INTRODUCCIÓN A LAS APLICACIONES WEB

En este curso aprenderemos a crear aplicaciones web desde cero. Pero, para ello, primero necesitamos una pequeña introducción a las mismas.

¿Qué es una aplicación web?

Es un software que se ejecuta en un servidor y al que se accede a través de un navegador web. A diferencia de las aplicaciones de escritorio, que requieren instalación en un sistema operativo específico, o de las aplicaciones móviles, que son desarrolladas para sistemas como iOS o Android, las aplicaciones web están diseñadas para funcionar en cualquier dispositivo con un navegador compatible y una conexión a Internet.

EJEMPLOS DE APLICACIONES WEB



VENTAJAS DE LAS APLICACIONES WEB

1. **Accesibilidad multiplataforma**: Se pueden utilizar desde cualquier dispositivo con un navegador (ordenadores, tabletas, móviles), lo cual permite una mayor flexibilidad de uso.
2. **Actualización centralizada**: No requieren actualizaciones manuales por parte del usuario; las mejoras y correcciones se aplican en el servidor, y todos los usuarios tienen acceso a la última versión automáticamente.
3. **Costo de desarrollo**: Desarrollar una aplicación web suele ser más económico, ya que no es necesario crear versiones para diferentes sistemas operativos, como Android o iOS.
4. **Fácil escalabilidad**: Permiten añadir o reducir funciones rápidamente, y los cambios se implementan en un solo punto (el servidor).
5. **Almacenamiento en la nube**: Los datos pueden almacenarse en la nube, lo cual permite una gestión y recuperación de datos más segura y accesible desde cualquier lugar.

DESVENTAJAS DE LAS APLICACIONES WEB

1. **Dependencia de la conexión a Internet**: Generalmente, requieren conexión a Internet para funcionar correctamente, lo cual puede ser una limitación en lugares sin acceso confiable.
2. **Rendimiento limitado**: En comparación con las aplicaciones de escritorio, pueden tener un rendimiento más bajo, especialmente para tareas que requieren gran capacidad de procesamiento o interacción intensiva.
3. **Seguridad**: Aunque existen prácticas de seguridad, las aplicaciones web están expuestas a amenazas como el phishing, los ataques de inyección de SQL y otros.
4. **Limitaciones de funcionalidad**: Algunas características (como el uso completo del hardware del dispositivo o acceso a ciertas APIs del sistema) pueden estar limitadas en aplicaciones web, en comparación con las de escritorio o móviles.
5. **Delegación del control de la información**: Nuestros datos se almacenan en el servidor de la empresa que ha creado la aplicación y, por lo tanto, dependemos de la buena gestión que ésta haga de ellos.



COMPARATIVA CON APLICACIONES DE ESCRITORIO Y MÓVILES



Aplicaciones de escritorio: Ofrecen mejor rendimiento y más funcionalidades relacionadas con el hardware (acceso a CPU, GPU, etc.), pero suelen estar limitadas a un solo sistema operativo y necesitan instalaciones y actualizaciones manuales.



Aplicaciones móviles: Están diseñadas específicamente para dispositivos móviles, lo que permite un mejor rendimiento y una experiencia optimizada para estos dispositivos. Sin embargo, deben ser desarrolladas y mantenidas en múltiples versiones (iOS, Android) y requieren actualización periódica.

En resumen, las aplicaciones web son una solución versátil y de fácil acceso, ideal para tareas que no exijan gran cantidad de procesamiento y para usuarios que valoren la conveniencia de la conectividad y accesibilidad en múltiples dispositivos.

EVOLUCIÓN DE LA WEB

La evolución de la web ha sido marcada por diferentes fases, conocidas como **Web 1.0**, **Web 2.0** y **Web 3.0**, con algunos desarrollos y experimentos hacia la **Web 4.0**. Cada fase tiene características y pilares fundamentales que reflejan los avances tecnológicos y la forma en que los usuarios interactúan con el contenido en línea.

1.0
WEB ESTÁTICA

2.0
WEB SOCIAL Y COLABORATIVA

3.0
WEB SEMÁNTICA E INTELIGENTE

4.0
WEB AUTÓNOMA Y UBICUA

ALGUNOS CONCEPTOS: SERVIDOR WEB

Es un software o un hardware que entrega contenido web a los navegadores de los usuarios a través del protocolo HTTP. Su función principal es gestionar las solicitudes de recursos estáticos como páginas HTML, hojas de estilo (CSS), imágenes, archivos JavaScript y otros elementos estáticos que forman una página web.

