Alumno:Liset Aielo
Materia: Programación avanzada

1. ¿Cómo surgió el sistema de documentos distribuidos WWW?

El sistema de documentos distribuidos conocido como la World Wide Web (WWW) surgió como una solución a la necesidad de compartir información de manera eficiente entre investigadores de diferentes instituciones. Tim Berners-Lee, un científico británico del CERN (Organización Europea para la Investigación Nuclear), desarrolló la idea de un sistema de hipertexto para enlazar y acceder a información distribuida a través de la red. La WWW permitió la conexión de documentos utilizando enlaces de hipertexto, facilitando la navegación entre diferentes fuentes de información.

2. ¿En qué año se creó la WWW? ¿Quién la creó? ¿Cuál fue el propósito inicial?

La WWW fue creada en **1989** por **Tim Berners-Lee** mientras trabajaba en el CERN. Su propósito inicial era mejorar la colaboración entre científicos de diferentes universidades e instituciones alrededor del mundo, permitiendo compartir información de manera rápida y estructurada. La primera propuesta fue escrita por Berners-Lee en marzo de 1989, y en 1990 desarrolló el primer navegador web, llamado WorldWideWeb, y el primer servidor web, llamado CERN HTTPd.

3. ¿Sobre qué red funcionaba? ¿Cuáles fueron los componentes iniciales?

La WWW funcionaba sobre internet, una red que ya existía y que conectaba computadoras alrededor del mundo. Internet proporcionaba la infraestructura de red necesaria para la transferencia de datos entre servidores y clientes.

Los componentes iniciales de la WWW fueron:

1. **HTML (Hypertext Markup Language)**: El lenguaje utilizado para crear páginas web.
2. **HTTP (Hypertext Transfer Protocol)**: El protocolo de comunicación utilizado para transmitir datos entre el navegador y el servidor web.
3. **URL (Uniform Resource Locator)**: El sistema de direcciones que permite identificar y localizar recursos en la web.
4. **Navegador web (Browser)**: La primera versión fue WorldWideWeb, desarrollada por Berners-Lee, que permitía visualizar documentos de hipertexto.
5. **Servidor web**: El software que aloja las páginas web y las entrega al navegador cuando se solicita.

Estos componentes formaron la base de la infraestructura que permitió que la WWW se convirtiera en el sistema global que conocemos hoy.

4. Nombre y describa las distintas etapas de evolución en la web. Realice una cronología de tecnologías que surgieron en la web hasta la actualidad.
La evolución de la web se puede dividir en varias etapas clave:

1. Web 1.0 (1991-2004):

Descripción: La primera etapa de la web, también conocida como la web estática o de solo lectura. Los usuarios podían acceder a información publicada, pero no interactuar con el contenido de manera significativa.
Tecnologías: HTML básico, navegadores web como Mosaic y Netscape, HTTP 0.9 y 1.0.
2. Web 2.0 (2004-presente):

Descripción: Introducción de la web dinámica, que permite la interacción del usuario y la generación de contenido. Las redes sociales, blogs y wikis son ejemplos claros. La participación y la colaboración en línea son sus características principales.
Tecnologías: AJAX, HTML5, CSS3, JavaScript, redes sociales, plataformas de blogging, APIs.
3. Web 3.0 (En desarrollo):

Descripción: También conocida como la web semántica. En esta etapa, se busca una mayor comprensión y organización de la información, permitiendo que los datos sean más accesibles y comprensibles tanto para humanos como para máquinas.
Tecnologías: RDF, OWL, SPARQL, blockchain, inteligencia artificial, aprendizaje automático.
4. Web 4.0 (Futuro):

