# PROGRAMA EDUCATIVO EN:

Tecnologías de la Información en Desarrollo y Gestión de Software

## Desarrollo Web Profesional

## NOMBRE DEL ESTUDIANTE:

Dana Lizbeth Castañeda Sánchez

# Arquitectura de información y navegación accesible

8vo Cuatrimestre “A”

# Bloque 1: Fundamentos y Arquitectura

1. Página Web vs. Aplicación Web

Aunque a menudo se usan como sinónimos, la diferencia radica en la interacción y el propósito.

| Característica | Página Web (Estática/Informativa) | Aplicación Web (Dinámica) |
|---|---|---|
| **Función principal** | Consumir información (leer, ver). | Realizar tareas o manipular datos. |
| **Interacción** | Baja (clics en enlaces, lectura). | Alta (crear cuentas, editar contenido, procesar pagos). |
| **Actualización** | El contenido cambia poco. | Los datos cambian constantemente en tiempo real. |
| **Ejemplo** | Un blog personal o una landing page de un producto. | Un gestor de tareas o un sistema de banca en línea. |

2. Ejemplos de Aplicaciones Web Profesionales

- SaaS (Software as a Service): Slack, Trello, Salesforce.
- Herramientas de Diseño: Figma, Canva.
- Gestión de Proyectos: Jira, Monday.com.
- Streaming/Contenido: Netflix (su interfaz de navegador), YouTube Studio.

3. ¿Qué problemas resuelve el software?

El software no es solo código sino es una herramienta de solución de problemas:

**Automatización:** Realizar tareas repetitivas sin error humano.

**Gestión de Datos:** Almacenar, organizar y recuperar volúmenes masivos de información.

**Comunicación:** Acortar distancias y facilitar la colaboración en tiempo real.

**Accesibilidad:** Permitir que servicios (como la banca o educación) lleguen a cualquier persona con internet.

4. Arquitectura General de Aplicaciones Web

**Frontend (Lado del cliente):** Es lo que el usuario ve y toca. Se construye principalmente con HTML, CSS y JavaScript (y frameworks como React o Vue). Su foco es la interfaz (UI) y la experiencia (UX).

**Backend (Lado del servidor):** Es la "lógica detrás de cámaras". Se encarga de procesar datos, gestionar usuarios y comunicarse con la base de datos. Lenguajes comunes: Python, Node.js, PHP, Java.

**Infraestructura / Entornos:**

- Desarrollo: Donde los programadores escriben código.
- Staging/Pruebas: Una copia idéntica a la realidad para buscar errores antes de lanzar.
- Producción: El entorno real donde entran los usuarios finales.

# Bloque 2: Arquitectura de Información y Accesibilidad

Diseñar un sistema usable requiere pensar en cómo el cerebro humano procesa la información.

**Conceptos Clave**

- Arquitectura de Información (AI): Es la disciplina de organizar y etiquetar sitios web para que los usuarios encuentren lo que buscan de forma intuitiva. Es el "plano" de la casa antes de decorarla.
- Jerarquías de Contenido: Organizar los elementos por orden de importancia. Se logra mediante tamaños de fuente, colores y ubicación (lo más importante suele ir arriba a la izquierda en culturas occidentales).
- Patrones de Navegación: Estructuras comunes que el usuario ya conoce, como:
    - *Menú Hamburguesa* (móvil).
    - *Barra de navegación persistente* (superior).
    - *Migas de pan (Breadcrumbs)* para saber dónde estás exactamente.

**Accesibilidad (A11y)**

Un sistema profesional debe ser usable por todos, incluyendo personas con discapacidades motrices o visuales.

- Orden de Tabulación: Es la secuencia en la que el foco se mueve cuando presionas la tecla Tab. Debe ser lógica (de arriba a abajo, de izquierda a derecha) y no saltar de forma errática.
- Navegación por Teclado: Un usuario debe poder completar todas las acciones críticas (comprar, registrarse, enviar) sin tocar el mouse. Esto incluye usar Enter para activar botones y Esc para cerrar modales.
- Accesibilidad sin Mouse: Se logra mediante el uso de etiquetas HTML semánticas (usar <button> en lugar de un  que parece botón) para que los lectores de pantalla y navegadores entiendan la función de cada elemento.

**Contribución Técnica Verificable**

La validez técnica de Rapiti se fundamenta en los siguientes puntos documentados:

1. Arquitectura de Tres Capas: La separación clara entre el Frontend (React), Backend (Node.js) y Base de Datos (PostgreSQL) asegura un sistema escalable y profesional.
2. Solución de Software vs. Métodos Manuales: La aplicación web es verificable como solución al centralizar datos de 15-20 productos básicos, reduciendo el sobrecosto estimado del 10-15% en compras locales.
3. Accesibilidad Operativa: El diseño garantiza que el orden de tabulación y la navegación por teclado permitan que tanto compradores como tenderos utilicen la plataforma sin necesidad de un mouse, cumpliendo estándares profesionales de usabilidad.
4. Infraestructura Moderna: El uso de Docker y CI/CD mediante GitHub asegura que el despliegue del MVP sea consistente y libre de errores críticos.

# Fuentes Bibliográficas


- Garrett, J. J. (2011). *The elements of user experience: User-centered design for the web and beyond*.New Riders http://www.jjg.net/elements/pdf/elements.pdf
- Haverbeke, M. (2018). *Eloquent JavaScript: A modern introduction to programming* (3ra ed.). No Starch Press. https://eloquentjavascript.net/
- Pressman, R. S., & Maxim, B. R. (2020). *Software engineering: A practitioner's approach* (9naed.).McGraw-Hill Education. https://www.mheducation.com/highered/product/software-engineering-practitioner-s-approach-pressman-maxim/M9781260235197.html
- Morville, P., & Rosenfeld, L. (2015). *Information architecture: For the web and beyond* (4ta ed.). O'Reilly Media. https://www.oreilly.com/library/view/information-architecture-4th/9781491913529/
- Nielsen, J. (1994). *Usability engineering*. Morgan Kaufmann. https://www.nngroup.com/books/usability-engineering/
- Horton, S., & Quesenbery, W. (2014). *A web for everyone: Designing accessible user experiences*. Rosenfeld Media. https://rosenfeldmedia.com/books/a-web-for-everyone/
- World Wide Web Consortium. (2023, 5 de octubre). *Web Content Accessibility Guidelines (WCAG) 2.2*. https://www.w3.org/TR/WCAG22/