VIETNAM GENERAL CONFEDERATION OF LABOR

**TON DUC THANG UNIVERSITY**

**FACULTY OF INFORMATION TECHNOLOGY**



**NGUYỄN GIA MỸ – 521H0272**

**NGUYỄN PHẠM THANH UYÊN – 521H0329**

**NGUYỄN ĐÌNH VIỆT HOÀNG – 522H0120**

**BUILDING A SYSTEM TO SUPPORT ONLINE BOOK TRADING**

**FINAL REPORT ENTERPRISE SERVICE-ORIENTED ARCHITECTURE**

**HO CHI MINH CITY, 2024**

VIETNAM GENERAL CONFEDERATION OF LABOR

**TON DUC THANG UNIVERSITY**

**FACULTY OF INFORMATION TECHNOLOGY**



**NGUYỄN GIA MỸ – 521H0272**

**NGUYỄN PHẠM THANH UYÊN – 521H0329**

**NGUYỄN ĐÌNH VIỆT HOÀNG – 522H0120**

**BUILDING A SYSTEM TO SUPPORT ONLINE BOOK TRADING**

**FINAL REPORT ENTERPRISE SERVICE-ORIENTED ARCHITECTURE**

Advised by

**Mr. DƯƠNG HỮU PHÚC**

**HO CHI MINH CITY, 2024**

**ACKNOWLEDGEMENT**

We sincerely thank Mr. Dương Hữu Phúc for teaching us the Enterprise Service-Oriented Architecture course with great enthusiasm. We want to express our deep appreciation for the dedication and professional knowledge that you shared with us. Through your classes, we gained a better understanding of the fundamental aspects of the Enterprise Service-Oriented Architecture, thanks to your detailed explanations and practical applications. You helped us grasp the knowledge and apply it effectively. Finally, we extend our heartfelt gratitude to Mr. Dương Hữu Phúc for your commitment and invaluable support throughout our learning journey in this course. The skills and knowledge we acquired will continue to impact our future development. We sincerely thank you and wish for your health, success, and happiness.

*Ho Chi Minh city, 3rd September 2023.*

*Author*

*(Signature and full name)*

***My***

Nguyễn Gia Mỹ

***Uyen***

Nguyễn Phạm Thanh Uyên

***Hoang***

Nguyễn Đình Việt Hoàng

**DECLARATION OF AUTHORSHIP**

We hereby declare that this is our own project and is guided by Mr. Dương Hữu Phúc; The content research and results contained herein are central and have not been published in any form before. The data in the tables for analysis, comments and evaluation are collected by the main author from different sources, which are clearly stated in the reference section.

In addition, the project also uses some comments, assessments as well as data of other authors, other organizations with citations and annotated sources.

**If something wrong happens, We’ll take full responsibility for the content of my project.** Ton Duc Thang University is not related to the infringing rights, the copyrights that We give during the implementation process (if any).

*Ho Chi Minh city, May 12, 2024*

*Author*

*(Signature and full name)*

***My***

Nguyen Gia My

***Uyen***

Nguyen Pham Thanh Uyen

***Hoang***

Nguyen Dinh Viet Hoang

**BUILDING A SYSTEM TO SUPPORT ONLINE BOOK TRADING**

In the modern commercial world, e-commerce has become an indispensable part, opening up new opportunities and creating breakthroughs in accessing the global market. E-commerce systems are not just tools but also a source of inspiration for development and progress in business. The system supporting online book transactions is also part of this development.

The online book transaction system enables users to update information, list books for sale, and help users search for and purchase their favorite books. Therefore, building a website to support online book transactions is a typical case of e-commerce activity. The system analysis and design project for online book transaction support is based on an e-commerce approach, programming platforms, and subsequent parts of the project presented as follows:

**Chapter 1:** System Overview. This chapter presents the approach and relationship to system development.

**Chapter 2:** System Analysis and Design. In this chapter, the system is analyzed and designed in an object-oriented manner using diagrams such as use-case, ERD, etc.

**Chapter 3:** System Implementation. This chapter presents the technologies used to implement the system and lists the corresponding APIs for the system functions.

**Chapter 4:** System Demonstration.

**TABLE OF CONTENT**

[**LIST OF FIGURES 5**](#_30j0zll)

[**LIST OF TABLES 6**](#_1fob9te)

[**ABBREVIATIONS 7**](#_3znysh7)

[**CHAPTER 1. OVERVIEW OF THE SYSTEM 1**](#_2et92p0)

[**1.1 Introduction of the topic 1**](#)

[**1.2 System Description 2**](#)

[**1.3 Topic Scope 4**](#)

[**1.3.1 Object and function limitations 4**](#_lnkxxxhf3vi6)

[**1.3.2 Technology limitations 5**](#_n3dedkfz2gwm)

[**1.4 Practical Significance 6**](#)

[**1.5 Report Structure 6**](#)

[**CHAPTER 2. SYSTEM ANALYSIS AND DESIGN 8**](#)

[**2.1 Functional Requirements 8**](#)

[**2.2 Non-functional Requirements 9**](#)

[**2.3 Use-case Diagram 9**](#)

[**2.3.1 Use-case Overview 10**](#)

[**2.3.2 Use-case Description 13**](#)

[**2.4 Entity-Relationship Diagram 31**](#)

[**2.4.1 ERD Diagram 31**](#_bm4wroakblo)

[**2.4.2 Physical Database Diagram 32**](#_nke03dhgzvjd)

[**CHAPTER 3. SYSTEM IMPLEMENTATION 33**](#)

[**3.1 Technical Stack 33**](#)

[**3.2 System Architecture 36**](#)

[**CHAPTER 4. DEMONSTRATION 67**](#)

[**CHAPTER 5. CONCLUSION 70**](#_tyjcwt)

[**REQUIREMENTS COLLECTION QUESTIONS LIST 71**](#_kmumrx9vrk4e)

[**REFERENCES 74**](#_4d34og8)

# LIST OF FIGURES

[Figure 2.1: Use case diagram 9](#_5p9jiiv2xcly)

[Figure 2.24: Supported Trading Books System Online ERD 31](#_inu4kq3zntty)

[Figure 2.25: Supported Trading Books Online System Physical Level Database Diagram 32](#_1cs5pfeebavh)

# LIST OF TABLES

[Table 2.2: Use-case Overview Table 13](#_ex43wvqfhbz9)

[Table 2.3: Sign Up Use-case Description 14](#_mo3d6kw4l3i9)

[Table 2.4: Login Use-case Description 15](#_7pscn6sz50rh)

[Table 2.5: Searching Books Use-case Description 16](#_yjzhhh1xzf9)

[Table 2.6: View Book Details Use-case Description 17](#_kyshff6x0392)

[Table 2.7: Enter Information Of Books Use-case Description 18](#_ovy6di1yw51u)

[Table 2.8: Sell Books Use-case Description 19](#_pdqqtyyciw3c)

[Table 2.9: Delete Books Use-case Description 20](#_hv0b8o8rr39b)

[Table 2.10: Update Account Use-case Description 21](#_wv6srj52l799)

[Table 2.11: Close Account Use-case Description 22](#_6qzg6q9nexbl)

[Table 2.12: Order Books Use-case Description 22](#_kwp9nwivm2i9)

[Table 2.13: Notice Delivery Price Use-case Description 23](#_kvdxfea2uva1)

[Table 2.14: Make Payment Use-case Description 24](#_mebw20tpfm0k)

[Table 2.15: Update Delivery Status Use-case Description 25](#_9r9zw2t3js66)

[Table 2.16: View Ordered History Use-case Description 25](#_i8c0gxulmnd9)

[Table 2.17: Approve Books Use-case Description 26](#_adgvg9q88bqv)

[Table 2.18: Delete Books Use-case Description 27](#_brz9pxy5cpat)

[Table 2.19: View Orders Use-case Description 28](#_ipi1roiinchp)

[Table 2.20: Delete Users Use-case Description 28](#_ro0y25mkfqvu)

[Table 2.21: View User Information Use-case Description 29](#_1w7w7p4leysq)

[Table 2.22: Send Notifications Use-case Description 30](#_tu1elgygzfkf)

[Table 2.23: Logout Use-case Description 31](#_nqklnpws88p0)

# ABBREVIATIONS

| GHN | Fast delivery |
| --- | --- |
| UI | User interface |
|  |  |
|  |  |
|  |  |
|  |  |

# OVERVIEW OF THE SYSTEM

## Introduction of the topic

Registration to create an account is mandatory for users. They need to provide information such as email, full name, password, phone number, and address. In case of forgetting the password, users can retrieve it through the registered email. When visiting the website, users can search or filter information such as authors or titles. Users can view detailed information about the book, including description, images, prices, and reviews from other users before deciding to buy.

When wanting to delete a book, sellers can select the books they have posted on the system. When wanting to sell a book, users need to provide information:

Information about the GHN (Fast Delivery) account

Details about the book, including title, author, description, and images. They can set the selling price for their book and manage the book buying status from other users. After selecting a book and providing necessary information, the purchase will be considered. When the seller confirms the order, the order will be transferred to the shipping unit (GHN) and the buyer can see the order status on the system.

The delivery will be handled by the Fast Delivery unit and quote the shipping cost to the customer.

The website only provides payment methods through Momo e-wallet.

Once users complete the transaction, they can rate the number of stars (maximum 5 stars and only rate once) for the book they have bought. Users can manage their account, including updating the account or closing the account.

The system administrator will monitor the system, manage user accounts, manage books, and view orders from users. For book management rights, the system administrator can confirm the book before the seller posts it on the system or delete the books posted on the system. For user management rights, administrators can view user information or delete user accounts.

