



SINGLE LADY

Report Finale

Data di rilascio: 14/07/2023

Metodi e Modelli per la qualità del software a.a. 2022-2023

Realizzato da

Stallone Claudia

matr.724680

Corso di Laurea ITPS

email c.stallone4@studenti.uniba.it

Indice

1. DESCRIZIONE DEL SISTEMA SOFTWARE.....	3
2. STRUMENTI DI LAVORO	4
3. OBIETTIVI(TARGET)	6
4. ANALISI PRELIMINARE SONARCLOUD-FORTIFY.....	7
5. ANALISI DI QUALITA'(INTRODUZIONE).....	10
5.1 Analisi1 SonarCloud.....	11
5.2 Analisi2 SonarCloud.....	12
5.3 Analisi3 SonarCloud.....	18
5.3 Analisi4 SonarCloud.....	21
5.3 Analisi5 SonarCloud.....	23
6. SICUREZZA.....	26
7. AFFIDABILITA'.....	27
8.MANUTENIBILITA'.....	28
9.CONCLUSIONE.....	29

DESCRIZIONE DEL SISTEMA SOFTWARE

Nome del Progetto analizzato: Laszip4j

Descrizione: LASzip è una libreria trasferita su java.

È una porta Java delle librerie LASzip e LAsTools di Martin Isenburg, il trasferimento di tale libreria su Java non è completo, poiché molte classi sono solo stub e non sono state trasferite.

Il driver principale era quello di avere un'implementazione Java dell'utilità laszip, cercando di essere in grado di decomprimere i file LAZ per il Canton Zurigo in Svizzera, il quale funziona correttamente.

L'utilizzo di tale libreria è lo stesso dell'utility laszip nativa, con l'unica differenza che viene invocato come jar eseguibile.

Questa libreria, oltre a LASzip e LAsTools, fornisce anche classi di convenienza per la lettura dei punti LAS.

Dalla versione 0.13, laszip4j supporta anche la scrittura di file las;

nel caso più semplice il file jar eseguibile può essere utilizzato per leggere dati compressi da un file laz e scriverli in un file las corrispondente.

Link del progetto GitHub:

<https://github.com/mreutegg/laszip4j>

STRUMENTI DI LAVORO

Gli strumenti di lavoro utilizzati durante il processo di analisi sono:

- **Maven:** Trattasi di un progetto open source, strumento per la gestione basato su java e build automation. Esso è ospitato da Apache Software Foundation, usa Project Object Model (POM) ossia un file XML che descrive le dipendenze fra progetto e librerie. Effettua automaticamente il download di librerie Java e plug-in da varie repository.
- **TortoiseSVN:** Trattasi di un client grafico di Subversion, programmato per funzionare come un'estensione di Microsoft Windows (programma gratuito). Esso può integrarsi con Microsoft Visual Studio utilizzando i plugin di terze parti come VisualSVN, VsTortoise e AnkhSVN.
- **Fortify:** (Utilizzato solo inizialmente prima dell'interruzione dell'uso della piattaforma)
Trattasi di un Software gratuito, il quale identifica le vulnerabilità di sicurezza nel codice sorgente nelle prime fasi del ciclo di vita di sviluppo del software e fornisce le migliori pratiche per cui gli sviluppatori possono codificare in modo più sicuro. Individua in maniera tempestiva i problemi di sicurezza.
- **Redmine:** (Utilizzato solo inizialmente prima dell'interruzione dell'uso della piattaforma)
Trattasi di un Software gratuito e open source, che ha come funzione la pianificazione di progetti e il tracciamento delle segnalazioni di bug.
Caratteristiche principali:
 - supporto per più progetti;
 - controllo degli accessi flessibile basato su ruoli;
 - sistema di localizzazione dei problemi;
 - diagramma e calendario gantt;
 - gestione di notizie;
 - documenti e file;
 - feed e notifiche e-mail;
 - forum di progetto;
 - tracciamento del tempo.

Redmine ha un'area personalizzata per problemi: time-entry, progetti e utenti; ed è un'integrazione SCM (ossia SVN, Git, CVS).

- **Microsoft Excel:** Trattasi di un programma per la produzione e gestione di fogli elettronici.
Esso permette di:
 - effettuare calcoli;
 - analizzare;
 - creare tabelle;
 - creare grafici ed elaborare i dati.

Fa parte della famiglia di Microsoft office.
- **SonarCloud:** Trattasi di un servizio di analisi del codice il quale si basa sul cloud per rilevare problemi di codifica in diversi linguaggi di programmazione. Inserendo il codice esso riesce a verificarlo controllando la manutenibilità, affidabilità e problemi di sicurezza. Aiuta a fornire un codice pulito e che sia di alta qualità.
- **Visual Studio Code:** Trattasi di un Software libero e gratuito per uso personale e non, esso è un'editor di codice sorgente sviluppato da Microsoft, supporta debugging, un controllo per GitHub integrato ecc.; si basa su Electron, un framework con cui è possibile sviluppare applicazioni Node.js. La sua caratteristica principale è il supporto a più linguaggi di programmazione ed è multiplatforme.

OBIETTIVI(TARGET)

SONARCLOUD

DATI INIZIALI:

Bug	Vulnerabilità	Hotspots Rewied	Code Smells	Technical Debt
E-67	A-0	A-100%	A-3,9K	39d

OBIETTIVO FINALE:

Bug	Vulnerabilità	Hotspots Rewied	Code Smells	Technical Debt
A-0	A-0	A-0	A-232	5 d

FORTIFY

OBIETTIVO INIZIALE:

Per tale piattaforma non posseggo i dati iniziali poiché fuori uso.

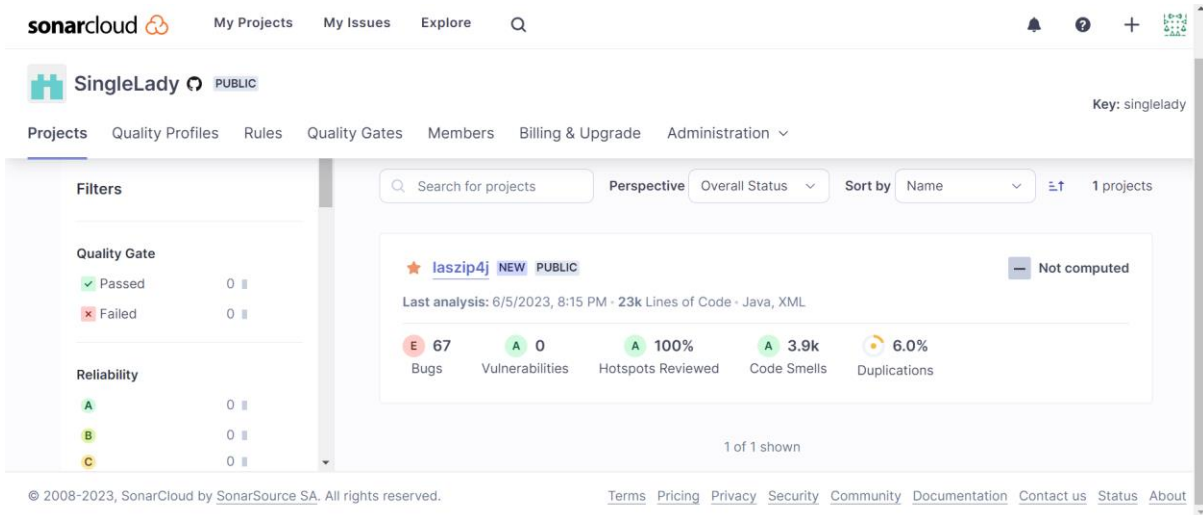
OBIETTIVO FINALE:

Critical	High	Medium	Low
0	100	0	244

ANALISI PRELIMINARE SONARCLOUD-FORTIFY

Link al progetto SonarCloud (dove poter visualizzare l'Analisi1):
https://sonarcloud.io/summary/overall?id=ClaudiaStallone_laszip4

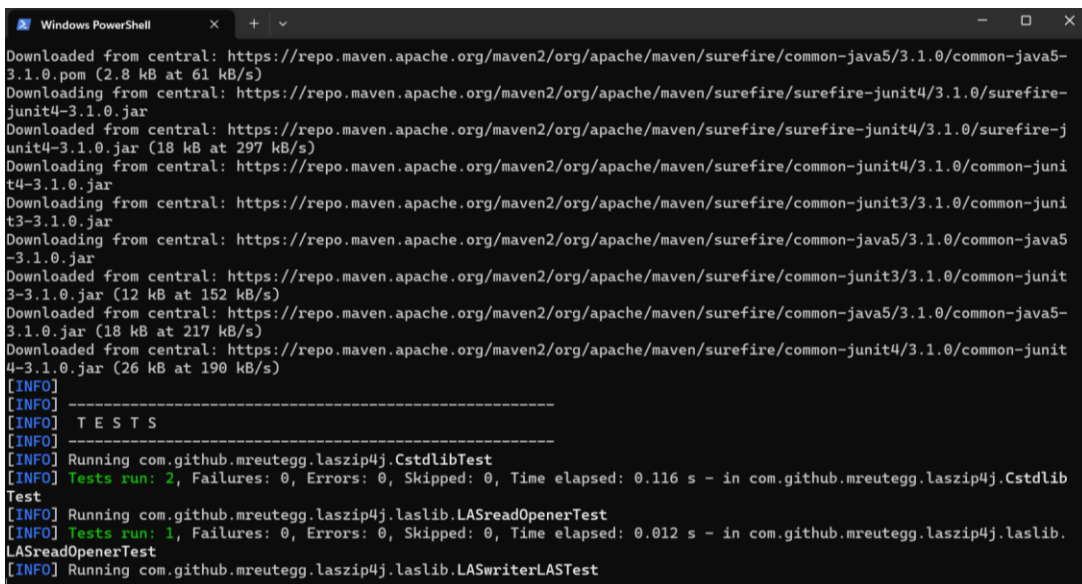
Screenshot SonarCloud (interfaccia principale con i dati del progetto) :



The screenshot shows the SonarCloud web interface for a project named 'SingleLady'. The interface includes a navigation bar with 'My Projects', 'My Issues', 'Explore', and a search icon. Below the navigation bar, there are tabs for 'Projects', 'Quality Profiles', 'Rules', 'Quality Gates', 'Members', 'Billing & Upgrade', and 'Administration'. The main content area displays project details for 'SingleLady', including a search bar, filters, and a summary of project metrics. The metrics section shows: 67 Bugs (E), 0 Vulnerabilities (A), 100% Hotspots Reviewed (A), 3.9k Code Smells (A), and 6.0% Duplications (A). The project status is 'Not computed'. The footer contains copyright information and links to Terms, Pricing, Privacy, Security, Community, Documentation, Contact us, Status, and About.

Screenshot Test Maven :

Test:



```
Windows PowerShell
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/surefire/common-java5/3.1.0/common-java5-3.1.0.pom (2.8 kB at 61 kB/s)
Downloading from central: https://repo.maven.apache.org/maven2/org/apache/maven/surefire/surefire-junit4/3.1.0/surefire-junit4-3.1.0.jar
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/surefire/surefire-junit4/3.1.0/surefire-junit4-3.1.0.jar (18 kB at 297 kB/s)
Downloading from central: https://repo.maven.apache.org/maven2/org/apache/maven/surefire/common-junit4/3.1.0/common-junit4-3.1.0.jar
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/surefire/common-junit4/3.1.0/common-junit4-3.1.0.jar
Downloading from central: https://repo.maven.apache.org/maven2/org/apache/maven/surefire/common-junit3/3.1.0/common-junit3-3.1.0.jar
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/surefire/common-junit3/3.1.0/common-junit3-3.1.0.jar
Downloading from central: https://repo.maven.apache.org/maven2/org/apache/maven/surefire/common-java5/3.1.0/common-java5-3.1.0.jar
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/surefire/common-java5/3.1.0/common-java5-3.1.0.jar (18 kB at 217 kB/s)
Downloading from central: https://repo.maven.apache.org/maven2/org/apache/maven/surefire/common-junit4/3.1.0/common-junit4-3.1.0.jar (26 kB at 190 kB/s)
[INFO]
[INFO] -----
[INFO] T E S T S
[INFO] -----
[INFO] Running com.github.mreutegg.laszip4j.CstdLibTest
[INFO] Tests run: 2, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 0.116 s - in com.github.mreutegg.laszip4j.CstdLibTest
[INFO] Running com.github.mreutegg.laszip4j.laslib.LASreadOpenerTest
[INFO] Tests run: 1, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 0.012 s - in com.github.mreutegg.laszip4j.laslib.LASreadOpenerTest
[INFO] Running com.github.mreutegg.laszip4j.laslib.LASwriterLASTest
```

```
Windows PowerShell

at org.junit.runners.BlockJUnit4ClassRunner.runChild(BlockJUnit4ClassRunner.java:63)
at org.junit.runners.ParentRunner$4.run(ParentRunner.java:331)
at org.junit.runners.ParentRunner$1.schedule(ParentRunner.java:79)
at org.junit.runners.ParentRunner.runChildren(ParentRunner.java:329)
at org.junit.runners.ParentRunner.access$100(ParentRunner.java:66)
at org.junit.runners.ParentRunner$2.evaluate(ParentRunner.java:293)
at org.junit.runners.ParentRunner$3.evaluate(ParentRunner.java:306)
at org.junit.runners.ParentRunner.run(ParentRunner.java:413)
at org.apache.maven.surefire.junit4.JUnit4Provider.execute(JUnit4Provider.java:316)
at org.apache.maven.surefire.junit4.JUnit4Provider.executeWithRerun(JUnit4Provider.java:240)
at org.apache.maven.surefire.junit4.JUnit4Provider.executeTestSet(JUnit4Provider.java:214)
at org.apache.maven.surefire.junit4.JUnit4Provider.invoke(JUnit4Provider.java:155)
at org.apache.maven.surefire.booter.ForkedBooter.runSuitesInProcess(ForkedBooter.java:385)
at org.apache.maven.surefire.booter.ForkedBooter.execute(ForkedBooter.java:162)
at org.apache.maven.surefire.booter.ForkedBooter.run(ForkedBooter.java:507)
at org.apache.maven.surefire.booter.ForkedBooter.main(ForkedBooter.java:495)

[INFO] Running com.github.mreutegg.laszip4j.LASWriterTest
[INFO] Tests run: 1, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 0.384 s - in com.github.mreutegg.laszip4j.LASWriterTest
[INFO]
[INFO] Results:
[INFO]
[ERROR] Failures:
[ERROR]   LASReaderTest.readExtraBytesShort:435 expected:<[0.80, 1.12, 1.00, 1.28, 1.52]> but was:<[0.80, 1.12, 1.00, 1.28, 1.52]>
[INFO]
[ERROR] Tests run: 16, Failures: 1, Errors: 0, Skipped: 0
[INFO]
[INFO] -----
```

