

# Ejercicios Tema 1

DWES

ALBERTO MÉNDEZ NÚÑEZ



## Contenido

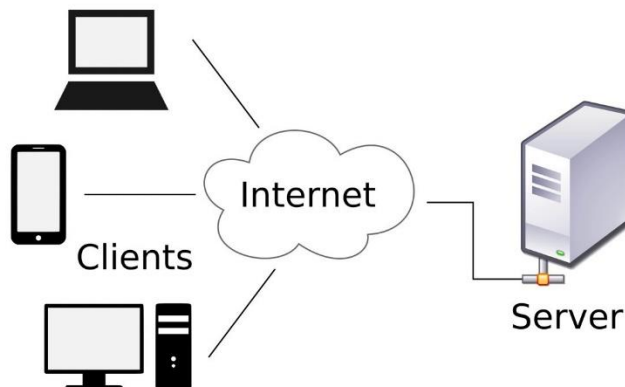
1. Protocolos de comunicaciones: IP, TCP, HTTP, HTTPS.....	2
2. Modelo de Comunicaciones Cliente-Servidor.....	3
3. Métodos de petición HTTP/HTTPS más utilizados.....	4
5. Modelo de desarrollo de aplicaciones multicapa.....	6
6. División funcional Front-End y Back-end.....	7
7. Página web estática, dinámica, aplicación web y mashup.....	8
8. Componentes de una aplicación web.....	9
9. Programas del lado del cliente y del lado del servidor.....	10
10. Lenguajes del lado del servidor de una aplicación web.....	11
11. Características y posibilidades de desarrollo de una plataforma XAMPP.....	12
XAMPP es una plataforma gratuita y multiplataforma que integra los componentes necesarios para desarrollar y probar aplicaciones web de manera local. Incluye Apache (servidor web), MySQL/MariaDB (base de datos), PHP y Perl.....	
12. En qué casos se instala (JVM) y JDK.....	12
13. IDE más utilizados (características y grado de implantación actual).....	13
14. Servidores HTTP /HTTPS más utilizados (características y grado de implantación actual).....	13
<b>Apache HTTP Server:</b> de código abierto, multiplataforma y muy flexible gracias a su sistema de módulos, es el servidor web más tradicional y popular. Es ligero y fácil de instalar y configurar por lo que funciona bien en prácticamente cualquier sistema operativo.....	
<b>Nginx:</b> más moderno y rápido que Apache HTTP, haciendo que sea muy eficaz para sitios web con mucho trafico, gestionando asi las conexiones de manera mas eficiente. Muy popular en los últimos años.....	
<b>Microsoft IIS:</b> es el servidor web de Microsoft, está integrado en los sistemas operativos Windows Server, su implantación es muy alta ya que las empresas tienden a utilizar productos de empresas fiables y conocidas como Microsoft.....	
<b>LiteSpeed:</b> es un servidor web gratuito, de código abierto y alto rendimiento, ofrede un rendimiento similar a NGINX. Su implantación es alta.....	
15. Apache HTTP vs Apache Tomcat.....	14
16. Navegadores HTTP /HTTPS más utilizados.....	15
17. Generadores de documentación HTML (PHPDoc).....	16
18. Repositorios de software – Sistemas de ctrl de versiones.....	16
19. Configuración del entorno de desarrollo.....	17
20. Configuración del entorno de explotación.....	17
21. Realizar un estudio sobre los siguientes conceptos.....	18
22. Elegir y realizar un estudio y una presentación para la exposición del trabajo sobre una de las siguientes arquitecturas de desarrollo de Aplicaciones Web:.....	19

## 1. Protocolos de comunicaciones: IP, TCP, HTTP, HTTPS.

- **IP: (Internet Protocol).** Existen varios tipos:
  - IP Privada:** es un número único que identifica tu dispositivo en una red privada como la de una clase, casa o un trabajo, el cual se lo asigna el router a cada dispositivo de la red interna.
  - IP Pública:** al igual que la IP Privada, es un número único que nos identifica en este caso en internet. El ISP (Proveedor de servicios de internet) nos asigna una de su grupo de IPs disponibles.
  - IP Estática:** es una dirección que en el momento en el que la asignemos no va a cambiar nunca dentro nuestra red interna. (Se puede elegir dentro de unos rangos teniendo en cuenta que parte identifica la red y el host).
  - IP Dinámica:** es una dirección la cual no podemos mantener de forma recurrente, se va asignando una diferente cada vez que nos conectamos a una red. Esto ocurre gracias al servidor DHCP que nos asigna una ip diferente aleatoria dentro de su red a cada dispositivo que se intenta conectar.
- **TCP: (Transmission Control Protocol)** al igual que los protocolos UDP y SCTP, un protocolo de Internet que está ubicado en la capa de transporte del modelo OSI. El objetivo de TCP es crear conexiones dentro de una red de datos compuesta por redes de computadoras para intercambiar datos. Además, en cuanto a su funcionamiento, garantiza que los datos llegarán a destino sin errores y en el mismo orden en el que fueron transmitidos.
- **HTTP: (Hypertext Transference Protocol)** principal protocolo utilizado para enviar datos entre un navegador web y un sitio web. Su puerto por defecto es el 80.
- **HTTPS: (Secure Hypertext Transference Protocol)** Es una versión más segura de HTTP que cifra los datos que se intercambian entre el navegador del cliente y el servidor web. Es importante cifrar esta conexión cuando queremos enviar o recibir datos sensibles. Su puerto por defecto es el 443.

