TRƯỜNG ĐẠI HỌC BÁCH KHOA HÀ NỘI

Viện công nghệ thông tin và truyền thông

──────── \* ───────



Báo cáo bài tập lớn

Project 2

Đề tài: Xây dựng ứng dụng tìm kiếm cây ATM trên hệ điều hành Android

Giảng viên hướng dẫn: Ts. Trịnh Thành Trung

Sinh viên thực hiện: Dương Thế Cường 20140557

***Hà Nội, tháng 5 năm 2017***

# Lời nói đầu

Hiện nay, với sự phát triển nhanh chóng của khoa học kỹ thuật, đặc biệt là ngành công nghệ thông tin, làm từng bước thay đổi cuộc sống của con người bởi các sản phẩm của ứng dụng công nghệ. Điện thoại thông minh đã trở thành vật bất ly thân đối với mỗi người, với giá thành ngày càng rẻ và các lợi ích của nó mang lại rất phong phú cho nên điện thoại thông minh ngày càng được sử dụng rộng rãi và phổ biến.

Việc sử dụng điện thoại thông minh ngoài khả năng hỗ trợ nghe, gọi, hỗ trợ trong công việc, giải trí mà còn giúp con người tiết kiệm thời gian hơn vào nhiều công việc thực tiễn trong đời sống hàng ngày. Hiện này, một hình thức lưu trữ tiền khá phổ biến trong cuộc sống đó là thẻ rút tiền ATM, nó được dùng để chi trả lương cho nhân viên của các công ty, doanh nghiệp, hay dùng để gửi tiền cho các sinh viên sống xa nhà. Vì vậy, việc tìm kiếm các địa điểm để dùng thẻ là khá hữu dụng trong cuộc sống, khi mà trong một số trường hợp chúng ta rất khó có thể xác định được địa điểm đặt cây ATM trong thành phố.Vì vậy, em đã lựa chọn đề tài “Xây dựng ứng dụng tìm kiếm máy rút tiền ATM trên hệ điều hành Android” để làm bài tập lớn.

Trong quá trình làm bài tập lớn, em đã nhận được sự hướng dẫn, chỉ bảo tận tình của thầy **Trịnh** **Thành** **Trung** đã giúp đỡ em hoàn thành bài tập lớn này. Tuy đã cố gắng tìm hiểu và phân tích nhưng chắc chắn rằng không tránh khỏi những thiếu sót, em rất mong nhận được sự thông cảm và góp ý của Thầy cùng các bạn để sản phẩm của em được hoàn thiện hơn.

Em xin chân thành cảm ơn !

# Mục lục

[Lời nói đầu 2](#_Toc482854708)

[Mục lục 3](#_Toc482854709)

[Danh mục hình ảnh 5](#_Toc482854710)

[Chương 1 Giới thiệu về đề tài 6](#_Toc482854711)

[1.1 Mục tiêu, nhiệm vụ 6](#_Toc482854712)

[1.2 Đối tượng nghiên cứu và phạm vi của hệ thống 6](#_Toc482854713)

[Chương 2 Hệ điều hành Android 7](#_Toc482854714)

[2.1 Giới thiệu hệ điều hành Anhdroid 7](#_Toc482854715)

[2.2 Những đặc trưng của hệ điều hành Android 7](#_Toc482854716)

[2.3 Kiến trúc hệ điều hành Android 8](#_Toc482854717)

[2.3.1 Linux Kernel (nhân Linux) 8](#_Toc482854718)

[2.3.2 Libraries 8](#_Toc482854719)

[2.3.3 Android Runtime 9](#_Toc482854720)

[2.3.4 Application Framework 9](#_Toc482854721)

[2.3.5 Applications 10](#_Toc482854722)

[2.4 Sơ lược về các phiên bản Android 10](#_Toc482854723)

[2.5 Các thành phần cơ bản của ứng dụng Android 11](#_Toc482854724)

[2.5.1 Activity 11](#_Toc482854725)

[2.5.2 Service 13](#_Toc482854726)

[2.5.3 Content provider 13](#_Toc482854727)

[2.5.4 Intent 13](#_Toc482854728)

[2.5.5 Broadcast Receiver 14](#_Toc482854729)

[2.5.6 View 14](#_Toc482854730)

[2.5.7 Notification 15](#_Toc482854731)

[Chương 3 JSON và XML 16](#_Toc482854732)

[3.1 JSON 16](#_Toc482854733)

[3.2 XML 17](#_Toc482854734)

[Chương 4 Hệ quản trị cơ sở dữ liệu SQLite 19](#_Toc482854735)

[4.1 Giới thiệu SQLite 19](#_Toc482854736)

[4.2 Các đặc điểm chính của SQLite 19](#_Toc482854737)

[4.3 Những trường hợp thích hợp sử dụng SQLite 20](#_Toc482854738)

[4.4 SQLite trong Android 20](#_Toc482854739)

[Chương 5 Google Maps và Google Maps API Web Services 21](#_Toc482854740)

[5.1 Google Maps 21](#_Toc482854741)

[5.1.1 Giới thiệu về Google Maps 21](#_Toc482854742)

[5.1.2 Google Maps for Mobile 22](#_Toc482854743)

[5.2 Google Maps API Web Service 22](#_Toc482854744)

[5.2.1 Giới thiệu về Google Maps API Web Service 22](#_Toc482854745)

[5.2.2 Sử dụng Google Maps API Web Service 22](#_Toc482854746)

[5.3 Google Geocoding API 23](#_Toc482854747)

[5.3.1 Mã hóa địa lý 23](#_Toc482854748)

[5.3.2 Google Geocoding API request 23](#_Toc482854749)

[5.3.3 Kết quả trả về dạng JSON 24](#_Toc482854750)

[5.4 Google Direction API 27](#_Toc482854751)

[5.4.1 Google Direction API request 27](#_Toc482854752)

[5.4.2 Kết quả trả về dạng JSON 28](#_Toc482854753)

[Chương 6 Phân tích thiết kế hệ thống 37](#_Toc482854754)

[6.1 Yêu cầu của hệ thống 37](#_Toc482854755)

[6.2 Phân tích thiết kê. 38](#_Toc482854756)

[6.2.1 Biểu đồ use case. 38](#_Toc482854757)

[6.2.2 Biểu đồ trình tự 40](#_Toc482854758)

[6.2.3 Biểu đồ giao tiếp 41](#_Toc482854759)

[6.2.3 Biểu đồ hoạt động 42](#_Toc482854760)

[6.2.4 Xây dựng cơ sở dữ liệu 42](#_Toc482854761)

[6.3 Chương trình minh họa 43](#_Toc482854762)

[Kết luận 46](#_Toc482854763)

[Tài liệu tham khảo 47](#_Toc482854764)

# Danh mục hình ảnh

[**Hình 1 Kiến trúc của hệ điều hành Android** 8](#_Toc482697240)

[**Hình 2 Các phiên bản Android** 11](#_Toc482697241)

[**Hình 3 Vòng đời của Activity** 12](#_Toc482697242)

[**Hình 4 Các loại Intent trong Android** 14](#_Toc482697243)

[**Hình 5 Sơ đồ phân cấp View trong Android** 15](#_Toc482697244)

[**Hình 6 JSON dạng Object** 16](#_Toc482697245)

[**Hình 7 JSON dạng Array** 17](#_Toc482697246)

[**Hình 8 Cấu trúc của tài liệu XML** 18](#_Toc482697247)

[**Hình 9 Google Maps** 21](#_Toc482697248)

[**Hình 10 Biểu đồ use-case tổng quan** 38](#_Toc482697249)

[**Hình 11 Biểu đồ use-case cấp 2 (Tìm kiếm ATM)** 38](#_Toc482697250)

[**Hình 12 Biểu đồ use-case cấp 2 (Cài đặt)** 39](#_Toc482697251)

[**Hình 13 Biểu đồ trình tự tìm đường đi** 40](#_Toc482697252)

[**Hình 14 Biểu đồ trình tự tìm kiếm cây ATM** 40](#_Toc482697253)

[**Hình 15 Biểu đồ giao tiếp tìm đường đi** 41](#_Toc482697254)

[**Hình 16 Biểu đồ giao tiếp tìm kiếm cây ATM** 41](#_Toc482697255)

[**Hình 17 Biểu đồ hoạt động tìm kiêm cây ATM** 42](#_Toc482697256)

[**Hình 18 Lớp ATM** 42](#_Toc482697257)

[**Hình 19 Demo cơ sở dữ liệu** 43](#_Toc482697258)

[**Hình 20 Giao diện chính của ứng dụng** 43](#_Toc482697259)

[**Hình 21 Giao diện tìm kiếm** 44](#_Toc482697260)

[**Hình 22 Giao diện cài đặt** 45](#_Toc482697261)

# Chương 1 Giới thiệu về đề tài

## Mục tiêu, nhiệm vụ

Tìm hiểu được các công nghệ lập trình cần thiết trên nền tảng Android để hoàn thành được ứng dụng có thể tìm kiếm, chỉ đường nhanh và chính xác đến vị trí của cây ATM.

Hiểu về nguyên lý hoạt động của hệ điều hành Android, các thành phần của hệ thống cũng như ứng dụng thực thi trên nền Android, tìm hiểu ngôn ngữ lập trình trên nền Android như ngôn ngữ Java, tương tác với các tài nguyên của hệ thống như GPS, Internet, sử dụng các dịch vụ của Google như Google Map, Google Directions API, Google Geocoding API.

## Đối tượng nghiên cứu và phạm vi của hệ thống

Đối tượng nghiên cứu là:

Lập trình Java trên nền Android.

Ngôn ngữ Java,Json, Xml.

Hệ cơ sở dữ liệu SQlite.

Công cụ Android SDK và Android Studio.

Hệ điều hành và tài nguyên của hệ thống Android.

Các dịch vụ của Google như Google Map, Google Directions API, Google Geocoding API.

Phạm vi: Đề tài chỉ dừng lại ở việc tìm kiếm địa chỉ đặt các cây ATM theo quận, tên ngân hàng ở Hà Nội, ứng dụng để tìm kiếm và chỉ đường từ vị trí người dùng đến vị trí cây ATM. Ứng dụng được viết trên nền Android 4.3 API 18.

# Chương 2 Hệ điều hành Android

## 2.1 Giới thiệu hệ điều hành Anhdroid

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 (và hiện nay cả trên một số đầu phát HD, HD Player, smartTV, oto …). Ban đầu, Android được phát triển bởi tổng công ty Android và sau này được Google mua lại vào năm 2005 như một phần chiến lược để Google lấn sâu vào lĩnh vực di động.Đây là con át chủ bài của Google để cạnh tranh thị phần hệ điều hành di động với Apple. Tháng 10 năm 2008, chiếc điện thoại đầu tiên chạy Android được bán ra ngoài thị trường.

