UNIVERSIDAD PRIVADA DEL VALLE

FACULTAD DE INFORMATICA

INGENIERIA DE SISTEMAS INFORMÁTICOS

LA PAZ

PROGRAMACION WEB I

Manual de técnico de la página web Nexo News

Grupo “B”

Estudiante:

Bustillos Serrano Julio Mateo

Docente: Ing. Miranda Ordoñez Henry

La Paz 11 de diciembre del 2025

Gestión II – 2025

Contenido

1. Introducción

Nexo News es una página web de noticias de videojuegos que centraliza contenido externo proveniente de APIs públicas. Su objetivo es ofrecer un espacio informativo y comunitario sin generar contenido propio. Responde a la necesidad de un proyecto académico que demuestre dominio de HTML, CSS y JavaScript en un entorno front-end puro.

2. Descripción General del Sistema

Nexo News es una aplicación web de naturaleza front-end puro, desarrollada como un proyecto académico para la materia Programación Web 1. No crea contenido propio; en cambio, suma y presenta información de fuentes externas auténticas, dirigiendo siempre al usuario a las fuentes originales y respetando el crédito correspondiente.
El sistema está formado únicamente por tecnologías en el lado del cliente:

- HTML5 como fundamento de contenido y estructura semántica.
- CSS3 para una estructura visualmente consistente, accesible y adaptable.
- JavaScript, sin utilizar marcos ni librerías externas, para la lógica dinámica.

Su arquitectura se fundamenta en un enfoque de "mejora continua": cada página contiene información estática directamente en el HTML como respaldo, garantizando que los usuarios siempre puedan acceder a datos útiles, aun sin conexión a internet.
El sistema solo tiene comunicación con un único servicio externo:

FreeToGame: Para conseguir una relación actualizada de juegos gratuitos, que incluya metadatos como la plataforma, el género y el vínculo para descargar.

CorsProxy.io, un proxy público gratuito, se emplea para eludir las limitaciones de seguridad del navegador (CORS) cuando se entra en esta API.

Por otra parte, las noticias son obtenidas de un archivo JSON local (json/json.json), este archivo contiene un conjunto estable de artículos auténticos obtenidos de fuentes confiables, como Xataka, IGN, Vandal y VidaExtra. Se carga directamente desde el cliente sin requerir conexión a internet. Esta decisión asegura que haya disponibilidad total, previene las dependencias externas y garantiza una experiencia de usuario uniforme en cualquier contexto, incluidas las exposiciones académicas sin conexión a Internet.

3. Arquitectura del sistema

Nexo News funciona con una arquitectura de front-end puro, es decir, sin base de datos ni servidor. El sistema se compone de páginas HTML estáticas que, al cargar, se enriquecen dinámicamente mediante el uso de JavaScript para consumir datos locales y, en un caso específico, una API externa mediante un proxy público.

Flujo General
Cuando un usuario accede a una página (como principal.html, por ejemplo), lo primero que hace el navegador es cargar el HTML base, que contiene contenido estático de respaldo (por ejemplo, juegos, noticias o comentarios de muestra). Después, JavaScript detecta la página actual a través de elementos únicos (por ejemplo, #contenedor-juegos o #img-principal) y actúa de acuerdo con el contexto:

- Para las noticias: se carga el archivo local json/json.json, que incluye un grupo invariable de artículos genuinos adquiridos de fuentes como Vandal, VidaExtra, IGN y Xataka. Este método asegura que siempre esté disponible, incluso sin conexión a internet, y evita la necesidad de depender de factores externos.
- Para los videojuegos: para prevenir errores de CORS, se usa CorsProxy.io como intermediario para hacer una solicitud a la API de FreeToGame a través de fetch(). Cuando la API responde satisfactoriamente, los datos reales sustituyen al contenido base. Si hay un fallo (por límite diario, caída del servicio o falta de conexión), el contenido estático sigue siendo visible.

Esta configuración apoya la filosofía de mejora continua: el cliente siempre obtiene una experiencia beneficiosa, ya sea con datos actualizados o con el soporte integrado en HTML.

