Apache Commons-email

Luigi Allocca, Simone Della Porta, Rocco Iuliano

ACM Reference Format:
Luigi Allocca, Simone Della Porta, Rocco Iuliano. 2023. Apache Commons-email. In *Proceedings of ACM Conference (Conference'17).* ACM, New York, NY, USA, 2 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

The aim of this project is to analyze and improve an existing project by Apache foundation. We will improve the availability (i), reliability (ii), safety (iii), security (iv) and resilience (v) of the project in order to improve the dependability. We will focus on improve security by improving the coverage of the test cases and creating new ones, fix the bugs related to security and reduce security hotspot issues which refer to a specific area or component of a software system that has a higher risk of security vulnerabilities or breaches. These issues are often identified through a security analysis or review, and they can pose a significant threat to the overall security of the system. By removing these security hotspots, the overall security of the project will be improved, reducing the risk of security breaches and protecting sensitive user data. This can lead to increased trust from users and stakeholders, and can help to ensure that the project is compliant with relevant security regulations and standards. Overall the general goals are:

- fix as many project bugs as possible, prioritizing the crucial bugs;
- improve the coverage of project testing by developing new test cases and improving the existing ones;
- minimize the number of code smells;
- reduce security hotspot issues;
- verify the project performance.

2 PROJECT PRE REQUIREMENTS

We choose a project with the following characteristics in order to improve the dependability of the software with CI/CD paradigm using the tools introduced in the course:

- Git Actions to check code quality, coverage, Java CI and security.
- The project should use Maven to manage the project build, so it should have the pom.xml file.

3 CONTEXT OF THE PROJECT

Apache Commons-Email aims to provide a API for sending email. It is built on top of the Java Mail API, which it aims to simplify. This project is structured in 3 package:

(1) **org.apache.commons.mail** that aims to provide a API for sending email;
(2) **org.apache.commons.mail.resolver** that contains implementation classes to resolve data sources from the following locations: class path file system URL (??);
(3) **org.apache.commons.mail.util** that contains some utility classes.

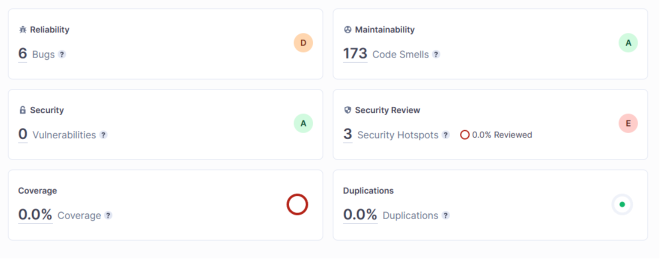
4 PRELIMINAR ANALISYS

After selecting the project Commons-email, we have created a fork of the repository, cloned the repository and built the project in order to run all test cases which result passed successfully. Then we performed a test push to verify the project actions and check the reults.

5 METRICS

6 METHODOLOGICAL STEPS CONDUCTED TO ADDRESS THE GOALS

Then we have conducted a preliminar analisys of the project by using *sonarcloud* [1] and *codecov* [2] and we obtained this results:



Reliability
6 Bugs ? D

Maintainability
173 Code Smells ? A

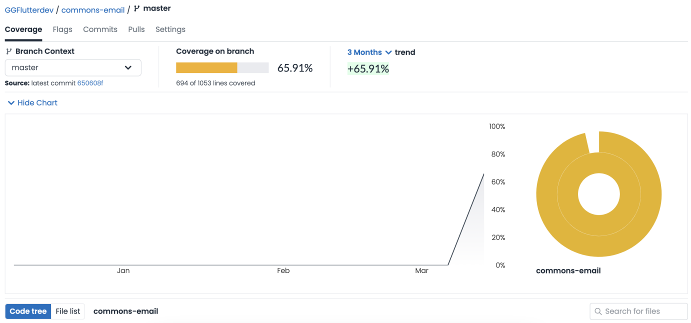
Security
0 Vulnerabilities ? A

Security Review
3 Security Hotspots ? ○ 0.0% Reviewed E

Coverage
0.0% Coverage ? ○

Duplications
0.0% Duplications ? ●

Figure 1: Sonarcloud analysis



GGFlutterdev / commons-email / master

Coverage Flags Commits Pulls Settings

Branch Context: master
Source: latest commit 650608f

Coverage on branch: 65.91%
694 of 1053 lines covered

3 Months trend: +65.91%

Hide Chart

Code tree File list commons-email

Search for files

Figure 2: Codecov analysis

The result of the analysis are:

(1) The coverage is 65,91%;
(2) The project has 6 bugs, of which 2 critic and 4 major;

Conference'17, July 2017, Washington, DC, USA
© 2023 Association for Computing Machinery.
ACM ISBN 978-x-xxxx-xxxx-x/YY/MM...$15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

(3) 3 security hotspot;
(4) 173 code smells.

So for having a continuous analisys of the project we decided to integrate these tools in the our project in order to analyze the code after every push or pull request.

7 RESULTS AND FINDINGS

8 CONCLUSIONS