**МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ**

**НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ "ЛЬВІВСЬКА ПОЛІТЕХНІКА"**

Інститут **ІКНІ**

Кафедра **ПЗ**



**ЗВІТ**

До лабораторної роботи № 1

**На тему:** *«Налаштування SonarQube для проекту і обчислення показників якості на основі отриманих результатів»*

**З дисципліни:** *«Управління якістю програмного забезпечення»*

**Лектор:**

ст. викладач

Нитребич О.О.

**Виконав:**

Ст. гр. ПЗІП-12

Голобородько М.В.

**Прийняв:**

викладач каф. ПЗ

Ваврук І.Є.

Львів – 2018

**Тема роботи:** налаштування SonarQube для проекту і обчислення показників якості на основі отриманих результатів.

**Мета роботи:** навчитись застосовувати засоби SonarQube для обчислення показників якості проекту.

**Хід роботи**

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))  
9.    Сформувати звіт.

**Результати виконання**

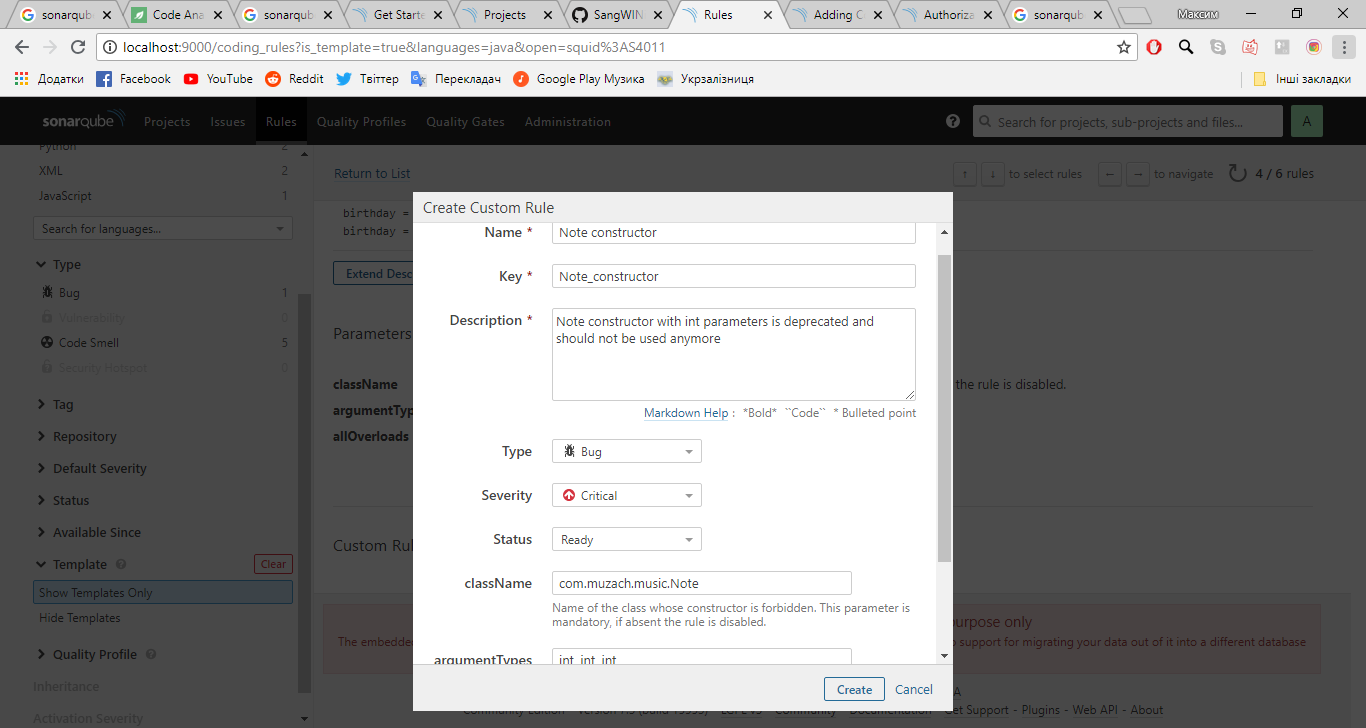
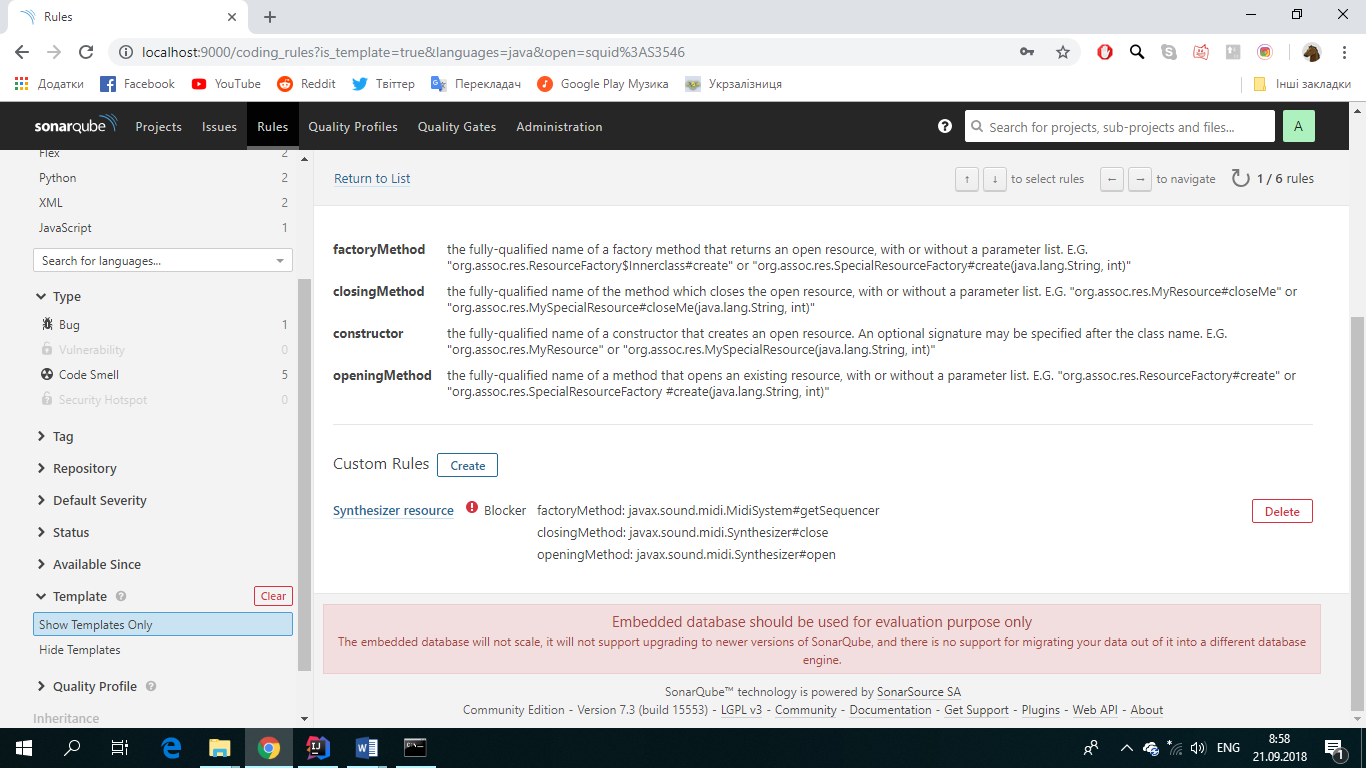
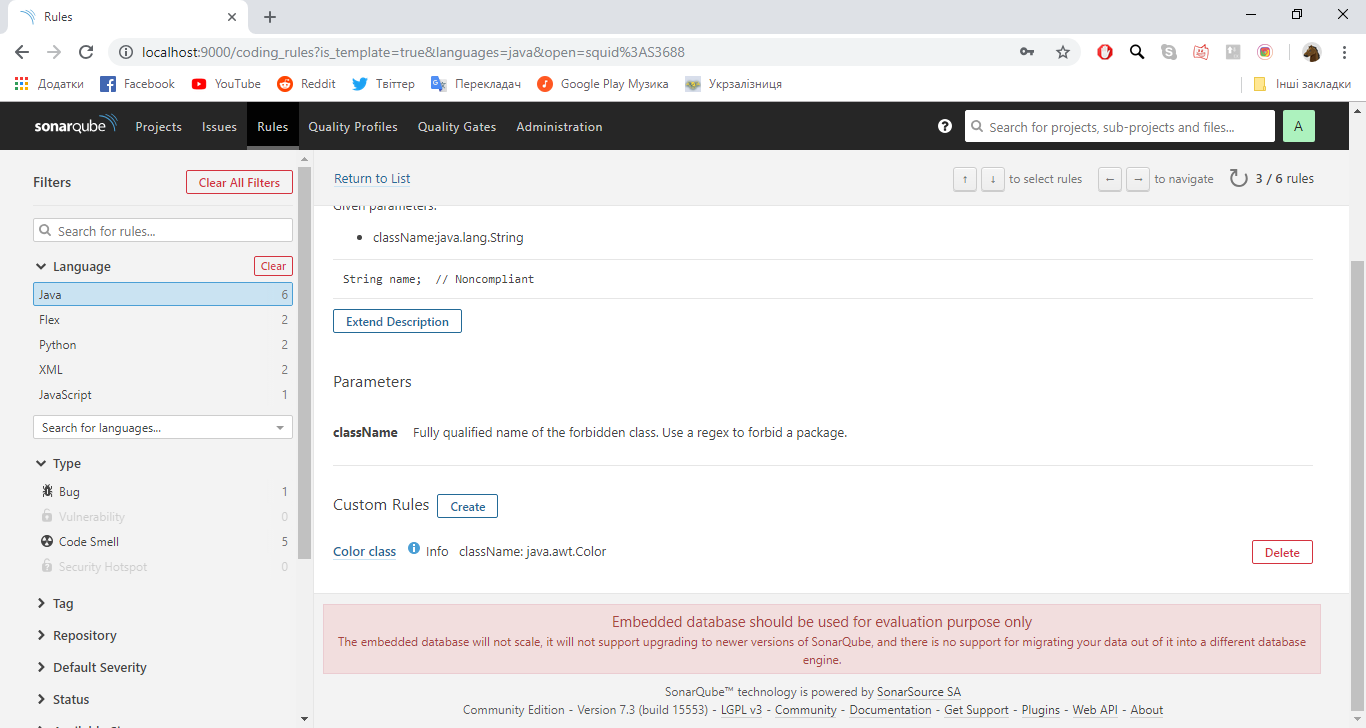
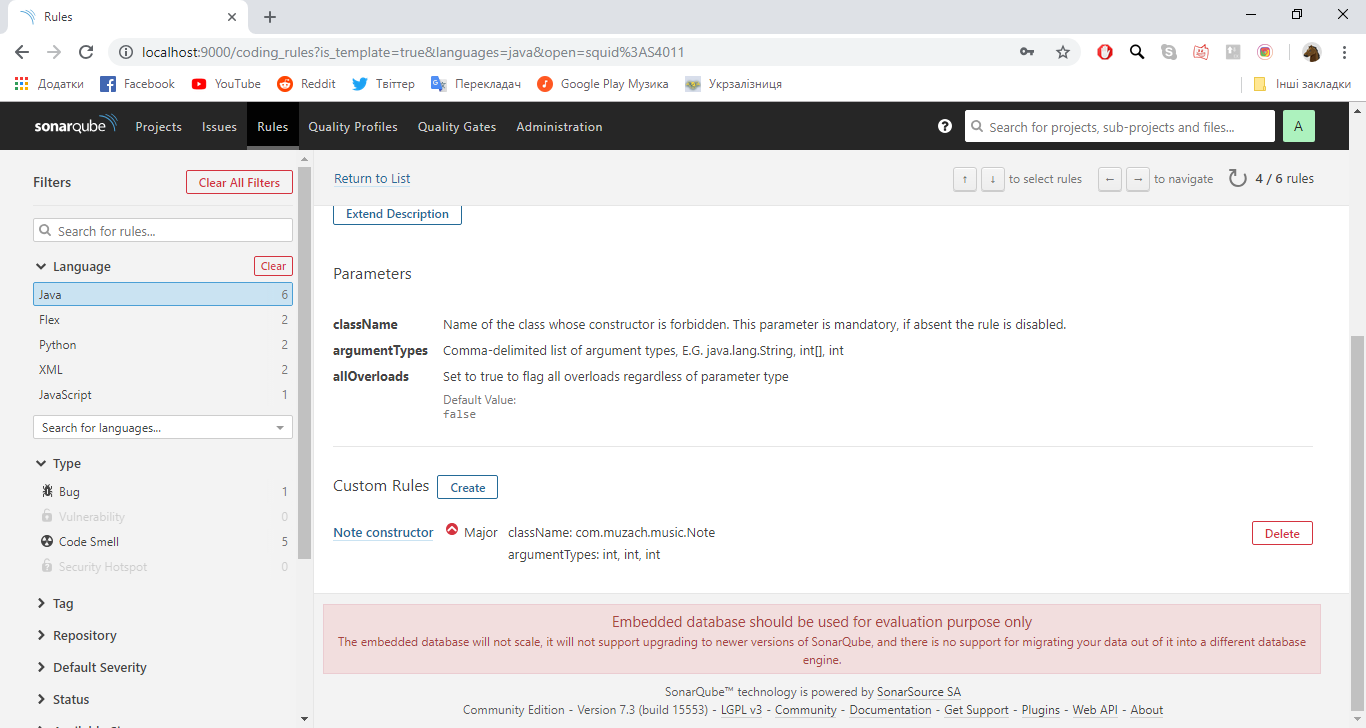
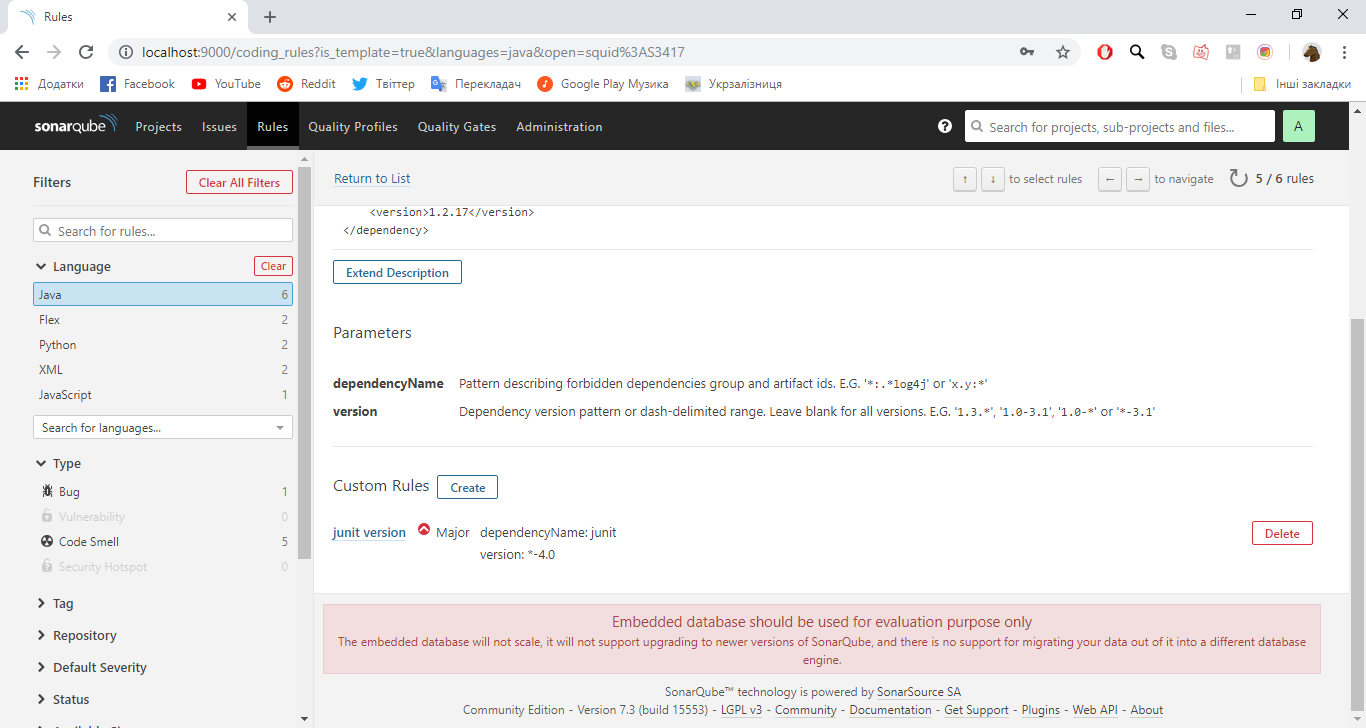


