**TRƯỜNG ĐẠI HỌC PHENIKAA**

**KHOA CÔNG NGHỆ THÔNG TIN**



Lập trình hướng đối tượng (N03)

# Đề tài: Xây dựng chương trình quản lý sách và cho mượn sách tại Đại học Phenikaa

# Giảng viên hướng dẫn: Trần Đăng Hoan

# Thành viên:

* Nguyễn Quốc Khánh - 21011495
* Nguyễn Thị Hà - 21010988
* Trần Long Hải – 21010586

*Hà Nội, Ngày ...Tháng ... Năm 2023*

# **Bảng phân công công việc**

|  |  |  |
| --- | --- | --- |
| Nguyễn Quốc Khánh | 21011495 |  |
| Nguyễn Thị Hà | 21010988 |  |
| Trần Long Hải | 21010586 |  |

### 

### 

**Mục lục**

**I. Giới thiệu 4**

1. Mục đích 4

2. Phạm vi 4

3. Đối tượng sử dụng 4

4. Công nghệ sử dụng 4

**II. Phân tích yêu cầu 4**

1. Quản lý thư viện 4

2. Quản lý thủ thư 5

3. Quản lý sinh viên và mượn sách 5

4. Quản lý sách 5

**III. Thiết kế hệ thống 6**

1. Cấu trúc gói (package) và lớp (class) 6

1.1 Package Admin 6

1.2 Package Books 7

1.3 Package Borrow 7

1.4 Package Database 7

1.5 Package Librarian 7

2. Thiết kế giao diện người dùng 7

2.1 Giao diện quản lý thư viện 8

2.2 Giao diện quản lý thủ thư 9

2.3 Giao diện quản lý sinh viên và quản lý sách 11

**IV. Use Cases 15**

1. Biểu đồ Use Case chung 15

2. Biểu đồ Use Case cho Quản trị viên 16

3. Biểu đồ Use Case cho Thủ thư 17

**V. Cài đặt và triển khai 18**

1. Cài đặt môi trường phát triển 18

2. Triển khai ứng dụng 18

3. Cách cài đặt chương trình 18

4. Kiểm thử và đánh giá 22

# **I. Giới Thiệu**

## **1. Mục đích**

Bài tập lớn " Xây dựng chương trình quản lý sách và cho mượn sách tại Đại học Phenikaa " được thực hiện với mục đích xây dựng một ứng dụng quản lý thư viện hiện đại, hiệu quả, dễ sử dụng và tiết kiệm thời gian cho người dùng. Đồng thời, bài tập lớn cũng giúp sinh viên nắm vững kiến thức về lập trình hướng đối tượng.

**2. Phạm vi**

Phạm vi của bài tập lớn bao gồm việc xây dựng một ứng dụng quản lý thư viện trên nền tảng Java, hỗ trợ quản lý sách, thủ thư, sinh viên và các giao dịch mượn-trả sách. Ứng dụng này hướng tới đối tượng người dùng là quản trị viên và thủ thư của thư viện.

**3. Đối tượng sử dụng**

Đối tượng sử dụng chính của ứng dụng quản lý thư viện này là quản trị viên (admin) và thủ thư (librarian). Quản trị viên có quyền truy cập vào toàn bộ hệ thống, quản lý thông tin của thủ thư, thêm, xóa hoặc cập nhật thông tin của thủ thư. Trong khi đó, thủ thư chịu trách nhiệm quản lý sách, đọc giả và các giao dịch mượn-trả sách.

**4. Công nghệ sử dụng**

Ứng dụng quản lý thư viện được xây dựng trên nền tảng Java, sử dụng ngôn ngữ lập trình Java và các công nghệ liên quan như Java Swing (để thiết kế giao diện người dùng), JDBC (để kết nối với cơ sở dữ liệu) và MySQL (là hệ quản trị cơ sở dữ liệu quan hệ). Ngoài ra, hệ thống còn tận dụng các đặc tính của lập trình hướng đối tượng như tính đóng gói, kế thừa, đa hình và tính trừu tượng…

**II. Phân tích yêu cầu**

**1. Quản lý thư viện**

Hệ thống quản lý thư viện cần cung cấp một giao diện đơn giản và trực quan để quản lý thông tin liên quan đến thư viện. Hệ thống phải cho phép quản trị viên (admin) thực hiện các công việc sau:

- Đăng nhập vào hệ thống để truy cập các chức năng quản lý.

- Thêm, chỉnh sửa và xóa thông tin về sách.

- Thêm, chỉnh sửa và xóa thông tin về thủ thư.

- Xem danh sách các sinh viên và quản lý thông tin mượn sách.

- Tạo báo cáo về tình hình mượn/trả sách và thống kê tổng quát về thư viện.

**2. Quản lý thủ thư**

Hệ thống quản lý thư viện phải cho phép quản trị viên quản lý thông tin của thủ thư. Các chức năng quản lý thủ thư bao gồm:

- Thêm thông tin về thủ thư mới, bao gồm tên, mật khẩu, email, địa chỉ, thành phố và số điện thoại liên lạc.

- Chỉnh sửa thông tin về thủ thư hiện có.

