TEMA 1 – DESARROLLO WEB EN ENTORNO SERVIDOR

Fecha última revisión 8/11/2024

Ejercicios

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

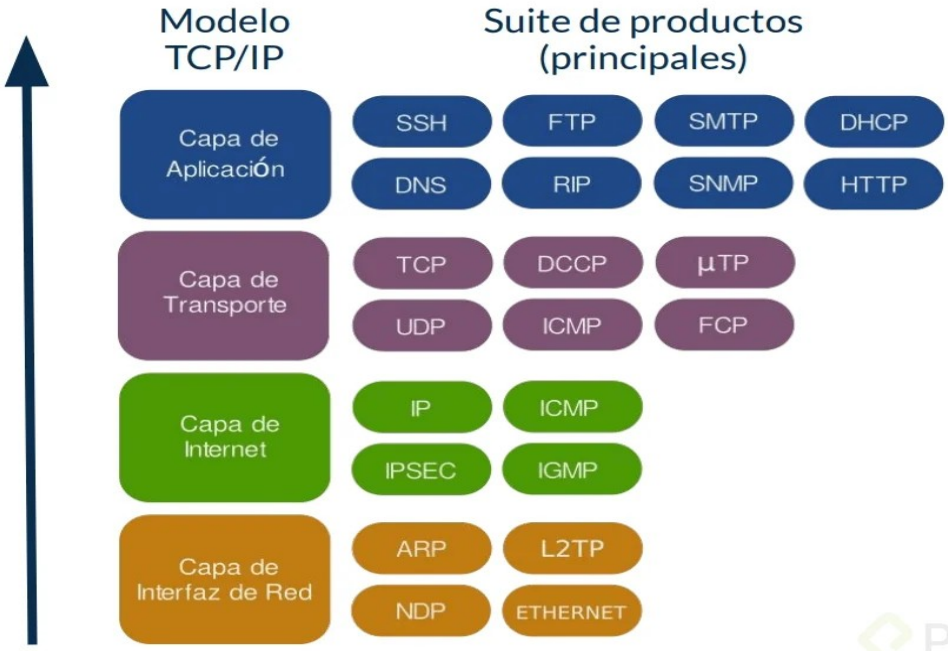
Consisten en un conjunto de normas que permiten que los ordenadores se comuniquen estableciendo la forma de identificación de estos en la red, la forma de transmisión de los datos y la forma en que la información debe procesarse.

IP (Internet Protocol): Protocolo de la capa de red del modelo OSI. Transmite datos en paquetes sin establecer una conexión previa, es decir, no garantiza la entrega.

TCP (Transmission Control Protocol): Protocolo de la capa de transporte. Establece una conexión fiable entre dos dispositivos, asegurando que los datos se entreguen completos y en orden. Usa puertos para identificar aplicaciones en un mismo dispositivo.

HTTP (Hypertext Transfer Protocol): Protocolo de la capa de aplicación que permite la transferencia de archivos web (HTML, XML). Usa el puerto 80 y no cifra la información. Sigue la estructura cliente-servidor.

HTTPS (Hypertext Transfer Protocol Secure): Versión segura de HTTP, que utiliza SSL/TLS para cifrar datos sensibles. Usa el puerto 443 y protege la información contra ataques.

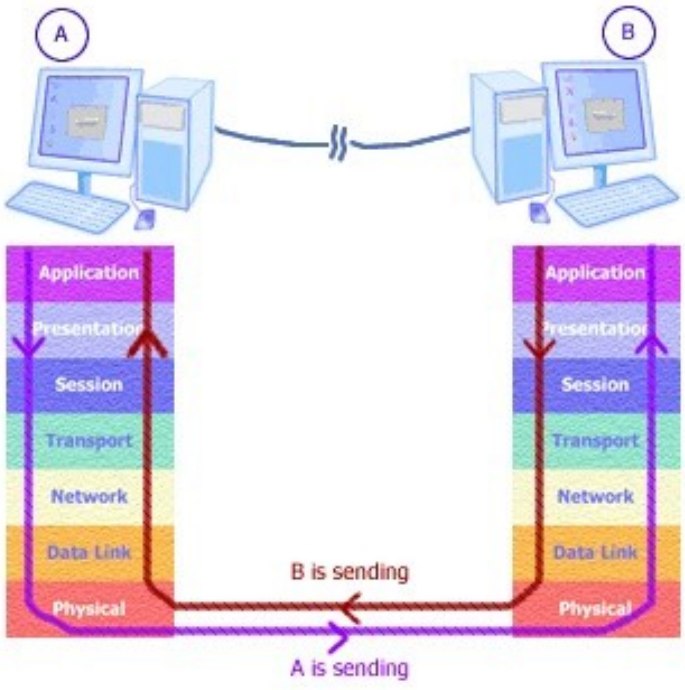


2. Modelo de comunicaciones cliente – servidor y su relación con las aplicaciones web.

Modelo de diseño de software en el que las tareas se reparten entre los proveedores de servicios (servidores) y los demandantes (clientes). El cliente realiza peticiones al servidor que las procesa y le da respuesta.

La relación con las aplicaciones web reside en que esta es una herramienta que los usuarios pueden utilizar accediendo a un servidor web (servidor) mediante un navegador (cliente).

https://acservidor.blogspot.com/2012/01/una-aproximacion-tecnica-la.html



3. Estudio sobre los métodos de petición HTTP/HTTPS más utilizados.

Hay muchos pero los más utilizados son los métodos GET y POST.

Método GET: *S*olicita una representación de un recurso específico. Las peticiones que usan el método GET sólo deben recuperar datos.

Método POST: *S*e utiliza para enviar una entidad a un recurso en específico, causando a menudo un cambio en el estado o efectos secundarios en el servidor.

Método HEAD: Pide una respuesta idéntica a la de una petición GET, pero sin el cuerpo de la respuesta.

Método PUT: Reemplaza todas las representaciones actuales del recurso de destino con la carga útil de la petición.

Método DELETE: Borra un recurso en específico.

Método CONNECT: Establece un túnel hacia el servidor identificado por el recurso.

Método OPTIONS: Es utilizado para describir las opciones de comunicación para el recurso de destino.

Método TRACE: Realiza una prueba de bucle de retorno de mensaje a lo largo de la ruta al recurso de destino.

Método PATCH: Es utilizado para aplicar modificaciones parciales a un recurso.

https://developer.mozilla.org/es/docs/Web/HTTP/Methods

4. Estudio sobre el concepto de URI (Identificador de Recursos Uniforme)/URL/URN, estructura, utilidad y relación con el protocolo HTTP/HTTPS.

Un URI es una cadena de caracteres que se utilizan para identificar un recurso o un nombre en internet. La URL es un tipo de URI que indica la ubicación del recurso y permite localizarlo y la URN es una cadena especializada que indica el nombre de un recurso único en internet.

Se relaciona con el protocolo HTTP/HTTPS ya que estos son parte fundamental de la URI ya sea URN O URL.

https://odiseageek.es/posts/en-que-se-diferencia-la-url-uri-y-urn/

Protocolo: El protocolo de una URL proporciona información sobre cómo se debe manejar la conexión entre el navegador del usuario y el servidor web que aloja el recurso. (HTTP, HTTPs, SFTP ...)

Subdominio: Un subdominio es una extensión de un dominio principal que actúa como un sitio web independiente dentro de ese dominio.

Dominio: Un dominio es una dirección única en la web que identifica un sitio web en particular. (wikipedia.org)

Extensión de dominio: La extensión de dominio son esas letras que siempre van detrás de un punto después del nombre del dominio.

