Graphical user interface, text, application

Description automatically generated

**TÀI LIỆU DỰ ÁN**

**QUẢN LÝ THƯ VIỆN - PNamLib**

A picture containing icon

Description automatically generated

|  |  |
| --- | --- |
| GIẢNG VIÊN | : TRẦN THỊ HƯỜNG |
| SINH VIÊN | : NGUYỄN TRỌNG LUÂN |
| LỚP | : CP17201/SU22 |

MAY 25, 2022

PHƯƠNG NAM LIBRARY

# **1.Giới thiệu dự án**

## Giới thiệu PNLib

Thư viện Phương Nam (PNLib) là một thư viện nhỏ, chuyên cho thuê các loại sách phục vụ cho các bạn đọc như “Kinh tế”, “Ngoại Ngữ”, “Công nghệ thông tin”, “Ẩm thực”, “Sức Khoẻ” … Việc quản lý các đầu sách, các phiếu mượn sách, thành viên hiện đang được thư viện quản lý trên sổ sách bằng tay. Việc này gây khó khăn cho thư viện, tốn thời gian ghi chép, và sai sót nhiều trong thống kê. Vì vậy, PNLib mong muốn xây dựng một phần mềm chạy trên thiết bị di động cá nhân để hỗ trợ giải quyết các khó khăn trên

## **1.1.Yêu cầu của PNLib**

Yêu cầu như sau:

**QUẢN LÝ**

Quản lý các thể loại sách

Quản lý sách

Quản lý thành viên

Quản lý phiếu mượn (mỗi phiếu chỉ cho mượn 01 quyển sách

**THỐNG KÊ**

Thống kê 10 sách mượn nhiều nhất.

Thống kê doanh thu theo thời gian chọn (từ ngày -> đến ngày)

**BẢO MẬT**

Muốn truy cập vào ứng dụng, admin và thủ thư phải đăng nhập.

Admin có quyền tạo tài khoản thủ thư.

Admin & thủ thư có quyền đổi mật khẩu.

**ĐÁNH GIÁ DỰ ÁN:**

Dự án với yêu cầu khả thi, trong khả năng thực hiện

# **2.Kế hoạch dự án**

Contents

[**1.Giới thiệu dự án** 2](#_Toc106089237)

[Giới thiệu PNLib 2](#_Toc106089238)

[**1.1.Yêu cầu của PNLib** 2](#_Toc106089239)

[**2.Kế hoạch dự án** 2](#_Toc106089240)

[**2.1. Sơ đồ use cases** 5](#_Toc106089241)

[**2.2. Đặc tả yêu cầu hệ thống** 5](#_Toc106089242)

[**2.2.1. Quản lý thành viên** 5](#_Toc106089243)

[**2.2.2. Quản lý sách** 5](#_Toc106089244)

[**2.2.3. Quản lý loại sách** 6](#_Toc106089245)

[**2.2.4. Quản lý phiếu mượn** 6](#_Toc106089246)

[**2.2.5. Thống kê** 6](#_Toc106089247)

[**2.2.6. Đăng nhập** 7](#_Toc106089248)

[**2.2.7. Đổi mật khẩu** 7](#_Toc106089249)

[**3.Sơ đồ triển khai và yêu cầu hệ thống** 7](#_Toc106089250)

[**3.1. Sơ đồ triển khai** 7](#_Toc106089251)

[**3.2. Yêu cầu hệ thống** 8](#_Toc106089252)

[**4.Thiết kế ứng dụng** 8](#_Toc106089253)

[**4.1.Mô hình công nghệ ứng dụng** 8](#_Toc106089254)

[**4.2. Thực thể** 10](#_Toc106089255)

[**4.2.1. Sơ đồ quan hệ thực thể** 10](#_Toc106089256)

[**4.2.2. Chi tiết thực thể** 10](#_Toc106089257)

[**4.3. Phác thảo giao diện** 15](#_Toc106089258)

[**4.3.1. Sơ đồ tổ chức giao diện** 33](#_Toc106089259)

[**4.3.2. Giao diện màn hình chính** 34](#_Toc106089260)

[**4.3.3. Giao diện quản lý** 37](#_Toc106089261)

[**4.3.4. Giao diện hỗ trợ khác** 45](#_Toc106089262)

[**5. Thực hiện dự án** 47](#_Toc106089263)

[**5.1. Tạo giao diện** 47](#_Toc106089264)

[**5.1.1. Các màn hình quản lý** 47](#_Toc106089265)

[**5.1.2. Các màn hình quản lý** 51](#_Toc106089266)

[**5.1.3 Màn hình thống kê theo lượt mượn** 55](#_Toc106089267)

[**5.1.4. Màn hình thống kê doanh thu theo tháng trong 1 năm** 57](#_Toc106089268)

[**5.1.5. Màn hình thống kê doanh thu theo năm trong 1 thập kỷ** 58](#_Toc106089269)

[**5.1.6. Các màn hình hỗ trợ khác** 60](#_Toc106089270)

[**5.2. Tạo CSDL với SQLITE và SQLSEVER** 62](#_Toc106089271)

[**5.2.1. Sơ đồ quan hệ** 62](#_Toc106089272)

[**5.2.2. Chi tiết các bảng** 63](#_Toc106089273)

[**5.3. Lập trình cho Controller chức năng thống kê bằng ASP.NET Core** 82](#_Toc106089274)

[**5.3.1. Thống kê doanh thu theo tháng trong năm** 82](#_Toc106089275)

[**5.3.2. Thống kê doanh thu theo năm trong 1 thập kỷ** 83](#_Toc106089276)

[**5.3.3. Thống kê tỉ lệ mượn theo từng loại sách** 83](#_Toc106089277)

[**5.3.4. Thống kê top 10 sách được mượn nhiều nhất** 84](#_Toc106089278)

[**5.4. Lập trình CSDL SQLite (SQLiteOpenHelper+Model+Dao)** 85](#_Toc106089279)

[**5.4.1. SQLiteOpenHelper** 85](#_Toc106089280)

[**5.4.2. Model Class và DAO** 88](#_Toc106089281)

[**5.5. Lập trình chức năng** 95](#_Toc106089282)

[**5.5.1. Màn hình chính User (UserMainActivity)** 95](#_Toc106089283)

[**5.4.2. Màn hình chính Admin (AdminMainActivity)** 95](#_Toc106089284)

[**5.5.3. Màn hình thống kê doanh thu tháng (RevenueFragment)** 96](#_Toc106089285)

[**5.5.4. Màn hình thống kê doanh thu năm(RevenueFragmentYear)** 96](#_Toc106089286)

[**5.5.5. Màn hình thống kê top 10 loại sách(TopBorrowedFragment)** 96](#_Toc106089287)

[**5.5s.6. Màn hình đăng nhập(LoginActivity)** 96](#_Toc106089288)

[**6. Kiểm thử phần mềm và sửa lỗi** 96](#_Toc106089289)

[**6.1. LoginActivity** 96](#_Toc106089290)

[**6.2. AdminMainActivity** 96](#_Toc106089291)

[**6.3. UserMainActivity** 97](#_Toc106089292)

[**7. Đóng gói và triển khai** 97](#_Toc106089293)

[**7.1. Sản phẩm phần mềm** 97](#_Toc106089294)

[**8. Kết luận** 97](#_Toc106089295)

[**8.1. Khó khăn** 97](#_Toc106089296)

[**8.2. Thuận lợi** 97](#_Toc106089297)

## **2.1. Sơ đồ use cases**

Diagram

Description automatically generated

## **2.2. Đặc tả yêu cầu hệ thống**

### **2.2.1. Quản lý thành viên**

**Mô tả chức năng:**

Chức năng quản lý thành viên được sử dụng để quản lý thông tin thành viên. Chức năng này liệt kê danh sách thành viên, them thành viên mới, cập nhật hoặc xóa thành viên đã tồn tại.

**Dữ liệu liên quan:**

Thông tin của mỗi thành viên bao gồm: Mã thành viên, Họ tên thành viên, cấp bậc thành viên và số lần mượn sách

**Đối tượng sử dụng:**

Sau khi đăng nhập và online thì user có thể sử dụng chức năng này, nếu đăng nhập ở chế độ offline thì thủ thư chỉ có thể xem không thể thêm, xóa, sửa.

### **2.2.2. Quản lý sách**

**Mô tả chức năng:**

Chức năng quản lý sách được sử dụng để quản lý thông tin sách. Chức năng này liệt kê danh sách của sách hiện có, thêm sách mới, cập nhật hoặc xóa sách đã tồn tại.

**Dữ liệu liên quan:**

Thông tin của mỗi thành viên bao gồm: Mã sách, tên sách, giá thuê, số lượng bản sao, thông tin loại sách, thông tin nhà xuất bản, thông tin ngôn ngữ sách.

**Đối tượng sử dụng:**

Sau khi đăng nhập và online thì user có thể sử dụng chức năng này, nếu đăng nhập ở chế độ offline thì user chỉ có thể xem không thể thêm, xóa, sửa.

### **2.2.3. Quản lý loại sách**

**Mô tả chức năng:**

Chức năng quản lý loại sách được sử dụng để quản lý thông tin loại sách. Chức năng này liệt kê danh sách loại sách, thêm loại sách mới, cập nhật hoặc xóa loại sách đã tồn tại.

**Dữ liệu liên quan:**

Thông tin của mỗi sách bao gồm: Mã loại sách, tên loại sách.

**Đối tượng sử dụng:**

Sau khi đăng nhập và online thì user có thể sử dụng chức năng này, nếu đăng nhập ở chế độ offline thì user chỉ có thể xem không thể thêm, xóa, sửa.

### **2.2.4. Quản lý phiếu mượn**

**Mô tả chức năng:**

Chức năng quản lý phiếu mượn được sử dụng để quản lý thông tin phiếu mượn. Chức năng này liệt kê danh sách phiếu mượn, thêm phiếu mượn mới, cập nhật hoặc xóa phiếu mượn đã tồn tại.

**Dữ liệu liên quan:**

Thông tin của mỗi thành viên bao gồm: Mã phiếu mượn, ngày bắt đầu, ngày hết hạn, sách được mượn, thành viên mượn và người tạo phiếu mượn.

**Đối tượng sử dụng:**

Sau khi đăng nhập và online thì user có thể sử dụng chức năng này, nếu đăng nhập ở chế độ offline thì user chỉ có thể xem không thể thêm, xóa, sửa.

### **2.2.5. Thống kê**

**Mô tả chức năng:**

Chức năng thống kê được sử dụng để xem thông tin thống kê của thư viện.bằng cách thực hiện các truy vấn trên các bảng bằng các mệnh đề JOIN, GROUP BY,…

**Dữ liệu liên quan:**

Thống kê theo các nội dung: Thống kê top 10 cuốn sách mượn nhiều nhất, thống kê tỉ lệ mượn theo từng loại sách, thống kê top 10 người mượn sách nhiều nhất

**Đối tượng sử dụng:**

Sau khi đăng nhập thì user có thể sử dụng chức năng này.

### **2.2.6. Đăng nhập**

**Mô tả chức năng:**

Chức năng đăng nhập được xử dụng đề xác thực user và admin. Sau khi đăng nhập hệ thống sẽ xác thực người dùng, nếu thông tin đăng nhập là đúng sever sẽ trả lại cho người dùng chuỗi token hợp lệ và chương trình sẽ dùng chuỗi token này như là chìa khóa để gửi các request đến sever.

**Dữ liệu liên quan:**

Thông tin của nhân viên thư viện: mã nhân viên, mật khẩu nhân viên

**Đối tượng sử dụng:**

Nhân viên thư viện bao gồm admin và user

### **2.2.7. Quản lý nhân viên thư viện**

**Mô tả chức năng:**

Chức năng quản lý nhân viên được sử dụng để quản lý thông tin nhân viên. Chức năng này liệt kê danh sách nhân viên, thêm nhân viên mới, cập nhật hoặc xóa nhân viên đã tồn tại.

**Dữ liệu liên quan:**

Thông tin của mỗi nhân viên bao gồm: Mã nhân viên, họ tên nhân viên, mật khẩu nhân viên, vai trò nhân viên.

**Đối tượng sử dụng:**

Sau khi đăng nhập và online thì admin có thể sử dụng chức năng này, nếu đăng nhập ở chế độ offline thì admin chỉ có thể xem không thể thêm, xóa, sửa.

### **2.2.8. Đổi mật khẩu**

**Mô tả chức năng:**

Chức năng thống kê được sử dụng để đổi mật khẩu của user.

**Dữ liệu liên quan:**

Thông tin của nhân viên thư viện: mã nhân viên, mật khẩu nhân viên

**Đối tượng sử dụng:**

Sau khi đăng nhập thì user có thể sử dụng chức năng này.