Рис. 1. Створення власного правила перевірки коду









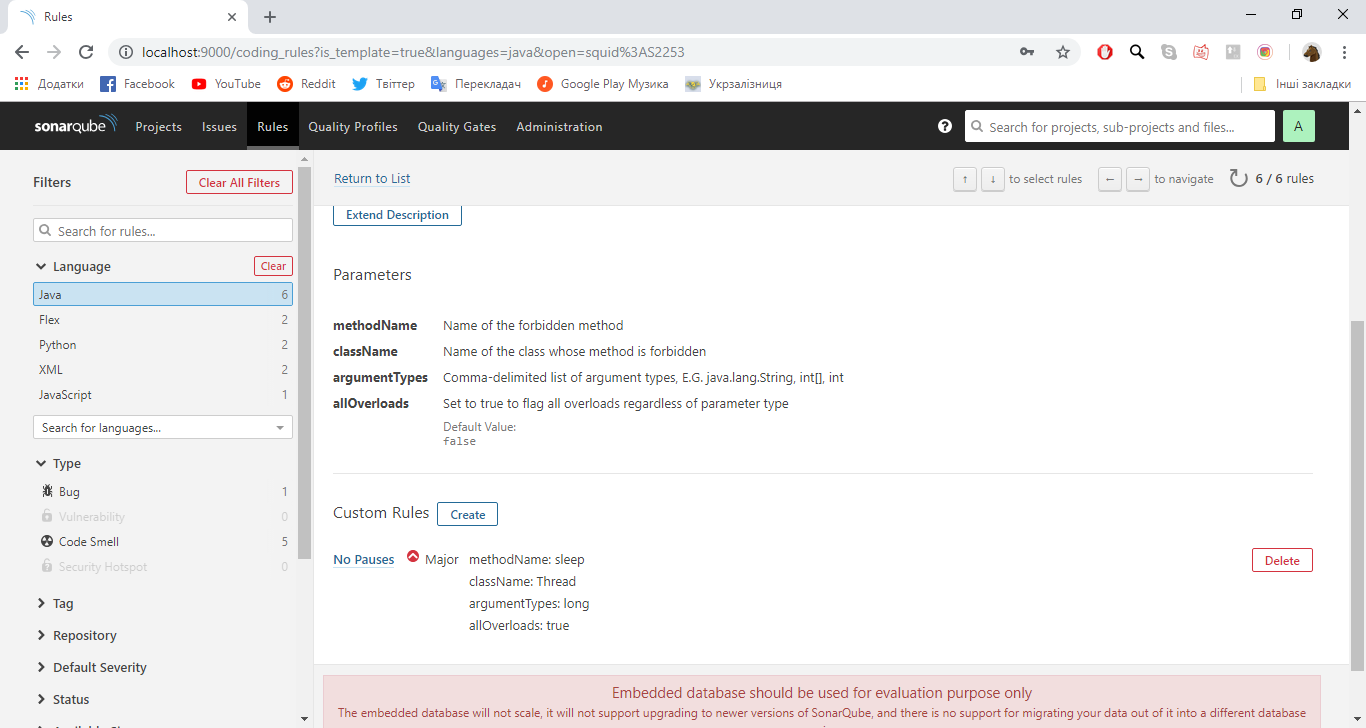


Рис. 2. Створені правила перевірки коду.