Uso de apis y recursos externos

- FreeToGame: única API externa usada, exclusivamente para la sección Top Juegos.
- CorsProxy.io: proxy público gratuito, utilizado solo para FreeToGame, para sortear las restricciones de seguridad del navegador (CORS).
- json/json.json: archivo local que almacena todas las noticias. No requiere conexión ni proxy, y se gestiona íntegramente desde el cliente.

4. Estructura del proyecto (carpetas y archivos)

Para simplificar la navegación y el mantenimiento, el proyecto está estructurado en carpetas de manera lógica y clara.

Los cinco archivos HTML principales se encuentran en la raíz del directorio:

noticias.html, principal.html, contactos.html, foro.html y top-juegos.html.

Cada uno de ellos es una sección del sitio y está conectado a través del menú de navegación.

El archivo estilos.css, ubicado en la carpeta css/, contiene todas las normas del sitio web: variables de color, diseño adaptable (con Flexbox y Grid), estilos de los componentes (botones, formularios, tarjetas) y consultas a medios.

El archivo scripts.js, que contiene la lógica dinámica, se encuentra en la carpeta scripts/:

- Carga y procesamiento de la información del archivo json/json.json (noticias locales).
- Uso de la API FreeToGame (a través de un proxy CORS).
- Administración del foro (filtrado, publicación).
- Redirecciones y menú hamburguesa.

Los activos visuales estáticos del sitio, que incluyen el logo (Pylon.png) y las imágenes de respaldo para los juegos y las noticias, se encuentran en la carpeta imagenes/.

La carpeta json/ es un componente fundamental de la arquitectura, pues guarda el archivo json.json, que tiene una colección fija de noticias auténticas adquiridas a través de fuentes como Xataka, Vandal, VidaExtra e IGN. El archivo se carga de manera local a través de fetch(), lo que asegura que el contenido noticioso funcione sin conexión y no dependa de elementos externos.
Esta organización pone de manifiesto el principio de división de responsabilidades, lo que posibilita modificaciones o expansiones futuras sin que estas repercutan en otras secciones del sistema.

5. Funcionamiento interno

El sistema se compone de tres niveles tecnológicos que operan en coordinación: Para la estructura, HTML; para el diseño, CSS; y para la interactividad, JavaScript. Siguiendo las normas de separación de responsabilidades, cada capa fue construida por separado.

5.1 HTML

Nexo News emplea HTML5 semántico con el objetivo de especificar con exactitud la finalidad de cada bloque de contenido. Cada página contiene contenido básico directamente en el HTML, lo que asegura que el usuario pueda acceder a información útil en todo momento, incluso si JavaScript se encuentra desactivado o no funciona.
Por ejemplo, en el archivo principal.html, las noticias principales y secundarias se establecen como tarjetas fijas que incluyen una imagen, un encabezado y una descripción:

```
<article id="noticia-principal" class="tarjeta_principal">
<img id="img-principal" src="imagenes/EldenRing.jpg" alt="Portada: Elden Ring DLC" />
<h2 id="titulo-principal" class="titulo-tarjeta">Elden Ring: Shadow of the Erdtree ya tiene fecha</h2>
<p id="descripcion-principal" class="texto-tarjeta">El tan esperado DLC de FromSoftware llegará el 21 de junio...</p>
<a id="enlace-principal" href="https://ejemplo.com/elden-dlc" target="_blank">Leer más</a>
</article>
```

5.2 CSS

El archivo de estilos, estilos.css, está dividido en secciones con comentarios que representan la estructura del sitio: encabezado, contenido principal, formularios, tarjetas, grillas (o cuadrículas), pie de página y diseño responsivo. Para centralizar sombras y colores, se emplean variables CSS en :root:

```
:root {
  --color-fondo: rgb(59, 59, 59);
  --color-borde: rgb(0, 147, 188);
  --sombra: 0px 4px 10px rgb(0, 255, 153);
}
```

El diseño se construye mediante la utilización de CSS Grid y Flexbox. La página principal utiliza Grid para obtener una disposición similar a la de una revista:

```
.seccion-superior {
  display: grid;
  grid-template-columns: 1fr 1fr 1fr;
  gap: 1.5rem;
}
.seccion-superior > .tarjeta_principal {
  grid-row: span 2;
  grid-column: 1;
}
```

