|  |
| --- |
| **SOFTWARE DESIGN SPECIFICATION REPORT**  **CMSE 322**  **Computer Engineering Department**  **Eastern Mediterranean University**  **PROJECT NAME:** University Secondhand Book Sales and Exchange Platform  **PROJECT START DATE: 05/03/2025**  **PROJECT END DATE: 21/05/2025**  **SUPERVISOR:** Prof. Dr. Duygu Çelik Ertuğrul  **SEMESTER TERM:** Spring 2024/2025  **PROJECT GR NO: 11** |

**Table of Contents**

[1. Introduction 3](#_Toc196692174)

[**1.1. Document Description 3**](#_Toc196692175)

[**1.1.1 Purpose 3**](#_Toc196692176)

[**1.1.2 Product Scope 3**](#_Toc196692177)

[**1.1.3 Intended Audience 3**](#_Toc196692178)

[**1.1.4 References 3**](#_Toc196692179)

[**1.2 System Overview 4**](#_Toc196692180)

[**1.2.1 Product Perspective 4**](#_Toc196692181)

[**1.2.2 Design Method 4**](#_Toc196692182)

[**1.2.3 Hardware Interfaces 4**](#_Toc196692183)

[**1.2.4 Software Interfaces 5**](#_Toc196692184)

[2 Design Considerations 6](#_Toc196692185)

[**2.1 Assumptions and Dependencies 6**](#_Toc196692186)

[**2.2 General Constraints 6**](#_Toc196692187)

[**2.3 Goals and Guidelines 7**](#_Toc196692188)

[**2.4 Development Methods 7**](#_Toc196692189)

[3 Architectural Strategies 8](#_Toc196692190)

[4 System Architecture 9](#_Toc196692191)

[5 Policies and Tactics 10](#_Toc196692192)

[6 Detailed System Design 11](#_Toc196692193)

[**6.1 Classification 11**](#_Toc196692194)

[**6.2 Definition 11**](#_Toc196692195)

[**6.3 Responsibilities 11**](#_Toc196692196)

[**6.4 Constraints 11**](#_Toc196692197)

[**6.5 Composition 12**](#_Toc196692198)

[**6.6 Uses/Interactions 12**](#_Toc196692199)

[**6.7 Resources 12**](#_Toc196692200)

[**6.8 Processing 13**](#_Toc196692201)

[**6.9 Interface/Exports 14**](#_Toc196692202)

[**6.10 Detailed Subsystem Design 18**](#_Toc196692203)

[7 Glossary 25](#_Toc196692204)

[8 Bibliography 25](#_Toc196692205)

[9 Appendix 26](#_Toc196692206)

[**9.1 Comparison table 26**](#_Toc196692207)

[**9.2 Use Case Tables 27**](#_Toc196692208)

[**9.3 Use Case Glossary: 30**](#_Toc196692209)

# Introduction

This Software Design Specification (SDS) document provides a comprehensive low-level blueprint for the design and development of the University Second-Hand Book Sales and Exchange Platform. It translates functional and non-functional requirements outlined in the Software Requirements Specification (SRS) document into a structured technical design. This SDS covers the following topics:

* Design Considerations and Constraints
* Architectural Strategy
* System and Subsystem Architecture
* Data and Design Models

## Document Description

### Purpose

This SDS serves as a guide for developers, architects, and testers, ensuring consistency in implementation, scalability, and maintainability. To ensure alignment between functional requirements (SRS) and technical implementation.

### Product Scope

With regards to the product scope, this document covers the:

* System architecture (subsystems, components, and interactions).
* Detailed design (database schema, APIs, UI workflows).
* Policies (security, performance, error handling).
* Dependencies (third-party services, hardware/software constraints).

### Intended Audience

This SDS is intended for:

* Developers: to implement features according to design.
* System Architects: to validate structural decisions.
* Testers: to derive test cases from designed and envisioned use cases.
* Project Managers: to track project progress against design milestones.

### References

IEEE Std 1016-2021 (System Design Documentation).

SRS for University Second-Hand Book Platform

## System Overview

### Product Perspective

The University Second-Hand Book Sales and Exchange Platform is intended to:

* Connect university students and staff interested in buying, selling, and exchanging books.
* Host a digital marketplace that lists books.
* Provide a simple and easy-to-use User Interface to enhance the exchange experience.

The system can interface with:

* University databases (for student/staff information)
* Mobile device hardware (camera, GPS, etc.)

### Design Method

The developer should use Object-Oriented Design, Component-Based Architecture and Iterative Development as part of the development process of the system. The system provides two primary interfaces:

* **Mobile Application (React Native)**:
* Home screen with featured listings
* Search interface with filters (price, condition, genre, distance etc.)
* Book detail view with seller information
* Messaging interface
* User profile management
* **Admin Web Portal (ASP.NET)**:
* Dashboard with system metrics
* User management page
* Content moderation tools

The UI features to be considered are accessible navigation (labeled buttons, simple UI etc.), responsive layout for various screen sizes, Dark/light mode support and features to improve screen readability.

### Hardware Interfaces

The system makes use of the following hardware with feature requirements:

* **Mobile Devices:**
* **Supported Hardware:** Smartphon**es** running iOS version 14.0+ or Android version 8.0+.
* **Hardware Requirements:**
* GPS to enhance book search using nearby book listings.
* Camera to take and upload book images.
* Internet connectivity (Wi-Fi/4G/5G).
* Touchscreen to navigate and input data.
* Speaker to notify users about new messages, exchange requests etc.
* **Server-Side:**
* Cloud hosting with a minimum of 4GB RAM and 2 vCPUs.
* 50GB SSD storage for the PostgreSQL database.

### Software Interfaces

* **Backend Services:**
* Node.js v18 + Express.js (RESTful API) for user authentication and book listing operations.
* PostgreSQL v15 databaseto store user data, book details, and transactions.
* **Third-Party APIs:**
* Firebase Authentication API (OAuth, email/password) and Firebase Cloud Messaging to implement push notifications for messages and exchange requests.
* Google Maps API for location-based search.
* **Shared Data:**
  + User profiles (ID, email, location) shared across auth and book modules.

# Design Considerations

This section describes many of the issues which need to be addressed or resolved before attempting to devise a complete design solution.

## Assumptions and Dependencies

* **Assumptions**
  + The system will be available 24/7 with minimal downtime.
  + Admins are pre-registered and authorized to manage disputes, listings, and users.
  + Users have a stable internet connection to access the platform.
  + Books listed shall be mostly academic, but other categories are also possible.
  + Users provide accurate book descriptions (title, author, condition, etc.) for their listing.
  + The system will flag suspicious activity (e.g., duplicate listings, many listings at once).
* **Related Software or Hardware**

We will use React Native to build the mobile app for both iOS and Android. For the backend system, we will use Node.js with Express.js to create the API and PostgreSQL for the database. We will connect Firebase to handle user accounts and notifications, and Google Maps for location features.

For designing the system, we will use Figma to make all the app screens and Microsoft Visio to plan the system structure. These technologies work well together, are reliable and commonly used, can scale well and fit our project budget and team skillset.

* **End-User Characteristics**

**Admin**: Manages user accounts, approves/rejects book listings, monitors transactions, and resolves disputes via a web dashboard.

**User (Students/Individuals)**: Lists books, searches/filters listings, initiates exchanges, and communicates via in-app chat.

## General Constraints

* The mobile app requires **iOS 14.0+** or **Android 8.0+** and **4GB RAM**.
* Users must enable GPS for location-based searches (or manually input locations).
* Real-time features (chat, notifications) depend on Firebase/WebSocket connectivity.
* All data transmissions use HTTPS (TLS 1.3); passwords require 8+ characters with special symbols.

## Goals and Guidelines

* **Goals**

We want to create a simple book exchange platform that helps students save money on textbooks. Our system will make it easy to find nearby books and talk to sellers directly. We'll focus on keeping it secure and making sure it works well even as more students join.

* Guidelines

Keep everything simple and easy to use. Make it work fast with quick loading times. Design it to look and feel like apps students are already familiar with. Build with reliability in mind so the system does not crash when many people use it.