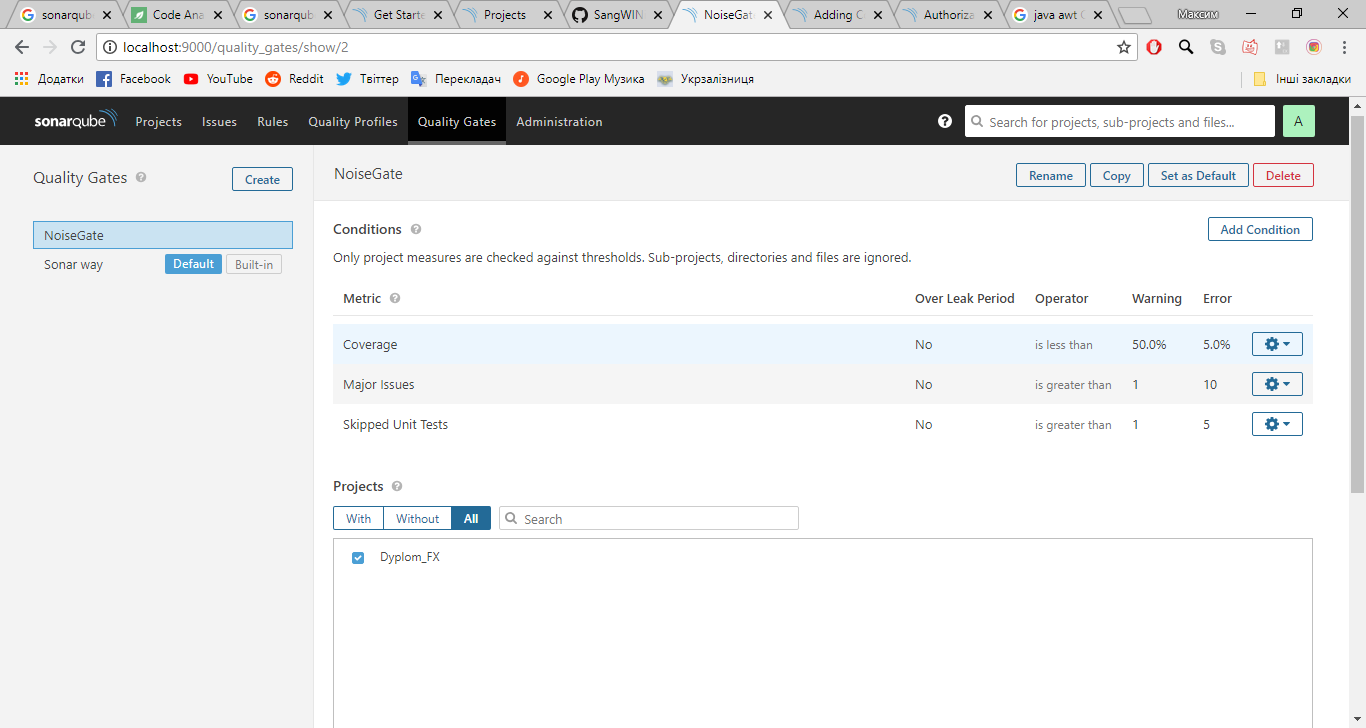


Рис. 3. Налаштування критерію успішного виконання проекту

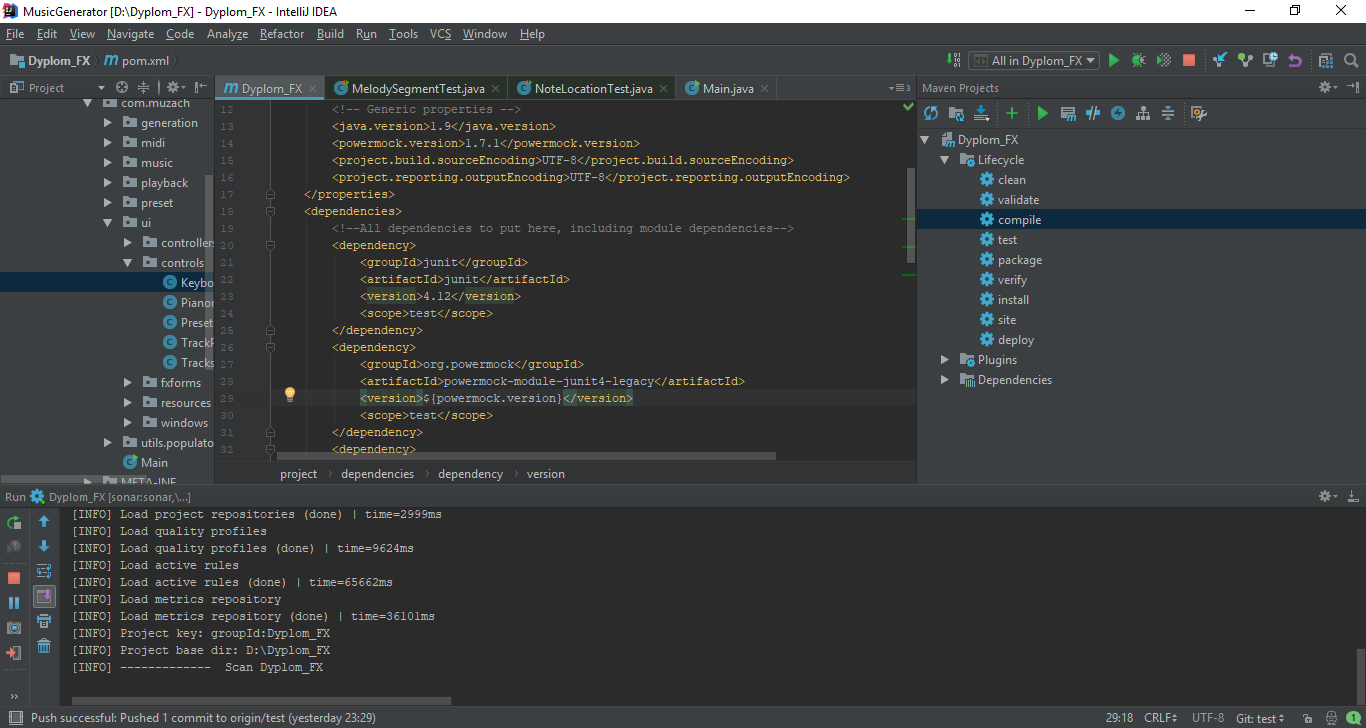


Рис. 4. Оцінка якості проекту з використанням SonarQube

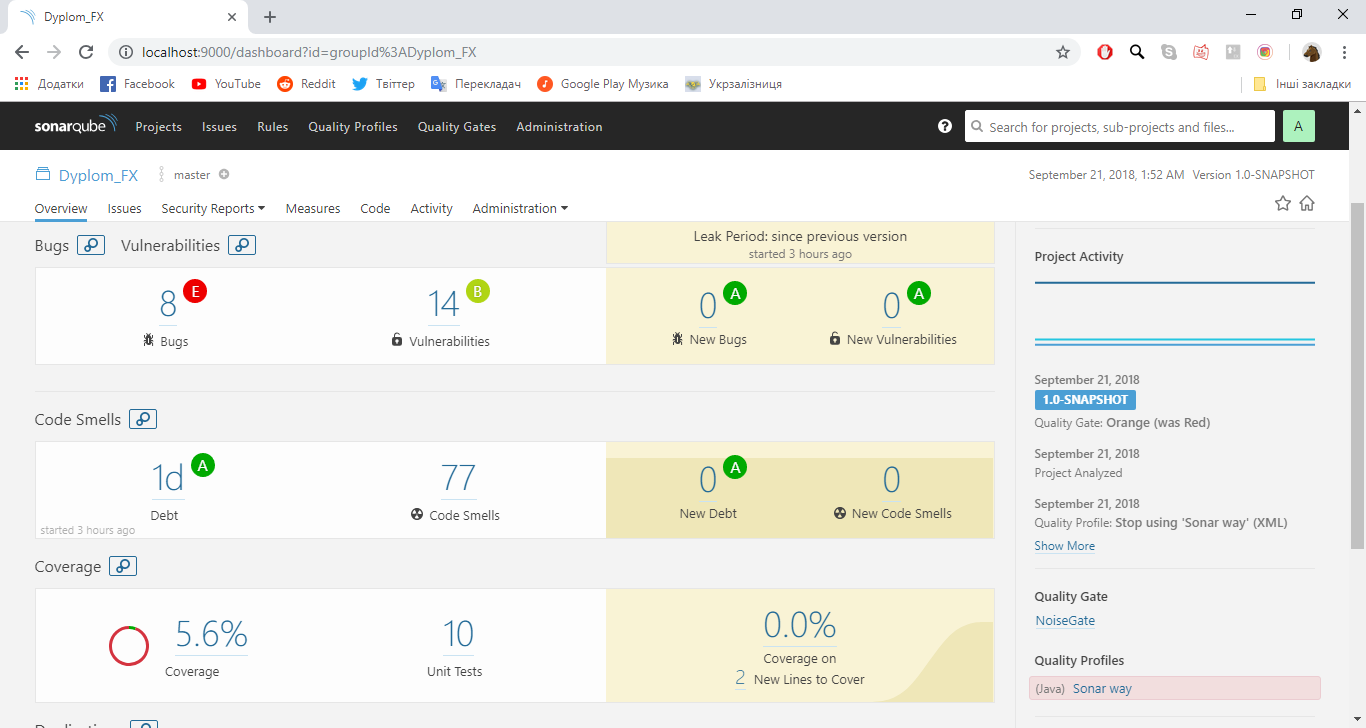


