**MỤC LỤC**

[**I . Đề bài: 2**](#_Toc164459192)

[**II. Mô tả thuật toán của chương trình quản lý thư viện: 3**](#_Toc164459193)

[**1. Khai báo cấu trúc dữ liệu: 3**](#_Toc164459194)

[**2. Khai báo mảng lưu trữ: 3**](#_Toc164459195)

[**3. Hàm kiểm tra chuỗi hợp lệ: 3**](#_Toc164459196)

[**4. Hàm kiểm tra ngày hợp lệ: 3**](#_Toc164459197)

[**5. Hàm kiểm tra ID hợp lệ: 3**](#_Toc164459198)

[**6. Các hàm thêm, xóa, sửa thông tin sách và người dùng: 3**](#_Toc164459199)

[**7. Các hàm hiển thị thông tin sách và người dùng: 5**](#_Toc164459200)

[**8. Hàm đăng nhập: 5**](#_Toc164459201)

[**9. Hàm main: 5**](#_Toc164459202)

[**10. Thoát khỏi chương trình: 5**](#_Toc164459203)

[**III . Phần Source Code và Giải thích Code 5**](#_Toc164459204)

[**1 . Source Code 5**](#_Toc164459205)

[**2 . Giải thích Code 31**](#_Toc164459206)

[**IV. Ưu và Nhược điểm của chương trình quản lý thư viện: 57**](#_Toc164459207)

[**1. Ưu điểm: 57**](#_Toc164459208)

[**1.1 Tính linh hoạt: 57**](#_Toc164459209)

[**2.1 Giao diện dễ sử dụng: 57**](#_Toc164459210)

[**3.1 Kiểm tra tính hợp lệ của dữ liệu: 57**](#_Toc164459211)

[**4.1 Tính bảo mật: 57**](#_Toc164459212)

[**2. Nhược điểm: 57**](#_Toc164459213)

[**1.2 Khả năng mở rộng hạn chế: 57**](#_Toc164459214)

[**2.2 Thiếu tính nhất quán trong xử lý lỗi: 58**](#_Toc164459215)

[**3.2 Thiếu tính bảo mật đầy đủ: 58**](#_Toc164459216)

[**4.2 Khả năng nhập liệu không linh hoạt: 58**](#_Toc164459217)

**BÀI TẬP LỚN – KĨ THUẬT LẬP TRÌNH**

# **I . Đề bài:**

Bài 2: Viết chương trình quản lý thư viện thực hiện các nhiệm vụ sau:

* Quản lý sách trong thư viện: cho phép thêm, xoá, sửa đổi thông tin về các sách trong cơ sở dữ liệu của chương trình
* Quản lý người dùng: cho phép thêm, xoá, sửa đổi thông tin về các user trong cơ sở dữ liệu của chương trình
* Phân cấp hai mức người dùng của chương trình: người dùng bình thường (user) và người dùng cao cấp (admin). Người dùng bình thường không cần đăng nhập và chỉ có thể xem thông tin về sách trong thư viện. Người dùng cao cấp có thể sửa đổi thông tin thư viện (thêm, sửa và xóa).
* Một quyến sách có các thông tin sau: mã số (isbn), tên sách (title), chủ đề (subject), tác giả (author), nhà xuất bản (publisher), ngày tháng xuất bản (date), số trang (pages), số bản copy trong thư viện (copies). Việc tìm kiếm được thực hiện trên các thông tin chính là: tên sách, chủ đề, tác giả, nhà xuất bản và năm xuất bản.
* Người dùng có các thông tin chính là: mã người dùng (userid), tên truy cập  
  (username), mật khẩu (password) và cấp người dùng

# **II. Mô tả thuật toán của chương trình quản lý thư viện:**

## **Khai báo cấu trúc dữ liệu:**

* Chương trình khai báo hai cấu trúc dữ liệu `Book` và `User` để lưu thông tin về sách và người dùng. Mỗi cấu trúc có các trường dữ liệu tương ứng.

## **Khai báo mảng lưu trữ:**

* Sử dụng mảng `library` để lưu danh sách các sách và mảng `users` để lưu thông tin về người dùng.

## **Hàm kiểm tra chuỗi hợp lệ:**

* `isValidString` kiểm tra xem một chuỗi có chứa ký tự không phù hợp không.

## **Hàm kiểm tra ngày hợp lệ:**

* `isValidDate` kiểm tra xem một chuỗi định dạng ngày tháng có hợp lệ không.

## **Hàm kiểm tra ID hợp lệ:**

* `isValidId` kiểm tra xem một chuỗi ID có chứa ký tự không phù hợp không.

## **Các hàm thêm, xóa, sửa thông tin sách và người dùng:**

* Hàm `addBook()`:
* Người dùng nhập thông tin của một cuốn sách mới, bao gồm ISBN, title, subject, author, publisher, date, pages, và copies.
* Hàm kiểm tra tính hợp lệ của mỗi trường thông tin đầu vào.
* Nếu thông tin hợp lệ, sách mới được thêm vào mảng `library`.
* Hàm `deleteBook()`:
* Người dùng nhập tên của một cuốn sách muốn xóa.
* Chương trình tìm kiếm sách có tên tương ứng trong mảng `library`.
* Nếu tìm thấy, sách đó được xóa khỏi mảng `library`.
* Hàm `editBook()`:
* Người dùng nhập tên của một cuốn sách muốn chỉnh sửa thông tin.
* Chương trình tìm kiếm sách có tên tương ứng trong mảng `library`.
* Người dùng chọn trường thông tin cụ thể muốn sửa đổi.
* Sau đó, người dùng cung cấp thông tin mới cho trường đó.
* Thông tin sách được cập nhật trong mảng `library`.
* Hàm `addUser()`:
* Người dùng nhập thông tin của một người dùng mới, bao gồm user ID, username, password và usertype.
* Hàm kiểm tra tính hợp lệ của mỗi trường thông tin đầu vào.
* Nếu thông tin hợp lệ, người dùng mới được thêm vào mảng `users`.
* Hàm `deleteUser()`:
* Người dùng nhập ID của một người dùng muốn xóa.
* Chương trình tìm kiếm người dùng có ID tương ứng trong mảng `users`.
* Nếu tìm thấy, người dùng đó được xóa khỏi mảng `users`.
* Hàm `editUser()`:
* Người dùng nhập ID của một người dùng muốn chỉnh sửa thông tin.
* Chương trình tìm kiếm người dùng có ID tương ứng trong mảng `users`.
* Người dùng chọn trường thông tin cụ thể muốn sửa đổi.
* Sau đó, người dùng cung cấp thông tin mới cho trường đó.
* Thông tin người dùng được cập nhật trong mảng `users`.

## **Các hàm hiển thị thông tin sách và người dùng:**

* `displayBooks`: Hiển thị thông tin của tất cả các cuốn sách trong mảng `library`.
* `displayUsers`: Hiển thị thông tin của tất cả người dùng trong mảng `users`.

## **Hàm đăng nhập:**

* Người dùng được yêu cầu nhập vai trò là user hoặc admin.
* Nếu là user, chương trình hiển thị menu cho người dùng.
* Nếu là admin, yêu cầu nhập tên đăng nhập và mật khẩu.
* Sau khi xác thực, hiển thị menu admin hoặc thông báo lỗi nếu thông tin không hợp lệ.

## **Hàm main:**

* Thực hiện vòng lặp chính của chương trình, cho phép người dùng chọn các chức năng để thao tác với thư viện hoặc hệ thống người dùng.

## **Thoát khỏi chương trình:**

* Khi người dùng chọn thoát, chương trình kết thúc và hiển thị thông báo cảm ơn.

