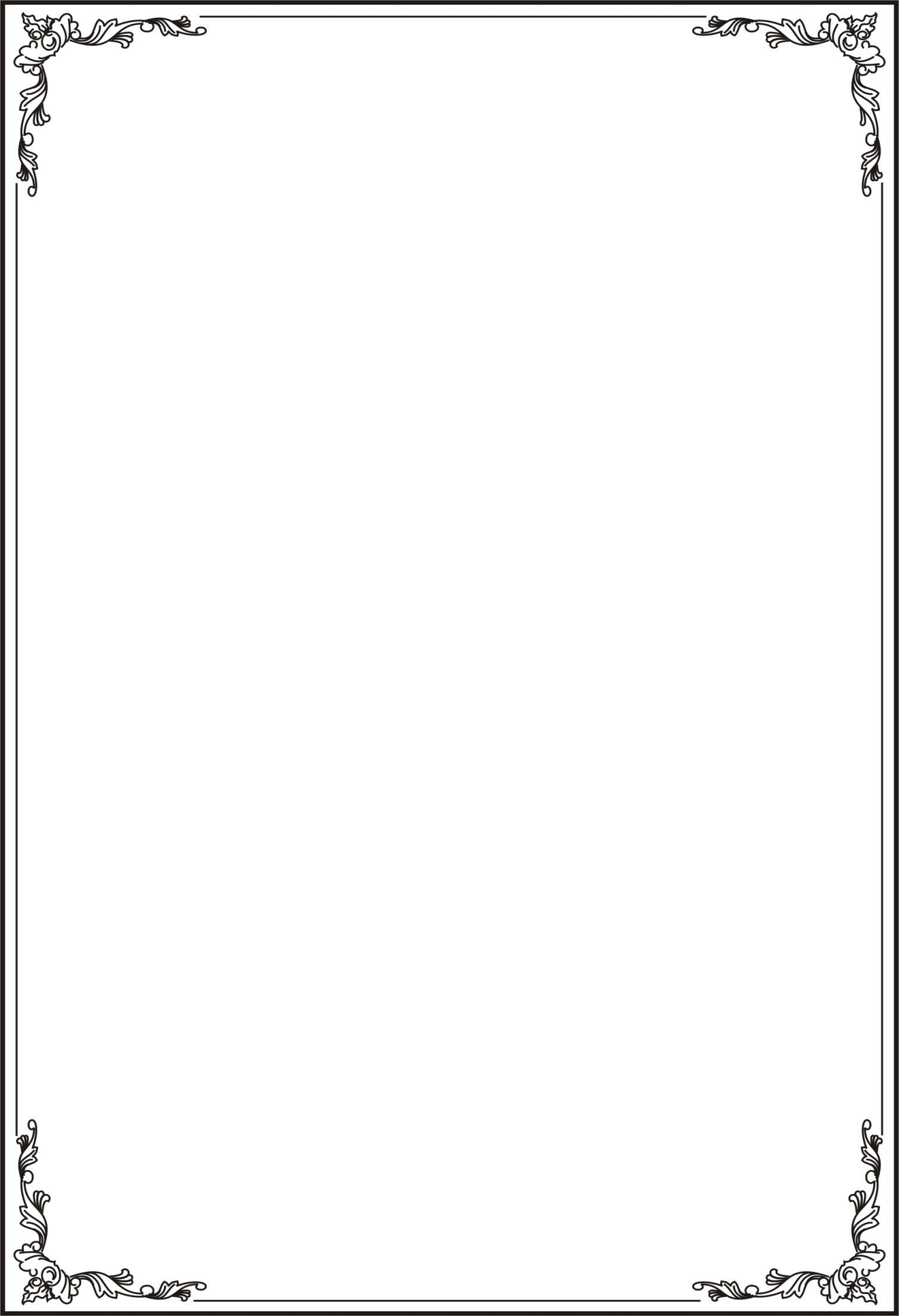
**TRƯỜNG ĐẠI HỌC NÔNG LÂM**

**THÀNH PHỐ HỒ CHÍ MINH**

**KHOA CÔNG NGHỆ THÔNG TIN**

-----🙞🙜🕮🙞🙜-----



***Báo cáo đồ án chuyên ngành***

***ĐỀ TÀI:***

**XÂY DỰNG ỨNG DỤNG THƯƠNG MẠI ĐIỆN TỬ CHO NHÃN HÀNG THỜI TRANG TARIFA TRÊN NỀN TẢNG ANDROID VÀ WEBSERVICE RESTFUL**

Giảng viên hướng dẫn: **Ths. MAI ANH THƠ**

Sinh viên thực hiện : **NGUYỄN VĂN CÔNG**

Mã số sinh viên: **13130136**

Lớp: **DH13DT**

Khóa: **2013-2017**

Mục lục

[MỞ ĐẦU 1](#_Toc492355539)