Рис. 5. Оцінка якості проекту до внесення змін

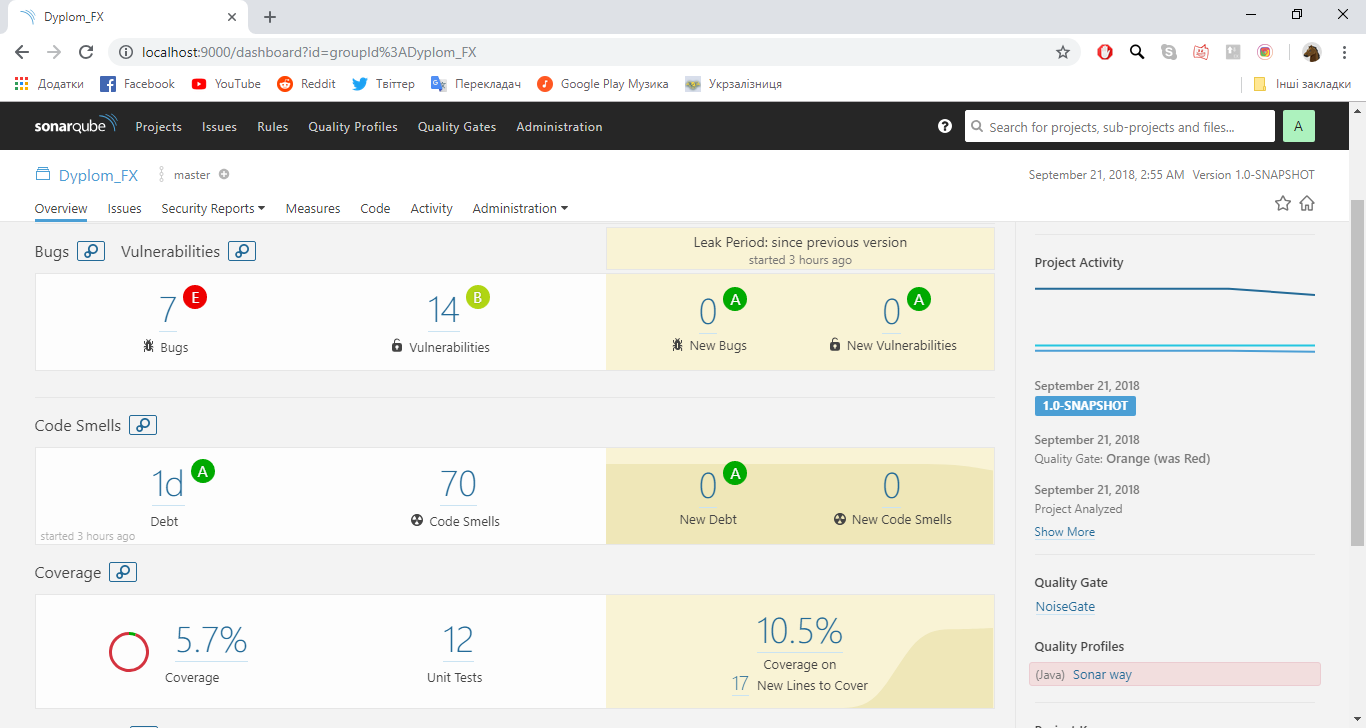


Рис. 6. Оцінка якості проекту після першої ітерації виправлень

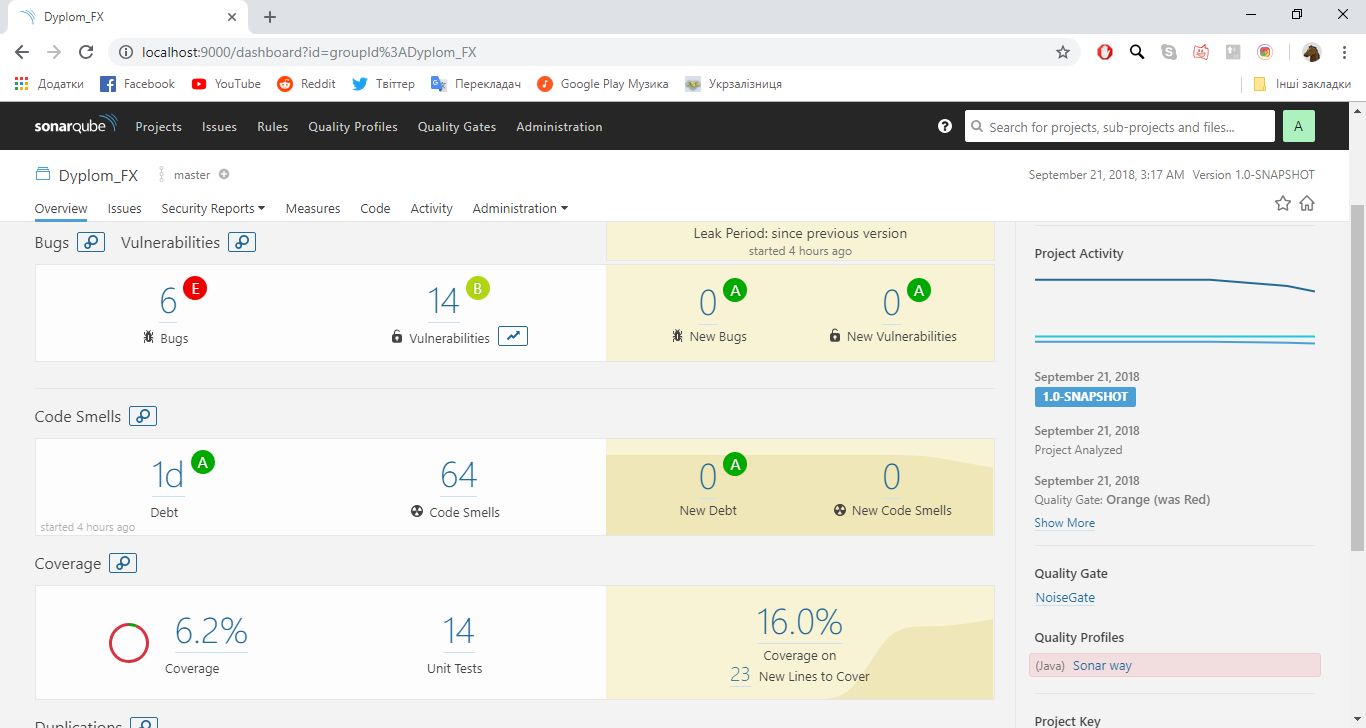


