedicado:**

* Un servidor completo para un solo usuario.
* Máximo control, rendimiento y seguridad.
* Ideal para sitios web con alto tráfico y requerimientos exigentes.

1. **Hosting en la Nube** (Cloud Hosting):

* Recursos distribuidos en múltiples servidores interconectados.
* Escalabilidad, flexibilidad y alta disponibilidad.
* Adecuado para proyectos con necesidades variables y picos de tráfico.

**Otros Tipos de Hosting**

1. **Hosting Reseller**:
   * Permite revender espacio y recursos a terceros.
   * Ideal para emprendedores y agencias web.
2. **Hosting WordPress:**
   * Optimizado para sitios web creados con WordPress.
   * Incluye herramientas y configuraciones específicas para este CMS.

**Criterios de Clasificación Adicionales**

* *Sistema Operativo*: Linux (popular y versátil) o Windows (para aplicaciones específicas).
* *Ubicación del Servidor*: Influencia en la velocidad de carga para los visitantes.
* *Tipo de Soporte*: Nivel de asistencia técnica ofrecida.

**Consideraciones al elegir un Hosting**

* *Necesidades del Proyecto*:

tráfico esperado, recursos requeridos, tipo de sitio web

* *Presupuesto*: Costos del plan y posibles gastos adicionales.
* *Nivel de Experiencia*: Facilidad de uso, herramientas disponibles, soporte técnico.