THOMPSON RIVERS UNIVERSITY

SENG 3210– Applied Software Engineering

Inked

Isaac Latta T00239008

Matia Landry T00707575

Date submitted

Table of Contents

- The table of contents should be automatically generated by selecting “References/ Table of Contents”. Remember that the table of contents should not have an entry of the “Table of Contents” itself.
- Proofread the text for typing and grammar mistakes.
- Follow the IEEE Bibliography style for the references by selecting "References/ Citations & Bibliography/ Style".

List of Figures

List of Tables

1 Introduction

Inked is a book recommender system that addresses a growing need for quick, reliable, and user-friendly book recommendations among students and instructors. Traditional methods—such as emailing lists or relying on word of mouth—are often inefficient, lack personalization, and do not scale well as participation increases. This report details a design for a Recommender System to aid individuals in discovering, rating and managing their book recommendations. Ultimately, the system fosters a community-driven reading culture, where students, instructors, and other stakeholders can collectively benefit from shared knowledge and ongoing book discovery.

The proposed system will integrate an external API to retrieve book titles and cover images while relying on a separate database to store and manage user-related data. The client side will use an intuitive Android interface, that will allow users to view, submit and search books and book recommendations. Administrators will be able to perform tasks such as adding, editing, or removing book entries. Creating a layered and modular system that ensures secure storage of user credentials and allows for seamless feature addition, as well as easy scalability for future developments.

Following the introduction, the report will discuss the project requirements, specific solution details and the final chosen architecture of the system. The report will then examine the environmental, societal, safety, and economic considerations, as well as present an overview of the team’s work distribution, meeting summaries, and proposed future enhancements.

2 Design Problem

Traditional methods of compiling recommended reading lists—such as email threads or scattered word-of-mouth suggestions—often lead to disorganized information, duplication of effort, and limited participation. In an environment where both students and instructors benefit from timely, relevant book recommendations, an online platform is needed to streamline the process of discovering, rating, and sharing book suggestions. Additionally, administrators require straightforward tools for managing the book database. The Book Recommender System addresses these needs by providing a centralized solution that merges real-time book information with secure user data handling.

2.1 Problem Definition

Design and implement a mobile Book Recommender System that allows users to search for books, submit and view recommendations, and update ranking information in real time. The system must include administrative capabilities for adding, editing, and removing books. By combining external APIs for up-to-date book details and a secure database for user data, the project will deliver an accessible, efficient, and scalable approach to book discovery.

2.2 Design Requirements

This section will outline the functional and non-functional requirements, as well as the objectives.

2.2.1 Functional Requirements

The Inked application is a mobile-based book recommendation system designed to facilitate seamless book discovery and rating. The functional requirements define the specific capabilities the system must provide to ensure its core functionality.

1. Book Search & Filtering
    - Users can search books by title, author, or genre.
    - Filtering options include rating, popularity, and latest additions.
2. Book Recommendation & Rating System
    - Users can rate books they have read.
    - Users can recommend books to other users on their friends list.
    - Users can preview books.
3. Real-Time Dashboard
    - Users can view their reading history and save books.
    - Displays a top 10 most recommended books list based on aggregate user ratings.
    - Updates automatically as new ratings are submitted.
4. Book Management (Admin Only)
    - Administrators can add, edit, and delete books from the system.
    - Each book entry includes a title, author, genre, description, cover image, and rating.
5. User Profile & Activity Tracking
6. Notifications & Alerts
    - Users receive alerts for new book additions and trending books.
7. Groups and Friend Lists
    - Users can search and add other users as friends.

    - Users can create reading groups
    - Reading groups maintain internal book recommendations.
8. User Authentication & Role Management
    - Users must be able to sign up and log in using an email and password.
    - Role-based access control:
        - Regular Users: Search for books, rate books, and receive recommendations.
        - Administrators: Add, edit, and remove books from the reading groups.

2.2.2 Objectives

The Inked application is designed to provide an interactive, community-driven book recommendation experience. It aims to achieve the following objectives:

1. Enable Dynamic Book Discovery
    - Integrate Google Books API or Open Books API for real-time book search.
    - Allow users to search for books by title, author, or genre, with filtering options
2. Facilitate Social Book Recommendations
    - Users can add friends within the app.
    - Books can be recommended directly to friends and shared within reading groups.
3. Support Collaborative Reading Groups
    - Users can create and join reading groups.
    - Each group will have its own book collection, managed by group admins.
    - Group members can rate, discuss, and recommend books to the group.
4. Enhance User Experience with Personalization & Tracking
    - Users can save books to their personal reading lists.
    - A reading history feature will allow users to track past reads and recommendations.
5. Implement Role-Based Book Management
    - Regular users can recommend books, but only reading group admins can add or remove books from their groups.
    - Users with admin roles in reading groups will have management rights over group book lists.
6. Create a Seamless & Engaging User Experience
    - The real-time dashboard will showcase saved books and search menu.
    - Users will receive notifications for book updates, friend recommendations, and group activity.
7. Ensure Secure & Scalable User Authentication
    - Users will be required to sign up and log in securely via email authentication.
    - The platform must support role-based access control (Regular Users vs. Reading Group Admins).
8. Prioritize Performance, Scalability, and Security
    - The system should efficiently handle multiple users and concurrent interactions without performance degradation.
    - All user data and recommendations should be securely encrypted.

2.2.3 Non-functional requirements and constraints

1. Performance
    - The application must meet standard mobile app performance expectations
    - Startup launch should take less than 2 seconds
    - The user interface should respond immediately (usually within 100–200 ms) to taps, swipes, and other interactions to feel fluid. Navigation between screens should be smooth, with minimal loading or blank screens.
    - performance-heavy tasks should be optimized or performed in the background.
    - Whenever possible, data transfers should be optimized and compressed to improve loading times and reduce data usage.

