Unidad 1:
Arquitectura desarrollo web

Índice

1. Introducción

Las aplicaciones web suponen un cambio a las tradicionales aplicaciones de escritorio. Mientras que estas últimas se ejecutan en una máquina y contienen toda la lógica necesaria para su uso, las aplicaciones web utilizan **modelos distribuidos** con varios elementos que se comunican entre sí utilizando un **protocolo de comunicaciones**, y la aplicación también está distribuida de forma diferenciada entre los diferentes actores que participan.

2. Modelo cliente-servidor

Las aplicaciones web se basan en el modelo cliente-servidor, que se trata de un **modelo distribuido** con dos elementos diferenciados: clientes y servidores. En este modelo existe un **proceso central que se ejecuta en el servidor**, que contiene toda la lógica de negocio y los datos, y ofrece una serie de servicios a uno o más procesos cliente. En general, el *software* que se ejecuta en los clientes es simplemente un navegador web.

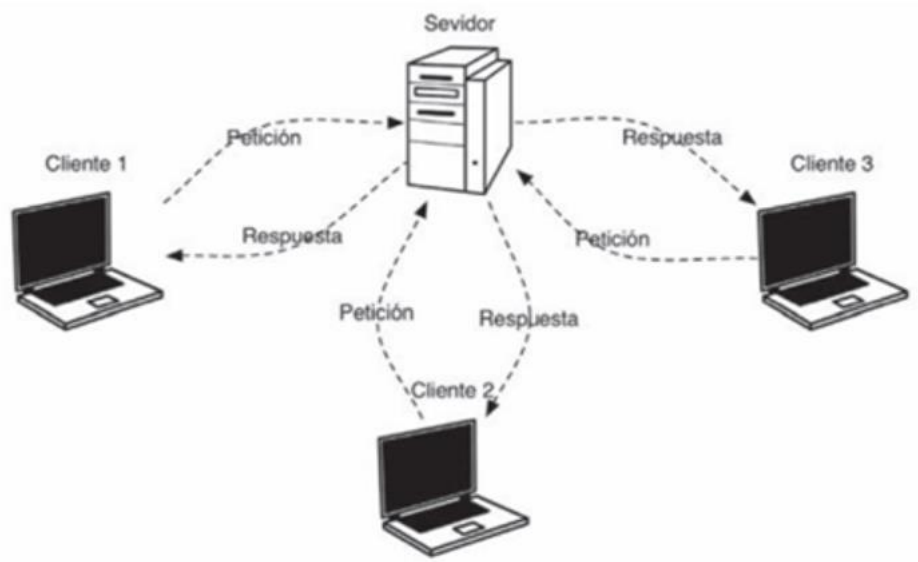

Ilustración 1. Modelo cliente-servidor.

El proceso servidor debe estar alojado en una máquina fácilmente accesible en la red, conociendo su dirección IP y número de puerto. Cuando un cliente requiere sus servicios, se conecta con el servidor, iniciando el proceso de comunicación. Cliente y servidor son **roles fijos que no cambian nunca**.

Los clientes realizan las peticiones al servidor utilizando un protocolo determinado (HTTP o HTTPS). El servidor recibe la petición y genera una respuesta que envía de vuelta al cliente.

2.1. Modelo de tres capas

La **arquitectura de tres capas** es una ampliación del modelo cliente-servidor visto anteriormente. La lógica de la aplicación se separa en:

1) **Capa de presentación**: se refiere a la interfaz de usuario. Muestra información y permite interactuar. Se ejecuta en un navegador, en el ordenador de la persona usuaria. Esta capa se comunica únicamente con la capa de negocio.
2) **Capa de negocio**: contiene la lógica propia de la aplicación. Se comunica con las otras dos capas. Se ejecuta en el servidor web. Recibe las acciones de usuaries de la capa de

presentación y, si es necesario, lanza las consultas de lectura o modificación necesarias a la capa de datos, pasando a la de presentación los resultados de dichas consultas.

3) **Capa de datos**: para gestionar la base de datos. Se comunica únicamente con la capa de negocio.

La finalidad de esta separación es crear aplicaciones más robustas, más fáciles de mantener y ampliar, con código reutilizable, en el que hacer cambios sea más sencillo.

