# Introducción a las aplicaciones web

## 1.1 Evolución de las aplicaciones web

Las aplicaciones web han experimentado una transformación significativa desde sus inicios en la década de 1990. Comenzaron como simples páginas estáticas HTML y han evolucionado hasta convertirse en complejas plataformas interactivas que utilizamos diariamente.

\*\*Etapas clave de evolución:\*\*

- \*\*Web 1.0 (1990-2000)\*\*: Páginas estáticas con contenido unidireccional donde los usuarios solo podían leer información. Un ejemplo clásico fueron los primeros sitios corporativos que funcionaban como folletos digitales.

- \*\*Web 2.0 (2000-2010)\*\*: Surgió la web participativa con contenido generado por usuarios y mayor interactividad. Plataformas como Facebook, YouTube y Wikipedia ejemplifican esta etapa donde los usuarios pasaron de ser consumidores a creadores de contenido.

- \*\*Web 3.0 (2010-actualidad)\*\*: Caracterizada por aplicaciones web semánticas, personalizadas e inteligentes. Servicios como Netflix que aprenden de tus preferencias o asistentes virtuales como Siri representan esta evolución hacia experiencias más contextuales.

- \*\*Web 4.0 (emergente)\*\*: Integración de inteligencia artificial avanzada, realidad aumentada/virtual y conexión omnipresente. Ejemplos incipientes incluyen experiencias de compra inmersivas o plataformas de aprendizaje adaptativas.

## 1.2 Arquitectura de las aplicaciones web

La arquitectura de aplicaciones web define cómo se organizan los componentes y cómo interactúan entre sí.

\*\*Arquitecturas principales:\*\*

- \*\*Cliente-Servidor\*\*: Modelo fundamental donde el navegador (cliente) solicita recursos a un servidor web. Por ejemplo, cuando accedes a un periódico digital, tu navegador solicita artículos al servidor del periódico.

- \*\*Tres capas\*\*:

- Capa de presentación (interfaz de usuario)

- Capa de lógica de negocio (procesamiento)

- Capa de datos (almacenamiento)

Un ejemplo práctico sería una tienda online como Amazon, donde la interfaz que ves es la capa de presentación, el sistema que procesa pedidos es la capa de lógica, y las bases de datos con productos e inventario conforman la capa de datos.

- \*\*Microservicios\*\*: Arquitectura moderna donde la aplicación se divide en servicios pequeños e independientes. Netflix utiliza más de 700 microservicios para gestionar diferentes funcionalidades como recomendaciones, streaming y facturación.

- \*\*Serverless\*\*: Modelo donde los desarrolladores no gestionan servidores sino que escriben funciones que se ejecutan en respuesta a eventos. Un ejemplo es una aplicación de procesamiento de imágenes que redimensiona automáticamente fotos subidas a la nube.

## 1.3 Tecnologías para el desarrollo de aplicaciones web

El ecosistema tecnológico para desarrollo web es amplio y continúa evolucionando rápidamente.

\*\*Frontend (lado del cliente):\*\*

- \*\*HTML5\*\*: Estructura del contenido web. Ejemplo: definición de secciones, formularios y elementos multimedia.

- \*\*CSS3\*\*: Estilos y diseño visual. Ejemplo: aplicar diseño responsive para que un sitio se adapte a diferentes dispositivos.

- \*\*JavaScript\*\*: Interactividad y dinamismo. Ejemplo: validación de formularios o animaciones sin recargar la página.

- \*\*Frameworks frontend\*\*: React, Angular, Vue.js. Ejemplo: Instagram utiliza React para crear una interfaz de usuario fluida y componentes reutilizables.

\*\*Backend (lado del servidor):\*\*

- \*\*Lenguajes\*\*: PHP, Python, Java, Node.js, Ruby. Ejemplo: WordPress usa PHP para generar dinámicamente páginas web.

- \*\*Frameworks backend\*\*: Laravel, Django, Spring, Express. Ejemplo: Twitter utiliza Ruby on Rails para gestionar millones de tweets.

- \*\*Bases de datos\*\*: MySQL, MongoDB, PostgreSQL. Ejemplo: una aplicación de comercio electrónico almacenando inventario en PostgreSQL.

\*\*Tecnologías complementarias:\*\*

- \*\*APIs y servicios web\*\*: Permiten la comunicación entre aplicaciones. Ejemplo: una app de clima que consume datos meteorológicos de una API externa.

- \*\*Herramientas DevOps\*\*: Docker, Kubernetes, Jenkins. Ejemplo: implementación automática de actualizaciones en una aplicación bancaria.

## 1.4 Planificación de aplicaciones web

La planificación adecuada es crucial para el éxito de cualquier proyecto web.

\*\*Fases clave en la planificación:\*\*

1. \*\*Análisis de requisitos\*\*: Identificación de necesidades y objetivos. Ejemplo: para una plataforma educativa, determinar qué funcionalidades necesitan estudiantes y profesores.

2. \*\*Diseño conceptual y prototipado\*\*: Creación de wireframes y maquetas. Ejemplo: diseñar la estructura de navegación y flujos de usuario para una aplicación de reservas de hotel.

3. \*\*Selección tecnológica\*\*: Elección de tecnologías adecuadas según requisitos. Ejemplo: optar por MongoDB para una aplicación que maneja datos no estructurados como perfiles de usuario con atributos variables.

4. \*\*Planificación de recursos\*\*: Estimación de tiempo, presupuesto y equipo. Ejemplo: determinar que se necesitarán tres desarrolladores durante cuatro meses para crear una aplicación de gestión de inventario.

5. \*\*Estrategia de implementación\*\*: Plan de desarrollo, pruebas y despliegue. Ejemplo: implementar primero el módulo de autenticación, luego funcionalidades principales y finalmente características secundarias.

6. \*\*Mantenimiento y actualización\*\*: Planificación para el ciclo de vida post-lanzamiento. Ejemplo: programar actualizaciones trimestrales y monitoreo continuo del rendimiento.

Una buena planificación debe considerar también aspectos como seguridad, escalabilidad, accesibilidad y experiencia de usuario desde el inicio, evitando costosas modificaciones posteriores.