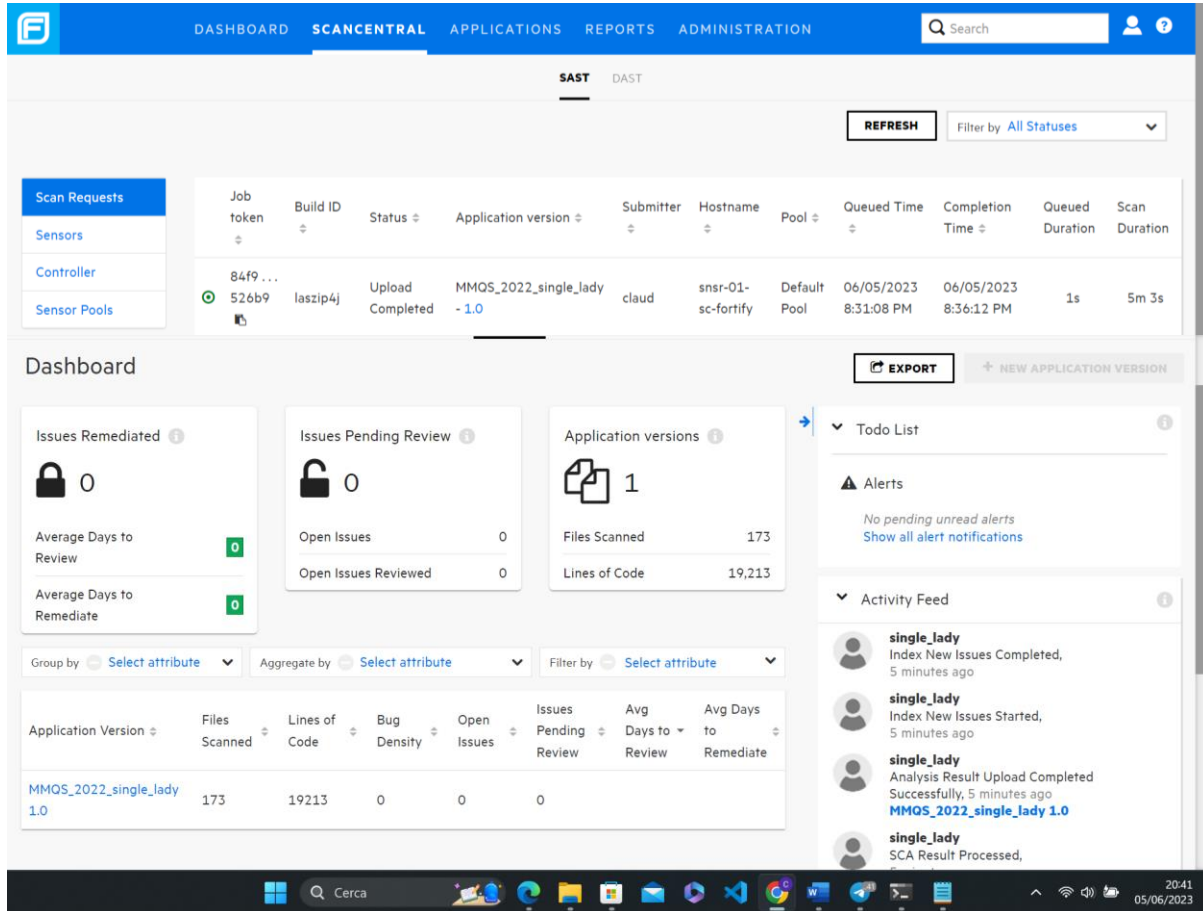
Screenshot Test Maven :

Esito positivo (**BUILD SUCCESS**):

```
Windows PowerShell

[INFO] -----[ jar ]-----
[INFO]
[INFO] --- resources:3.3.0:resources (default-resources) @ laszip4j ---
[INFO] skip non existing resourceDirectory C:\Users\claud\OneDrive\Desktop\esami ITPS\modelli e metodi\progetto\laszip4j\src\main\resources
[INFO]
[INFO] --- compiler:3.11.0:compile (default-compile) @ laszip4j ---
[INFO] Changes detected - recompiling the module! :source
[INFO] Compiling 172 source files with javac [debug target 1.8] to target\classes
[WARNING] bootstrap class path not set in conjunction with -source 8
[INFO] /C:/Users/claud/OneDrive/Desktop/esami ITPS/modelli e metodi/progetto/laszip4j/src/main/java/com/github/mreutegg/laszip4j/laszip/LASwritePoint.java: C:\Users\claud\OneDrive\Desktop\esami ITPS\modelli e metodi\progetto\laszip4j\src\main\java\com\github\mreutegg\laszip4j\laszip\LASwritePoint.java uses unchecked or unsafe operations.
[INFO] /C:/Users/claud/OneDrive/Desktop/esami ITPS/modelli e metodi/progetto/laszip4j/src/main/java/com/github/mreutegg/laszip4j/laszip/LASwritePoint.java: Recompile with -Xlint:unchecked for details.
[INFO]
[INFO] BUILD SUCCESS
[INFO]
[INFO] Total time: 5.534 s
[INFO] Finished at: 2023-06-05T20:21:11+02:00
[INFO]
[WARNING]
[WARNING] Plugin validation issues were detected in 1 plugin(s)
[WARNING]
[WARNING] * org.apache.maven.plugins:maven-resources-plugin:3.3.0
[WARNING]
[WARNING] For more or less details, use 'maven.plugin.validation' property with one of the values (case insensitive): [BRIEF, DEFAULT, VERBOSE]
[WARNING]
PS C:\Users\claud\OneDrive\Desktop\esami ITPS\modelli e metodi\progetto\laszip4j>
```


Screenshot Fortify :



The screenshot displays the Fortify SCANCENTRAL interface. The top navigation bar includes links for DASHBOARD, SCANCENTRAL, APPLICATIONS, REPORTS, and ADMINISTRATION. The main content area is divided into several sections:

- Scan Requests:** A table showing scan results. The first entry is for 'MMQS_2022_single_lady - 1.0', which is in a 'Completed' status. The table columns include Job token, Build ID, Status, Application version, Submitter, Hostname, Pool, Queued Time, Completion Time, Queued Duration, and Scan Duration.
- Dashboard:** A summary section with three main cards:
 - Issues Remediated:** Shows 0 issues.
 - Issues Pending Review:** Shows 0 open issues and 0 open issues reviewed.
 - Application versions:** Shows 1 version scanned, 173 files scanned, and 19,213 lines of code.
- Activity Feed:** A list of recent events, including 'Index New Issues Completed', 'Index New Issues Started', 'Analysis Result Upload Completed Successfully', and 'SCA Result Processed', all associated with the 'single_lady' user.

