Analysis Document

Feeler

1 INDEX

2 INTRODUCTION

In this document the second iteration of the Feeler project will be analyzed.

The feeler application is a managing application for book readers. You can use the application to discover new books and keep track of your own bookshelf, which includes books you want to read, books you are reading, and books you finished.

The first iteration of the project was created for the fundamentals course of the 4th semester. It was used to satisfy the set learning goals for the course, but unfortunately did not fully comply with the set standards.

The main problem with the first iteration was that it was not developed following a design process.

This second iteration of the project is developed to correctly implement the design process and any other missing learning goals.

3 PROJECT DEFINITION

3.1 GOAL

The goal of the project is to expand onto the first version of the feeler application and create a more in-debt version of the first application.

3.2 INITIAL CONCEPT

The detailed documentation of the first project can be found here.

When starting this project, a first iteration of the application exists. The concept of the first version of the application consists of the following points:

- The user can explore book based on an emotion.
- The user can read details about books.
- The user can store books they want to read on their bookshelf.
- The user can set the status of books on their bookshelf to reading and read.

These points are the starting-off point of the project and will be further developed.

3.3 FINAL CONCEPT

3.3.1 Stakeholders

This project has the following stakeholders:

- PO
- Developer
- User

3.3.2 Requirements

ID	Name	Description
R01	GUI Language	The application is available in English.
R02	Mobile	The application is optimized for mobile devices.
R03	Book Language	The provided books are written in English.
R04	Installable	The application is installable.
R05	Offline use	The bookshelf is available while offline.
R06	Data Security	Users' data is securely stored (most importantly, passwords).
R07	Account Security	Passwords must be at least 6 characters.
R08	Test coverage	The application has at least 80% test coverage.

3.3.3 User stories

ID	Name	Actor	Description
US01	Account	PO	As the actor I want users to be able to register and authenticate accounts so bookshelves can be implemented for each user. To register, the user must provide a unique username and a password.

US02	Search	User	As the actor I want to be able to search for books by title, so I can use the application for books I found without the application. For the same reason, I would like this search function to also work for a book's ISBN or author.
US03	Delete Account	User	As the actor I want to be able to delete my account, so I know my data will be secure once I am not interested in the application anymore.
US04	Encouragement	PO	As the actor I want users to be complemented when they finish reading a book, so the experience of using the application is more meaningful and they will want to keep reading and using the application.
US05	Random	User	As the actor I want to be able to explore random books, so finding new books to read becomes easier.

4 TECHNOLOGY CHOICE

4.1 FRONTEND

The frontend will be a progressive web application created with React.

4.2 BACKEND

The backend will be a Spring Boot application.

5 QUALITY CONTROL

5.1 TESTING APPROACH

Both the front- and back-end tests are defined in the test plan. For every user story the connected tests must be implemented before it is finished.

Whenever code is pushed to the main branch the project will be build, tests will be run, and a production version will be deployed.

When all the user stories have been implemented a test report is generated.

5.2 VERSION MANAGEMENT

The entire project, including documentation, will be stored using GitHub. Additionally, every part of the documentation will contain a document history containing the version, date, author, and changes made.

5.3 TEST ENVIRONMENT

The testing environment will be deployed on an Ubuntu server on Digital Ocean.

The CI/CD tool that will be used is Jenkins running in a Docker container.

A jenkins pipeline will be created for both the back- and frontend, which will build, test, and deploy the application with use of Docker containers.