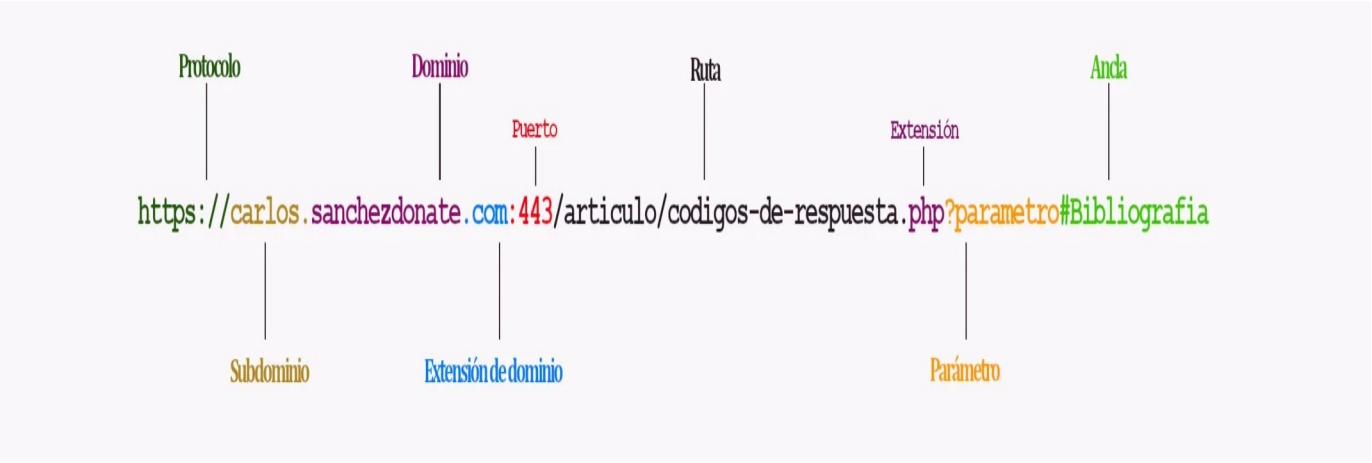
Puerto: Son puntos virtuales gestionados por el sistema operativo que permiten la comunicación entre aplicaciones y servicios en una red. (HTTP emplea el 80, HTTP el 443 ...)

Ruta: La ruta de una URL es la parte de la dirección web que sigue al dominio y que identifica la ubicación específica de una página dentro del sitio web.

Extensión de archivo: Un archivo en una dirección web y que indican el tipo de archivo que es.

Parámetro: Los parámetros son una variante de la propia URL. Estos pueden modificar el contenido o simplemente utilizarse con fines de analítica y otras características que pueda necesitar o presentar el proyecto.

Hashbang o Ancla: El hashbang se utiliza en algunas aplicaciones web para permitir que los usuarios naveguen por diferentes secciones de la aplicación sin tener que actualizar la página web completa.



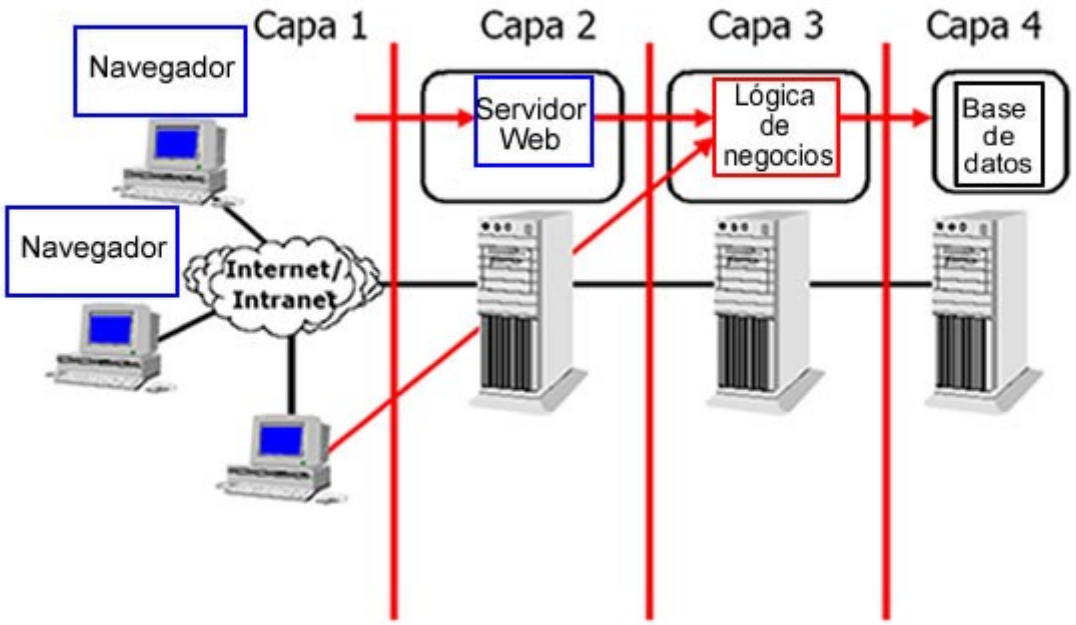
5. Modelo de desarrollo de aplicaciones multicapa – comunicación entre capas – componentes – funcionalidad de cada capa.

Una arquitectura multicapa es una arquitectura cliente-servidor en la que las funciones de presentación, lógica de negocio y gestión de datos están separadas físicamente.

Capa de presentación: Es la interfaz gráfica que el usuario ve. Muestra la información y recoge los datos del usuario. Debe ser fácil de usar y se comunica con la capa de negocio.

Capa de negocio: Procesa las peticiones del usuario y genera respuestas. Se conecta con la capa de presentación para recibir solicitudes y con la capa de datos para acceder o almacenar información.

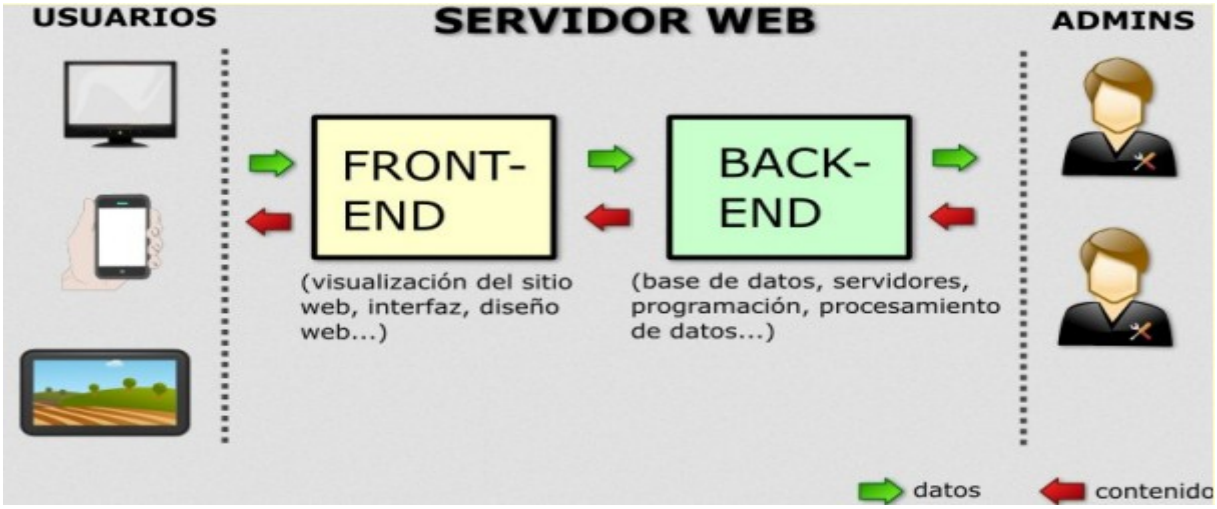
Capa de datos: es donde residen los datos y es la encargada de acceder a los mismos. Está formada por uno o más gestores de bases de datos que realizan todo el almacenamiento de datos, reciben solicitudes de almacenamiento o recuperación de información desde la capa de negocio.



6. Modelo de división funcional front-end / back-end para aplicaciones web.

La funcionalidad front-end es la que tiene un usuario normal de una aplicación web esté o no autenticado y la funcionalidad back-end es la que tiene un usuario administrador, publicador, censor etc.

Al contrario de lo que muchos piensan no guarda relación ser programador del front o del back con este tipo de funcionalidades aunque tengan el mismo nombre.



