UNIVERSIDAD PRIVADA DE TACNA

FACULTAD DE INGENIERIA

Escuela Profesional de Ingeniería de Sistemas

Análisis y Mejoramiento del sistema intranet de la empresa LADITEC

Curso: Calidad y Pruebas de Software

Docente: Ing. Patrick Cuadros

Integrantes:

Condori Loayza, Helbert Andres	**(2020067571)**
Aranda Reyes, Diego Andre	**(2019063855)**
Mamani Lima, Erick Mauricio	**(2020066321)**
Rivera Mendoza, Jhonny	**(2020067144)**

Tacna – Perú
2023

CONTROL DE VERSIONES					
Versión	Hecha por	Revisada por	Aprobada por	Fecha	Motivo
1.0	Helbert Condori, Erick Mamani, Jhonny Rivera y Diego Aranda	Helbert Condori, Erick Mamani, Jhonny Rivera y Diego Aranda	Helbert Condori, Erick Mamani, Jhonny Rivera y Diego Aranda	23/04/2023	Realización del documento versión 1.0

INDICE GENERAL

Resumen....3

Abstract....3

1. Antecedentes o introducción....3

2. Titulo....3

3. Autores....4

4. Planteamiento del problema....4

4.1. Problema....4

4.2. Justificación....4

4.3. Alcance....4

5. Objetivos....4

5.1. General....4

5.2. Específicos....4

6. Referentes teóricos....5

- Diagramas de Casos de Uso, Diagrama de Clases, Diagrama de Componentes y Arquitectura. 5

7. Desarrollo de la propuesta (Aqui va el analisis de su aplicación con SonarQube, para que les muestre todos los aspectos a mejorar de su aplicación) 6

7.1. Tecnología de información 8

7.2. Metodología, técnicas usadas....8

8. Cronograma (personas, tiempo, otros recursos) Basado en las observaciones que la herramienta SonarQube les informara sobre la aplicación, a fin de reducir la deuda tecnica, vulnerabilidades, fallas, etc. a 0. 8

Resumen

El objetivo es reducir la deuda técnica, vulnerabilidades, fallas, etc. a 0 en un proyecto determinado. Para lograr esto, se utilizará una herramienta de análisis estático de código como SonarQube para identificar las áreas problemáticas y definir las prioridades. Se asignará un equipo de desarrolladores para corregir los problemas identificados, utilizando las mejores prácticas de desarrollo de software y siguiendo las recomendaciones de la herramienta. Se establecerá un cronograma y se realizarán revisiones periódicas para evaluar el progreso del proyecto y ajustar los plazos según sea necesario. Al finalizar el proyecto, se lanzará la aplicación libre de deuda técnica y vulnerabilidades, y se mantendrá la calidad del código continuamente para evitar la acumulación de deuda técnica y vulnerabilidades en el futuro.

Abstract

The objective is to reduce technical debt, vulnerabilities, failures, etc. to 0 in a specific application. To achieve this, a static code analysis tool such as SonarQube will be used to identify problem areas and define priorities. A team of developers will be assigned to correct the identified issues, using best software development practices and following the tool's recommendations. A schedule will be established, and periodic reviews will be conducted to assess project progress and adjust timelines as necessary. At the end of the project, the application will be launched free of technical debt and vulnerabilities, and code quality will be maintained continuously to prevent the accumulation of technical debt and vulnerabilities in the future.

1. Antecedentes o introducción

El tema de reducción de deuda técnica y vulnerabilidades en el desarrollo de software es la creación de la técnica de refactorización por parte de Martin Fowler en la década de 1990. La refactorización es una técnica que se utiliza para mejorar el diseño del código fuente, sin cambiar su comportamiento externo. Esta técnica ayuda a reducir la deuda técnica al mejorar la calidad del código y la facilidad de mantenimiento. Cabe mencionar que otro antecedente importante es la creciente conciencia en la industria del software sobre la importancia de la seguridad del software. A medida que las empresas dependen cada vez más de las aplicaciones de software para operar, los ataques de ciberseguridad se han convertido en una preocupación cada vez mayor. Los desarrolladores y los equipos de seguridad han comenzado a centrarse en la identificación y eliminación de vulnerabilidades en el software durante todo el ciclo de vida del desarrollo del software, desde el diseño hasta la implementación.