# **III . Phần Source Code và Giải thích Code**

## **1 . Source Code**

#include <iostream>

#include <string>

#include <algorithm>

#include <limits>

using namespace std;

const int MAX\_BOOKS = 100;

const int MAX\_USERS = 100;

struct Book

{

string isbn;

string title;

string subject;

string author;

string publisher;

string date;

int pages;

int copies;

};

struct User

{

string userid;

string username;

string password;

string usertype;

};

Book library[MAX\_BOOKS];

User users[MAX\_USERS];

int numBooks = 0;

int numUsers = 0;

User currentUser;

bool hasUserInformation = false;

bool isValidString(const string& str)

{

for (size\_t i = 0; i < str.length(); ++i)

{

char ch = str[i];

if (!isalnum(ch) && ch != ' ')

{

return false;

}

}

return true;

}

bool isValidDate(const string& str)

{

if (str.length() != 7) return false;

if (str[2] != '/') return false;

if (!isdigit(str[0]) || !isdigit(str[1])) return false;

int month = (str[0] - '0') \* 10 + (str[1] - '0');

if (month < 1 || month > 12) return false;

for (int i = 3; i < 7; ++i)

{

if (!isdigit(str[i])) return false;

}

return true;

}

bool isValidId(const string& id)

{

for (size\_t i = 0; i < id.length(); ++i)

{

if (!isdigit(id[i]))

{

return false;

}

}

return true;

}

bool isExistingUser(const string& userId)

{

for (int i = 0; i < numUsers; ++i)

{

if (users[i].userid == userId)

{

return true;

}

}

return false;

}

void addBook()

{

if (numBooks >= MAX\_BOOKS)

{

cout << "Thu vien da dat toi gioi han so sach." << endl;

return;

}

Book newBook;

cout << "Nhap thong tin sach:" << endl;

cout << "ISBN: ";

cin >> newBook.isbn;

while (!isValidString(newBook.isbn))

{

cout << "ISBN phai la chuoi. Nhap lai: ";

cin >> newBook.isbn;

}

cout << "Title: ";

cin.ignore();

getline(cin, newBook.title);

while (!isValidString(newBook.title))

{

cout << "Title phai la chuoi. Nhap lai: ";

getline(cin, newBook.title);

}

cout << "Subject: ";

getline(cin, newBook.subject);

while (!isValidString(newBook.subject))

{

cout << "Subject phai la chuoi. Nhap lai: ";

getline(cin, newBook.subject);

}

cout << "Author: ";

getline(cin, newBook.author);

while (!isValidString(newBook.author))

{

cout << "Author phai la chuoi. Nhap lai: ";

getline(cin, newBook.author);

}

cout << "Publisher: ";

getline(cin, newBook.publisher);

while (!isValidString(newBook.publisher))

{

cout << "Publisher phai la chuoi. Nhap lai: ";

getline(cin, newBook.publisher);

}

cout << "Date (mm/yyyy): ";

cin >> newBook.date;

while (!isValidDate(newBook.date))

{

cout << "Date phai dung dinh dang (mm/yyyy). Nhap lai: ";

cin >> newBook.date;

}

cout << "Pages: ";

cin >> newBook.pages;

while (!cin)

{

cout << "Pages phai la mot so. Nhap lai: ";

cin.clear();

cin.ignore(numeric\_limits<streamsize>::max(), '\n');

cin >> newBook.pages;

}

cout << "Copies: ";

cin >> newBook.copies;

while (!cin)

{

cout << "Copies phai la mot so. Nhap lai: ";

cin.clear();

cin.ignore(numeric\_limits<streamsize>::max(), '\n');

cin >> newBook.copies;

}

library[numBooks++] = newBook;

cout << "Sach da duoc them vao thu vien." << endl;

}

void deleteBook()

{

if (numBooks == 0)

{

cout << "Thu vien hien chua co sach nao." << endl;

return;

}

string title;

cout << "Nhap ten sach muon xoa: ";

cin.ignore();

getline(cin, title);

bool found = false;

for (int i = 0; i < numBooks; ++i)

{

if (library[i].title == title)

{

found = true;

for (int j = i; j < numBooks - 1; ++j)

{

library[j] = library[j + 1];

}

numBooks--;

cout << "Sach da duoc xoa khoi thu vien." << endl;

break;

}

}

if (!found)

{

cout << "Khong tim thay sach co ten nay trong thu vien." << endl;

}

}

void editBook()

{

if (numBooks == 0)

{

cout << "Thu vien hien chua co sach nao." << endl;

return;

}

string title;

bool found = false;

while (!found)

{

cout << "Nhap ten sach muon sua thong tin: ";

cin.ignore();

getline(cin, title);

for (int i = 0; i < numBooks; ++i)

{

if (library[i].title == title)

{

found = true;

cout << "Nhap thong tin moi cho sach '" << title << "':" << endl;

string field;

bool validField = false;

while (!validField)

{

cout << "Chon phan muon sua (isbn, title, subject, author, publisher, date, pages, copies): ";

cin >> field;

if (field == "isbn" || field == "title" || field == "subject" || field == "author" || field == "publisher" || field == "date" || field == "pages" || field == "copies")

{

validField = true;

}

else

{

cout << "Lua chon khong hop le. Vui long nhap lai." << endl;

}

}

if (field == "isbn")

{

cout << "ISBN: ";

cin >> library[i].isbn;

while (!isValidString(library[i].isbn))

{

cout << "ISBN phai la chuoi. Nhap lai: ";

cin >> library[i].isbn;

}

}

else if (field == "title")

{

cout << "Title: ";

cin.ignore();

getline(cin, library[i].title);

while (!isValidString(library[i].title))

{

cout << "Title phai la chuoi. Nhap lai: ";

getline(cin, library[i].title);

}

}

else if (field == "subject")

{

cout << "Subject: ";

cin.ignore();

getline(cin, library[i].subject);

while (!isValidString(library[i].subject))

{

cout << "Subject phai la chuoi. Nhap lai: ";

getline(cin, library[i].subject);

}

}

else if (field == "author")

{

cout << "Author: ";

cin.ignore();

getline(cin, library[i].author);

while (!isValidString(library[i].author))

{

cout << "Author phai la chuoi. Nhap lai: ";

getline(cin, library[i].author);

}

}

else if (field == "publisher")

{

cout << "Publisher: ";

cin.ignore();

getline(cin, library[i].publisher);

while (!isValidString(library[i].publisher))

{

cout << "Publisher phai la chuoi. Nhap lai: ";

getline(cin, library[i].publisher);

}

}

else if (field == "date")

{

cout << "Date (mm/yyyy): ";

cin >> library[i].date;

while (!isValidDate(library[i].date))

{

cout << "Date phai dung dinh dang (mm/yyyy). Nhap lai:";

cin >> library[i].date;

}

}

else if (field == "pages")

{

cout << "Pages: ";

cin >> library[i].pages;

while (!cin)

{

cout << "Pages phai la mot so. Nhap lai: ";

cin.clear();

cin.ignore(numeric\_limits<streamsize>::max(), '\n');

cin >> library[i].pages;

}

}

else if (field == "copies")

{

cout << "Copies: ";

cin >> library[i].copies;

while (!cin)

{

cout << "Copies phai la mot so. Nhap lai: ";

cin.clear();

cin.ignore(numeric\_limits<streamsize>::max(), '\n');

cin >> library[i].copies;

}

}

else

{

cout << "Phan muon sua khong hop le." << endl;

return;

}

cout << "Thong tin sach da duoc cap nhat." << endl;

break;

}

}

if (!found)

{

cout << "Khong tim thay sach co ten nay trong thu vien. Vui long nhap lai." << endl;

}

}

}

void displayBooks()

{

cout << "Danh sach trong thu vien:" << endl;

if (numBooks == 0)