# **3.Sơ đồ triển khai và yêu cầu hệ thống**

## **3.1. Sơ đồ triển khai**

Diagram

Description automatically generated

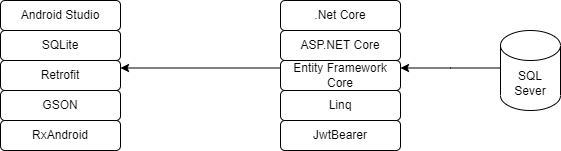
## **3.2. Yêu cầu hệ thống**

Thiết bị: thiết bị Android 9+

Công cụ: host, domain, SQLite, SQL Sever, Enity Framework Core, ASP.NET core.

# **4.Thiết kế ứng dụng**

## **4.1.Mô hình công nghệ ứng dụng**



* + Sử dụng SQL Sever để lưu dữ liệu.
  + Sử dụng Framework ASP.NET Core để ánh xạ các đối tượng của CSDL và viết các API thao tác với sever.
  + Sử dụng JWT để xác thực.
  + Sử dụng Retrofit, GSON, RxAndroid.
  + Sử dụng SQLite để lưu trữ dữ liệu tải về từ sever.
  + Sử dụng Android Studio để lập trình cho controllers.

**Ý tưởng sử dụng Framework ASP.NET Core:**

Sau khi tạo các bảng trên SQL Sever Studio, sử dụng Entity Framework Core bằng công cụ Database First để chuyển đổi các đối tượng trong cơ sở dữ liệu hiện có thành các Models trong ASP.NET Core và sử dụng thư viện LinQ để thực hiện các câu truy vấn với CSDL qua ASP.NET Core.

**Ý tưởng sử dụng JWT:**

Sau khi xác thực thành công hệ sever sẽ sinh ra một đoạn mã Token và gửi lại cho phía client, và client sẽ lưu lại đoạn Token này để dùng cho các request lên sever sau này

**Ý tưởng sử dụng Retrofit, GSON và RxAndroid:**

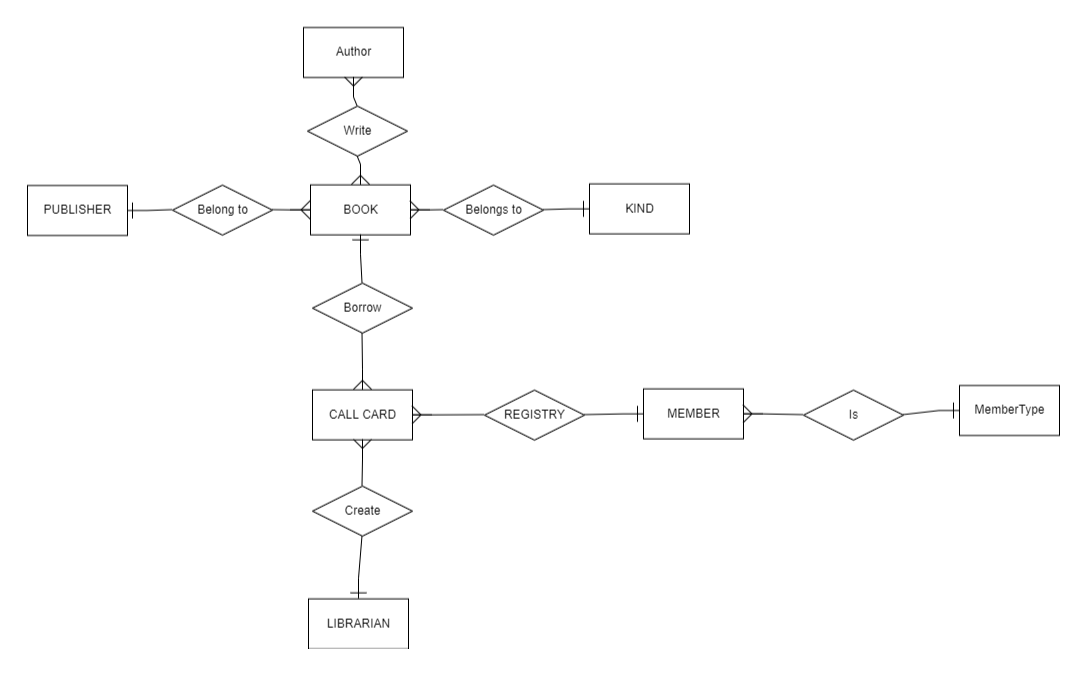
Để thao tác với các API trên ứng dụng Android sẽ sử dụng Retrofit, Retrofit sử dụng cho việc truy xuất và tải lên JSON thông qua web services. Để truy xuất và tải lên JSON ta dùng GSON là một thư viện Java để chuyển đổi đối tượng JSON thành đối tượng Java hoặc ngược lại, và cuối cùng là sử dụng RxAndroid để phát dữ liệu và nhận dữ liệu được truy xuất từ Retrofit sau khi đã được chuyển đổi bằng GSON qua Observable và thực hiện cập nhật bất đồng bộ trên UI.

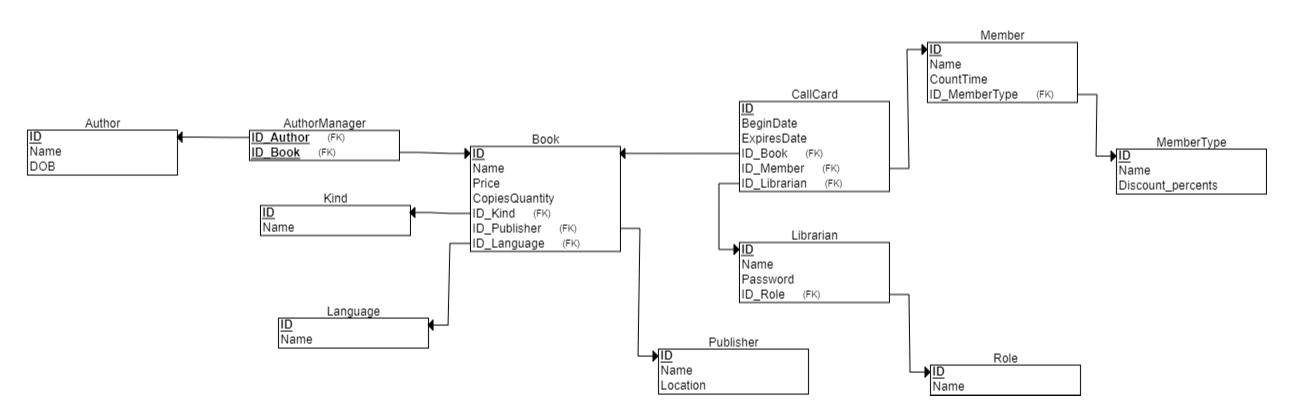
**Ý tưởng sử dụng SQLite, SQL Sever:**

Sau khi khởi động lần đầu, ứng dụng sẽ gửi các request lên sever để lấy về dữ liệu và cập nhật vào SQLite cho lần sử dụng sau và các thao tác với dữ sau lần khởi động đầu tiên cũng sẽ được cập nhật lên SQLite đồng thời SQL Sever

## **4.2. Thực thể**

### **4.2.1. Sơ đồ quan hệ thực thể**





### **4.2.2. Chi tiết thực thể**

#### **4.2.2.1. Thực thể Author**

|  |  |  |
| --- | --- | --- |
| Author | | |
| **PK** | ID | Int |
|  | Name | Nvarchar |
|  | DOB | Date |
|  |  |  |

|  |  |  |
| --- | --- | --- |
| **Thuộc tính** | **Kiểu dữ liệu** | **Mô tả** |
| ID | Int | Mã tác giả |
| Name | String | Tên tác giả |
| DOB | Date | Ngày sinh |

#### **4.2.2.2. Thực thể Kind**

|  |  |  |
| --- | --- | --- |
| Kind | | |
| **PK** | ID | Int |
|  | Name | Nvarchar |

|  |  |  |
| --- | --- | --- |
| **Thuộc tính** | **Kiểu dữ liệu** | **Mô tả** |
| ID | Int | Mã loại sách |
| Name | String | Tên loại sách |

#### **4.2.2.3. Thực thể Language**

|  |  |  |
| --- | --- | --- |
| Language | | |
| **PK** | ID | Int |
|  | Name | Nvarchar |

|  |  |  |
| --- | --- | --- |
| **Thuộc tính** | **Kiểu dữ liệu** | **Mô tả** |
| ID | Int | Mã ngôn ngữ |
| Name | String | Tên ngôn ngữ |

#### **4.2.2.4. Thực thể Book**

|  |  |  |
| --- | --- | --- |
| Language | | |
| **PK** | ID | Int |
|  | Name | Nvarchar |
|  | Price | Int |
|  | CopiesQuantities | Int |
|  | ID\_Kind | Int |
|  | ID\_Publisher | Int |
|  | ID\_Language | Int |

|  |  |  |
| --- | --- | --- |
| **Thuộc tính** | **Kiểu dữ liệu** | **Mô tả** |
| ID | Int | Mã tác giả |
| Name | String | Tên tác giả |
| Price | Int | Ngày sinh |
| CopiesQuantities | Int | Số bản sao |
| ID\_Kind | Int | Mã loại sách |
| ID\_Publisher | Int | Mã nhà xuất bản |
| ID\_Language | Int | Mã ngôn ngữ |

#### **4.2.2.5. Thực thể Publisher**

|  |  |  |
| --- | --- | --- |
| Publisher | | |
| **PK** | ID | Int |
|  | Name | Nvarchar |
|  | Location | Nvarchar |

|  |  |  |
| --- | --- | --- |
| **Thuộc tính** | **Kiểu dữ liệu** | **Mô tả** |
| ID | Int | Mã tác giả |
| Name | String | Tên nhà xuất bản |
| Location | String | Địa chỉ |

#### **4.2.2.6. Thực thể Librarian**

|  |  |  |
| --- | --- | --- |
| Librarian | | |
| **PK** | ID | Varchar |
|  | Name | Nvarchar |
|  | Password | Varchar |
|  | ID\_Role | Int |
|  |  |  |

|  |  |  |
| --- | --- | --- |
| **Thuộc tính** | **Kiểu dữ liệu** | **Mô tả** |
| ID | String | Mã nhân viên |
| Name | String | Tên nhân viên |
| Password | String | Mật khẩu |
| ID\_Role | Int | Mã vai trò |

#### **4.2.2.7. Thực thể Callcard**

|  |  |  |
| --- | --- | --- |
| Callcard | | |
| **PK** | ID | Int |
|  | BeginDate | Date |
|  | ExpiresDate | Date |
|  | ID\_Book | Int |
|  | ID\_Member | Int |
|  | ID\_Librarian | Int |

|  |  |  |
| --- | --- | --- |
| **Thuộc tính** | **Kiểu dữ liệu** | **Mô tả** |
| ID | Int | Mã phiếu mượn |
| BeginDate | String | Tên thủ thư |
| ExpiresDate | String | Mật khẩu |
| ID\_Book | Int | Mã sách |
| ID\_Member | Int | Mã thành viên |
| ID\_Librarian | Int | Mã thủ thư |

#### **4.2.2.8. Thực thể Member**

|  |  |  |
| --- | --- | --- |
| Member | | |
| **PK** | ID | Int |
|  | Name | Nvarchar |
|  | CountTime | Int |
|  | ID\_MemberType | Int |

|  |  |  |
| --- | --- | --- |
| **Thuộc tính** | **Kiểu dữ liệu** | **Mô tả** |
| ID | Int | Mã phiếu mượn |
| Name | String | Tên thủ thư |
| CountTime | Int | Số lần mượn |
| ID\_MemberType | Int | Mã loại thành viên |

#### **4.2.2.9. Thực thể MemberType**

|  |  |  |
| --- | --- | --- |
| MemberType | | |
| **PK** | ID | Int |
|  | Name | Nvarchar |
|  | Discount\_percents | Int |

|  |  |  |
| --- | --- | --- |
| **Thuộc tính** | **Kiểu dữ liệu** | **Mô tả** |
| ID | Int | Mã loại thành viên |
| Name | String | Tên loại thành viên |
| Discount\_percents | Int | Phần trăm được giảm giá |

#### **4.2.2.10. Thực thể Role**

|  |  |  |
| --- | --- | --- |
| Role | | |
| **PK** | ID | Int |
|  | Name | Nvarchar |

|  |  |  |
| --- | --- | --- |
| **Thuộc tính** | **Kiểu dữ liệu** | **Mô tả** |
| ID | Int | Mã vai trò |
| Name | String | Tên vài trò |

