SERVICIO NACIONAL DE ARENDIZAJE SENA

ANALISIS Y DESARROLLO DE SOFTWARE

ESTANDARES DE CODIFICACION

ELKIN ALBERTO PEREZ

MIGUEL BECERRA

YORMAN CABALLERO

2023

CUCUTA, NORTE DE SANTANDER

INDICE

ESTANDARES DE CODIFICACION.

[INTRODUCCION 3](#_Toc164970376)

[OBJETIVOS 4](#_Toc164970377)

[TECNOLOGÍAS Y LENGUAJES UTILIZADOS 5](#_Toc164970378)

[ESTANDARES DE CODIFICACION. 5](#_Toc164970379)

[1. JAVA 5](#_Toc164970380)

[2. HTML 6](#_Toc164970381)

[3. CSS 6](#_Toc164970382)

[4. JAVASCRIP 6](#_Toc164970383)

[HERRAMIENTAS DE ANÁLISIS DE CÓDIGO. 7](#_Toc164970384)

[CONCLUSION 8](#_Toc164970385)

# INTRODUCCION

En el desarrollo de software, especialmente en proyectos colaborativos, es fundamental establecer y seguir estándares de codificación consistentes. Estos estándares mejoran la legibilidad, el mantenimiento y la calidad del código fuente, facilitando la colaboración entre los miembros del equipo y asegurando un estilo coherente en todo el proyecto.

# OBJETIVOS

* Comprender la estructura correcta y las tendencias de código más actuales.
* Entender los lenguajes usados para la creación de nuestro proyecto.
* Comprender los estándares y reglas de codificación de código.
* Aprender de las herramientas de edición y prueba de código.

# TECNOLOGÍAS Y LENGUAJES UTILIZADOS

Para el desarrollo de la aplicación web de un establecimiento de bebidas alcohólicas, se utilizarán las siguientes tecnologías y lenguajes:

- Java: Lenguaje de programación principal para el desarrollo de la lógica de la aplicación web.

- HTML: Lenguaje de marcado para la estructura y contenido de las páginas web.

- CSS: Lenguaje de hojas de estilo para el diseño y presentación visual de las páginas web.

- JavaScript: Lenguaje de programación para agregar interactividad y funcionalidades del lado del cliente en las páginas web.

## ESTANDARES DE CODIFICACION.

A continuación, se detallan los estándares de codificación recomendados para cada tecnología y lenguaje:

### 1. JAVA

- Convención de nomenclatura: Clases en UpperCamelCase (ej. NombreClase), métodos y variables en lowerCamelCase (ej. nombreMetodo), constantes en UPPERCASE\_WITH\_UNDERSCORES (ej. CONSTANTE\_NOMBRE).

- Indentación: Utilizar 4 espacios en lugar de tabuladores.

- Llaves: Colocar la llave de apertura en la misma línea que la declaración y la llave de cierre en una nueva línea.

- Comentarios: Utilizar comentarios descriptivos y claros para explicar el propósito y funcionamiento del código.

- Organización de paquetes: Seguir una estructura de paquetes lógica y coherente, agrupando clases relacionadas en el mismo paquete.

- Nombres descriptivos: Utilizar nombres claros y descriptivos para clases, métodos y variables, evitando abreviaturas o nombres ambiguos.

### 2. HTML

- Estructura semántica: Utilizar etiquetas HTML adecuadas para representar la estructura y el contenido de la página web (ej. <header>, <nav>, <main>, <footer>).

- Indentación y anidamiento: Indentar correctamente los elementos anidados para mejorar la legibilidad del código.

- Minúsculas: Utilizar minúsculas para todos los nombres de etiquetas y atributos.

- Accesibilidad: Implementar prácticas de accesibilidad, como textos alternativos para imágenes y etiquetas de formulario adecuadas.

### 3. CSS

- Nomenclatura: Utilizar una convención de nomenclatura consistente, como BEM (Bloque-Elemento-Modificador) o Kebab-case (ej. bloque\_\_elemento--modificador).

- Organización: Agrupar reglas CSS relacionadas y separar secciones con comentarios descriptivos.

- Especificidad: Evitar selectores excesivamente específicos y utilizar una especificidad adecuada para cada regla.

- Uso de preprocesadores (opcional): Considerar el uso de preprocesadores CSS como Sass o Less para mejorar la modularidad y mantenibilidad del código CSS.

### 4. JAVASCRIP

- Convención de nomenclatura: Utilizar lowerCamelCase para variables y funciones (ej. nombreVariable, nombreFuncion).

- Indentación: Utilizar 2 espacios en lugar de tabuladores.

- Comillas: Utilizar comillas simples ('') para delimitar cadenas de texto, a menos que se requieran comillas dobles para escapar caracteres.

- Comentarios: Utilizar comentarios descriptivos y claros para explicar el propósito y funcionamiento del código.

- Modularidad: Dividir el código en módulos y archivos separados para mejorar la organización y reutilización del código.

- Estándar de codificación: Considerar adoptar un estándar de codificación reconocido, como el de Google o el de Airbnb, para mantener la consistencia en todo el proyecto.

## HERRAMIENTAS DE ANÁLISIS DE CÓDIGO.

Se recomienda utilizar herramientas de análisis de código estático, como SonarQube, ESLint o StyleLint, para identificar y corregir problemas de calidad, errores y violaciones de estándares de codificación en todas las tecnologías y lenguajes utilizados.

# CONCLUSION

Establecer y seguir estándares de codificación consistentes es crucial para el desarrollo exitoso de la aplicación web. Estos estándares facilitan la colaboración en equipo, mejoran la legibilidad y mantenibilidad del código, y aseguran una calidad más alta en el producto final. Es importante que todos los miembros del equipo de desarrollo se adhieran a estos estándares desde el inicio del proyecto y se realicen revisiones periódicas para garantizar su cumplimiento.