HW1: Mid-term assignment report

Filipe Maia Antão [103470], v2023-04-03

1 Introduction

1.1 Overview of the work

This report presents the midterm individual project required for TQS, covering both the software product features and the adopted quality assurance strategy. The Application works as a tool to search and query the air quality of a given city.

1.2 Current limitations

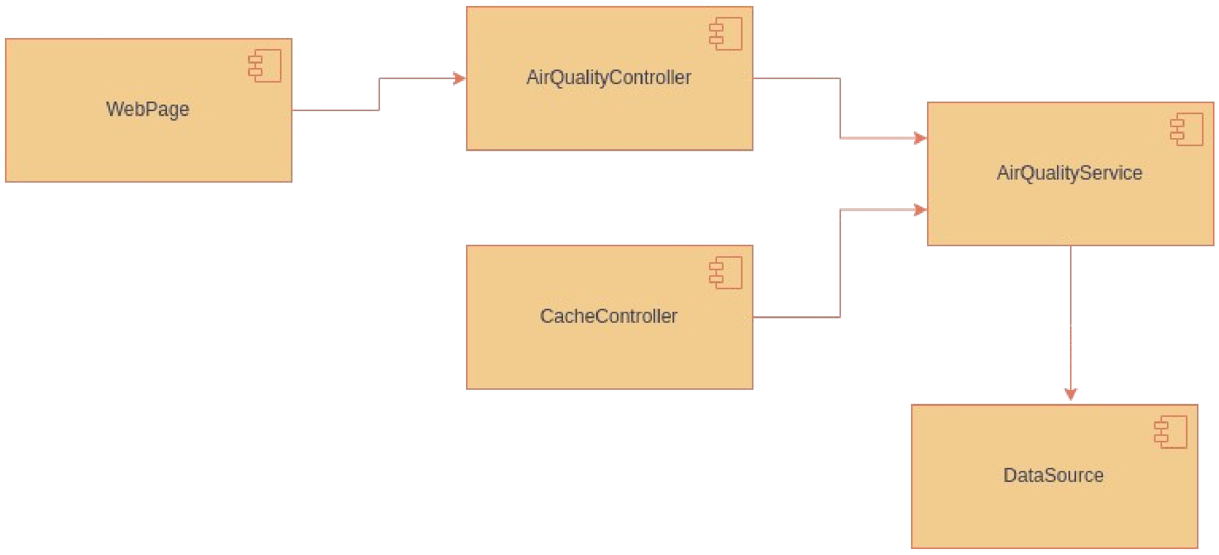
The application works as expected to most cities(upon searching the city in the search bar, values of ozone, pm25 and pm10 for the day in question and next days). However, depending on the city and the station that provides the data, some values may be missing and this application does not deal with those missing values, returning an error page when it happens.

2 Product specification

2.1 Functional scope and supported interactions

This application can be used to query values of air quality of a given city, therefore that are a lot of use cases. A good one would be a person who's going on vacation. He would search the city in the application and would get the values of air quality as well as a brief assessment of the danger.

2.2 System architecture



Technologies used:

- Spring Framework
- Thymeleaf
- External API for the retrieval of the air quality data (https://aqicn.org/api/)

2.3 API for developers

With this project, developers can take advantage of 2 API's that were developed. One relates to the air quality theme that was proposed for the project, the other one provides information relating to some cache statistics.

Air Quality API:

GET */airquality/*?city={city}	return the air quality information of a given city

Cache Statistics API:

GET */cache/*hits	return the number of cache hits
GET */cache/*misses	return the number of cache misses
GET */cache/*requests	return the number of cache requests