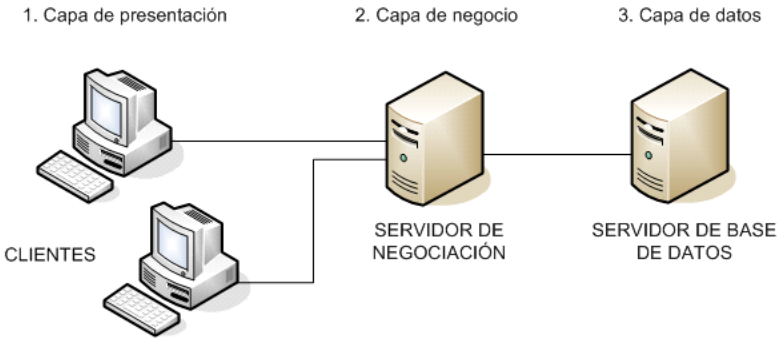

Ilustración 2. Arquitectura de tres capas.

3. Generación dinámica de páginas web

Desde la irrupción de la web 2.0 a principios de la década de los 2000, se empieza a hablar de web dinámica, dejando de lado a las webs estáticas utilizadas hasta ese momento.

Una página web estática estará creada utilizando solamente HTML+CSS, y mostrará siempre el mismo contenido. Como ventajas, podemos señalar que son más sencillas y más ligeras.

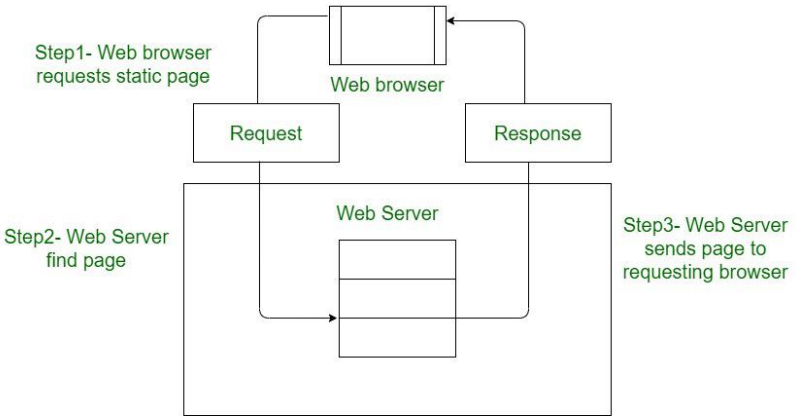

Ilustración 3. Página web estática. GeekforGeeks.

Cuando se utiliza un lenguaje de programación del lado del servidor, se pueden generar páginas dinámicas, es decir, que cambian en función de la petición del cliente, con la posibilidad de consultar una base de datos.

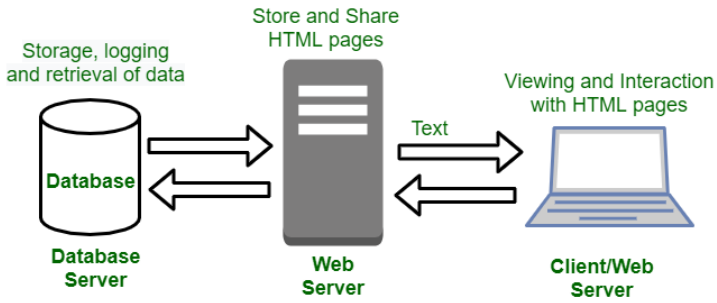

Ilustración 4. Página web dinámica. GeekforGeeks.

Las tecnologías empleadas en la generación de páginas web dinámicas son:

- En la parte ***front-end*** (cliente), que se ejecuta en un navegador web, se utiliza HTML, CSS y JavaScript.
- En la parte ***back-end*** (servidor), que se ejecuta en un servidor web + BBDD, se utilizan lenguajes de programación de entorno servidor, como pueden ser, entre otros:
    - PHP: Lenguaje de propósito general diseñado para el desarrollo de aplicaciones web dinámicas. Es no tipado, se embebe dentro de HTML entre las instrucciones `<?php` y `?>`, y tiene multitud de librerías y *frameworks* (Laravel, Symfony, CodeIgniter, Zend, etc.
    - Python
    - Ruby
    - JSP: versión Java de PHP
    - ASP.NET, de Microsoft

Una persona profesional se la denomina ***full-stack*** cuando domina tanto la parte *front* como *end*.

Anualmente, Stack Overflow, una de las mayores comunidades de desarrolladoras y desarrolladores del mundo, publica los resultados de la encuesta sobre las tecnologías más utilizadas. Puedes consultar los datos de 2024 aquí: https://survey.stackoverflow.co/2024/

3.1. Integración con los lenguajes de marcas

Las páginas dinámicas se componen de una parte estática basada en HTML (y, opcionalmente, CSS y JavaScript), y una parte dinámica en algún lenguaje de programación de servidor.

```
<!DOCTYPE html>
<html>
<head>
    <title>Prueba PHP</title>
</head>
<body>
    <?php
        echo "Hola mundo";
    ?>
</body>
</html>
```

Cuando solicitamos esta página desde un cliente, al acceder al código fuente de la página desde el navegador, comprobaremos que la parte de PHP no se envía al cliente, sino que se ejecuta en el servidor, se genera una salida, y es ésta la que se envía al cliente.