[Chương 1 HỆ ĐIỀU HÀNH ANDROID VÀ LẬP TRÌNH TRÊN ANDROID 3](#_Toc492355540)

[1. Giới thiệu Android 3](#_Toc492355541)

[1.1. Khái niệm Android 3](#_Toc492355542)

[1.2. Khả năng Android 4](#_Toc492355543)

[1.3. Đặc tính 4](#_Toc492355544)

[2. Lịch sử Android 5](#_Toc492355545)

[3. Kiến trúc Android 6](#_Toc492355546)

[3.1. Applications 6](#_Toc492355547)

[3.2. Application Framework 6](#_Toc492355548)

[3.3. Android Runtime & Libraries 7](#_Toc492355549)

[3.4. Linux Kernel 7](#_Toc492355550)

[4. Lập trình Android 8](#_Toc492355551)

[4.1. Máy ảo Dalvik 8](#_Toc492355552)

[4.2. Các gói Java-Android cần thiết 8](#_Toc492355553)

[4.3. Các thành phần quan trọng trong ứng dụng 9](#_Toc492355554)

[4.4. Activity 9](#_Toc492355555)

[4.5. Service 12](#_Toc492355556)

[4.6. Broadcast Receivers 13](#_Toc492355557)

[4.7. Content Provider 14](#_Toc492355558)

[Chương 2 ỨNG DỤNG MUA SẮM CHO NHÃN HÀNG THỜI TRANG TARIFA 15](#_Toc492355559)

[1. Xác định yêu cầu 15](#_Toc492355560)

[1.1. Yêu cầu chức năng 15](#_Toc492355561)

[1.2. Yêu cầu hệ thống 15](#_Toc492355562)

[2. Mô hình biểu đồ dữ liệu chung 15](#_Toc492355563)

[3. Webservice 15](#_Toc492355564)

[3.1. Phân tích yêu cầu 15](#_Toc492355565)

[3.2. Hướng xây dựng Webservice 16](#_Toc492355566)

[3.3. Class Diagram 16](#_Toc492355567)

[3.4. Chi tiết từng phương thức 17](#_Toc492355568)

[3.5. Sequence Diagram chung 18](#_Toc492355569)

[4. Ứng dụng 18](#_Toc492355570)

[4.1. Tổng quan 18](#_Toc492355571)

[4.2. Usecase 19](#_Toc492355572)

[4.3. Sơ đồ luồng Activity 19](#_Toc492355573)

[4.4. Mô hình MVP 20](#_Toc492355574)

[5. Firebase 21](#_Toc492355575)

[5.1. Authentication 21](#_Toc492355576)

[5.2. Realtime database 21](#_Toc492355577)

[Chương 3 TỔNG KẾT 21](#_Toc492355578)