Google muốn Android trở thành nền tảng mở và miễn phí, do vậy hầu hết các mã nguồn của Android được phát hành theo giấy phép mã nguồn mở Apache. Chính mã nguồn mở cùng với giấy phép không có nhiều ràng buộc đã cho phép các nhà phát triển thiết bị, mạng di động và các lập trình viên được điều chỉnh và phân phối Android một cách tự do. Hơn nữa, các nhà sản xuất phần cứng có thể thêm các thành phần mở rộng độc quyền của hãng hoặc tùy biến cho Android để tạo ra các sản phẩm khác nhau. Chính mô hình phát triển đơn giản này khiến Android trở nên rất hấp dẫn do đó khơi dậy sự quan tâm của nhiều nhà cung cấp.

Tháng 10 năm 2012, có khoảng 700 000 ứng dụng trên Android và số lượt tải trên Google Play ước tính khoảng 25 tỷ lượt. Vào quý 4 năm 2010 Android vượt Symbian trở thành nền tảng điện thoại thông minh phổ biến nhất thế giới. Android chiếm 75% thị phần điện thoại thông minh trên toàn thế giới vào quý 3 năm 2012, với tổng cộng 500 triệu thiết bị được kích hoạt và 1,3 triệu lượt kích hoạt mỗi ngày.

Ưu điểm chính của việc áp dụng Android là nó cung cấp một cách tiếp cận thống nhất để phát triển ứng dụng. Các nhà phát triển chỉ cần phát triển cho Android và các ứng dụng của họ có thể chạy trên nhiều thiết bị khác nhau. Trong thế giới điện thoại di động thông minh, ứng dụng là một phần quan trọng nhất mang đến thành công. Do đó, các nhà sản xuất thiết bị coi Andoid là hy vọng tốt nhất để thách thức IOS của Apple.

## 2.2 Những đặc trưng của hệ điều hành Android

**Aplication** **framework**: cho phép sử dụng lại và thay thế các thành phần trong lập trình ứng dụng.

**Dalvik** **virtual machine**: tối ưu hóa cho thiết bị di động.

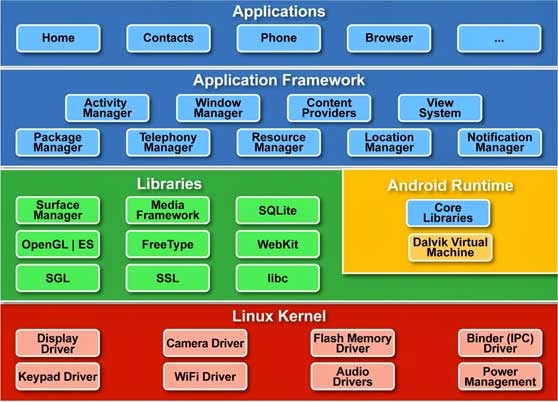
**Intergrated browser:** trình duyệt tích hợp, dựa trên cơ chế WebKit mã nguồn mở.

**SQLite**: cơ sở dữ liệu trong môi trường di động.

**Media support**: hỗ trợ các định dạng audio, video và các định dạng hình ảnh.

**GSM Telephony**: mạng điện thoại di động (phụ thuộc vào phần cứng).

## 2.3 Kiến trúc hệ điều hành Android



**Hình 1 Kiến trúc của hệ điều hành Android**

### 2.3.1 Linux Kernel (nhân Linux)

Android có một hạt nhân dựa trên nhân Linux phiên bản 2.6, kể từ Android 4.0 Ice Cream Sandwich trở về sau là phiên bản 3.x.

Kernel Linux hoat động như một lớp trừu tượng hóa giữa phần cứng và tầng dưới của phần mềm. Lớp này chứa tất cả các thiết bị mức thấp điều khiển các thành phần cứng khác nhau của một thiết bị Android.

### 2.3.2 Libraries

Libraries bao gồm một tập hợp các thư viện lập trình chứa mã lệnh cung cấp những tính năng và thao tác chính trên hệ điều hành. Một số thư viện cơ bản được liệt kê dưới đây:

**System C Library**: a BSD-derived triển khai các thư viện hệ thống ngôn ngữ C chuẩn, được nhúng vào các thiết bị dựa trên hệ điều hành Linux.

**Media Libraries**: dựa trên PacketVideo’s OpenCORE, thư viện này hỗ trợ cho việc chơi nhạc, quay phim, chụp hình theo các định dạng file MPEG4, MP3, AAC, JPG và PNG.

**Surface Manager**: quản lý truy cập đến các hệ thống con hiển thị cũng như các lớp đồ họa 2D, 3D từ tầng ứng dụng.

**LibWebCore**: thư viện được dùng để tạo nên thành phần webview trong Anhdroid và có thể được nhúng vào nhiều ứng dụng.

**SGL**: thư viện hỗ trợ đồ họa 2D.

**3D Libraries**: thư viện hỗ trợ đồ họa 3D.

**FreeType**: bitmap and vector font rendering.

**SQLite**: Một cơ sở dữ liệu được dùng cho các thiết bị cầm tay có bộ nhớ hạn chế. SQLite không có quan hệ như các cơ sở dữ liệu khác.

**SSL**: chịu trách nhiệm về bảo mật Internet.

### 2.3.3 Android Runtime

Nằm cùng một tầng với Libraries, Android Runtime cung cấp một bộ phận quan trọng là Dalvik Vitual Machine là 1 loại Java Virtual Machine được thiết kế đặc biệt để tối ưu cho Android.

Dalvik Vitual Machine sử dụng các đặc trưng của nhân Linux như quản lý bộ nhớ và đa luồng, những thứ đã có sẵn trong Java. Dalvik Vitual Machine cho phép tất cả các ứng dụng Android chạy trong tiến trình riêng của nó.

Android Runtime cũng cung cấp một tập các thư viện cho phép các nhà phát triển ứng dụng Android viết các ứng dụng Android sử dụng ngôn ngữ lập trình Java.

### 2.3.4 Application Framework

Application Framework cung cấp các dịch vụ ở cấp độ cao hơn cho các ứng dụng dưới dạng các lớp Java. Các nhà phát triển ứng dụng được phép sử dụng các dịch vụ này trong ứng dụng của mình.

Application Framework cung cấp các dịch vụ chính sau:

**Activity Manager**: kiểm soát vòng đời của ứng dụng và điều hướng các activity.

**Content Providers**: cho phép các ứng dụng chia sẻ dữ liệu với các ứng dụng khác.

**Resource Manager**: cung chấp quyền truy xuất vào các tài nguyên của hệ thống.

**Notificatons Manager**: cho phép ứng dụng hiển thị thông báo với người dùng.

**View System**: bao gồm các thành phần giao diện được sử dụng để tạo giao diện và để người dùng tương tác.

### 2.3.5 Applications

Lớp trên cùng sẽ là các ứng dụng cho Android như gọi điện, nhắn tin, danh bạ, trình duyệt… cũng như các ứng dụng được tải về và cài đặt hay bất kỳ ứng dụng nào bạn viết được cài đặt trên tầng này.

## 2.4 Sơ lược về các phiên bản Android

Alpha – Android 1.0 (23/09/2008) : Phát hành phiên bản Android 1.0 trên điện thoại Android đầu tiên HTC Dream (G1).

Beta – Android 1.1 (09/02/2009) : phát hành phiên bản cập nhật 1.1 riêng cho T-Mobile G1.

Kể từ tháng 04/2009 các phiên bản Android được đặt tên theo tên một loại bánh kẹo và phát hành theo thứ tự bảng chữ cái.

Cupcake – Android 1.5 (30/04/2009).

Donut – Android 1.6 (15/09/2009).

Eclair – Android 2.0 (26/10/2009).

Froyo – Android 2.2 (20/05/2010).

Gingerbread – Android 2.3 (06/12/2010).

Honeycomb – Android 3.0 (22/02/2011).

IceCream Sandwich – Android 4.0 (18/10/2011).

Jelly Bean – Android 4.1 (09/07/2012).

Kitkat – Android 4.4 (31/10/2013).

Lollipop – Android 5.0 (12/11/2014).

Marshmallow – Android 6.0 (05/10/2015).

Nougat – Android 7.0 (22/08/2016).

Hiện nay Google dự kiến công bố Android O (Android 8.0) vào tháng 5/2017, sau đó sẽ tung bản thử nghiệm vào mùa thu.



**Hình 2 Các phiên bản Android**

## 2.5 Các thành phần cơ bản của ứng dụng Android

### 2.5.1 Activity

Là lớp thể hiện của ứng dụng. Mỗi màn hình trong ứng dụng là một mở rộng của lớp Activity. Các Activity sử dụng các View tạo thành giao diện đồ họa tương tác người dùng để thể hiện thông tin và phản hồi hành động của người dùng.

Một ứng dụng có thể gồm một hay nhiều Activity. Activity chính phải được hiển thị đầu tiên khi khởi động ứng dụng. Chuyển từ một Activity này sang Activity khác bằng cách cho Activity hiện thời khởi động Activity kế tiếp. Mặc dù chúng cùng làm việc với nhau để tạo thành một giao diện người dùng thế nhưng mỗi Activity lại độc lập với nhau.

Một Activity có thể là một cửa sổ toàn màn hình, nhưng nó có thể nhỏ hơn màn hình và nằm trên các cửa sổ khác.

Khi ứng dụng được mở thì Activity chính sẽ được tạo và them vào stack, khi Activity mới được tạo, nó sẽ được đặt trên cùng của stack. Chỉ có một Activity duy nhất ở trên cùng là hiển thị nội dung với người dùng, các Activity còn lại sẽ bị tạm dừng và chỉ hoạt động trở lại khi Activity mới được giải phóng.

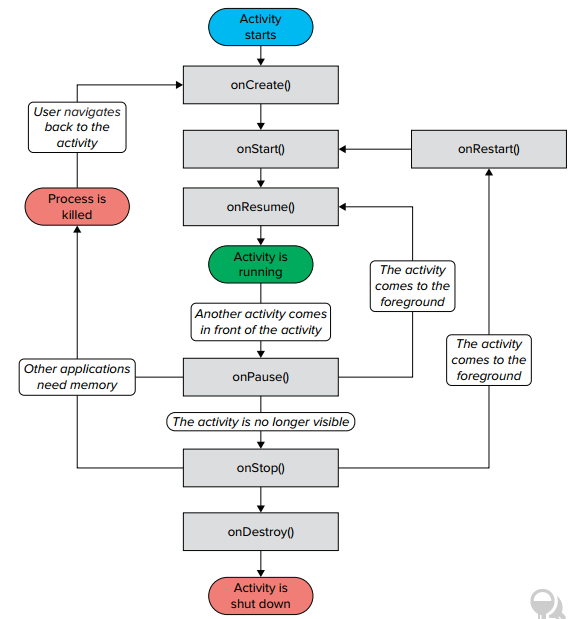
Một Activity có bốn trạng thái chính:

**Active** hay **Running**: Activity đang chạy trên màn hình.

**Pause**: Activity vẫn hiển thị trên màn hình nhưng không thể tương tác. Ví dụ bị một Activity trong suốt hoặc Activity không full màn hình đè lên.