#### **4.2.2.11. Thực thể AuthorManager**

|  |  |  |
| --- | --- | --- |
| AuthorManager | | |
| **PK** | ID\_Author | Int |
| **PK** | ID\_Book | Int |

|  |  |  |
| --- | --- | --- |
| **Thuộc tính** | **Kiểu dữ liệu** | **Mô tả** |
| ID\_Author | Int | Mã tác giả |
| ID\_Book | Int | Mã sách |

## **4.3. Phác thảo giao diện**

Logo

Description automatically generatedGraphical user interface, application

Description automatically generatedGraphical user interface, text, application, chat or text message

Description automatically generatedTable

Description automatically generatedGraphical user interface

Description automatically generatedText, email

Description automatically generatedTable

Description automatically generatedTable

Description automatically generatedGraphical user interface, application

Description automatically generatedGraphical user interface, application

Description automatically generatedGraphical user interface, application

Description automatically generatedGraphical user interface, text, application, chat or text message

Description automatically generatedPie chart

Description automatically generatedChart

Description automatically generated with low confidenceGraphical user interface, text, application, chat or text message

Description automatically generatedGraphical user interface

Description automatically generated with low confidenceGraphical user interface, text, application, chat or text message

Description automatically generated

### **4.3.1. Sơ đồ tổ chức giao diện**

Diagram

Description automatically generated

### **4.3.2. Giao diện màn hình chính**

#### **4.3.2.1. Màn hình chính vai trò User**

Graphical user interface, application

Description automatically generated

Graphical user interface, text, application

Description automatically generated

|  |  |  |  |
| --- | --- | --- | --- |
| **TT** | **@ID** | **Sự kiện** | **Mô tả hoạt động** |
| 1 | navigationView | Vuốt từ trái/ Click Menu icon | Hiển thị Navigation Drawer |
| 2 | nav\_callcard | Click | Ánh xạ layout chứa recycler view để hiển thị danh sách phiếu mượn lên layout rỗng trong activity |
| 3 | nav\_book | Click | Ánh xạ layout chứa recycler view để hiển thị danh sách sách lên layout rỗng trong activity |
| 4 | nav\_type | Click | Ánh xạ layout chứa recycler view để hiển thị danh sách thể loại sách lên layout rỗng trong activity |
| 5 | nav\_member | Click | Ánh xạ layout chứa recycler view để hiển thị danh sách thành viên lên layout rỗng trong activity |
| 6 | nav\_top10 | Click | Chuyển fragment thống kê theo lượt mượn |
| 7 | nav\_revenue\_month | Click | Chuyển fragment thống kê doanh thu theo tháng |
| 8 | nav\_revenue\_year | Click | Chuyển fragment thống kê doanh thu theo năm |
| 9 | nav\_change\_password | Click | Chuyển fragment đổi mật khẩu |
| 10 | nav\_logout | Click | Đăng xuất khỏi user |
| 11 | tvOffline |  | Thông báo đăng nhập offline |

#### **4.3.2.1. Màn hình chính vai trò Admin**

Graphical user interface, application

Description automatically generated with medium confidence

|  |  |  |  |
| --- | --- | --- | --- |
| **TT** | **@ID** | **Sự kiện** | **Mô tả hoạt động** |
| 1 | swSearch | Lắng nghe thay đổi trên thanh tìm kiếm | Lọc dữ liệu trên recycler view theo dữ liệu trên thanh tìm kiếm |
| 2 | rcvLibrarian | Long Click | Kích hoạt contextual action mode, xem danh sách nhân viên |
| 3 | nav\_book | Click | Ánh xạ layout hiển thị danh sách các sách vào layout rỗng trong activity |
| 4 | btnLogout | Click | Đăng xuất khỏi vai trò admin |
| 5 | btnNewUser | Click | Tạo nhân viên mới |

### **4.3.3. Giao diện quản lý**

#### **4.3.3.1. Màn hình quản lý thành viên**

Graphical user interface, application

Description automatically generated

|  |  |  |  |
| --- | --- | --- | --- |
| **TT** | **@ID** | **Sự kiện** | **Mô tả hoạt động** |
| 1 | swUserSearch | Lắng nghe thay đổi trên thanh tìm kiếm | Lọc dữ liệu trên recycler view theo dữ liệu trên thanh tìm kiếm |
| 2 | rcvRecyclerView | Long Click | Kích hoạt contextual action mode, xem danh sách thành viên |
| 3 | btnNewItem | Click | Tạo thành viên mới |
| 4 | tvOffline |  | Thông báo đăng nhập offline |

#### **4.3.3.2. Màn hình quản lý phiếu mượn**

Table

Description automatically generated

|  |  |  |  |
| --- | --- | --- | --- |
| **TT** | **@ID** | **Sự kiện** | **Mô tả hoạt động** |
| 1 | swUserSearch | Lắng nghe thay đổi trên thanh tìm kiếm | Lọc dữ liệu trên recycler view theo dữ liệu trên thanh tìm kiếm |
| 2 | rcvRecyclerView | Long Click | Kích hoạt contextual action mode, xem danh sách phiếu mượn |
| 3 | btnNewItem | Click | Tạo phiếu mượn mới |

#### **4.3.3.3. Màn hình quản lý sách**

Table

Description automatically generated

|  |  |  |  |
| --- | --- | --- | --- |
| **TT** | **@ID** | **Sự kiện** | **Mô tả hoạt động** |
| 1 | swUserSearch | Lắng nghe thay đổi trên thanh tìm kiếm | Lọc dữ liệu trên recycler view theo dữ liệu trên thanh tìm kiếm |
| 2 | rcvRecyclerView | Long Click | Kích hoạt contextual action mode, xem dah sách sách |
| 3 | btnNewItem | Click | Tạo sách mới |

#### **4.3.3.4. Màn hình quản lý loại sách**

Table

Description automatically generated with medium confidence

|  |  |  |  |
| --- | --- | --- | --- |
| **TT** | **@ID** | **Sự kiện** | **Mô tả hoạt động** |
| 1 | swUserSearch | Lắng nghe thay đổi trên thanh tìm kiếm | Lọc dữ liệu trên recycler view theo dữ liệu trên thanh tìm kiếm |
| 2 | rcvRecyclerView | Long Click | Kích hoạt contextual action mode, xem danh sách loại sách |
| 3 | btnNewItem | Click | Tạo loại sách mới |

#### **4.3.3.5. Màn hình thống kê theo lượt mượn của từng loại sách**

Chart, sunburst chart

Description automatically generated

Table

Description automatically generated

|  |  |  |  |
| --- | --- | --- | --- |
| **TT** | **@ID** | **Sự kiện** | **Mô tả hoạt động** |
| 1 | piechart | Click | Hiển thị tỉ lệ phần trăm theo lượt mượn của loại sách trên tổng các loại sách |
| 2 | rcvTop10 |  | Xem danh sách các sách trong top 10 |

#### **4.3.3.6. Màn hình thống kê theo doanh thu 12 tháng trong năm**

Chart

Description automatically generated

|  |  |  |  |
| --- | --- | --- | --- |
| **TT** | **@ID** | **Sự kiện** | **Mô tả hoạt động** |
| 1 | barchart |  | Hiển thị doanh thu các tháng trong 1 năm theo biểu đồ cột |

#### **4.3.3.7. Màn hình thống kê theo doanh thu từng năm trong 1 thập kỷ**

Chart, histogram

Description automatically generated

|  |  |  |  |
| --- | --- | --- | --- |
| **TT** | **@ID** | **Sự kiện** | **Mô tả hoạt động** |
| 1 | barchartYear |  | Hiển thị doanh thu các năm trong 1 thập kỷ theo biểu đồ cột |

### **4.3.4. Giao diện hỗ trợ khác**

#### **4.3.4.1. Màn hình chào**

Logo

Description automatically generated

|  |  |  |  |
| --- | --- | --- | --- |
| **TT** | **@ID** | **Sự kiện** | **Mô tả hoạt động** |
| 1 | llIntroduction |  | Giới thiệu về ứng dụng Phương Nam Book LMS |

#### **4.3.4.2. Màn hình đăng nhập**

Graphical user interface

Description automatically generated

|  |  |  |  |
| --- | --- | --- | --- |
| **TT** | **@ID** | **Sự kiện** | **Mô tả hoạt động** |
| 1 | edtUsername | Touch | Nhập tài khoản nhân viên |
| 2 | edtPassword | Touch | Nhập mật khẩu nhân viên |
| 3 | forgetPassword | Click | Hiển thị hướng dẫn khi quên mật khẩu |
| 4 | btnLogin | Click | Đăng nhập vào giao diện nhân viên theo vai trò |

# **5. Thực hiện dự án**

## **5.1. Tạo giao diện**

### **5.1.1. Các màn hình quản lý**

#### **5.1.1.1. Màn hình chính vai trò user (UserMainActivity)**

**Giao diện**

Graphical user interface, application

Description automatically generated

Graphical user interface, text, application

Description automatically generated

**Đặt ID các View**

|  |  |  |
| --- | --- | --- |
| **VIEW** | **ID** | **Các thuộc tính** |
| **Textview** | android:id="@+id/tvOffline" | android:layout\_width="match\_parent" android:layout\_height="30dp" android:background="@color/light\_ocean" android:text="Chế độ offline" android:gravity="center" android:textSize="18dp" android:visibility="gone" |
| **Drawerlayout** | android:id="@+id/drawer\_layout" | android:layout\_width="match\_parent" android:layout\_height="match\_parent" |
| **RelativeLayout** | android:id="@+id/body\_container" | android:layout\_width="match\_parent" android:layout\_height="match\_parent" android:layout\_marginTop="60dp" |
| **RelativeLayout** |  | android:layout\_width="match\_parent" android:layout\_height="match\_parent" |
| **Toolbar** | android:id="@+id/tbToolbar" | android:layout\_width="match\_parent" android:layout\_height="40dp" android:elevation="4dp" android:theme="@style/ThemeOverlay.AppCompat.ActionBar" app:popupTheme="@style/ThemeOverlay.AppCompat.Light" app:navigationIcon="@drawable/ic\_menu" app:titleTextColor="@color/black" android:layout\_margin="10dp" android:background="@drawable/shape\_toolbar" |
| **Searchview** | android:id="@+id/swUserSearch" | app:iconifiedByDefault="false" android:layout\_width="wrap\_content" app:searchIcon="@null" android:importantForContentCapture="auto" app:searchHintIcon="@drawable/ic\_person" android:layout\_height="match\_parent" android:background="@drawable/shape\_toolbar" app:titleTextColor="@color/black" |
| **Navigationview** | android:id="@+id/navigationView" | android:layout\_width="match\_parent" android:layout\_height="match\_parent" app:headerLayout="@layout/layout\_header\_drawer" app:menu="@menu/menu\_drawer" android:layout\_gravity="start" android:fitsSystemWindows="true" app:itemRippleColor="@android:color/darker\_gray" app:itemIconSize="30dp" android:layout\_marginTop="53dp" app:itemIconTint="@drawable/item\_selector" |

#### **5.1.1.1. Màn hình chính vai trò admin (AdminMainActivity)**

Graphical user interface, application

Description automatically generated with medium confidence

|  |  |  |
| --- | --- | --- |
| **VIEW** | **ID** | **Các thuộc tính** |
| **Textview** | android:id="@+id/tvOffline" | android:id="@+id/tvOffline" android:layout\_width="match\_parent" android:layout\_height="30dp" android:background="@color/light\_ocean" android:text="Chế độ offline" android:gravity="center" android:textSize="18dp" android:visibility="gone" |
| **Searchview** | android:id="@+id/swSearch" | android:layout\_width="match\_parent" android:layout\_height="40dp" android:layout\_marginTop="5dp" android:layout\_marginRight="10dp" android:background="@drawable/shape\_toolbar" android:importantForContentCapture="auto" android:layoutDirection="rtl" app:iconifiedByDefault="false" app:queryBackground="@android:color/transparent" app:titleTextColor="@color/black" |
| **Recyclerview** | android:id="@+id/rcvLibrarian" | android:layout\_width="match\_parent" android:layout\_height="wrap\_content" android:layout\_marginTop="5dp" |
| **LinearLayout** | android:id="@+id/btnNewUser" | android:layout\_width="60dp" android:layout\_height="60dp" android:layout\_marginBottom="20dp" android:gravity="center" android:background="@drawable/button" android:layout\_alignParentEnd="true" android:layout\_alignParentBottom="true" |
| **LinearLayout** | android:id="@+id/btnLogout" | android:layout\_width="50dp" android:layout\_height="40dp" android:gravity="center" android:layout\_marginTop="5dp" |

