**BỘ GIÁO DỤC VÀ ĐÀO TẠO**

**TRƯỜNG ĐẠI HỌC NÔNG LÂM TP HCM**

**KHOA CÔNG NGHỆ THÔNG TIN**

LUẬN VĂN TỐT NGHIỆP

ĐỀ TÀI:

XÂY DỰNG ỨNG DỤNG THƯƠNG MẠI ĐIỆN TỬ CHO NHÃN HÀNG

THỜI TRANG TARIFA TRÊN NỀN TẢNG ANDROID VÀ RESTFUL WEBSERVICE

**Ngành : Công Nghệ Thông Tin**

**Niên khóa : 2013 - 2017**

**Lớp : DH13DT**

**SVTH : Nguyễn Văn Công**

TP. HỒ CHÍ MINH, tháng 7 năm 2017

**BỘ GIÁO DỤC VÀ ĐÀO TẠO**

**TRƯỜNG ĐẠI HỌC NÔNG LÂM TP HCM**

**KHOA CÔNG NGHỆ THÔNG TIN**

LUẬN VĂN TỐT NGHIỆP

ĐỀ TÀI:

XÂY DỰNG ỨNG DỤNG THƯƠNG MẠI ĐIỆN TỬ CHO NHÃN HÀNG

THỜI TRANG TARIFA TRÊN NỀN TẢNG ANDROID VÀ RESTFUL WEBSERVICE

**Ngành : Công nghệ thông tin**

**Niên khóa : 2013 - 2107**

**Lớp : DH13DT**

**SVTH : Nguyễn Văn Công**

TP. HỒ CHÍ MINH, tháng 7 năm 2017

**CÔNG TRÌNH HOÀN TẤT TẠI**

**TRƯỜNG ĐẠI HỌC NÔNG LÂM TP HCM**

Cán bộ hướng dẫn: Thạc sĩ Mai Anh Thơ

Cán bộ phản biện: Thạc sĩ Lê Phi Hùng

Luận văn cử nhân được bảo vệ tại HỘI ĐỒNG CHẤM LUẬN VĂN CỬ NHÂN TRƯỜNG ĐẠI HỌC NÔNG LÂM TP HCM ngày tháng năm

*Nhận xét của giáo viên hướng dẫn*

*Nhận xét của giáo viên phản biện*

**NHIỆM VỤ LUẬN VĂN CỬ NHÂN**

**CỘNG HOÀ XÃ HỘI CHỦ NGHĨA VIỆT NAM**

**Độc Lập-Tự Do-Hạnh Phúc**

**BỘ GIÁO DỤC VÀ ĐÀO TẠO**

**TRƯỜNG ĐH NÔNG LÂM TPHCM**

Phái: **Nam**

Nơi sinh: Quảng Nam

Ngành: Công nghệ thông tin

Họ tên sinh viên: **NGUYỄN VĂN CÔNG**

Ngày tháng năm sinh: 14/06/1994

Chuyên ngành: Công nghệ thông tin

1. TÊN ĐỀ TÀI: **Xây dựng ứng dụng thương mại điện tử cho nhãn hàng thời trang Tarifa trên nền tảng Android và Webservice - Restful**
2. NHIỆM VỤ VÀ NỘI DUNG

* Nhiệm vụ: Xậy dựng một ứng dụng điện thoại di động trên nền tảng Android đáp ứng nhu cầu mua hàng trực tuyến của khách hàng trên điện thoại.
* Nội dung: Nghiên cứu lâp trình Android, Spring Java để tạo ra một ứng dụng thương mại điện tử cho nhãn hàng thời trang Tarifa.

1. NGÀY GIAO NHIỆM VỤ: **03/03/2017**
2. NGÀY HOÀN THÀNH NHIỆM VỤ: **25/08/2017**
3. HỌ VÀ TÊN CÁN BỘ HƯỚNG DẪN: Thạc sĩ Mai Anh Thơ

Ngày / /

CÁN BỘ HƯỚNG DẪN

(Ký và ghi rõ họ tên)

Ngày / /

KHOA CNTT

(Ký và ghi rõ họ tên)

Ngày / /

CÁN BỘ PHẢN BIỆN

(Ký và ghi rõ họ tên)

**LỜI CẢM ƠN**

Trong bốn năm học tập và rèn luyện tại trường đại học Nông Lâm thành phố Hồ Chí Minh, chúng em được sự dạy dỗ chân thành và tận tâm từ thành cô. Chúng em đã tích lũy được nhiều kiến thức bổ ích và các kỹ năng cần thiết cho cuộc sống.

Đầu tiên, chúng em xin bày tỏ lòng biết ơn sâu sắc đến Cô ThS.Mai Anh Thơ. Cô đã tận tình hướng dẫn, truyền đạt kiến thức và kinh nghiệm cho chúng em trong suốt quá trình thực hiện.

Tiếp theo, chúng em xin chân thành cảm ơn quý Thầy Cô Khoa công nghệ thông tin trường đại học Nông Lâm thành phố Hồ Chí Minh đã truyền đạt kiến thức cho chúng em trong bốn năm học tập và tạo điều kiện thuận lợi cho chúng em hoàn thành khóa luận này.

Bên cạnh đó, chúng mình xin gửi lời cảm ơn đến các bạn lớp DH13DT đã chia sẻ và giúp đỡ chúng mình trong suốt quá trình học tập.

Cuối cùng, chúng con xin chân thành cảm ơn cha mẹ đã sinh chúng con, nuôi dưỡng, tạo mọi điều kiện vật chất lẫn tinh thần cho chúng con học tập đến ngày hôm nay.

Trong quá trình thực hiện, mặc dù chúng em đã cố gắng tìm hiểu, học tập rất nhiều nhưng chúng em cũng không tránh khỏi những thiếu sót. Kính mong sự thông cảm và chỉ bảo của Thầy Cô.

Chúng em xin kính chúc cha mẹ, Thầy Cô tràn đầy sức khỏe, hạnh phúc và làm việc tốt.

Chúng em xin chân thành cảm ơn!

Sinh viên thực hiện

Nguyễn Văn Công

DANH SÁCH CHỮ VIẾT TẮT

JPA Java Persistence API

SQL Structured Query Language

JSON Java Object notation

HTTP Hypertext Transfer Protocol

MVC Model View Controller

REST Representation State Transfer Application Program Interface

MVP Model View Presenter

SQL Structured Query Language

GSM Global System for Mobile Communications

XML Extensible Markup Language

DANH MỤC CÁC HÌNH

[Hình 3.1 Biểu tượng Android. 18](file:///C:\Users\Cong\Desktop\FinalProject\trunk\document\LVTN_v7.docx#_Toc492719708)

[Hình 3.2 Điện thoại Android. HTC Dream (trái), HTC Magic ( Phải) 22](file:///C:\Users\Cong\Desktop\FinalProject\trunk\document\LVTN_v7.docx#_Toc492719709)

[Hình 3.3 Kiến trúc Android 23](file:///C:\Users\Cong\Desktop\FinalProject\trunk\document\LVTN_v7.docx#_Toc492719710)

[Hình 3.4 Vòng đời của Activity 29](file:///C:\Users\Cong\Desktop\FinalProject\trunk\document\LVTN_v7.docx#_Toc492719711)

[Hình 3.5 Các trạng thái của chu kì sống của Activity 30](file:///C:\Users\Cong\Desktop\FinalProject\trunk\document\LVTN_v7.docx#_Toc492719712)

[Hình 3.6 Sơ đồ chuyển trạng thái của Service 33](file:///C:\Users\Cong\Desktop\FinalProject\trunk\document\LVTN_v7.docx#_Toc492719713)

[Hình 3.7 Biểu đồ dữ liệu 35](#_Toc492719714)

[Hình 3.8 Hàm của Webservice 37](file:///C:\Users\Cong\Desktop\FinalProject\trunk\document\LVTN_v7.docx#_Toc492719715)

[Hình 3.9 Sequence Diagram Webservice 39](file:///C:\Users\Cong\Desktop\FinalProject\trunk\document\LVTN_v7.docx#_Toc492719716)

[Hình 3.10 Class Diagram Webservice 40](file:///C:\Users\Cong\Desktop\FinalProject\trunk\document\LVTN_v7.docx#_Toc492719717)

[Hình 3.11 Usecase ứng dụng 41](#_Toc492719718)

[Hình 3.12 Luồng Activity 42](#_Toc492719719)

[Hình 3.13 Mô hình MVP 43](#_Toc492719720)

[Hình 3.15 Quản lí các tài khoản đăng nhập 44](file:///C:\Users\Cong\Desktop\FinalProject\trunk\document\LVTN_v7.docx#_Toc492719721)

[Hình 3.14 Đăng nhập với tài khoản của các trang xã hội 44](file:///C:\Users\Cong\Desktop\FinalProject\trunk\document\LVTN_v7.docx#_Toc492719722)

[Hình 3.16 Cấu trúc lưu database của một tài khoản trên Firebase 45](file:///C:\Users\Cong\Desktop\FinalProject\trunk\document\LVTN_v7.docx#_Toc492719723)

[Hình 3.17 Cấu trúc lưu database review của một sản phẩm trên Firebase 46](#_Toc492719724)

[Hình 3.18 Cấu trúc lưu database của các đơn đặt hàng từ khách hàng trên Firebase 46](#_Toc492719725)

[Hình 4.1 Màn hình đăng nhập 49](file:///C:\Users\Cong\Desktop\FinalProject\trunk\document\LVTN_v7.docx#_Toc492719726)

[Hình 4.2 Màn hình chính 50](file:///C:\Users\Cong\Desktop\FinalProject\trunk\document\LVTN_v7.docx#_Toc492719727)

[Hình 4.3 Màn hình xem sản phẩm 51](file:///C:\Users\Cong\Desktop\FinalProject\trunk\document\LVTN_v7.docx#_Toc492719728)

[Hình 4.4 Màn hình danh mục riêng 52](file:///C:\Users\Cong\Desktop\FinalProject\trunk\document\LVTN_v7.docx#_Toc492719729)

[Hình 4.5 Màn hình mua hàng 53](file:///C:\Users\Cong\Desktop\FinalProject\trunk\document\LVTN_v7.docx#_Toc492719730)

DANH MỤC CÁC BẢNG

[Bảng 2.1: Bảng so sách các ứng dụng thời trang. 17](#_Toc492721517)

[Bảng 6.1 Mô tả usecase đăng nhập i](#_Toc492721518)

[Bảng 6.2 Mô tả usecase tìm kiếm ii](#_Toc492721519)

[Bảng 6.3 Mô tả usecase quản lý danh sách sản phẩm yêu thích ii](#_Toc492721520)

[Bảng 6.4 Mô tả usecase đánh giá sản phẩm ii](#_Toc492721521)

MỤC LỤC

[DANH SÁCH CHỮ VIẾT TẮT ix](#_Toc492721589)

[DANH MỤC CÁC HÌNH ix](#_Toc492721590)

[DANH MỤC CÁC BẢNG xi](#_Toc492721591)

[MỤC LỤC xii](#_Toc492721592)

[Chương 1 Mở đầu 15](#_Toc492721593)