Рис. 7. Оцінка якості проекту після другої ітерації виправлень

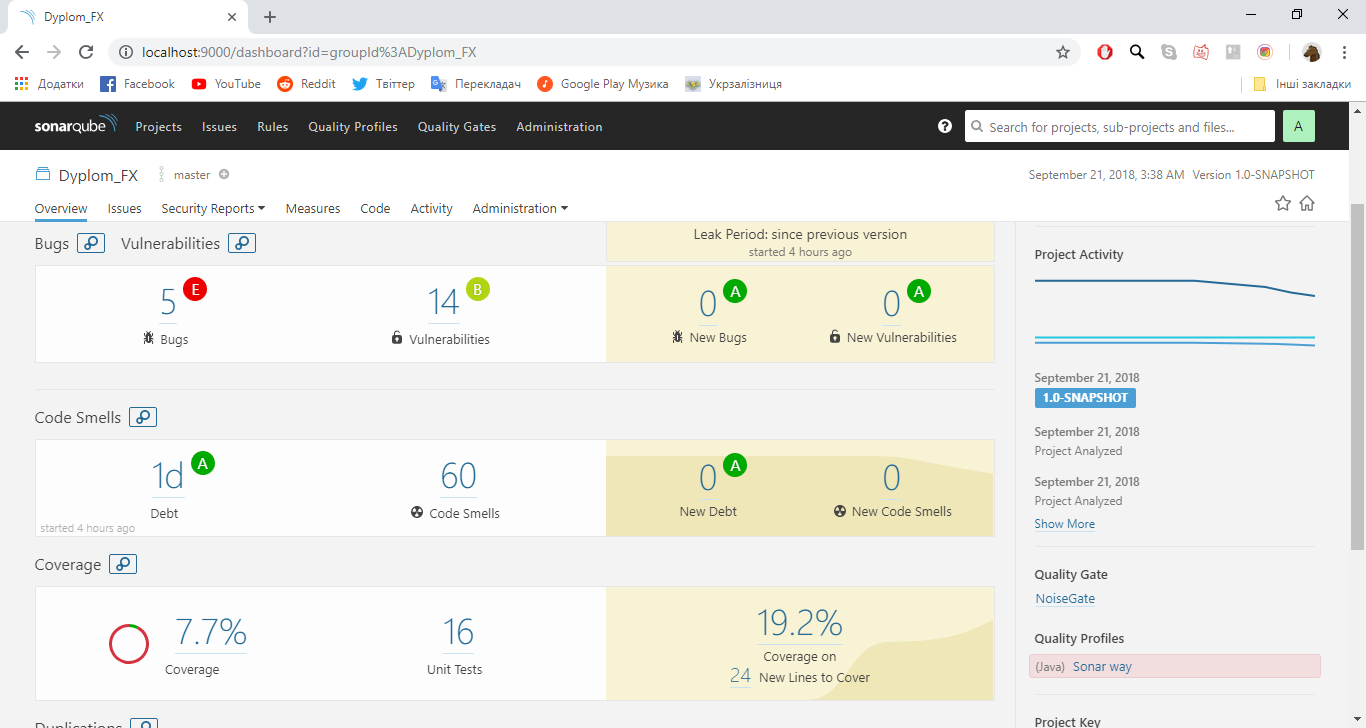


Рис. 8. Оцінка якості проекту після третьої ітерації виправлень

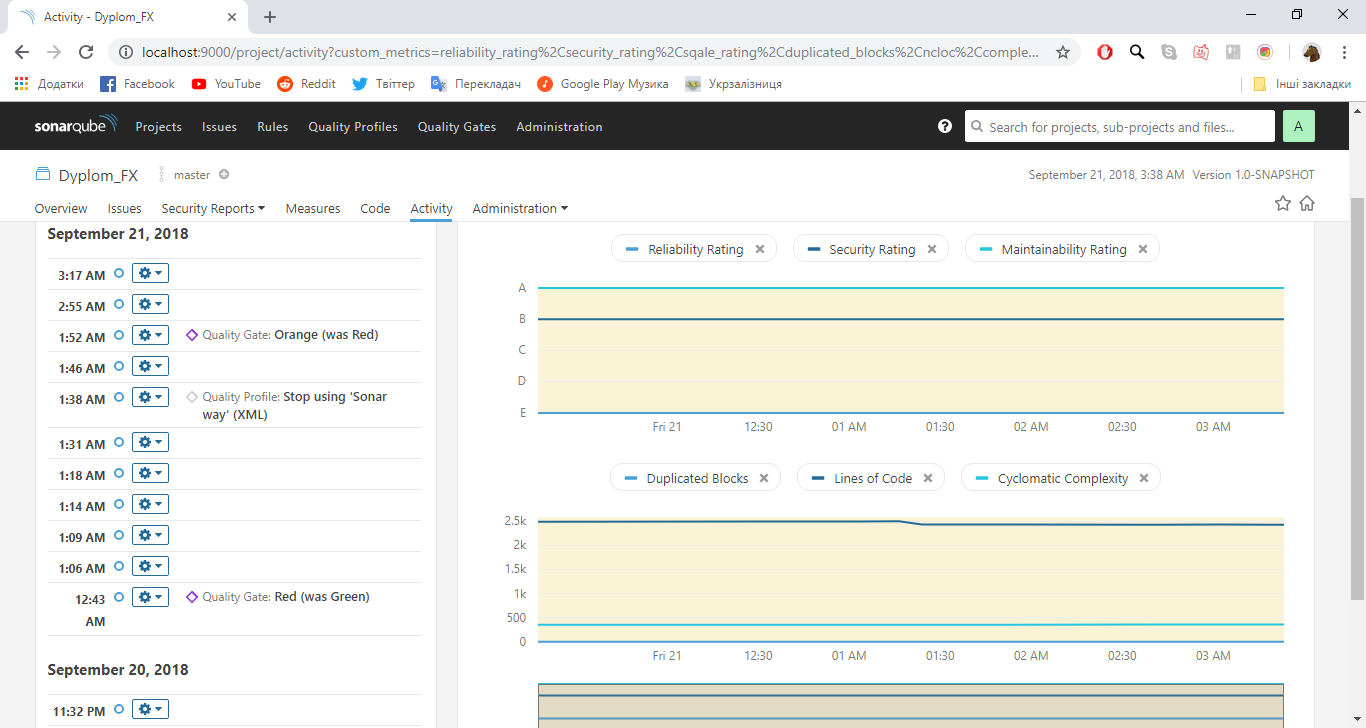


Рис. 9. Статистика покращення якості коду (початок)