Descripción: Se anticipa como la web inteligente o autónoma. Se espera que esté totalmente integrada con la vida cotidiana, con la interacción en tiempo real y la personalización a gran escala.
Tecnologías: IoT (Internet of Things), realidad aumentada, realidad virtual, inteligencia artificial avanzada.
Cronología de Tecnologías Clave:
1991: Nace la World Wide Web (HTML, HTTP).
1993: Mosaic, el primer navegador web gráfico.
1994: Netscape Navigator, primera versión comercial de un navegador.
1995: JavaScript, lenguaje de scripting para la web.
1998: Google se funda, transformando la búsqueda web.
2001: Primeros pasos de la Web Semántica (Tim Berners-Lee).
2004: Web 2.0, auge de las redes sociales (Facebook, YouTube).
2008: HTML5, estándar para contenido multimedia.
2010: HTML5 se convierte en un estándar clave, reemplazando Flash.
2018: Web 3.0 toma forma, integrando blockchain y AI.
5. ¿Qué es un Servidor WWW?
Un Servidor WWW es un software o sistema de hardware que almacena, procesa y entrega páginas web a los usuarios a través de Internet. Cuando un usuario solicita una página web, el

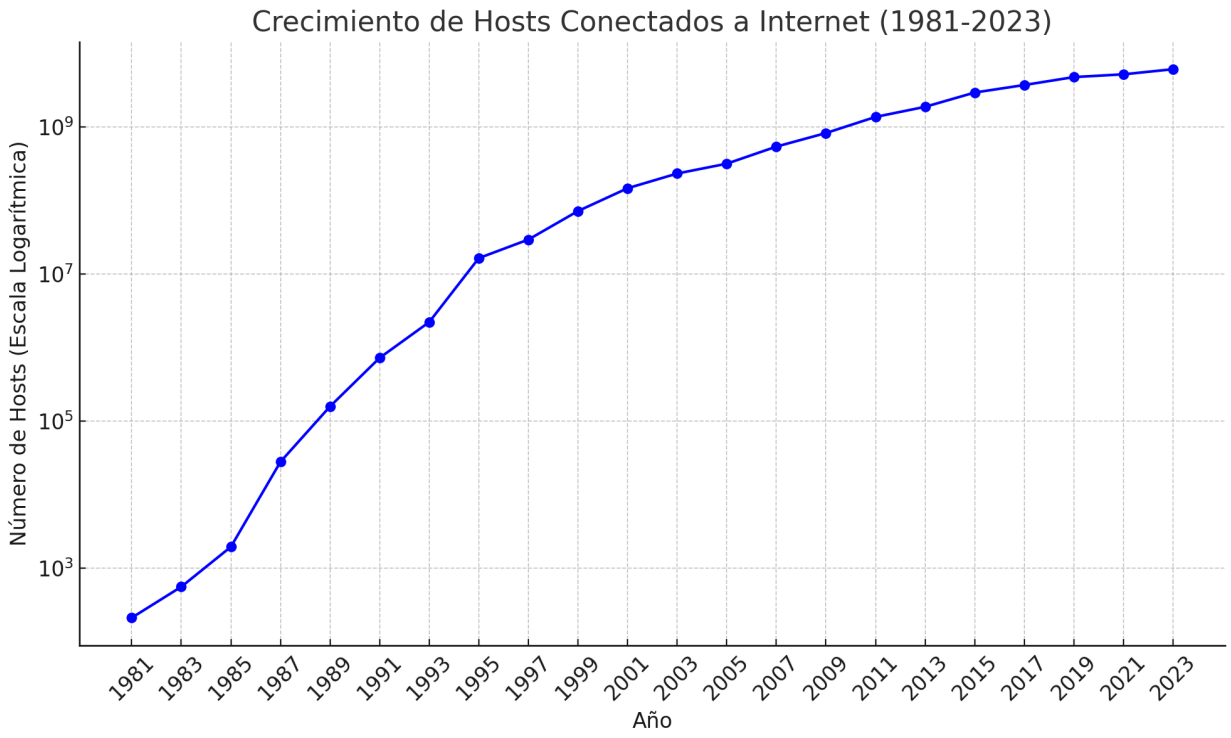
servidor responde enviando el contenido solicitado al navegador del cliente utilizando el protocolo HTTP o HTTPS. Los servidores WWW son fundamentales para el funcionamiento de la web, ya que permiten acceder a la información desde cualquier parte del mundo.

