Міністерство освіти і науки України

Національний університет «Львівська політехніка»

Інститут комп’ютерних наук та інформаційних технологій

Кафедра програмного забезпечення

****

**ЗВІТ**

**Про виконання лабораторної роботи № 1**

**з дисципліни «Управління якістю програмного забезпечення»**

**«Налаштування Sonarqube для проекту і обчислення показників якості на основі отриманих результатів»**

**Лектор:**

Ваврук І.Є.

**Виконав:**

студ. групи ПЗМ-11

Хижняк Д. Е.

**Прийняв:**

Ваврук І.Є.

«\_\_\_» \_\_\_\_\_\_ 2019 р.

∑ = \_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_

Львів – 2019

**Тема роботи:** Налаштування Sonarqube для проекту і обчислення показників якості на основі отриманих результатів.

**Мета роботи:** Навчитись застосовувати Sonarqube для аналізу показників якості програмних проектів.

**Завдання**

Передумовою виконання завдання є наявність великого проекту (курсова робота, бакалаврська робота або що) і мінімум 10-15 написаних юніт-тестів для даного проекту.

1. Локально встановити SonarQube.
2. Налаштувати 5 власних правил перевірки коду (quality rules) з різним типом (bug, code smell, etc.) та пріоритетом впливу на систему (severity).
3. Налаштувати критерії успішного виконання проекту (quality gates).
4. Оцінити якість проекту з використанням SonarQube.
5. Виправити частину знайдених помилок, а також дописати 2 юніт-тести, коментарі.
6. Оцінити якість проекту з використанням SonarQube.
7. Повторювати кроки 5-6 тричі.
8. Зібрати статистику покращення якості коду (Reliability (Надійність), Security (Безпека), Maintainability (Підтримуваність), Coverage (Покриття тестами), Duplications (Дублювання), Size (Розмір проекту), Complexity (Цикломатична складність), Documentation (Документування коду) і Issues (Помилки))

**Хід роботи**

1. Локально встановили SonarQube та налаштували проект для роботи з ним.

У директорії проекту створено \*.bat файл, що виконує ряд дій для проведення аналізу:

* Запускає наявні у проекті юніт тести.
* Генерує xml файл з даними про покриття коду.
* Запускає SonarScanner.
* Виконує збірку проекту, під час чого SonarScanner аналізує його.

У панелі адміністрування SonarQube було виключено з аналізу JavaScript бібліотеки, що використовуються у проекті.

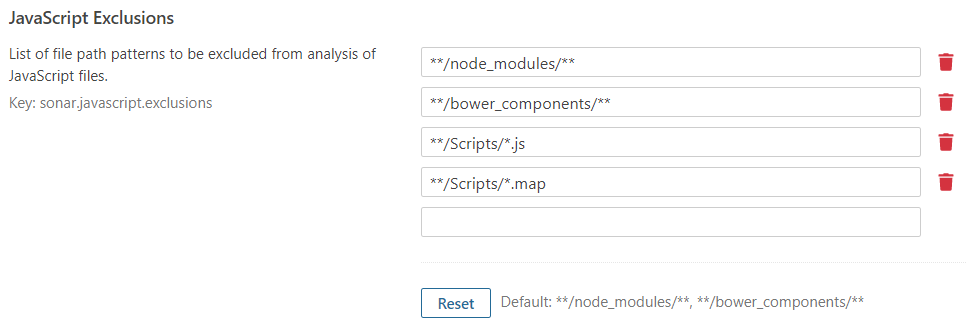


Рис. 1. Виключені з аналізу JavaScript коду директорії

На момент першого аналізу у рішення містило 12 юніт тестів. Результати аналізу подано на Рис. 3.