[1.1 Lý do chọn đề tài 15](#_Toc492721594)

[1.2 Mục đích, đối tượng và phạm vi nghiên cứu 16](#_Toc492721595)

[1.2.1 Mục đích 16](#_Toc492721596)

[1.2.2 Đối tượng nghiên cứu 16](#_Toc492721597)

[1.2.3 Phạm vi nghiên cứu 16](#_Toc492721598)

[Chương 2 Tổng quan 17](#_Toc492721599)

[2.1 Khảo sát các ứng dụng thời trang 17](#_Toc492721600)

[2.2 Kết luận, đề xuất giải pháp cho bài toán 18](#_Toc492721601)

[2.2.1 Kết luận 18](#_Toc492721602)

[2.2.2 Giải pháp 18](#_Toc492721603)

[Chương 3 Nội dung nghiên cứu 19](#_Toc492721604)

[3.1 Hệ điều hành Android 19](#_Toc492721605)

[3.1.1 Giới thiệu Android 19](#_Toc492721606)

[3.1.2 Khái niệm Android 19](#_Toc492721607)

[3.1.3 Khả năng Android 20](#_Toc492721608)

[3.1.4 Đặc tính 21](#_Toc492721609)

[3.1.5 Lịch sử Android 22](#_Toc492721610)

[3.1.6 Kiến trúc Android 24](#_Toc492721611)

[3.1.7 Android Runtime & Libraries 25](#_Toc492721612)

[3.1.8 Linux Kernel 26](#_Toc492721613)

[3.2 Lập trình Android 27](#_Toc492721614)

[3.2.1 Máy ảo Dalvik 27](#_Toc492721615)

[3.2.2 Các gói Java 28](#_Toc492721616)

[3.2.3 Các gói Android 28](#_Toc492721617)

[3.2.4 Các thành phần quan trọng trong ứng dụng 29](#_Toc492721618)

[3.3 Xác định yêu cầu 36](#_Toc492721619)

[3.3.1 Yêu cầu chức năng 36](#_Toc492721620)

[3.3.2 Yêu cầu hệ thống 36](#_Toc492721621)

[3.4 Mô hình biểu đồ dữ liệu chung 36](#_Toc492721622)

[3.5 Webservice 37](#_Toc492721623)

[3.5.1 Phân tích yêu cầu 37](#_Toc492721624)

[3.5.2 Hướng xây dựng Webservice 37](#_Toc492721625)

[3.5.3 Chi tiết từng phương thức 38](#_Toc492721626)

[3.5.4 Sequence Diagram chung 40](#_Toc492721627)

[3.5.5 Class Diagram 41](#_Toc492721628)

[3.6 Ứng dụng 41](#_Toc492721629)

[3.6.1 Tổng quan 41](#_Toc492721630)

[3.6.2 Usecase 42](#_Toc492721631)

[3.6.3 Sơ đồ luồng Activity 42](#_Toc492721632)

[3.6.4 Mô hình MVP 44](#_Toc492721633)

[3.7 Firebase 45](#_Toc492721634)

[3.7.1 Authentication 45](#_Toc492721635)

[3.7.2 Realtime database 46](#_Toc492721636)

[3.8 Một số thư viện quang trọng 48](#_Toc492721637)

[3.8.1 Retrofit 48](#_Toc492721638)

[3.8.2 Picasso 48](#_Toc492721639)

[Chương 4 Kết quả đạt được 50](#_Toc492721640)

[4.1 Màn hình đăng nhập 50](#_Toc492721641)

[4.2 Màn hình chính 51](#_Toc492721642)

[4.3 Màn hình xem sản phẩm 52](#_Toc492721643)

[4.4 Màn hình danh mục riêng 53](#_Toc492721644)

[4.5 Màn hình mua hàng 54](#_Toc492721645)

[Chương 5 Kết luận và kiến nghị 55](#_Toc492721646)

[5.1 Các kết luận 55](#_Toc492721647)

[5.2 Các hạn chế 55](#_Toc492721648)

[5.3 Các đề xuất mới 55](#_Toc492721649)

[Chương 6 TÀI LIỆU THAM KHẢO 57](#_Toc492721650)

[PHỤ LỤC i](#_Toc492721651)

[Phụ lục: Mô tả usecase i](#_Toc492721652)

# Mở đầu

## Lý do chọn đề tài

Hiện nay ngành công nghiệp phần mềm đang rất phát triển và ngành công nghiệp phần mềm điện thoại di động cũng không nằm ngoài xu thế đó. Tuy còn nhiều hạn chế trong phần cứng của điện thoại di động nhưng nó đã thể hiện được sức mạnh của mình trong rất nhiều các lĩnh vực khác nhau như giải trí, các tiện ích, thanh toán điện tử v.v… Ngành công nghệ phần mềm di động vẫn còn khá non trẻ ở Việt Nam vì vậy việc nghiên cứu và phát triển các ứng dụng trên di động là rất cần thiết.

Xuất phát từ nhu cầu thực tế đó, Google đã mạnh dạn nghiên cứu và đưa ra một nền tảng hệ điều hành mới, đó chính là Android. Một nền tảng hệ điều hành mã nguồn mở hoàn toàn và dựa vào một ngôn ngữ lập trình rất mạnh là Java để phát triển ứng dụng. Chính vì Android hỗ trợ Java ngay từ trong nền tảng của nó, nên việc viết ứng dụng, ngay cả sử dụng các chức năng của phần cứng là hoàn toàn dễ dàng thông qua việc sử dụng bộ công cụ hỗ trợ lập trình được cung cấp hoàn toàn miễn phí là Android SDK.

Ngày nay, với sự phát triển nhanh chóng của xã hội, nhu cầu cập nhật tin tức xã hội mọi lúc mọi nơi là rất cần thiết. Ngoài những trang mạng xã hội như Facebook, Instagram thì các ứng dụng thời trang (fashion app) còn là một phần không thể thiếu trong cuộc sống của các tín đồ thời trang. Cùng với sự phát triển của smartphone, các ứng dụng thời trang ngày càng tích hợp nhiều chức năng hơn. Song song đó, các fashion app này còn đem lại những gợi ý phối đồ hoàn toàn mới cho các tín đồ thời trang. Chính vì vậy nhóm thực hiện chọn đề tài “Xây dựng ứng dụng thương mại điện tử thời trang trên nền tảng Android và Webservice – Restful” với mục đích nghiên cứu, tìm hiểu hệ điều hành Android và Webservice – Restful, nhằm hiểu rõ về các kỹ thuật lập trình trên thiết bị di động.

## Mục đích, đối tượng và phạm vi nghiên cứu

### Mục đích

Mục đích quan trọng của việc thực hiện đề tài này là xây dựng ứng dụng thương mại điện tử cho nhãn thời trang Tarifa trên nền tảng android và webservice. Bên cạnh đó, thực hiện đề tài nhằm đạt được các mục tiêu chi tiết như sau:

* Tìm hiểu các ứng dụng thời trang như Vivid linen, Fynd, Myntra, Flipkart, cách thức thanh toán hiển thị sản phẩm trên điện thoại như thề nào.
* Xây dựng giao diện người dùng theo phong cách vintage trên điện thoại android.
* Xử lý back-end bằng ngôn ngữ Java, Spring framework.

### Đối tượng nghiên cứu

Các ứng dụng mua sắm thời trang trực tuyến trên điện thoại thông minh sử dụng hệ điều hành android như Fynd, Myntra, Flipkart.

Ngoài ra, nội dung cần phải nghiên cứu một số vấn đề sau:

* + Lập trình di động trên Android.
  + Framework để tạo webservice Restful.
  + Service cung cấp giải pháp thời gian thực.

### Phạm vi nghiên cứu

Để đáp ứng được các mục đích trên, đề tài đưa ra phạm vi nghiên cứu sau:

* Đề tài tập trung tìm hiểu các tính năng nổi bật của các ứng dụng trên nền tảng Android.
* Đề tài tìm hiểu và áp dụng Spring framework, JPA, MySQL để tạo web serive.
* Nghiên cứu Firebase google serivce để đáp ứng cho việc tương tác thời gian thực.

# Tổng quan

## Khảo sát các ứng dụng thời trang

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Fynd** | **Myntra** | **Flipkart** |
| **Giao diện** | * Cách bố trí hình ảnh hợp lí, tốc độ xử lý nhanh. * Hình ảnh có độ phân giải cao giúp người xem dễ dàng quan sát được chất liệu vải. Đặc biệt, tại mỗi vị trí hiển thị, hình ảnh sẽ có độ phân giải khác nhau. * Cách bố trí các thành phần của giao diện hợp lý giúp người xem dễ dàng tìm kiếm trang phục yêu thích. | * Giao diện đẹp tốc độ xử lý trung bình. * Các thành phần bố trí hợp lý. * Đưa ra gợi ý trang phục dựa trên các trang phục đã xem trước đó | * Giao diện tương đối đơn giản, tốc độ xử lý trung bình. * Bán hỗn hợp mặt hàng không chỉ riêng thời trang. |
| **Đặt hàng, thanh toán** | * Có luồng thực hiện thanh toán chi tiết giúp người đặt hàng xác định được thông tin trang phục trước khi quyết định mua hàng * Ứng dụng có tích hợp cổng thanh toán như: PayPal hoặc thanh toán bằng VISA. | * . Có luồng thực hiện thanh toán chi tiết giúp người đặt hàng xác định được thông tin trang phục trước khi quyết định mua hàng   Ứng dụng có tích hợp cổng thanh toán như: PayPal hoặc thanh toán bằng VISA. | Chức năng đặt hàng, thanh toán đơn giản. |

Bảng .: Bảng so sách các ứng dụng thời trang.

## Kết luận, đề xuất giải pháp cho bài toán

### Kết luận

Sau khi phân tích và đánh giá các ứng dụng thời trang trên thì đối với việc phát triển một ứng dụng Android trên điện thoại cho một nhãn hàng thời trang là rất cần thiết.

Khi phát triển ứng dụng giao diện phải đẹp, tốc độ xủ lý nhanh, linh hoạt trong mục đích sử dụng của người dùng.

Ngoài ra còn phải thể hiện những gì mà nhãn hàng thời trang muốn người dùng biết đến.

### Giải pháp

Sử dụng công nghệ Spring framework với cơ chế bảo mật hiêu quả giúp người dùng bảo mật tài khoản của họ. Ngoài ra sử dụng Spring framework để cải thiện tốc độ lấy và trả dữ liệu của website.

Thiết kế giao diện theo hướng material.

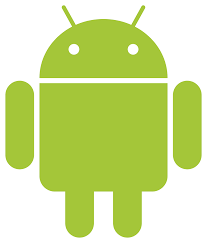
Dùng FireBase service để cung cấp những tính năng tiện lợi cho người dùng.

# Nội dung nghiên cứu

## Hệ điều hành Android

### Giới thiệu Android

Như chúng ta biết, hiện tại đã có hơn nửa nhân loại sử dụng máy di động để thoại và giao tiếp qua các mạng không dây. Con số 4 tỉ người này sẽ còn tăng lên và máy di động càng ngày càng "thông minh" với nhiều chức năng và dịch vụ rất hấp dẫn, cho nên thị trường máy di động thông minh sẽ vượt xa máy vi tính trong một tương lai rất gần...

