Project Report: Online Bookstore Management System

411021390 賈俊佑

1. Introduction

This project aims to design and implement an online bookstore management system that allows users to browse books, add them to the shopping cart, and place orders. Additionally, it provides administrators with the functionality to manage book inventory and orders. The front-end of the system is built using HTML and CSS, and the back-end uses PHP for dynamic content generation and MySQL for the database.

2.System structure

System Structure
The Online Bookstore Management System is designed with a three-tier architecture consisting of the Presentation Layer, the Business Logic Layer, and the Data Access Layer. Below is a detailed explanation of each layer and their interaction within the system:

2.1 Presentation Layer

The Presentation Layer is the front-end of the application, which includes HTML, CSS, and JavaScript files. This layer is responsible for displaying the user interface and handling user interactions.

HTML: Used for structuring the content on web pages.
CSS: Used for styling the web pages to make them visually appealing.
JavaScript: Used for enhancing user interactions and providing dynamic content updates without requiring page reloads.
This layer communicates with the Business Logic Layer through form submissions and AJAX requests, which are handled by PHP scripts.

2.2 Business Logic Layer

The Business Logic Layer is the back-end of the application, implemented using PHP. This layer contains the core functionality and business rules of the application.

PHP Scripts: Handle requests from the Presentation Layer, process data, and apply

business logic.
The PHP scripts in this layer interact with the Data Access Layer to retrieve or store data in the database. They also generate dynamic HTML content that is sent back to the Presentation Layer for display.

2.3 Data Access Layer

The Data Access Layer is responsible for interacting with the database. This layer is implemented using MySQL and contains the database schema and SQL queries.

MySQL Database: Stores all the data related to users, books, orders, and order items.
SQL Queries: Used within PHP scripts to perform CRUD (Create, Read, Update, Delete) operations on the database.
This layer ensures data integrity and handles all database interactions required by the Business Logic Layer.

3. Database Design

Database Structure

3.1 User

UserID (int, primary key)
Name (varchar)
Account (varchar)
Password (varchar)
Address (varchar)

3.2 Book

BookID (int, primary key)
Title (varchar)
Author (varchar)
Genre (varchar)
ISBN (varchar)
Price (decimal)
StockQuantity (int)

3.3 Order

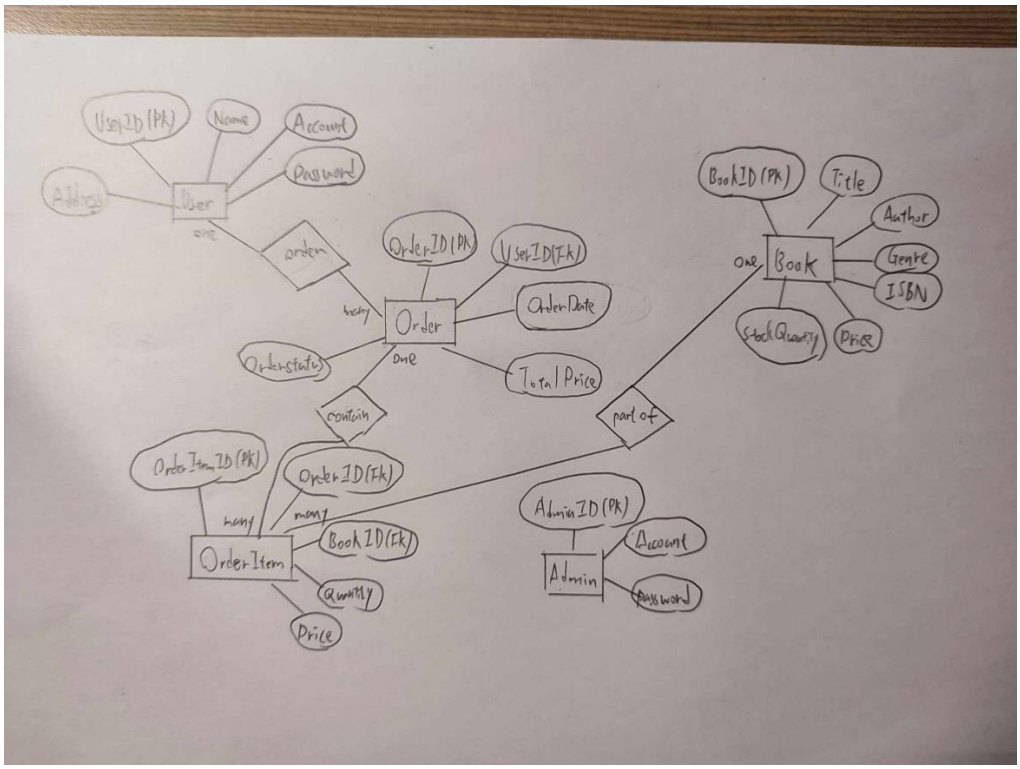
OrderID (int, primary key)
UserID (int, foreign key)
OrderDate (datetime)
TotalPrice (decimal)
OrderStatus (varchar)

