IMPLANTACIÓN DE APLICACIONES WEB EN ENTORNOS INTERNET, INTRANET Y EXTRANET.

INTERNET.

- Conjunto descentralizado de redes de comunicaciones interconectadas, que utilizan la familia de protocolos TCP/IP, lo cual garantiza que las redes físicas heterogéneas que la componen constituyen una red lógica única de alcance mundial. (Wikipedia).

HISTORIA.

CONCEPTOS DE INTERNET.

URL-URI-URN.

- Cadenas de texto utilizadas como identificadores o direcciones de un recurso disponible en Internet.
- URI = URL + URN.
- URI engloba a URL y URN, es decir, ambas son un tipo de URI.
- **URI.**
    - Identificador Uniforme de Recursos - Uniform Resource Identifier.
    - Cadena que identifica un recurso en Internet. Puede ser un nombre, un localizador o ambas cosas.
- **URL**.
    - Localizador Uniforme de Recursos - Uniform Resource Locator.
    - Sistema de localización o direccionamiento de los documentos web.
    - Identifica un recurso porque es una URI, pero permite localizarlo.
    - Ejemplos:
        - http://www.unsitio.com/index.html
        - https://www.otrositio.com/welcome.html
- **URN**.
    - Nombre Uniforme de Recursos - Uniform Resource Name.
    - Nombre único y público de un recurso en Internet.
    - URN no da información sobre la localización del recurso.
    - Es una URI porque identifica un recurso, pero no permite localizarlo.
    - urn: isbn: 0238943023-23098230
- **Estructura de una URI:**
    - Protocolo o esquema://Tipo de servidor o máquina.nombre del servidor.dominio/directorio/archivo.html?parámetro o #fragmento

PROTOCOLOS DE INTERNET.

- **Concepto.**
    - Los **protocolos de red** son un conjunto de reglas que gobiernan la comunicación entre dispositivos que están conectados a una red.
- **Orientados a conexión**:
    - Antes de transferir datos, se establece una conexión entre las entidades participantes en la comunicación.

- **TCP**.
    - Protocolo de control de transmisión - Transmission Control Protocol.
    - Basado en el intercambio de datagramas.
    - Garantiza que los datos son enviados destino sin errores y en el orden en que se transmitieron.
    - Utiliza puertos para entregar los datos dentro de una máquina a la aplicación que los solicito.
    - El destino se encarga de reorganizar los paquetes y entregarlos a la aplicación correspondiente.
    - Navegadores, clientes FTP y muchos protocolos de aplicación lo utilizan.
- **No orientados a conexión**:
    - Se transfieren los datos sin establecerse una conexión entre las entidades participantes en la comunicación.
    - Un punto de la comunicación transmite los datos sin necesidad de asegurarse de que el receptor esté disponible y listo para recibirlos.
    - **IP**.
        - Protocolo de Internet – Internet Protocol.
        - Pilar básico de Internet, ya que permite el desarrollo y transporte de paquetes de datos denominados datagramas IP, aunque su recepción no está asegurada.
        - Forma parte del protocolo TCP/IP.
        - Responsable del direccionamiento, encaminamiento o enrutamiento de los datagramas.
        - No fiable, intentará hacer llegar a su destino los datagramas, aunque tengan que viajar por diferentes caminos.
        - Datagrama:
            - Unidad de transferencia de información en las redes IP.
            - Es un paquete de datos.
            - Estructura:
                - Cabecera o encabezado con información de control y sobre el transporte del datagrama.
                - Datos que se transmiten.
    - **UDP.**
        - Protocolo de datagrama de usuario - User Datagram Protocol.
        - Protocolo ligero de transporte de datos que funciona sobre IP.
        - Basado en el intercambio de datagramas.
        - En caso de detectar un error, UDP no entrega el datagrama al destino, sino que lo descarta, de modo que no garantiza la entrega de los datagramas, por lo que es un protocolo poco fiable.
        - Su simplicidad y rapidez lo hace ideal para aplicaciones que requieren pocos retrasos como aplicaciones en tiempo real (video en streaming)
        - Ideal para sistemas tan limitados que no pueden implementar un sistema complejo como el TCP.
        - Como TCP, también utiliza puertos.

PUERTOS.

- Interfaz de comunicación que permite la transmisión de datos en ambos sentidos.
- Vía de comunicación usada por las aplicaciones y servicios para transferir datos y comunicarse.
- La interfaz puede ser lógica o física.
    - **Puerto físico**:
        - Se usan para comunicar equipos físicos.
    - **Puerto lógico**:
        - Se usan para envío y recepción de información y para establecer la comunicación entre dispositivos.
        - Rango de puertos lógicos:
            - 1 al 65.534.
        - Algunos ejemplos:
            - http: 80
            - telnet: 23
            - pop3: 110.
            - SSh: 22.
            - ftp: 21
- Diversos programas TCP/IP pueden ejecutarse simultáneamente en Internet (por ejemplo, pueden abrirse diferentes navegadores de manera simultánea o navegar por páginas HTML mientras se descarga un archivo de un FTP). Cada uno de estos programas funciona con un protocolo. A veces el equipo debe poder distinguir las diferentes fuentes de datos.
- Por lo tanto, para facilitar este proceso, a cada una de estas aplicaciones puede serle asignada una dirección única en equipo, codificada en 16 bits.
- Número de puerto: número de 16 bits (de 1 a 65535) que sirve para identificar el proceso al que entregar el mensaje dentro de la máquina.

- **Socket.**
    - Dirección única resultante de la combinación de una *dirección IP + puerto:*

0-255.0-255.0-255.0-255:1-65535

- De esta manera, la dirección IP sirve para identificar de manera única un equipo en la red mientras que el número de puerto especifica la aplicación a la que se dirigen los datos. Así, cuando el equipo recibe información que va dirigida a un puerto, los datos se envían a la aplicación relacionada.

PROTOCOLOS DE APLICACIÓN.

- Protocolos para obtener recursos concretos en Internet.
- Tipos más comunes:
- **http**:
    - Protocolo de Transferencia de Hipertexto - Hipertext Transfer Protocol.

    - Transfiere archivos en formato HTML entre un cliente (navegador) y un servidor web localizado mediante una dirección URL.
- **https**:
    - Protocolo de Transferencia de Hipertexto Seguro - Hipertext Transfer Protocol Secure.
    - Transfiere archivos en formato HTML entre un cliente (navegador) y un servidor web localizado mediante una dirección URL de forma segura a través de un canal exclusivo y encriptando la información.
- **ftp**:
    - Protocolo de Transferencia de Archivos – File Transfer Protocol.
    - Transfiere archivos a través de una red TCP/IP según los siguientes objetivos:
        - Permite que equipos remotos compartan archivos.
        - Permite transferencias de datos más eficaces.
        - Permite independencia ente sistemas de archivos del cliente y el servidor.
- **telnet**:
    - Red de telecomunicaciones - Telecomunication Network.
    - Protocolo de conexión remota que permite conectar terminales y aplicaciones en internet.
    - Permite el manejo de otra máquina de forma remota.
- **file**:
    - Protocolo de archivos locales.
- **mailto**.
    - Protocolo de envío y recepción de correo electrónico.
    - Sirve para especificar la cuenta de correo del usuario destinatario.
    - Sintaxis:
        - mailto: nombre de usuario@servidor
- **smtp**.
    - Protocolo simple de transferencia de correo.
    - Permite el envío de correo.
    - Protocolo de correo saliente (Cliente a Internet).
- **pop3.**
    - Protocolo de oficina de correos.
    - Protocolo de recepción de correo (Servidor a Cliente).
    - Los correos se borran del servidor tras su envío al cliente.
- **imap.**
    - Protocolo de recepción de correo (Servidor a Cliente).
    - Los correos se mantienen en el servidor, enviándose una copia al cliente.
    - Permite administrar varias cuentas de correo electrónico.
    - Alternativo al pop3.
- **irc.**
    - Internet Relay Chat.
    - Protocolo de comunicación en tiempo real mediante mensajería.
- **voip.**

        - Telefonía vía internet.
    - **p2p**
        - Peer to Peer.
        - Compartir archivos de cliente a cliente.
    - **ssh.**
        - Protocolo de seguridad.
        - Usado para iniciar una sesión remota en otro ordenador creando una conexión segura.
        - Tunelización.
    - **ssl.**
        - Protocolo de seguridad.
        - Usado para el cifrado o encriptación de los datos que se transmitirán entre ordenadores.

MODELO OSI.

- Modelo de interconexión de sistemas abiertos - Open Systems Interconnection.
- Modelo de referencia para los protocolos de la red.
- Describe una estructura con siete niveles o capas para las operaciones de red. Cada nivel tiene asociados uno o más protocolos.
- Capas:
    - **Física (1).**
        - Define las características del hardware de red.
    - **Enlace de datos (2).**
        - Gestiona la transferencia de datos.
    - **Red (3).**
        - Gestiona el encaminamiento o enrutamiento de datos y su transferencia entre redes.
        - El objetivo de esta capa es que los datos lleguen al destino.
        - Protocolos IPv4 e IPv6.
    - **Transporte (4).**
        - Gestiona la transferencia o transporte de los datos y también, garantiza la integridad de éstos.
        - Protocolos TCP o UDP.
    - **Sesión (5).**
        - Gestiona la conexión entre 2 ordenadores manteniendo ésta mientras se transfieren los datos.
    - **Presentación (6).**
        - Gestiona la representación de la información.
        - Permite cifrar y comprimir los datos.
    - **Aplicación (7).**
        - Servicios de red a las aplicaciones.
        - Define los protocolos que utilizan las aplicaciones para intercambiar datos.
        - Protocolos ftp, smtp, pop3, ldap, …

DIRECCIONES IP - DIRECCIONES DE INTERNET.

- Identifican recursos, equipos o dispositivos en red.
- **Tipos:**
    - **Direcciones IPv4.**
        - El protocolo IPv4 emplea direcciones numéricas compuestas por un conjunto de 4 números enteros ente 0 y 255 separados por un punto.
        - Cada número representa a un byte u 8 bits.
        - Formato:
            - 0-255.0-255.0-255.0-255
            - 4 bytes o 32 bits (2^{32}), que dan lugar a 4.294.967.296.
        - Ejemplos:
            - http://210.45.23.1
            - ftp://145.99.0.0
            - http://127.0.0.1
    - **Direcciones IPv6.**
        - Diseñadas para reemplazar a las ipv4.
        - El protocolo IPv6 emplea direcciones numéricas compuestas por un conjunto de 8 grupos de 4 números hexadecimales separados por 2 puntos.
        - Permite 2^{128} direcciones distintas porque se emplean 16 bytes para codificarlas y dan 340.282.366.920.938.463.463.374.607.431.768.211.456.
        - Formato:
            - xxxx:xxxx:xxxx:xxxx:xxxx:xxxx:xxxx:xxxx
        - Ejemplo:
            - 8000:a23e:0534:aae3:0000:45d3:0d22:76a3
        - Optimización o simplificación direcciones ipv6:
            - Primer cero de un grupo puede eliminarse.
            - Grupos de 4 ceros pueden sustituirse por 2 puntos.
                - 8000:a23e:534:aae3::45d3:d22:76a3
                - ::1

- **Máscara de red.**
    - Al igual que las direcciones Ipv4, también tienen 32 bits o 4 bytes.
    - Una dirección IP puede tener una parte fija y otra variable, para saber cuál es cual se usa una máscara de subred.
    - La parte fija identifica una red y la variable a los dispositivos conectados a ella.
    - Una máscara de red en una dirección IP, casi siempre compuesta por los valores 0 y 255.
    - Con ellos, la máscara identifica la parte fija de la variable.
    - Expresada en binario, utiliza unos y ceros para indicar qué bits de la dirección son de red (unos) y cuáles son de host (ceros).
    - Ejemplo de máscara.
        - 255.255.255.0 (En rojo la parte fija, en verde la variable).

        - Si usamos esta máscara y tenemos conectados un ordenador, un portátil y una Tablet que están identificados con las direcciones IP 192.168.1.2, 192.168.1.3 y 192.168.1.4 respectivamente, la parte fija que identifica nuestra red es 192.168.1, mientras que el cuarto número, que es variable, identifica cada dispositivo dentro de nuestra red.
    - También permite distinguir direcciones IP a simple vista parecidas:
        - Las direcciones 192.168.1.2 con máscara 255.255.255.0 y 192.168.1.2 con máscara 255.255.254.0
    - **Máscara expresada en binario.**
        - 255.255.255.0 equivale 11111111.11111111.11111111.0000000
    - **Prefijos de red**.
        - Cantidad de bits en la dirección que conforma la porción de red.
        - Es otra forma de expresar una máscara de red.
        - Se añade como numero decimal a una dirección IP.
        - Sintaxis:
            - Dirección IPv4/Prefijo de red
        - Ejemplo:
            - 192.168.4.89/24
            - Como una dirección IP tiene 32 bits, el ejemplo anterior indica que de esos 32, 24 se corresponden con la parte de la red y los 8 restantes con los hosts.

- **Dirección de red.**
    - Dirección que identifica a una red o a una subred.
    - Es la primera dirección IP o dirección más baja de una subred.
    - Esta dirección tiene ceros en la parte correspondiente al host.
    - Ejemplo:
        - Si tenemos una subred con 6 direcciones que van desde la 192.168.0.40 a 192.168.0.45, 192.168.0.40 será la dirección de red o de la subred.

- **Dirección de Broadcast o multidifusión**.
    - Última dirección IP o dirección más alta de una subred.
    - Dirección especial que permite la comunicación con todos los nodos o host de una subred.
    - Permite la transmisión de datos a una multitud de nodos receptores de la misma subred y de forma simultánea.
    - Esta dirección tiene unos en la parte correspondiente al host.
    - Ejemplo:
        - Si tenemos una subred con 6 direcciones que van desde la 192.168.0.40 a 192.168.0.45, 192.168.0.45 será la dirección de broadcast.
- **Direcciones host**.
    - Cada dispositivo conectado a una red o subred necesita una dirección única para ser identificado y así poder para enviar paquetes a dicho host.
    - En IPv4 los valores que se asignan a cada host están entre el valor de la dirección de red y la dirección de broadcast.

- Ejemplo:
  - Si en la subred con direcciones que van desde la 192.168.0.40 a 192.168.0.45, tenemos conectados 2 ordenadores, un portátil y una impresora, las direcciones IP de éstos host serán 192.168.0.41, 192.168.0.42, 192.168.0.43 y 192.168.0.44

- **Puerta de enlace.**
  - Intermediario que sirve de enlace entre dos redes informáticas.
  - Así, los dispositivos de una red se comunican con los de otra red.
  - Si un dispositivo solicita una página web de Internet, la solicitud atravesará la puerta de enlace para de salir de la red local y llegar a Internet.
  - Identificada mediante una dirección IP.
    - Las puertas de enlaces predeterminadas normalmente con las direcciones 192.168.1.1 o 192.168.0.1**.**

- **IP pública**.
  - Dirección que se asigna a cualquier dispositivo que se conecte directamente a Internet, como un router a nivel doméstico o un servidor que aloja sitios web.
  - Son visibles desde Internet.
- **IP privada.**
  - Dirección fija que se asigna a cada dispositivo conectado a una red privada o doméstica.
  - Dirección IP que el router asigna a cada ordenador, Tablet, televisión inteligente, videoconsola o cualquier otro dispositivo conectado a él.
  - De esta forma, cada dispositivo conectado al router tiene su propia dirección IP privada, pero todos la misma pública.
  - No son accesibles o visibles desde Internet.
  - No cambian, a no ser que se modifiquen manualmente.
  - Conocer nuestra ipv4 e ipv6 privadas.
    - Acceder a CMD o símbolo de sistema como administrador.
    - Teclear el comando *ipconfig* y pulsar Enter.
- **Clases de direcciones IPv4**.
  - **Clase A**.
    - Red de gran tamaño.
    - Direcciones comprendidas entre 0.0.0.0 y 127.255.255.255.
    - Dirección de Red:
      - 0.0.0.0
    - Dirección de Broadcast:
      - 127.255.255.255
    - Máscara de red:
      - 255.0.0.0
  - **Clase B**.
    - Red de tamaño mediano.
    - Direcciones comprendidas entre 128.0.0.0 - 191.255.255.255.
    - Dirección de Red:
      - 128.0.0.0

        - Dirección de Broadcast:
            - 191.255.255.255
        - Máscara de red:
            - 255.255.0.0
    - **Clase C**.
        - Red de tamaño pequeño.
        - Direcciones comprendidas entre 192.0.0.0 - 223.255.255.255.
        - Dirección de Red:
            - 192.0.0.0
        - Dirección de Broadcast:
            - 223.255.255.255
        - Máscara de red:
            - 255.255.255.0
    - **Clase D**.
        - Uso multicast, para transmitir información a un grupo de receptores interesados que estén configurados para tal fin.
        - Direcciones comprendidas entre 224.0.0.0 - 239.255.255.255.
        - Dirección de Red:
            - 224.0.0.0
        - Dirección de Broadcast:
            - 239.255.255.255
        - Máscara de red:
            - 255.255.255.255 o Sin definir.
    - **Clase E**.
        - Uso experimental.
        - Direcciones comprendidas entre 240.0.0.0 - 255.255.255.255.
        - Dirección de Red:
            - 240.0.0.0
        - Dirección de Broadcast:
            - 255.255.255.255
        - Máscara de red:
            - 255.255.255.255 o Sin definir.

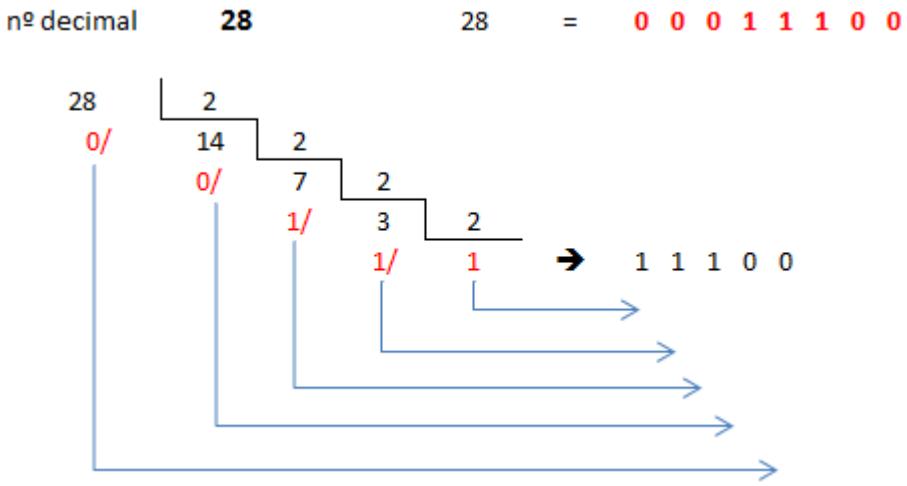
SUBNETTING.