### **5.1.2. Các màn hình quản lý**

#### **5.1.2.1. Màn hình quản lý phiếu mượn**

Table

Description automatically generated

|  |  |  |
| --- | --- | --- |
| **VIEW** | **ID** | **Các thuộc tính** |
| **RecyclerView** | android:id="@+id/rcvRecyclerView" | android:layout\_width="match\_parent" android:layout\_height="wrap\_content" |
| **Searchview** | android:id="@+id/swSearch" | android:layout\_width="match\_parent" android:layout\_height="40dp" android:layout\_marginTop="5dp" android:layout\_marginRight="10dp" android:background="@drawable/shape\_toolbar" android:importantForContentCapture="auto" android:layoutDirection="rtl" app:iconifiedByDefault="false" app:queryBackground="@android:color/transparent" app:titleTextColor="@color/black" |
| **Linearlayout** | android:id="@+id/btnNewItem" | android:layout\_width="60dp" android:layout\_height="60dp" android:layout\_marginBottom="20dp" android:gravity="center" android:background="@drawable/button" android:layout\_alignParentEnd="true" android:layout\_alignParentBottom="true" |

#### **5.1.2.2. Màn hình quản lý sách**

Table

Description automatically generated

|  |  |  |
| --- | --- | --- |
| **VIEW** | **ID** | **Các thuộc tính** |
| **RecyclerView** | android:id="@+id/rcvRecyclerView" | android:layout\_width="match\_parent" android:layout\_height="wrap\_content" |
| **Searchview** | android:id="@+id/swSearch" | android:layout\_width="match\_parent" android:layout\_height="40dp" android:layout\_marginTop="5dp" android:layout\_marginRight="10dp" android:background="@drawable/shape\_toolbar" android:importantForContentCapture="auto" android:layoutDirection="rtl" app:iconifiedByDefault="false" app:queryBackground="@android:color/transparent" app:titleTextColor="@color/black" |
| **Linearlayout** | android:id="@+id/btnNewItem" | android:layout\_width="60dp" android:layout\_height="60dp" android:layout\_marginBottom="20dp" android:gravity="center" android:background="@drawable/button" android:layout\_alignParentEnd="true" android:layout\_alignParentBottom="true" |

#### **5.1.2.3. Màn hình quản lý loại sách**

Table

Description automatically generated with medium confidence

|  |  |  |
| --- | --- | --- |
| **VIEW** | **ID** | **Các thuộc tính** |
| **RecyclerView** | android:id="@+id/rcvRecyclerView" | android:layout\_width="match\_parent" android:layout\_height="wrap\_content" |
| **Searchview** | android:id="@+id/swSearch" | android:layout\_width="match\_parent" android:layout\_height="40dp" android:layout\_marginTop="5dp" android:layout\_marginRight="10dp" android:background="@drawable/shape\_toolbar" android:importantForContentCapture="auto" android:layoutDirection="rtl" app:iconifiedByDefault="false" app:queryBackground="@android:color/transparent" app:titleTextColor="@color/black" |
| **Linearlayout** | android:id="@+id/btnNewItem" | android:layout\_width="60dp" android:layout\_height="60dp" android:layout\_marginBottom="20dp" android:gravity="center" android:background="@drawable/button" android:layout\_alignParentEnd="true" android:layout\_alignParentBottom="true" |

#### **5.1.2.3. Màn hình quản lý thành viên**

Graphical user interface, application

Description automatically generated

|  |  |  |
| --- | --- | --- |
| **VIEW** | **ID** | **Các thuộc tính** |
| **RecyclerView** | android:id="@+id/rcvRecyclerView" | android:layout\_width="match\_parent" android:layout\_height="wrap\_content" |
| **Searchview** | android:id="@+id/swSearch" | android:layout\_width="match\_parent" android:layout\_height="40dp" android:layout\_marginTop="5dp" android:layout\_marginRight="10dp" android:background="@drawable/shape\_toolbar" android:importantForContentCapture="auto" android:layoutDirection="rtl" app:iconifiedByDefault="false" app:queryBackground="@android:color/transparent" app:titleTextColor="@color/black" |
| **Linearlayout** | android:id="@+id/btnNewItem" | android:layout\_width="60dp" android:layout\_height="60dp" android:layout\_marginBottom="20dp" android:gravity="center" android:background="@drawable/button" android:layout\_alignParentEnd="true" android:layout\_alignParentBottom="true" |

### **5.1.3 Màn hình thống kê theo lượt mượn**

Chart, sunburst chart

Description automatically generated

Table

Description automatically generated

|  |  |  |
| --- | --- | --- |
| **VIEW** | **ID** | **Các thuộc tính** |
| **CoordinatorLayout** |  | xmlns:tools="http://schemas.android.com/tools" android:layout\_width="match\_parent" android:layout\_height="match\_parent" xmlns:app="http://schemas.android.com/apk/res-auto" tools:context=".Fragments.TopBorrowedFragment" android:clickable="true" |
| **AppBarLayout** | android:id="@+id/appbar" | android:layout\_width="match\_parent" android:layout\_height="600dp" android:fitsSystemWindows="true" android:background="@color/light\_ocean" |
| **CollapingToolbarLayout** | android:id="@+id/collapsingToolBar" | android:layout\_width="match\_parent" android:layout\_height="match\_parent" android:fitsSystemWindows="true" app:contentScrim="?attr/colorPrimary" app:layout\_scrollFlags="scroll|snap|exitUntilCollapsed" |
| **FrameLayout** |  | android:layout\_width="match\_parent" android:layout\_height="match\_parent" |
| **TextView** | android:id="@+id/touchToShow" | android:layout\_width="match\_parent" android:layout\_height="match\_parent" android:text="Chạm vào để hiện biểu đồ" android:background="@color/light\_ocean" android:gravity="center" |
| **PieChart** | android:id="@+id/piechart" | android:layout\_width="match\_parent" android:layout\_height="match\_parent" android:text="asdasdasd" android:focusable="true" android:visibility="gone" android:focusedByDefault="true" android:focusableInTouchMode="true" android:fitsSystemWindows="true" android:layout\_margin="10dp" tools:ignore="HardcodedText" |
| **CoordinatorLayout** |  | android:layout\_width="match\_parent" android:layout\_height="match\_parent" android:background="@color/light\_ocean" app:layout\_behavior="com.google.android.material.appbar.AppBarLayout$ScrollingViewBehavior" |
| **Include** |  | layout="@layout/layout\_top10" |

### **5.1.4. Màn hình thống kê doanh thu theo tháng trong 1 năm**

Chart

Description automatically generated

|  |  |  |
| --- | --- | --- |
| **VIEW** | **ID** | **Các thuộc tính** |
| **Barchart** | android:id="@+id/barchart" | android:layout\_width="match\_parent" android:layout\_height="match\_parent" android:layout\_margin="20dp" |

### **5.1.5. Màn hình thống kê doanh thu theo năm trong 1 thập kỷ**

Chart, histogram

Description automatically generated

|  |  |  |
| --- | --- | --- |
| **VIEW** | **ID** | **Các thuộc tính** |
| **Barchart** | android:id="@+id/barchartYear" | android:layout\_width="match\_parent" android:layout\_height="match\_parent" android:layout\_margin="20dp" |

### **5.1.6. Các màn hình hỗ trợ khác**

#### **5.1.6.1. Màn hình chào**

Logo

Description automatically generated

|  |  |  |
| --- | --- | --- |
| **VIEW** | **ID** | **Các thuộc tính** |
| **LinearLayout** | android:id="@+id/llIntroduction" | android:layout\_width="match\_parent" android:layout\_height="match\_parent" android:visibility="visible" |

#### **5.1.6.1. Màn hình đăng nhập**

Graphical user interface

Description automatically generated

|  |  |  |
| --- | --- | --- |
| **VIEW** | **ID** | **Các thuộc tính** |
| **ScrollView** |  | android:layout\_width="wrap\_content" android:layout\_height="wrap\_content" |
| **FrameLayout** |  | android:layout\_width="match\_parent" android:layout\_height="match\_parent" |
| **LinearLayout** |  | android:layout\_width="wrap\_content" android:layout\_height="wrap\_content" android:orientation="vertical" android:gravity="center\_horizontal" |
| **ImageView** |  | android:layout\_width="350dp" android:layout\_height="180dp" android:src="@mipmap/librarian" |
| **TextView** |  | android:layout\_width="wrap\_content" android:layout\_height="wrap\_content" android:textSize="30dp" android:textColor="@color/light\_ocean" android:text="Đăng nhập" android:textStyle="bold" android:layout\_marginBottom="20dp" |
| **Include** |  | layout="@layout/layout\_login\_form" |
| **TextView** |  | android:id="@+id/forgetPassword" android:layout\_width="wrap\_content" android:layout\_height="wrap\_content" android:padding="10dp" android:textSize="18dp" android:text="Quên mật khẩu?" android:textColor="@color/light\_ocean" |
| **TextView** | android:id="@+id/btnLogin" | android:layout\_width="200dp" android:layout\_height="50dp" android:text="ĐĂNG NHẬP" android:textColor="@color/light\_ocean" android:textSize="20dp" android:clickable="true" android:gravity="center" android:background="@drawable/button" |
| **ProgressBar** | android:id="@+id/progressbar" | android:visibility="gone" android:layout\_width="wrap\_content" android:layout\_height="wrap\_content" android:layout\_marginTop="10dp" |

## **5.2. Tạo CSDL với SQLITE và SQLSEVER**

### **5.2.1. Sơ đồ quan hệ**

Diagram, engineering drawing

Description automatically generated

Diagram

Description automatically generated

### **5.2.2. Chi tiết các bảng**

Gán tên các bảng, các thuộc tính trong bảng bằng các biến static

private static final String *TABLE\_AUTHOR*="Author";  
private static final String *KEY\_AUTHOR\_ID*="ID\_Author";  
private static final String *KEY\_AUTHOR\_NAME*="Name\_Author";  
private static final String *KEY\_AUTHOR\_DOB*="Dob\_Author";  
  
private static final String *TABLE\_KIND* = "Kind";  
private static final String *KEY\_KIND\_ID* = "ID\_Kind";  
private static final String *KEY\_KIND\_NAME* = "Name\_Kind";  
  
private static final String *TABLE\_LANGUAGE* = "Language";  
private static final String *KEY\_LANGUAGE\_ID* = "ID\_Language";  
private static final String *KEY\_LANGUAGE\_NAME* = "Name\_Language";  
  
private static final String *TABLE\_BOOK* = "Book";  
private static final String *KEY\_BOOK\_ID* = "ID\_Book";  
private static final String *KEY\_BOOK\_NAME* = "Name\_Book";  
private static final String *KEY\_BOOK\_PRICE* = "Price";  
private static final String *KEY\_BOOK\_QUANTITY* = "CopiesQuantity";  
  
private static final String *TABLE\_PUBLISHER* = "Publisher";  
private static final String *KEY\_PUBLISHER\_ID* = "ID\_Publisher";  
private static final String *KEY\_PUBLISHER\_NAME* = "Name\_Publisher";  
private static final String *KEY\_PUBLISHER\_LOCATION* = "Location\_Publisher";  
  
private static final String *TABLE\_CALLCARD* = "CallCard";  
private static final String *KEY\_CALLCARD\_ID* = "ID\_CallCard";  
private static final String *KEY\_CALLCARD\_BEGINDATE* = "BeginDate";  
private static final String *KEY\_CALLCARD\_EXPIRESDATE* = "ExpiresDate";  
  
private static final String *TABLE\_LIBRARIAN* = "Librarian";  
private static final String *KEY\_LIBRARIAN\_ID* = "ID\_Librarian";  
private static final String *KEY\_LIBRARIAN\_NAME* = "Name\_Librarian";  
private static final String *KEY\_LIBRARIAN\_PASSWORD* = "Password\_Librarian";  
  
private static final String *TABLE\_MEMBER* = "Member";  
private static final String *KEY\_MEMBER\_ID* = "ID\_Member";  
private static final String *KEY\_MEMBER\_NAME* = "Name\_Member";  
private static final String *KEY\_MEMBER\_COUNTTIME* = "CountTime";  
  
private static final String *TABLE\_ROLE* = "Role";  
private static final String *KEY\_ROLE\_ID* = "ID\_Role";  
private static final String *KEY\_ROLE\_NAME* = "Name\_Role";  
  
private static final String *TABLE\_MEMBERTYPE* = "MemberType";  
private static final String *KEY\_MEMBERTYPE\_ID* = "ID\_MemberType";  
private static final String *KEY\_MEMBERTYPE\_NAME* = "Name\_MemberType";  
private static final String *KEY\_MEMBERTYPE\_DISCOUNT* = "Discount\_MemberType";  
  
private static final String *TABLE\_AUTHORMANAGER* = "AuthorManager";