3.4 OrderItem

OrderItemID (int, primary key)
OrderID (int, foreign key)
BookID (int, foreign key)
Quantity (int)
Price (decimal)

3.5 Admin

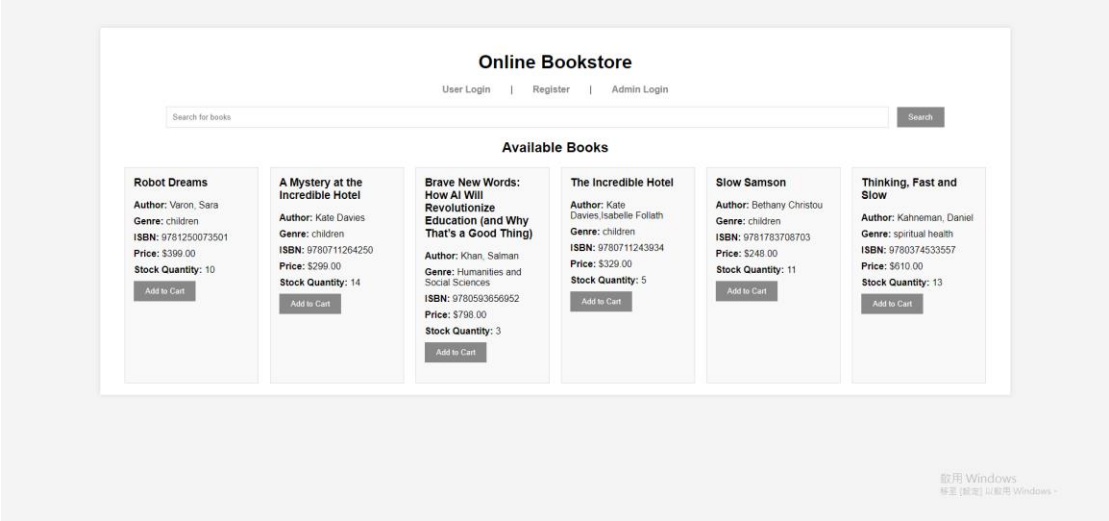
AdminID (int, primary key)
Account (varchar)
Password (varchar)
EE/EER graph



4.System implement

4.1 Homepage(index.php)

The homepage displays the list of available books and provides navigation links to other parts of the application such as the shopping cart, user profile, and admin interface. Users can search for books, and add books to their shopping cart.



Important SQL Queries:

a.Retrieve All Books

SELECT * FROM Book;

Purpose: To display all available books on the homepage.

b. Search Books

SELECT * FROM Book WHERE Title LIKE ? OR Author LIKE ? OR Genre LIKE ? OR ISBN LIKE ?;

Purpose: To display all available books on the homepage.

c. Check Existing Pending Order

SELECT OrderID FROM `Order` WHERE UserID = ? AND OrderStatus = 'Pending';

Purpose: To check if the user already has a pending order.

d. Create New Order

INSERT INTO `Order` (UserID, OrderDate, TotalPrice, OrderStatus) VALUES (?, ?, 0, 'Pending');

Purpose: To create a new order for the user if no pending order exists.

e. Update Order Item Quantity

UPDATE OrderItem SET Quantity = Quantity + 1 WHERE OrderID = ? AND BookID = ?;

Purpose: To update the quantity of a book in the user's shopping cart.

f. Add New Order Item

INSERT INTO OrderItem (OrderID, BookID, Quantity, Price) VALUES (?, ?, 1, (SELECT Price FROM Book WHERE BookID = ?));

Purpose: To add a new book to the user's shopping cart.