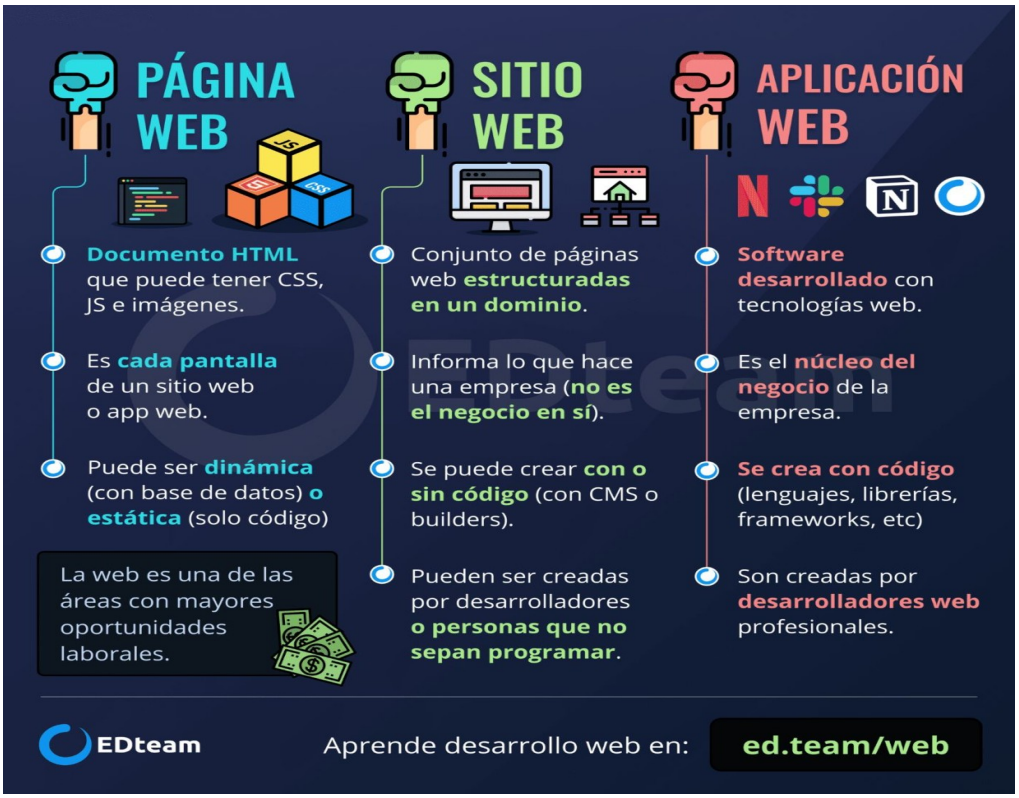
7. Página web estática – página web dinámica – aplicación web – mashup.

Página web estática: Contiene contenido fijo, no cambia a menos que se modifique manualmente. Los usuarios solo pueden ver la información, pero no interactuar con ella. Se basa en HTML y CSS. No tiene persistencia y sin control de acceso.

Página web dinámica: El contenido puede cambiar en respuesta a las interacciones del usuario o según datos externos de una base de datos (persistencia). Puede o no tener control de acceso.

Aplicación web: Software que solo se puede usar desde un navegador. Necesita un servidor web y un cliente web que se comunican con el protocolo HTTP.

Aplicación web híbrida: Combinación de datos o funcionalidades de diferentes fuentes o aplicaciones en un solo servicio. Puede cambiar lo que estoy viendo y es más compleja e interactiva.



8. Componentes de una aplicación web.

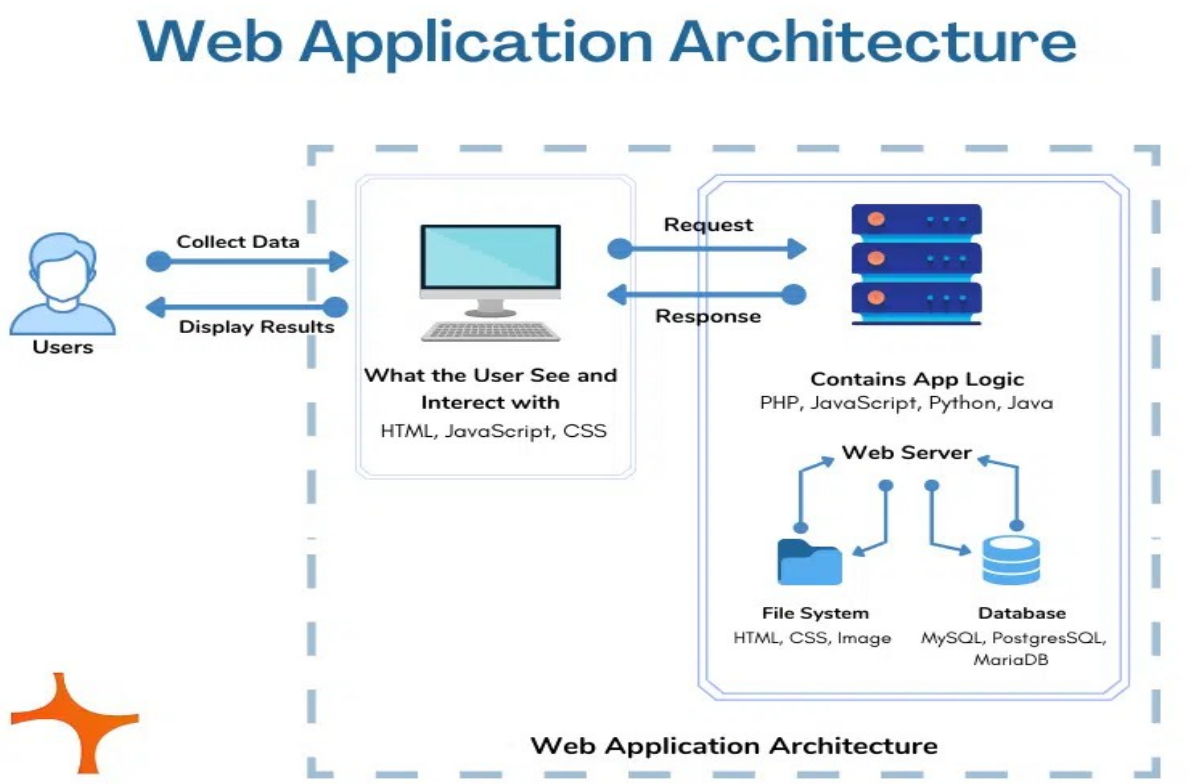
Cliente Web: Interfaz a través de la cual los usuarios interactúan con la aplicación, generalmente en un navegador.

Servidor Web: Programa que procesa solicitudes del cliente y genera respuestas. Puede establecer conexiones bidireccionales o unidireccionales y asíncronas.

Módulo encargado de ejecutar el código: Programa que ejecuta archivos de código escritos en diferentes lenguajes de programación, como PHP o Java.

Sistema gestor de base de datos (si se necesita): Software que permite administrar una base de datos (MySQL, MariaDB…).

(Ficheros escritos en) Lenguajes de programación: ficheros utilizados para realizar la aplicación interpretados y ejecutados por el módulo que se mencionó anteriormente. Dependiendo del lado de la aplicación en el que nos encontremos los lenguajes de programación irán variando.



9. Programas ejecutados en el lado del cliente y programas ejecutados en el lado del servidor – lenguajes de programación utilizados en cada caso.

Lado Cliente: Navegador Web. Los lenguajes que se utilizan en el lado del cliente son principalmente ***HTML***, ***JavaScript***, ***Java***, ***VBScript***, ***Flash***, ***CSS*** etc.

Lado Servidor: Servidor Web. Los lenguajes que se utilizan en el lado del servidor son ***PHP***, ***ASP*** (Visual Basic .Net, C#), ***JSP*** y ***Perl***.

https://alfredcmmx.wordpress.com/wp-content/uploads/2013/02/tema-2-lenguajes-del-lado-del-cliente-y-servidor.pdf

10. Lenguajes de programación utilizados en el lado servidor de una aplicación web (características y grado de implantación actual).

PHP: Fácil de aprender y multiplataforma. Ofrece muchas funciones y simplifica la conexión con diversas bases de datos, como Oracle y MySQL. Orientado a objetos

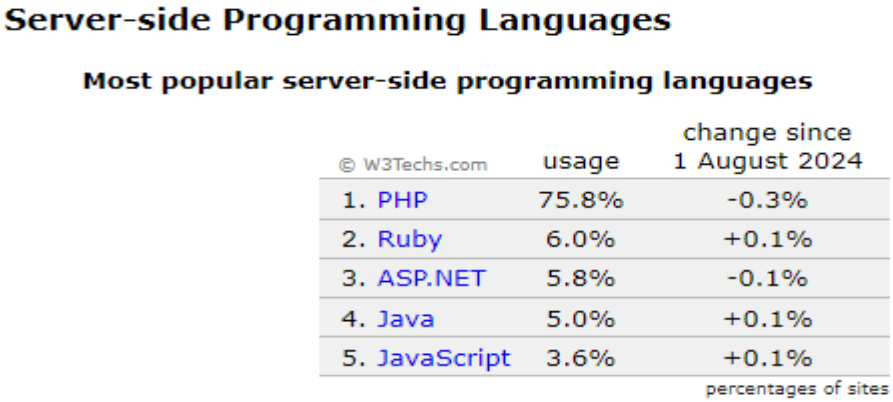
Ruby: Lenguaje interpretado y de código abierto, orientado a objetos y funcional. Destaca por su simplicidad y eficiencia, permitiendo escribir menos código para realizar tareas complejas.

