

TRƯỜNG ĐẠI HỌC VĂN HIẾN
KHOA CÔNG NGHỆ THÔNG TIN



TIỂU LUẬN MÔN HỌC
LẬP TRÌNH TRÊN CÁC THIẾT BỊ DI ĐỘNG

Đề tài:

XÂY DỰNG ỨNG DỤNG DI ĐỘNG
QUẢN LÝ SÁCH iBOOK

GVHD: ThS Nguyễn Hữu Vĩnh

SVTH: Hà Văn Được

MSSV: 181A010208

TP.HCM, tháng 4 năm 2021

TRƯỜNG ĐẠI HỌC VĂN HIẾN
KHOA CÔNG NGHỆ THÔNG TIN



TIỂU LUẬN MÔN HỌC
LẬP TRÌNH TRÊN CÁC THIẾT BỊ DI ĐỘNG

Đề tài:

XÂY DỰNG ỨNG DỤNG DI ĐỘNG
QUẢN LÝ SÁCH iBOOK

GVHD: ThS Nguyễn Hữu Vĩnh

SVTH: Hà Văn Được

MSSV: 181A010208

TP.HCM, tháng 4 năm 2021

NHẬN XÉT VÀ CHẤM ĐIỂM CỦA GIẢNG VIÊN

Họ và tên giảng viên: **Nguyễn Hữu Vĩnh**

Tên đề tài: **XÂY DỰNG ỨNG DỤNG DI ĐỘNG QUẢN LÝ SÁCH iBOOK.**

Nội dung nhận xét:

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

Điểm:

Bảng số:

Bảng chữ:

GIẢNG VIÊN CHẤM

(Ký, ghi rõ họ tên)

NGUYỄN HỮU VĨNH

MỤC MỤC

LỜI MỞ ĐẦU	1
CHƯƠNG 1. KHẢO SÁT VÀ THU THẬP YÊU CẦU	2
1.1. Khảo sát hiện trạng.....	2
1.1.1. Phân quản lý của Admin.....	2
1.1.2. Giao diện người dùng	2
1.2. Các chức năng của bài toán.....	2
CHƯƠNG 2. CƠ SỞ LÝ THUYẾT	3
2.1. Các công nghệ sử dụng trong đề tài	3
2.1.1. Android	3
2.1.2. Android Studio.....	4
2.1.3. SQLite	5
2.1.4. Java	7
2.1.5. Kiến trúc MVP.....	9
CHƯƠNG 3. PHÂN TÍCH HỆ THỐNG.....	16
3.1. Sơ đồ Use case	16
3.2. Đặc tả Use case	16
3.2.1. Use case Đăng Nhập:.....	16
3.2.3. Use case Đăng xuất.....	17
3.2.4. Use case Đặt hàng.....	18
3.2.5. Use case Xem thông tin sản phẩm.....	18
3.2.6. Use case đổi ngôn ngữ.....	19
3.3. Sơ đồ hoạt động.....	20
3.3.1. Đăng nhập	20
3.3.2. Đăng ký.....	20
3.3.3. Đăng xuất.....	21
3.3.4. Đặt hàng.....	21

3.3.5. Xem thông tin sản phẩm	22
3.4. Sơ đồ tuần tự	23
3.4.1. Đăng nhập	23
3.4.2. Đăng ký	23
3.4.3. Đăng xuất	24
3.4.4. Đặt hàng	24
3.4.5. Xem thông tin sản phẩm	25
CHƯƠNG 4. THIẾT KẾ VÀ XÂY DỰNG ỨNG DỤNG.....	26
4.1. Giao diện khách hàng.....	26
4.1.1. Giao diện trang mở đầu	26
4.1.2. Giao diện trang chủ.....	27
4.1.3. Giao diện chi tiết sản phẩm.	29
4.1.4. Giao diện Thêm sách	31
4.1.5. Giao diện đăng nhập	32
4.1.6. Giao diện đăng ký	33
4.2. Giao diện quản lý	35
4.2.1. Giao diện trang quản trị người dùng.....	35
CHƯƠNG 5: KẾT LUẬN.....	36
5.1. Kết quả đạt được	36
5.2. Hạn chế của đề tài	36
5.3. Hướng phát triển của đề tài	36
TÀI LIỆU THAM KHẢO	37

DANH MỤC HÌNH

Hình 1	Sơ đồ Use Case tổng quan.....	16
Hình 2	Sơ đồ hoạt động chức năng Đăng nhập.....	20
Hình 3	Sơ đồ hoạt động chức năng Đăng ký.....	20
Hình 4	Sơ đồ hoạt động chức năng Đăng xuất.....	21
Hình 5	Sơ đồ chức năng Đặt hàng.....	21
Hình 6	Sơ đồ hoạt động chức năng xem thông tin sản phẩm.....	22
Hình 7	Sơ đồ hoạt động chức năng đổi ngôn ngữ.....	22
Hình 8	Sơ đồ tuần tự chức năng Đăng nhập.....	23
Hình 9	Sơ đồ tuần tự chức năng Đăng ký.....	23
Hình 10	Sơ đồ tuần tự chức năng Đăng xuất.....	24
Hình 11	Sơ đồ tuần tự chức năng Đặt hàng.....	24
Hình 12	Sơ đồ tuần tự chức năng Xem thông tin sản phẩmĐổi ngôn ngữ.....	25
Hình 13	Sơ đồ tuần tự chức năng Đổi ngôn ngữ.....	25
Hình 14	Giao diện trang mở đầu.....	26
Hình 15	Giao diện trang chủ.....	27
Hình 16	Chức năng giao diện trang chủ.....	28
Hình 17	Giao diện trang thông tin sản phẩm.....	29
Hình 18	Giao diện đóng giao diện chi tiết sản phẩm.....	30
Hình 19	Giao diện thêm sách.....	31
Hình 20	Giao diện đăng nhập.....	32
Hình 21	Giao diện Đăng ký.....	33
Hình 22	Giao diện Quên mật khẩu.....	34
Hình 23	Giao diện danh sách độc giả.....	35

DANH SÁCH CÁC KÝ TỰ CHỮ VIẾT

Từ viết tắt	Giải thích
Tiếng Việt	
CSDL	Cơ sở dữ liệu
ĐVT	Đơn vị tính
TV	Thành viên
DH	Đơn hàng
SP	Sản phẩm
Tiếng Anh	
UC	Use Case
ADO.NET	ActiveX Data Object.NET
ASP.NET	Active Server Pages.NET
CSS	Cascading Style Sheets

LỜI MỞ ĐẦU

1. Tính cấp thiết của đề tài

Cùng với sự phát triển mạnh mẽ của Internet, ngày nay, việc sở hữu một ứng dụng không còn là điều xa lạ, thậm chí trong một số trường hợp còn là tiêu chuẩn bắt buộc đối với doanh nghiệp – công ty trong thời điểm cạnh tranh mang tính toàn cầu như hiện nay.

Cùng với đó công ty cổ phần thương mại DL là công ty kinh doanh mua bán game bản quyền đang rất cần một ứng dụng để có thể giới thiệu sản phẩm, thực hiện bán hàng trực tuyến.

2. Mục tiêu của đề tài

- + Xây dựng các chức năng cơ bản của một ứng dụng bán hàng thương mại.
- + Ứng dụng hiển thị sản phẩm đẹp, thu hút người dùng.
- + Hỗ trợ khách hàng một cách nhanh nhất khi nhận được yêu cầu liên hệ.
- + Cho phép khách hàng đăng ký, đăng nhập để sử dụng ứng dụng
- + Nắm bắt được công nghệ thiết kế web bằng Andoird Studio, SQLite,...

CHƯƠNG 1. KHẢO SÁT VÀ THU THẬP YÊU CẦU

1.1. Khảo sát hiện trạng

1.1.1. Phần quản lý của Admin

Admin quản lý toàn bộ hoạt động của cửa hàng.

Quản lý tài khoản khách hàng: Admin có thể xem thông tin tài khoản khách hàng, xoá tài khoản,...

1.1.2. Giao diện người dùng

- Ứng dụng không nên quá phức tạp.
- Dung lượng file không quá lớn.
- Thanh menu thật đơn giản.
- Phải có thông tin liên hệ.
- Font chữ đơn giản, dễ nhìn, màu sắc hài hòa.
- An toàn và bảo mật dữ liệu.
- Cho phép khách hàng xem thông tin chi tiết.
- Cho phép khách hàng đặt mua sản phẩm.
- Cho phép khách hàng đăng ký tài khoản.

1.2. Các chức năng của bài toán

- Admin:
 - + Quản lý khách hàng.
- Khách hàng:
 - + Đăng nhập. / Đăng ký
 - + Xem thông tin sản phẩm.
 - + Quản lý đơn hàng.

CHƯƠNG 2. CƠ SỞ LÝ THUYẾT

2.1. Các công nghệ sử dụng trong đề tài

2.1.1. Android

Android là một hệ điều hành dựa trên nền tảng Linux được thiết kế dành cho các thiết bị di động có màn hình cảm ứng như điện thoại thông minh và máy tính bảng. Ban đầu, Android được phát triển bởi Android, Inc. với sự hỗ trợ tài chính từ Google và sau này được chính Google mua lại vào năm 2005.

Android ra mắt vào năm 2007 cùng với tuyên bố thành lập Liên minh thiết bị cầm tay mở: một hiệp hội gồm các công ty phần cứng, phần mềm, và viễn thông với mục tiêu đẩy mạnh các tiêu chuẩn mở cho các thiết bị di động. Chiếc điện thoại đầu tiên chạy Android được bán vào năm 2008.

Android có mã nguồn mở và Google phát hành mã nguồn theo Giấy phép Apache. Chính mã nguồn mở cùng với một giấy phép không có nhiều ràng buộc đã cho phép các nhà phát triển thiết bị, mạng di động và các lập trình viên nhiệt huyết được điều chỉnh và phân phối Android một cách tự do. Ngoài ra, Android còn có một cộng đồng lập trình viên đông đảo chuyên viết các ứng dụng để mở rộng chức năng của thiết bị, bằng một loại ngôn ngữ lập trình Java có sửa đổi. Tháng 10 năm 2012, có khoảng 700.000 ứng dụng trên Android, và số lượt tải ứng dụng từ Google Play, cửa hàng ứng dụng chính của Android, ước tính khoảng 25 tỷ lượt.

Những yếu tố này đã giúp Android trở thành nền tảng điện thoại thông minh phổ biến nhất thế giới, vượt qua Symbian OS vào quý 4 năm 2010, và được các công ty công nghệ lựa chọn khi họ cần một hệ điều hành không nặng nề, có khả năng tinh chỉnh, và giá rẻ chạy trên các thiết bị công nghệ cao thay vì tạo dựng từ đầu. Kết quả là mặc dù được thiết kế để chạy trên điện thoại và máy tính bảng, Android đã xuất hiện trên TV, máy chơi game và các thiết bị điện tử khác. Bản chất mở của Android cũng khích lệ một đội ngũ đông đảo lập trình viên và những người đam mê sử dụng mã nguồn mở để tạo ra những dự án do cộng đồng quản lý. Những dự án này bổ sung các

tính năng cao cấp cho những người dùng thích tìm tòi hoặc đưa Android vào các thiết bị ban đầu chạy hệ điều hành khác.

Android chiếm 87,7% thị phần điện thoại thông minh trên toàn thế giới vào thời điểm quý 2 năm 2017, với tổng cộng 2 tỷ thiết bị đã được kích hoạt và 1,3 triệu lượt kích hoạt mỗi ngày. Sự thành công của hệ điều hành cũng khiến nó trở thành mục tiêu trong các vụ kiện liên quan đến bằng phát minh, góp mặt trong cái gọi là "cuộc chiến điện thoại thông minh" giữa các công ty công nghệ.

2.1.2. Android Studio

Có nhiều công cụ để phát triển Android nhưng đến nay công cụ chính thức và mạnh mẽ nhất là Android Studio. Đây là IDE (Môi trường phát triển tích hợp) chính thức cho nền tảng Android, được phát triển bởi Google và được sử dụng để tạo phần lớn các ứng dụng mà bạn có thể sử dụng hàng ngày.

Android Studio lần đầu tiên được công bố tại hội nghị Google I/O vào năm 2013 và được phát hành cho công chúng vào năm 2014 sau nhiều phiên bản beta khác nhau. Trước khi được phát hành, các nhà phát triển Android thường sử dụng các công cụ như Eclipse IDE, một IDE Java chung cũng hỗ trợ nhiều ngôn ngữ lập trình khác.

Android Studio khiến việc tạo ứng dụng trở nên dễ dàng hơn đáng kể so với phần mềm không chuyên dụng. Đối với người mới bắt đầu, có rất nhiều thứ để học và nhiều thông tin có sẵn, thậm chí thông qua các kênh chính thức nhưng chúng có thể đã lỗi thời hoặc quá nhiều thông tin khiến họ cảm thấy choáng ngợp. Bài viết này sẽ giải thích ngắn gọn nhưng chi tiết về một số chức năng cơ bản của nó để bạn có thể nắm bắt được bước đầu trong công cuộc phát triển Android của mình.

Chức năng của Android Studio là cung cấp giao diện để tạo các ứng dụng và xử lý phần lớn các công cụ quản lý file phức tạp đằng sau hậu trường. Ngôn ngữ lập trình được sử dụng ở đây là Java và được cài đặt riêng trên thiết bị của bạn. Android Studio rất đơn giản, bạn chỉ cần viết, chỉnh sửa và lưu các dự án của mình và các file trong dự án đó. Đồng thời, Android Studio sẽ cấp quyền truy cập vào Android SDK.

