

**BỘ GIÁO DỤC VÀ ĐÀO TẠO  
TRƯỜNG ĐẠI HỌC THĂNG LONG**



# **KHÓA LUẬN TỐT NGHIỆP**

**TÌM HIỂU VÀ PHÁT TRIỂN ỨNG DỤNG  
TRA CỨU THÔNG TIN TÀU - XE  
TRÊN THIẾT BỊ DI ĐỘNG  
SỬ DỤNG HỆ ĐIỀU HÀNH ANDROID**

**GIÁO VIÊN HƯỚNG DẪN : THS. LÊ MINH TUẤN  
SINH VIÊN THỰC HIỆN : NGUYỄN HOÀNG LONG  
MÃ SINH VIÊN : A10805  
CHUYÊN NGÀNH : KHOA HỌC MÁY TÍNH**

**HÀ NỘI – 2013**

**Mục Lục**

**DANH MỤC HÌNH ẢNH – BẢNG BIỂU ..... 1**

**DANH MỤC VIẾT TẮT VÀ THUẬT NGỮ..... 2**

**LỜI MỞ ĐẦU ..... 3**

**CHƯƠNG 1. TỔNG QUAN về Dự ÁN..... 5**

**1.1. Khảo sát thực tế..... 5**

**1.2. Giải pháp ..... 7**

**CHƯƠNG 2. PHÂN TÍCH YÊU CẦU BÀI TOÁN ..... 8**

**2.1. Mục tiêu..... 8**

**2.2. Yêu cầu kỹ thuật..... 9**

**2.3. Yêu cầu nghiệp vụ ..... 9**

**2.4. Ảnh xạ yêu cầu nghiệp vụ và chức năng: ..... 12**

**2.5. Đặc tả hệ thống ..... 13**

        2.5.1. Sơ đồ hoạt động của chức năng Giờ tàu..... 13

        2.5.2. Sơ đồ hoạt động của chức năng Tuyến buýt..... 13

        2.5.3. Sơ đồ hoạt động của chức năng Chuyển tuyến ..... 14

        2.5.4. Sơ đồ hoạt động của chức năng Định vị và Bến xe buýt..... 15

**2.6. Kế hoạch dự án ..... 15**

**2.6. Đặc tả chức năng ..... 16**

        2.6.1. Chức năng Giờ tàu..... 16

        2.6.2. Chức năng Tuyến buýt..... 18

        2.6.3. Chức năng Chuyển tuyến..... 21

        2.6.4. Chức năng Định vị..... 24

        2.6.5. Chức năng Bến xe buýt..... 27

**CHƯƠNG 3. THIẾT KẾ VÀ TRIỂN KHAI..... 31**

**3.1. Hệ điều hành Android..... 31**

        3.1.1. Giới thiệu:..... 31

        3.1.2. Kiến trúc hệ điều hành Android: ..... 32

**3.2. Chu trình sống của một ứng dụng ..... 33**

        3.2.1. Activity là gì?..... 33

3.2.2. Chu trình sống của một activity (Activity lifecycle).....	33
<b>3.3. Kiến trúc mạng.....</b>	<b>35</b>
<b>3.4. Mô hình hóa dữ liệu .....</b>	<b>36</b>
3.4.1. Sơ đồ cơ sở dữ liệu .....	36
3.4.2. Cấu trúc các bảng dữ liệu.....	37
<b>3.5. Triển khai.....</b>	<b>38</b>
3.5.1. Cấu trúc của một project Android.....	39
3.5.2. Cấu trúc của file AndroidManifest.xml .....	39
3.5.3. File R.java .....	41
<b>3.6. Xử lý dữ liệu đầu vào .....</b>	<b>42</b>
3.6.1. Xử lý thông tin giờ tàu.....	42
3.6.2. Xử lý thông tin tuyến buýt.....	44
<b>3.7. Cài đặt các chức năng chính.....</b>	<b>48</b>
3.7.1. Chức năng Chuyển tuyến.....	48
3.7.2. Chức năng Định vị.....	51
<b>3.8. Cấu hình .....</b>	<b>54</b>
<b>CHƯƠNG 4. KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN.....</b>	<b>55</b>
<b>PHỤ LỤC 1 - CÁC THUẬT NGỮ.....</b>	<b>56</b>
<b>PHỤ LỤC 2 – DANH MỤC TÀI LIỆU THAM KHẢO .....</b>	<b>60</b>