6. ¿Qué es un cliente web? Explique detalladamente.
Un cliente web es cualquier dispositivo o software que solicita, recibe y presenta información desde un servidor web. El cliente más común es un navegador web como Chrome, Firefox o Safari, que permite a los usuarios acceder y navegar por páginas web. Los clientes web pueden también ser aplicaciones móviles o software especializado que interactúa con la web. El cliente envía solicitudes HTTP al servidor, que luego devuelve la respuesta con los recursos solicitados, como HTML, CSS, JavaScript o datos multimedia.

7. ¿Cómo está compuesto el sistema WWW actual?
El sistema WWW actual está compuesto por varios elementos clave:

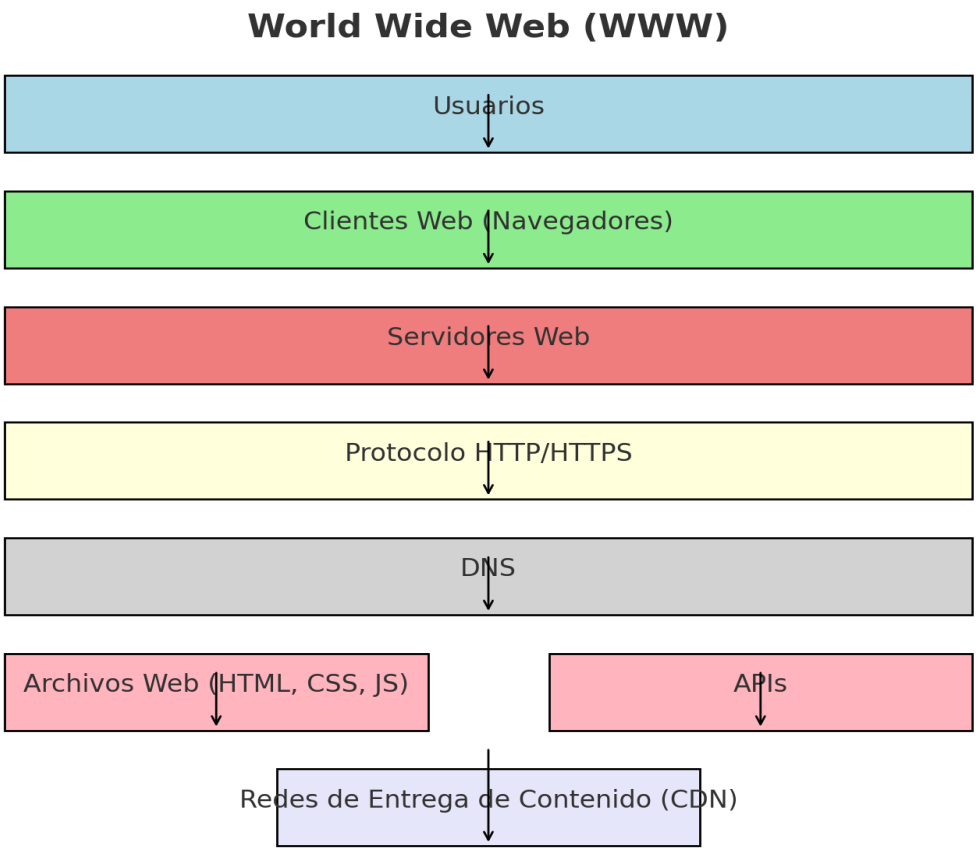
Servidores Web: Almacenan y entregan contenido a los usuarios.
Clientes Web (Navegadores): Solicitan y muestran el contenido.
Protocolos (HTTP/HTTPS): Normas que gobiernan la transferencia de datos entre servidores y clientes.
Lenguajes de Marcado y Scripting (HTML, CSS, JavaScript): Lenguajes que estructuran y presentan contenido en la web.
DNS (Sistema de Nombres de Dominio): Convierte nombres de dominio legibles (como www.ejemplo.com) en direcciones IP.
APIs: Permiten la interacción entre diferentes servicios y aplicaciones en la web.
Redes de Entrega de Contenido (CDN): Mejoran la velocidad de entrega del contenido distribuyendo copias de los recursos en diferentes ubicaciones geográficas.

