**PROCESO DE GESTIÓN DE FORMACIÓN PROFESIONAL INTEGRAL FORMATO GUÍA DE APRENDIZAJE**

1. **IDENTIFICACIÓN DE LA GUIA DE APRENDIZAJE** 
   * + **Denominación del Programa de Formación**: Tecnología en Análisis y Desarrollo de Software
     + **Código del Programa de Formación:** 3065369
     + **Nombre del Proyecto:** Construcción de software integrador de tecnologías orientadas a servicios
     + **Fase del Proyecto:** Ejecución
     + **Actividad de Proyecto:** Introducciónal diseño de aplicaciones web**,** maquetación, de una página web.
     + **Competencia:** DESARROLLAR LA SOLUCIÓN DE SOFTWARE DE ACUERDO CON EL DISEÑO Y METODOLOGÍAS DE DESARROLLO
     + **Resultados de Aprendizaje Alcanzar:** CREAR COMPONENTES FRONT-END DEL SOFTWARE DE ACUERDO CON LAS NECESIDADES DEL CLIENTE.
     + **Duración de la Guía:** 12 Horas

1. **PRESENTACIÓN Aprendiz SENA:**

El diseño eficiente y mantenible en CSS es una habilidad fundamental en el desarrollo web moderno. Esta guía de aprendizaje te propone sumergirte en el estudio de metodologías de diseño CSS con énfasis en BEM (Bloque, Elemento, Modificador), una estrategia que potencia la escalabilidad, claridad y reutilización del código. Al dominar estas metodologías, fortalecerás tu capacidad para construir sistemas de estilos coherentes, robustos y de fácil mantenimiento, cualidades esenciales para proyectos colaborativos y profesionales.

Esta guía está diseñada para organizar tu aprendizaje de forma progresiva, orientándote hacia un desarrollo integral tanto a nivel técnico como personal. A través de actividades autónomas, colaborativas y reflexivas, conectarás conocimientos previos sobre CSS con nuevas prácticas y patrones de diseño. Se promueve el autoaprendizaje organizado y el trabajo en equipo como base para una experiencia significativa, activa y aplicada.

1. **FORMULACIÓN DE LAS ACTIVIDADES DE APRENDIZAJE**

**A. Actividades de reflexión inicial.**

Analizarás cómo organizas actualmente tus hojas de estilo y qué problemas has identificado. Reflexionarás sobre la necesidad de estructuras claras en CSS y el papel de las metodologías.

**Actividades de aprendizaje:**

* + Escribe un breve ensayo (300 palabras) sobre tu experiencia con CSS: ¿Qué problemas has tenido al diseñar los ejercicios expuestos en formación?

Mi experiencia con css

En este ensayo presentare mis dificultades o debilidades a la hora de trabajar con css dicha herramienta que se utiliza para dar estilos a un código HTML, con el objetivo de dar a conocer y reflexionar sobre mis deficiencias con dicha herramienta.

Problemas con los que me he enfrentado con el css durante mi aprendizaje han sido poder modificar diferentes herramientas que tienen ciertas funcionalidades muy similares y el punto de poder adaptarme a ellas sin confusión, el movimiento o el posicionamiento de los elementos en el HTML han sido una leve dificultad, en otros aspectos de la hoja de estilos el poder identificar los tamaños las proporciones de las mismas, poder identificar un rango para dichos elementos un espaciado.

Breakpoints este fue un tema que me ha costado un poco entender utilizando el max- width o el min-width el poder identificar en qué momento se aplicaban los estilos al modificar el tamaño de la interfaz de la web y en su momento poder lograr que la pagina tome un estilo justificado sin que se desborde o se dañe la apariencia de la misma. En las imágenes lograr obtener un tamaño que se agradable al usuario de manera que no se vean pixeladas y obtener un buen posicionamiento.

Estas deficiencias se superarán creando una ruta de aprendizaje poniendo en práctica u apoyándome con diferentes metodologías como “bem” para solidificar mis conocimientos en el área para agregar hay otro método el cual me llamo la atención el Smacss “arquitectura escalable y modular para css” es una guía de estilos simple que sigue 5 categorías a fin de documentar y adaptar el proceso de diseños de un sitio web y siguiendo estas reglas simples se puede ordenar el código fácilmente , el código será escalable y fácil de entender.

* + Realiza una lista de problemas técnicos comunes asociados al desorden en CSS.

Conflicto de especificidad.

Cascada y orden de declaración.

Selectores anidados.

Posicionamiento de los elementos.

Tener una mala distribución del código.

Redundancias.

* + Investiga y define el concepto de "deuda técnica" en CSS.

Deuda técnica: es la acumulación de malas prácticas un código desorganizado con atajos o soluciones temporales para agilizar el progreso del código, en definición será un código tedioso de leer, estilos sobrescritos, estilos innecesarios, entre otros.

El rendimiento del css se vera afectado por redundancias, haciendo esto un difícil mantenimiento se complicarán las modificaciones de los estilos y el desarrollo de nuevas funcionalidades y la colaboración de entre diferentes desarrolladores.