- Técnica de subdividir una red IP física en varias redes lógicas más pequeñas**.**
- De esta manera se aprovecha el espacio de direcciones que de otra manera podría perderse.
- Cada subred creada funcionará como una red individual, aunque sigan formando parte de una misma red principal y un mismo dominio.
- Ventajas:
    - **Ajuste del rango de direcciones**.
        - Permite decidir qué tamaños deben tener las subredes.
    - **Conexión más rápida**.

        - Los paquetes de datos llegan directamente al receptor y no se transmiten por toda la red.
    - **Mejor organización lógica**.
        - En una organización o en una empresa se puede segmentar una red por criterios departamentales, locales, geográficos, etc.
    - **Mayor seguridad**.
        - Ante un ataque, se puede aislar una subred del resto.
- **Conversiones**.
    - **Binario a decimal**.
        - Para hacer la conversión de binario a decimal, hay que sumar el valor decimal correspondiente a la posición que tenga un 1 en el número binario.
        - Como un byte tiene 8 bit cada posición de izquierda a derecha vale 128, 64, 32 16,8 4, 2 y 1 respetivamente, siendo el valor decimal final, la suma de todos los valores de las posiciones donde coincida un uno.
        - Ejemplo:

Valor decimal	128	64	32	16	8	4	2	1
Binario	1	1	0	0	1	0	1	0
Resultado	128+	64+	0	0	8+	0	2+	0

        - Valor decimal final: 128 + 64 + 8 + 2 = 202
        - El byte 11001010 equivale al valor decimal 202.
    - Otros ejemplos**:**
        - 10001 = 00010001 = 16 +1 = 17
        - 01100111 = 64 + 32 + 4 + 2 + 1 = 103
        - 0000110010101110 = 2048 + 1024 + 128 + 32 + 8 + 4 + 2 = 3246.
    - **Decimal a binario.**
        - Se realizan sucesivas divisiones entre dos y se utilizan, tanto los restos década división, como el último cociente.
        - Los resultados se anotan en orden inverso (derecha a izquierda).
        - Si es necesario completar el byte hasta alcanzar los ocho dígitos, se añaden 0 por la izquierda.
        - Ejemplo:



- Para un número decimal mayor, hace falta un segundo byte, o más bytes.
- En un segundo byte se continua la secuencia de valores según posición, siendo un valor el doble que el anterior.
- Al igual que con un solo byte, el valor decimal resultante será la suma de los valores posicionales donde haya un 1.

Valor decimal	32.768	16.384	8.192	4.096	2.048	1.024	512	256

- Ejemplo:
  - El valor binario 1011011000 equivale al número decimal 728.
  - La parte azul corresponde al primer byte.
  - La parte roja al segundo.
  - La suma de valores posicionales:
    - 512 + 128 + 64 + 16 + 8 = 728

Dirección de red /subred a la que pertenece una IP.

- Para saber a qué red o subred pertenece un host identificado con una dirección IP se realiza una operación o comparación lógica AND.
- Se compara la dirección IP con su máscara bit a bit.
- El resultado de una comparación es 1 si los dos bits comparados son 1 y si no, el resultado es 0.
- Ejemplo:
  - IP de un Host: 192.168.0.42
  - Mascara de red: 255.255.255.0
  - Dirección de red: 192.168.0.0
- Operación AND:
  - 11000000 10101000 00000000 00101010 IP Host.
  - 11111111 11111111 11111111 00000000 Mascara.
  - 11000000 10101000 00000000 00000000 Dirección de red.

Dirección de broadcast asociada a una red / subred a la que pertenece una dirección IP.

- Para conocer la dirección de broadcast de una red o subred a la que pertenece un host identificado con una dirección IP se realiza una operación o comparación lógica OR.
- Se compara la dirección IP con su máscara bit a bit.
- Los valores de la máscara se invierten.
- El resultado de una comparación es 1 al menos uno de los comparados es 1. Si ambos bits son cero, el resultado es 0.
- Ejemplo:
  - IP de un Host: 192.168.0.42
  - Mascara de red: 255.255.255.0
  - Dirección de broadcast: 192.168.0.255
- Operación OR:
  - 11000000 10101000 00000000 00101010 IP Host.
  - 00000000 00000000 00000000 11111111 Mascara invertida.
  - 11000000 10101000 00000000 11111111 Dirección de broadcast.

Subredes.

- Para crear subredes, hay que coger bits de la porción de hosts y reasignarlos a la porción de red.
- Numero de subredes creadas: 2^n.
- Si reasignamos 4 bits, el total de subredes creadas será de $2^4 = 16$ subredes.
- Ejemplo:
  - Una red única con dirección de red 192.168.168.0, es de tipo C y tiene los siguientes valores binarios:
    - 11000000 10101000 10101000 00000000 Dirección de red.
    - 11111111 11111111 11111111 00000000 Máscara.
  - Para crear varias subredes a partir de esta red única, se añaden a la máscara unos en la parte de los hosts, empezando por la izquierda del byte.
  - 11111111 11111111 11111111 11100000. Nueva Máscara binario.
  - 255.255.255.224. Nueva máscara en decimal.
  - Se han creado $2^3 = 8$ subredes. (000, 001, 010, 011, 100, 101, 110, 111) cuyas direcciones de red son:
    - 11000000 10101000 10101000 00000000 ->192.168.168.0
    - 11000000 10101000 10101000 00100000 ->192.168.168.32
    - 11000000 10101000 10101000 01000000 ->192.168.168.64
    - 11000000 10101000 10101000 01100000 ->192.168.168.96
    - 11000000 10101000 10101000 10000000 ->192.168.168.128
    - 11000000 10101000 10101000 10100000 ->192.168.168.160
    - 11000000 10101000 10101000 11000000 ->192.168.168.192
    - 11000000 10101000 10101000 11100000 ->192.168.168.224
  - Total de host en cada subred $2^5 = 32$
  - Total de host útiles cada subred: 32 - 2 = 30, ya que cada subred tiene una IP de subred y otra de broadcast.

Crear subredes con un determinado número de hosts.

- Si en lugar de conocer el número de bits de hosts que se convertirán en bit de red, conocemos el número de host que debe tener cada subred el proceso es el siguiente:
    - Sumar los valores posicionales de derecha izquierda hasta encontrar el resultado igual o mayor más cercano al número de host solicitados.
    - Los ceros más a la izquierda de la posición encontrada en el punto anterior se convierten en unos.
- De esta forma se obtiene una nueva mascara, a partir de la cual se pueden conocer las direcciones de cada subred, de broadcast, etc.
- Ejemplo:
    - A partir de la red identificada como 192.168.19.0 crear varias subredes que contengan cada un 8 host.
    - Mascara por defecto:
        - 225.225.255.0
    - Valores posicionales sumados de derecha a izquierda:
        - 0 + 0 + 0 + 8 = 8
    - Los ceros a la izquierda de la posición cuarta, que dio el número exacto o más cercano mayor se convierten en unos (1), de modo que la máscara adaptada queda como:
        - 11111111.11111111.11111111.11110000
    - A partir de aquel se pude conocer la IP de red, broadcast, número de host y número de bits de red y cada IP de cada host en cada subred.

Calcular direcciones de red, broadcast y host.

- Conocida una IP, saber a qué red pertenece (cuál es su dirección de red), cuál es su dirección de broadcast y cuáles son las direcciones del primer y último host.
- Pasos:
    - **1. Conocer el número de bits para la red y el número de bits para los hosts.**
        - Esto se puede averiguar a través de la máscara de red.
        - Ésta puede estar expresada como como dirección IP o como valor de decimal como prefijo incluido en la dirección IP que se está consultando.
        - Con valor decimal como prefijo:
            - Restar los 32 bit de la dirección IP al valor del prefijo.
            - Ejemplo:
                - IP para consultar: 192.168.20.45/20
                - Mascara de subred = 20.
                - Dirección IP = 32.
                - Bits utilizados para host: 32- 20 = 12.
                - Numero de host que permite: 2^{12} = 4.096.
                - Numero de host útiles o disponibles: 4.094 (4.096 – 2, se quita la dirección de red y la de broadcast).

- **2. Calcular las direcciones.**
  - Para ello hay que pasar IP a binario:
    - 192.168.20.45 = 11000000 10101000 00010100 00101101 (Rojo bits de red, azul, bits de hosts).
  - **Dirección de Red**.
    - Todos los bits de hosts se ponen a cero.
    - Ejemplo:
      - 11000000 10101000 00010100 00101101 -> 11000000 10101000 00010000 00000000.
      - IP de Red: 11000000 10101000 00010000 00000000.
  - **Dirección de Broadcast**.
    - Todos los bits de hosts se ponen a uno.
    - Ejemplo:
      - 11000000 10101000 00010100 00101101 -> 11000000 10101000 00011111 11111111.
      - IP de Broadcast: 11000000 10101000 00011111 11111111.
  - **Dirección del primer host**.
    - Dirección siguiente a la de red, de modo que se pone el número 1 en binario en la parte del host correspondiente. En este caso, se cambia el último cero por un uno.
    - Ejemplo:
      - 11000000 10101000 00010100 00101101 -> 11000000 10101000 00010000 00000001.
      - IP del primer Host: 11000000 10101000 00010000 00000001.
  - **Dirección del último host**.
    - Dirección anterior a la de broadcast, de modo que se pone el número 0 en binario en la parte del host correspondiente. En este caso, se cambia el último uno por un cero.
    - Ejemplo:
      - 11000000 10101000 00010100 00101101 -> 11000000 10101000 00011111 11111110.
      - IP del último Host: 11000000 10101000 00011111 11111110.
  - **Dirección de un host cualquiera**.
    - Dirección siguiente a la de red en un numero de posición determinado, de modo que se pone ese número en binario empezando por el ultimo byte.
    - Ejemplo:
      - Host 3:
        - Un uno en última posición que vale 1.
        - Un uno en penúltima posición que vale 2.
        - 11000000 10101000 00010100 00101101 -> 11000000 10101000 00010000 00000011.
        - IP del Host 3: 11000000 10101000 00010000 00000011.
      - Host 42:
        - Un uno en penúltima posición que vale 2.
        - Un uno en cuarta posición que vale 8.
        - Un uno en sexta posición que vale 32.

        - 11000000 10101000 00010100 00101101 -> 11000000 10101000 00010000 00101010.
        - IP del Host 42: 11000000 10101000 00010000 00101010.
- **3. Pasar las direcciones de binario a decimal**
    - **Dirección de Red**.
        - IP de Red: 11000000 10101000 00010000 00000000.
        - 192.168.16.0
    - **Dirección de Broadcast**.
        - IP de Broadcast: 11000000 10101000 00011111 11111111.
        - 192.168.31.255
    - **Dirección del primer host**.
        - IP del primer Host: 11000000 10101000 00010000 00000001.
        - 192.168.16.1
    - **Dirección del último host**.
        - IP del último Host: 11000000 10101000 00011111 11111110.
        - 192.168.31.254
    - **Dirección de un host cualquiera**.
        - Ejemplo:
            - Host 3:
                - IP del Host 3: 11000000 10101000 00010000 00000011.
                - 192.168.16.3
            - Host 42:
                - IP del Host 42: 11000000 10101000 00010000 00101010.
                - 192.168.16.42

DOMINIOS Y ALOJAMIENTO.

Concepto.

- Dirección de un sitio web.
- Apunta a un sitio web que contiene un conjunto de páginas web.
- Denominación que se da a un sitio web.
- Nombre que debe ser único y exclusivo.
- El nombre del dominio debe registrarse pagando una cuota anual.
- Los navegadores acceden a los sitios web a través de una dirección IP (IPv4 o IPv6).

Propósito o ventajas de un dominio.

- Traducir las direcciones IP (abstracción) a términos más intuitivos, memorizables y fáciles de encontrar.
- Se pueden tener más sitios en Internet, ya que muchos dominios pueden compartir la misma IP del servidor donde están alojados.
- Mayor flexibilidad al cambiar un sitio de un alojamiento web a otro, ya que la IP puede cambiar, pero el dominio puede seguir siendo el mismo.

Categorías:

- **Dominio Principal.**

    - Dominio enlazado o que apunta a un sitio web y que está alojado en un servidor de alojamiento (hosting).
    - Se puede conseguir al contratar el hosting o conseguirlo sin hosting.
    - Permite el acceso a un sitio web.
    - Ejemplo:
        - misitio.com
- **Alias de dominio.**
    - Dominio que redirecciona o apunta a un dominio principal.
    - Ejemplo:
        - Dominio principal:
            - misitio.com
        - Alias de dominio:
            - misitio.es, misitio.org, misitio.info, misitio.biz, otrositio.com, sitio3.org
- **Dominio adicional:**
    - Dominio que tiene un contenido o configuración independiente y por tanto no es igual al principal.
    - En el hosting o servidor de alojamiento se crea una carpeta nueva con el nombre de dominio adicional.
    - Ejemplos:
        - Queen.com (Dominio principal del grupo).
        - QueenMusical.com (Dominio adicional del musical sobre Queen).
        - QueenDiscografia.net(Dominio adicional sobre la discografía de Queen).
        - Queen.es(Dominio adicional sobre relación de Queen con España (conciertos, fans,..)).

DNS.

- Domain Name System.
- Sistema de nombres de dominio.
- Administra el espacio de nombres de dominio.
- Servicio de resolución de nombres de dominio que traduce estos a sus correspondientes direcciones IP.

Ejemplos de direcciones de nombres de dominio.

- Ejemplo de URL con nombre de dominio**.**
    - http://www.misitio.com
    - http://www.misitio.es
- Ejemplo de URL con nombre de dominio, ruta de acceso y archivo.
    - http://www.tienda.com/jardineria/herramientas.html
- Páginas de inicio de un sitio Web.
    - Index.html, index.php, default.aspx, etc.

Organismos y protocolos que gestionan los nombres de dominio.

ICANN

- Internet Corporation for Assigned Names and Numbers.
- Corporación de Internet para la asignación de nombres y números.
- Organización sin ánimo de lucro que regula la concesión única de dominios de nivel superior (TLD) a personas u organizaciones, así como sus correspondientes direcciones IP.
- También coordina los sistemas de nombres de dominio (DNS), es decir, se ocupa de que a cada domino se le asigne una IP correcta.

RED.ES

- El registro de nombres de domino está sometido a las reglas de cada país.
- En España los gestiona RED.ES y otras entidades colaboradoras.

WHOIS.

- Protocolo TCP que permite conocer quién es el propietario de un nombre de dominio y su dirección IP.
- Existe una base de datos que almacena la titularidad del dominio y la fecha de expiración.
- Se pueden hacer consultas por comandos o aplicación con interfaces visuales vías páginas web. (dominios.es)
- Existen servicios de privacidad para evitar accesos a datos de contacto mediante WHOIS.
    - Servicios de privacidad con datos de contacto alternativo.
    - Servicios de representación con datos del proveedor del dominio.

Tipos de dominios.

- **TLD (Top Level Domain) - Dominio de Primer Nivel**.
    - También se denomina extensión de dominio.
    - Genéricos:
        - com, mil, edu, org, gov, net, nato
    - Territoriales o geográficos:
        - es, pt, br, uk, ve….
    - Nuevos genéricos:
        - biz, mobi, museum, info, tv, jobs, kids, xxx,….
    - Nuevos territoriales:
        - asia, lat, mad, cat, eu,…
- **SLD (Second Level Domain) - Dominio de Segundo Nivel**.
    - Nombre del sitio que puede coincidir con marcas comerciales, nombres de instituciones, organismos, personas o empresas, etc., …
    - Ejemplo:
        - misitio(SLD).com(TLD)
- **TrLD (Third Level Domain) - Dominio de Tercer Nivel.**
    - Combinación de un dominio genérico con uno geográfico.
        - misitio.com.es
        - tienda.co.uk

- **Subdominio.**
    - Subconjunto de un dominio.
    - Sintaxis:
        - subdominio.dominio de segundo nivel.extensión o dominio de primer nivel
    - Ejemplos:
        - tienda.com (dominio principal).
        - novedades.tienda.com (subdominio de tienda.com).
        - blog.tienda.com (subdominio de tienda.com).
        - marketing.tienda.com (subdominio de tienda.com).

Ciclo de vida de un dominio.

- Etapas por las que pasa un dominio desde su registro hasta su eliminación.
- Etapas:
    - **Disponible**.
        - El dominio está libre y puede registrarse.
    - **Registrado**.
        - Registro del dominio por un periodo de tiempo.
        - Normalmente de 1 a 10 años.
        - Tras completar registro se tramita el pago.
        - Nadie más puede registrarlo.
    - **Activo**.
        - Periodo de tiempo por el que se ha registrado y el domino está en uso.
    - **Renovación**.
        - Mientras esta activo y antes de llegar a la fecha de expiración, puede renovarse.
    - **Caducidad**.
        - Si no se renueva, caduca y deja de funcionar.
    - **Periodo de gracia.**
        - En este periodo puede renovarse por el precio habitual de renovación.
        - Entre 30 y 45 días.
        - No todos los tipos de dominio tienen el mismo periodo de gracia.
    - **Periodo de castigo**.
        - Si no se ha renovado durante el periodo de gracia se puede renovar a un precio mayor.
    - **Tasa adicional**.
        - Dura 30 días.
    - **Periodo de eliminación**.
        - Si se llega a este periodo ya no se puede renovar.
        - Se bloquea durante 5 días y luego se libera para que lo registre otro.

Consejos para elegir un dominio.

- Nombre fácil de escribir, verbalizar o deletrear.
- Nombre corto y sencillo.

