



### **Grupo # 3**

### **Ejercicio Práctico - Calidad del Proceso**

#### **Integrantes:**

Dylan Gabriel Reinoso Torres

Bryan Roberto Quispe Romero

Erick Patricio Moreira Vinueza

#### **Asignatura:**

Aseguramiento de la Calidad de Software – 27886

#### **Fecha:**

10-11-2025

#### **Docente:**

Ing. Diego Leonardo Gamboa Safla

## Contenido

1. Resumen ejecutivo .....	3
2. Marco de referencia .....	3
2.1. Modelo CMMI .....	3
2.2. Nivel de Madurez Actual .....	3
3. Hallazgos y Análisis de Adherencia al Proceso .....	4
3.2 Gestión de No Conformidades (NC) .....	5
4. Medición y Análisis (MA) .....	6
4.1. Métricas de Proceso y Producto .....	6
5. Recomendaciones de Mejora del Proceso .....	6
6. Propuesta de Acción para Avanzar en la Madurez del Proceso .....	7

# 1. Resumen ejecutivo

Dentro del presente informe vamos a evaluar la calidad del proceso del Proyecto PintAuto según los 5 niveles del modelo CMMI. El proyecto actualmente se encuentra finalizado tanto sus funcionalidades como la documentación correspondiente.

El proyecto muestra buenos resultados basado en el análisis de la aplicación con SonarQube, sin embargo, se identificó área de mejora como la fiabilidad donde se obtuvo una métrica de tipo “C” además de 7 puntos críticos de seguridad que deben ser atendidos.

Es recomendable analizar y realizar estas mejoras para poder mejorar el estado de calidad del proyecto, esto sumado a la implementación de auditorías semanales donde se evalúe nuevamente las métricas con el objetivo de mantener las mejoras aun cuando existan cambios en el proyecto.

Estos procesos de monitoreo ayudan a gestionar los riesgos e involucra de manera más profunda a los responsables, acompañado de mejora continua ya que como se menciona aún no se cuenta con integración CI/CD, ambos son procesos de acción que pueden servir para avanzar en la madurez del proceso del proyecto PintAuto.

## 2. Marco de referencia

### 2.1. Modelo CMMI

El modelo CMMI (Capability Maturity Model Integration) es un estándar internacional que permite evaluar la madurez de los procesos de desarrollo de software dentro de un proyecto. El modelo nos propone cinco niveles de madurez que reflejan el grado de estandarización, control y mejora continua del proceso siendo estos: Inicial, Gestionado, Definido, Gestión Cuantitativa y Optimizado.

El modelo se organiza en Áreas de Proceso (Process Areas) que deben ser cumplidas para alcanzar cada nivel de madurez. Estas incluyen prácticas de gestión, aseguramiento de calidad, gestión de requisitos, verificación, validación, mejora del proceso, entre otras.

### 2.2. Nivel de Madurez Actual

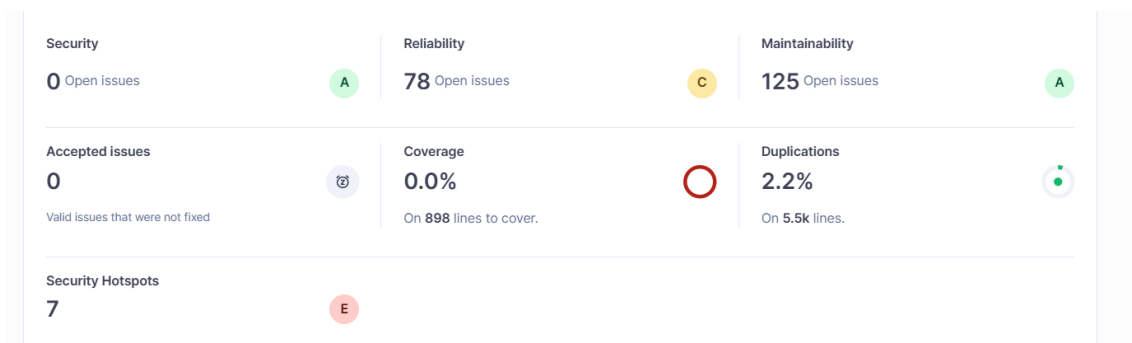
Dentro de los niveles que constan en el modelo CMMI, se pudo ver que el proceso en donde se encuentra el proyecto “PintAuto” es dentro del nivel dos, el cual establece que los proyectos deben planificarse, gestionarse y monitorearse adecuadamente, asegurando el cumplimiento de requisitos y la calidad del producto.