Vì thế việc lập trình trên thiết bị di động ngày càng phổ biến và phát triển rất mạnh mẽ. Từ nền tảng mã nguồn mở, Google đã cho ra mắt Android chạy trên các thiết bị di động. Android có rất nhiều công cụ và dụng cụ miễn phí để nghiên cứu và phát triển phần mềm trên nền tảng của nó. Tài liệu này sẽ giúp chúng ta tìm hiểu về Android.

Hình . Biểu tượng Android.

### Khái niệm Android

Android là nền tảng phần mềm dựa trên mã nguồn mở Linux OS cho máy di động và những phần mềm trung gian để hổ trợ các ứng dụng mà người sử dụng cần đến. Định nghĩa một cách không quá chuyên môn thì có thể coi Android là tên một nền tảng mở cho thiết bị di động của Google (gồm hệ điều hành, phần mền trung gian và một số ứng dụng cơ bản). Android sẽ đương đầu với một số hệ điều hành dành cho thiết bị di dộng khác trên thị trường như Windows Mobile, Symbian và IOS.

Android hiện nay không chỉ giới hạn là hệ điều hành trên điện thoại di động mà nó còn được phát triển tích hợp trên các thiết bị điện tử khác như TV, đầu HD, tủ lạnh…

Android đã thu hút được sự chú ý của giới công nghệ khắp toàn cầu khi đứa con của Google sử dụng giấy phép mã nguồn mở. Đó là một sản phẩm kết tinh từ ý tưởng của Khối Liên minh thiết bị cầm tay mở do Google dẫn đầu, gồm 34 thành viên với các công ty hàng đầu về công nghệ và di động toàn cầu như Qualcomm, Intel, Motorola, Texas Instruments và LG Electronics, các nhà mạng như T-Mobile, Sprint Nextel, NTT DoCoMo và China Mobile.

### Khả năng Android

#### Tính năng mở của Android

Android được xây dựng từ dưới đi lên cho phép người phát triển tạo các ứng dụng di động hấp dẫn với đầy đủ các điểm mạnh của các thiết bị cầm tay hiện có. Android hoàn toàn mở, một ứng dụng có thể gọi tới bất kỳ chức năng lõi của điện thoại như tạo cuộc gọi, gửi tin nhắn hay sử dụng máy ảnh, cho phép người phát triển tạo ra phong phú hơn, liên kết hơn các ứng dụng hướng tới người dùng. Android là một mã nguồn mở, nó có thể được mở rộng để kết hợp tự do giữa các công nghệ nổi trội. Nền tảng này sẽ tiếp tục phát triển bởi cộng động phát triển để tạo ra các ứng dụng di động hoàn hảo.

#### Không có rào cản ứng dụng trong Android

Android phá vỡ mọi rào cản đối với người phát triển để tạo ứng dụng mới và cái tiến. Một người phát triển có thể kết hợp thông tin trang web với dữ liệu trên điện thoại cá nhân, chẳng han như danh bạn, lịch hay vị trí trên bản đồ, để cung cấp chính xác hơn những thông tin cần thiết cho người khác. Với Android, người phát triển có thể xây dựng một ứng dụng mà cho phép người dùng xem vị trí của những người bạn và thông báo khi họ đang ở vị trí lân cận. Tất cả được lập trình dễ dàng thông qua sự hỗ trợ của MapView và dịch vụ định vị toàn cầu GPS.

#### Android với tốc độ nhanh và phát triển ứng dụng dễ dàng

Android cung cấp bộ thư viện giao diện lập trình ứng dụng đồ sồ và các công cụ dễ dàng sử dụng để viết các ứng dụng từ đơn giản tới phức tạp. Android có thể cho phép người phát triển biết được vị trí của thiết bị và cho phép các thiết bị giao tiếp với nhau để có thể tạo nên mạng xã hội chia sẻ ngang hàng rộng khắp.

### Đặc tính

Ứng dụng framework cho phép tái sử dụng và thay thế các thành phần.

Các nhà phát triển viết ứng dụng Android dựa trên nền tảng Java.

Dalvik máy ảo được tối ưu hóa cho các thiết bị di động.

Tích hợp trình duyệt dựa trên động cơ WebKit mã nguồn mở.

Tối ưu hóa đồ họa được hỗ trợ bởi một tùy chỉnh đồ họa 2D thư viện; đồ họa 3D dựa trên những đặc điểm kỹ thuật OpenGL ES 1,0 (Tùy chôn tang tốc phần cứng ).

SQLite cho việc lưu trữ dữ liệu cấu trúc.

Phương tiện truyền thông hỗ trợ cho âm thanh phổ biến, video, và vẫn c.n định dạng h.nh ảnh (MPEG4, H.264, MP3, AAC, AMR, JPG, PNG, GIF).

GSM điện thoại (phụ thuộc phần cứng).

Bluetooth, EDGE, 3G và WiFi (phụ thuộc phần cứng ).

Camera, GPS, la bàn, và gia tốc (phụ thuộc phần cứng).

Môi trường phát triển phong phú bao gồm một mô phỏng thiết bị, công cụ để gỡ lỗi, bộ nhớ và profiling hiệu suất, và một plugin cho IDE Eclipse.

### Lịch sử Android

Android là công ty được thành lập tại Palo Alto, California vào tháng 10 năm 2003 bởi Andy Rubin (đồng sáng lập công ty Danger), Rich Miner (đồng sáng lập Tổng công ty Viễn thông Wildfire), Nick Sears (từng là Phó giám đốc T-Mobile), và Chris White (trưởng thiết kế và giao diện tại WebTV).

Google mua lại Android Inc vào ngày 17 tháng 8 năm 2005, biến nó thành một bộ phận trực thuộc Google. Những nhân viên chủ chốt của Android, gồm Rubin, Miner và White vẫn tiếp tục ở lại làm việc sau khi được mua lại. Nhiều dự đoán cho rằng Google dự tính tham gia thị trường điện thoại di động sau bước đi này. Tại Google, nhóm do Rubin đứng đầu đã phát triển một nền tảng thiết bị di động phát triển trên nền nhân Linux.

Google quảng bá nền tảng này cho các nhà sản xuất điện thoại và các nhà mạng với lời hứa sẽ cung cấp một hệ thống uyển chuyển và có khả năng nâng cấp. Google đã liên hệ với hàng loạt hang phần cứng cũng như đối tác phần mềm,.

Ngày 5 tháng 11 năm 2007, Liên minh thiết bị cầm tay mở (Open Handset Alliance), một hiệp hội bao gồm nhiều công ty trong đó Texas instrucments, Tập đoàn Broadcom, Google, HTC, Intel, LG, Tập đoàn Marvell Technology, Motorola, Nvidia, Qualcomm, Samsung Electronics, Sprint Nextel và T-Mobile được thành lập với mục đích phát triển các tiêu chuẩn mở cho thiết bị di động đo Google đứng đầu. Cùng ngày, Android cũng được ra mắt với vai trò là sản phẩm đầu tiên của Liên minh, một nền tảng thiết bị di động được xây dựng trên nhân Linux phiên bản 2.6.

Chiếc điện thoại chạy Android đầu tiên được bán ra là HTC Dream, phát hành ngày 22 tháng 10 năm 2008. Biểu tượng của hệ điều hành Android mởi là một con robot màu xanh lá cây do hang thiết kế Irina Blok tại California vẽ.

Từ năm 2008, Android trải qua nhiều lần cập nhật nâng cấp để dần dần cải tiến hệ điều hành, bổ sung các tính năng mới và sửa các lỗi trong những lần phát hành trước. Mỗi bản nâng cấp được đặt tên lần lượt theo thứ tự bảng chữ cái, theo tên của một món ăng tráng miệng, điều đó thật thú vị. Ví dụ như phiên bản 1.5 Cupcake ( bánh bông lan nhỏ có kem), tiếp nối là 1.6 Donut (bánh vòng). Phiên bản mới nhất hiện nay là 4.4 Kitkat.



Hình . Điện thoại Android. HTC Dream (trái), HTC Magic ( Phải)

### Kiến trúc Android

Hình . Kiến trúc Android

#### Applications

Là lớp trên cùng của kiến trúc Android. Hệ điều hành Android tích hợp sẳn một số ứng dụng cơ bản như email client, SMS, lịch điện tử, bản đồ, trình duyệt web, sổ liên lạc và một số ứng dụng khác. Ngoài ra tầng này cũng chính là tầng chứa các ứng dụng được phát triển bằng ngôn ngữ Java.

#### Application Framework

Google xây dựng cho các nhà phát triển tầng này để phát triển các ứng dụng của họ trên Android chỉ bằng cách gọi các API. Đây là tầng mà ứng dụng của chúng ta tương tác trực tiếp. Các chương trình quản lý các chức năng cơ bản của điện thoại như quản lý tài nguyên, quản lý cuộc gọi, v.v… Là một nhà phát triển, bạn chỉ cần xem xét đây là một số công cụ cơ bản mà chúng ta xây dựng các ứng dụng của chúng ta.

View UI - để xây dựng layout của ứng dụng bao gồm: list view, text field, button, dialog, form …

Content Providers - cho phép các ứng dụng truy cập dữ liệu từ các ứng dụng khác hoặc để chia sẻ dữ liệu của riêng ứng dụng.

Resource Manager - cung cấp cách thức truy cập đến non-code resources như các asset, graphic, image, music, video …

Notification Manager - cho phép tất cả các ứng dụng hiển thị thông báo của mình trên hệ điều hành.

Activity Manager - quản lý vòng đời của các ứng dụng.

### Android Runtime & Libraries

#### Android Runtime

Android có một tập các thư viện nòng cốt để cung cấp hầu hết các chức năng sẵn có trong thư viện cốt lõi của ngôn ngữ lập trình Java. Android Runtime bao gồm máy ảo Dalvik và các thư viện Android.

Máy ảo Dalvik là máy ảo để chạy các ứng dụng trên Android, đã được tối ưu để đảm bảo rằng một thiết bị có thể chạy được nhiều chương trình một cách hiệu quả. Nó dựa vào nhân Linux để thực hiện đa luồng và quản lý bộ nhớ cấp thấp.

Các thư viện cơ bản: các ứng dụng Android được phát triển trên môi trường Java, nhưng Dalvik lại không phải là một máy ảo Java JVM (Java Virtual Machine). Các thư viện cơ bản của Android cung cấp hầu hết các chức năng có trong thư viện cơ bản của Java cũng như là thư viện riêng của Android.

#### Libraries

Đây là tầng thư viện gốc của Android. Tầng này cho phép các thiết bị xử lý các loại dữ liệu khác nhau. Các thư viện này được viết bằng ngôn ngữ C hoặc C++ và được chỉ định như là một phần cứng cụ thể. Sẽ được các nhà phát triển phát triển ứng dụng Android sử dụng khai thác.

