Online Social Bookstore Application

Product Backlog Specification

VERSIONS HISTORY

Date	Version	Description	Author
10/7/24	<1.0>	1st version of the requirements definition document	A. Zarras

1 Introduction

The objective of this project is to develop an online social bookstore application that allows individuals to exchange used books for free. The rest of this document is structured as follows. In Section 2, we focus on the development process that shall be followed and other scoring and organizational issues. Sections 3 and 4 provide the Product Backlog, i.e., the "raw" functional and non-functional requirements that should be further analyzed to drive the design, implementation and testing of the application.

2 Functional Requirements / User Stories

2.1 General user stories

US ID	User story
US1	As a user, I want to create a new account, so that I have access to the functionalities of the social bookstore application.
US2	As a user, I want login to my account, so that I have access to the functionalities of the social bookstore application.
US2	As a user, I want to logout from my account, so as to terminate my interaction with the social bookstore application.
US4	As a user, I want to create a profile that includes my full name, address, age, phone number, the categories of books that I prefer (e.g., Art, Comic, Fantasy, Fiction, Biographies, History, Science, Literature, Adventure, Crime, Other) and my favorite authors, to facilitate the search/recommendation of books that may be interesting for me.

2.2 User stories for book offers

US ID	User story
US5	As a user, I want to be able to add a book offer in personal list of book offers with a description that includes the book title, the author(s), the category that the book belongs to (Art, Comic, Fantasy, Fiction, Biographies, History, Science, Literature, Adventure, Crime, Other) and a summary of the book, to facilitate the search/recommendation of books by/to other users.
US6	As a user, I want to browse a list of requests from other users who are interested in a book offer that I have made, so that I can decide to whom I shall give the book.
US7	As a user, I want to select a user who requested a book from my list and notify him that he can take the book. I also want to notify the rest of the users who requested the book that the book has been taken by another user.
US8	As a user, I want to have access to the contact information of a selected user who requested a book that I offer, so that I can contact him to arrange the delivery of the book.
US9	As a user, I want to remove a book that is no longer available from my personal list of book offers and from the book request lists of other users who requested for this book, to enable the accurate book search/recommendation.

2.3 User stories for book requests

US ID	User story
US10	As a user, I want to be able to search for book offers with a certain title and authors to find interesting books. Besides specifying the search criteria, I want to choose between an exact or an approximate search strategy. Then, I want to be able to make a request for a book that is included in the search results to the user who offers the book so that he becomes aware that I want to get the book.
US11	As a user, I would like to browse a list of recommended book offers from the social bookstore application, to find interesting books. It would be nice to be able to choose among various recommendation strategies that consider information given in my profile. Then, I want to be able to make a request for a recommended book to the user who offers the book so that he becomes aware that I want to get the book.
US12	As a user, I would like to remove a book request so that the user who offers the book knows that I am no longer interested in this book.

3 Non-Functional Requirements

[NF1] Maintainability: In software engineering, maintainability is the degree of effectiveness and efficiency with which a product or system can be modified by the maintainers. In the case of this project, we specifically focus on the following concerns:

- **[NF1.1]** A first concern is to employ an architecture that promotes low coupling and high cohesion. To this end, you can employ **Fowler's enterprise application architecture patterns** [2] that allow to clearly separate the different parts (views, controllers, domain model, database) of the application and facilitate the mapping of the domain model to the underlying database schema.

- **[NF1.2]** In the long term, we want to be able to **easily extend** and **configure** the application with **different book recommendation strategies.** The initial set of **recommendation strategies** that will be supported by the application should include the following strategies.

  - The first strategy recommends a list of books that belong to the categories the user prefers, as specified in the user's profile.

  - The second strategy recommends a list that contains books written by the user's preferred authors, as specified in the user's profile.

  - The third strategy is a composite strategy that combines the first two - it recommends a list of books written by the user's preferred authors and books that belong to the categories the user prefers, as specified in the user's profile.

- **[NF1.3]** In the long term we want to be able to easily **extend** and **configure** the application with **different search strategies, without having to change the code of the application.** We would also like to **avoid duplicate code** that may result from the implementation of similar, but different strategies. The initial set of **search strategies** that will be supported by the application should include the following strategies.

- The first (exact) strategy should produce a list of books such that the title and the authors of each book match exactly the book title and authors, specified in the query.
- The second (approximate) strategy should produce a set of books such that the title of each book contains the title specified in the query, and the authors list of each book includes at least the authors, specified in the query.

HINT To achieve the [**NF1.2] and [NF1.3]** concerns the application should be designed according to well-known **principles** and exploit **best practices** like the **GoF design patterns. Specifically, consider using the GoG strategy pattern and the GoF template method pattern for the implementation of alternative/interchangeable strategies.**

[NF2] Usability: In software engineering, usability concerns the ease of use and learnability. In the context of this project the application should provide a simple and user-intuitive interface. The application should also provide help, in the form of user guidelines, concerning its main functionalities of the application.

4 Technical Requirements/Constraints/Recommendations

Following, there is a list of technologies, frameworks and tools to consider for the development of the project.

- Java
- Spring Boot
- MySql
- Junit, Mockito
- Eclipse, Intellij
- PlantUML

5 References

[1] Erich Gamma, Richard Helm, Ralph Johnson, and John Vlissides. *Design Patterns: Elements of Reusable Object-Oriented Software.* Addison-Wesley.

[2] Martin Fowler. Catalog of Patterns of Enterprise Application Architecture. Addison-Wesley. https://martinfowler.com/eaaCatalog/