## 2. Modelo de Comunicaciones Cliente-Servidor

### ■ Arquitectura cliente-servidor



En la imagen podemos observar el comportamiento general del modelo cliente servidor y podemos identificar dos partes clave:

- **Cliente:** Con cliente nos podemos referir a un dispositivo o un proceso que en este caso está realizando una petición de un recurso mediante el navegador a un servidor web.
- **Servidor Web:** El servidor web es el encargado de recibir, procesar y si procede responder las peticiones realizadas por el cliente. Esta acción del servidor es la responsable de que cuando nosotros busquemos un recurso en nuestro navegador nos devuelva la página que buscamos. Es importante recalcar que, si el servidor web recibe muchas solicitudes de un mismo recurso seguidas, lo mas probable y lo mas seguro es que el servidor ignore dichas peticiones ya que podrían tratarse de un ataque de denegación de servicio (Dos) o un ataque de denegación de servicio distribuido (DDoS).

### 3. Métodos de petición HTTP/HTTPS más utilizados

Las peticiones HTTP/HTTPS están formadas por un método, una URL y una versión.

Métodos de petición más utilizados:

**HEAD:**

Solo devuelve las cabeceras de la respuesta, sin el cuerpo.  
Comprueba si un recurso existe, su tamaño y última fecha de modificación.

**GET:**

Es el método de petición mas utilizado. Lo utilizamos para solicitar un recurso al servidor. El ejemplo más claro del método GET seria al buscar una URL en nuestro navegador, estaríamos enviando una petición al servidor que aloja dicha página mediante un GET.

**POST:**

Al contrario que el método GET, en este caso lo que hacemos es enviar información al servidor para que la procese o la almacene. Un ejemplo sería al rellenar un formulario de una aplicación web, clicamos en el botón de envío, dicha información que hemos introducido es enviada, procesada y almacenada en el servidor.

**PUT:**

El método PUT reemplaza un recurso ya existente en el servidor, seria como hacer una actualización de algo que el servidor ya tiene almacenado y procesado.

**DELETE:**

Elimina un recurso que especifiquemos del servidor.

**TRACE:**

Devuelve la misma petición que el cliente envió, tal cual llegó al servidor.

**OPTIONS:**

Devuelve los métodos HTTP soportados por un recurso o servidor.

**CONNECT:**

Se usa para establecer un túnel de comunicación con el servidor y comprobar si existe comunicación con el host.

#### 4. Estudio sobre el concepto de URI/URL/URN.



**URI = Identificador general de un recurso.**

Ejemplo de URI:

<https://www.ejemplo.com/recurso?id=123>

https-> protocolo que utiliza.

[www.ejemplo.com](http://www.ejemplo.com)-> dominio.

/recurso-> ruta del recurso que queremos ver.

?id=123-> parámetro de consulta (query string).

**URL = URI que incluye la forma de localizar el recurso.**

Ejemplo de URL:

[https://es.wikipedia.org/wiki/Varanus\\_macraei](https://es.wikipedia.org/wiki/Varanus_macraei)

https-> protocolo que utiliza. (como se accede al recurso)

[www.wikipedia.org](http://www.wikipedia.org)-> dominio (donde está el recurso)

/wiki/Varanus\_macraei-> ruta (recurso específico solicitado)

**URN = URI que solo nombra el recurso de manera única pero no dice como encontrarlo.**

Ejemplo de URN:

urn:isbn:9780140449136

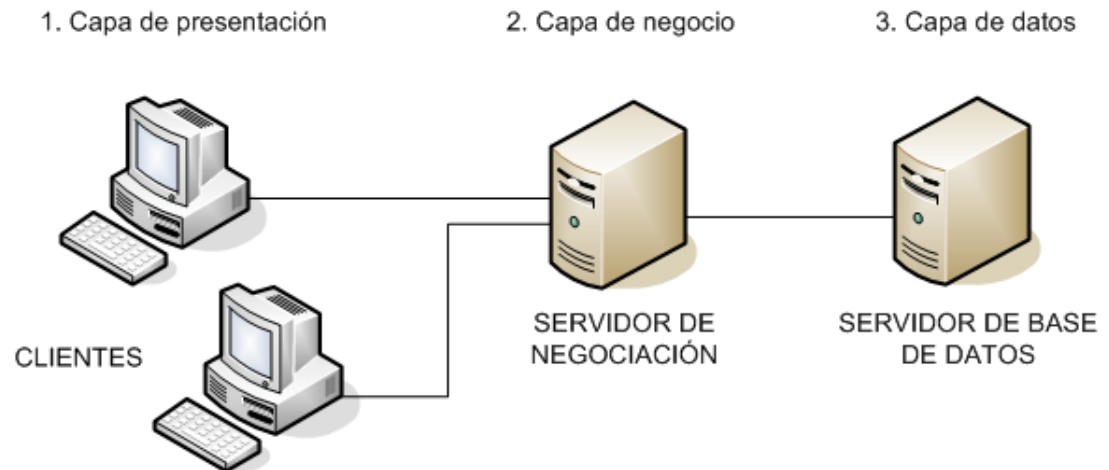
urn-> esquema que indica que es un nombre uniforme de recurso.