Một vài thư viện gốc:

* System C library: một thể hiện được xây dựng từ BSD của bộ thư viện hệ thống C chuẩn (libc), được điều chỉnh để tối ưu hóa cho các thiết bị chạy trên nền Linux.
* Media libraries: Bộ thư viện hổ trợ trình diễn và ghi các định dạng âm than và hình ảnh phổ biến.
* Surface manager: Quản lý hiển thị nội dung 2D và 3D.
* LibWebCore: Một web browser engine hiện đại được sử dụng trong trình duyệt của Android lần trong trình duyệt nhúng web view được sử dụng trong ứng dụng.
* SGL: Engine hổ trợ đồ họa 2D.
* 3D libraries: Một thể hiện được xây dựng dựa trên các APIs của OpenGL ES 1.0. Những thư viện này sử dụng các tăng tốc 3D bằng phần cứng lẫn phần mềm để tối ưu hóa hiển thị 3D.
* FreeType: Bitmap and vector font rendering.
* SQLite: hệ cơ sở dữ liệu nhỏ gọn và mạnh mẽ.
* Webkit: công cụ trình duyệt được sử dụng để hiển thị nội dung HTML.

### Linux Kernel

* Toàn bộ hệ thống Android được xây dựng trên Linux 2.6 Kernel với một vài thay đổi bởi Google. Hệ thống Linux này tương tác với phần cứng và tất cả các trình điều khiển cần thiết. Trình điều khiển được chương trình kiểm soát và giao tiếp với phần cứng. Linux Kernel cũng hoạt động như một tầng trừu tượng giữa phần cứng và các lớp phần mềm khác. Android sử dụng Linux cho tất cả các chức năng cốt lõi như quản lý bộ nhớ, quản lý tiến trình, mạng, thiết lập bảo mật…
* Tầng này có các thành phần chủ yếu:
* Display Driver: Điều khiển việc hiển thị lên màn hình cũng như thu nhận những điều khiển của người dùng trên màn hình (di chuyển, cảm ứng…).
* Camera Driver: Điều khiển hoạt động của camera, nhận luồng dữ liệu từ camera trả về.
* Bluetooth Driver: Điều khiển thiết bị thu và phát sóng Bluetooth.
* USB driver: Quản lý hoạt động của các cổng giao tiếp USB.
* Keypad driver: Điều khiển bàn phím.
* Wifi driver: Chịu trách nhiệm về việc thu phát sóng wifi.
* Audio Driver: Điều khiển các bộ thu phát âm thanh, giải mã các tín hiệu dạng audio thành tín hiệu số và ngược lại.
* Binder IPC Driver: Chịu trách nhiệm về việc kết nối và liên lạc với mạng vô tuyến như CDMA, GSM, 3G, 4G, E để đảm bảo những chức năng truyền thông được thực hiện.
* M-System Driver: Quản lý việc đọc ghi…lên các thiết bị như thẻ nhớ SD, flash.
* Power Management: Giám sát việc tiêu thụ điện năng.

## Lập trình Android

### Máy ảo Dalvik

Máy ảo Dalvik đã được giới thiệu sơ lượt ở phần trên. Nó là một trong những thành phần quan trọng của Android. Thay vì sử dụng máy ảo Java như trước kia, Android sử dụng máy ảo của riêng nó được thiết kế để đảm bảo rằng đa số ứng dụng có thể chạy mượt mà trên một thiết bị di động.

Máy ảo Dalvik sử dụng lõi Linux để xử lý các chức năng ở mức thấp bao gồm bảo mật, các tiến trình, các luồng và quản lý vùng nhớ. Nó cũng có thể viết bằng ứng dụng C/C++ để chạy trực tiếp lên lõi Linux. Giữa phần cứng và các dịch vụ hệ thống được quản lý bởi máy ảo Dalvik, nó là một thành phần ở giữa. Bằng cách sử dụng máy ảo này để chạy ứng dụng, các nhà phát triển hoàn toàn không phải bận tâm gì về các phần cứng bên dưới.

### Các gói Java

* java.lang – gói chứa các lớp lõi của Java.
* java.io - xuất nhập.
* java.net - kết nối mạng.
* java.util - chứa các lớp tiện ích. Bao gồm: Log thường được dùng đề viết LogCat.
* java.text - tiện ích sử lý văn bản.
* java.math - các lớp toán học.
* javax.net - các lớp mạng.
* javax.security - Security-related classes.
* javax.xml - các lớp liên quan đến DOM-based XML.
* org.apache.\* - các lớp liên quan đến HTTP-related.
* org.xml - các lớp liên quan đến SAX-based XML.

### Các gói Android

* Android.app—Android application model access.
* Android.content—truy xuất dữ liệu trong Android.
* Android.net—bao gồm Uri class dùng cho việc truy xuất các nội dung khác.
* Android.graphics—đồ họa.
* Android.opengl—các lớp OpenGL.
* Android.os—truy cập ở mức độ hệ thông đến môi trường Android.
* Android.provider—các lớp liên quan đến ContentProvider.
* Android.telephony—khả năng truy cập.
* Android.text—Text layout.
* Android.util—tập các tiện thao tác trên văn bản,bao gồm XML.
* Android.view—thành phần UI (giao diện người dùng).
* Android.webkit—Browser functionality.
* Android.widget—thành phần UI mức độ cao hơn.

### Các thành phần quan trọng trong ứng dụng

* Một ứng dụng trên Android được cấu thành từ bốn thành phần cơ bản:
* Activity
* Service
* Broadcast Receivers
* Content Provider
* Các thành phần này không nhất thiết phải có mặt đầy đủ trong ứng dụng nhưng nó cấu thành nên một ứng dụng Android. Chúng ta có thể xem các thành phần này và quản lý chúng trong file AndroidManifest.xml – một file quan trọng.

#### Activity

##### Khái niệm

* Một Activity là một giao diện người dùng trực quan mà người dùng có thể thực hiện tương tác trực tiếp trên đó mỗi khi được kích hoạt. Một ứng dụng có thể có nhiều Activity và chúng có thể gọi đến nhau, chuyển đổi giữa các Activity với nhau. Mỗi Activity là một dẫn xuất của lớp android.app.Activity
* Mỗi Activity có một khung giao diện để vẽ lên. Thông thường các khung này phủ đầy màn hình, ngoài ra nó cũng có thể có thêm các cửa sổ con khác như là hộp thoại, tiêu đề…

##### Vòng đời của Activity

* Các Activity trong hệ thống được quản lý bởi một cấu trúc dữ liệu ngăn xếp. Khi có một Activity được khởi tạo, nó được đẩy vào trong ngăn xếp, chuyển sang trạng thái thực thi và Activity trước đó sẽ chuyển sang trạng thái chờ. Activity này chỉ trở lại trạng thái thực thi khi mà Activity vừa khởi tạo kết thúc việc thực thi.



Hình . Vòng đời của Activity

##### Các trạng thái của chu kỳ sống

* Running khi nó ở trên nhất màn hình và nhận tương tác người dùng.
* Paused khi Activity không còn là trọng tâm trên màn hình, nhưng vẫn hiện thị trước người dùng. Có nghĩa là một Activity mới ở trên nó nhưng không bao phủ đầy màn hình.
* Stopped khi một Activity hoàn toàn bị che khuất, nó sẽ rơi vào trạng thái Stopped. Tuy nhiên, nó vẫn còn lưu trữ toàn bộ thông tin trạng thái. Người dùng không thấy nó và thường bị loại bỏ trong trường hợp hệ thống cần vùng nhớ cho tác vụ khác.
* Killed Khi hệ thống bị thiếu bộ nhớ, nó sẽ giải phóng các tiến trình theo nguyên tắc ưu tiên. Các Activity ở trạng thái stop hoặc paused cũng có thể bị giải phóng và khi nó được hiển thị lại thì các Activity này phải khởi động lại hoàn toàn và phục hồi lại trạng thái trước đó.

Hình . Các trạng thái của chu kì sống của Activity

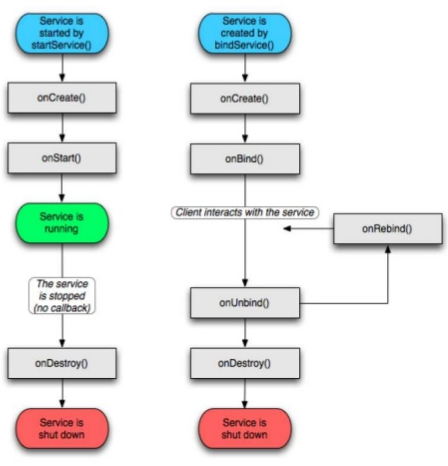
* Khi chuyển giữa các trạng thái, ứng dụng sẽ gọi các hàm callback ứng với các bước chuyển:
* OnCreate(...): hàm này được gọi khi lớp Activity được khởi tạo, dùng để thiết lập giao diện ứng dụng và thực thi những thao tác cơ bản.
* onStart(): hàm này được gọi khi lớp ứng dụng xuất hiện trên màn hình.
* onResume(): hàm được gọi ngay sau OnStart hoặc khi người dùng focus ứng dụng, hàm này sẽ đưa ứng dụng lên top màn hình.
* onPause(): hàm được gọi khi hệ thống đang focus đến 1 activity trước đó.
* onStop(): hàm được gọi khi một activity khác được khởi động và focus.
* onRestart(): đưọc gọi khi ứng dụng chuyển sang onStop(), nhưng muốn khởi động lại bằng onStart().
* Vòng đời của một Activity có thể được thực hiện trong quá trình sau:
* Toàn bộ thời gian sống của một Activity bắt đầu từ lời gọi đầu tiên tới phương thức onCreate(Bundle) tới lời gọi phương thức onDestroy(). Trong quá trình này, một Activity sẽ khởi tạo lại tất cả các tài nguyên cần thiết sử dụng trong phương thức onCreate() và giải phóng chúng khi phương thức onDestroy() được thực thi.
* Thời gian sống có thể nhìn thấy của một Activity bắt đầu từ lời gọi tới phương thức onStart(), cho tới khi phương thức onStop() của nó được thực thi. Toàn bộ các tài nguyên đang được sử dụng bởi Activity vẫn tiếp tục được lưu giữ, người dùng có thể thấy giao diện nhưng không tương tác được với Activity do trong quá trình này Activity không ở trạng thái tương tước được.
* Thời gian sống của một Activity thực sự là quá trình bắt đầu từ khi có lời gọi tới phương thức onResume() và kết thúc bằng lời gọi tới phương thức onPause(). Trong thời gian này, Activity có thể thương tác với người dùng.

#### Service

##### Khái niệm

* Một Service là các đoạn mã được thực thi ngầm bởi hệ thống mà người sử dụng không thấy và không tương tác được. Mỗi Service đều được extends từ lớp cơ sở là Server trong gói android.app. Có thể kết nối tới hoặc kích hoạt một Service thông qua interface mà Service đưa ra. Ví dụ như một chương trình chơi nhạc, sẽ có vài hoạt động cho phép người dùng duyệt danh sách các bài hát và lựa chọn bài nào để phát. Tuy nhiên, chức năng chơi nhạc không được thiết kế như một Activity bởi chúng ta có thể sẽ muốn chuyển qua cửa sổ khác, như khi soạn tin nhắn thì bài hát vẫn tiếp tục được chơi. Trong trường hợp này, ứng dụng chơi nhạc sẽ khởi tạo một Service bằng cách sử dụng phương thức Context.startService().
* Một ứng dụng có thể dễ dàng thực hiện liên kết tới một Service đang chạy (thậm chí khởi động nếu nó chưa thực thi) bằng phương thức Context.bindService(). Khi đó Service này sẽ cung cấp cho ứng dụng cơ chế để giao tiếp với chúng thông qua giao diện gọi là IBinder (đối với Service chơi nhạc có thể cho phép dừng hoặc chuyển qua bài nhạc kế tiếp).