ASP.NET: Desarrollado por Microsoft, permite control de usuario personalizado y separa la capa de diseño del código. Sin embargo, consume muchos recursos, lo que puede ser una desventaja. Es compilado.

Java: Lenguaje compilado e interpretado a la vez y orientado a objetos. Es conocido por su portabilidad (escribe una vez, ejecuta en cualquier lugar) y robustez, además de contar con una extensa biblioteca y comunidad de soporte.

JavaScript: Es un lenguaje interpretado y permite construir aplicaciones web altamente escalables y eficientes, utilizando un único lenguaje tanto en el cliente como en el servidor. Su ecosistema cuenta con una amplia variedad de bibliotecas y frameworks que facilitan el desarrollo.

https://w3techs.com/



Server-side Programming Languages

Most popular server-side programming languages

© W3Techs.com	usage	change since 1 August 2024
1. PHP	75.8%	-0.3%
2. Ruby	6.0%	+0.1%
3. ASP.NET	5.8%	-0.1%
4. Java	5.0%	+0.1%
5. JavaScript	3.6%	+0.1%

percentages of sites

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

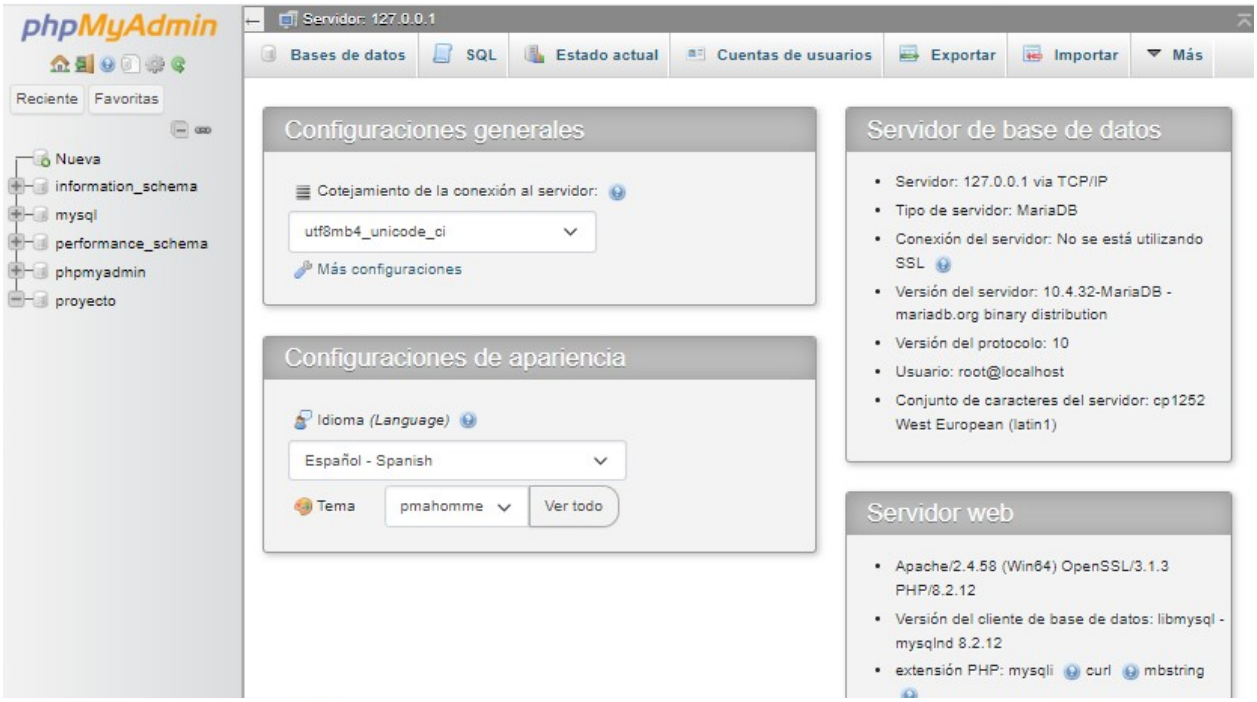
XAMPP es un paquete de software que incluye Apache, MySQL, PHP y Perl, lo que le permite ofrecer todos los componentes necesarios para desarrollar aplicaciones web con total garantía.

Sus características principales:

Portabilidad y compatibilidad multiplataforma: XAMPP es compatible con diferentes sistemas operativos, incluyendo Windows, macOS y Linux. Además, la portabilidad de XAMPP facilita el traslado de proyectos entre diferentes ordenadores y sistemas operativos.

Soporte para múltiples lenguajes de programación y bases de datos: Incluye soporte para PHP y Perl, además de gestionar bases de datos a través de MariaDB y MySQL.

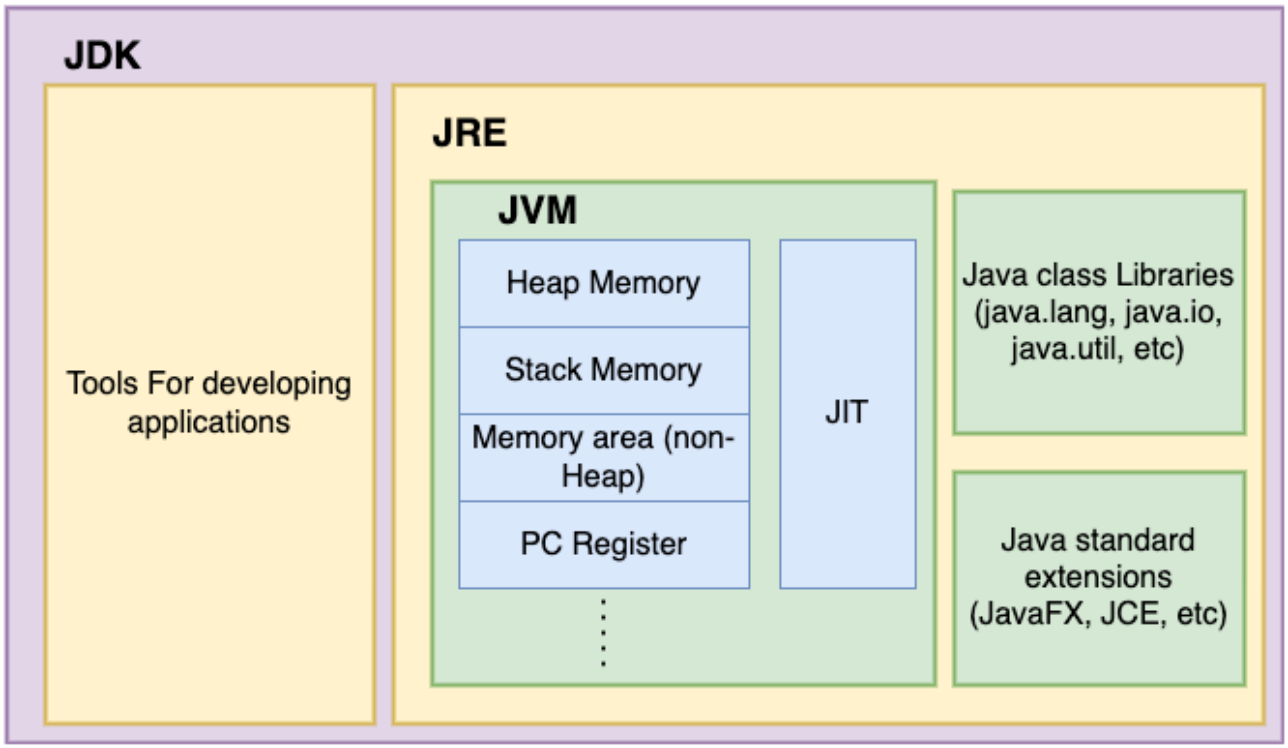
https://www.nettix.com.pe/blog/web-blog/que-es-xampp-y-como-puedo-usarlo



12. En que casos es necesaria la instalación de la máquina virtual Java (JVM) y el software JDK en el entorno de desarrollo y en el entorno de explotación.

Entorno de desarrollo: Se necesita **JDK** para compilar y desarrollar aplicaciones Java.