The main page for users will provide information about popular books, as well as provide options to log in or register an account.

Each book product on the website will be presented with detailed information such as book name, book cover, quantity, price, author, product description, and reviews from users.

Users when buying or selling books will be provided with order code, buyer code or seller code, details about the book product (book name, book cover, author, description, selling price, rating), order status, total order value (order cost and shipping cost) and transaction time.

Users can check the order history after logging into the system.

Fast Delivery will receive orders from the seller including the seller code and buyer code, order information, and verify the seller’s order to make a shipping order. The shipping order will include information from the order that the buyer sends to the seller and the shipping code of the shipping unit.

The system can send notifications to the user’s email address when successfully registered.

## System Description

Reading books is an indispensable part of many people's lives. Providing an online platform for buying and selling books makes it easy for users to search for and purchase their favorite books. Through the website, users can easily search for detailed information about books such as book titles, authors, genres, prices, book conditions, and view illustrations. The platform also allows users to post information about the books they want to sell so that others can have more options when buying books.

Registration to create an account is required for users.

They need to provide information such as email, full name, password, phone number, and address.

In case of forgetting the password, users can recover it via the registered email.

When accessing the website, users can search or filter information such as authors or titles. Users can view detailed information about books, including descriptions, images, prices, and reviews from other users before deciding to buy."

"When wanting to delete books, sellers can select the books they have posted on the system.

When wanting to sell books, users need to provide information:

1. Information about the GHN (Fast Delivery) account

2. Details about the book, including title, author, description, and images. They can set the selling price for their books and manage the purchase status from other users.

After selecting books and providing the necessary information, the purchase will be considered. When the seller confirms the order, the order will be transferred to the shipping company (GHN), and the buyer can see the order status on the system.

The shipping will be handled by the Giao Hang Nhanh company, and the shipping fee will be quoted to the customer.

The website only provides payment methods through the Momo e-wallet.

Users complete the transaction, they can rate the book (up to 5 stars and only rate once) they have purchased.

Users can manage accounts, including updating accounts or closing accounts.

The system administrator will monitor the system, manage user accounts, manage books, and view orders from users. For book management rights, the administrator can confirm books before sellers post them on the system or delete books posted on the system. For user management rights, the administrator can view user information or delete user accounts.

The main page for users will provide information about popular books, as well as provide options to log in or register for an account.

Each book product on the website will be presented with detailed information such as book title, book cover, quantity, price, author, product description, and user reviews.

When users buy or sell books, they will be provided with an order code, buyer code, or seller code, details about the book product (book title, book cover, author, description, selling price, review), order status, total order value (order cost and shipping cost), and transaction time.

Users can check their order history after logging into the system.

Giao Hang Nhanh will receive orders from sellers including seller code and buyer code, order information, and seller order authentication to create a shipping order. In the shipping order, it will include information from the buyer's order sent to the seller and the shipping code of the shipping company.

The system can send notifications to users' email addresses when they successfully register.

## Topic Scope

### 1.3.1 Object and function limitations

*Users:*

* Registering and logging into the system.
* Browsing, searching, and purchasing books from the online store.
* Adding books to the shopping cart.
* Making online payments through the Momo electronic wallet.
* Posting book information for sale and managing the status of posted books.
* Tracking the status of orders and receiving books after placing an order.
* Can view order history after logging into the system.

*Administrators*:

* Logging into the system with administrative privileges.
* Managing user account information, including viewing information or deleting user accounts.
* Managing books, including reviewing detailed book information before users post them for sale or deleting books.
* Can view user orders if necessary.

*Fast Delivery Service:*

* Receiving information about confirmed orders from the website.
* Delivering goods to the address specified in the order.
* Updating shipping status and related information for users.

*System:*

* Integrating the Momo electronic wallet API to process payments.
* Sending notifications and updating order status to users and administrators.
* Managing the database, including storing information about books, users, and orders.

### 1.3.2 Technology limitations

* *Programming language:* No problem.
* *Database:* Difficulty in optimizing.
* *User interface:* Difficult to design user-friendly interface.
* *Security:* Decentralize user rights in the system: Send a request to the server to check the API key and the API key does not contain the user's role (permission). Ensure the security of cash flow.

## Practical Significance

The system enables users to buy and sell books conveniently from home without having to visit a bookstore. This is particularly beneficial in today's digital age, where online shopping is becoming increasingly popular.

The system provides users with a wide range of book titles to choose from, catering to various interests and preferences. This often offers more diversity than what can be found in a physical bookstore.

The system streamlines the process of buying and selling books, making the transaction process faster and more efficient. Users can easily search for specific books, make payments, and track their orders, all in one place.

Additionally, the system serves as a platform for book lovers to connect and share their passion for books. Users can share reviews and ratings, helping others make informed decisions about their purchases.

By creating favorable conditions for selling used books, this system will promote sustainability and reduce waste. This aligns with global efforts to promote a circular economy and minimize environmental impact.

The system provides economic opportunities for individuals to earn income from selling used books. It also supports the development of the online retail sector and contributes to the digital economy.

By making books more accessible, this system supports literacy and education. It allows more people to access a wealth of educational resources, contributing to lifelong learning and personal development.

## Report Structure

**Chapter 1: Overview of the system**

* 1. Introduction of the topic
  2. System Description
  3. Topic Scope
  4. Practical Significance
  5. Report Structure

**Chapter 2: System Analysis and Design**

2.1 Functional Requirements

2.2 Non-functional Requirements

2.3 Use-case Diagram

2.5 Entity-Relationship Diagram

2.6 Physical Database Diagram

**Chapter 3: System Implementation**

3.1 Technical Stack

3.2 System Architecture

**Chapter 4: Demonstration**

**Chapter 5: Conclusion**

5.1 Conclusion

5.2 Potential and Development Direction

**Requirements collection questions list**

# SYSTEM ANALYSIS AND DESIGN

## Functional Requirements

The system must allow users to register for an account.

The system will send a notification to the user's email address upon successful registration.

Administrators must verify the detailed information of books before approving listings or deleting books if they are not valid.

The system must display detailed book information so that users can search for books based on various criteria.

Users must be able to search for and purchase books from the website.

The system must allow users to select the quantity of books during checkout.

The system must display the order status when customers successfully place an order.

Users need to be able to view their order information.

The system must allow customers to rate the product once after receiving it.

The system must integrate the Momo e-wallet to provide customers with a transaction method.

The system must allow users to view their order history.

Administrators can view order information.

The system must allow user accounts to create new attributes.

Users can view their account information to update and make changes.

Users can close their accounts.

Administrators can view user information or delete user accounts.

The system must check to prevent duplicate orders and protect users' personal information.

## Non-functional Requirements

The web application will run on all web browser platforms.

The system must be available throughout the year.

The system's interaction time must not exceed 5 minutes for any function.

The system must have a capacity of at least 200 users at any given time during the day.

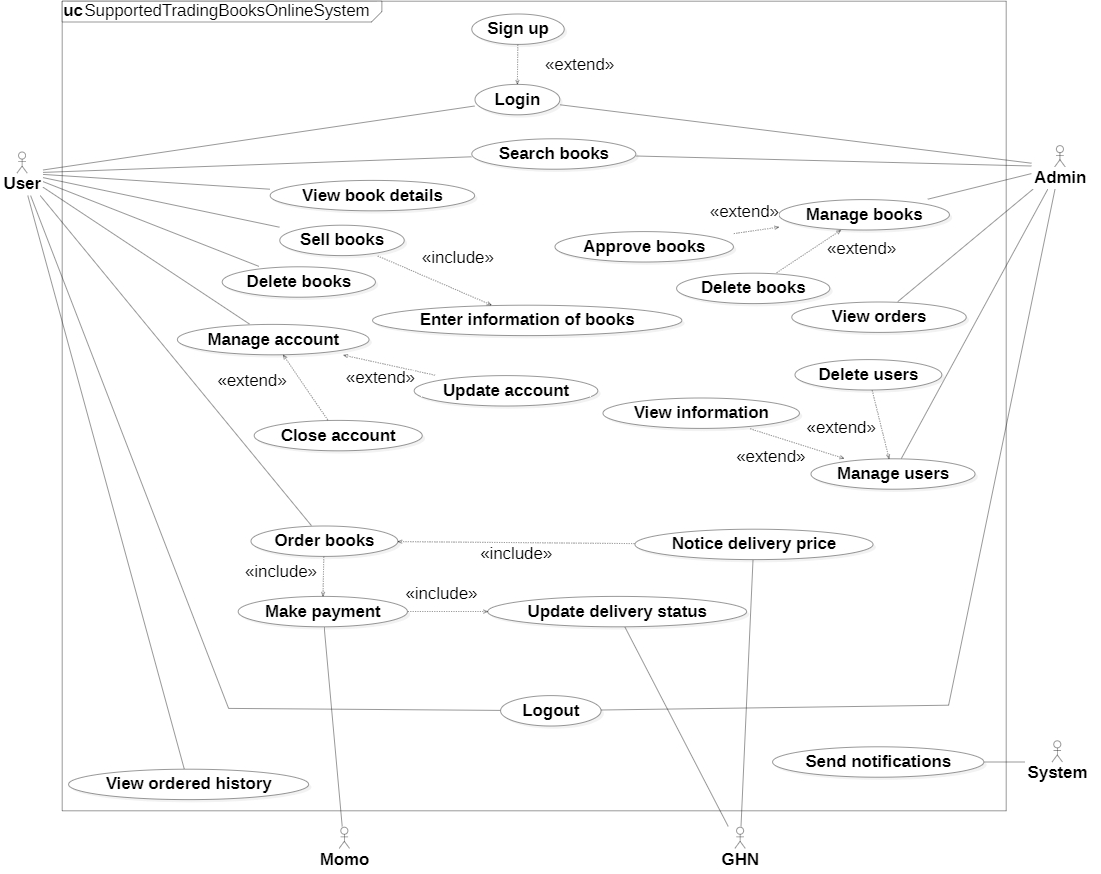
The system can only be disrupted for a maximum of 10 minutes.

