



ANÁLISIS Y MEJORAMIENTO DEL SISTEMA INTRANET DE LA EMPRESA LADITEC

INTEGRANTES:

Condori Loayza, Helbert Andres

(2020067571)

Aranda Reyes, Diego Andre

(2019063855)

Mamani Lima, Erick Mauricio

(2020066321)

Rivera Mendoza, Jhonny

(2020067144)

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.

PLANTEAMIENTO DEL PROBLEMA

- 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.

PLANTEAMIENTO DEL PROBLEMA

- 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.

- 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.

OBJETIVOS

- **General**

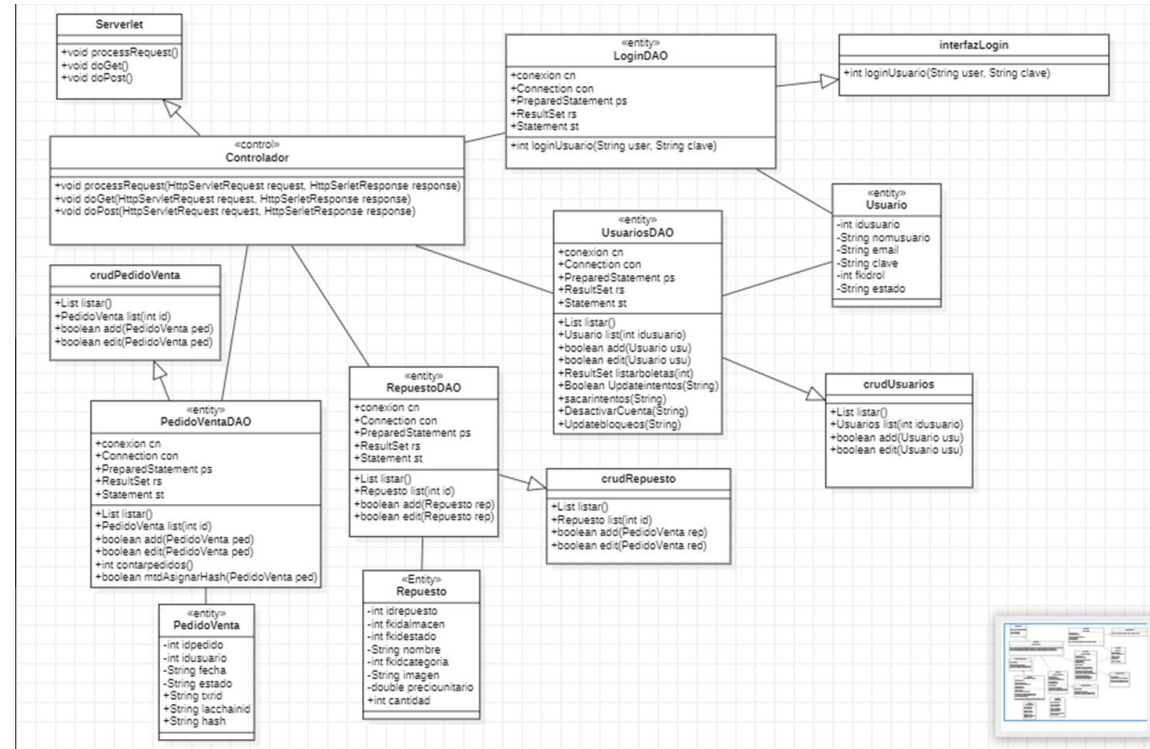
- Reducir la deuda técnica, vulnerabilidades, y fallas en la aplicación a 0 mediante el uso de una herramienta de análisis estático de código, mejores prácticas de desarrollo de software y la asignación de un equipo de desarrolladores capacitados.