##### Vòng đời của một Service

* Vòng đời của một Service được hiểu là quá trình hoạt động từ khi nó được tạo ra cho tới khi bị loại khỏi hệ thống. Có hai cách thức để một Service có thể được chạy trong hệ thống:
* Khi hệ thống có lời gọi tới phương thức Context.StartService(). Trong trường hợp này, dịch vụ sẽ được thực hiện liên tục cho tới khi hệ thống gọi phương thức Context.stopService().
* Khi các ứng dụng gọi phương thức Context.bindService() để tạo kết nối tới Service (Service sẽ được khởi tạo nếu tại thời điểm đón ó đang không hoạt động). Ứng dụng sẽ nhận được một đối tượng IBinder do Serivce trả lại để có thể gọi các phương thức Callback phù hợp để truy cập tới các trạng thái của dịch vụ. Nếu do lời gọi Context.bindService() mà dịch vụ được khởi tạo thì nó sẽ được thực thi cho tới khi nào kết nối trên (tức là đối tượng IBinder) vẫn còn tồn tại.

Hình . Sơ đồ chuyển trạng thái của Service

#### Broadcast Receivers

##### Khái niệm

* Broadcast Receivers là một thành phần không làm gì cả nhưng nó nhận và phản hồi lại các thông báo. Các thông báo bắt nguồn từ mã hệ thống – như pin yếu, hay có một ảnh mới vừa chụp, … Một ứng dụng thông thường cũng có thể phát ra các thông báo – ví dụ như, nó để các ứng dụng khác biết là dữ liệu mới được tải về hay nó vừa trả ra kết quả cho các ứng dụng khác. Một ứng dụng có nhiều Broadcast Receivers. Tất cả các receivers thừa kế từ lớp BroadcastReceiver cơ sở.
* Ví dụ, chúng ta xây dựng một ứng dụng gọi điện thoại. Chúng ta đăng ký BroadcastReceiver nhận cuộc gọi đến. Khi có cuộc gọi điến thì hệ thống sẽ hiện danh sách các ứng dụng có đăng ký BroadcastReceiver nhận cuộc gọi, trong đó có ứng dụng của ta vừa đăng ký. Nếu chỉ có một ứng dụng đăng ký thì mặc định khởi chạy ứng dụng đó.

#### Content Provider

##### Khái niệm

* Content Providers là thành phần cung cấp dữ liệu từ một ứng dụng đến một ứng dụng khác dựa trên các yêu cầu. Mỗi yêu cầu được xử lý bằng các phương thức của class ContentResolver. Một Content Provider có thể sử dụng các cách lưu trữ dữ liệu khác nhau, dữ liệu có thể được lưu trữ trong databases, file, thậm chí thông qua kết nối mạng.
* Mỗi ứng dụng Android chạy trong các tiến trình riêng của chính mình và nó có các điều khoản riêng của nó, điều mà giữ dữ liệu của ứng dụng ẩn với các ứng dụng khác. Tuy nhiên, thỉnh thoảng nó được yêu cầu chia sẻ dữ liệu đến các ứng dụng khác. Sử dụng Content Provider trong việc này rất hữu ích.
* Content Providers cho phép bạn tập trung dữ liệu ở một nơi và các ứng dụng khác nhau sẽ truy xuất vào nó khi cần thiết. Content Provider hoạt động rất giống với một cơ sở dữ liệu, và bạn có thể truy vấn nó, chỉnh sửa nội dung, cũng như là thêm xóa các nội dung sử dụng các phương thức: insert(), update(), delete(), query(). Trong nhiều trường hợp dữ liệu được lưu trữ trong SQLite.

## Xác định yêu cầu

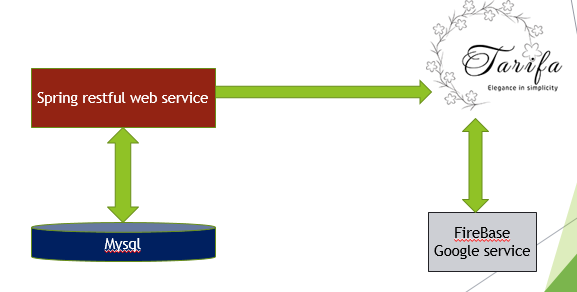
### Yêu cầu chức năng

* Xây dựng .ứng dụng trên nền tảng android mô phỏng lại trang web của nhãn hàng thời trang Tarifa.
* Ứng dụng cung cấp đầy đủ chức năng mua hàng.
* Cho phép web và ứng dụng cùng sử dụng chung nguồn dữ liệu.
* Có thể cho người dùng tương tác thời gian thực trên sản phẩm khi đánh giá sản phẩm.
* Có thể lưu lại những sản phẩm mà người dùng đã chọn trước đó ở phần giỏ hàng và sản phẩm yêu thích để hiển thị khi khách hàng đang nhập ở thiết bị khác.

### Yêu cầu hệ thống

* Xây dựng hai database phục vụ cho ứng dụng. Gồm database cho server để lưu trữ dữ liệu lâu dài và database google firebase service để lưu lại hoạt động người dùng và tương tác thời gian thực.
* Server đáp ứng được khả năng truy cập dữ liệu từ nhiều điện thoại cùng một lúc, và đồng bộ chúng để tránh mất mát thông tin.

## Mô hình biểu đồ dữ liệu chung



Hình . Biểu đồ dữ liệu

* Dữ liệu này được đồng bộ từ trên server. Ứng dụng gửi yêu cầu dữ liệu lên webservice, webservice truy xuất dữ liệu từ database trên server và gửi trả lại phản hồi cho ứng dụng. Ứng dụng sẽ hiển thị dữ liệu cho người dùng và tương tác với Firebase Google service.

## Webservice

### Phân tích yêu cầu

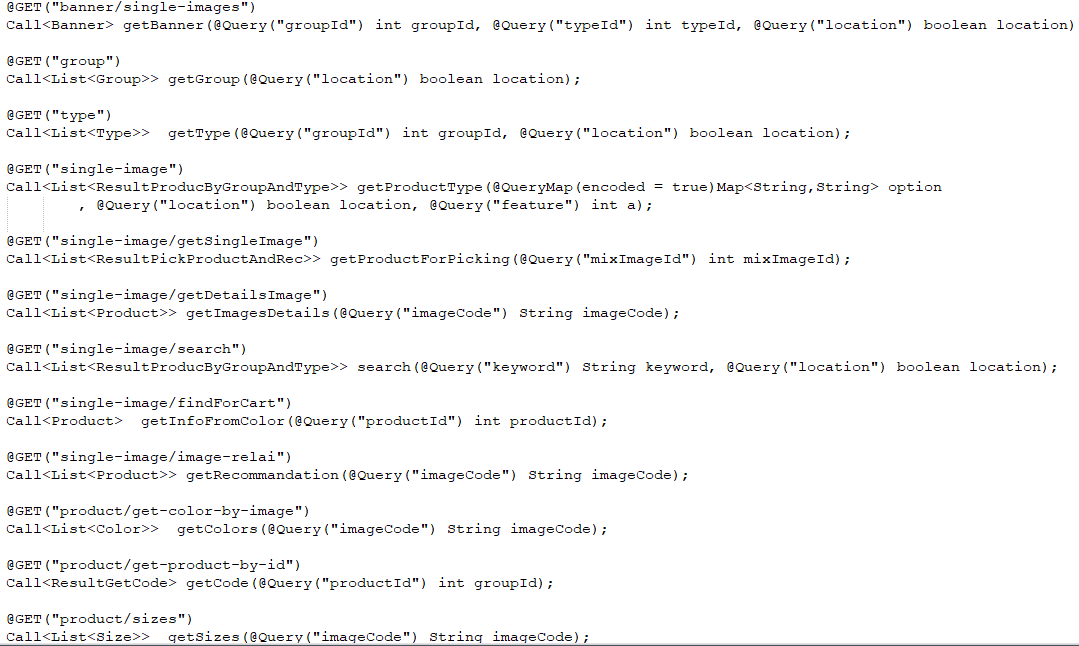
Dựa vào tính thực tế trong việc yêu cầu một server cung cấp khả năng nhận yêu cầu -> tiến hành xử lý -> kết nối database để truy xuất dữ liệu -> trả lại phản hồi dữ liệu từ yêu cầu. Vậy nên việc xây dựng một Webservice để đáp ứng những nhu cầu đó là cần thiết hơn so với các nền tảng khác.

### Hướng xây dựng Webservice

Webservice được xây dựng trên ngôn ngữ Java sử dụng Spring framework, kết nối tới database sử dụng MySQL thông qua Spring Data.

Sử dụng định dạng REST trong việc gửi request và trả về response.

Webservice cung cấp các phương thức phục vụ cho việc lấy dữ liệu từ server của ứng dụng.

* 

Hình 4.2: Danh s

Hình . Hàm của Webservice

### Chi tiết từng phương thức

**getBanner**

* Phương thức này dùng để trả về thông tin banner cho phần xem sản phẩm theo loại.

**getGroup**

* Phương thức này dùng để lấy danh sách các nhóm.

**getType**

* Phương thức này dùng để trả về danh sách các type có trong một nhóm nhất định.

**getProductType**

* Phương thức này dùng để trả về danh sách các sản phẩm có trong một type.

**getProductForPicking**

* Phương thức này dùng để trả về danh sách các sản phẩm có trong một sản phẩm phối.

**getImagesDetails**

* Phương thức này dùng để trả về danh sách các hình ảnh góc nhìn của một sản phẩm.

**search**

* Phương thức này dùng để tìm kiếm sản phẩm theo tên.

**getInfoFromColor**

* Phương thức này dùng để trả về thông tin của một sản phẩm theo một màu nất đinh.

**getRecommandation**

* Phương thức này dùng để trả về những sản phẩm gợi ý cho người dùng khi đang xem một sản phẩm nào đó.

**getColors**

* Phương thức này dùng để trả về tất cả mày của một sản phẩm nhất định.

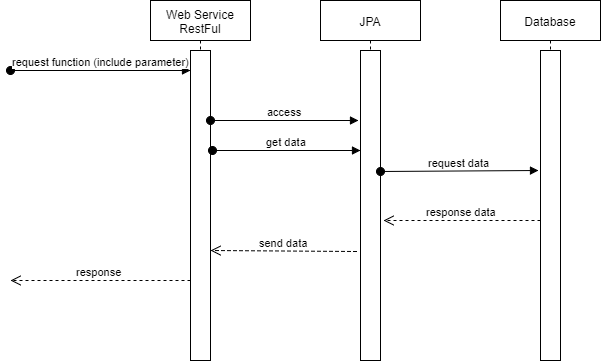
**getCode**

Phương thức này dùng để trả về mã code của sản phẩm dựa trên id sản phẩm đó.