8. Realice un diagrama de crecimiento de host conectados a Internet desde su inicio hasta la actualidad.



Este diagrama de crecimiento de hosts conectados a Internet es desde 1981 hasta 2023. El gráfico utiliza una escala logarítmica en el eje vertical para mostrar el crecimiento exponencial de los hosts conectados a lo largo del tiempo. Como se puede observar, el número de hosts ha crecido de manera espectacular, especialmente a partir de los años 90, coincidiendo con la popularización de la World Wide Web.

9. Realiza un diagrama conceptual de composición de la web. ¿Cuáles son los componentes de la WWW?



Los componentes clave del sistema WWW actual están organizados y conectados, mostrando cómo interactúan entre sí. Cada caja representa un componente esencial, desde los usuarios y navegadores web hasta servidores, protocolos, DNS, archivos web, APIs y redes de entrega de contenido (CDN).

10. ¿Cuál es la arquitectura del sistema distribuido de documentos?

La arquitectura del sistema distribuido de documentos (WWW) es de tipo cliente-servidor. Se compone de:

1. **Clientes (Navegadores Web):** Interactúan con los usuarios, solicitando documentos de los servidores.
2. **Servidores:** Almacenan, gestionan y entregan documentos a los clientes.
3. **Protocolos (HTTP/HTTPS):** Controlan la comunicación entre clientes y servidores.
4. **DNS:** Resuelve nombres de dominio en direcciones IP para la localización de servidores.

Esta arquitectura permite la distribución de documentos a través de múltiples servidores a nivel mundial.

11. ¿Cómo funciona el sistema WWW?

El sistema WWW funciona mediante una serie de interacciones entre clientes y servidores usando protocolos estándar:

1. **Solicitud (Request):** Un cliente web (navegador) envía una solicitud HTTP a un servidor, generalmente a través de una URL.
2. **Resolución de Dominio:** El DNS traduce el nombre de dominio en la URL a una dirección IP del servidor.
3. **Respuesta (Response):** El servidor procesa la solicitud y envía de vuelta el contenido solicitado, como una página HTML.
4. **Renderización:** El cliente (navegador) procesa y muestra el contenido al usuario.
5. **Interacción:** El usuario puede interactuar con la página, generando nuevas solicitudes (como clics en enlaces o formularios) que repiten el proceso.

Este ciclo de solicitudes y respuestas es lo que permite la navegación en la web.

12. ¿Qué es HTML? ¿De dónde proviene?

HTML (HyperText Markup Language) es el lenguaje de marcado utilizado para estructurar y presentar contenido en la web. HTML utiliza una serie de etiquetas (tags) que definen elementos como encabezados, párrafos, enlaces, imágenes y otros componentes multimedia en una página web.

Origen: HTML fue creado por Tim Berners-Lee en 1991 en el CERN, como parte de la creación de la World Wide Web. Fue diseñado para permitir la vinculación de documentos a través de hipertextos, lo que posibilitó la navegación entre páginas web.

13. ¿A qué se le llama página web estática? ¿y página dinámica? ¿y página activa?

- **Página web estática:** Es una página cuyo contenido no cambia en función de la interacción del usuario. Se muestra tal como fue diseñada, y su contenido solo se actualiza mediante cambios manuales en el código HTML. Ejemplos típicos son sitios de información básica o portafolios.
- **Página web dinámica:** Es una página cuyo contenido se genera en tiempo real, basado en la interacción del usuario o en datos provenientes de un servidor. Utiliza lenguajes de programación como PHP, Python o JavaScript para cambiar el contenido dinámicamente. Ejemplos incluyen redes sociales, blogs, y sitios de comercio electrónico.
- **Página web activa:** Similar a la página dinámica, pero con un enfoque en la interacción compleja con el usuario. Se actualiza constantemente y puede involucrar la ejecución de scripts del lado del cliente o servidor para actualizar el contenido sin recargar la página, utilizando tecnologías como AJAX.

14. ¿Qué es una URL? ¿Qué función cumple en la Web? ¿Qué es un servidor de archivos basado en URL?