Isbn-> espacio de nombres (en este caso, libros identificados por su ISBN).

9780140449136-> identificador único dentro de ese espacio.

Relación con HTTP y HTTPS= El navegador interpreta la URL, usa HTTP o HTTPS, solicita el recurso y devuelve el contenido al usuario.

## 5. Modelo de desarrollo de aplicaciones multicapa.



**- Capa de presentación:**

Sirve para interactuar con el usuario.

Los componentes pueden ser: formularios, pantallas, páginas web, apps.

**- Capa de negocio:**

Es la encargada de las reglas de negocio y procesos de aplicación.

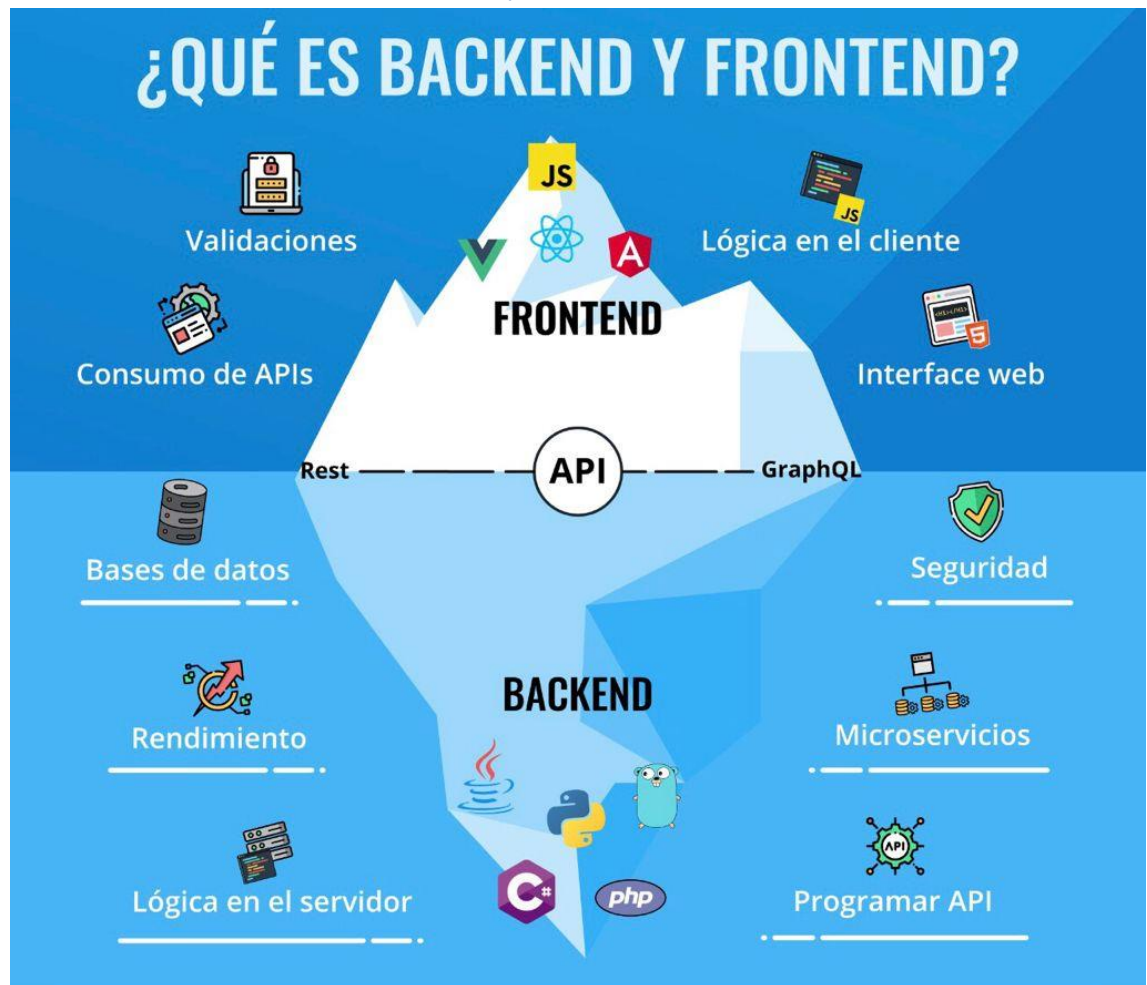
Servicios, controladores, lógica de validación.

**- Capa de datos:**

Gestiona el acceso a la base de datos y sistemas de almacenamiento.

ORM, consultas SQL, repositorios, controladores de BBDD.