The system must comply with the laws and regulations stipulated by the authorities in Vietnam.

The system must adhere to industry standards.

The system must be user-friendly for first-time users to navigate.

## Use-case Diagram



#### Figure 2.1: Use case diagram

### Use-case Overview

| **ID** | **Use Case’s Name** | **Description** | **Actor** |
| --- | --- | --- | --- |
| **UC01** | Sign up | Users and Admins will need to have an account to use the system. Users can sign up if they want to sign in to the website and use resources of the website. | User |
| **UC02** | Login | Users can login to the website by the account that they’ve created. Admins can login by using an account that is provided already. However, Admins can login by an account that they actively created by themselves. | User, Admin |
| **UC03** | Search Book | Users can find the books they want by filtering the keywords. | User |
| **UC04** | View book details | Users can see the details of the books if they want. | User |
| **UC05** | Enter information of books | Users can enter the information of the books (Name, Author, Quantity, Price, Description, etc) for sale. | User |
| **UC06** | Sell books | Users can post the books to the website that he/she wants to sell. | User |
| **UC07** | Delete books | Users can delete books they have posted for sale. | User |
| **UC08** | Update account | Users can update their account information. | User |
| **UC09** | Close account | Users can close their account if they want. | User |
| **UC10** | Order books | Users can place orders for books. | User |
| **UC11** | Notice delivery price | Users can view the delivery price. GHN provides the estimated shipping fee. | User, GHN |
| **UC12** | Make payment | Users can make payments by Momo for their orders. | User, Momo |
| **UC13** | Update delivery status | GHN updates the delivery status of orders. | GHN |
| **UC14** | View ordered history | Users can view their ordered history. | User |
| **UC15** | Approve books | Admins can approve books posted by users. | Admin |
| **UC16** | Delete books | Admins can delete books. | Admin |
| **UC17** | View orders | Admins can view orders made by users. | Admin |
| **UC18** | Delete users | Admins can delete user accounts. | Admin |
| **UC19** | View information | Admins can view user information. | Admin |
| **UC20** | Send notifications | The system can send notifications to users or admins. | System |
| **UC21** | Logout | Users and Admins can log out if they don’t want to use the system anymore. | User, Admin |

#### Table 2.2: Use-case Overview Table

### Use-case Description

| Use case: Sign up | | ID: UC01 | | Priority: High | |
| --- | --- | --- | --- | --- | --- |
| Actor: User | | | | | |
| Description: User creates a new account on the system. | | | | | |
| Trigger: User creates many accounts on the system. | | | | | |
| Pre-condition: None | | | | | |
| Basic Flow:   1. User navigates to the sign up page. 2. User enters their desired full name, email, phone number, address and password. 3. User submits the sign up form. 4. System verifies the entered information. 5. System creates a new user account. 6. System sends a confirmation email to the user’s email address. | | | Alternative Flow:  1a. If the email has already been used by another user:   1. The system displays an error message requesting the user to choose a different email. 2. The user adjusts their registration information. 3. Return to step 3 in the Basic Flow.   3a. If the user does not enter sufficient information or the information provided is invalid:   1. The system displays an error message requesting the user to enter complete and accurate information. 2. The user adjusts their registration information. 3. Return to step 3 in the Basic Flow. | | |
| Exception Flow:   * If the entered email or username is already in use, the system prompts the user to choose a different one. | | | | | |
| Post-condition:   * Users successfully sign up and can now access the system. | | | | | |
| Input | | Source | | Outputs | Destination |
| Email, full name, phone number, address, and password | | User input | | Confirmation email | User’s email inbox, display the notice on the screen. |

#### Table 2.3: Sign Up Use-case Description

| Use case: Login | | ID: UC02 | | Priority: High | |
| --- | --- | --- | --- | --- | --- |
| Actor: User, Admin | | | | | |
| Description: Users and admins log in to the system using their credentials. | | | | | |
| Trigger: User or admin wants to access the system. | | | | | |
| Pre-condition: None | | | | | |
| Basic Flow:   1. User/Admin navigates to the sign in page. 2. User/Admin enters their email and password. 3. User/Admin submits the sign in form. 4. System verifies the entered credentials. 5. System grants access to the user/admin if the credentials are valid. | | | Alternative Flow:   * If the user/admin forgets their password, they can click on the “Forgot password” link and follow the password recovery process. | | |
| Exception Flow:   * If the entered email or password is correct, the system displays an error message and prompts the user/admin to enter the correct credentials. | | | | | |
| Post-condition:   * User or admin successfully signs in and gains access to the system. | | | | | |
| Input | | Source | | Outputs | Destination |
| Email, password | | User input | | Access granted confirmation | User or admin interface |

#### Table 2.4: Login Use-case Description

### 

| Use case: Search books | | ID: UC03 | | Priority: Medium | |
| --- | --- | --- | --- | --- | --- |
| Actor: User, Admin | | | | | |
| Description: User searches for books based on specified criteria. | | | | | |
| Trigger: User wants to find books matching specific criteria. | | | | | |
| Pre-condition: User is logged in to the system. | | | | | |
| Basic Flow:   1. User navigates to the search section on the website. 2. User enters search criteria (e.g., title, author, genre). 3. System retrieves and displays books matching the criteria. | | | Alternative Flow:  3a. If no matching books are found, the system displays a message indicating no results. | | |
| Exception Flow:   * None. | | | | | |
| Post-condition:   * User views a list of books matching the search criteria. | | | | | |
| Input | | Source | | Outputs | Destination |
| Search criteria (e.g., title, author, genre, etc) | | User input | | List of matching books | User interface |

#### Table 2.5: Searching Books Use-case Description

| Use case: View book details | | ID: UC04 | | Priority: High | |
| --- | --- | --- | --- | --- | --- |
| Actor: User | | | | | |
| Description: Users can view detailed information about specific books if they wish. | | | | | |
| Trigger: User wants to view details about a specific book. | | | | | |
| Pre-condition: At least one book is available in the system. | | | | | |
| Basic Flow:   1. User searches for or selects a specific book on the website. 2. The system displays detailed information about the book including book title, author, image, description, price and reviews from other users. | | | Alternative Flow:   * None. | | |
| Exception Flow:   * If no book exists or the requested book cannot be found, the system displays an error message and asks the user to try again. | | | | | |
| Post-condition:   * The user can view detailed information about the book and proceed with any further actions (if applicable). | | | | | |
| Input | | Source | | Outputs | Destination |
| None | | User’s selection | | Detailed information about the book | User interface |

#### Table 2.6: View Book Details Use-case Description

| Use case: Enter information of books | | ID: UC05 | | Priority: High | |
| --- | --- | --- | --- | --- | --- |
| Actor: User | | | | | |
| Description: Users can enter detailed information about the books (Name, Author, Quantity, Price, Description, etc.) for sale. | | | | | |
| Trigger: User wants to enter information about a book for sale. | | | | | |
| Pre-condition: User is logged in and wants to sell a book. | | | | | |
| Basic Flow:   1. User navigates to the book management page on the website. 2. User selects the option "Add New Book". 3. User enters detailed information about the book, including title, author, quantity, price, description, etc. 4. User submits the information of the book. | | | Alternative Flow:   * None. | | |
| Exception Flow:   * If the user does not enter all the required information, the system displays an error message and prompts the user to complete all fields. * If there is an issue with adding book information, such as a database error or network problem, the system displays an error message and prompts the user to try again later. | | | | | |
| Post-condition:   * Information about the book is successfully added to the system. | | | | | |
| Input | | Source | | Outputs | Destination |
| Detailed information about the book | | User input | | Confirmation message | User interface |

#### Table 2.7: Enter Information Of Books Use-case Description

| Use case: Sell books | | ID: UC06 | | Priority: High | |
| --- | --- | --- | --- | --- | --- |
| Actor: User | | | | | |
| Description: Users can post the books they want to sell on the website. | | | | | |
| Trigger: User wants to sell a book. | | | | | |
| Pre-condition: User is logged in and has at least one book to sell.. | | | | | |
| Basic Flow:   1. User navigates to the “Sell Books” section on the website. 2. User selects the option to “Post a New Book for Sale”. 3. User enters the details of the book they want to sell, including the title, author, quantity, price, description, etc. 4. User submits the information of the book. 5. The system verifies the information and adds the book to the list of draft books available for sale. | | | Alternative Flow:   * None. | | |
| Exception Flow:   * If the user does not enter all the required information, the system displays an error message and prompts the user to complete all fields. * If there is an issue with posting the book, such as database error or network problem, the system displays an error message and prompts the user to try again later. | | | | | |
| Post-condition:   * The book is successfully posted for sale on the website. | | | | | |
| Input | | Source | | Outputs | Destination |
| Book details | | User input | | Confirmation message | User interface |

