**ĐẠI HỌC QUỐC GIA TP.HCM**

**TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN**

**KHOA CÔNG NGHỆ THÔNG TIN**



**ĐỒ ÁN­**

**MÔN: NHẬP MÔN LẬP TRÌNH**

**NỘI DUNG: ỨNG DỤNG QUẢN LÝ THƯ VIỆN,**

**SỬ DỤNG STRUCT VÀ LƯU TRỮ DỮ LIỆU BẰNG CSV**

GIẢNG VIÊN: PHẠM MINH TUẤN

HỌ VÀ TÊN: HÀ THANH TUÂN­­­­

MSSV: 25880061

Ngày hoàn thành: 02/12/2025

Mục lục

[1. Miêu tả ứng dụng 4](#_Toc215555137)

[2. Cấu trúc mã nguồn 5](#_Toc215555138)

[3. Báo cáo chi tiết các tính năng đã hoàn thành 7](#_Toc215555139)

## **1. Miêu tả ứng dụng**

Đây là một ứng dụng **Hệ thống Quản lý Thư viện** được viết bằng ngôn ngữ C++. Ứng dụng này chạy trên giao diện dòng lệnh (console) và cho phép người dùng (giả định là thủ thư hoặc nhân viên thư viện) thực hiện các thao tác nghiệp vụ cơ bản.

**Các đặc điểm kỹ thuật được cập nhật:**

* **Mô hình hóa dữ liệu**: Sử dụng struct (User, Book, BorrowRecord) để đóng gói thông tin, thay thế hoàn toàn cho cách dùng ma trận.
* **Lưu trữ bền vững:** Hệ thống tích hợp module csv\_utils để tự động Đọc/Ghi dữ liệu ra file .csv (users.csv, books.csv, borrows.csv) mỗi khi khởi động hoặc tắt ứng dụng.

**Các chức năng chính của ứng dụng bao gồm:**

* **Quản lý Người dùng**: Cho phép đăng ký người dùng mới, xem thông tin, chỉnh sửa thông tin và xóa người dùng khỏi hệ thống.
* **Quản lý Sách**: Cho phép thêm sách mới (hoặc cập nhật số lượng nếu ISBN đã tồn tại), xem danh sách, chỉnh sửa thông tin sách, xóa sách, và tìm kiếm sách theo ISBN hoặc tên sách.
* **Quản lý Mượn/Trả**: Xử lý logic nghiệp vụ chính, bao gồm tạo phiếu mượn (liên kết một người dùng với nhiều sách) và tạo phiếu trả. Hệ thống tự động tính toán số lượng sách, ghi nhận phí phạt nếu trả sách trễ hạn hoặc làm mất sách.
* **Thống kê & Báo cáo**: Cung cấp các báo cáo thống kê về tình hình thư viện, như:
  + Tổng số sách.
  + Số lượng sách theo thể loại.
  + Tổng số người dùng (và phân loại theo giới tính).
  + Số lượng sách đang được mượn (chưa trả).
  + Danh sách những người dùng đang mượn sách quá hạn.

Chương trình có một cờ (SHOULD\_INIT\_DATA\_FOR\_TESTING) trong configs.h cho phép tự động nạp dữ liệu mẫu khi khởi động để thuận tiện cho việc kiểm thử.

## **2. Cấu trúc mã nguồn**

Dự án được tổ chức theo mô-đun hóa, tách biệt rõ ràng các chức năng khác nhau vào các tệp tin header (.h) và tệp tin triển khai (.cpp). Cấu trúc này giúp mã nguồn dễ đọc, dễ bảo trì và dễ mở rộng.