{

cout << "Thu vien hien chua co sach nao." << endl;

}

else

{

for (int i = 0; i < numBooks; ++i)

{

Book& book = library[i];

cout << endl;

cout << "-----------------------------------------" << endl;

cout << "QUYEN SACH " << i+1 << " - " << library[i].title << endl;

cout << "THONG TIN CUA SACH " << endl;

cout << "Title: " << library[i].title << "\nSubject: " << library[i].subject

<< "\nAuthor: " << library[i].author << "\nPublisher: " << library[i].publisher << "\nDate: " << library[i].date << endl;

}

}

}

void addUser()

{

if (numUsers >= MAX\_USERS)

{

cout << "He thong da dat toi gioi han so nguoi dung." << endl;

return;

}

User newUser;

cout << "Nhap thong tin nguoi dung:" << endl;

cout << "User ID: ";

cin >> newUser.userid;

while (!isValidId(newUser.userid) || isExistingUser(newUser.userid))

{

if (!isValidId(newUser.userid))

{

cout << "User ID phai la chuoi so. Nhap lai: ";

cin >> newUser.userid;

}

else

{

cout << "User ID da ton tai. Nhap lai: ";

cin >> newUser.userid;

}

}

cout << "Username: ";

cin.ignore();

getline(cin, newUser.username);

while (!isValidString(newUser.username))

{

cout << "Username phai la chuoi. Nhap lai: ";

getline(cin, newUser.username);

}

cout << "Password: ";

getline(cin, newUser.password);

while (!isValidString(newUser.password))

{

cout << "Password phai la chuoi. Nhap lai: ";

getline(cin, newUser.password);

}

cout << "User type (user/admin): ";

cin >> newUser.usertype;

while (newUser.usertype != "user" && newUser.usertype != "admin")

{

cout << "User type phai la 'user' hoac 'admin'. Nhap lai: ";

cin >> newUser.usertype;

}

users[numUsers++] = newUser;

cout << "Nguoi dung da duoc them vao co so du lieu." << endl;

}

void deleteUser()

{

if (numUsers == 0)

{

cout << "Co so du lieu hien khong co nguoi dung nao." << endl;

return;

}

string userId;

cout << "Nhap ma nguoi dung muon xoa: ";

cin >> userId;

bool found = false;

for (int i = 0; i < numUsers; ++i)

{

if (users[i].userid == userId)

{

found = true;

for (int j = i; j < numUsers - 1; ++j)

{

users[j] = users[j + 1];

}

numUsers--;

cout << "Nguoi dung da duoc xoa khoi co so du lieu." << endl;

break;

}

}

if (!found)

{

cout << "Khong tim thay nguoi dung co ma nay trong co so du lieu." << endl;

}

}

void editUser()

{

if (numUsers == 0)

{

cout << "Co so du lieu hien khong co nguoi dung nao." << endl;

return;

}

string userId;

cout << "Nhap ma nguoi dung muon sua thong tin: ";

cin >> userId;

bool found = false;

for (int i = 0; i < numUsers; ++i)

{

if (users[i].userid == userId)

{

found = true;

cout << "Nhap thong tin moi cho nguoi dung '" << userId << "':" << endl;

string field;

bool validField = false;

while (!validField)

{

cout << "Chon phan muon sua (username, password, usertype): ";

cin >> field;

if (field == "username" || field == "password" || field == "usertype")

{

validField = true;

}

else

{

cout << "Lua chon khong hop le. Vui long nhap lai." << endl;

}

}

if (field == "username")

{

cout << "Username: ";

cin.ignore();

getline(cin, users[i].username);

while (!isValidString(users[i].username))

{

cout << "Username phai la chuoi. Nhap lai: ";

getline(cin, users[i].username);

}

}

else if (field == "password")

{

cout << "Password: ";

cin.ignore();

getline(cin, users[i].password);

while (!isValidString(users[i].password))

{

cout << "Password phai la chuoi. Nhap lai: ";

getline(cin, users[i].password);

}

}

else if (field == "usertype")

{

cout << "User type (user/admin): ";

cin >> users[i].usertype;

while (users[i].usertype != "user" && users[i].usertype != "admin")

{

cout << "User type phai la 'user' hoac 'admin'. Nhap lai: ";

cin >> users[i].usertype;

}

}

else

{

cout << "Phan muon sua khong hop le." << endl;

return;

}

cout << "Thong tin nguoi dung da duoc cap nhat." << endl;

break;

}

}

if (!found)

{

cout << "Khong tim thay nguoi dung co ma nay trong he thong. Vui long nhap lai." << endl;

}

}

void displayUsers()

{

cout << "Danh sach nguoi dung trong co so du lieu:" << endl;

if (numUsers == 0)

{

cout << "Co so du lieu hien khong co nguoi dung nao." << endl;

}

else

{

for (int i = 0; i < numUsers; ++i)

{

cout << "-----------------------------------------" << endl;

cout << "NGUOI DUNG " << i + 1 << endl;

cout << "THONG TIN CUA NGUOI DUNG " << endl;

cout << "UserID: " << users[i].userid << endl;

cout << "Username: " << users[i].username << endl;

cout << "Password: " << users[i].password << endl;

cout << "User type: " << users[i].usertype << endl;

}

}

}

void login()

{

string userType;

cout << "Ban la user hay admin? (user/admin): ";

cin >> userType;

if (userType == "user")

{

cout << "Chao mung user!" << endl;

while (true)

{

cout << endl;

cout << "Menu user:" << endl;

cout << "1. Hien thi tat ca sach" << endl;

cout << "2. Thoat" << endl;

cout << "Chon: ";

int choice;

cin >> choice;

switch (choice)

{

case 1:

displayBooks();

break;

case 2:

cout << "Thoat ra ngoai thanh cong!" << endl;

return;

default:

cout << "Lua chon khong hop le. Vui long chon lai." << endl;

}

}

}

else if (userType == "admin")

{

string username, password;

bool isValidLogin = false;

cout << "Nhap ten dang nhap: ";

cin >> username;

cout << "Nhap mat khau: ";

cin >> password;

for (int i = 0; i < numUsers; ++i)

{

if (users[i].username == username && users[i].password == password && users[i].usertype == "admin")

{

isValidLogin = true;

currentUser = users[i];

cout << "Dang nhap thanh cong! Chao mung admin." << endl;

while (true)

{

cout << endl;

cout << "Menu admin:" << endl;

cout << "1. Them sach" << endl;

cout << "2. Xoa sach" << endl;

cout << "3. Sua thong tin sach" << endl;

cout << "4. Hien thi tat ca sach" << endl;

cout << "5. Dang xuat" << endl;

cout << "Chon: ";

int choice;

cin >> choice;

switch (choice)

{

case 1:

addBook();

break;

case 2:

deleteBook();

break;

case 3:

editBook();

break;

case 4:

displayBooks();

break;

case 5:

cout << "Dang xuat...";

return;

default:

cout << "Lua chon khong hop le. Vui long chon lai." << endl;

}

}

}

}

if (!isValidLogin)

{

cout << "Thong tin dang nhap khong hop le. Vui long nhap lai." << endl;

}

}

else

{

cout << "Loai nguoi dung khong hop le." << endl;

}

}

int main()

{

while (true)

{

cout << "\n===== CHUONG TRINH QUAN LY THU VIEN =====" << endl;

cout << "1. Them sach" << endl;

cout << "2. Xoa sach" << endl;

cout << "3. Sua thong tin sach" << endl;

cout << "4. Hien thi tat ca sach" << endl;

cout << "5. Them nguoi dung" << endl;

cout << "6. Xoa nguoi dung" << endl;

cout << "7. Sua thong tin nguoi dung" << endl;

cout << "8. Hien thi tat ca nguoi dung" << endl;

cout << "9. Dang nhap nguoi dung" << endl;

cout << "0. Thoat chuong trinh " << endl;

cout << "Nhap lua chon cua ban: ";

int choice;

cin >> choice;

switch (choice)

{

case 1:

addBook();

break;

case 2:

deleteBook();

break;

case 3:

editBook();

break;

case 4:

displayBooks();

break;

case 5:

addUser();

break;

case 6:

deleteUser();

break;

case 7:

editUser();

break;

case 8:

displayUsers();

break;

case 9:

login();

break;

case 0:

cout << "Cam on ban da su dung chuong trinh." << endl;

return 0;

default:

cout << "Lua chon khong hop le. Vui long nhap lai." << endl;

}

}

return 0;

}

## **2 . Giải thích Code**

#include <iostream> // Thư viện dùng để nhập xuất dữ liệu.