**Stop**: Khi một Activity bị che khuất hoàn toàn bởi một Activity khác. (Vẫn tồn tại nhưng thường bị kill để giảm bộ nhớ).

**Killed** hay **Shutdown**: Khi một Activity kết thúc và bị xóa khỏi bộ nhớ.

**  
Hình 3 Vòng đời của Activity**

Tùy từng sự kiện mà các phương thức được gọi để đảm bảo trạng thái các Ativity được hoạt động đúng. Chi tiết các phương thức và sự kiện thay đổi trạng thái tương ứng:

**onCreate():** Phương thức này khởi tạo Activity.

**onRestart():** Được gọi sau khi một Activity dừng và muốn bắt đầu lại.

**onStart():** Được gọi khi Activity chuyển sang trạng thái có thể nhìn thấy nhưng chưa tương tác được.

**onResume():** Activity có thể tương tác được với người dùng.

**onPause():** Khi một Activity khác đè lên Activity hiện tại, thì phương thức này được Activity hiện tại gọi.

**onStop():** Khi Activity không còn hiển thị với người dùng.

**onDestroy():** Được gọi trước khi hệ thống giải phóng bộ nhớ hay ứng dụng loại bỏ Activity.

Một Activity có ba vòng đời chính:

**Entire Lifetime:** diễn ra từ lần gọi onCreate() đầu tiên và kéo dài tới lần gọi onDestroy() cuối cùng.

**Visible Lifetim:** diễn ra từ khi gọi onStart() và kéo dài tới khi gọi onStop(). Ở vòng đời này, Activity được hiển thị trên màn hình mặc dù nó có thể không tương tác được.

**Foreground Lifetime:** diễn ra từ khi gọi onResume() và kéo dài tới khi gọi onPause(). Ở vòng đời này Activity nằm trên một Activity khác và có thể tương tác được với người dùng.

### 2.5.2 Service

Một Service không có giao diện trực quan, nó dùng để thực thi một công việc dưới nền hệ điều hành (ẩn với người dùng) trong một khoảng thời gian không xác định.

Giống như Activity và các thành phần khác, Service chạy trong thread chính của ứng dụng. Nó thường dành cho những công việc mất nhiều thời gian để thực thi hoặc để điều khiển thực thi bởi một tiến trinh khác.

### 2.5.3 Content provider

Một Content provider dùng để quản lý việc chia sẻ một tập dữ liệu ứng dụng. Dữ liệu có thể được lưu trữ trong hệ thống file, trong một cơ sở dữ liệu SQLite, trên web hay lưu trữ bằng bất cứ cách nào khác. Thông qua Content provider, một ứng dụng có thể truy cập và chỉnh sửa dữ liệu của ứng dụng khác.

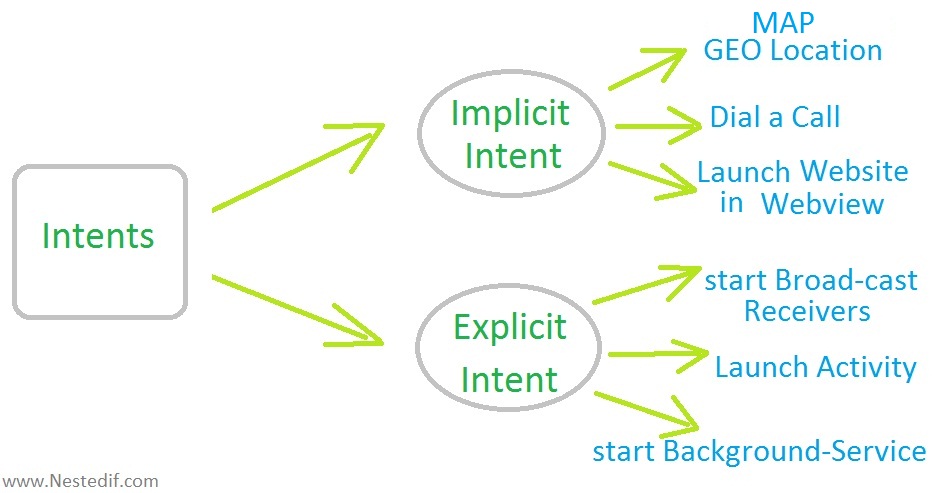
### 2.5.4 Intent

Intent là cầu nối giữa các Activity, mỗi ứng dụng Android thường bao gồm nhiều Activity, mỗi Activity hoạt động độc lập với nhau và thực hiện những công việc khác nhau. Chúng ta có thể dùng Intent để gọi hoặc truyền thông điệp giữa các Activity với nhau, hay giữa Activity với Service.

Có hai loại Intent:

Explicit Intents: là Intent đã xác định rõ thuộc tính component, nghĩa là đã chỉ rõ thành phần sẽ nhận và xử lý Intent. Explicit Intents thường được dùng để khời chạy giữa các Activity trong cùng một ứng dụng.

Implicit Intents: là Intent không chỉ rõ component xử lý, thay vào đó nó bổ sung thông tin trong các thuộc tính. Khi Intent được gửi đi, hệ thống sẽ dựa vào những thông tin này để quyết định component nào thích hợp nhất để xử lý.



**Hình 4 Các loại Intent trong Android**

### 2.5.5 Broadcast Receiver

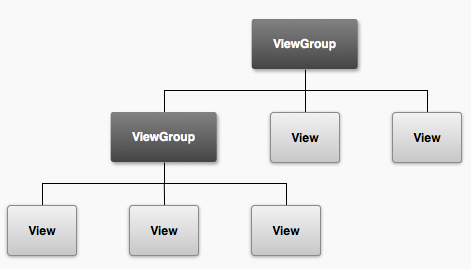
Broadcast Receiver có chức nang dùng để nhận các sự kiện mà hệ thống hoặc ứng dụng phát đi. Ví dụ như báo pin yếu, wifi đã được mở, có tin nhắn đến, cần cập nhật phiên bản ứng dụng…

Broadcast Receiver không có giao diện. Tuy nhiên, nó có thể tạo các thông báo trên Notifications, bật một Activity hoặc một Service.

### 2.5.6 View

Trong một ứng dụng Android, giao diện người dùng được xây dựng từ các đối tượng View và ViewGroup. Có nhiều kểu View và ViewGroup. Mỗi một kiểu là một class con của class View và tất cả các kiểu đó gọi là các Widget.

Một giao diện người dùng là một tập hợp các Layout và các Widget được sắp xếp có thứ tự. Để thể hiện một giao diện người dùng thì trong hàm onCreat() của mỗi Activity cần phải được gọi một hàm là setContentView(R.layout.main), hàm này sẽ load giao diện từ file XML lên để phân tích thành mã bytecode.



**Hình 5 Sơ đồ phân cấp View trong Android**

### 2.5.7 Notification

Notification là thành phần thông báo cho người dùng các thay đổi từ hệ thống hoặc từ ứng dụng mà không gián đoạn các Activity hiện thời. Ví dụ khi nhận được một tin nhắn hay cuộc gọi đến thì sẽ nhấp nháy đèn flash hoặc âm thanh…

# Chương 3 JSON và XML

## 3.1 JSON

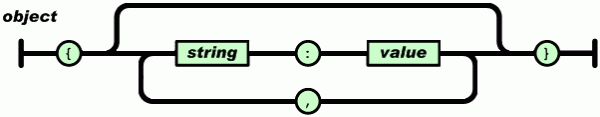
JSON (JavaScript Object Noation) là một chuẩn đề định dạng dữ liệu. Rất dễ để đọc, viết và phân tích. JSON dựa trên cở sở tập hợp của ngôn ngữ JavaScrip, tiêu chuẩn ECMA – 262 phiên bản 3 tháng 12 năm 1999.Cấu trúc JSON là một định dạng văn bản đơn giản với các trường dữ liệu lồng vào nhau.JSON được sử dụng để trao đổi dữ liệu giữa các thành phần của hệ thống tương thích với hầu hết các ngôn ngữ C, C++, C#, Java, JavaScript, Python.... Khi sử dụng JSON không cần phải có các bước phân tích phức tạp như đối với XML, mà có thể truy vấn trực tiếp giá trị theo tên (khóa) được định nghĩa trong JSON. Đứng về mặt tốc độ của các ứng dụng di động thì đây là một ưu điểm rất lớn.

JSON được xây dựng dựa trên hai cấu trúc chính :

* Là tập hợp các cặp giá trị name/value
* Là tập hợp danh sách các giá trị, có thể là array, vector hay list.

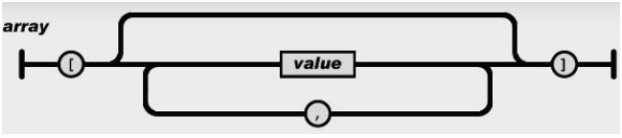
Và tùy thuộc vào dữ liệu cần trao đổi, JSON có thể có nhiều dạng khác nhau, tuy nhiên có thể tổng hợp thành hai dạng chính sau :

* Một đối tượng Object chứa các cặp giá trị string/value không cần thứ tự, được bao trong cặp  ‘{}’, các giá trị bên trong được định dạng ‘string :value’ và chia các nhau bởi dấu ‘,’. Value ở đây có thể là chuỗi, số, true – false, null…



**Hình 6 JSON dạng Object**

* Một đối tượng mảng bao gồm nhiều phần tử con có thứ tự. Các phần tử con được bao trong cặp ‘[]’ và chia cách nhau bởi dâu ‘,’. Mỗi phần tử con có thể là một giá trị đơn lẻ như : chuỗi, số, true – false, null… hoặc một Object khác, thậm chí có thể là một mảng.



**Hình 7 JSON dạng Array**

Các đặc điểm của JSON:

JSON là văn bản trơn (không có định dạng màu sắc, cỡ chữ…).

JSON là tự mô tả (người dùng có thể hiểu được).

JSON có cấu trúc cây.

JSON có thể được phân tích cú pháp (parse).

Dữ liệu JSON có thể truyền đi bằng Ajax.

Không có thẻ kết thúc.

Ngắn hơn, nhanh hơn để đọc và ghi.

## 3.2 XML

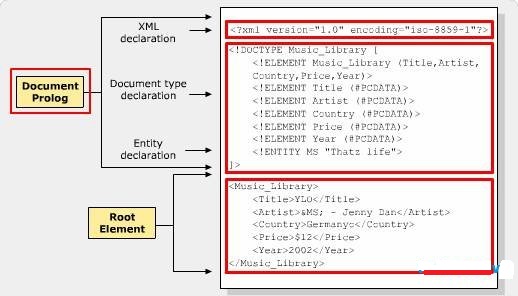
XML (eXtensible Markup Language) là ngôn ngữ đánh dấu do W3C đề nghị. Mục đích chính của XML là đơn giản hóa việc chia sẻ dữ liệu giữa các hệ thống khác nhau, đặc biệt là các hệ thông được kết nối với Internet. XML được tạo ra nhằm khắc phục những hạn chế của HTML – ngôn ngữ đánh dấu siêu văn bản, là cơ sở của mọi trang Web. Giống như HTML, XML cũng được dựa trên SGML (Standard Generalized Markup Language).

