deti universidade de aveiro
departamento de eletrónica, telecomunicações e informática

HW1: Mid-term assignment report

Rodrigo França Lima [98475], v2022-04-26

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 product's name is CoviData, it's an open API for Covid information about specific countries, with an integrated cache system and filtering. It also provides cache metrics and resilient information due to multiple sources of data gathering.

1.2 Current limitations

Currently the product lacks more types of filtering, this is due to the lack of similar api's that can be used as source for the main api. Also the product lacks a responsive and intuitive frontend on the resources page.

2 Product specification

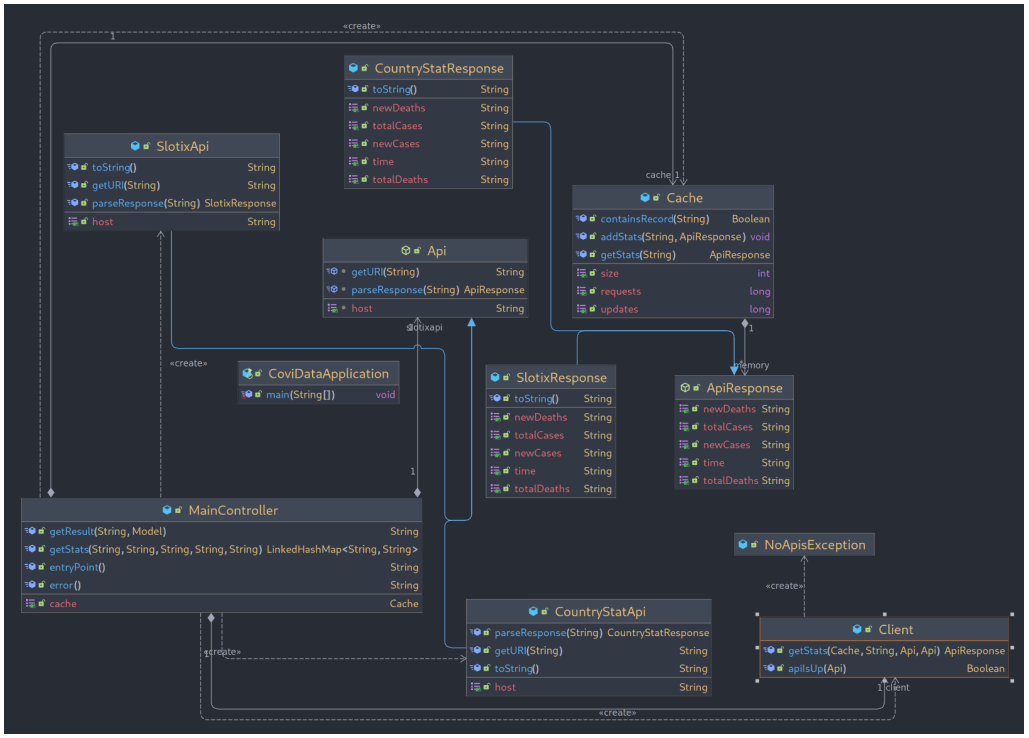
2.1 Functional scope and supported interactions

The application is aimed for users who have interest in knowing the updated information of the situation on covid in a certain country and also for developers who want a consistent and resilient API about Covid in certain countries that they can rely on as a backbone.

2.2 System architecture

The application is built with java and with the spring framework. It was made with maintenance in mind. That means that each api is an object of the type Api that can enable for future migrations of apis in the backend without affecting the main code, this is enabled also by the use of translators that translate parameters of apis to a specific syntax used later on as the main format.

Resilience is also a goal on the project so the application has various api to be used and it checks if an api is up or not, if it isn't it changes to one that is up. The response time is also shortened by the use of a caching system.



2.3 API for developers

The application has two main endpoints for use:

GET	/api/v1/stats/{Country name}	Get covid information about a given country
GET	/api/v1/cache	Get cache statistics, number of requests, updates and size

For filtering of stats results the user can give as argument of the properties they don't want to receive and give as value 0 if they want to omit it.

3 Quality assurance

3.1 Overall strategy for testing

The strategy used on the project development was of TDD, the tests were made first and then with those in mind the implementation was built. This is facilitated by giving a clear view of the end goals of each component of the final project. Also for testing it was used cucumber so that it can be easier to review the code and understand it, by giving specific scenarios and a more verbose explanation of the testing at hand.