**DANH MỤC HÌNH ẢNH – BẢNG BIỂU**

Hình 1. 1. Dữ liệu giờ tàu dạng bảng.....	5
Hình 1. 2. Dữ liệu giờ tàu dạng ảnh.....	6
Hình 1. 3. Dữ liệu tuyến buýt dạng web (liệt kê) .....	6
Hình 1. 4. Dữ liệu tuyến buýt dạng web (dạng bảng).....	7
Hình 2. 1. Sơ đồ USECASE của ứng dụng.....	11
Hình 2. 2. Sơ đồ hoạt động của chức năng Giờ tàu .....	13
Hình 2. 3. Sơ đồ hoạt động của chức năng Tuyến buýt.....	13
Hình 2. 4. Sơ đồ hoạt động của chức năng Chuyển tuyến.....	14
Hình 2. 5. Sơ đồ hoạt động của chức năng Định vị và Bến xe buýt.....	15
Hình 2. 6. Kế hoạch dự án .....	15
Hình 3. 1. Kiến trúc hệ điều hành Android.....	32
Hình 3. 2. Chu trình sống của một Activity.....	34
Hình 3. 3. Kiến trúc mạng.....	36
Hình 3. 4. Sơ đồ cơ sở dữ liệu .....	36
Hình 3. 5. Cấu trúc một project Android .....	39
Hình 3. 6. Minh họa cấu trúc file đầu vào .....	43
Hình 3. 7. Quy trình xử lý thông tin giờ tàu .....	44
Hình 3. 8. Minh họa LookUpTable.....	47
Hình 3. 9. Sơ đồ hành động chức năng chuyển tuyến .....	48
Hình 3. 10. Minh họa mô hình hóa dữ liệu.....	49
Hình 3. 11. Các lớp hỗ trợ thuật toán Dijkstra.....	50
Hình 3. 12. Lấy debug keystore .....	52
Hình 3. 13. Lấy API key .....	52
Bảng 2. 1. Bảng ánh xạ yêu cầu nghiệp vụ và chức năng .....	12
Bảng 3. 1. Lược sử phát triển hệ điều hành Android.....	31
Bảng 3. 2. Bảng dữ liệu tuyến xe buýt.....	37
Bảng 3. 3. Bảng dữ liệu các đoạn của tuyến xe buýt.....	37
Bảng 3. 4. Bảng dữ liệu các bến xe buýt .....	38

**DANH MỤC VIẾT TẮT VÀ THUẬT NGỮ**

<b>Từ viết tắt</b>	<b>Tên tiếng Anh đầy đủ</b>	<b>Tên tiếng Việt đầy đủ</b>
UC	Usercase	Chức năng
API	Application Programming Interface	Giao diện lập trình ứng dụng
CSDL	Database	Cơ sở dữ liệu
HTML	Hyper Text Markup Language	Ngôn ngữ hiển thị siêu văn bản
SSL	Secure Sockets Layer	
	Server	Máy chủ

## **LỜI MỞ ĐẦU**

Trong 20 năm trở lại đây, thu nhập bình quân đầu người của Việt Nam đã tăng gấp 3 lần kéo theo việc gia tăng phương tiện cá nhân và sự phát triển của hệ thống vận tải công cộng. Trong đó vận tải công cộng bao gồm, đường bộ, hàng không và đường thủy là xương sống giúp lưu chuyển người và hàng hóa nhằm duy trì và phát triển kinh tế.

Theo quy hoạch phát triển mạng lưới giao thông của Chính phủ:

+ **Giao thông đường sắt:**

Đến năm 2020: giao thông vận tải đường sắt cần chiếm tỷ trọng tối thiểu 13% về nhu cầu luân chuyển hành khách và 14% về luân chuyển hàng hoá trong tổng khối lượng vận tải của toàn ngành giao thông vận tải; trong đó vận tải hành khách đô thị bằng đường sắt đạt ít nhất là 20% nhu cầu vận tải hành khách công cộng tại Thủ đô Hà Nội và thành phố Hồ Chí Minh.

+ **Giao thông đường bộ:**

Giai đoạn đến năm 2020: Khối lượng khách vận chuyển 5,5 tỷ hành khách với 165,5 tỷ hành khách luân chuyển. Khối lượng hàng hóa vận chuyển 760 triệu tấn với 35 tỷ tấn hàng hóa luân chuyển.

Phương tiện ô tô các loại khoảng 2,8 – 3,0 triệu xe.

Qua các mục tiêu phát triển trên ta thấy được vai trò quan trọng của vận tải đường sắt và đường bộ trong hiện tại và tương lai.

Mặc dù có tầm ảnh hưởng quan trọng đến sự phát triển kinh tế như vậy nhưng hệ thống thông tin hỗ trợ nhu cầu theo dõi lịch trình của tàu, xe buýt hay tìm đường đi giữa các bến... lại chưa phát triển tương xứng.

Bên cạnh đó, mặc dù mới bước vào thị trường vào năm 2007 nhưng tốc độ phát triển của smartphone (Điện thoại thông minh) và tablet (Máy tính bảng) đã tiến gần tới vị trí của máy tính vốn đã phát triển trước đó 30 năm. Riêng với smartphone, năm 2012, lượng thiết bị bán ra đã đạt hơn 600 triệu, một con số đáng mơ ước đối với các ngành khác. Số liệu này càng được khẳng định khi so sánh tương quan với các thiết bị kết nối của Microsoft: từng chiếm trên 80% năm 2009 (Windows, Windows Mobile) nhưng tính đến tháng 3/2013 chỉ còn dưới 25% do sự xuất hiện của iOS và Android. Qua đó, ta thấy sự bùng nổ mạnh mẽ của thị trường thiết bị di động và đây thực sự là xu hướng phát triển của tương lai.

Dựa vào các yếu tố nêu trên, em đã quyết định lựa chọn đề tài “Xây dựng hệ thống hỗ trợ tra cứu thông tin tàu xe trên nền tảng di động Android”. Sự thành công của đề tài

mang lại cho cộng đồng một ứng dụng có tính thực tiễn cao, giúp người sử dụng có thêm công cụ để tra cứu thông tin về tàu xe cũng như tìm đường đi giữa các bến một cách thuận tiện, giúp giảm thời gian và chi phí làm tăng chất lượng cuộc sống.

Nội dung của tài liệu này được tổ chức thành 4 chương:

Chương 1: Tổng quan về dự án.

Chương 2: Phân tích yêu cầu bài toán.

Chương 3: Thiết kế và triển khai.

Chương 4: Kết luận và hướng phát triển.