Hãy coi đây là đuôi cho code Java cho phép nó chạy trơn tru trên các thiết bị Android và tận dụng lợi thế của phần cứng gốc. Bạn cần sử dụng ngôn ngữ lập trình Java để viết các chương trình, Android SDK có nhiệm vụ kết nối các phần này lại với nhau. Cùng lúc đó Android Studio kích hoạt để chạy code, thông qua trình giả lập hoặc qua một phần cứng kết nối với thiết bị. Sau đó, bạn cũng có thể “gỡ rối” chương trình khi nó chạy và nhận phản hồi giải thích sự cố, v.v... để bạn có thể nhanh chóng giải quyết vấn đề.

Google đã nỗ lực rất nhiều để làm cho Android Studio trở nên mạnh mẽ và hữu ích nhất có thể. Nó cung cấp những gợi ý trực tiếp trong khi viết code và thường đề xuất những thay đổi cần thiết để sửa lỗi hoặc làm code hiệu quả hơn. Ví dụ, nếu không sử dụng biến, biến đó sẽ được tô đậm bằng màu xám. Và khi bắt đầu gõ một dòng code, Android Studio sẽ cung cấp danh sách gợi ý tự hoàn thành để giúp bạn hoàn thiện dòng code đó. Chức năng này rất hữu ích khi bạn không nhớ được chính xác cú pháp hoặc để tiết kiệm thời gian.

2.1.3. SQLite

SQLite là gì?

SQLite là một thư viện phần mềm mà triển khai một SQL Database Engine, không cần máy chủ, không cần cấu hình, khép kín và nhỏ gọn. Nó là một cơ sở dữ liệu, không cần cấu hình, có nghĩa là giống như các cơ sở dữ liệu khác mà bạn không cần phải cấu hình nó trong hệ thống của mình.

SQLite engine không phải là một quy trình độc lập (standalone process) như các cơ sở dữ liệu khác, bạn có thể liên kết nó một cách tĩnh hoặc động tùy theo yêu cầu của bạn với ứng dụng của bạn. SQLite truy cập trực tiếp các file lưu trữ (storage files) của nó.

Tại sao lại là SQLite?

SQLite không yêu cầu một quy trình hoặc hệ thống máy chủ riêng biệt để hoạt động.

SQLite không cần cấu hình, có nghĩa là không cần thiết lập hoặc quản trị.

Một cơ sở dữ liệu SQLite hoàn chỉnh được lưu trữ trong một file disk đa nền tảng (cross-platform disk file).

SQLite rất nhỏ và trọng lượng nhẹ, dưới 400KiB được cấu hình đầy đủ hoặc dưới 250KiB với các tính năng tùy chọn bị bỏ qua.

SQLite là khép kín (self-contained), có nghĩa là không có phụ thuộc bên ngoài.

Các transaction trong SQLite hoàn toàn tuân thủ ACID, cho phép truy cập an toàn từ nhiều tiến trình (process) hoặc luồng (thread).

SQLite hỗ trợ hầu hết các tính năng ngôn ngữ truy vấn (query language) được tìm thấy trong tiêu chuẩn SQL92 (SQL2).

SQLite được viết bằng ANSI-C và cung cấp API đơn giản và dễ sử dụng.

SQLite có sẵn trên UNIX (Linux, Mac OS-X, Android, iOS) và Windows (Win32, WinCE, WinRT).

Lịch sử tóm tắt của SQLite

2000 - D. Richard Hipp đã thiết kế SQLite cho mục đích không yêu cầu quản trị để vận hành chương trình.

2000 - Vào tháng 8, SQLite 1.0 được phát hành với trình quản lý cơ sở dữ liệu GNU.

2011 - Hipp tuyên bố bổ sung giao diện UNQL vào SQLite DB và phát triển UNQLite (Cơ sở dữ liệu hướng tài liệu - Document oriented database).

Hạn chế SQLite

Một số tính năng của SQL92 không được hỗ trợ trong SQLite được liệt kê trong bảng sau:

STT	Đặc điểm	Mô tả
-----	----------	-------

1	RIGHT OUTER JOIN	Chỉ có LEFT OUTER JOIN được thực hiện.
2	FULL OUTER JOIN	Chỉ có LEFT OUTER JOIN được thực hiện.
3	ALTER TABLE	Các biến thể RENAME TABLE và ADD COLUMN của lệnh ALTER TABLE được hỗ trợ. DROP COLUMN, ALTER COLUMN, ADD CONSTRAINT không được hỗ trợ.
4	Trigger support	Trigger FOR EACH ROW được hỗ trợ nhưng không hỗ trợ FOR EACH STATEMENT.
5	VIEWS	VIEWS trong SQLite là chỉ đọc. Bạn không thể thực thi câu lệnh DELETE, INSERT hoặc UPDATE trên một view.
6	GRANT và REVOKE	Các quyền truy cập duy nhất có thể được áp dụng là các quyền truy cập file thông thường (normal file) của hệ điều hành.

2.1.4. Java

Java là gì?

Java là một ngôn ngữ lập trình hiện đại, bậc cao, hướng đối tượng, bảo mật và mạnh mẽ. và là một Platform.

Platform: Bất cứ môi trường phần cứng hoặc phần mềm nào mà trong đó có một chương trình chạy, thì được hiểu như là một Platform. Với môi trường runtime riêng cho mình (JRE) và API, Java được gọi là Platform.

Ngôn ngữ lập trình Java ban đầu được phát triển bởi Sun Microsystems do James Gosling khởi xướng và phát hành vào năm 1995. Phiên bản mới nhất của Java Standard Edition là Java SE 8. Với sự tiến bộ của Java và sự phổ biến rộng rãi của nó, nhiều cấu hình đã được xây dựng để phù hợp với nhiều loại nền

tăng khác nhau. Ví dụ: J2EE cho các ứng dụng doanh nghiệp, J2ME cho các ứng dụng di động.

Các phiên bản J2 mới đã được đổi tên thành Java SE, Java EE và Java ME. Phương châm của java là "Write Once, Run Anywhere" - viết một lần chạy nhiều nơi, nghĩa là bạn chỉ cần viết một lần trên window chẳng hạn, sau đó vẫn chương trình đó bạn có thể chạy trên Linux, Android, các thiết bị J2ME...

Các tính năng của Java

Hướng đối tượng - Trong Java, mọi thứ đều là một Object. Java có thể dễ dàng mở rộng và bảo trì vì nó được xây dựng dựa trên mô hình Object.

Nền tảng độc lập - Không giống nhiều ngôn ngữ lập trình khác bao gồm cả C và C ++, khi Java được biên dịch, nó không được biên dịch thành ngôn ngữ máy nền tảng cụ thể, thay vào mã byte - nền tảng độc lập. Mã byte này được thông dịch bởi máy ảo (JVM) trên nền tảng nào đó mà nó đang chạy.

Đơn giản - Java được thiết kế để dễ học. Nếu bạn hiểu khái niệm cơ bản về OOP Java, sẽ rất dễ để trở thành master về java.

Bảo mật - Với tính năng an toàn của Java, nó cho phép phát triển các hệ thống không có virus, giả mạo. Các kỹ thuật xác thực dựa trên mã hoá khóa công khai.

Kiến trúc - trung lập - Trình biên dịch Java tạo ra định dạng tệp đối tượng kiến trúc trung lập, làm cho mã biên dịch được thực thi trên nhiều bộ vi xử lý, với sự hiện diện của hệ điều hành Java.

Portable - Là kiến trúc tập trung và không có khía cạnh thực hiện phụ thuộc của đặc tả này làm cho Java khả chuyển. Trình biên dịch trong Java được viết bằng ANSI C, đó là một tập con POSIX.

Mạnh mẽ - Java làm nỗ lực để loại trừ các tình huống dễ bị lỗi bằng cách kiểm tra lỗi tại thời gian biên dịch và kiểm tra lỗi tại runtime.

Đa luồng - Với tính năng đa luồng của Java có thể viết các chương trình có thể thực hiện nhiều tác vụ đồng thời. Tính năng thiết kế này cho phép các nhà phát triển xây dựng các ứng dụng tương tác có thể chạy trơn tru hơn.

Thông dịch - Mã byte Java được dịch trực tiếp tới các máy tính gốc và không được lưu trữ ở bất cứ đâu.

Hiệu năng cao - Với việc sử dụng trình biên dịch Just-In-Time, Java cho phép thực hiện hiệu năng cao.

Phân tán - Java được thiết kế cho môi trường phân tán của Internet.

Năng động - Java là năng động hơn C hoặc C++ vì nó được thiết kế để thích nghi với môi trường đang phát triển. Các chương trình Java có thể mang một lượng lớn thông tin tại runtime mà có thể được sử dụng để xác minh và giải quyết các truy cập vào các đối tượng tại runtime.

2.1.5. Kiến trúc MVP

Khi chúng ta đang phát triển một ứng dụng phức tạp, chúng ta thường bắt gặp những thách thức mà có lẽ đã được giải quyết trước đó và đã có một số giải pháp khá tuyệt vời.

Các giải pháp như vậy thường được gọi là patterns. Chúng ta thường nói về design patterns và architectural patterns. Chúng đơn giản hóa phát triển và nên sử dụng chúng bất cứ khi nào thích hợp.

Hướng dẫn này sẽ giúp bạn hiểu được tầm quan trọng của một dự án được thiết kế tốt và tại sao kiến trúc tiêu chuẩn của Android không phải là luôn luôn đủ.

Chúng ta sẽ thảo luận về một số vấn đề tiềm năng mà bạn có thể gặp phải khi phát triển các ứng dụng Android và mình sẽ chỉ cho các bạn làm thế nào để giải quyết các vấn đề bằng cách cải thiện khả năng kiểm thử và độ tin cậy của ứng dụng thông qua mô hình Model View Presenter (MVP).

Kiến trúc Android

Các thiết kế của một dự án là mối quan tâm ngay từ đầu. Một trong những điều đầu tiên chúng ta nên xem xét là kiến trúc mà chúng ta dự định áp dụng vì nó sẽ xác định cách các yếu tố khác nhau của ứng dụng liên quan đến nhau như thế nào. Nó cũng sẽ thiết lập một số nguyên tắc cơ bản để hướng dẫn chúng ta trong phát triển.

Nói chung, một framework hay SDK hy vọng mọi việc được thực hiện một cách nhất định, nhưng điều đó không phải luôn luôn là một trong những quyền cho một dự án. Đôi khi, không có cách nào xác định trước hoặc đúng làm việc, để lại các quyết định thiết kế lên đến các nhà phát triển. Android SDK hy vọng mọi việc được thực hiện một cách nhất định, nhưng đó không phải là luôn luôn đủ hoặc sự lựa chọn tốt nhất.

Mặc dù Android cung cấp một SDK xuất sắc, mô hình kiến trúc của nó là khá bất thường và có thể dễ dàng có được theo cách của bạn trong thời gian phát triển, đặc biệt là khi xây dựng các ứng dụng phức tạp cần phải được kiểm tra và duy trì trong một thời gian dài. May mắn thay, chúng ta có thể chọn từ nhiều giải pháp kiến trúc để giải quyết vấn đề này.

Vấn đề là gì ?

Đây là một câu hỏi khó. Một số có thể nói rằng không có bất kỳ vấn đề với các kiến trúc được cung cấp bởi Android. Chắc chắn, nó được công việc làm. Nhưng chúng ta có thể làm tốt hơn? Tôi mạnh mẽ tin rằng chúng ta có thể.

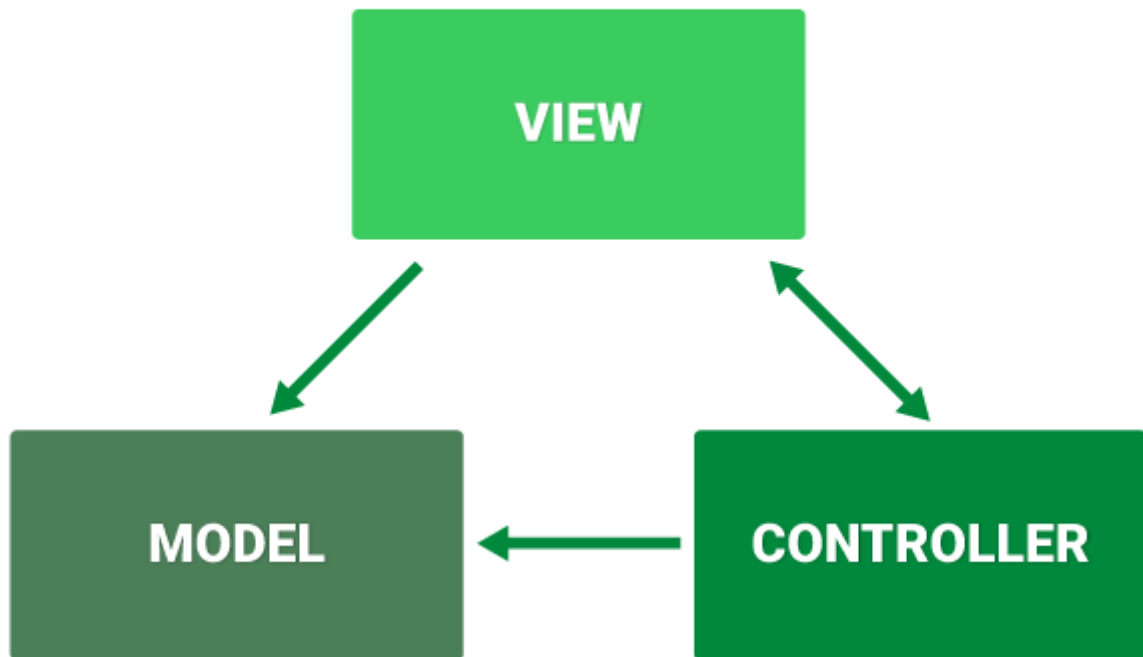
Các công cụ được cung cấp bởi Android, với cách bài trí, hoạt động, và các cấu trúc dữ liệu, dường như hướng chúng ta theo hướng mô hình Model View Controller (MVC). MVC là một mô hình rắn thành lập nhằm mục đích cô lập các vai trò khác nhau của ứng dụng. Điều này được gọi là tách quan tâm.

