HW1: Mid-term assignment report

Testing and Software Quality

Rafael Remígio 102435

April 7, 2023

Departamento de Electrónica, Telecomunicações e Informática
Universidade de Aveiro

Year 2022/2023

Contents

1 Introduction

1.1 Overview of the work

In this assignment the web application developed provides information about parameters related to air quality of a certain location.

1.2 Current limitations

2 Product specification

2.1 Functional scope and supported interactions

The WebApp provides two use cases:

1. search for a specific location and receive Air Quality information. This information consists of the specific parameters:

    - Particulate Matter PM2.5

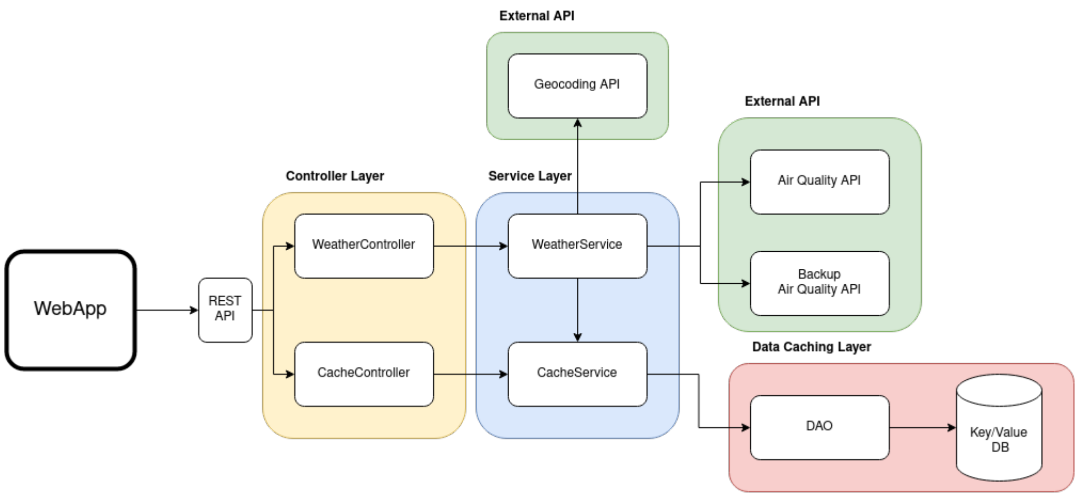
- Particulate Matter PM10
- Carbon Monoxide CO
- Nitrogen Dioxide NO2

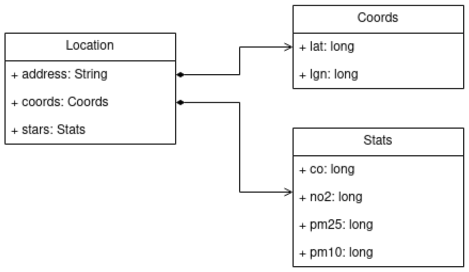
2. access and view statistics of the cache's usage:

- Misses
- Hits
- Acesses

2.2 System architecture

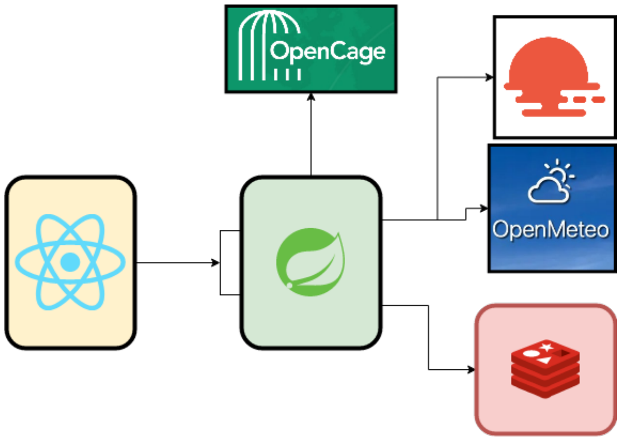
2.2.1 Architecture





2.2.2 Technologies and API's Used

The very basic User Interface was constructed using ReactJS. It interacts with the Backedn througth a RestAPI. The backend was built using Spring Boot. For an in memory database I used Redis. The Geolocation Api chosen was Open Cage. The primary Air Quality API is the OpenWeather API. The backup Air Quality API is the OpenMeteo API.



2.3 API

The API has two endpoints:

1. **GET** /weather?local={*local*}
   Query Params:

   - **Required** - local: Address, Country or City .

   Response Body:

```
{
 "coords":{
          "lat":40.7275536,
          "lgn":-8.5209649
          },
 "stats":{
          "co":150.0,
          "no2":6.5,
          "pm25":9.4,
          "pm10":11.5
          },
 "location":"aveiro"
}
```

2. **GET** /cacheInfo
   Response Body:

```
{
"hits":5,
"misses":10,
"acesses":15
}
```

3 Quality assurance

3.1 Logging

Logging is a crucial aspect of software development that plays a vital role in supporting production and debugging activities.
Logging was conducted using the slf4j Logger class has it provides an easy integration with the Spring Boot Application.
Each log message contains a timestamp, method invocation details, custom log messages, and other contextual data. Additionally, each log entry is assigned a logging level identifier, providing a way to categorize and prioritize log entries based on their significance.
I followed logging principles and best practices - such as: logging at the correct evel, providing meaningfull messages, etc - in order to provide meaningfull logs.

3.2 Overall strategy for testing

3.3 Unit and integration testing

implemented and

3.4 Functional testing

3.5 Code quality analysis

SonarQube

3.6 Continuous integration pipeline

4 References & resources

GitRepository can be found at m̈y github:

4.1 Reference materials