#### **5.2.2.1. Bảng Author**

**Cấu trúc bảng**

|  |  |  |  |
| --- | --- | --- | --- |
| **Tên cột** | **Kiểu dữ liệu** | **Ràng buộc** | **Mô tả** |
| ID | INT | PK, NOT NULL, AUTO INCREMENT | Mã tác giả |
| Name | TEXT | NOT NULL | Tên tác giả |
| DOB | DATE | NOT NULL | Ngày sinh tác giả |

**Lệnh tạo bảng SQlite**

String sqlAuthor="create table "+*TABLE\_AUTHOR*+"("  
 +*KEY\_AUTHOR\_ID*+" integer primary key autoincrement "+","  
 +*KEY\_AUTHOR\_NAME*+" text" +","  
 +*KEY\_AUTHOR\_DOB*+" text" +")";  
db.execSQL(sqlAuthor);

**Lệnh tạo bảng SQLSever**

CREATE TABLE Author

(

ID INT NOT NULL IDENTITY(1,1),

Name NVARCHAR(255) NOT NULL,

DOB DATE NOT NULL,

PRIMARY KEY (ID)

);

**Câu lệnh SQLite thêm hàng vào bảng**

db.insert(PhuongNamLibSQLite.*getTableAuthor*(),null,values);

#### **5.2.2.2. Bảng Kind**

**Cấu trúc bảng**

|  |  |  |  |
| --- | --- | --- | --- |
| **Tên cột** | **Kiểu dữ liệu** | **Ràng buộc** | **Mô tả** |
| ID | INT | PK,NOT NULL, AUTO INCREMENT | Mã Loại sách |
| Name | TEXT | NOT NULL | Tên loại sách |

**Lệnh tạo bảng SQlite**

String sqlPublisher="create table "+*TABLE\_PUBLISHER*+"("  
String sqlKind="create table "+*TABLE\_KIND*+"("  
 +*KEY\_KIND\_ID*+" integer primary key autoincrement "+","  
 +*KEY\_KIND\_NAME*+" text"+")";  
db.execSQL(sqlKind);

**Lệnh tạo bảng SQLSever**

CREATE TABLE Kind

(

ID INT NOT NULL IDENTITY(1,1),

Name NVARCHAR NOT NULL,

PRIMARY KEY (ID)

);

**Câu lệnh truy vấn SQLite**

Thêm một hàng

db.insert(PhuongNamLibSQLite.*getTableKind*(),null,values);

Lấy số lượng loại sách hiện có

select count("+PhuongNamLibSQLite.*getKeyKindId*()+") "  
 +"from "+PhuongNamLibSQLite.*getTableKind*()

Lấy tất cả cột trong bảng

select \*from "+ PhuongNamLibSQLite.*getTableKind*()

Xóa một hàng trong bảng loại sách

db.delete(PhuongNamLibSQLite.*getTableKind*()  
 ,PhuongNamLibSQLite.*getKeyKindId*()+"=?"  
 ,new String[]{kinds.get(position).getId()+""});

Cập nhật một hàng trong bảng loại sách

db.update(PhuongNamLibSQLite.*getTableKind*(),values  
 ,PhuongNamLibSQLite.*getKeyKindId*()+"=?"  
 ,new String[]{kind.getId()+""});

**Lập trình cho controller quản lý loại sách trên asp.net core**

namespace PhuongNamLib.Controllers

{

[Route("/api/")]

[ApiController]

public class KindController : ControllerBase

{

private readonly PNLibEntities db;

public KindController(PNLibEntities context)

{

db = context;

}

[Route("all-kind")]

public IActionResult GetAllKind()

{

List<Kind> list = db.Kinds.ToList();

return Ok(list);

}

[Authorize]

[HttpPost]

[Route("add-kind")]

public IActionResult AddKind(Kind Kind)

{

try

{

if (!ModelState.IsValid)

{

return BadRequest();

}

db.Kinds.Add(Kind);

db.SaveChanges();

}

catch (Exception)

{

return BadRequest();

}

return Ok(new Message("Okay"));

}

[Authorize]

[HttpPost]

[Route("delete-kind")]

public IActionResult DeleteKind(Kind Kind)

{

try

{

if (!ModelState.IsValid)

{

return BadRequest();

}

db.Kinds.Remove(Kind);

db.SaveChanges();

}

catch (Exception)

{

return BadRequest();

}

return Ok(new Message("Okay"));

}

[Authorize]

[HttpPost]

[Route("update-kind")]

public IActionResult UpdateKind(Kind Kind)

{

Kind kind = db.Kinds.Single(x => x.Id == Kind.Id);

if (kind != null)

{

kind.Name = Kind.Name;

db.SaveChanges();

return Ok(new Message("Okay"));

}

else

return BadRequest();

}

}

}

#### **5.2.2.3. Bảng Publisher**

Cấu trúc bảng

|  |  |  |  |
| --- | --- | --- | --- |
| **Tên cột** | **Kiểu dữ liệu** | **Ràng buộc** | **Mô tả** |
| ID | INT | PK, AUTO INCREMENT | Mã NXB |
| Name | TEXT | NOT NULL | Tên NXB |
| Location | TEXT | NOT NULL | Địa chỉ NXB |

**Lệnh tạo bảng SQlite**

String sqlPublisher="create table "+*TABLE\_PUBLISHER*+"("  
 +*KEY\_PUBLISHER\_ID*+" integer primary key autoincrement "+","  
 +*KEY\_PUBLISHER\_NAME*+" text" +","  
 +*KEY\_PUBLISHER\_LOCATION*+" text" +")";  
db.execSQL(sqlPublisher);

**Lệnh tạo bảng SQLSever**

CREATE TABLE Publisher

(

ID INT NOT NULL IDENTITY(1,1),

Name NVARCHAR(255) NOT NULL,

Location NVARCHAR(255) NOT NULL,

PRIMARY KEY (ID)

);

**Câu lệnh SQLite thêm hàng vào bảng**

db.insert(PhuongNamLibSQLite.*getTablePublisher*(),null,values);

#### **5.2.2.4. Bảng Role**

Cấu trúc bảng

|  |  |  |  |
| --- | --- | --- | --- |
| **Tên cột** | **Kiểu dữ liệu** | **Ràng buộc** | **Mô tả** |
| ID | INT | PK, AUTO INCREMENT | Mã vai trò |
| Name | TEXT | NOT NULL | Tên vai trò |

**Lệnh tạo bảng SQlite**

String sqlRole="create table "+*TABLE\_ROLE*+"("  
 +*KEY\_ROLE\_ID*+" integer primary key autoincrement "+","  
 +*KEY\_ROLE\_NAME*+" text"+")";  
db.execSQL(sqlRole);

**Lệnh tạo bảng SQLSever**

CREATE TABLE Role

(

ID INT NOT NULL IDENTITY(1,1),

Name NVARCHAR(255) NOT NULL,

PRIMARY KEY (ID)

);

**Câu lệnh SQLite thêm hàng vào bảng**

db.insert(PhuongNamLibSQLite.*getRole*(),null,values);

##### **5.2.2.5. Bảng Language**

Cấu trúc bảng

|  |  |  |  |
| --- | --- | --- | --- |
| **Tên cột** | **Kiểu dữ liệu** | **Ràng buộc** | **Mô tả** |
| ID | INT | PK, NOT NULL,AUTO INCREMENT | Mã ngôn ngữ |
| Name | TEXT | NOT NULL | Tên ngôn ngữ |

**Lệnh tạo bảng SQlite**

String sqlLanguage="create table "+*TABLE\_LANGUAGE*+"("  
 +*KEY\_LANGUAGE\_ID*+" integer primary key autoincrement "+","  
 +*KEY\_LANGUAGE\_NAME*+" text"+")";  
db.execSQL(sqlLanguage);

**Lệnh tạo bảng SQLSever**

CREATE TABLE Language

(

ID INT NOT NULL IDENTITY(1,1),

Name NVARCHAR(255) NOT NULL,

PRIMARY KEY (ID)

);

**Câu lệnh SQLite thêm hàng vào bảng**

db.insert(PhuongNamLibSQLite.*getLanguage*(),null,values);

#### **5.2.2.6. Bảng MemberType**

Cấu trúc bảng

|  |  |  |  |
| --- | --- | --- | --- |
| **Tên cột** | **Kiểu dữ liệu** | **Ràng buộc** | **Mô tả** |
| ID | INT | PK, NOT NULL,AUTO INCREMENT | Mã loại thành viên |
| Name | TEXT | NOT NULL | Tên loại thành viên |
| Discount\_percents | INT | NOT NULL | Phần trăm giảm giá |

**Lệnh tạo bảng SQlite**

String sqlMemberType="create table "+*TABLE\_MEMBERTYPE*+"("  
 +*KEY\_MEMBERTYPE\_ID*+" integer primary key autoincrement "+","  
 +*KEY\_MEMBERTYPE\_NAME*+" text"+","  
 +*KEY\_MEMBERTYPE\_DISCOUNT*+" integer"+")";  
db.execSQL(sqlMemberType);

**Lệnh tạo bảng SQLSever**

CREATE TABLE MemberType

(

ID INT NOT NULL IDENTITY(1,1),

Name NVARCHAR(255) NOT NULL,

Discount\_percents INT NOT NULL,

PRIMARY KEY (ID)

);

**Câu lệnh SQLite thêm hàng vào bảng**

db.insert(PhuongNamLibSQLite.*getMemberType*(),null,values);

#### **5.2.2.7. Bảng Book**

**Cấu trúc bảng**

|  |  |  |  |
| --- | --- | --- | --- |
| **Tên cột** | **Kiểu dữ liệu** | **Ràng buộc** | **Mô tả** |
| ID | INT | PK, NOT NULL, AUTO INCREMENT | Mã sách |
| Name | TEXT | NOT NULL | Tên sách |
| Price | INT | NOT NULL | Giá thuê |
| CopiesQuantity | INT | NOT NULL | Số lượng bản sao |
| ID\_Kind | INT | NOT NULL | Mã loại sách |
| ID\_Publisher | INT | NOT NULL | Mã nhà xuất bản |
| ID\_language | INT | NOT NULL | Mã ngôn ngữ |

**Lệnh tạo bảng SQlite**

String sqlBook="create table "+*TABLE\_BOOK*+"("  
 +*KEY\_BOOK\_ID*+" integer primary key autoincrement "+","  
 +*KEY\_BOOK\_NAME*+" text" +","  
 +*KEY\_BOOK\_PRICE*+" integer" +","  
 +*KEY\_BOOK\_QUANTITY*+" integer" +","  
 +*KEY\_KIND\_ID*+" integer" +","  
 +*KEY\_PUBLISHER\_ID*+" integer" +","  
 +*KEY\_LANGUAGE\_ID*+" integer"+")";  
db.execSQL(sqlBook);

**Lệnh tạo bảng SQLSever**

CREATE TABLE Book

(

ID INT NOT NULL IDENTITY(1,1),

Name VARCHAR NOT NULL,

Price INT NOT NULL,

CopiesQuantity INT NOT NULL,

ID\_Kind INT NOT NULL,

ID\_Publisher INT NOT NULL,

ID\_Language INT NOT NULL,

PRIMARY KEY (ID),

FOREIGN KEY (ID\_Kind) REFERENCES Kind(ID),

FOREIGN KEY (ID\_Publisher) REFERENCES Publisher(ID),

FOREIGN KEY (ID\_Language) REFERENCES Language(ID)

);

KEY (ID)

);

**Câu lệnh truy vấn SQLite**

Thêm một hàng

db.insert(PhuongNamLibSQLite.*getTableBook*(),null,values);

Lấy số lượng loại sách hiện có

select count("+PhuongNamLibSQLite.*getKeyBookId*()+") "  
 +"from "+PhuongNamLibSQLite.*getTableBook*()

Lấy tất cả cột trong bảng

select \*from "+ PhuongNamLibSQLite.*getTableBook*()

Xóa một hàng trong bảng sách

db.delete(PhuongNamLibSQLite.*getTableBook*()  
 ,PhuongNamLibSQLite.*getKeyBookId*()+"=?"  
 ,new String[]{alBooks.get(position).getId()+""});

Cập nhật một hàng trong bảng sách

db.update(PhuongNamLibSQLite.*getTableBook*(),values  
 ,PhuongNamLibSQLite.*getKeyBookId*()+"=?"  
 ,new String[]{book.getId()+""});

