Contents

[1 Development process 2](#_Toc189349109)

[1.1 Choosing the development model 2](#_Toc189349110)

[1.2 Test 2](#_Toc189349111)

[1.3 Implementation of services 3](#_Toc189349112)

[3.1.1 ProofreadingServiceAPI 3](#_Toc189349113)

[3.1.2 Textpost 3](#_Toc189349114)

[1.4 Implementation of controllers 4](#_Toc189349115)

# Development process

## Choosing the development model

For this project I have chosen to go **with test driven development.** Main reasons for choosing this model are abundance of time and lack of external “Proofreading” service.

After that I have initialized project with all required libraries and started development.

## Test

First I have made all required models and validators. Custom validators were made by implementing ConstraintValidator and they are used like normal validation annotations.

Then I created the **IProofreadingServiceAPI** which will be used for communication with external API. Then, I created the **ITextPolishBL** service, which will have **IProofreadingServiceAPI** as a dependency and it will be used for text processing and similarity check.

All tests were written using Junit and Mockito libraries.

**TextPolishControllerTest** includes mocked version of **IProofreadingServiceAPI** which will always return valid data.

|  |  |
| --- | --- |
| Test name | Expected result |
| polish\_success | success |
| polish\_whenContentHasTags\_success | success with removed tags |
| polish\_whenInvalidLanguage\_fail | Invalid language error |
| polish\_whenInvalidDomain\_fail | Invalid domain error |
| polish\_whenInvalidContentCount\_fail | Invalid Content count error |

## Implementation of services

### ProofreadingServiceAPI

**ProofreadingServiceAPI** uses spring boot`s built in scheduler for scheduling reset for configuration

A computer screen shot of a program code

Description automatically generated

API calls to external service were implemented by using RestTemplate client.

### Textpost

**TextPolishBL** uses IProofreadingServiceAPI as a dependency.

There are two ways for removing tags from text. One is by using regex which is less performant and the other is by using Stacks.

## Implementation of controllers

We only need one controller **TextPolishController** which will have also have only one route polish.

It has only one dependency which is TextPolishBL.

# API documentation

# README

Before starting the project you only need to change **PROOFREADING\_SERVICE\_API\_URL** inside of application.properties.