Lập trình ứng dụng cho thiết bị di động dựa trên nền tảng Android luôn là một mảnh đất màu mỡ cho các lập trình viên. Do làm việc độc lập, kiến thức còn hạn chế nên trong quá trình thực hiện đề tài không tránh khỏi những thiếu sót. Em rất mong nhận được sự đóng góp, chỉ bảo của các Thầy, Cô để em có thể hoàn thiện hơn ứng dụng, đáp ứng được phần nào nhu cầu về các ứng dụng có tính thực tiễn cao trong cuộc sống.

Một lần nữa, em xin chân thành cảm ơn

*Hà Nội, ngày .....tháng.....năm 2013*

Sinh viên

**Nguyễn Hoàng Long**

CHƯƠNG 1. TỔNG QUAN VỀ DỰ ÁN

1.1. Khảo sát thực tế

Ở thời điểm hiện tại, đã có những ứng dụng hỗ trợ người dùng tra cứu giờ tàu, xe buýt như sau:

Theo dõi giờ tàu qua chức năng “Xem giờ tàu” tại địa chỉ:

<http://www.vr.com.vn/gio-tau.html> (Báo ĐSVN thiết kế và quản lý)

HÀ NỘI	19.07	23.00	15.45	13.30	6.30	14.40	20.40	07.30	04.00
Giấy Bút									18.18
Phụ Lý			16.58		9.35		15.13	16.13	11.17
		20.35	17.28						
Tram Định		20.37	17.37		15.38	10.12		16.10	16.50
			18.07						11.57
Minh Bình					10.46				12.34
			18.06						

Hình 1.1. Dữ liệu giờ tàu dạng bảng



<http://vetau24h.com>

**GIỜ TÀU HÀ NỘI - SÀI GÒN**

- ↪ Giờ tàu Hỏa Tàu SE1
- ↪ Giờ tàu Hỏa Tàu SE3
- ↪ Giờ tàu Hỏa Tàu SE5
- ↪ Giờ tàu Hỏa Tàu SE7
- ↪ Giờ tàu Hỏa Tàu TN1

**GIỜ TÀU SÀI GÒN - HÀ NỘI**

- ↪ Giờ tàu Hỏa Tàu SE2
- ↪ Giờ tàu Hỏa Tàu SE4
- ↪ Giờ tàu Hỏa Tàu SE6
- ↪ Giờ tàu Hỏa Tàu SE8
- ↪ Giờ tàu Hỏa Tàu TN2

**Vận Tải Nhanh**

- ↪ SE1 Tàu nhanh m-sg
- ↪ SE2 Tàu nhanh sg-m
- ↪ SE3 Tàu nhanh nhất m-sg
- ↪ SE4 Tàu nhanh nhất sg-m
- ↪ SE5 Tàu chậm m-sg
- ↪ SE6 Tàu chậm sg-m
- ↪ SE7 Tàu nhanh m-sg
- ↪ SE8 Tàu nhanh sg-m
- ↪ TN1 Tàu chậm nhất m-sg
- ↪ TN2 Tàu chậm nhất sg-m

**Ghi chú:**

KH: đường thẳng từ ga xuất phát đến ga đến mà quý khách lựa chọn, ví dụ quý khách muốn đi từ ga Hà Nội đến ga Thanh Hóa thì ga xuất phát là ga Hà Nội và ga đến là ga Thanh Hóa.  
 Giờ đến, giờ đi: là giờ tàu đến và giờ đi từ một ga mà quý khách lựa chọn, ví dụ nếu quý khách muốn đi tàu từ ga Thanh Hóa thì giờ tàu đến là 22h15 và tàu sẽ đứng tại đây 3 phút và xuất phát lúc 22h18, năng 05 với ga Hà Nội, vì là ga xuất phát, Ga Sài Gòn là ga cuối cùng nên giờ đến và giờ đi trùng nhau.

Ga	KM	Giờ đến	Giờ đi
Nam Định	87	20:35	20:39
Thanh Hóa	175	22:15	22:18
Vinh	319	00:44	00:49
Đồng Hới	522	04:31	04:43
Đồng Hà	622	06:43	06:46
Huế	688	07:57	08:02
Đà Nẵng	791	10:31	10:46
Tam Kỳ	865	12:05	12:08
Quảng Ngãi	928	13:28	13:31
Điêu Trì	1096	16:16	16:28
Tuy Hòa	1198	18:33	18:36
Nha Trang	1315	20:28	20:33
Tháp Chàm	1408	22:04	22:07
Sài Gòn	1726	04:10	04:10

Hình 1.2. Dữ liệu giờ tàu dạng ảnh

+ Danh sách các tuyến xe buýt và các điểm dừng được liệt kê ở dạng text

<http://www.xebushanoi.com/forum/showthread.php?t=1267> (có chức năng tìm đường nhưng không hoạt động)

**03. Bến xe Giáp Bát - Bến xe Gia Lâm**

Tần suất: 10 - 15 phút/chuyến  
 Thời gian hoạt động: 5h08 - 21h08

**Lộ trình:**

**Lộ trình đi:** Bến xe Giáp Bát - Giải Phóng - Lê Duẩn - Nguyễn Thượng Hiền - Yết Kiêu - Trần Hưng Đạo - Trần Quang Khải - Trần Nhật Duật (Quay đầu trước phố Hàng Khoai) - Nguyễn Văn Cừ - Nguyễn Sơn - Ngọc Lâm - Ngõ Gia Khâm - Bến xe Gia Lâm.