## 6. División funcional Front-End y Back-end.



**Front-End:** Es la parte a la que los usuarios corrientes acceden en una página web, por ejemplo, al acceder a la web de Amazon ya sea que hayamos iniciado sesión con una cuenta de cliente normal o hayamos accedido sin iniciar sesión, podremos visitar la página, movernos por los distintos apartados y comprar productos.

**Back-end:** En este caso sería el apartado de la aplicación donde se realizan los cambios como cambiar precios, añadir productos, actualizar información etc. A este apartado solamente pueden acceder los usuarios encargados del mantenimiento de la aplicación.



## 7. Página web estática, dinámica, aplicación web y mashup.

### **Web estática:**

El contenido es fijo, no cambia nada a menos que el programador la modifique.  
Ejemplo: El índice de la asignatura de DWES, está hecho con HTML y CSS y no cambia a no ser que yo lo modifique.

### **Web dinámica:**

El contenido cambia constantemente cada vez que accedemos, por ejemplo, a el País. Su contenido se genera en el servidor cada vez que alguien lo visita.

### **Aplicación web:**

Se asemeja a una página web dinámica pero mucho mas completa. La aplicación web debe tener control de acceso, no puede acceder cualquiera, solamente los usuarios registrados en la aplicación (Diferentes accesos para cada usuario).

### **Mashup:**

Es una aplicación web que consulta y recoge información de varias paginas web para utilizarlo en una sola.

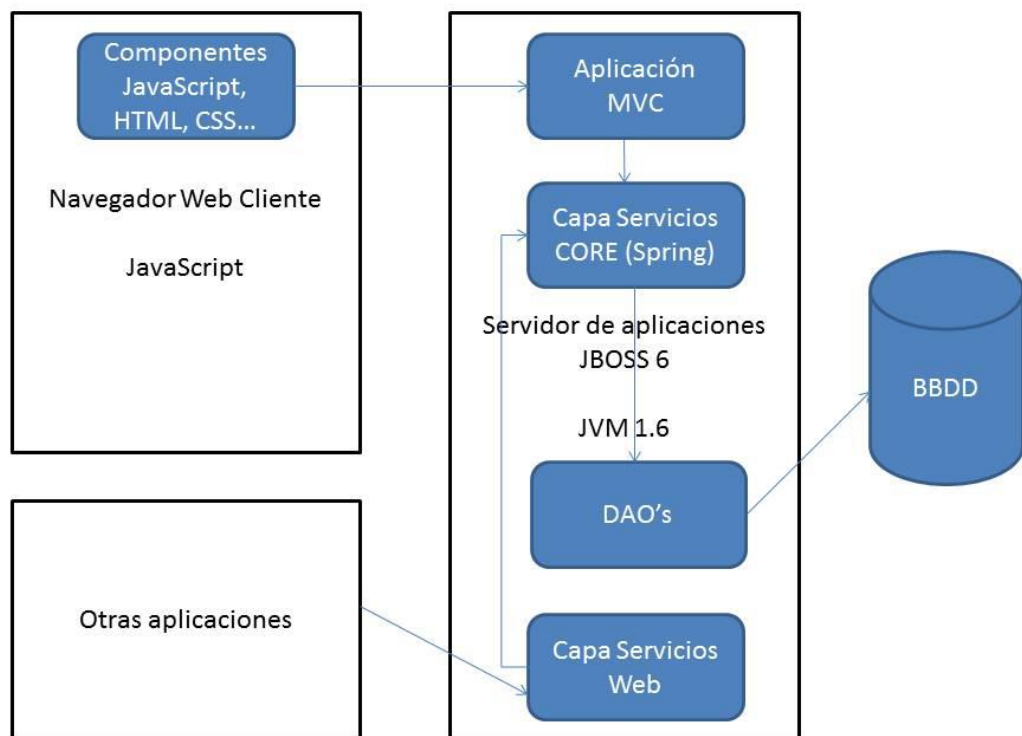
Por ejemplo, una página que recopile diferentes precios de un videojuego de diferentes plataformas de venta online y te muestre los precios ordenados de menor a mayor.

## 8. Componentes de una aplicación web.

**Servidor web (Apache):** Es el encargado de procesar las solicitudes de los clientes y devolver el recurso que solicitan. Algunos populares son Apache, Nginx y Apache Tomcat.

**Servicio que interpreta el código (php8.3-fpm):** Se encarga de procesar los datos y de la autenticación de los usuarios. Nosotros utilizamos php8.3-fpm para programar la funcionalidad de la aplicación.

**Sistema gestor de base de datos (MariaDB):** Almacena y administra la información de la aplicación, además de tener usuarios propios para su administración. Nosotros usamos MariaDB.



## 9. Programas del lado del cliente y del lado del servidor.

### Lado del servidor:

Los programas del lado del servidor son los que se ejecutan en el servidor web y no en el navegador del usuario.

- En primer lugar reciben las solicitudes del cliente, por ejemplo, cuando alguien visita una página o rellena un formulario.
- Después procesan la información, por ejemplo:
  - o Consultar o actualizar una base de datos.
  - o Validar datos del usuario.
  - o Ejecutar lógica de negocio.
  - o Generar contenido dinámico.
- Envían una respuesta al cliente, generalmente en HTML, JSON O XML.
- Los lenguajes más comunes del lado del servidor son PHP, Python, Node.js, Java, Ruby, C#.

