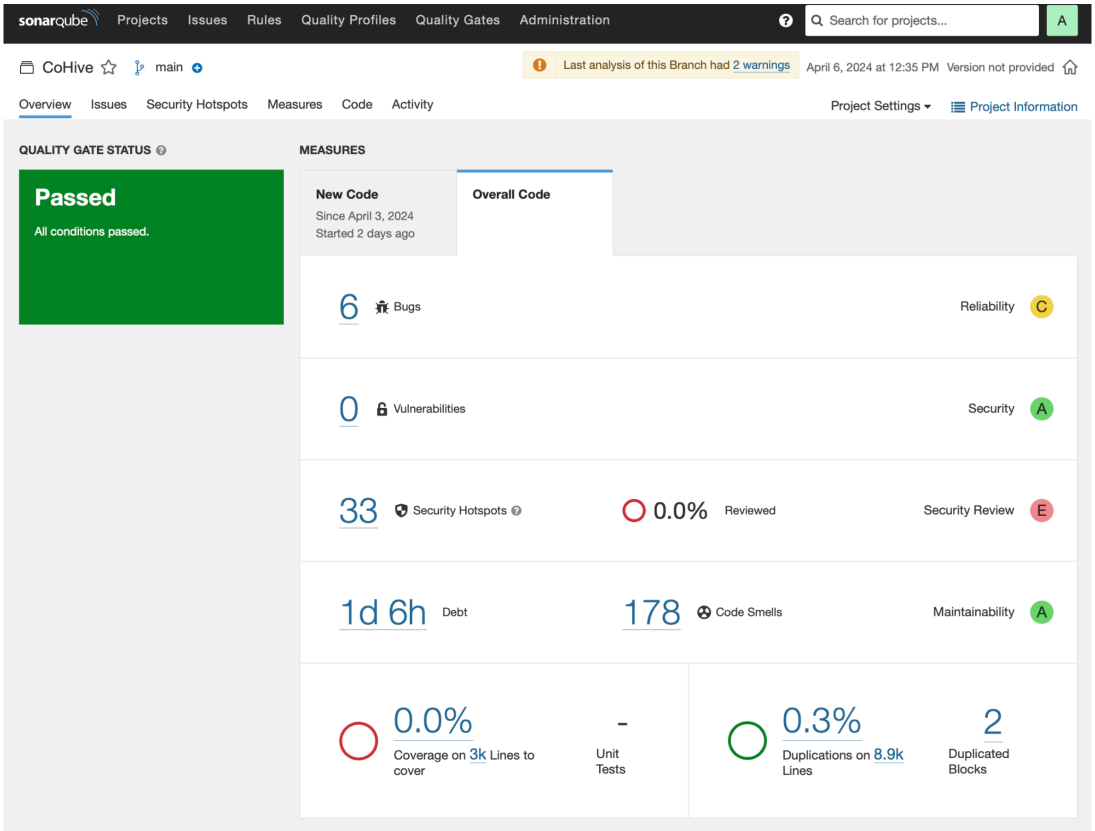


CoHive ☆ main

Last analysis of this Branch had 2 warnings April 6, 2024 at 12:35 PM Version not provided

QUALITY GATE STATUS

Passed

All conditions passed.