Entorno de explotación: Se necesita **JVM** para ejecutar aplicaciones Java ya compiladas.



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

Un IDE es una herramienta de software que ofrece un conjunto completo de funciones para el desarrollo de aplicaciones informáticas, facilitando y agilizando el proceso de programación mediante una interfaz de usuario integrada. Los IDEs más utilizados son:

- **Visual Studio Code** (Microsoft):
    - Amplia biblioteca de extensiones.
    - Sugerencias de código.
    - Integración de depuración.
    - Control de versiones.
    - Multiplataforma y código abierto.
    - >70% de cuota de mercado, muy popular y en constante crecimiento.

- **Eclipse** (originalmente de IBM, ahora mantenido por la Eclipse Foundation):
    - Soporte para múltiples lenguajes (Java, C++, Python, etc.).
    - Altamente personalizable con extensiones.
    - Resaltado de sintaxis.
    - Depuración de código y control de versiones.
    - Plataforma RCP para crear aplicaciones de escritorio personalizables.
    - 15-20% de cuota de mercado, relevante en entornos empresariales.

- **NetBeans** (desarrollado por Sun Microsystems, ahora mantenido por Oracle):
    - Fácil de usar y versátil, especialmente en Java.
    - Función de arrastrar y soltar para crear interfaces gráficas.
    - Altamente extensible mediante complementos y módulos.
    - Soporta varios lenguajes, como Java, PHP, HTML5 y JavaScript.
    - Herramientas para desarrollo web y móvil.
    - <5% de cuota de mercado, utilizado principalmente en educación y algunos proyectos de código abierto.

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

Nginx: Servidor web de alto rendimiento que funciona como proxy inverso y balanceador de carga. Es eficiente en el manejo de conexiones simultáneas, soporta WebSockets y permite caché de contenido, lo que lo hace ideal para sitios de alto tráfico debido a su escalabilidad y bajo consumo de recursos.

Apache: Servidor web de código abierto y gratuito, conocido por su flexibilidad y robustez. Compatible con varios sistemas operativos, permite la adición de módulos para personalizar sus funciones y ofrece altos estándares de seguridad con actualizaciones regulares.

Cloudflare Server: Proporciona un servicio gratuito de servidor de nombres de dominio (DNS) utilizando una red Anycast. Maneja más del 35% de los dominios DNS administrados y se destaca por su velocidad de búsqueda, con un tiempo promedio de 8,66 ms, según datos de 2016.

https://w3techs.com/

Web Servers

Most popular web servers

© W3Techs.com	usage	change since 1 August 2024
1. Nginx	33.8%	-0.2%
2. Apache	28.8%	-0.3%
3. Cloudflare Server	22.7%	+0.2%
4. LiteSpeed	13.8%	+0.3%
5. Microsoft-IIS	4.4%	-0.1%

percentages of sites

15. Apache HTTP vs Apache Tomcat

Apache HTTP: Servidor web desarrollado por apache , para servir cosas estáticas y mediante extensiones para hostear aplicaciones dinámicas PHP, Python, .Net y otros.

Apache Tomcat: Servidor web desarrollado por apache también, para servir aplicaciones que soporten el JVM aunque también soporta aplicaciones estáticas no es su fuerte. y no tiene algunas cosas para escalamiento que si tiene apache pero pueden trabajar juntos para tener esas.

La principal diferencia reside en que Apache HTTP está pensado para desarrollar en PHP, Perl, Python en cambio Apache Tomcat está pensado para desarrollar en Java.

Aspect	Apache HTTP Server	Apache Tomcat
Primary use case	Serves static content, handles HTTP requests.	Specialised for running Java Servlets and JSP applications.
Dynamic content	Limited support for dynamic content through modules.	Focuses on dynamic content using Java Servlets and JSP.
Architecture	Modular, process-based or threaded.	Java Servlet container, designed for Java-based applications.
Configuration	Configuration via text files.	Configuration through XML files and Java properties.
Language support	Supports various programming languages.	Primarily supports Java applications.
Performance	Efficient for serving static content.	Optimised for Java-based dynamic content.
Use cases	General web hosting, static content delivery.	Java web applications, servlets, and JSP deployment.

16. Navegadores HTTP /HTTPS más utilizados (características y grado de implantación actual).

Un navegador web es un programa que permite ver la información que contiene una página web. El navegador interpreta el código, HTML generalmente, en el que está escrita la página web y lo presenta en pantalla permitiendo al usuario interactuar con su contenido y navegar. Los más utilizados se detallan a continuación.

Google Chrome: Alto rendimiento, gran velocidad de carga, extensiones personalizables, y sincronización entre dispositivos. Es el navegador más utilizado a nivel mundial, con una cuota de mercado superior al 65%.

Mozilla Firefox: Enfoque en la privacidad del usuario, personalización a través de complementos, y herramientas de desarrollo integradas. Mantiene una cuota de mercado de alrededor del 15%, siendo una opción popular entre usuarios que priorizan la privacidad.

Microsoft Edge: Basado en Chromium, con integración de Windows 10/11, rendimiento mejorado, y herramientas de seguridad avanzadas. Ha ganado popularidad rápidamente, alcanzando aproximadamente el 5-10% del mercado.

Safari: Navegador predeterminado en dispositivos Apple, optimizado para rendimiento y eficiencia energética, con enfoque en la privacidad. Principalmente utilizado en dispositivos Apple, con una cuota de mercado de alrededor del 10-15%.

Opera: Integración de VPN gratuita, bloqueador de anuncios incorporado, y modo de ahorro de batería. Su cuota de mercado es menor, alrededor del 2-3%, pero tiene una base de usuarios leales.

17. Generadores de documentación HTML (PHPDoc): PHPDocumentor, ApiGen, …

PHPDocumentor: Herramienta ampliamente utilizada para generar documentación a partir de comentarios en el código PHP. Produce documentación en formato HTML, PDF y otros. Soporta múltiples formatos de salida, incluye diagramas de clases, y permite personalización mediante plantillas.

https://manual.phpdoc.org/HTMLSmartyConverter/HandS/phpDocumentor/tutorial_tags.pkg.html

ApiGen: Generador de documentación que se enfoca en la creación de documentación API a partir de código PHP. Ofrece una interfaz limpia y rápida, soporte para namespaces y una opción de búsqueda, ideal para proyectos con muchas clases y funciones.

```php
<?php
/**
 * Convert a string to UTF8 encoding
 *
 * @param string $string The string which has to be converted
 * @return string A string compatible with UTF8 encoding
 */
function to_utf8($string)
{
    // If we're not dealing with ASCII input
    if(!preg_match('/[^\x00-\x7F]/S', $string))
    {
```

18. Repositorios de software – sistemas de control de versiones: GIT , CVS, Subversion, ...

El control de versiones es un sistema que registra los cambios realizados en un conjunto de archivos a lo largo del tiempo, permitiendo gestionar, rastrear y revertir esos cambios de manera eficiente. Es fundamental en el desarrollo de software, ya que facilita la colaboración entre múltiples desarrolladores y asegura la integridad del código.

1. **Control de Versiones Centralizado (CVS)**:
    - **Descripción**: Un único servidor central almacena la versión completa del proyecto, y los desarrolladores trabajan en sus copias locales. Los cambios se envían al servidor central.
    - **Ventajas**: Fácil de entender y utilizar para equipos pequeños.
    - **Desventajas**: La falta de acceso a la historia del proyecto sin conexión y la posibilidad de conflictos si varios desarrolladores modifican los mismos archivos simultáneamente.
    - **CVS: https://es.wikipedia.org/wiki/CVS**
    - **Subversion: https://es.wikipedia.org/wiki/Subversion_(software)**
2. **Control de Versiones Distribuido (DVCS)**:
    - **Descripción**: Cada desarrollador tiene una copia completa del repositorio, incluyendo su historial. Esto permite trabajar sin conexión y realizar cambios localmente antes de enviar las actualizaciones al repositorio central.
    - **Ventajas**: Mejor manejo de ramas y fusiones, trabajo sin conexión, y mayor seguridad ante la pérdida de datos en el servidor.
    - **Desventajas**: Puede ser más complejo para usuarios nuevos, ya que hay más funciones y opciones.
    - **Git: https://es.wikipedia.org/wiki/Git**
    - **Mercurial: https://es.wikipedia.org/wiki/Mercurial**

19. Propuesta de configuración del entorno de desarrollo para la asignatura de Desarrollo web del lado servidor en este curso (incluyendo las versiones): xxx-USED y xxx-WXED.