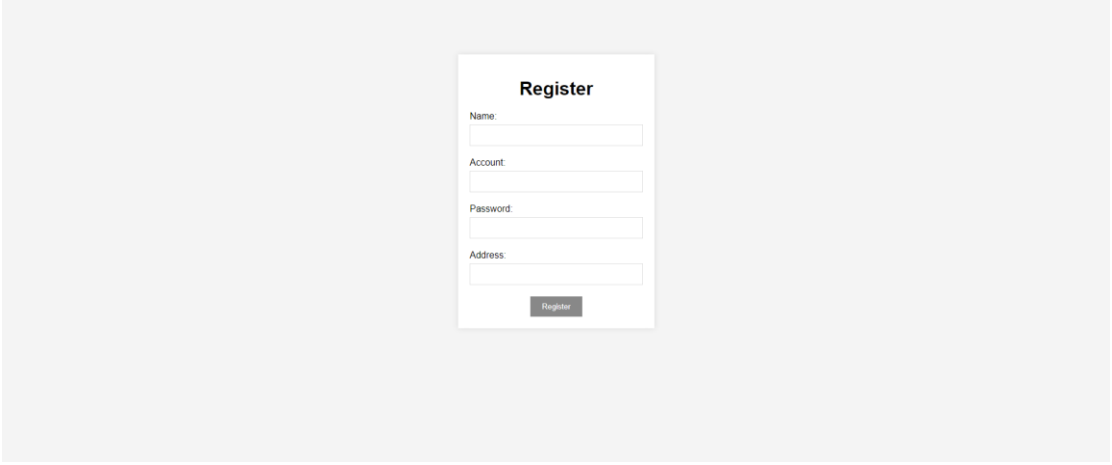
g. Update Order Total Price

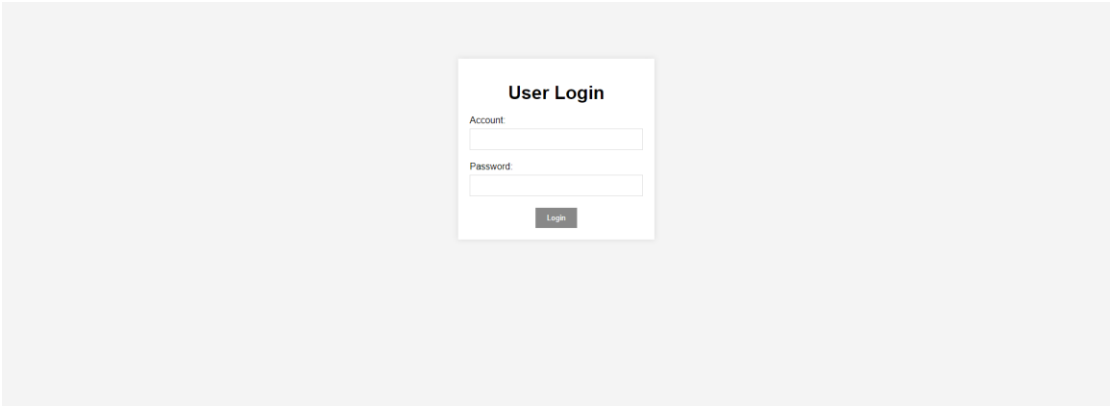
UPDATE `Order` SET TotalPrice = (SELECT SUM(Price * Quantity) FROM OrderItem WHERE OrderID = ?) WHERE OrderID = ?;

Purpose: To update the total price of the order based on the items in the cart.

4.2 User Registration and Login(login.php and register.php)

Users can register for an account and log into the system using their credentials.





Important SQL Queries:

a.Check if the account is already registered(register.php):

SELECT * FROM User WHERE Account = ?

Purpose: This query checks if there is already a user with the given account name in the database. It helps to prevent duplicate account registrations.

b. Insert new user data(register.php):

INSERT INTO User (Name, Account, Password, Address) VALUES (?, ?, ?, ?)

Purpose: This query inserts the new user's data into the database, including their

name, account name, hashed password, and address. It creates a new user record in the User table.