#### Table 2.8: Sell Books Use-case Description

### 

| Use case: Delete books | | ID: UC07 | | Priority: High | |
| --- | --- | --- | --- | --- | --- |
| Actor: User | | | | | |
| Description: Users can delete books that they no longer want to sell. | | | | | |
| Trigger: User wants to delete a book from their list of books for sale. | | | | | |
| Pre-condition: User is logged in and has books listed for sale. | | | | | |
| Basic Flow:   1. User navigates to the "Manage Books" section on the website. 2. User selects the book they want to delete from the list of their books for sale. 3. User confirms the deletion action. 4. The system removes the selected book from the list of books for sale. | | | Alternative Flow:   * None. | | |
| Exception Flow:   * If the user cancels the deletion action, the system aborts the process and returns the user to the previous state. | | | | | |
| Post-condition:   * The selected book is successfully removed from the list of books for sale. | | | | | |
| Input | | Source | | Outputs | Destination |
| Selection of the book to delete | | User input | | Confirmation message | User interface |

#### Table 2.9: Delete Books Use-case Description

| Use case: Update account | | ID: UC08 | | Priority: High | |
| --- | --- | --- | --- | --- | --- |
| Actor: User | | | | | |
| Description: Users can update their personal account information. | | | | | |
| Trigger: User wants to change or update their personal account information. | | | | | |
| Pre-condition: User is logged in to the system and wants to update personal account information. | | | | | |
| Basic Flow:   1. User navigates to the account update page on the website. 2. User edits or changes their personal information. 3. User confirms and saves the changes. 4. The system updates the user's account information. | | | Alternative Flow:   * None. | | |
| Exception Flow:   * If there is an error when updating account information, such as entering the wrong format or system error, the system displays an error message and prompts the user to try again later. | | | | | |
| Post-condition:   * User's account information is successfully updated. | | | | | |
| Input | | Source | | Outputs | Destination |
| Personal information to be updated | | Input from the user | | Confirmation message | User interface |

#### Table 2.10: Update Account Use-case Description

| Use case: Close account | | ID: UC09 | | Priority: High | |
| --- | --- | --- | --- | --- | --- |
| Actor: User | | | | | |
| Description: Users can close their account if they no longer wish to use the system. | | | | | |
| Trigger: User decides to close their account temporarily or permanently. | | | | | |
| Pre-condition: User is logged in to the system and wants to close their account. | | | | | |
| Basic Flow:   1. User navigates to the account settings or close account page on the website. 2. User selects the option to close their account. 3. User chooses whether to close the account. 4. User confirms the action to close the account. 5. The system either temporarily suspends or permanently deactivates the user's account based on the user's choice. | | | Alternative Flow:   * None. | | |
| Exception Flow:   * None. | | | | | |
| Post-condition:   * User's account is either temporarily suspended or permanently closed and deactivated. | | | | | |
| Input | | Source | | Outputs | Destination |
| Choice of temporary or permanent closure | | Input from the user | | Confirmation message | User interface |

#### Table 2.11: Close Account Use-case Description

| Use case: Order books | | ID: UC10 | | Priority: High | |
| --- | --- | --- | --- | --- | --- |
| Actor: User | | | | | |
| Description: Users can place an order to purchase books from the system. | | | | | |
| Trigger: User wants to buy books from the system. | | | | | |
| Pre-condition: User is logged in to the system and has selected the books they want to purchase. | | | | | |
| Basic Flow:   1. User adds desired books to their shopping cart on the website. 2. User proceeds to checkout to place the order. 3. User provides necessary details such as shipping address and payment information. 4. User confirms the order. 5. The system processes the order and sends a confirmation to the user. | | | Alternative Flow:   * None. | | |
| Exception Flow:   * If there is an error during the order process, such as payment failure or invalid shipping address, the system displays an error message and prompts the user to correct the issue. | | | | | |
| Post-condition:   * Order is successfully placed and the user receives confirmation. | | | | | |
| Input | | Source | | Outputs | Destination |
| Shipping details, payment information | | Input from the user | | Order confirmation | User interface |

#### Table 2.12: Order Books Use-case Description

| Use case: Notice delivery price | | ID: UC11 | | Priority: High | |
| --- | --- | --- | --- | --- | --- |
| Actor: User, GHN | | | | | |
| Description: Users receive information about the delivery price for their order. | | | | | |
| Trigger: User wants to know the delivery price for their order. | | | | | |
| Pre-condition: User has placed an order and GHN is responsible for delivery. | | | | | |
| Basic Flow:   1. The system retrieves the delivery price from the GHN service. 2. The system displays the delivery price to the user. | | | Alternative Flow:  None. | | |
| Exception Flow:  None. | | | | | |
| Post-condition:  User receives information about the delivery price for their order. | | | | | |
| Input | | Source | | Outputs | Destination |
| Request for delivery price | | Input from the user | | Delivery price information | User interface |

#### Table 2.13: Notice Delivery Price Use-case Description

### 

| Use case: Make payment | | ID: UC12 | | Priority: High | |
| --- | --- | --- | --- | --- | --- |
| Actor: User, Momo | | | | | |
| Description: Users make payment for their orders using the Momo e-wallet. | | | | | |
| Trigger: User wants to pay for their order. | | | | | |
| Pre-condition: User has completed the order process and selected Momo as the payment method. | | | | | |
| Basic Flow:   1. User selects the payment. 2. The system redirects the user to the Momo payment gateway. 3. User logs in to their Momo account and confirms the payment. 4. The system processes the payment and notifies the user of the successful transaction. | | | Alternative Flow:   * None. | | |
| Exception Flow:  If the payment fails, the system displays an error message and prompts the user to retry. | | | | | |
| Post-condition:  Payment for the order is successfully processed using Momo. | | | | | |
| Input | | Source | | Outputs | Destination |
| Payment details | | Input from the user | | Payment confirmation | User interface |

#### Table 2.14: Make Payment Use-case Description

### 

| Use case: Update delivery status | | ID: UC13 | | Priority: High | |
| --- | --- | --- | --- | --- | --- |
| Actor: GHN | | | | | |
| Description: GHN updates the delivery status of an order. System updates the delivery status | | | | | |
| Trigger: GHN receives information about the delivery status of an order. | | | | | |
| Pre-condition: GHN has received the order and is responsible for its delivery. | | | | | |
| Basic Flow:   1. GHN receives information about the delivery status of an order. 2. GHN updates the delivery status in the system. System updates delivery status. | | | Alternative Flow:   * None. | | |
| Exception Flow:   * None. | | | | | |
| Post-condition:   * The delivery status of the order is updated in the system. | | | | | |
| Input | | Source | | Outputs | Destination |
| Delivery status information | | Input from GHN | | Updated delivery status | System database |

#### Table 2.15: Update Delivery Status Use-case Description

### 

| Use case: View ordered history | | ID: UC14 | | Priority: High | |
| --- | --- | --- | --- | --- | --- |
| Actor: User | | | | | |
| Description: Users view their order history to track past purchases. | | | | | |
| Trigger: User wants to view their order history. | | | | | |
| Pre-condition: User is logged in to the system. | | | | | |
| Basic Flow:   1. User navigates to the order history section of their account. 2. The system retrieves and displays the user's past orders. | | | Alternative Flow:   * None. | | |
| Exception Flow:   * None. | | | | | |
| Post-condition:   * Users successfully view their order history. | | | | | |
| Input | | Source | | Outputs | Destination |
| User's request for order history | | User input | | List of past orders | User interface |

#### Table 2.16: View Ordered History Use-case Description

| Use case: Approve books | | ID: UC15 | | Priority: High | |
| --- | --- | --- | --- | --- | --- |
| Actor: Admin | | | | | |
| Description: Admin approves the books posted by users for sale. | | | | | |
| Trigger: Admin wants to review and approve books for sale. | | | | | |
| Pre-condition: Admin is logged in to the system. | | | | | |
| Basic Flow:   1. Admin navigates to the section for reviewing posted books. 2. Admin reviews the details and content of the books. 3. Admin approves the books that meet the criteria for sale. | | | Alternative Flow:   * None. | | |
| Exception Flow:   * None. | | | | | |
| Post-condition:   * Books approved by the admin are marked as available for sale. | | | | | |
| Input | | Source | | Outputs | Destination |
| Admin's review and approval of books | | Admin input | | Approved books status | System database |

#### Table 2.17: Approve Books Use-case Description

### 

| Use case: Delete books | | ID: UC16 | | Priority: High | |
| --- | --- | --- | --- | --- | --- |
| Actor: Admin | | | | | |
| Description: Admin deletes books from the system. | | | | | |
| Trigger: Admin wants to remove books from the system. | | | | | |
| Pre-condition: Admin is logged in to the system. | | | | | |
| Basic Flow:   1. Admin navigates to the section for managing books. 2. Admin selects the books to be deleted. 3. Admin confirms the deletion action. 4. System removes the selected books from the database. | | | Alternative Flow:   * None. | | |
| Exception Flow:   * None. | | | | | |
| Post-condition:   * Selected books are successfully deleted from the system. | | | | | |
| Input | | Source | | Outputs | Destination |
| Admin's selection of books for deletion | | Admin input | | Confirmation message | System database |

#### Table 2.18: Delete Books Use-case Description

### 

| Use case: View orders | | ID: UC17 | | Priority: High | |
| --- | --- | --- | --- | --- | --- |
| Actor: Admin | | | | | |
| Description: Admin views the list of orders placed by users. | | | | | |
| Trigger: Admin wants to check the list of orders. | | | | | |
| Pre-condition: Admin is logged in to the system. | | | | | |
| Basic Flow:   1. Admin navigates to the section for viewing orders. 2. System retrieves and displays the list of orders. | | | Alternative Flow:   * None. | | |
| Exception Flow:   * None. | | | | | |
| Post-condition:   * Admin successfully views the list of orders. | | | | | |
| Input | | Source | | Outputs | Destination |
| None | | System database | | List of orders | Admin interface |