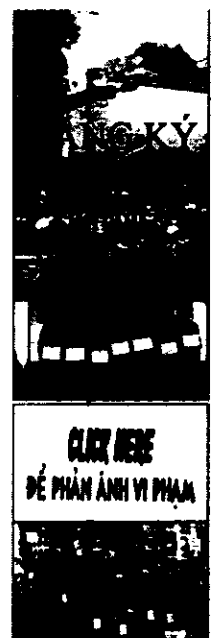
**Lộ trình về:** Bến xe Gia Lâm - Ngõ Gia Khâm - Ngọc Lâm - Nguyễn Văn Cừ - Trần Nhật Duật (Quay đầu trước phố Hàng Khoai) - Trần Quang Khải - Trần Khánh Dư - Trần Hưng Đạo - Lê Duẩn - Giải Phóng - Ngõ 3 Dùi Cỏ - Bến xe Giáp Bát.

**03B. BX Giáp Bát - KĐT Việt Hưng, Vincom Village**

Tần suất: 25 - 30 phút/chuyến  
 Thời gian hoạt động:  
 Sáng: Đầu A 6h20 - 8h10, Đầu B: 6h00 - 8h20  
 Trưa: Đầu A: 11h00 - 12h50, Đầu B: 11h10 - 13h00  
 Chiều: Đầu A: 16h30 - 19h50, Đầu B: 16h50 - 20h00  
 Giá vé: 3.000đ/1 lượt

**Lộ trình:**

**Lộ trình đi:** BX Giáp Bát - Giải Phóng - Lê Duẩn - Nguyễn Thượng Hiền - Yết Kiêu - Trần Hưng Đạo - Trần Khánh Dư - Trần Quang Khải - Trần Nhật Duật - Long Biên (Điểm quay đầu trước phố Hàng Khoai) - Cầu Chương Dương - Nguyễn Văn Cừ - Ngõ Gia Tự - Nguyễn Cao Luyện - Tòa nhà P3 KĐT Việt Hưng - Khu Green House - Trường Lâm - Vincom Village (TT Thương Mại Vincom Center Long Biên)



Hình 1.3. Dữ liệu tuyến buýt dạng web (liệt kê)

+ Trang web của Công ty Vận tải và Dịch vụ công cộng Hà Nội (có chức năng tìm đường nhưng không hoạt động)

The screenshot shows the HANOIBUS website interface. On the left is a navigation menu with items like 'Trang chủ hanoibus', 'Giới thiệu', 'Cơ cấu tổ chức', 'Định hướng phát triển', 'Tín tức hanoibus', 'Lộ trình tuyến', 'Các loại vé', 'Đăng ký vé tháng', 'Tiểu chí - Nội quy', 'Hướng dẫn đi xe', 'Nhận diện thương hiệu', and 'Người lái việc tốt'. Below the menu is a 'Thông tin điều hành tuyến' section with a small image of a bus. The main content area is titled 'Lộ trình các tuyến xe buýt' and features a table for 'LỘ TRÌNH CÁC TUYẾN XE BUÝT THÀNH PHỐ HÀ NỘI'. The table has four columns: 'SỐ HIỆU TUYẾN', 'TÊN TUYẾN THỜI GIAN HOẠT ĐỘNG TẦN SUẤT HOẠT ĐỘNG', 'LƯỢT ĐI QUA CÁC ĐƯỜNG PHỐ CHÍNH', and 'LƯỢT VỀ QUA CÁC ĐƯỜNG PHỐ CHÍNH'. The first row is for route 01, 'Long Biên - BX Yên Nghĩa', with a detailed list of stops and directions.

Hình 1.4. Dữ liệu tuyến buýt dạng web (dạng bảng)

Qua những ví dụ trên ta thấy, thông tin về giờ tàu hay các tuyến xe buýt đã được quan tâm phát triển nhưng vẫn đang ở dạng rất đơn giản (liệt kê, ảnh), hoặc chức năng tìm đường giữa các tuyến buýt ở Hà Nội đã có nhưng không hoạt động. Việc này gây lãng phí thời gian khi người dùng chỉ muốn tra cứu giờ tàu giữa 2 ga cụ thể hay tìm đường giữa các tuyến buýt. Ngoài ra, nếu muốn tra cứu, người dùng phải sử dụng máy tính cá nhân và có kết nối Internet mới thực hiện được. Hạn chế này làm giảm tốc độ lưu thông của người và hàng hóa, vì không phải lúc nào cũng có sẵn máy tính có kết nối Internet cũng như các trang web này không phải lúc nào cũng sẵn sàng đáp ứng (trường hợp trang web gặp sự cố). Do đó, giải pháp đưa các thông tin này lên thiết bị di động sẽ khắc phục được điểm yếu này.

## 1.2. Giải pháp

Vì dữ liệu về giờ tàu (liên quan đến các tuyến, ga và giờ tàu) có tính ổn định. Ít có sự thay đổi trong nhiều năm. Hay dữ liệu về các tuyến buýt cũng ít thay đổi trong nhiều tháng. Vì vậy, việc xây dựng một ứng dụng cho phép tra cứu các thông tin này trên các thiết bị di động chạy hệ điều hành Android là một giải pháp đạt được nhiều ưu thế:

Theo số liệu của IDC: tính đến quý 3 năm 2012, hệ điều hành Android đã chiếm 75% thị phần smartphone trên toàn cầu, khẳng định tính phổ biến của hệ điều hành này.

Người dùng có thể tiếp cận các dữ liệu này mọi lúc, mọi nơi do ứng dụng này được phát triển cho các thiết bị di động.

Không nhất thiết phải có kết nối Internet mới có thể tra cứu được, vì các dữ liệu này là ổn định trong một khoảng thời gian dài và có thể được lưu trữ offline.