**Como evitamos la “deuda técnica”.**

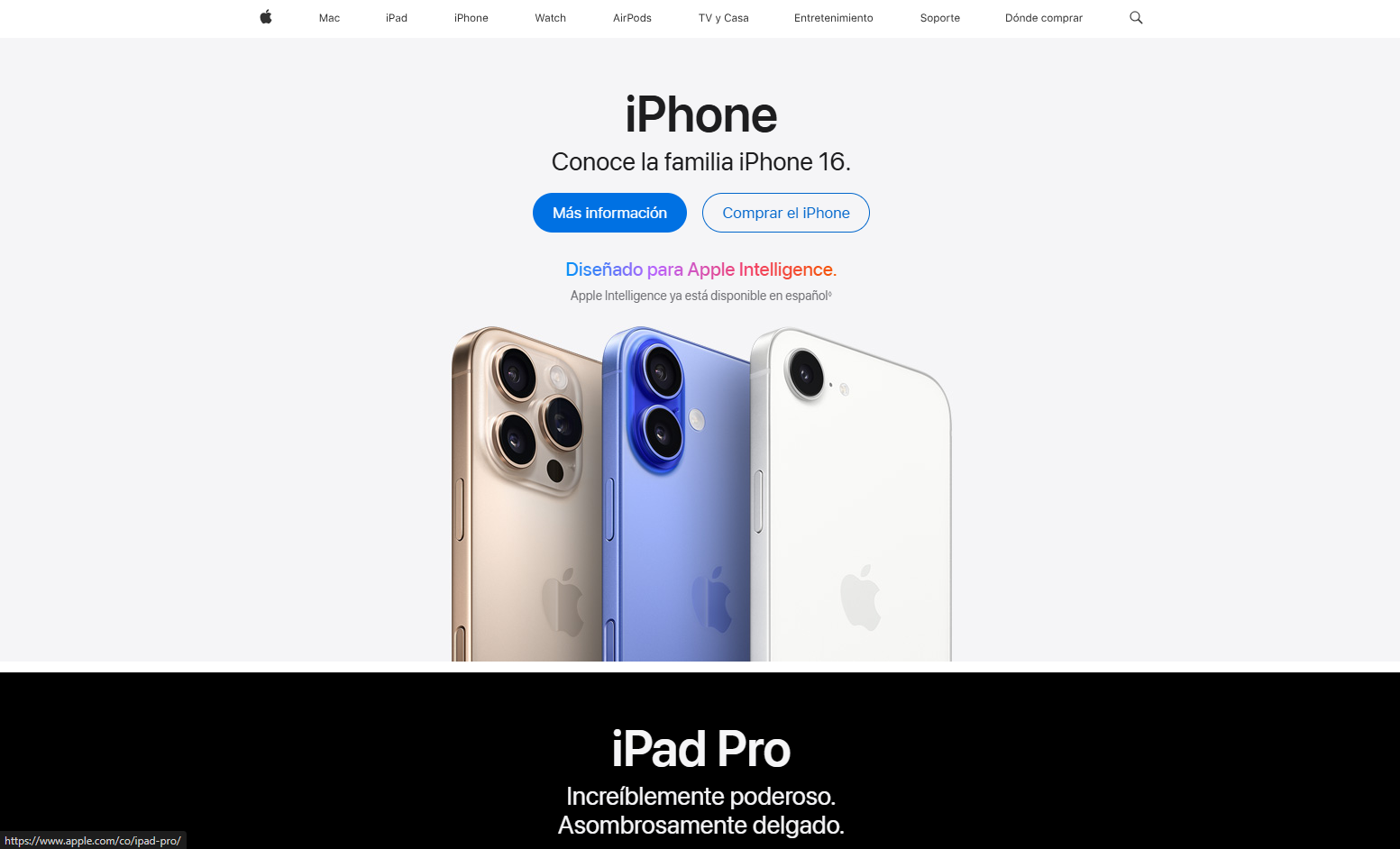
Utilizamos métodos modulares “bem”, adoptar esta metodología como también smacss para organizar el código y facilitar su mantenimiento, útil

izar un sistema de nomenclaturas consistentes para los estilos, refactorizar el código para eliminar la deuda técnica acumulada, utilizar herramientas Stylelint para detectar problemas en el código y mejorar su calidad y tener un mejor rendimiento a la hora de mejorar el código.

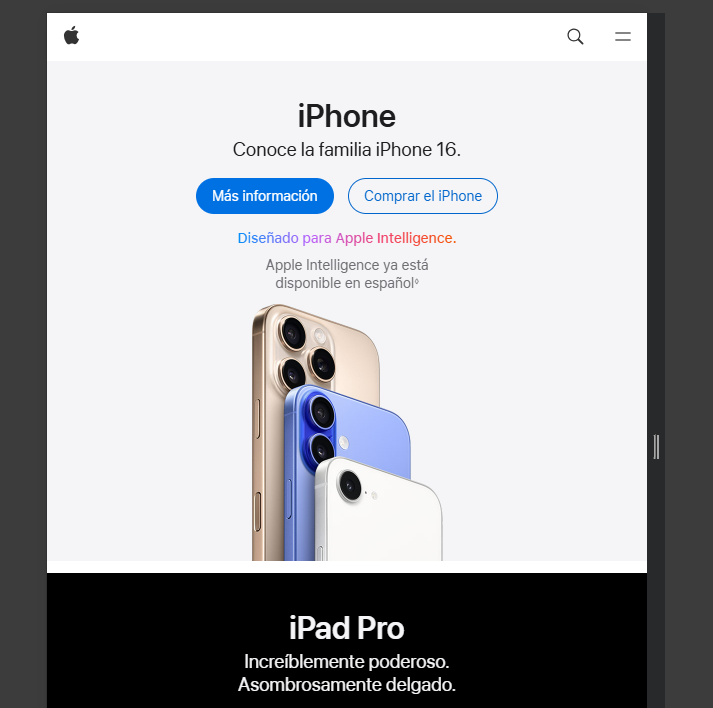
* + Describe qué entiendes por "reutilización de componentes de estilo".

Definir estilos en clases y aplicarlos a múltiples elementos, esencial para mantener un código limpio consistente y eficiente en el desarrollo web

* + Observa un sitio web que te guste y describe cómo crees que están estructurados sus componentes CSS.



Esta página que elegí se refleja un menú de navegación <navbar> con espaciados entre cada opción al parecer margin en cada elemento y un buscador desplegable. Tiene una sección como cuerpo de la página un background imge, dos buttom e información “texto” por ultimo se muestra un footer con texto y un background black.



Esta pagina también se muestra responsiva , adaptativa maneja breackpoints y se notan cambios como un menú hamburguesa para el menú de navegación un en el secttion un background imge mas pequeña

**Materiales de formación:**

**Estrategias didácticas activas:**

* + Lluvia de ideas
  + Diario reflexivo

**Duración de la actividad 1 hora.**

**B. Actividades de contextualización e identificación de conocimientos necesarios para el aprendizaje.**

Identificarás las metodologías CSS más utilizadas y analizarás sus propósitos y diferencias. Te enfocarás especialmente en comprender el enfoque de BEM.

**Actividades de aprendizaje:**

* + Investiga y define las metodologías: OOCSS, SMACSS, BEM, ITCSS.

OOCSS (css orientado a objetos) se centra en la creación de componentes reutilizable y en la separación de la estructura y la apariencia, su objetivo principal es reducir la duplicación de código y fomentar la creación de estilos modulares.

SMACSS (arquitectura modular para css) proporciona una guía para organizar de manera eficiente el código una metodología con cinco pasos esta metodologia ayuda a tener coherencia en la estructura del código en estilos y el sitio web.

BEM (block element modifier) es una metodologia de nomenclatura para clases en html y css ayuda a crear código mas estructurado, mantenible y reutilizable, especialmente en proyectos grandes.