#### Table 2.19: View Orders Use-case Description

| Use case: Delete users | | ID: UC18 | | Priority: High | |
| --- | --- | --- | --- | --- | --- |
| Actor: Admin | | | | | |
| Description: Admin deletes users from the system. | | | | | |
| Trigger: Admin wants to remove users from the system. | | | | | |
| Pre-condition: Admin is logged in to the system. | | | | | |
| Basic Flow:   1. Admin navigates to the section for managing users. 2. Admin selects the users to be deleted. 3. Admin confirms the deletion action. 4. System removes the selected users from the database. | | | Alternative Flow:   * None. | | |
| Exception Flow:   * None. | | | | | |
| Post-condition:   * Selected users are successfully deleted from the system. | | | | | |
| Input | | Source | | Outputs | Destination |
| Admin's selection of users for deletion | | Admin input | | Confirmation message | System database |

#### Table 2.20: Delete Users Use-case Description

### 

| Use case: View user information | | ID: UC19 | | Priority: High | |
| --- | --- | --- | --- | --- | --- |
| Actor: Admin | | | | | |
| Description: Admin views the information stored in the system. | | | | | |
| Trigger: Admin wants to access information from the system. | | | | | |
| Pre-condition: Admin is logged in to the system. | | | | | |
| Basic Flow:   1. Admin navigates to the section for viewing information. 2. System retrieves and displays the information stored in the database. | | | Alternative Flow:   * None. | | |
| Exception Flow:   * None. | | | | | |
| Post-condition:   * Admin successfully views the information from the system. | | | | | |
| Input | | Source | | Outputs | Destination |
| None | | System database | | Information displayed on the admin interface | Admin interface |

#### Table 2.21: View User Information Use-case Description

### 

| Use case: Send notifications | | ID: UC20 | | Priority: High | |
| --- | --- | --- | --- | --- | --- |
| Actor: System | | | | | |
| Description: The system sends notifications to users or admins. | | | | | |
| Trigger: The system needs to notify users or admins about specific events. | | | | | |
| Pre-condition: None. | | | | | |
| Basic Flow:   1. The system identifies the event that requires notification. 2. The system generates a notification message. 3. The system sends the notification to the intended recipients. | | | Alternative Flow:   * None. | | |
| Exception Flow:   * None. | | | | | |
| Post-condition:   * Users or admins receive the notifications from the system. | | | | | |
| Input | | Source | | Outputs | Destination |
| Event information, recipient information | | System | | Notification message | Users or admins |

#### Table 2.22: Send Notifications Use-case Description

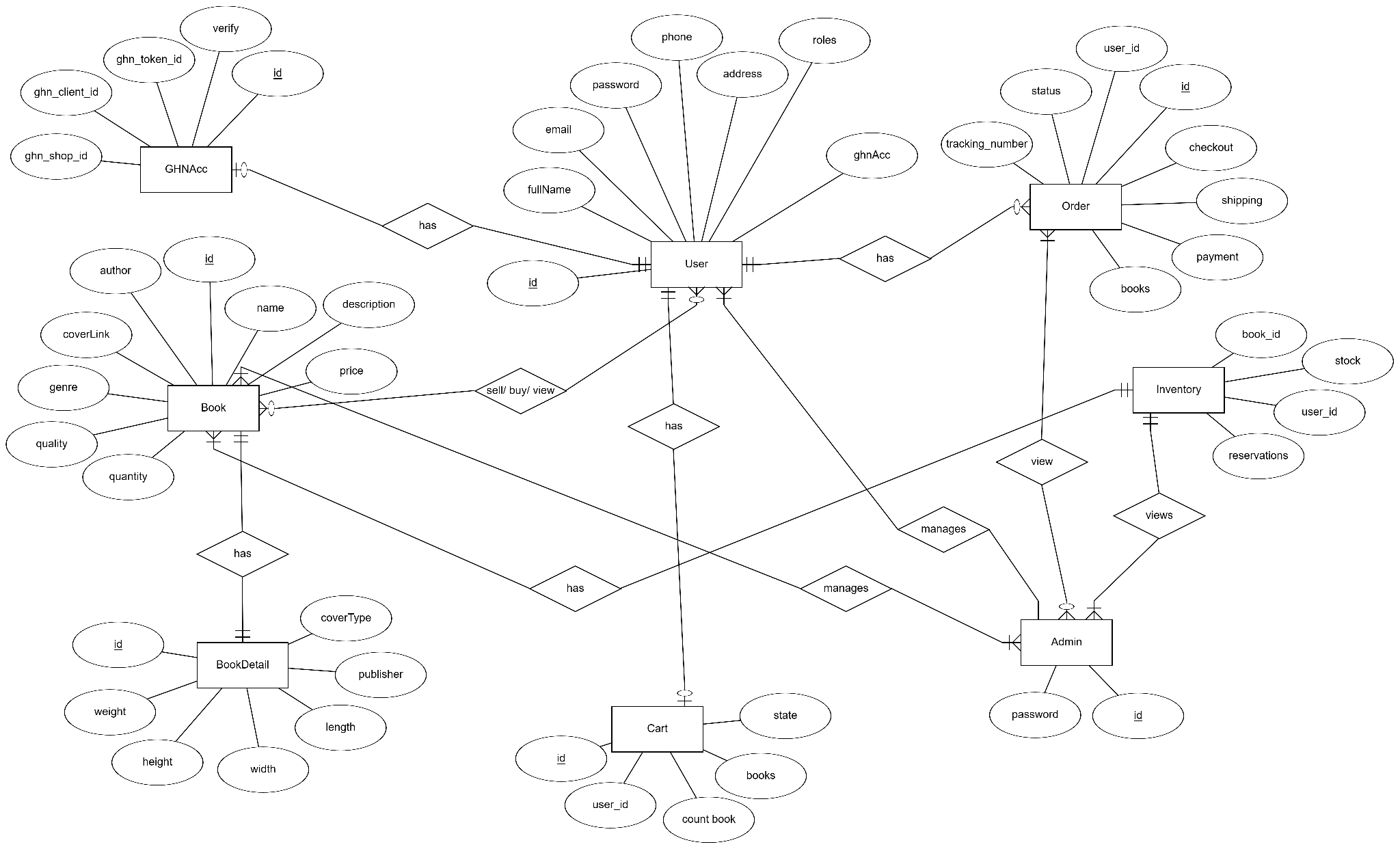
### 

| Use case: Logout | | ID: UC21 | | Priority: High | |
| --- | --- | --- | --- | --- | --- |
| Actor: User, Admin | | | | | |
| Description: This use case describes the process of logging out from the system for both users and administrators. | | | | | |
| Trigger: The user or administrator decides to log out of the system. | | | | | |
| Pre-condition: The user or administrator must be logged into the system. | | | | | |
| Basic Flow:   1. The user or administrator navigates to the logout option in the system interface. 2. The system logs out the user or administrator and redirects them to the login page. | | | Alternative Flow:   * None. | | |
| Exception Flow:   * None. | | | | | |
| Post-condition:   * The user or administrator is successfully logged out of the system. | | | | | |
| Input | | Source | | Outputs | Destination |
| None | | System interface | | Redirect to the login page | Login page |

#### Table 2.23: Logout Use-case Description

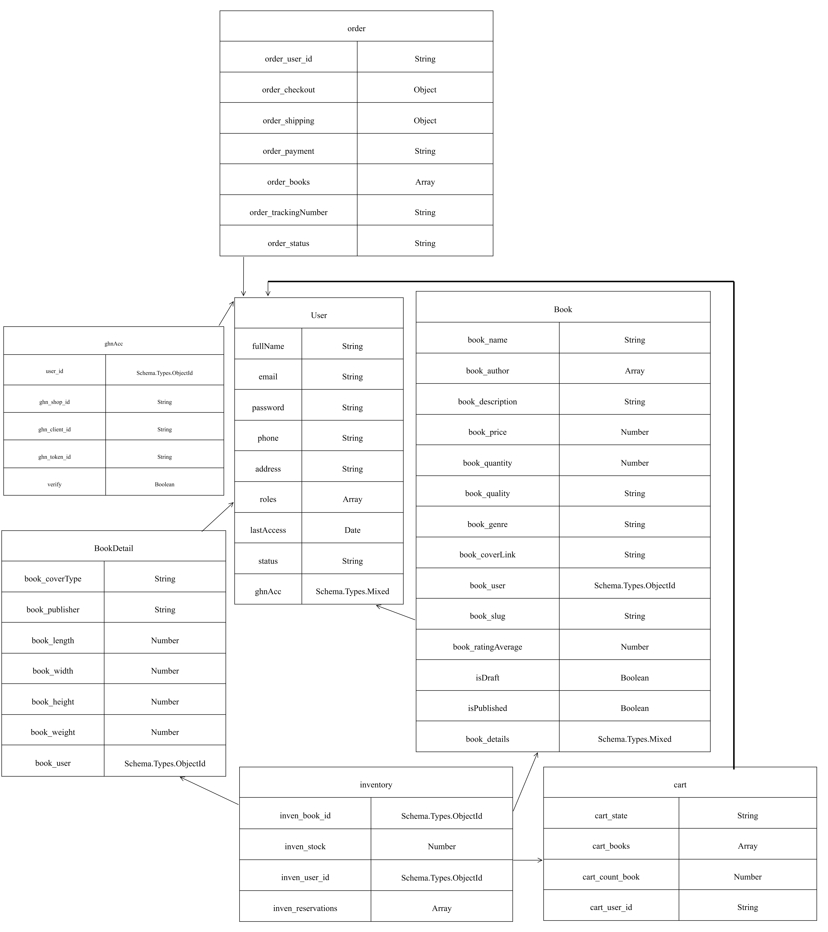
## Entity-Relationship Diagram

### 2.4.1 ERD Diagram



#### Figure 2.24: Supported Trading Books System Online ERD

### 2.4.2 Physical Database Diagram



#### Figure 2.25: Supported Trading Books Online System Physical Level Database Diagram

# SYSTEM IMPLEMENTATION

## Technical Stack

Some of the main goals of the system include that the website must run on popular web browsers and the system must also be stable as it needs to accommodate at least 200 users accessing it at any given time. The system should also be light and simple as we will prioritize performance and keep the user interface simple for users to navigate.