- **Específicos**

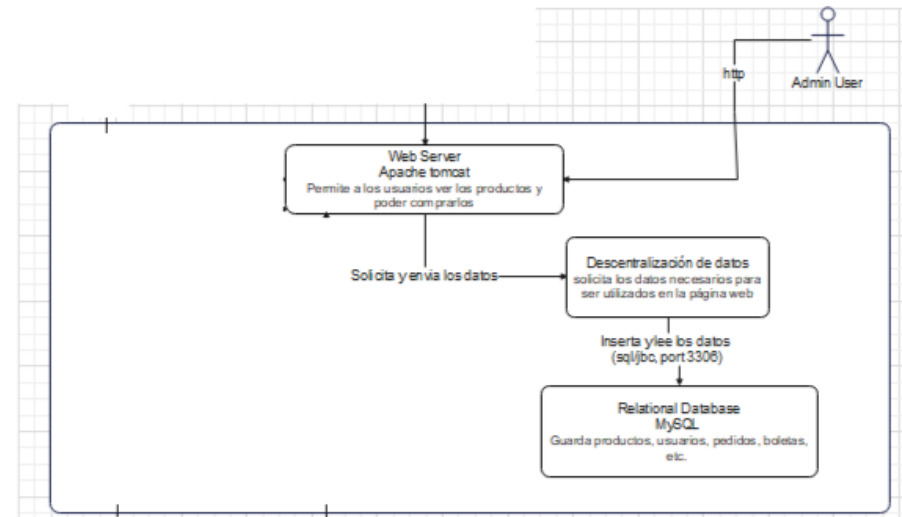
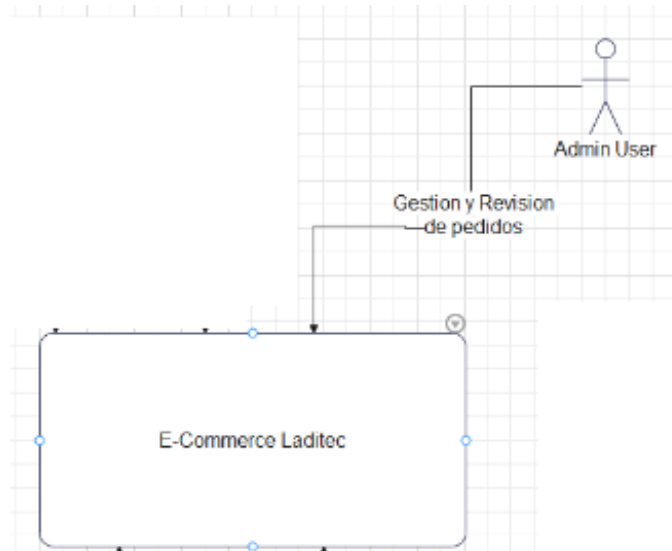
- Identificar y priorizar las áreas problemáticas de la aplicación mediante el uso de una herramienta de análisis estático de código.
- Corregir las áreas problemáticas de la aplicación utilizando las mejores prácticas de desarrollo de software y siguiendo las recomendaciones de la herramienta de análisis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