- Xóa thông tin của thủ thư.

- Xem danh sách các thủ thư.

**3. Quản lý sinh viên và mượn sách** Hệ thống quản lý thư viện cần cung cấp chức năng để quản lý thông tin của sinh viên và việc mượn sách. Các chức năng cần thiết bao gồm:

- Thêm thông tin sinh viên mới, bao gồm tên, số thẻ sinh viên, số điện thoại, và email.

- Chỉnh sửa thông tin sinh viên hiện có.

- Xóa thông tin sinh viên.

- Xem danh sách các sinh viên.

- Ghi nhận thông tin mượn sách của sinh viên.

- Ghi nhận thông tin trả sách của sinh viên.

- Tạo báo cáo về việc mượn/trả sách của từng sinh viên.

**4. Quản lý sách**

Hệ thống quản lý thư viện phải cung cấp các chức năng để quản lý thông tin sách. Các chức năng quản lý sách bao gồm:

- Thêm thông tin sách mới, bao gồm tên sách, tác giả, nhà xuất bản, năm xuất bản, thể loại, số lượng, và mã sách.

- Chỉnh sửa thông tin sách hiện có, cho phép cập nhật các thông tin như tên sách, tác giả, nhà xuất bản, năm xuất bản, thể loại, số lượng, và mã sách.

- Xóa thông tin sách khỏi hệ thống.

- Xem danh sách các sách hiện có trong thư viện, bao gồm tên sách, tác giả, nhà xuất bản, năm xuất bản, thể loại, số lượng, và mã sách.

- Tìm kiếm sách theo tiêu chí như tên sách, tác giả, nhà xuất bản, năm xuất bản, thể loại, hoặc mã sách.

- Theo dõi số lượng sách hiện có và sách đã được mượn, cũng như thông báo khi số lượng sách gần hết.

- Tạo báo cáo thống kê về số lượng sách, tình hình mượn/trả sách và các xu hướng mượn sách theo thời gian, thể loại, hoặc sinh viên.

**III. Thiết kế hệ thống**

**1. Cấu trúc gói (package) và lớp (class)**

**1.1 Package Admin**

**AdminLogin.java**: Tạo ra một giao diện đăng nhập cho quản trị viên (admin) với các trường nhập tên đăng nhập và mật khẩu. Khi nhấn nút "Login", hệ thống sẽ xác thực thông tin đăng nhập và chuyển đến giao diện **AdminSuccess** nếu đúng, ngược lại sẽ hiển thị thông báo lỗi.

**AdminSuccess.java**: Tạo ra một giao diện thành công cho quản trị viên sau khi họ đăng nhập thành công. Giao diện này cung cấp các tùy chọn như thêm, xóa, xem danh sách thủ thư và đăng xuất.

**LibrarianLogin.java**: Tạo ra một giao diện đăng nhập cho thủ thư với các trường nhập tên đăng nhập và mật khẩu. Khi nhấn nút "Login", hệ thống sẽ xác thực thông tin đăng nhập và chuyển đến giao diện **LibrarianSuccess** nếu đúng, ngược lại sẽ hiển thị thông báo lỗi.

**LibrarianSuccess.java**: Tạo ra một giao diện thành công cho thủ thư sau khi họ đăng nhập thành công. Giao diện này cung cấp các tùy chọn như thêm sách, xem sách, mượn sách, xem sách đã mượn, trả sách và đăng xuất.

**Library.java**: Tạo ra một giao diện chính cho hệ thống quản lý thư viện. Giao diện này cung cấp hai tùy chọn: "Admin Login" để quản trị viên đăng nhập và "Librarian Login" để thủ thư đăng nhập.

**1.2 Package Books**

**BookDao.java**: Chứa các phương thức truy vấn cơ sở dữ liệu để thêm sách, cập nhật sách và kiểm tra sách có sẵn hay không.

**BooksForm.java**: Tạo ra giao diện người dùng cho việc nhập thông tin sách mới và gọi phương thức từ lớp **BookDao** để thêm sách vào cơ sở dữ liệu.

**ReturnBook.java**: Tạo ra giao diện người dùng để nhập thông tin trả sách và gọi phương thức từ lớp **ReturnBookDao** để trả sách và cập nhật cơ sở dữ liệu.

**ReturnBookDao.java**: Chứa các phương thức truy vấn cơ sở dữ liệu để xóa thông tin mượn sách và cập nhật lại số lượng sách sau khi trả sách.

**ViewBooks.java**: Tạo ra giao diện người dùng để hiển thị danh sách các sách có trong cơ sở dữ liệu.

**1.3 Package Borrow**

**BorrowBookDao.java**: Kiểm tra sách có sẵn trong cơ sở dữ liệu, lưu thông tin mượn sách và cập nhật số lượng sách khi mượn.

**BorrowBookForm.java**: Tạo ra giao diện người dùng cho việc nhập thông tin mượn sách và thêm thông tin mượn sách vào cơ sở dữ liệu.