Kiến trúc này tạo ra ba lớp:

- + Model
- + View
- + Controller

Mỗi layer có trách nhiệm một khía cạnh của ứng dụng. Model phản ứng với logic kinh doanh, View là giao diện người dùng, và Controller trung gian Xem tiếp cận với Model.

Model View Controller



Nhưng nếu chúng ta phân tích chặt chẽ kiến trúc Android, đặc biệt là mối quan hệ giữa View (Activities, Fragments, vv) và Model (cấu trúc dữ liệu), chúng ta có thể kết luận rằng nó không thể được coi là MVC. Nó hoàn toàn khác với MVC và theo những luật riêng của nó. Nó chắc chắn có thể nhận được theo cách của bạn khi mục tiêu của bạn là để tạo ra các ứng dụng tốt nhất có thể.

Cụ thể hơn, nếu chúng ta nghĩ về sự kết nối cộng sinh giữa Loaders và Activities hoặc Fragments, bạn có thể hiểu lý do tại sao chúng ta nên chú ý tới kiến trúc của Android. Một Activity hoặc Fragment có trách nhiệm gọi Loader, thứ mà sẽ lấy dữ liệu và gửi trả lại cho cha mẹ của nó. Sự tồn tại của nó là hoàn toàn gắn liền với mẹ của nó và không có sự tách biệt giữa vai trò View (Activity/Fragment) và business logic được thực hiện bởi các Loader.



Chúng ta sử dụng unit test như thế nào, trong một ứng dụng, trong đó dữ liệu (Loader) được kết hợp rất chặt chẽ với các View (Hoạt động hoặc Fragment) nếu bản chất của đơn vị kiểm tra là kiểm tra các mảnh nhỏ nhất có thể của mã? Nếu bạn đang làm việc trong một đội và bạn cần phải thay đổi điều gì đó trong mã của người khác, làm thế nào bạn có thể tìm ra vấn đề nếu dự án không dính vào một mô hình kiến trúc nổi tiếng và bất cứ điều gì có thể là nghĩa đen bất cứ nơi nào?

Giải pháp là gì ?

Chúng tôi có thể giải quyết điều này bằng cách thực hiện một mô hình kiến trúc khác nhau, và may mắn thay, Android SDK cho phép chúng ta lựa chọn giữa nhiều giải pháp. Chúng tôi có thể thu hẹp lựa chọn của chúng tôi đến các giải pháp phù hợp nhất cho Android. Các mô hình Model View Controller (MVC) là một lựa chọn tốt, nhưng một một thậm chí tốt hơn là Model View Presenter (MVP) mô hình liên quan chặt chẽ. MVP đã được phát triển bằng cách sử dụng cơ sở giống như MVC, nhưng với một mô hình hiện đại hơn mà tạo ra một tách thậm chí tốt hơn các mối quan tâm và tối đa hóa khả năng kiểm thử của ứng dụng.

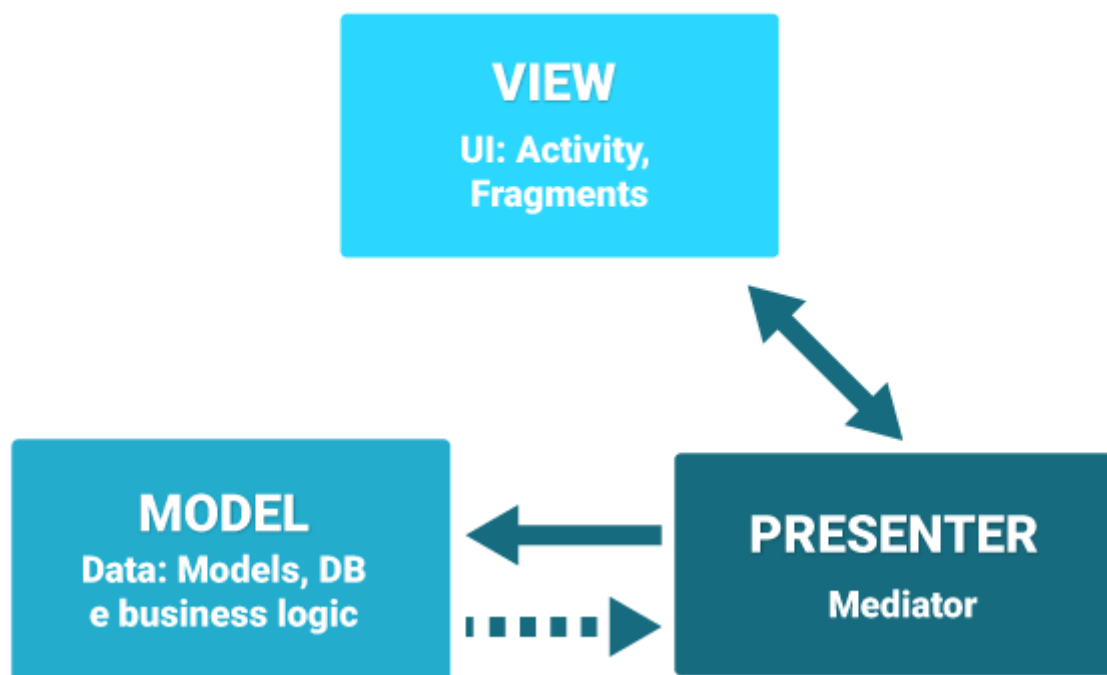
Model View Presenter

Như tôi đã đề cập trước đó, tách mối quan tâm không phải là điểm mạnh của Android. May mắn thay, các mô hình Model View Presenter cải thiện này đáng kể. MVP tách ứng dụng thành ba lớp:

- + Model
- + View
- + Presenter

Mỗi một người có trách nhiệm của mình và thông tin liên lạc giữa các lớp được quản lý bởi các Presenter. Các thuyết trình hoạt động như một trung gian hòa giải giữa các bộ phận khác nhau.

Model View Presenter



Model tổ chức business logic của ứng dụng. Nó kiểm soát như thế nào dữ liệu có thể được tạo ra, lưu trữ, và sửa đổi.

View là một giao diện thụ động hiển thị dữ liệu và các tuyến đường hành động dùng cho Presenter.

Presenter đóng vai trò như người đàn ông trung niên. Nó lấy dữ liệu từ Model và cho thấy nó trong các View. Nó cũng xử lý hành động người dùng mong muốn được nó bằng các View.

Khác nhau giữa MVC và MVP

Các mô hình Model View Presenter được dựa trên mô hình Model View Controller. Kể từ khi họ chia sẻ một số khái niệm, nó có thể được khó khăn để phân biệt chúng. Các Presenter và Controller có vai trò tương tự. Chúng có trách nhiệm giao tiếp giữa Model và View. Điều đó nói rằng, Controller không quản lý Model và View như đúng như trình bày không.

Trong mô hình MVC, View lớp có phần thông minh và có thể lấy dữ liệu trực tiếp từ Model. Trong các mô hình MVP, View là hoàn toàn thụ động và dữ liệu luôn được phân phối đến các Xem theo Presenter. Bộ điều khiển trong MVC cũng có thể được chia sẻ giữa nhiều lượt xem. Trong MVP, View và trình bày có một mối quan hệ một-một, do đó, các Presenter được gắn với một View.

- + Trong MVP, View không thể truy cập các Model.
- + Các thuyết trình được gắn với một Xem đơn.
- + Xem là hoàn toàn thụ động trong các mô hình MVP.

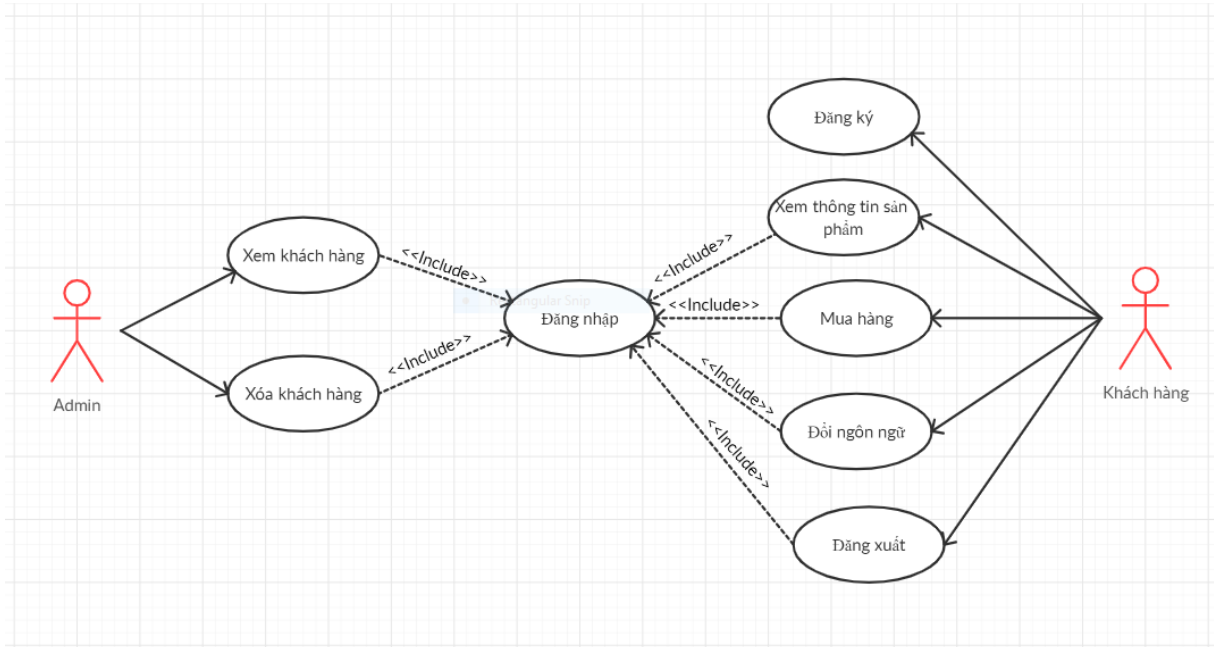
Những khác biệt về khái niệm làm cho rằng các mô hình MVP đảm bảo một tách tốt hơn các mối quan tâm và nó cũng làm tăng đáng kể khả năng kiểm thử của ứng dụng bằng cách thúc đẩy một tách lớn hơn trong ba lớp lõi: Activity, Fragment, and View Objects

Có rất nhiều cách hiểu về cách MVP có thể được thực hiện trên Android. Nhìn chung, mặc dù, hoạt động và các mảnh vỡ được giao vai trò của Xem và chịu trách nhiệm cho việc tạo ra các Presenter và Model. The View cũng là trách nhiệm duy trì mô hình và thuyết trình giữa những thay đổi cấu hình, thông báo cho họ về sự phá hủy cuối cùng của View.

Xem các đối tượng khác, chẳng hạn như RecyclerView, cũng có thể được coi là một phần của lớp View trong MVP. Có một mối quan hệ một-một giữa View và các thuyết trình và các tình huống phức tạp đôi khi có thể yêu cầu nhiều diễn giả.

CHƯƠNG 3. PHÂN TÍCH HỆ THỐNG

3.1. Sơ đồ Use case



Hình 1 Sơ đồ Use Case tổng quan

3.2. Đặc tả Use case

3.2.1. Use case Đăng Nhập:

Tên Use case	Đăng Nhập
Mô tả ngắn	Cho phép Admin và khách hàng đăng nhập vào hệ thống
Tác nhân	Admin, Khách hàng
Tiền điều kiện	Admin và Khách hàng đăng nhập vào hệ thống
Dòng sự kiện	
Dòng hành động chính	Chọn chức năng Đăng Nhập: <ul style="list-style-type: none"> - Nhập tên đăng nhập và mật khẩu vào giao diện đăng nhập. - Hệ thống kiểm tra tên đăng nhập và mật khẩu. Hệ thống sẽ kiểm tra tài khoản đó nếu của là tài khoản của Admin sẽ chuyển sang trang quản trị, nếu là tài khoản của khách hàng sẽ chuyển sang trang mua hàng.