- **URL (Uniform Resource Locator):** Es la dirección que se utiliza para acceder a recursos en la web. Una URL indica la localización de un recurso y el protocolo que se debe utilizar para acceder a él (como HTTP o HTTPS).
- **Función en la Web:** La URL es esencial para la navegación web, ya que permite a los usuarios localizar y acceder a diferentes recursos como páginas web, imágenes, videos y archivos.
- **Servidor de archivos basado en URL:** Es un servidor que entrega archivos a los clientes en respuesta a solicitudes URL específicas. Esto puede incluir la entrega de documentos HTML, imágenes, videos, y otros archivos multimedia a través de la web.

15. ¿Dónde y cómo se ejecuta el lenguaje HTML?

HTML no se "ejecuta" en el sentido tradicional como un lenguaje de programación. En su lugar, se **interpreta** y **se renderiza** por el navegador web en el dispositivo del usuario. Cuando un navegador recibe un archivo HTML, lo interpreta para estructurar y mostrar el contenido de la página web de acuerdo con las etiquetas y elementos definidos en el archivo.

16. ¿Qué es un lenguaje script en la tecnología WWW?

Un **lenguaje script** en la tecnología WWW es un lenguaje de programación que se ejecuta dentro de un entorno específico, como un navegador web (JavaScript) o un servidor web (PHP). Estos lenguajes se utilizan para automatizar tareas, generar contenido dinámico y mejorar la interactividad de las páginas web.

17. ¿Qué es un plugin? ¿Qué es una cookie?

- **Plugin:** Es un software adicional que se integra con un navegador o aplicación para proporcionar funciones adicionales que no están incluidas de forma predeterminada. Por ejemplo, plugins para reproducir videos, mostrar gráficos en 3D o bloquear publicidad.
- **Cookie:** Es un pequeño archivo de texto almacenado en el navegador del usuario por un sitio web. Las cookies se utilizan para recordar información sobre la visita del usuario, como preferencias, datos de inicio de sesión o artículos en un carrito de compras. Ayudan a personalizar la experiencia de navegación y son fundamentales para el funcionamiento de muchas funcionalidades web.

18. ¿Qué es JavaScript? ¿Dónde y quién ejecuta JavaScript? ¿Cómo nació JavaScript?

- **JavaScript:** Es un lenguaje de programación ligero y orientado a objetos utilizado para crear contenido interactivo en páginas web, como formularios dinámicos, menús desplegables, animaciones, y mucho más.
- **Ejecución:** JavaScript se ejecuta en el navegador web del usuario, permitiendo que las páginas web respondan a la interacción del usuario sin necesidad de recargar la página. Además, JavaScript también puede ejecutarse en el servidor (Node.js).
- **Nacimiento:** JavaScript fue creado por Brendan Eich en 1995 mientras trabajaba en Netscape. Inicialmente llamado "Mocha", luego "LiveScript" y finalmente "JavaScript", fue diseñado para proporcionar una manera sencilla de añadir interactividad a las páginas web. Se desarrolló rápidamente como un estándar y ahora es uno de los lenguajes más utilizados en el desarrollo web.

19. ¿Qué es un Applet de Java? ¿Cómo puede un Browser ejecutar código Java?

- Un applet de Java es un pequeño programa escrito en el lenguaje de programación Java que se puede insertar en una página web. Los applets se ejecutan dentro de un contenedor en el navegador, llamado máquina virtual Java (JVM), que interpreta y ejecuta el código Java del applet. El navegador debe tener el plugin de Java instalado para poder ejecutar estos applets.

20. ¿Cómo funciona la tecnología Flash de Macromedia?

- La tecnología Flash de Macromedia (ahora Adobe) utiliza un formato de archivo específico (.swf) para crear contenido multimedia interactivo y animado en la web. Flash se ejecuta en un plugin de navegador que interpreta los archivos .swf, permitiendo la visualización de animaciones, gráficos y aplicaciones interactivas. Este contenido se presenta de manera fluida y con gráficos ricos, y el plugin maneja la ejecución del código de Flash dentro del navegador.

21. ¿Para qué se utiliza la tecnología CGI? ¿Qué lenguajes de programación se pueden utilizar para los scripts CGI?