The goals can be achieved using the following technical stack:

* Website: HTML, CSS, Javascript
* Backend: Javascript
* Database: MongoDB.
* Frontend: Reactjs

***3.1.1 Javascript***

JavaScript serves as an integral part of our system, providing dynamic functionality and interactivity to enhance user experience. We have chosen JavaScript due to its widespread popularity and versatility in web development.

JavaScript offers excellent support for handling client-side scripting tasks, allowing us to create interactive elements and real-time updates on the user interface. Its extensive libraries and frameworks, such as Node.js and Express.js, empower us to build scalable and efficient backend services.

Additionally, JavaScript seamlessly integrates with MongoDB, our chosen database management system, facilitating smooth data handling and manipulation. This compatibility makes JavaScript a suitable choice for our technology stack.

Furthermore, JavaScript boasts a large and active community, providing ample resources, documentation, and support for developers. Its simple syntax makes it beginner-friendly, while the availability of online courses and tutorials makes it accessible to learners of all levels.

***3.1.2 MongoDB***

MongoDB serves as the backbone of our system's data management infrastructure. We have opted for MongoDB due to its flexible and scalable document-oriented database model, which aligns well with the dynamic nature of our application.

One of the key reasons for choosing MongoDB is its ability to handle unstructured and semi-structured data efficiently. This is particularly advantageous for us as we deal with diverse book-related information, including varying book formats, genres, and authors.

MongoDB's support for JSON-like documents allows us to store and retrieve data in a format that closely resembles the objects used in our application's codebase. This native compatibility streamlines the development process and enhances data consistency across the system.

Furthermore, MongoDB offers robust features for data replication, sharding, and high availability, ensuring that our system can scale seamlessly to accommodate growing data volumes and user loads. This scalability is crucial for supporting our platform's potential expansion and future growth.

Additionally, MongoDB's query language and indexing capabilities enable efficient data retrieval and analysis, empowering us to deliver responsive and performant user experiences. Its rich ecosystem of tools and libraries further enhances development productivity and system maintainability. MongoDB's combination of flexibility, scalability, and developer-friendly features makes it an ideal choice for managing the diverse and evolving data needs of our online book trading platform.

***3.1.3 Reactjs***

React.js plays a pivotal role in shaping the user interface (UI) of our online book trading platform. We have chosen React.js for its component-based architecture, virtual DOM rendering, and declarative approach to building user interfaces.

The component-based architecture of React.js aligns perfectly with our goal of creating modular and reusable UI components. This allows us to break down complex user interfaces into smaller, self-contained components, facilitating better code organization and maintenance.

Furthermore, React.js's virtual DOM rendering mechanism ensures optimal performance by efficiently updating only the necessary parts of the UI when data changes. This results in faster rendering and improved user experience, especially in applications with dynamic and interactive interfaces like ours.

The declarative nature of React.js simplifies UI development by allowing developers to describe the desired UI state and let React.js take care of updating the DOM to match that state. This abstraction minimizes the need for manual DOM manipulation and reduces the risk of bugs and inconsistencies.

Moreover, React.js enjoys strong community support and a rich ecosystem of libraries and tools, which accelerates development and enhances the overall productivity of our team. The availability of pre-built UI components, state management solutions, and routing libraries further streamlines the development process and enables us to focus on delivering value-added features.

Additionally, React.js's compatibility with modern JavaScript features, such as ES6 syntax and JSX syntax extensions, enhances code readability and maintainability. Its support for server-side rendering and mobile app development with React Native also provides opportunities for future expansion and platform diversification. React.js empowers us to build high-performance, scalable, and maintainable user interfaces for our online book trading platform, making it the preferred choice for front-end development.

***3.1.4 HTML/CSS/Javascript***

Last but not least, the combination of HTML, CSS and Javascript has always been the backbone of many modern websites. The simple and concise nature of these languages reduces the learning curve and makes it possible for beginners to build a simple website from scratch.

Because they are standard technologies, they can run on all different web browsers for a long time. It also has a large community that helps create various external libraries that can enhance the website development process.

## System Architecture

**3.2.1 API Docs**

1/ Sign up

1. Use this API to create accounts for users
2. Method: POST

{host}: localhost

{port}: 3056

{host}:{port}/v1/api/user/signup

**I/ Request Structure:**

| **Data field** | **Required** | **Data type** | **Description** |
| --- | --- | --- | --- |
| fullName | X | String | user’s full name |
| email | X | String | user’s email |
| password | X | String | password |
| address | X | String | home address |

{

"fullName": "Nguyễn Gia Minh",

"email": "ethanngia@gmail.com",

"password": "ethanngia",

"phone": "0356346175",

"address": "159/4/1/19 Phạm Thái Bường, Phường 4, Vĩnh Long, Vĩnh Long"

}

**II/ Response Structure:**

1. Successfully created

| **Data field** | **Description** |
| --- | --- |
| message | message |
| status | status code |
| fullName | user’s fullname |
| email | user’s email |
| phone | user’s mobile phone |
| address | user’s home address |
| roles | user's role |
| lastAccess | user’s last time logged in |
| status | Active/ Closed |
| \_id | user id |

{

"message": "Signed Up Successfully",

"status": 200,

"metadata": {

"fullName": "Nguyễn Gia Minh",

"email": "[ethanngia@gmail.com](mailto:ethanngia@gmail.com)",

"password": "$2b$10$Atj4rG18bWCr2j3uCBfDMOEiMFyln3v8FhgQcpFXzGwQWt7zmigC2",

"phone": "0356346175",

"address": "159/4/1/19 Phạm Thái Bường, Phường 4, Vĩnh Long, Vĩnh Long",

"roles": [

"0001",

"0005"

],

"lastAccess": null,

"status": "Active",

"\_id": "6640b4891e48d20852864ded",

"createdAt": "2024-05-12T12:22:33.376Z",

"updatedAt": "2024-05-12T12:22:33.376Z",

"\_\_v": 0

}

}

2. Error

| **Data field** | **Description** |
| --- | --- |
| status | ‘error’ |
| code | error status code |
| message | error message |

{

"status": "error",

"code": "ER",

"message": "User Already signed up."

}

**2/ Login**

1. Use this API to verify credentials
2. Method: POST

{host}: localhost

{port}: 3056

{host}:{port}/v1/api/user/login

**I/ Request Structure:**

| **Data field** | **Required** | **Data type** | **Description** |
| --- | --- | --- | --- |
| email | X | String | user’s email |
| password | X | String | password |

{

"email": "giamynguyen1906@gmail.com",

"password": "giamynguyen1906"

}

**II/ Response Structure:**1.Success

| **Data field** | **Description** |
| --- | --- |
| message | success message |

{

"message": "Logged in successfully"

}

2. Error

| **Data field** | **Description** |
| --- | --- |
| code | error status code |
| status | ‘error’ |
| message | error message |

User not signed up

{

"status": "error",

"code": "ER",

"message": "User not signed up."

}

Wrong password

{

"status": "error",

"code": "ER",

"message": "Wrong Credential."

}

**3/ Logout**

1. Method: POST

{host}: localhost

{port}: 3056

{host}:{port}/v1/api/user/logout

**I/ Request Structure:**1. Success

| **Data field** | **Description** |
| --- | --- |
| - | - |

2. Error

| **Data field** | **Description** |
| --- | --- |
| code | error status code |
| status | ‘error’ |
| message | error message |

HEADER: Not have refresh token

{

"status": "error",

"code": "ER",

"message": "No refresh token."

}

**4/ Update user account**  
1. Use this API when users need to change personal information.

2. Note:

* For addresses, users can only change when they have not registered to sell books. (GHN does not provide an API for updating shop addresses)
* Users can't change emails.
* This API is also used to update information about book sales claims (verify) in the ghnAcc Document (GHN Account)

3. Method: PATCH

{host}: localhost

{port}: 3056

{host}:{port}/v1/api/user/account/update/:id

localhost:3056/v1/api/user/account/update/6640d8effbe102cef9a6ef43

{

"fullName": "Gia Minh Nguyễn",

"phone": "0913750412"

}

**I/** **Response Structure**1. Success

{

"message": "Update Account successfully",

"status": 200,

"metadata": {

"\_id": "6640d8effbe102cef9a6ef43",

"fullName": "Gia Minh Nguyễn",

"email": "[ethanngia@gmail.com](mailto:ethanngia@gmail.com)",

"password":"$2b$10$Atj4rG18bWCr2j3uCBfDMOEiMFyln3v8FhgQcpFXzGwQWt7zmigC2",

"phone": "0913750412",

"address": "159/4/1/19 Phạm Thái Bường, Phường 4, Vĩnh Long, Vĩnh Long",

"roles": [

"0001",

"0005"

],

"lastAccess": null,

"status": "Active",

"createdAt": "2024-05-12T14:57:51.287Z",

"updatedAt": "2024-05-12T15:54:47.846Z",

"\_\_v": 0

}

}

**5/ Close account by user**  
1. Use this API when users close their accounts. After closing the account, the user no longer has access to system resources (roles)