**getSizes**

* Phương thức này dùng để trả về tất cả các size của một sản phẩm.

### Sequence Diagram chung



Hình . Sequence Diagram Webservice

### Class Diagram

Hình . Class Diagram Webservice

## Ứng dụng

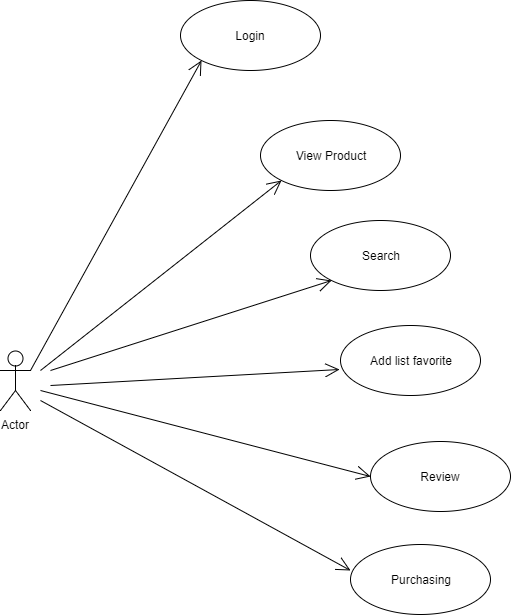
### Tổng quan

Ứng dụng được viết trên nền tảng android đáp ứng đủ tiêu chí của đặc tả yêu cầu đề tài:

* Cho phép xem sản phẩm phối, xem sản phẩm đơn và chi tiết sản phẩm.
* Có khả năng quản lý danh sách yêu thích và giỏ hàng của người dùng một cách hợp lý, linh hoạt và chính xác.
* Cho phép tìm kiếm sản phẩm theo tên một cách nhanh chóng.
* Ứng dụng cho người dùng tương tác với nhau, nhận xét sản phẩm thời gian thực.
* Có thể đặt hàng nhanh chóng và tiện lợi.

### Usecase

Từ những yêu cầu trên, ta có usecase cho ứng dụng.

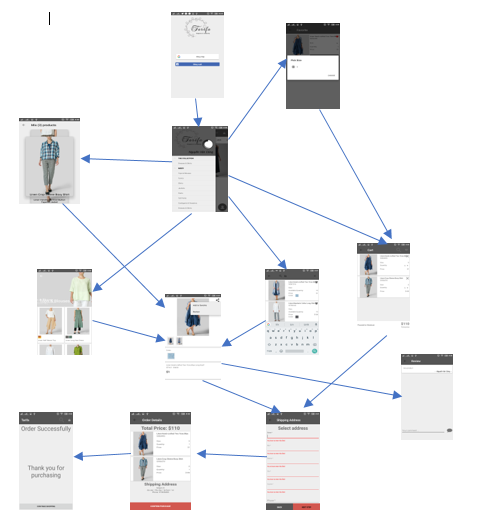


Hình . Usecase ứng dụng

### Sơ đồ luồng Activity

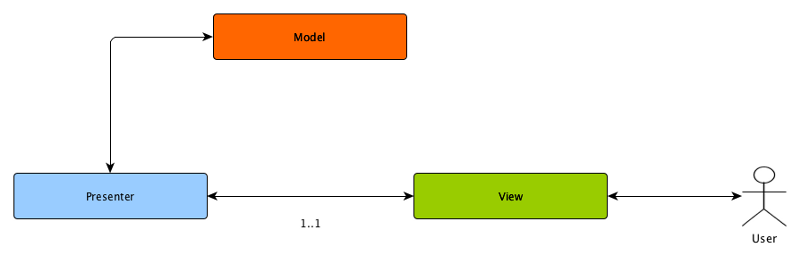
Activity trong Android là một trong những thành phần quan trọng nhất. Sơ đồ luồng Activity dưới đây sẽ thể hiện rõ việc người dùng có thể tương tác ứng dụng như thế nào, một cái nhìn tổng quan.

Sơ đồ luồng này chỉ gồm những Activity chính. Ứng dụng còn gồm nhiều giao diện là các Activity và Dialog khác nhằm đáp ứng toàn bộ yêu cầu của ứng dụng.



Hình . Luồng Activity

### Mô hình MVP



Hình . Mô hình MVP

Ứng dụng được xây dựng trên mô hình MVP – Model View Presenter. MVP là mô hình cải tiến của mô hình MVC.

M – Model: lớp xử lí dữ liệu. Lớp này sẽ chịu trách nhiệm lấy dữ liệu từ database hoặc network một cách bất đồng bộ. Sau đó sẽ trả về dữ liệu cho Presenter thông qua các hàm callback.

V- View: lớp xử lí view. Lớp này chịu trách nhiệm tìm view (bind view), đưa dữ liệu vào view, animation, kiểm soát các input event của user và gửi cho present các event

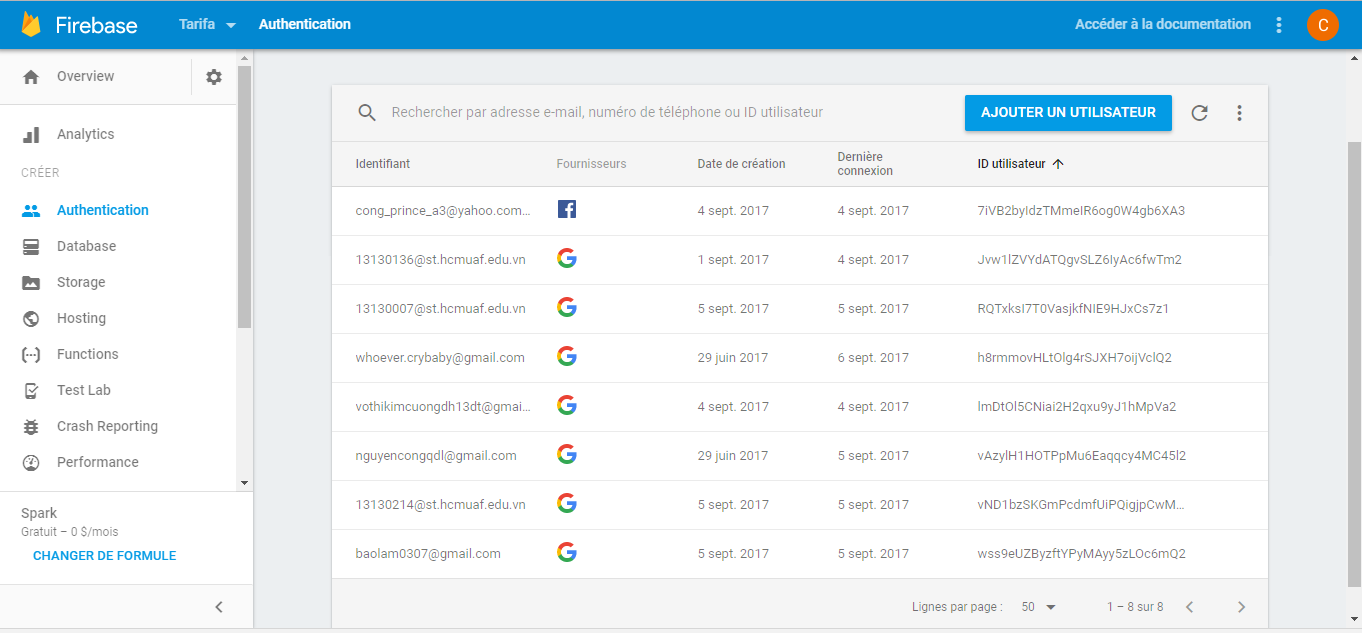
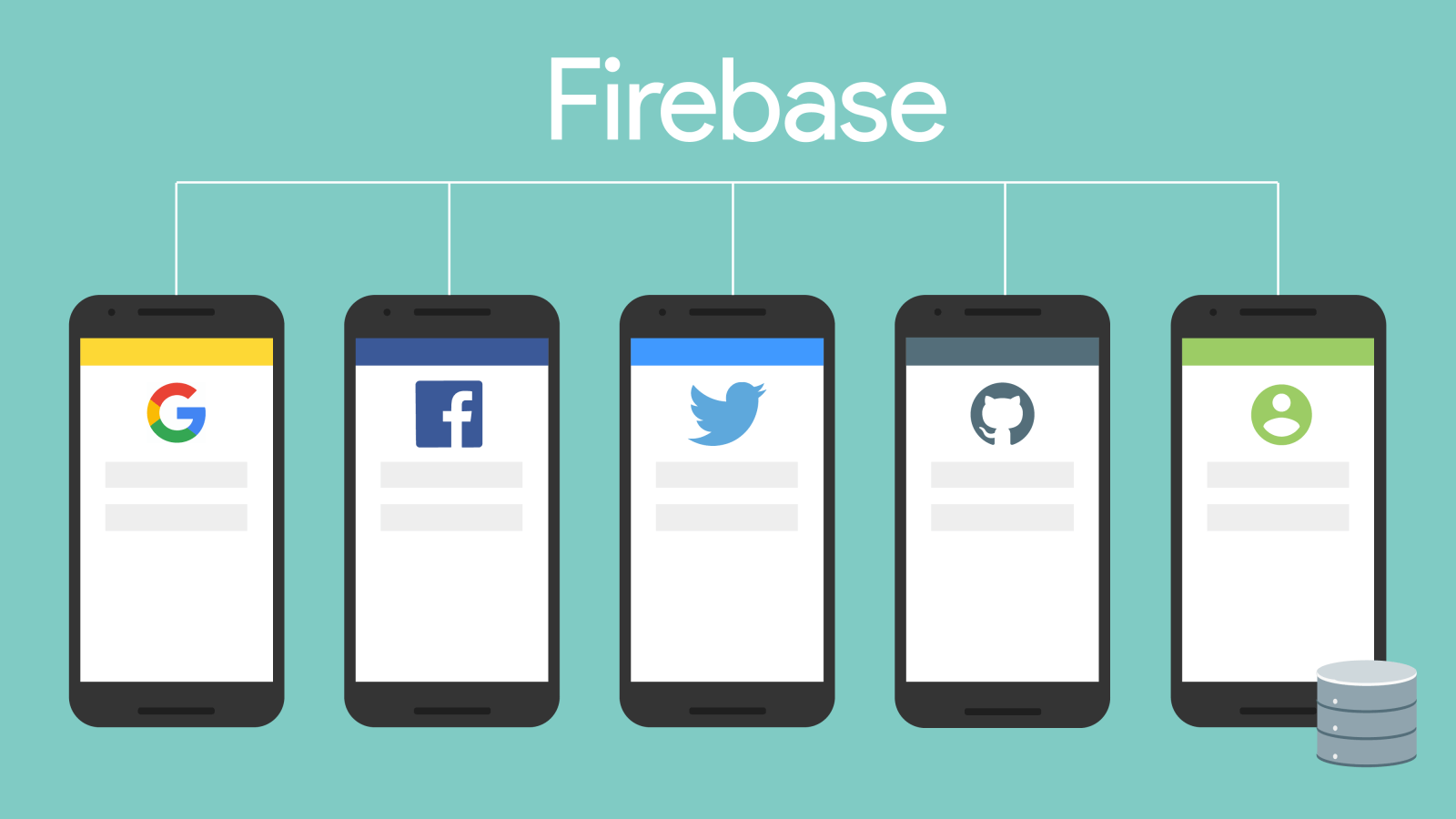
P – Presenter: lớp này sẽ là lớp xử lí các bussiness logic. Đây là sẽ lớp trung gian có thể giao tiếp được với hai lớp M & V (chú ý M – V ko  giao tiếp trực tiếp với nhau). Khi lớp View nhận một input event sẽ gửi sự kiện xuống lớp P, lớp P lấy sẽ liệu từ lớp Model và gửi lại cho lớp V và hướng dẫn lớp V cách hiển thị.