**CHƯƠNG 2. PHÂN TÍCH YÊU CẦU BÀI TOÁN****2.1. Mục tiêu**

Để xây dựng được một ứng dụng hữu ích và phù hợp với đề tài đặt ra thì ứng dụng này phải đạt được các mục tiêu sau đây:

- Cho phép người dùng xem thông tin chi tiết về giờ tàu xuất/đến một bến, trong một tuyến tàu nằm trong hệ thống vận tải hành khách đường sắt của Việt Nam. Thông tin này phải bao gồm
  - Tuyến tàu: tên tuyến.
  - Mã tàu: mỗi tuyến lại có nhiều tàu với mã khác nhau.
  - Tên bến-giờ tàu: mỗi tàu được phân biệt bởi mã tàu lại đi qua các bến khác nhau, giờ đến/xuất bến cũng khác nhau.
- Cho phép người dùng xem thông tin chi tiết về các tuyến xe buýt thuộc hệ thống xe buýt nội đô thành phố Hà Nội. Các thông tin này bao gồm
  - Danh sách tuyến xe buýt: phải có đầy đủ thông tin về các tuyến buýt nội đô thành phố Hà Nội được đăng tải tại địa chỉ <http://www.transerco.vn>, bao gồm tên tuyến và mã tuyến.
  - Chi tiết tuyến: bao gồm tên các bến lần lượt theo chiều đi hoặc chiều về của mỗi tuyến.
- Cho phép người dùng tìm hướng dẫn chuyển tuyến giữa hai bến bất kì trong hệ thống tuyến xe buýt nội đô thành phố Hà Nội. Trường hợp có hướng dẫn chuyển tuyến phù hợp, ứng dụng phải hiển thị hướng dẫn đó, còn ngược lại thì phải có thông báo không có hướng dẫn phù hợp. Thông tin chỉ dẫn cần bao gồm:
  - Tên bến: là tên của bến người dùng cần lên hoặc xuống xe buýt.
  - Mã tuyến: là mã số của tuyến ứng với tên bến bến người dùng cần lên hoặc xuống xe buýt.
  - Tên tuyến: là tên của tuyến xe buýt ứng với tên bến bến người dùng cần lên hoặc xuống xe buýt.
  - Hướng dẫn: là chỉ dẫn cho người dùng biết tại mỗi bến họ cần “lên” hay “xuống” xe.
- Hỗ trợ người dùng để họ biết được vị trí hiện tại của mình, đồng thời tìm kiếm và hiển thị các bến xe buýt gần với vị trí hiện tại của người dùng trên bản đồ trên bản đồ số(Google Map). Các thông tin hiển thị cần bao gồm:

- Đánh dấu vị trí hiện tại của người dùng và tên của vị trí đó.
- Đánh dấu vị trí các bến xe buýt gần vị trí hiện tại và thông tin về tên địa điểm gắn với các đánh dấu đó.

## **2.2. Yêu cầu kỹ thuật**

Để trở thành một sản phẩm có tính thực tế, hỗ trợ tốt cho nhu cầu của người dùng thì ứng dụng phải đạt được những yêu cầu kỹ thuật sau:

- Về chức năng:
  - Các chức năng phải được phân chia rõ ràng, có thể hoạt động độc lập mà không cần dựa vào các chức năng khác.
  - Tốc độ xử lý yêu cầu phải nhanh và ổn định
- Về giao diện:
  - Ứng dụng phải hiển thị được trên tất cả các dòng điện thoại có cấu hình được đưa ra ở mục 3.7
  - Các thành phần hiển thị như thông báo, danh sách, bản đồ... phải rõ ràng, dễ đọc, dễ theo dõi
- Về khả năng cập nhật:
  - Ứng dụng phải đảm bảo khả năng cập nhật dễ dàng khi có các thay đổi của nguồn dữ liệu như thay đổi lộ trình tuyến buýt hay dữ liệu bản đồ Google Map...

## **2.3. Yêu cầu nghiệp vụ**

Sau khi đánh giá và phân tích hiện trạng của các ứng dụng trên thị trường khi giải quyết yêu cầu cho bài toán này, ứng dụng cần phải có được các yêu cầu nghiệp vụ như sau:

### **BR1: Xem giờ tàu đi/đến bến**

Trợ giúp người dùng theo dõi giờ tàu xuất/vào các bến thuộc một tuyến cụ thể.

Đối tượng sử dụng: người dùng đầu cuối

Các thông tin cần quản lý:

- Tên các tuyến tàu thuộc hệ thống đường sắt vận tải hành khách của Việt Nam
- Mã các tàu ứng với từng tuyến, thông tin về giờ tàu ứng với mỗi mã tàu

### **BR2: Xem thông tin tuyến buýt**

Cho phép người dùng xem các tuyến xe buýt thuộc hệ thống xe buýt nội đô (Hà Nội), đưa ra danh sách các bến theo trình tự chiều đi/về ứng với mỗi tuyến.

Đối tượng sử dụng: người dùng đầu cuối

Các thông tin cần quản lý:

- Tên, mã các tuyến xe buýt thuộc hệ thống xe buýt nội đô Hà Nội
- Tên các bến thuộc một tuyến, trình tự các bến theo chiều đi/về

**BR3: Xem hướng dẫn chuyển tuyến xe buýt**

Hỗ trợ người dùng tìm ra một phương án chuyển bến ít nhất để di chuyển giữa hai bến bất kì bằng xe buýt.

Đối tượng sử dụng: người dùng đầu cuối

Các thông tin cần quản lý: tương tự như BR2 nhưng bổ sung thêm thông tin về khoảng cách giữa các bến.

**BR4: Xác định vị trí hiện tại trên bản đồ số**

Đánh dấu vị trí hiện tại của người dùng trên bản đồ số kèm thông tin về địa chỉ của vị trí đó.