Dưới đây là mô tả vai trò của từng cụm tệp tin chính được cập nhật:

* **main.cpp:**
  + Là điểm khởi đầu (entry point) của chương trình.
  + Thực hiện việc khởi tạo dữ liệu mẫu (nếu được bật) và khởi tạo ngày giờ hệ thống.
  + Gọi hàm display\_main\_menu() để bắt đầu vòng lặp chính của ứng dụng.
  + Thực hiện chu trình: Đọc CSV -> Khởi tạo dữ liệu mẫu (nếu cần) -> Hiển thị Menu -> Ghi CSV khi thoát.
* **configs.h:**
  + Đây là tệp cấu hình trung tâm của toàn bộ dự án.
  + Nó sử dụng các chỉ thị #define để định nghĩa các hằng số quan trọng như đường dẫn các file csv, MAX\_USERS, MAX\_BOOKS, LATE\_FINE\_PER\_DAY (phí phạt), MAX\_BORROW\_DAYS (số ngày mượn tối đa), và cờ SHOULD\_INIT\_DATA\_FOR\_TESTING.
* **users.h / users.cpp:**
  + Mô-đun quản lý người dùng.
  + Định nghĩa struct User, khai báo các mảng toàn cục User USERS[MAX\_USERS] để lưu trữ dữ liệu người dùng
  + users.cpp định nghĩa các mảng này và triển khai logic cho các hàm như register\_user, edit\_user\_info, delete\_user, print\_user\_info.
* **books.h / books.cpp:**
  + Mô-đun quản lý kho sách (inventory).
  + Định nghĩa struct Book: Chứa ISBN, tên, tác giả, giá, số lượng tồn kho...
  + Khai báo mảng toàn cục: extern Book BOOKS[MAX\_BOOKS].
  + books.cpp triển khai logic cho các hàm add\_book, edit\_book\_info, delete\_book, borrow\_book (chỉ giảm số lượng), return\_book (chỉ tăng số lượng).
* **book\_ops.h / book\_ops.cpp:**
  + Mô-đun xử lý nghiệp vụ mượn/trả sách.
  + Định nghĩa struct BorrowRecord: Chứa thông tin phiếu mượn, danh sách ISBN sách mượn, ngày mượn/trả, tiền phạt.
  + Khai báo mảng toàn cục: extern BorrowRecord BORROW\_RECORDS[...].
  + Các hàm chính là create\_borrow\_card (tạo phiếu mượn mới) và create\_return\_card (tạo phiếu trả, xử lý phạt).
* **csv\_utils.h / csv\_utils.cpp:**
  + Chịu trách nhiệm nhập xuất file.
  + Các hàm: read\_users\_from\_csv, write\_users\_to\_csv (và tương tự cho books, borrows).
  + Sử dụng strtok để phân tích chuỗi CSV.

