Políticas de Nomenclatura para Desarrollo

Bases de Datos:

1. **Nombres de Bases de Datos:**
    - Utilizar nombres descriptivos y concisos que reflejen el propósito y contenido de la base de datos.
    - Preferir el formato snake_case o camelCase.
2. **Nombres de Tablas:**
    - Nombrar las tablas de forma singular y utilizar nombres que describan claramente la entidad que representan.
    - Evitar prefijos innecesarios.
3. **Nombres de Columnas:**
    - Nombrar las columnas de manera descriptiva, utilizando camelCase o snake_case según la preferencia del equipo.
    - Evitar abreviaciones ambiguas y preferir claridad sobre brevedad.

Variables:

1. **Nombres de Variables:**
    - Utilizar nombres descriptivos y representativos del propósito de la variable.
    - Emplear camelCase para variables en JavaScript y otros lenguajes de estilo similar.
2. **Variables Constantes:**
    - Utilizar MAYÚSCULAS_SNAKE_CASE para representar variables constantes.

Funciones:

1. **Nombres de Funciones:**
    - Nombrar las funciones de manera descriptiva, indicando su propósito y acción.
    - Utilizar camelCase para funciones en JavaScript y otros lenguajes de estilo similar.
2. **Verbos en Funciones:**
    - Utilizar verbos en el nombre de las funciones para indicar acciones (e.g., `calcularPrecio`, `guardarUsuario`).

Clases:

1. **Nombres de Clases:**
    - Utilizar nombres en formato PascalCase para clases.
    - Nombrar las clases de manera que reflejen su responsabilidad y función en el sistema.

Git:

1. **Nombres de Ramas:**
    - Utilizar nombres descriptivos para las ramas, prefiriendo nombres que reflejen la tarea o función que se está desarrollando.
    - Evitar nombres genéricos como "feature" o "fix".
2. **Mensajes de Commit:**
    - Escribir mensajes de commit claros y concisos, indicando de manera comprensible los cambios realizados.
    - Utilizar un formato consistente, como "Agregar", "Modificar", "Corregir", etc.
3. **Tags:**
    - Utilizar tags para versiones de software, siguiendo un formato semántico como "v1.0.0".

Documentación:

1. **Documentación de Código:**
    - Incluir comentarios descriptivos en el código para explicar decisiones, algoritmos complejos o partes críticas.
    - Mantener la documentación actualizada con el código.
2. **README:**
    - Incluir un README en cada proyecto que proporcione información clara sobre cómo configurar, ejecutar y contribuir al proyecto.