XXX-USED

Sistema operativo: Ubuntu Server 24.04.1 LTS

Servidor administración remota: SSH

Servidor de transferencia de ficheros: SFTP (SSH)

Servidor Web: Apache HTTP

Sistema Gestor de Base de Datos: MySQL

XXX-WXED

Sistema operativo: Windows 10 Pro

Servidor administración remota :SSH

Navegador: Google Chrome, Microsoft Edge, Mozilla Firefox

IDE: NetBeans, Visual Studio Code

Ofimática: Libre Office Writer

Cliente SSH: Filezilla 3.67.0

REPOSITORIO

GitHub

23. Tipos de arquitecturas, frameworks asociados y grado de implantación.

Hay muchos tipos de arquitecturas pero las principales son: Java, AMP, .NET y JS.

Arquitectura Java EE

Componentes de cliente:

1.Clientes web (Thin Clients): Páginas dinámicas que se envían al navegador del usuario, conectándose a la capa web del servidor JEE para acceder a la capa de negocio.

2. Aplicaciones de cliente (Smart Clients): Ofrecen una interfaz más rica, ejecutándose directamente en el cliente y accediendo a la capa de negocio sin pasar por la capa web.

Componentes web:

Se ejecutan en el contenedor web del servidor JEE e incluyen:

1. Servelts: Clases Java que procesan peticiones del cliente y generan respuestas.

2. JSP (JavaServer Pages): Servlets orientados a la presentación, más fáciles de usar para los programadores.

3. JSF (JavaServer Faces): Evolución de JSP con una interfaz de usuario más avanzada, que soporta Ajax.

4. JavaBeans: Componentes que encapsulan la lógica de negocio de la aplicación.

Componentes de la capa de negocio: Son los que implementan la lógica de negocio y se ejecutan en la capa de negocio. Son los EJB Enterprise Java Beans. Los EJBs los hay de tres tipos, de sesión de entidad y dirigidos por mensajes. Las tecnologías básicas de esta capa son los EJBs, JPA (Java Persistence API) , la api-Rest o las de persistencia .

Componentes de la capa de datos: No son componentes de java en si,sino que son 3party. Nos permite explotar las características de esos servicios. Para ello Java EE nos proporciona un API de conectividad con bdd, otro de persistencia para poder guardar información, API de transacciones, etc.

Frameworks

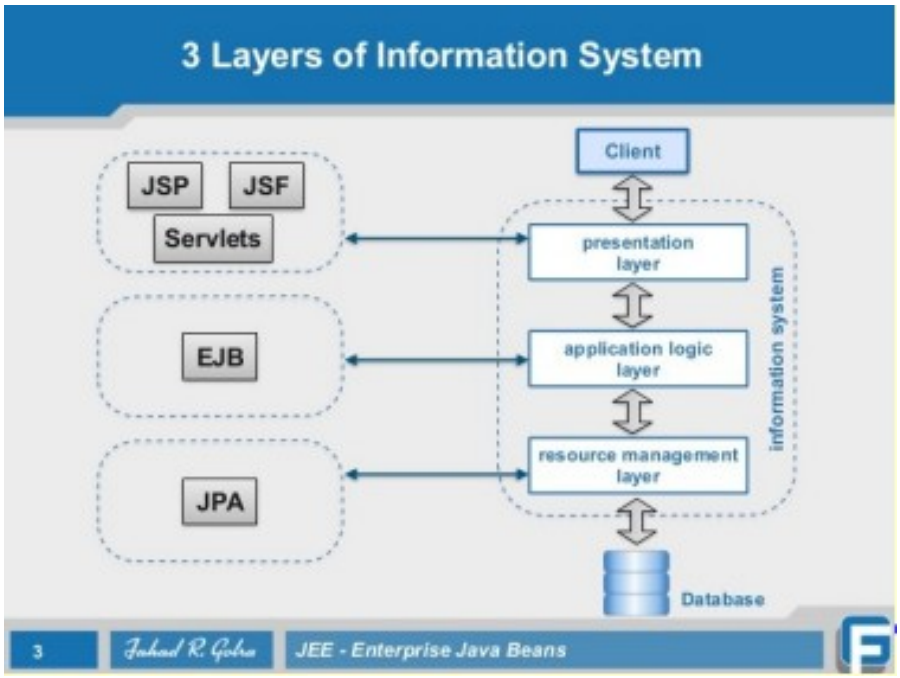
Los principales frameworks para la arquitectura Java EE incluyen:

Spring: Ofrece un contenedor de inversión de control y soporte para la programación orientada a aspectos, facilitando el desarrollo de aplicaciones robustas.

Hibernate: Framework de mapeo objeto-relacional (ORM) que simplifica la interacción con bases de datos y gestiona la persistencia de datos.

JavaServer Faces (JSF): Framework para construir interfaces de usuario en aplicaciones web, permitiendo la integración de componentes y la gestión del estado.

Jakarta EE (anteriormente Java EE): Proporciona un conjunto de especificaciones para construir aplicaciones empresariales, integrando varios frameworks y tecnologías.



Arquitectura AMP (Apache, MySQL, PHP)

Componentes de cliente:

1. **Clientes Web (Thin Clients):** Aplicaciones que acceden a las páginas web servidas por el servidor AMP. Usan HTML, CSS, y JavaScript para la interacción, sin necesidad de instalar software adicional en el cliente.

2. **Aplicaciones de Cliente (Smart Clients):** Pueden ser aplicaciones más ricas, como las basadas en JavaScript (usando frameworks como Angular o React), que se ejecutan en el cliente pero interactúan con la capa de negocio a través de APIs.

Componentes Web:

1. **Apache Web Server:** Servidor HTTP que maneja las peticiones del cliente, sirviendo contenido estático (HTML, imágenes, archivos CSS) y redirigiendo a los scripts PHP o aplicaciones dinámicas.

2. **PHP:** Lenguaje de programación del lado del servidor que ejecuta la lógica de negocio y genera contenido dinámico para las aplicaciones web.

3. **MySQL:** Sistema de gestión de bases de datos que almacena la información y se conecta con PHP para realizar operaciones CRUD (crear, leer, actualizar, eliminar).

Componentes de la Capa de Negocio:

- En la arquitectura AMP, la lógica de negocio se encuentra principalmente en los scripts PHP. Estos pueden interactuar con la base de datos MySQL utilizando consultas SQL o frameworks ORM como Doctrine o Eloquent.

Componentes de la Capa de Datos:

- **MySQL** se encarga de la persistencia de datos en la base de datos. En algunos casos, se usan otros sistemas de almacenamiento como Redis o Elasticsearch para optimizar ciertas tareas de búsqueda o almacenamiento en caché.

Frameworks:

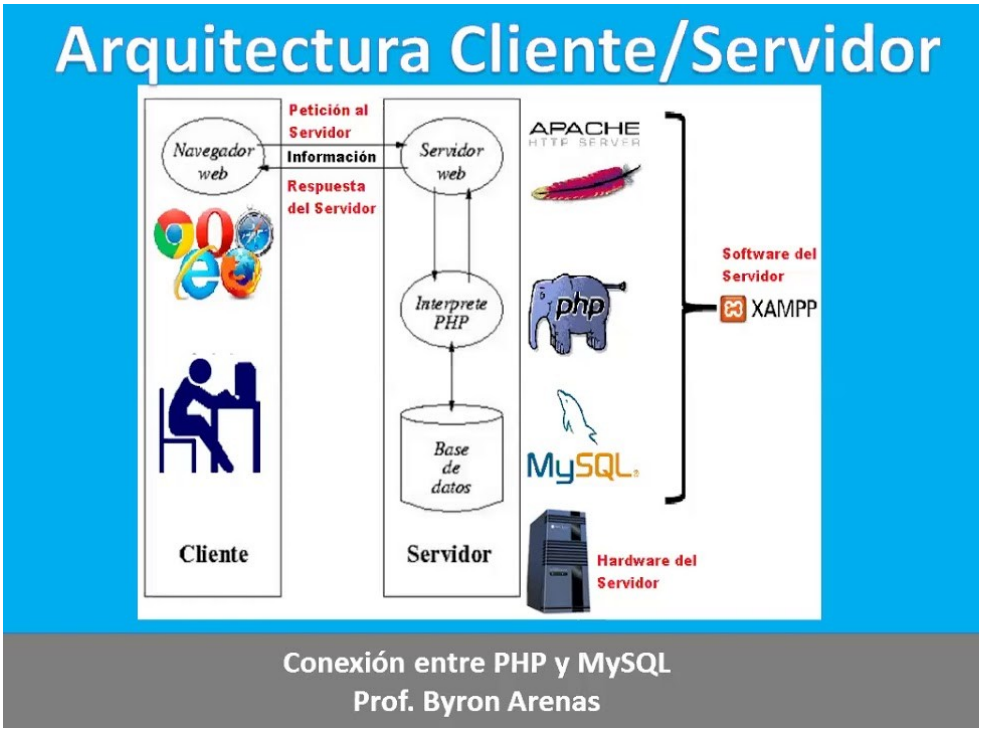
1. **Laravel:** Framework PHP que facilita el desarrollo de aplicaciones web robustas y escalables, con características como Eloquent ORM para trabajar con bases de datos y un sistema de rutas eficiente.