Đối tượng sử dụng: người dùng đầu cuối

Các thông tin cần quản lý:

- Nội dung hiển thị về địa điểm hiện tại

**BR5: Tìm các bến xe buýt gần vị trí hiện tại**

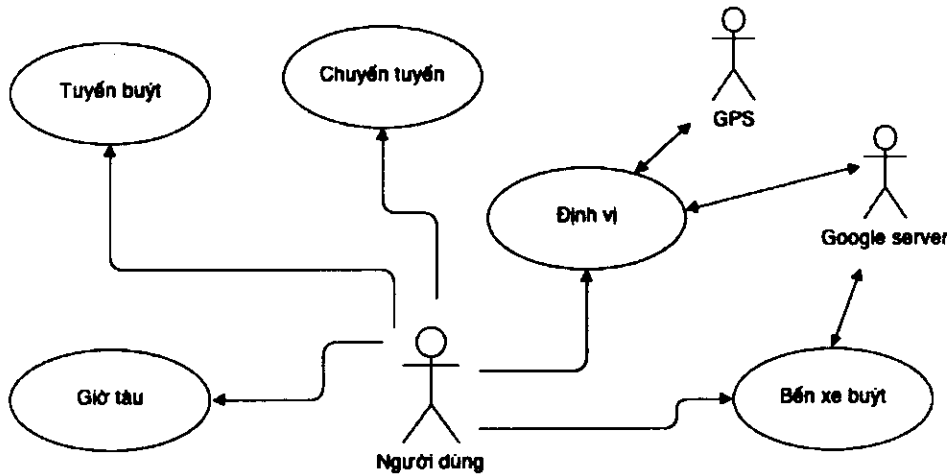
Đánh dấu các bến xe buýt gần với vị trí hiện tại của người dùng kèm theo thông tin về địa chỉ của các bến đó.

Đối tượng sử dụng: người dùng đầu cuối

Các thông tin cần quản lý:

- Bán kính tìm kiếm bến xe buýt gần nhất
- Thông tin về địa điểm của các bến xe buýt đó

Sơ đồ USECASE



Hình 2.1. Sơ đồ USECASE của ứng dụng

**Các tác nhân:**

**Người dùng:** là tác nhân chính sử dụng các chức năng của chương trình.

**Google server:** là hệ thống bên ngoài cung cấp dữ liệu để các chức năng liên quan có thể hoạt động.

**Các chức năng:**

UC #001: Giờ tàu

Chức năng này cho phép người dùng theo dõi bảng giờ tàu của một tuyến nhất định, bao gồm giờ tàu xuất bến hoặc đến bến.

UC #002: Tuyến buýt

Chức năng này giúp người dùng xem thông tin chi tiết của một tuyến buýt, bao gồm số hiệu tuyến, tên tuyến, liệt kê các bến theo trình tự chiều đi/về.

UC #003: Chuyển tuyến

Chức năng này sẽ hỗ trợ người dùng tìm ra một phương án chuyển tuyến xe buýt ngắn nhất giữa bến đầu và bến cuối do người dùng nhập vào.

UC #004: Định vị

Chức năng này cho biết vị trí hiện tại của người dùng trên bản đồ số (Google Map) và địa chỉ của vị trí đó.

UC #005: Bến xe buýt

Chức năng này sẽ đánh dấu vị trí các bến xe buýt gần với vị trí hiện tại của người dùng kèm theo địa chỉ của các bến đó trên bản đồ số (Google Map).

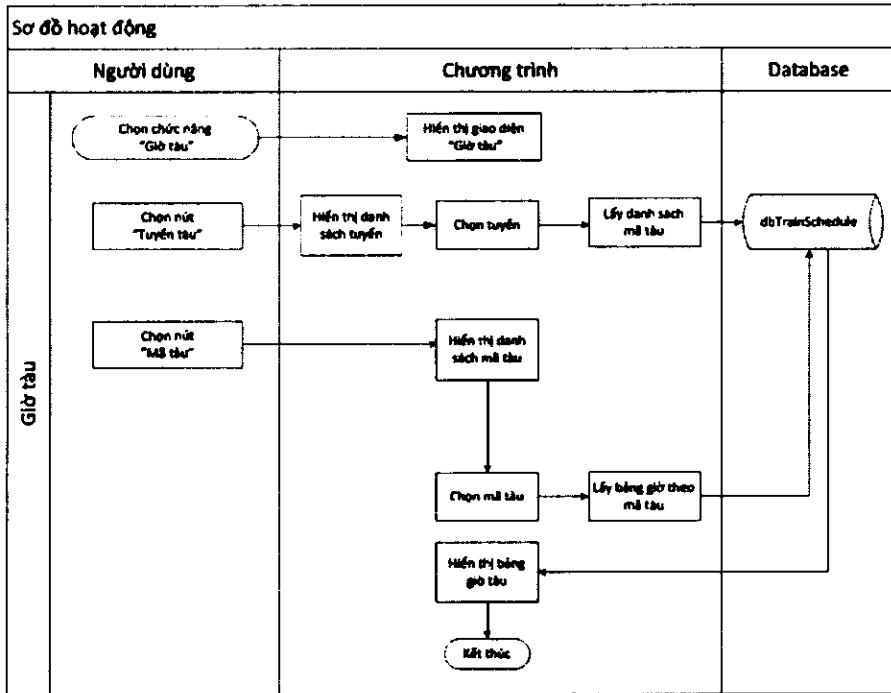
**2.4. Ánh xạ yêu cầu nghiệp vụ và chức năng:**

