

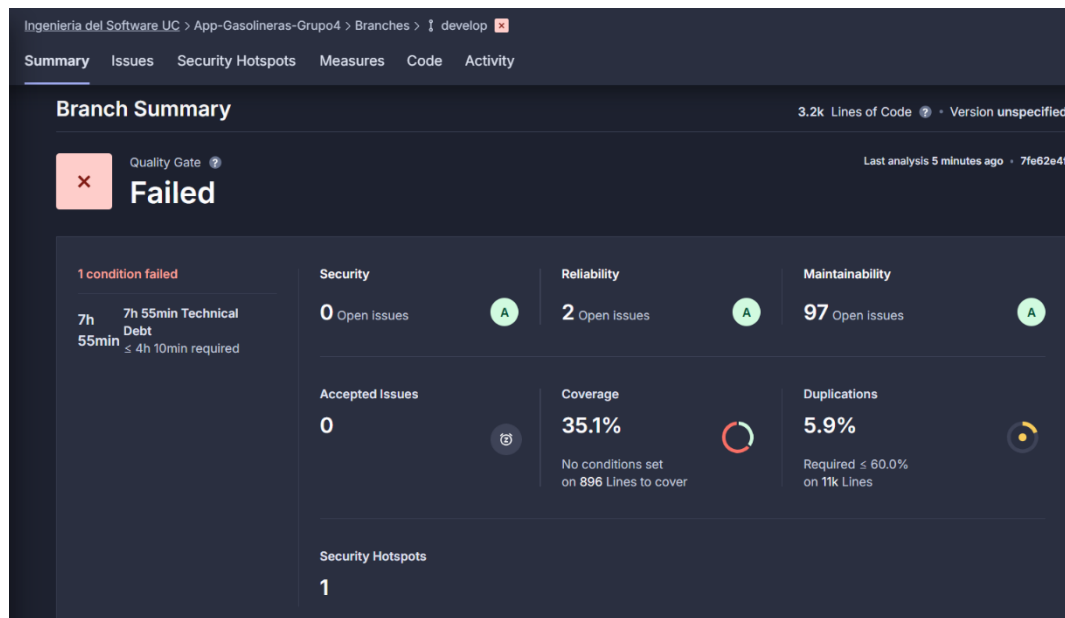
Informe de Calidad

Segundo Sprint, Grupo 4

Fecha de realización: 31 de octubre de 2024

Hash del commit: 7fe62e4f

Captura del Resumen del Análisis:

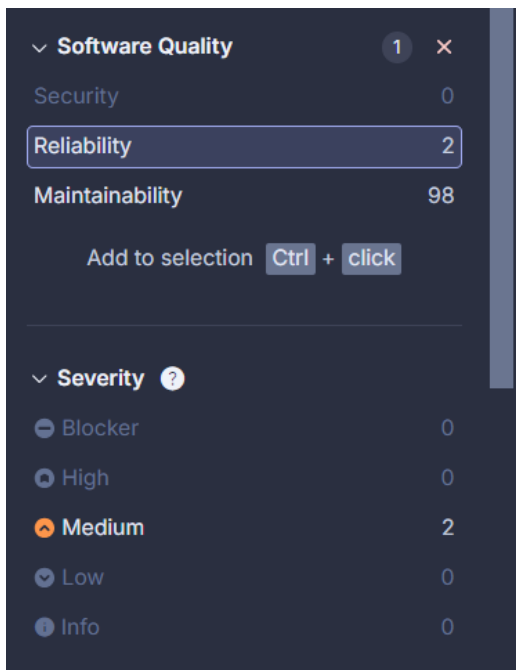


El quality gate, en este segundo análisis, falló por dos razones principales: la primera, fue porque la deuda técnica superó las cuatro horas con diez minutos. La segunda razón, es porque al realizar la prueba con SonarQube dentro de la aplicación, solamente el 35.1% de líneas de código están siendo probadas. Esto indica que hay un 64.9% de líneas de código que no han sido probadas y esto permite que existan partes del algoritmo que pueden contener bugs no detectados al no haber sido analizados.

Además, un detalle a tener muy presente es que se está detectando una posible vulnerabilidad en un security hotspots o punto de acceso de seguridad. Los otros factores de calidad tienen los resultados dentro de un rango permitido.