ITCSS (arquitectura css del triangulo invertido) es una metodologia para organizar hojas de estilo css especialmente en proyectos grandes, se basa en una estructura de capas con diferentes niveles de especificidad asemeja a un triángulo invertido.

* + Crea una tabla comparativa de ventajas y desventajas de cada metodología.

**OCSS**

|  |  |
| --- | --- |
| Ventajas | Desventajas |
| \* Reutilización de código  \* Modularidad, permite dividir el css en módulos independientes.  \* escalabilidad, facilita la ampliación y la modificación de estilos a medida que el proyecto crece  \* fácil mantenimiento y rendimiento mejorado. | \* Complejidad en el hmtl  \* excesos de clases lo que dificulta la gestión y el mantenimiento del código.  \* no apto para proyectos pequeños ya que la sobrecarga de clases y la complejidad del HTML pueden ser menos problemáticas en proyectos más pequeño. |

**SMACSS**

|  |  |
| --- | --- |
| Ventajas | Desventajas |
| * Organización y estructura:   SMACSS divide el código CSS en categorías (base, layout, módulos, estado y tema) facilitando la comprensión y el mantenimiento del código.   * Modularidad:   Al separar el código en módulos, se promueve la reutilización de estilos y se facilita la creación de componentes independientes.   * Escalabilidad:   La estructura de SMACSS permite que el código CSS crezca de manera organizada y predecible, evitando la complejidad y el caos que pueden surgir en proyectos grandes.   * Mantenimiento:   La separación de estilos en categorías facilita la identificación y modificación de estilos específicos, lo que agiliza las tareas de mantenimiento y actualización.   * Colaboración:   Al definir categorías claras y convenciones de nomenclatura, SMACSS ayuda a que varios desarrolladores puedan trabajar en el mismo proyecto de manera más eficiente y consistente | * Curva de aprendizaje:   Aunque SMACSS es relativamente fácil de entender, puede tomar tiempo para que los desarrolladores se acostumbren a la estructura y a las convenciones de nomenclatura.   * Disciplina en la aplicación:   Para aprovechar al máximo SMACSS, es importante ser consistente con la estructura y las convenciones de nomenclatura, lo que requiere disciplina por parte del equipo de desarrollo.   * Posible exceso de especificidad:   En algunos casos, la estructura de SMACSS podría llevar a una especificidad excesiva en los selectores CSS, lo que podría dificultar la anulación de estilos en situaciones específicas.   * No es una solución mágica:   Aunque SMACSS es una excelente metodología, no resuelve todos los problemas de CSS. Se necesita combinar con otras buenas prácticas y herramientas para crear código CSS robusto y eficiente. |

**BEM**

|  |  |
| --- | --- |
| Ventajas | Desventajas |
| * Organización y estructura:   BEM promueve una estructura clara y predecible para el código CSS, facilitando la comprensión y el mantenimiento, especialmente en proyectos grandes y colaborativos.   * Reutilización de componentes:   Al definir bloques, elementos y modificadores, BEM permite crear componentes reutilizables que pueden ser implementados en diferentes partes del sitio web sin necesidad de reescribir estilos.   * Evita conflictos de estilos:   Al usar nombres de clase únicos para cada componente y sus partes, BEM ayuda a prevenir conflictos de estilos que pueden surgir en proyectos grandes | * Longitud de nombres de clase:   La nomenclatura de BEM puede resultar en nombres de clase largos y complejos, especialmente para elementos y modificadores anidados, lo que puede dificultar la lectura y escritura del código en algunos casos.   * Curva de aprendizaje inicial:   BEM requiere un cierto aprendizaje inicial para comprender su estructura y nomenclatura, lo que puede ralentizar el desarrollo al principio.   * Posible complejidad en proyectos pequeños:   En proyectos pequeños, la complejidad de la metodología BEM podría no ser necesaria, y podría agregar una sobrecarga innecesaria al código |

**ITCSS**

|  |  |
| --- | --- |
| Ventajas | Desventajas |
| * Organización:   ITCSS estructura el CSS en capas, desde lo más genérico a lo más específico, lo que facilita la comprensión y el mantenimiento del código.   * Escalabilidad:   Permite que el código CSS crezca de manera organizada y predecible, evitando conflictos y redundancias a medida que el proyecto se expande.   * Mantenibilidad:   La estructura en capas y la separación de responsabilidades hacen que sea más fácil encontrar y modificar estilos específicos sin afectar otras partes del código.   * Reutilización:   Al dividir el código en objetos y componentes con sus propios estilos, se facilita la reutilización de estos elementos en diferentes partes del proyecto o en otros proyectos.   * Colaboración:   La estructura organizada facilita el trabajo en equipo, ya que varios desarrolladores pueden trabajar en diferentes partes del CSS sin interferir entre si | * Curva de aprendizaje:   Requiere familiarizarse con la metodología y sus capas, lo que puede tomar tiempo para los desarrolladores que no están acostumbrados a trabajar con arquitecturas CSS complejas.   * Configuración inicial:   La configuración inicial de ITCSS puede llevar más tiempo que simplemente escribir CSS sin una estructura definida, especialmente en proyectos pequeños.   * Sobrecarga en proyectos pequeños:   En proyectos pequeños o de corta duración, la estructura de ITCSS puede ser excesiva y agregar complejidad innecesaria.   * Puede ser propietario:   La documentación de código abierto de ITCSS no es tan amplia como otras arquitecturas, lo que puede dificultar su adopción por parte de algunos desarrolladores |

* + Investiga el origen de la metodología BEM.

La historia de BEM comenzó en 2005. En aquel entonces, desde la perspectiva del frontender, un proyecto típico de Yandex era un conjunto de páginas HTML estáticas que servían como base para crear plantillas XSL.

Las páginas HTML se guardaron en una carpeta separada con la siguiente estructura:

acerca de.html

índice.html

…

proyecto.css

proyecto.js

i/

yandex.png

Había un archivo HTML independiente para cada página. El diseño utilizado IDsy classes.

Todos los scripts del proyecto se guardaron en un solo project.jsarchivo. JavaScript se utilizó como herramienta auxiliar para los elementos interactivos, por lo que project.jsera pequeño.

Las imágenes se guardaron en una carpeta aparte. Debido a la necesidad de compatibilidad con IE 5 y la falta de CSS3 en los navegadores, se usaron imágenes para todo tipo de elementos de diseño, incluso para crear esquinas redondeadas.

