UNIVERSITY OF NAPLES “FEDERICO II”

UNIVERSITÀ DEGLI STUDI DI NAPOLI
FEDERICO II

SOFTWARE ENGINEERING PROJECT A.A. 2023/2024

CODE ANALISYS DIETIDEALS24

VERSION DEVELOP

CANDIDATES

GIORDANO VINCENZO N86003039
LEONE GIANLUCA N86003082
MIGLIORISI ANDREA N86002979

This page was intentionally left blank.

ingsw_dietideals24_app

This SonarQube report provides a comprehensive analysis of the codebase to identify areas for improvement in terms of code quality, reliability, and maintainability. SonarQube is a powerful tool that detects code smells, bugs, security vulnerabilities, and other quality issues across multiple programming languages. By identifying these issues, SonarQube assists development teams in enhancing code quality, ensuring adherence to best practices, and reducing technical debt, which in turn supports sustainable development and long-term project scalability.

The purpose of this report is to summarize the key findings from the latest SonarQube scan, including detailed insights into code smells, duplications, and critical security risks. Each identified issue is categorized by severity and type, providing a clear roadmap for prioritizing fixes. The report includes an overview of the most significant issues and highlights the specific components or modules within the codebase that require immediate attention. Furthermore, the analysis considers historical trends, offering insights into how the code quality has evolved over time and whether previous remediation efforts have had a measurable impact.

This document also provides recommended actions to address specific issues, accompanied by actionable suggestions that developers can integrate directly into their workflows. These recommendations include leveraging automated testing, improving documentation, and refactoring problematic areas to reduce complexity. In addition to presenting the raw data, the report interprets key metrics such as maintainability index, reliability ratings, and security posture, emphasizing their implications for the long-term health of the project.

By addressing the findings in this report, the development team can improve code stability, enhance user trust by mitigating risks, and create a more efficient development environment. Furthermore, the insights provided can inform future development cycles, allowing teams to prevent similar issues from arising. The recommendations support not only short-term fixes but also strategic planning for continuous integration and delivery pipelines.

In summary, this report serves as a critical resource for assessing the current state of the codebase and outlines actionable steps to achieve a cleaner, more robust, and more secure code foundation. Its findings and insights will enable the team to make data-driven decisions, ensuring the software evolves to meet both current and future needs effectively.

CONTENT

ingsw_dietideals24_app

INTRODUCTION

This document presents the detailed results of the code analysis conducted for the **ingsw_dietideals_app**. The analysis was performed with the objective of identifying potential issues in the source code, optimizing performance, improving readability, and ensuring adherence to software development best practices. The results include key metrics, identified issues categorized by severity (bugs, vulnerabilities, code smells), and recommendations for their resolution.
Additionally, the document provides an overview of critical areas requiring immediate attention and proposes specific corrective actions for each identified issue. By reviewing this information, the development team can effectively plan refactoring activities and enhance the overall quality of the software product

CONFIGURATION

- Quality Profiles
  - Names: Sonar way [JavaScript];
  - Files:
    - Db.js
    - Db.js.old
    - passportConfig.js
    - auctionController.js
    - bidController.js
    - categoryController.js
    - notificationCategory.js
    - orderController.js
    - paymentController.js
    - searchController.js
    - sellerController.js
    - userController.js
    - auctionScheduler.js
    - auth.js
    - associations.js
    - auction.js
    - bid.js
    - category.js
    - notification.js

ingsw_dietideals24_app

        - order.js
        - payment.js
        - product.js
        - user.js
        - auctionRoutes.js
        - authRoutes.js
        - bidRoutes.js
        - categoryRoutes.js
        - notificationRoutes.js
        - orderRoutes.js
        - paymentRoutes.js
        - searchRoutes.js
        - sellerRoutes.js
        - userRoutes.js
        - auctionService.js
        - dayjs.js
        - app.js
        - docker-compose.yml
        - Dockerfile
- Quality Gate
    - Name: Sonar way
    - File: Sonar way.xml

SYNTHESIS

ANALYSIS STATUS

Reliability	Security	Security Review	Maintainability
A	A	A	A

QUALITY GATE STATUS

Quality Gate Status	Passed

Metric	Value
Reliability Rating on New Code	OK
Security Rating on New Code	OK
Maintainability Rating on New Code	OK
Coverage on New Code	OK
Duplicated Lines (%) on New Code	OK
Security Hotspots Reviewed on New Code	OK