2. **Symfony:** Framework PHP modular y flexible, que ofrece una base sólida para aplicaciones web complejas y es utilizado como base para otros frameworks como Laravel.

3. **CodeIgniter:** Framework ligero y fácil de usar para PHP, ideal para crear aplicaciones web rápidas con menos configuración.

4. **Zend Framework:** Framework PHP orientado a objetos para crear aplicaciones empresariales de alto rendimiento y escalables.

5. **CakePHP:** Framework que promueve el desarrollo rápido, basado en el patrón MVC, facilitando la construcción de aplicaciones PHP dinámicas.



Arquitectura .NET

Componentes de Cliente:

1. **Clientes Web (Thin Clients):** Navegadores que interactúan con aplicaciones ASP.NET a través de HTTP/HTTPS.
2. **Aplicaciones de Cliente (Smart Clients):** Pueden ser aplicaciones .NET de escritorio, móviles (Xamarin), o aplicaciones ricas en la web usando Blazor (tecnología basada en WebAssembly que permite ejecutar C# en el navegador).

Componentes Web:

1. **ASP.NET Core:** Framework que permite crear aplicaciones web modernas y de alto rendimiento. Se encarga de manejar peticiones HTTP y servir contenido estático o dinámico.
2. **Razor Pages y MVC:** Razor es un motor de plantillas que se utiliza dentro de ASP.NET para crear páginas dinámicas. En el modelo MVC (Model-View-Controller), Razor es la tecnología de vista que facilita la separación de lógica de presentación.
3. **SignalR:** Tecnología que facilita la comunicación en tiempo real entre el servidor y el cliente, útil para aplicaciones interactivas.

Componentes de la Capa de Negocio:

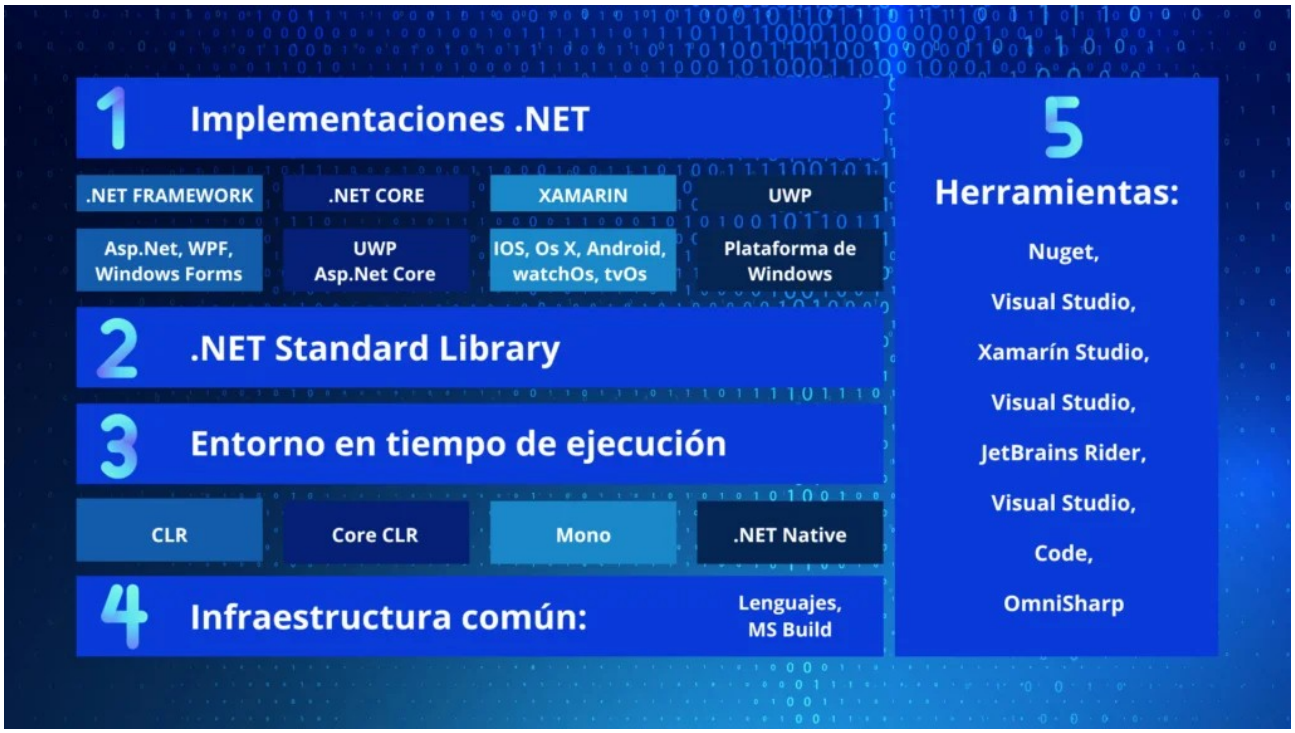
- **.NET Core y .NET Framework:** En esta capa se desarrollan las clases que implementan la lógica de negocio. Pueden incluir servicios y APIs RESTful con ASP.NET Core Web API.
- **Entity Framework (EF):** Framework ORM que simplifica la persistencia de datos, permitiendo trabajar con bases de datos relacionales sin necesidad de escribir SQL explícitamente.

Componentes de la Capa de Datos:

- **SQL Server o cualquier DBMS compatible:** Usado para almacenar los datos de las aplicaciones. Entity Framework se conecta a esta capa para realizar operaciones CRUD sobre las tablas de la base de datos.
- **NoSQL (Opcional):** Si la aplicación requiere una base de datos NoSQL, puede integrarse con tecnologías como MongoDB.

Frameworks:

- **ASP.NET Core:** Framework principal para construir aplicaciones web, APIs y servicios.
- **Entity Framework Core:** ORM que facilita el mapeo de objetos a bases de datos relacionales.
- **Blazor:** Framework de Microsoft para crear aplicaciones interactivas en el navegador con C#.
- **SignalR:** Para comunicación en tiempo real.



Arquitectura MEAN (MongoDB, Express.js, Angular, Node.js)

Componentes de Cliente:

1. **Clientes Web (Thin Clients):** Navegadores que interactúan con la aplicación MEAN a través de HTTP.
2. **Aplicaciones de Cliente (Smart Clients):** Aplicaciones Angular que se ejecutan en el navegador, interactuando con la API del servidor Node.js.

Componentes Web:

1. **Express.js:** Framework minimalista de Node.js que gestiona la lógica de servidor. Maneja las rutas y las peticiones HTTP, y conecta con la base de datos MongoDB.
2. **Node.js:** Entorno de ejecución JavaScript en el servidor, utilizado para manejar las solicitudes HTTP y ejecutar el código de backend.
3. **MongoDB:** Base de datos NoSQL que se utiliza para almacenar los datos de manera flexible y escalable.

Componentes de la Capa de Negocio:

- **Node.js y Express.js:** La lógica de negocio se maneja principalmente en estas tecnologías, que ejecutan las operaciones y gestionan la comunicación con la base de datos MongoDB.
- **APIs RESTful:** Generalmente, Express.js se usa para crear APIs RESTful que interactúan con la base de datos y proporcionan servicios al cliente (en este caso, las aplicaciones Angular).

Componentes de la Capa de Datos:

- **MongoDB:** Sistema de base de datos NoSQL que guarda los datos en un formato flexible (documentos BSON). Es escalable y eficiente para manejar grandes volúmenes de datos no estructurados.