Para que haya responsividad, se implementan media queries con una perspectiva mobile-first. En dispositivos móviles (≤850px), el menú se transforma en botón hamburguesa y todas las cuadrículas se compactan en una sola columna:

```
@media (max-width: 850px) {
  .seccion-superior { grid-template-columns: 1fr; }
  .grilla { grid-template-columns: 1fr; }
}
```

5.3 JavaScript

La lógica dinámica está contenida en scripts.js, que se ejecuta después de que ocurra el evento DOMContentLoaded. La siguiente es la secuencia principal:

- Para las noticias: se utiliza fetch() para cargar el archivo local json/json.json. Un grupo de artículos auténticos, que se obtuvieron a partir de fuentes como Vandal, VidaExtra e IGN, es lo que contiene este archivo. El contenido estático de respaldo es reemplazado por los datos que se inyectan en el DOM a través de appendChild() y createElement().

- Para los juegos: se utiliza la API de FreeToGame por medio de CorsProxy.io a fin de prevenir fallos CORS. El contenido base sigue siendo visible en caso de que la API no funcione.
- Para el foro: los comentarios se crean de manera local utilizando insertBefore() y luego se filtran a través del atributo data-categoria.

En ningún momento el sistema guarda datos. Los comentarios del foro y los mensajes de contacto solo están presentes en la sesión activa del navegador. Esta resolución fortalece la naturaleza académica del proyecto y previene dependencias que no son necesarias.
Además, la página leer-mas.html obtiene el índice de la noticia por medio de la URL (por ejemplo, ?noticia=3) y carga todo el contenido desde el archivo json.json, lo que asegura una experiencia consistente y totalmente fuera de línea.

6. Gestión de errores

El diseño de Nexo News se enfoca en ser robusto frente a fallos, asegurando que el usuario tenga siempre una experiencia provechosa, aun cuando los servicios externos no estén disponibles. Esta filosofía se aplica en tres niveles esenciales.
Como las noticias se extraen de un archivo JSON local (json/json.json), su acceso está asegurado en cualquier contexto, incluso sin conexión a la red. Así pues, la única parte susceptible de fallar es la sección de juegos, que sí está sujeta a la API externa de FreeToGame.
Si esta API no funciona (por un límite diario, por que el servicio se cae o por falta de conexión), el contenido de juegos de muestra que está directamente en el HTML sigue siendo visible, lo cual garantiza que la página nunca quede vacía.

En segundo lugar, cuando hay un error al cargar imágenes (por recursos eliminados, servidores caídos o URLs rotas), cada imagen se protege a través del evento onerror. Si no es posible cargar una imagen, de manera automática se sustituye por el logo de respaldo Pylon.png.

En tercer lugar, se aplican validaciones fundamentales desde el lado del cliente en los formularios. En el foro, si el usuario intenta enviar un comentario vacío, se muestra una alerta y se interrumpe la publicación.

7. Consideraciones de seguridad

A pesar de que Nexo News es un proyecto front-end sin backend, se han tomado decisiones deliberadas en cuanto a la seguridad, en concordancia con las mejores prácticas para los entornos académicos de desarrollo.
Para incluir contenido creado por el usuario en los formularios, no se utiliza innerHTML. En el foro, el texto del comentario se incorpora de manera implícita a través de textContent al crear la tarjeta:

```
article.innerHTML=`<p><strong>${categoria}•${fecha}</strong><br>${texto}</p>;
```

Aunque aquí sí se usa innerHTML, el texto del usuario (${texto}) no se filtra, lo que podría permitir inyección de scripts (XSS) si el proyecto tuviera persistencia o fuera compartido entre usuarios. Dado que los comentarios no se almacenan ni se comparten (solo existen en la sesión local), el riesgo es prácticamente nulo, pero se reconoce como una limitación del enfoque académico. En una aplicación real, se sanitizaría el texto antes de insertarlo.

8. Glosario

8.1 css

***:** selector universal que, al emplearse con padding: 0, margin: 0 y box-sizing: border-box, se utiliza para establecer una base de diseño ordenada y predecible; su función es aplicar estilos a todos los elementos del documento.

box-sizing: border-box: Modifica el modelo de caja del navegador para que la altura y el ancho de un elemento contengan su borde (border), relleno (padding) y contenido; evita así desbordamientos y permite tener más control sobre los layouts.