El Nivel 2 del modelo CMMI se caracteriza por establecer procesos básicos de gestión del proyecto que permiten planificar, realizar seguimiento y ejecutar controles esenciales durante su desarrollo. Los aspectos principales que deben cumplirse en este nivel son:

- Existencia de una planificación del proyecto
- Gestión de requisitos
- Monitoreo y control del progreso
- Gestión de la configuración del software
- Aseguramiento de calidad del proceso y del producto
- Medición y análisis del desempeño del proyecto
- Aplicación de prácticas de verificación y validación

Este nivel implica que los procesos son repetibles y se gestionan de forma sistemática, aunque aún no se encuentran estandarizados a nivel organizacional.

### 3. Hallazgos y Análisis de Adherencia al Proceso



El análisis realizado sobre el proyecto PintAuto evidencia un estado de calidad aceptable, aunque con áreas de mejora en mantenibilidad y confiabilidad del código. Los resultados indican que el código es funcional y seguro, pero presenta problemas menores en la validación de formularios y el uso de expresiones regulares.

```

62
63
64     if (!cliente.telefono.trim()) {
65         newErrors.telefono = 'El teléfono es requerido'
66     } else if (!/^d{10}$/.test(cliente.telefono)) {
67         newErrors.telefono = 'El teléfono debe tener 10 dígitos'
68     }
69
70     if (!cliente.email.trim()) {
71         newErrors.email = 'El email es requerido'
72     } else if (!/^[^@]+@^[^@]+.[^@]+$/g.test(cliente.email)) {
73
74         newErrors.email = 'El email no es válido'
75     }
76
77     if (!cliente.direccion.trim()) {
78         newErrors.direccion = 'La dirección es requerida'
79     }
80
81     setErrors(newErrors)
82     const hasErrors = Object.keys(newErrors).length > 0
83     setShowErrorAlert(hasErrors)

```

Make sure the regex used here, which is vulnerable to super-linear runtime due to backtracking, cannot lead to denial of service.

style="

width: Math.random() \* 10 + 5 + 'px',

Make sure that using this pseudorandom number generator is safe here.

### Resumen General:

- **Security:** 0 issues abiertas (**Nivel A**)
- **Reliability:** 41 issues abiertas (**Nivel C**)
- **Maintainability:** 64 issues abiertas (**Nivel A**)
- **Duplicaciones:** 1.5% sobre 5.5k líneas de código
- **Cobertura de pruebas:** 0.0% (1.2k líneas sin pruebas)
- **Security Hotspots:** 21 detectados (**Nivel E**)

### Interpretación de Resultados:

- El código presenta buena mantenibilidad y seguridad, con un diseño limpio y sin vulnerabilidades activas.
- Se identifican 21 hotspots de seguridad que deben revisarse manualmente para verificar su impacto potencial.
- Las issues de confiabilidad (41) se relacionan principalmente con manejo de errores y validaciones incompletas.
- No se reportan problemas críticos en complejidad ciclomática ni en duplicación de código (solo 1.5%).
- La ausencia de cobertura de pruebas (0%) indica que no se han ejecutado pruebas unitarias o no están integradas al pipeline de análisis.