3.2. Single Page Application

Hoy en día está ganando peso las arquitecturas conocidas como **Single Page Application**. En ellas, en lugar de que con cada petición el servidor envía al cliente una web completa, en SPA el servidor solamente suministra una vez la web completa, y después solamente envía, en formato JSON, las nuevas informaciones solicitadas por el cliente. Se da más peso al cliente, liberando de carga de trabajo al servidor.

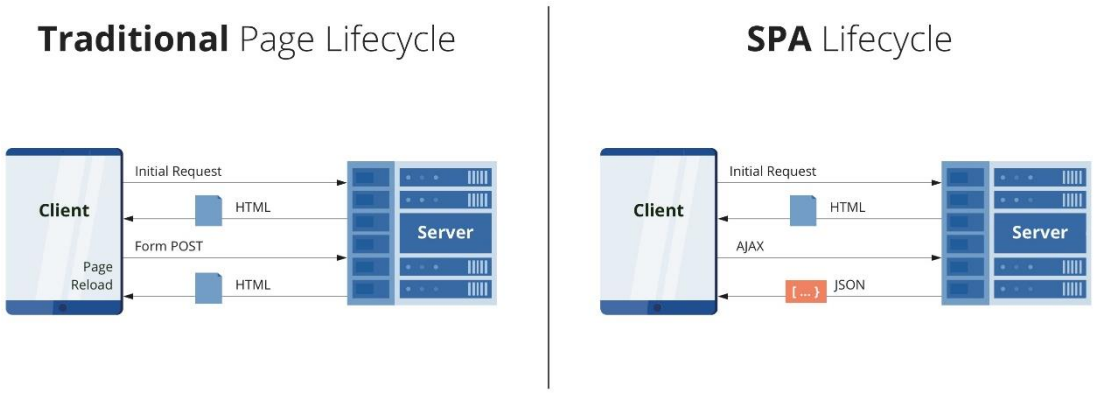

Ilustración 5. Arquitectura SPA (Single Page Application). DZone.

4. Herramientas de programación

4.1. Servidor web

Como hemos venido viendo, las aplicaciones web se ejecutan dentro de un servidor, por lo que es una pieza fundamental de este tipo de programación. Las aplicaciones se instalarán en dicho servidor, y los clientes, a través de un navegador web, consumirán (solicitarán) los servicios prestados por nuestro programa.

Existen multitud de servidores web. Entre los más conocidos y utilizadas podremos destacar Apache, Nginx (ambos de código abierto), Microsoft-IIS, LiteSpeed, etc.

4.2. Servidor de base de datos

Como hemos visto, el servidor web puede interactuar con la información almacenada en una base de datos. Esta base de datos puede seguir el modelo relacional (MySQL/MariaDB, PostgreSQL, Oracle, Microsoft SQL Server, SQLite…) o no relacionales / NoSQL (MongoDB, Firebase Realtime Database…).

4.3. XAMPP

XAMPP, o LAMP en su versión de Linux, es un paquete que incluye lo necesario para la programación web, pues incluye un servidor Apache con PHP y un servidor de base de datos MariaDB, entre otras cosas.

4.4. Docker

Docker utiliza contenedores online, una especie de máquina virtual muy reducida que solamente carga los elementos necesarios para nuestras aplicaciones. Así, podemos tener un contenedor con Apache, otro contenedor con MySQL, etc. Pueden interactuar entre ellos, y es la solución que vamos a emplear a lo largo del curso.

4.5. IDE

Los Entornos Integrados de Desarrollo son programas que aglutinan una serie de funcionalidades muy útiles a la hora de programar (editor de texto, acceso a documentación, *debug*, compilador, consola, control de versiones y dependencias, etc.).

En este curso, y para PHP, utilizaremos **Visual Studio Code**, con las siguientes extensiones:

- PHP: contiene el intérprete de PHP y otras herramientas útiles.
- PHP Extension Pack: incluye herramientas que facilitan la programación en PHP.
- PHP Server: muy interesante, pues nos lanza la aplicación que estemos desarrollando en un servidor web, de manera que la ejecución de nuestras aplicaciones se hace más rápida y sencilla.