# 

# 

**POLÍTICAS DE NOMENCLATURA**

# 

# 

# 

# 

# 

# 

# 

# 

[**Políticas de Nomenclatura para el Equipo de Desarrollo 3**](#_ni3kc0oq766v)

[1. Bases de Datos 3](#_cec2985yh7p)

[1.1. Nombres de Tablas 3](#_nznfvg7zfj42)

[1.2. Nombres de Columnas 3](#_hnwo9tk7szar)

[1.3. Índices 3](#_g660v5tzdx3t)

[1.4. Claves Foráneas 3](#_s6yxuxgg6iab)

[2. Variables 4](#_y06e3tjo2ggc)

[2.1. Variables en JavaScript/TypeScript 4](#_duqhznepkjvo)

[2.2. Variables en CSS/SCSS 4](#_51uhvhj3cwb7)

[3. Funciones 4](#_rtr763mhimbo)

[3.1. Nombres de Funciones en JavaScript 4](#_o1y72ni67esp)

[4. Clases 4](#_qx99uouzavco)

[4.1. Nombres de Clases en JavaScript/TypeScript 4](#_pzldm6y6si47)

[5. Git 5](#_15lzntn2riej)

[5.1. Ramas 5](#_iv3xr5wg8r1o)

[5.2. Commits 5](#_vlvy8fud5jcp)

[6. Consideraciones 6](#_23kp00dpqm2y)

[6.1. Archivos y Directorios 6](#_b5gb79v2159v)

[6.2. Configuración de Entorno 6](#_rg5rxu4o7exd)

[6.3. Documentación y Comentarios 6](#_r9if4lb28dv6)

# 

# **Políticas de Nomenclatura para el Equipo de Desarrollo**

## **1. Bases de Datos**

### **1.1. Nombres de Tablas**

* Utilizar nombres en inglés.
* Usar nombres en plural.
* Separar palabras con guiones bajos (\_).

**Ejemplo:**

* users
* order\_items

### **1.2. Nombres de Columnas**

* Utilizar nombres en inglés.
* Usar snake\_case (minúsculas y guiones bajos).
* Ser descriptivo pero conciso.

**Ejemplo:**

* user\_id
* created\_at

### **1.3. Índices**

* Prefijo idx\_ seguido del nombre de la tabla y las columnas involucradas.

**Ejemplo:**

* idx\_users\_email
* idx\_order\_items\_order\_id\_product\_id

### **1.4. Claves Foráneas**

* Prefijo fk\_ seguido del nombre de la tabla y la columna de referencia.

**Ejemplo:**

* fk\_orders\_user\_id
* fk\_order\_items\_product\_id

## **2. Variables**

### **2.1. Variables en JavaScript/TypeScript**

* Utilizar camelCase para variables locales y parámetros de funciones.
* Utilizar UPPER\_SNAKE\_CASE para constantes.

**Ejemplo:**

let userName = '';

const MAX\_RETRIES = 5;

### **2.2. Variables en CSS/SCSS**

* Utilizar kebab-case.

**Ejemplo:**

$primary-color: #3498db;

## **3. Funciones**

### **3.1. Nombres de Funciones en JavaScript**

* Utilizar camelCase.
* Ser descriptivo y empezar con un verbo.

**Ejemplo:**

function getUserById(userId) {

// código de la función

}

const calculateTotalPrice = (items) => {

// código de la función

};

## **4. Clases**

### **4.1. Nombres de Clases en JavaScript/TypeScript**

* Utilizar PascalCase.
* Ser descriptivo y usar sustantivos.

**Ejemplo:**

class UserController {

// métodos y propiedades de la clase

}

class ShoppingCart {

// métodos y propiedades de la clase

}

## **5. Git**

### **5.1. Ramas**

* Utilizar kebab-case.
* Prefijar con el tipo de rama: feature, bugfix, hotfix, release.

**Ejemplo:**

* feature/add-user-authentication
* bugfix/fix-login-error

### **5.2. Commits**

* Utilizar mensajes claros y descriptivos.
* Estructura: [tipo]: [descripción].

**Tipos comunes:**

* feat: Nueva funcionalidad.
* fix: Corrección de errores.
* refactor: Cambios en el código que no afectan su funcionamiento.
* docs: Cambios en la documentación.
* style: Cambios de formato, no de código.
* test: Añadir o corregir tests.
* chore: Tareas de mantenimiento.

**Ejemplo:**

**feat:** add user authentication module

**fix:** correct typo in login function

**refactor:** optimize user data fetching logic

## **6. Consideraciones**

### **6.1. Archivos y Directorios**

* Utilizar kebab-case para nombres de archivos y directorios.
* Ser descriptivo y conciso.

**Ejemplo:**

components/

user-profile/

user-profile.component.js

user-profile.styles.css

### **6.2. Configuración de Entorno**

* Utilizar UPPER\_SNAKE\_CASE para variables de entorno.

**Ejemplo:**

DATABASE\_URL=postgres://user:password@localhost:5432/mydb

API\_KEY=1234567890abcdef

### **6.3. Documentación y Comentarios**

* Escribir comentarios claros y descriptivos.
* Utilizar JSDoc para funciones, métodos y clases.

**Ejemplo:**

javascript

Copiar código

/\*\*

\* Obtiene un usuario por su ID.

\* @param {number} userId - El ID del usuario.

\* @returns {Object} El usuario encontrado.

\*/

function getUserById(userId) {

// código de la función

}