The bottom of the screenshot shows a Windows taskbar with various application icons and the system clock indicating 20:41 on 05/06/2023.

ANALISI DI QUALITÀ (INTRODUZIONE)

L'analisi di qualità è una tecnica importante utilizzata nella fase di sviluppo del software che vede la valutazione del codice sorgente di quest'ultimo; essa viene eseguita per identificare delle anomalie o dei problemi all'interno del software che possono provocare ed influire negativamente sulla qualità.

Generalmente si utilizzano strumenti automatici in grado di poter rilevare problemi di vario tipo, come ad esempio cicli di controllo non definiti correttamente; errori di formattazione del codice; violazione delle pratiche di programmazione; immoderata complessità del codice; ecc...

Essa permette di correggere ed individuare problematiche all'interno del codice, prima di rilascio, in modalità immediata; questo quindi tende a garantire un'alta qualità del software ed un corretto funzionamento.

L'analisi di qualità può avvenire in due modi:

- Manualmente
- Automaticamente

L'analisi di qualità effettuata all'interno del progetto è avvenuta tramite l'utilizzo di SonarCloud.



Esso è un servizio di analisi del codice, open source, il quale si basa sul cloud per rilevare problemi di codifica in diversi linguaggi di programmazione, effettuando un'ispezione.

Una volta inserito il codice sorgente, esso lo verifica controllandone la manutenibilità, l'affidabilità e i problemi di sicurezza.

Inoltre aiuta a fornire un codice pulito di alta qualità e permette di eseguire in maniera automatica le revisioni con le analisi statiche del codice sorgente.

Analisi1 SonarCloud

La prima analisi eseguita in data 15/06/2023 evidenzia le seguenti problematiche:

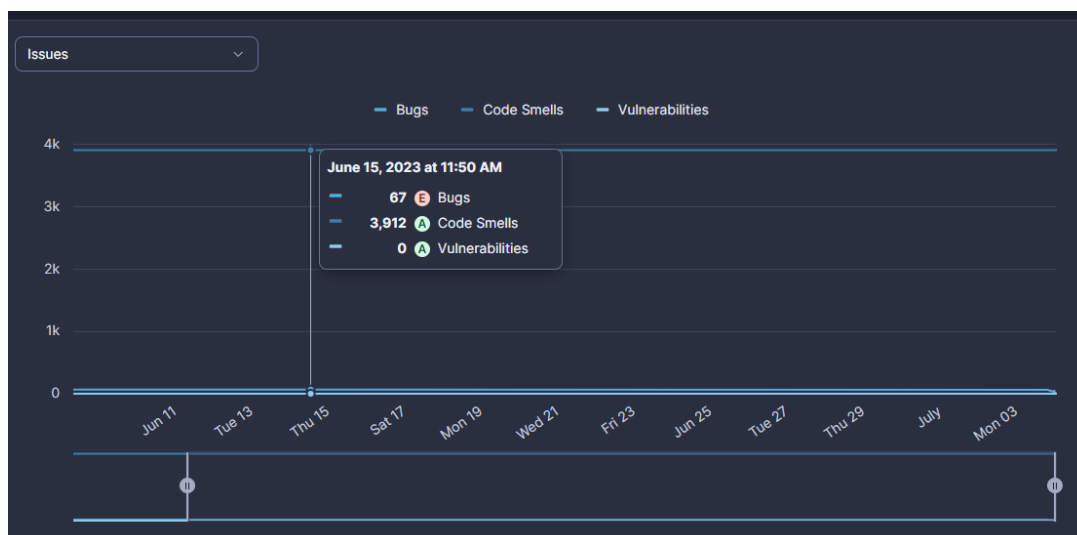
- 67 bugs:
 - 55 Major
 - 4 Minor
 - 7 Blocker
 - 1 Critical

La maggior parte dei bug è di tipo :

- ❖ "A NullPointerException could be thrown; getExtraBytes is nullable here."
- ❖ "A NullPointerException could be thrown; getWavepacket() can return null."

- 3,9k code smells
 - 12 Blocker
 - 174 Critical
 - 324 Major
 - 3.9k Minor

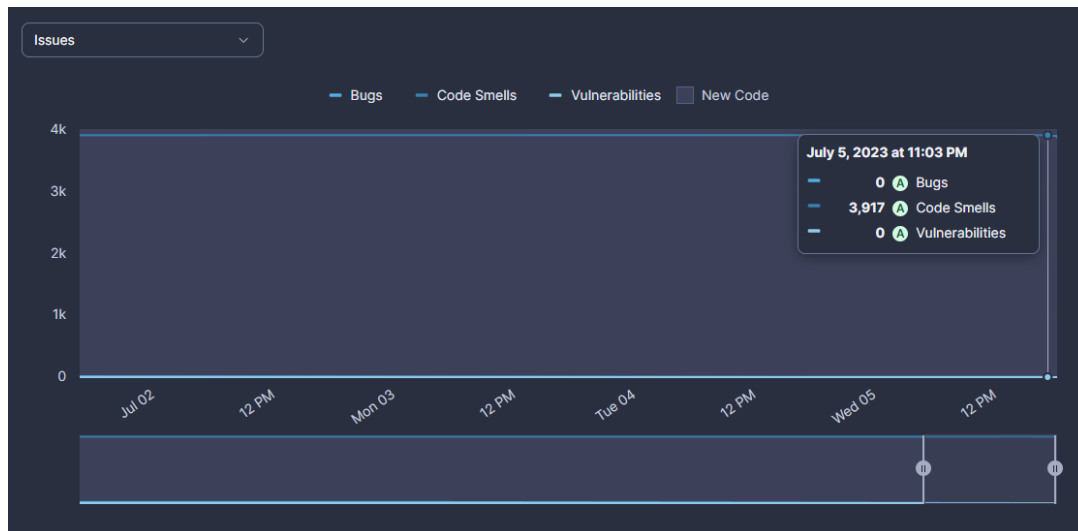
Il debito tecnico iniziale è di 39d dovuto alle issue di categoria 'code smells'.



Il termine dell'Analisi1, eseguita in data 05/07/2023 evidenzia una riduzione dei bugs da 67 a 0, dovuta alla risoluzione delle issues assegnate.

