Javier Santiago Giraldo Jimenez jgiraldoji@unal.edu.co
Manuel Alejandro Navas Bohorquez mnavas@unal.edu.co
Alvaro Andres Romero Castro alromeroca@unal.edu.co
Tomas Sebastian Vallejo Fonseca tvallejof@unal.edu.co

Proyecto Final

Ingeniería de Software I
Departamento de Ingeniería de Sistemas e Industrial
UNAL

Punto 7: Clean Code en ReservaPlus

Dado que aún no hay código en el proyecto, este documento establecerá las bases para la implementación de buenas prácticas de Clean Code y herramientas de análisis estático en ReservaPlus.

- **Selección de herramientas de análisis estático**

Para mantener un código limpio y fácil de mantener en ReservaPlus, seleccionaremos herramientas de análisis estático según cada tecnología usada en el proyecto.

Componente	**Lenguaje / Framework**	**Herramienta de análisis**	**Motivo de elección**
Frontend	JavaScript, TypeScript (React.js)	ESLint, Prettier	Detecta errores, asegura estilo y formato automático
Backend	JavaScript (Node.js, Express.js)	ESLint, SonarQube	Evalúa calidad del código, detecta errores comunes
Base de Datos	PostgreSQL, MongoDB	SQLFluff (SQL), MongoDB Atlas Performance Advisor	Revisión de consultas SQL y optimización en MongoDB

- **Automatización del análisis de código**
-

Para integrar estas reglas en el flujo de trabajo, proponemos la validación del código mediante:

- **Ejecución manual del linter**: Se recomienda ejecutar los linters manualmente antes de cada commit para garantizar que el código cumple con los estándares establecidos.
- **Integración con GitHub Actions**:
  - Analiza el código con ESLint y SonarQube en cada Pull Request.

    - Bloquea la fusión de código si no cumple con los estándares.
- **Revisión de código entre desarrolladores**:
    - Uso de una plantilla de Pull Request con checklist de calidad.
    - Comentarios en código para mejorar legibilidad y estructura.

En cuanto a la siguiente parte, estamos en fase de planeación por lo que escribir código no es útil aún.