Dòng hành động thay thế	Nếu nhập sai tài khoản, mật khẩu. Hệ thống sẽ thông báo đăng nhập không thành công do sai tài khoản hoặc mật khẩu.
Hậu điều kiện	Thoát ứng dụng

3.2.2. Use case đăng ký

Tên Use case	Đăng ký
Mô tả ngắn	Cho phép khách hàng đăng ký tài khoản để mua hàng
Tác nhân	Khách hàng
Tiền điều kiện	Khách hàng cần vào hệ thống
Dòng sự kiện	
Dòng hành động chính	<p>Chọn đăng ký:</p> <p>Khách hàng sẽ nhập các thông tin trên giao đăng ký. Hệ thống sẽ kiểm tra thông tin. Nếu thành công hệ thống sẽ chuyển sang giao diện đăng nhập với tài khoản vừa đăng ký trên</p>
Dòng hành động thay thế	Nếu dữ liệu nhập vào không hợp lệ, hệ thống sẽ thông báo "dữ liệu không hợp lệ" và yêu cầu nhập lại
Hậu điều kiện	Thoát ứng dụng

3.2.3. Use case Đăng xuất

Tên Use case	Đăng xuất
Mô tả ngắn	Cho phép khách hàng đăng xuất tài khoản để mua hàng
Tác nhân	Khách hàng
Tiền điều kiện	Khách hàng cần vào hệ thống
Dòng sự kiện	

Dòng hành động chính	Chọn đăng xuất: Khách hàng đăng xuất ra khỏi hệ thống.
Dòng hành động thay thế	
Hậu điều kiện	Thoát ựng

3.2.4. Use case Đặt hàng

Tên Use case	Đặt hàng
Mô tả ngắn	Cho phép khách hàng đặt mua các sản phẩm
Tác nhân	Khách hàng
Tiền điều kiện	Khách hàng cần đăng nhập vào hệ thống
Dòng sự kiện	
Dòng hành động chính	Chọn đặt hàng: Khách hàng sẽ chọn các sản phẩm cần mua, sau đó nhấn Đặt Hàng, các sản phẩm sẽ được thêm vào giỏ hàng.
Dòng hành động thay thế	Nếu dữ liệu nhập vào không hợp lệ, hệ thống sẽ thông báo "dữ liệu không hợp lệ" và yêu cầu nhập lại
Hậu điều kiện	Thoát ứng dụng

3.2.5. Use case Xem thông tin sản phẩm

Tên Use case	Xem thông tin sản phẩm
Mô tả ngắn	Cho phép khách hàng xem thông tin sản phẩm
Tác nhân	Khách hàng
Tiền điều kiện	Khách hàng vào trang chủ

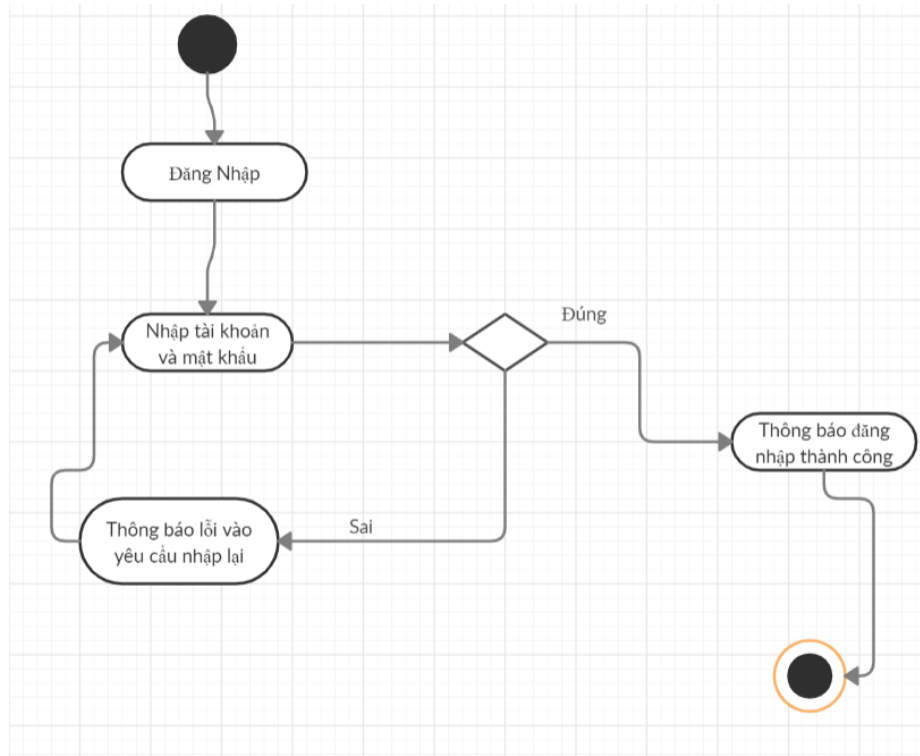
Dòng sự kiện	
Dòng hành động chính	Khách hàng chọn vào sản phẩm cần xem: - Hệ thống sẽ hiển thị thông tin sản phẩm bao gồm: nhà sản xuất, loại sản phẩm và đặc điểm nổi bật
Dòng hành động thay thế	
Hậu điều kiện	Thoát ứng dụng

3.2.6. Use case đổi ngôn ngữ

Tên Use case	Đổi ngôn ngữ
Mô tả ngắn	Cho phép khách hàng đổi sang ngôn ngữ mong muốn
Tác nhân	Khách hàng
Tiền điều kiện	Không
Dòng sự kiện	
Dòng hành động chính	- Hệ thống sẽ tự động thay đổi ngôn ngữ theo ngôn ngữ cài đặt ở máy khách hàng.
Dòng hành động thay thế	Nếu khách hàng sử dụng ngôn ngữ không có trong danh sách ngôn ngữ của ứng dụng thì ứng dụng sẽ tự động chuyển sang Tiếng Việt.
Hậu điều kiện	Thoát ứng dụng

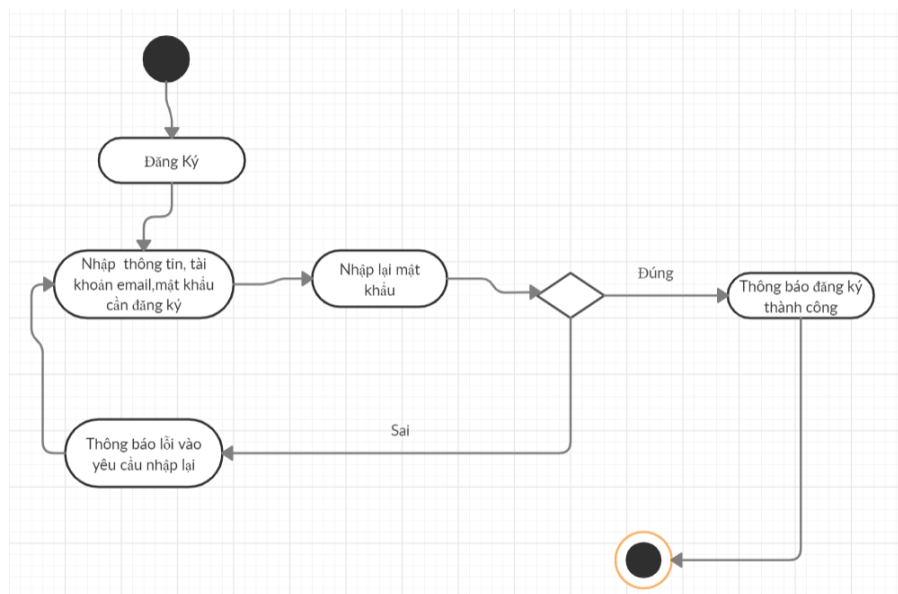
3.3. Sơ đồ hoạt động

3.3.1. Đăng nhập



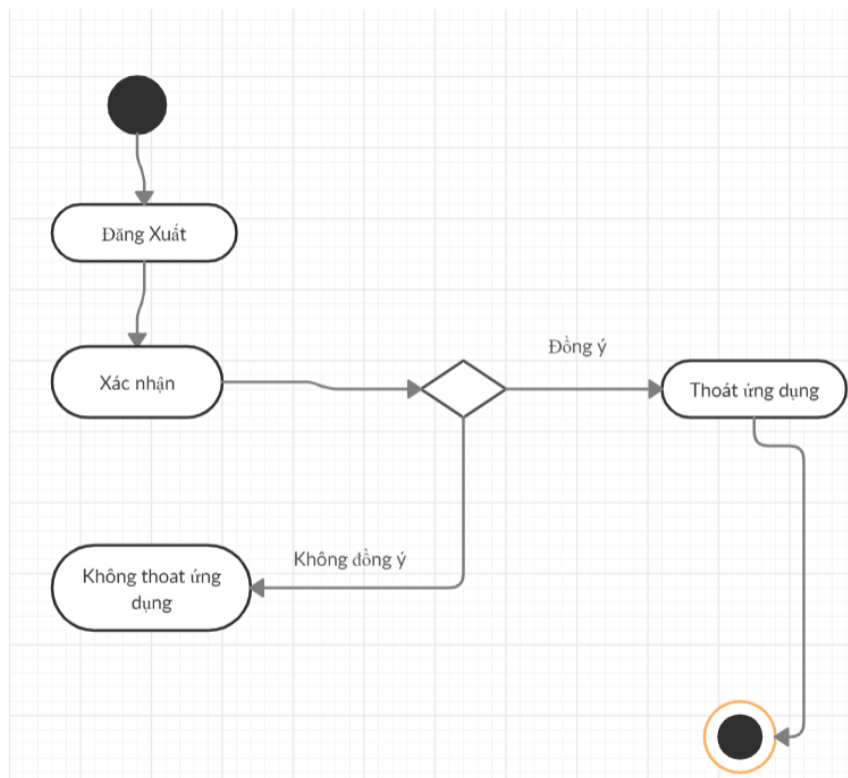
Hình 2 Sơ đồ hoạt động chức năng Đăng nhập

3.3.2. Đăng ký



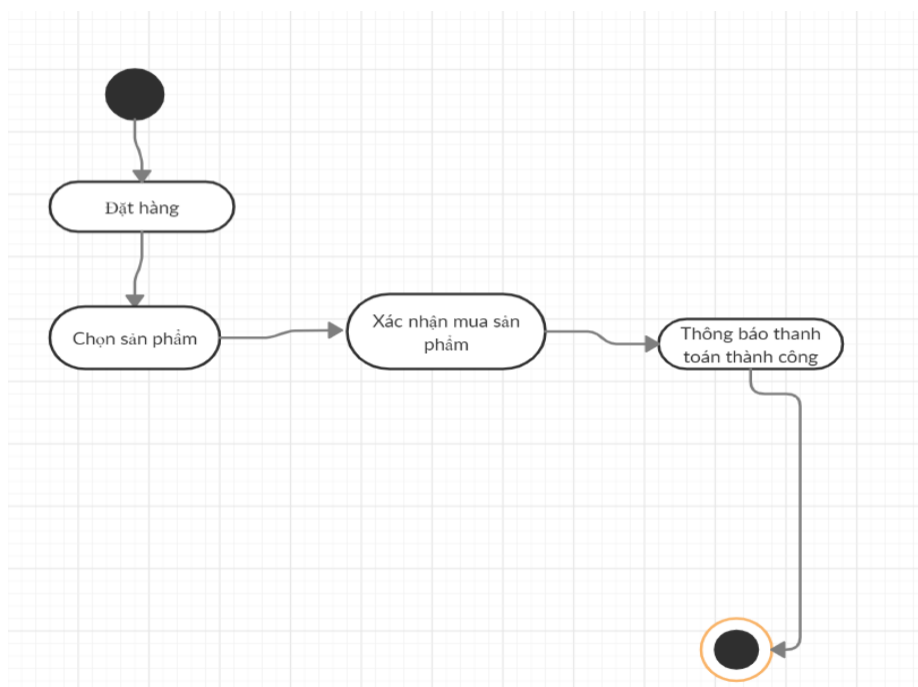
Hình 3 Sơ đồ hoạt động chức năng Đăng ký

3.3.3. Đăng xuất



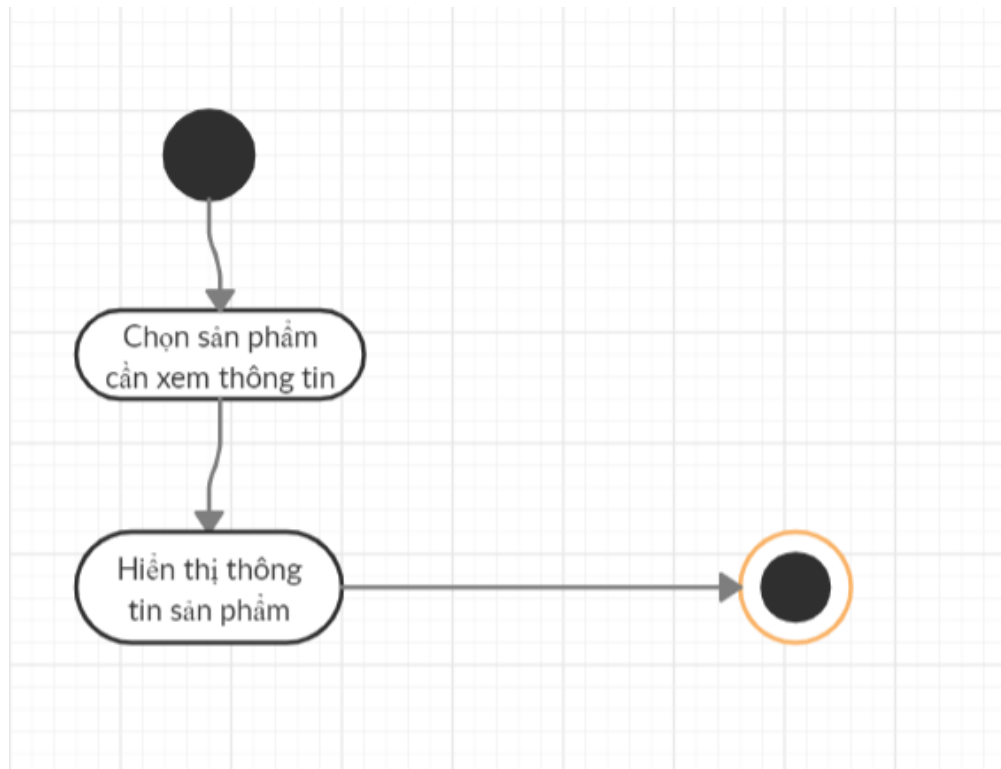
Hình 4 Sơ đồ hoạt động chức năng Đăng xuất