### Lado del cliente:

Los programas del lado del cliente son los que se ejecutan directamente en el dispositivo del usuario normalmente en el navegador.

- Se ejecutan en el navegador del usuario.
- Manipulan la interfaz mostrando textos, imágenes, botones etc.
- Validan los datos antes de enviarlos al servidor.
- Pueden comunicarse con el servidor mediante AJAX o fetch() para obtener datos sin recargar la página.
- Mejoran la experiencia del usuario haciéndolo todo mas interactivo y dinámico.
- Los lenguajes más comunes del lado del cliente son HTML, CSS, JavaScript y a veces frameworks como React, Vue o Angular.

## 10. Lenguajes del lado del servidor de una aplicación web.

**PHP:** Lenguaje interpretado, sencillo de aprender y ampliamente usado para crear páginas web dinámicas. Se integra fácilmente con servidores como Apache y sigue siendo muy popular en CMS como WordPress.

**Python:** Interpretado, de sintaxis clara y fácil mantenimiento. Con frameworks como Django y Flask, permite un desarrollo rápido y seguro de aplicaciones web. Su uso ha crecido notablemente en los últimos años.

**Java:** Compilado a bytecode que se ejecuta en la JVM, lo que lo hace portátil entre plataformas. Muy usado en entornos empresariales por su robustez, seguridad y escalabilidad.

**C#:** Lenguaje compilado y orientado a objetos, usado principalmente con .NET. Ofrece gran integración con herramientas Microsoft y, gracias a .NET Core, es actualmente multiplataforma.

**JavaScript (Node.js):** Permite usar JavaScript en el servidor, unificando el desarrollo front-end y back-end. Destaca por su rendimiento, arquitectura basada en eventos y su amplio ecosistema de módulos npm, siendo muy popular en aplicaciones en tiempo real.



## 11. Características y posibilidades de desarrollo de una plataforma XAMPP.

XAMPP es una plataforma gratuita y multiplataforma que integra los componentes necesarios para desarrollar y probar aplicaciones web de manera local. Incluye Apache (servidor web), MySQL/MariaDB (base de datos), PHP y Perl.

### Características principales:

- Fácil instalación y configuración.
- Compatible con Windows, macOS y Linux.
- Incluye panel de control para gestionar servicios.
  - Apache-> Servidor web encargado de recibir procesar y responder solicitudes enviadas desde un navegador.
  - MySQL/MariaDB-> Sistema gestor de base de datos para almacenar y gestionar la información.
  - PHP-> Lenguaje de programación interpretado que se utiliza para crear sitios web dinámicos.
  - Perl-> Lenguaje de programación también incluido en la instalación de XAMPP.

### Posibilidades de desarrollo:

- Desarrollo: Permite a los desarrolladores crear y probar sitios web y aplicaciones en sus propios ordenadores sin necesidad de un servidor en línea.
- Configuración: Sencillo de instalar y configurar, agrupa todos los componentes necesarios en un solo paquete, simplificando la configuración de un servidor web.

## 12. En qué casos se instala (JVM) y JDK.

- **Máquina Virtual Java (JVM):** Es el intérprete que traduce el código Java ya compilado a instrucciones que entiende tu ordenador. Es la razón por la cual Java puede ejecutarse en cualquier sistema operativo: Windows, macOS, Linux, etc.
- **Kit de desarrollo de java (JDK):** Es el conjunto de herramientas necesarias para desarrollar aplicaciones Java. Contiene JRE (Java Runtime Environment) y herramientas de desarrollo como javac, java, jar, javadoc y jdb.

El JDK sirve para programar en Java y la JVM sirve para ejecutar dichos programas.

### 13. IDE más utilizados (características y grado de implantación actual).

- **Visual Studio:** Editor de código cada vez más popular, es gratuito, posee miles de plugins, lo que lo hace muy versátil a la hora de programar en casi cualquier lenguaje de programación. Implantación muy alta.
- **Eclipse:** IDE gratuito como Visual Studio, fue popular para programar en Java y otros lenguajes. Su interfaz es más compleja que la de Visual Studio pero aun así sigue siendo bastante usada.
- **IntelliJ IDEA:** IDE muy potente pensado para Java, posee herramientas que facilitan su desarrollo. Tiene una versión gratuita llamada Community Edition la cual es muy usada.
- **NetBeans:** IDE robusto con muchas funcionalidades especialmente para entornos Java, buen soporte multiplataforma y es completamente libre aunque a día de hoy no se utilice tanto como otros IDEs del mercado es una buena opción que sigue siendo muy utilizada.
- **PyCharm:** Generalmente utilizada para programar en Python ya que incluye potentes herramientas de análisis de código y refactorización.

### 14. Servidores HTTP /HTTPS más utilizados (características y grado de implantación actual).

**Apache HTTP Server:** de código abierto, multiplataforma y muy flexible gracias a su sistema de módulos, es el servidor web más tradicional y popular. Es ligero y fácil de instalar y configurar por lo que funciona bien en prácticamente cualquier sistema operativo.