#include <string> // Thư viện dùng để làm việc với chuỗi.

#include <algorithm> // Thư viện chứa các hàm xử lý thuật toán như tìm kiếm, sắp xếp.

#include <limits> // Thư viện chứa giá trị tối đa của các kiểu dữ liệu.

using namespace std; // Sử dụng không gian tên std để tránh việc phải ghi std:: trước mỗi lệnh.

// Khai báo cấu trúc Book (Sách) chứa thông tin về một quyển sách.

struct Book {

string isbn; // Mã số ISBN của sách.

string title; // Tiêu đề của sách.

string subject; // Chủ đề của sách.

string author; // Tác giả của sách.

string publisher; // Nhà xuất bản của sách.

string date; // Ngày xuất bản của sách.

int pages; // Số trang của sách.

int copies; // Số bản sao của sách.

};

// Khai báo cấu trúc User (Người dùng) chứa thông tin về một người dùng.

struct User {

string userid; // Mã số người dùng.

string username; // Tên đăng nhập của người dùng.

string password; // Mật khẩu của người dùng.

string usertype; // Loại người dùng (user hoặc admin).

};

Book library[MAX\_BOOKS]; // Mảng chứa thông tin về các quyển sách trong thư viện.

User users[MAX\_USERS]; // Mảng chứa thông tin về các người dùng.

int numBooks = 0; // Số lượng sách trong thư viện.

int numUsers = 0; // Số lượng người dùng.

User currentUser; // Người dùng hiện tại đang đăng nhập.

bool hasUserInformation = false; // Biến kiểm tra xem đã có thông tin người dùng hay chưa.

// Hàm kiểm tra xem một chuỗi có hợp lệ không (chỉ chứa ký tự chữ và số).

bool isValidString(const string& str) {

for (size\_t i = 0; i < str.length(); ++i) { // Duyệt qua từng ký tự trong chuỗi.

char ch = str[i]; // Lấy ký tự tại vị trí i.

if (!isalnum(ch) && ch != ' ') { // Nếu ký tự không phải là chữ cái, số hoặc dấu cách.

return false; // Trả về false vì chuỗi không hợp lệ.

}

}

return true; // Trả về true nếu chuỗi hợp lệ.

}

// Hàm kiểm tra xem một chuỗi đại diện cho ngày tháng có hợp lệ không (định dạng mm/yyyy).

bool isValidDate(const string& str) {

if (str.length() != 7) return false; // Nếu chuỗi không có đúng 7 ký tự (mm/yyyy) thì không hợp lệ.

if (str[2] != '/') return false; // Nếu ký tự thứ 3 không phải là dấu '/' thì không hợp lệ.

if (!isdigit(str[0]) || !isdigit(str[1])) return false; // Nếu hai ký tự đầu không phải là số thì không hợp lệ.

int month = (str[0] - '0') \* 10 + (str[1] - '0'); // Lấy giá trị tháng từ chuỗi.

if (month < 1 || month > 12) return false; // Nếu giá trị tháng không nằm trong khoảng từ 1 đến 12 thì không hợp lệ.

for (int i = 3; i < 7; ++i) { // Duyệt qua các ký tự từ vị trí thứ 4 đến cuối chuỗi (năm).

if (!isdigit(str[i])) return false; // Nếu có ký tự không phải là số thì không hợp lệ.

}

return true; // Trả về true nếu chuỗi hợp lệ.

}

// Hàm kiểm tra xem một chuỗi có hợp lệ là một ID không (chỉ chứa các ký tự số).

bool isValidId(const string& id) {

for (size\_t i = 0; i < id.length(); ++i) { // Duyệt qua từng ký tự trong chuỗi.

if (!isdigit(id[i])) { // Nếu ký tự không phải là số.

return false; // Trả về false vì chuỗi không hợp lệ.

}

}

return true; // Trả về true nếu chuỗi hợp lệ.

}

// Kiểm tra xem userId đã tồn tại trong mảng người dùng hay chưa

bool isExistingUser(const string& userId)

{

// Duyệt qua mảng người dùng

for (int i = 0; i < numUsers; ++i)

{

// Kiểm tra xem userId truyền vào có trùng với userId trong mảng không

if (users[i].userid == userId)

{

// Nếu tìm thấy userId trùng, trả về true

return true;

}

}

// Nếu không tìm thấy userId trùng, trả về false

return false;

}

void addBook()

{

// Kiểm tra xem số lượng sách trong thư viện đã đạt đến giới hạn chưa.

if (numBooks >= MAX\_BOOKS)

{

// In ra thông báo khi thư viện đã đạt đến giới hạn số sách.

cout << "Thu vien da dat toi gioi han so sach." << endl;

// Kết thúc hàm khi không thể thêm sách mới.

return;

}

// Khai báo một biến mới kiểu Book để lưu thông tin của sách mới.

Book newBook;

// Yêu cầu người dùng nhập thông tin cho sách mới.

cout << "Nhap thong tin sach:" << endl;

// Yêu cầu người dùng nhập mã số ISBN của sách.

cout << "ISBN: ";

cin >> newBook.isbn;

// Kiểm tra tính hợp lệ của chuỗi ISBN vừa nhập.

while (!isValidString(newBook.isbn))

{

// Yêu cầu nhập lại nếu chuỗi ISBN không hợp lệ.

cout << "ISBN phai la chuoi. Nhap lai: ";

cin >> newBook.isbn;

}

// Yêu cầu người dùng nhập tiêu đề của sách.

cout << "Title: ";

// Bỏ qua các ký tự thừa trong bộ đệm dữ liệu nhập trước đó.

cin.ignore();

// Nhận dữ liệu đầu vào dưới dạng chuỗi từ bàn phím và lưu vào trường title của sách mới.

getline(cin, newBook.title);

// Kiểm tra tính hợp lệ của tiêu đề sách vừa nhập.

while (!isValidString(newBook.title))

{

// Yêu cầu nhập lại nếu tiêu đề sách không hợp lệ.

cout << "Title phai la chuoi. Nhap lai: ";

// Nhận dữ liệu đầu vào dưới dạng chuỗi từ bàn phím và lưu vào trường title của sách mới.

getline(cin, newBook.title);

}

// Yêu cầu người dùng nhập chủ đề của sách.

cout << "Subject: ";

// Nhận dữ liệu đầu vào dưới dạng chuỗi từ bàn phím và lưu vào trường subject của sách mới.

getline(cin, newBook.subject);

// Kiểm tra tính hợp lệ của chủ đề sách vừa nhập.

while (!isValidString(newBook.subject))

{

// Yêu cầu nhập lại nếu chủ đề sách không hợp lệ.

cout << "Subject phai la chuoi. Nhap lai: ";

// Nhận dữ liệu đầu vào dưới dạng chuỗi từ bàn phím và lưu vào trường subject của sách mới.

getline(cin, newBook.subject);

}

// Yêu cầu người dùng nhập tác giả của sách.