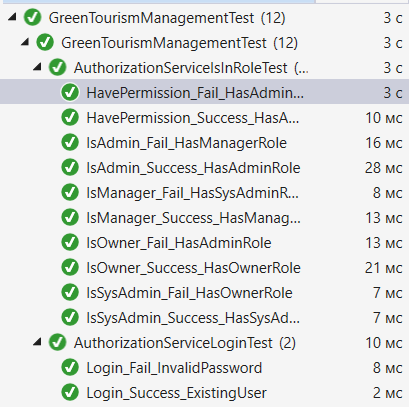


Рис. 2. Перелік початкових юніт тестів

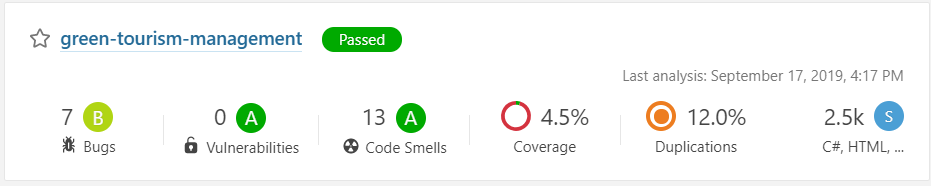


Рис. 3. Результати першого аналізу проекту

2. Налаштовано 5 додаткових правил перевірки коду (Quality Rules):

* Не пройдені юніт тести мають бути виправлені (Bug, Major).
* Не має використовуватись логування у консоль (Vunerability, Minor).
* Оператор «==» не повинен використовуватись якщо метод «Equals» перевизначений (Code smell, Major).
* Виклики «async» методів не повинні блокувати потік (Code smell, Blocker).
* Конструкції «if ... else if» повинні закінчуватись виразом «else».

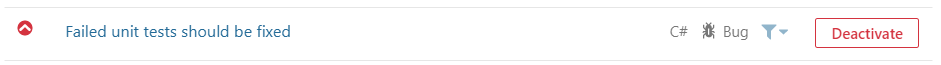


Рис. 4. Приклад вигляду активованого правила перевірки коду

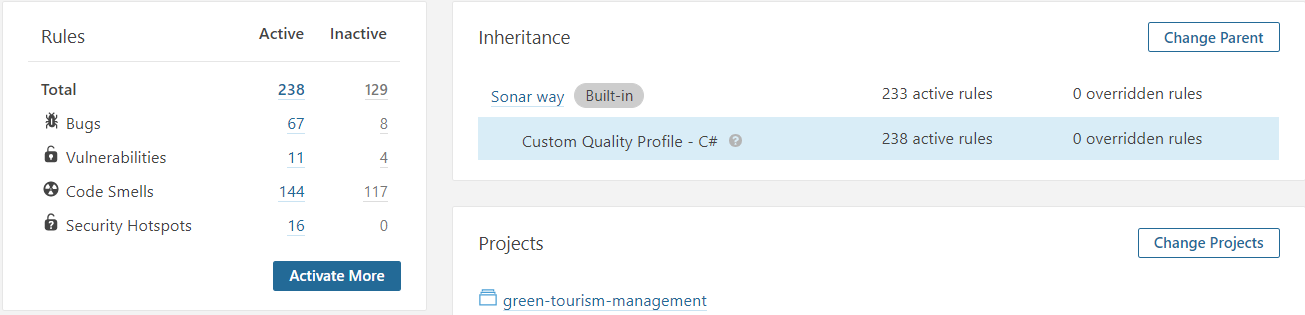


Рис. 5. Інформація про налаштований профіль перевірки якості

3. Налаштовано критерії успішного виконання проекту (Quality Gates).

* За основу взято стандартні критерії «Sonar Way». До них було додано:
* Кількість критичних проблем не більше 0.
* Кількість продубльованих блоків коду не більше 15.
* Не більше 5 нових багів за версію.
* 0 пропущених юніт тестів.
* 0 не пройдених юніт тестів.



Рис. 6. Перелік застосованих умов у створеному Quality Gate

4. Оцінено якість проекту з використанням SonarQube.

Після проведеного аналізу було виявлено, що проект не пройшов Quality Gates, через нову критичну умову: конструкції «if ... else if» повинні закінчуватись виразом «else». Інші показники якості лишились не змінними.

5. Додано 2 юніт теста та коментарі до протестованих ділянок коду, виправлено 7 багів та 2 блоки коду, що дублюється. Виправлено критичну проблему. Виправлений код не був покритий тестами, тому виникла нова проблема Рис. 9.