**Nginx:** más moderno y rápido que Apache HTTP, haciendo que sea muy eficaz para sitios web con mucho tráfico, gestionando así las conexiones de manera más eficiente. Muy popular en los últimos años.

**Microsoft IIS:** es el servidor web de Microsoft, está integrado en los sistemas operativos Windows Server, su implantación es muy alta ya que las empresas tienden a utilizar productos de empresas fiables y conocidas como Microsoft.

**LiteSpeed:** es un servidor web gratuito, de código abierto y alto rendimiento, ofrece un rendimiento similar a NGINX. Su implantación es alta.

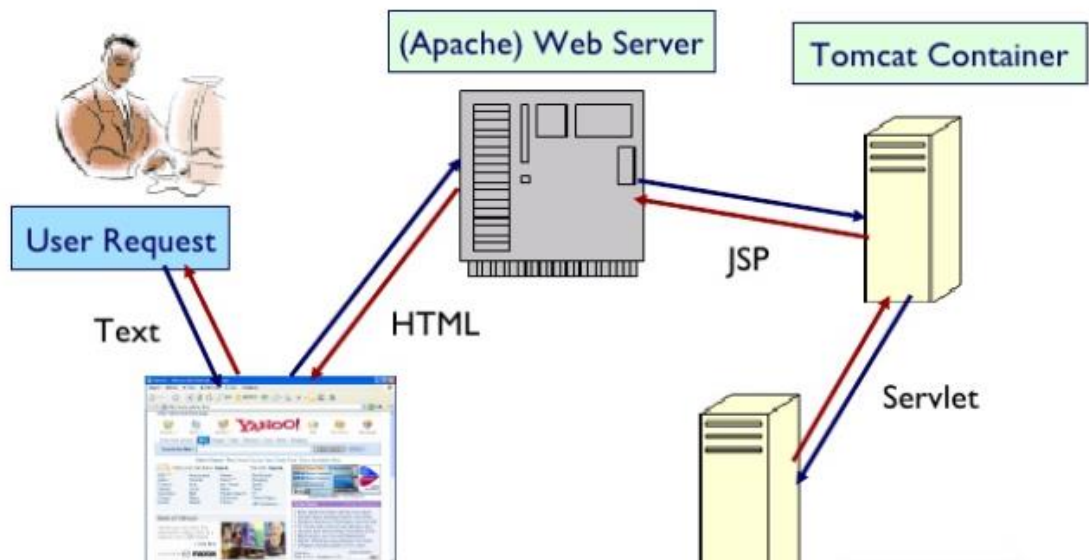
## 15. Apache HTTP vs Apache Tomcat

**Apache HTTP Server:** es un servidor web orientado a ofrecer contenido estático (HTML, imágenes, CSS, etc.) y puede ejecutar contenido dinámico mediante módulos externos, pero no está diseñado para aplicaciones Java. Su funcionamiento se basa en la estructura cliente-servidor, el cliente envía una petición HTTP desde el navegador al servidor, y este procesa la solicitud y envía la respuesta al cliente (el index de la página que ha solicitado).

**Apache Tomcat,** en cambio, es un servidor de aplicaciones especializado en Java (Servlets y JSP). Su principal función es ejecutar aplicaciones web Java, gestionar sesiones y generar contenido dinámico, aunque también puede servir archivos estáticos.

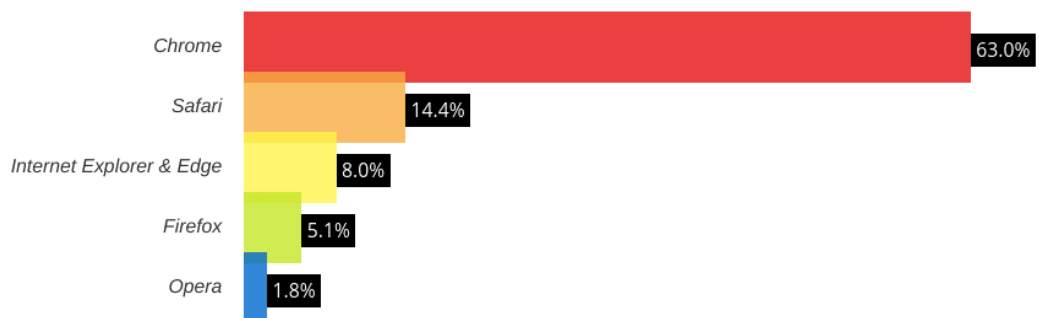
**Conclusión:** En definitiva, ambos son buenos servidores web, todo depende de lo que estes buscando y lo que pretendes almacenar o ofrecer. Si quieres tener una página web estática (HTML, imágenes, CSS, etc) deberías usar Apache HTTP, en cambio si lo que quieres es ejecutar aplicaciones java, gestionar sesiones y generar contenido dinámico deberías usar Apache Tomcat ya que esta mas enfocado a ello.