- Fácil de recordar.
- Los dominios con antigüedad son mejor valorados por los buscadores.
- Usar palabras clave relevantes.
- Nombre del dominio que tenga relación con la actividad del sitio.
- Si se tiene una empresa, usar su nombre como dominio.
- Evitar guiones medios (-) y bajos (_) por inducir a confusión.
- Evitar las palabras raras y poco comunes.
- Evitar números en los dominios.
- Evitar la ñ, si se quieren las visitas desde otros países.
- Evitar las mayúsculas.
- Evitar exceso de consonantes o letras dobles.
- Evitar acrónimos y abreviaturas.
- Registra el dominio cuanto antes, valen poco y así se evita que alguien nos lo quite.
- Elige el dominio .com como primera opción. Es fácil de recordar y el sitio se posicionará globalmente.
- Si el sitio se va a centrar en un país, usar también su geográfico (es, pt, mx, pe, etc.).
- Registra todas las variantes posibles (.com, .es, .net, .org). Se evitan así sitios ajenos con el mismo nombre.
- Registrar, si es posible, también el plural, el singular, el femenino y el masculino del nombre del dominio.
- Para posicionarme mejoren un país, es mejor que el dominio esté alojado en el mismo lugar que el sitio web y la empresa.
- Para evitar problemas legales, asegurarse de que el registro se hace a nuestro nombre o empresa.
- Antes de registrar, pedir opinión a otras personas. Una idea genial, puede ser en realidad absurda.
- Se pueden crear dominios con herramientas como Domain Name Generator o Domain Typo Generator.
- Usar diccionarios de sinónimos para ver ideas.
- Asegurarse que el dominio no tenga otro significado en otro idioma o sea difícil de entender o escribir.
- No uses marcas registradas o nombres de otras empresas, aunque el dominio esté libre.
- No cambiar de dominio constantemente.
- Estar atento a la fecha de caducidad para no perderlo.
- Registrarlo por un periodo de tiempo largo.
- Registrar el dominio en una empresa segura y de confianza.

Características y prestaciones para considerar en un hosting.

Además del espacio para el sitio web en un servidor, medido en GB y mejor con discos SSD, los proveedores de alojamiento pueden ofrecer otros **servicios** como:

- Cantidad de memoria RAM para el rendimiento.
- Certificados SSL (https) gratuitos para una web segura.
    - Pueden ofrecerlos a través de programas como Encryption Everywhere o la Fundación Linux.

- Creación de bases de datos, lo que implica espacio suficiente y un SGBD incluido como MySQL, MariaDB y PostgreSQL.
- Correo.
    - Cuentas de correo profesionales para que los visitantes puedan ponerse en contacto.
    - Cuantas más mejor, con el nombre del dominio.
- Aplicaciones adicionales fáciles de instalar como CMS como WordPress, Joomla, Prestashop o WooComerce para gestionar una tienda.
- Herramientas para desarrolladores (PHP, Perl, Python, JavaScript, etc.).
- Acceso y cuentas FTP.
- Posibilidad de tener varios dominios según necesidades.
- Creación de subdominios para dividir una web en subsecciones, si ésta crece y es necesario.
- Aparcado (parking) de dominios o alias de dominio.
- Incluir un panel de control para gestionar el sitio de forma fácil e intuitiva.
- Ancho de banda amplio (Bandwidth).
    - Indica cómo de rápido pueden transferirse los datos.
    - Límite máximo de datos que se pueden transmitir al mismo tiempo
    - Cantidad de datos a transmitir por segundo, se mide en Mbit (mbps) o Gbit (gbps).
- Transferencia de datos.
    - Cantidad total de datos que se trasferirán en un tiempo determinado, normalmente un mes (tráfico mensual).
    - Volumen de datos generados al visitar una página e interactuar con sus elementos.
    - Incluye:
        - Elementos que carga la página en cada visita.
        - Datos transferidos vía FTP.
        - Uso de correo electrónico.
        - Otros.
    - Suele ser un valor mensual que se restea al acabar el mes (si se tienen 10 GB, sólo se pueden gastar 10GB).
    - Su control implica cuidar el tamaño de las imágenes, no subir videos, controlar los archivos adjuntos del correo electrónico, etc.
- Copia de seguridad local y externa, automática y programable.
- Migración fácil a otros servidores o a otros planes de alojamiento.
- Foros, FAQ's, video tutoriales, …
- Filtros de spam, antivirus, cortafuegos, ...
- Alta en buscadores.
- Soporte técnico y atención al cliente 24/7.

ENTORNOS WEB.

Las aplicaciones web se emplean en tres entornos informáticos diferentes:

- **Internet.**

    - Al contrario que otros servicios en línea controlados de forma centralizada, Internet tiene un diseño descentralizado.

- Cada ordenador (host) en Internet es independiente.
- Sus operadores pueden elegir qué servicio de Internet usar y que servicios locales quieren proporcionar al resto de la Internet.
- Existe una gran variedad de formas de acceder a la Internet, siendo el método común, el acceso a través de un Proveedor de Servicios de Internet - Internet Service Provider (ISP).

- **Intranet.**

  - Red de ordenadores basada en los protocolos de Internet (TCP/IP), que pertenece a una organización y que es accesible únicamente por los miembros de ésta.
  - Es una red privada.
  - Puede estar o no conectada a Internet.
  - Un sitio web en una Intranet es igual que cualquier otro sitio web, pero los cortafuegos lo protegen de accesos no autorizados.
  - Es decir, una Intranet se ubica detrás de un cortafuegos y sólo es accesible por las personas que forman parte de la organización propietaria de la Intranet.
  - Como Internet, se utilizan para distribuir y compartir información.
  - Amplio crecimiento porque son más económicas de montar y administrar, que otras redes privadas basadas en protocolos propietarios.
  - Ejemplos de usos dentro de una intranet:
    - Dentro de la una empresa, todos los empleados están conectados al mismo «switch» sin servicio de Internet, pero todas comparten documentos y dispositivos para su trabajo diario.
    - **Switch de red o conmutador.**
      - Dispositivo que permite conectar todos los equipos en una red (ordenadores, impresoras, servidores, etc.), de modo que a través de ellos se pueden compartir archivos, impresoras, acceso a Internet a través de un router, …
    - A través de un programa común los empleados de la organización pueden obtener:
      - Acceso a directorios internos: búsqueda de teléfonos, direcciones, agendas, programaciones, etc.
      - Acceso a las bases de datos de la empresa.
      - Publicación de documentos internos: informes económicos, listas de precios, publicaciones y manuales de producto, etc.
      - Distribución de aplicaciones.
    - Creación de aplicaciones para trabajo en equipo.
- **Extranet.**

  - Intranet a la que pueden acceder parcialmente personas autorizadas ajenas a la organización o empresa propietaria de la intranet.
  - Proporciona diferentes niveles de acceso a personas que se encuentran en el exterior de la organización.
  - Acceso mediante nombre de usuario y una contraseña, determinado la identidad del usuario, a que partes de la Extranet puede acceder.

- Suelen usar protocolos de seguridad como SSL (Secure Socket Layer), y redes privadas virtuales (VPN - Virtual Private Network).
- Ejemplo de extranet:
  - Son muy utilizadas por empresas que comparten información entre ellas, por ejemplo, transacciones comerciales entre un fabricante y el distribuidor de un producto, o entre un distribuidor y un comercio minorista.
  - Uso frecuente como medio de comunicación de una empresa con sus clientes, proveedores o socios.
  - Son la base del comercio electrónico entre empresas (Business To Business, B2B).

SERVIDORES.

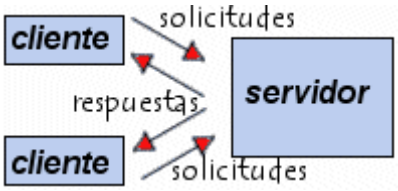
MODELO CLIENTE-SERVIDOR.

- Diversas aplicaciones se ejecutan en un entorno cliente/servidor.
- Esto significa que los **equipos clientes** (equipos que forman parte de una red), se conectan a un **servidor**, un equipo generalmente muy potente en materia de capacidad de entrada-salida, que proporciona servicios a los equipos clientes.
- Los servicios son utilizados por programas denominados **programas clientes** que se ejecutan en equipos clientes.
- Se usa el término "cliente" (cliente FTP, cliente de correo electrónico, etc.), cuando un programa se ha diseñado para ejecutarse en un equipo cliente, poder realizar peticiones a un servidor y ser capaz de procesar los datos recibidos de éste.
- El modelo cliente-servidor es recomendable para redes que requieran un alto grado de fiabilidad.
- Ventajas**:**
  - **Recursos centralizados.**
    - Como el servidor es el centro de la red, puede administrar los recursos que son comunes a todos los usuarios, por ejemplo: una base de datos centralizada evitaría problemas provocados por datos contradictorios y redundantes en bases de datos alojadas en los clientes.
  - **Seguridad mejorada.**
    - Mayores y de mejor calidad los sistemas de seguridad que se pueden usar en un servidor que los que se usan en un cliente.
    - Personal dedicado a asegurar la seguridad.
  - **Administración al nivel del servidor**
    - Como el servidor juega un papel más importante que los clientes en el modelo, estos últimos requieren menos administración.
  - **Escalabilidad.**
    - En esta arquitectura es posible aumentar las características o número de los servidores, así como, quitar o agregar clientes de forma fácil y rápida, sin afectar al funcionamiento de la red.
- Desventajas:
  - **Costo elevado.**
    - Debido a la complejidad técnica del servidor.

- **Eslabón débil.**
  - El servidor es el único eslabón débil en modelo cliente/servidor, debido a que toda la red está construida en torno a él.
  - Para minimizar este problema, los servidores son altamente tolerante a los fallos.

Funcionamiento del sistema cliente-servidor

- El cliente envía una solicitud al servidor mediante su dirección IP y el puerto, que está reservado para un servicio en particular que se ejecuta en el servidor.
- El servidor recibe la solicitud y responde con la dirección IP del equipo cliente y su puerto.



SERVIDOR.

Concepto.

- Es un ordenador y sus programas al servicio de otros ordenadores.
- Los clientes piden o solicitan, y el servidor proporciona los recursos o servicios.
- Un servidor deberá estar siempre encendido, ya que si se apaga dejará de dar servicio a los demás.
- Un servidor se puede definir desde el punto de vista del hardware o del software.
  - **Hardware**.
    - Ordenador o equipo físico sobre el que se ejecutan las aplicaciones.
    - Integrado en una red, contiene un sistema operativo y servidores basados en software.
    - Características físicas.
      - CPU gran cantidad de núcleos y una gran cantidad de memoria caché.
      - Varias CPUs montadas en la misma placa base.
      - CPUs diseñadas para estar funcionando continuamente sin apagarse.
      - Gran cantidad de memoria RAM ECC (Error Correcting Code).
      - Varios discos duros de gran velocidad (mecánico-magnéticos y SSD), incluso en configuraciones RAID.
      - Hardware de red con anchos de banda que permiten transmitir información de ida y vuelta a gran velocidad.
    - Rack o bastidor.

        - Estructura que permite contener varios servidores junto con otros dispositivos de red como routers, switch, etc., o equipos de audio o video, …
    - Procesadores para servidores:
        - Intel Xeon.
        - AMD Opteron y Epyc.
- **Software**.
    - Un servidor es un programa informático que proporciona servicios a otros programas o usuarios.

TIPOS DE SERVIDORES

SERVIDOR WEB.

- Lugar donde se almacena físicamente un sitio web.
- Puede alojar a la vez varios hostings que, a su vez, pueden tener varios sitios web distintos.

- **Servidores web más utilizados.**
    - Apache.
    - Nginx.
    - Microsoft IIS.
    - LiteSpeed.
    - Lighttpd.
    - Jigsaw
    - Sun Java System Web Server.
- **Hosting o alojamiento web.**
    - Espacio donde se aloja un sitio web para que cualquier persona pueda verlo a través del dominio que tenga.

	Hosting 1	Web 1	
	Hosting 2	Web 1	
SERVIDOR		Web 2	**CLIENTE**
			(Acceso a través de dominio con Internet)
	Hosting 3	Web 1	
		Web 2	
		Web 3	

- **Tipos de hosting.**
    - **Compartido o Shared Hosting.**
        - Varios hostings comparten el mismo servidor físico (procesador, memoria, disco duro, etc.).
        - Se comparten los recursos por varios clientes.

- Ventajas.
  - Más económico, ya que se comparten gastos.
  - El proveedor se encarga de la gestión y mantenimiento.
  - No hacen falta grandes conocimientos para gestionar un hosting (se suele hacer a través de un panel de control).
  - Si se necesita, se puede autorizar a paquetes hosting más potentes o con más prestaciones.
- Inconvenientes al compartir:
  - Menos recursos disponibles
  - Web más lenta, si no se gestionan bien los recursos.
  - Una web con muchas visitas puede ralentizar al resto del hosting.

- **Dedicado o Servidor dedicado**.
  - Todos los recursos de un servidor físico están asignados en exclusiva a un único hosting.
  - Uso en sitios con muchas visitas o empresariales que necesitan mucha potencia.
  - El proveedor se encarga de la administración y mantenimiento del servidor.
  - Atención al cliente, personalizada.
  - Se puede gestionar el hosting desde un panel de control (cpanel, WHM, Plesk).
  - Inconvenientes.
    - Si el cliente se encarga de la gestión, hay que tener conocimientos más avanzados.
    - Más caro.
    - Indispensable una buena configuración y optimización de los recursos del servidor.
    - Es más complicado actualizar el hosting, porque implica mejorar el servidor físico o cambiar a otro más potente.
- **Servidor Virtual Privado (VPS).**
  - Se contrata un servidor virtual privado cuando se necesita más potencia que la que ofrece un hosting compartido, pero que el servidor sea más económico que uno dedicado.
  - Muy popular.
  - Uso de tecnología de virtualización para proporcionar recursos dedicados (privados), en un servidor con múltiples usuarios.
  - VPS:
    - Partición virtual dentro de un servidor físico a la que se asignan recursos exclusivos (memoria, espacio en disco, capacidad de procesamientos, etc.), y se instala un Sistema Operativo.
  - VPS administrado:
    - El proveedor del hosting se encarga de su gestión.
  - VPS no administrado:
    - El cliente se encarga de su gestión.
  - Ventajas:

        - Más económico que un servidor dedicado.
        - Más recursos que el compartido, que no se comparten con otro hosting.
        - Si es no administrado, no hacen falta grandes conocimientos técnicos.
        - Se puede administrar / gestionar desde un panel de control.
        - Se puede actualizar a servidores privados más potentes cuando se necesiten más recursos.
    - Inconvenientes:
        - Más caro que compartido y más, si es administrado.
        - Dispone de menos recursos que uno dedicado.
        - Es importante que este bien optimizado y configurado, para aprovechar toda su potencia.
- **Hosting elástico – Elastic Sites (LVE).**
    - Se sitúa ente medias del compartido y VPS, siendo más potente que el primero y más barato que el segundo.
    - Parecido a VPS, en el que a cada usuario de un hosting se le asignan recursos del servidor físico encapsulados en un entorno virtualizado independientes (LVE).
    - Cada usuario tiene su hosting como en el compartido, pero con recursos asignados sólo para él.
    - Ventajas:
        - Más económico que un VPS.
        - Recursos asignados que no se comparten.
        - Como en el compartido, el proveedor se encarga de la gestión y mantenimiento.
        - No hacen falta conocimientos avanzados para gestionar el hosting, puede hacerse desde un panel de control.
        - Actualizable a opciones más potentes si se necesita.
    - Inconvenientes:
        - Más caro que el compartido.
        - No tiene administración personal.
- **Alojamiento en la nube - Cloud Hosting.**
    - Más avanzado.
    - Usado en aplicaciones potentes y sitios web con muchas visitas.
    - Más que un servidor físico, se contratan los recursos que se necesitan en cada momento, es decir, se contrata un servicio.
    - Usa una red de servidores físicos conectados entre sí, que se comportan como un superordenador muy potente con recursos ilimitados.
    - Si un servidor falla se seguirán usando los recursos del resto como si no hubiera pasado nada.
    - Puede ser administrado o no administrado.
    - Ventajas:
        - Recursos ilimitados, mayor potencia posible.

        - Más rentable para proyectos que necesitan mucha potencia
        - Se paga por lo que se consume encada momento (en uno dedicado, se paga se consuma lo que consuma).
        - Mayor fiabilidad ante fallos. (Tolerancia a fallos).
        - Atención al cliente más personalizada, estén o no administrados.
        - Se puede gestionar desde un panel de control.
    - Inconvenientes:
        - Caro.
        - Sea o no administrado, requiere conocimientos técnicos para su gestión.
        - Indispensable su correcta configuración y optimización para aprovechar toda su potencia.

SERVIDOR FTP.

- Uso para la transferencia segura de archivos entre ordenadores.
- Garantiza la seguridad de los archivos y control de su transferencia.
- Uso habitual:
    - Subir archivos de páginas web a los servidores web, archivos de imágenes, videos, hacer copias de seguridad o backup, descarga de archivos de todo tipo, etc.

SERVIDOR DE BASE DE DATOS.

- Son ordenadores utilizados para alojar bases de datos que luego serán usadas por uno o más clientes.
- Suministra servicios de gestión y consulta de las bases a los clientes que se conectan.
- Además, realizan otras tareas con los datos como su análisis, almacenamiento o manipulación.
- Muchos servidores realizan este tipo de funciones, porque casi toda aplicación suele necesitar una base de datos, no obstante, los hay dedicados en exclusiva a almacenar información en bases de datos (Un servidor de un banco).

SERVIDOR DE APLICACIONES

- Ejecuta aplicaciones y ofrece los servicios de éstas a los clientes conectados.
- La ejecución de las aplicaciones suele producir la conexión a otros tipos de servidores como, por ejemplo, los de bases de datos.
- Asociado al concepto de sistema distribuido, en el que un conjunto de programas informáticos usa recursos ubicados en varios nodos distintos con el objetivo de obtener un resultado compartido común.
- Ejemplos de servidores de aplicaciones:
    - BEA WebLogic.
    - IBM WebSphere.
    - Sun-Netscape IPlanet.
    - Sun One.
    - Oracle IAS.

- Borland AppServer.
- HP Bluestone.

SERVIDORES PROXY O SERVIDORES DE RED.