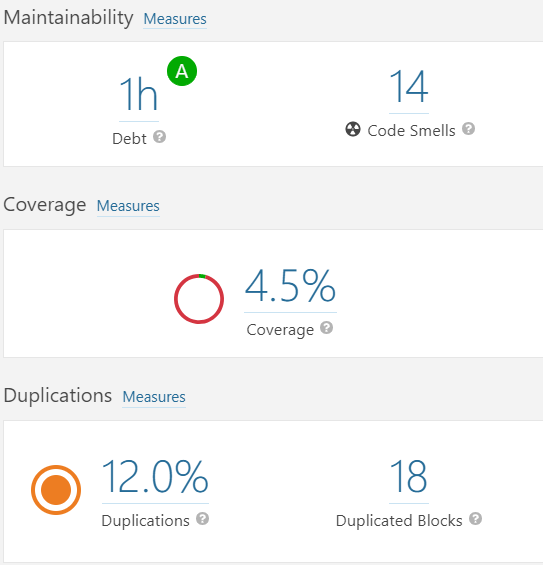
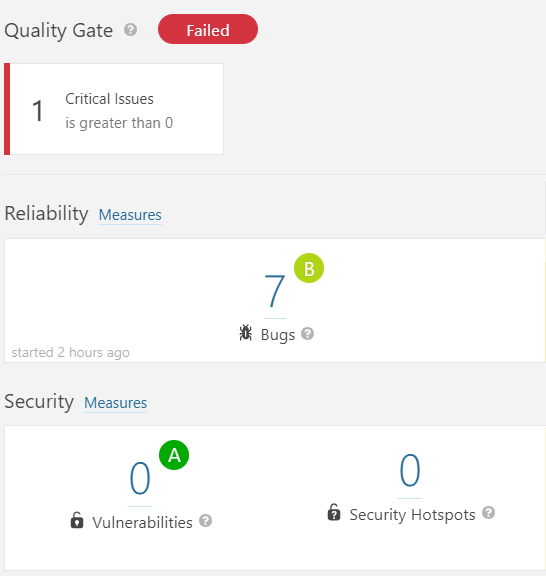


Рис. 7. Результати аналізу проекту після налаштування Quality Gates i Quality Rules

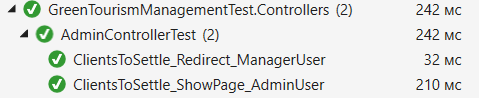


Рис. 8. Додані юніт тести для 2 ітерації

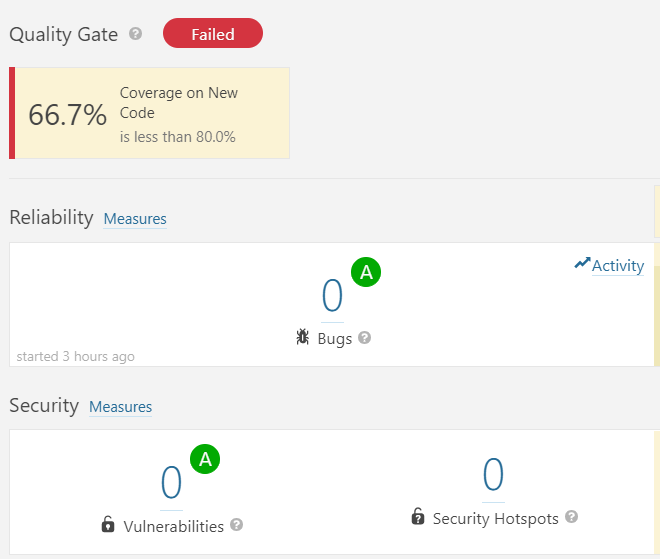
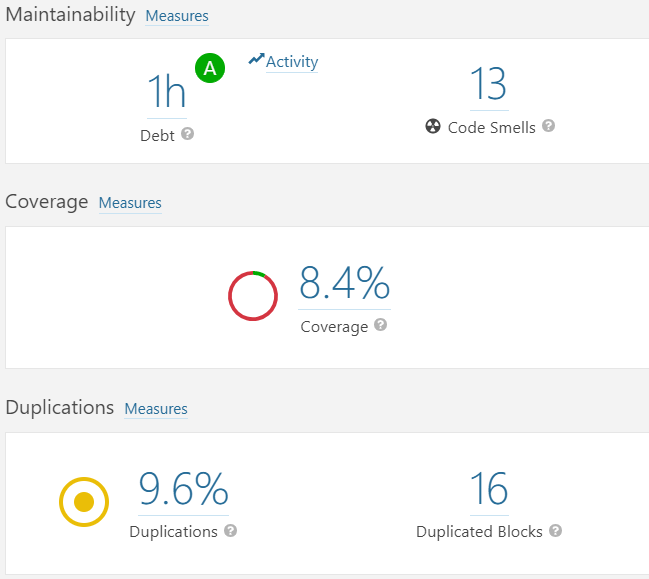
 