MEASURES

New Code
Since April 3, 2024
Started 2 days ago

Overall Code

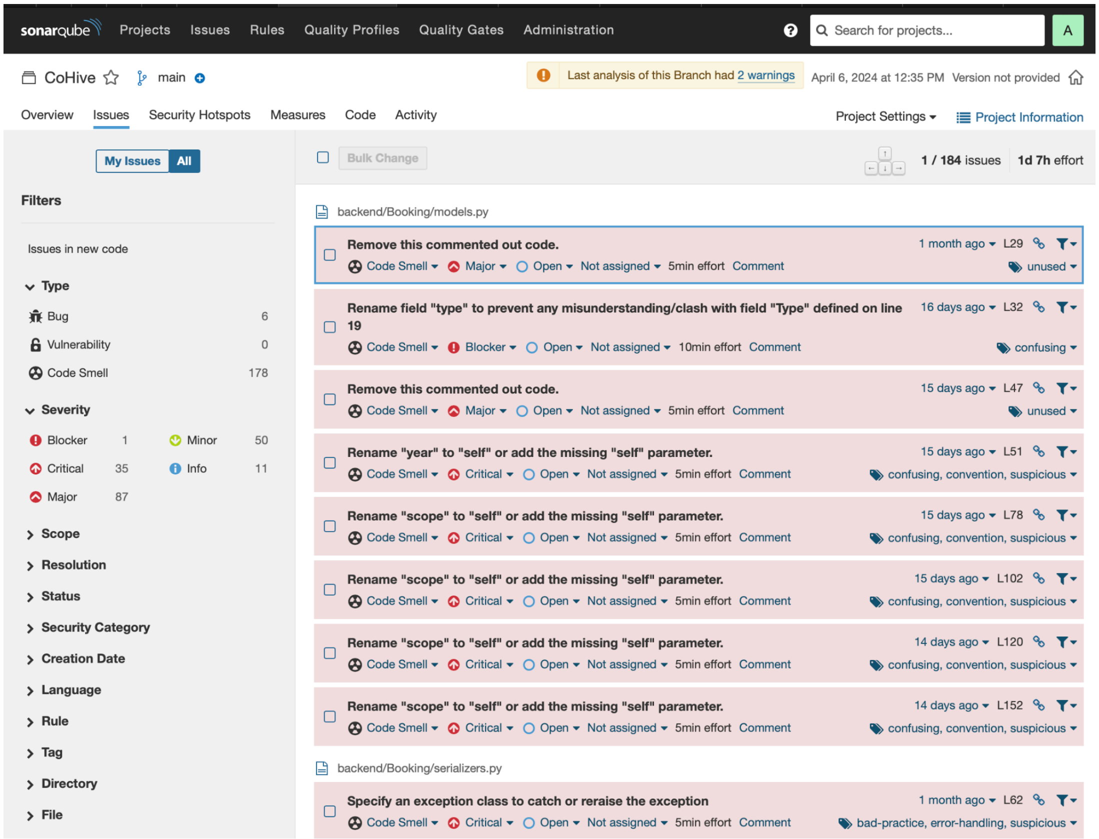
6 Bugs		Reliability C
0 Vulnerabilities		Security A
33 Security Hotspots	0.0% Reviewed	Security Review E
1d 6h Debt	178 Code Smells	Maintainability A

0.0% Coverage on 3k Lines to cover

- Unit Tests

0.3% Duplications on 8.9k Lines

2 Duplicated Blocks



CoHive ☆ main

Last analysis of this Branch had 2 warnings April 6, 2024 at 12:35 PM Version not provided

1 / 184 issues 1d 7h effort

Filters

Issues in new code

Type

Bug	6
Vulnerability	0
Code Smell	178

Severity

Blocker	1	Minor	50
Critical	35	Info	11
Major	87		

Scope
Resolution
Status
Security Category
Creation Date
Language
Rule
Tag
Directory
File

backend/Booking/models.py

□ Remove this commented out code.
Code Smell Major Open Not assigned 5min effort Comment 1 month ago L29 unused

□ Rename field "type" to prevent any misunderstanding/clash with field "Type" defined on line 19
Code Smell Blocker Open Not assigned 10min effort Comment 16 days ago L32 confusing

□ Remove this commented out code.
Code Smell Major Open Not assigned 5min effort Comment 15 days ago L47 unused

□ Rename "year" to "self" or add the missing "self" parameter.
Code Smell Critical Open Not assigned 5min effort Comment 15 days ago L51 confusing, convention, suspicious

□ Rename "scope" to "self" or add the missing "self" parameter.
Code Smell Critical Open Not assigned 5min effort Comment 15 days ago L78 confusing, convention, suspicious

□ Rename "scope" to "self" or add the missing "self" parameter.
Code Smell Critical Open Not assigned 5min effort Comment 15 days ago L102 confusing, convention, suspicious

□ Rename "scope" to "self" or add the missing "self" parameter.
Code Smell Critical Open Not assigned 5min effort Comment 14 days ago L120 confusing, convention, suspicious