- Se utilizan para administrar una red de ordenadores, permitiendo el acceso o no a ésta por parte de los clientes.
- Suelen incluir sistema de protección cortafuegos.
- Pueden realizar muchas tareas como buscar virus, actuar como cortafuegos, acelerar la conexión guardando en la memoria caché y ocultar una dirección IP pública.

SERVIDOR DE IMPRESIÓN.

- Permite enviarle trabajos de impresión desde diferentes equipos.
- Típico en oficinas, se encarga de procesar los trabajos de impresión y enviarlos a las impresoras correspondientes.

SERVIDOR DE CORREO ELECTRÓNICO.

- Es un ordenador que funciona como una oficina de correo virtual.
- Transfiere y almacena mensajes de correo electrónico a través de una red.
- Suelen dividirse en dos distintos:
  - Uno para el correo entrante (POP3, IMAP).
  - Otro para el correo saliente (SMTP).
- Un dominio de Internet puede tener uno o más para gestionar sus cuentas de correo.
- Ejemplo:
  - Un dominio como www.misitio.com podría tener un servidor de correo que gestionaría cuentas como usuario@misitio.com, por ejemplo.

SERVIDOR DE DNS.

- Domain Name System o Sistema de Nombres de Dominio.
- Se encarga de traducir las direcciones IP a nombres de dominio fácilmente identificables y recordables.

SERVIDOR COPIA DE SEGURIDAD O BACKUP.

- Ofrecen un servicio de copia de seguridad remota en línea.
- Incluyen la instalación de un programa en el ordenador cliente que se encarga de realizar la copia con la frecuencia y en el periodo temporal establecido, además de seleccionar los datos a copiar, comprimirlos, cifrarlos y enviarlos al servidor remoto.

CLOUD COMPUTING - COMPUTACIÓN EN LA NUBE.

CONCEPTO DE NUBE:

- Físicamente son ordenadores (servidores) conectados a Internet para ofrecer distintos servicios a los clientes que los soliciten, incluido los discos que permiten un almacenamiento masivo y el hardware de red que permite dicha conexión.

- Red de servidores remotos de todo el mundo que están interconectados. Estos servidores están diseñados para almacenar y administrar datos, ejecutar aplicaciones o entregar contenido o servicios, como streaming de vídeos, correo web, software de ofimática o medios sociales. En lugar de acceder a archivos y datos desde un equipo personal o local, accede a ellos en línea desde cualquier dispositivo conectado a Internet, es decir, la información está disponible dondequiera que esté el usuario y siempre que la necesite.

CONCEPTO:

- Tecnología que permite acceso remoto a programas, datos y almacenamiento.
- Alternativa a la ejecución y almacenamiento local.

CARACTERÍSTICAS:

- **Multiplataforma - Ubicación.**
  - Acceso desde cualquier dispositivo (móvil, PC, Tablet,)
  - Acceso con cualquier sistema operativo (Android, Windows, Linux, iOS.)**.**
- **Concurrencia:**
  - Varios usuarios pueden usar las aplicaciones y los datos de forma simultánea y sincronizada.
- **Bajo Demanda.**
  - Capacidad de computación se adapta a las necesidades del usuario.
- **Flexibilidad-escalamiento.**
  - Recursos de hardware y software pueden dimensionarse según el usuario lo necesite.
- **Servicio Medido:**
  - El proveedor del servicio monitoriza y controla el uso de recursos.

VENTAJAS

- **Acceso desde cualquier sitio y desde varios dispositivos**:
  - Capacidad de trabajar desde cualquier sitio.
- **Todo el software está en un único sitio**:
  - No hay que instalar programas o aplicaciones a nivel local.
  - Necesario un navegador para acceder a la nube.
- **Colaboración**:
  - Los usuarios pueden compartir aplicaciones y documentos al mismo tiempo.
- **Ahorro en software y hardware.**
- **Ahorro en mantenimiento técnico**.
- **Escalabilidad**:
  - Capacidad de un sistema para crecer cuando sea necesario.
- **Seguridad**:
  - Seguridad mayor que a nivel local.
  - Uso de replicas, copias de seguridad, seguridad física y lógica.
  - Recuperación frente a desastres.
- **Actualizaciones automáticas de software**.

INCONVENIENTES

Seguridad y Privacidad.

- Almacenamiento fuera del equipo local implica dejar de tener control sobre los datos.
- Datos almacenados en el soporte físico de terceros.
- Riesgo al confiar en un tercero.
- Riesgo de ataques e intrusión.

Falta Conexión o acceso Internet.

- Problemas de cobertura legal.
- Servidores al estar en cualquier lugar del mundo, si hay problemas, ¿qué ley protege al usuario?

Conflictos de propiedad intelectual.

- Al no estar la información en poder del usuario pueden surgir problemas sobre quien es el propietario real.

Dependencia del proveedor del servicio.

- Tanto aplicaciones como datos están centralizados en sus servidores hay una dependencia de éste.

Falta mecanismo de escalabilidad.

- Al aumentar el número de usuarios se sobrecargan los servidores y si no hay un plan de crecimiento habrá un deterioro en la calidad del servicio.

CONTRATACIÓN DEL SERVICIO.

El servicio de acceso a la nube suministrado por el proveedor puede ser:

- **Gratuito.**
- **De pago.** (Existiendo varios sistemas de pago):
  - **Suscripción**
    - Precio predefinido.
    - Tarifa plana.
  - **Pago por uso.**

CONSEJOS USO NUBE.

- Hacer copias de seguridad a nivel local.
- Respetar la privacidad propia.
- Tener presente siempre la seguridad y cambiar las contraseñas con frecuencia.

SERVICIOS EN LA NUBE.

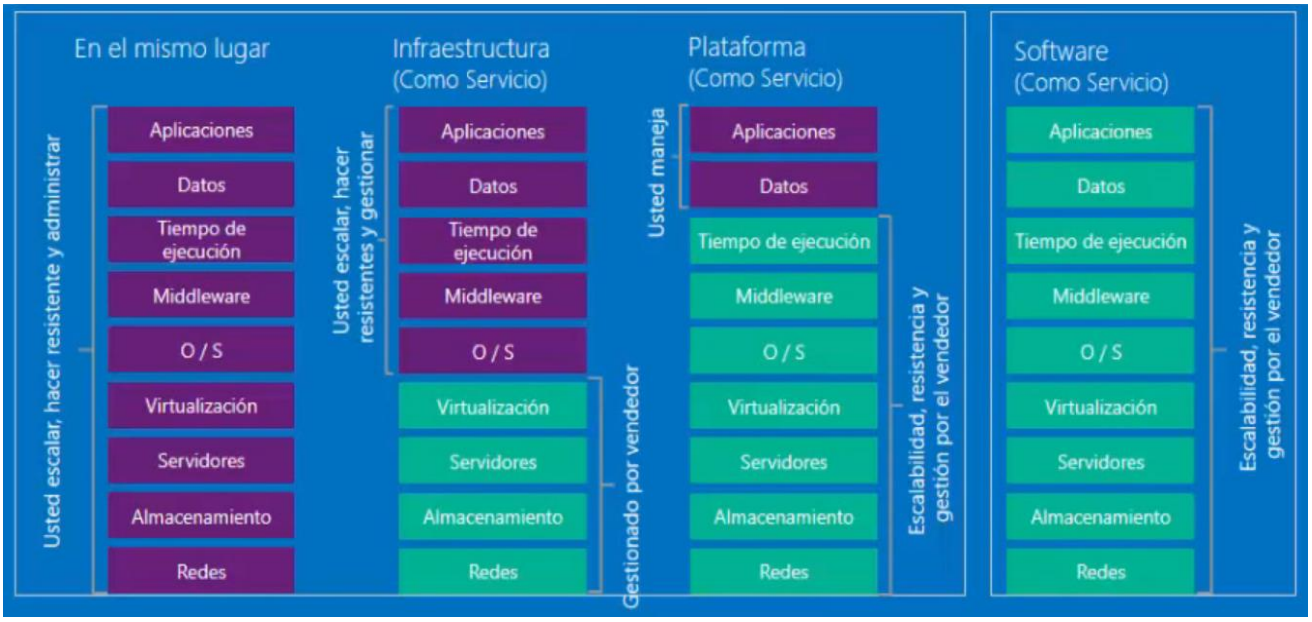
- **Uso de espacio de almacenamiento** (Cloud Storage).
  - Guardar archivos en un servidor de Internet.
  - Acceso vía Web o FTP.
  - Almacenar copias de seguridad.
  - Proveedores de almacenamiento en la nube:
    - Google Drive, OneDrive Microsoft, Mega, Amazon Cloud Drive, Dropbox, Apple iCloud.

- **Uso de Aplicaciones y Software**.
- **Uso de hardware**.

TIPOS DE IMPLEMENTACIÓN O SERVICIOS EN LA NUBE.

(Vertientes, tipos de nubes o tipos de computación en la nube).

1. **Infraestructura como servicio (IaaS) - Hardware como servicio.**
   - Alquiler de infraestructura a través de la red como servidores, almacenamiento, conexiones, etc.
   - La administración de la plataforma y software es responsabilidad del proveedor.
   - Algunos proveedores: Amazon Web Services, Joyent, Akamai, Microsoft Azure.
2. **Plataforma como servicio (PaaS).**
   - Permite el desarrollo de aplicaciones o programas personalizados.
   - El proveedor suministra las herramientas, complementos, módulos, API's, y componentes para su desarrollo.
   - El proveedor se encarga del hardware, la red, el sistema operativo, la seguridad...
   - Algunos proveedores: Google App Engine, Plataforma G.
3. **Software como servicio (SaaS).**
   - El proveedor suministra una aplicación como un servicio bajo demanda.
   - Acceso mediante un navegador.
   - El usuario tiene un control mínimo sobre las aplicaciones.
   - No necesario instalar las aplicaciones a nivel local.
   - El proveedor se encarga del hardware, la red, el sistema operativo, la seguridad, …
   - Algunos Proveedores: Google Docs, Microsoft Office 365, Correo Electrónico, CMS (Worpress, Joomla, Drupal, Prestashop).



TIPOS DE NUBES.

- **Pública.**
  - Todos los recursos (hardware y software), son compartidos por los usuarios a los que se le presta el servicio en la nube.

        - Acceso remoto vía internet.
        - Propietario gestiona, mantiene y actualiza el servicio y la infraestructura.
    - **Privada.**
        - El proveedor suministra el servicio a un único cliente.
        - Los usuarios suelen ser empresas, entidades y grandes organizaciones.
        - El usuario utiliza sus propios servidores, almacenamiento, hardware de red y aplicaciones sin ser dueño de ellos. (real o virtual).
        - Infraestructura bajo demanda fácilmente escalable.
        - Proveedor gestiona, mantiene y actualiza el servicio y la infraestructura, mientras que el cliente gestiona y mantiene sus propias aplicaciones y datos.
        - Alta protección de datos y privacidad de la información.
    - **Privada virtual. VPC (Virtual Private Cloud)**
        - Nube pública dentro de la cual un cliente bajo demanda puede solicitar aislarse del resto de usuarios públicos o de otros VPC.
        - Creando redes privada virtuales y canales encriptados.
        - Garantía de seguridad.
        - Proveedores: Amazon Web Services, Microsoft Azure, Cloud-Bricks, Forty-Cloud.
    - **Híbrida.**
        - Combinación de nube pública y nube privada.
        - El cliente es propietario de una parte privada de la nube mientras que también usa los servicios y recursos de la parte pública.
        - El cliente puede contratar y ampliar servicios según necesidades ya que tienen una alta escalabilidad.
    - **Comunitarias.**
        - Diferentes organizaciones o empresas comparten sus recursos en la nube para resolver un problema común.
        - Pueden ser administradas o gestionadas por esas empresa u organizaciones o por terceras partes.
    - **Multinube.**
        - Combinación de varios servicios suministrados por varios proveedores de nube privada o pública.
        - Surgen al aumentar y expandirse la empresas u organizaciones porque aumentan sus clientes y por ello deben aumentar sus servicios.

FUNCIONALIDADES AVANZADAS DE LOS SERVIDORES.

ARQUITECTURA DE ALTA DISPONIBILIDAD (HIGH AVAILABILITY).

- Es un protocolo utilizado tanto para el diseño de un sistema informático, como para su puesta en práctica para asegurar su funcionamiento.
- **Fiabilidad**:
    - Probabilidad de que un sistema funcione normalmente durante un periodo determinado de tiempo, lo que garantiza la continuidad de servicio.

- En las empresas el impacto de un fallo en el sistema por una parada puede implicar:
    - Pérdida de información.
    - Pérdida de ingresos y reputación.
    - Baja productividad.
    - Frustración y pérdida de clientes
    - Estrés del empleado
- **Sistema Informático de Misión Crítica.**
    - Se habla de ellos cuando del correcto funcionamiento de un sistema informático depende la vida, seguridad o propiedades de las personas.
    - Es necesario y obligatorio la redundancia de equipos y en especial, el uso de Sistemas de Alimentación Ininterrumpida (SAI).
    - Están incluidos en este tipo de sistemas el control aéreo, el control de las industrias críticas, el transporte, las finanzas, el equipamiento médico, etc.
- **Índice de Disponibilidad.**
    - La disponibilidad es la medida en que un sistema informático es capaz de proveer servicio ininterrumpido a sus usuarios.
    - Para medirla, se usa el índice de disponibilidad, expresado como un porcentaje que se calcula con la siguiente fórmula:

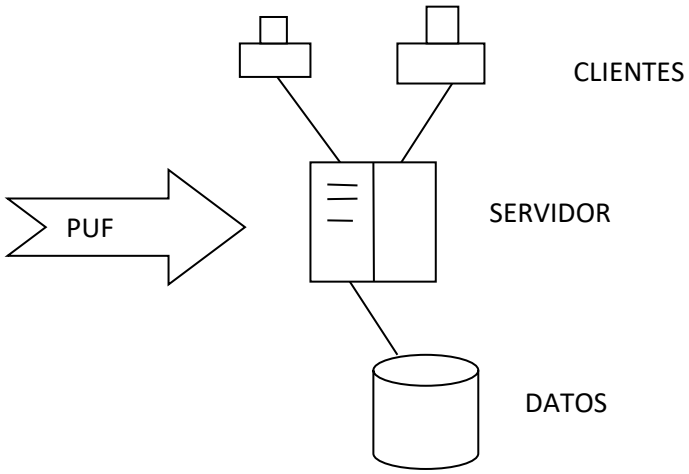
$$D = \frac{tiempo\ real\ funcionamiento = (tiempo\ estimado - tiempo\ de\ inactividad) * 100}{tiempo\ estimado\ funcionamiento}$$

    - El tiempo se mide en minutos.
    - El resultado se expresa en porcentaje.
    - Tiempos:
        - **Tiempo de real funcionamiento o Uptime:**
            - Tiempo en que se provee un servicio eficaz y aceptable.
        - **Tiempo estimado de funcionamiento**:
            - Tiempo en que, por configuración, diseño u otras características debería haber estado funcionando el sistema informático.
        - **Tiempo de no disponibilidad o Unavailable**:
            - El sistema no está disponible o está fuera de servicio.
        - **Tiempo de inactividad o Downtime:**
            - Tiempo en que un sistema informático está detenido y no presta servicio.
        - El **tiempo de inactividad** puede ser de 2 **tipos:**
            - **Planificado (Scheduled Downtime)**.
                - Sistema detenido por mantenimiento, operaciones de rutina, actualizaciones, cambios de configuración…
            - **No planificado (Unscheduled Downtime).**
                - Sistema detenido por fallos en el hardware, fallos en el software, ataques, catástrofes naturales, errores humanos…
    - Ejemplo:

- Un sistema debe estar funcionando 24/7/365 al año, sin embargo, por distintos motivos ha estado detenido durante 5 días. ¿Cuál ha sido su índice de disponibilidad?
  - 5 días -> 5 * 24 * 60 = 7.200 minutos.
  - 1 año = 525.600 minutos.
  - D = ((525.600 – 7.200) * 100) / 525.600 = 98,63%

TOLERANCIA A FALLOS.

- Característica del diseño de un sistema que le permite seguir funcionando en caso de fallo de uno o varios de sus componentes.
- **Diseño de sistemas sin Puntos Únicos de Fallo.**
  - **Punto Único de Fallo (PUF) - Simple Point of Failure (SPOF)**
    - Componente de un sistema informático que si falla provoca un fallo en todo el sistema, o impide que este funcione correctamente.
    - Puede ser un componente electrónico, de hardware, de software o incluso una persona responsable de un determinado servicio cuya ausencia provoque el mismo resultado que un fallo.
    - Para evitar los PUF se utiliza redundancia y esto nos lleva a la arquitectura de alta disponibilidad.



- **Redundancia.**
  - Réplica del hardware, software e información para su uso en caso de fallo de un sistema.
  - **Redundancia geográfica de los equipos.**
    - Consistente en que los componentes que pueden provocar una caída del sistema estén duplicados y configurados de modo que cuando uno falle, el sistema continúe funcionando mediante el componente redundante al que ha fallado.
    - **<u>Componente Redundante o Sistema Redundante:</u>**
      - Aquellos en los que se repiten equipos para asegurarlos frente a posibles daños.

        - **Dispositivos redundantes en un servidor**:
            - Procesadores con varias CPU y módulos de memoria.
            - Discos duros (uso RAID).
            - Tarjetas de red.
                - Los servidores suelen venir con 2 o más, para sumar capacidades y asegurar ante fallos.
            - Ejemplos:
                - Un servidor puede ser tolerante a fallos usando un servidor idéntico que se ejecuta en paralelo, con todas las operaciones duplicadas en el servidor de respaldo.
    - **Redundancia geográfica de los datos y aplicaciones.**
        - El software y los datos se replican en otros equipos para su uso en caso de que los programas o los equipos que los soportan dejen de funcionar.
        - Copias de seguridad y respaldo de la información.
        - Ejemplos**:**
            - Una base de datos puede replicarse continuamente en otro equipo. Si la principal deja de funcionar, las operaciones se redirigen automáticamente a la segunda base de datos.
    - **Otros**:
        - **Sistemas eléctricos**:
            - Fuentes de alimentación alternativas.
            - UPS (Sistemas de Alimentación Ininterrumpida).
            - Generadores auxiliares.
            - Uso de suministradores eléctricos independientes.
        - **Componentes de red**:
            - Permiten crear caminos diferentes entre dos componentes de red, por si alguno falla.
            - Los más usuales son los routers, los conmutadores o switchs, las tarjetas de red, el cableado coaxial o de fibra óptica, etc.