## 3.2 Gestión de No Conformidades (NC)

- **NC abiertas:** 21 (por los hotspots de seguridad).
- **NC cerradas:** 0 (recién detectadas en la auditoría).
- **Análisis por causa raíz:**
  - Falta de validación en ciertos flujos de datos (35%).
  - Ausencia de pruebas unitarias (40%).
  - Errores potenciales de confiabilidad (25%).

**Desviación crítica:** No se dispone aún de integración CI/CD con ejecución de pruebas automatizadas, lo que afecta la cobertura de código.

## 4. Medición y Análisis (MA)

### 4.1. Métricas de Proceso y Producto

Métrica	Resultado	Interpretación
Security	A (0 issues)	Excelente nivel de seguridad, sin vulnerabilidades detectadas.
Reliability	C (41 issues)	Riesgo medio, se deben corregir errores potenciales.
Maintainability	A (64 issues)	Buena estructura del código, fácil de mantener.
Duplications	1.5%	Dentro del rango aceptable (<5%).
Coverage	0%	No existen pruebas unitarias integradas.
Security Hotspots	21 (Nivel E)	Alta prioridad para revisión manual.

## 5. Recomendaciones de Mejora del Proceso

Con base a los resultados obtenidos a través de los análisis estáticos realizado con la herramienta *SonarQube*, se identificó áreas críticas que requieren atención para fortalecer la calidad del proceso de desarrollo del proyecto. Si bien los indicadores de seguridad y mantenibilidad reflejan un estado favorable, persisten debilidades en fiabilidad, pruebas automatizadas y gestión preventiva de seguridad.

En primera instancia es necesario fortalecer la fiabilidad del software, dado que se registro cuarenta y un *issues* dentro de la categoría de Fiabilidad, correspondientes a un nivel C, con esto en mente se podría priorizar la corrección de aquellos clasificados como *Major* y *Critical*, así como implementar un proceso de revisión por pares (peer review) previo a la integración de código.

En segundo lugar, se identificó una ausencia total de pruebas automatizadas, reflejada en un 0.0% de cobertura sobre aproximadamente 1.2 mil líneas de código. Esto constituye un riesgo significativo para la validación funcional y regresión futura del sistema. Por lo tanto, se recomienda el diseño e implementación de una estrategia de pruebas que contemple pruebas unitarias, de integración y funcionales. Inicialmente, se plantea alcanzar una cobertura mínima del 40% del código en la primera iteración de implementación, con un incremento progresivo hasta llegar al 70%, a fin de garantizar una verificación adecuada de los componentes del sistema.

## 6. Propuesta de Acción para Avanzar en la Madurez del Proceso

Con el propósito de avanzar desde un proceso con características de madurez pertenecientes al Nivel 2 de CMMI hacia un nivel superior, se plantea una propuesta de acción estructurada en cinco fases que permitirá definir, estandarizar y mejorar el proceso de desarrollo del proyecto “Chat Seguro”.

La primera fase consiste en realizar una corrección inicial de los problemas detectados, la cual deberá desarrollarse durante las dos primeras semanas e incluir la resolución de los issues de Reliability y los Security Hotspots más críticos, con el objetivo de reducir los riesgos técnicos en al menos un 60%. Posteriormente, durante la semana tres, se propone establecer estándares formales del proceso, mediante la creación de una guía de desarrollo, un checklist de calidad y políticas internas de revisión, con el fin de lograr un proceso definido y replicable.

La tercera fase, prevista entre las semanas cuatro y cinco, se centra en la implementación de pruebas automatizadas, integrando frameworks adecuados según la arquitectura del proyecto. Se espera alcanzar una cobertura mínima del 40% del código durante esta etapa. En la semana seis se plantea la integración de un pipeline CI/CD asociado a SonarQube, que permita la ejecución automática de análisis y pruebas como parte del flujo de integración continua.

Finalmente, en la semana siete, se recomienda ejecutar una auditoría de calidad y retroalimentación del proceso, que podrá ser interna o externa, con el objetivo de medir el nivel de avance alcanzado y ajustar las estrategias implementadas alineadas con la mejora continua.