## Development Methods

We will build the system step by step using incremental model. First, the high priority tasks - user accounts and basic book listings will be developed. After testing properly, we'll create the exchange system and messaging features. Lastly, implement the extra features like wishlists and admin tools.

Using this methodology, we can fix problems in each part before moving forward. Also, it’s easier to manage than building everything at once. Feedback testing would also help us iteratively improve the codebase. And if the project budget runs out or timeline constraints occur, we will have core features implemented.

# Architectural Strategies

* We will use standard web and mobile technologies to build the platform, combining frontend and backend solutions with a reliable database system.
* The system will follow common software design approaches that work well for similar applications, and will use Integration Testing to make sure different parts of the system work together properly.
* All parts of the platform will be available online, with both user and admin interfaces accessible through internet connections.
* We will use a single main database that keeps all information up-to-date in real time across the whole system.
* The system won't connect to or depend on any outside third party databases or services, we will host everything ourselves so it is easier for us to handle any hardware issues.

# System Architecture

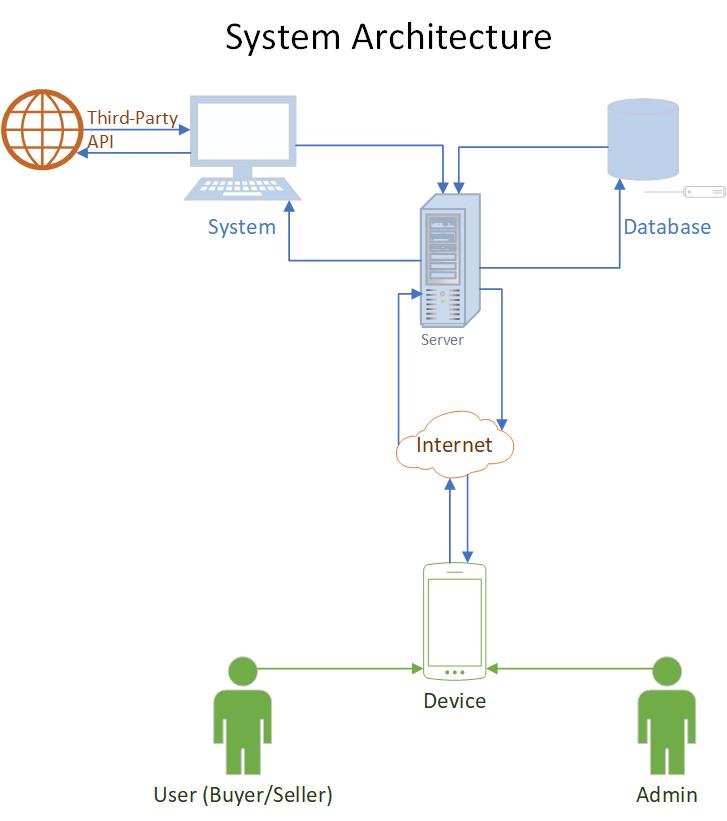


Figure 1: System Architecture

**Context Diagram**

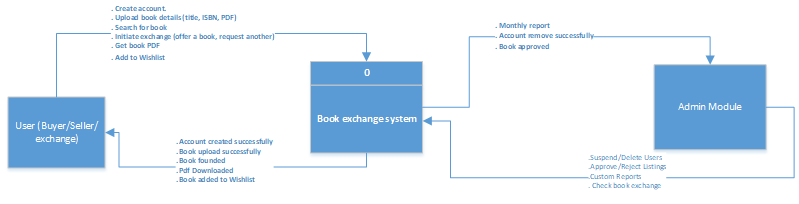


Figure 2: Context Diagram

# Policies and Tactics

The book exchange system will be coded in JavaScript, HTML, and CSS languages using PostgreSQL for the database. The default OS will be Windows 11. We will use React and React Native for the framework. Throughout the implementation process, all needs will be outlined in the SRS document and use cases will be employed to ensure that all functionalities are established in accordance with the requirements. All the features and requirements will be tested efficiently with all bugs and reported problems fixed. Our app will require users to have access to the internet.

# Detailed System Design

## Classification

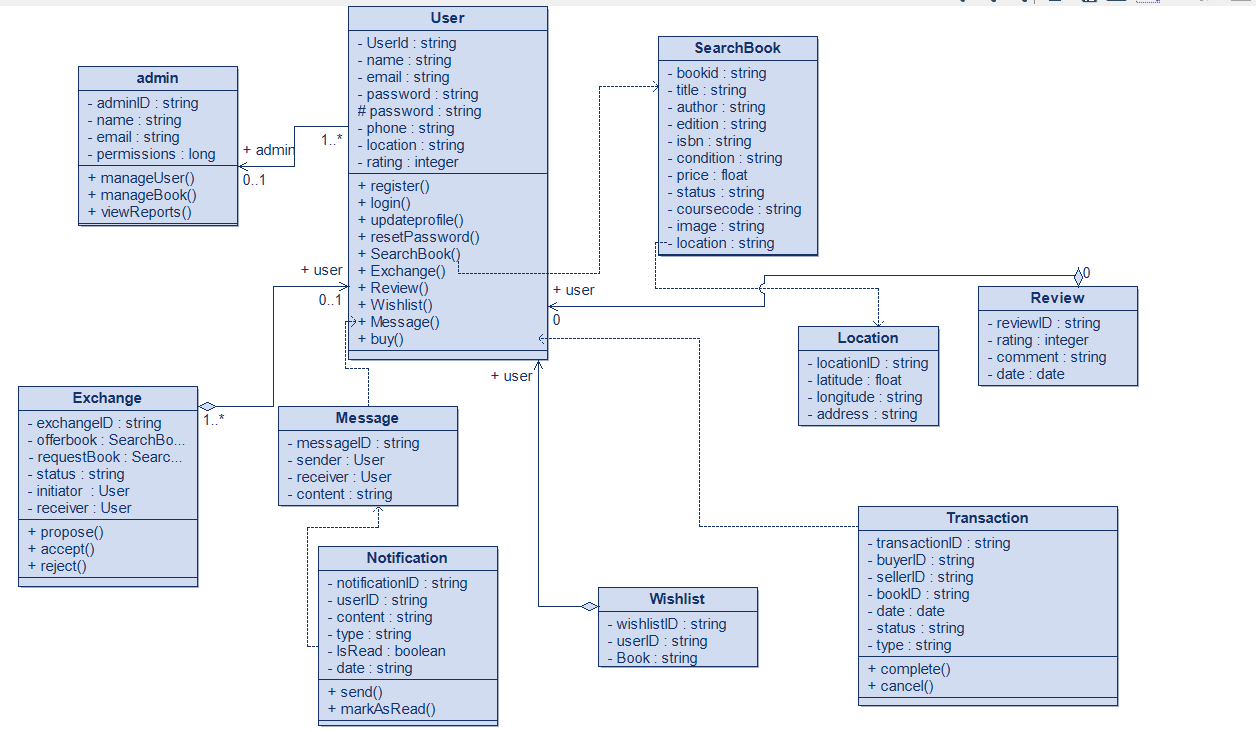


Figure 3: Class Diagram

## Definition

Our system manages interactions for buying, selling, and exchanging used books among users. It will handle user login/register, book listings, search functionality, messaging, and lastly admin moderation.

## Responsibilities

* Authentication Module: Validates user credentials and manages session tokens
* Listing Module: Allow users to create, view, edit, or delete book listings made.
* Search Module: Provides keywords and filters to let users perform quick search.
* Messaging Module: Allows users to message in real time
* Admin Panel: Allows admins to moderate the application in real time.