3.2 Unit and integration testing

The unit tests were used in various key components of the project. It was used to test the client's behavior. First to check if the client can check if an api is down or not:

```
@Test
@DisplayName("Client returns false if Api is down")
void clientReturnsFalseWhenApiDown() throws IOException, InterruptedException {
    // Country
    String country = "DoesntExist";

    // 1 - instantiate the mock substitute
    SlotixApi slotixapi = Mockito.mock( SlotixApi.class );
    CountryStatApi countrystatapi = Mockito.mock( CountryStatApi.class );

    // 3 - "teach" the required expectations (prepare the mock)
    Mockito.when( slotixapi.getURI( country )).thenReturn( t: "https://doesntexistatallllllllll.com" );
    Mockito.when( slotixapi.getHost()).thenReturn( t: "covid-19-tracking.p.rapidapi.com" );
    Mockito.when( countrystatapi.getURI( country )).thenReturn( t: "https://doesntexistatallllllllll.com" );
    Mockito.when( countrystatapi.getHost()).thenReturn( t: "covid-193.p.rapidapi.com" );

    // SlotixApi broken
    assertThat(client.apiIsUp(slotixapi), is( value: false));

    // CountryStatApi broken
    assertThat(client.apiIsUp(countrystatapi), is( value: false));
}
```

Secondly to check if an api is up if it can get the right information from the caching system:

```java
@Test
@DisplayName("Client gets results from the Cache")
void clientGetsResultsFromCache() throws IOException, InterruptedException, NoApisException, ParseException {
    // Country
    String country = "Portugal";

    // APIs
    SlotixApi slotixapi = new SlotixApi();
    CountryStatApi countrystatapi = new CountryStatApi();

    // Generate static result
    String dateNow = java.time.LocalDate.now().toString();
    SlotixResponse response = slotixapi.parseResponse("{\"error\":false,\"statusCode\":200,\"message\":\"OK\",\"data

    // Add response to cache
    cache.addStats(country, response);

    // Assert API response
    assertThat(client.getStats(cache, country, slotixapi, countrystatapi), is(response));
}
```

Finally to check if the fallback is triggered if all apis are down:

```java
@Test
@DisplayName("Client gives error if both apis are down")
void clientGivesErrorIfBothApisDown() {
    // Country
    String country = "Portugal";

    // Broken APIs mocks
    SlotixApi slotixapi = Mockito.mock( SlotixApi.class );
    CountryStatApi countrystatapi = Mockito.mock( CountryStatApi.class );
    Mockito.when( slotixapi.getURI( country: "DoesntExist" )).thenReturn( t: "https://doesntexistatallllllll.com" );
    Mockito.when( slotixapi.getHost()).thenReturn( t: "covid-19-tracking.p.rapidapi.com" );
    Mockito.when( countrystatapi.getURI( country: "DoesntExist" )).thenReturn( t: "https://doesntexistatallllllll.com" );
    Mockito.when( countrystatapi.getHost()).thenReturn( t: "covid-19-tracking.p.rapidapi.com" );

    assertThrows(NoApisException.class, ()->client.getStats(cache, country, slotixapi, countrystatapi));
}
```

For each api class it was tested if the right URI was given and if the right host and if the api result was parsed correctly:

```java
@Test
@DisplayName("Get right URI for country")
void apiGetURIForCountry() {
    String result = api.getURI( country: "Portugal");
    assertThat(result, is( value: "https://countrystat.p.rapidapi.com/coronavirus/who_latest_stat_by_country.php?count
}

Pengrey
@Test
@DisplayName("Get right Host for API")
void apiGetApiHost() { assertThat(api.getHost(), is( value: "countrystat.p.rapidapi.com")); }

Pengrey
@Test
@DisplayName("Parse result from API to object")
void apiParseResult() {
    CountryStatResponse parsedResponse = api.parseResponse("{\"recordDate\":\"2022-04-16\",\"isoCode\":\"PRT\",\"con
    assertThat(parsedResponse.toString(), is( value: "SlotixResponse{Time='2022-04-14', NewCases='10459', NewDeaths='2
}
```