ESCALAMIENTO.

- Capacidad de crecimiento del software o de una aplicación para adaptarse a un creciente número de usuarios, transacciones y, en general, para adaptarse a mayores cargas de trabajo sin degradaciones del servicio.
- Tipos**:**
    - **Horizontal.**
        - Consiste en aumentar el número de servidores (nodos) que atienden una aplicación**.**
        - Varios servidores que trabajan como un todo en un clúster**.**
        - Cuando vuelva a ser necesario, se añaden más nodos al clúster**.**
        - Suele haber un servidor primario que gestiona el clúster**.**
        - Si uno de los nodos falla, los demás siguen funcionado**.**
        - El escalamiento horizontal soporta alta disponibilidad y balanceo de carga**.**
    - **Vertical.**
        - Consiste en aumentar el hardware de uno de los nodos, es decir, se cambia el hardware (disco duro, memoria, procesador, etc. …), por otro más potente.

CLÚSTER.

- Sistema distribuido compuesto por varios equipos autónomos interconectados, que se emplean como un recurso computacional único.
- **Clúster de balanceo de carga o de cómputo adaptativo.**
    - Uno o más ordenadores que actúan como puerta de entrada del clúster, encargándose de las peticiones que recibe éste; y su envío a otros ordenadores del clúster.
    - Sistema fácilmente escalable y robusto, ya que, aunque caigan algunos ordenadores del clúster, otros seguirán funcionando.
- **Balanceo de Carga:**
    - Técnica que permite repartir las operaciones o tareas a realizar entre varios de los recursos que componen el clúster (ordenadores, servidores, etc.).
    - Ventajas:
        - Aumento de la velocidad de acceso por parte del usuario al servidor.
        - Aumento de la fiabilidad del sistema.
        - Aumento de la tolerancia a fallos.

MANTENIMIENTO.

- Relacionado con lo versátil o no que puede ser un servidor a la hora de actualizar, depurar y corregir errores y mantenerlo en funcionamiento.
- Una mejora en el mantenimiento se da al crear aplicaciones o programas reusables y modulares.

APLICACIONES.

- **Concepto.**
    - Programa informático diseñado para realizar una o varias funciones determinadas.
    - Puede ser genéricas o hechas a medida.
- **Tipos**.
    - **Aplicación de escritorio.**
        - Aquella instaladas en un equipo determinado (ordenador, Tablet, móvil), que no necesitan Internet para funcionar.
        - Ventajas:
            - Rápidas porque aprovechan adecuadamente los recursos del ordenador.
            - Pueden acceder a los dispositivos y periféricos del ordenar (disco duro, memoria, impresora, tarjeta gráfica, cámara, etc.) para su funcionamiento.
        - Desventajas:
            - Algunas pueden consumir muchos recursos del ordenador.
            - Puede ser complicado mantener actualizadas las aplicaciones que se ejecutan en varios ordenadores.
            - Pueden requerir una instalación y configuración compleja, que no todo usuario puede conocer.
            - Pueden no existir versiones para todos los sistemas operativos.
        - Desarrollo:

- Su desarrollo implica crear la aplicación y también puede ser necesario crear una base de datos.
    - **Aplicación**:
        - Los más cómodo es utilizar un IDE (Entorno de Desarrollo Integrado que incorporara lenguajes de programación, librerías, frameworks, etc.
        - Tras realizarse el código fuente, éste debe ser compilado para obtener los archivos ejecutables necesarios.
    - **Base de datos**:
        - Se necesita un gestor de base de datos.
        - Para poder acceder a los datos desde la aplicación puede ser necesario usar un driver o conector de base de datos.

- **Aplicación web.**
    - Aplicación que se utiliza y es ejecutada desde un navegador, ya que está alojada en un servidor o en la nube.
    - Ventajas:
        - Se puede ejecutar en cualquier dispositivo con un navegador web y conexión a Internet.
        - No hay que instalarlas, ni actualizarlas, esas operaciones se hacen en el servidor.
        - Pueden acceder a dispositivos del equipo local (cámara, micrófono, etc.).
        - No requieren de ninguna tarea en el ordenador local.
    - Desventajas:
        - No se pueden usar sin no hay conexión a Internet.
        - Si son aplicaciones propias, pero instaladas en un servidor, requieren contratar éste.
    - Desarrollo:
        - **Aplicación.**
            - Uso de lenguajes de programación para su creación, por ejemplo, a través de un IDE**.**
        - **Bases de datos.**
            - Pueden necesitar acceso a datos, por lo que también puede ser necesario el uso de un Sistema Gestor de Bases de Datos**.**
- **Aplicación cliente/ servidor.**
    - La aplicación está dividida y distribuida en tareas.
    - Unas tareas se ejecutan en el servidor y otras en el cliente.
    - Tras establecerse la conexión cliente - servidor, en el cliente se abre una interfaz de trabajo para realizar las tareas.
    - El cliente envía la información necesaria al servidor, se realizan las acciones necesarias y posteriormente éste envía al cliente el resultado mostrando por pantalla la información procesada.
    - Desarrollo:

- Se utilizan las mismas herramientas que para las aplicaciones de escritorio, es decir, lenguajes de programación que pueden usarse desde un IDE.
- Pueden o no necesitar bases de datos o interfaz.

TECNOLOGÍAS DE DESARROLLO DE APLICACIONES.

- Antes de crear, usando lenguajes de programación, el código que definirá cada programa de una aplicación, hay que pasar por etapas previos de diseño y desarrollo de una aplicación.
- Para estas tareas se pueden usar diversas herramientas.
- **Herramientas CASE**.
    - Computer Aided Software Engineering - Ingeniería de Software Asistida por Ordenador.
    - Conjunto de aplicaciones informáticas empleadas para automatizar las actividades del ciclo de vida del desarrollo de software.
    - Son usadas por los directores de proyectos de software, analistas e ingenieros para desarrollar sistemas de software.
    - Su uso acérela el desarrollo de un proyecto, ayuda a encontrar errores y permite ir avanzando por las distintas etapas del desarrollo.
    - Componentes:
        - **Depósito central.**
            - Requieren de un depósito central que sirve de fuente de común, consistente e integrada información.
            - Es un lugar central de almacenamiento, donde se guarda información sobre los requisitos, documentación, informes y diagramas relacionados que el software a desarrollar.
            - El depósito central también sirve como diccionario de datos.
        - **Herramientas Upper CASE.**
            - Son las que se usan en las fases de planificación, análisis y diseño.
        - **Herramientas Lower CASE.**
            - Se usan en la implementación, pruebas y en el mantenimiento.
        - **Herramientas Integrated CASE.**
            - Son usan en todas las fases del desarrollo.
    - Tipos**:**
        - Herramienta CASE diagrama.
        - Herramientas de documentación.
        - Herramientas de la ingeniería de la información.
        - Herramientas modelado de procesos.
        - Herramientas de administración.
        - Herramientas de planificación de proyectos.
        - Herramientas de administración de proyectos.
        - Herramientas de seguimiento de requisitos
        - Herramientas de métricas y gestión.
        - Herramientas de documentación
        - Herramientas de software de sistema.
        - Herramientas de control de calidad.
        - Herramientas de gestión como base de datos.

        - Herramientas de codificación de cuarta generación.
        - Herramientas de mantenimiento.
        - Herramientas de gestión de configuración de software.
        - Herramientas de análisis y diseño.
        - Herramientas de desarrollo y diseño de interfaz.
        - Herramientas de generación de prototipos.
        - Herramientas de programación.
        - Herramientas de integración y comprobación.
        - Herramientas de análisis estático.
        - Herramientas de análisis dinámico.
        - Herramientas de gestión de comprobación.
        - Herramientas de comprobación clientes/servidor.
        - Herramientas de reingeniería.
        - Herramientas de desarrollo Web.
        - Herramientas de aseguramiento de la calidad.
        - Herramientas de control de cambios.
- **UML.**
    - Lenguaje Unificado de Modelado – Unified Modelling Language.
    - Lenguaje gráfico empleado para visualizar, especificar, construir y documentar un sistema.
    - Lenguaje grafico porque utiliza diagramas, pero no es un lenguaje de programación.
    - Última versión: 2.5.1.
    - Es muy utilizado en la programación orientada a objetos porque incluye varias notaciones orientadas a objetos como diseño orientado a objetos, técnicas de modelado de objetos e ingeniería de software orientada a objetos.
    - Tipos de diagramas:
        - **Estructura**.
            - Son estáticos porque representan los elementos individuales de un sistema.
            - Tipos:
                - Clases.
                - Objetos.
                - Componentes.
                - Estructura compuesta.
                - Paquetes.
                - Despliegue o distribución.
        - **Comportamiento**.
            - Son dinámicos porque representan procesos y situaciones dinámicas.
            - Tipos:
                - Casos de uso.
                - Actividades.
                - Estados.
                - **Interacción**.
                    - Subtipo de los diagramas de comportamientos utilizados para modelar el comportamiento de los elementos cuando intercambian información.
                    - Tipos:

- Secuencia.
- Comunicación.
- Tiempo.
- Interacción.

TECNOLOGÍAS Y HERRAMIENTAS DE DESARROLLO.

- **Lado cliente**.
  - HTML5, CSS3 y JavaScript.
- **Lado Servidor**.
  - Lenguajes de programación PHP, Perl, Node.js, Python, C#, Visual Basic, Java, etc.
- **Bases de datos**:
  - MySQL, MariaDB, MS SQL Server, PostgreSQL, Oracle Database, etc.
- **Gestores de contenido**.
  - CMS (Content Management System)
  - Aplicación web que sirve para desarrollar un sitio web.
  - Utilizan una interfaz gráfica para la creación con editor WYSIWYG.
  - Funciones básicas:
    - Creación del sitio web.
    - Gestión y mantenimiento del sitio web.
    - Administración del sitio web y del propio CMS.
  - CMS más usados:
    - WordPress, Drupal, Moodle, Joomla, PrestaShop, etc.

ENTORNOS DE DESARROLLO.

- **Entorno de trabajo**.
  - Equipo en el que se va a crear la ampliación web.
  - Para ello se necesita:
    - Programas de edición de código (HTML, CSS, Lenguajes de programación, etc.)
    - Navegadores.
    - Programas FTP.
    - Sistemas Gestores de Bases de Datos.
    - Conexión de red.
- **Entorno de pruebas o pre-produccion.**
  - Equipo donde se prueba el código o la aplicación creada, usando datos o supuestos los más próximos a los reales posibles.
  - Debe ser un entorno lo más parecido posible al de producción, siendo aconsejable, que tenga el mismo sistema operativo que éste.
- **Entorno de producción.**
  - Equipo donde se alojará el sitio o la aplicación web para estar disponible para los usuarios.
  - Es necesario que tenga un dominio.

ORGANIZACIÓN DEL CONTENIDO DE UN SITIO WEB.

- Para tener mejor organizado el contenido de un sitio web es necesario crear una estructura de carpetas que alojaran los distintos archivos que conformen el sitio web.

- La organización depende del creador, si bien, hay algunas carpetas que son comunes a muchos sitios web o son creada por defecto si usan CMS (Sistemas de Gestión de Contenidos). También pueden ser incluida determinados hostings.
- Algunos archivos deben estar ocultos al usuario. A ellos se le asignan propiedades de ocultación. En Linux los archivos ocultos llevan un punto delante del nombre.
- Tipos de contenido.
    - **Páginas o programas de inicio.**
        - Páginas como *index.html, index. php, default.htm, default.asp*, etc., se guardan en la carpeta raíz el sitio.
        - Archivos de robots.
        - Para algunos CMS o sitios web existe una carpeta, *public_html*, que actúa como directorio raíz.
    - **Archivos de administración.**
        - Se guardan en la carpeta *admin*.
    - **Programas.**
        - En carpetas con su nombre o, si son scripts, en una carpeta con el nombre de *scripts*.
    - **Hojas de estilo.**
        - Se pueden guardar en la carpeta *css*.
        - Si hay diversos temas, se pueden crear subcarpetas con los temas y sus estilos asociados.
    - **Elementos multimedia**.
        - Imágenes, videos, audios, etc., se pueden guardar en carpetas exclusivas (imagenes, images, fotos, audio, etc.), o en una carpeta multimedia con o sin subcarpetas.
    - **Ficheros de configuración.**
        - Se guardan en la carpeta raíz el sitio.
        - Un ejemplo es el fichero *.htaccess* que, usado en servidores Apache, es un archivo oculto que se permite configurar un sitio web y realizar entre otras acciones como:
            - Reescribir la URL.
            - Proteger directorios con contraseña.
            - No permitir el acceso a direcciones IP específicas.
            - Cambiar la zona horaria del sitio web.
        - Otro fichero es *.htpasswd* que permite proteger el acceso al sitio web con una contraseña y un nombre de usuario.
    - **Documentos.**
        - Se pueden guardar en carpetas con el nombre de documentos o con nombres relativos a la función de los propios documentos.
        - También puede accederse a ellos mediante aplicaciones ftp y, por tanto, almacenarse en carpetas con nombres como *ftp, public_FTP*, etc.
    - **Correo electrónico**
        - Pueden guardarse en una carpeta con el nombre *email.*

        - En ella se guardan los correos, las cuentas de los usuarios que se han creado y todo lo relacionado con el correo electrónico.
        - Puede tener subcarpetas como spam, enviados, recibidos, etc.
    - **Usuarios y contraseñas**.
        - Las bases de datos con la información sobre los usuarios, contraseñas, etc., se guardan en la carpeta *etc*.
    - **Otras carpetas**.
        - Pueden almacenar archivos temporales, librerías, etc.

SEGURIDAD EN APLICACIONES WEB.

- Los ataques más comunes a los sitios web son a las bases de datos para robar información, normalmente aprovechando vulnerabilidades en los servidores o en las aplicaciones web.
- Si bien, puede producirse fallos en equipo y software, la mayoría de los problemas relativos a la seguridad de sitios web se producen a nivel ampliación y tienen que ver con una programación incorrecta realizada por los programadores.
- Para garantizar la seguridad es necesario:
    - Políticas de seguridad.
    - Mecanismos de seguridad.
- **Políticas de seguridad**.
    - Para desarrollar una buena política de seguridad y garantizar la integridad, confidencialidad y disponibilidad de la información, es necesario conocer los estándares de ciberseguridad para las tecnologías de la información (TI):
    - **Estándares de ciberseguridad**. (Wikipedia).
        - Técnicas destinadas a proteger el entorno cibernético de un usuario u organización.
        - Éste entorno incluye a los usuarios, redes, dispositivos, software, procesos, información almacenada o en transferencia, aplicaciones, servicios y sistemas que pueden conectarse directa o indirectamente a las redes.
        - Objetivo:
            - Reducir los riesgos, incluyendo prevención o atenuación de ciberataques.
        - Los estándares incluyen:
            - Herramientas, políticas seguridad, conceptos de seguridad, salvaguardas de seguridad, guías, enfoques de gestión de riesgos, acciones, capacitación, mejores, prácticas, aseguramiento y tecnologías.
    - Estándares actuales:
        - ISO/IEC 27000: Gestión de la seguridad de la información (SGSI).
        - ISO/IEC 27032: Directrices para la ciberseguridad.
        - ISO/IEC 27033: Seguridad de las redes.
        - ISO/IEC 27034: Seguridad de las aplicaciones.
        - ISO/IEC 27035: Gestión de incidentes de seguridad de TI.
        - ISO/IEC 27036: Gestión de la seguridad de la información en relaciones con terceros.
        - ISA/IEC 62443-1-1: Redes de comunicaciones industriales: seguridad de la red y del sistema.

        - ISA/IEC 62443-2-1: Establecimiento de un programa de gestión de ciberseguridad para IACS (sistemas de control y automatización).
        - ISA/IEC 62443-2-3: Gestión de parches en entorno IACS.
        - ISA/IEC 62443-2-4: Requisitos del programa de seguridad para los proveedores de servicios IACS.
        - ISA/IEC 62443-3-1: Tecnologías de seguridad para IACS.
        - ISA/IEC 62443-3-2: Evaluación del riesgo de seguridad y diseño de sistema.
        - ISA/IEC 62443-3-3: Requisitos de seguridad para IACS.
        - IEC 61158: Comunicaciones industriales. Especificaciones para buses de campo y redes de tiempo real.
        - IEC 61784-3: Seguridad funcional de buses de campo.
        - ISO 10219-1: Requisitos de seguridad para robots industriales.
        - ISO 20218-2: Requisitos de seguridad para la integración de robots industriales.

- **Seguridad de la información.**
    - **Disponibilidad.**
        - Característica, cualidad o condición de la información de encontrarse a disposición de quienes deben acceder a ella, ya sean personas, procesos o aplicaciones, en el momento en que la requieran.
    - **Confidencialidad.**
        - Propiedad de la información por la que se garantiza que ésta sólo es accesible por personal autorizado.
        - Definida en la normativa ISO/IIEC 27002.
        - La Información que se da entre un emisor y un receptor debe protegerse frente a terceros. Y esa protección debe de ser independiente del sistema de comunicación. Si un tercero intercepta la comunicación para él debe ser inteligible.
        - Criptosistema: Procedimiento para codificar la información y garantizar su confidencialidad.
    - **Integridad**:
        - Exactitud, precisión, coherencia y validez de los datos.
        - Es la garantía de que la información no pueda ser modificada sin autorización o permiso para ello.

AMENAZAS A UN SISTEMA INFORMATICO.

- **Factores físicos naturales**.
    - Inundaciones, hundimientos, daños por viento, descargas eléctricas atmosféricas, nieve, hielo, terremotos...
- **Factores físicos de vecindad** (proceden del entorno creado por el hombre).
    - Proximidad a servicios de agua, gas, electricidad, etc.
    - Proximidad a equipamientos o sistemas que pueden fallar.
    - Actos vandálicos, robo, atentados...
- **Factores humanos**.

- Incompetencia, intrusión, robo, programas maliciosos, fallos de software, errores de programación, problemas internos en las empresas que pueden producir problemas laborales, disputas, venganzas, sabotajes...