cout << "Author: ";

// Nhận dữ liệu đầu vào dưới dạng chuỗi từ bàn phím và lưu vào trường author của sách mới.

getline(cin, newBook.author);

// Kiểm tra tính hợp lệ của tác giả sách vừa nhập.

while (!isValidString(newBook.author))

{

// Yêu cầu nhập lại nếu tác giả sách không hợp lệ.

cout << "Author phai la chuoi. Nhap lai: ";

// Nhận dữ liệu đầu vào dưới dạng chuỗi từ bàn phím và lưu vào trường author của sách mới.

getline(cin, newBook.author);

}

// Yêu cầu người dùng nhập nhà xuất bản của sách.

cout << "Publisher: ";

// Nhận dữ liệu đầu vào dưới dạng chuỗi từ bàn phím và lưu vào trường publisher của sách mới.

getline(cin, newBook.publisher);

// Kiểm tra tính hợp lệ của nhà xuất bản sách vừa nhập.

while (!isValidString(newBook.publisher))

{

// Yêu cầu nhập lại nếu nhà xuất bản sách không hợp lệ.

cout << "Publisher phai la chuoi. Nhap lai: ";

// Nhận dữ liệu đầu vào dưới dạng chuỗi từ bàn phím và lưu vào trường publisher của sách mới.

getline(cin, newBook.publisher);

}

// Yêu cầu người dùng nhập ngày xuất bản của sách.

cout << "Date (mm/yyyy): ";

// Nhận dữ liệu đầu vào dưới dạng chuỗi từ bàn phím và lưu vào trường date của sách mới.

cin >> newBook.date;

// Kiểm tra tính hợp lệ của ngày xuất bản sách vừa nhập.

while (!isValidDate(newBook.date))

{

// Yêu cầu nhập lại nếu ngày xuất bản sách không hợp lệ.

cout << "Date phai dung dinh dang (mm/yyyy). Nhap lai: ";

// Nhận dữ liệu đầu vào dưới dạng chuỗi từ bàn phím và lưu vào trường date của sách mới.

cin >> newBook.date;

}

// Yêu cầu người dùng nhập số trang của sách.

cout << "Pages: ";

// Nhận dữ liệu đầu vào dưới dạng số từ bàn phím và lưu vào trường pages của sách mới.

cin >> newBook.pages;

// Kiểm tra tính hợp lệ của số trang sách vừa nhập.

while (!cin)

{

// Yêu cầu nhập lại nếu số trang sách không hợp lệ.

cout << "Pages phai la mot so. Nhap lai: ";

// Xóa trạng thái lỗi của dòng nhập trước đó và bỏ qua ký tự thừa trong bộ đệm dữ liệu nhập.

cin.clear();

cin.ignore(numeric\_limits<streamsize>::max(), '\n');

// Nhận dữ liệu đầu vào dưới dạng số từ bàn phím và lưu vào trường pages của sách mới.

cin >> newBook.pages;

}

// Yêu cầu người dùng nhập số bản sao của sách.

cout << "Copies: ";

// Nhận dữ liệu đầu vào dưới dạng số từ bàn phím và lưu vào trường copies của sách mới.

cin >> newBook.copies;

// Kiểm tra tính hợp lệ của số bản sao sách vừa nhập.

while (!cin)

{

// Yêu cầu nhập lại nếu số bản sao sách không hợp lệ.

cout << "Copies phai la mot so. Nhap lai: ";

// Xóa trạng thái lỗi của dòng nhập trước đó và bỏ qua ký tự thừa trong bộ đệm dữ liệu nhập.

cin.clear();

cin.ignore(numeric\_limits<streamsize>::max(), '\n');

// Nhận dữ liệu đầu vào dưới dạng số từ bàn phím và lưu vào trường copies của sách mới.

cin >> newBook.copies;

}

// Thêm sách mới vào thư viện và tăng số lượng sách trong thư viện lên 1.

library[numBooks++] = newBook;

// In ra thông báo khi sách mới đã được thêm vào thư viện.

cout << "Sach da duoc them vao thu vien." << endl;

}

void deleteBook()

{

// Kiểm tra xem thư viện có sách nào không.

if (numBooks == 0)

{

// In ra thông báo nếu thư viện không có sách.

cout << "Thu vien hien chua co sach nao." << endl;

// Kết thúc hàm khi không có sách nào để xóa.

return;

}

string title;

// Yêu cầu người dùng nhập tên sách muốn xóa.

cout << "Nhap ten sach muon xoa: ";

// Bỏ qua các ký tự thừa trong bộ đệm dữ liệu nhập.

cin.ignore();

// Nhận dữ liệu đầu vào dưới dạng chuỗi từ bàn phím và lưu vào biến title.

getline(cin, title);

bool found = false;

// Duyệt qua từng quyển sách trong thư viện.

for (int i = 0; i < numBooks; ++i)

{

// Nếu tìm thấy sách có tiêu đề trùng khớp với tiêu đề được nhập.

if (library[i].title == title)

{

found = true;

// Dịch chuyển các quyển sách phía sau về trước để xóa sách.

for (int j = i; j < numBooks - 1; ++j)

{

library[j] = library[j + 1];

}

// Giảm số lượng sách trong thư viện đi 1.

numBooks--;

// In ra thông báo khi sách đã được xóa khỏi thư viện.

cout << "Sach da duoc xoa khoi thu vien." << endl;

break;

}

}

// Nếu không tìm thấy sách có tiêu đề trùng khớp.

if (!found)

{

// In ra thông báo không tìm thấy sách.

cout << "Khong tim thay sach co ten nay trong thu vien." << endl;

}

}

void editBook()

{

// Kiểm tra xem thư viện có sách nào không.

if (numBooks == 0)

{

// In ra thông báo nếu thư viện không có sách.

cout << "Thu vien hien chua co sach nao." << endl;

// Kết thúc hàm khi không có sách nào để chỉnh sửa.

return;

}

string title;

bool found = false;

// Vòng lặp này sẽ tiếp tục chạy cho đến khi người dùng nhập tên sách có trong thư viện.

while (!found)

{

// Yêu cầu người dùng nhập tên sách muốn chỉnh sửa thông tin.

cout << "Nhap ten sach muon sua thong tin: ";

// Bỏ qua các ký tự thừa trong bộ đệm dữ liệu nhập.

cin.ignore();

// Nhận dữ liệu đầu vào dưới dạng chuỗi từ bàn phím và lưu vào biến title.

getline(cin, title);

// Duyệt qua từng quyển sách trong thư viện.

for (int i = 0; i < numBooks; ++i)

{

// Nếu tìm thấy sách có tiêu đề trùng khớp với tiêu đề được nhập.

if (library[i].title == title)

{

found = true;

// In ra thông báo và yêu cầu nhập thông tin mới cho sách.

cout << "Nhap thong tin moi cho sach '" << title << "':" << endl;

// Biến lưu trữ loại thông tin mà người dùng muốn chỉnh sửa.

string field;

// Biến kiểm tra xem loại thông tin được nhập có hợp lệ không.

bool validField = false;

// Vòng lặp này sẽ tiếp tục chạy cho đến khi người dùng nhập một loại thông tin hợp lệ.

while (!validField)

{

// Yêu cầu người dùng nhập loại thông tin muốn chỉnh sửa.

cout << "Chon phan muon sua (isbn, title, subject, author, publisher, date, pages, copies): ";

// Nhận dữ liệu đầu vào dưới dạng chuỗi từ bàn phím và lưu vào biến field.

cin >> field;

// Kiểm tra xem loại thông tin nhập vào có hợp lệ không.

if (field == "isbn" || field == "title" || field == "subject" || field == "author" || field == "publisher" || field == "date" || field == "pages" || field == "copies")