- La tecnología CGI (Common Gateway Interface) se utiliza para permitir que los servidores web interactúen con programas externos que generan contenido dinámico. Los scripts CGI se ejecutan en el servidor para procesar solicitudes y generar respuestas que luego se envían al navegador del cliente. Los lenguajes de programación que se pueden utilizar para los scripts CGI incluyen Perl, Python, C, C++, y más recientemente, también se pueden utilizar lenguajes como PHP y Ruby.

22. ¿Qué es un “form” HTML? ¿para que se utiliza?

- Un "form" (formulario) HTML es un elemento que permite a los usuarios ingresar datos que luego se envían al servidor para su procesamiento. Se utiliza para recolectar información de los usuarios, como datos de registro, búsquedas, encuestas, etc. El formulario HTML contiene elementos como campos de texto, botones de opción, casillas

de verificación y botones de envío, que facilitan la interacción entre el usuario y el servidor.

23. ¿Qué son y cómo funcionan las tecnologías ASP, PHP y JSP?

- **ASP (Active Server Pages)**: Es una tecnología de Microsoft que permite la creación de páginas web dinámicas. Los scripts ASP se ejecutan en el servidor, generando contenido HTML dinámico que se envía al navegador del cliente.
- **PHP (Hypertext Preprocessor)**: Es un lenguaje de scripting del lado del servidor que se utiliza para crear contenido web dinámico. Los scripts PHP se ejecutan en el servidor, y el resultado se envía al navegador del cliente. PHP es conocido por su integración con bases de datos y su facilidad de uso.
- **JSP (JavaServer Pages)**: Es una tecnología de Java que permite la creación de contenido web dinámico utilizando Java. Los archivos JSP se procesan en el servidor para generar HTML que se envía al cliente. JSP permite incluir código Java en las páginas web para generar contenido dinámico.

24. ¿Qué es DHTML? ¿y XHTML?

- **DHTML (Dynamic HTML)**: Es una colección de tecnologías que permite la creación de páginas web interactivas y dinámicas. Incluye HTML, CSS y JavaScript. DHTML permite modificar el contenido y el diseño de una página web en respuesta a las acciones del usuario, sin necesidad de recargar la página.
- **XHTML (Extensible Hypertext Markup Language)**: Es una reformulación de HTML como un lenguaje XML. XHTML sigue reglas más estrictas que HTML y requiere que todos los elementos estén correctamente anidados y cerrados. Su propósito es mejorar la interoperabilidad entre navegadores y otros dispositivos.

25. ¿Qué es XML? Explique detalladamente.

- XML (Extensible Markup Language) es un lenguaje de marcado diseñado para almacenar y transportar datos de manera estructurada y legible tanto para humanos como para máquinas. XML permite definir sus propias etiquetas y estructuras de datos, lo que lo hace muy flexible y extensible. Es ampliamente utilizado para intercambiar datos entre sistemas y plataformas, y en aplicaciones como configuraciones de software y documentos web. XML proporciona una estructura jerárquica para los datos, facilitando su organización y validación.

26. ¿Qué son los Servicios Web? De un ejemplo de aplicación utilizando Servicios WEB. URL´s, Dominios y Direcciones.

- Los servicios web son aplicaciones basadas en estándares que permiten la comunicación entre diferentes sistemas a través de la web. Utilizan protocolos como HTTP y formatos de datos como XML o JSON para intercambiar información. Un ejemplo de aplicación utilizando servicios web es una aplicación de reserva de vuelos

que utiliza un servicio web para consultar la disponibilidad de vuelos en tiempo real desde una base de datos externa y mostrar los resultados al usuario.

27. ¿Qué es una URL? ¿Qué función cumple en la web? ¿Cuál es la estructura de una URL?

