**Políticas de nomenclaturas**

**Estándares de Nombres y**

**Convenciones en Desarrollo de Software**

Bemaster

Versión 1.0.0

Año 2024

**Índice**

[Descripción 3](#_Toc156811856)

[Objetivo 3](#_Toc156811857)

[Entornos de desarrollo 3](#_Toc156811858)

[Entorno de pruebas 4](#_Toc156811859)

[Entorno de producción 4](#_Toc156811860)

[Tabla de las políticas 5](#_Toc156811861)

[Políticas nomenclaturas carpeta del proyecto 5](#_Toc156811862)

[Políticas nomenclaturas Git 5](#_Toc156811863)

[Políticas nomenclaturas base de datos 6](#_Toc156811864)

[Políticas nomenclaturas variables 6](#_Toc156811865)

[Políticas nomenclaturas clases 7](#_Toc156811866)

[Políticas nomenclaturas funciones 7](#_Toc156811867)

# Descripción

Este documento permite establecer las políticas necesarias para la correcta nomenclatura al momento de implementar un desarrollo de software. Siendo así, una guía sobre como se debe de nombrar los diversos componentes dentro de un proyecto, permitiendo una mejor legibilidad, mantenibilidad y colaboración entre los integrantes de equipo o personal externo que participen durante la creación de proyectos.

# Objetivo

Establecer un estándar en la nomenclatura de los componentes que hacen parte de la elaboración de proyectos durante las fases temprana del desarrollo.

1. **Que son las políticas de nomenclatura**

Son un conjunto de reglas establecidas para guiar en la selección de nombres en un entorno de desarrollo de software, permitiendo consistencia, claridad y legibilidad en nombramiento de proyectos y desarrollo de código y ambientes en los cuales se desarrolla, facilitando la colaboración dentro de un equipo de desarrollo.

# Entornos de desarrollo

Durante el desarrollo de software, se establece la convención de nomenclatura específica para diferencia entre entornos de pruebas y entornos de producción. Esta convención se implementa mediante la adición de sufijos.

## 4.1 Entorno de pruebas

Para los elementos relacionados con pruebas, ya sea en el código fuente, bases de datos o cualquier otro recurso destinado a ambientes de prueba, se adopta la convención de agregar el sufijo -qa al final del nombre.

**Ejemplo para base de datos:**

ecommerce\_user-qa.sql

**Ejemplo para nombre del proyecto**

tienda\_ecommerce\_bemaster-qa

## 4.2 Entorno de producción

Para los elementos relacionados con pruebas, ya sea en el código fuente, bases de datos o cualquier otro recurso destinado a ambientes de prueba, se adopta la convención de agregar el sufijo -pdn al final del nombre.

**Ejemplo para base de datos:**

ecommerce\_user-pdn.sql

**Ejemplo para nombre del proyecto**

tienda\_ecommerce\_bemaster-pdn

# Tabla de las políticas

|  |  |  |  |
| --- | --- | --- | --- |
| **Política** | **Identificador** | **Descripción** | **Cumplimiento** |
| Elemento al cual va dirigida la política especificando la correcta nomenclatura | código único asignado a cada política para facilitar la identificación y referencia. | Explicación de para que sirve la política. | Mecanismo por el cual se valida que se cumple de forma adecuada la nomenclatura. |

## 5.1 Políticas nomenclaturas carpeta del proyecto

|  |  |  |  |
| --- | --- | --- | --- |
| **Politica** | **Identificador** | **Descripción** | **Cumplimiento** |
| Nombre carpeta de proyecto | PRO-001 | Utilización de un nombre descriptivo que de claridad de que será el proyecto | Validación por el equipo de que el nombre es apropiado y sea claro su intención. Ejemplos:  ecommerce-nombre-empresa.  blog-nombre-empresa |

## 5.2 Políticas nomenclaturas Git

|  |  |  |  |
| --- | --- | --- | --- |
| **Politica** | **Identificador** | **Descripción** | **Cumplimiento** |
| Nombre de la rama en Git | GIT-001 | Generar nombres de rama acorde a la solución que se va estar elaborando | Estandarizar las ramas necesarias que se utilizaran dentro de los proyectos. Ejemplos:  Qa : developer  Pdn : main  Cambios: feature |
| Creación de commit | GIT-002 | Poder generar commits adecuados que señalen si lo que se está revisando es una corrección o una funcionabilidad nueva | Revisar dentro del equipo la manera adecuada de gestionar cuando es un cambio o algo nuevo, por prefijos antes de que inicie la descripción del commit como puede ser FIX – ADD – DELETE para estandarizar. Ejemplos:  Git commit -m “ADD se agregó nuevo navbar x”  Git commit -m “FIX solución de estilos del navbar x” |

## Políticas nomenclaturas base de datos

|  |  |  |  |
| --- | --- | --- | --- |
| **Politica** | **Identificador** | **Descripción** | **Cumplimiento** |
| Nombre de las bases de datos | BAS-001 | En la creación de las bases de datos, los nombres serán separados por guion bajo. | Verificar que las bases de datos tengan correctamente la nomenclatura. Ejemplo:  ecommerce\_nombre\_empresa-qa.sql |
| Nombramiento de tablas | BAS-002 | Poder estandarizar el nombramiento de las tablas dentro de las bases de datos para agilizar las consultas SQL | Revisar dentro de las bases de datos la forma correcta de nombramiento de las tablas, siendo esta de forma singular. Ejemplos:  Evento – Producto – Usuario |

## 5.4 Políticas nomenclaturas variables

|  |  |  |  |
| --- | --- | --- | --- |
| **Politica** | **Identificador** | **Descripción** | **Cumplimiento** |
| Nombramiento de variables | VAR-001 | Utilización adecuada de nombres dentro del código que ayuden a identificar de forma clara para que se está utilizando | Revisar en los refinamientos o reuniones de proyecto la adecuada creación de variables. |
| Nombramiento de constantes | VAR-002 | Las variables que durante la ejecución del programa se sabe que no cambiarán sus valores tendrán un formato diferente. | Revisar en los refinamientos o reuniones de proyecto la adecuada creación de variables constantes siendo esta de letra capital/mayúscula. Ejemplo:  api key de mongo: API\_KEY\_MONGO |

## 5.5 Políticas nomenclaturas clases

|  |  |  |  |
| --- | --- | --- | --- |
| **Politica** | **Identificador** | **Descripción** | **Cumplimiento** |
| Nombres descriptivos de herencia | CLA-001 | Utilizar nombres descriptivos en caso de herencia para reflejar la relación entre clases. | Validación por el equipo de que el nombre es apropiado y sea claro su intención. Ejemplo:  Creación de un coche o vehiculo lo ideal es que hereden de una clase como Vehiculo |

## 5.6 Políticas nomenclaturas funciones

|  |  |  |  |
| --- | --- | --- | --- |
| **Politica** | **Identificador** | **Descripción** | **Cumplimiento** |
| Nombres claros para las funciones | FUN-001 | Utilizar nombres descriptivos que sean claros para que sirven dentro del código | Validación por el equipo de que el nombre es apropiado y no sea excesivamente largo, limitando la longitud para mejorar la legibilidad de estos. Ejemplo:  Evitar utilizar nombres largos calcularSumaYDevolverResultado por calcularSuma |