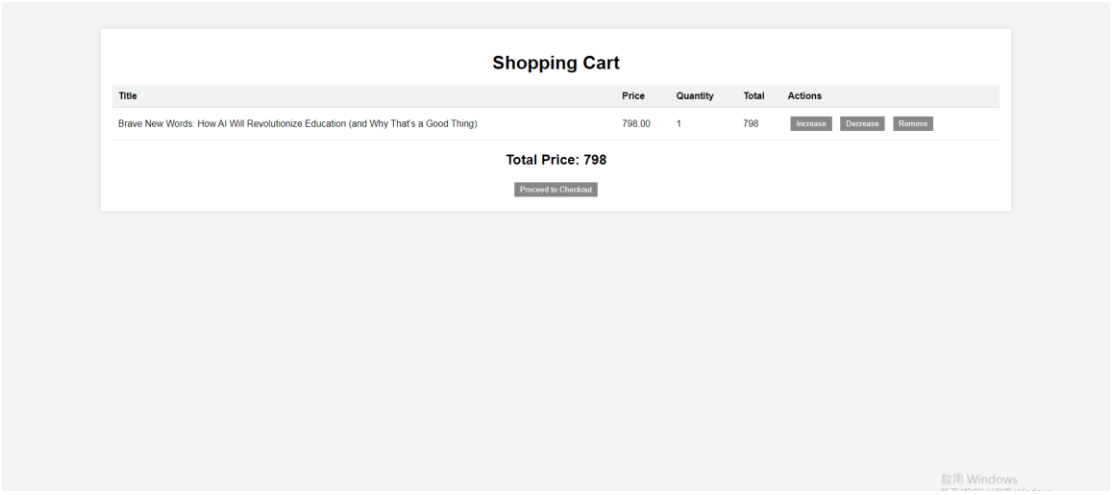
c.Verify User Credentials(login.php):

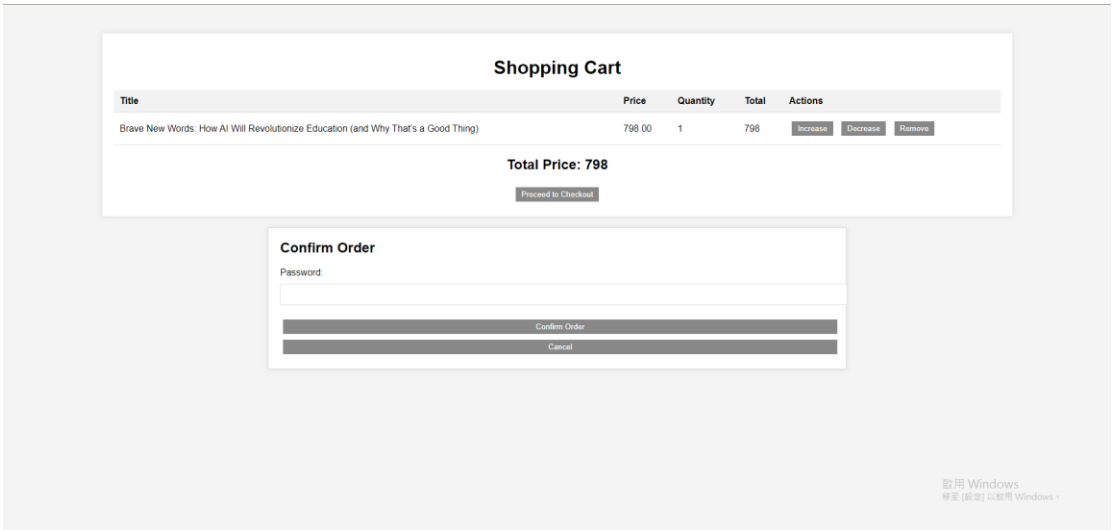
SELECT UserID, Password FROM User WHERE Account = ?;

Purpose: This query retrieves the user's ID and hashed password from the database to verify the login credentials.

4.3 Shopping cart management and transaction confirm(cart.php)

Manages the shopping cart by allowing users to add, remove, or adjust quantities of items in their cart. It also handles the order confirmation process, including password validation and stock adjustments.





Important SQL Queries:

a.Remove Item from Cart:

DELETE FROM OrderItem WHERE OrderID = (SELECT OrderID FROM `Order` WHERE UserID = ? AND OrderStatus = 'Pending') AND BookID = ?;

Purpose: This query removes a specific book from the user's shopping cart.
b. Increase Item Quantity in Cart:
UPDATE OrderItem SET Quantity = Quantity + 1 WHERE OrderID = (SELECT OrderID FROM `Order` WHERE UserID = ? AND OrderStatus = 'Pending') AND BookID = ?;