{

validField = true;

}

else

{

// In ra thông báo lỗi nếu loại thông tin nhập vào không hợp lệ.

cout << "Lua chon khong hop le. Vui long nhap lai." << endl;

}

}

// Thực hiện chỉnh sửa thông tin tương ứng với loại thông tin được nhập.

if (field == "isbn")

{

cout << "ISBN: ";

cin >> library[i].isbn;

// Kiểm tra tính hợp lệ của mã ISBN được nhập.

while (!isValidString(library[i].isbn))

{

cout << "ISBN phai la chuoi. Nhap lai: ";

cin >> library[i].isbn;

}

}

else if (field == "title")

{

cout << "Title: ";

cin.ignore();

getline(cin, library[i].title);

// Kiểm tra tính hợp lệ của tiêu đề sách được nhập.

while (!isValidString(library[i].title))

{

cout << "Title phai la chuoi. Nhap lai: ";

getline(cin, library[i].title);

}

}

else if (field == "subject")

{

// Yêu cầu người dùng nhập chủ đề mới cho sách.

cout << "Subject: ";

// Bỏ qua các ký tự thừa trong bộ đệm dữ liệu nhập.

cin.ignore();

// Nhận dữ liệu đầu vào dưới dạng chuỗi từ bàn phím và lưu vào biến subject của sách.

getline(cin, library[i].subject);

// Kiểm tra tính hợp lệ của chủ đề sách được nhập.

while (!isValidString(library[i].subject))

{

cout << "Subject phai la chuoi. Nhap lai: ";

// Nhận dữ liệu đầu vào dưới dạng chuỗi từ bàn phím và lưu vào biến subject của sách.

getline(cin, library[i].subject);

}

}

else if (field == "author")

{

// Yêu cầu người dùng nhập tác giả mới cho sách.

cout << "Author: ";

// Bỏ qua các ký tự thừa trong bộ đệm dữ liệu nhập.

cin.ignore();

// Nhận dữ liệu đầu vào dưới dạng chuỗi từ bàn phím và lưu vào biến author của sách.

getline(cin, library[i].author);

// Kiểm tra tính hợp lệ của tác giả sách được nhập.

while (!isValidString(library[i].author))

{

cout << "Author phai la chuoi. Nhap lai: ";

// Nhận dữ liệu đầu vào dưới dạng chuỗi từ bàn phím và lưu vào biến author của sách.

getline(cin, library[i].author);

}

}

else if (field == "publisher")

{

// Yêu cầu người dùng nhập nhà xuất bản mới cho sách.

cout << "Publisher: ";

// Bỏ qua các ký tự thừa trong bộ đệm dữ liệu nhập.

cin.ignore();

// Nhận dữ liệu đầu vào dưới dạng chuỗi từ bàn phím và lưu vào biến publisher của sách.

getline(cin, library[i].publisher);

// Kiểm tra tính hợp lệ của nhà xuất bản sách được nhập.

while (!isValidString(library[i].publisher))

{

cout << "Publisher phai la chuoi. Nhap lai: ";

// Nhận dữ liệu đầu vào dưới dạng chuỗi từ bàn phím và lưu vào biến publisher của sách.

getline(cin, library[i].publisher);

}

}

else if (field == "date")

{

// Yêu cầu người dùng nhập ngày xuất bản mới cho sách.

cout << "Date (mm/yyyy): ";

// Nhận dữ liệu đầu vào dưới dạng chuỗi từ bàn phím và lưu vào biến date của sách.

cin >> library[i].date;

// Kiểm tra tính hợp lệ của ngày xuất bản sách được nhập.

while (!isValidDate(library[i].date))

{

cout << "Date phai dung dinh dang (mm/yyyy). Nhap lai: ";

// Nhận dữ liệu đầu vào dưới dạng chuỗi từ bàn phím và lưu vào biến date của sách.

cin >> library[i].date;

}

}

else if (field == "pages")

{

// Yêu cầu người dùng nhập số trang mới cho sách.

cout << "Pages: ";

// Nhận dữ liệu đầu vào dưới dạng số từ bàn phím và lưu vào biến pages của sách.

cin >> library[i].pages;

// Kiểm tra tính hợp lệ của số trang sách được nhập.

while (!cin)

{

cout << "Pages phai la mot so. Nhap lai: ";

// Xóa trạng thái lỗi của dòng nhập trước đó và bỏ qua các ký tự thừa trong bộ đệm dữ liệu nhập.

cin.clear();

cin.ignore(numeric\_limits<streamsize>::max(), '\n');

// Nhận dữ liệu đầu vào dưới dạng số từ bàn phím và lưu vào biến pages của sách.

cin >> library[i].pages;

}

}

else if (field == "copies")

{

// Yêu cầu người dùng nhập số lượng bản sao mới cho sách.

cout << "Copies: ";

// Nhận dữ liệu đầu vào dưới dạng số từ bàn phím và lưu vào biến copies của sách.

cin >> library[i].copies;

// Kiểm tra tính hợp lệ của số lượng bản sao sách được nhập.

while (!cin)

{

cout << "Copies phai la mot so. Nhap lai: ";

// Xóa trạng thái lỗi của dòng nhập trước đó và bỏ qua các ký tự thừa trong bộ đệm dữ liệu nhập.

cin.clear();

cin.ignore(numeric\_limits<streamsize>::max(), '\n');

// Nhận dữ liệu đầu vào dưới dạng số từ bàn phím và lưu vào biến copies của sách.

cin >> library[i].copies;

}

}

else

{

// In ra thông báo nếu loại thông tin muốn sửa không hợp lệ.

cout << "Phan muon sua khong hop le." << endl;

// Kết thúc hàm.

return;

}

// In ra thông báo sau khi chỉnh sửa thông tin sách thành công.

cout << "Thong tin sach da duoc cap nhat." << endl;

// Kết thúc vòng lặp.

break;

}

}

if (!found)

{

cout << "Khong tim thay sach co ten nay trong thu vien. Vui long nhap lai." << endl;

}

}

}

void displayBooks()

{

// In ra tiêu đề danh sách sách trong thư viện.

cout << "Danh sach sach trong thu vien:" << endl;

// Kiểm tra xem thư viện có sách nào không.

if (numBooks == 0)

{

// In ra thông báo nếu thư viện không có sách.

cout << "Thu vien hien chua co sach nao." << endl;

}

else

{

// Duyệt qua từng quyển sách trong thư viện.

for (int i = 0; i < numBooks; ++i)

{

// Lưu tham chiếu đến quyển sách hiện tại.

Book& book = library[i];

// In ra dấu phân cách để phân biệt giữa các quyển sách.

cout << endl;

cout << "-----------------------------------------" << endl;

// In ra tiêu đề của sách.

cout << "QUYEN SACH " << i+1 << " - " << library[i].title << endl;

// In ra thông tin chi tiết của sách.

cout << "THONG TIN CUA SACH " << endl;

cout << "Title: " << library[i].title << "\nSubject: " << library[i].subject

<< "\nAuthor: " << library[i].author << "\nPublisher: " << library[i].publisher << "\nDate: " << library[i].date << endl;

}

}

}

// Hàm thêm người dùng mới vào hệ thống

void addUser()

{

// Kiểm tra xem số lượng người dùng đã đạt đến giới hạn tối đa hay chưa

if (numUsers >= MAX\_USERS)

{

// Nếu đã đạt, thông báo và kết thúc hàm

cout << "Hẹ thong da dat toi gioi han so nguoi dung." << endl;

return;

}

// Khai báo một biến newUser kiểu User để lưu thông tin về người dùng mới

User newUser;

cout << "Nhap thong tin nguoi dung:" << endl;

// Yêu cầu người dùng nhập userId của người dùng mới

cout << "User ID: ";

cin >> newUser.userid;