Cấu trúc của tài liệu XML:

Gồm có hai phần:

**Document Prolog**: lưu trữ metadata của XML gồm hai phần đó là khai báo XML và khai báo kiểu dữ liệu trong XML. Phần khai báo XML (XML declaration) bao gồm các thông tin về version của XML, charset, endcoding… Phần khai báo kiểu dữ liệu trong XML (DTD) dùng để khai báo cấu trúc của các thẻ dùng trong XML.

**Root Element** hay còn gọi là **Document Element**: chưa tất cả các phần tử và nội dung của nó. Một phần tử của XML phải có thẻ mở và thẻ đóng.



**Hình 8 Cấu trúc của tài liệu XML**

Các đặc điểm của XML:

Dễ dàng xử lý, chuyển tải và trao đổi dữ liệu.

Mô tả dữ liệu và cách thể hiện thông qua các thẻ.

Tổ chức dữ liệu và cấu trúc phân cấp.

Dữ liệu độc lập là ưu điểm chính của XML. Do XML chỉ dùng để mô tả dữ liệu dạng text nên tất cả các chương trình đều có thể đọc được.

Dễ dàng để tạo một file XML.

Lưu trữ cấu hình cho Website.

# Chương 4 Hệ quản trị cơ sở dữ liệu SQLite

## 4.1 Giới thiệu SQLite

SQLite là một hệ quản trị cơ sở dữ liệu quan hệ nhúng. Chứa trong nó là một thư viện nhỏ bằng ngôn ngữ C (khoảng 250kB) và SQLite là một hệ quản trị cơ sở dữ liệu mã nguồn mở.

SQLite không cần cài đặt, không cần cấu hình, không cần server, những điểm này rất khác so với việc sử dụng SQL Server hoặc Oracle, nhưng nó vẫn có transaction để đảm bảo tính toàn vẹn và an toàn trong quá trình thao tác dữ liệu.

SQLite được đánh giá là nhanh, tin cậy và đặc biệt là làm cho việc học và sử dụng nó trở nên đơn giản nhất.

SQLite là một lựa chọn phổ biến để lưu trữ dữ liệu cục bộ. Có thể kết hợp với nhiều ngôn ngữ lập trình khác nhau. Nó là công cụ lưu trữ cơ sở dữ liệu được sử dụng rộng rãi nhất trên các thiết bị di động.

## 4.2 Các đặc điểm chính của SQLite

Đảm bảo đầy đủ bốn đặc tính ACID là tính nguyên tố (Atomic), tính nhất quán (Consisten), tính độc lập (Isolated) và tính bền vững (Durable) ngay cả khi hệ thống bị treo hoặc mất điện.

Không cần cấu hình hay quản trị phức tạp.

SQLite có gần như toàn bộ các đặc tính phổ biến của SQL theo chuẩn SQL92.

Toàn bộ dữ liệu được lưu trữ trong một file duy nhất không phụ thuộc vào nền tảng.

Hỗ trợ cơ sở dữ liệu lên đến hàng TetraByte trong đó kiểu chuỗi và kiểu Blob lên đến hàng GigaByte.

Bộ thư viện quản lý rất nhỏ gọn.

Nhanh hơn các hệ quản trị cơ sở dữ liệu dạng client/server.

Các bộ API đơn giản và dễ sử dụng.

Mã nguồn mở được viết bằng ANSI-C có thể tái sử dụng với bất cứ mục đích gì.

SQLite có thể được tải về và nhúng vào nhiều dự án khác nhau.

Tổ chức lưu trữ độc lập, không phụ thuộc vào các thư viện bên ngoài. Đây là một đặc điểm khá quan trọng để SQLite trở thành cơ sở dữ liệu phù hợp để nhúng vào các thiết bị di động hoặc tích hợp vào các ứng dụng muốn chạy mà không cần phải điều chỉnh cấu hình hệ thống.

## 4.3 Những trường hợp thích hợp sử dụng SQLite

Ứng dụng sử dụng dạng flat file để lưu trữ dữ liệu như từ điển, các ứng dụng nhỏ, lưu trữ cấu hình ứng dụng.

Sử dụng trong các thiết bị nhúng như smart phone, PDA (Personal Digital Assistant) bởi vì SQLite nhỏ, sử dụng bộ nhớ, không gian lưu trữ hiệu quả, tin cậy mà không đòi hỏi phải quản trị.

SQLite có thể sử dụng làm cơ sở dữ liệu cho các website vừa và nhỏ.

Thay thế cho một cơ sở dữ liệu doanh nghiệp đang trong quá trình demo hoặc kiểm tra.

Bởi vì tính đơn giản dễ cài đặt và sử dụng, nên SQLite là một cơ sở dữ liệu tốt để sử dụng giảng dạy cho người mới làm quen với ngôn ngữ truy vấn SQL.

## 4.4 SQLite trong Android

Nếu ứng dụng tạo ra một cơ sở dữ liệu, nó sẽ mặc định được lưu trong thư mục DATA/data/app\_name/databases/file\_name.

Trong đó:

DATA: là đường dẫn có thể lấy được bằng câu lệnh Enviroment.getDataDirectory().

App\_name: là tên ứng dụng.

File\_name : là tên cơ sở dữ liệu.

Để sử dụng SQLite cần import gói android.database trong ứng dụng, gói này chứa tất cả các phương thức cần thiết để làm việc với SQLite. Để làm việc với cơ sở dữ liệu SQLite trong Android, chúng ta sử dụng hai lớp chính là SQLiteOpenHelper và SQLiteDatabase.

SQLiteOpenHelper cung cấp các phương thức getReadableDatabase() và getWriteableDatabase() để truy cập đến các đối tượng của lớp SQLiteDatabase để đọc và ghi dữ liệu.

SQLiteDatabase là lớp cung cấp các phương thức thao tác với cơ sở dữ liệu.

# Chương 5 Google Maps và Google Maps API Web Services

## 5.1 Google Maps

### 5.1.1 Giới thiệu về Google Maps

Google Maps (thời gian trước gọi là Google Local) là một dịch vụ ứng dụng và công nghệ bản đồ trực tuyến trên web miễn phí được cung cấp bởi Google và hỗ trợ nhiều dịch vụ dựa vào bản đồ như Google Ride Finder và một số có thể dùng để nhúng vào các trang web của bên thứ ba thông qua Google Maps API. Nó cho phép thấy bản đồ đường xá, đường đi cho xe đạp, cho người đi bộ và xe hơi, và những địa điểm kinh doan trong khu vực cũng như khắp nơi trên thế giới.



**Hình 9 Google Maps**

Google Maps là một trong những ứng dụng phổ biến nhất của Google hiện nay. Khi mà việc sử dụng bản đồ giấy đx trở nên lỗi thời thì dịch vụ tìm kiếm bản đồ sơ, bản đồ vệ tinh, hệ thống GPS là hết sức cần thiết.

### 5.1.2 Google Maps for Mobile

Năm 2006, Google đã giới thiệu một ứng dụng Java có tên là Google Maps for Mobile. Mục đích để chạy trên bất kỳ thiết bị di động nào có cung cấp nền tảng Java. Rất nhiều tính năng chính của phiên bản Google Maps trên web cũng có mặt trong ứng dụng.

Tháng 11 năm 2007, Google Maps for Mobile 2.0 được phát hành. Nó cung cấp một dịch vụ xác định vị trí giống như GPS nhưng vẫn hoạt động nếu thành phần GPS trong phần cứng không có hoạt động. Tính năng định vị không cần thành phần GPS dựa vào phần mềm xác định trạm phát sóng không dây và trạm BTS (Base Transceiver Station - trạm thu phát sóng di động) gần nhất. Sau đó phần mềm sẽ tìm trong cơ sở dữ liệu của nó vị trí của các điểm phát song Wifi và các trạm BTS mà nó biết được. Bằng cách xác định cường độ tín hiệu từ nhiều trạm phát sóng Wifi và BTS đã biết vị trí, phần mềm sẽ tính toán ra vị trí hiện tại của người dùng.

Thứ tự ưu tiên tính toán vị trí sẽ là:

Thiêt bị bắt vị trí GPS.

Trạm Wifi.

Trạm BTS.

## 5.2 Google Maps API Web Service

### 5.2.1 Giới thiệu về Google Maps API Web Service

Google Maps API Web Service là một tập các giao diện HTTP cung cấp thong tin địa lý cho ứng dụng.

Google Maps API Web Service bao gồm:

Directions API.

Distance Matrix API.

Elevation API.

Geocoding API.

Places API.

### 5.2.2 Sử dụng Google Maps API Web Service

Google Maps API cung cấp các dịch vụ như là các giao diện phục vụ cho việc yêu cầu dữ liệu địa lý và sử dụng dữ liệu đó trong ứng dụng của nhà phát triển. Các dịch vụ này được thiết kế để sử dụng với một ứng dụng bản đồ.

Các dịch vụ này sử dụng HTTP request, thiết lập các chuỗi URL request để gửi yêu cầu cho web service. Thông thường, web service sẽ trả về kết quả là tập tin dạng JSON hoặc XML, nhà phát triển phân tích cú pháp kết quả trả về để sử dụng.

URL request có dạng:

[*http://maps.googleapis.com/maps/api/service/ouput?parameters*](http://maps.googleapis.com/maps/api/service/ouput?parameters)

service: loại dịch vụ.

ouput: kết quả trả về ở dạng tập tin JSON hay XML.

parameter: các tham số phù hợp.

Có thể truy cập Google Maps API Web Service thông qua HTTPS, nó được khuyến khích sử dụng nếu ứng dụng có chứa những dữ liệu nhạy cảm như tọa độ người dùng.

Sử dụng Google Maps API yêu phải chỉ ra rằng ứng dụng có sử dụng một cảm biến (ví dụ cảm biến GPS) để xác định vị trí người dùng trong bất cứ yêu cầu nào cho service thì phải thiết lập giá trị của tham số sensor=true, nếu không vẫn phải gán giá trị sensor=false.

## 5.3 Google Geocoding API

### 5.3.1 Mã hóa địa lý

Là quá trình chuyển địa chỉ vật lý (ví dụ số: 1, đường Đại Cồ Việt, quận Hai Bà Trưng, Hà Nội) thành tọa độ địa lý (ví dụ: vĩ độ 21.007065, kinh độ 105.843032) để đánh dấu trên bản đồ. Ngoài ra Google Geocoding API còn cho phép giải mã từ tọa độ địa lý sang địa chỉ vật lý.

### 5.3.2 Google Geocoding API request

Định dạng chuỗi URL request:

[*http://maps.googleapis.com/maps/api/geocode/ouput?parameters*](http://maps.googleapis.com/maps/api/geocode/ouput?parameters)

Output có thể là JSON hoặc XML.