## Constraints

* Platform Constraints: The mobile app is limited to Android 8.0+ and IOS 14.0+, also admin panel requires a modern web browser
* Security Constraints: All communication must be encrypted using HTTPS
* Performance Constraints: Server response time must not exceed a certain period of time
* Storage Constraints: Images that will be uploaded must not exceed 5MB per listing
* Operational Constraints: The system must be connected to the internet; offline is not supported

## Composition

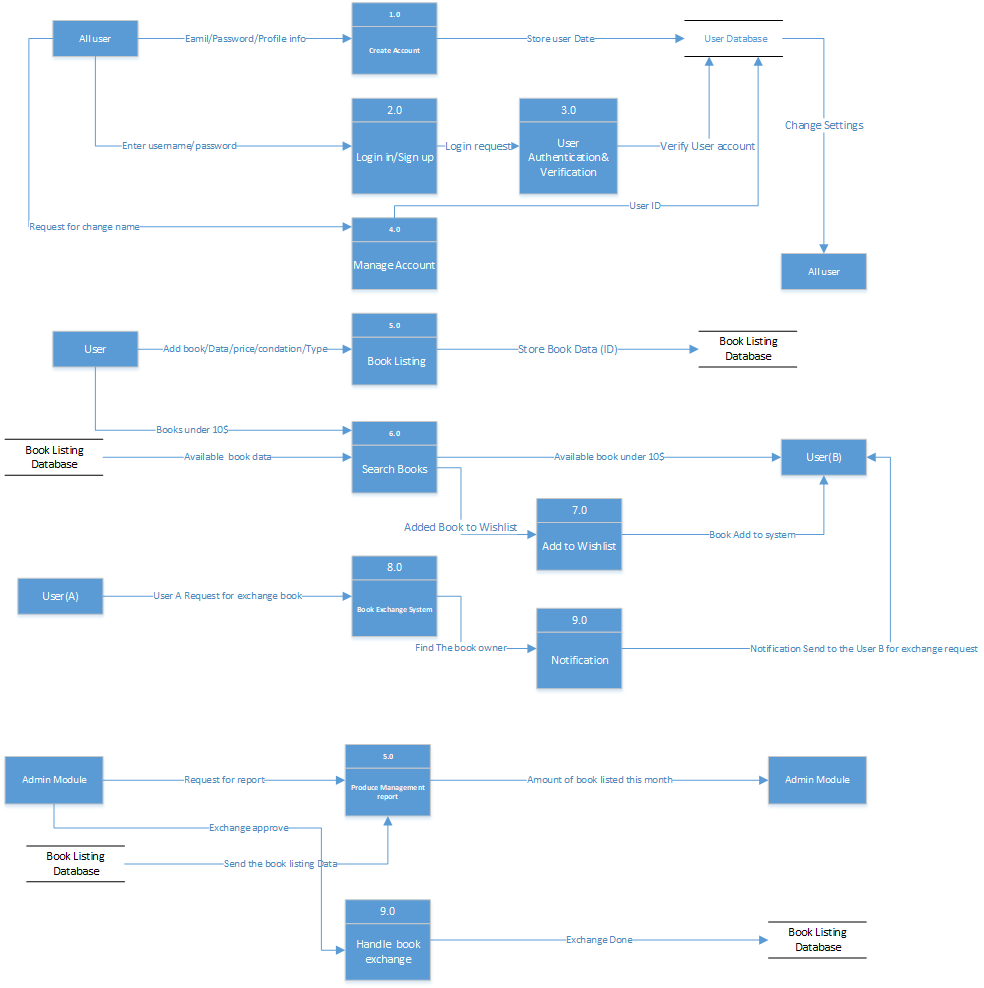


Figure 4: DFD Level-0 Diagram

## Uses/Interactions

The interactions between the classes are defined in the class diagrams drawn and stated above in 6.1 (Figure 3)

## Resources

A server and database are the primary resources. This resource will be used to run and manage most of the functions.

## Processing

|  |  |  |  |
| --- | --- | --- | --- |
| **Function (F) \ Module (M)** | **Input** | **Output** | **Access** |
| Create an Account (Sign up) (F) | Personal Details | Login Screen | All |
| Log in to the system (F) | User Credentials (Email/Password) | System Homepage | Registered Users, Admin |
| View Profile/Edit (F) | User Credentials | Profile Page (View/Update) | Registered Users, Admin |
| List a Book for Sale or Exchange (F) | Book Information (Details, Images, Type) | Book Listing Created | Registered Users |
| Offer a PDF Version (F) | PDF File Upload | PDF Available for Listing | Registered Users |
| Search for Books (F) | Search Filters (Category, Price, Location) | Search Results | Registered Users, Admin |
| Save Books to a Wishlist (F) | Selected Book | Wishlist Updated | Registered Users |
| Start a Chat (F) | Selected User/Listing | Chat Session Opened | Registered Users |
| Purchase a Book you want (F) | Payment Details | Successful Purchase | Registered Users |
| Confirm Sale in the System (F) | Sale Confirmation | Sale Finalized | Registered Users |
| Request a Book Exchange (F) | Book Proposal | Exchange Request Sent | Registered Users |
| Respond to Exchange Requests (F) | Exchange Offer | Exchange Accepted/Declined | Registered Users |
| View Transaction History (F) | User Selection | Transaction History Displayed | Registered Users, Admin |
| Manage User Accounts (M) | User Management Actions | User Account Managed | Admin |
| Approve/Reject Book Listings (M) | Book Listing | Approval/Rejection Notification | Admin |
| Monitor Transactions & Handle Fraud (M) | Reported Transactions | Investigation Outcome | Admin |
| Manage Promotions & Advertisements (M) | Promotion/AD Details | Promotion Managed | Admin |

## Interface/Exports

Our mobile application will be a responsive and user-friendly interface. Our mobile application will be developed using HTML, CSS, and JavaScript for the front end with React and React Native frameworks. For the database, we are using PostgreSQL.

Our database will be completely integrated with our mobile application, allowing immediate updates and real-time data modification via the hosting server. For the system to function properly, internet connection will be needed.

Important Export/Interface Features:

* synchronization of user profiles, messaging, and book listings in real time.
* updates to the front end without needing to reload the entire page, without user noticing it.
* options for administrative database exports (e.g., transaction history, user data).
* Cross-platform compatibility: The system will run on both iOS and Android.