**ViewBorrowBooks.java**: Tạo ra giao diện người dùng để hiển thị danh sách các sách đã mượn

**1.4 Package Database**

**DB,java**: Chứa phương thức để thiết lập kết nối với cơ sở dữ liệu MySQL. Phương thức getConnection() sẽ trả về đối tượng Connection để kết nối với cơ sở dữ liệu 'library\_managerment'.

**LM.sql**: Chứa các câu lệnh SQL để tạo cơ sở dữ liệu 'library\_managerment', tạo các bảng 'librarian', 'issuebooks' và 'books', cũng như chèn dữ liệu mẫu vào các bảng đó. Các câu lệnh SQL giúp tạo ra cấu trúc và dữ liệu ban đầu cho ứng dụng quản lý thư viện.

**1.5 Package Librarian**

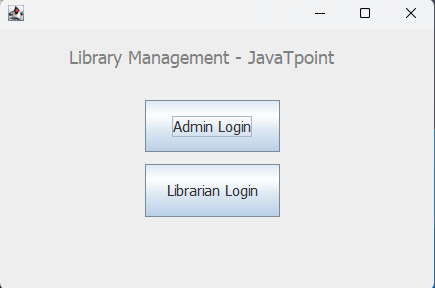
**DeleteLibrarian.java**: Dùng để gọi và thực hiện việc xóa thủ thư thông qua phương thức LibrarianDao.delete(id) xóa thông tin của một thủ thư thông qua ID.

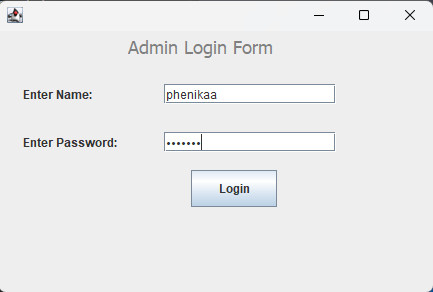
**LibrarianDao.java**: Để thao tác với cơ sở dữ liệu liên quan đến thủ thư như lưu, xóa, kiểm tra tên đăng nhập và mật khẩu

**LibrarianForm.java**: Dùng để tạo một giao diện để nhập thông tin thủ thư và lưu vào cơ sở dữ liệu

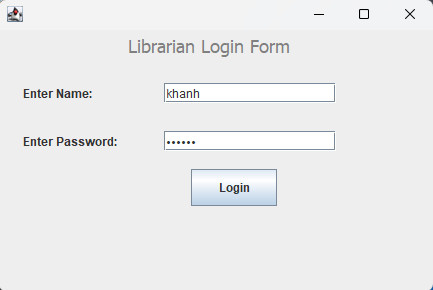
**ViewLibrarian.java**: Dùng để hiển thị danh sách các thủ thư trong cơ sở dữ liệu