□ Rename "scope" to "self" or add the missing "self" parameter.
Code Smell Critical Open Not assigned 5min effort Comment 14 days ago L152 confusing, convention, suspicious

backend/Booking/serializers.py

□ Specify an exception class to catch or reraise the exception
Code Smell Critical Open Not assigned 5min effort Comment 1 month ago L62 bad-practice, error-handling, suspicious

The screenshots from the SonarQube dashboard provide a detailed overview of the project's code quality analysis. The Quality Gate Status shows a "Passed" result, indicating compliance with the default quality thresholds set by SonarQube. SonarQube identified six bugs in the code but didn't detect any vulnerabilities, suggesting that there are no immediate security concerns. Even though 33 security hotspots were detected, it is still rated with an 'A.' In addition, the maintainability has also received an 'A,' indicating good code maintainability. However, the project has approximately one day and 6 hours of technical debt, representing the time it would take to address code quality issues. Notably, a 0.0% coverage metric implies that coverage reports still need to be integrated into SonarQube's analysis. Additionally, code duplication is minimal, which, when further checked, all 0.3% is on the bookingstests.py, which is a Selenium test and not a part of the project's code. Thus, the duplication metric across 8.9K lines of code is zero.

In the issues section of the dashboard, the project has 178 code smell issues, with a breakdown of severity indicating one 'blocker,' 35 'critical,' 87 'major,' 50 'minor,' and 11 'info' issues. These issues range from straightforward fixes, such as removing commented-out code, to more significant concerns, like renaming ambiguously named fields that could lead to confusion or clashes within the codebase. SonarQube also specified the files and lines of code that require attention, and with this kind of breakdown, it would be easier to know which issues to prioritize first in the future. The dashboard emphasizes the project's strengths while highlighting areas for improvement, especially in integrating testing in the SonarQube for better analysis and addressing the identified code smells to mitigate technical debt and improve code reliability.

The team has recently focused on addressing some of the issues, particularly those that are easy to fix, during the last sprint. This included resolving code smells related to naming conventions. Additionally, we've cleaned up numerous files by removing old and commented-out code blocks to enhance readability and maintainability. These efforts are crucial for improving the overall quality of the codebase and reducing technical debt.

While the team has been proactive in addressing many issues, it's worth noting that some have intentionally been overlooked. For instance, suggestions to rename variables like "type" have been deemed non-problematic by the team, given their context within the codebase. Additionally, a significant effort has been made to remove unused code, improving overall code cleanliness and performance. Furthermore, numerous print and log statements have either been removed entirely or commented out, contributing to a cleaner and more streamlined codebase. These actions underscore our commitment to enhancing code quality and maintainability.

The absence of vulnerability issues in the SonarQube analysis provides assurance to the team regarding the correct implementation of Google Auth and JWT (JSON Web Tokens), alleviating concerns over potential safety issues. Additionally, during this sprint, the team implemented token expiry, further enhancing security measures. Furthermore, a new feature was added to address previous concerns, whereby a 401 error now redirects the user back to the sign-in screen, mitigating the risk of complete system breakdowns. These proactive steps demonstrate our commitment to ensuring the security and reliability of the system.

In response to feedback from the tool regarding the prevention of unsafe and undesired API methods, we've taken steps to enhance the security and functionality of our codebase. Specifically, we've focused on cleaning up the view.py files, particularly those under the User section. This cleanup involved adding necessary checks for HTTP methods to ensure proper handling of user authentication and authorization. Additionally, we've improved error handling by ensuring that unauthorized requests return a proper 401 error. Furthermore, we've conducted general cleanup tasks, such as removing unused views, commented-out code, and print statements. These actions not only enhance the security and robustness of our system but also contribute to overall code cleanliness and maintainability.

One of the reported issues was regarding the apparent lack of utilization of the ErrorNotification. This observation stemmed from the absence of hits to this component in the code. However, it's important to note that this absence is indicative of the component functioning correctly, thus not triggering any errors necessitating notification. Despite being flagged as a bug by the tool, this scenario underscores the effectiveness of the ErrorNotification component in ensuring smooth operation without encountering errors.