For the caching system it was tested CRUD operations and checked status responses to various requests:

```java
@Test
@DisplayName("If cache is checked for a country that was added in the same day it should say record is present.")
void cacheCheckTodayExistentStats() throws ParseException {
    String country = "Exists";

    // Create Api and generate static response
    SlotixApi api = new SlotixApi();
    String dateNow = java.time.LocalDate.now().toString();
    SlotixResponse response = api.parseResponse("{\"error\":false,\"statusCode\":200,\"message\":\"OK\",\"data\":{\"

    // Add response to cache
    cache.addStats(country, response);

    // Check cache and assert that the response is saved
    assertThat(cache.containsRecord(country), is( value: true));

    // Check Stats of cache
    assertThat(cache.getSize(), is( value: 1));
    assertThat(cache.getRequests(), is( value: 1L));
    assertThat(cache.getUpdates(), is( value: 1L));
}
```

```java
@Test
@DisplayName("If cache is checked for a country that was added not on the same day it should say record is not prese
void cacheCheckAnotherDayExistentStats() throws ParseException {
    String country = "Shouldnt Exist";

    // Create Api and generate static response
    SlotixApi api = new SlotixApi();
    String dateYesterday = java.time.LocalDate.now().minusDays( daysToSubtract: 1).toString();
    SlotixResponse response = api.parseResponse("{\"error\":false,\"statusCode\":200,\"message\":\"OK\",\"data\":{\"

    // Add response to cache
    cache.addStats(country, response);

    // Check cache and assert that the response is saved
    assertThat(cache.containsRecord(country), is( value: false));

    // Check Stats of cache
    assertThat(cache.getSize(), is( value: 1));
    assertThat(cache.getRequests(), is( value: 1L));
    assertThat(cache.getUpdates(), is( value: 1L));
}
```

```java
@Test
@DisplayName("If its added response to cache, it should save it")
void cacheAddStats() throws ParseException {
    String country = "Portugal";
    // Create Api and generate static response
    SlotixApi api = new SlotixApi();
    SlotixResponse response = api.parseResponse("{\"error\":false,\"statusCode\":200,\"message\":\"OK\",\"data\":{\"

    // Add response to cache
    cache.addStats(country, response);

    // Check if saved result is the same that was passed in
    assertThat(cache.getStats(country), is(response));

    // Check Stats of cache
    assertThat(cache.getSize(), is( value: 1));
    assertThat(cache.getRequests(), is( value: 0L));
    assertThat(cache.getUpdates(), is( value: 1L));
}

Pengrey
@Test
@DisplayName("If cache is checked for an unkown country it should say record isn't present.")
void cacheCheckNonExistentStats() throws ParseException {
    String country = "Doesnt exist";
    assertThat(cache.containsRecord(country), is( value: false));

    // Check Stats of cache
    assertThat(cache.getSize(), is( value: 0));
    assertThat(cache.getRequests(), is( value: 1L));
    assertThat(cache.getUpdates(), is( value: 0L));
}
```

3.3 Functional testing

To test user-facing features it was considered the base example of a simple user using the GUI version to get stats on covid for a given country, this was tested with a scenario in cucumber.

```
Feature: Get covid status for a country

  Scenario: Get status
    When I navigate to "http://localhost:8080/"
    And I choose the country "Portugal"
    And I click Submit
    Then I should be see in the page body the pattern "Updated:  New Cases:  New Deaths:  Total Cases:  Total Deaths: "
```