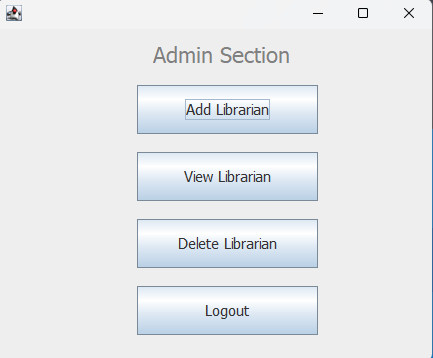
**2. Thiết kế giao diện người dùng**

2.1 Giao diện quản lý thư viện

Ảnh 1: Giao diện mở đầu

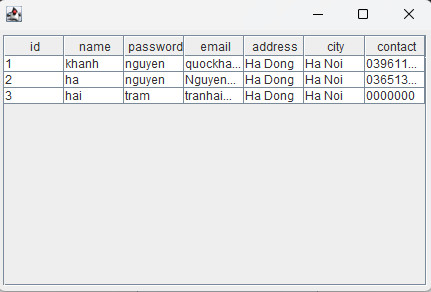
Ảnh 2: Đăng nhập quản trị

Ảnh 3: Đăng nhập thủ thư

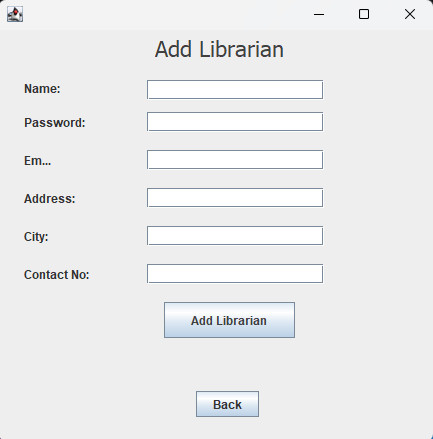
2.2 Giao diện quản lý thủ thư

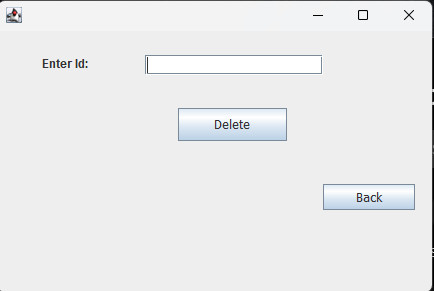
Ảnh 4: Giao diện quản lý thủ thư

Ảnh 5: Thêm thủ thư

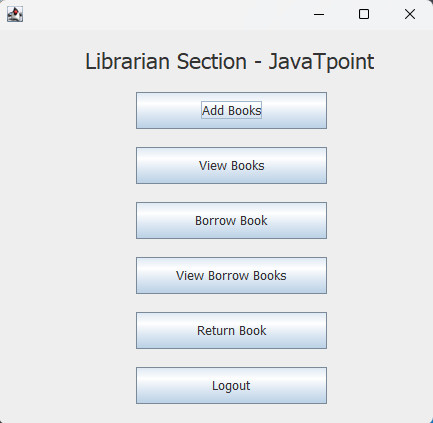


Ảnh 6: Xem thông tin thủ thư

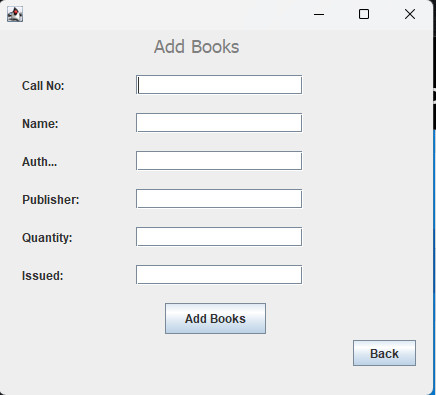
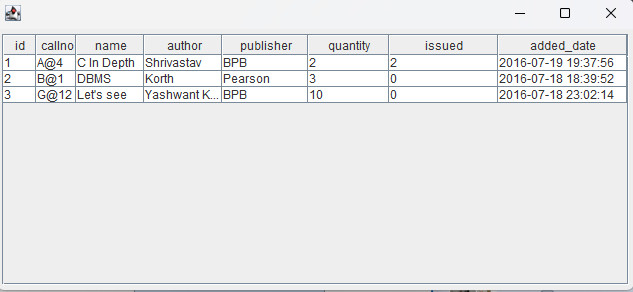




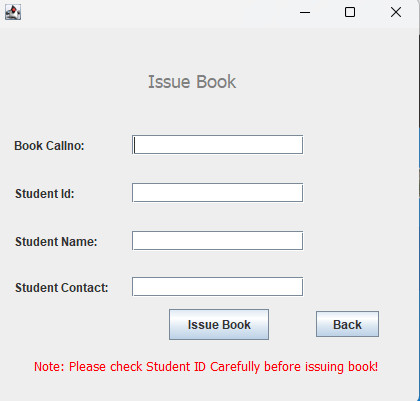
Ảnh 7: Xóa thủ thư

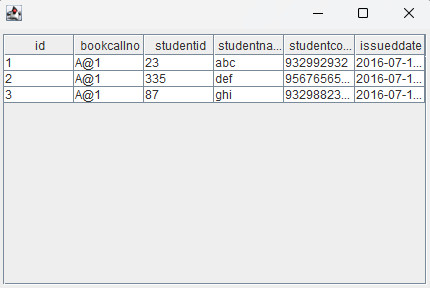
2.3 Giao diện quản lý sinh viên và quản lý sách