- Una URL (Uniform Resource Locator) es una dirección que se utiliza para acceder a recursos en la web. Su función es identificar de manera única un recurso en Internet y permitir que los navegadores lo localicen. La estructura de una URL típicamente incluye:
    - **Esquema**: Indica el protocolo utilizado (por ejemplo, http, https).
    - **Host**: El dominio del servidor (por ejemplo, www.ejemplo.com).
    - **Ruta**: La ubicación del recurso en el servidor (por ejemplo,Los métodos principales de HTTP/1.1 son:
- **GET**: Solicita un recurso del servidor. Es un método seguro y idempotente, lo que significa que no debe alterar el estado del recurso en el servidor y puede repetirse sin efectos adicionales.
  Ejemplo: GET /index.html HTTP/1.1
- **POST**: Envía datos al servidor para crear o actualizar un recurso. A diferencia de GET, POST no es idempotente, lo que significa que múltiples solicitudes POST pueden tener efectos diferentes.
  Ejemplo: POST /submit-form HTTP/1.1
- **PUT**: Envía datos al servidor para reemplazar un recurso existente. Es un método idempotente, por lo que repetir la misma solicitud PUT no debería cambiar el resultado.
  Ejemplo: PUT /update-resource HTTP/1.1
- **DELETE**: Solicita la eliminación de un recurso en el servidor. Este método es idempotente, y repetir la misma solicitud DELETE no debería tener efectos adicionales si el recurso ya ha sido eliminado.
  Ejemplo: DELETE /delete-resource HTTP/1.1
- **HEAD**: Similar a GET, pero solo solicita los encabezados del recurso sin el cuerpo. Se utiliza para obtener metainformación sobre el recurso.
  Ejemplo: HEAD /index.html HTTP/1.1
- **OPTIONS**: Solicita información sobre los métodos HTTP soportados por el servidor para un recurso específico. Es útil para conocer las capacidades del servidor.
  Ejemplo: OPTIONS /index.html HTTP/1.1
- **TRACE**: Devuelve la solicitud recibida por el servidor para propósitos de diagnóstico. Este método no es comúnmente utilizado y puede ser un riesgo de seguridad si no se maneja adecuadamente.
  Ejemplo: TRACE /index.html HTTP/1.1
    - **CONNECT**: Establece un túnel de comunicación con el servidor, comúnmente utilizado para establecer conexiones a través de un proxy HTTP.
      Ejemplo: CONNECT www.ejemplo.com:443 HTTP/1.1 /pagina.html).
    - **Parámetros de consulta** (opcional): Información adicional pasada al servidor (por ejemplo, ?id=123).

- **Fragmento** (opcional): Una referencia a una sección específica dentro del recurso (por ejemplo, #seccion1).

28. ¿Qué es un servidor de archivos basado en URL?

- Un servidor de archivos basado en URL es un servidor que permite a los usuarios acceder a archivos mediante una URL. Cuando un usuario solicita una URL, el servidor busca el archivo correspondiente en su sistema de archivos y lo envía al cliente. Esto es típico en servidores web que proporcionan acceso a archivos estáticos como documentos, imágenes y videos.

29. ¿Cuál es el formato de una URL en IPv6?

- En IPv6, las direcciones se incluyen entre corchetes en una URL. El formato general es:
  - http://[direccion_ipv6]/ruta
  - Por ejemplo: http://[2001:db8::1]/index.html

30. ¿Qué es un Dominio www? ¿Qué relación tiene con una URL?

- Un dominio www es una parte de una URL que se refiere a un nombre de dominio asociado con un sitio web. En una URL, el dominio identifica el servidor donde se aloja el recurso. La relación es que el dominio forma parte de la URL y es utilizado por el navegador para localizar el servidor que alberga el recurso solicitado.

31. ¿Cuál es la función del DNS en la WEB? Protocolos utilizados en la www

- El DNS (Domain Name System) se encarga de traducir nombres de dominio legibles por humanos en direcciones IP que los computadores utilizan para identificar y localizar servidores en la web. Así, cuando se ingresa una URL en un navegador, el DNS convierte el nombre de dominio en una dirección IP para que el navegador pueda conectarse al servidor adecuado.

32. ¿Cuál es la relación entre HTML y HTTP?

- HTML (Hypertext Markup Language) es el lenguaje utilizado para estructurar y presentar contenido en la web, mientras que HTTP (Hypertext Transfer Protocol) es el protocolo utilizado para transmitir ese contenido entre el servidor web y el navegador del cliente. En esencia, HTML define cómo se presenta la información, y HTTP se encarga de la transferencia de esa información a través de la red.