## **3. Báo cáo chi tiết các tính năng đã hoàn thành**

|  |  |  |
| --- | --- | --- |
| **Chức năng** | **% Hoàn thành** | **Ghi chú** |
| 1a. Xem danh sách độc giả | 100% | Hàm **print\_all\_users():** Duyệt mảng USERS, truy cập trực tiếp các trường USERS[i].name, USERS[i].id, hàm sẽ gọi print\_user\_info() để hiển thị thông tin chi tiết của người dùng với id khác 0 và dừng lại khi gặp id = 0 (slot trống). |
| 1b. Thêm độc giả | 100% | Hàm **register\_user()**:   1. Kiểm tra CMND đã tồn tại chưa bằng is\_existing\_user(). 2. Tìm một vị trí (index) trống trong mảng USERS (vị trí có id là 0). 3. Nếu còn chỗ, yêu cầu người dùng nhập tuần tự các thông tin (tên, ngày sinh, giới tính, email, địa chỉ). 4. Ngày tạo tài khoản được gán bằng ngày hệ thống (CURRENT\_YEAR/MONTH/DAY). 5. Ngày hết hạn được tính bằng cách gọi hàm get\_expiration\_date() (cộng thêm 48 tháng). |
| 1c. Chỉnh sửa thông tin một độc giả | 100% | Hàm **edit\_user\_info()**:   1. Tìm index nội bộ của người dùng dựa trên CMND. 2. Lần lượt hiển thị từng thông tin hiện tại (Tên, Ngày sinh, v.v.). 3. Sau mỗi thông tin, gọi hàm ask\_to\_edit\_field() để hỏi người dùng có muốn thay đổi không (nhập 'y' hoặc bỏ qua bằng Enter). 4. Nếu người dùng đồng ý, chương trình sẽ yêu cầu nhập giá trị mới và ghi đè vào vị trí tương ứng trong các mảng dữ liệu. |
| 1d. Xóa thông tin một độc giả | 100% | Hàm **delete\_user()**:   * Tìm vị trí i cần xóa. * Thực hiện vòng lặp USERS[j] = USERS[j+1] từ j=i đến cuối mảng để lấp đầy khoảng trống (Shift Left). * Reset phần tử cuối cùng. |
| 1e. Tìm kiếm độc giả theo CMND | 100% | Hàm **print\_user\_info():**   1. Nhận CMND làm đầu vào. 2. Tìm index tương ứng trong mảng USERS có id là CMND. 3. Nếu tìm thấy, in ra tất cả thông tin được lưu struct. |
| 1f. Tìm kiếm sách đã mượn theo họ tên | 100% | Hàm **print\_borrowed\_books\_by\_username**():   1. Nhận tên cần tìm kiếm làm đầu vào 2. Tìm tất cả CMND có cùng tên với đầu vào trong mảng USERS. 3. Nếu tìm thấy, in ra tất cả và phiếu mượn sách kèm thông tin chi tiết của từng CMND |
| 2a. Xem danh sách các sách trong thư viện | 100% | Hàm **print\_all\_books():** Lặp qua toàn bộ mảng BOOKS. Với mỗi phần tử có ISBN khác 0, hàm sẽ in ra thông tin sách, ngược lại, hàm dừng nếu gặp slot trống. |
| 2b. Thêm sách | 100% | Hàm **add\_book()**:   1. Kiểm tra xem có sách với ISBN tương tự đã tồn tại trong mảng chưa. 2. Nếu đã tồn tại, chỉ cộng thêm số lượng bản sao vào tại vị trí đó và kết thúc. 3. Nếu chưa tồn tại, tìm một vị trí trống đầu tiên trong mảng BOOKS để thêm vào. 4. Yêu cầu người dùng nhập đầy đủ thông tin cho sách mới (tên, tác giả, NXB, v.v.). |
| 2c. Chỉnh sửa thông tin một quyển sách | 100% | Hàm **edit\_book\_info():** Logic tương tự như edit\_user\_info().   1. Tìm index của sách qua ISBN 2. Hiển thị thông tin hiện tại và dùng ask\_to\_edit\_field() để hỏi người dùng có muốn cập nhật từng trường thông tin hay không. 3. Nếu người dùng đồng ý, chương trình sẽ yêu cầu nhập giá trị mới và ghi đè vào vị trí tương ứng trong các mảng dữ liệu. |
| 2d. Xóa thông tin sách | 100% | Hàm **delete\_book()**: Tương tự xóa độc giả, sử dụng kỹ thuật dời mảng (Shift Left) để duy trì tính liên tục của danh sách sách. |
| 2e. Tìm kiếm sách theo ISBN | 100% | Hàm **print\_book\_info\_by\_isbn()**: Tìm kiếm tuyến tính và in ra thông tin sách trên mảng BOOKS theo ISBN. |
| 2f. Tìm kiếm sách theo tên sách | 100% | Hàm **print\_book\_info\_by\_name()**:   1. Lặp qua mảng BOOKS và sử dụng hàm strcmp() để so sánh chuỗi. 2. Khi tìm thấy tên sách trùng khớp, hàm lấy ra ISBN tại index đó và gọi print\_book\_info\_by\_isbn() để hiển thị đầy đủ thông tin |