Ảnh 8: Giao diện quản lý sinh viên và sách

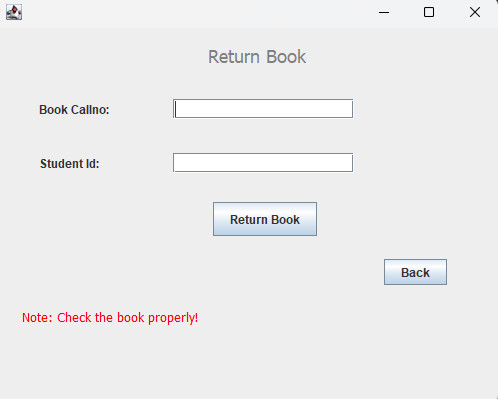
Ảnh 9: Thêm sách

Ảnh 10: Xem danh sách sách

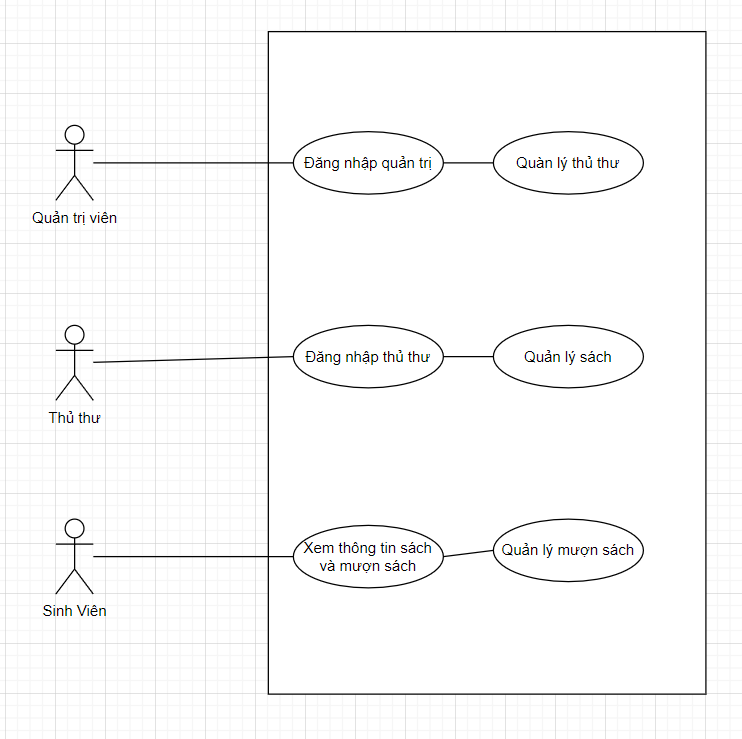
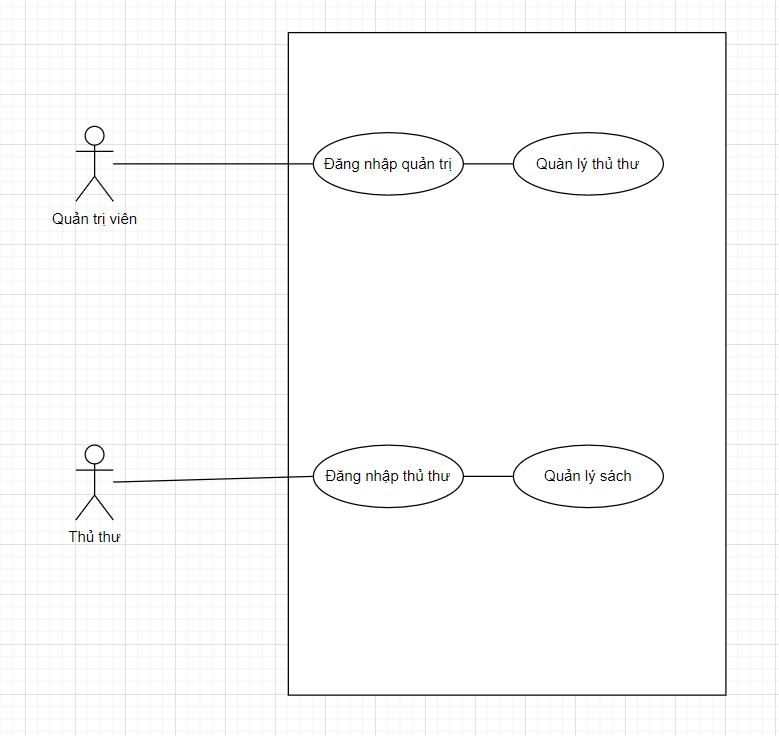