- **Otros**.
  - Fallo del suministro eléctrico, fallos de las comunicaciones, fallos en los equipos, etc.

Causas de desastre más comunes:

- Factor humano.
- Fallo en las comunicaciones.
- Fallo en el suministro eléctrico.
- Desastres naturales.
- Fuego.
- Fallos en el equipamiento y en el hardware.
- Daños por agua.

SALVAGUARDA FÍSICA Y LÓGICA.

- Para evitar o minimizar los daños producidos por las distintas amenazas a las que está sometido un sistema informático, se toman medidas tanto a nivel físico, como lógico que garanticen la disponibilidad, integridad y confidencialidad de la información.
- Medidas de seguridad física:

  - Ubicación adecuada para evitar desastres naturales.
  - Edificaciones a prueba de desastres.
  - Redundancia geográfica de instalaciones y equipos.
  - Sistemas eléctricos, antiincendios, etc. redundantes.
  - Vigilancia (personas, cámaras).
  - Acceso restringido con cerrojos (convencionales, magnéticos, biométricos...).
  - SAI: Sistemas de alimentación ininterrumpida.
- Medidas de seguridad lógica:

  - Replica y respaldo de datos y programas (Copias de seguridad, clonación de discos, particiones y puntos de restauración)
  - Control del acceso a los usuarios (contraseñas y logins)
  - Controlar el uso de los programas e información (permisos)
  - Garantizar la integridad de la información que se transmite.
  - Controlar que la información enviada sea recibida por los usuarios adecuados.
  - Controlar que la información recibida sea la misma que la trasmitida.
  - Disponer de sistemas alternativos de trasmisión entre diferentes puntos.
  - Preservar los sistemas operativos, programas, etc. con programas específicos de protección (antimalware, cortafuegos...)

MALWARE.

- Concepto.
  - **Malware** (Malicious Software - Software Malicioso)

- Todo tipo de software cuya función es dañar o causar mal funcionamiento del sistema o de las aplicaciones instaladas.

- **Tipos**.
  - **Virus**: Programa informático que sin conocimiento o permiso del usuario infecta a otros archivos del sistema con intención de modificarlos o dañarlos.
  - **Troyano**: Programa alojado dentro de otro programa normal que se instala al ejecutarse el huésped y realiza tareas ocultas para el usuario de distintos tipos.
  - **Gusano**: Conjuntos de programas que se diferencian de los virus en que no necesitan un archivo anfitrión para seguir vivos y replicarse.
  - **Adware**: Software que despliega publicidad de distintos productos o servicios que aparece en ventanas emergentes o en barras que simulan ofrecer distintos servicios. Suelen estar ocultos en programas gratuitos.
  - **Backdoor**: Programas que abren puertas traseras para que el creador tome el control del sistema y haga lo que quiera con él.
  - **Botnet**: Sirven para crear redes de equipos infectados controladas por el atacante para que todos los equipos trabajen de forma conjunta y distribuida. A estos ordenadores infectados se les llama robots o zombis.
  - **Hijacker**: Programas que secuestran algunas funciones del navegador, por ejemplo, modifican la página de inicio y las de búsqueda.
  - **Keylogger**: Se encargan de almacenar en un archivo todo lo que el usuario teclea y así para robar contraseñas, nombres de usuario, etc.
  - **Rootkit**: Se instalan profundamente en el sistema operativo sustituyendo archivos críticos de éste por archivos controlados por el rootkit. Suministran información, realizan seguimiento, etc., y pueden inutilizar el sistema operativo al ser desinstalados o eliminados.
  - **Spam**: Correo electrónico basura no solicitado que es enviado masivamente.
  - **Hoax**: correos electrónicos distribuidos en forma de cadena cuyo objetivo es hacer pasar por real lo que es falso.
  - **Rogue**: Programa falso que dice ser o hacer lo que no es.
  - **Phising**: Sirve para robar información personal y/o financiera a través de la falsificación de un ente de confianza.
  - **Spyware**: Programas espías que recopilan información sobre personas u organizaciones sin su consentimiento y normalmente con fines publicitarios. Envía hábitos de navegación, contenido o datos de las páginas que se visitan, direcciones de páginas, etc., con objeto de crear perfiles de los datos de los usuarios.
  - **Ransomware**: Secuestran el ordenador o encriptan la información de éste, dando instrucciones al usuario para que pueda recuperar el control a cambio del pago de dinero.
  - **Bomba lógica**: Malware insertado en un programa informático que permanece oculto hasta cumplirse una o más condiciones preprogramadas, en ese momento se ejecuta una acción maliciosa. El código malicioso se ejecuta cuando se cumple una o varias condiciones o, cuando se cumple un tiempo predeterminado.

- **Consejos de protección.**

  - Instalar programas de protección. (Antiespías, antivirus, cortafuegos, etc.).

- Tener sentido común.
- Estar siempre actualizado.
- Mantenerse informado sobre nuevas amenazas en foros, páginas de seguridad, etc.

CORTAFUEGOS (FIREWALL).

- Permiten controlar el acceso a un sistema informático o a un equipo, bloqueando el acceso no autorizado y permitiendo las conexiones autorizadas.
- Amplio uso para controlar el acceso a redes privadas (Intranets).
- **Funciones**:
    - Preservar nuestra seguridad y privacidad.
    - Proteger una red doméstica o empresarial.
    - Mantener a salvo la información almacenada en redes, servidores, ordenadores.
    - Evitar intrusiones de usuarios no deseados.
    - Evitar distintos tipos de ataques como:
        - Denegación de servicio (DOS - Denial of Service):
            - Saturación de los puertos con un flujo de información que provoca que un servicio o recurso acabe siendo inaccesible.
        - Spoofing:
            - Suplantación indebida de IP, de nombre de dominio, de página web, …
- **Funcionamiento**:
    - Todo el tráfico e información que pasa por el router es analizado.
    - Si cumple una serie de reglas configuradas en el firewall, podrá entrar o salir de nuestra red, si no, se bloqueará.
- **Reglas**:
    - Administrar a usuarios acceso a los servicios privados de la red.
    - Registro de intentos de entradas y salidas de la red.
    - Filtrar paquetes (datagramas) en función de su origen, destino y número de puerto (Filtro de direcciones).
    - Filtrar determinados tipos de tráfico, por ejemplo, no dejar pasar tráfico FTP, Telnet, etc. (Filtrado de protocolo).
    - Controlar número de conexiones que se producen desde un mismo punto y bloquearlos desde un mismo punto y bloquearlos si se sobrepasa un límite (Evitar así ataques por denegación de servicio o DoS).
    - Control de aplicaciones que pueden acceder a internet.
    - Dirección de puertos en escucha que no deberían estar abiertos (una aplicación quiere usar un puerto para esperar conexiones entrantes).
- **Implementación**:
    - **Cortafuegos por Hardware.**

- Dispositivo o conjunto de dispositivos instalado en una red para levantar una defensa que proteja ésta del exterior.
- Se ubican entre el punto de unión de 2 redes, por ejemplo, entre una red privada e Internet y entre router y switch).
- Pueden ser dispositivos específicos (Cisco) o routers que disponen de la función firewall.
- Solución excelente para proteger una red empresarial.
- Implementan otras funcionalidades:
  - Tecnologías **SSL** (Secure Sockets Layer - Capa de Puertos Seguros).
  - **VPN** (Virtual Private Network - Red Privada Virtual)
  - Programas antivirus y/o antispam integrados.

- **Cortafuegos por software.**

  - Programa que se instala directamente en los ordenadores o servidores protegiendo sólo al equipo donde esté instalado.
  - Funcionalidad más limitada que por hardware.
  - Consume recursos del equipo.
  - Algunas aplicaciones cortafuegos:
    - Las incorporadas en los Sistemas Operativos:
      - Windows Firewall.
    - Software de terceros:
      - Comodo Firewall.
      - PC Tools Firewall Plus.
      - Zone Alarm.
      - Outpost Firewall.
- **Ambos.**
  - Implementación de cortafuegos por hardware y software a la vez.

COPIAS DE SEGURIDAD:

Concepto.

- Es una copia de los datos en un momento temporal determinado que es depositada en un lugar seguro para su posterior uso en caso de necesidad.
- Se puede duplicar todo o parte de un sistema.

Tipos Principales De Copias De Seguridad.

- **Total o Completa:**
  - Copia de todos los archivos que contienen la información a salvaguardar o si es necesario de todo el sistema. Ésta desactiva el atributo modificado a todos los archivos.
  - Ventajas**:**
    - Se copia todo, sólo se necesita una única copia para la restauración.

- Inconvenientes**:**
    - Necesita un soporte de almacenamiento lo suficientemente grande si la cantidad de información a salvaguardar es muy voluminosa.
    - El proceso de copia puede demorarse en el tiempo. Esto influye en el RTO (Tiempo Objetivo de Recuperación).

- **Incremental:**
    - Copia sólo los datos o ficheros nuevos o que han cambiado desde la última copia total o incremental. Solo copia archivos con atributo o fecha modificados activado. Tras la copia del archivo este atributo se desactiva.
    - Ventajas**:**
        - Ocupa poco espacio y se hace muy rápido.
    - Inconvenientes**:**
        - Se generan multitud de copias y son todas necesarias para la restauración; si se corrompe alguna, no se puede recuperar la información.
    - **Diferencial:**
        - Sólo se copian los datos o ficheros nuevos o que han cambiado desde la última copia completa. Es una copia acumulativa. Es igual que la incremental pero el atributo modificado no se desactiva.
        - Ventajas**:**
            - Sólo necesita tener la última diferencial más la completa.
        - Ocupa poco espacio y se hace muy rápido.
        - Inconvenientes**:**

Genera copias que ya tiene en las copias previas. Por lo que desperdicia espacio.

RESTAURACIÓN:

- Es un proceso que deja un sistema o archivos en el mismo estado en el que se hizo la copia de seguridad.
- **Motivos para hacer una restauración**:
    - Por eliminación accidental por parte del usuario.
    - Ataques al sistema.
    - Virus.
    - Fallos del hardware y en general cualquier problema que implique pérdida de datos.
- **Proceso De Restauración:**
    - Restablecer la última copia total.
    - Restablecer todas las incrementales en orden descendente desde la última copia total.
    - Restablecer la última diferencial siempre y cuando sea la última copia realizada.
- **Programas para copias de seguridad:**
    - Los propios del S.O. o los servidores Web.
    - Programas de terceros, son gratuitos y de pago.
- **Compresión y cifrado**
    - **Compresión:**
        - Al realizar una copia de seguridad, quizá sea necesario ajustar la compresión en función de la capacidad de almacenamiento.

- Para optimizar el espacio disponible, es necesario ajustar el nivel de compresión hasta que se adecue a las necesidades.

- **Cifrado:**
  - Para aumentar la seguridad de las copias realizadas, éstas pueden cifrarse o encriptarse.

ACCESO RESTRINGIDO POR CUENTAS DE USUARIO.

- **Usuario:**
  - Persona que utiliza un sistema informático.
- **Identificación:**
  - Requisito necesario para que el usuario pueda acceder al sistema y utilice sus recursos.
- **Cuenta de Usuario:**
  - Necesaria para identificarse ante un sistema.
  - Están compuestas de un nombre de usuario (login) y de una contraseña (password), así como de una serie de permisos o privilegios asociados.
- **User ID:**
  - Identificador que el sistema operativo asocia a un usuario identificado.
  - Posteriormente, el sistema operativo utiliza el User ID en lugar del nombre del usuario.
  - Normalmente es un número entero.
- **Gestión de Usuarios:**
  - La gestión de una cuenta posibilita autentificar la identidad y controlar el acceso a recursos. Además, permite auditar las acciones realizadas usando la cuenta.
- **Autenticación:**
  - Proceso de identificación de un usuario frente a un sistema informático.
  - Asegura que las personas o entidades que acceden a la información son quien dicen ser.
  - Sistemas de autenticación.
    - **Sistemas basados en algo conocido por el usuario.**
      - Contraseña, pin, etc.
    - **Sistemas basados en objetos poseídos.**
      - Tarjetas de identidad, DNI electrónico, tarjeta bancaria, llaves USB Epass Token, etc.
      - USB Epass Token.
        - Dispositivo electrónico que almacena firmas digitales o datos biométricos para facilitar el proceso de autenticación y pueden incluir teclados exclusivos para teclear pines.
    - **Sistemas basados en datos biométricos**.
      - Huella dactilar, patrones oculares, verificación de escritura, verificación de voz, etc.
    - **Sistemas de autenticación multifactor.**

        - Combinación de varios métodos de identificación.
    - Factores de autenticación humanos:
        - **Factores que hacen referencia a lo que se es**.
            - Huellas dactilares, reconocimiento de voz, ADN, etc.
        - **Factores que hacen referencia a lo que se tiene**.
            - Tarjetas, llaves USB ePass Token, etc.
        - **Factores que hacen referencia a lo que se sabe**.
            - Contraseñas, pines, frase, etc.
        - **Factores que hacen referencia a lo que se hace**.
            - Escribir, firmar, hablar, etc.
        - Ejemplos:
            - Autenticación mediante 2 factores:
                - Tengo + Sé → Tarjeta + Pin.
                - En un cajero automático la identificación es de doble factor, ya que se necesita una tarjeta y un pin.
            - Autenticación mediante 3 factores:
                - Tengo + Sé + Soy → Dispositivo Criptográfico + Pin + Huella.

IDENTIFICADOR ÚNICO DE ACCESO.

- Permite al usuario que se identifique una sola vez dentro de un sistema.
- **Single Sign-On o Reduced Sign-On Systems. (SSO).**
    - **Concepto**.
        - Sistemas de identificación que permiten acceder a varios sistemas, o a los distintos recursos de un mismo sistema, con una única identificación de usuario.
    - **Características.**
        - **Multiplataforma**.
            - Facilitan las tareas de inicio de sesión y acceso a recursos desde múltiples plataformas.
        - **Trasparencia**.
            - Se accede a los recursos de forma transparente debido a la automatización del inicio de sesión.
        - **Facilidad de uso**.
        - **Gestión sencilla**.
            - Permite la sincronización de contraseñas e información de usuarios.
            - Facilita la gestión de los administradores.
        - **Control de acceso**.
            - SSO no modifica los permisos de los usuarios.
        - **Seguridad**.
            - Los SSO incluyen sistemas de seguridad, por ejemplo, la información del usuario se transmite encriptada.
    - **Tipos Principales de SSO**.

- E-SSO (Enterprise Single Sign-On).
- Web-SSO
- Kerberos
- Open ID
- Identidad Federada.

PERMISOS PARA ARCHIVOS Y DIRECTORIOS.

- Los permisos son directivas que controlan o determinan lo que un usuario puede hacer con los archivos o los directorios.
- Para poder usar la información por parte de distintos usuarios se pueden compartir directorios y archivos asignándoles previamente a éstos una serie de permisos.
- **Permisos sobre directorios compartidos**:
  - Se establecen permisos para acceder a los directorios y su contenido desde otros equipos.
  - Además de los permisos para el acceso desde otros equipos, se establece el nº de usuarios con acceso permitido.
- **Permisos de acceso local**:
  - Necesarios cuando se accede desde el mismo equipo.
- **Herencia**:
  - Los permisos además de afectar a determinados archivos y directorios pueden también afectar a los que dependen de ellos o a los que se creen posteriormente.
- **TIPOS DE PERMISOS EN WINDOWS.**
  - **Control total**.
    - Permiso de máximo nivel.
    - Permite todo tipo de acciones.
  - **Modificar**.
    - Otorga todos los permisos menos eliminar archivos y subdirectorios, cambiar permisos o tomar posesión.
  - **Lectura**.
    - Permite ver archivos y subdirectorios, ver datos, atributos, permisos y mostrar directorios.
  - **Ejecución**.
    - Permite ejecutar programas.
  - **Escritura**.
    - Permite crear archivos y subdirectorios, añadir datos a archivos, modificar atributos y leer permisos.
- **TIPOS DE PERMISOS EN LINUX.**
  - En Linux los permisos son más o menos los mismos que en Windows, pero se otorgan a tres niveles:
    - los aplicados al propietario del archivo.
    - los aplicados al grupo que tiene el archivo.
    - los aplicados a todos los usuarios del sistema.
  - Sintaxis:
    - x xxx xxx xxx
  - Primer carácter:

        - Identifica un archivo o un directorio:
            - Un archivo se identifica con un guión (-).
            - Una carpeta se identifica con la letra d.
    - Resto de caracteres:
        - Posiciones:
            - Primera:
                - Permiso de lectura.
            - Segunda:
                - Permiso de escritura.
            - Tercera:
                - Permiso de ejecución.
        - Caracteres:
            - Sin permiso (-)
            - Letra r permiso de lectura.
            - Letra w permiso de escritura.
            - Letra x permiso de ejecución.
        - Grupos de caracteres:
            - De 2 a 4:
                - Identifican a un usuario.
            - De 5 a 7:
                - Identifican a un grupo.
            - De 8 a 10:
                - Identifican al resto de usuarios.
            - Ejemplos:
                - -rwxr—r--
                - drwxrwx---
        - Permisos con número.
            - Se especifica un numero de tres cifras, siendo la primera la que representa al usuario, la segunda al grupo y la tercera al resto de usuarios.
        - Valores:
            - 0 representa al guion (-).
            - 1 representa permiso de ejecución.
            - 2 permiso de escritura.
            - 4 permiso de lectura.
        - Ejemplos:
            - 704 equivale a rwx---r--
            - 444 equivale a r-- r-- r--

CERTIFICADO DIGITAL.

- **Concepto**.
    - Sistema que permite verificar online una identidad real de forma inequívoca.
    - Documento electrónico que asocia una clave pública con la identidad de un usuario.
- **Contenido**.

    - Datos de identificación del usuario y la clave pública asociada.
- **Características**.
    - Periodo de vida limitado asociado a 3 estados:
        - Activo (en uso).
        - Inactivo (no está en uso).
        - Revocado.
    - Validez.
        - Exige de una tercera parte de confianza (TCP-TTP). En España, la autoridad de certificación es la FNMT-RCM (Fábrica Nacional de la moneda y Timbre - Real Casa de la Moneda).
    - Formato estándar es X.509 y V.30.
- **Tipos.**
    - Certificado personal.
        - Por ejemplo, el que identifica a uno mismo para poder realizar operaciones con la Administración.
    - Certificados de servidor.
        - Son certificados de la web basados en un protocolo.
    - Certificados de autoridad de certificación.
        - Aseguran que el expedidor es correcto y está autorizado.
    - Certificado para la firma de código.
        - Lo llevan los programas para verificar que el software es verdadero, que no lleva virus, etc.