| 3. Lập phiếu mượn sách | 100% | Hàm **create\_borrow\_card()**:   1. Tìm một vị trí trống trong mảng BORROW\_RECORDS để tạo phiếu mới. 2. Yêu cầu nhập CMND, sau đó xác thực bằng is\_existing\_user() và is\_user\_expired(). 3. Gán ngày mượn là ngày hiện tại của hệ thống. 4. Lặp để người dùng nhập ISBN của các sách muốn mượn; mỗi lần nhập, gọi borrow\_book() để kiểm tra và giảm số lượng trong kho đi 1 cuốn. 5. Nếu không mượn được cuốn nào, phiếu sẽ bị hủy. |
| 4. Lập phiếu trả sách | 100% | Hàm **create\_return\_card()**:   1. Yêu cầu nhập ID phiếu mượn và tìm index tương ứng. 2. Kiểm tra xem phiếu đã được trả chưa. 3. Lặp qua danh sách từng ISBN sách đã mượn trong phiếu, hỏi người dùng sách có bị mất không. 4. Nếu sách bị mất, gọi calculate\_lost\_penalty\_by\_isbn() để tính phí phạt. 5. Nếu sách không mất, gọi return\_book() để tăng lại số lượng trong kho lên 1 cuốn. 6. Cuối cùng, gọi calculate\_late\_penalty() để tính phí trả trễ và gán ngày trả thực tế là ngày hiện tại. |
| 5a. Thống kê số lượng sách trong thư viện |  | Hàm **print\_total\_number\_of\_books()**:   1. Khởi tạo biến total\_books = 0. 2. Lặp qua toàn bộ mảng BOOKS và cộng dồn giá trị count của từng phần tử vào biến total\_books. |
| 5b. Thống kê số lượng sách theo thể loại | 100% | Hàm **print\_number\_books\_by\_genre()**: Sử dụng 2 vòng lặp lồng nhau. Vòng lặp ngoài duyệt qua từng sách để lấy thể loại, vòng lặp trong kiểm tra xem thể loại đó đã được thống kê trước đó chưa. Nếu chưa, một vòng lặp nữa sẽ chạy để cộng dồn số lượng của tất cả các sách có cùng thể loại. |
| 5c. Thống kê số lượng độc giả | 100% | Hàm **print\_number\_users()**:   1. Khởi tạo biến user\_counts = 0. 2. Lặp qua mảng USERS và tăng biến đếm cho mỗi phần tử có id khác 0. |
| 5d. Thống kê số lượng độc giả theo giới tính | 100% | Hàm **print\_number\_users\_by\_gender()**:   1. Khởi tạo 2 biến đếm cho Nam và Nữ. 2. Lặp qua mảng USERS. Nếu giá trị là 0, tăng biến đếm của Nữ; nếu là 1, tăng biến đếm của Nam. |
| 5e. Thống kê số sách đang được mượn | 100% | Hàm **print\_number\_unreturned\_books():**   1. Lặp qua mảng BORROW\_RECORDS. 2. Nếu một phiếu mượn đang hoạt động và chưa được trả (actual\_return\_date là 0), chương trình sẽ lặp tiếp qua danh sách borrowed\_isbns của phiếu đó để đếm số lượng sách. |
| 5f. Thống kê danh sách độc giả bị trễ hạn | 100% | Hàm **print\_current\_late\_users()**:   1. Lặp qua các phiếu mượn chưa trả. 2. Với mỗi phiếu, gọi hàm print\_late\_return\_user(). Hàm này dùng calculate\_days\_between() để tính số ngày từ ngày mượn đến ngày hiện tại. Nếu số ngày này lớn hơn MAX\_BORROW\_DAYS (được định nghĩa là 7), thông tin người dùng và số ngày trễ hạn sẽ được in ra. |
| Chức năng thêm 1: Xem danh sách tất cả phiếu mượn | 100% | Hàm **print\_all\_borrow\_records():**   1. Lặp qua toàn bộ mảng BORROW\_RECORDS. 2. Nếu ID phiếu mượn khác 0 (tồn tại), hàm sẽ gọi print\_a\_borrow\_record() để in chi tiết phiếu đó, bao gồm thông tin người mượn, ngày mượn, ngày trả (nếu có) và danh sách ISBN các sách đã mượn. |
| Chức năng thêm 2: Xem danh sách phiếu mượn chưa trả | 100% | Hàm **print\_unreturned\_borrows()**:   1. Lặp qua mảng BORROW\_RECORDS. 2. Hàm chỉ chọn và in ra các phiếu mượn thỏa mãn hai điều kiện: ID phiếu khác 0 (tồn tại) VÀ ngày trả thực tế actual\_return\_date bằng 0 (nghĩa là chưa trả). |
| Chức năng thêm 3: Xem danh sách phiếu trả | 100% | Hàm **print\_finished\_returns()**:   1. Lặp qua mảng BORROW\_RECORDS 2. Hàm chỉ chọn và in ra các phiếu mượn thỏa mãn hai điều kiện: ID phiếu khác 0 (tồn tại) VÀ ngày trả thực tế actual\_return\_date khác 0 (nghĩa là đã trả). Báo cáo này sẽ bao gồm tổng phí phạt (phạt trễ hạn và phạt mất sách). |