Ảnh 11: Mượn sách

Ảnh 12: Quản lý mượn sách

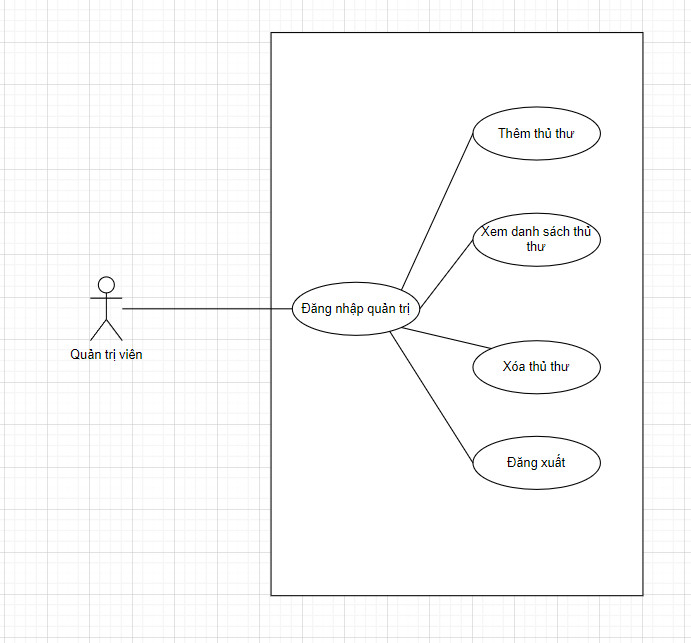
Ảnh 13: Trả sách

**IV. Use Cases**

**1. Biểu đồ Use Case chung**

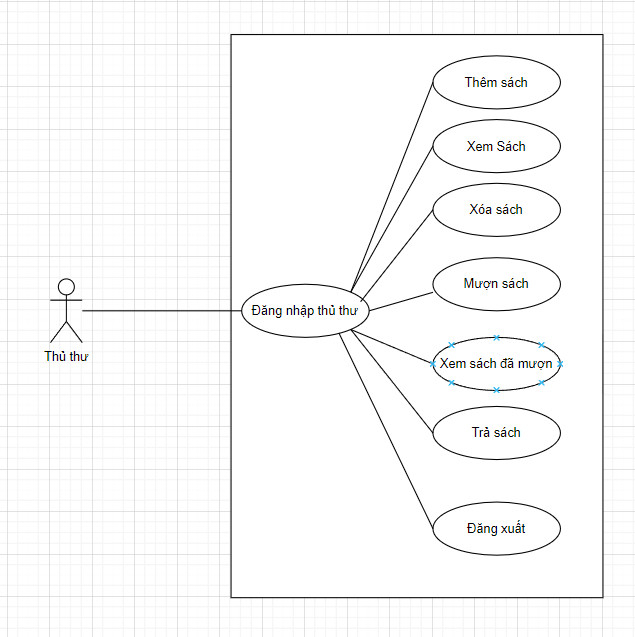
Ảnh 14: Use Cases Chung

**2. Biểu đồ Use Case cho Quản trị viên**



Ảnh 15: Use Cases Quản trị viên

|  |  |
| --- | --- |
| **Tên Use Case** | Quản trị viên |
| **Mô tả** | Cho phép người quản trị đăng nhập để thực hiện các chức năng |
| **Actor** | Admin |
| **Luồng sự kiện** | 1. Hệ thống hiển thị trang chủ đăng nhập admin 2. Admin tạo các tài khoản, xem, xóa thủ thư 3. Use Case kết thúc |

**3. Biểu đồ Use Case cho Thủ thư**

Ảnh 16: Use Cases Thủ thư

|  |  |
| --- | --- |
| **Tên Use Case** | Thủ thư |
| **Mô tả** | Cho phép thủ thư đăng nhập để thực hiện các chức năng |
| **Actor** | Librarian |
| **Luồng sự kiện** | 1. Hệ thống hiển thị trang chủ đăng nhập libraryan 2. Thủ thư sẽ thực hiện các chức năng quản lsi sách và quản lý sinh viên mượn sách 3. Use Case kết thúc |

**V. Chức năng và giao diện**

**VI. Cài đặt và triển khai**

**1. Cài đặt môi trường phát triển**

- Cài đặt Java Development Kit (JDK) phiên bản thích hợp (ít nhất JDK 8).

- Cài đặt một môi trường phát triển (IntelliJ IDEA)

- Cài đặt và cấu hình cơ sở dữ liệu MySQL

- Thêm thư viện JDBC Connector (như MySQL Connector/J) vào dự án để kết nối với cơ sở dữ liệu.

**2. Triển khai ứng dụng**

- Tạo cấu trúc thư mục dự án theo cấu trúc gói (package) và lớp (class) đã thiết kế.

- Tạo các lớp (class) và viết mã cho các chức năng theo yêu cầu đã phân tích.

- Tạo cơ sở dữ liệu và các bảng liên quan (như: books, librarian, borrow, …) trên hệ quản trị cơ sở dữ liệu.

- Kiểm tra kết nối giữa ứng dụng và cơ sở dữ liệu thông qua các phương thức trong lớp Database

- Triển khai giao diện người dùng cho các chức năng liên quan (quản lý thư viện, quản lý thủ thư, quản lý sinh viên và mượn sách, quản lý sách).

- Thêm các sự kiện và xử lý sự kiện cho các thành phần giao diện (như nút, menu, bảng).