SESIÓN.

- Es la duración de una conexión a un determinado sistema o red.
- Suele incluir intercambio de paquetes de información entre un usuario y un servidor.
- Para iniciar una sesión normalmente es necesario identificarse.
- La información asociada a la conexión se guarda en el servidor.

SISTEMAS DE CONTROL DE VERSIONES.

- **Concepto**.
    - Software que ayuda a realizar un seguimiento de los cambios realizados en el código a lo largo del tiempo.
    - A medida que se edita código, el sistema de control de versiones toma una instantánea de los archivos, que guarda permanentemente para que se pueda recuperar si fuese necesario.
    - Sin control de versiones, se suele tener varias copias del código en un equipo con distintas variantes o modificaciones. Un cambio incorrecto o la pérdida de uno de estos archivos podría suponer perder el trabajo.
    - Para solucionar esto, los sistemas de control de versiones gestionan todas las versiones del código mostrando al usuario una sola versión a la vez.
- **Terminología.**

- **Repositorio.**
  - Ubicación centralizada usada para almacenar, organizar y compartir archivos, datos o programas.
  - Ubicaciones:
    - **Local.**
      - El almacén o repositorio está alojado en un equipo local.
    - **Remoto**.
      - El almacén o repositorio esta almacenado en uno o varios servidores a los que se accede a través de una red.
  - Repositorios en la nube para código**:**
    - Github, Gitlab, Source Forge, Bitbucket, Gitkraken o Cloud Source Repositories, entre otros.
- **Rama (branch)**
  - Línea de desarrollo independiente aislada de la principal, o de otras ramas, para que no afecten sus cambios a éstas, o no se vea afectada por los cambios en ellas.
  - En realidad, son copias de los archivos de un proyecto que se mantienen aisladas de los archivos de repositorio principal, pero siguen ancladas a él.
  - Tras trabajar con una rama secundaria, ésta se fusiona a la principal.
  - **Ramificación**.
    - Proceso de creación de ramas.
- **Bifurcación (fork).**
  - Son copias de los archivos de un proyecto que se separan de éste manteniéndose independientes y no volviéndose a fusionarse con él.
- **Fusión (merge).**
  - Integración o combinación de 2 ramas separadas en una sola.
- **Etiquetado (tag).**
  - Identificación de varios archivos con una etiqueta.

Tipos de sistemas de control de versiones.

- **Local.**
  - El seguimiento de los cambios de los archivos se realiza dentro del equipo local.
- **Centralizado.**
  - El seguimiento de los cambios de los archivos se realiza en un servidor centralizado.
  - El servidor centralizado incluye el repositorio con toda la información de los archivos versionados y la lista de usuarios que trabajan con ese repositorio centralizado.
- **Distribuido.**
  - Los clientes clonan completamente el repositorio incluido su historial completo, por lo que el repositorio se encuentra en todos los equipos clientes que lo han clonado, además de en uno o varios servidores en la nube.
  - Sistema tolerante a fallos, ya que si algún cliente pierde el repositorio o el servidor falla o está inactivo por un tiempo determinado, cualquiera de los repositorios de los clientes se puede copiar al servidor para restaurarlo o desde éste se puede clonar de nuevo al cliente.
  - Cada clon es una copia de seguridad completa de todos los archivos.

Software de control de versiones.

- **Modelo centralizado.**
  - **CVS.**
    - Sistema de control de versiones muy popular.
    - Es un modelo de repositorio cliente-servidor o centralizado, donde varios desarrolladores pueden trabajar en el mismo proyecto en paralelo.
    - El cliente CVS mantiene actualizada la copia de trabajo del archivo y requiere intervención manual sólo cuando ocurre un conflicto de edición.
  - **Apache Subversion (SVN).**
    - Es un modelo de repositorio cliente-servidor o centralizado, donde los directorios están versionados junto con las operaciones de copia, eliminación, movimiento y cambio de nombre.
- **Modelo distribuido.**
  - **GIT.**
    - Una de las herramientas de control de versiones más utilizadas.
    - Es un modelo de repositorio distribuido compatible con sistemas y protocolos existentes como HTTP, FTP, SSH.
    - Capaz de gestionar eficientemente tanto proyectos pequeños, como grandes.
  - **Mercurial.**
    - Herramienta distribuida de control de versiones escrita en Python y destinada a desarrolladores de software.
    - Alto rendimiento y escalabilidad con capacidades avanzadas de ramificación y fusión.
    - Desarrollo colaborativo totalmente distribuido.
    - Además, posee una interfaz web integrada.
  - **Monotone.**
    - Herramienta distribuida de control de versiones escrita en C ++.
    - Buen apoyo para la internacionalización y localización.
    - Utiliza un protocolo personalizado eficiente y robusto denominado Netsync.

GIT.

Concepto.

- Herramienta para el control de versiones.

Funcionamiento básico.

- Crear un repositorio en la nube.
- Copia o clona el repositorio al equipo local.
- Cambios:
  - Añade carpetas, archivos o modificaciones a éstos dentro del repositorio local y confirma (commit) los cambios.
  - Envía (push) los cambios a la rama principal del servidor.

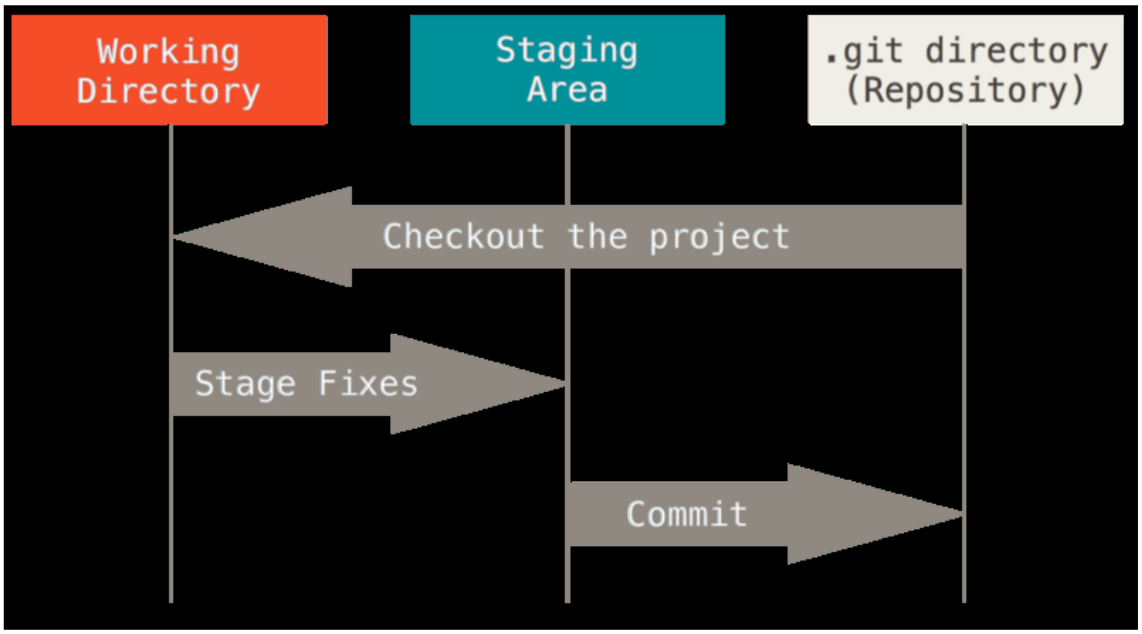
    - Añade carpetas, archivos o modificaciones a éstos dentro del repositorio remoto y confirma (commit) los cambios.
    - Copia (pull) los cambios a tu máquina local.
- Si es necesario, crea rama (branch), haz en ellas cambios y confírmalos.
- Solicitar añadir cambios a la rama principal (pull request)".
- Fusiona ("merge") la rama con la principal.

Instalador.

- https://git-scm.com/

Etapas o estados del flujo de trabajo en GIT.

- Para gestionar un archivo con GIT, éste pasa por los siguientes estados:
    - **Working directory.**
        - Directorio de trabajo.
        - Repositorio o carpeta donde se crean y modifican los archivos.
        - En este directorio están los archivos que no están versionados y no tienen seguimiento.
    - **Staging area**. (add)
        - Área temporal o intermedia de trabajo o ensayo donde está un archivo antes de confirmar los cambios**.**
        - A ella se añade el contenido modificado para empezar su seguimiento y el registro de sus cambios.
    - **Repositorio local**. (commit)
        - Si se confirman los cambios los archivos pasan al repositorio, que es donde está el archivo versionado.
        - A esta zona se añade el contenido de la staging área y se realiza una instantánea.
        - Si se sigue trabajando y se realizan más instantáneas, se trabaja con la última confirmación realizada (HEAD).
        - En caso de necesitar volver a una versión previa de un archivo, se retrocede a la instantánea que interese.
    - **Repositorio remoto**.
        - Al igual que uno local, contiene los archivos con su historial de versiones.
        - Se puede subir (push) o copiar a local (pull) el contenido de un repositorio.



COMANDOS GIT.

- Se utilizan desde una terminal, que puede ser la propia que incorpora GIT (git-bash), CMD de Windows o cualquier otra.
- Uso con Visual Studio Code.
    - Posicionarse en la carpeta que contiene los archivos y el general el proyecto a versionar.
    - Utilizar los comandos GIT desde una consola o terminal.
    - Abrir consola en VSC.
    - Terminal / Nuevo Terminal.

COMANDOS.

- **Registro de usuarios.**
    - Para registrar un usuario en GIT se usa su nombre y su cuenta de correo.
    - Sintaxis:
        - git config --global user.name “Nombre de usuario”
        - git config --global user.email “Correo electrónico del usuario”
    - Si no se registrase bien, no se podrán usar algunas funciones.
    - Otra opción:
        - Editar el archivo config, que se encuentra en la carpeta oculta .git.
        - Tras editarlo, incluir en él las siguientes líneas de código:

```
[user]
name = Nombre del usuario
email = dirección de correo electrónico
```

- **Visualizar usuario registrado y su correo electrónico.**
    - Nombre del usuario:
        - Sintaxis:
            - git config user.name
    - Correo electrónico:
        - Sintaxis:
            - git config user.email
- **Visualizar configuración.**
    - Muestra la configuración por defecto, más los ajustes que haya realizado el usuario.
    - Sintaxis:
        - git config --list.
    - Para salir de config, pulsar *q*.
- **Versión.**
    - Muestra la versión de GIT instalada.
    - Sintaxis:
        - git version.
- **Ayuda.**
    - Muestra los comandos y sus parámetros, que pueden usarse en modo consola con GIT Bash o CMD.
    - Sintaxis:
        - git help
- **Crear un repositorio en un directorio existente.**
    - Permite establecer una carpeta como repositorio en GIT.
    - Relaciona los archivos y carpetas con un repositorio determinado.
    - Dentro de la carpeta se crea la carpeta oculta .git.
    - El comando se ejecuta una sola vez por repositorio a establecer.
    - Sintaxis:
        - git init
    - Pasos:
        - Colocarse en la carpeta a convertir.
        - Ejecutar git init
- **Eliminar un repositorio local.**
    - Borrar la carpeta *.git* de la carpeta que se utiliza como repositorio local.
- **Borrar carpetas y todo su contenido en línea de comandos.**
    - **Pasos:**
        - Posicionarse en la carpeta que incluye la carpeta a borrar.
        - Borrar:
            - **Con git bash.**
                - Abrir git bash como administrador.
                - Ubicarse en la carpeta que contiene el repositorio a borrar.
                - rm -rf nombre de la carpeta a borrar
                - Ejemplo:
                    - rm -rf .git
            - **Con CMD.**
                - rd nombre de la carpeta a borrar /s /q

- **Status**.

- Muestra los archivos sin seguimiento o sin registrar.
- En general, muestra los archivos que se modificaron, eliminaron o crearon.
- Con el parámetro -s muestra solo los archivos sin información adicional, y sólo aquellos que han sido modificados, una vez que todos están ya en proceso de versionado.
- Sintaxis:
  - git status
  - git status -s
- Resultados:
  - Archivos archivos sin seguimiento:
    - Aparecen con interrogaciones delante del nombre.
    - ?? nombre del archivo.extensión.
  - Ejemplo:
    - ??Pagina1.html
    - ??Codigo1.php
  - Archivos añadidos al área temporal de trabajo y por tanto en seguimiento:
    - Muestran una A delante del nombre.
    - A nombre del archivo.extensión.
    - Ejemplo:
      - A estilos.css
  - Archivos modificados:
    - Muestran una m delante del nombre.
    - m nombre del archivo.extensión.
    - Ejemplo:
      - m estilos.css

- **Leyendas en Explorador Visual Studio Code.**
  - **U.**
    - Archivo sin seguimiento.
  - **A.**
    - Archivo registrado o en seguimiento.
    - El archivo ha pasado al área intermedia para seguir sus cambios.
  - **M.**
    - Archivo modificado.
  - **R.**
    - Un archivo ha sido borrado o renombrado y ha pasado del área de trabajo a la de preparación hasta que se confirmen las acciones.
- **Agregar un archivo al área temporal de trabajo.**
  - Permite comenzar a hacer un seguimiento de los cambios que se produzcan en un archivo.
  - Sintaxis:
    - **Agregar un archivo**.
      - git add Nombre del archivo.extensión.
    - **Agregar todos los archivos**.
      - git add .
      - Usando el punto, se añade al índice cualquier archivo nuevo o cualquier archivo que haya sido modificado.
- **Crear una instantánea.**

- Para tener distintas versiones de los archivos de un proyecto en seguimiento, hay que crear copias de un instante determinado de éstos para poder volver atrás en el tiempo a una versión concreta que se quiera utilizar.
- Solo se añaden al repositorio local los archivos añadidos al área de trabajo.
- A los archivos sin seguimiento no se le hace una instantánea o copia en el repositorio local.
- Esto crea la instantánea en el repositorio local desde el área intermedia de trabajo.
- Se usa el comando commit, que guarda los cambios en el repositorio.
- Si hay algún cambio antes de usar commit hay que utilizar add, es decir, no se puede guardar el cambio en el repositorio local, antes hay que agregarlo al área intermedia de trabajo.
- Sintaxis:
  - git commit
  - git commit -m “Mensaje”
    - Con el parámetro -m se puede incluir un texto descriptivo que posteriormente puede ayudar para reconocer cada versión.
- Ejemplo:
  - git commit -m “Primer cambio 01-07-2022”

- **Crear una instantánea sin agregar los cambios al área intermedia.**
  - Se puede crear una instantánea sin pasar por la zona intermedia o índex usando el parámetro -a.
  - Sintaxis:
    - git commit -a -m “Mensaje”
  - Ejemplo:
    - git commit -a -m “Archivo nuevo añadido”
- **Listado de confirmaciones realizadas.**
  - Para ver un listado o el historial de todos los commit que se han realizado ordenados de forma descendente, se utiliza el comando log.
  - Sintaxis:
    - git log
    - git log --oneline
      - Con el parámetro - -oneline sólo se muestran los textos o mensaje asociado a los distintos commit realizados.
  - Resultados:
    - Se muestra un código o identificador distinto para cada commit, que se empleará para volver a una versión o instantánea previa.
    - El último commit realizado, aparece en primer lugar con la expresión [HEAD -> rama]
    - Es el HEAD que apunta al último commit realizado en la rama actual.
- **Consultar el historial completo.**
  - Para ver el historial completo de cambios realizados en el repositorio local e identificar el hash (identificador) del commit que tiene el cambio al que queremos volver.
  - Permite encontrar una referencia anterior a la ejecución de reset.
  - Sintaxis:
    - git reflog
- **Volver a una versión previa. (Retroceder en el tiempo).**

- Para volver a un punto previo en el que un archivo o un proyecto tenía un contenido diferente al actual, se usa el código identificador de cada instantánea commit realizado.
- Sintaxis:
  - git reset --hard Código o Identificador.
- Ejemplo:
  - git reset --hard abf66n4.

- **Volver a una versión posterior. (Avanzar en el tiempo).**
  - Igualmente, se usa el código identificador de la instantánea a la que se quiere volver
  - Sintaxis:
    - git reset --hard Código o Identificador.
  - Ejemplo:
    - git reset --hard 145ju03.
- **clear.**
  - Permite limpiar código de la terminal git bash
  - Sintaxis:
    - clear.
- **cls.**
  - Permite limpiar código de la terminal CMD y Powershell.
  - Sintaxis:
    - cls.
- **show**.
  - Muestra los cambios confirmados y que se hayan realizado en el último archivo utilizado.
  - Muestra el último commit, incluyendo información adicional como autor, nombre del commit, fecha, etc.
  - Se muestra en color rojo el contenido cambiado y en verde el cambio realizado.
  - Sintaxis:
    - git show
- **mv.**
  - Sirve para cambiar el nombre a un archivo.
  - Sintaxis:
    - git mv Nombre original del archivo.extension Nombre nuevo del archivo .extension
  - Ejemplo:
    - git mv index.html index2.html
- **rm.**
  - Permite eliminar un archivo.
  - Se puede usar el asterisco (*) como comodín para eliminar conjuntos de archivos que sigan un patrón.
  - Sintaxis:
    - git rm ruta/nombre del archivo a eliminar.
  - Ejemplo:
    - git rm pagina1.html
    - git rm documentos/info.pdf
    - git rm pagina1.*

        - git rm *.css
- **Clonar un repositorio.**
    - Sirve para descargar el contenido de un repositorio remoto al local.
    - Útil cuando en caso de que se haya deteriorado o borrado accidentalmente los archivos en local.
    - Necesario para un nuevo miembro de un equipo que necesite los archivos de un determinado repositorio para trabajar con ellos.
    - Sintaxis:
        - git clone URL al repositorio.
    - Ejemplo:
        - git clone https://github.com/juanPHP/cosasPHP.git
- **Respaldo de archivos en la nube - Repositorio en la nube.**
- **Conexión a un repositorio remoto.**
    - Crear una cuenta en un repositorio en la nube como, por ejemplo, en github.
    - Asociarlo al repositorio local para guardar en remoto los archivos y carpetas del proyecto que interese.
    - Esta acción se realiza una sola vez, a no ser que se quieren almacenar o subir a remoto otros proyectos a otros repositorios.
    - Sintaxis:
        - git remote add origin URL
    - Ejemplo:
        - git remote add origin https://github.com/juan_proyecto23/repositorio23.git
- **Actualizar la versión remota de un repositorio desde otro local.**
    - Para ello hay que subir los archivos y sus modificaciones al repositorio.
    - Acción que hay que realizar también cada vez que se realicen cambios y sus commit correspondientes.
    - La primera vez hay que identificarse en el sitio web en el que tengamos alojado nuestro repositorio.
    - Sintaxis:
        - **Crear una rama principal en el repositorio remoto**:
            - git branch -M main
        - **Subir el contenido por primera vez**.
            - git push -u origin main
        - **Subir contenido por segunda o siguientes veces**.
            - git push rama
            - git push
                - En este caso, si ha producido un cambio en un archivo, primero hay que añadir los archivos cambiados al área temporal, realizar un commit y después usar push.
                - Si no se especifica, los archivos y sus cambios se envían a la rama en uso, si no, a la rama que se indique.
- **Eliminar un origen remoto del repositorio**.
    - Si cambiamos de repositorio o hay algún problema en la conexión, se puede eliminar el origen remoto y volver a crear la conexión:
    - Sintaxis:
        - git remote remove origin
- **Cambiar un origen remoto por otro**.