2. Security & Data Integrity
    - Implement user authentication and privacy mechanisms.
    - Ensure data protection against unauthorized modifications.

3. Modifiability
    - The system should be designed to allow easy integration of new UI components with minimal development effort.
    - Modules have compact simple tasks that they will be in charge of executing
4. Compatibility
    - The application must be compatible with Android devices running API level 19 (KitKat) or higher

3 Solution

In this section, we present three proposed solutions derived from our initial brainstorming sessions. The first two are early concepts, while the final solution incorporates the most suitable elements to address all requirements and constraints.

3.1 Solution 1: Single app with local storage

Write a brief description of your first solution and provide the reasons for not selecting this one.

You can use the component diagram, sequence diagram, and class diagram.

3.2 Solution 2: External Books API

This solution is an improved solution but might not be the final solution that you select. Give a brief description of this solution here.

You can use the component diagram, sequence diagram, and class diagram.

3.3 Final Solution: Hybrid Model

This solution is the final solution. Explain why it is better than other solutions. You may use a table for comparison purposes. After providing the reason for selecting this solution, detail it below.

3.3.1 Features and the software architecture

Discuss all the features of your final solution. Describe the functionalities of the top-level components and how they will be used for enabling those features. The product features may be tabulated (with a title) for improved comprehension. Use component diagrams to model the internal structures (i.e., sub-components or second-level components) of two major components. Describe the functionalities of the sub-components and the interactions (e.g., the interfaces) between the sub-components. Explain the interfaces between the top-level architecture and the internal structures (i.e., explaining how the internal structures interact with other top-level components).

3.3.2 The system interfaces

Describe the temporal events (i.e., the time-triggered events) and the signal events (i.e., events received from external components) for the intended application. Describe the expected response of the system to each event.

3.3.3 The user interface design

Design the user interface components. Describe the user interface components, the possible business events, and the responses to the triggered events.

3.3.4 The requirements traceability matrix

List the system's requirements and map the requirements to the corresponding design component, code component (e.g., java class file or XML configuration file), and the required testing scenario.

3.3.5 Environmental, Societal, Safety, and Economic Considerations

Explain how your design project considered environmental, societal, and economic considerations. It may include how your implementation has positive contributions to the environment and society. What type of financial decisions did you make? How did you make sure that the implementation is safe to use?

3.3.5.1 Environmental considerations

Explain how your design project considered environmental considerations.

3.3.5.2 Societal considerations

Explain how your design project considered societal considerations.

3.3.5.3 Safety considerations

Explain how your design project considered safety considerations.

3.3.5.4 Economic considerations

Explain how your design project considered economic considerations.

3.3.6 Limitations

Every product has some limitations, so is the case with your design solution. Highlight some of the limitations of your implementation here.

4 Teamwork

Since this is a group project, you must have a fair distribution of tasks among yourselves. To this end, you must hold meetings to discuss the distribution of tasks and keep track of the project progress.

4.1 Meeting 1

Time: February 28, 2025, 7:30pm

Agenda: Brainstorming session; determine project

roles, and initial app design

Team Member	Previous Task	Completion State	Next Task
Isaac Latta	N/A	N/A	- Implement book api(e.g. Open Books/Google Books) - Setup Database
Matia Landry	N/A	N/A	- Design and Implement login page

Outcome: Isaac will begin setting up the database. We have narrowed the database choices down to two options.

1. Firebase
    a. Simpler and directly usable within android studio
    b. Simple JSON storage
    c. Can hande email authentication directly
2. Amazon RDS with custom EC2 backend
    a. A Postgresql instance can be hosted using using AWS services (free micro instance)
    b. A separate AWS instance will manage JWT authentication and middle man the database with the app (free micro instance)
    c. The backend api will be written in Python using either Flask or FastAPI
    d. Will provide both hands on experience with AWS services and industry standard backend frameworks

First Isaac will set up the AWS instances and write the backend, if too complicated or deemed infeasible a more viable option such as Firebase will be chosen.
Matia will implement the front end, and non database dependent in-app features.

Both the MVVM and MVC design pattern seem like obvious choices for the app.

4.2 Meeting 2

Time: March 3, 2025, 12:00 pm

Agenda: Regroup on progress made, assign new tasks

Team Member	Previous Task	Completion State	Next Task
Isaac Latta	- Implement book api(e.g. Open Library/Google Books) - Setup Database	75% - Missing Book api implementation	- Implement book api(e.g. Open Library/Google Books) - Begin implementing basic database functions
Matia Landry	- Design and Implement login page	100%	- Design and Implement the Main page

Isaac made the decision to use Amazon's AWS services. We have set up an aws api endpoint which forwards to our EC2 instance that runs the flask backend. The flask backend then runs the queries through an AWS RDS Postgres instance hosting the database. The core functionality of the database schema has been implemented(e.g. usernames, passwords, ids). Matia has implemented the login screen and successfully interfaced it with the controller class.

Next, Isaac will choose and implement the book api and Matia will implement the main page.

4.3 Meeting 3

Provide a similar description here.

4.4 Meeting 4

Provide a similar description here.

5 Conclusion and Future Work

- Provide a concise summary of your accomplishments, outlining the design functions and objectives successfully achieved while adhering to the specified constraints.
- In consideration of the limitations inherent in the application design, offer recommendations for potential enhancements in future iterations of the design.

6 References

- Use the IEEE reference style.
- Do not put any reference if it is not cited in the text.

Appendix

If you want to provide any additional information, use this appendix.