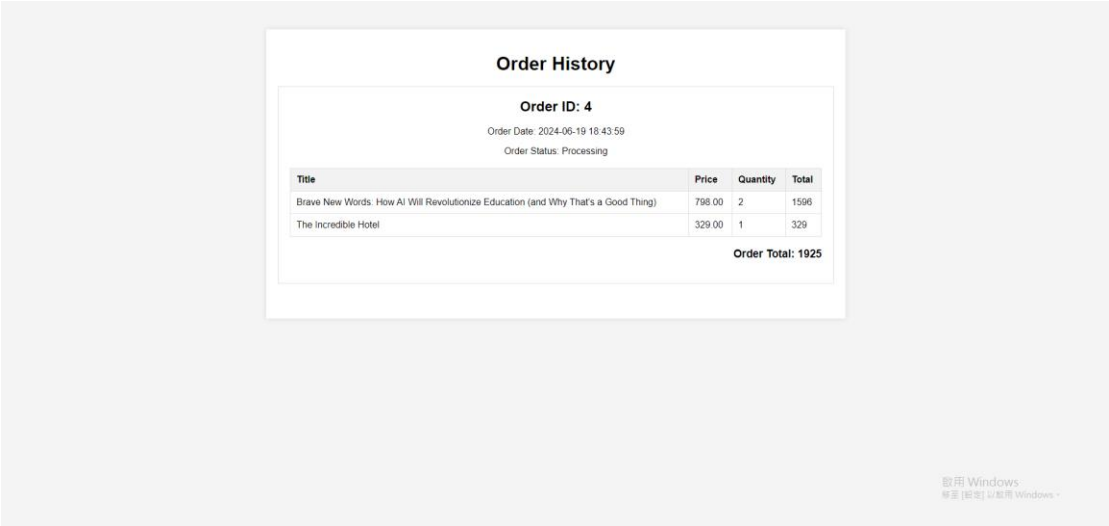
Purpose: This query increases the quantity of a specific book in the user's shopping cart, provided that there is sufficient stock.
c. Decrease Item Quantity in Cart:
UPDATE OrderItem SET Quantity = CASE WHEN Quantity > 1 THEN Quantity - 1 ELSE Quantity END WHERE OrderID = (SELECT OrderID FROM `Order` WHERE UserID = ? AND OrderStatus = 'Pending') AND BookID = ?;

Purpose: This query decreases the quantity of a specific book in the user's shopping cart but ensures that the quantity does not go below one.
d. Confirm Order:
UPDATE `Order` SET OrderStatus = 'Processing', TotalPrice = ? WHERE OrderID = ?;

Purpose: This query updates the order status to 'Processing' and sets the total price of the order after the user confirms the purchase. It also checks stock availability before confirming the order.

4.4 Order status check(order.php)

The order status check page allows users to view their current orders that are in the 'Processing' status. It displays the order ID, order date, order status, and details of the items in each order, including the title, price, quantity, and total price for each item.



Important SQL Queries:

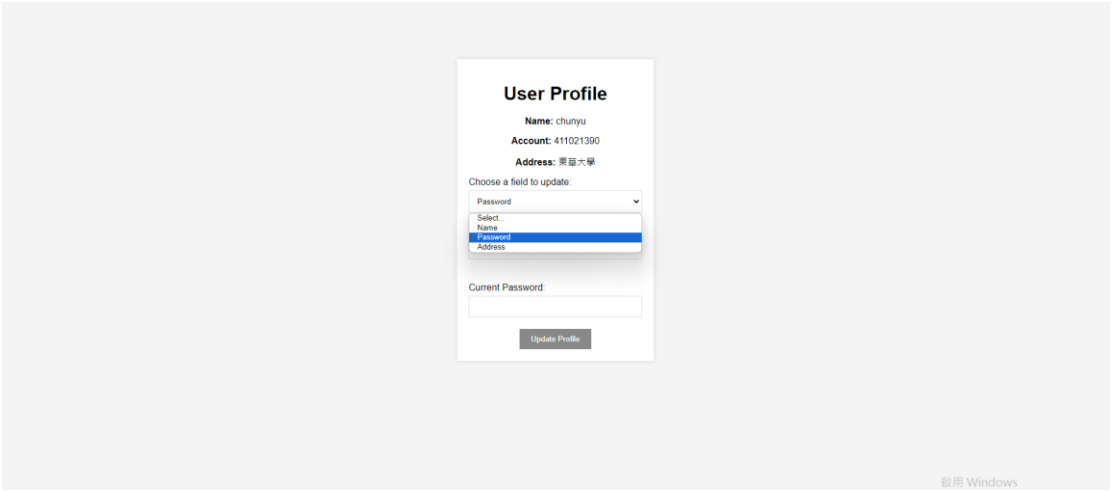
a. Retrieve User's Processing Orders:

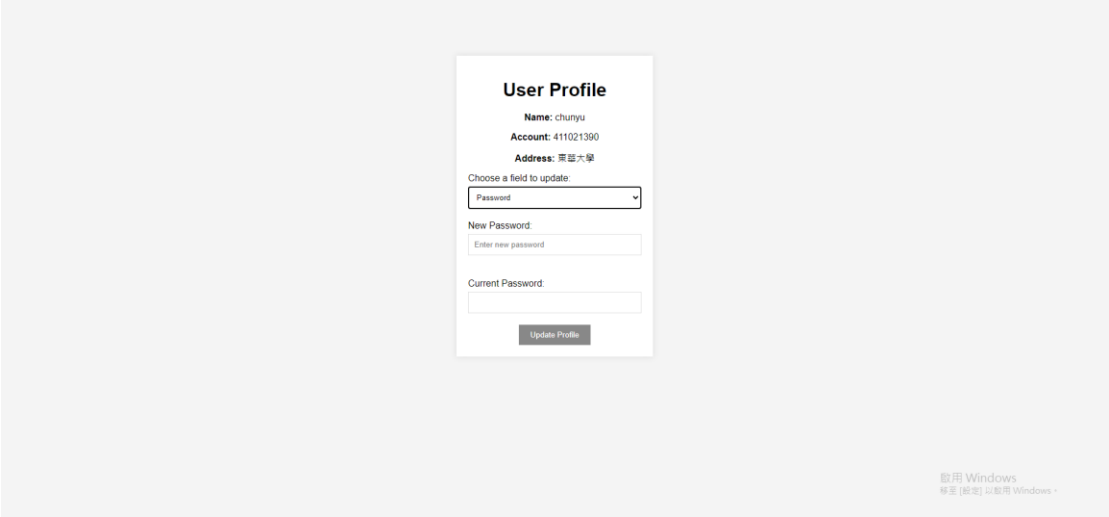
```
SELECT `Order`.OrderID, `Order`.OrderDate, `Order`.OrderStatus, Book.Title,
Book.Price, OrderItem.Quantity
FROM `Order`
JOIN OrderItem ON `Order`.OrderID = OrderItem.OrderID
JOIN Book ON OrderItem.BookID = Book.BookID
WHERE `Order`.UserID = ? AND `Order`.OrderStatus = 'Processing'
ORDER BY `Order`.OrderDate DESC;
```

Purpose: This query retrieves all orders for a specific user where the order status is 'Processing'. It joins the Order, OrderItem, and Book tables to get the order details, including the order ID, order date, order status, book title, book price, and quantity of each book in the order.

4.5User profile(profile.php)

The user profile management page allows users to view and update their profile information, including their name, password, and address. Users need to verify their current password before making any updates. The page includes a dropdown to select the field to update and dynamically displays the corresponding input fields.





Important SQL Queries:

a.Retrieve Current User Profile:

SELECT Name, Account, Address FROM User WHERE UserID = ?;

Purpose: This query retrieves the current user's name, account, and address from the database to display on the profile page.

b. Verify Current Password:

SELECT Password FROM User WHERE UserID = ?;

Purpose: This query retrieves the current hashed password of the user to verify the user's identity before allowing updates to their profile.

c. Update User Name:

UPDATE User SET Name = ? WHERE UserID = ?;

Purpose: This query updates the user's name in the database with the new name provided.

d. Update User Password:

UPDATE User SET Password = ? WHERE UserID = ?;

Purpose: This query updates the user's password in the database with the new hashed password provided.

e. Update User Address:

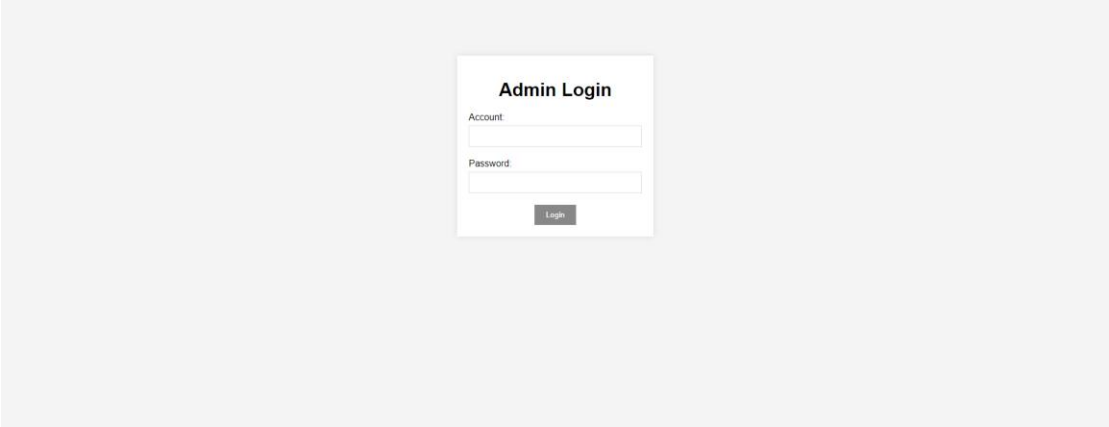
UPDATE User SET Address = ? WHERE UserID = ?;

Purpose: This query updates the user's address in the database with the new address

provided.

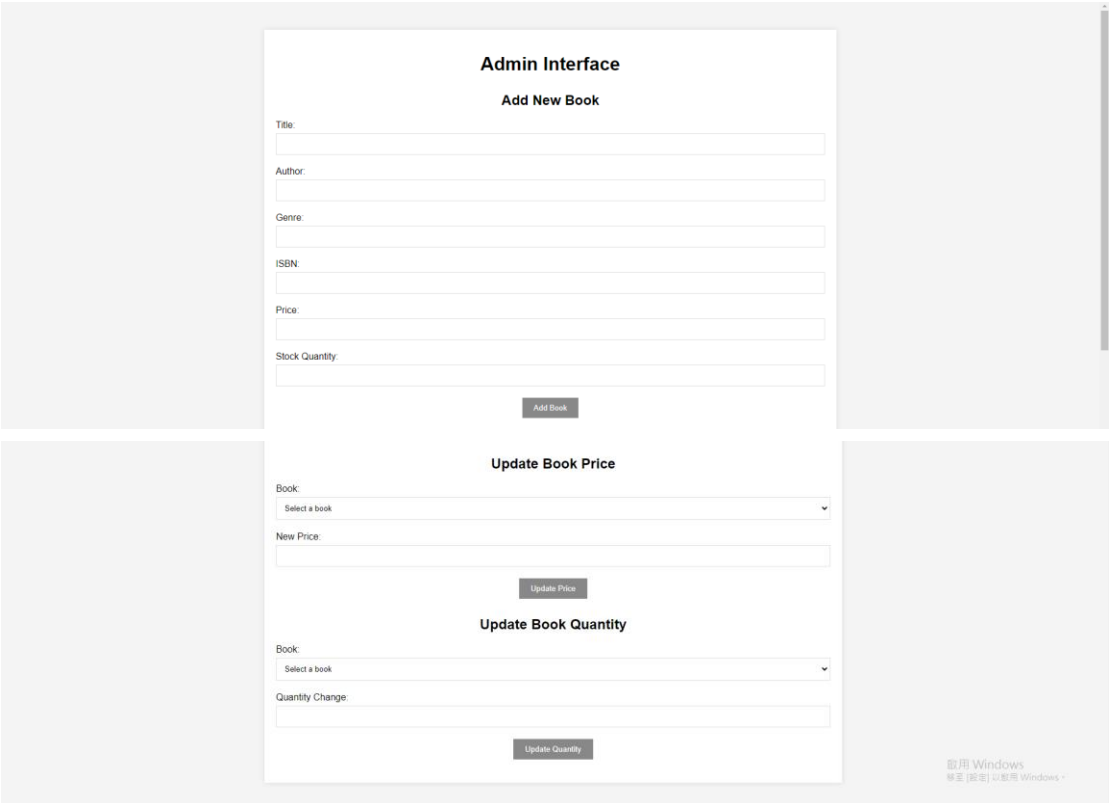
4.5 Admin interface(admin_login.php admin_interface.php and admin_order.php)

admin_login.php



Admin_interface.php

The admin interface page allows administrators to manage the book inventory. It provides functionalities to add new books, update the price of existing books, and update the stock quantity of books. This page is only accessible to users with admin privileges.