En general, los antecedentes del tema de reducción de deuda técnica y vulnerabilidades en el desarrollo de software demuestran la importancia de utilizar las mejores prácticas de desarrollo de software y de mantener la calidad del código a lo largo del tiempo para evitar problemas de seguridad y de calidad en el futuro.

2. Titulo

Análisis y Mejoramiento del sistema intranet de la empresa LADITEC

3. Autores

    - Diego Andre Aranda Reyes
    - Erick Mauricio Mamani Lima
    - Helbert Andres Condori Loayza
    - Jhonny Rivera Mendoza

4. Planteamiento del problema

    4.1. Problema

La empresa Laditec se ha visto afectada por un sistema de software ineficiente, que presenta vulnerabilidades de seguridad, líneas de código duplicadas y un mantenimiento inadecuado. Como resultado, la empresa ha experimentado una disminución en la productividad y calidad del servicio ofrecido. El problema radica en la falta de una estrategia de análisis y mejora del sistema de software, lo que ha llevado a una falta de control sobre el código y sus vulnerabilidades.

    4.2. Justificación

Es crucial para la empresa Laditec mejorar su sistema de software para aumentar su eficiencia y mejorar su calidad de servicio. Esto se logrará mediante la identificación y corrección de vulnerabilidades y errores en el código, así como la eliminación de líneas duplicadas y la mejora del mantenimiento del sistema.

    4.3. Alcance

Este proyecto se centrará en el análisis y mejora del sistema de software de la empresa Laditec a nivel general. Por lo que se hará uso de herramientas de análisis de código, como Sonarqube y Sonar Scanner. Además, incluirá la identificación de vulnerabilidades, errores y líneas duplicadas, así como la propuesta de soluciones para mejorar la calidad del código y la eficiencia del sistema.

5. Objetivos

    5.1. General

Reducir la deuda técnica, vulnerabilidades, y fallas en la aplicación a 0 mediante el uso de una herramienta de análisis estático de código, mejores prácticas de desarrollo de software y la asignación de un equipo de desarrolladores capacitados.

    5.2. Específicos

a. Identificar y priorizar las áreas problemáticas de la aplicación mediante el uso de una herramienta de análisis estático de código.
b. Corregir las áreas problemáticas de la aplicación utilizando las mejores prácticas de desarrollo de software y siguiendo las recomendaciones de la herramienta de análisis estático de código.
c. Evaluar periódicamente el progreso del proyecto mediante la realización de revisiones y ajustar los plazos según sea necesario.