# MỞ ĐẦU

* **Nội dung đề tài:** Tại IUH, mỗi buổi lên lớp giảng viên điểm danh sinh viên và cập nhật vào hệ thống quản lý sinh viên theo định kỳ. Phần mềm này giúp giảng viên điểm danh bằng cách quét mã vạch trên thẻ sinh viên. Thiết bị dùng để quét thẻ là điện thoại di động sử dụng hệ điều hành Android.
* **Lý do chọn đề tài:**
* Điểm danh sinh viên là một trong những công việc tương đối tốn nhiều thời gian và công sức trong mỗi buổi học. Chính vì vậy, tin học hóa trong lĩnh vực điểm danh sinh viên là một yêu cầu tất yếu. Muốn điểm danh tốt cần có được các phần mềm tốt, phần mềm phải đảm bảo được độ chính xác cao, dễ sử dụng và nhiều tiện ích.
* Đề tài là một yêu cầu thiết thực trong việc điểm danh của hầu hết tất cả các trường học đang hoạt động hiện nay.
* Tính thực tế và hữu ích của nó sẽ giúp các thầy cô quản lý tốt sinh viên của mình trong mỗi buổi học.
* Qua đề tài cũng là cách kiểm tra hiệu quả những kiến thức đã học và học được vô số kiến thức mới. Cũng là cách tiếp cận với thực tế có hiệu quả nhất đối với em. Giúp em có nền tảng về quy trình nghiệp vụ xây dựng một phần mềm.
* **Các bước thực hiện:**
* Tìm hiểu về hệ điều hành Android và cách lập trình trên Android.
* Tìm hiểu các loại mã vạch và các phần mềm nguồn mở đọc mã vạch dùng cho điện thoại sử dụng hệ điều hành Android.
* Tìm hiểu về cách tương tác để lấy dữ liệu và cập nhật dữ liệu vào hệ thống quản lý sinh viên của nhà trường.
* Xây dựng phần mềm điểm danh.
* Thực hiện các chức năng hổ trợ cho người dùng: import/ export file, các thống kê.
* Testing và viết báo cáo.
* **Bố cục:**

**Chương 1 MÃ VẠCH, ỨNG DỤNG TRONG PHẦN MỀM ĐIỂM DANH SINH VIÊN**

Trình bày các kiến thức cơ bản về mã vạch cũng như các loại mã vạch hiện có trên thế giới. Phân loại các chức năng, công dụng và cách xử lý của từng loại mã vạch

**Chương 2 HỆ ĐIỀU HÀNH ANDROID VÀ LẬP TRÌNH TRÊN ANDROID**

Tìm hiểu về hệ điều hành Android, lịch sử hình thành, cách thức hoạt động của Android. Tiếp cận hệ thống Android dưới con mắt của một người lập trình để có những cái nhìn tổng quan nhất “Android là gì”.

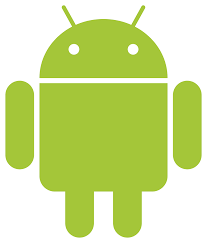
**Chương 3 PHẦN MỀM ĐIỂM DANH SINH VIÊN BẰNG MÃ VẠCH**

Phát biểu bài toán, phân tích và thiết kế hệ thống dựa trên các nghiệp vụ đưa ra. Thiết kế mô hình dữ liệu, các giao thức truyền tải dữ liệu, xây dựng cơ sở dữ liệu, xây dựng hoàn chỉnh phần mềm quản lý và điểm danh sinh viên bằng mã vạch.

**NỘI DUNG**

**Chương 1 HỆ ĐIỀU HÀNH ANDROID VÀ LẬP TRÌNH TRÊN ANDROID**

1. **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 1.1 Biểu tượng Android.

* 1. **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.
  1. **Khả năng Android**
     1. **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.

* + 1. **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.

* + 1. **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.

* 1. **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.

1. **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 2.1 Điện thoại Android. HTC Dream (trái), HTC Magic ( Phải)

1. **Kiến trúc Android**

Hình 3.1: Sơ đồ các thành phần chính của hệ điều hành Android

Hệ điều hành Android được xây dựng dựa trên 4 tầng cơ bản: Applications, Application Framework, Android Runtime & Libraries, Linux Kernel. Mỗi tầng làm việc đều được sự hỗ trợ từ tầng bên dưới nó.

* 1. **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.

* 1. **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.

* 1. **Android Runtime & Libraries**
     1. **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.
  + 1. **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
  1. **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.

1. **Lập trình Android**
   1. **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.
  1. **Các gói Java-Android cần thiết**
     1. **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.
  + 1. **Các gói Android** [[10](#exp)]
* 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.
  1. **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.
  1. **Activity**
     1. **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…
  + 1. **Vòng đời của Activity**