Los estilos y scripts se escribieron en los archivos: project.cssy project.js. Para separar los estilos de las diferentes partes de una página, usamos comentarios para indicar el principio y el final:

En 2006, comenzamos a trabajar en nuestros primeros grandes proyectos: Yandex.Music y Ya.Ru. Estos proyectos, con decenas de páginas, revelaron las principales desventajas del enfoque actual de desarrollo:

Cualquier cambio en el código de una página afectaba el código de otras páginas.

Fue difícil elegir los nombres de las clases.

Un CSS típico en aquel entonces contenía una larga cascada:

* + Redacta un resume de los principios claves de BEM.

Principios clave de BEM:

Bloque (Block):

Representa un componente independiente y reutilizable, como un menú, un formulario o un botón.

Elemento (Element):

Forma parte de un bloque y no tiene sentido sin él, por ejemplo, un ítem dentro de una lista.

Modificador (Modifier):

Define la apariencia o comportamiento de un bloque o elemento, como un color diferente o un estado activo

* + Analiza 10 ejemplos de nombres de clases con la metodología BEM y explica su composición.

La metodologia bem organiza el código en tres partes , bloques, elementos y modificadores, un bloque es un componente independiente, un elemento es una parte de un bloque y el modificador cambia la apariencia o comportamiento de un bloque o elemento

button: Un bloque, representando un botón genérico.

button\_\_text: Un elemento, el texto dentro del botón.

button--primary: Un modificador, un botón con estilo primario.

button--disabled: Un modificador, un botón deshabilitado.

header: Un bloque, representando la cabecera de la página.

header\_\_logo: Un elemento, el logotipo dentro de la cabecera.

header\_\_nav: Un elemento, la barra de navegación dentro de la cabecera.

header\_\_nav--fixed: Un modificador, una barra de navegación fija.

form: Un bloque, representando un formulario.

form\_\_input--error: Un elemento con modificador, un campo de entrada con error.

**Materiales de formación:**

* + <https://en.bem.info/methodology/>
  + [https://www.youtube.com/watch?v=FsoOdePCdiQ&ab\_channel=RubenTerr%C3%A](https://www.youtube.com/watch?v=FsoOdePCdiQ&ab_channel=RubenTerr%C3%A9)

[9](https://www.youtube.com/watch?v=FsoOdePCdiQ&ab_channel=RubenTerr%C3%A9)

**Estrategias didácticas activas:**

* + Aprendizaje basado en investigación.
  + Aprendizaje visual

**Duración de la actividad 1 hora.**

**C. Actividades de apropiación del conocimiento.**

Aplicarás la metodología BEM para estructurar componentes de interfaz. Crearás clases CSS con nombres BEM siguiendo una estructura clara.

**Ejercicios prácticos:** en cada uno, crea el marcado **HTML** y las clases **CSS** correspondientes utilizando **BEM**.

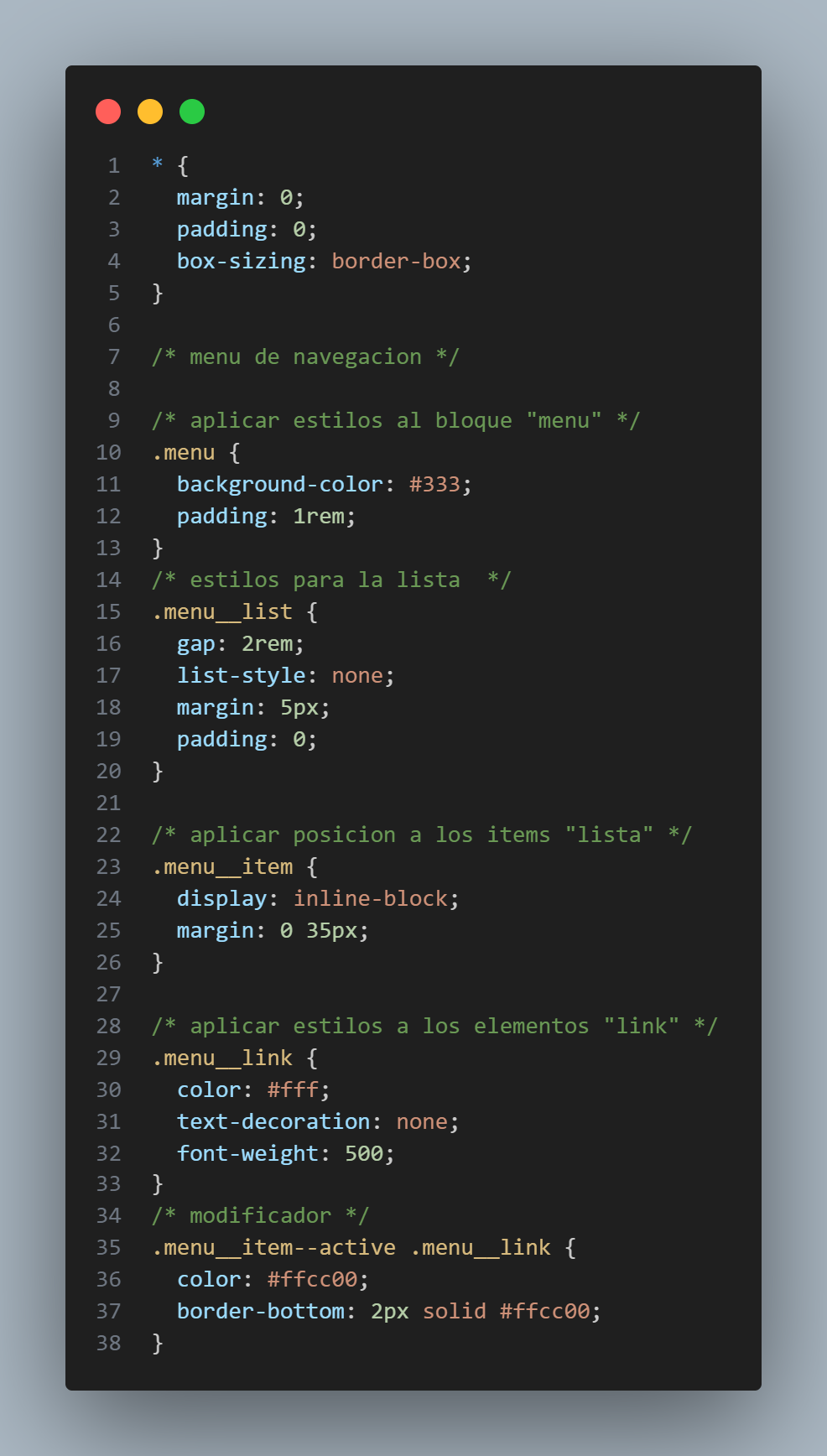
* 1. Componente de tarjeta de producto.