Severidad de los issues de cada elemento principal:

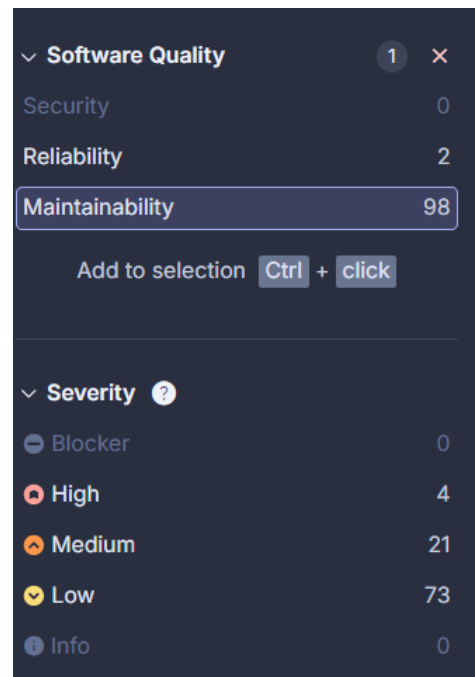
Reliability:



Software Quality	
Security	0
Reliability	2
Maintainability	98
Add to selection Ctrl + click	
Severity	
Blocker	0
High	0
Medium	2
Low	0
Info	0

En esta captura de pantalla, se aprecia que existen dos errores con severidad media de Reliability pero en esta ocasión, se puede dar prioridad a otros aspectos de mejora ya que la prueba de reliability en el quality gate, fue aprobada.

Maintainability:



Software Quality	
Security	0
Reliability	2
Maintainability	98
Add to selection Ctrl + click	
Severity	
Blocker	0
High	4
Medium	21
Low	73
Info	0

En esta imagen, se puede observar que hay varios problemas de distinta severidad, existiendo cuatro con severidad alta de mantenibilidad y es urgente resolverlos

Security:

Debido a que no existen problemas de seguridad, no se anexa una captura relacionada con este software quality o información adicional relacionada.

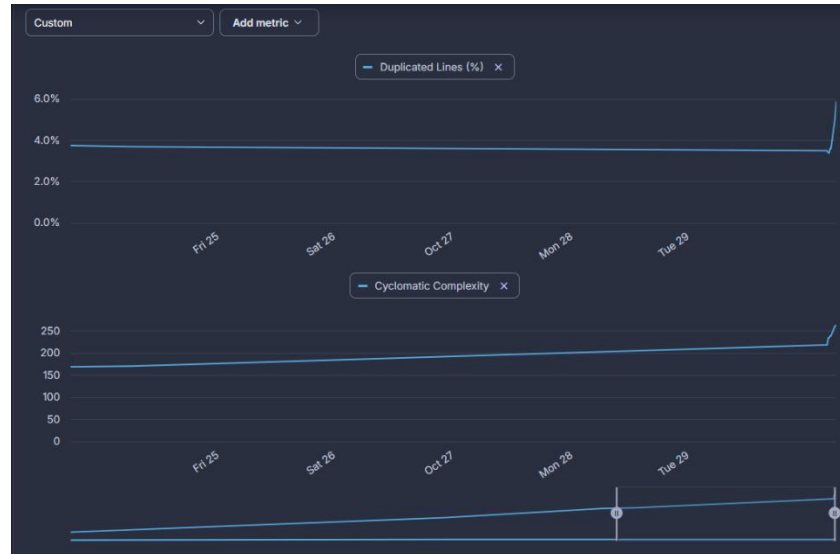
Gráfica de número de issues de cada uno de los tres elementos:

Se realiza desde el 24 de octubre, ya que fue la fecha en que se realizó el último análisis de calidad con SonarQube para el informe de calidad.



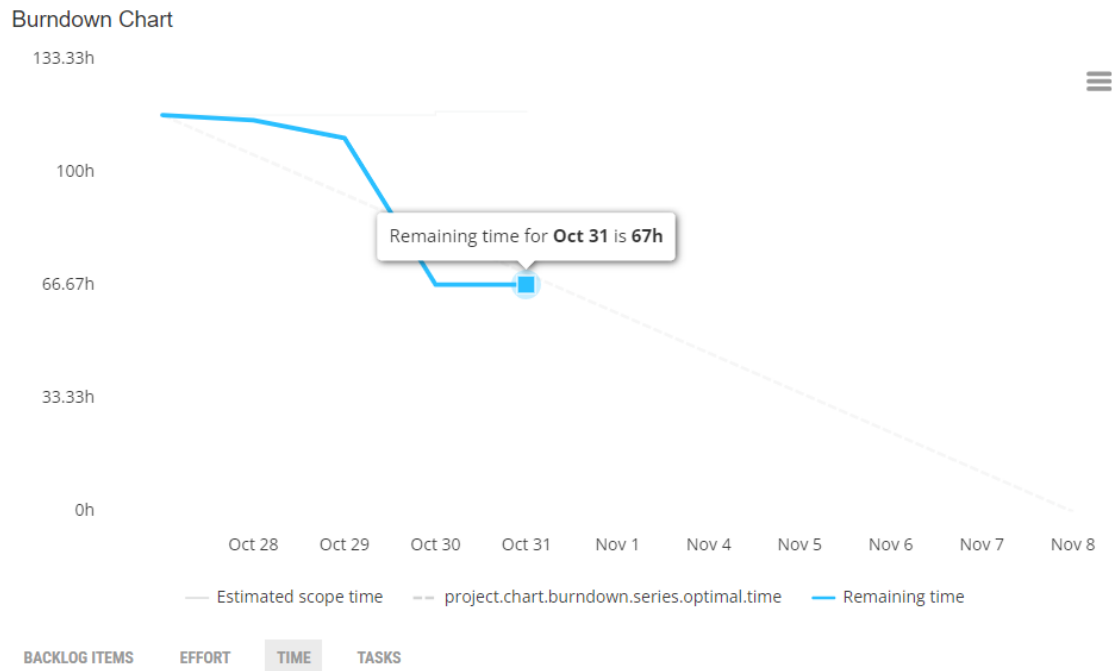
En esta gráfica, la mantenibilidad es el área más crítica, con un total de 98 incidencias. La fiabilidad o Reliability, también presenta problemas, pero únicamente con 2 incidencias.