3.3.4. Đặt hàng

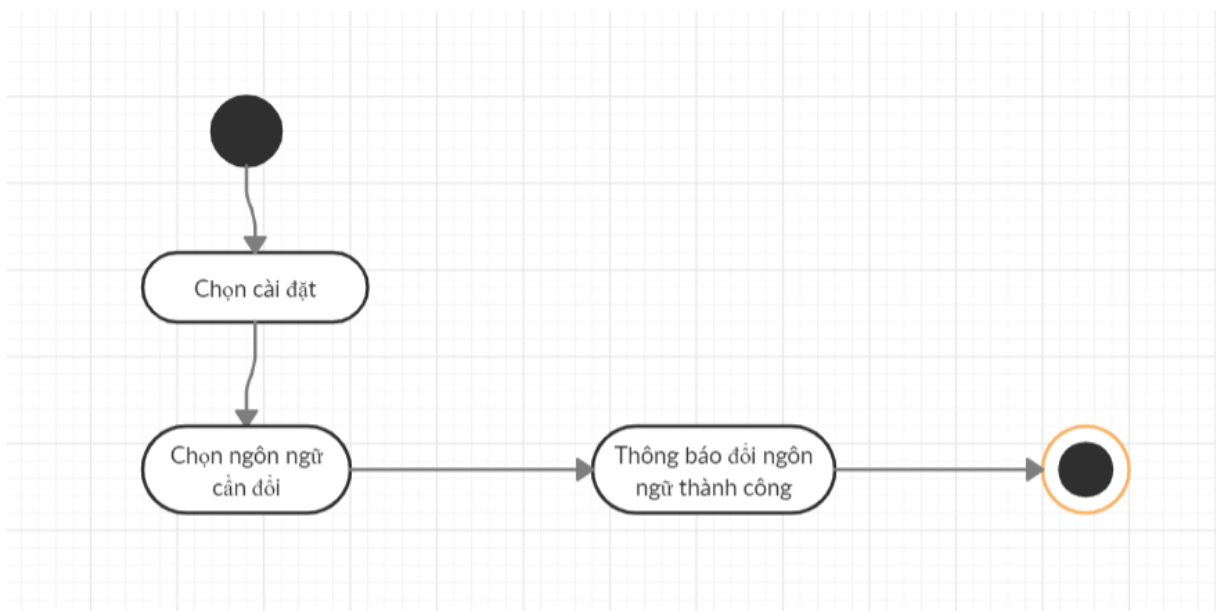


Hình 5 Sơ đồ chức năng Đặt hàng

3.3.5. Xem thông tin sản phẩm



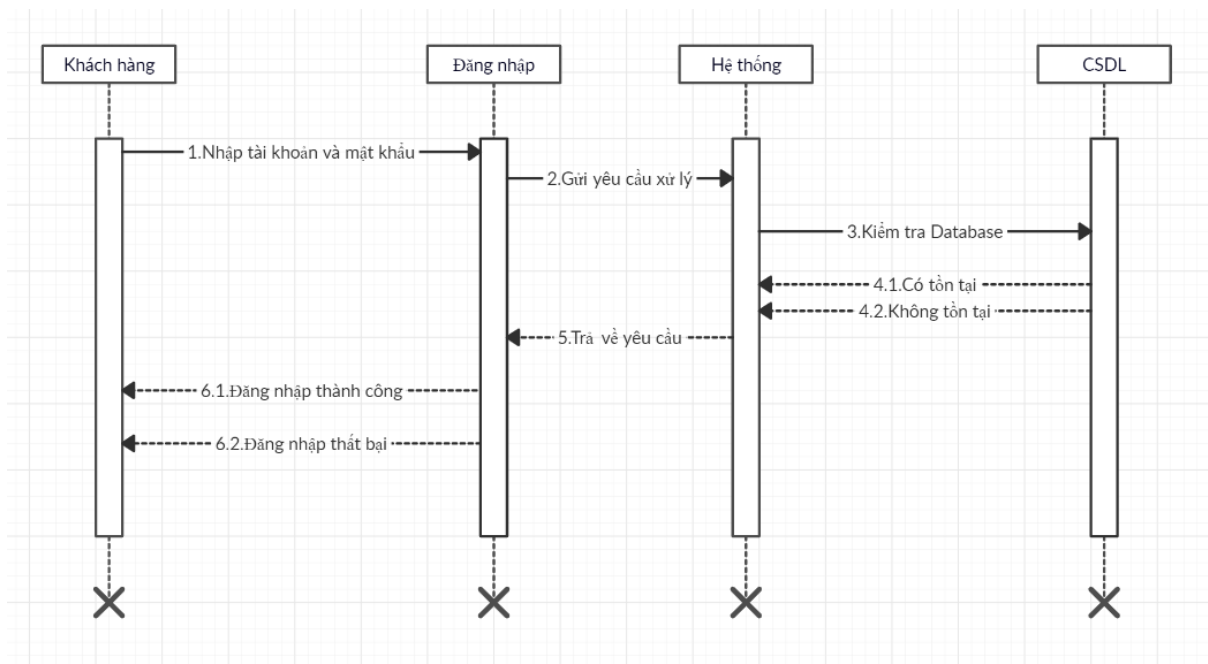
Hình 6 Sơ đồ hoạt động chức năng xem thông tin sản phẩm



Hình 7 Sơ đồ hoạt động chức năng đổi ngôn ngữ

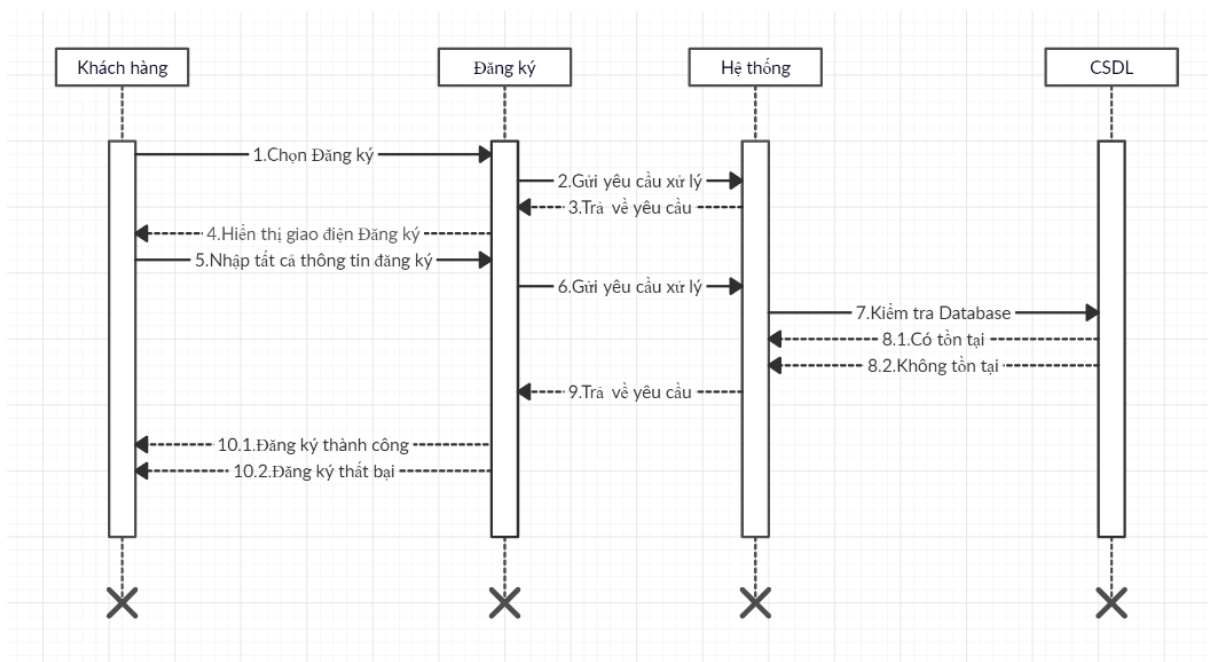
3.4. Sơ đồ tuần tự

3.4.1. Đăng nhập



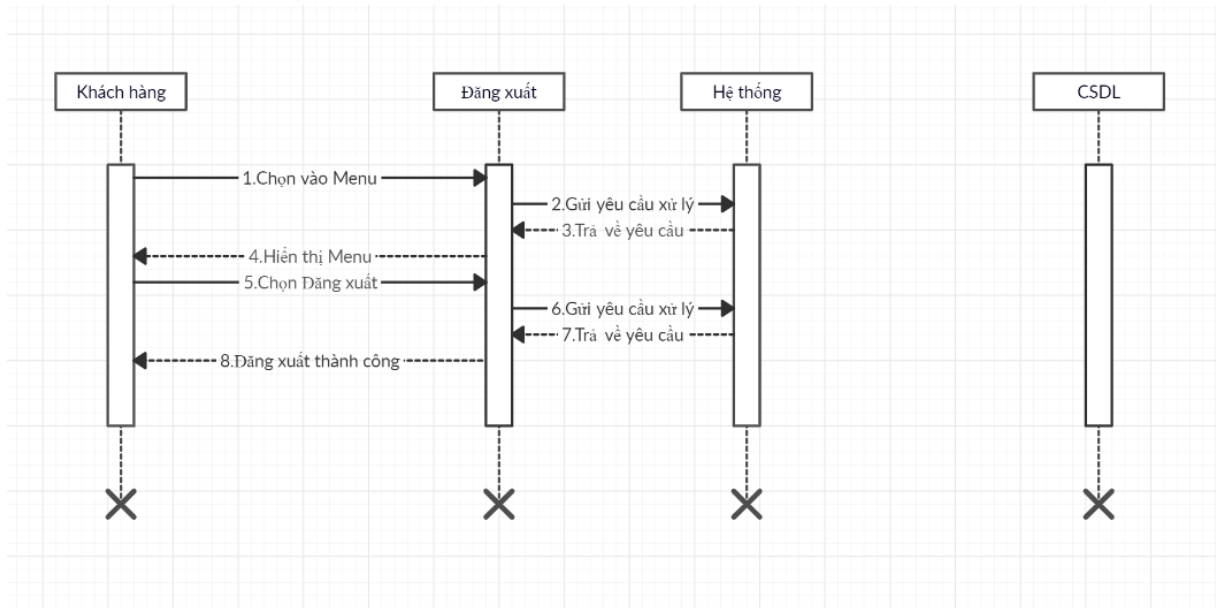
Hình 8 Sơ đồ tuần tự chức năng Đăng nhập

3.4.2. Đăng ký



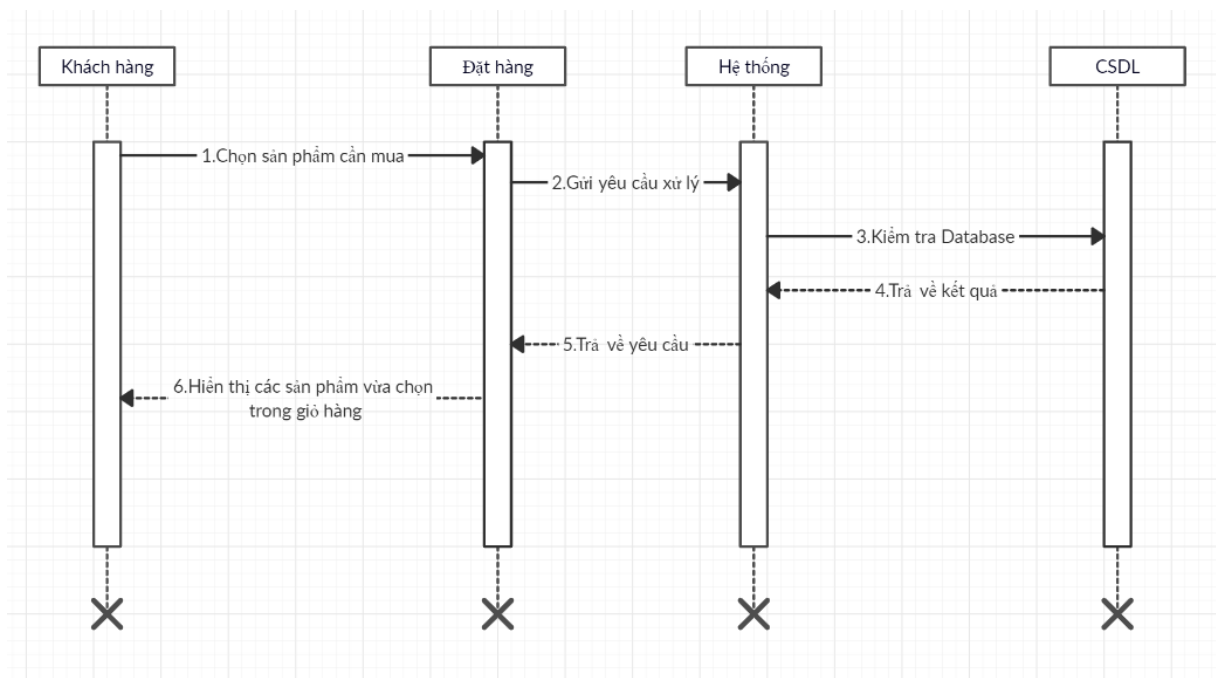
Hình 9 Sơ đồ tuần tự chức năng Đăng ký

3.4.3. Đăng xuất



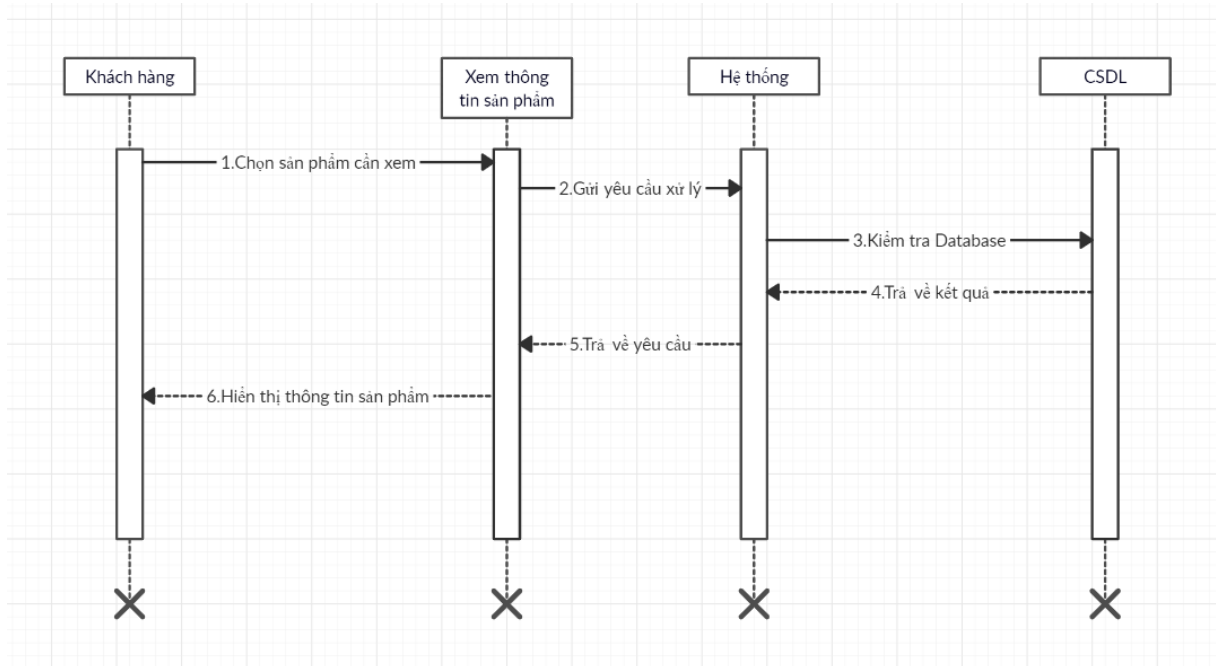
Hình 10 Sơ đồ tuần tự chức năng Đăng xuất