<b>BR</b>	<b>Mô tả</b>	<b>UC</b>
BR1	Xem giờ tàu đi/đến bến	UC#001
BR2	Xem thông tin tuyến buýt	UC#002
BR3	Xem hướng dẫn chuyển tuyến xe buýt	UC#003
BR4	Xác định vị trí hiện tại trên bản đồ số	UC#004
BR5	Tìm các bến xe buýt gần vị trí hiện tại	UC#005

**Bảng 2.1. Bảng ánh xạ yêu cầu nghiệp vụ và chức năng**

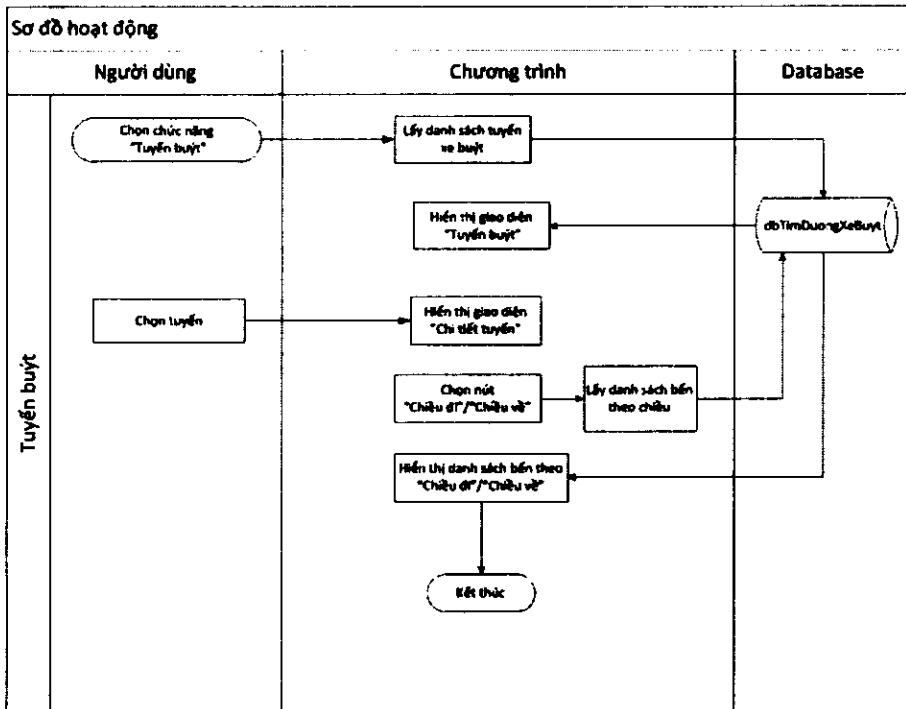
2.5. Đặc tả hệ thống

Sơ đồ hoạt động của chức năng Giờ tàu



Hình 2.2. Sơ đồ hoạt động của chức năng Giờ tàu

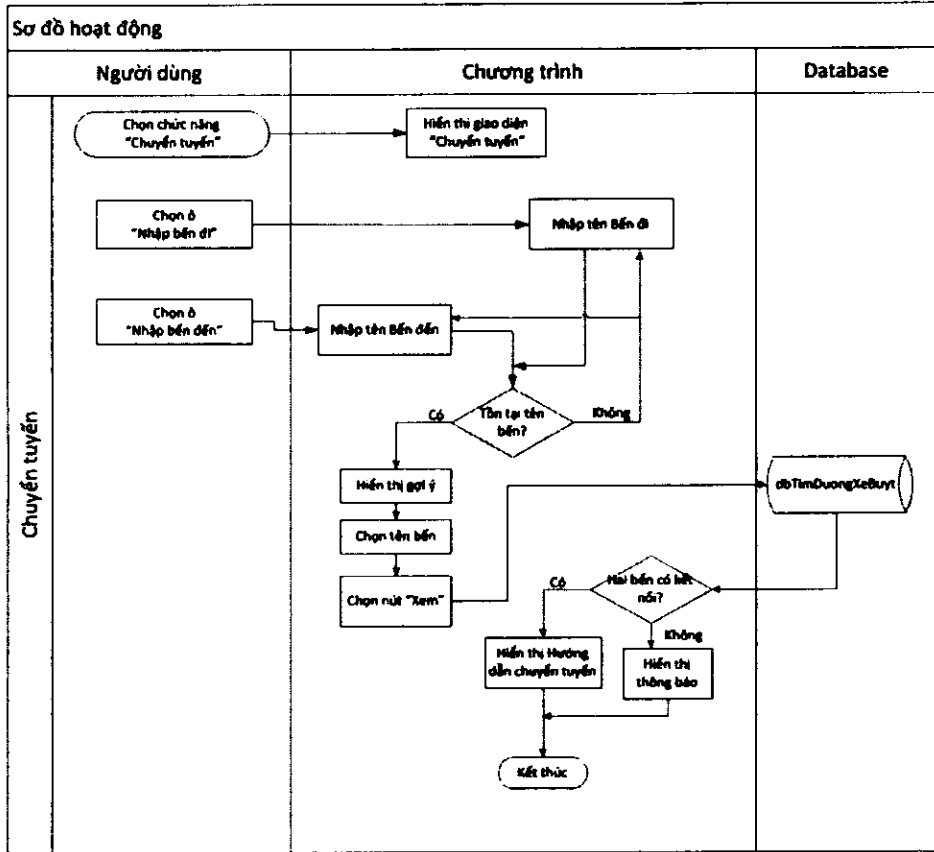
Sơ đồ hoạt động của chức năng Tuyến buýt