Porcentaje de código repetido y complejidad ciclomática:



En esta gráfica, se puede evidenciar que la duplicidad de los algoritmos se mantuvo constante hasta el 30 de octubre y desde entonces, han aumentado. Sobre la complejidad ciclomática, ha ido en aumento, pero se dispara notoriamente el 30 de octubre. El aumento de estas dos variables, afectan negativamente la mantenibilidad del código y también demuestra que ese día se realizaron merge con algoritmos que se deben mejorar.

Burndown Chart:



Plan de Acción

Al analizar todas las gráficas, nos podemos dar cuenta que existen varios problemas por solucionar. En este sprint, según el burndown chart y el tiempo disponible de trabajo, la prioridad será resolver la incidencia de Security Hotspots y reducir la deuda técnica para poder cumplir con el Quality Gate.

1. Incidencias a Solventar

Incidencia	Prioridad	Justificación
Security Hotspots	Alta	Este tipo de error es crítico porque puede causar vulnerabilidades a la seguridad de la aplicación y que personas no autorizadas accedan a datos confidenciales. Hay que revisarlo y solucionarlo lo más pronto posible.
Reducción de deuda técnica en paymentHistory, registerDiscount y discountList.	Alta	Reducir la deuda técnica es prioritario porque mejora la mantenibilidad del código y evita que problemas menores se acumulen, lo que puede llevar a complicaciones mayores en el futuro.

Define a constant instead of duplicating this literal "Error" 11 times.	Media	Duplicar strings es una mala práctica, ya que realizar un cambio en solo uno, provocaría que se tenga que cambiar en todas las otras apariciones, provocando errores de mantenibilidad a futuro.
---	-------	--

Actividades agregadas a ScrumDesk

Para la brevedad y concisión en el informe, la lista se realizó en forma de tabla.

Incidencia	Descripción
Eliminar toda la deuda técnica en RegisterDiscount	En las clases RegisterDiscountView y RegisterDiscountPresenter se han de eliminar los errores que causan deuda técnica: <ul style="list-style-type: none"> • Renombrado de variables que no cumplen las reglas • Refactorización de métodos muy complejos • Eliminación de imports sin uso
Resolver code smell	En la clase RegisterDiscountPresenter se encuentra un code smell por duplicidad de una constante de tipo String "Error", esta debe ser sustituida de alguna forma para evitar el code smell.
Resolver security hotspot en Utils	Dentro de la clase Utils se ha de resolver el hotspot de seguridad que se encuentra en el metodo obtenerRotulosUnicos en el cual se hace un "e.printStackTrace()".
Code smell fix	Arreglar un método duplicado en RegisterPaymentView, el método relacionado con el lanzado del historial de pagos esta duplicado.
Code smell en register discount	La db es una variable publica y deberia ser o bien privada o estatica y final.
Resolver "Extract this nested code block into a method" en DiscountArrayAdapter	Arreglar problema de Code Smell en clase DiscountArrayAdapter, en el método getView extraer a métodos privados las asignaciones de los textView.
Resolver "Extract this nested code block into a method." en PagosArrayAdapter	Arreglar problema de Code Smell en clase PagosArrayAdapter, en el método getView

	extraer a métodos privados las asignaciones de los textView.
Resolver "Extract this nested code block into a method." en CombustibleArrayAdapter	Arreglar problema de Code Smell en clase CombustibleArrayAdapter, en el método getView extraer a métodos privados las asignaciones de los textView.
Resolver "Extract this nested code block into a method." en GasolinerasArrayAdapter	Arreglar problema de Code Smell en clase GasolinerasArrayAdapter, en el método getView extraer a métodos privados las asignaciones de los textView.
Remove code smell private field not used	Remove a private field in the HistoryPaymentUITest that it is not used (DecorView)