3.4.4. Đặt hàng

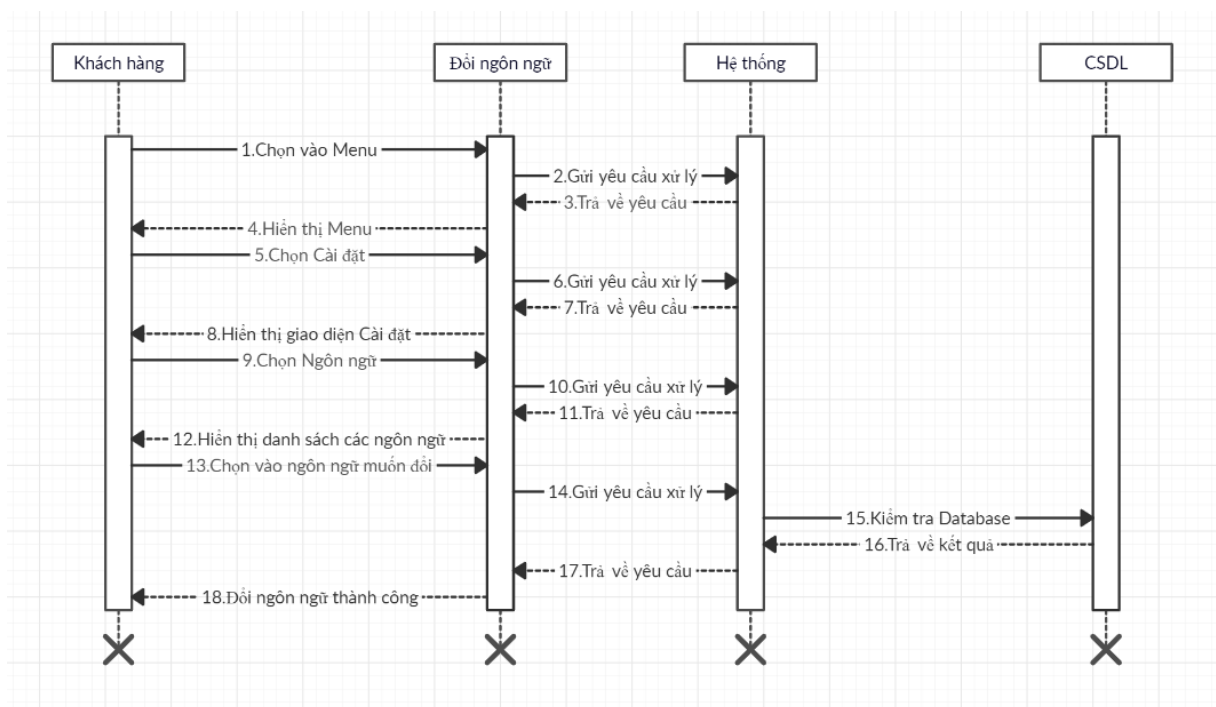


Hình 11 Sơ đồ tuần tự chức năng Đặt hàng

3.4.5. Xem thông tin sản phẩm



Hình 12 Sơ đồ tuần tự chức năng Xem thông tin sản phẩm

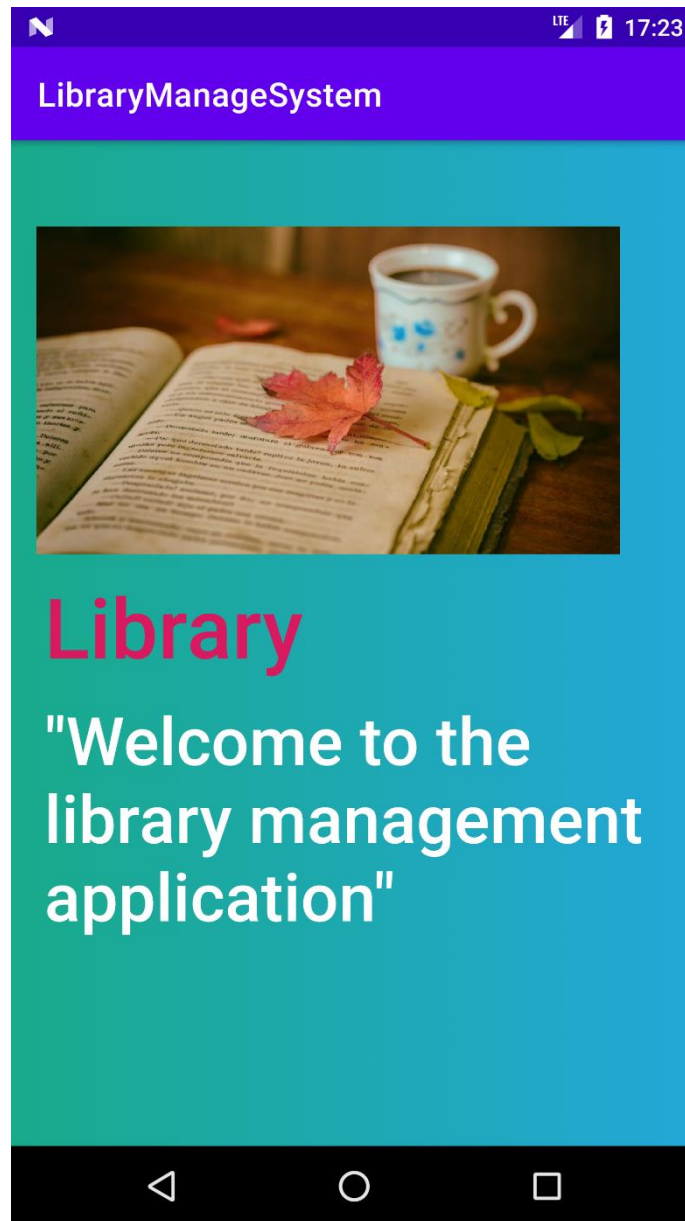


Hình 13 Sơ đồ tuần tự chức năng Đổi ngôn ngữ

CHƯƠNG 4. THIẾT KẾ VÀ XÂY DỰNG ỨNG DỤNG

4.1. Giao diện khách hàng

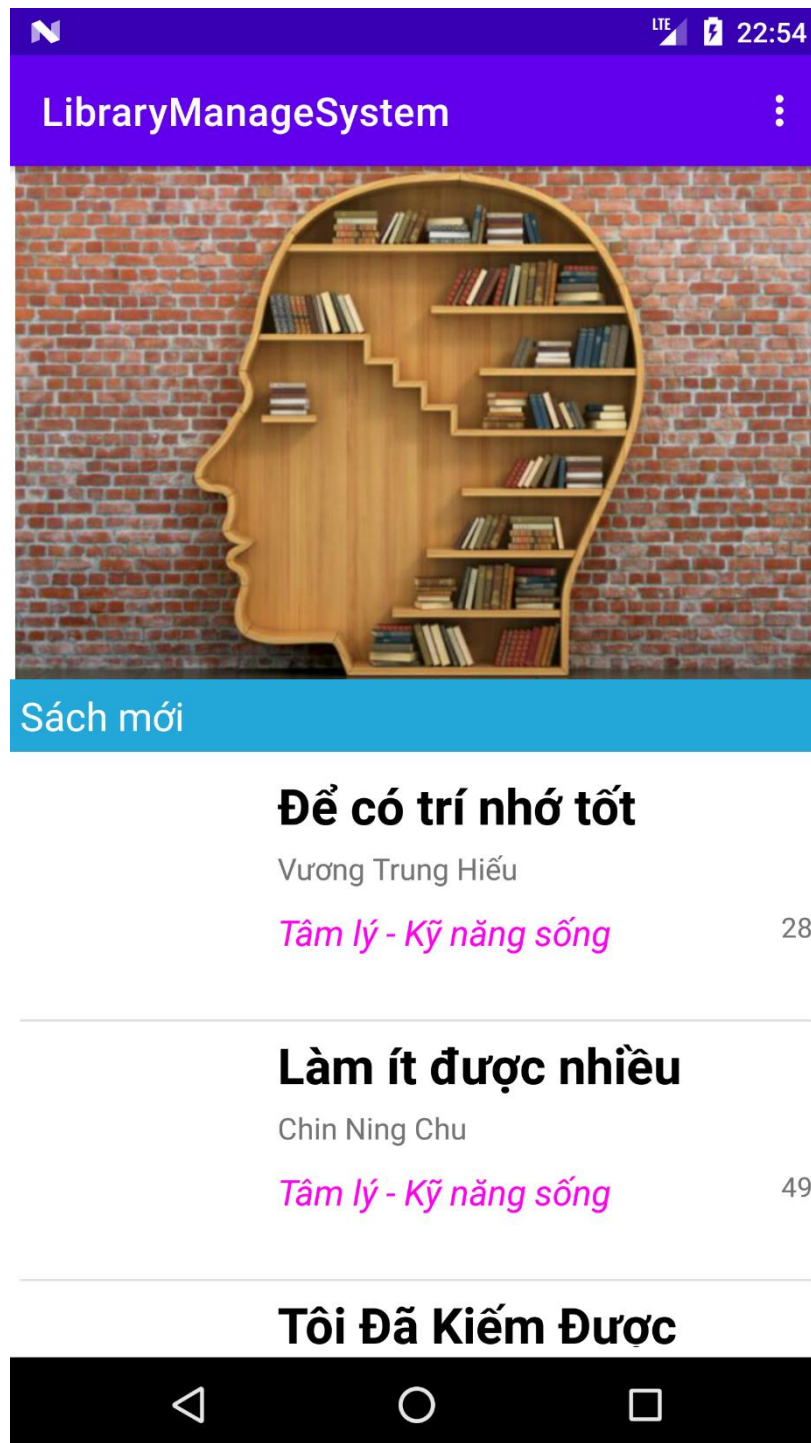
4.1.1. Giao diện trang mở đầu



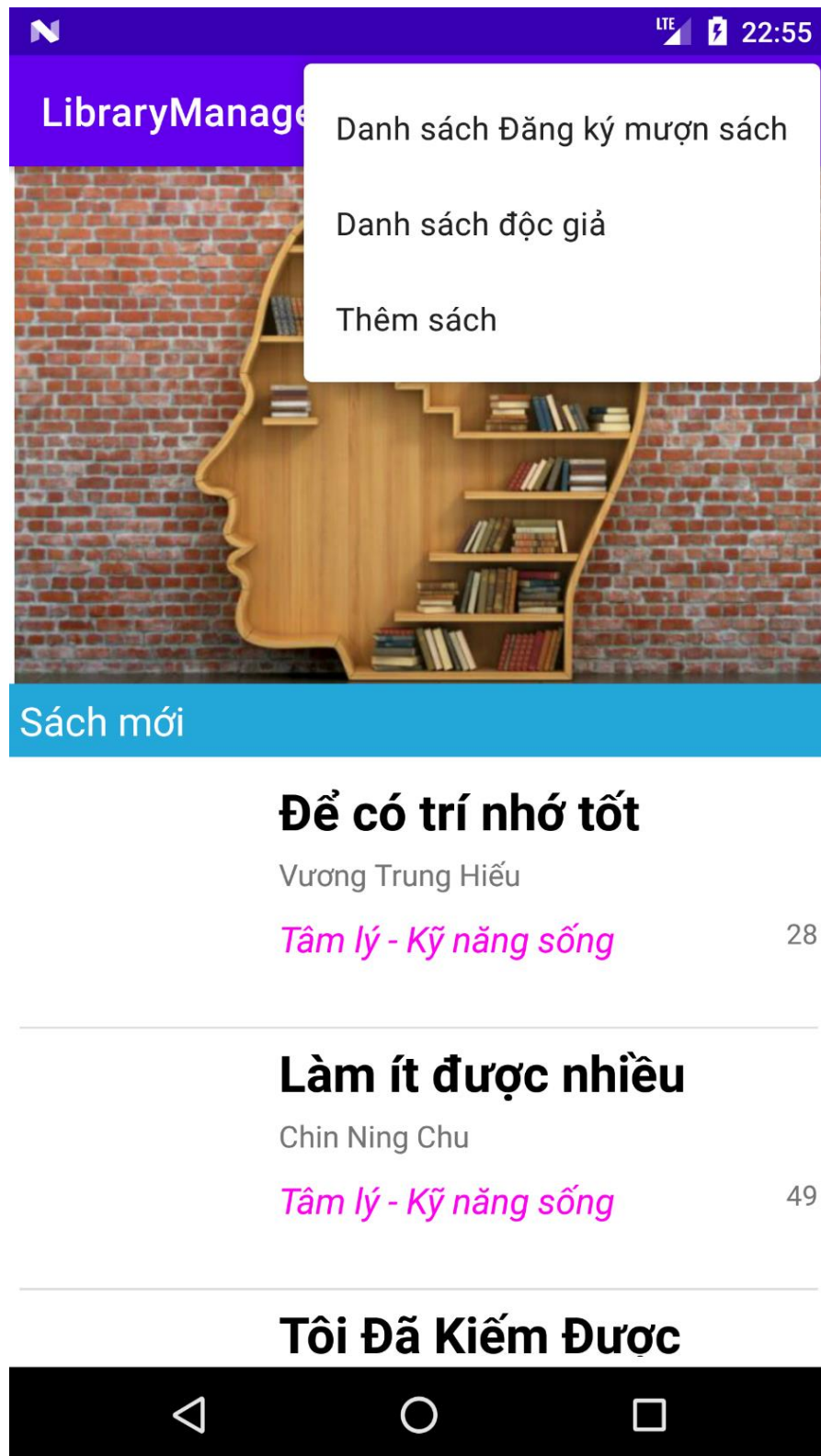
Hình 14 Giao diện trang mở đầu

4.1.2. Giao diện trang chủ

Mô tả: Khách hàng có xem thông tin các sản phẩm mới trên trang chủ của ứng dụng



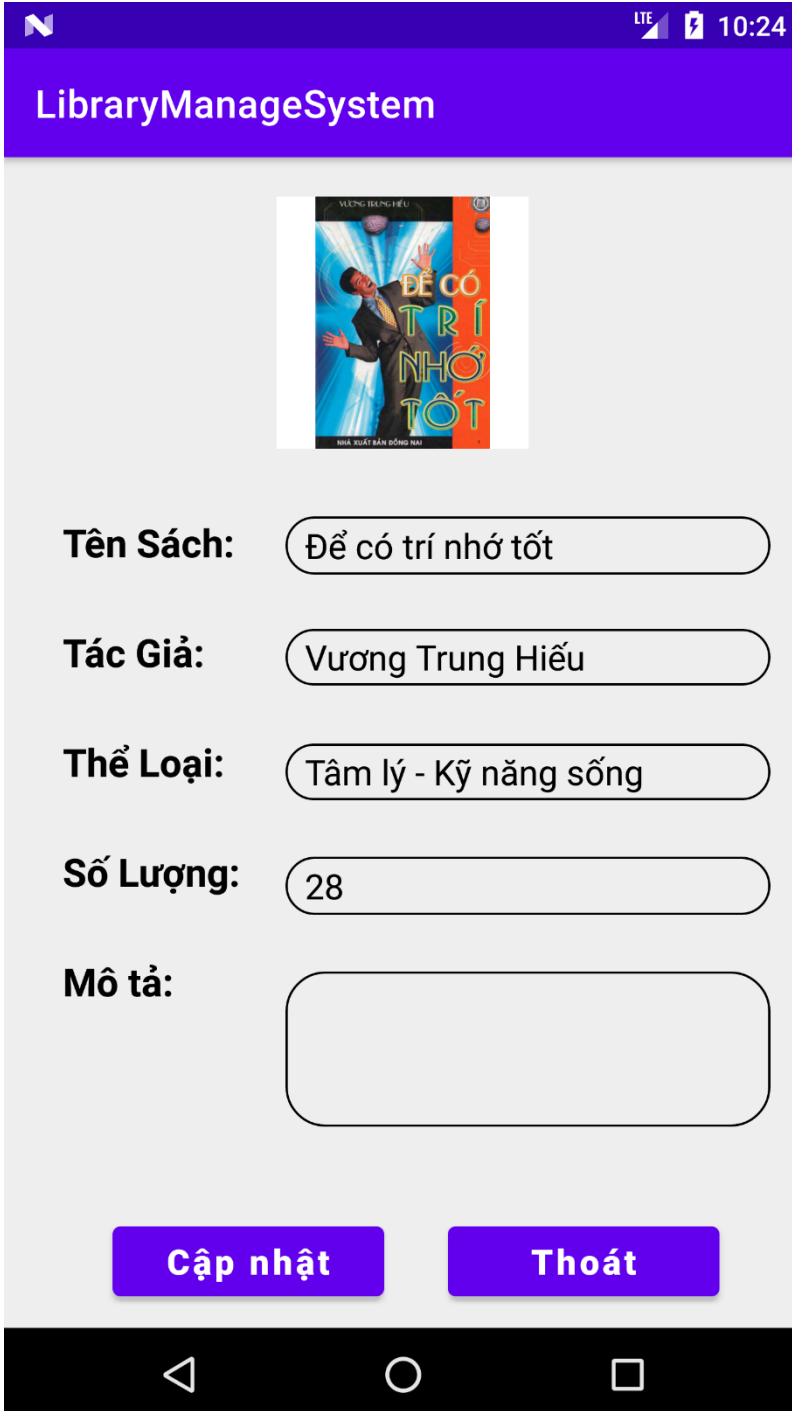
Hình 15 Giao diện trang chủ



Hình 16 Chức năng giao diện trang chủ

4.1.3. Giao diện chi tiết sản phẩm.

Mô tả: Khách hàng chọn vào sản phẩm sẽ xem được hình ảnh, mô tả chi tiết về sản phẩm và giá tiền

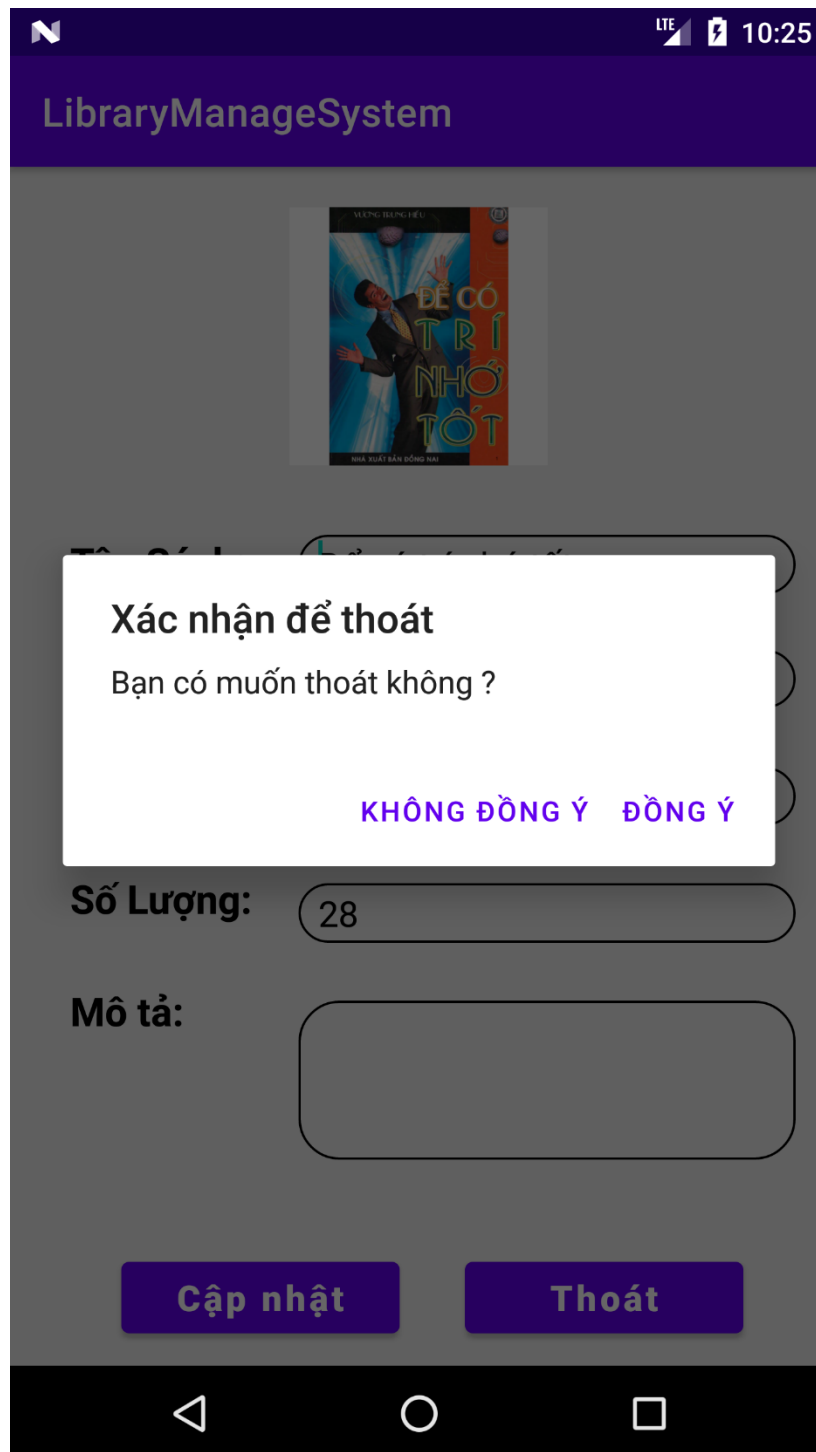