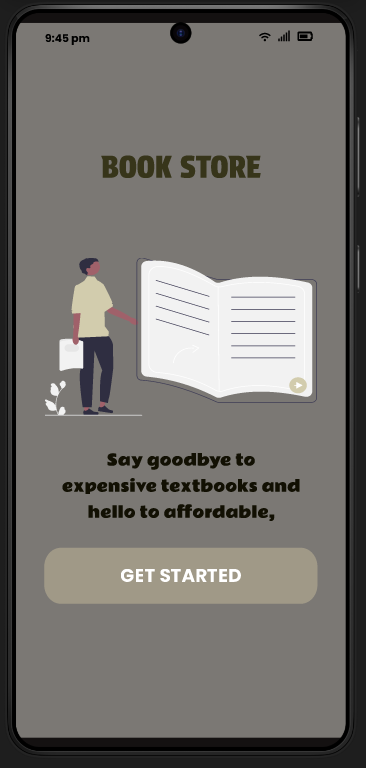


Figure 5: UI Welcome Screen

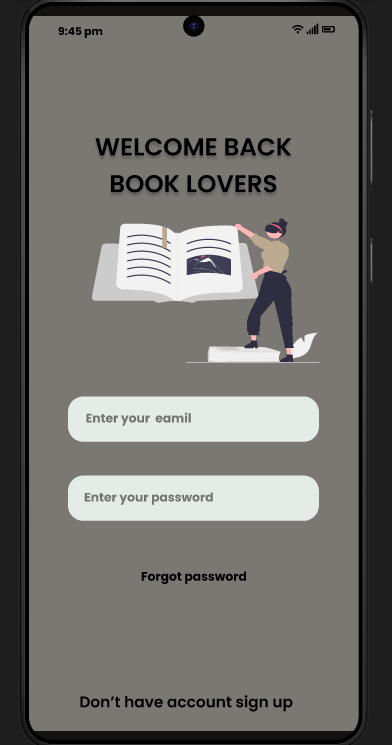


Figure 6: UI Login Screen

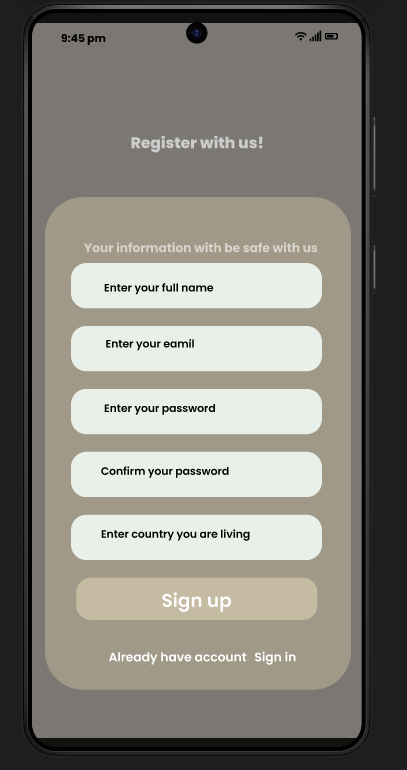


Figure 7: UI User Registration

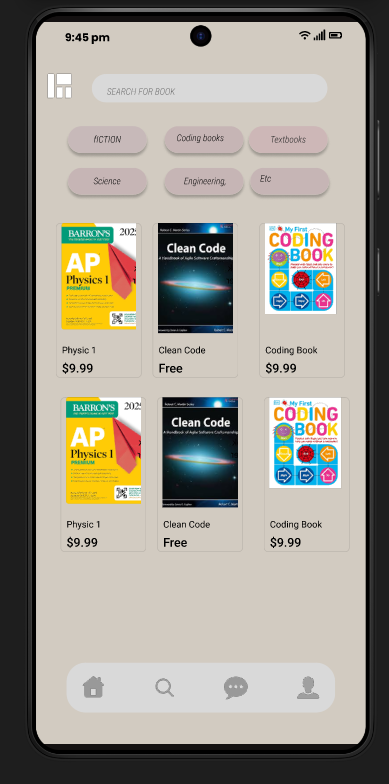


Figure 8: UI Home Screen

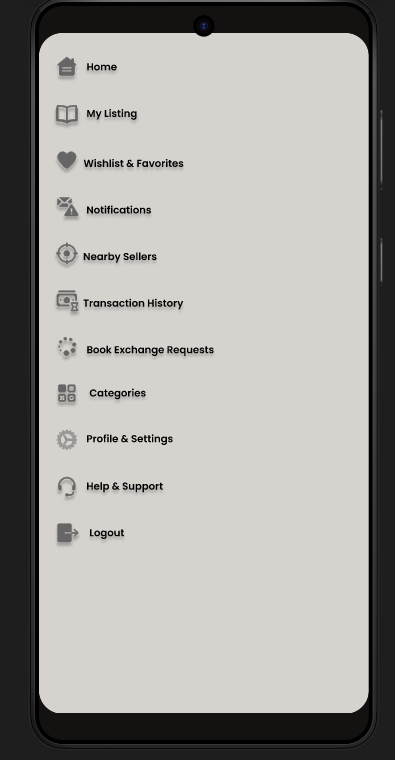


Figure 9: UI Sidebar Menu

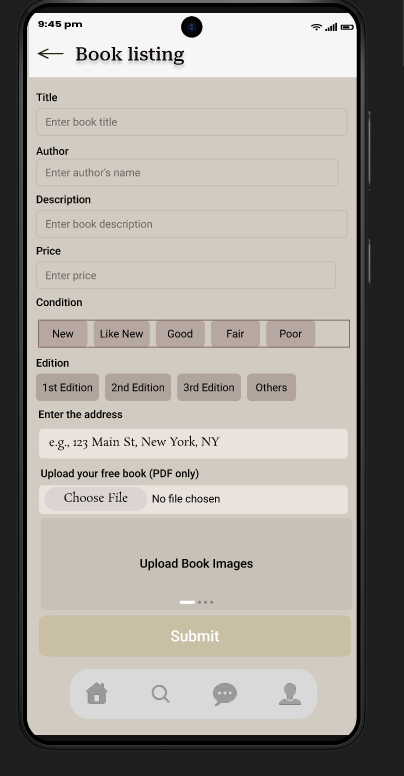


Figure 10: UI New Book Listing

A screen shot of a phone

AI-generated content may be incorrect.

