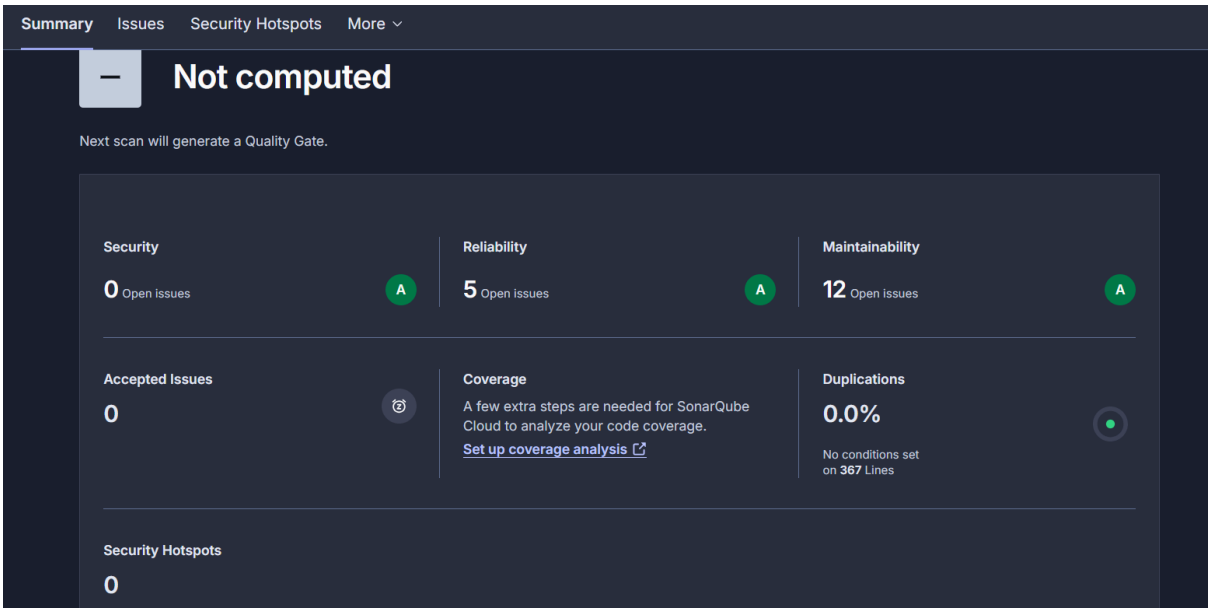
Documento de Evidencia de Auditoría de Calidad (Versión Refactorizada)

Asignatura y Tema

- Asignatura: Arquitectura y Diseño del Software
- Autor:Melanny Guate
- Fecha: 11 de noviembre de 2025
- Tema: Auditoría de Calidad Post-Refactorización con GitHub y SonarCloud.
- Objetivo de la Clase: Documentar la auditoría de calidad de la versión optimizada (Refactorizada) y los problemas pendientes.

Paso 1: Evidencia de Conexión y Ejecución de la Auditoría (Dashboard)

1. Versión Refactorizada



Paso 2: Análisis de Calidad de la Versión Refactorizada

- Instrucción: Documentar el estado del Quality Gate, las clasificaciones (A, B, C, D), el número de Issues para las métricas clave y el porcentaje de duplicación, utilizando la información de la sección Summary de SonarCloud.

1. Auditoría de Métricas

- Nota: La versión Refactorizada presenta 17 problemas abiertos en total, principalmente de Mantenibilidad.

Métrica / Atributo	Versión Refactorizada
Quality Gate	[Registrar el estado: Pasó/Falló]
Reliability (Fiabilidad)	Clasificación: **A** / Issues Abiertos: **5**
Maintainability (Mantenibilidad)	Clasificación: **A** / Issues Abiertos: **12**
Security (Seguridad)	Clasificación: **A** / Issues Abiertos: **0**
Duplicación	Porcentaje: **0.0%**

2. Resumen de Problemas (Issues y Security Hotspots)

- Instrucción: Documentar la cantidad total de problemas encontrados en la sección Issues y clasificarlos por severidad.

Versión Refactorizada

- Cantidad Total de Problemas: 17 (Estimado: 5 Fiabilidad + 12 Mantenibilidad)
- Bugs (Estimado por Fiabilidad): Total Fiabilidad: 5
- Vulnerabilities (Estimado por Seguridad): 0
- Code Smells (Estimado por Mantenibilidad): 12
- Security Hotspots: 0

Paso 3: Documentación de Hallazgos Prioritarios Pendientes

Hallazgo Prioritario 1: Declaración Léxica Inesperada en switch

- Versión del Issue: Refactorizada
- Tipo de Problema: Code Smell (Major - Mantenibilidad)
- Captura de Pantalla del Problema (SonarCloud):
  [Incluya una captura de pantalla del Code Smell Unexpected lexical declaration in case block.]
- Descripción del Problema e Impacto: Se encontró una declaración de variables con let o const directamente dentro de un bloque case de una estructura switch. Esto puede generar errores de ámbito de variables, ya que la declaración no está contenida dentro de su propio bloque de código ({}). El impacto es en la Mantenibilidad y la Fiabilidad, ya que el comportamiento del código puede ser inesperado o llevar a errores de lógica difíciles de depurar.