Các tham số của chuỗi URL request, mỗi tham số được phân cách bởi ký tự ‘&’:

**address**: (bắt buộc) địa chỉ muốn mã hóa địa lý.

Hoặc

**latlng**: (bắt buộc) giá trị kinh độ - vĩ độ của điểm muốn lấy địa chỉ gần nhất.

**bounds**: (tùy chọn) hình chữ nhật mà kết quả mã hóa địa lý sẽ hiện lên nổi bật.

**region**: (tùy chọn) mã vùng, dùng hệ thống tên miền quốc gia.

**language**: (tùy chọn) thiết lập ngôn ngữ sẽ trả về trong kết quả.

**sensor**: (bắt buộc) cho biết yêu cầu có phải đến từ một thiết bị có cảm biến vị trí hay không.

Chú ý: chỉ có thể dùng một trong hai tham số address hoặc latlng để tìm.

### 5.3.3 Kết quả trả về dạng JSON

Google API sẽ trả về một trong hai loại tập tin là JSON hoặc XML.

Trong ví dụ dưới đây, kết quả trả về của mã hóa địa lý địa chỉ số 1, đường Đại Cồ Việt, quận Hai Bà Trưng, Hà Nội dưới dạng tập tin JSON.

Chuỗi URL request:

[http://maps.googleapis.com/maps/api/geocode/json?address=sô 1 đại cồ việt hai bà trưng hà nội việt nam &sensor=false](http://maps.googleapis.com/maps/api/geocode/json?address=sô%201%20đại%20cồ%20việt%20hai%20bà%20trưng%20hà%20nội%20việt%20nam%20&sensor=false)

Kết quả trả về định dạng JSON:

{

"results" : [

{

"address\_components" : [

{

"long\_name" : "1",

"short\_name" : "1",

"types" : [ "street\_number" ]

},

{

"long\_name" : "Đại Cồ Việt",

"short\_name" : "Đại Cồ Việt",

"types" : [ "route" ]

},

{

"long\_name" : "Lê Đại Hành",

"short\_name" : "Lê Đại Hành",

"types" : [ "political", "sublocality", "sublocality\_level\_1" ]

},

{

"long\_name" : "Hai Bà Trưng",

"short\_name" : "Hai Bà Trưng",

"types" : [ "administrative\_area\_level\_2", "political" ]

},

{

"long\_name" : "Hà Nội",

"short\_name" : "Hà Nội",

"types" : [ "administrative\_area\_level\_1", "political" ]

},

{

"long\_name" : "Việt Nam",

"short\_name" : "VN",

"types" : [ "country", "political" ]

}

],

"formatted\_address" : "1 Đại Cồ Việt, Lê Đại Hành, Hai Bà Trưng, Hà Nội, Việt Nam",

"geometry" : {

"location" : {

"lat" : 21.0070854,

"lng" : 105.8428817

},

"location\_type" : "ROOFTOP",

"viewport" : {

"northeast" : {

"lat" : 21.0084343802915,

"lng" : 105.8442306802915

},

"southwest" : {

"lat" : 21.0057364197085,

"lng" : 105.8415327197085

}

}

},

"place\_id" : "ChIJ9d6owIqrNTERRSTQrdWEZKc",

"types" : [ "street\_address" ]

}

],

"status" : "OK"

}

Phân tích kết quả trả về:

Cấu trúc tập tin trả về có hai phần từ chính:

**results**: chứa một tập thông tin kết quả.

**status**: thông báo trạng thái kết quả.

Trong đó **status** thường có các giá trị:

**OK**: kết quả trả về bình thường.

**ZERO\_RESULTS**: địa chỉ không tồn tại hoặc giá trị kinh độ, vĩ độ không đúng.

**OVER\_QUERY\_LIMIT**: thông báo đã gửi quá số lượng yêu cầu trong một khoảng thời gian.

**REQUEST\_DENIED**: ứng dụng đã bị dịch vụ Google Geocoding API chặn.

**INVALID\_REQUEST**: thông báo chuỗi URL request không hợp lệ.

**Results** bao gồm:

**address\_components[]:** chứa một mảng các thành phần địa lý. Ví dụ Hà Nội là một thành phần, Việt Nam là một thành phần. Mỗi phần tử của mảng address\_component chứa các thuộc tính:

**long\_name:** tên đầy đủ của thành phần địa lý.

**short\_name:** tên viết tắt của thành phần địa lý.

**types[]:** kiểu của thành phần địa lý. Chứa một hay nhiều kiểu của địa danh, ví dụ khi mã hóa địa lý Hà Nội sẽ trả về hai giá trị là administrative\_area\_level\_1 và political.

**formatted\_address:** địa chỉ bằng chữ.

**geometry:** chứa thông tin hình học của địa điểm mã hóa địa lý. Trong geometry bao gồm các thuộc tính sau:

**location**: chứa kết quả vĩ độ, kinh độ của địa điểm.

**location\_type**: chứa thông tin bổ sung về địa đểm.

**viewport:** chưa thông tin về khung nhìn của kết quả. Gồm có tọa độ điểm Tây-Bắc và Đông-Nam. Viewport dùng để hiển thị trên bản đồ cho người dùng.

**bound:** (có hoặc không) chứa thông tin một hình chữ nhật bao toàn bộ kết quả trả về, có thể khác viewport.

Các giá trị của types[] thường dùng là:

**street\_address:** địa chỉ đường.

**street\_number:** số nhà.

**route:** địa chỉ tuyến đường.

**intersection:** ngã tư lớn.

**political:** địa chỉ hành chính.

**country:** quốc gia.

**administrative\_area\_level\_1:** địa chỉ hành chính cấp 1, dưới quốc gia, ở Việt Nam là tỉnh hoặc thành phố trực thuộc trung ương.

**administrative\_area\_level\_1:** địa chỉ hành chính cấp 2, ở Việt Nam là quận, huyện.

**locality:** thành phố hoặc thị xã.

**airport:** sân bay.

**park:** công viên.

## 5.4 Google Direction API

Google Direction API là dịch vụ tìm kiếm đường đi giữa hai điểm sử dụng HTTP request. Thông tin đầu vào của điểm đầu, điểm kết thúc có thể ở dạng địa chỉ vật lý hoặc tọa độ địa lý. Google Direction API có thể trả về nhiều đường đi khác nhau.

### 5.4.1 Google Direction API request

Định dạng của chuỗi URL request:

[*http://maps.googleapis.com/maps/api/directions/output?parameters*](http://maps.googleapis.com/maps/api/directions/output?parameters)

**Output** có thể là JSON hoặc XML.

Các tham số của chuỗi URL request, mỗi tham số được phân cách bởi ký tự ‘&’:

**origin**: (bắt buộc) địa chỉ vật lý hay giá trị kinh độ - vĩ độ của điểm bắt đầu tìm đường đi.

**destination**: (bắt buộc) địa chỉ vật lý hay giá trị kinh độ - vĩ độ của điểm kết thúc tìm đường đi.

**mode**: (bắt buộc) xác định kiểu di chuyển (đi bộ, xe đạp, oto).

**driving**: (mặc định) lái xe hơi.

**walking**: đi bộ.

**bicycling**: đi xe đạp.

**waypoints**: (tùy chọn) xác định một tập hợp các địa điểm trung gian, các địa điểm này có thể là địa chỉ vật lý hoặc tọa độ địa lý.

**alternatives**: (tùy chọn) nếu bằng true thì Google Directions sẽ trả về nhiều hơn một đường đi để lựa chọn.

**avoid**: (tùy chọn) chỉ ra tuyến đường sẽ phải tránh đường cao tốc hoặc tránh trạm thu phí.

**tolls**: tránh trạm thu phí.

**highways**: tránh đường cao tốc.

**units**: (tùy chọn) xác định đơn vị cho kết quả.

**region**: (tùy chọn) mã vùng.

**sensor**: (bắt buộc) cho biết yêu cầu có phải đến từ một thiết bị có cảm biến vị trí hay không.

Khi tìm kiếm đường đi sử dụng Google Direction API sử dụng nhiều điểm trung gian thì mỗi điểm trung gian cách nhau bằng ký tự ‘|’.Mặc định sẽ tìm kiếm đường đi qua các điểm trung gian dựa trên sự sắp xếp địa điểm trên URL request.

### 5.4.2 Kết quả trả về dạng JSON

Google API sẽ trả về một trong hai loại tập tin là JSON hoặc XML.

Mặc định chỉ có một giá trị route trong kết quả trả về, trừ khi muốn kết quả trả về hơn hai tuyến đường.

Trong ví dụ dưới đây, kết quả trả về của tìm kiếm đường đi từ số 1, đường Đại Cồ Việt, quận Hai Bà Trưng, Hà Nội đến hồ Hoàn Kiếm, quận Hoàn Kiếm, Hà Nội dưới dạng tập tin JSON.

Chuỗi URL request:

*[http://maps.googleapis.com/maps/api/directions/json?origin=số 1 đại cồ việt hai bà trưng hà nội việt nam&destination=hồ hoàn kiếm quận hoàn kiếm hà nội việt nam&sensor=false&units=metric&mode=walking](Cốc%20Cốc.lnk)*

Kết quả trả về định dạng JSON:

{

"geocoded\_waypoints" : [

{

"geocoder\_status" : "OK",

"place\_id" : "ChIJ9d6owIqrNTERRSTQrdWEZKc",

"types" : [ "street\_address" ]

},

{

"geocoder\_status" : "OK",

"place\_id" : "ChIJlclXM5WrNTERDqL5tGu\_ugE",

"types" : [ "establishment", "natural\_feature" ]

}

],

"routes" : [

{

"bounds" : {

"northeast" : {

"lat" : 21.0317032,

"lng" : 105.8538372

},

"southwest" : {

"lat" : 21.0072596,

"lng" : 105.8426445

}

},

"copyrights" : "Dữ liệu bản đồ ©2017 Google",

"legs" : [

{

"distance" : {

"text" : "4,1 km",

"value" : 4106

},

"duration" : {

"text" : "51 phút",

"value" : 3053

},

"end\_address" : "Hồ Hoàn Kiếm, Hàng Trống, Hoàn Kiếm, Hà Nội, Việt Nam",

"end\_location" : {

"lat" : 21.0286521,

"lng" : 105.8514472

},

"start\_address" : "1 Đại Cồ Việt, Lê Đại Hành, Hai Bà Trưng, Hà Nội, Việt Nam",

"start\_location" : {

"lat" : 21.0072596,

"lng" : 105.8428845

},

"steps" : [

{

"distance" : {

"text" : "14 m",

"value" : 14

},

"duration" : {

"text" : "1 phút",

"value" : 10

},

"end\_location" : {

"lat" : 21.0072618,

"lng" : 105.842746

},

"html\_instructions" : "Đi về hướng \u003cb\u003eTây\u003c/b\u003e",

"polyline" : {

"points" : "k~e\_C\_m\_eS?X"

},

"start\_location" : {

"lat" : 21.0072596,

"lng" : 105.8428845

},

"travel\_mode" : "WALKING"