Figure 11: UI Wishlist

## Detailed Subsystem Design

**Use Case Diagrams**

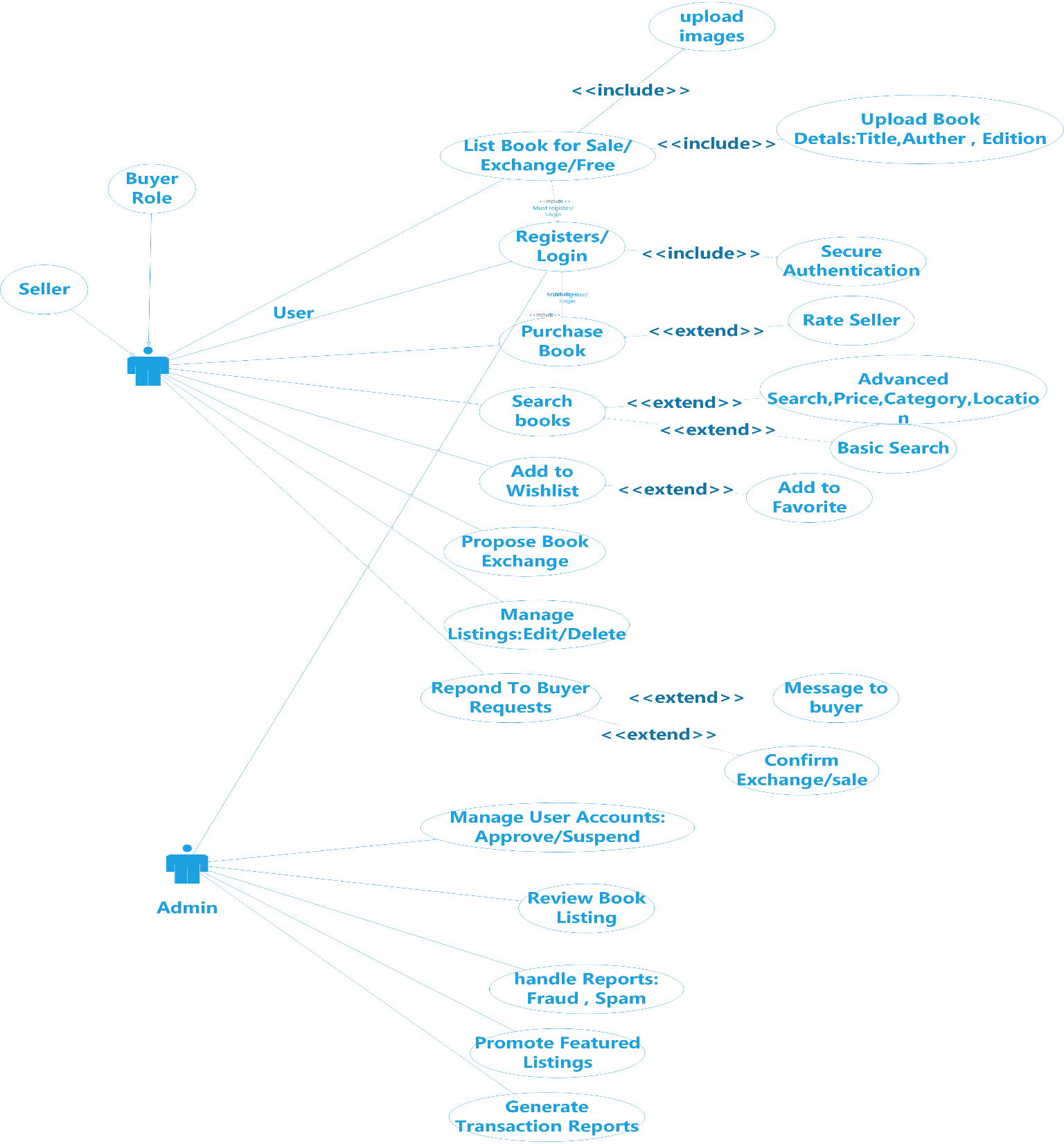


Figure 12: Use Case Diagram

**Activity Diagrams**

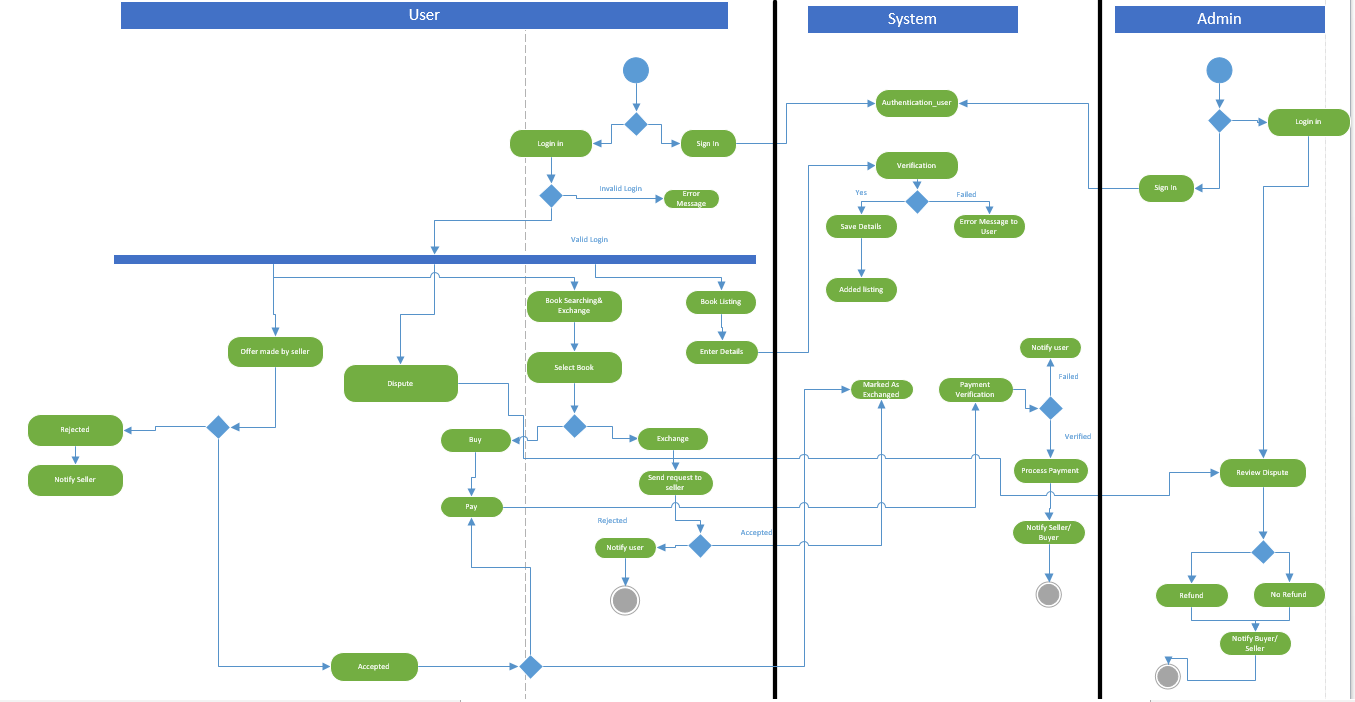


Figure 13: Activity Diagram

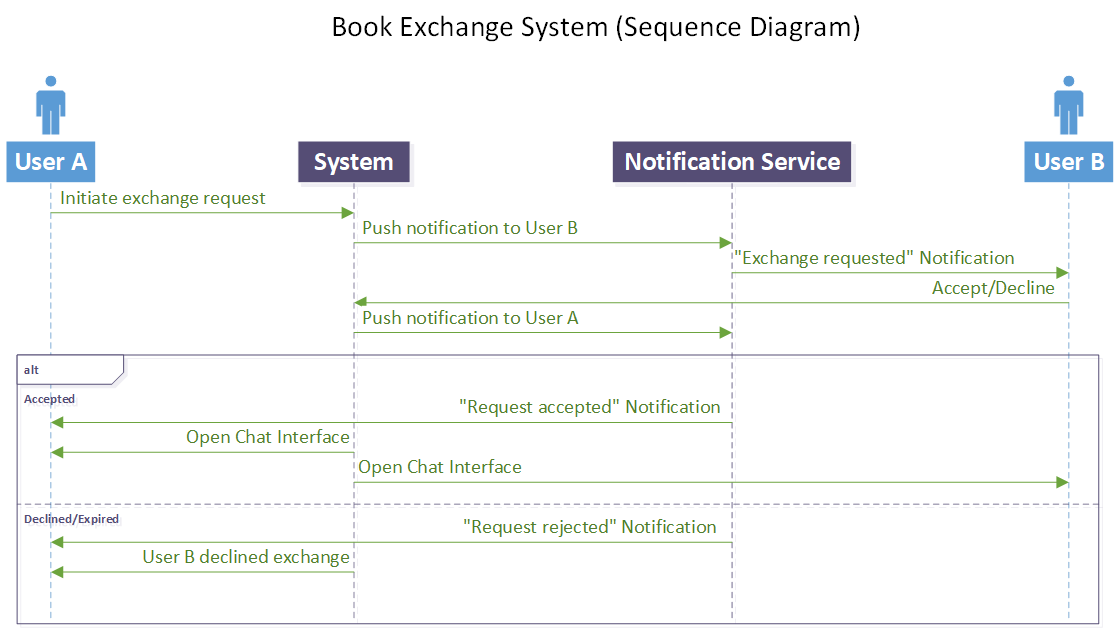
**Sequence Diagram**

Figure 14: Sequence Diagram (Book Exchange System)