Important SQL Queries:

a. Add New Book:

INSERT INTO Book (Title, Author, Genre, ISBN, Price, StockQuantity) VALUES

(?, ?, ?, ?, ?, ?);

Purpose: This query inserts a new book into the Book table with the provided title, author, genre, ISBN, price, and stock quantity.

b. Update Book Price:

UPDATE Book SET Price = ? WHERE BookID = ?;

Purpose: This query updates the price of a specific book in the Book table identified by its BookID.

c. Update Book Quantity:

UPDATE Book SET StockQuantity = StockQuantity + ? WHERE BookID = ?;

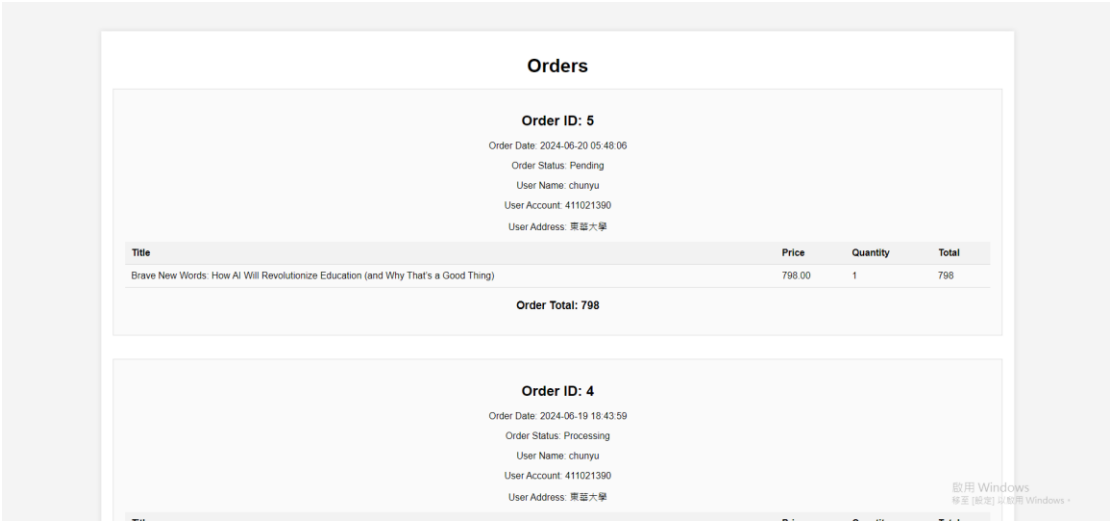
Purpose: This query updates the stock quantity of a specific book in the Book table by adding the specified quantity change to the current stock, identified by its BookID.

d. Retrieve All Books:

SELECT BookID, Title FROM Book;

Purpose: This query retrieves the BookID and Title of all books from the Book table to populate the dropdown selection in the admin interface for updating price and quantity.

Admin_order.php



Important SQL Queries:

a.Retrieve All Orders:

SELECT `Order`.OrderID, `Order`.OrderDate, `Order`.OrderStatus, User.Name, User.Account, User.Address

FROM `Order`

```
JOIN User ON `Order`.UserID = User.UserID
ORDER BY `Order`.OrderDate DESC;
```

Purpose: This query retrieves all orders from the Order table, along with the associated user information from the User table, ordered by the date of the order.

b.Retrieve Items for Each Order:

```
SELECT Book.Title, Book.Price, OrderItem.Quantity
FROM OrderItem
JOIN Book ON OrderItem.BookID = Book.BookID
WHERE OrderItem.OrderID = ?;
```

Purpose: This query retrieves the details of the items in a specific order, including the book title, price, and quantity, for the order identified by the given OrderID.

c. Update Order Status:

```
UPDATE `Order` SET OrderStatus = ? WHERE OrderID = ?;
```

Purpose: This query updates the status of a specific order in the Order table to mark it as completed, identified by the given OrderID.

5.project program

https://github.com/ChiaChunYu/bookstore-system

6.demo video

https://www.youtube.com/watch?v=1c0rEpuh6Y4

7.Data source

The book data used in this project was sourced from Books.com.tw (博客來外文書區):https://www.books.com.tw/?gad_source=1&gclid=CjwKCAjwg8qzBhAoEiwAWagLrKlyeyX0OZdQ0D1t7bK0Se2jjlGxHQmFijK4Xi3Pb0YJxBAEK7IGghoCQ7kQAvD_BwE

8.Other

If the video or project program has any problem please contact me.