* 1. Componente de menú de navegación.





* 1. Componente de formulario de contacto.





* 1. Componente de lista de tareas.

1. <!-- bloque -->
2. <div class="lista">
3. <!-- elementos -->
4. <ul class="lista\_\_tareas">
5. <h3 class="lista\_\_titulo">Lista De Tareas Supr</h3>
6. <li class="lista\_\_items">
7. <input type="checkbox" class="list\_\_checkbox">
8. <span class="lista\_\_opcion">C:\Windows\System32</span>
9. <!-- modificador -->
10. <button class="lista\_\_delete-button">Eliminar</button>
11. </li>
12. <!-- modificador -->
13. <li class="lista\_\_items  lista\_\_items--completo">
14. <!-- elementos -->
15. <input type="checkbox" class="lista\_\_checkbox"checked>
16. <span class="lista\_\_opcion">Database</span>
17. <!-- modificador -->
18. <button class="lista\_\_delete-button">Eliminar</button>
19. </li>
20. </ul>
21. <!-- elementos -->
22. <h3 class="lista\_\_titulo">Informacion</h3>
23. <p class="lista\_\_texto">Lorem ipsum dolor, sit amet consectetur adipisicing elit. Iste laudantium quae sint corrupti .</p>
24. </div>

/\* lista de tareas \*/

/\* estilos al bloque "lista" \*/

.lista {

  width: 300px;

  height: 380px;

  margin: 3.7% auto;

  padding: 10px;

  box-shadow: 0 15px 15px #3333335e;

  border: 1px solid #ccc;

  border-radius: 4px;

  display: inline-block;

  margin-left: 20px;

  vertical-align: top;

}

/\* estilo al titulo  \*/

.lista\_\_titulo{

  text-align: center;

  margin: 5%;

}

/\* estilo a la lista de tareas \*/

.lista\_\_tareas {

  list-style: none;

  padding: 0;

}

.lista\_\_items {

  margin: 10% auto;

  padding: 0 10px;

}

/\* estilos botones eliminar \*/

.lista\_\_delete-button {

  background-color: #007bff;

  padding: 5px 10px;

  border-radius: 5px;

  border: none;

  cursor: pointer;

  color: white;

  display: inline-block;

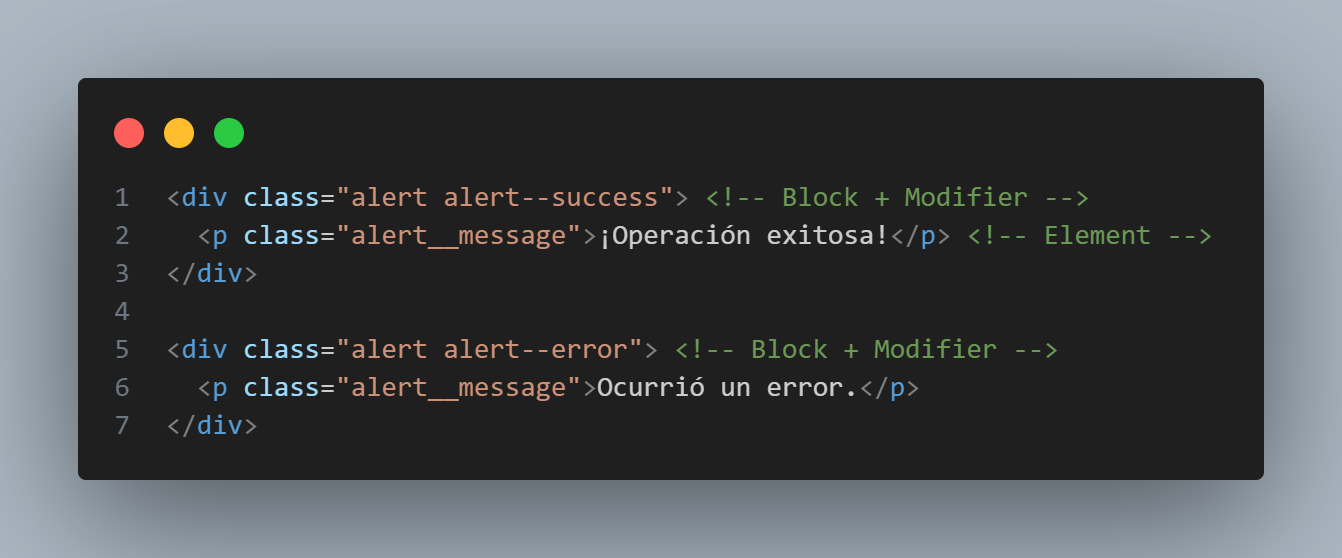
}

* 1. Componente de galería de imágenes.





* 1. Componente de alerta o notificación.



/\* -------------------------- \*/

/\* alerta notificacion \*/

/\* Block \*/

.alert {

  border: 1px solid;

  padding: 10px;

  margin-bottom: 10px;

}

/\* Element \*/

.alert\_\_message {

  margin: 0;

}

/\* Modifier \*/

.alert--success {

  border-color: green;

  background-color: #e0fbe0;

}

.alert--error {

  border-color: red;

  background-color: #fde0e0;

}

* 1. Componente de tabla de datos.

<table class="data-table"> <!-- Block -->

  <thead class="data-table\_\_head"> <!-- Element -->

    <tr class="data-table\_\_row">

      <th class="data-table\_\_header">Nombre</th>

      <th class="data-table\_\_header">Edad</th>

    </tr>

  </thead>

  <tbody class="data-table\_\_body"> <!-- Element -->

    <tr class="data-table\_\_row data-table\_\_row--highlighted"> <!-- Element + Modifier -->