},

{

"distance" : {

"text" : "32 m",

"value" : 32

},

"duration" : {

"text" : "1 phút",

"value" : 23

},

"end\_location" : {

"lat" : 21.0075319,

"lng" : 105.8426445

},

"html\_instructions" : "Rẽ \u003cb\u003ephải\u003c/b\u003e về hướng \u003cb\u003eĐại Cồ Việt\u003c/b\u003e",

"maneuver" : "turn-right",

"polyline" : {

"points" : "k~e\_Cel\_eSIDKFE@G@G@I@"

},

"start\_location" : {

"lat" : 21.0072618,

"lng" : 105.842746

},

"travel\_mode" : "WALKING"

},

{

"distance" : {

"text" : "0,6 km",

"value" : 644

},

"duration" : {

"text" : "8 phút",

"value" : 479

},

"end\_location" : {

"lat" : 21.0083929,

"lng" : 105.8487696

},

"html\_instructions" : "Rẽ \u003cb\u003ephải\u003c/b\u003e vào \u003cb\u003eĐại Cồ Việt\u003c/b\u003e\u003cdiv style=\"font-size:0.9em\"\u003eBăng qua Nhà C1 (ở bên phải cách 43&nbsp;m)\u003c/div\u003e",

"maneuver" : "turn-right",

"polyline" : {

"points" : "a`f\_Cok\_eSGgBMcCGy@KoAIe@KaCOmCSeCYkCKgAQeBEe@Iw@"

},

"start\_location" : {

"lat" : 21.0075319,

"lng" : 105.8426445

},

"travel\_mode" : "WALKING"

},

{

"distance" : {

"text" : "0,3 km",

"value" : 276

},

"duration" : {

"text" : "3 phút",

"value" : 201

},

"end\_location" : {

"lat" : 21.0086008,

"lng" : 105.8514167

},

"html\_instructions" : "Chếch sang \u003cb\u003ephải\u003c/b\u003e tại Công Ty Cp Sx Kinh Doanh Lâm Hồng Đạt để vào \u003cb\u003eĐại Cồ Việt\u003c/b\u003e\u003cdiv style=\"font-size:0.9em\"\u003eBăng qua Trường Học (ở bên phải cách 85&nbsp;m)\u003c/div\u003e",

"maneuver" : "turn-slight-right",

"polyline" : {

"points" : "mef\_Cyq`eSGq@WgEEg@AWCcAAi@?g@?MBs@"

},

"start\_location" : {

"lat" : 21.0083929,

"lng" : 105.8487696

},

"travel\_mode" : "WALKING"

},

{

"distance" : {

"text" : "1,3 km",

"value" : 1252

},

"duration" : {

"text" : "16 phút",

"value" : 946

},

"end\_location" : {

"lat" : 21.0197878,

"lng" : 105.8519505

},

"html\_instructions" : "Rẽ \u003cb\u003etrái\u003c/b\u003e tại Cửa Hàng Tạp Hóa Số 2 vào \u003cb\u003eBạch Mai\u003c/b\u003e/\u003cb\u003eHuế\u003c/b\u003e\u003cdiv style=\"font-size:0.9em\"\u003eĐi tiếp tục theo Huế\u003c/div\u003e\u003cdiv style=\"font-size:0.9em\"\u003eBăng qua cầu (ở bên phải)\u003c/div\u003e",

"maneuver" : "turn-left",

"polyline" : {

"points" : "wff\_CkbaeSg@?cAGuCKyBIyFOcGScA?g@BcFPsAHkI^}AAaAEs@IoAQoC\_@\_Dc@"

},

"start\_location" : {

"lat" : 21.0086008,

"lng" : 105.8514167

},

"travel\_mode" : "WALKING"

},

{

"distance" : {

"text" : "0,6 km",

"value" : 641

},

"duration" : {

"text" : "8 phút",

"value" : 481

},

"end\_location" : {

"lat" : 21.0254261,

"lng" : 105.8532119

},

"html\_instructions" : "Tại Quán Cafe My Cloud 1A, tiếp tục vào \u003cb\u003eHàng Bài\u003c/b\u003e\u003cdiv style=\"font-size:0.9em\"\u003eBăng qua Cửa Hàng Đồ Đồng - Đồ Thờ (ở phía bên phải)\u003c/div\u003e",

"polyline" : {

"points" : "ulh\_CueaeSoFw@oFs@eAOa@G}De@gBQ{Ew@]G"

},

"start\_location" : {

"lat" : 21.0197878,

"lng" : 105.8519505

},

"travel\_mode" : "WALKING"

},

{

"distance" : {

"text" : "0,6 km",

"value" : 601

},

"duration" : {

"text" : "7 phút",

"value" : 442

},

"end\_location" : {

"lat" : 21.0307187,

"lng" : 105.853582

},

"html\_instructions" : "Tại Trung tâm Thương mại Tràng Tiền, tiếp tục vào \u003cb\u003eĐinh Tiên Hoàng\u003c/b\u003e\u003cdiv style=\"font-size:0.9em\"\u003eBăng qua Cục Hóa Chất (ở bên phải cách 44&nbsp;m)\u003c/div\u003e",

"polyline" : {

"points" : "}oi\_CqmaeSaCYkBOkCQsBSwD]eCGeDGMDODQJ\_@Z"

},

"start\_location" : {

"lat" : 21.0254261,

"lng" : 105.8532119

},

"travel\_mode" : "WALKING"

},

{

"distance" : {

"text" : "39 m",

"value" : 39

},

"duration" : {

"text" : "1 phút",

"value" : 27

},

"end\_location" : {

"lat" : 21.0307676,

"lng" : 105.8532365

},

"html\_instructions" : "Rẽ \u003cb\u003etrái \u003c/b\u003e tại Tháp Bút",

"maneuver" : "turn-left",

"polyline" : {

"points" : "\_qj\_C{oaeSDF?F?D?H?FCJKN"

},

"start\_location" : {

"lat" : 21.0307187,

"lng" : 105.853582

},

"travel\_mode" : "WALKING"

},

{

"distance" : {

"text" : "0,6 km",

"value" : 607

},

"duration" : {

"text" : "7 phút",

"value" : 444

},

"end\_location" : {

"lat" : 21.0286521,

"lng" : 105.8514472

},

"html\_instructions" : "Rẽ \u003cb\u003ephải\u003c/b\u003e\u003cdiv style=\"font-size:0.9em\"\u003eBăng qua Cà Phê Bon Bon (ở bên phải cách 400&nbsp;m)\u003c/div\u003e\u003cdiv style=\"font-size:0.9em\"\u003eĐiểm đến sẽ ở bên trái\u003c/div\u003e",

"maneuver" : "turn-right",

"polyline" : {

"points" : "iqj\_CwmaeS]?OCE@I@a@R]RMHEB?@A@A@ADCLId@MpAE`A@t@B\\FNJRRXVNTHTDH?VEREXEZG^G`@EVI\\Cd@IZCl@C`@CZGf@Ar@?@?"

},

"start\_location" : {

"lat" : 21.0307676,

"lng" : 105.8532365

},

"travel\_mode" : "WALKING"

}

],

"traffic\_speed\_entry" : [],

"via\_waypoint" : []

}

],

"overview\_polyline" : {

"points" : "k~e\_C\_m\_eS?XIDQHOBI@GgBU}DKoAIe@[oGm@qGm@kGg@yHEmB?u@Bs@g@?yESsJYcGScA?kGT\_Lh@\_DGcC[oWoDgBWeHw@yF\_AmFi@\_Ge@wD]eCGeDGMDa@P\_@ZDF?L?POZm@COBsAt@EJMr@SrCDrARb@RXVNj@N`@El@KtB\_@bAMhAG|@KzAA@?"

},

"summary" : "Huế",

"warnings" : [

"Chỉ đường đi bộ đang ở giai đoạn thử nghiệm. Sử dụng cảnh báo – Tuyến đường này không có vỉa hè hoặc đường dành cho người đi bộ."

],

"waypoint\_order" : []

}

],

"status" : "OK"

}

Phân tích kết quả trả về:

Cấu trúc tập tin trả về có hai phần từ chính:

**results**: chứa một tập thông tin kết quả.

**status**: thông báo trạng thái kết quả.

Trong đó **status** thường có các giá trị:

**OK**: kết quả trả về bình thường.

**NOT\_FOUND:** một điểm nào đó nhập vào không thể mã hóa địa lý để lấy tọa độ.

**ZERO\_RESULTS**: không thể tìm thấy đường đi.

**MAX\_WAYPOINTS\_EXCEEDED:** thông báo vượt quá giới hạn các điểm trung gian cho phép.

**OVER\_QUERY\_LIMIT**: thông báo đã gửi quá số lượng yêu cầu trong một khoảng thời gian.

**REQUEST\_DENIED**: ứng dụng đã bị dịch vụ Google Direction API chặn.

**INVALID\_REQUEST**: thông báo chuỗi URL request không hợp lệ.

**UNKNOWN\_ERROR:** lỗi không xác định.

**Results** bao gồm:

**routes[]:** là một mảng chưa thông tin về một tuyến đường riêng biệt từ điểm bắt đầu đến điểm kết thúc. Trong kết quả trả về của Google Direction mảng routes sẽ có nhiều giá trị hoặc sẽ không có nếu không thể tìm thấy đường đi.

Trong mỗi mảng routes[] sẽ có những thuộc tính sau:

**summary**: tên tóm tắt của đoạn đường.

**legs[]:** là một mảng chứa mô tả đường đi giữa hai điểm. Nếu không yêu cầu điểm trung gian thì kết quả trả về mảng legs sẽ có một phần tử.

**waypoint\_order[]:** chứa một mảng quy định thứ tự sắp xếp của các điểm trung gian.

**overview\_polyline**: chứa một đối tượng point giữ một mảng các điểm đã mã hóa đại diện cho một tuyến đường trong kết quả.

**bounds**: quy định hình chữ nhật bao quanh tuyến đường trên bản đồ.

**copyrights**: chứa thông tin bản quyền.

**warnings[]:** chứa một tập các cảnh bảo cho người dùng.

Trong mỗi mảng legs sẽ chứa các đối tượng sau:

**steps[]:** mảng chứa thông tin từng đoạn đường.

**distance**: chứa thông tin về độ dài đoạn đường được mô tả trong mảng legs, bao gồm hai thuộc tính:

**value**: độ dài kiểu số nguyên, đơn vị mét.

**text**: mô tả bằng chữ độ dài của đoạn đường, đơn vị phụ thuộc vào vùng địa lý đã thiết lập trong chuỗi URL request hoặc bằng giá trị trong units.