2. Method: PATCH  
{host}: localhost

{port}: 3056

{host}:{port}/v1/api/user/account/close

**I/ Response Structure**

| **Data field** | **Description** |
| --- | --- |
| message | success message |
| status | status code |
| metadata | - |

{

"message": "Closed Account successfully",

"status": 200,

"metadata": {}

}

**6/ Create book**  
1. Use this API to create books for sale.

2. Method: POST

{host}: localhost

{port}: 3056

{host}:{port}/v1/api/book/create

**I/ Request Structure**

| **Data field** | **Required** | **Data type** | **Description** |
| --- | --- | --- | --- |
| book\_name | X | String | name of the book |
| book\_author | X | String | book’ authors |
| book\_description |  | String | description of the book |
| book\_price | X | Number | price of the book |
| book\_quantity | X | Number | quality |
| book\_quality | X | String | quality of the book [New/ Like New] |
| book\_genre | X | String | Fiction/ Non-Fiction |
| book\_coverLink |  | String | book cover image // thumbnail |
| book\_slug |  | String | slug |
| book\_ratingAverage |  | Number | Default: 4.5. (1.0 → 5.0) |
| book\_coverType | X | String | Paperback/ Hardcover |
| book\_publisher |  | String | publisher |
| book\_length | X | Number | length of the book |
| book\_width | X | Number | width of the book |
| book\_height | X | Number | height of the book |
| book\_weight | X | Number | weight of the book |

{

"book\_name": "Gentle Discipline",

"book\_author": ["Sarah Ockwell-Smith"],

"book\_description": "Using Emotional Connection--Not Punishment--to Raise Confident, Capable Kids",

"book\_price": 476000,

"book\_quality": 1,

"book\_quantity": "New",

"book\_genre": "Non-Fiction",

"book\_coverLink": "<https://i.imgur.com/3oZBFos.jpg>",

"book\_details": {

"book\_coverType": "Paperback",

"book\_publisher": "HighBridge, a Division of Recorded Books",

"book\_length": 20,

"book\_width": 16,

"book\_height": 3,

"book\_weight": 227

}

}

**II/ Response Structure**1. Success

| **Data field** | **Description** |
| --- | --- |
| message | success message |
| status | status code |
| book\_name | name of the book |
| book\_author | book’ authors |
| book\_description | description of the book |
| book\_price | price of the book |
| book\_quantity | quality |
| book\_quality | quality of the book [New/ Like New] |
| book\_genre | Fiction/ Non-Fiction |
| book\_coverLink | book cover image |
| book\_user | user id |
| book\_slug | slug |
| book\_ratingAverage | Default: 4.5. (1.0 → 5.0) |
| book\_coverType | Paperback/ Hardcover |
| book\_publisher | publisher |
| book\_length | length of the book |
| book\_width | width of the book |
| book\_height | height of the book |
| book\_weight | weight of the book |
| isDraft | Draft stage. Wating for admin to approve |
| isPublished | the book is officially published to the web |

{

"message": "Create book successfully",

"status": 200,

"metadata": {

"book\_name": "Gentle Discipline",

"book\_author": [

"Sarah Ockwell-Smith"

],

"book\_description": "Using Emotional Connection Not Punishment to Raise Confident, Capable Kids",

"book\_price": 476000,

"book\_quantity": 1,

"book\_quality": "New",

"book\_genre": "Non-Fiction",

"book\_coverLink": "<https://i.imgur.com/3oZBFos.jpg>",

"book\_user": "663ee0eb22b48f22b302ac94",

"book\_slug": "gentle-discipline",

"book\_ratingAverage": 4.5,

"isDraft": true,

"isPublished": false,

"book\_details": {

"book\_coverType": "Paperback",

"book\_publisher": "HighBridge, a Division of Recorded Books",

"book\_length": 20,

"book\_width": 16,

"book\_height": 3,

"book\_weight": 227

},

"\_id": "663f9a95d0274cd459356081",

"createdAt": "2024-05-11T16:19:33.539Z",

"updatedAt": "2024-05-11T16:19:33.539Z",

"\_\_v": 0

}

}

**7/ Get all published books**

1. Use this API to get all officially listed books for sale

2. Method: GET

{host}: localhost

{port}: 3056

{host}:{port}/v1/api/book/

**I/ Response Structure**1. Success

| **Data field** | **Description** |
| --- | --- |
| message | message |
| status | status code |
| metadata | list of published books |

{

"message": "Get all published books successfully",

"status": 200,

"metadata": [

{

"\_id": "663f9c6424c94f18854a6f52",

"book\_name": "Gentle Discipline",

"book\_price": 476000,

"book\_coverLink": "<https://i.imgur.com/3oZBFos.jpg>"

}

]

}

**8/ Get all draft books**  
1. Use this API to get all books listed for sale. However, the books, which are in "Draft" status, have not been published on the website.

2. Method: GET

{host}: localhost

{port}: 3056

{host}:{port}/v1/api/book/draft/all

**I/** **Response Structure**

| **Data field** | **Description** |
| --- | --- |
| message | success message |
| status | status code |
| \_id | book id |
| book\_name | name of the book |
| book\_author | book’ authors |
| book\_description | description of the book |
| book\_price | price of the book |
| book\_quantity | quality |
| book\_quality | quality of the book [New/ Like New] |
| book\_genre | Fiction/ Non-Fiction |
| book\_coverLink | book cover image |
| book\_user | user id |
| book\_slug | slug |
| book\_ratingAverage | Default: 4.5. (1.0 → 5.0) |
| book\_coverType | Paperback/ Hardcover |
| book\_publisher | publisher |
| book\_length | length of the book |
| book\_width | width of the book |
| book\_height | height of the book |
| book\_weight | weight of the book |
| isDraft | Draft stage. Waiting for admin to approve |
| isPublished | the book is officially published to the web |

{

"message": "Get all draft book successfully",

"status": 200,

"metadata": [

{

"\_id": "663f9c6424c94f18854a6f52",

"book\_name": "Gentle Discipline",

"book\_author": [

"Sarah Ockwell-Smith"

],

"book\_description": "Using Emotional Connection Not Punishment to Raise Confident, Capable Kids",

"book\_price": 476000,

"book\_quantity": 1,

"book\_quality": "New",

"book\_genre": "Non-Fiction",

"book\_coverLink": "<https://i.imgur.com/3oZBFos.jpg>",

"book\_user": "663ee0eb22b48f22b302ac94",

"book\_slug": "gentle-discipline",

"book\_ratingAverage": 4.5,

"book\_details": {

"book\_coverType": "Paperback",

"book\_publisher": "HighBridge, a Division of Recorded Books",

"book\_length": 20,

"book\_width": 16,

"book\_height": 3,

"book\_weight": 227

},

"createdAt": "2024-05-11T16:27:16.085Z",

"updatedAt": "2024-05-12T02:59:18.927Z",

"\_\_v": 0

}

]

}

**9/ Get user’s all published books**  
1. Use this API to get all the official books for sale by 1 user

2. Method: GET

{host}: localhost

{port}: 3056

{host}:{port}/v1/api/book/published/:id

**I/ Response Structure**1. Success

| **Data field** | **Description** |
| --- | --- |
| message | message |
| status | status code |
| metadata | list of user’ published books |

{

"message": "Get user published books successfully",

"status": 200,

"metadata": [

{

"\_id": "663fa1281b826da79390432c",

"book\_name": "The Art of War",

"book\_author": [

"Sun Tzu",

"Aidan Gillen"

],

"book\_description": "The 13 chapters of The Art of War, each devoted to one aspect of warfare, were compiled by the high-ranking Chinese military general, strategist, and philosopher Sun-Tzu. In spite of its battlefield specificity, The Art of War has found new life in the modern age, with leaders in fields as wide and far-reaching as world politics, human psychology, and corporate strategy finding valuable insight in its timeworn words.",

"book\_price": 259000,

"book\_quantity": 1,

"book\_quality": "New",

"book\_genre": "Fiction",

"book\_coverLink": "<https://i.imgur.com/9S1XLzZ.jpg>",

"book\_user": "663ee0eb22b48f22b302ac94",

"book\_slug": "the-art-of-war",

"book\_ratingAverage": 4.5,

"book\_details": {

"book\_coverType": "HardCover",

"book\_publisher": "Macmillan Collector's Library",

"book\_length": 20,

"book\_width": 16,

"book\_height": 3,

"book\_weight": 136

},

"createdAt": "2024-05-11T16:47:36.888Z",

"updatedAt": "2024-05-12T02:47:51.253Z",

"\_\_v": 0

}

]

}

2. Error

| **Data field** | **Description** |
| --- | --- |
| code | error status code |
| status | ‘error’ |
| message | error message |

Invalid user id

{

"status": "error",

"code": "ER",

"message": "User not found."