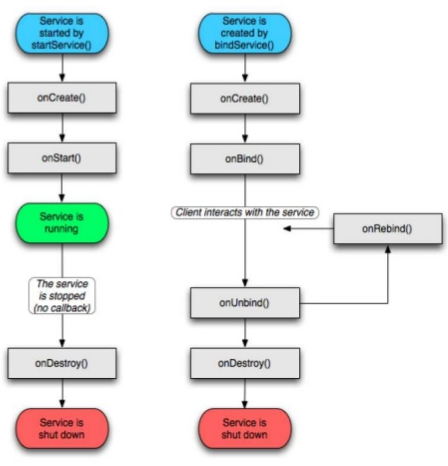
Các Activity trong hệ thống được quản lý bởi một cấu trúc dữ liệu ngăn xếp. Khi có một Activity được khởi tạo, nó được đẩy vào trong ngăn xếp, chuyển sang trạng thái thực thi và Activity trước đó sẽ chuyển sang trạng thái chờ. Activity này chỉ trở lại trạng thái thực thi khi mà Activity vừa khởi tạo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



Hình 4.1: Vòng đời của Activity

* + 1. **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 4.2: 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.
  1. **Service**
     1. **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).
  + 1. **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 4.3: Sơ đồ chuyển trạng thái của Service

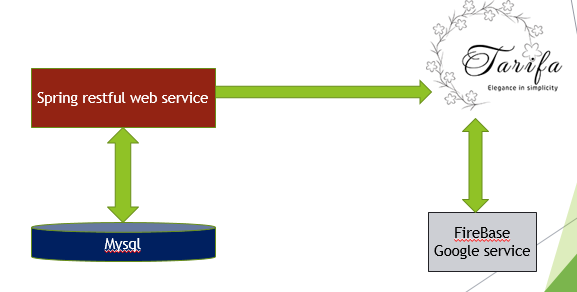
* 1. **Broadcast Receivers**
     1. **Khái niệm**
* Broadcast Receivers là một thành phần không làm gì cả nhưng nó nhận và phản hồi lại các thông báo. Các thông báo bắt nguồn từ mã hệ thống – như pin yếu, hay có một ảnh mới vừa chụp, … Một ứng dụng thông thường cũng có thể phát ra các thông báo – ví 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.
  1. **Content Provider**
     1. **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.

**Chương 2** **ỨNG DỤNG MUA SẮM CHO NHÃN HÀNG THỜI TRANG TARIFA**

1. **Xác định yêu cầu**
   1. **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.
* Đồng bộ hóa dữ liệu giữa web và ứng dụng.
* 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.
  1. **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.

1. **Mô hình biểu đồ dữ liệu chung**

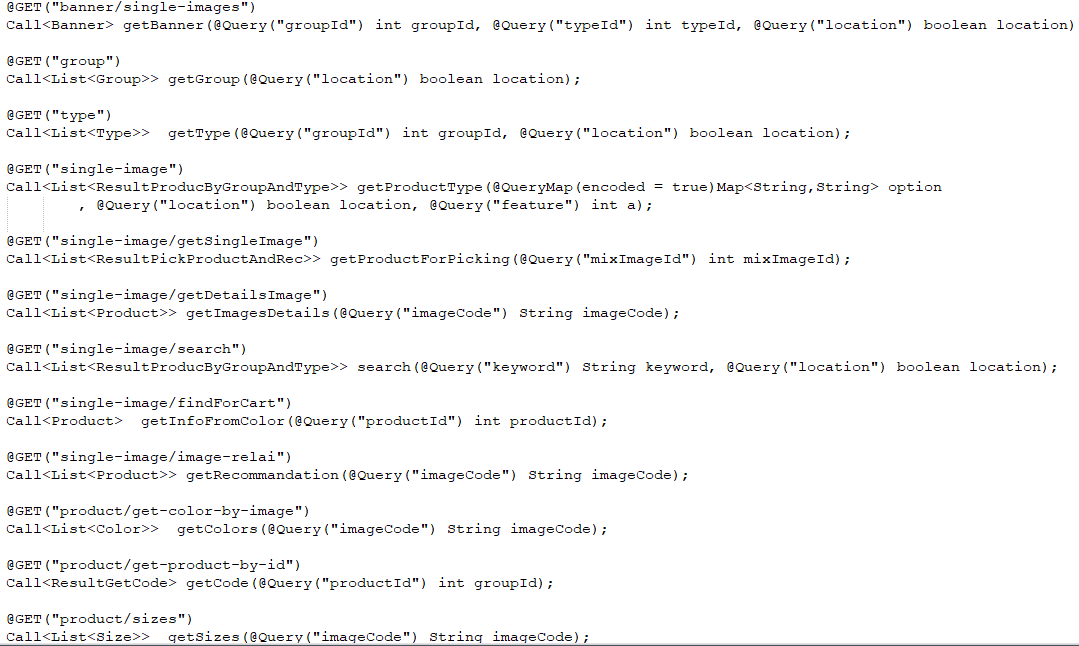


Hình 2.1: Biểu đồ dữ liệu

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.