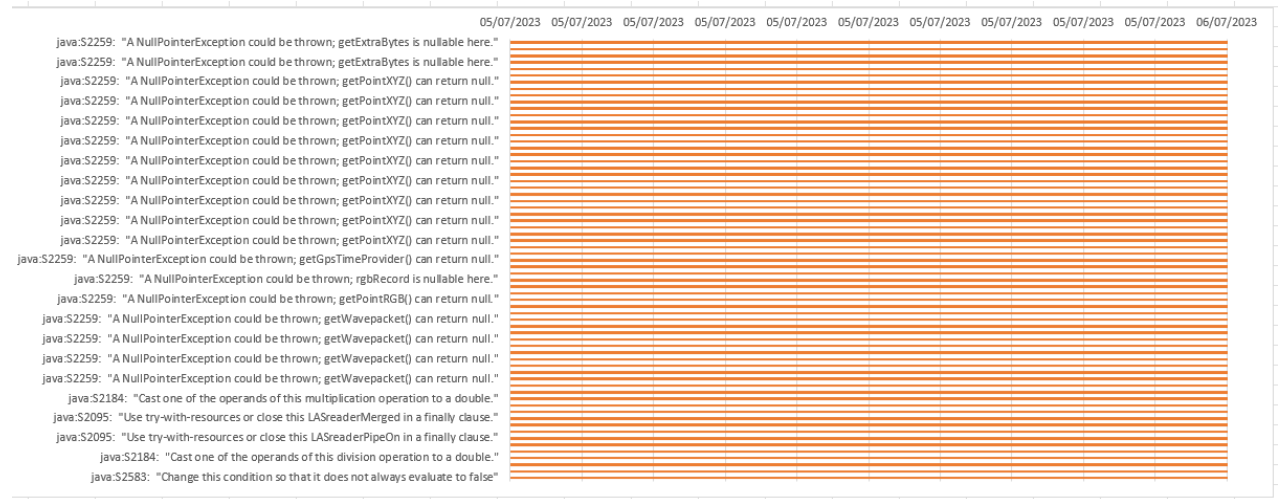
Il debito tecnico non è sceso poiché nella risoluzione dei bugs il numero di ore è poco rilevante.

Nonostante ciò la risoluzione assume comunque un'importanza nel ridurre il debito.



Gantt Finale

Nel Gantt dell'ActionPlan1 finale si può osservare, dal grafico, la data di conclusione della risoluzione delle issues dove viene messo in evidenza un leggero ritardo nella risoluzione dei bug.



Analisi2 SonarCloud

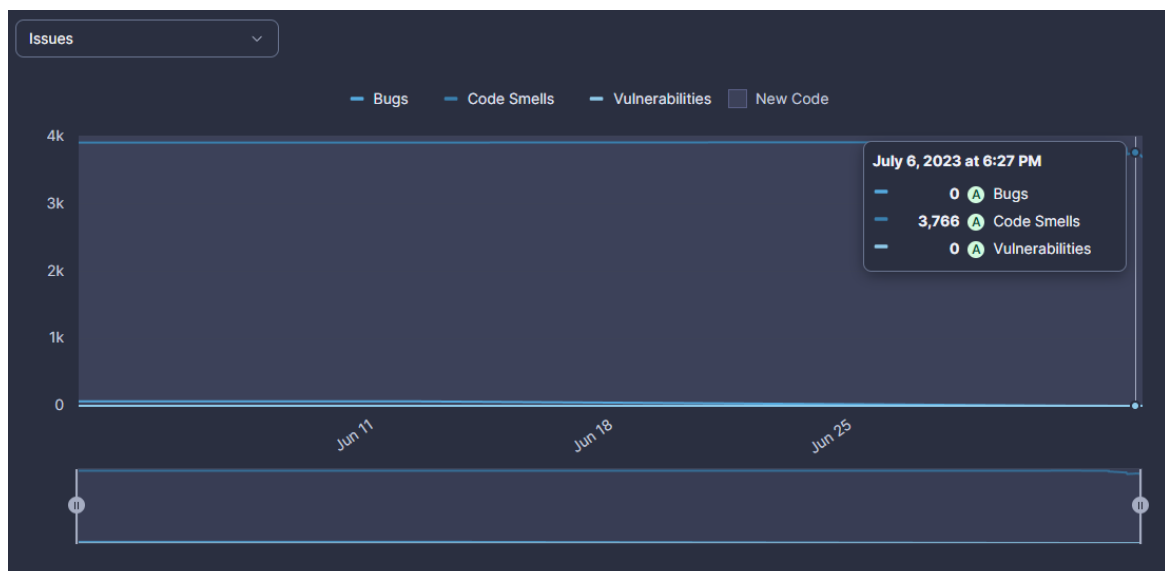
La seconda analisi eseguita in data 06/07/2023 presenta 3,766 code smells.

Rispetto all' analisi precedente si nota una riduzione, seppur minima, di essi dovuta alla loro risoluzione.

L' analisi generata il giorno successivo dalla riduzione dei bugs a 0, ha contribuito ad una minima diminuzione del debito del tempo.

La scelta strategica adottata di suddividere il lavoro in diverse tipologie : bug e code smells, permette di velocizzare e facilitare codesto servizio.

In questa analisi sono risultati tutti i code smells in un unico file, che è stato suddiviso a sua volta in altre 3 parti (con richiesta effettuata precedentemente).



ActionPlan2 (Introduzione)

I due Gantt iniziale e finale di ogni Action Plan sono stati generati manualmente da Excel poiché la piattaforma Redmine è fuori uso.

Gantt iniziale

Nel Gantt dell'ActionPlan2 iniziale si può osservare, dal grafico, la data di inizio della risoluzione delle issues dove viene indicata la data di conclusione di ogni issue dell'Action Plan.



Gantt Finale

Nel Gantt dell'ActionPlan2 finale si può osservare, dal grafico, la data di conclusione della risoluzione delle issues, dove i code smells sono stati risolti.



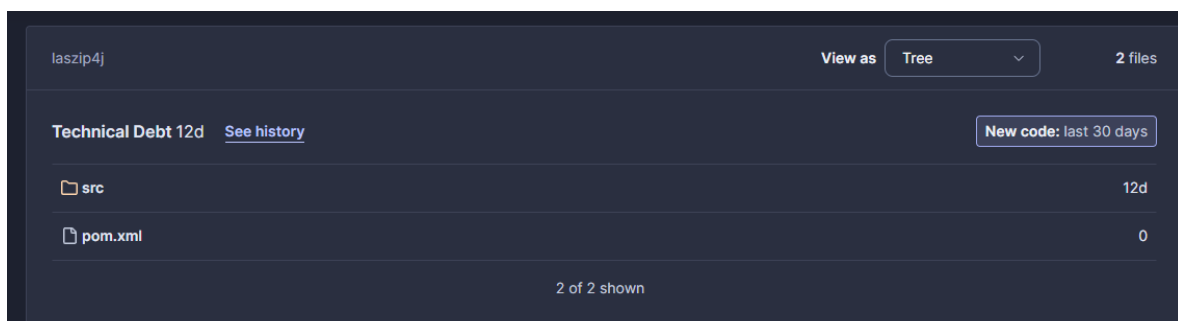
Analisi3 SonarCloud

La terza analisi eseguita in data 11/07/2023 evidenzia una riduzione dei code smells.

Questa analisi pone in rilievo una notevole riduzione per quanto riguarda i giorni di debito da 39d iniziali a 12d "finali".

È stato opportuno risolvere i code smells della categoria aventi un tempo stimato di 0 min.