**3. Cách cài đặt chương trình**

**-** Clone code từ git:

git clone <https://github.com/qkhanh711/OOPproject.git>

- Sau khi clone có folder:  
 OOPproject/

LMP/

idea/

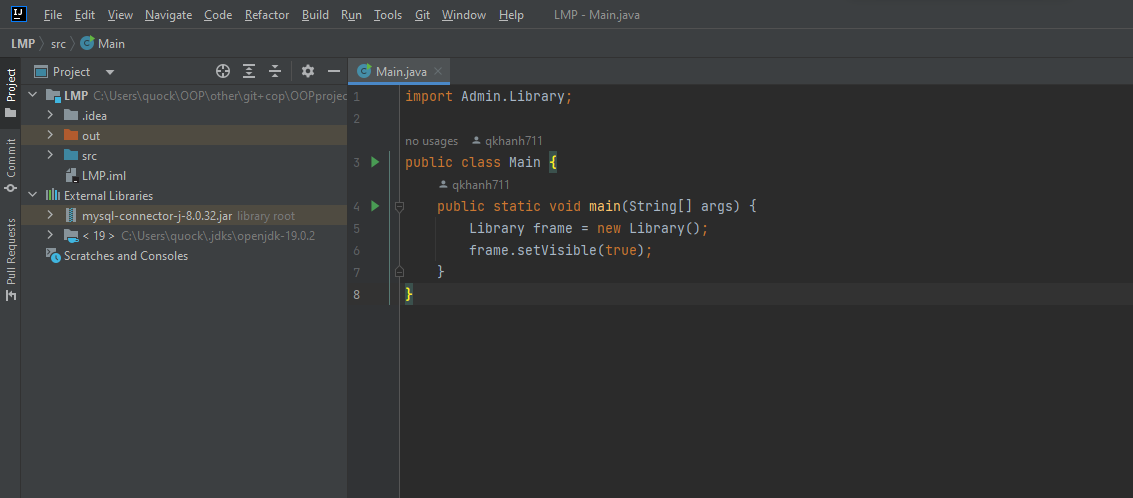
out/

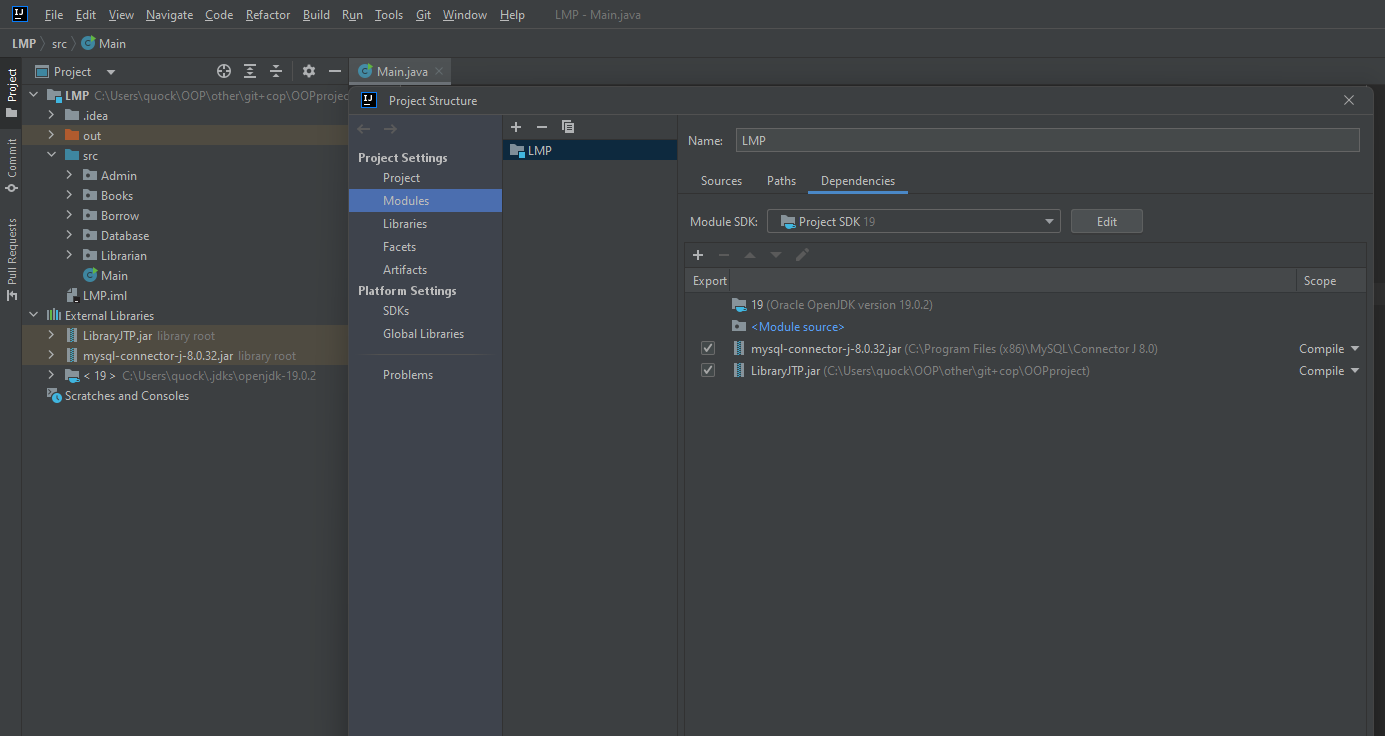
src/

LibraryJTP/

**Bước 1:** Cài IDE Intelij tại link”:

<https://www.jetbrains.com/idea/download/#section=windows>

**Bước 2:** Đăng ký tài khoản jetbrain để sử dụng IDE, sau khi mở Intelij chọn Open -> LMP -> OK kết quả như hình dưới:

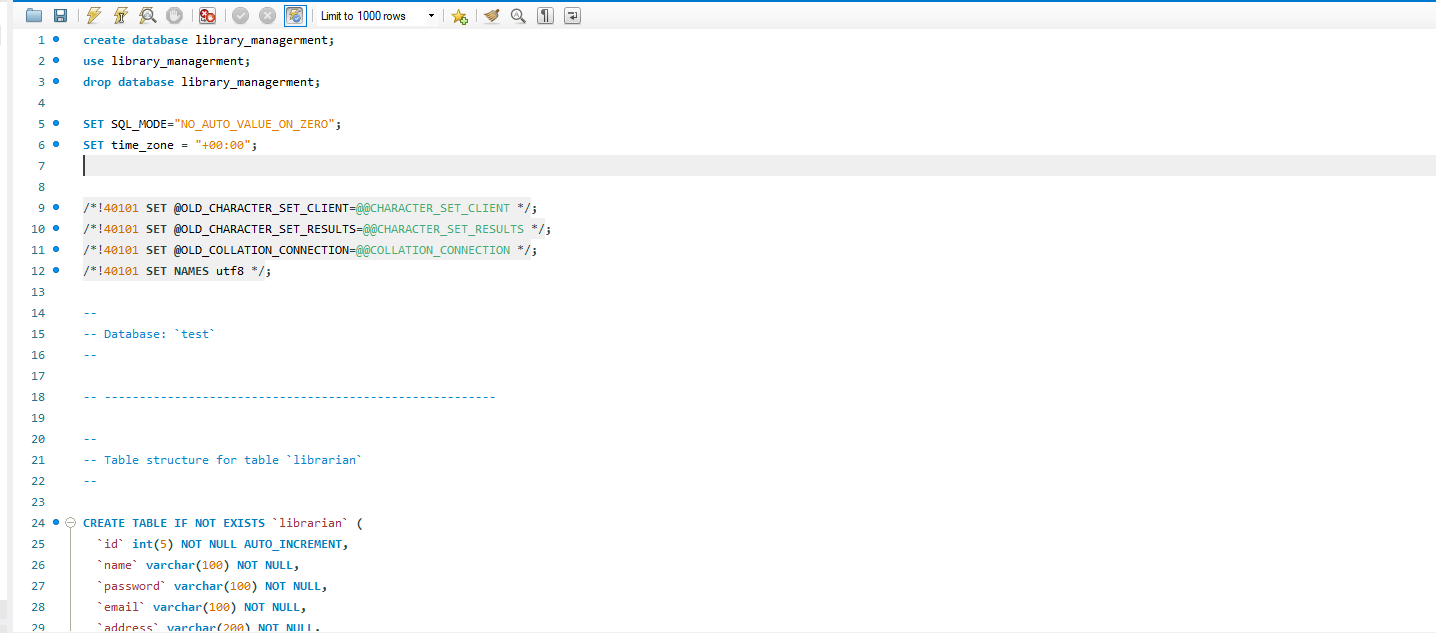
**Bước 3** Chọn File -> Project Structure -> Modules -> Dependencies -> + -> JARs or Directories... -> tìm đường dẫn LibraryJPT.jar + mysql-connector