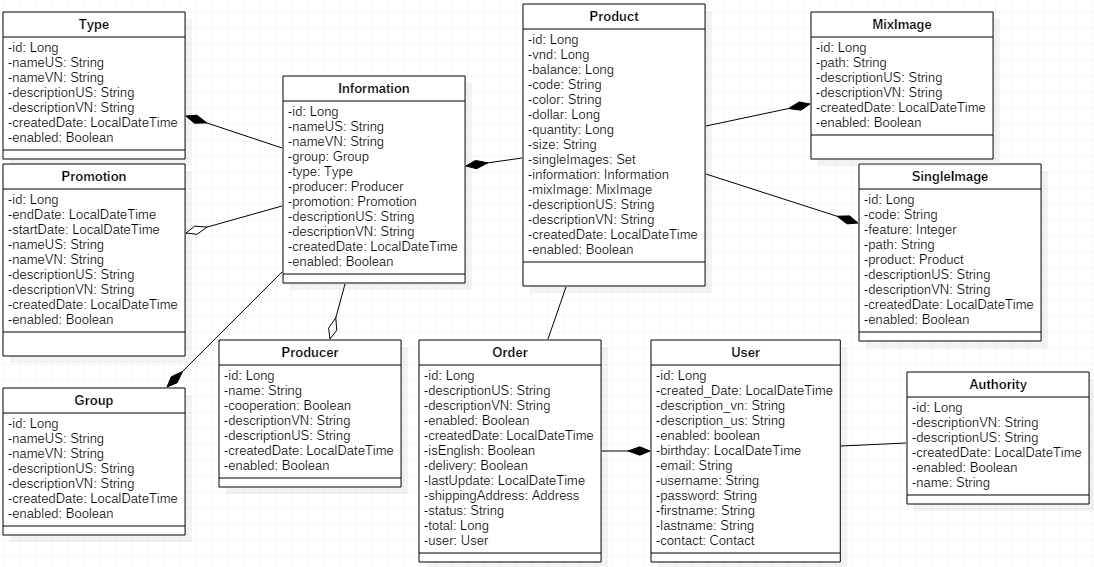
1. **Webservice**
   1. **Phân tích yêu cầu**

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.

* 1. **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.

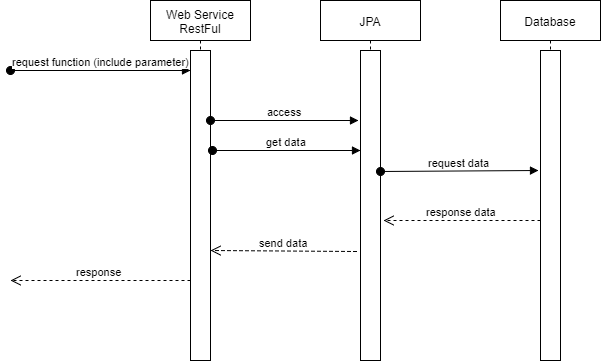
Hình 4.2: Danh sách hàm của Webservice

* 1. **Class Diagram**



Hình 4.5: Class Diagram Webservice

* 1. **Chi tiết từng phương thức**
     1. **getBanner**
* Phương thức này dùng để trả về thông tin banner cho phần xem sản phẩm theo loại.
  + 1. **getGroup**
* Phương thức này dùng để lấy danh sách các nhóm.
  + 1. **getType**
* Phương thức này dùng để trả về danh sách các type có trong một nhóm nhất định.
  + 1. **getProductType**
* Phương thức này dùng để trả về danh sách các sản phẩm có trong một type.
  + 1. **getProductForPicking**
* Phương thức này dùng để trả về danh sách các sản phẩm có trong một sản phẩm phối.
  + 1. **getImagesDetails**
* Phương thức này dùng để trả về danh sách các hình ảnh góc nhìn của một sản phẩm.
  + 1. **Search**
* Phương thức này dùng để tìm kiếm sản phẩm theo tên.
  + 1. **getInfoFromColor**
* Phương thức này dùng để trả về thông tin của một sản phẩm theo một màu nất đinh.
  + 1. **getRecommandation**
* Phương thức này dùng để trả về những sản phẩm gợi ý cho người dùng khi đang xem một sản phẩm nào đó.
  + 1. **getColors**
* Phương thức này dùng để trả về tất cả mày của một sản phẩm nhất định.
  + 1. **getCode**
* Phương thức này dùng để trả về mã code của sản phẩm dựa trên id sản phẩm đó.
  + 1. **getSizes**
* Phương thức này dùng để trả về tất cả các size của một sản phẩm.
  1. **Sequence Diagram chung**



