Estándares de Código y Nomenclatura de Ramas - Proyecto Bookshare

Integrante responsable: Néstor Ayala (Tester / QA)

Proyecto: Bookshare

Fecha: Octubre 2025

1. Convenciones de Código

Estas reglas buscan mantener un estilo de código coherente, limpio y fácil de mantener en todo el proyecto Bookshare. Aplican para el frontend (React) y backend (Node.js + Express).

JavaScript / Node.js

- Usar camelCase para variables y funciones → getUserData()
- Usar PascalCase para clases o componentes → UserCard.jsx
- Cada archivo debe exportar una sola función o clase principal.
- Usar const y let (no var).
- Indentación de 2 espacios.
- Comentarios breves y descriptivos.
- Validar funciones con pruebas unitarias (Jest).

React

- Un componente por archivo.
- Archivos con nombre de componente → BookForm.jsx
- Evitar lógica compleja dentro del JSX.
- Usar TailwindCSS para el estilo.

2. Estándares de Commits

Los mensajes de commit deben seguir un formato estandarizado que describa claramente el cambio realizado.

Formato: <tipo>: <descripción corta>

Ejemplo:
feat: agregar validación de registro de usuario
fix: corregir error en formulario de publicación
test: agregar pruebas unitarias para endpoint /login
refactor: simplificar componente BookCard

Tipos de commit:
- feat → nueva funcionalidad
- fix → corrección de errores
- test → creación o modificación de pruebas
- refactor → mejoras internas del código
- docs → cambios en documentación

3. Estrategia de Ramas Git

Aunque el proyecto Bookshare se desarrolla en un entorno monolítico, se utilizará una estrategia de ramas simple para
evitar conflictos y mantener el control de versiones.

Rama	Propósito
main	Código estable y probado.
dev	Rama de desarrollo general donde se integran las nuevas características.
feature/<nombre>	Desarrollo de una nueva funcionalidad. Ej: feature/chat o feature/login.
hotfix/<nombre>	Corrección urgente de errores detectados en producción.

4. Control de Calidad

Antes de subir cualquier cambio al repositorio, se deben cumplir las siguientes verificaciones:
- Ejecutar npm run lint (si se utiliza ESLint).
- Ejecutar npm test para validar pruebas unitarias y TDD.

- Revisar que el código cumpla con las convenciones establecidas.
- Realizar revisión de código cruzada (pair review) antes del merge.

5. Convenciones de Nombres

Las siguientes reglas aseguran consistencia en la nomenclatura de archivos, ramas y variables.

Elemento	Ejemplo	Regla
Variables	userName	camelCase
Componentes React	BookList	PascalCase
Archivos JS	userController.js	minúsculas y guion medio
Ramas Git	feature/login-form	minúsculas separadas con guion (-)