Función principal:

- Servir contenido estático: Un servidor web recibe solicitudes de archivos y responde con el contenido solicitado. Esto incluye archivos HTML, imágenes (JPEG, PNG), CSS, JavaScript, etc.
- Procesamiento de solicitudes HTTP: Recibe las solicitudes HTTP o HTTPS de los navegadores y devuelve los recursos adecuados.

Ejemplos: Apache, Nginx, Microsoft IIS, LiteSpeed.

ALGUNOS CONCEPTOS: SERVIDOR DE APLICACIONES WEB

Es un software que gestiona aplicaciones dinámicas. Además de servir contenido estático, un servidor de aplicaciones ejecuta la lógica de negocio, maneja las interacciones con bases de datos, y genera contenido dinámico en tiempo real antes de enviarlo al cliente. Este tipo de servidor suele ejecutar código de servidor, como Java, Python, Ruby, PHP, Node.js, etc.

Función principal:

- Procesar contenido dinámico: El servidor de aplicaciones es responsable de ejecutar la lógica de la aplicación y generar contenido dinámico. Esto incluye consultar bases de datos, autenticar usuarios, gestionar sesiones, y construir las páginas dinámicamente antes de enviarlas al cliente.
- Ejecutar código de backend: Además de recibir solicitudes HTTP, el servidor de aplicaciones ejecuta scripts o aplicaciones en el backend, gestionando las interacciones con bases de datos y otros sistemas.

Ejemplos: Tomcat, Node.js, Django.

ENFOQUES PARA EL DESARROLLO: DESARROLLO DEL LADO CLIENTE (FRONTEND)

El **frontend** es todo lo que el usuario ve e interactúa en el navegador web. Se refiere al desarrollo de la interfaz de usuario (UI) y la experiencia de usuario (UX) de la aplicación web. En este enfoque, el código corre en el navegador del usuario.

Herramientas y tecnologías comunes:

HTML, CSS, JavaScript, frameworks como React, Angular, Vue.js, etc.

Ventajas:

- Rendimiento mejorado
- Interactividad y dinamismo
- Desacoplamiento del backend

