**Evaluación Módulo 4**

**Automatización de Pruebas en una Plataforma de Salud**

**Nombre:** Juan Pablo Urra Jara

**Curso:** Fundamentos DevOps

Contenido

[**Análisis del estado actual de la plataforma** 3](#_Toc202131971)

[Descripción del error en la lógica del código 3](#_Toc202131972)

[Impacto del error en la experiencia del usuario. 3](#_Toc202131973)

[Falta de procesos de validación y pruebas en el desarrollo actual. 3](#_Toc202131974)

[Diseño y desarrollo de pruebas automatizadas 4](#_Toc202131975)

[Pruebas unitarias: Implementación de pruebas en JUnit para validar la lógica de actualización de peso. 4](#_Toc202131976)

[Pruebas funcionales: Simulación de flujo completo de usuario con Selenium o Cypress. 4](#_Toc202131977)

[Pruebas de regresión: Estrategia para detectar errores en futuras actualizaciones. 4](#_Toc202131978)

[Pruebas de rendimiento: Evaluación del tiempo de respuesta usando JMeter. 4](#_Toc202131979)

[Automatización del proceso de pruebas con CI/CD 4](#_Toc202131980)

[Creación de un pipeline en GitHub Actions o Jenkins para ejecutar automáticamente las pruebas. 4](#_Toc202131981)

[Configuración de reportes de resultados y alertas de fallos. 4](#_Toc202131982)

[Validación de calidad del código con herramientas como SonarQube. 4](#_Toc202131983)

# **Análisis del estado actual de la plataforma**

## Descripción del error en la lógica del código

public void actualizarPeso(double nuevoPeso) {

this.peso -= 1;

}

En lugar de asignar el nuevo peso, se está restando 1kg.

El sistema no está utilizando nuevoPeso, lo que causa una disminución de 1 kg sin importar el valor ingresado.

## Impacto del error en la experiencia del usuario.

* Resultados incorrectos en el historial del usuario.
* Pérdida de confianza en la aplicación.
* Imposibilidad de monitoreo efectivo, potencialmente perjudicial para personas con requerimientos médicos.

## Falta de procesos de validación y pruebas en el desarrollo actual.

* No existen pruebas unitarias, de integración ni funcionales.
* No hay CI/CD, lo que permite que errores lleguen al entorno de producción.

# Diseño y desarrollo de pruebas automatizadas

## Pruebas unitarias: Implementación de pruebas en JUnit para validar la lógica de actualización de peso.

## Pruebas funcionales: Simulación de flujo completo de usuario con Selenium o Cypress.

## Pruebas de regresión: Estrategia para detectar errores en futuras actualizaciones.

## Pruebas de rendimiento: Evaluación del tiempo de respuesta usando JMeter.

# Automatización del proceso de pruebas con CI/CD

## Creación de un pipeline en GitHub Actions o Jenkins para ejecutar automáticamente las pruebas.

## Configuración de reportes de resultados y alertas de fallos.

## Validación de calidad del código con herramientas como SonarQube.

El desarrollo de los puntos anteriores se puede encontrar en el siguiente repositorio de Github: <https://github.com/ojitxslml/Eva4-devops-adalid>