The screenshot displays the 'LibraryManageSystem' app interface on a mobile device. At the top, a purple header bar contains the app name. Below this, a book cover for 'Để có trí nhớ tốt' by Vương Trung Hiếu is shown. The book cover features a man in a suit with his arms raised against a blue and orange background. Below the cover, the book's details are listed in a form-like structure:

- Tên Sách:** Để có trí nhớ tốt
- Tác Giả:** Vương Trung Hiếu
- Thể Loại:** Tâm lý - Kỹ năng sống
- Số Lượng:** 28
- Mô tả:** (An empty text box for the description)

At the bottom of the form, there are two purple buttons: 'Cập nhật' (Update) and 'Thoát' (Exit). The entire interface is set against a light gray background, and the bottom of the screen shows the standard Android navigation bar.

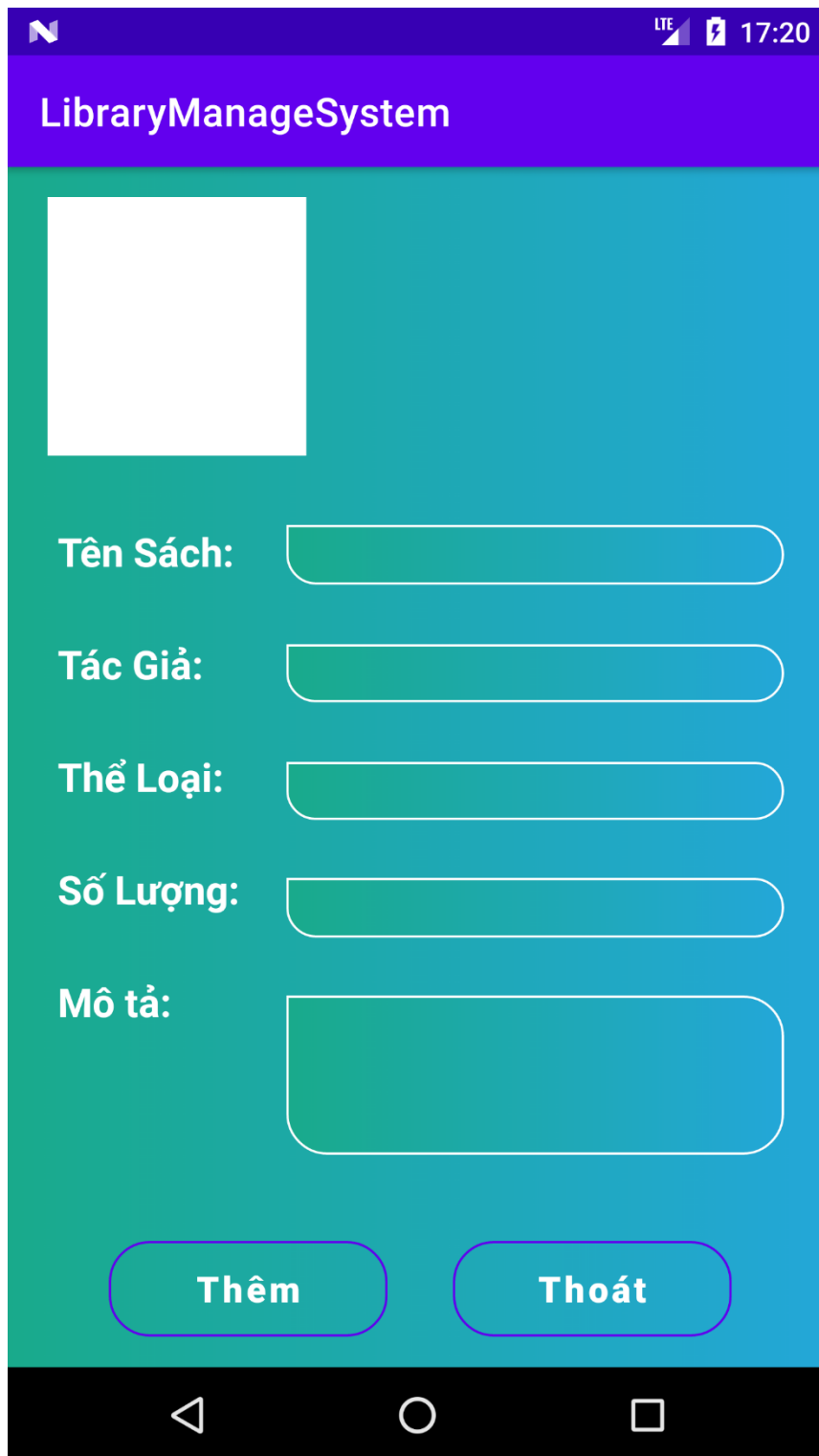
Hình 17 Giao diện trang thông tin sản phẩm



Hình 18 Giao diện đóng giao diện chi tiết sản phẩm

4.1.4. Giao diện Thêm sách

Mô tả: chọn chức năng thêm sách ở trang chủ, sau đó thêm thông tin sách cần update.