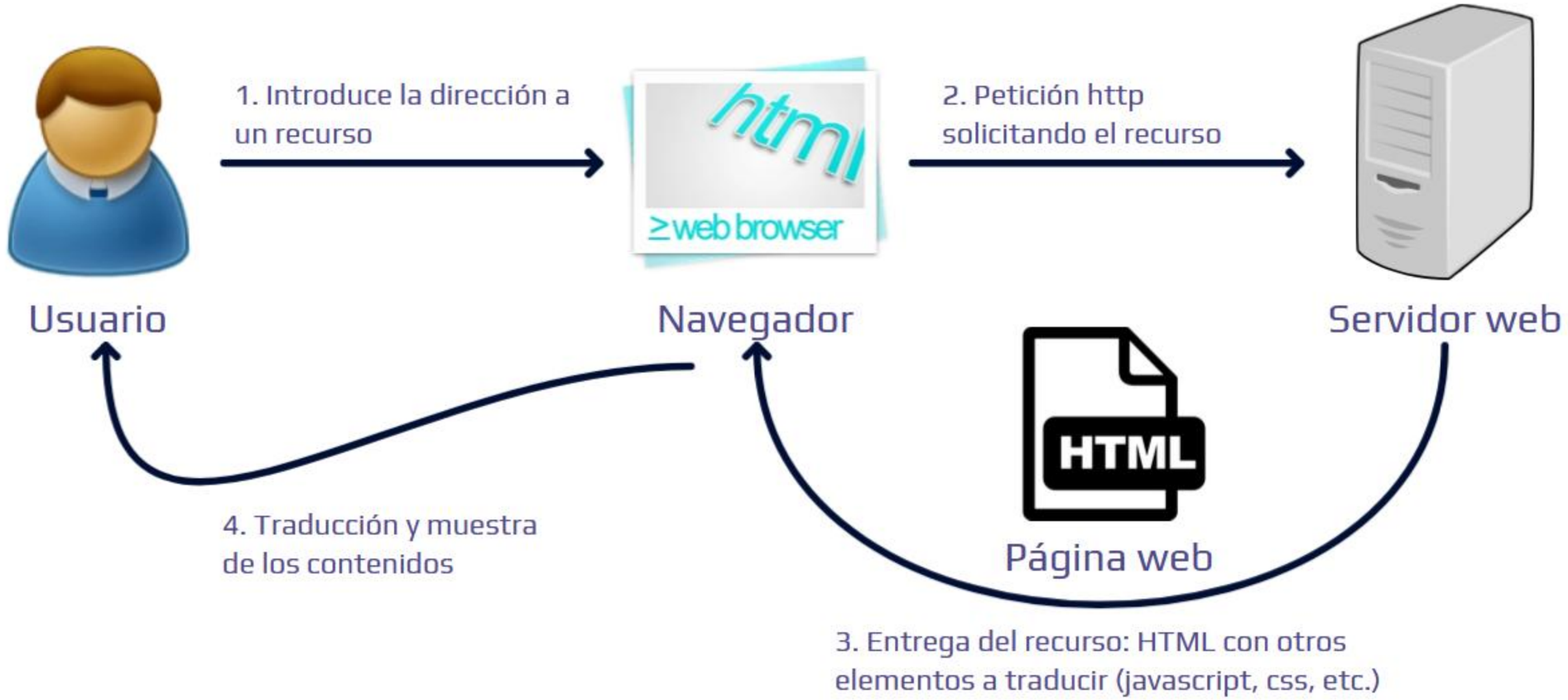


Desventajas:

- Dependencia de los recursos del cliente
- SEO y accesibilidad
- Limitaciones de seguridad



ENFOQUES PARA EL DESARROLLO: DESARROLLO DEL LADO CLIENTE (FRONTEND)



ENFOQUES PARA EL DESARROLLO: DESARROLLO DEL LADO SERVIDOR (BACKEND)

El **backend** se refiere a todo lo que ocurre "detrás de las cámaras" en el servidor. El servidor es el responsable de gestionar las bases de datos, la lógica de negocio, la autenticación de usuarios, la gestión de peticiones y respuestas HTTP, y otros procesos que no son visibles directamente para el usuario.

Herramientas y tecnologías comunes:

Lenguajes como Node.js, Python, Ruby, PHP, Java, C#, etc. Bases de datos como MySQL, PostgreSQL, MongoDB, etc. Frameworks como Express.js, Django, Flask, Laravel, Spring Boot, etc.

Ventajas:

- Seguridad
- Control total sobre los datos
- Escalabilidad
- SEO y accesibilidad