In questa analisi vi sono molti code smells risolti implicitamente, poiché in precedenza uno o più tra questi, riguardanti la complessità del codice sono stati risolti, questo ha determinato appunto una risoluzione implicita.



Technical Debt 12d [See history](#) New code: last 30 days

Item	Days
src	12d
pom.xml	0

2 of 2 shown

ActionPlan3 (Introduzione)

I due Gantt iniziale e finale di ogni Action Plan sono stati generati manualmente da Excel poiché la piattaforma Redmine è fuori uso.

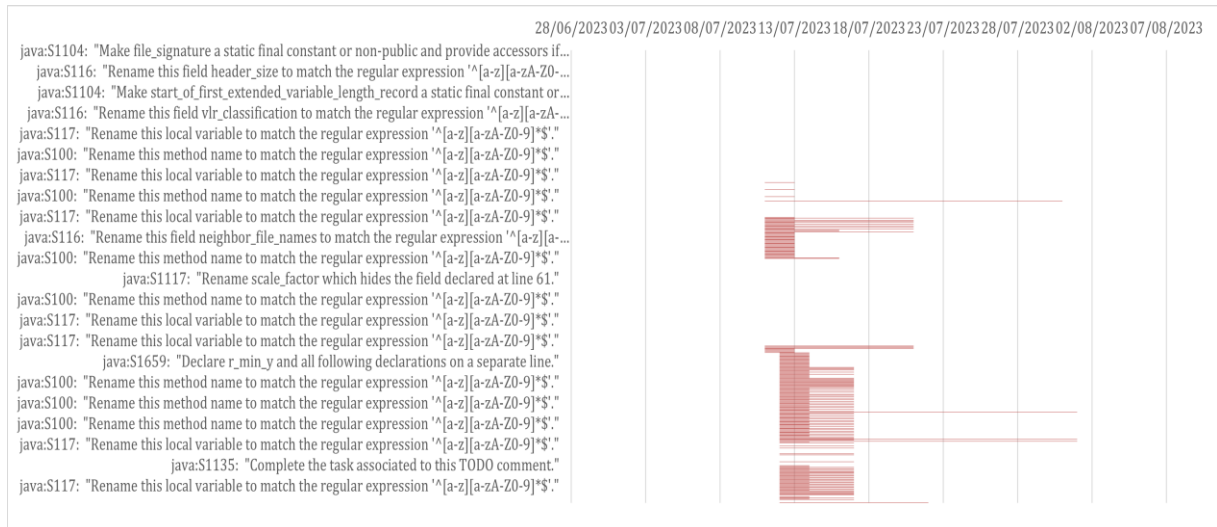
Gantt iniziale

Nel Gantt dell'ActionPlan3 iniziale si può osservare, dal grafico, la data di inizio della risoluzione delle issues dove viene indicata la data di conclusione di ogni issue dell'Action Plan.



Gantt Finale

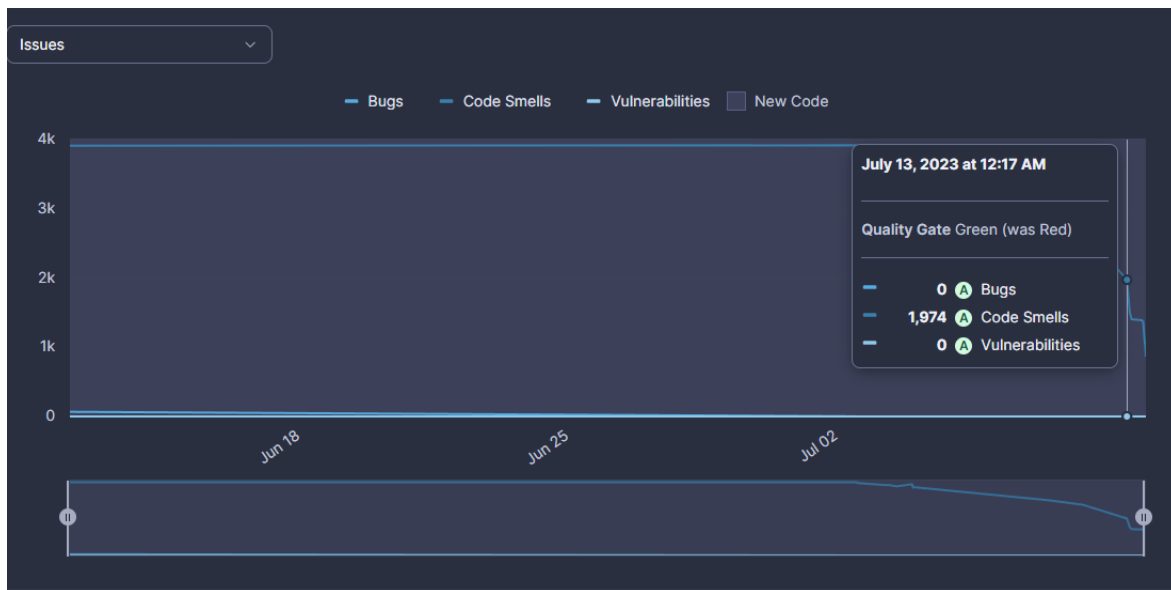
Nel Gantt dell'ActionPlan3 finale si può osservare, dal grafico, la data di conclusione della risoluzione delle issues, dove i code smells sono stati risolti.



Analisi4 SonarCloud

La quarta analisi eseguita in data 12/07/2023, presenta un ulteriore calo, ottimale, dei code smells.

Bisogna tener conto però che il target dei code smells non è stato definito ma per poter raggiungere l'obiettivo del Technical Debt a 5d, è necessario ridurre al minimo le issues rimanenti.



ActionPlan4 (Introduzione)

I due Gantt iniziale e finale di ogni Action Plan sono stati generati manualmente da Excel poiché la piattaforma Redmine è fuori uso.

Gantt iniziale

Nel Gantt dell'ActionPlan4 iniziale si può osservare, dal grafico, la data di inizio della risoluzione delle issues dove viene indicata la data di conclusione di ogni issue dell'Action Plan.



Gantt Finale

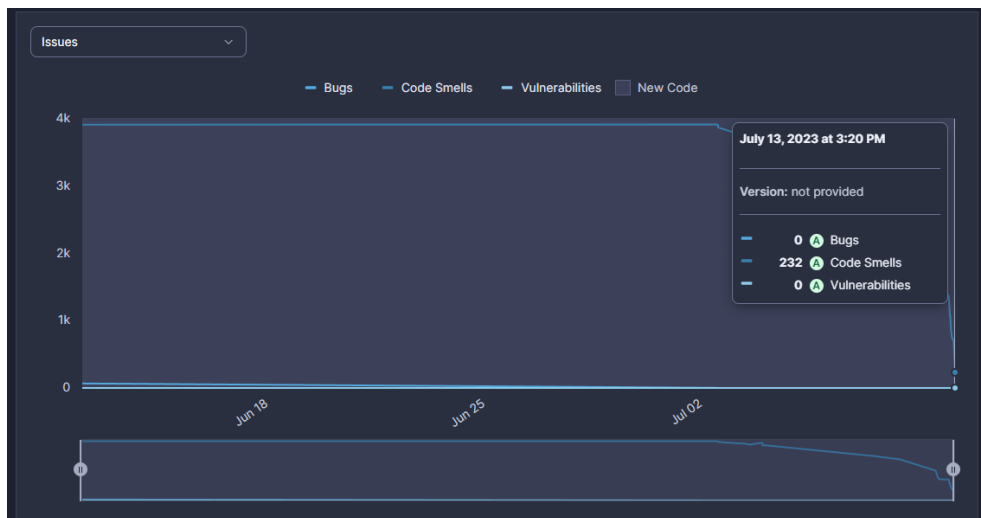
Nel Gantt dell'ActionPlan4 finale si può osservare, dal grafico, la data di conclusione della risoluzione delle issues, dove i code smells sono stati risolti.