6. Referentes teóricos

Diagramas de Casos de Uso

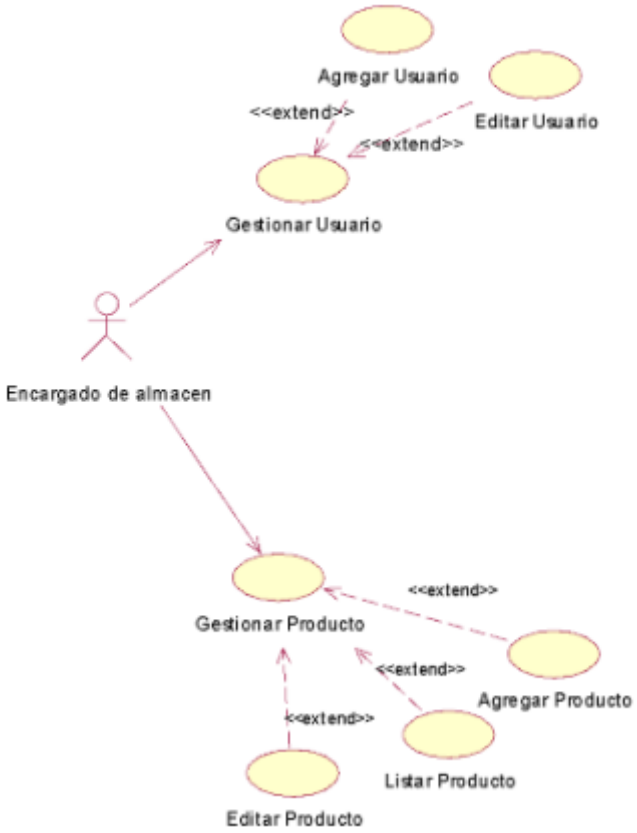


Diagrama de Arquitectura

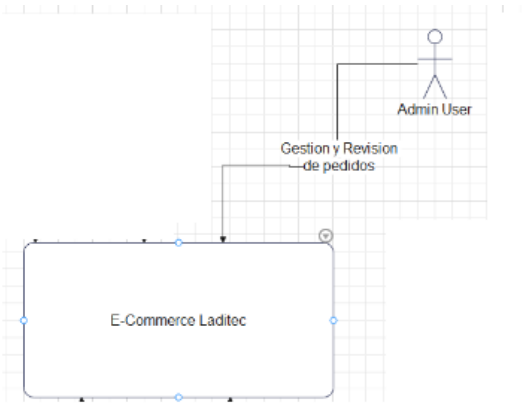


Diagrama de Componentes

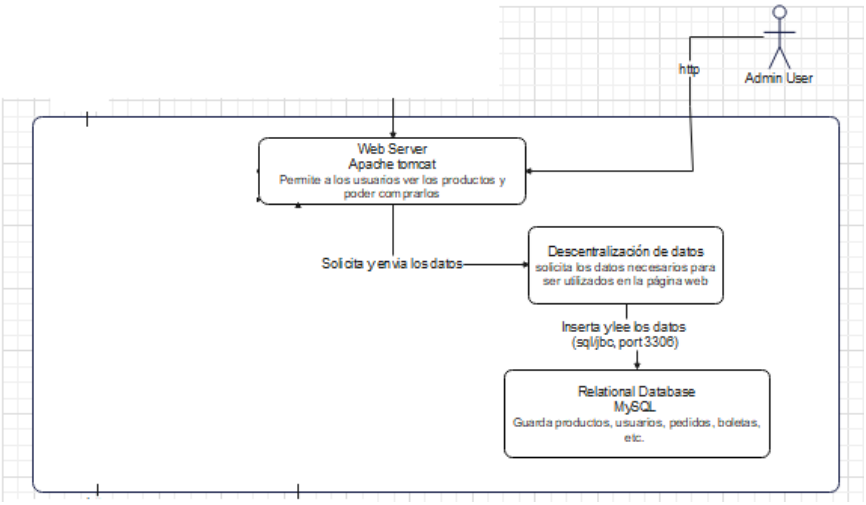
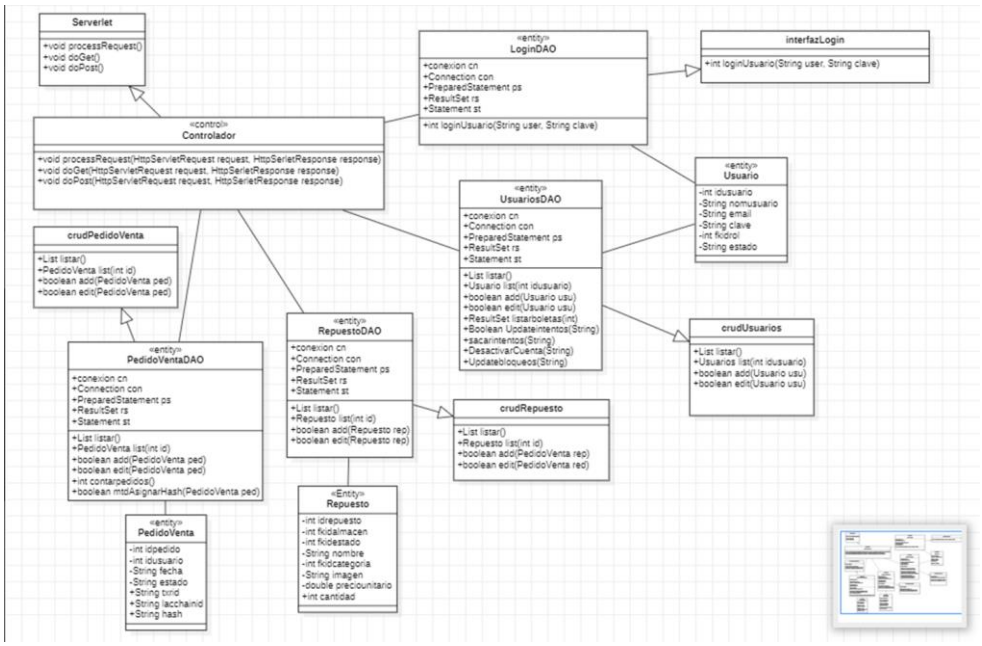
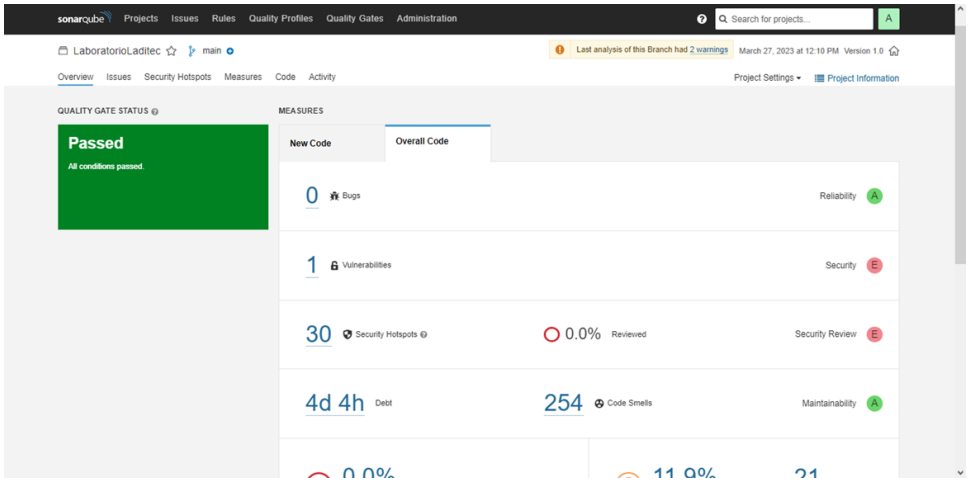


