Trifonov,Dobri D.V.

Test PLan

Contents

[Testing Strategy 2](#_Toc88852716)

[Context 2](#_Toc88852717)

[Front-End 2](#_Toc88852718)

[Back-End 2](#_Toc88852719)

[Database 2](#_Toc88852720)

[Testing 3](#_Toc88852721)

[Front-End Testing: Cypress 3](#_Toc88852722)

[Back-End Testing: Unit tests and SonarQube 3](#_Toc88852723)

[Database Testing 3](#_Toc88852724)

|  |  |  |
| --- | --- | --- |
| **Version** | **Description** | **Date** |
| 1.0 | Creation of test plan document | *26/11/****2021*** |

# Testing Strategy

## Context

To create a test plan first we need to understand the components of our application

### Front-End

The front end uses React, an open source library, that is used for building single page applications and is compatible with both web and mobile apps. React consists of HTML, CSS and JavaScript. In the project we also implement some libraries/modules that help to provide needed functionality.

### Back-End

The back end uses Spring boot, a Spring framework that is optimized for shortening the code length and providing an easy way to run a Spring application. At it’s core it is Java that is heavily reliant on microservices.

### Database

For the database we use MySQL since it is one of the most popular and reliable SQL databases.

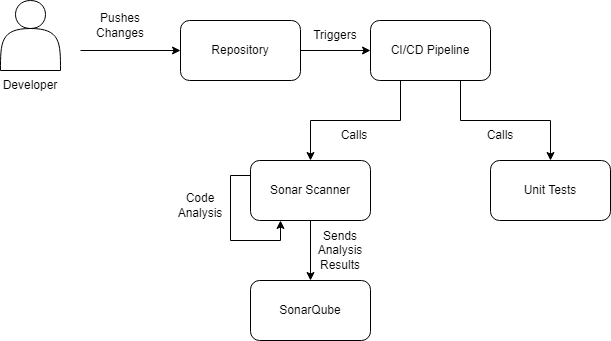
# Testing

## Front-End Testing: Cypress

For the Front end we have decided to use Cypress, since it allows to write automated web tests in JavaScript and has a relatively simple set up process.

## Back-End Testing: Unit tests and SonarQube

The nature of the application makes it difficult to implement Unit testing, since most of it consists of passing objects and variables around, but we will attempt our best to reach 40% code coverage and if possible go for 60%. Even tough there isn’t that much logic, unit tests will ensure that code behaviour remains the same through the application’s life cycle.

SonarQube will also be implemented to ensure the code quality remains. Since SonarQube can detect small bugs and security concerns it will be vital for developing a bug free application.

## Database Testing

There really isn’t a reason to test the database. It’s a tried and tested product that has plethora of inbuilt checks to ensure that data remains consistent.