Рис. 9. Результати аналізу проекту після 1 ітерації виправлень

6. Повторено дії по покращенню якості коду, цього разу Quality Gate пройшли успішно.

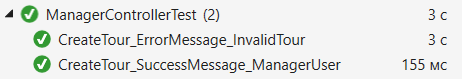


Рис. 10. Додані юніт тести для 2 ітерації

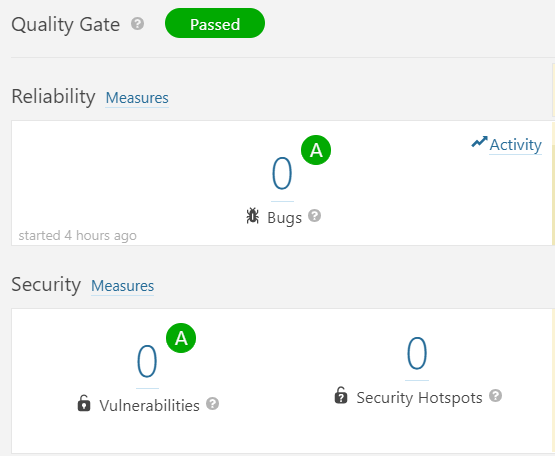
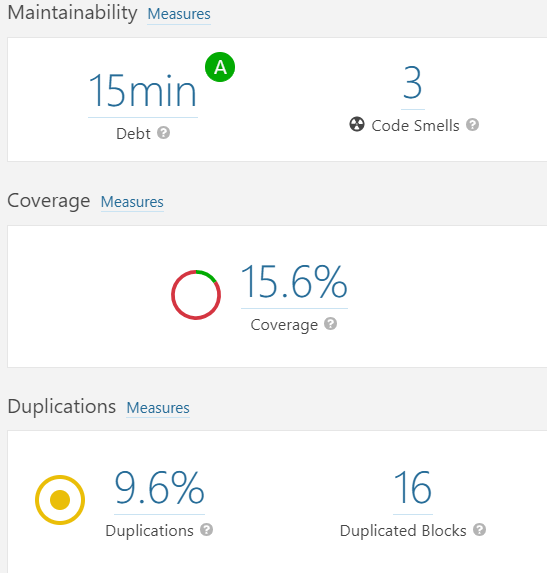
 

Рис. 11. Результати аналізу проекту після 2 ітерації виправлень

7. Отримано кінцеві результати аналізу якості коду після 3 ітерацій покращень.