REFERENTES TEÓRICOS

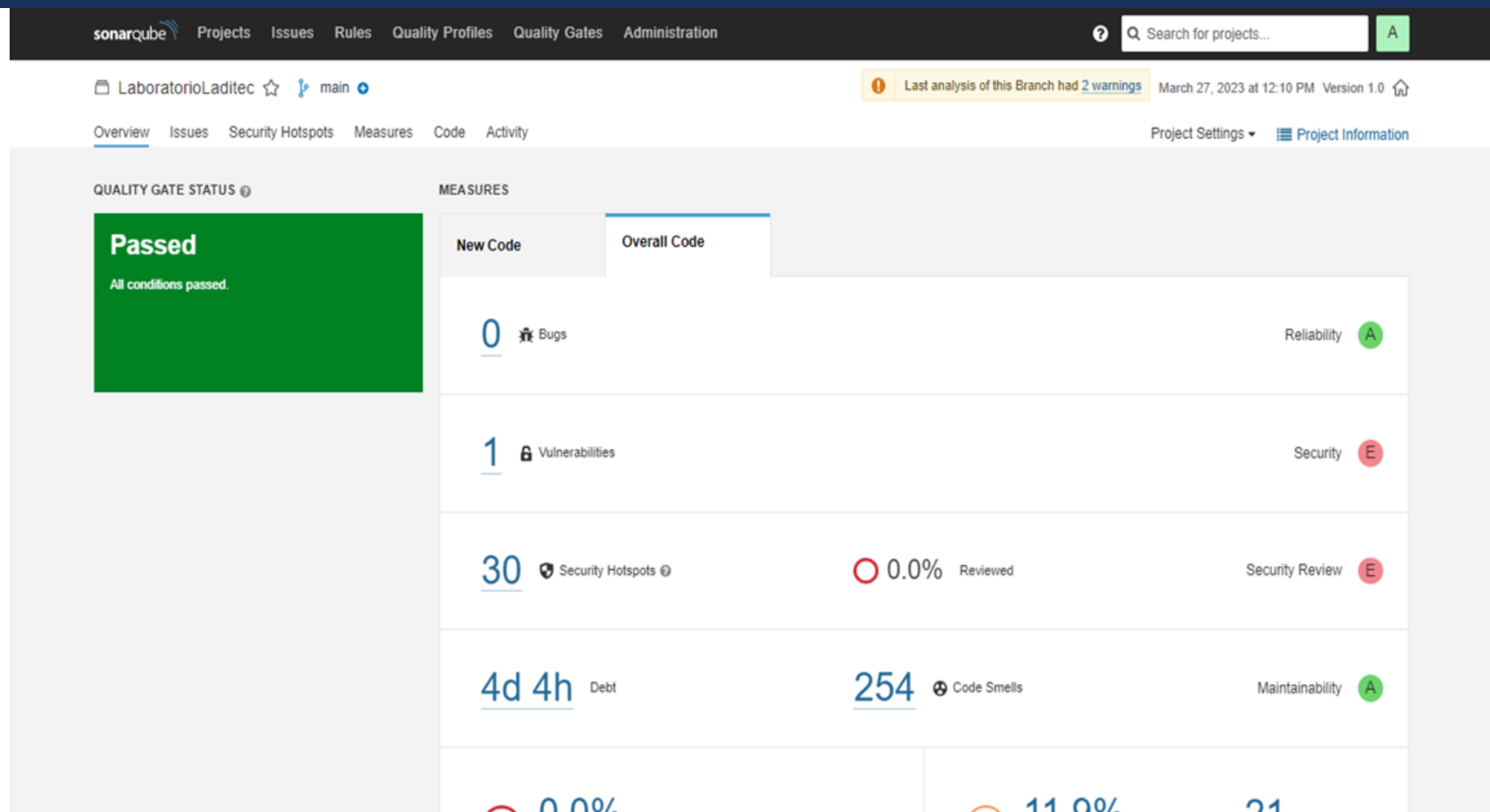
Diagramas de Casos de Uso, Diagrama de Clases,



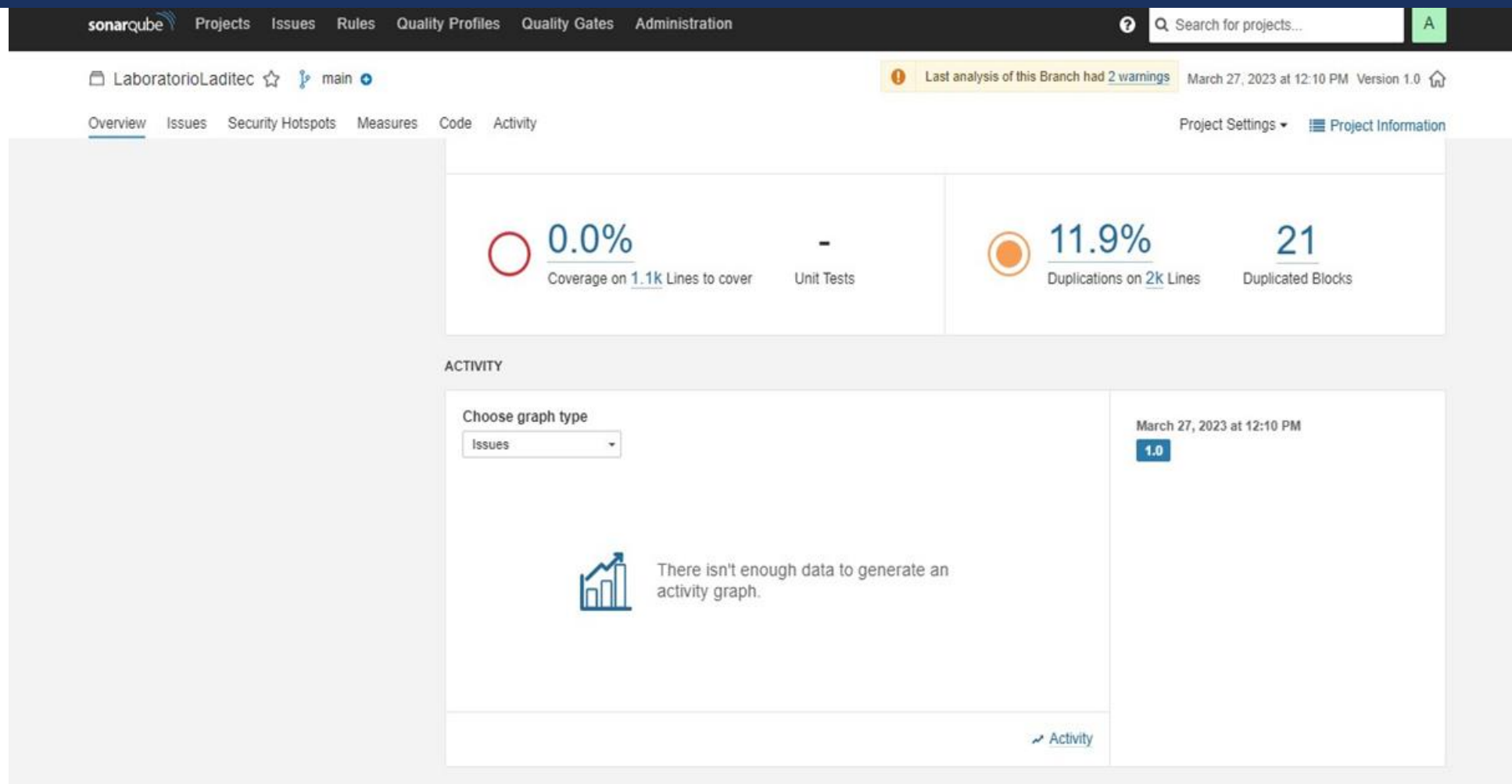
- Diagrama de Componentes y Arquitectura.