Frameworks:

- **Angular:** Framework frontend de JavaScript para crear aplicaciones web dinámicas, que se conecta con las APIs RESTful de Node.js.
- **Express.js:** Framework de backend para Node.js, que facilita el desarrollo de aplicaciones web y servicios API.
- **Node.js:** Entorno de ejecución de JavaScript en el servidor.
- **MongoDB:** Base de datos NoSQL orientada a documentos.

MEAN (MongoDB, Express.js, Angular, Node.js)

MEAN es una pila tecnológica basada en JavaScript que utiliza **Angular** como framework para el frontend. Angular es un framework completo que ofrece una solución integral para el desarrollo de aplicaciones web, usando TypeScript como su lenguaje principal. Angular es ideal para aplicaciones empresariales grandes, ya que ofrece una estructura rígida, manejo de rutas, inyección de dependencias, y una completa integración con servicios backend.

- **Frontend**: Utiliza **Angular**, un framework robusto y opinado que facilita la creación de aplicaciones de una sola página (SPA) interactivas. Angular viene con una arquitectura bien definida que promueve el uso de módulos, componentes, y servicios, lo cual es útil para proyectos de gran escala.
- **Backend**: Se utiliza **Node.js** para el entorno de ejecución JavaScript del servidor, y **Express.js** como el framework minimalista que maneja las rutas, peticiones HTTP y middleware.
- **Base de Datos**: **MongoDB** como base de datos NoSQL, que almacena los datos en formato BSON, adecuado para manejar grandes volúmenes de datos no estructurados o semiestructurados.

MERN (MongoDB, Express.js, React, Node.js)

MERN es otra pila basada en JavaScript, pero en lugar de **Angular**, usa **React** para el frontend. React no es un framework completo como Angular, sino una **biblioteca** para construir interfaces de usuario. React es muy popular por su enfoque basado en componentes, lo que facilita la reutilización y la organización del código. Además, React es más flexible que Angular y no impone una estructura tan estricta, lo que le da al desarrollador más libertad.

- **Frontend**: Utiliza **React**, una biblioteca que permite crear interfaces de usuario dinámicas y componentes reutilizables. A diferencia de Angular, React no ofrece tantas soluciones "listas para usar" como en el caso de Angular, pero su simplicidad y flexibilidad hacen que sea ideal para proyectos de todo tipo, desde pequeños hasta grandes.
- **Backend**: Al igual que en MEAN, se usa **Node.js** y **Express.js** para gestionar las peticiones y ejecutar la lógica de backend.
- **Base de Datos**: **MongoDB** como base de datos NoSQL, igual que en MEAN, aprovechando su flexibilidad y escalabilidad para proyectos que requieren un almacenamiento ágil y fácil de ajustar.

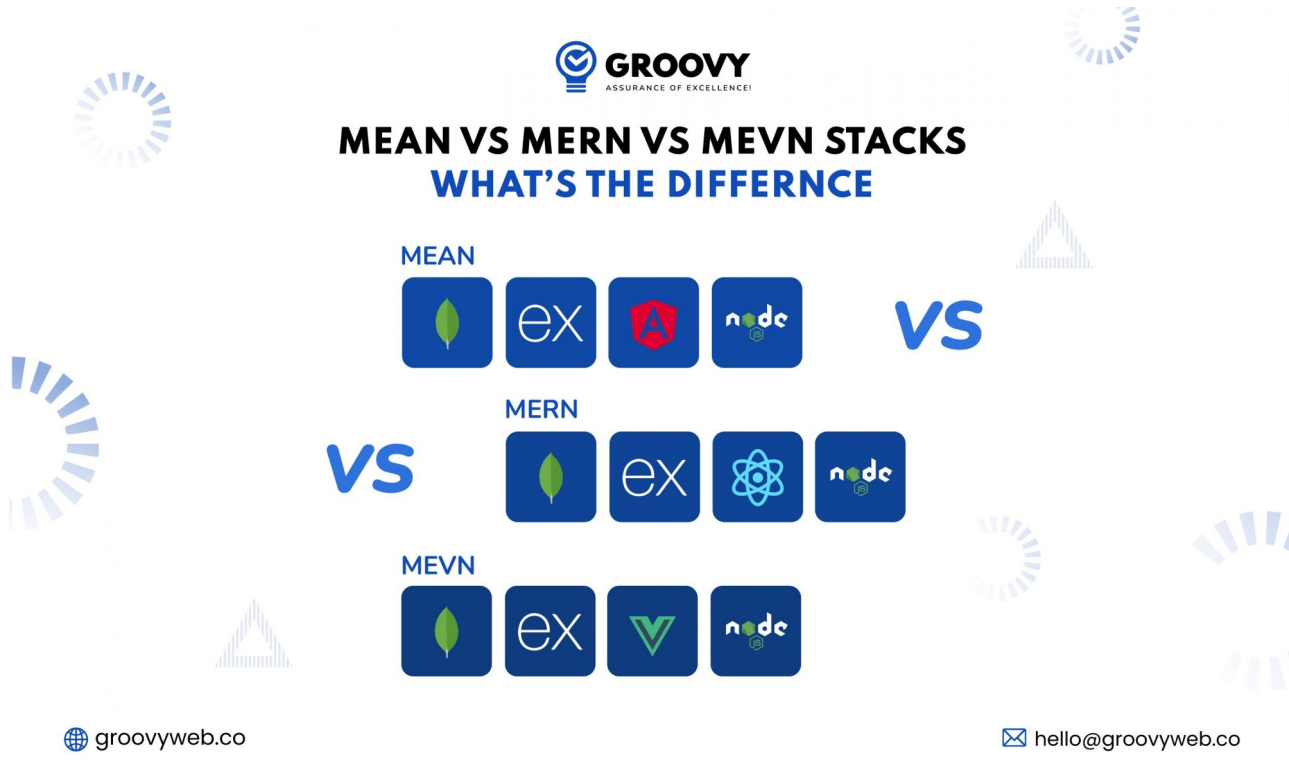
MEVN (MongoDB, Express.js, Vue.js, Node.js)

MEVN es una variación similar a **MERN**, pero en lugar de **React**, usa **Vue.js** como el framework para el frontend. Vue.js es un framework progresivo, lo que significa que se puede adoptar de manera gradual. Es conocido por ser más fácil de aprender que Angular y React, y por su flexibilidad. Vue tiene una curva de aprendizaje mucho más suave y permite integrarse fácilmente con otros proyectos.

- **Frontend**: **Vue.js** es un framework ligero y fácil de aprender para construir interfaces de usuario interactivas. A diferencia de React, que se centra solo en la vista, Vue también ofrece un sistema completo de gestión de estado (Vuex) y un enrutador (Vue Router). Vue es ideal para desarrolladores que buscan algo menos estructurado que Angular pero más completo que React por sí solo.
- **Backend**: Al igual que en las otras dos pilas, **Node.js** y **Express.js** se encargan de manejar el backend, gestionando las rutas y las peticiones HTTP.
- **Base de Datos**: **MongoDB** como base de datos NoSQL, utilizado por igual en las tres pilas debido a su compatibilidad con JavaScript y su flexibilidad para manejar datos semiestructurados.

Comparación y Elección de la Pila

- **MEAN**: Es ideal para aplicaciones grandes y empresariales que requieren una solución todo en uno. Angular ofrece una arquitectura más rígida, lo cual es útil en equipos grandes donde el código debe estar muy bien estructurado. Además, Angular incluye muchas características integradas, como la inyección de dependencias, que facilitan el desarrollo de aplicaciones complejas.
- **MERN**: Es adecuado para proyectos donde se desea mayor flexibilidad y una interfaz de usuario más interactiva, usando React. MERN es ideal para aplicaciones con una UI dinámica que requiere un control más granular sobre los componentes y su ciclo de vida. React es más ligero y permite una integración más sencilla con otras bibliotecas o herramientas de terceros.
- **MEVN**: Es una opción intermedia entre Angular y React. Vue.js combina lo mejor de ambos mundos, ofreciendo una experiencia más sencilla que Angular pero más completa que React por sí sola. Es ideal para quienes buscan facilidad de uso, flexibilidad y rapidez en el desarrollo, sin la complejidad de Angular.


GROOVY
ASSURANCE OF EXCELLENCE!
MEAN VS MERN VS MEVN STACKS
WHAT'S THE DIFFERNCE
MEAN
VS
MERN
VS
MEVN
groovyweb.co
hello@groovyweb.co