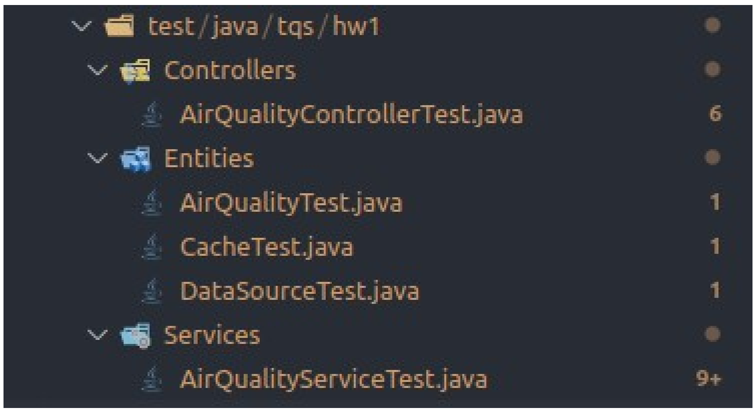
3 Quality assurance

3.1 Overall strategy for testing

The main approach used in this project was Test Driven Development(TDD). This way, the tests were designed before writing the code, this way bugs were minimized.To produce the tests, dependencies such as jUnit and Mockito were used in the maven project.

3.2 Unit and integration testing

Testing was done in all entities of our project(AirQuality, Cache, DataSource), and the service and air quality controller were also tested.



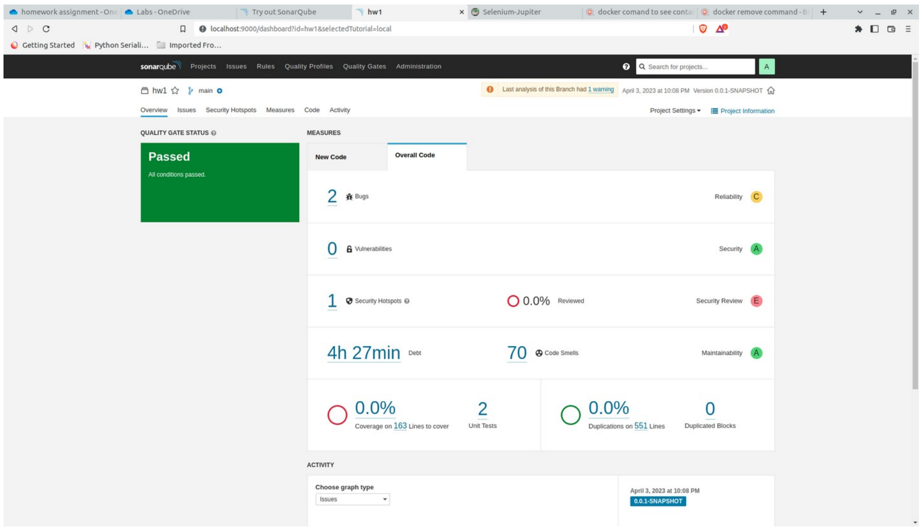
```
test / java / tqs / hw1
  Controllers
    AirQualityControllerTest.java      6
  Entities
    AirQualityTest.java                1
    CacheTest.java                     1
    DataSourceTest.java                1
  Services
    AirQualityServiceTest.java         9+
```

3.3 Functional testing

Functional testing wasn't completed in this project.

3.4 Code quality analysis

To execute the code analysis, I used SonarQube to evaluate my code. Below is a screenshot of the results.



Last analysis of this Branch had 1 warning April 3, 2023 at 10:08 PM Version 0.0.1-SNAPSHOT

QUALITY GATE STATUS

Passed
All conditions passed.

MEASURES

2 Bugs — Reliability C

0 Vulnerabilities — Security A

1 Security Hotspots — 0.0% Reviewed — Security Review E

4h 27min Debt — 70 Code Smells — Maintainability A

0.0% Coverage on 163 Lines to cover — 2 Unit Tests

0.0% Duplications on 551 Lines — 0 Duplicated Blocks

ACTIVITY

April 3, 2023 at 10:08 PM
0.0.1-SNAPSHOT

4 References & resources

Project resources

Resource:	**URL/location:**
Git repository	https://github.com/MrFantao/TQS_HW
Video demo	The video is in the Git Repository in folder docs

Reference materials

Air Quality API : https://aqicn.org/api/