Với mô hình MVP chúng ta sẽ quản lý được mã nguồn ứng dụng một cách dễ dàng và rõ ràng hơn.

## Firebase

### Authentication

Firebase cung cấp cho ứng dụng khả năng đăng nhập và quản lí tài khoản tương thích với email, facebook, twitter và nhiều tài khoản mạng xã hội khác.

Việc đăng nhập này cũng có tính bảo mật khá cao được đảm bảo bởi google.

Hình . Quản lí các tài khoản đăng nhập

Hình . Đăng nhập với tài khoản của các trang xã hội

### Realtime database

Fiebase cung cấp cho ta một database quản lý dữ liệu dạng JSON.

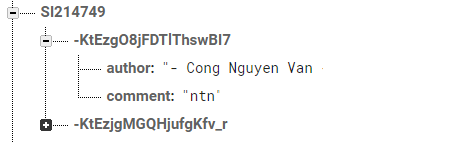
Với database này cũng cung cấp cho ta cách để cập nhập dữ liệu thời gian thực.

Việc bảo mật dữ liệu cũng được đảm bảo ở đây.

Với database này chúng ta có thể cho người dung nhận xét sản phẩm thời gian thực từ đó cho khách hàng cảm nhận được sự tương tác nhiều hơn.



Hình . Cấu trúc lưu database của một tài khoản trên Firebase



Hình . Cấu trúc lưu database review của một sản phẩm trên Firebase



Hình . Cấu trúc lưu database của các đơn đặt hàng từ khách hàng trên Firebase

## Một số thư viện quang trọng

### Retrofit

#### Giới thiệu

Retrofit là một Rest Client cho Android và Java và được tạo ra bởi Square. Họ làm cho việc nhận và tải lên JSON (hoặc dữ liệu khác) một cách khá dễ dàng tới một WebService dựa trên mô hình REST.

Các gói trang bị thêm cho phép sử dụng các bộ chuyển đổi sau đây:

* *Gson*: com.squareup.retrofit:converter-gson
* *Jackson*: com.squareup.retrofit:converter-jackson
* *Moshi*: com.squareup.retrofit:converter-moshi
* *Protobuf*: com.squareup.retrofit:converter-protobuf
* *Wire*: com.squareup.retrofit:converter-wire
* *Simple XML*: com.squareup.retrofit:converter-simplexml

#### Cài đặt

Để làm việc với **Retrofit** bạn cần triển khai cơ bản 3 lớp:

1. Model class để kết nối JSON Data
2. Interfaces để định nghĩa các API cho Webservice
3. Retrofit.Builder Lớp để định nghĩa URL Endpoint cho các hoạt động liên quan tới Http

### Picasso

#### Giới thiệu

[Picaso](http://square.github.io/picasso/) là một thư viện tải và xử lý hình ảnh cho Android. Nó được tạo ra và duy trì bởi [Square](http://square.github.io/), và cung cấp việc tải và xử lý hình ảnh. Nó đơn giản hoá quá trình hiển thị hình ảnh từ một nguồn bên ngoài. Trong nhiều trường hợp, chỉ một ít code là đủ để cài đặt thư việc đơn giản này.

Picaso làm việc rất tốt trong việc hiển thị hình ảnh từ xa. Thư viện xử lý mọi giai đoạn của quá trình, từ khởi tạo các yêu cầu HTTP đến lưu hình ảnh vào bộ nhớ đệm. Điều này có thể hơi dài dòng khi tự mình viết code để thực hiện những hành động này. Trong hướng dẫn nhanh này, chúng ta sẽ tìm hiểu một số trường hợp sử dụng thông dụng.

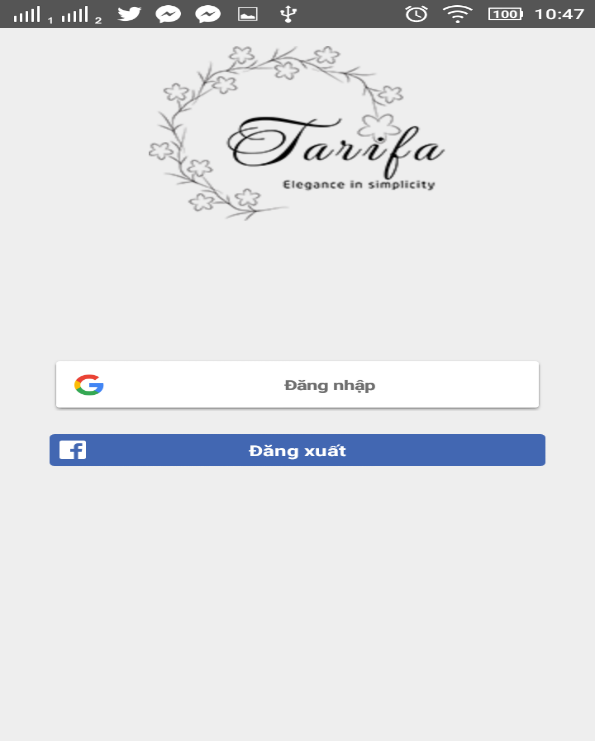
#### Cài đặt

Thêm compile 'com.squareup.picasso:picasso:2.3.3'vào tập tin build.gradle trong phần dependency.

# Kết quả đạt được

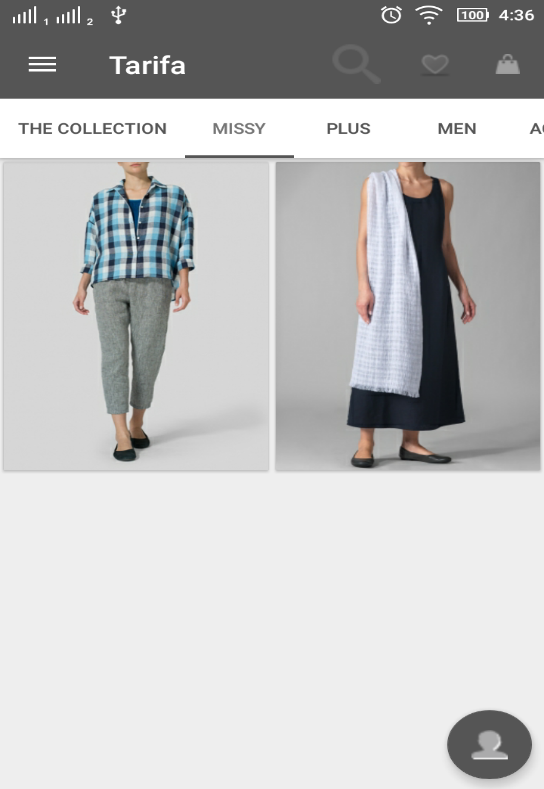
Dưới đây là những hình ảnh về ứng dụng:

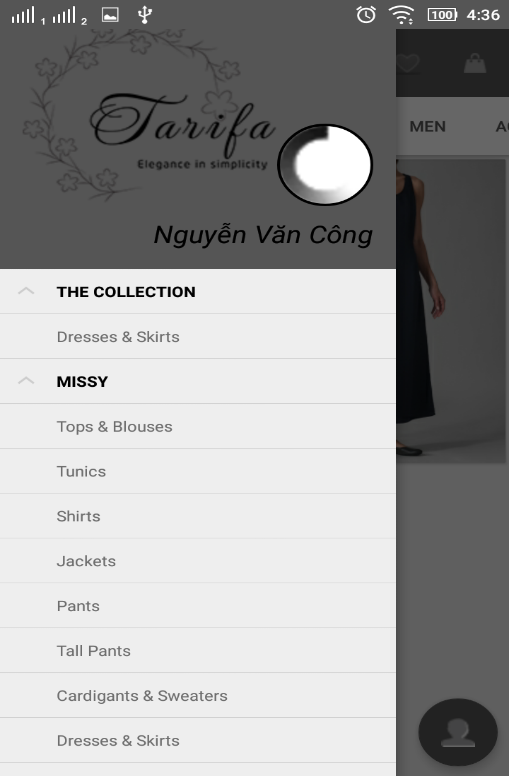
## Màn hình đăng nhập

Màn hình đăng nhập cho phép đăng nhập với 2 lựa chọn là tài khoản email google hoặc facebook.

Hình . Màn hình đăng nhập

## Màn hình chính

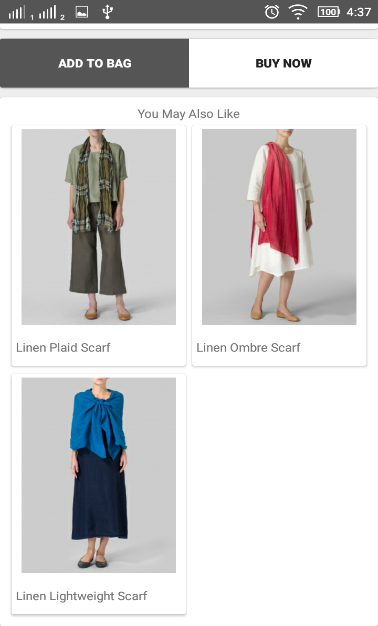
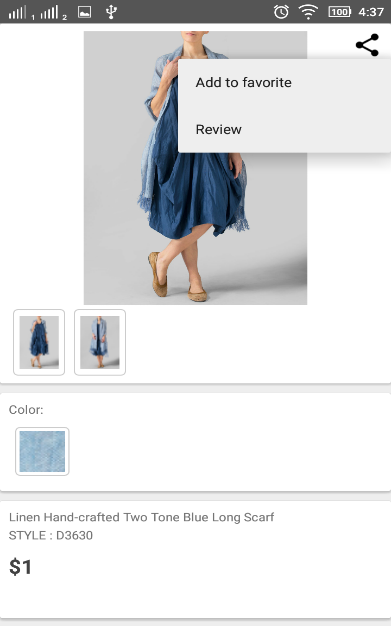
Màn hình chính là màn hình đầu tiên hiển thị sau khi login thành công, tại đay hiển thị các sản phẩm mix của các nhóm sản phẩm.