### ■ Apache Tomcat Structure



## 16. Navegadores HTTP /HTTPS más utilizados.

- **Google Chrome:**  
Es el navegador más utilizado, por su rapidez, estabilidad y amplio soporte de estándares web. Destaca por su integración con los servicios de Google. Su implantación global ronda el 63%.
- **Modzilla Firefox:**  
Se caracteriza por su enfoque en la privacidad, seguridad y personalización. Ofrece soporte completo para estándares web y potentes herramientas para desarrolladores. Su implantación global ronda el 5%.
- **Microsoft Edge:**  
Reemplazó a Internet Explorer y combina la compatibilidad con estándares modernos con integración en Windows y servicios de Microsoft. Actualmente es el navegador predeterminado de Microsoft, con una implantación global del 8%.
- **Safari:**  
Navegador predeterminado de productos Apple. Se caracteriza por su eficiencia energética, velocidad y optimización para dispositivos Apple. Su grado de implantación es alto entre usuarios de estos dispositivos, aunque es menor fuera del ecosistema Apple. Su implantación global es del 14%.
- **Opera:**  
Conocido por su enfoque en velocidad, bajo consumo de recursos y funciones integradas como VPN y bloqueador de anuncios. Su implantación es más limitada, aunque mantiene un nicho de usuarios fieles que valoran estas características constando así de aproximadamente el 2% de los usuarios globales. Tiene también una variante llamada OperaGX más enfocada a la optimización de los recursos en el navegador.





## 17. Generadores de documentación HTML (PHPDoc)

Los generadores de documentación HTML crean documentación técnica automáticamente a partir del código y sus comentarios. En PHP, analizan clases, funciones y métodos para generar páginas web que facilitan la comprensión y el mantenimiento del proyecto.

- **Código JavaScript:**

**JSDoc:** Es el estándar de JavaScript y el formato de comentario y herramienta de generación de documentación mas común. Genera una página web HTML en la que explica las clases, métodos y propiedades de tu código.

**TypeDoc:** Es similar a JSDoc pero en este caso está diseñado para leer anotaciones de tipo de TypeScript y generar documentación HTML.

- **Código PHP:**

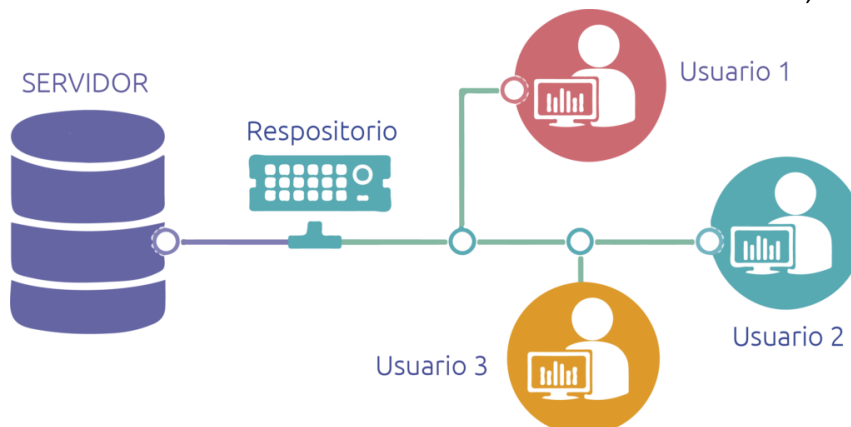
Los generadores de código de PHP son “phpDocumentator” y “Doxygen”. Funcionan leyendo los comentarios phpDoc de tu código y generan una documentación en base a tus comentarios.

**phpDocumentator:** Es el generador más común de PHP ya que respeta la especificación phpDoc oficial. Es capaz de generar documentación completa para bibliotecas y proyectos php.

**Doxygen:** Es un generador multilenguaje, es decir, se utiliza para diversos lenguajes de programación como Java Python y C++, aunque también soporta PHP generando la documentación con sintaxis phpDoc.

## 18. Repositorios de software – Sistemas de ctrl de versiones

Son lugares donde se almacena, organiza y gestiona el código fuente de un proyecto, incluyendo todas sus versiones y cambios a lo largo del tiempo. Permiten a los equipos de desarrollo trabajar de manera colaborativa, realizar seguimiento de modificaciones y mantener un historial completo del proyecto. Para gestionar estos repositorios se utilizan sistemas de control de versiones, que registran cada cambio, permiten revertir errores y facilitan la integración de aportes de distintos desarrolladores. Entre los sistemas de control de versiones más conocidos se encuentran GIT, CVS y Subversion.



## 19. Configuración del entorno de desarrollo

**Servidor Web:** Máquina virtual Ubuntu Server 22.04 con Apache HTTP/HTTPS, servicio php8.3-fpm para interpretar el código php de nuestra aplicación web y servicio MariaDB para la gestión de la información.

**Lenguaje de programación:** Utilizaremos el PHP ya que es el lenguaje más utilizado en el desarrollo de aplicaciones web del lado del servidor.

**IDE:** NetBeans para programar en PHP y Visual Studio para html y css, aunque es ideal aprender a manejar un solo IDE para poder programar en PHP, gestionar la base de datos y modificar la apariencia de nuestra página sin tener que estar cambiando constantemente de IDE.

**Control de Versiones:** Para el control de versiones utilizaremos Git y almacenaremos nuestros archivos en repositorios de GitHub para cada módulo.