7. DESARROLLO DE LA PROPUESTA



7. DESARROLLO DE LA PROPUESTA




7. DESARROLLO DE LA PROPUESTA – DIA MARTES

1  Vulnerabilities

Security E

15  Security Hotspots 

 0.0% Reviewed

Security Review E

2d 2h Debt

128  Code Smells

Maintainability A

 0.0%
Coverage on 558 Lines to cover

-
Unit Tests

 16.0%
Duplications on 918 Lines

14
Duplicated Blocks

7. DESARROLLO DE LA PROPUESTA – DIA JUEVES

The screenshot displays the SonarQube web interface for a project named 'Laditec'. The browser address bar shows 'localhost:9000/dashboard?id=Laditec'. The SonarQube logo is in the top left, and navigation tabs for 'Projects', 'Issues', 'Rules', 'Quality Profiles', 'Quality Gates', and 'Administration' are at the top. The 'main' branch is selected. A yellow warning banner states: 'The last analysis has warnings. See details'. Below this, the 'Overview' tab is active, showing a 'Passed' quality gate status with the message 'All conditions passed.' and a note: 'Some Quality Gate conditions on New Code were ignored because of the small number of New Lines'. The 'MEASURES' section is divided into 'New Code' (since April 27, 2023) and 'Overall Code'. Metrics include: 0 Bugs (Reliability A), 0 Vulnerabilities (Security A), 0 Security Hotspots (Security Review A), 1d 4h Debt (Maintainability A), 0.0% Coverage on 611 Lines to cover (Unit Tests -), 17.3% Duplications on 976 Lines (Duplicated Blocks 16), and 80 Code Smells. An 'ACTIVITY' section is at the bottom. The Windows taskbar at the bottom shows the time as 3:44.

localhost:9000/dashboard?id=Laditec

sonarqube

Projects Issues Rules Quality Profiles Quality Gates Administration

Laditec / main

The last analysis has warnings. See details

Version not provided

Overview Issues Security Hotspots Measures Code Activity

Project Settings Project Information

QUALITY GATE STATUS

Some Quality Gate conditions on New Code were ignored because of the small number of New Lines

Passed

All conditions passed.

MEASURES

New Code

Since April 27, 2023

Started 29 minutes ago

Overall Code

0 Bugs

Reliability A

0 Vulnerabilities

Security A

0 Security Hotspots

Reviewed

Security Review A

1d 4h Debt

80 Code Smells

Maintainability A

0.0% Coverage on 611 Lines to cover

Unit Tests -

17.3% Duplications on 976 Lines

Duplicated Blocks 16

ACTIVITY

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

- 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.



GRACIAS