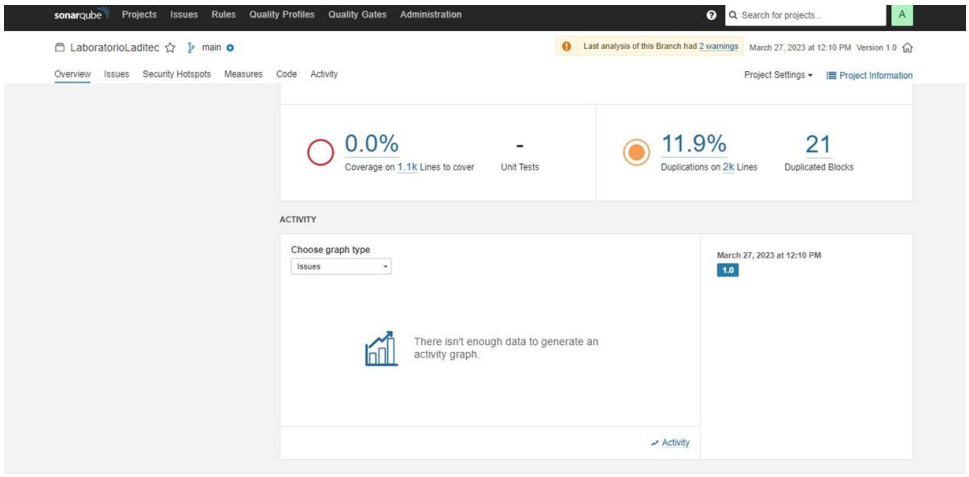
Diagrama de Clases



7. Desarrollo de la propuesta (Aquí va el análisis de su aplicación con SonarQube, para que les muestre todos los aspectos a mejorar de su aplicación)

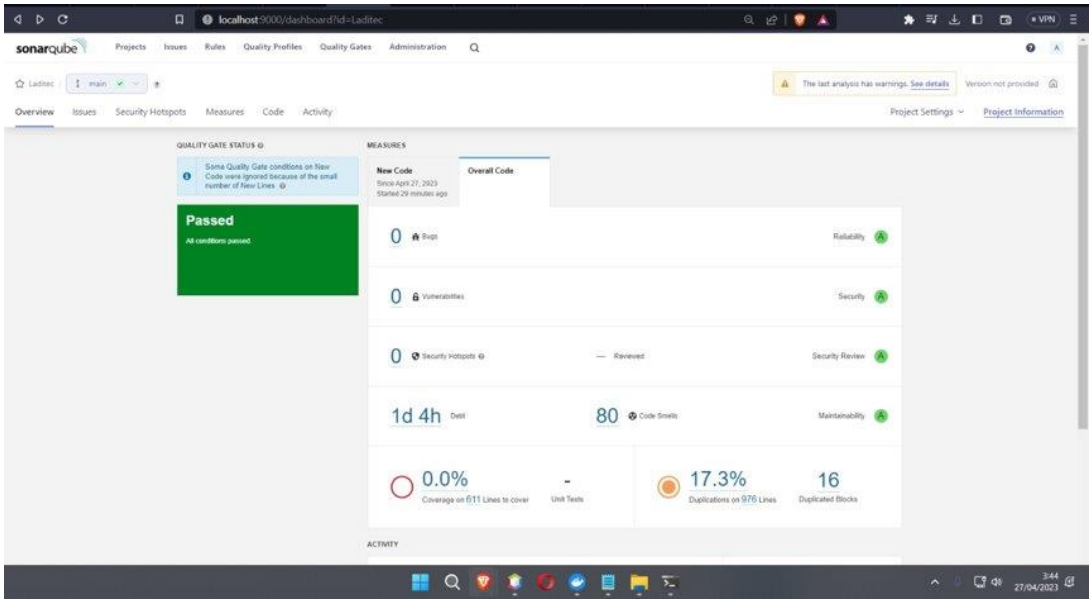
En la siguiente imagen podemos observar el análisis del código el día 27 de marzo del 2023, donde encontramos 0 bugs, 1 vulnerabilidad y 30 Security hostspots, además de que 254 code smells y más de 2k de líneas duplicadas.