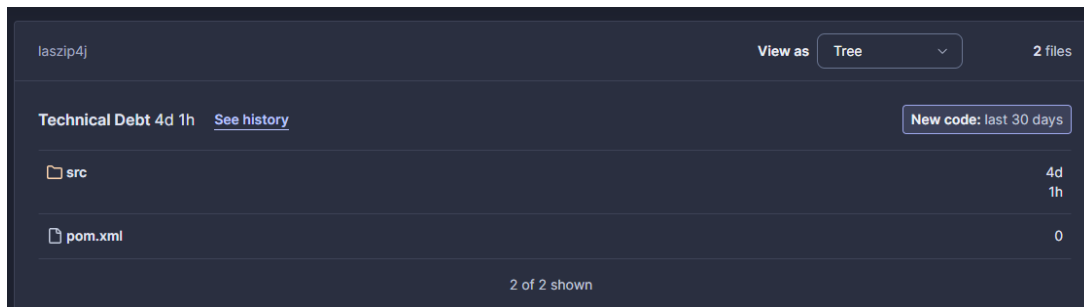
Analisi5 SonarCloud

La quinta analisi eseguita in data 13/07/2023, dimostra che i code smells non sono stati risolti totalmente, poiché l'obiettivo non è quello di ridurre i code smells a 0 ma quello di raggiungere il Technical Debt a 5d.

Infatti come si può osservare ne rimangono 232 code smells da risolvere, ciò nonostante i giorni di debito sono diminuiti.



In questo screen possiamo osservare che i giorni di debito sono arrivati a 4d,1h

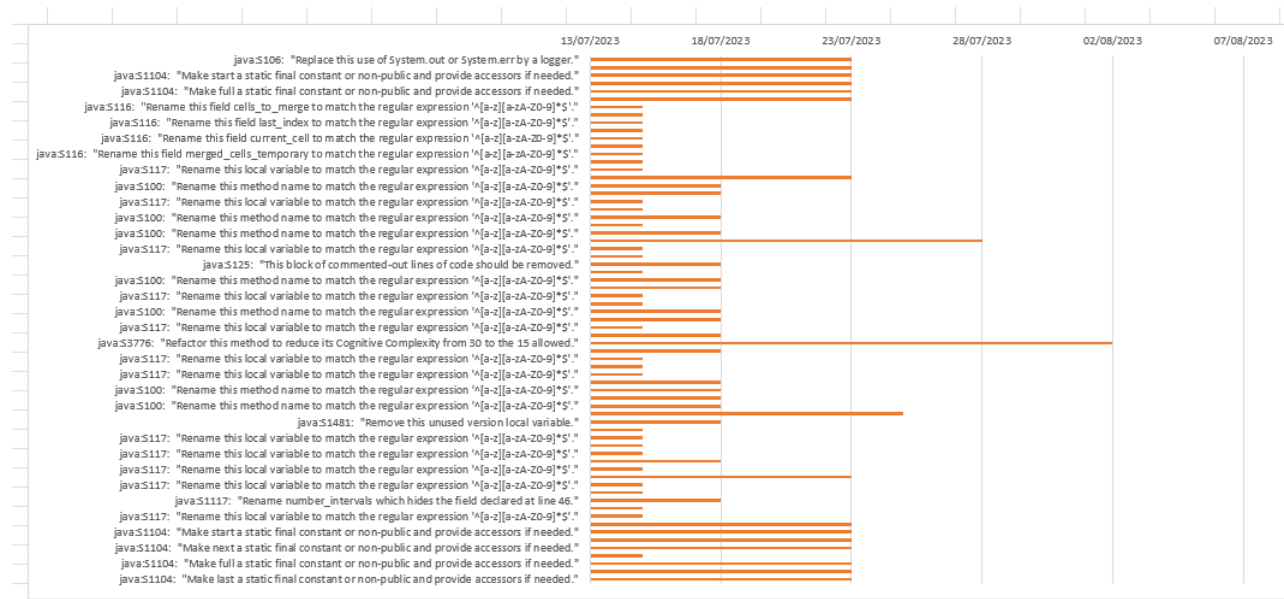


ActionPlan5 (Introduzione)

I due Gantt iniziale e finale di ogni Action Plan sono stati generati manualmente da Excel poiché la piattaforma Redmine è fuori uso.

Gantt iniziale

Nel Gantt dell'ActionPlan5 iniziale si può osservare, dal grafico, la data di inizio della risoluzione delle issues dove viene indicata la data di conclusione di ogni issue dell'Action Plan.



Gantt Finale

Nel Gantt dell'ActionPlan5 finale si può osservare, dal grafico, la data di conclusione della risoluzione delle issues, seppur non tutte sono state risolte poiché il Technical Debt deve raggiungere i 5d .

Come precedentemente detto grazie alla risoluzione dei code smells i giorni di debito sono arrivati a 4d,1h.