// Kiểm tra tính hợp lệ của userId và xem liệu userId đã tồn tại trong hệ thống hay chưa

while (!isValidId(newUser.userid) || isExistingUser(newUser.userid))

{

// Nếu userId không hợp lệ, hiển thị thông báo yêu cầu nhập lại

if (!isValidId(newUser.userid))

{

cout << "User ID phai la chuoi so. Vui long nhap lai ";

cin >> newUser.userid;

}

// Nếu userId đã tồn tại, hiển thị thông báo yêu cầu nhập lại

else

{

cout << "User ID da ton tai. Vui long nhap lai: ";

cin >> newUser.userid;

}

}

// Nhập tên người dùng từ bàn phím và lưu vào newUser.username.

cout << "Username: ";

cin.ignore();

getline(cin, newUser.username);

// Kiểm tra tính hợp lệ của tên người dùng.

while (!isValidString(newUser.username))

{

// Nếu tên người dùng không hợp lệ, yêu cầu nhập lại.

cout << "Username phai la chuoi. Nhap lai: ";

getline(cin, newUser.username);

}

// Nhập mật khẩu từ bàn phím và lưu vào newUser.password.

cout << "Password: ";

getline(cin, newUser.password);

// Kiểm tra tính hợp lệ của mật khẩu.

while (!isValidString(newUser.password))

{

// Nếu mật khẩu không hợp lệ, yêu cầu nhập lại.

cout << "Password phai la chuoi. Nhap lai: ";

getline(cin, newUser.password);

}

// Nhập loại người dùng (user hoặc admin) từ bàn phím và lưu vào newUser.usertype.

cout << "User type (user/admin): ";

cin >> newUser.usertype;

// Kiểm tra tính hợp lệ của loại người dùng.

while (newUser.usertype != "user" && newUser.usertype != "admin")

{

// Nếu loại người dùng không hợp lệ, yêu cầu nhập lại.

cout << "User type phai la 'user' hoac 'admin'. Nhap lai: ";

cin >> newUser.usertype;

}

// Lưu thông tin người dùng vào mảng users và tăng biến đếm numUsers.

users[numUsers++] = newUser;

// In thông báo sau khi người dùng được thêm thành công.

cout << "Nguoi dung da duoc them vao co so du lieu." << endl;

}

void deleteUser()

{

// Kiểm tra xem có người dùng nào trong cơ sở dữ liệu không.

if (numUsers == 0)

{

// Nếu không có, hiển thị thông báo và kết thúc hàm.

cout << "Co so du lieu hien khong co nguoi dung nao." << endl;

return;

}

// Nhập mã số người dùng cần xóa từ người dùng.

string userId;

cout << "Nhap ma nguoi dung muon xoa: ";

cin >> userId;

// Biến kiểm tra xem người dùng cần xóa có tồn tại trong cơ sở dữ liệu không.

bool found = false;

// Duyệt qua mảng người dùng để tìm người dùng cần xóa.

for (int i = 0; i < numUsers; ++i)

{

if (users[i].userid == userId)

{

// Nếu tìm thấy, đánh dấu đã tìm thấy.

found = true;

// Dời các phần tử sau về trước để xóa người dùng khỏi mảng.

for (int j = i; j < numUsers - 1; ++j)

{

users[j] = users[j + 1];

}

// Giảm số lượng người dùng đi 1 và kết thúc hàm.

numUsers--;

cout << "Nguoi dung da duoc xoa khoi co so du lieu." << endl;

break;

}

}

// Nếu không tìm thấy người dùng cần xóa, hiển thị thông báo.

if (!found)

{

cout << "Khong tim thay nguoi dung co ma nay trong co so du lieu." << endl;

}

}

void editUser()

{

// Kiểm tra xem có người dùng nào trong cơ sở dữ liệu không.

if (numUsers == 0)

{

// Nếu không có, hiển thị thông báo và kết thúc hàm.

cout << "Co so du lieu hien khong co nguoi dung nao." << endl;

return;

}

// Nhập mã số người dùng cần chỉnh sửa thông tin từ người dùng.

string userId;

cout << "Nhap ma nguoi dung muon sua thong tin: ";

cin >> userId;

// Biến kiểm tra xem người dùng cần chỉnh sửa có tồn tại trong cơ sở dữ liệu không.

bool found = false;

// Duyệt qua mảng người dùng để tìm người dùng cần chỉnh sửa thông tin.

for (int i = 0; i < numUsers; ++i)

{

if (users[i].userid == userId)

{

// Nếu tìm thấy, đánh dấu đã tìm thấy.

found = true;

// Hiển thị yêu cầu nhập thông tin mới cho người dùng.

cout << "Nhap thong tin moi cho nguoi dung '" << userId << "':" << endl;

// Biến lưu trữ phần muốn chỉnh sửa.

string field;

bool validField = false;

// Lặp cho đến khi người dùng nhập phần muốn chỉnh sửa hợp lệ.

while (!validField)

{

cout << "Chon phan muon sua (username, password, usertype): ";

cin >> field;

// Kiểm tra tính hợp lệ của phần muốn chỉnh sửa.

if (field == "username" || field == "password" || field == "usertype")

{

validField = true;

}

else

{

cout << "Lua chon khong hop le. Vui long nhap lai." << endl;

}

}

// Thực hiện chỉnh sửa dựa trên phần muốn chỉnh sửa.

if (field == "username")

{

// Nhập và kiểm tra tính hợp lệ của tên người dùng mới.

cout << "Username: ";

cin.ignore();

getline(cin, users[i].username);

while (!isValidString(users[i].username))

{

cout << "Username phai la chuoi. Nhap lai: ";

getline(cin, users[i].username);

}

}

else if (field == "password")

{

// Nhập và kiểm tra tính hợp lệ của mật khẩu mới.

cout << "Password: ";

cin.ignore();

getline(cin, users[i].password);

while (!isValidString(users[i].password))

{

cout << "Password phai la chuoi. Nhap lai: ";

getline(cin, users[i].password);

}

}

else if (field == "usertype")

{

// Nhập và kiểm tra tính hợp lệ của loại người dùng mới.

cout << "User type (user/admin): ";

cin >> users[i].usertype;

while (users[i].usertype != "user" && users[i].usertype != "admin")

{

cout << "User type phai la 'user' hoac 'admin'. Nhap lai: ";

cin >> users[i].usertype;

}

}

else

{

cout << "Phan muon sua khong hop le." << endl;

return;

}

// In thông báo sau khi thông tin người dùng được cập nhật.

cout << "Thong tin nguoi dung da duoc cap nhat." << endl;

break;

}

}

// Nếu không tìm thấy người dùng cần chỉnh sửa, hiển thị thông báo.

if (!found)

{

cout << "Khong tim thay nguoi dung co ma nay trong he thong. Vui long nhap lai." << endl;

}

}

void displayUsers()

{

// Hiển thị tiêu đề danh sách người dùng trong cơ sở dữ liệu.

cout << "Danh sach nguoi dung trong co so du lieu:" << endl;

// Kiểm tra xem có người dùng nào trong cơ sở dữ liệu không.

if (numUsers == 0)

{

// Nếu không có, hiển thị thông báo và kết thúc hàm.

cout << "Co so du lieu hien khong co nguoi dung nao." << endl;

}

else

{

// Nếu có, duyệt qua mảng người dùng và hiển thị thông tin của từng người dùng.

for (int i = 0; i < numUsers; ++i)

{

cout << "-----------------------------------------" << endl;

cout << "NGUOI DUNG " << i + 1 << endl;

cout << "THONG TIN CUA NGUOI DUNG " << endl;

cout << "UserID: " << users[i].userid << endl;

cout << "Username: " << users[i].username << endl;

cout << "Password: " << users[i].password << endl;

cout << "User type: " << users[i].usertype << endl;