**Pruebas:** Las pruebas las realizaremos en local accediendo desde el navegador de nuestro cliente antes de subir la aplicación a explotación.

**Entorno de explotación:** Cuando comprobemos que nuestra aplicación funciona correctamente y no tiene errores, la subiremos mediante SFTP a nuestro hosting virtual, nosotros estamos utilizando Plesk.

## 20. Configuración del entorno de explotación

**Servidor Web:** Apache HTTP/HTTPS u otro.

**Base de Datos:** MySQL/MariaDB.

**Lenguaje de programación:** PHP8.3-fpm.

## 21. Realizar un estudio sobre los siguientes conceptos

### **CMS – Sistema de gestión de contenidos:**

Es una aplicación web que permite crear, editar, organizar y publicar contenido digital en sitios web, sin necesidad de tener conocimientos técnicos avanzados en programación.

Los CMS sirven como base para desarrollar sitios y aplicaciones dinámicas.

Los desarrolladores pueden personalizar plantillas o crear plugins.

Ejemplos:

- WordPress.
- Joomla.
- Drupal.

### **ERP – Sistema de planificación de los recursos empresariales:**

Es un sistema de software que integra y automatiza los procesos esenciales de una organización: finanzas, recursos humanos, compras, inventarios, ventas, producción, contabilidad, entre otros.

Muchos ERP modernos son basados en la web o en la nube, accesibles desde cualquier navegador.

Los desarrolladores pueden crear módulos personalizados para adaptar el ERP a las necesidades de la empresa.

Ejemplos:

- SAP.
- Oracle ERP Cloud.
- Microsoft Dynamics 365.

## 22. Elegir y realizar un estudio y una presentación para la exposición del trabajo sobre una de las siguientes arquitecturas de desarrollo de Aplicaciones Web:

### • MEAN (con MongoDB y con MySQL)

Es un stack de Desarrollo full-stack JavaScript que se usa para crear aplicaciones web dinámicas. Sus siglas M.E.A.N significan:

M-> MongoDB: es una base de datos NoSQL orientada a documentos.

E-> Express.js: es un framework de Node.js para crear servidores y manejar rutas HTTP.

A-> Angular: es un framework front-end para crear interfaces interactivas.

N-> Node.js: es el entorno de ejecución de JavaScript en el servidor.

MEAN usa por defecto MongoDB, que es una base de datos NoSQL.

### • Java EE vs Spring

**Java EE (Enterprise Edition):** es un estandar oficial de java para aplicaciones empresariales.

Está basado en estándares oficiales, necesita un servidor de aplicaciones que implemente la especificación y es más prescriptive sobre APIs.

**Spring:** es un framework de Desarrollo java creado para simplificar el Desarrollo de aplicaciones empresariales.

Componentes clave:

Spring Core: Inyección de dependencias (IoC).

Spring MVC: Framework web.

Spring Data: Acceso a bases de datos.

Spring Boot: Configuración automática y microservicios.

Es un framework flexible y modular.

### • Microsoft .NET

Es un framework y ahora también una plataforma, desarrollada por Microsoft para crear aplicaciones de todo tipo (web, escritorio, móviles, juegos y servicios en la nube).

Permite usar varios lenguajes de programación como C#, VR.NET y F#.

Funciona sobre Windows, aunque .NET Core y .NET 5+ son multiplataforma (Windows, Linux, MacOS).

### • Angular 7

Angular es un framework de JavaScript/TypeScript desarrollado por Google que se utiliza para construir aplicaciones web de una sola página (SPA, Single Page Applications).

Angular 7 es una versión lanzada en Octubre de 2018 que trajo mejoras importantes sobre Angular 6, aunque hoy en día ya está un poco desactualizada frente a Angular 16+.

Angular divide la UI en componentes reutilizables:

Cada componente tiene un Template (HTML), una lógica (TypeScript) y estilos (CSS/SCSS).

### • Symfony

Symfony es un framework de PHP para desarrollar aplicaciones web y APIs robustas.

Está diseñado para proyectos grandes y complejos, con un enfoque en modularidad, reutilización de componentes y buenas prácticas.

Es muy popular en Europa y usado por plataformas grandes como Drupal, Magento y Prestashop.

### • Laravel

Laravel es un framework PHP de código abierto para desarrollar aplicaciones web modernas.

Está diseñado para hacer más fácil y rápido el desarrollo, proporcionando una sintaxis limpia y herramientas integradas.

### • CakePHP

CakePHP es un framework PHP open-source que sigue el patrón MVC (Model-View-Controller).

CakePHP promueve la convención sobre configuración, lo que significa que si sigues las reglas de nomenclatura y estructura del framework, muchas cosas funcionan automáticamente sin configurarlas manualmente.

### • CodeIgniter

CodeIgniter es un framework PHP open-source diseñado para crear aplicaciones web rápidas y ligeras.

Fue creado en 2006 y se caracteriza por su curva de aprendizaje baja y requerimientos mínimos de configuración.

Sigue el patrón MVC (Model-View-Controller), pero no obliga estrictamente a usarlo; puedes escribir código procedural si lo deseas.