- Otra opción para crear una conexión con el repositorio remoto sin eliminar la conexión previa es cambiarla.
- Sintaxis:
  - git remote set-url origin nueva URL
- Ejemplo:
  - git remote set-url origin https://github.com/proyectos/trabajo.git

- **Actualizar la versión local de un repositorio desde otro remoto.**
  - Para copiar en local los cambios realizados en remoto se usa el comando pull, es decir se actualiza el directorio local al commit remoto más nuevo.
  - Es un comando combinado, que equivale a git fetch + git merge.
  - Comprueba si hay cambios en el repositorio remoto y, en caso de que los haya, se actualizan los archivos en el repositorio local para sincronizar la rama remota con la local.
  - Sintaxis:
    - git pull
    - git pull --verbose
    - Con la opción verbose se produce una salida más detallada de los cambios que se están realizando.
- **Etiquetas.**
  - Es posible etiquetar commit o puntos específicos del historial como importantes.
  - Se usa dar nombre a los commit o para indicar versiones de la aplicación o del proyecto.
  - **Creación de una etiqueta**.
    - **Asociándola a la instantánea de cabecera**
      - Para crear una etiqueta se usa el comando tag.
      - Sintaxis:
        - git tag etiqueta -m “Descripción de la etiqueta”
      - Ejemplo:
        - git tag version1.0 -m “Versión 1.0”
    - **Asociándola a cualquier instantánea.**
      - Sintaxis:
        - git tag etiqueta Identificador de la instantánea a al que se quiere asociar.
      - Ejemplo:
        - git tag cambio1 3cdcb18
  - **Listado de etiquetas**.
    - Para ver un listado de todas las etiquetas creadas:
    - Sintaxis:
      - git tag.
  - **Información sobre etiquetas**.
    - Para visualizar información sobre una etiqueta en particular, se usa el comando show.
    - Sintaxis:
      - git show nombre de la etiqueta
    - Ejemplo:
      - git show version1.0
  - **Subir etiquetas a remoto**.

        - Hay que subirlas manualmente.
        - Una etiqueta:
            - Sintaxis:
                - git push origin nombre de la etiqueta
            - Ejemplo:
                - git push origin version1.0
        - Todas las etiquetas:
            - Sintaxis:
                - git push - -tags
    - **Ir a una etiqueta**.
        - Para desplazarse a una etiqueta, y por tanto al commit para el cual se ha creado, se usa el comando checkout.
        - Equivale a volver a un punto previo en el historial.
            - Sintaxis:
                - git checkout Nombre de la etiqueta
            - Ejemplo:
                - git checkout version1.0
    - **Eliminar una etiqueta**.
        - Para eliminar una etiqueta se usa el parámetro -d.
        - Sintaxis:
            - git tag -d Nombre de la etiqueta
        - Ejemplo:
            - git tag -d version1.0
- **Ramas.**
    - **Crear una rama nueva.**
        - Sintaxis:
            - git branch Nombre de la rama.
        - Ejemplo:
            - git branch secundaria1
    - **Crear una rama nueva y cambiarse a ella.**
        - Sintaxis:
            - git branch -b Nombre de la rama.
        - Ejemplo:
            - git branch -b secundaria2
    - **Visualizar en que rama estamos.**
        - Sintaxis:
            - git branch
    - **Cambiar a otra rama.**
        - Sintaxis:
            - git checkout Nombre de la rama a la que se cambia.
        - Ejemplo:
            - git checkout secundaria2
    - **Subir la nueva rama al repositorio.**
        - Sintaxis:
            - git push origin Nombre de la rama.
        - Ejemplo:
            - git push origin secundaria2

- **Eliminar una rama.**
  - Sintaxis:
    - git branch -d Nombre de la rama a eliminar.
  - Ejemplo:
    - git branch -d secundaria1.

- **Fusión de ramas.**
  - Unión o combinación de 2 ramas.
  - La fusión se realiza entre las últimas instantáneas de cada rama.
  - Para combinar 2 ramas entre sí, se utiliza el comando merge.
  - Para unir 2 ramas, hay que colocarse en la principal (master) y luego unirlas
  - Sintaxis:
    - git checkout Nombre de la rama principal
    - git merge Nombre de la rama a fusionar con la principal -m "Mensaje"
  - Ejemplo:
    - git checkout master
    - git merge secundaria1 -m "Fusión de rama secundaria1 con la principal"
  - Conflictos.
    - Al fusionar 2 ramas se pueden producir conflictos y no realizarse correctamente la fusión.
    - El conflicto se resuelve aplicando los cambios de una rama solo, los de la otra, los de ambas o ninguno.
    - Podría ser necesario editar los archivos que se muestra en conflicto y luego añadirlos a la rama fusionada con:
      - git add nombre de archivo.extension
- **Comprobar ramas fusionadas.**
  - Para mostrar que ramas se ha fusionado:
    - git log --all –graph
    - git show-branch

PRUEBAS DE SOFTWARE. (SOFTWARE TESTING).

- **Concepto.**
  - Proceso de evaluación y verificación de un determinado software para comprobar que funciona correctamente y hace aquello para lo que ha sido creado.
  - Antes de poner en producción una aplicación es necesario probarla para comprobar su correcto funcionamiento. De esta forma se puede confirmar que se obtienen los resultados correctos tanto en valor como en formato, que todos los enlaces de un sitio web son funcionales o que los módulos, métodos, clases etc., se ejecutan correctamente.
  - Otro de los parámetros que suele evaluarse es la seguridad.
- **Tipos.**
  - **Estáticas.**
  - **Dinámicas.**
    - **Funcionales**
      - Pruebas unitarias.

        - Pruebas de componentes.
        - Pruebas de integración.
        - Pruebas de sistema.
        - Pruebas de humo.
        - Pruebas alpha.
        - Pruebas beta.
        - Pruebas de aceptación.
        - Pruebas de regresión.
    - **No funcionales.**
        - Pruebas de compatibilidad.
        - Pruebas de seguridad.
        - Pruebas de estrés.
        - Pruebas de usabilidad.
        - Pruebas de rendimiento.
        - Pruebas de internacionalización y localización.
        - Pruebas de escalabilidad.
        - Pruebas de mantenibilidad.
        - Pruebas de instalabilidad.
        - Pruebas de portabilidad.

ESTÁTICAS.

Análisis estático

- Conjunto de pruebas que para su aplicación no requieren la ejecución de un programa.
- Se usa para detectar reglas que no se cumplen, o errores léxicos, sintácticos e incluso algunos semánticos.
- También se emplea para encontrar vulnerabilidades en el código como:
    - **Inyección SQL**.
        - Un atacante usa un fragmento de código SQL para manipular una base de datos y acceder a información potencialmente valiosa.
    - **Ataque XSS o scripting entre sitios**.
        - Inclusión de un script malicioso en un sitio web que luego es enviado junto con el contenido legitimo del sitio, al navegador de la víctima.
        - Suelen realizarse con JavaScript.
    - **Desbordamiento de buffer**.
        - Se produce cuando la cantidad de datos en el buffer (espacio en la memoria en la que se almacenan los datos de forma temporal), excede su capacidad de almacenamiento. Los datos se desbordan ocupando posiciones de memoria adyacentes que sobrescriben o corrompen los datos que hubiera en ellas, lo que podría dar lugar a alteraciones y fallos en la ejecución de un programa, es decir, dar lugar a vulnerabilidades, que podrían aprovecharse por parte de un atacante.
        - Así, un atacante, al mandar más datos de los que caben en el bloque de memoria asignado, puede sobrescribir datos en otras partes de la memoria con el objetivo de:

- Bloquear un programa.
- Inyectar código malicioso.
- Cambiar el flujo de ejecución de un programa.

- **Herramientas.**
  - SonarQube, Embold, Collabolator, PVS-Studio, Helix QAC, Veracode, PMD, ESlint, Ideone, Rextester y otras.
- **Otras herramientas para lenguajes de Programación.**
  - https://rextester.com/l/fortran_online_compiler
  - https://ideone.com/
  - https://www.writephponline.com/
- **Otras herramientas para HTML, CSS y JavaScript.**
  - https://jsfiddle.net/
  - https://liveweave.com/#
  - https://validator.w3.org/nu/
- **Otras herramientas para JavaScript.**
  - https://playcode.io/

DINÁMICAS.

- Todas aquellas pruebas que para su ejecución requieren la ejecución de la aplicación.
- Las pruebas dinámicas permiten el uso de técnicas de caja negra y caja blanca con mayor amplitud. Debido a la naturaleza dinámica de la ejecución de pruebas es posible medir con mayor precisión el comportamiento de la aplicación desarrollada.
- **Tipos.**
  - **Pruebas funcionales.**
  - **Pruebas no funcionales.**

PRUEBAS FUNCIONALES.

- Se emplean para confirmar si una aplicación produce resultados correctos, tiene algún fallo o si le falta algo especificado en los requisitos del software.
- No se emplean para comprobar cómo se produce el procesamiento y no prueban todo un sistema, si no solo algunas características de éste.
- **Tipos.**
  - **Pruebas de aceptación.**
    - Verifican si todo el sistema funciona según lo previsto.
  - **Pruebas de integración.**
    - Garantizan que los componentes, módulos o las funciones individuales están conectados y funcionan conjuntamente.
  - **Prueba de humo.**
    - Permiten verificar si las funcionalidades más significativas de la aplicación funcionan o no.
    - De forma que lo más básico del software se ejecute de forma correcta con pruebas sencillas y rápidas.
    - Son una revisión rápida para comprobar el funcionamiento básico del software.
  - **Pruebas beta.**

- Permiten que algunos clientes reales prueben el software pruebas para asegurarse de que éste o alguna nueva actualización funciona correctamente antes de que el software sea usado por todo el mundo.

- **Pruebas de rendimiento.**
  - Comprueban cómo funciona el software bajo diferentes cargas de trabajo. Las pruebas de carga, por ejemplo, se utilizan para evaluar el rendimiento en condiciones de carga reales.
- **Pruebas de regresión.**
  - Comprueban si nuevas características (código nuevo, corrección de errores, actualizaciones), no rompen o degradan la funcionalidad del software.
- **Pruebas unitarias o Unit Test**
  - Estas pruebas están diseñadas para comprobar el correcto funcionamiento una unidad de código.
  - Una unidad de código es la parte más pequeña a de una aplicación, por ejemplo, una función o un método.
  - Su objetivo es aislar una parte del código para probar que funciona adecuadamente.
  - Sólo deben acceder a las unidades de código para la que han sido desarrolladas, no a otras para las que no se han diseñado, ni a otras partes del sistema.
- **Otra clasificación.**
  - **Pruebas de caja negra.**
    - Pruebas sin acceso al código fuente.
    - Comprueban la respuesta del software a varias entradas bajo un cierto estado interno de los programas.
  - **Pruebas de caja blanca.**
    - Pruebas con acceso al código fuente y a los documentos de diseño.
    - Su objetivo en comprobar cómo funciona internamente una aplicación y si lo hace como se espera.
    - Algunos tipos de pruebas de caja blanca:
      - Pruebas de flujo de control.
      - Pruebas de flujo de datos:
        - Seguimiento de variables a lo largo de las rutas de ejecución.
      - Pruebas de bifurcación.
      - Pruebas de caminos básicos:
        - Explorar distintas rutas de ejecución.

PRUEBAS NO FUNCIONALES.

- Se centran en medir el rendimiento y la usabilidad del software o de las aplicaciones, más que en si éstas funcionan.
- **Tipos.**
  - **Revisión del código. (Revisión entre pares).**
    - Confirman que el software nuevo y modificado esté siguiendo los estándares de codificación de una organización y se adhiera a sus mejores prácticas.
    - También se denomina revisión entre pares, ya que 2 desarrolladores (autor y revisor), comprueban el código.

- **Pruebas de estrés.**
  - Prueban cuánta tensión puede soportar el sistema antes de fallar.
- **Pruebas de usabilidad.**
  - Validan lo bien que un usuario puede utilizar un sistema o una aplicación web para completar una tarea.
- **Pruebas de accesibilidad.**
  - Evalúan si el software es accesible para personas con discapacidades.
- **Pruebas de seguridad.**
  - Comprueban que el software no esté expuesto a vulnerabilidades y ataque maliciosos que puedan hacer que no funcione, o lo haga de forma incorrecta.
- **Pruebas de fiabilidad.**
  - Evalúan la fiabilidad del software y aseguran que éste puede realizar continuamente sus funciones sin fallos.
- **Pruebas de supervivencia.**
  - Garantizan que, en caso de errores y fallos, el software va a responder adecuadamente, va a minimizar la perdida de datos y se va a recuperar adecuadamente.
- **Pruebas de disponibilidad.**
  - La disponibilidad del software hace referencia al grado en que el usuario puede depender del sistema durante su funcionamiento.
  - También se denominan pruebas de estabilidad.
- **Pruebas de escalabilidad.**
  - Comprueban hasta qué punto una aplicación puede aumentar su capacidad de procesamiento para responder a una demanda creciente.
- **Pruebas de Portabilidad.**
  - Se utilizan para comprobar la flexibilidad y facilidad para transferir el software desde su entorno actual de hardware o software a otro.

DESARROLLO DIRIGIDO POR PRUEBAS (TDD)

- Metodología en la que las pruebas se escriben antes que el código con el objetivo de conseguir:
  - Un diseño de software más reflexivo.
  - Mayor cobertura de pruebas.

DOCUMENTACIÓN TÉCNICA.

- Parte importante en la ingeniería de software, ya que explica cómo funciona éste y cómo debe usarse.
- Esencial en cada etapa del trabajo, durante todo proceso de desarrollo, es decir, no hay que esperar a terminar un trabajo para hacerla, sino que su creación va en paralelo con el proceso de desarrollo.

Destinatarios de la documentación.

- **Arquitectura / Diseño.**
  - Descripción general del software.
  - Incluye las relaciones con otros programas, componentes y/o hardware, así como las pautas que se utilizarán en el diseño y producción del software.

- **Técnico.**
  - Documentación sobre:
    - Código.
    - Algoritmos.
    - Interfaces.
    - Interfaces de programación (API).
- **Usuario.**
  - Manuales para:
    - Usuarios.
    - Administradores de sistemas.
    - Personal de soporte.
- **Marketing.**
  - Instrucciones de producto y garantía promocional.

Tipos de documentación.

- **Documentación técnica.**
  - Incluye las especificaciones técnicas del producto, así como los requisitos de funcionamiento.
- **Documentación de soporte.**
  - Cuya función es resolver y explicar los problemas más comunes.
- **Documentación general.**
  - Información sobre una aplicación y sus funciones.
- **Documentación de ayuda.**
  - Utilizada para explicar al usuario el manejo.
- **Documentación legal.**
  - Que incluye las condiciones legales de uso de la aplicación.
- **Manual de usuario.**
  - Documento o guía que ayuda a entender el funcionamiento de algo.
  - Su función es dar asistencia técnica a las personas que utilizan la aplicación, software, sistema, etc., para el cual se ha creado el manual.
  - Puede incluir también:
    - FAQ's (*Frecuently Asked Questions*).
    - Guías de Uso e Instalación.
    - Glosario.
    - Otros.

Soportes.

- **Soporte papel.**
  - Cuando la información se incluye en documentos impresos**.**
- **Soporte informático.**
  - Cuando se incluye en documentos digitales que funcionan como manuales.
- **Soporte físico.**
  - Si se incluye en discos (CD's, DVD's, etc.), memorias USB, etc.
- **Soporte online.**
  - Cuando la documentación o la ayuda debe ser consultada través de Internet.

Herramientas de documentación técnica.

- **Confluence.**
  - Herramienta para gestionar documentación.
  - Con aspecto similar a la Wikipedia, permite compartir en formato web, textos e imágenes que ayudan a informar y documentar.
  - https://www.atlassian.com/es/software/confluence
- **Read the Docs.**
  - Plataforma gratuita para el alojamiento de documentación de software con código fuente disponible gratuitamente.
  - Facilita la redacción de documentación técnica al automatizar la creación y el control de versiones.
  - Puede generar documentación automáticamente para el contenido de repositorios ubicados en github, gitlab, etc.
  - https://readthedocs.org/
- **Javadoc.**
  - Generador de documentación creado por Sun Microsystems para el lenguaje Java.
  - Genera documentación en formato HTML a partir del código fuente de Java.
  - https://www.oracle.com/es/technical-resources/articles/java/javadoc-tool.html
- **Markdown.**
  - Herramienta de conversión de texto plano a HTML.
  - Uso también para formatear archivos README, escribir mensajes en foros de discusión online o para crear texto simple.
  - https://markdown.es/
- **Swagger.**
  - Herramienta útil durante el desarrollo de todo el ciclo de vida de una API, desde el diseño y la documentación, hasta la prueba y la implementación.
  - https://swagger.io/