      <td class="data-table\_\_cell">José</td>

      <td class="data-table\_\_cell">30</td>

    </tr>

    <tr class="data-table\_\_row">

      <td class="data-table\_\_cell">Ana</td>

      <td class="data-table\_\_cell">25</td>

    </tr>

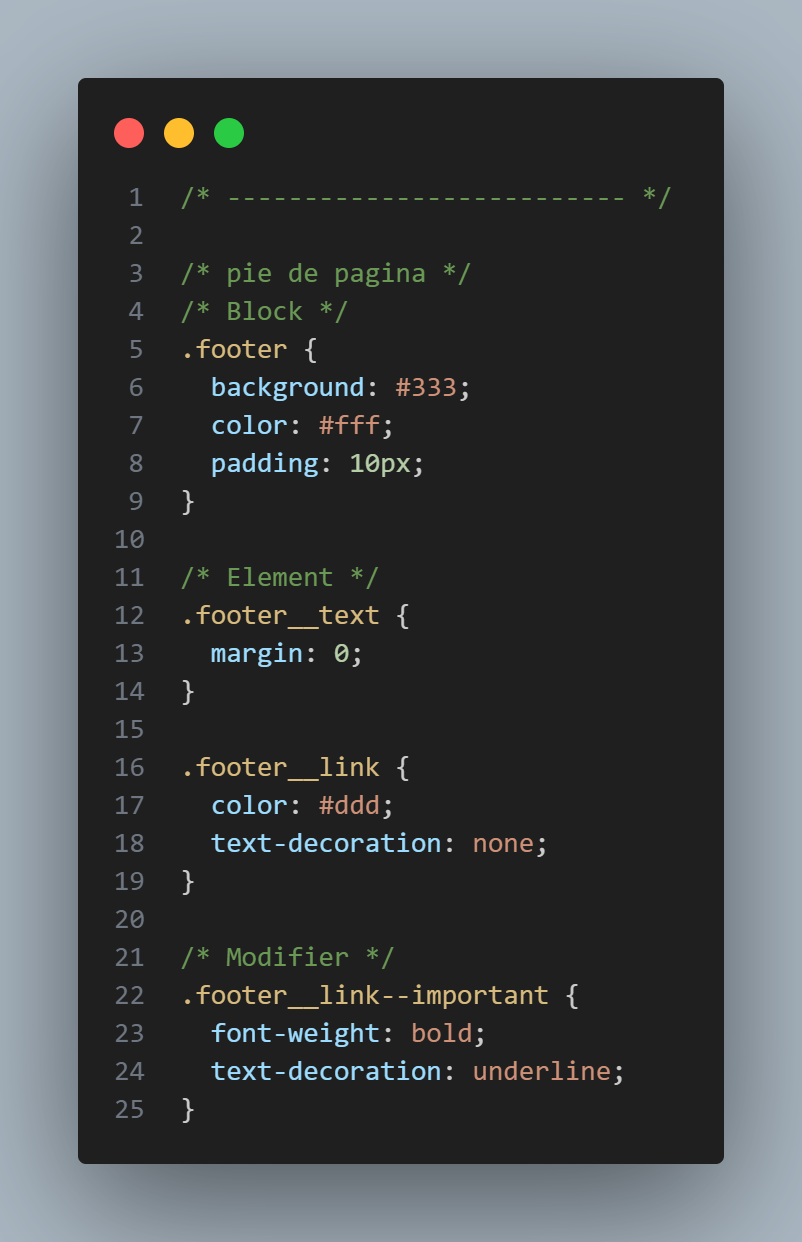
  </tbody>

</table>



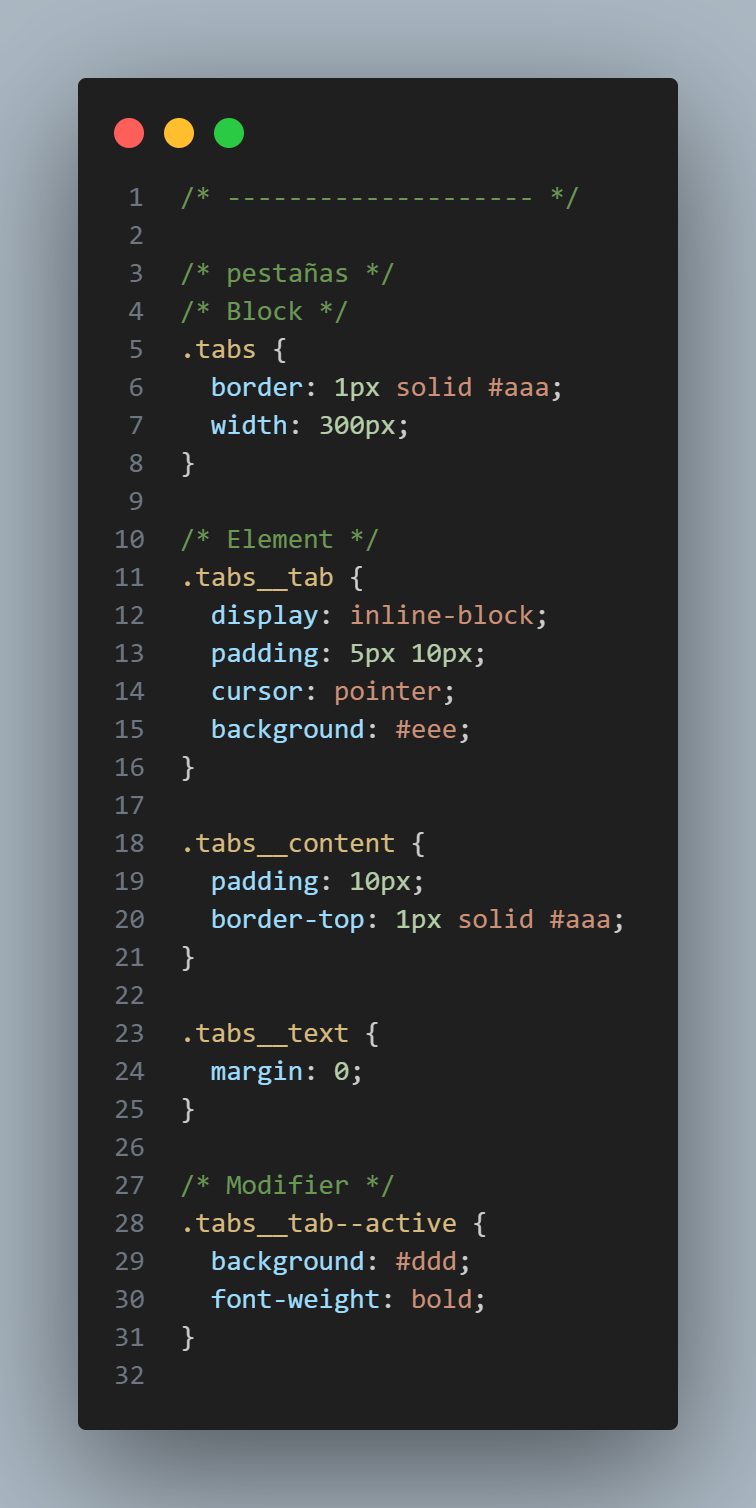
* 1. Componente de pie de página.





* 1. Componente de pestañas (tabs).