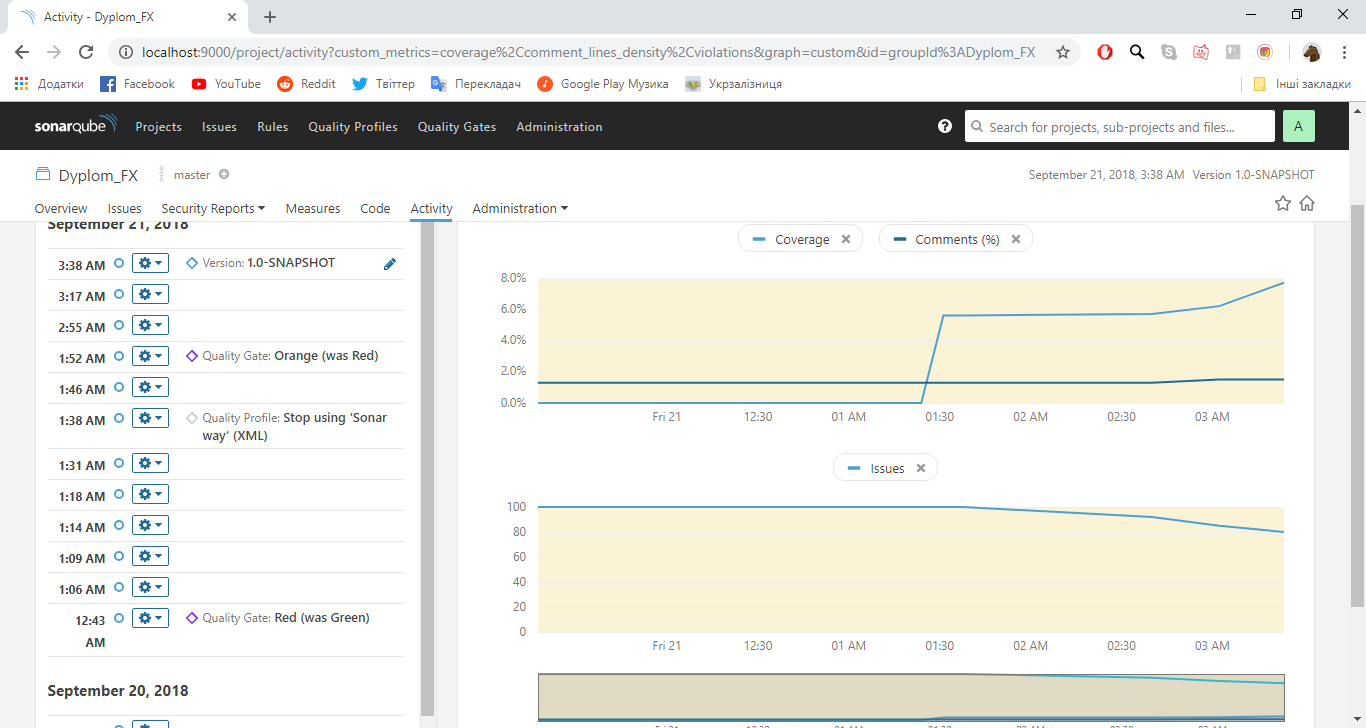


Рис. 10. Статистика покращення якості коду (продовження)

***Налаштування SonarQube через Maven***

<?xml version="1.0" encoding="UTF-8"?>

<project xmlns="http://maven.apache.org/POM/4.0.0"

xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">

<modelVersion>4.0.0</modelVersion>

<groupId>groupId</groupId>

<artifactId>Dyplom\_FX</artifactId>

<version>1.0-SNAPSHOT</version>

<properties>

<!-- Generic properties -->

<java.version>1.9</java.version>

<powermock.version>1.7.1</powermock.version>

<project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>

<project.reporting.outputEncoding>UTF-8</project.reporting.outputEncoding>

<sonar.sources>src/main</sonar.sources>

<sonar.tests>src/test</sonar.tests>

<sonar.java.coveragePlugin>jacoco</sonar.java.coveragePlugin>

<sonar.dynamicAnalysis>reuseReports</sonar.dynamicAnalysis>

<sonar.jacoco.reportPath>${project.basedir}/../target/jacoco.exec</sonar.jacoco.reportPath>

<sonar.language>java</sonar.language>

</properties>

<dependencies>

<!--All dependencies to put here, including module dependencies-->

<dependency>

<groupId>junit</groupId>

<artifactId>junit</artifactId>

<version>4.12</version>

<scope>test</scope>

</dependency>

</dependencies>

<build>

<resources>

<resource>

<directory>${project.basedir}/src/main/java</directory>

<excludes>

<exclude>\*\*/\*.java</exclude>

</excludes>

</resource>

<resource>

<directory>${project.basedir}/src/main/resources</directory>

<includes>

<include>\*</include>

</includes>

</resource>

</resources>

<plugins>

<plugin>

<groupId>org.apache.maven.plugins</groupId>

<artifactId>maven-compiler-plugin</artifactId>

<version>3.5.1</version>

<configuration>

<source>${java.version}</source>

<target>${java.version}</target>

</configuration>

</plugin>

<plugin>

<groupId>org.jacoco</groupId>

<artifactId>jacoco-maven-plugin</artifactId>

<version>0.7.8</version>

<configuration>

<destFile>${sonar.jacoco.reportPath}</destFile>

<append>true</append>

</configuration>

<executions>

<execution>

<id>agent</id>

<goals>

<goal>prepare-agent</goal>

</goals>

</execution>

</executions>

</plugin>

</plugins>

</build>

<profiles>

<profile>

<id>sonar</id>

<activation>

<activeByDefault>true</activeByDefault>

</activation>

<properties>

<sonar.host.url>

http://localhost:9000

</sonar.host.url>

<sonar.login>

6eafb482f61e1040009656c1e5642a4de107a610

</sonar.login>

</properties>

</profile>

</profiles>

</project>

**Висновок**

Виконуючи лабораторну роботу, я налаштував SonarQube для своєї кваліфікаційної роботи, здійснив перевірку якості коду проекту та на основі отриманих результатів здійснив правки та збільшив покриття проекту юніт тестами.