METRICS

Coverage	Duplication	Comment density	Median number of lines of code per file	Adherence to coding standard
0 %	3.4 %	3.7 %	unknown	99.5 %

ingsw_dietideals24_app

TESTS

Total	Success Rate	Skipped	Errors	Failures
0	0 %	0	0	0

METRICS RANGE

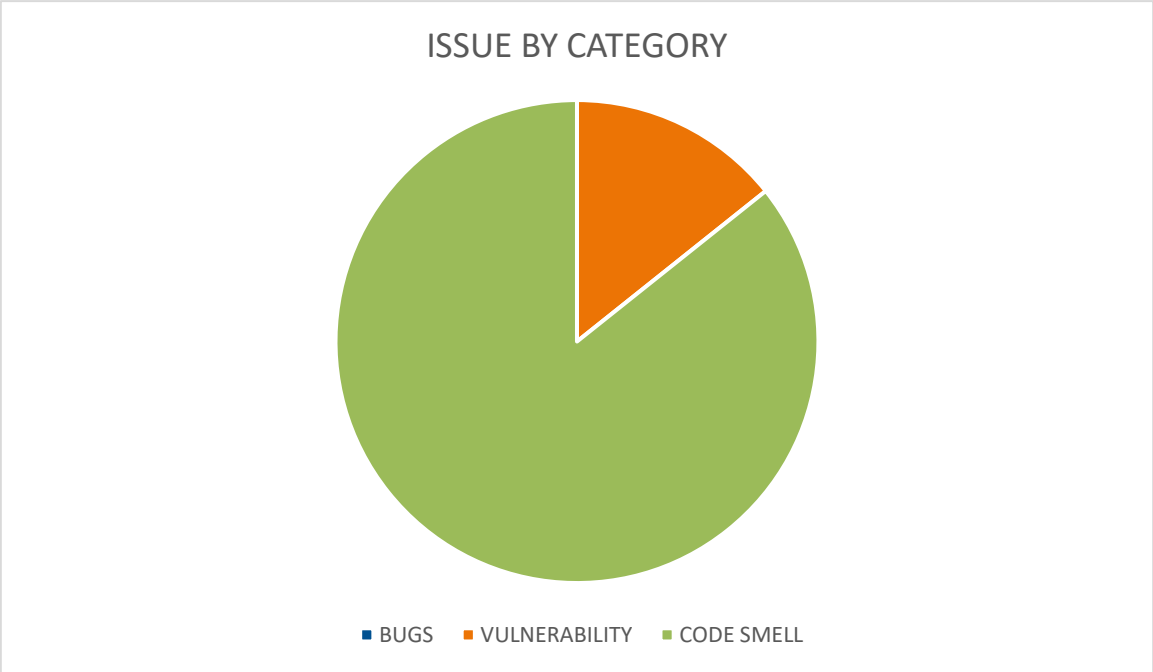
	Cyclomatic Complexity	Cognitive Complexity	Lines of code per file	Comment density (%)	Coverage	Duplication (%)
Min	unknown	unknown	unknown	unknown	0	unknown
Max	unknown	unknown	unknown	unknown	0	unknown