* 1. Componente de selector de categorías.



/\* --------------- \*/

/\* selector de categorias \*/

/\* Block \*/

.category-selector {

  padding: 5px;

}

/\* Element \*/

.category-selector\_\_button {

  margin: 2px;

  padding: 5px 10px;

  background: #f0f0f0;

  border: 1px solid #aaa;

  cursor: pointer;

}

/\* Modifier \*/

.category-selector\_\_button--selected {

  background: #c0e0ff;

  border-color: #0055aa;

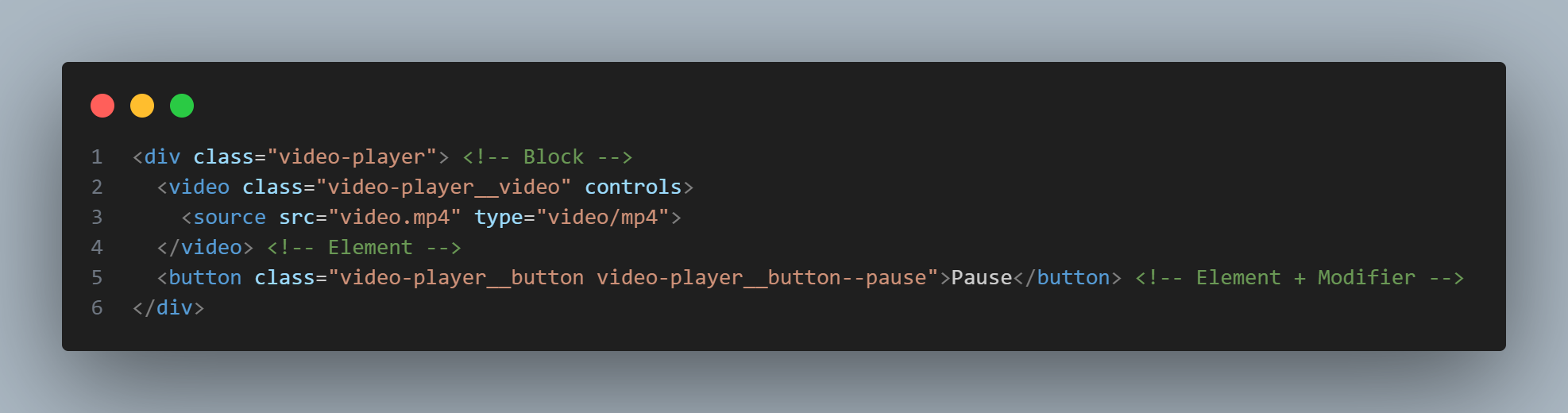
}

* 1. Componente de sistema de calificaciones.

1. <div class="rating"> <!-- Block -->
2. <span class="rating\_\_star rating\_\_star--filled">★</span> <!-- Element + Modifier -->
3. <span class="rating\_\_star rating\_\_star--filled">★</span>
4. <span class="rating\_\_star rating\_\_star--filled">★</span>
5. <span class="rating\_\_star">★</span>
6. <span class="rating\_\_star">★</span>
7. </div>



* 1. Componente de reproductor de video.



/\* ------------------------ \*/

/\* reproductor de video \*/

/\* Block \*/

.video-player {

  width: 320px;

  border: 1px solid #999;

  padding: 5px;

}

/\* Element \*/

.video-player\_\_video {

  width: 100%;

}

.video-player\_\_button {

  margin-top: 5px;

  padding: 5px 10px;

}

/\* Modifier \*/

.video-player\_\_button--pause {

  background: #f88;

}

* 1. Componente de barra lateral sidebar.



/\* -------------------------- \*/

/\* sidebar \*/

/\* Block \*/

.sidebar {

  width: 200px;

  background: #f5f5f5;

  padding: 10px;

}

/\* Element \*/

.sidebar\_\_title {

  margin: 0 0 10px;

}

.sidebar\_\_list {

  list-style: none;

  padding: 0;

}

.sidebar\_\_item {

  padding: 5px;

  cursor: pointer;

}

/\* Modifier \*/

.sidebar\_\_item--active {

  background: #ddd;

  font-weight: bold;

}

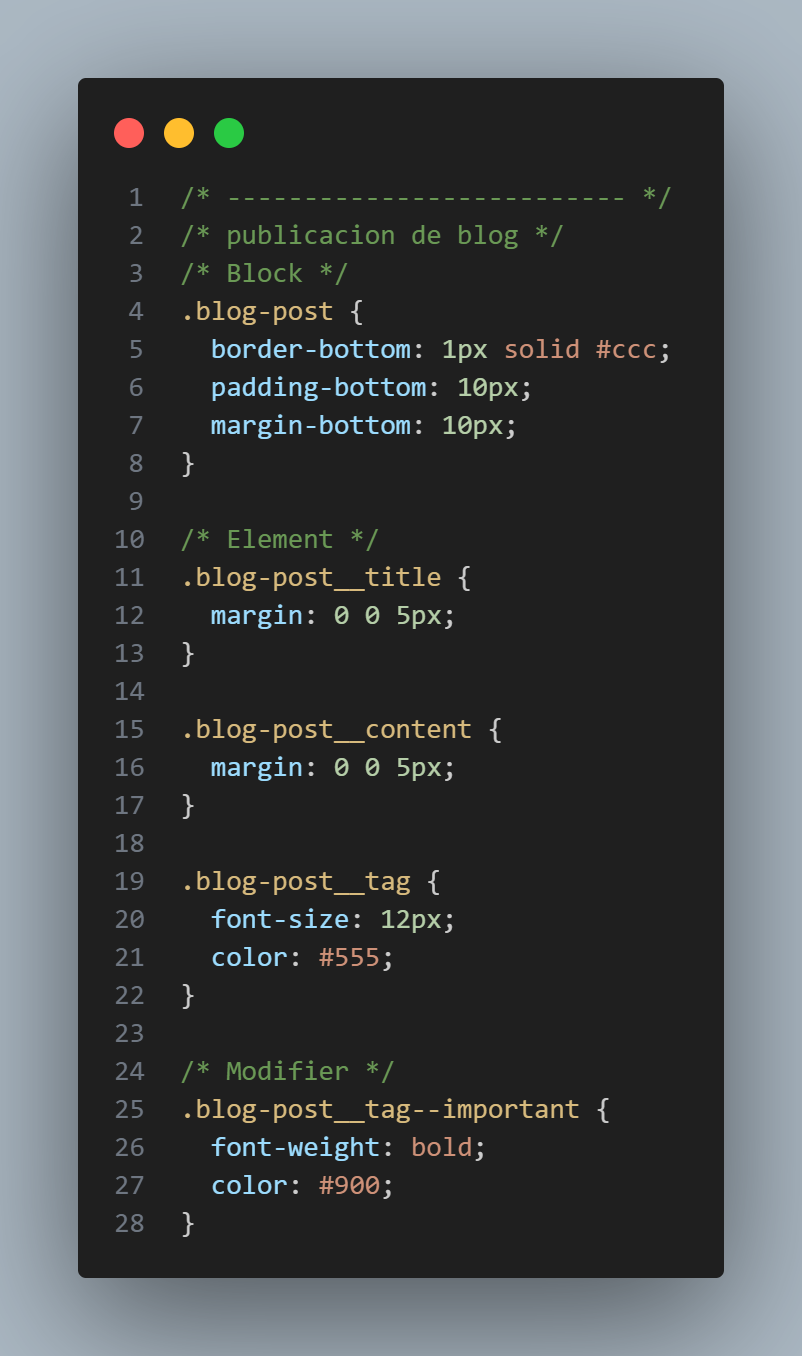
* 1. Componente de resumen de carrito.