Hình . Màn hình chính

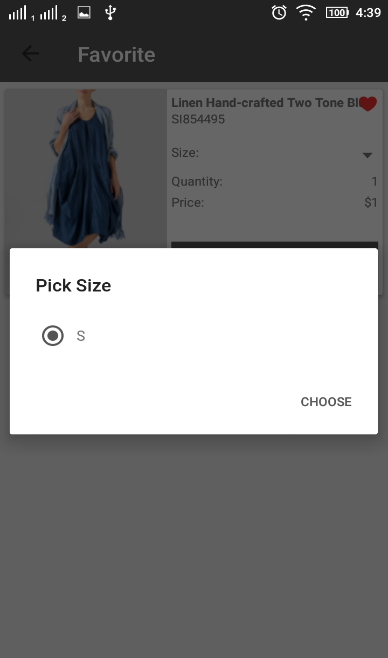
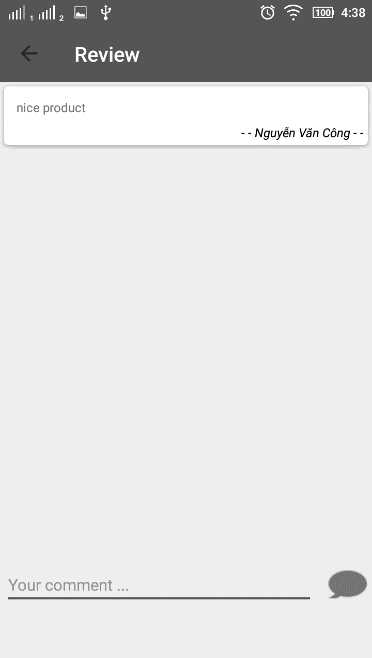
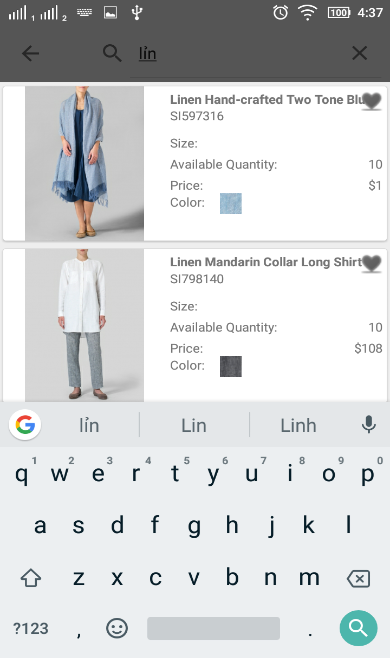
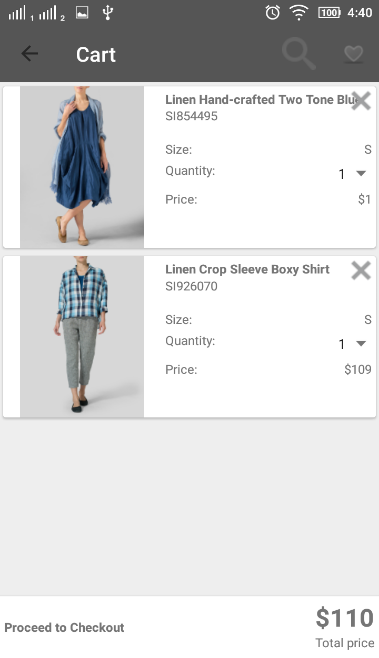
## Màn hình xem sản phẩm

Những màn hình xem sản phẩm cho phép người dùng xem sản phẩm theo mix, nhóm – loại hoặc chi tiết …



Hình . Màn hình xem sản phẩm

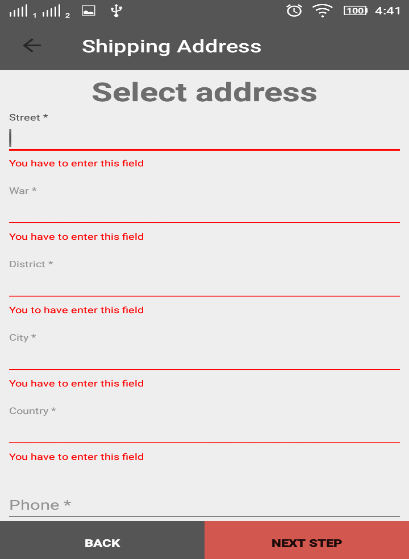
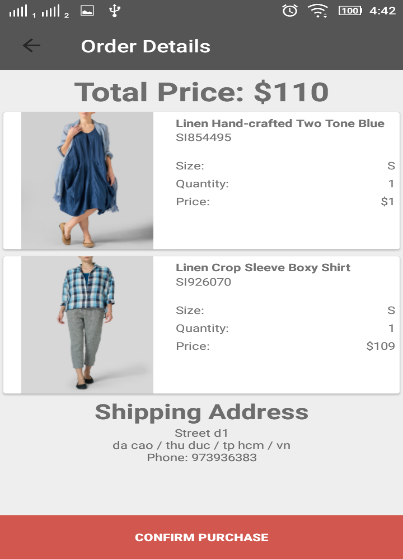
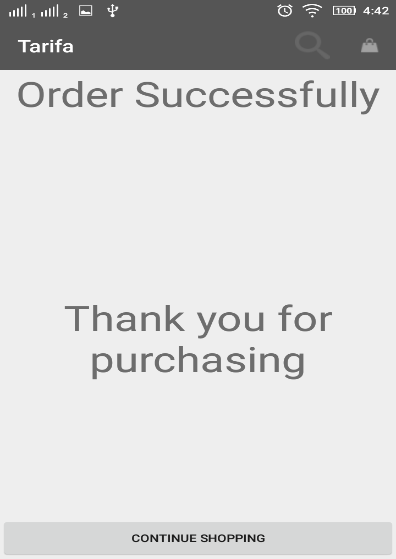
## Màn hình danh mục riêng

Những màn hình này cho phép người dùng tìm kím sản phẩm hay quản lý giỏ hàng, danh sách sản phẩm yêu thích hoặc có thể đánh giá sản phẩm.

Hình . Màn hình danh mục riêng

## Màn hình mua hàng

Những màn hình này tương ứng với quá trình đặt hàng: nhập địa chỉ -> xác nhận -> mua hàng thành công



Hình . Màn hình mua hàng

# Kết luận và kiến nghị

## Các kết luận

* Phát triển được ứng dụng mua sắm trực tuyến cho nhãn hàng Tarifa trên nên tảng Android.
* Giao diện đẹp mắt với material design.
* Cho phép khách hàng tương tác thời gian thực khi đánh giá sản phẩm.
* Hiển thị cho khác hàng xem sản phẩm ở nhiều kiểu khác nhau như : đồ phối, nhóm đồ, loại đồ, chi tiết. sản phẩm có thể yêu thích …
* Có chức năng tim kiếm, lưu lại những sản phẩm yêu thích, giỏ hàng của khách hàng để hiển thị cho khách hàng khi đăng nhập ở một điện thoại khác.
* Dùng spring framework để xây dựng Web service.

## Các hạn chế

* Chưa cho phép dùng tài khoản bên web để đăng nhập vào ứng dụng.
* Quản lý phần đặt hàng của khác hàng riêng với cơ sở dữ liệu bên web.
* Chưa áp dụng được các chính sách của thương mại điện tử như các chính sách chăm sóc khách hàng thân thiết cũng như các phương thức thanh toán bằng cổng thanh toán điện tử.

## Các đề xuất mới

* Xây dựng hệ thống phân tích dữ liệu để biết khách hàng hay vào xem những sản phẩm nào, hoặc phân tích độ tuổi trung bình để đưa ra gợi ý sản phẩm chính xác cho người dùng.
* Xây dựng thêm chức năng phối đồ cho khách hàng : ví dụ khi xem một sản phẩm từ sản phẩm đó có thể phối một cách hợp lý với các sản phẩm khác để ra một bộ đồ hoàn chỉnh.
* Tích hợp thanh toán trực tuyến vào ứng dụng.
* Cho phép đăng nhâp ứng dụng bằng tài khoản của web.
* Phát triển ứng dụng trên nên tảng IOS.

# TÀI LIỆU THAM KHẢO

[1] https://developer.android.com/index.html

[2] http://guides.codepath.com/android/

[3] https://spring.io

[4] <https://github.com/>

[5] <https://console.firebase.google.com>

[6] <https://stackoverflow.com>

[7] Android Programming: The Big Nerd Ranch Guide (3rd Edition) (Big Nerd Ranch Guides)

[8] Head First Android Development: A Brain-Friendly Guide, Dawn Griffiths (Author)

[9] Teach Yourself VISUALLY Android Phones and Tablets (Teach Yourself VISUALLY (Tech)), [Guy Hart-Davis](https://www.amazon.com/s/ref=dp_byline_sr_book_1?ie=UTF8&text=Guy+Hart-Davis&search-alias=books&field-author=Guy+Hart-Davis&sort=relevancerank) (Author)

PHỤ LỤC

Phụ lục: Mô tả usecase

* **Đăng nhập**

|  |  |
| --- | --- |
| Usecase | Đăng nhập |
| Actor | Khách hàng |
| Description | Usecase cho phép khách hàng và người quản trị đăng nhập vào hệ thống. Người quản trị đăng nhập bằng tài khoản google email hoặc facebook. |
| Main flows | 1. Chọn button đăng nhập ở màn hình login. 2. Chọn email hoặc facebook. |
| Alternative flows | 1. Hệ thống sẽ hiển thị thông báo faild login nếu đăng nhập không thành công. 2. Nếu đăng nhập thành công thì sẽ vào màn hình chính. |

Bảng . Mô tả usecase đăng nhập

* **Tìm kiếm trang phục**

|  |  |
| --- | --- |
| Usecase | Tìm kiếm trang phục |
| Actor | Khách hàng |
| Description | Usecase cho phép khách hàng tìm kiếm trang phục yêu thích dựa vào tên trang phục. |
| Main flows | 1. Người tìm kiếm điền tên trang phục vào thanh search. 2. Hệ thống sẽ tự động tìm kiếm các sản phẩm tương ứng với mỗi chữ được nhập vào kết hợp với những từ đã nhập trước đó. |

Bảng . Mô tả usecase tìm kiếm

* **Lưu sản phẩm yêu thích**

|  |  |
| --- | --- |
| Usecase | Add list favorite (thêm danh sách yêu thích) |
| Actor | Khách hàng |
| Description | Usecase cho phép khách hàng thêm các trang phục yêu thích vào một danh sách và hệ thống sẽ lưu trữ lại để khách hàng vẫn có thể xem khi đang nhập ở một điện thoại khác. |
| Main flows | Xem chi tiết sản phẩm và thêm sản phẩm đó vào mục yêu thích. |

Bảng . Mô tả usecase quản lý danh sách sản phẩm yêu thích

* **Đánh giá sản phẩm**

|  |  |
| --- | --- |
| Usecase | Review |
| Actor | Khách hàng |
| Description | Usecase cho phép khách hàng đánh giá sản phẩm tương tác với những khách hàng khác tại một thời điểm |
| Main flows | Xem chi tiết sản phẩm và vào mục review để vào đánh giá sản phẩm. |

Bảng . Mô tả usecase đánh giá sản phẩm