Join với bảng Kind và bảng Publisher

"select "+PhuongNamLibSQLite.*getKeyBookId*()+","  
 +PhuongNamLibSQLite.*getKeyBookName*()+","  
 +PhuongNamLibSQLite.*getKeyBookPrice*()+","  
 +PhuongNamLibSQLite.*getKeyBookQuantity*()+","  
 +PhuongNamLibSQLite.*getKeyKindName*()+","  
 +PhuongNamLibSQLite.*getKeyPublisherName*()+","  
 +PhuongNamLibSQLite.*getKeyLanguageName*()  
 +" from "+ PhuongNamLibSQLite.*getTableBook*()  
 +" inner join "  
 +PhuongNamLibSQLite.*getTableKind*()  
 +" on "  
 +PhuongNamLibSQLite.*getTableBook*()+"."+PhuongNamLibSQLite.*getKeyKindId*()  
 +" = "  
 +PhuongNamLibSQLite.*getTableKind*()+"."+PhuongNamLibSQLite.*getKeyKindId*()  
 +" inner join "  
 +PhuongNamLibSQLite.*getTablePublisher*()  
 +" on "  
 +PhuongNamLibSQLite.*getTableBook*()+"."+PhuongNamLibSQLite.*getKeyPublisherId*()  
 +" = "  
 +PhuongNamLibSQLite.*getTablePublisher*()+"."+PhuongNamLibSQLite.*getKeyPublisherId*()  
 +" inner join "  
 +PhuongNamLibSQLite.*getTableLanguage*()  
 +" on "  
 +PhuongNamLibSQLite.*getTableBook*()+"."+PhuongNamLibSQLite.*getKeyLanguageId*()  
 +" = "  
 +PhuongNamLibSQLite.*getTableLanguage*()+"."+PhuongNamLibSQLite.*getKeyLanguageId*()

**Lập trình cho controller quản lý sách trên asp.net core**

namespace PhuongNamLib.Controllers

{

[Route("/api/")]

[ApiController]

public class BookController : ControllerBase

{

private readonly PNLibEntities db;

public BookController(PNLibEntities context)

{

db = context;

}

[Route("all-book")]

public IActionResult GetAllBook()

{

List<Book> list = db.Books.ToList();

return Ok(list);

}

[Authorize]

[HttpPost]

[Route("add-book")]

public IActionResult AddBook(Book Book)

{

try

{

if (!ModelState.IsValid)

{

return BadRequest();

}

db.Books.Add(Book);

db.SaveChanges();

}

catch (Exception)

{

return BadRequest();

}

return Ok(new Message("Okay"));

}

[Authorize]

[HttpPost]

[Route("delete-book")]

public IActionResult DeleteBook(Book Book)

{

try

{

if (!ModelState.IsValid)

{

return BadRequest();

}

db.Books.Remove(Book);

db.SaveChanges();

}

catch (Exception)

{

return BadRequest();

}

return Ok(new Message("Okay"));

}

[Authorize]

[HttpPost]

[Route("update-book")]

public IActionResult UpdateBook(Book Book)

{

Book book = db.Books.Single(x => x.Id == Book.Id);

if (book != null)

{

book.Name = Book.Name;

book.IdKind = Book.IdKind;

book.Price = Book.Price;

book.IdPublisher = Book.IdPublisher;

book.IdLanguage = Book.IdLanguage;

db.SaveChanges();

return Ok(new Message("Okay"));

}

else

return BadRequest();

}

}

}

#### **5.2.2.8. Bảng Member**

**Cấu trúc bảng**

|  |  |  |  |
| --- | --- | --- | --- |
| **Tên cột** | **Kiểu dữ liệu** | **Ràng buộc** | **Mô tả** |
| ID | INT | PK, NOT NULL, AUTO INCREMENT | Mã thành viên |
| Name | TEXT | NOT NULL | Tên thành viên |
| CountTime | INT | NOT NULL | Lượt mượn sách |
| ID\_MemberType | INT | NOT NULL | Mã loại thành viên |

**Lệnh tạo bảng SQlite**

String sqlMember="create table "+*TABLE\_MEMBER*+"("  
 +*KEY\_MEMBER\_ID*+" integer primary key autoincrement "+","  
 +*KEY\_MEMBER\_NAME*+" text" +","  
 +*KEY\_MEMBER\_COUNTTIME*+" integer"+","  
 +*KEY\_MEMBERTYPE\_ID*+" text" +")";  
db.execSQL(sqlMember);

**Lệnh tạo bảng SQLSever**

CREATE TABLE Member

(

ID INT NOT NULL IDENTITY(1,1),

Name VARCHAR(255) NOT NULL,

CountTime INT NOT NULL,

ID\_MemberType INT NOT NULL,

PRIMARY KEY (ID),

FOREIGN KEY (ID\_MemberType) REFERENCES MemberType(ID)

);

**Câu lệnh truy vấn SQLite**

Thêm một hàng

db.insert(PhuongNamLibSQLite.*getTableMember*(),null,values);

Lấy số lượng thành viên hiện có

"select count("+PhuongNamLibSQLite.*getKeyMemberId*()+") "  
 +"from "+PhuongNamLibSQLite.*getTableMember*()

Lấy tất cả cột trong bảng

"select \*from "+ PhuongNamLibSQLite.*getTableMember*()

Xóa một hàng trong bảng thành viên

db.delete(PhuongNamLibSQLite.*getTableMember*()  
 ,PhuongNamLibSQLite.*getKeyMemberId*()+"=?"  
 ,new String[]{members.get(position).getId()+""});

Cập nhật một hàng trong bảng thành viên

db.update(PhuongNamLibSQLite.*getTableMember*(),values  
 ,PhuongNamLibSQLite.*getKeyMemberId*()+"=?"  
 ,new String[]{member.getId()+""});

Join với bảng MemberType

"select "+PhuongNamLibSQLite.*getKeyMemberId*()+","  
 +PhuongNamLibSQLite.*getKeyMemberName*()+","  
 +PhuongNamLibSQLite.*getKeyMemberCounttime*()+","  
 +PhuongNamLibSQLite.*getKeyMembertypeName*()  
 +" from "+ PhuongNamLibSQLite.*getTableMember*()  
 +" inner join "+ PhuongNamLibSQLite.*getTableMembertype*()  
 +" on "+PhuongNamLibSQLite.*getTableMember*()+"."+PhuongNamLibSQLite.*getKeyMembertypeId*()  
 +" = "+PhuongNamLibSQLite.*getTableMembertype*()+"."+PhuongNamLibSQLite.*getKeyMembertypeId*()

+PhuongNamLibSQLite.*getTableLanguage*()+"."+PhuongNamLibSQLite.*getKeyLanguageId*()

**Lập trình cho controller quản lý thành viên trên asp.net core**

public class MemberController : ControllerBase

{

private readonly PNLibEntities db;

public MemberController(PNLibEntities context)

{

db = context;

}

[Route("all-member")]

public IActionResult GetAllMember()

{

List<Member> list = db.Members.ToList();

return Ok(list);

}

[Authorize]

[HttpPost]

[Route("add-member")]

public IActionResult AddMember(Member Member)

{

try

{

if (!ModelState.IsValid)

{

return BadRequest();

}

db.Members.Add(Member);

db.SaveChanges();

}

catch (Exception)

{

return BadRequest();

}

return Ok(new Message("Okay"));

}

[Authorize]

[HttpPost]

[Route("delete-member")]

public IActionResult DeleteMember(Member Member)

{

try

{

if (!ModelState.IsValid)

{

return BadRequest();

}

db.Members.Remove(Member);

db.SaveChanges();

}

catch (Exception)

{

return BadRequest();

}

return Ok(new Message("Okay"));

}

[Authorize]

[HttpPost]

[Route("update-member")]

public IActionResult UpdateMember(Member Member)

{

Member mem = db.Members.Single(x => x.Id == Member.Id);

if (mem != null)

{

mem.Name = Member.Name;

mem.CountTime = Member.CountTime;

db.SaveChanges();

return Ok(new Message("Okay"));

}

else

return BadRequest();

}

}

#### **5.2.2.9. Bảng Librarian**

**Cấu trúc bảng**

|  |  |  |  |
| --- | --- | --- | --- |
| **Tên cột** | **Kiểu dữ liệu** | **Ràng buộc** | **Mô tả** |
| ID | INT | PK, NOT NULL, AUTO INCREMENT | Mã nhân viên |
| Name | TEXT | NOT NULL | Tên nhân viên |
| Password | TEXT | NOT NULL | Mật khẩu nhân viên |

**Lệnh tạo bảng SQlite**

String sqlLibrarian="create table "+*TABLE\_LIBRARIAN*+"("  
 +*KEY\_LIBRARIAN\_ID*+" text primary key "+","  
 +*KEY\_LIBRARIAN\_NAME*+" text" +","  
 +*KEY\_LIBRARIAN\_PASSWORD*+" text"+","  
 +*KEY\_ROLE\_ID*+" integer" +")";  
db.execSQL(sqlLibrarian);

**Lệnh tạo bảng SQLSever**

CREATE TABLE Librarian

(

ID VARCHAR NOT NULL,

Name VARCHAR NOT NULL,

Password VARCHAR NOT NULL,

ID\_Role INT NOT NULL,

PRIMARY KEY (ID),

FOREIGN KEY (ID\_Role) REFERENCES Role(ID)

);

**Câu lệnh truy vấn SQLite**

Thêm một hàng

db.insert(PhuongNamLibSQLite.*getTableLibrarian*(),null,values);

Lấy tất cả cột trong bảng

"select \*from "+ PhuongNamLibSQLite.*getTableLibrarian*()

Xóa một hàng trong bảng nhân viên

db.delete(PhuongNamLibSQLite.*getTableLibrarian*()  
 ,PhuongNamLibSQLite.*getKeyLibrarianId*()+"=?"  
 ,new String[]{librarians.get(position).getId()});

Cập nhật một hàng trong bảng nhân viên

db.update(PhuongNamLibSQLite.*getTableLibrarian*(),values  
 ,PhuongNamLibSQLite.*getKeyLibrarianId*()+"=?"  
 ,new String[]{librarian.getId()});

**Lập trình cho controller quản lý nhân viên trên asp.net core**

namespace PhuongNamLib.Controllers

{

[Route("/api/")]

[ApiController]

public class LibrarianController : ControllerBase

{

private readonly PNLibEntities db;

public LibrarianController(PNLibEntities context)

{

db = context;

}

[Authorize]

[HttpGet]

[Route("all-librarian")]

public IActionResult GetAllLibrarian()

{

List<Librarian> list = db.Librarians.ToList();

return Ok(list);

}

[Authorize]

[HttpPost]

[Route("add-librarian")]

public IActionResult AddLibrarian(Librarian librarian)

{

try

{

if (!ModelState.IsValid)

{

return BadRequest();

}

db.Librarians.Add(librarian);

db.SaveChanges();

}

catch (Exception)

{

return BadRequest();

}

return Ok(new Message("Okay"));

}

[Authorize]

[HttpPost]

[Route("delete-librarian")]

public IActionResult DeleteLibrarian(Librarian librarian)

{

try

{

if (!ModelState.IsValid)

{

return BadRequest();

}

db.Librarians.Remove(librarian);

db.SaveChanges();

}

catch (Exception)

{

return BadRequest();

}

return Ok(new Message("Okay"));

}

[Authorize]

[HttpPost]

[Route("update-librarian")]

public IActionResult UpdateLibrarian(Librarian librarian)

{

Librarian lib = db.Librarians.Single(x => x.Id.ToLower() == librarian.Id.ToLower());

if(lib != null)

{

lib.Name = librarian.Name;

lib.Password = librarian.Password;

lib.IdRole = librarian.IdRole;

db.SaveChanges();

return Ok(new Message("Okay"));

}

else

return BadRequest();

}

}

}

#### **5.2.2.10. Bảng CallCard**

**Cấu trúc bảng**

|  |  |  |  |
| --- | --- | --- | --- |
| **Tên cột** | **Kiểu dữ liệu** | **Ràng buộc** | **Mô tả** |
| ID | INT | PK, NOT NULL, AUTO INCREMENT | Mã phiếu mượn |
| BeginDate | DATE | NOT NULL | Ngày mượn |
| ExpiresDate | DATE | NOT NULL | Ngày hết hạn |
| ID\_Book | INT | NOT NULL | Mã sách |
| ID\_Member | INT | NOT NULL | Lượt thành viên mượn |
| ID\_Librarian | INT | NOT NULL | Mã nhân viên tạo phiếu |

**Lệnh tạo bảng SQlite**

String sqlCallCard="create table "+*TABLE\_CALLCARD*+"("  
 +*KEY\_CALLCARD\_ID*+" integer primary key autoincrement "+","  
 +*KEY\_CALLCARD\_BEGINDATE*+" text" +","  
 +*KEY\_CALLCARD\_EXPIRESDATE*+" text" +","  
 +*KEY\_BOOK\_ID*+" integer" +","  
 +*KEY\_MEMBER\_ID*+" integer" +","  
 +*KEY\_LIBRARIAN\_ID*+" text" +")";  
db.execSQL(sqlCallCard);