**Bước 4** Vào package Database:

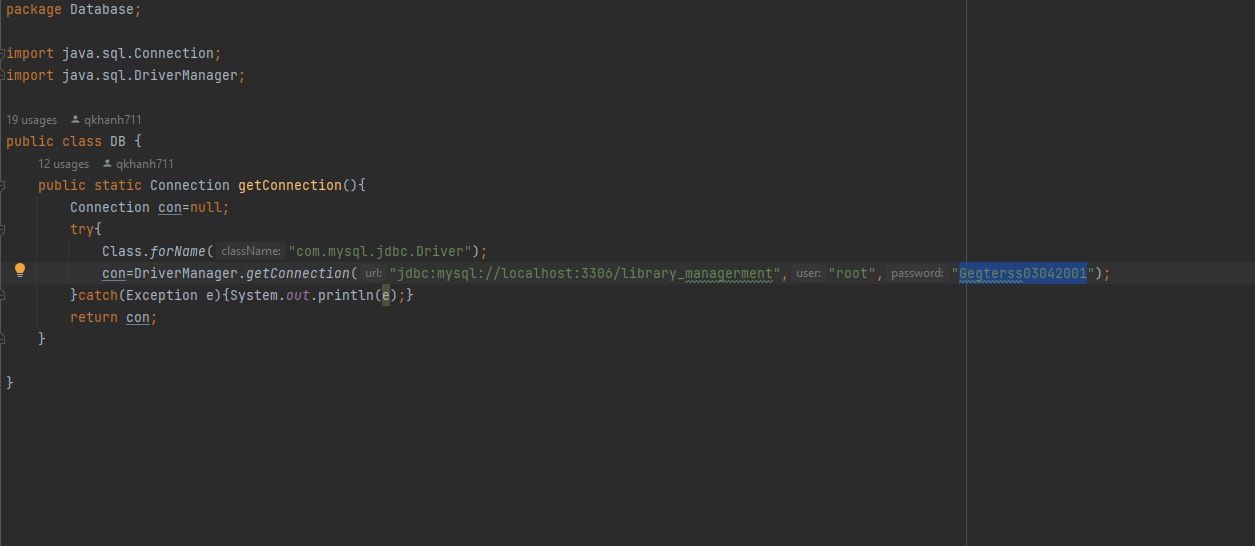
Mở mysql trên máy, chạy từng dòng trong LM.sql (trừ dòng drop database ...) để:

- tạo database

- tạo bảng

 - thêm nội dung vào trong bảng

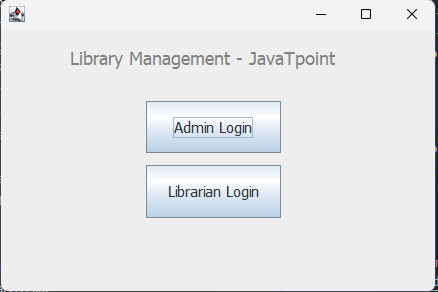
Chọn DB đổi mật khẩu mysql (phần bôi đen)



**Bước 5:** Chạy file Main: Giao diện ra như hình dưới, chọn Admin Login để tạo tài khoản Libararian -> nhập tài khoản mật khẩu là:

**phenikaa**

**k15cntt**



**4. Kiểm thử và đánh giá**

- Kiểm tra chức năng đảm bảo chúng hoạt động đúng yêu cầu.

- Kiểm tra giao diện người dùng, đảm bảo rõ ràng, dễ sử dụng và thân thiện.

- Kiểm tra hiệu năng và độ ổn định của ứng dụng trong quá trình sử dụng.