:root: se trata de un pseudo-selector que representa el elemento ; sirve como contenedor global para la definición de variables CSS personalizadas (con la sintaxis --nombre-variable) que se pueden utilizar en todo el documento.

--nombre-variable: sintaxis para declarar una variable CSS; posibilita el almacenamiento de valores que se pueden reutilizar (como las sombras o los colores) y la centralización de la administración del diseño, lo cual simplifica futuras actualizaciones.

rgb(r, g, b): función que establece colores a través de sus elementos azul, verde y rojo (valores entre 0 y 255). Para este proyecto se elige esta opción ya que es más explícita y legible que la notación hexadecimal, sobre todo en situaciones pedagógicas.

display: flex: activa el modelo de diseño Flexible Box dentro de un contenedor; hace posible la alineación, distribución y ajuste de elementos secundarios en una única dimensión (ya sea fila o columna), lo que es perfecto para menús, formularios y encabezados.

display: grid: habilita el modelo de layout CSS Grid; posibilita la elaboración de diseños bidimensionales (columnas y filas) con un control exacto sobre el tamaño y la posición de los elementos, tal como en una página principal de revista.

fr (fracción): es una unidad de medida en CSS Grid que denota una fracción del espacio libre disponible. Para generar columnas con el mismo ancho que se ajustan al contenedor, se emplea en grid-template-columns: 1fr 1fr 1fr.

gap: propiedad que determina el espacio homogéneo entre columnas y filas en diseños de Grid o Flexbox; sustituye los márgenes y previene que se produzcan colapsos del diseño.

position: sticky: un valor de posición que combina el comportamiento relativo y fijo; el elemento se mueve normalmente hasta llegar a una posición de desplazamiento específica (por ejemplo, top: 0) y, después, se "pega" a esa posición.

z-index: propiedad que regula la profundidad (orden de apilamiento) de los elementos posicionados; las cifras más altas se presentan encima de las más bajas, lo cual es fundamental para que el encabezado permanezca visible al desplazarse.

@media: regla que solo aplica estilos bajo ciertas condiciones (por ejemplo, el ancho de la pantalla); es el fundamento del diseño adaptable en Nexo News, con puntos de interrupción a 850px, 768px y 1024px.

flex-wrap: wrap: propiedad de Flexbox que posibilita que los elementos hijos se distribuyan en varias líneas si no tienen espacio en una única línea, lo que previene el desbordamiento en pantallas con dimensiones reducidas.

transition: propiedad que describe una animación suave cuando se modifican los valores de otras propiedades (como el color o la transformación); aumenta la sensación de interactividad y fluidez en la interfaz.

transform: scale(): función que modifica (ya sea al aumentar o disminuir) el tamaño de un elemento; se utiliza en :hover para producir un leve efecto de "elevación" sobre tarjetas.

box-shadow: propiedad que crea una sombra alrededor de un elemento; en Nexo News se utiliza con un color verde neón (rgb(0, 255, 153)) para enfatizar la identidad visual y proporcionar una impresión de profundidad.

border-radius: propiedad que permite suavizar las esquinas de un componente; valores como 20px generan un aspecto moderno y atractivo en botones, tarjetas y contenedores.

object-fit: cover: propiedad que, al recortar lo necesario para llenar el espacio, mantiene la proporción de una imagen dentro de su contenedor. Asegura que las imágenes de juegos y noticias no se deformen.

:hover: Es una pseudo-clase que se aplica a los elementos cuando el cursor del mouse pasa por encima de ellos; se utiliza para proporcionar comentarios visuales en botones, enlaces y tarjetas.

:focus: pseudoclase que implementa estilos cuando un componente (por ejemplo, un campo de formulario) es seleccionado por el teclado o mediante clic; contribuye a mejorar la usabilidad y la accesibilidad.

min-height: propiedad que define la altura mínima que un elemento tiene que tener; se utiliza con 100vh en para garantizar que el diseño cubra, como mínimo, toda la pantalla del navegador.

max-width: establece un límite para el ancho de un elemento. En , al aplicar max-width: 80%, se logra centrar el contenido y optimizar la legibilidad en pantallas anchas.

var(--nombre): función que obtiene el valor de una variable CSS que ha sido definida en :root; posibilita la reutilización, coherente y mantenible, de colores, sombras y otros valores.