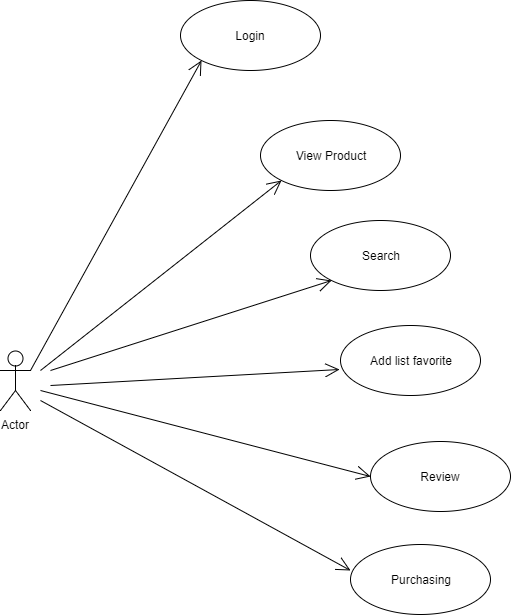
Hình 4.4: Sequence Diagram Webservice

1. **Ứng dụng**
   1. **Tổng quan**

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

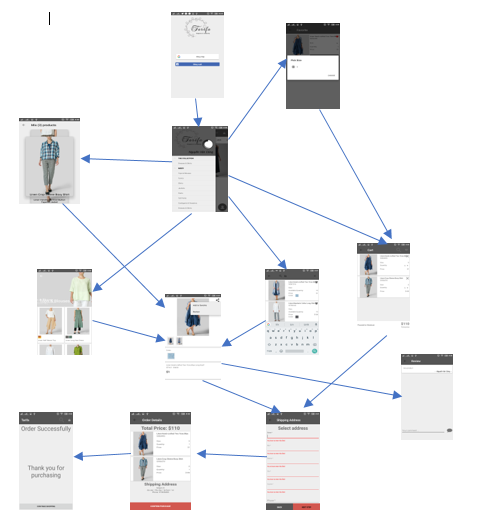
* 1. **Usecase**

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

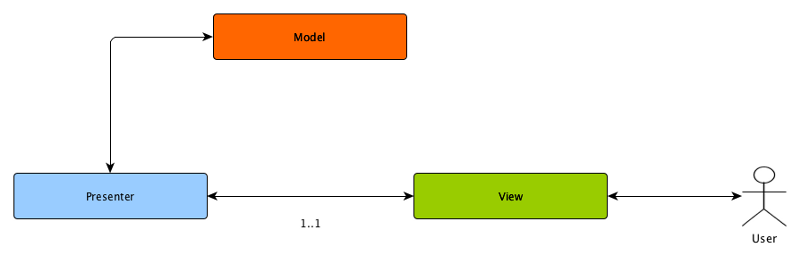


Hình 5.2: Usecase ứng dụng

* 1. **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.



Hình 5.4: Luồng Activity

* 1. **Mô hình MVP**
* 

Hình 5.5 : 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.

1. **Firebase**
   1. **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.
  1. **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.

**Chương 3 TỔNG KẾT**

1. **Kết quả đạt được**

* Hoàn thành đầy đủ chức năng cũng như yêu cầu nghiệp vụ mà đề tài đưa ra.
* Tạo được ứng dụng có giao diện đẹp mắt, khả năng tương tác cao.

1. **Hạn chế**

* Chưa đồng bộ được tài khoản web và ứng dụng khi đăng nhập.
* Chưa tích hợp cổng thanh toán trực tuyến.

1. **Hướng phát triển**

* Cải thiện UI/UX.
* Tích hợp thanh toán trực tuyến.