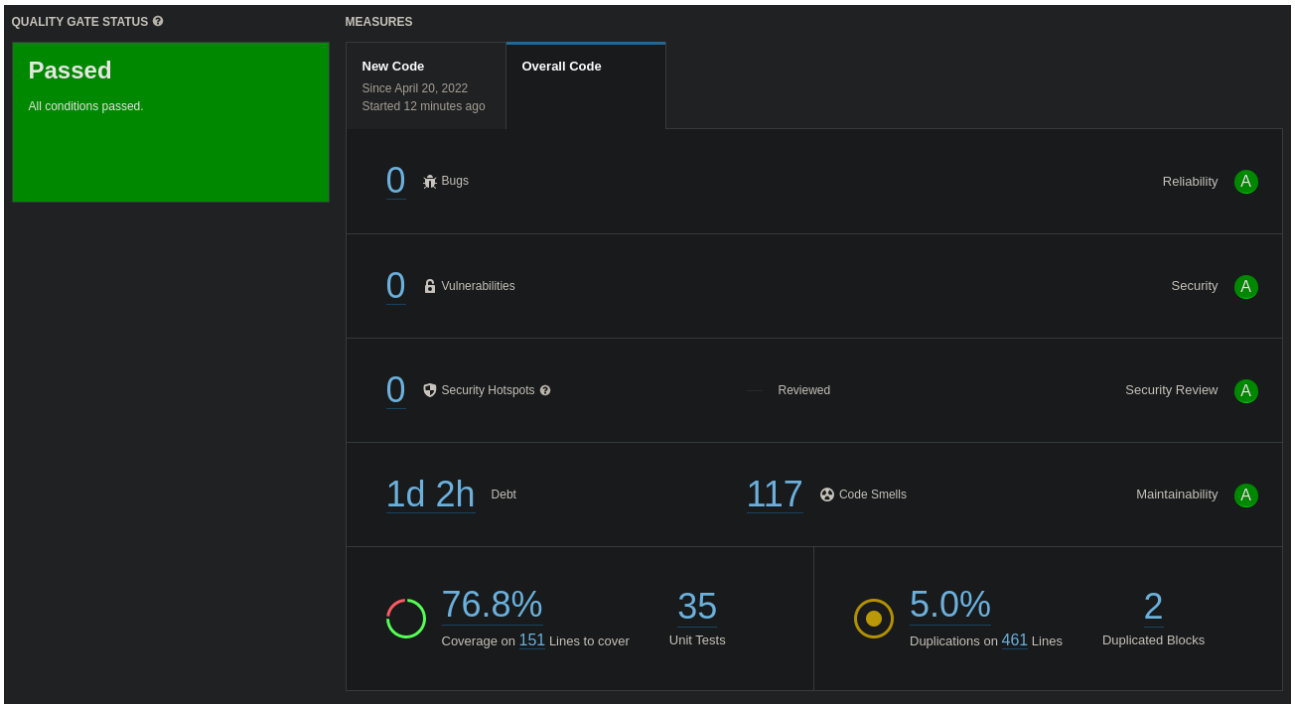
For developer users it was tested the retrieving of api statistics, use of filters and for cache status.

```
@Test
public void whenGoodCountry_returnStats() throws IOException {
    String result = doHttpGet( url: "http://localhost:8080/api/v1/stats/Portugal");
    assertThat(result.replaceAll( regex: "[0-9-]", replacement: ""), is( value: "{\"time\":\"\",\"newCases\":\"\",\"newDeath
}

Pengrey
@Test
public void whenBadCountry_returnNothing() throws IOException {
    String result = doHttpGet( url: "http://localhost:8080/api/v1/stats/Prtugal");
    assertThat(result.replaceAll( regex: "[0-9-]", replacement: ""), is( value: "{}"));
}

Pengrey
@Test
public void whenFilterOutnewCases_getRest() throws IOException {
    String result = doHttpGet( url: "http://localhost:8080/api/v1/stats/Portugal?newCases=0");
    assertThat(result.replaceAll( regex: "[0-9-]", replacement: ""), is( value: "{\"time\":\"\",\"newDeaths\":\"\",\"totalCa
}
```

```
@Test
public void getCacheStatusFromApi() throws IOException {
    String result = doHttpGet( url: "http://localhost:8080/api/v1/cache");
    assertThat(result.replaceAll( regex: "[0-9-]", replacement: ""), is( value: "{\"requests\":,\"updates\":,\"size\":}"));
}
```

3.4 Code quality analysis

For static code analysis it used a CI/CD platform using Codacy Analysis and also it used the SonarQube platform. With the SonarQube analysis it was discovered some code smells at start with the use of various APIs with similar code, this was later refactored so that the code was less duplicated. The report also addressed the fact that some conditions were not tested in filtering, those conditions were later added to be also tested. Altho there was an addition of tests the coverage remained low, this is due to the fact that some tests test two or more things at a time, for example a test to retrieve data from the api tests the parsing and the request.The final SonarQube report is the following:



3.5 Continuous integration pipeline [optional]

```yaml
# This workflow will build a Java project with Maven, and cache/restore any dependencies to improve the workflow execution time
# For more information see: https://help.github.com/actions/language-and-framework-guides/building-and-testing-java-with-maven

name: Java CI with Maven

on:
  push:
    branches: [ hw1 ]
  pull_request:
    branches: [ master ]

jobs:
  verify-hw1:
    runs-on: ubuntu-latest

    defaults:
      run:
        working-directory: HW1/hw1

    steps:
    - uses: actions/checkout@v3
    - name: Set up OpenJDK 11
      uses: actions/setup-java@v3
      with:
        java-version: '11'
        distribution: 'zulu'
        cache: maven
    - name: Build with Maven
      run: mvn -B verify --file pom.xml
    - name: Run Codacy Analysis CLI
      uses: codacy/codacy-analysis-cli-action@master
```

4 References & resources

Project resources

Resource:	URL/location:
Git repository	https://github.com/Pengrey/tqs_98475
Video demo	https://raw.githubusercontent.com/Pengrey/tqs_98475/main/HW1/recording.gif
CI/CD pipeline	https://github.com/Pengrey/tqs_98475/actions

Reference materials

https://github.com/cypress-io/github-action
https://rapidapi.com/slotixsro-slotixsro-default/api/covid-19-tracking
https://rapidapi.com/api-sports/api/covid-193/
https://cucumber.io/
https://github.com/google/gson
https://www.tutorialspoint.com/gson/gson_quick_guide.htm