8.2 JavaScript

fetch(): función nativa para realizar peticiones HTTP a APIs o servidores; devuelve una promesa que resuelve en un objeto Response

.then(): método de una promesa que se ejecuta si la operación anterior se resuelve con éxito; se usa para procesar la respuesta de una API tras convertirla a JSON.

.catch(): método de una promesa que se ejecuta si ocurre un error en cualquier punto de la cadena; en Nexo News, se usa para evitar que la página falle y mantener el contenido base visible.

res.ok: propiedad del objeto Response que es true si el código de estado HTTP está en el rango 2xx (éxito); se usa para validar que la respuesta del servidor es válida antes de procesarla.

res.json(): método que lee el cuerpo de la respuesta y lo convierte en un objeto JavaScript; se utiliza porque las APIs externas devuelven datos en formato JSON.

Promise.reject(): función que crea una promesa rechazada manualmente; se usa para forzar un error si la respuesta del servidor no cumple con las expectativas (ej. status !== 'ok').

document.getElementById(): método que selecciona un elemento del DOM por su atributo id único; es la forma más directa y eficiente de acceder a elementos específicos como la noticia principal o el formulario.

document.querySelector(): método que selecciona el primer elemento que coincida con un selector CSS; se usa para obtener el formulario del foro o el botón hamburguesa.

createElement(): método que crea un nuevo elemento HTML en memoria; se utiliza para construir dinámicamente tarjetas de noticias, juegos o comentarios sin inyección directa de HTML.

.appendChild(): método que añade un nodo como último hijo de otro; se usa para insertar nuevas tarjetas en la grilla del foro o de noticias.

.**insertBefore():** método que inserta un nodo antes de otro existente; en el foro, se usa para añadir el nuevo comentario al inicio de la lista.

.innerHTML: propiedad que establece o devuelve el contenido HTML de un elemento; se usa con precaución para insertar texto estructurado (como y
 en los comentarios del foro), reconociendo que en entornos con persistencia requeriría sanitización.

.textContent: propiedad que establece o devuelve solo el texto de un elemento (sin HTML); se usa para actualizar títulos y descripciones de noticias porque es más seguro y evita inyección de código.

.style.cssText: propiedad que permite asignar múltiples estilos CSS en una sola cadena; se usa para aplicar reglas específicas a enlaces dentro de tarjetas dinámicas.

.classList.toggle(): método que añade una clase si no está presente, o la quita si está; se utiliza en el menú hamburguesa para alternar entre los estados "abierto" y "cerrado".

.dataset: propiedad que permite acceder a atributos de datos personalizados en HTML (como data-categoria="opinion"); se usa en el foro para identificar la categoría de cada comentario y habilitar el filtrado.

window.open(): función que abre una URL en una nueva pestaña del navegador; se utiliza en los botones "Jugar" y enlaces "Leer más" para redirigir al usuario a fuentes externas.

onerror (en <img>): evento que se dispara si una imagen falla al cargar; se usa para reemplazar automáticamente cualquier imagen rota por el logo de respaldo Pylon.png.

new Date(): constructor que crea un objeto con la fecha y hora actual del sistema; se utiliza para generar la fecha de publicación de los comentarios en el foro.

.padStart(2, '0'): método de cadena que rellena el inicio de una cadena con un carácter específico hasta alcanzar una longitud mínima; se usa para formatear días y meses con dos dígitos (ej. 5 → "05").

.slice(inicio, fin): método de arreglo que extrae una porción desde el índice inicio hasta fin (sin incluirlo); se usa para dividir las noticias de la API en grupos (principal, secundarias, inferiores).

.trim(): método de cadena que elimina espacios en blanco al inicio y al final; se utiliza en el foro para validar que el comentario no esté vacío.

encodeURIComponent(): función que codifica una URI para que caracteres especiales (como : o /) no rompan la estructura de la URL; se usa al construir la URL del proxy para FreeToGame.

window.location.origin: propiedad que devuelve el origen de la página actual (protocolo + dominio + puerto); se usa en la función crearImagen() para construir la URL absoluta del logo de respaldo y evitar bucles infinitos en onerror.