}

}

}

void login()

{

string userType;

// Hỏi người dùng là user hay admin.

cout << "Ban la user hay admin? (user/admin): ";

cin >> userType;

// Nếu là user:

if (userType == "user")

{

// Hiển thị chào mừng và hiển thị menu cho user.

cout << "Chao mung user!" << endl;

while (true)

{

// Hiển thị menu cho user.

cout << endl;

cout << "Menu user:" << endl;

cout << "1. Hien thi tat ca sach" << endl;

cout << "2. Thoat" << endl;

cout << "Chon: ";

int choice;

cin >> choice;

switch (choice)

{

case 1:

// Hiển thị tất cả sách.

displayBooks();

break;

case 2:

// Thoát khỏi chương trình.

cout << "Thoat ra ngoai thanh cong!" << endl;

return;

default:

// Thông báo khi người dùng nhập lựa chọn không hợp lệ.

cout << "Lua chon khong hop le. Vui long chon lai." << endl;

}

}

}

// Nếu là admin:

else if (userType == "admin")

{

string username, password;

bool isValidLogin = false;

// Yêu cầu người dùng nhập tên đăng nhập và mật khẩu.

cout << "Nhap ten dang nhap: ";

cin >> username;

cout << "Nhap mat khau: ";

cin >> password;

for (int i = 0; i < numUsers; ++i)

{

if (users[i].username == username && users[i].password == password && users[i].usertype == "admin")

{

isValidLogin = true;

currentUser = users[i];

// Hiển thị thông báo đăng nhập thành công và chào mừng admin.

cout << "Dang nhap thanh cong! Chao mung admin." << endl;

while (true)

{

// Hiển thị menu cho admin.

cout << endl;

cout << "Menu admin:" << endl;

cout << "1. Them sach" << endl;

cout << "2. Xoa sach" << endl;

cout << "3. Sua thong tin sach" << endl;

cout << "4. Hien thi tat ca sach" << endl;

cout << "5. Dang xuat" << endl;

cout << "Chon: ";

int choice;

cin >> choice;

switch (choice)

{

case 1:

// Thêm sách mới.

addBook();

break;

case 2:

// Xóa sách.

deleteBook();

break;

case 3:

// Sửa thông tin sách.

editBook();

break;

case 4:

// Hiển thị tất cả sách.

displayBooks();

break;

case 5:

// Đăng xuất.

cout << "Dang xuat...";

return;

default:

// Thông báo khi người dùng nhập lựa chọn không hợp lệ.

cout << "Lua chon khong hop le. Vui long chon lai." << endl;

}

}

}

}

if (!isValidLogin)

{

// Thông báo khi thông tin đăng nhập không hợp lệ.

cout << "Thong tin dang nhap khong hop le. Vui long nhap lai." << endl;

}

}

// Nếu không phải user hoặc admin:

else

{

// Hiển thị thông báo lỗi khi loại người dùng không hợp lệ.

cout << "Loai nguoi dung khong hop le." << endl;

}

}

int main()

{

// Vòng lặp chính của chương trình.

while (true)

{

// Hiển thị tiêu đề của chương trình quản lý thư viện.

cout << "\n===== CHUONG TRINH QUAN LY THU VIEN =====" << endl;

// Hiển thị các lựa chọn cho người dùng.

cout << "1. Them sach" << endl;

cout << "2. Xoa sach" << endl;

cout << "3. Sua thong tin sach" << endl;

cout << "4. Hien thi tat ca sach" << endl;

cout << "5. Them nguoi dung" << endl;

cout << "6. Xoa nguoi dung" << endl;

cout << "7. Sua thong tin nguoi dung" << endl;

cout << "8. Hien thi tat ca nguoi dung" << endl;

cout << "9. Dang nhap nguoi dung" << endl;

cout << "0. Thoat chuong trinh " << endl;

cout << "Nhap lua chon cua ban: ";

// Nhập lựa chọn của người dùng.

int choice;

cin >> choice;

// Xử lý lựa chọn của người dùng.

switch (choice)

{

case 1:

// Thêm sách vào cơ sở dữ liệu.

addBook();

break;

case 2:

// Xóa sách khỏi cơ sở dữ liệu.

deleteBook();

break;

case 3:

// Sửa thông tin sách trong cơ sở dữ liệu.

editBook();

break;

case 4:

// Hiển thị tất cả sách trong cơ sở dữ liệu.

displayBooks();

break;

case 5:

// Thêm người dùng vào cơ sở dữ liệu.

addUser();

break;

case 6:

// Xóa người dùng khỏi cơ sở dữ liệu.

deleteUser();

break;

case 7:

// Sửa thông tin người dùng trong cơ sở dữ liệu.

editUser();

break;

case 8:

// Hiển thị tất cả người dùng trong cơ sở dữ liệu.

displayUsers();

break;

case 9:

// Đăng nhập người dùng vào hệ thống.

login();

break;

case 0:

// Kết thúc chương trình.

cout << "Cam on ban da su dung chuong trinh." << endl;

return 0;

default:

// Thông báo khi người dùng nhập lựa chọn không hợp lệ.

cout << "Lua chon khong hop le. Vui long nhap lai." << endl;

}

}

return 0;

}

# **IV. Ưu và Nhược điểm của chương trình quản lý thư viện:**

## **Ưu điểm:**

### **1.1 Tính linh hoạt:**

Chương trình cho phép người dùng thêm, xóa, sửa thông tin sách và người dùng, cũng như hiển thị danh sách sách và người dùng trong thư viện.

### **2.1 Giao diện dễ sử dụng:**

Giao diện dựa trên dòng lệnh, dễ hiểu và dễ sử dụng, giúp người dùng tương tác với chương trình một cách thuận tiện.

### **3.1 Kiểm tra tính hợp lệ của dữ liệu:**

Chương trình kiểm tra tính hợp lệ của dữ liệu đầu vào như ISBN, date, pages, copies, username, password, usertype, đảm bảo rằng thông tin được nhập vào là chính xác và đúng định dạng.

### **4.1 Tính bảo mật:**

Chương trình yêu cầu xác thực cho các tác vụ của admin, đảm bảo rằng chỉ có người dùng có quyền mới có thể thực hiện các tác vụ quản trị như thêm, sửa, xóa sách hoặc người dùng.

## **Nhược điểm:**

### **1.2 Khả năng mở rộng hạn chế:**

Dữ liệu được lưu trữ trong mảng cố định, giới hạn bởi số lượng tối đa của sách và người dùng. Điều này có thể làm hạn chế khả năng mở rộng của chương trình nếu cần quản lý một lượng lớn hơn các mục sách hoặc người dùng.

### **2.2 Thiếu tính nhất quán trong xử lý lỗi:**

Mặc dù chương trình có kiểm tra tính hợp lệ của dữ liệu đầu vào, nhưng việc xử lý lỗi và thông báo lỗi không luôn nhất quán hoặc chi tiết. Có thể cần cải thiện để cung cấp thông báo lỗi rõ ràng và hướng dẫn người dùng cách sửa lỗi.

### **3.2 Thiếu tính bảo mật đầy đủ:**

Mặc dù yêu cầu xác thực cho admin, nhưng chương trình không cung cấp cơ chế bảo mật phức tạp như quản lý phiên đăng nhập, mã hóa mật khẩu, hoặc kiểm tra vai trò của người dùng trong mỗi yêu cầu.

### **4.2 Khả năng nhập liệu không linh hoạt:**

Chương trình có thể gặp khó khăn trong việc xử lý nhập liệu không đúng định dạng hoặc không hợp lệ từ người dùng, dẫn đến lỗi hoặc hiệu suất kém.