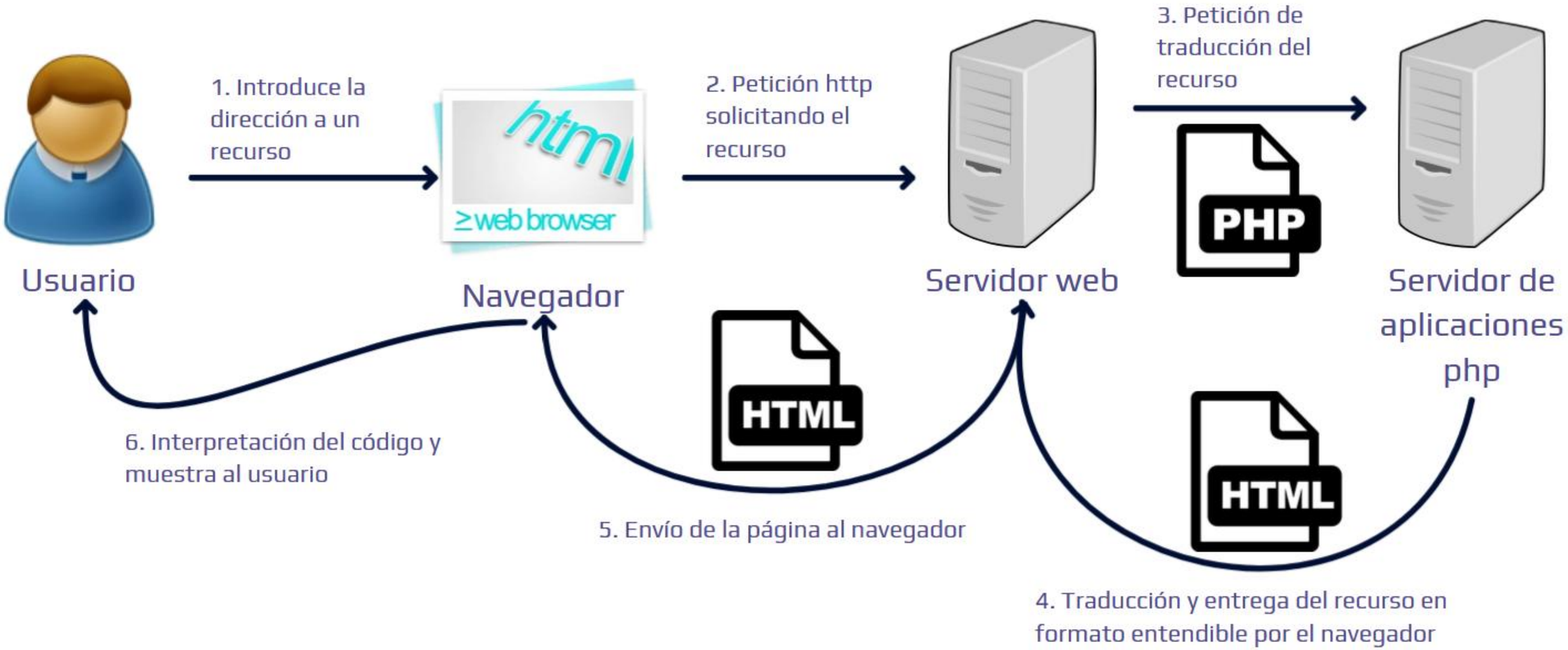


Desventajas:

- Carga en el servidor
- Dependencia de la red
- Menos interactividad
- sin JavaScript



ENFOQUES PARA EL DESARROLLO: DESARROLLO DEL LADO SERVIDOR (BACKEND)



ENFOQUE HÍBRIDO: FULL STACK

En la mayoría de las aplicaciones web modernas, se utiliza una combinación de ambos enfoques, conocido como desarrollo **full-stack**. Esto significa que se emplea un backend robusto para manejar datos y lógica de negocio, mientras que el frontend se encarga de la interfaz y la interacción del usuario.



DECISIONES PREVIAS

Antes de empezar a desarrollar una aplicación web, debemos configurar el entorno de desarrollo. Y para eso, antes tendremos que tomar una serie de decisiones sobre esa configuración.

- LENGUAJE DE PROGRAMACIÓN
- SERVIDOR DE APLICACIONES WEB
- ENTORNO DE DESARROLLO
- SERVIDOR WEB
- SERVIDOR DE BASES DE DATOS

XAMPP

Ahora que hemos tomado todas las decisiones, llega el momento de instalarlo todo. En un entorno de trabajo final, el que usaría realmente el cliente (**entorno de producción**), conviene instalar y configurar cada componente por separado, ya que hay que tener en cuenta aspectos como el número de usuarios que se prevee que tendrá la aplicación, el número de transacciones por minuto, la escalabilidad, la seguridad, etc.

Pero esta es una tarea difícil, tediosa y totalmente innecesaria para el objetivo que buscamos nosotros, que es configurar un entorno de trabajo apto para crear una aplicación web en nuestro ordenador, sin necesidad de conectarnos a internet, y hacer en él todas las pruebas necesarias. Es decir, queremos hacer una instalación en el **entorno de desarrollo.**

Para ello, existen lo que se conoce como **soluciones integradas**.
Concretamente, para PHP tenemos **XAMPP**, cuyas siglas vienen de **Cross Platform** (X), **Apache**, **MySQL**, **PHP** y **Perl**, ya que automáticamente instala y configura todas ellas.
Lo descargaremos e instalaremos desde su página oficial:
https://www.apachefriends.org/es/index.html



INSTALACIÓN DE VISUAL STUDIO CODE

Con el XAMPP instalado, ya solo nos queda instalar el IDE que, como habíamos dicho, será **Visual Studio Code**:
https://code.visualstudio.com/

Visual Studio Code

Después de instalarlo tenemos que agregarle la extensión **PHP: All-in-One PHP support.**
Con esto, tendremos nuestro entorno perfectamente configurado para empezar a programar.