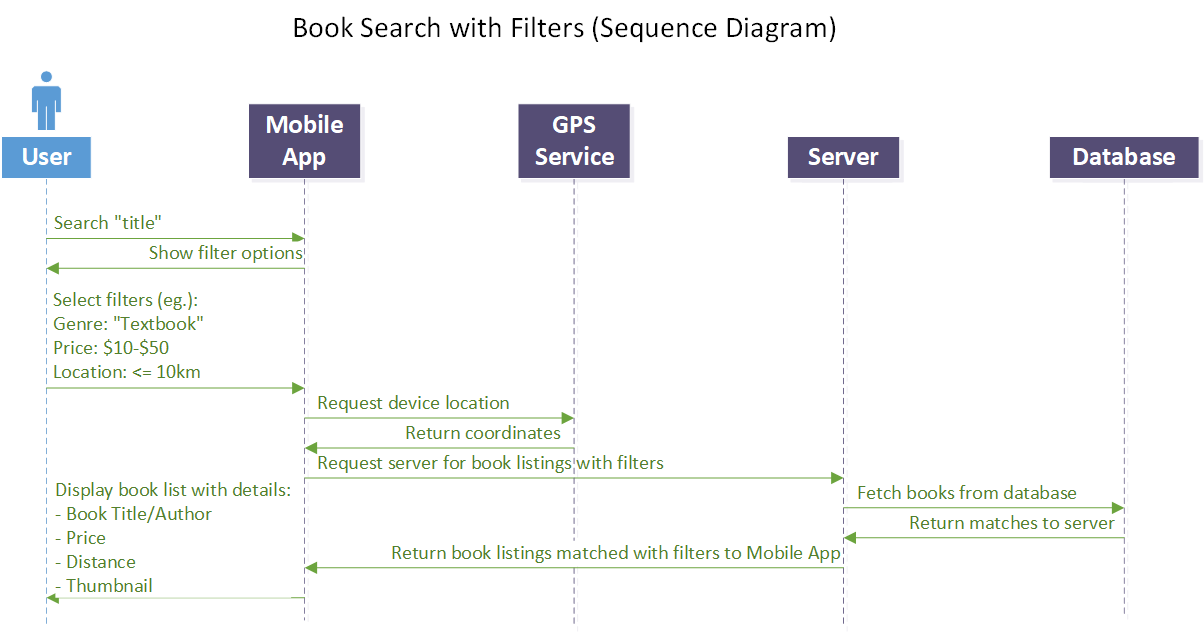


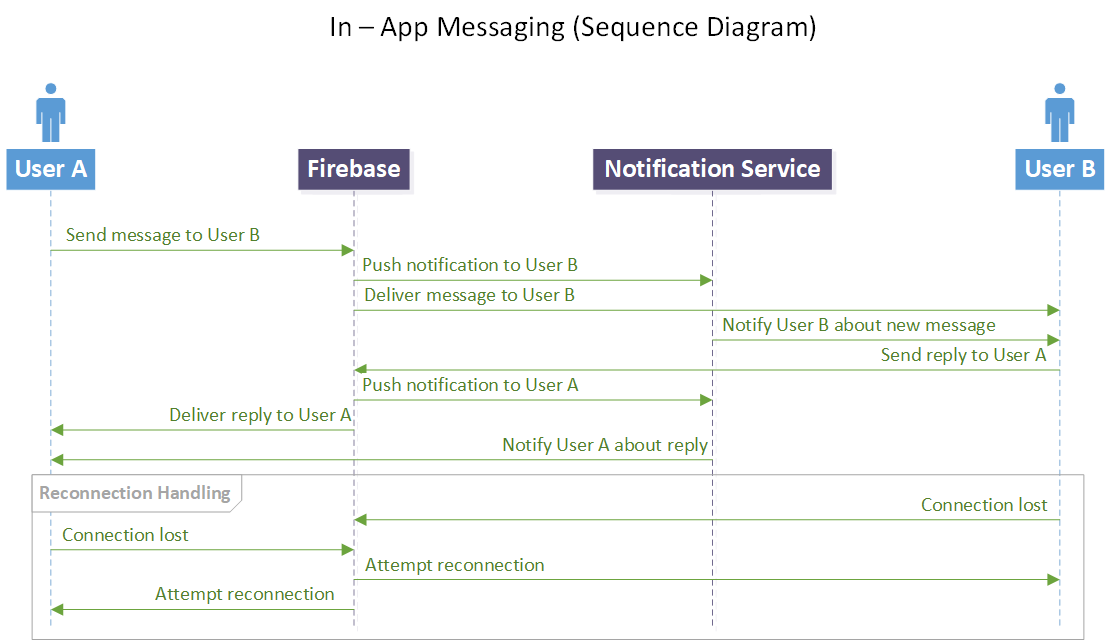
Figure 15: Sequence Diagram (Book Search with Filters) 

Figure 16: Sequence Diagram (In App Messaging)