**Lệnh tạo bảng SQLSever**

CREATE TABLE CallCard

(

ID INT NOT NULL IDENTITY(1,1),

BeginDate DATE NOT NULL,

ExpiresDate DATE NOT NULL,

ID\_Book INT NOT NULL,

ID\_Member INT NOT NULL,

ID\_Librarian VARCHAR NOT NULL,

PRIMARY KEY (ID),

FOREIGN KEY (ID\_Book) REFERENCES Book(ID),

FOREIGN KEY (ID\_Member) REFERENCES Member(ID),

FOREIGN KEY (ID\_Librarian) REFERENCES Librarian(ID)

);

**Câu lệnh truy vấn SQLite**

Thêm một hàng

db.insert(PhuongNamLibSQLite.*getTableCallcard*(),null,values);

Lấy số lượng phiếu mượn hiện có

"select count("+PhuongNamLibSQLite.*getKeyCallcardId*()+") "  
 +"from "+PhuongNamLibSQLite.*getTableCallcard*()

Lấy tất cả cột trong bảng

"select \*from "+ PhuongNamLibSQLite.*getTableCallcard*()

Xóa một hàng trong bảng phiếu mượn

db.delete(PhuongNamLibSQLite.*getTableCallcard*()  
 ,PhuongNamLibSQLite.*getKeyCallcardId*()+"=?"  
 ,new String[]{alCallcards.get(position).getId()+""});

Cập nhật một hàng trong bảng phiếu mượn

db.update(PhuongNamLibSQLite.*getTableCallcard*(),values  
 ,PhuongNamLibSQLite.*getKeyCallcardId*()+"=?"  
 ,new String[]{callcard.getId()+""});

Join với bảng Member và Book

"select "+PhuongNamLibSQLite.*getKeyCallcardId*()+","  
 +PhuongNamLibSQLite.*getKeyCallcardBegindate*()+","  
 +PhuongNamLibSQLite.*getKeyCallcardExpiresdate*()+","  
 +PhuongNamLibSQLite.*getKeyBookName*()+","  
 +PhuongNamLibSQLite.*getKeyMemberName*()+","  
 +PhuongNamLibSQLite.*getKeyLibrarianId*()  
 +" from "+ PhuongNamLibSQLite.*getTableCallcard*()  
 +" inner join "  
 +PhuongNamLibSQLite.*getTableBook*()  
 +" on "  
 +PhuongNamLibSQLite.*getTableCallcard*()+"."+PhuongNamLibSQLite.*getKeyBookId*()  
 +" = "  
 +PhuongNamLibSQLite.*getTableBook*()+"."+PhuongNamLibSQLite.*getKeyBookId*()  
 +" inner join "  
 +PhuongNamLibSQLite.*getTableMember*()  
 +" on "  
 +PhuongNamLibSQLite.*getTableCallcard*()+"."+PhuongNamLibSQLite.*getKeyMemberId*()  
 +" = "  
 +PhuongNamLibSQLite.*getTableMember*()+"."+PhuongNamLibSQLite.*getKeyMemberId*()

**Lập trình cho controller quản lý phiếu mượn trên asp.net core**

namespace PhuongNamcal.Controllers

{

[Route("/api/")]

[ApiController]

public class CallCardController : ControllerBase

{

private readonly PNLibEntities db;

public CallCardController(PNLibEntities context)

{

db = context;

}

[Route("all-callcard")]

public IActionResult GetAllcallcard()

{

List<CallCard> list = db.CallCards.ToList();

return Ok(list);

}

[Authorize]

[HttpPost]

[Route("add-callcard")]

public IActionResult Addcallcard(CallCard CallCard)

{

try

{

if (!ModelState.IsValid)

{

return BadRequest();

}

db.CallCards.Add(CallCard);

db.SaveChanges();

}

catch (Exception)

{

return BadRequest();

}

return Ok(new Message("Okay"));

}

[Authorize]

[HttpPost]

[Route("delete-callcard")]

public IActionResult Deletecallcard(CallCard CallCard)

{

try

{

if (!ModelState.IsValid)

{

return BadRequest();

}

db.CallCards.Remove(CallCard);

db.SaveChanges();

}

catch (Exception)

{

return BadRequest();

}

return Ok(new Message("Okay"));

}

[Authorize]

[HttpPost]

[Route("update-callcard")]

public IActionResult Updatecallcard(CallCard CallCard)

{

CallCard cal = db.CallCards.Single(x => x.Id == CallCard.Id);

if (cal != null)

{

cal.BeginDate = CallCard.BeginDate;

cal.ExpiresDate = CallCard.ExpiresDate;

db.SaveChanges();

return Ok(new Message("Okay"));

}

else

return BadRequest();

}

}

}

#### **5.2.2.11. Bảng Role**

Cấu trúc bảng

|  |  |  |  |
| --- | --- | --- | --- |
| **Tên cột** | **Kiểu dữ liệu** | **Ràng buộc** | **Mô tả** |
| ID | INT | PK, AUTO INCREMENT | Mã vai trò |
| Name | TEXT | NOT NULL | Tên vai trò |

**Lệnh tạo bảng SQlite**

String sqlRole="create table "+*TABLE\_ROLE*+"("  
 +*KEY\_ROLE\_ID*+" integer primary key autoincrement "+","  
 +*KEY\_ROLE\_NAME*+" text"+")";  
db.execSQL(sqlRole);

**Lệnh tạo bảng SQLSever**

CREATE TABLE Role

(

ID INT NOT NULL IDENTITY(1,1),

Name NVARCHAR(255) NOT NULL,

PRIMARY KEY (ID)

);

**Câu lệnh SQLite thêm hàng vào bảng**

db.insert(PhuongNamLibSQLite.*getRole*(),null,values);

#### **5.2.2.12. Bảng AuthorManager**

Cấu trúc bảng

|  |  |  |  |
| --- | --- | --- | --- |
| **Tên cột** | **Kiểu dữ liệu** | **Ràng buộc** | **Mô tả** |
| ID\_Book | INT | PK, NOT NULL | Mã sách |
| ID\_Author | INT | PK, NOT NULL | Tên tác giả |

**Lệnh tạo bảng SQlite**

String sqlAuthorManager="create table "+*TABLE\_AUTHORMANAGER*+"("  
 +*KEY\_BOOK\_ID*+" integer " +","  
 +*KEY\_AUTHOR\_ID*+" integer "+","+"primary key"+"("+*KEY\_BOOK\_ID*+","+*KEY\_AUTHOR\_ID*+")"+")";  
db.execSQL(sqlAuthorManager);

**Lệnh tạo bảng SQLSever**

CREATE TABLE AuthorManager

(

ID\_Author INT NOT NULL,

ID\_Book INT NOT NULL,

PRIMARY KEY (ID\_Author, ID\_Book),

FOREIGN KEY (ID\_Author) REFERENCES Author(ID),

FOREIGN KEY (ID\_Book) REFERENCES Book(ID)

);

**Câu lệnh SQLite thêm hàng vào bảng**

db.insert(PhuongNamLibSQLite.*getTableAuthormanager*(),null,values);

## **5.3. Lập trình cho Controller chức năng thống kê bằng ASP.NET Core**

### **5.3.1. Thống kê doanh thu theo tháng trong năm**

CODE:

[HttpGet]

[Route("revenue-by-month")]

public IActionResult GetAllRevenueByMonth()

{

List<Revenue> revenueList = new List<Revenue>();

int revenue = 0;

for (int i = 1; i <= 12; i++)

{

var kq = db.CallCards.Join(db.Books, c => c.IdBook, b => b.Id, (c, b) => new

{

Revenue = b.Price,

BeginDate = c.BeginDate

})

.Where(d => d.BeginDate.Month == i && d.BeginDate.Year == DateTime.Now.Year);

foreach (var k in kq)

{

revenue = revenue + k.Revenue;

}

revenueList.Add(new Revenue(i, revenue));

revenue = 0;

}

return Ok(revenueList.ToList());

}

### **5.3.2. Thống kê doanh thu theo năm trong 1 thập kỷ**

CODE:

[HttpGet]

[Authorize]

[Route("revenue-by-year")]

public IActionResult GetAllRevenueByYear()

{

List<Revenue> revenueList = new List<Revenue>();

int revenue = 0;

for (int i = DateTime.Now.Year; i >= DateTime.Now.Year - 10; i--)

{

for (int j = 0; j < 12; j++)

{

var kq = db.CallCards.Join(db.Books, c => c.IdBook, b => b.Id, (c, b) => new

{

Revenue = b.Price,

BeginDate = c.BeginDate

})

.Where(d => d.BeginDate.Month == j && d.BeginDate.Year == i);

foreach (var k in kq)

{

revenue = revenue + k.Revenue;

}

}

revenueList.Add(new Revenue(i, revenue));

revenue = 0;

}

return Ok(revenueList.ToList());

}

### **5.3.3. Thống kê tỉ lệ mượn theo từng loại sách**

CODE:

[HttpGet]

[Authorize]

[Route("top-borrowed")]

public IActionResult GetTopBorrowed()

{

List<BorrowedKind> list = new List<BorrowedKind>();

var kq = db.Kinds

.Join(db.Books, k => k.Id, b => b.IdKind, (k, b) => new

{

namekind = k.Name,

idbook = b.Id

})

.Join(db.CallCards, t => t.idbook, c => c.IdBook, (t,c)=> new

{

t.namekind

}).GroupBy( n => n.namekind)

.Select(n => new

{

namekind=n.Key,

namekindCount=n.Count()

});

foreach (var item in kq)

{

list.Add(new BorrowedKind(item.namekind, item.namekindCount));

}

return Ok(list.ToList());

}

### **5.3.4. Thống kê top 10 sách được mượn nhiều nhất**

CODE:

[HttpGet]

[Route("top-10-book")]

public IActionResult GetAllTop10Book()

{

List<TopTenBook> list = new List<TopTenBook>();

var kq = db.Kinds

.Join(db.Books, k => k.Id, b => b.IdKind, (k, b) => new

{

namekind = k.Name,

idbook = b.Id,

namebook=b.Name

})

.Join(db.CallCards, t => t.idbook, c => c.IdBook, (t, c) => new

{

t.namebook

}).GroupBy(n => n.namebook)

.Select(n => new

{

namebook = n.Key,

namebookcount = n.Count()

})

.Join(db.Books, t => t.namebook, b => b.Name, (t, b) => new

{

nameofbook = t.namebook,

countofbook = t.namebookcount,

idkind = b.IdKind

})

.Join(db.Kinds, t => t.idkind, k=>k.Id, (t, k) => new

{

bookname = t.nameofbook,

kindname = k.Name,

bookcount = t.countofbook

}).OrderByDescending(n=>n.bookcount).Take(10);

foreach (var item in kq)

{

list.Add(new TopTenBook(item.bookname, item.kindname, item.bookcount));

}

return Ok(list.ToList());

}

## **5.4. Lập trình CSDL SQLite (SQLiteOpenHelper+Model+Dao)**

### **5.4.1. SQLiteOpenHelper**

**Mô tả:**

SQLiteOpenHelper là lớp tiện ích hỗ trợ lập trình Android với SQLite. Khi kế thừ Lớp này cần ghi đè 2 phương thức được mô tả trong phần hướng dẫn sử dụng sau đây.