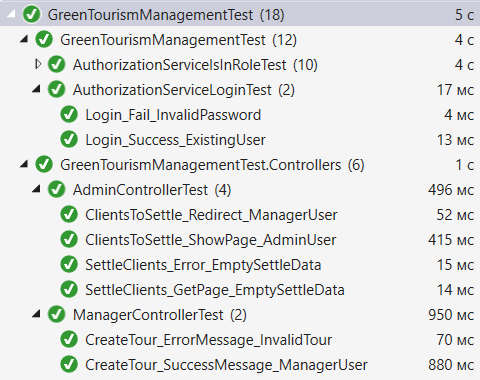


Рис. 12. Юніт тести наявні після 3 ітерацій

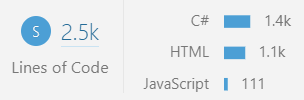


Рис. 13. Розмір проекту

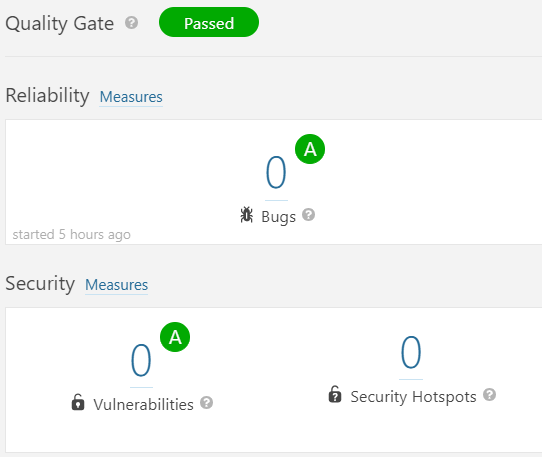
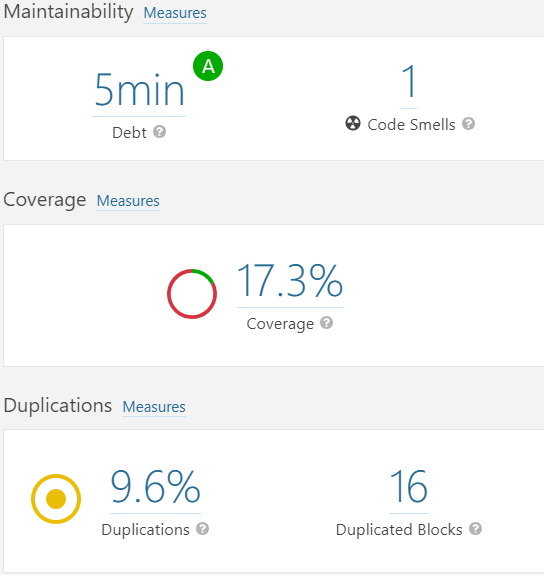
 