* 1. Componente de publicación en blog.





**Materiales de formación:**

* + Editor de código (VS Code)
  + Navegador web con herramientas de inspección
  + Repositorio de Git

**Estrategias didácticas activas:**

* + Aprendizaje basado en proyectos
  + Resolución de problemas prácticos
  + Trabajo colaborativo

**Evidencias de aprendizaje:**

* + Archivos **HTML** y **CSS** con estructura **BEM**.
  + Documentación del diseño de clases.
  + Código fuente con comentarios
  + Proyecto publicado en GitHub

**Duración de la actividad 6 hora.**

**D. Actividades de transferencia del conocimiento.**

Pondrás en práctica lo aprendido en un proyecto web real o simulado. Aplicarás BEM en una interfaz compuesta por varios componentes.

**Actividades de aprendizaje:**

* 1. Reestructura una página que hayas creado previamente usando BEM.
  2. Crea una página de inicio de e-commerce con componentes estructurados
  3. Crea una guía visual (presentación) con los bloques, elementos y modificadores usados para diseñar el e-commerce donde se pueda identificar el porqué de cada componente y realiza una exposición al grupo. “**Este punto servirá para subir la nota de las personas que redactaron su nombre en color rojo en la presentación de la guían anterior**”

https://github.com/jogard24/ActividadBem.git

**Materiales de formación:**

**Estrategias didácticas activas:**

* + Aprendizaje basado en proyectos
  + Evaluación entre pares **Evidencias de aprendizaje:**
  + Repositorio con el código del proyecto
  + Presentación oral o escrita de la experiencia
  + Cuadro comparativo antes/después de aplicar BEM

**Duración de la actividad 6 horas.**

1. **ACTIVIDADES DE EVALUACIÓN**

Tome como referencia la técnica e instrumentos de evaluación citados en la guía de Desarrollo Curricular

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Fase del proyecto formativo** | **Actividad del proyecto formativo** | **Actividad de**  **Aprendizaj e** | **Evidencias de**  **Aprendizaj e** | **Criterios de**  **Evaluación** | **Técnicas e**  **Instrumento s de**  **Evaluación** |
| EJECUCIÓ  N | DESARROLLAR  LA STRUCTURA  DE DATOS,  CONSTRUCCIÓ  N,  CODIFICACIÓN  E INTERFAZ DE  USUARIO,  PRUEBAS Y  VALIDACIÓN DE LOS MÓDULOS  DEL SISTEMA  DE  INFORMACIÓ | GENERAR  PLANTILLAS Y ESTILOS.  CREAR  INTERFACES  GRÁFICAS DE  USUARIO EN APLICACIONE S DE  ESCRITORIO, WEB Y  MÓVILES. | Conocimient  o y  Producto | 1. GENERA   PLANTILLAS Y  ESTILOS DE  ACUERDO  CON EL  DISEÑO  ESTABLECID O   1. CREA   INTERFACES DE USUARIO  APLICANDO  BUENAS  PRÁCTICAS  DE  USABILIDAD Y  ACCESIBILIDA  D | Listas de chequeo |

1. **GLOSARIO DE TÉRMINOS**

* + **BEM (Bloque, Elemento, Modificador):** Metodología para nombrar clases CSS de forma estructurada y legible.
  + **Bloque:** Componente independiente que representa una entidad significativa (ej. menu).
  + **Elemento:** Parte interna de un bloque que depende del mismo (ej. menu\_\_item).
  + **Modificador:** Variación de apariencia o comportamiento de un bloque o elemento (ej. menu\_\_item--active).
  + **Metodología CSS:** Estrategia para organizar el código CSS de forma escalable y mantenible.
  + **Escalabilidad:** Capacidad del código para crecer sin perder claridad ni funcionalidad.
  + **Reutilización:** Uso del mismo código en distintos contextos o componentes.
  + **Deuda técnica:** Consecuencias negativas acumuladas por soluciones rápidas o desorganizadas en el código.
  + **Autoaprendizaje:** Capacidad de aprender de forma autónoma y continua.
  + **Aprendizaje colaborativo:** Estrategia en la que los aprendices trabajan juntos para alcanzar metas comunes.

1. **REFERENTES BILBIOGRÁFICOS**

* GetBEM. (s.f.). *Introduction to BEM methodology*. Recuperado de[: https://getbem.com](https://getbem.com/)
* Smashing Magazine. (2012). *CSS Methodologies: BEM, OOCSS, SMACSS, Atomic CSS*. https://www.smashingmagazine.com
* CSS-Tricks. (s.f.). *BEM 101*. https://css-tricks.com/bem-101/
* MDN Web Docs. (s.f.). *CSS: Cascading Style Sheets*. https://developer.mozilla.org/enUS/docs/Web/CSS

1. **CONTROL DEL DOCUMENTO**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **Nombre** | **Cargo** | **Dependencia** | **Fecha** |
| **Autor**  **(es)** | **John Freddy Becerra**  **Castellanos** | **Instructor**  **Contratista** | **CIMI** | **06/02/2025** |

1. **CONTROL DE CAMBIOS** (diligenciar únicamente si realiza ajustes a la guía)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | **Nombre** | **Cargo** | **Dependencia** | **Fecha** | **Razón del**  **Cambio** |
| **Autor**  **(es)** |  |  |  |  |  |