**duration**: ước lượng thời gian di chuyển hết đoạn đường ứng với một đoạn đường được mô tả trong mảng legs, bao gồm hai thuộc tính:

**value**: giá trị số nguyên, đơn vị giây.

**text**: thời gian ước lượng bằng chữ.

**start\_location**: chứa vĩ độ, kinh độ của điểm bắt đầu.

**end\_location:** chứa vĩ độ, kinh độ của điểm kết thúc.

**start\_address:** địa chỉ điểm bắt đầu bằng chữ.

**end\_address:** địa chỉ điểm kết thúc bằng chữ.

Trong mỗi mảng steps mô tả một đoạn đường. Đoạn đường trong mảng steps quy định là đơn vị nhất và không cần phải tách them nữa. Mỗi phần tử trong mảng steps chứa các đối tượng sau:

**html\_instructions**: hướng dẫn cho đoạn đường bằng HTML.

**distance:** chứa thông tin về độ dài đoạn đường được mô tả trong mảng steps, bao gồm hai thuộc tính:

**value:** độ dài kiểu số nguyên, đơn vị mét.

**text:** mô tả bằng chữ độ dài của đoạn đường, đơn vị phụ thuộc vào vùng địa lý đã thiết lập trong chuỗi URL request hoặc bằng giá trị trong units.

**duration**: ước lượng thời gian di chuyển hết đoạn đường ứng với một đoạn đường được mô tả trong mảng steps, bao gồm hai thuộc tính:

**value**: giá trị số nguyên, đơn vị giây.

**text**: thời gian ước lượng bằng chữ.

**start\_location**: chứa vĩ độ, kinh độ của điểm bắt đầu.

**end\_location:** chứa vĩ độ, kinh độ của điểm kết thúc.

# Chương 6 Phân tích thiết kế hệ thống

## 6.1 Yêu cầu của hệ thống

**Yêu cầu chức năng:**

Hiển thị vị trí của người sử dụng..

Tìm kiếm được địa điểm đặt các cây ATM tùy theo yêu cầu của người sử dụng.

Hiển thị được đường đi từ vị trí của người sử dụng đến vị trí của cây ATM bất kỳ.

Lưu lại trạng thái của ứng dụng khi người sử dụng thoát ứng dụng.

**Yêu cầu phi chức năng:**

Chương trình có thể vận hành tốt khi mà cơ sở dữ liệu có thể rấ lớn.

Kết quả hiển thị vị trí, đường đi chính xác với sai số nhỏ, có thể chấp nhận được.

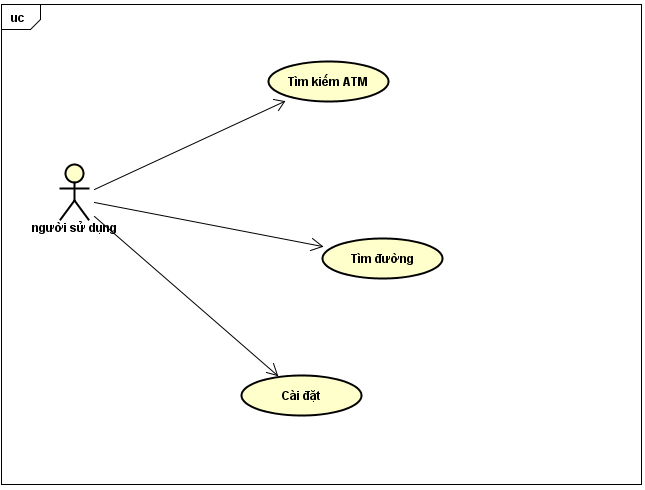
Giao diện chương trình đẹp, đơn giản để những người có ít kinh nghiệm có thể sử dụng dễ dàng.

Sử dụng hệ điều hành Android phiên bản lớn hơn 4.0.

Điện thoại có kết nối với Internet và hỗ trợ GPS.

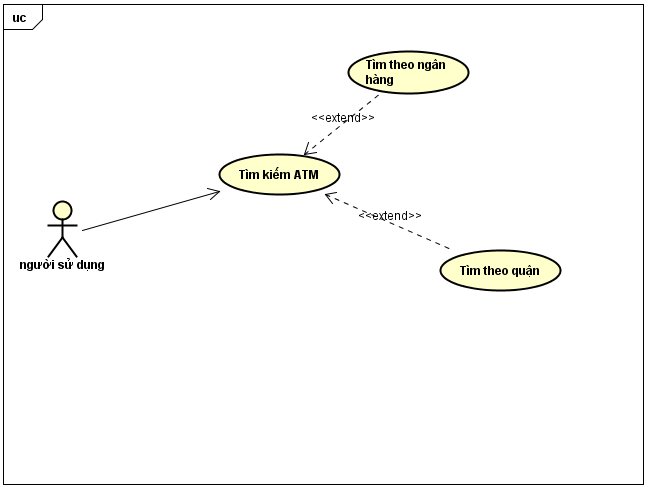
## 6.2 Phân tích thiết kê.

### 6.2.1 Biểu đồ use case.



**Hình 10 Biểu đồ use-case tổng quan**

Phân rã use-case “Tìm kiếm ATM”:

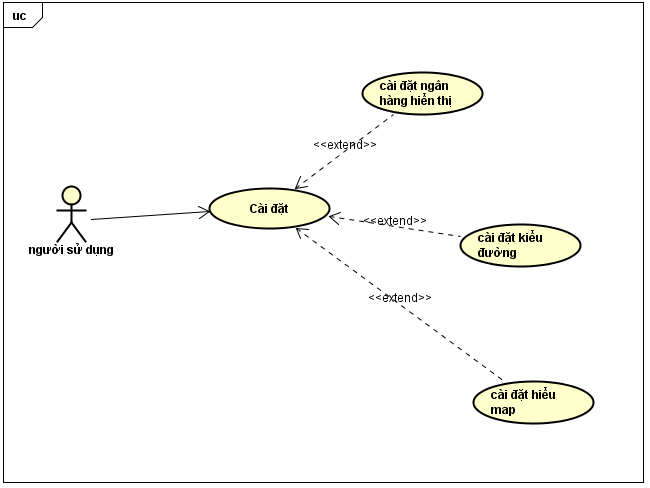


**Hình 11 Biểu đồ use-case cấp 2 (Tìm kiếm ATM)**

Đặc tả use-case “Tìm kiếm ATM theo ngân hàng”:

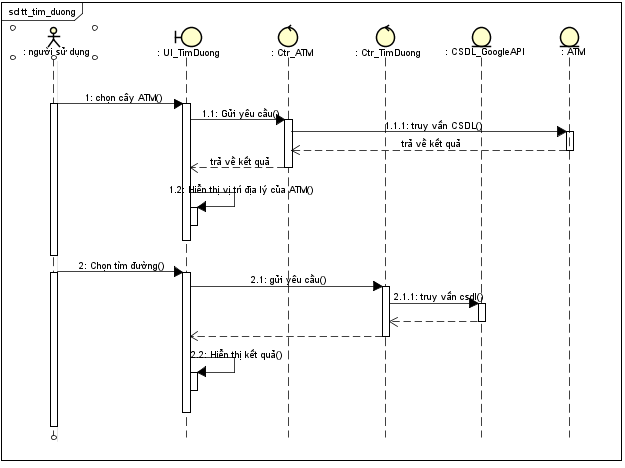
|  |  |
| --- | --- |
| Tên | Tìm kiếm ATM theo ngân hàng |
| Precondition | Chọn ngân hàng muốn tìm kiếm |
| Main flow event (succees) | |  |  |  | | --- | --- | --- | | # | Actor | Action | | 1 | Người sử dụng | Chọn ngân hàng muốn tìm cây ATM. | | 2 | System | Tìm kiếm trong csdl. | | 3 | System | Hiển thị kết quả. | |
| Alternative flow of event | |  |  |  | | --- | --- | --- | | # | Actor | Action | | 1 | System | Hiển thị thông báo lỗi. | |
| Post condition | Không có |

Phân rã use-case “Cài đặt”:

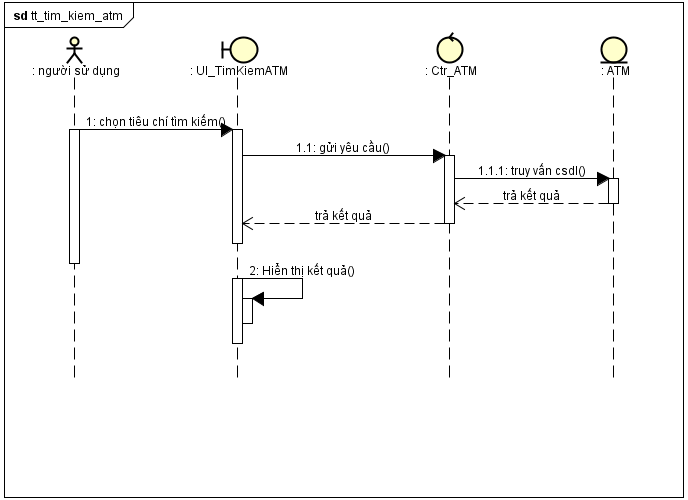


**Hình 12 Biểu đồ use-case cấp 2 (Cài đặt)**

### 6.2.2 Biểu đồ trình tự

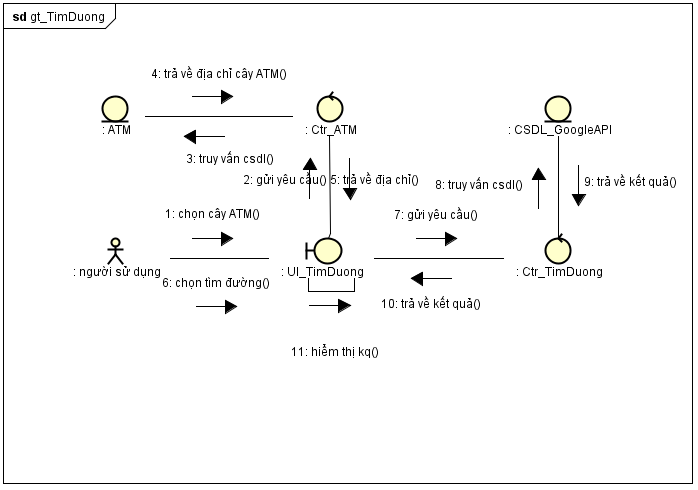


**Hình 13 Biểu đồ trình tự tìm đường đi**

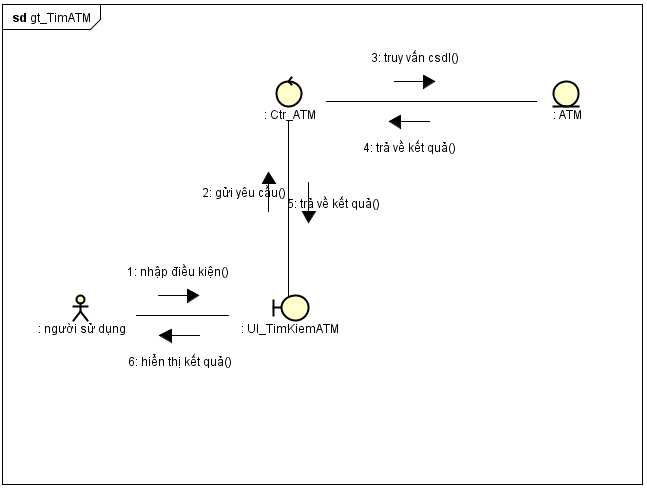


**Hình 14 Biểu đồ trình tự tìm kiếm cây ATM**

### 6.2.3 Biểu đồ giao tiếp

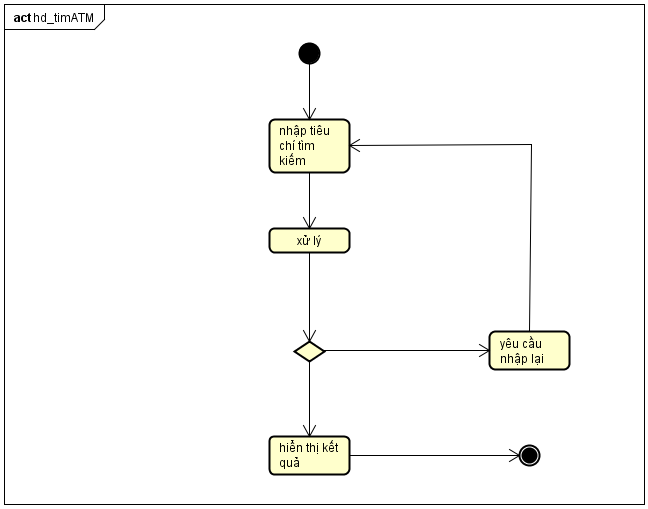


**Hình 15 Biểu đồ giao tiếp tìm đường đi**



**Hình 16 Biểu đồ giao tiếp tìm kiếm cây ATM**

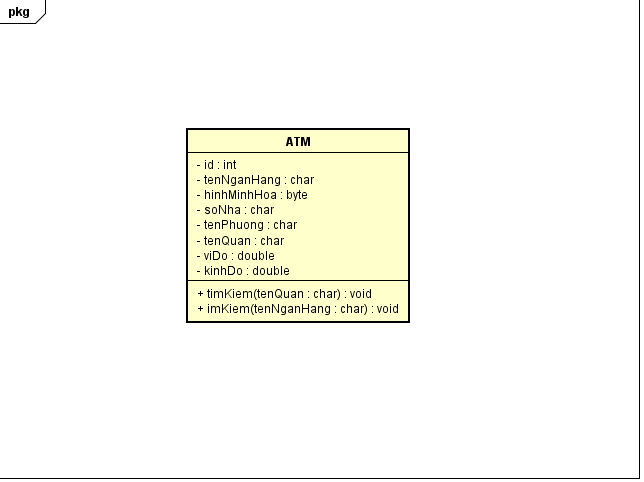
### 6.2.3 Biểu đồ hoạt động



**Hình 17 Biểu đồ hoạt động tìm kiêm cây ATM**

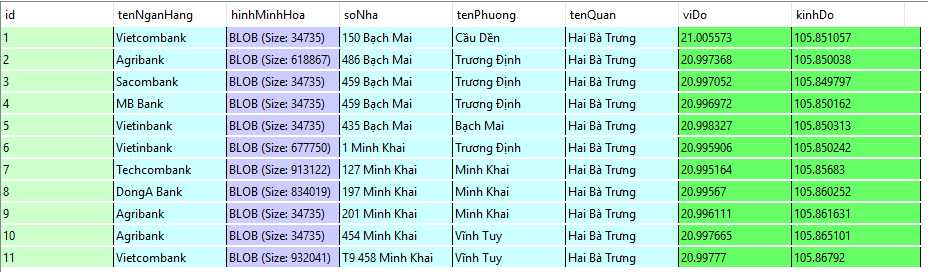
### 6.2.4 Xây dựng cơ sở dữ liệu

Lớp ATM:



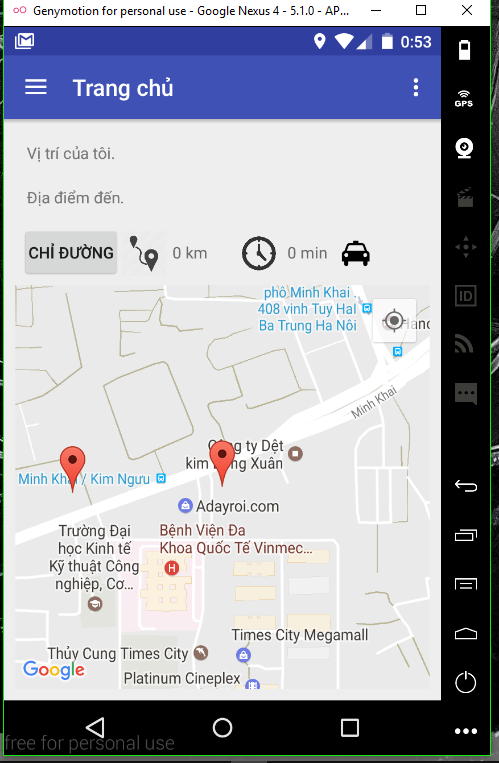
**Hình 18 Lớp ATM**

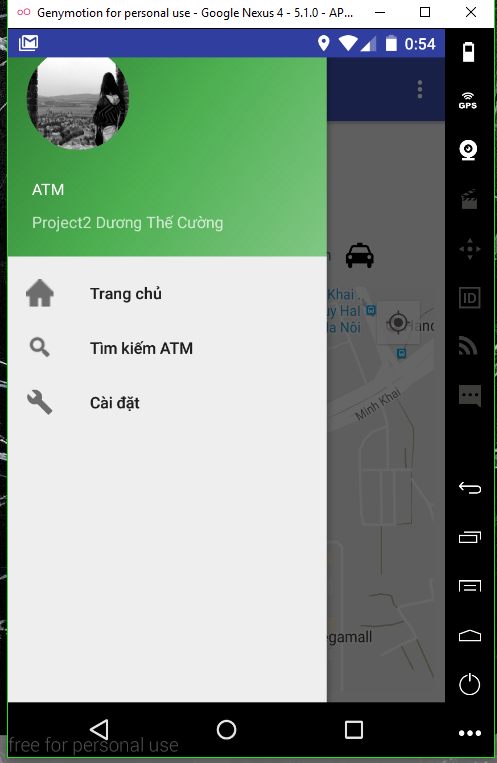
Demo cơ sở dữ liệu:



**Hình 19 Demo cơ sở dữ liệu**

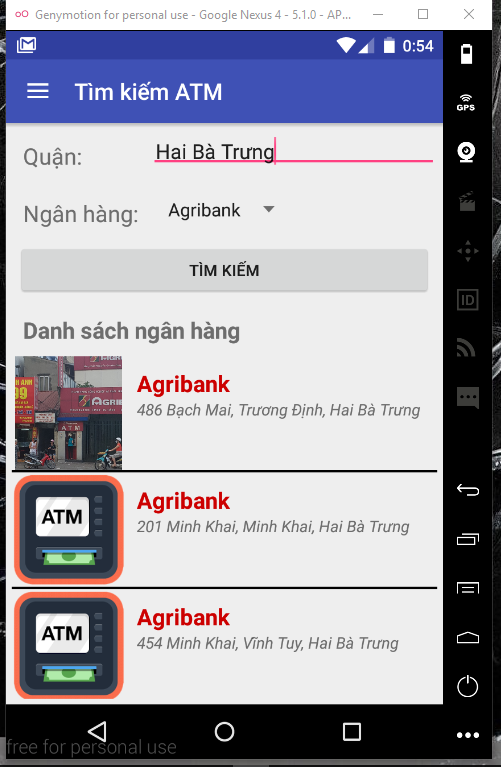
## 6.3 Chương trình minh họa

Giao diện chính của ứng dụng:



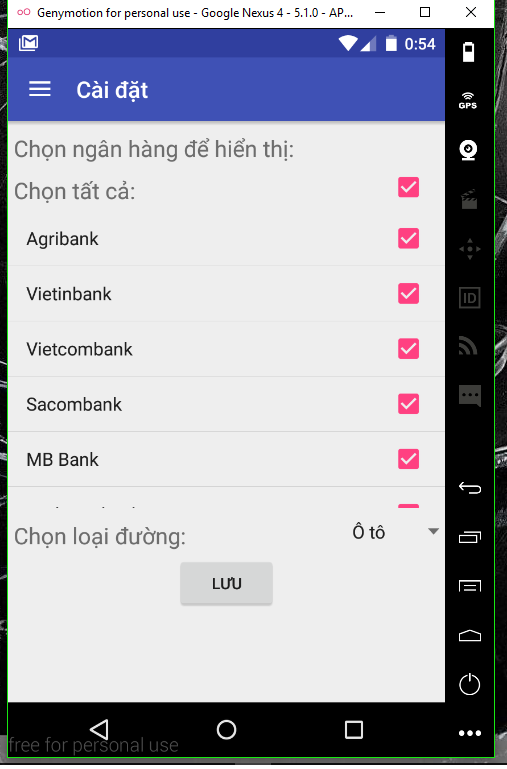
**Hình 20 Giao diện chính của ứng dụng**

Giao diện “Tìm kiếm”:



**Hình 21 Giao diện tìm kiếm**

Giao diện “Cài đặt”:



**Hình 22 Giao diện cài đặt**

# Kết luận

Ưu điểm: Ứng dụng đã giải quyết được vấn đề tìm kiếm ATM hàng ngày, giao diện thân thiện, dễ sử dụng.

Nhược điểm: Việc tìm kiếm ATM còn chưa được nhanh, chưa có cơ chế cập nhật dữ liệu thông qua server, muốn cập nhật dữ liệu phải cập nhật cả phần mềm, chưa cho phép người dùng thêm địa điểm mới.

Hướng phát triển trong tương lai: Giải quyết triệt để những nhược điểm còn tồn tại và phát triển một số chức năng mới như có thể chia sẻ vị trí người dùng với nhau. Thêm cơ chế sao lưu dữ liệu người dùng. Mở rộng giải quyết những nhu cầu thường ngày như tìm kiếm trạm xăng, bến xe bus, nhà hàng…

Do mới làm quen với ngôn ngữ lập trình Android, còn nhiều bỡ ngỡ nên ứng dụng còn nhiều sai sót. Em rất mong được sự chỉ bảo của Thầy cùng sự góp ý của các bạn để ứng dụng của em ngày càng hoàn thiện hơn.

Em xin chân thành cảm ơn Thầy!

# Tài liệu tham khảo

[1] **Google** **INC**. Android Developer Docs. (Online).

[2] **Google** **INC**. Android Maps API Web Services. (Online).

[3] **Wikipedia**. Android (Hệ điều hành). (Online).

[4] **Mark** **L**. **Murphy** The Busy Coder’s Guide to Android Development version 4.1, 8/2012.

[5] **Wei**-**Meng** **Lee**. Beginning Android 4 Application Development, 2012.

[6] **FPT** **Software**. Giáo trình đào tạo Android FPT Software, 2011.