El día 25 de abril se realizó el siguiente análisis y se encontró lo siguiente, 1 vulnerabilidades, 15 security hotstpots, 128 code smells y 918 líneas duplicadas

Por último el día 27 de abril se hizo un último análisis y se corrieron los últimos errores, donde el análisis salió: 0 bugs, 0 vulnerabilidades, 0 security hotspots y 80 code smells.



La prueba resulto Passed en 3 aspectos importantes, 0 bugs, 0 vulnerabilidades y 0 security hotspots.

7.1. Tecnología de información

a. Herramientas de análisis estático de código, como SonarQube, para identificar problemas de calidad y seguridad en el código fuente.
b. Se ha utilizado el Lenguaje de programación Java para desarrollar la aplicación.
c. Se utilizo Docker como virtualizador de aplicaciones.
d. Se ha utilizado la herramienta de control de versiones Git, para rastrear los cambios en el código y mantener un historial de versiones.
e. Plataformas de gestión de proyectos en este caso se ha utilizado Git para planificar y organizar las tareas del equipo de desarrollo.

7.2. Metodología, técnicas usadas

Se ha escaneado el código mediante Sonnarqube para poder realizar los cambios respectivos en el proyecto logrando así una mejora en la calidad del mismo para esto se va a realizar:

- Análisis estático de código
- Pruebas unitarias y de integración
- Pruebas del código

8. Cronograma (personas, tiempo, otros recursos) Basado en las observaciones que la herramienta SonarQube les informara sobre la aplicación, a fin de reducir la deuda técnica, vulnerabilidades, fallas, etc. a 0.

Semana 1-2:

- Revisión detallada de los informes de SonarQube para identificar las áreas problemáticas y definir las prioridades.

- Asignar un equipo de desarrolladores y un líder de proyecto para el esfuerzo de reducción de deuda técnica.
- Realizar reuniones con el equipo de desarrollo para establecer los objetivos y alcances del proyecto.

Semana 3-4:

- Comenzar a abordar las áreas problemáticas de mayor prioridad, utilizando las mejores prácticas de desarrollo de software y siguiendo las recomendaciones de SonarQube.
- Asignar a los desarrolladores responsables para corregir cada problema identificado.
- Programar pruebas manuales y automatizadas para asegurar la calidad del código corregido.

Semana 5-6:

- Continuar con la corrección de problemas identificados, revisando y probando continuamente el código para asegurar que se cumplan las mejores prácticas y recomendaciones.
- Asignar tareas adicionales a los desarrolladores según la capacidad disponible.

Semana 7-8:

- Comenzar a abordar las áreas de menor prioridad y problemas menos críticos.
- Establecer hitos y metas para evaluar el progreso del proyecto y ajustar los plazos según sea necesario.

Semana 9-10:

- Revisar el progreso del proyecto y realizar ajustes en consecuencia.
- Identificar y abordar cualquier obstáculo o problema que impida el progreso del proyecto.
- Continuar con el trabajo de corrección de problemas.

Semana 11-12:

- Realizar una revisión final para asegurarse de que no haya deuda técnica, vulnerabilidades, fallas, etc. pendientes.
- Realizar una auditoría de seguridad final para identificar cualquier problema de seguridad y corregirlos antes del lanzamiento de la aplicación.
- Lanzamiento de la aplicación libre de deuda técnica y vulnerabilidades.