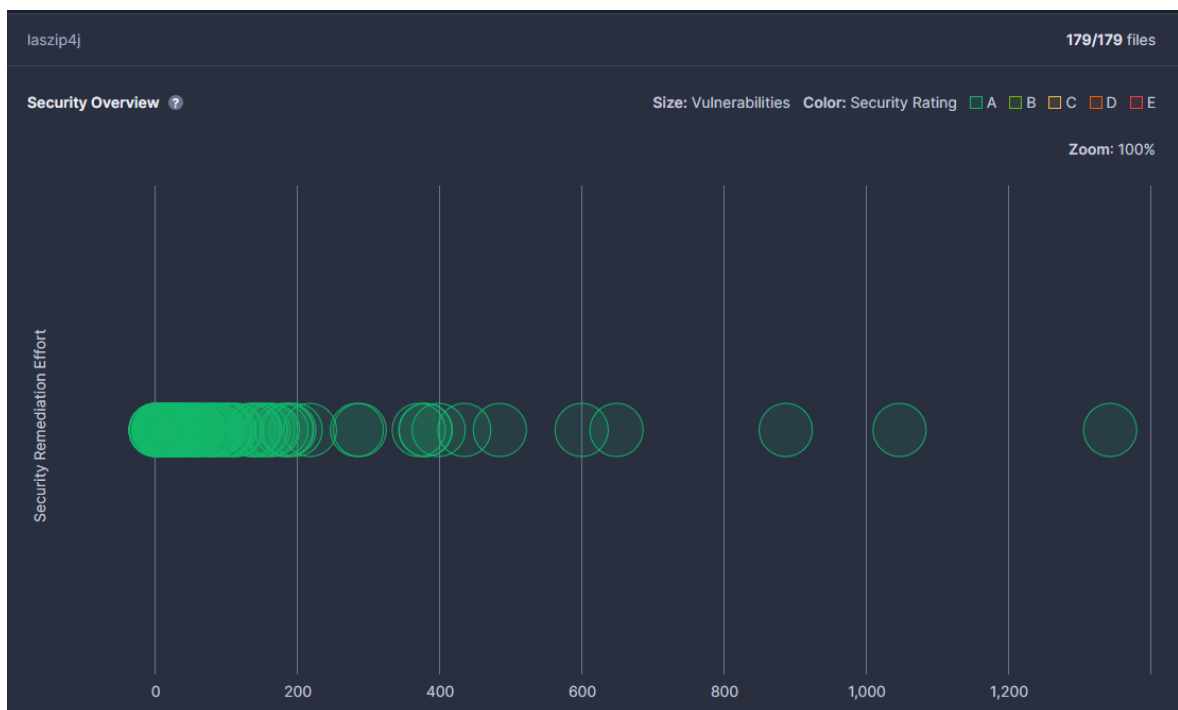
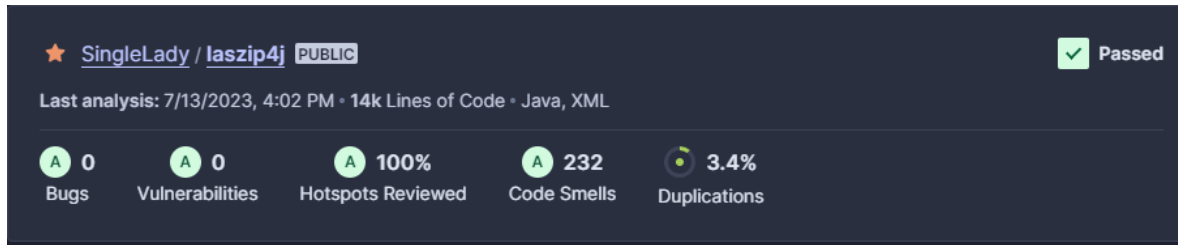


SICUREZZA(VULNERABILITA')

La "Sicurezza" è una caratteristica di qualità; essa si valuta grazie alle misure di Vulnerabilità e alle Security Hotspots.

All' interno del mio progetto tali misure erano già definite di classe A.

Come da screenshot:



QUALITA' INIZIALE VS QUALITA' FINALE AFFIDABILITA' (BUGS)

L'“Affidabilità” è una caratteristica di qualità, viene misurata per mezzo dei bugs.

L'analisi ha messo in rilievo la presenza di 67 bugs i quali sono stati risolti grazie alle indicazioni proposte da Sonar Cloud.

Il prodotto finito risulta privo di bugs, più affidabile e quindi più funzionante.

Ciò si evince dal grafico qui di seguito:



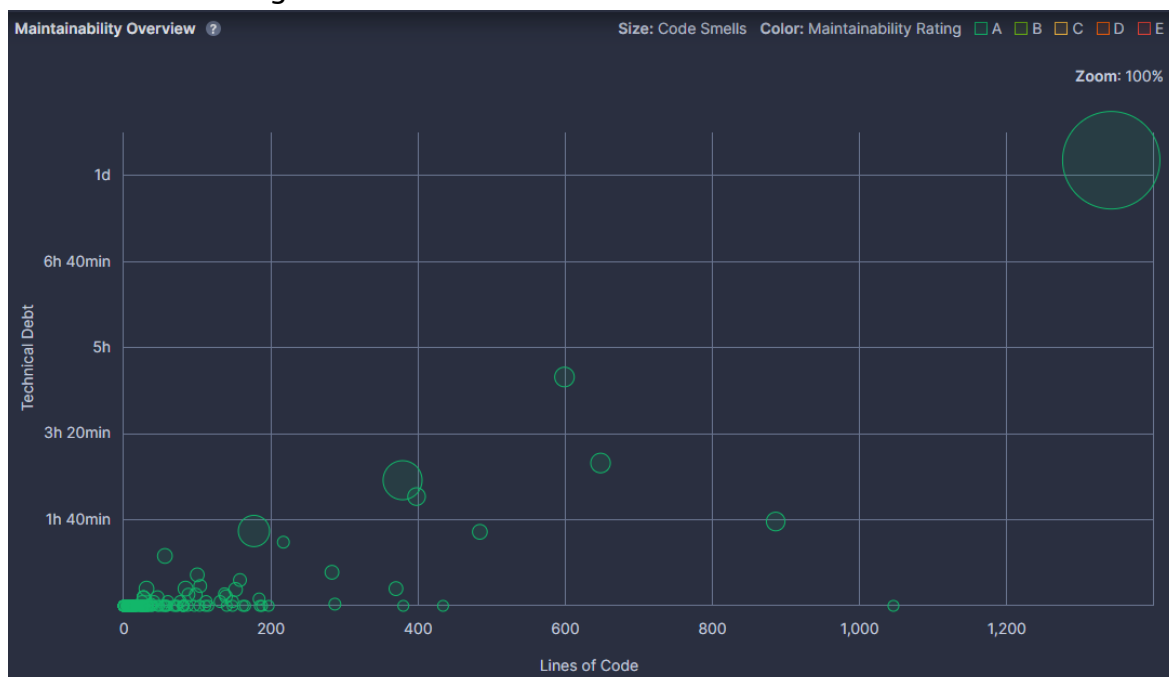
MANUTENIBILITA' (CODE SMELLS)

La "Manutenibilità" è una caratteristica di qualità, valutata per mezzo delle misure dei Code Smells.

L'analisi della qualità mi ha posto in evidenza la presenza di ben 3,9K code smells.

Grazie ad essa ho rilevato l'effort complessivo di 39 giorni di debito il quale aveva come problema principale la risoluzione dei code smells.

Una volta giunta alla risoluzione dei code smells vi è stato il raggiungimento della "Maintainability Rating" pari ad A e ho raggiunto l'obiettivo dei 5 giorni di debito come richiestomi.



CONCLUSIONI

Nel progetto Laszip4j sono stati sviluppati 5 Action Plan, riguardanti le risoluzioni identificate dall'analisi di qualità di Sonar Cloud, dove l'Action Plan 1 vede la risoluzione dei bug mentre gli altri 4 vedono la risoluzione dei code smells utili per portare il debito dei giorni a 5 come richiesto, anche se ho raggiunto il debito di 4 giorni e 1 ora.

Ogni Action Plan mi ha messo davanti a molteplici sfide, ho imparato l'importanza del prefissare e del raggiungere gli obiettivi.

Non ho potuto usufruire delle piattaforme Fortify e Redmine indicate per la realizzazione del progetto poiché non in uso, come riportato nella descrizione degli strumenti utilizzati, il mio obiettivo era comunque quello di raggiungere un buon risultato contando solamente sulle mie forze.