}

**10/ Get user’s all draft books**  
1. Use this API to get all books listed for sale by users but not yet approved (status: "draft")

2. Method: GET  
{host}: localhost

{port}: 3056

{host}:{port}/v1/api/book/draft/user/:id

Note: id - user id

**I/ Response Structure**1. Success

| **Data field** | **Description** |
| --- | --- |
| message | success message |
| status | status code |
| metadata | list of user’ all draft book |

{

"message": "Get user draft books successfully",

"status": 200,

"metadata": [

{

"\_id": "663fa1281b826da79390432c",

"book\_name": "The Art of War",

"book\_author": [

"Sun Tzu",

"Aidan Gillen"

],

"book\_description": "The 13 chapters of The Art of War, each devoted to one aspect of warfare, were compiled by the high-ranking Chinese military general, strategist, and philosopher Sun-Tzu. In spite of its battlefield specificity, The Art of War has found new life in the modern age, with leaders in fields as wide and far-reaching as world politics, human psychology, and corporate strategy finding valuable insight in its timeworn words.",

"book\_price": 259000,

"book\_quantity": 1,

"book\_quality": "New",

"book\_genre": "Fiction",

"book\_coverLink": "<https://i.imgur.com/9S1XLzZ.jpg>",

"book\_user": "663ee0eb22b48f22b302ac94",

"book\_slug": "the-art-of-war",

"book\_ratingAverage": 4.5,

"book\_details": {

"book\_coverType": "HardCover",

"book\_publisher": "Macmillan Collector's Library",

"book\_length": 20,

"book\_width": 16,

"book\_height": 3,

"book\_weight": 136

},

"createdAt": "2024-05-11T16:47:36.888Z",

"updatedAt": "2024-05-12T02:47:51.253Z",

"\_\_v": 0

}

]

}

2. Error

| **Data field** | **Description** |
| --- | --- |
| code | error status code |
| status | ‘error’ |
| message | error message |

Invalid user id

{

"status": "error",

"code": "ER",

"message": "User not found."

}

**11/ Publish book**  
1. This API is used to publicize the user's book on the website, which occurs after the Admin reviews the book (Allow sale)

2. Method: POST  
{host}: localhost

{port}: 3056

{host}:{port}/v1/api/book/publish/:id

Note: id - book id

**I/ Response Structure**1. Success

| **Data field** | **Description** |
| --- | --- |
| message | message |
| status | status code |
| metadata | 1 |

{

"message": "Publish book successfully",

"status": 200,

"metadata": 1

}

2. Error

| **Data field** | **Description** |
| --- | --- |
| code | error status code |
| status | ‘error’ |
| message | error message |

{

"status": "error",

"code": "ER",

"message": "Invalid Book."

}

**12/ Unpublish book**  
1. Use this API to hide books for sale (books in "draft" status), which occurs when users after being browsed by admins, books are public on the website and users hide books when users no longer need to sell books.

2. Method: POST  
  
{host}: localhost

{port}: 3056

{host}:{port}/v1/api/book/unpublish/:id

Note: id - book id

**I/ Response Structure**1. Success

| **Data field** | **Description** |
| --- | --- |
| message | message |
| status | status code |
| metadata | 1 |

{

"message": "Unpublish book successfully",

"status": 200,

"metadata": 1

}

2. Error

| **Data field** | **Description** |
| --- | --- |
| code | error status code |
| status | ‘error’ |
| message | error message |

{

"status": "error",

"code": "ER",

"message": "Invalid Book."

}

**13/ Search Book By User**  
1. Use this API to search for (public) books based on keyword/text.

2. Method: GET  
  
{host}: localhost

{port}: 3056

{host}:{port}/v1/api/book/search/:keySearch

For example: localhost:3056/v1/api/book/search/war

**I/ Response Structure:**

| **Data field** | **Description** |
| --- | --- |
| message | error message |
| status | status code |
| metadata | list of books searched |

{

"message": "Get all search books successfully",

"status": 200,

"metadata": [

{

"\_id": "663fa1281b826da79390432c",

"book\_name": "The Art of War",

"book\_author": [

"Sun Tzu",

"Aidan Gillen"

],

"book\_description": "The 13 chapters of The Art of War, each devoted to one aspect of warfare, were compiled by the high-ranking Chinese military general, strategist, and philosopher Sun-Tzu. In spite of its battlefield specificity, The Art of War has found new life in the modern age, with leaders in fields as wide and far-reaching as world politics, human psychology, and corporate strategy finding valuable insight in its timeworn words.",

"book\_price": 259000,

"book\_quantity": 1,

"book\_quality": "New",

"book\_genre": "Fiction",

"book\_coverLink": "<https://i.imgur.com/9S1XLzZ.jpg>",

"book\_user": "663ee0eb22b48f22b302ac94",

"book\_slug": "the-art-of-war",

"book\_ratingAverage": 4.5,

"book\_details": {

"book\_coverType": "HardCover",

"book\_publisher": "Macmillan Collector's Library",

"book\_length": 20,

"book\_width": 16,

"book\_height": 3,

"book\_weight": 136

},

"createdAt": "2024-05-11T16:47:36.888Z",

"updatedAt": "2024-05-12T04:51:35.167Z",

"\_\_v": 0,

"score": 0.75

}

]

}

localhost:3056/v1/api/book/search/account

{

"message": "Get all search books successfully",

"status": 200,

"metadata": []

}

**14/ Get a book**  
1. Use this API to get details of a book.

2. Method: GET

{host}: localhost

{port}: 3056

{host}:{port}/v1/api/book/:id

Note: id - book id

For example: localhost:3056/v1/api/book/663f9c6424c94f18854a6f52

**I/ Response Structure**1. Success

| **Data field** | **Description** |
| --- | --- |
| message | error message |
| status | status code |
| metadata | book details |

{

"message": "Get book successfully",

"status": 200,

"metadata": {

"\_id": "663f9c6424c94f18854a6f52",

"book\_name": "Gentle Discipline",

"book\_author": [

"Sarah Ockwell-Smith"

],

"book\_description": "Using Emotional Connection Not Punishment to Raise Confident, Capable Kids",

"book\_price": 476000,

"book\_quantity": 1,

"book\_quality": "New",

"book\_genre": "Non-Fiction",

"book\_coverLink": "<https://i.imgur.com/3oZBFos.jpg>",

"book\_user": "663ee0eb22b48f22b302ac94",

"book\_slug": "gentle-discipline",

"book\_ratingAverage": 4.5,

"book\_details": {

"book\_coverType": "Paperback",

"book\_publisher": "HighBridge, a Division of Recorded Books",

"book\_length": 20,

"book\_width": 16,

"book\_height": 3,

"book\_weight": 227

},

"createdAt": "2024-05-11T16:27:16.085Z",

"updatedAt": "2024-05-12T08:26:07.505Z"

}

}

No book found:

{

"message": "Get book successfully",

"status": 200,

"metadata": null

}

**3.2.2 Scenario**

1. **About book selling service**

* Users are obliged to enter the following information accurately:
  + Full name → This is the user's store name created in Quick Delivery (GHN)
  + Address
  + Phone number
* Users must confirm registration to sell books.
* In Fast Delivery:
  + The main store is the responsibility of the BookTrader team.
  + Substore is the name of the user registered to sell books on the BookTrader system.
  + Users do not have access to the Quick Delivery system to query order information. All order information is provided through the BookTrader system.

# DEMONSTRATION

| *Order* | *Specification* | *User Interface* |
| --- | --- | --- |
| 1 | Step 1: User creates account  Step 2: User logs into the system  Step 3: Users create books for sale.  Step 4: User updates account.  Note:   * As for the address, users can only change it when they have not registered to sell books. (GHN does not provide API to update shop address) * Users cannot change their email. | Step 1:  Step 2:  Step 3:  Step 4:  Step 5:  Step 6:  Step 7:  Step 8:  Step 9: |

# CONCLUSION

* 1. **Conclusion**

In summary, the online book trading support system has integrated many business processes from various popular platforms in the e-commerce field. We have developed a simple but prioritized platform user experience, and at the same time support users in searching, buying and selling books easily.

Our platform is designed to work on any modern device and is easily accessible to people of all ages via web browser.

Through this project, we were exposed to many new techniques and business concepts, from system implementation to business processes such as account verification and asset management. We also explored and learned a lot about popular technologies in the online business sector and how they can interact with our systems.

Ultimately, the project helped us better understand the process of selecting the right technologies and how they make an impact on the system. We are confident that, with the learning and experience accumulated from this project, we have accelerated our development in the field of e-commerce and information technology.

* 1. **Potential and Development Direction**

Going forward, we will continue to gather constructive feedback and ideas from users, the book community, and other stakeholders to improve the system's online book trading process.

We will also research and apply new technologies to enhance user experience and optimize business processes. Implementing new business processes to create a competitive and efficient environment is our top priority in the development of our online book trading system.

# REQUIREMENTS COLLECTION QUESTIONS LIST

**1. List of requirements gathering questions:**

***Interview Questions for Users:***

1. Can you describe your experience with buying books online?
2. Which platforms have you used before?
3. What are the main challenges you face when buying books online?
4. What features do you consider essential for an online platform for buying and selling books?

***Interview Questions for Administrators/System:***

1. In your opinion, what is the most important feature that users expect from an online book transaction system?
2. How do you plan to manage books and orders on the system?
3. What security features does the system need to ensure the safety of user information and transactions?

**2. Questionnaire to collect requirements**

|  | **Question** | **Question Type** | **Importance Level** | **Note** |
| --- | --- | --- | --- | --- |
| **1** | Have you ever purchased books online before? | Single-choice | High | Inquire about users' past experience with purchasing books online. |
| **2** | What do you expect from an ideal online book-selling website? | Open-ended | High | Gather expectations and specific requirements from users regarding their online book-buying experience. |
| **3** | What are the basic features you want the system to provide? | Open-ended | High | Identify important features that users expect from the system. |
| **4** | What factors do you usually consider when deciding to purchase a book? | Open-ended | Medium | Reflect users' personal opinions on important factors when buying books. |
| **5** | Do you want the system to provide a book rating and review feature? | Single-choice | Medium | Determine users' expectations for the book rating and review feature. |
| **6** | Which payment method do you prefer when buying books online? | Multiple-choice | Medium | Identify users' preferences for payment methods when purchasing books online. |
| **7** | What are your expectations for the user interface of the system? | Open-ended | High | Collect information on user interface requirements from users. |
| **8** | Do you have any other requirements or suggestions for the system? | Open-ended | Low | Open up to users to share any other requirements or suggestions they may have. |

# REFERENCES