HW1: Mid-term assignment report

*Pedro Figueiredo [97487]*, v2022-05-02

[1 Introduction 1](#_Toc100228981)

[1.1 Overview of the work 1](#_Toc100228982)

[1.2 Current limitations 1](#_Toc100228983)

[2 Product specification 1](#_Toc100228984)

[2.1 Functional scope and supported interactions 1](#_Toc100228985)

[2.2 System architecture 2](#_Toc100228986)

[2.3 API for developers 2](#_Toc100228987)

[3 Quality assurance 2](#_Toc100228988)

[3.1 Overall strategy for testing 2](#_Toc100228989)

[3.2 Unit and integration testing 2](#_Toc100228990)

[3.3 Functional testing 2](#_Toc100228991)

[3.4 Code quality analysis 2](#_Toc100228992)

[3.5 Continuous integration pipeline [optional] 3](#_Toc100228993)

[4 References & resources 3](#_Toc100228994)

<All remarks like this should be removed from the final document!

This a template for the expected **content/structure**. You may use any editing tool to prepare the report (LaTeX included).

Feel free to write in Portuguese or English, but do not mix languages between headings and body…>

# Introduction

## Overview of the work

The application Covid Data is a simple app designed to provide data of different countries, filtered by day and Country, regarding several covid-19 related metrics.

It allows easy access and visualization of multiple values

## Current limitations

It’s data has limited scope. The cache endpoints are exposed, ready to go, and tested but they still have no frontend access due to time costraints.

# Product specification

## Functional scope and supported interactions

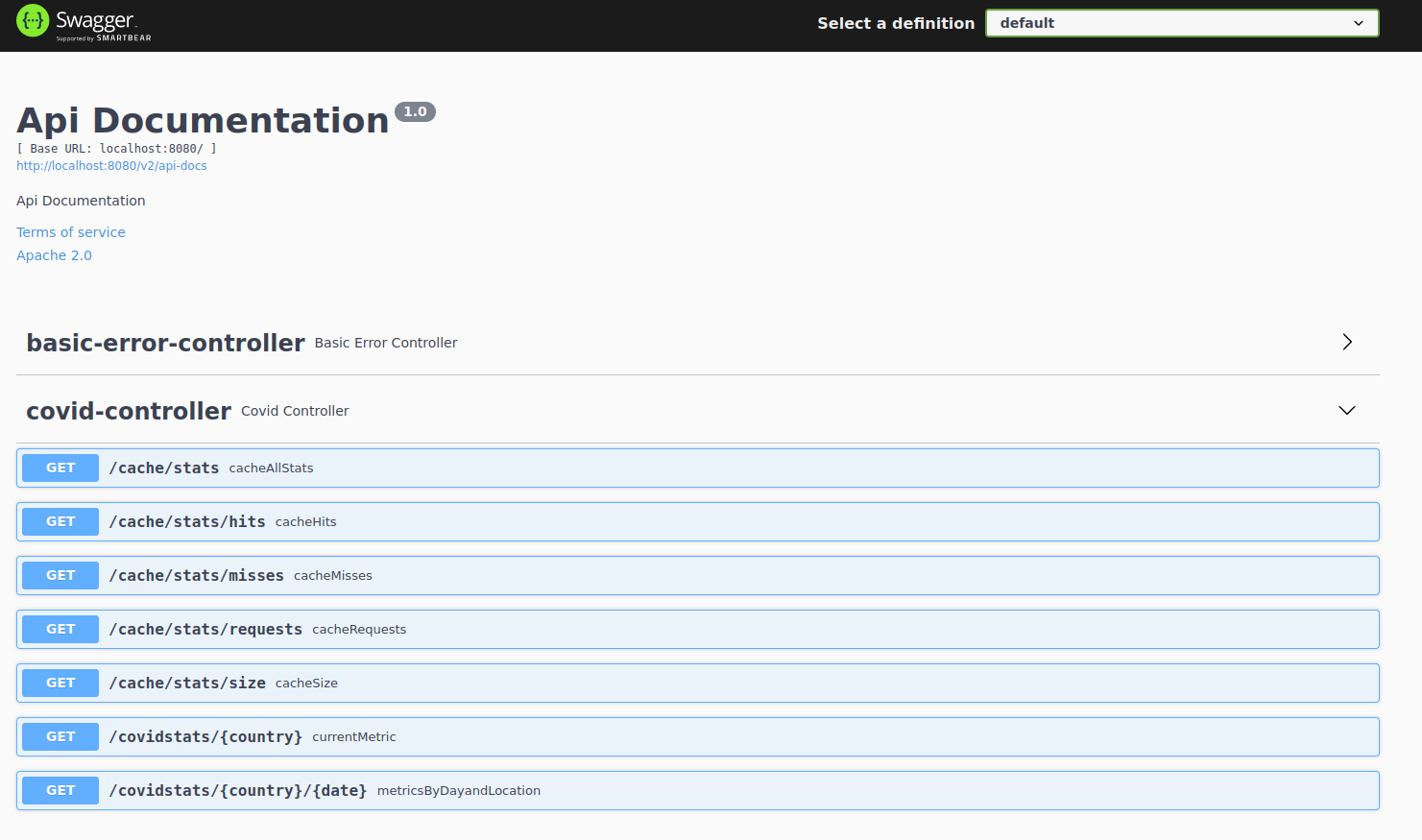
<functional description of the application: who (actors) will use the application and for what? Briefly explain the main **usage scenarios.** >

The main usage scenario considered is that of a standard internet user, who, intending to search for different Covid related metrics, comes to our website to search for information.  
  
For developers looking to examine cache behaviour, there are endpoints, thought not yet exposed.

## System architecture

The architecture is a base layer architecture, consisting of a frontend in base web technologies (HMTL, CSS and JS), then a Spring Boot backend, connected to the John Hopkins API to gather new information.

## API for developers



# Quality assurance

## Overall strategy for testing

In regards to strategy, an attempt was made to perform TDD for the backend, especially the cache section of development. As for the frontend, though we did attempt to implement BDD, we sadly ran out time.

As such, the development on that end was driven by use cases mainly.

## Unit and integration testing

Unit and integration testing were used in the backend REST API development. The strategy used to determine the development was the following, determination of what the application should do, followed by the development of tests to fit the required need. To conclude, we had the actual app development to fit the tests criteria.

## Functional testing

Regarding functional testing, tests were developed to test the application behaviour to respond to user input.

The cases considered include the testing of correct response values.

## Code quality analysis

These were unable to be implemented due to time constraints.

# References & resources

Project resources

|  |  |
| --- | --- |
| **Resource:** | **URL/location:** |
| Git repository | <link to your TQs repo> |
| Video demo | Included in the repository |
| QA dashboard (online) | [**optional**; if you have a quality dashboard available (e.g.: sonarcloud), place the URL here] |
| CI/CD pipeline | [**optional**; if you have th CI pipeline definition in a server, place the URL here] |
| Deployment ready to use | [**optional**; if you have the solution deployed in server, place the URL here] |