Team 5 - Code review report
(Emil, Emmi, Musa, Verner)

**Objective**

As a team, we conducted a code review using SonarQube and code coverage metrics, also for proper practice we used a default ESlint configuration and a common formatting and naming etiquette. The findings were meant to increase maintainability and readability of code.

**Summary**

| Tool | Report |
|---|---|
| Code Coverage | 71.1% |
| SonarQube - Reliability | B - 110 open issues |
| SonarQube - Maintainability | A - 153 open issues |
| SonarQube - Duplications | 0.0% |
| SonarQube - Security | A - 0 open issues |

Common & Priority Issues

Reliability:

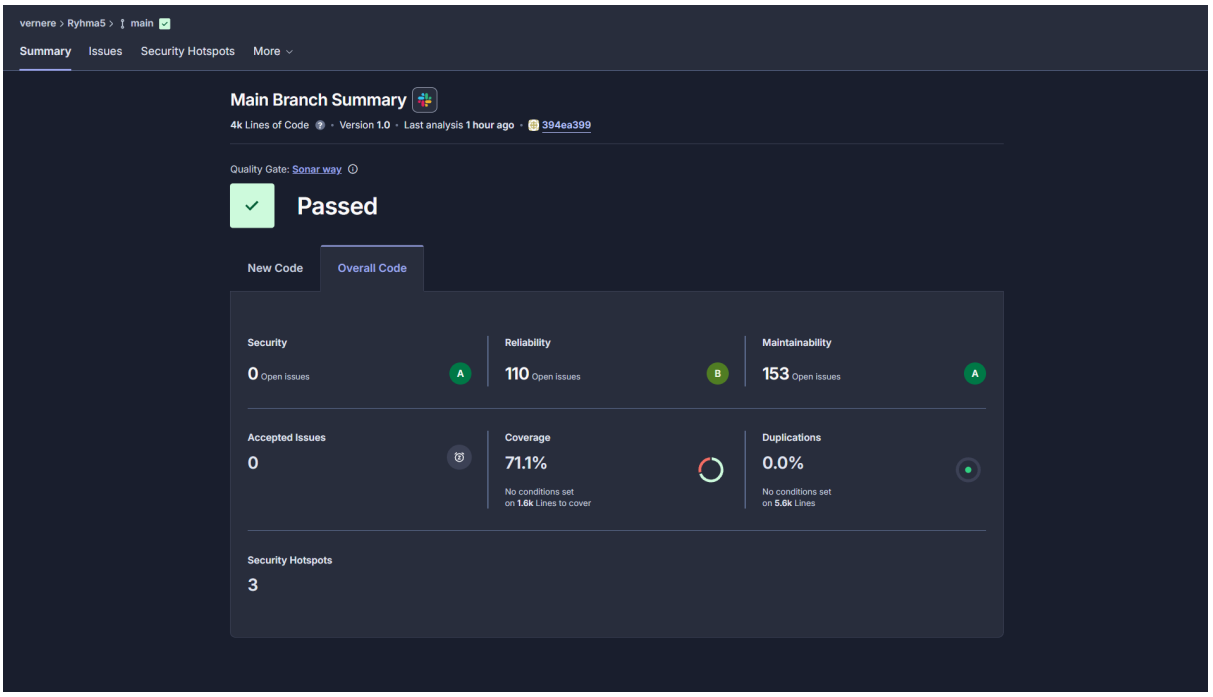
1. Missing prop validation
    - Reliability and maintainability low to medium priority
    - When passing props to components it is good practice to validate the existence of of the prop
2. Visible, non-interactive elements with click handlers must have at least one keyboard listener.
    - Reliability low priority
    - None interactive elements are not meant to have click handlers, causes readability issues

Maintainability:

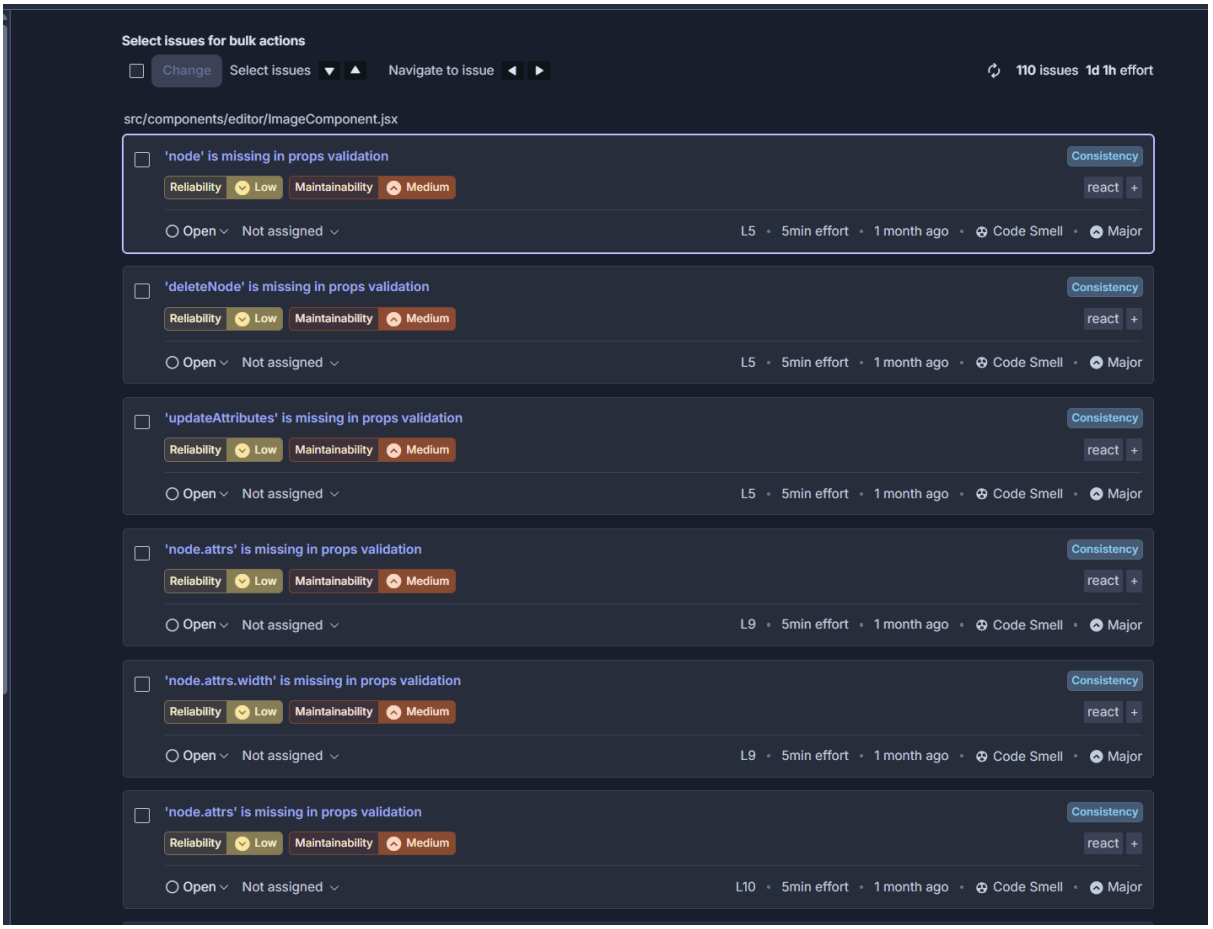
1. Components imported multiple times
    - Maintainability low priority
    - Affects readability
2. Multiple checks instead of optional chaining (?.)
    - Maintainability medium priority
    - Makes code more readable and concise
3. Unused imports
    - Maintainability low priority
    - Takes up less space

Examples:

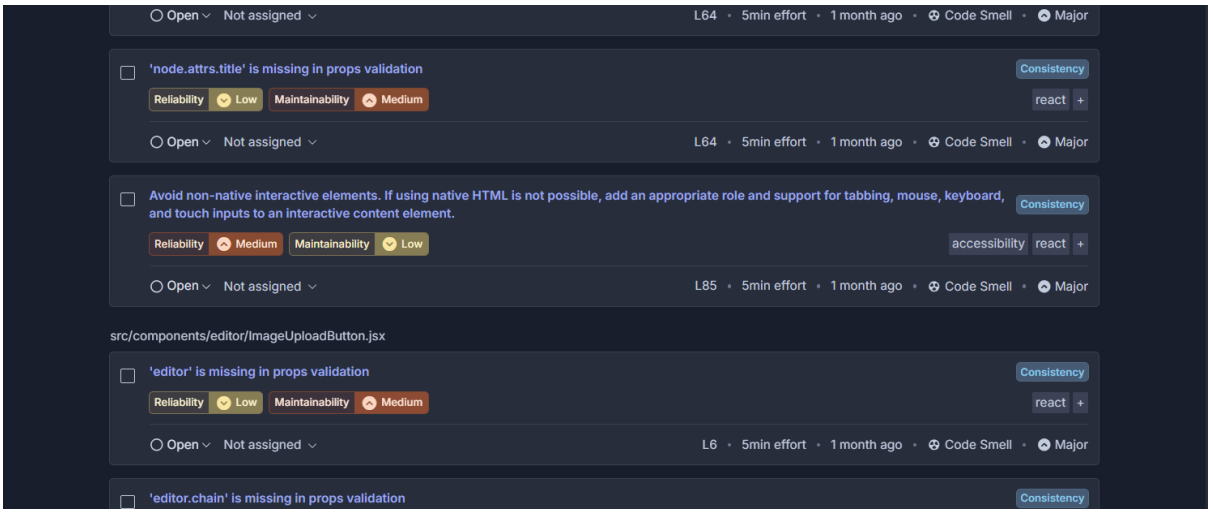
## 1. SonarQube report



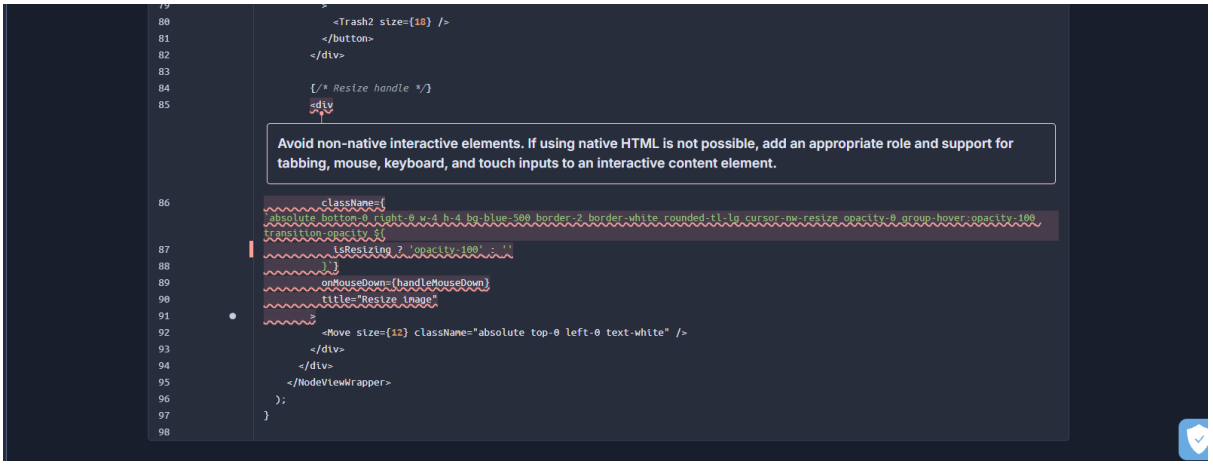

## 2. Prop validation



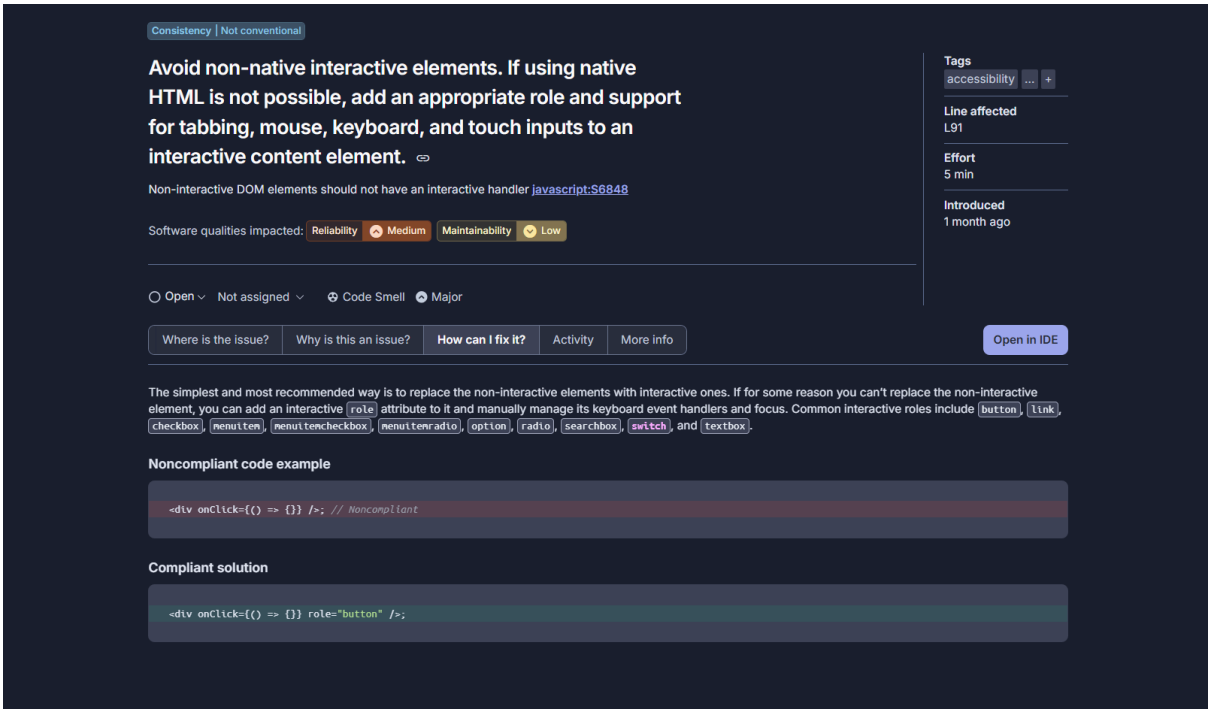
## 3. Missing non interactive elements




## 4. Example error




## 5. Sample solution

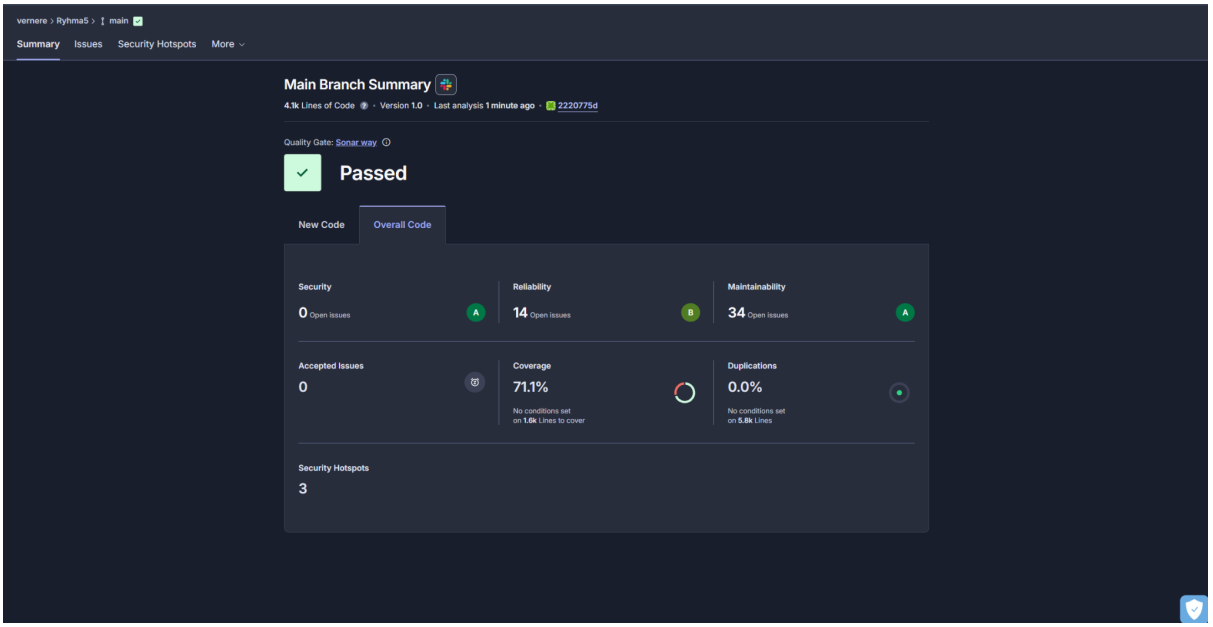


**Short-Term Actions(completed):**

- Added prop validation to all components that get passed props
- Upped code readability through removing duplicate imports
- Changed non-interactive elements with click handlers  to have at least one keyboard listener

**Long-Term Improvements:**

- Refactor collaboration invitation logic for better reliability




**Conclusion:**

From the beginning the code was designed and implemented with clear intent. As a result the code quality was reasonably good starting the cleanup. With the cleanup and refactoring, issues were reduced from 263 to 48 ( 81.75% reduction), which can be considered a successful result. These corrections improved the software’s reliability and maintainability.