- Plan de Corrección Futuro: [Describa brevemente cómo se planea corregir este issue en la siguiente iteración de desarrollo. (Solución: Encerrar las declaraciones de variables dentro de un bloque de llaves {} en el case correspondiente).]

Hallazgo Prioritario 2: Contraste Mínimo de Texto (Accesibilidad)

- Versión del Issue: Refactorizada
- Tipo de Problema: Code Smell (Major - Mantenibilidad)
- Captura de Pantalla del Problema (SonarCloud):
  [Incluya una captura de pantalla del Code Smell Text does not meet the minimal contrast requirement with its background.]
- Descripción del Problema e Impacto: El código CSS utiliza combinaciones de colores de texto y fondo que no cumplen con los requisitos mínimos de contraste de accesibilidad (WCAG 2.1). Esto afecta directamente la Usabilidad para usuarios con deficiencias visuales o para cualquier usuario en condiciones de mucha luz. Aunque es un *Code Smell* de Mantenibilidad, su impacto real es en la Accesibilidad y la experiencia del usuario.
- Plan de Corrección Futuro: [Describa brevemente cómo se planea corregir este issue en la siguiente iteración de desarrollo. (Solución: Ajustar los valores hexadecimales del color del texto o del fondo para asegurar que la relación de contraste sea al menos 4.5:1).]

Hallazgo Prioritario 3: Uso de la Función Obsoleta parseInt()

- Versión del Issue: Refactorizada
- Tipo de Problema: Code Smell (Minor - Fiabilidad y Mantenibilidad)
- Captura de Pantalla del Problema (SonarCloud):
  [Incluya una captura de pantalla del Code Smell Prefer 'Number.parseInt()' over 'parseInt()'.]
- Descripción del Problema e Impacto: El código está utilizando la función global parseInt(). Aunque sigue siendo funcional, el estándar moderno de ECMAScript (ES2015) promueve el uso de Number.parseInt(). El uso de la versión global puede causar confusión de contexto y, en ciertas optimizaciones, puede tener un rendimiento ligeramente inferior. El impacto es en la Mantenibilidad y el cumplimiento de las convenciones de código moderno.
- Plan de Corrección Futuro: [Describa brevemente cómo se planea corregir este issue en la siguiente iteración de desarrollo. (Solución: Reemplazar todas las instancias de parseInt() por la función estática Number.parseInt()).]

Hallazgo Prioritario 4: Uso de la Función Obsoleta parseFloat()

- Versión del Issue: Refactorizada
- Tipo de Problema: Code Smell (Minor - Fiabilidad y Mantenibilidad)
- Captura de Pantalla del Problema (SonarCloud):
  [Incluya una captura de pantalla del Code Smell Prefer 'Number.parseFloat()' over 'parseFloat()'.]
- Descripción del Problema e Impacto: Similar al Hallazgo 3, se está utilizando la función global parseFloat(), que se considera obsoleta por la convención ES2015 en

favor de Number.parseFloat(). El impacto es en la Mantenibilidad y la uniformidad del código, ya que el uso de funciones globales sin el prefijo de su objeto puede ser menos claro sobre su origen.

- Plan de Corrección Futuro: [Describa brevemente cómo se planea corregir este issue en la siguiente iteración de desarrollo. (Solución: Reemplazar todas las instancias de parseFloat() por la función estática Number.parseFloat()).]

Hallazgo Prioritario 5: Bug de Fiabilidad Pendiente (Estimación)

- Versión del Issue: Refactorizada
- Tipo de Problema: Bug (Medium - Fiabilidad)
- Captura de Pantalla del Problema (SonarCloud):
  [Incluya una captura de pantalla de uno de los 5 Bugs de Fiabilidad restantes (por ejemplo, una asignación potencialmente nula o un error en un bucle).]
- Descripción del Problema e Impacto: Dado que el Dashboard indica 5 Issues de Fiabilidad, este hallazgo representa uno de los Bugs restantes que podrían causar un comportamiento inesperado. Por ejemplo, podría tratarse de una validación faltante en un bucle (while o for) o una comparación incorrecta de tipos. El impacto principal es en la Fiabilidad, ya que, aunque el código se refactorizó, este Bug específico aún tiene el potencial de fallar en producción.
- Plan de Corrección Futuro: [Describa brevemente cómo se planea corregir este issue en la siguiente iteración de desarrollo. (Solución: Revisar la lógica de la línea reportada y aplicar el *fail-fast principle* o asegurar la inicialización correcta de variables para eliminar la posibilidad de fallo en tiempo de ejecución).]