**Hướng dẫn sử dụng:**

* onCreate(SQLiteDatabase db): chứa các lệnh tạo table (có thể chèn dữ liệu test)
* onUpgrade(SQLiteDatabase db, int i, int i1): chứa các lệnh xoá table và gọi lại onCreate khi nâng version

CODE:

@Override  
public void onCreate(SQLiteDatabase db) {  
  
  
 String sqlKind="create table "+*TABLE\_KIND*+"("  
 +*KEY\_KIND\_ID*+" integer primary key autoincrement "+","  
 +*KEY\_KIND\_NAME*+" text"+")";  
 db.execSQL(sqlKind);  
  
 String sqlRole="create table "+*TABLE\_ROLE*+"("  
 +*KEY\_ROLE\_ID*+" integer primary key autoincrement "+","  
 +*KEY\_ROLE\_NAME*+" text"+")";  
 db.execSQL(sqlRole);  
  
  
  
 String sqlLanguage="create table "+*TABLE\_LANGUAGE*+"("  
 +*KEY\_LANGUAGE\_ID*+" integer primary key autoincrement "+","  
 +*KEY\_LANGUAGE\_NAME*+" text"+")";  
 db.execSQL(sqlLanguage);  
  
 String sqlMemberType="create table "+*TABLE\_MEMBERTYPE*+"("  
 +*KEY\_MEMBERTYPE\_ID*+" integer primary key autoincrement "+","  
 +*KEY\_MEMBERTYPE\_NAME*+" text"+","  
 +*KEY\_MEMBERTYPE\_DISCOUNT*+" integer"+")";  
 db.execSQL(sqlMemberType);  
  
 String sqlMember="create table "+*TABLE\_MEMBER*+"("  
 +*KEY\_MEMBER\_ID*+" integer primary key autoincrement "+","  
 +*KEY\_MEMBER\_NAME*+" text" +","  
 +*KEY\_MEMBER\_COUNTTIME*+" integer"+","  
 +*KEY\_MEMBERTYPE\_ID*+" text" +")";  
 db.execSQL(sqlMember);  
  
 String sqlAuthor="create table "+*TABLE\_AUTHOR*+"("  
 +*KEY\_AUTHOR\_ID*+" integer primary key autoincrement "+","  
 +*KEY\_AUTHOR\_NAME*+" text" +","  
 +*KEY\_AUTHOR\_DOB*+" text" +")";  
 db.execSQL(sqlAuthor);  
  
  
  
 String sqlLibrarian="create table "+*TABLE\_LIBRARIAN*+"("  
 +*KEY\_LIBRARIAN\_ID*+" text primary key "+","  
 +*KEY\_LIBRARIAN\_NAME*+" text" +","  
 +*KEY\_LIBRARIAN\_PASSWORD*+" text"+","  
 +*KEY\_ROLE\_ID*+" integer" +")";  
 db.execSQL(sqlLibrarian);  
  
 String sqlPublisher="create table "+*TABLE\_PUBLISHER*+"("  
 +*KEY\_PUBLISHER\_ID*+" integer primary key autoincrement "+","  
 +*KEY\_PUBLISHER\_NAME*+" text" +","  
 +*KEY\_PUBLISHER\_LOCATION*+" text" +")";  
 db.execSQL(sqlPublisher);  
  
 String sqlBook="create table "+*TABLE\_BOOK*+"("  
 +*KEY\_BOOK\_ID*+" integer primary key autoincrement "+","  
 +*KEY\_BOOK\_NAME*+" text" +","  
 +*KEY\_BOOK\_PRICE*+" integer" +","  
 +*KEY\_BOOK\_QUANTITY*+" integer" +","  
 +*KEY\_KIND\_ID*+" integer" +","  
 +*KEY\_PUBLISHER\_ID*+" integer" +","  
 +*KEY\_LANGUAGE\_ID*+" integer"+")";  
 db.execSQL(sqlBook);  
  
 String sqlCallCard="create table "+*TABLE\_CALLCARD*+"("  
 +*KEY\_CALLCARD\_ID*+" integer primary key autoincrement "+","  
 +*KEY\_CALLCARD\_BEGINDATE*+" text" +","  
 +*KEY\_CALLCARD\_EXPIRESDATE*+" text" +","  
 +*KEY\_BOOK\_ID*+" integer" +","  
 +*KEY\_MEMBER\_ID*+" integer" +","  
 +*KEY\_LIBRARIAN\_ID*+" text" +")";  
 db.execSQL(sqlCallCard);  
  
 String sqlAuthorManager="create table "+*TABLE\_AUTHORMANAGER*+"("  
 +*KEY\_BOOK\_ID*+" integer " +","  
 +*KEY\_AUTHOR\_ID*+" integer "+","+"primary key"+"("+*KEY\_BOOK\_ID*+","+*KEY\_AUTHOR\_ID*+")"+")";  
 db.execSQL(sqlAuthorManager);  
}  
  
@Override  
public void onUpgrade(SQLiteDatabase sqLiteDatabase, int i, int i1) {  
 db.execSQL("drop table if exists "+*TABLE\_AUTHORMANAGER*);  
  
 db.execSQL("drop table if exists "+*TABLE\_AUTHOR*);  
  
 db.execSQL("drop table if exists "+*TABLE\_PUBLISHER*);  
  
 db.execSQL("drop table if exists "+*TABLE\_LANGUAGE*);  
  
 db.execSQL("drop table if exists "+*TABLE\_ROLE*);  
  
 db.execSQL("drop table if exists " +*TABLE\_BOOK*);  
  
 db.execSQL("drop table if exists " +*TABLE\_KIND*);  
  
 db.execSQL("drop table if exists " +*TABLE\_CALLCARD*);  
  
 db.execSQL("drop table if exists " +*TABLE\_MEMBER*);  
  
 db.execSQL("drop table if exists " +*TABLE\_LIBRARIAN*);  
  
 db.execSQL("drop table if exists "+*TABLE\_MEMBERTYPE*);  
 onCreate(db);  
}

### **5.4.2. Model Class và DAO**

#### **5.4.2.1. Book, BookToItem và BookManager**

Diagram, schematic

Description automatically generated with medium confidence

**Diễn giải**

|  |  |
| --- | --- |
| **Thành phần** | **Mô tả** |
| Book | Là lớp thực thể mô tả dữ liệu thủ thư |
| BookToItem | Là lớp thực thể mô tả dữ liệu thủ thư sau khi Inner Join |
| BookManager | Là lớp DAO cụ thể hóa các hoạt động truy xuất CSDL với bảng TABLE\_BOOK |

#### **5.4.2.2. CallCard, CallcardToItem và CallcardManager**

A picture containing chart

Description automatically generated

|  |  |
| --- | --- |
| **Thành phần** | **Mô tả** |
| Callcard | Là lớp thực thể mô tả dữ liệu phiếu mượn |
| CallcardToItem | Là lớp thực thể mô tả dữ liệu phiếu mượn sau khi Inner Join |
| CallcardManager | Là lớp DAO cụ thể hóa các hoạt động truy xuất CSDL với bảng TABLE\_CALLCARD |

#### **5.4.2.3. Librarian, LibrarianToItem và LibrarianManager**

A picture containing table

Description automatically generated

|  |  |
| --- | --- |
| **Thành phần** | **Mô tả** |
| Librarian | Là lớp thực thể mô tả dữ liệu nhân viên thư viện |
| LibrarianToItem | Là lớp thực thể mô tả dữ liệu nhân viên thư viện sau khi Inner Join |
| LibrarianManager | Là lớp DAO cụ thể hóa các hoạt động truy xuất CSDL với bảng TABLE\_LIBRARIAN |

#### **5.4.2.4. Member, MemberToItem và MemberManager**

Chart

Description automatically generated

|  |  |
| --- | --- |
| **Thành phần** | **Mô tả** |
| Member | Là lớp thực thể mô tả dữ liệu thành viên |
| MemberToItem | Là lớp thực thể mô tả dữ thành viên sau khi Inner Join |
| MemberManager | Là lớp DAO cụ thể hóa các hoạt động truy xuất CSDL với bảng TABLE\_MEMBER |

#### **5.4.2.5. Kind và KindManager**

Chart, scatter chart

Description automatically generated

|  |  |
| --- | --- |
| **Thành phần** | **Mô tả** |
| Kind | Là lớp thực thể mô tả dữ liệu loại sách |
| KindManager | Là lớp DAO cụ thể hóa các hoạt động truy xuất CSDL với bảng TABLE\_KIND |

#### **5.4.2.6. Role và RoleManager**

Chart

Description automatically generated

|  |  |
| --- | --- |
| **Thành phần** | **Mô tả** |
| Kind | Là lớp thực thể mô tả dữ liệu vai trò |
| KindManager | Là lớp DAO cụ thể hóa các hoạt động truy xuất CSDL với bảng TABLE\_ROLE |

## **5.5. Lập trình chức năng**

### **5.5.1. Màn hình chính User (UserMainActivity)**

|  |  |  |
| --- | --- | --- |
| **TT** | **Phương thức/ sự kiện** | **Mô tả** |
| 1 | setNavigationItemSelectedListener() | Điều hướng Navigation |
| 2 | setSupportActionBar() | Sử dụng toolbar |
| 3 | addOnItemTouchListener | Bắt sự kiện click cho recycler view |
| 4 | onOptionsItemSelected() | Bắt sự kiện khi touch vào các item trên toolbar |
| 5 | onActivityResult() | Nhận kết quả trả về từ activity khác |
| 6 | StartActivityForResult() | Chạy activity để lấy kết quả trả về |
| 7 | ActionMode.Callback() | Lập trình cho phản hồi của contextual action mode |
| 8 | startActionMode(ActionMode.Callback()) | Chạy contextual action mode |
| 9 | getSupportFragmentManager() | Quản lý các hành động của fragment |
| 10 | setOnQueryTextListener() | Sự kiện khi nhập vào SearchView |
| 11 | removeAllViewInLayout() | Xóa tất cả view khỏi layout hiện tại |

### **5.4.2. Màn hình chính Admin (AdminMainActivity)**

|  |  |  |
| --- | --- | --- |
| **TT** | **Phương thức/ sự kiện** | **Mô tả** |
| 1 | addOnItemTouchListener | Bắt sự kiện click cho recycler view |
| 2 | onOptionsItemSelected() | Bắt sự kiện khi touch vào các item trên toolbar |
| 3 | onActivityResult() | Nhận kết quả trả về từ activity khác |
| 4 | StartActivityForResult() | Chạy activity để lấy kết quả trả về |
| 5 | ActionMode.Callback() | Lập trình cho phản hồi của contextual action mode |
| 6 | startActionMode(ActionMode.Callback()) | Chạy contextual action mode |
| 7 | setOnQueryTextListener() | Sự kiện khi nhập vào SearchView |

### **5.5.3. Màn hình thống kê doanh thu tháng (RevenueFragment)**

|  |  |  |
| --- | --- | --- |
| **TT** | **Phương thức/ sự kiện** | **Mô tả** |
| 1 | RevenueByMonth() | Đổ dữ liệu cho Bar Chart thống kê doanh thu theo từng tháng |

### **5.5.4. Màn hình thống kê doanh thu năm(RevenueFragmentYear)**

|  |  |  |
| --- | --- | --- |
| **TT** | **Phương thức/ sự kiện** | **Mô tả** |
| 1 | RevenueByYear() | Đổ dữ liệu cho Bar Chart thống kê doanh thu theo từng năm |

### **5.5.5. Màn hình thống kê top 10 loại sách(TopBorrowedFragment)**

|  |  |  |
| --- | --- | --- |
| **TT** | **Phương thức/ sự kiện** | **Mô tả** |
| 1 | SetPieChart() | Đổ dữ liệu cho Pie Chart thống kê tỉ lệ mượn cho từng loại sách |
| 2 | SetTopTen() | Đổ dữ liệu cho Bar Chart thống kê top 10 sách mượn nhiều nhất |

### **5.5s.6. Màn hình đăng nhập(LoginActivity)**

|  |  |  |
| --- | --- | --- |
| **TT** | **Phương thức/ sự kiện** | **Mô tả** |
| 1 | Login() | Lấy id và mật khẩu thực hiện authentication nếu hợp lệ thì chuyển đến màn hình tương ứng với vai trò nếu không sẽ báo lỗi |
| 2 | ForgetPassDialog() | Hiển thị thông báo nếu quên mật khẩu |
| 3 | onBackPressed() | Sự kiện khi chạm vào nút back trên điện thoại |

# **6. Kiểm thử phần mềm và sửa lỗi**

## **6.1. LoginActivity**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **TH** | **Mô tả tình huống** | **Dữ liệu mẫu** | **Kết quả trông đợi** | **Fix** |
|  |  |  |  |  |

## **6.2. AdminMainActivity**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **TH** | **Mô tả tình huống** | **Dữ liệu mẫu** | **Kết quả trông đợi** | **Fix** |
|  |  |  |  |  |

## **6.3. UserMainActivity**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **TH** | **Mô tả tình huống** | **Dữ liệu mẫu** | **Kết quả trông đợi** | **Fix** |
|  |  |  |  |  |

# **7. Đóng gói và triển khai**

## **7.1. Sản phẩm phần mềm**

|  |  |  |
| --- | --- | --- |
| **TT** | **Thành phần** | **Mô tả** |
| 1 | LibMana.apk | File cài đặt |

# **8. Kết luận**

## **8.1. Khó khăn**

Khó khăn trong quá trình thực hiện dự án mẫu là em vẫn chưa hoàn thành dự án 1 cách hoàn thiện nhất trong thời gian môn học và sản phẩm vẫn còn nhiều thiếu sót trong giao diện và do lần đầu thực hiện 1 project có nhiều chức năng phải code để xử lý nên trình bày code trong project vẫn chưa được tối ưu nhất.

## **8.2. Thuận lợi**

Thuận lợi trong quá trình thực hiện dự án mẫu là những chức năng xử lý trong đề tài này em đã từng làm qua và em đã có tìm hiểu cơ bản để viết controller bằng C# nên việc xử lý thống kê dễ dàng hơn và theo em các chức năng trong đề tài này cũng không quá khó để code xử lý.