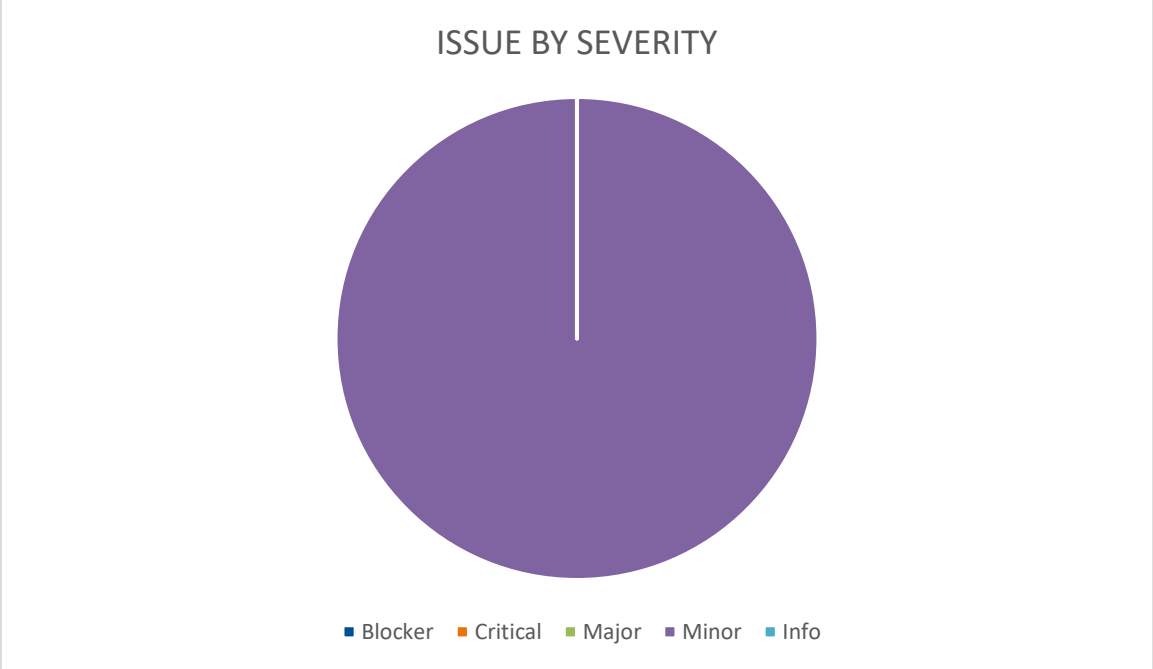
VOLUME

Language	Line of codes
JSON	3409
JavaScript	2136
YAML	43
Total	5588

ISSUES

CHARTS





ISSUES COUNT BY SEVERITY AND TYPE

Type / Severity	INFO	MINOR	MAJOR	CRITICAL	BLOCKER
BUG	0	0	0	0	0
VULNERABILITY	0	2	0	0	0
CODE_SMELL	0	12	0	0	0

ISSUES LIST

Name	Description	Type	Severity	Number
Prefer using an optional chain expression instead, as it's more concise and easier to read	SonarQube detects the opportunity to use **optional chaining** to make the code more concise and secure. Traditionally, when accessing a property of an object that might be null or undefined. This practice can be verbose and less readable. Optional chaining simplifies the code and makes it more readable by eliminating the need for repeated existence checks.	CODE_SMELL	MEDIUM	10
Unused local variables and functions should be removed	SonarQube detects a **useless assignment** to a variable (in this case, emails). This code smell occurs when a variable is assigned a value but is never used in the code, which leads to unnecessary clutter and can make the code harder to understand and maintain. In this case, the variable emails is assigned a value but is not utilized anywhere in the code. This leads to a redundant assignment that doesn't contribute to the program's functionality. SonarQube suggests removing such assignments to improve code clarity and reduce unnecessary memory usage. By eliminating unused variables, the code becomes cleaner and more efficient, enhancing maintainability and readability.	CODE_SMELL	MINOR	2

SECURITY HOTSPOTS

SECURITY HOTSPOTS COUNT BY CATEGORY AND PRIORITY

Category / Priority	LOW	MEDIUM	HIGH
LDAP Injection	0	0	0
Object Injection	0	0	0
Server-Side Request Forgery (SSRF)	0	0	0
XML External Entity (XXE)	0	0	0
Insecure Configuration	1	0	0
XPath Injection	0	0	0
Authentication	0	0	0
Weak Cryptography	0	0	0
Denial of Service (DoS)	0	0	0
Log Injection	0	0	0
Cross-Site Request Forgery (CSRF)	0	0	0
Open Redirect	0	0	0
Permission	0	0	0
SQL Injection	0	0	0
Encryption of Sensitive Data	0	0	0
Traceability	0	0	0
Buffer Overflow	0	0	0
File Manipulation	0	0	0
Code Injection (RCE)	0	0	0

Cross-Site Scripting (XSS)	0	0	0
Command Injection	0	0	0
Path Traversal Injection	0	0	0
HTTP Response Splitting	0	0	0
Others	1	0	0

SECURITY HOTSPOTS LIST

Name	Description	Type	Severity	Number
Make sure that enabling CORS is safe here	Having a permissive Cross-Origin Resource Sharing policy is security-sensitive. It has led in the past to the following vulnerabilities: • CVE-2018-0269 • CVE-2017-14460 Same origin policy in browsers prevents, by default and for security-reasons, a javascript frontend to perform a cross-origin HTTP request to a resource that has a different origin (domain, protocol, or port) from its own. The requested target can append additional HTTP headers in response, called CORS, that act like directives for the browser and change the access control policy / relax the same origin policy.	Insecure Configuration	LOW	1
This framework implicitly discloses version information by default. Make sure it is safe here	Disclosure of version information, usually overlooked by developers but disclosed by default by the systems and frameworks in use, can pose a significant security risk depending on the production environement. Once this information is public, attackers can use it to identify potential security holes or vulnerabilities specific to that version. Furthermore, if the published version information indicates the use of outdated or unsupported software, it becomes easier for attackers to exploit known vulnerabilities. They can search for published vulnerabilities related to that version and launch attacks that specifically target those vulnerabilities.	Version Disclosure	MINOR	1