Рис. 14. Результати аналізу проекту після 3 ітерації

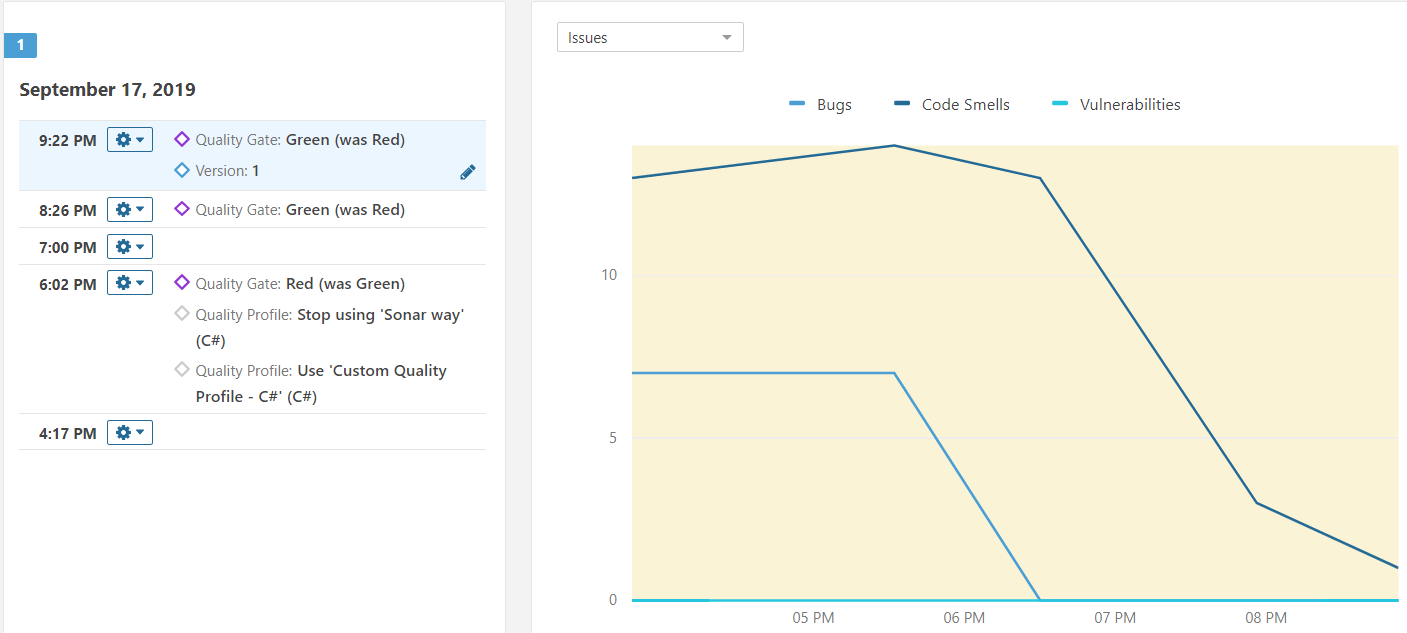


Рис. 15. Графік наявних проблем на кожну ітерацію

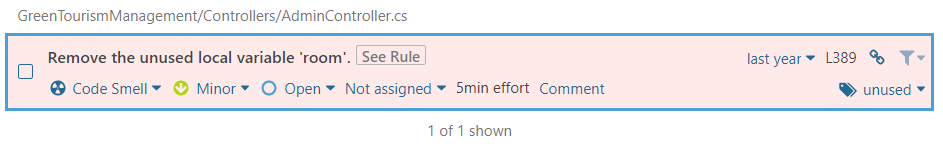


Рис. 16. Наявні у проекті проблеми

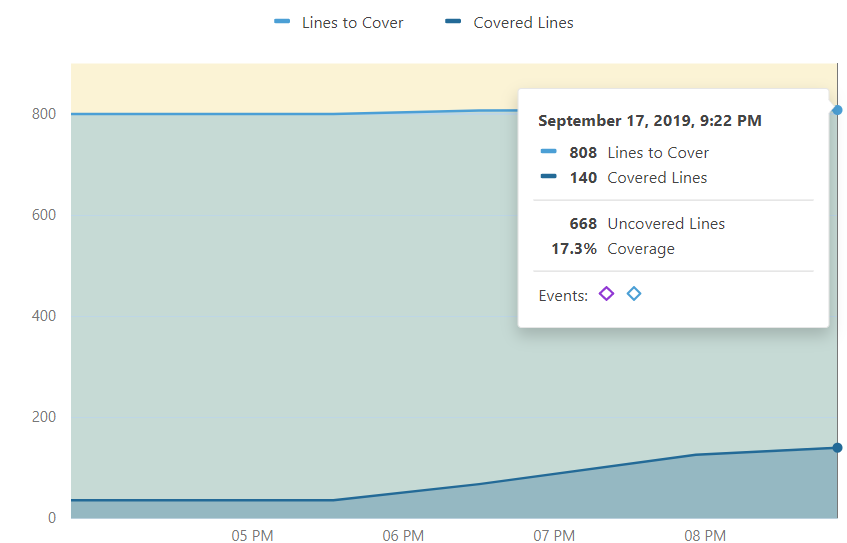


Рис. 17. Графік інформації про покриття коду на кожну ітерацію

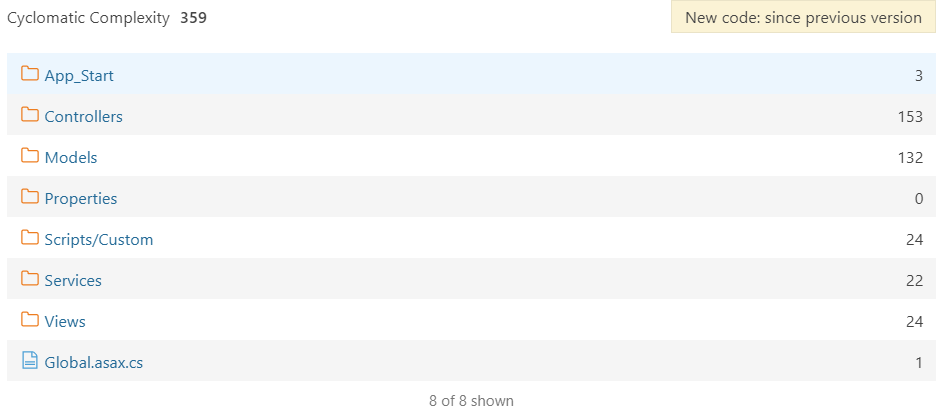


Рис. 18. Інформація про цикломатичну складність проекту

**Висновки**

Навчились аналізувати якість проекту з використанням програмного забезпечення SonarQube. Покращили обраний проект, за допомогою ітераційного аналізу якості виконаного даним ПЗ.