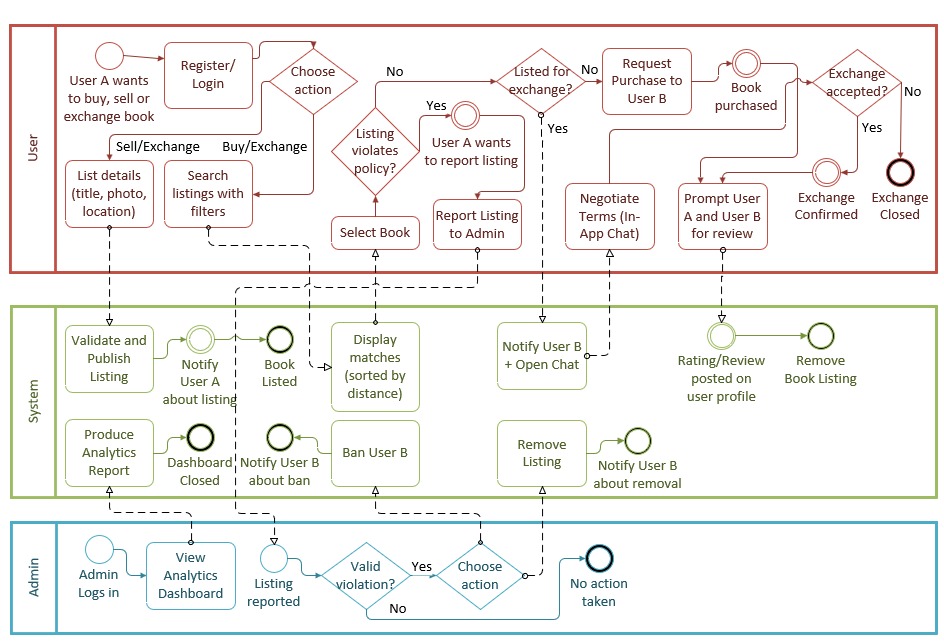
**BPMN Diagram**

Figure 17: BPMN Diagram

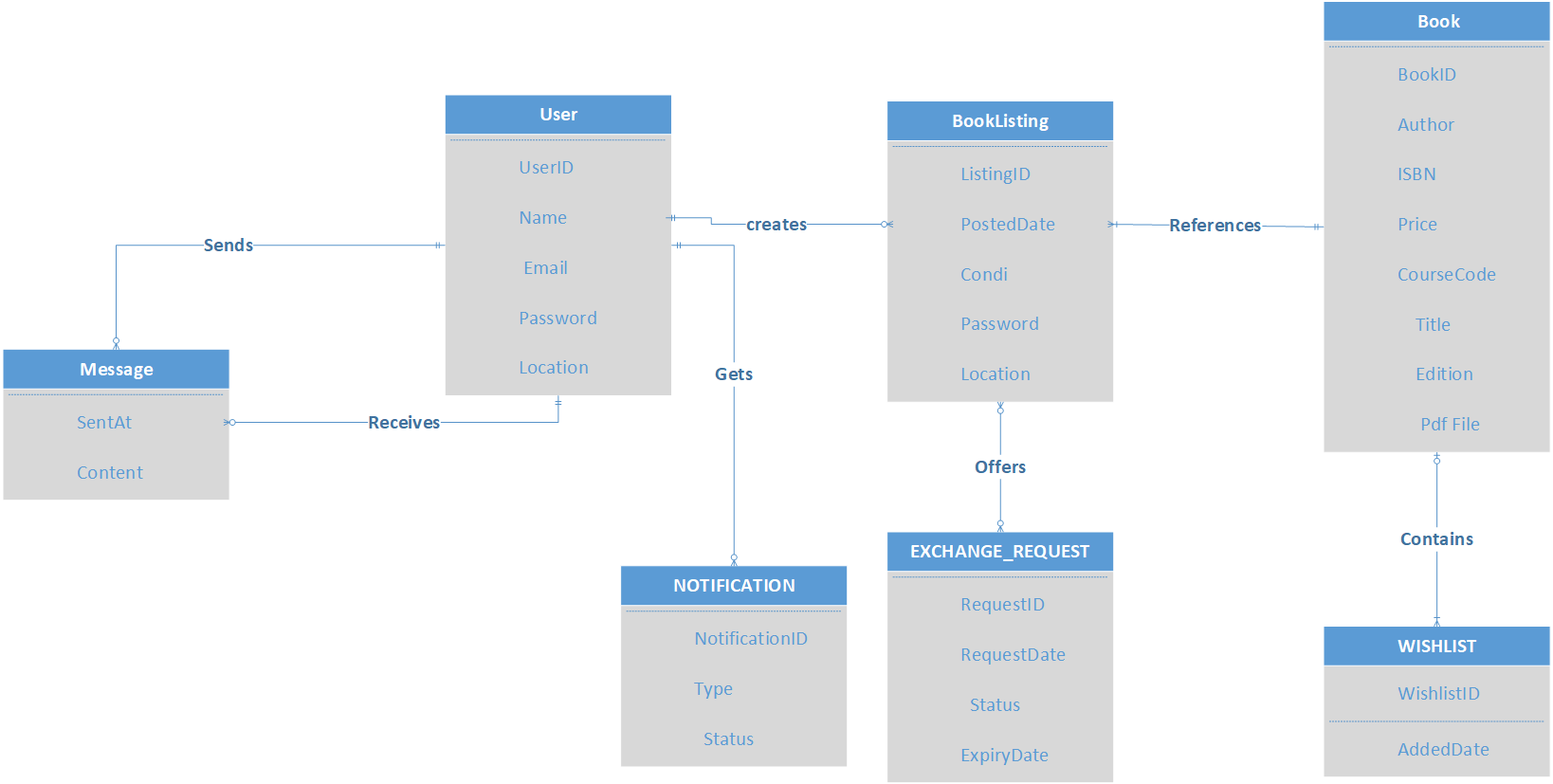
**E-R Diagrams** 

Figure 18: Conceptual E-R Diagram

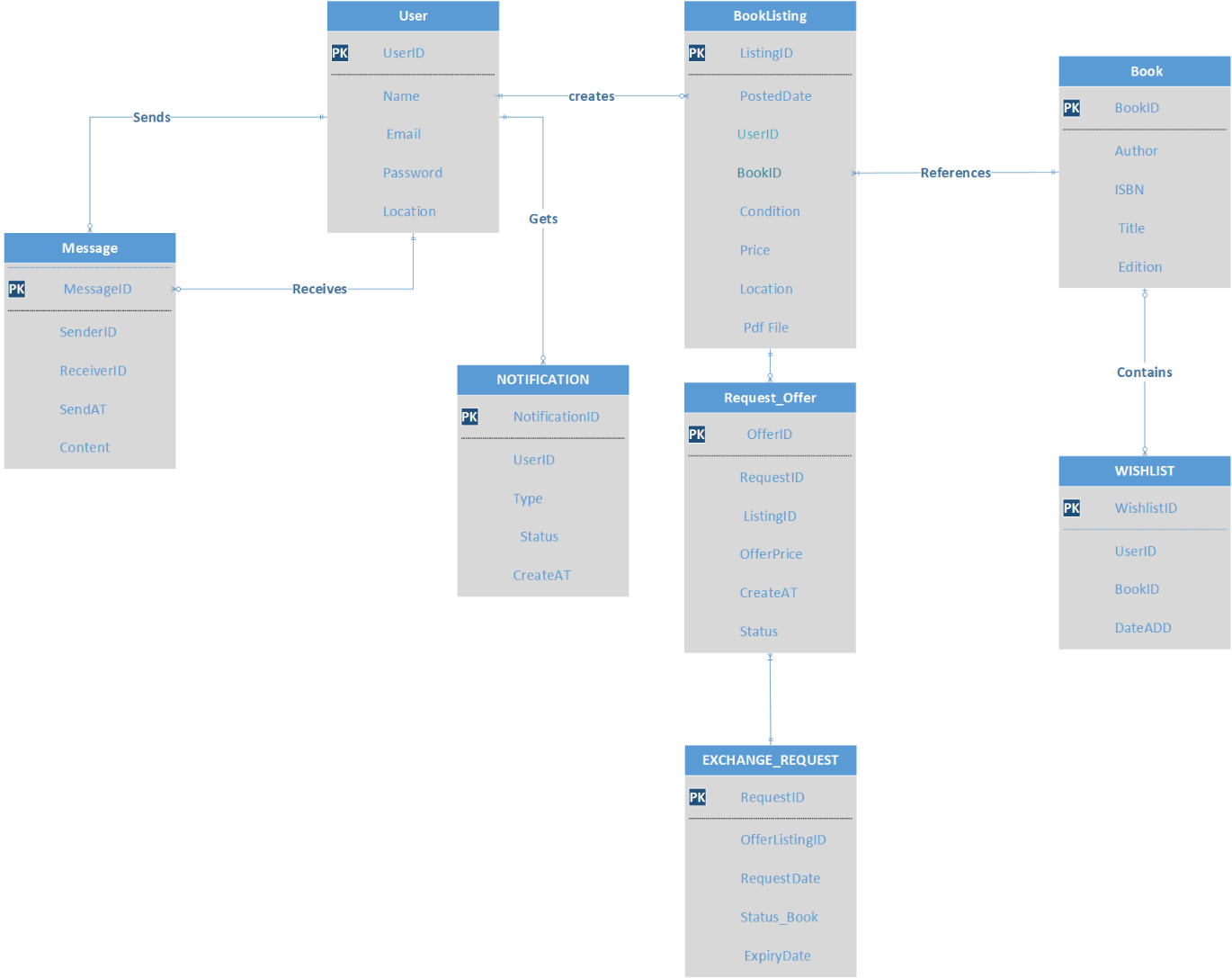


Figure 19: Logical Database Diagram

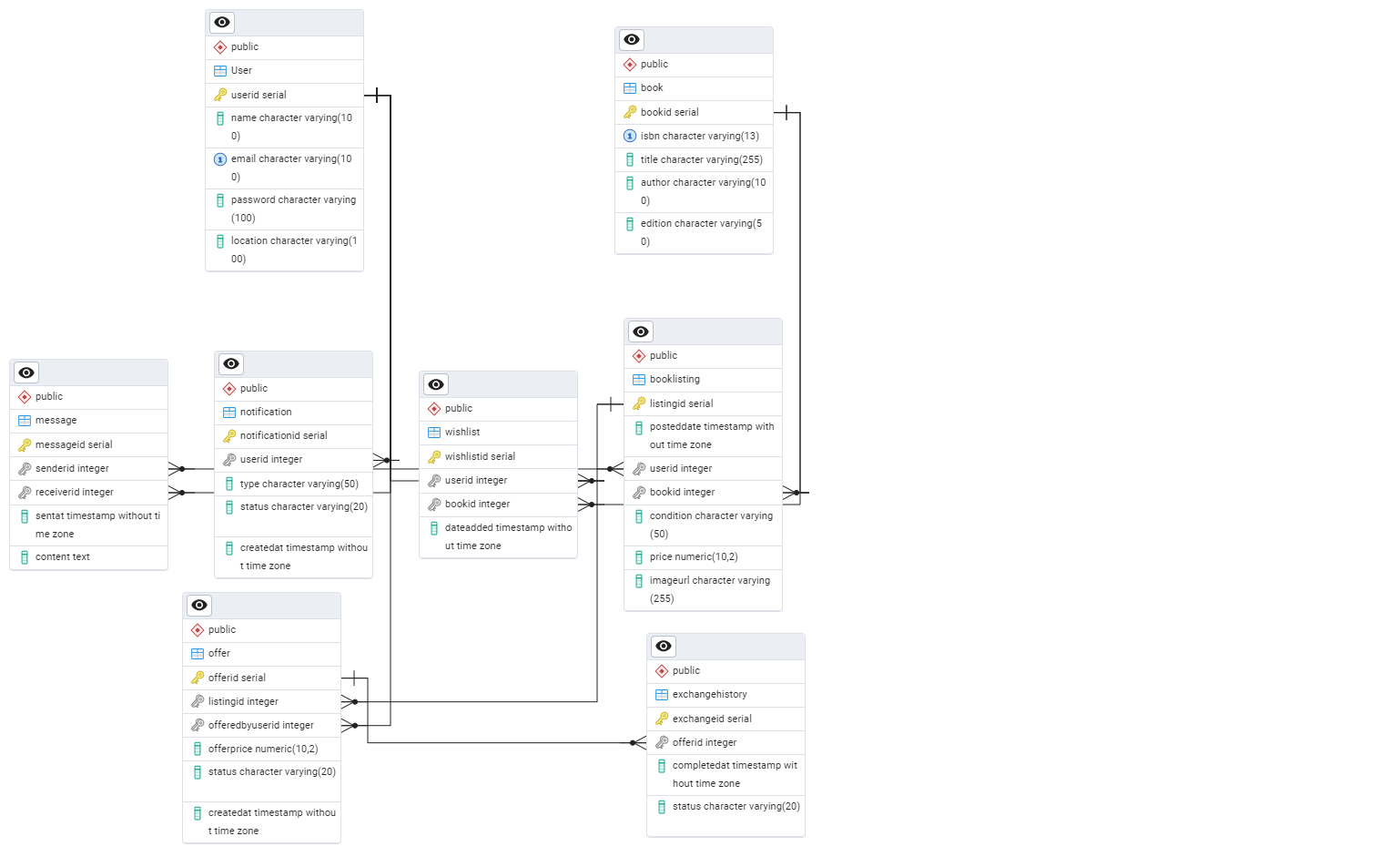


Figure 20: Physical Database

# Glossary

**BPMN –** Business Process Model and Notation

**CSS** – Cascading Style Sheets

**DB**- Database

**GUI** – Graphical User Interface

**HTML** – Hyper-Text Markup Language

**IDE** – Integrated Development Environment

**OS** – Operating System

**SDS** – Software Design Specification

**SQL** – Structured Query Language

**SRS** – Software Requirements Specification

**UI** – User Interface

**UML** – Unified Modeling Language

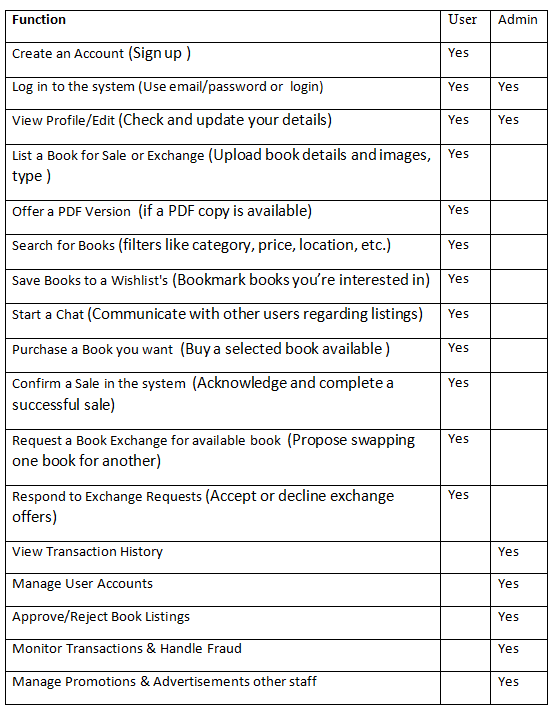
# Bibliography

Figma <https://www.figma.com/>

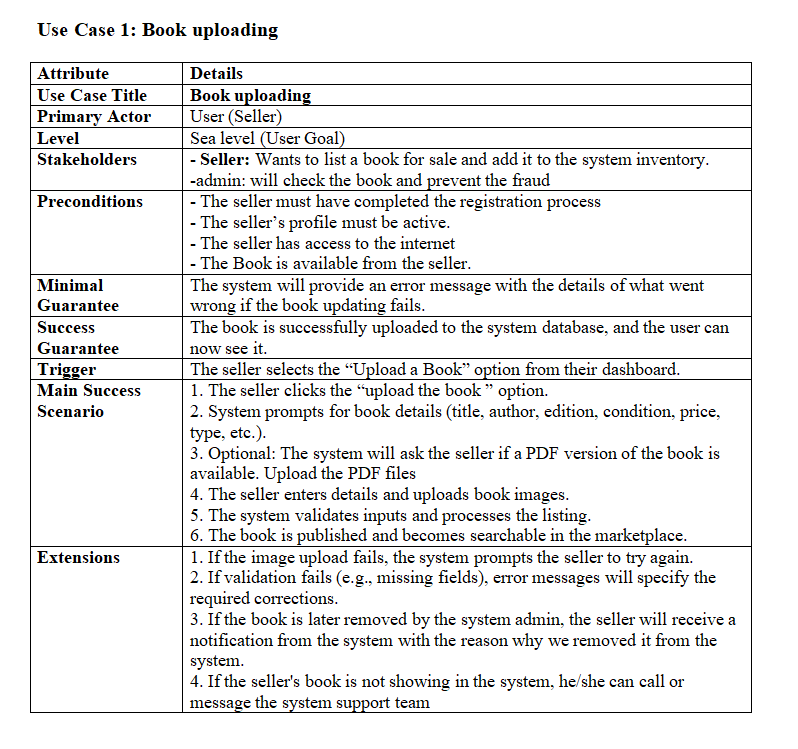
[4] Modelio <https://www.modeliosoft.com/en/>

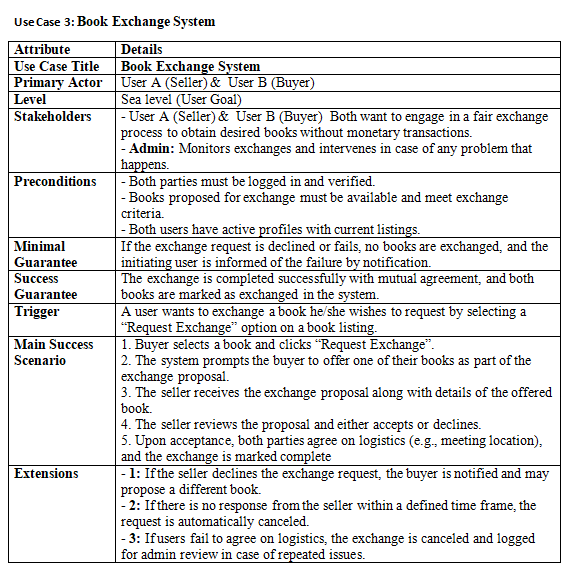
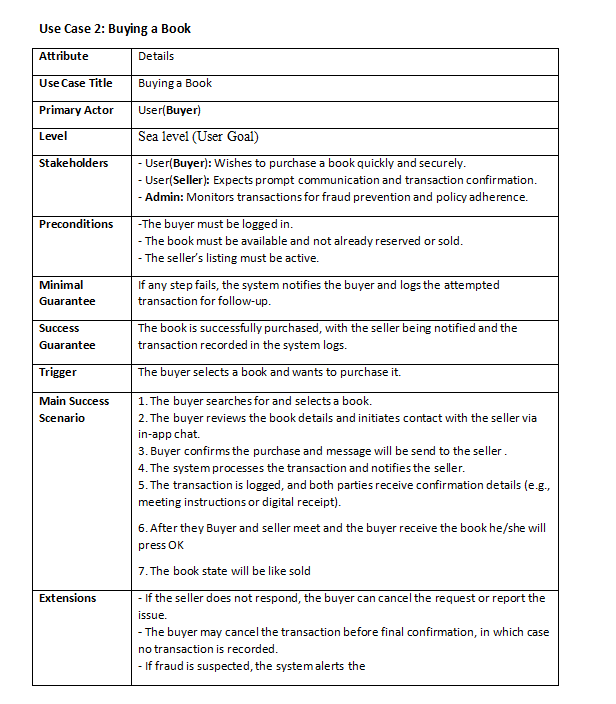
# Appendix

## Comparison table



## 9.2 Use Case Tables





## 9.3 Use Case Glossary:

|  |  |  |  |
| --- | --- | --- | --- |
| ID | Use Case Name | Description | Actors |
| 01 | User Registration/Login | Process for users to create accounts or log in via email. Includes security checks (OAuth, verification). | User (Buyer/Seller), Admin |
| 02 | Book Listing | A seller uploads the book details (title, price, condition, images) for sale/exchange. | User (Seller) |
| 03 | Secure Authentication | System features ensure secure login via OAuth 2.0, email/phone verification, or password encryption. | User (Buyer/Seller), Admin |
| 04 | In-App Chat | Real-time messaging system for buyers and sellers to negotiate transactions. | User (Buyer), User (Seller) |
| 05 | Fraud Detection | Automated system detection for suspicious listings/accounts for admin review and check it. | Admin |
| 06 | Book Exchange | A buyer’s request to exchange a book for an agreement with the seller. | User (Buyer), User (Seller) |
| 07 | Location-Based Search | Feature allowing users to filter book listings by location-based search. | User (Buyer) |
| 08 | Transaction Logs | Records of completed sales/exchanges of the user will give reports | Admin |
| 09 | Wishlist | A feature where buyers save books they want to buy. | User (Buyer) |
| 10 | Admin Dashboard | Web-based interface for admins to manage users, listings, and reports. | Admin |