LibraryManageSystem

Tên Sách:

Tác Giả:

Thể Loại:

Số Lượng:

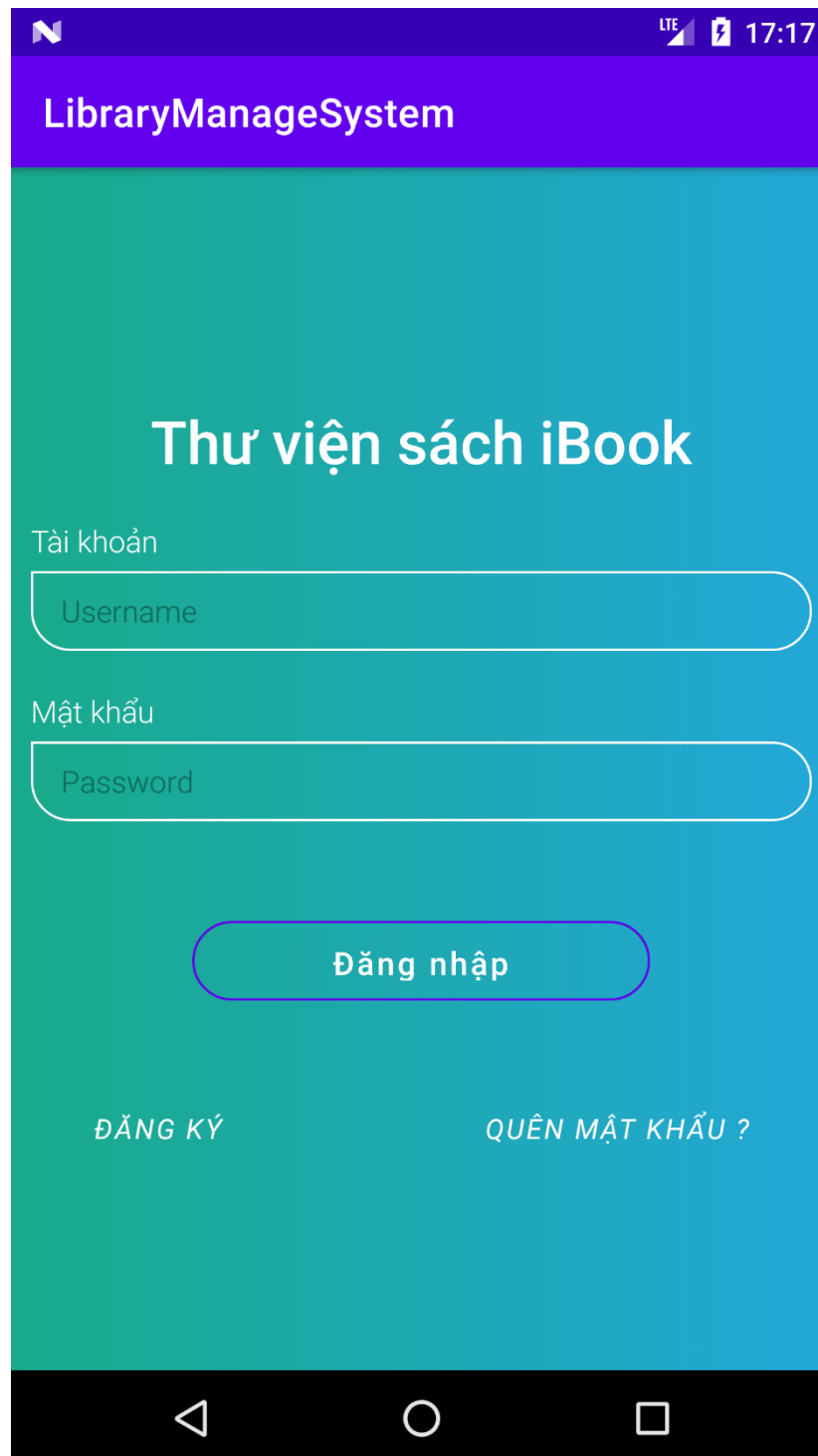
Mô tả:

Thêm Thoát

Hình 19 Giao diện thêm sách

4.1.5. Giao diện đăng nhập

Mô tả: Khách hàng nhập thông tin tài khoản đã đăng ký sau đó nhấn nút Đăng nhập để vào Trang chủ, nếu khách hàng chưa có tài khoản có thể nhấn nút đăng ký để tạo tài khoản.



LibraryManageSystem

Thư viện sách iBook

Tài khoản

Username

Mật khẩu

Password

Đăng nhập

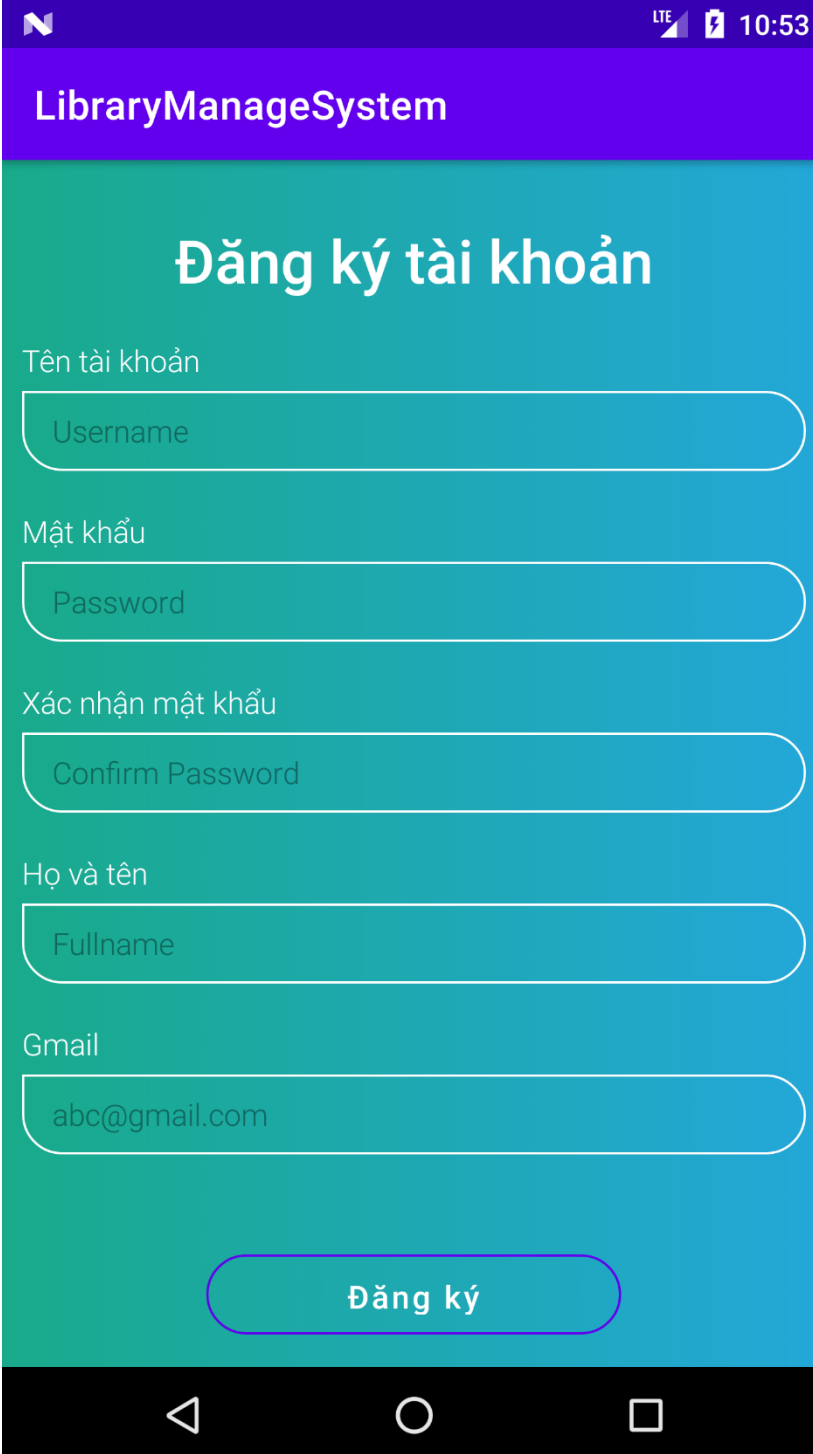
ĐĂNG KÝ

QUÊN MẬT KHẨU ?

Hình 20 Giao diện đăng nhập

4.1.6. Giao diện đăng ký

Mô tả: Khách hàng chưa có tài khoản có thể chọn nút Đăng ký sau đó nhập đầy đủ thông tin để tạo tài khoản.



The screenshot shows a mobile application interface for 'LibraryManageSystem'. The top status bar is purple with a white 'N' logo, LTE signal, battery icon, and the time 10:53. Below the status bar is a purple header with the text 'LibraryManageSystem' in white. The main content area has a teal background and is titled 'Đăng ký tài khoản' (Register account) in large white text. Below the title are five input fields, each with a label above it: 'Tên tài khoản' (Account name) with 'Username' placeholder, 'Mật khẩu' (Password) with 'Password' placeholder, 'Xác nhận mật khẩu' (Confirm password) with 'Confirm Password' placeholder, 'Họ và tên' (Full name) with 'Fullname' placeholder, and 'Gmail' with 'abc@gmail.com' placeholder. At the bottom of the form is a rounded rectangular button with a purple border and the text 'Đăng ký' (Register) in white. The bottom of the screen shows the standard Android navigation bar with back, home, and recent apps icons.

Hình 21 Giao diện Đăng ký

4.1.7. Giao diện quên mật khẩu

LibraryManageSystem

Quên Mật khẩu

Gmail

abc@gmail.com

Tên tài khoản

Username

Gửi

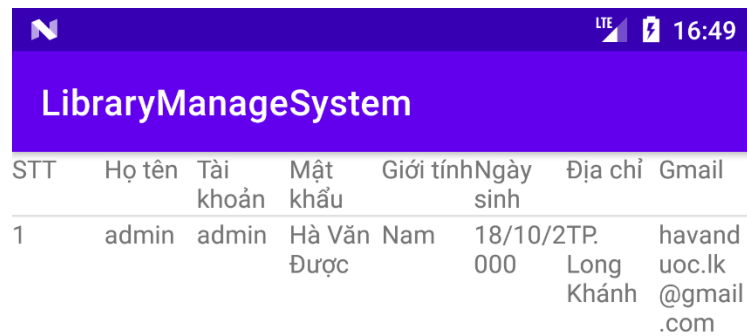
QUAY LẠI

Hình 22 Giao diện Quên mật khẩu

4.2. Giao diện quản lý

4.2.1. Giao diện trang quản trị người dùng

Mô tả: Hiển thị danh sách tài khoản của khách hàng. Admin có thể chọn Xóa tài khoản khách hàng



The screenshot shows a mobile application interface with a purple header bar containing the text "LibraryManageSystem". Below the header is a table with the following columns: STT, Họ tên, Tài khoản, Mật khẩu, Giới tính, Ngày sinh, Địa chỉ, and Gmail. The table contains one row of data for an admin user.

STT	Họ tên	Tài khoản	Mật khẩu	Giới tính	Ngày sinh	Địa chỉ	Gmail
1	admin	admin	Hà Văn Được	Nam	18/10/2000	TP. Long Khánh	havanduc.lk@gmail.com



Hình 23 Giao diện danh sách độc giả

CHƯƠNG 5: KẾT LUẬN

5.1. Kết quả đạt được

Sau một thời gian tập trung triển khai đề tài, em đã hoàn thành được đề tài *Xây dựng ứng dụng di động quản lý sách iBook* với giao diện tương đối ổn, các thao tác sử dụng dễ dàng, thân thiện với người dùng.

Sau khi hoàn thành xong được ứng dụng bán hàng Online, em có thêm những kỹ năng nền tảng để xây dựng được một ứng dụng bằng Android Studio, SQLite, Java và sử dụng những kiến thức đã học vào việc phát triển ứng dụng di động. Bên cạnh đó, em cũng được mở rộng thêm các kiến thức chuyên ngành mới nhờ việc tìm hiểu một số kiến thức lập trình để áp dụng vào việc phát triển ứng dụng này. Nhờ vậy, kỹ năng tự học và vận dụng các kỹ năng mới của em được nâng cao hơn.

Khi thực hiện và hoàn thành dự án, em cũng đã biết thêm được nhiều kinh nghiệm quý giá khi xây dựng ứng dụng nói riêng và phát triển phần mềm. Những kỹ năng này sẽ nền tảng để giúp em nâng cao trình độ bản thân, kinh nghiệm trong thực tế để có thể làm việc ngoài các doanh nghiệp sau này.

5.2. Hạn chế của đề tài

Bên cạnh những kết quả khả quan đạt được chúng em còn nhận thấy những tồn tại hạn chế sau:

- Giao diện người dùng chưa được đẹp mắt.
- Tốc độ xử lý chưa cao.
- Vẫn còn phát sinh một số lỗi khi vận hành hệ thống.

5.3. Hướng phát triển của đề tài

Sau khi hoàn thành đề tài *Xây dựng ứng dụng di động quản lý sách iBook*, em sẽ tiếp tục nghiên cứu và phát triển ứng dụng nhằm cải thiện giao diện người dùng, nâng cấp các chức năng và tối ưu hoá tốc độ xử lý để đem lại hiệu quả cao hơn, phát triển để đưa vào ứng dụng thực tế.

TÀI LIỆU THAM KHẢO

Tiếng Việt

1. Bryan Syverson, Joel Murach, *SQL hướng dẫn học qua ví dụ*, NXB Khoa học và Kỹ thuật, 2013.

Tiếng Anh

2. Floyd Sally, Jacobson Van (1993), *Random Early Detection gateways for Congestion Avoidance*, IEEE/ACM Transactions on Networking.
3. Shankland, Stephen (ngày 12 tháng 11 năm 2007). “Google's Android parts ways with Java industry group”. CNET News.

Website

5. <https://hiepsiit.com/> Truy cập ngày 13/07/2020
6. <http://www.w3schools.com/>. Truy cập ngày 11/07/2020