33. ¿Cuáles son las versiones de HTTP y cuáles sus diferencias?

- **HTTP/1.0**: La primera versión que introdujo el protocolo de transferencia de hipertexto. Su principal limitación es la falta de soporte para conexiones persistentes.
- **HTTP/1.1**: Introdujo mejoras significativas como conexiones persistentes, pipelining de solicitudes y respuestas, y mejores mecanismos de cacheo.

- **HTTP/2**: Introdujo multiplexión de solicitudes/respuestas en una sola conexión, compresión de encabezados y priorización de recursos, mejorando la eficiencia y velocidad de las comunicaciones.
- **HTTP/3**: Basado en QUIC (Quick UDP Internet Connections), ofrece mejoras en la latencia y la resistencia a la pérdida de paquetes, proporcionando una transferencia de datos más rápida y confiable.

34. ¿Qué es MIME?

- MIME (Multipurpose Internet Mail Extensions) es un estándar que extiende el formato de los mensajes de correo electrónico para soportar diferentes tipos de contenido, como texto en múltiples formatos, imágenes, audio y video. También se utiliza en HTTP para indicar el tipo de contenido de los archivos enviados por el servidor al cliente, ayudando a los navegadores a manejar correctamente los datos.

35. Explique cómo es una solicitud HTTP.

- Una solicitud HTTP se compone de varias partes:
    - **Línea de solicitud**: Esta línea contiene tres componentes principales:
        - **Método HTTP**: Indica la acción a realizar. Los métodos comunes incluyen GET (solicitar un recurso), POST (enviar datos al servidor), PUT (actualizar un recurso) y DELETE (eliminar un recurso).
        - **URL**: La dirección del recurso solicitado.
        - **Versión del protocolo HTTP**: Indica la versión del protocolo que el cliente está utilizando, como HTTP/1.1 o HTTP/2.
    - **Encabezados**: Proporcionan información adicional sobre la solicitud. Incluyen pares clave-valor que informan al servidor sobre el cliente y la solicitud. Ejemplos de encabezados son Host, User-Agent, Accept, Content-Type, y Authorization.
    - **Cuerpo (opcional)**: Contiene los datos que el cliente envía al servidor, especialmente en solicitudes como POST o PUT. Por ejemplo, en un formulario web enviado por POST, el cuerpo de la solicitud contendrá los datos del formulario.

36. ¿Cuáles son los métodos de HTTP 1? Enumere y describa el funcionamiento de cada uno.

- Los métodos principales de HTTP/1.1 son:
    - **GET**: Solicita un recurso del servidor. Es un método seguro y idempotente, lo que significa que no debe alterar el estado del recurso en el servidor y puede repetirse sin efectos adicionales.
    - **POST**: Envía datos al servidor para crear o actualizar un recurso. A diferencia de GET, POST no es idempotente, lo que significa que múltiples solicitudes POST pueden tener efectos diferentes.

- **PUT**: Envía datos al servidor para reemplazar un recurso existente. Es un método idempotente, por lo que repetir la misma solicitud PUT no debería cambiar el resultado.
- **DELETE**: Solicita la eliminación de un recurso en el servidor. Este método es idempotente, y repetir la misma solicitud DELETE no debería tener efectos adicionales si el recurso ya ha sido eliminado.
- **HEAD**: Similar a GET, pero solo solicita los encabezados del recurso sin el cuerpo. Se utiliza para obtener metainformación sobre el recurso.
- **OPTIONS**: Solicita información sobre los métodos HTTP soportados por el servidor para un recurso específico. Es útil para conocer las capacidades del servidor.
- **TRACE**: Devuelve la solicitud recibida por el servidor para propósitos de diagnóstico. Este método no es comúnmente utilizado y puede ser un riesgo de seguridad si no se maneja adecuadamente.
- **CONNECT**: Establece un túnel de comunicación con el servidor, comúnmente utilizado para establecer conexiones a través de un proxy HTTP.