Hình 2.3. Sơ đồ hoạt động của chức năng Tuyến buýt

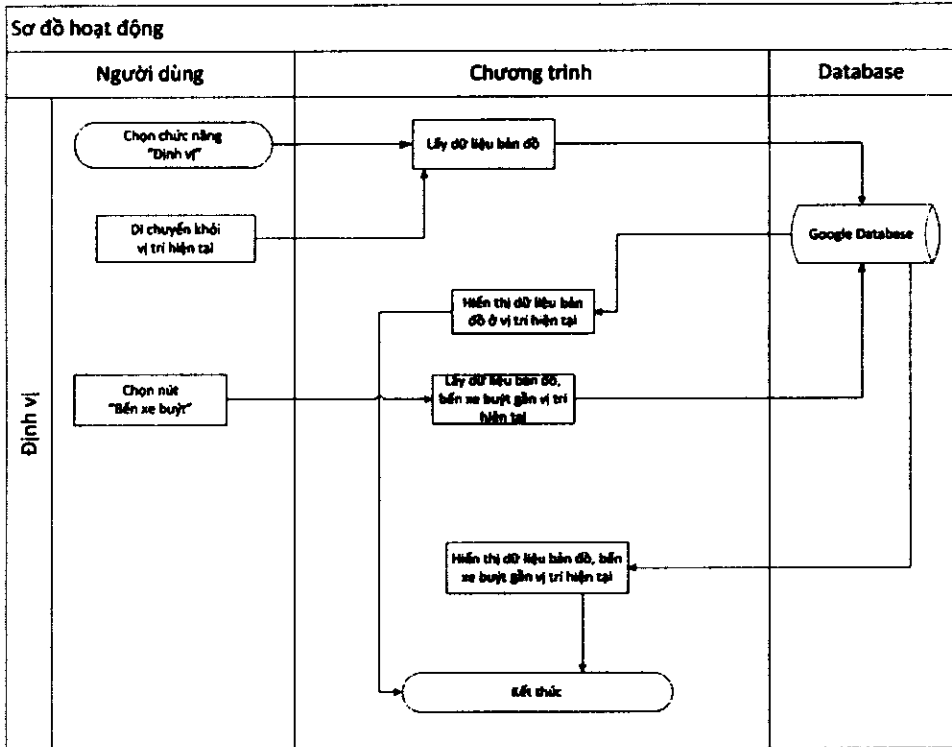


Sơ đồ hoạt động của chức năng Chuyển tuyến



Hình 2.4. Sơ đồ hoạt động của chức năng Chuyển tuyến

Sơ đồ hoạt động của chức năng Định vị và Bến xe buýt



Hình 2.5. Sơ đồ hoạt động của chức năng Định vị và Bến xe buýt

2.3. Kế hoạch dự án

Để dự án đảm bảo tiến độ nhằm cho ra sản phẩm đạt được các yêu cầu đã đề ra, dưới đây là bảng kế hoạch chi tiết từng giai đoạn của dự án.

STT	Tên kế hoạch	Thời gian
1	Khởi tạo dự án, lập kế hoạch thực hiện dự án.	2 ngày
3	Phân tích yêu cầu chức năng, đưa ra mô hình và luồng công việc tổng quan.	1 tuần
4	Thiết kế mô hình UC và đặc tả các chức năng hệ thống.	3 tuần
5	Thiết kế CSDL.	5 ngày
6	Thiết kế giao diện hệ thống.	3 tuần
7	Lập trình.	4 tháng
8	Kiểm thử từng giai đoạn.	Theo tuần
9	Kiểm thử sau khi hệ thống hoàn thành.	1 tuần

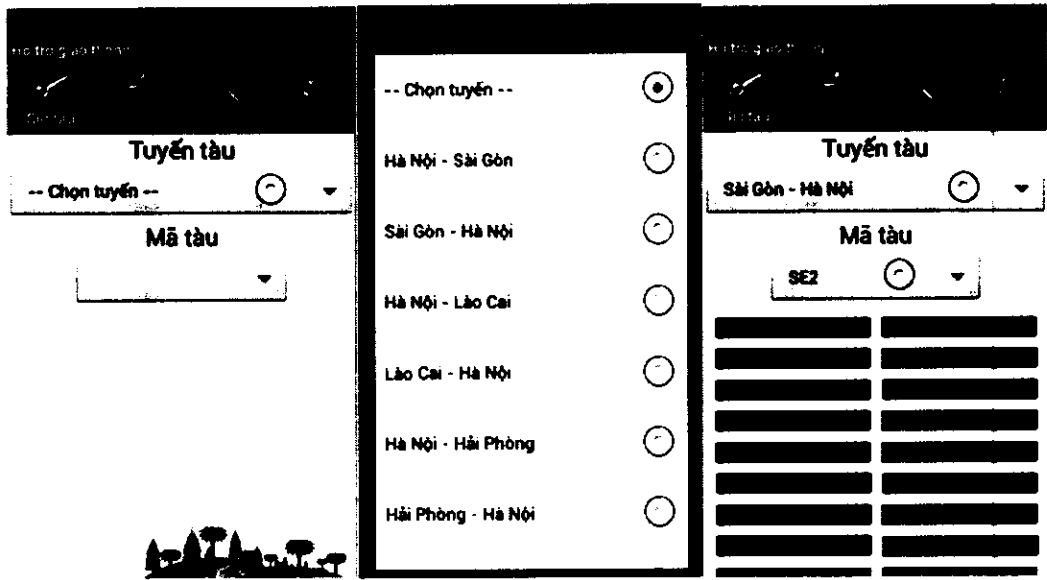
Hình 2.6. Kế hoạch dự án

2.6. Đặc tả chức năng

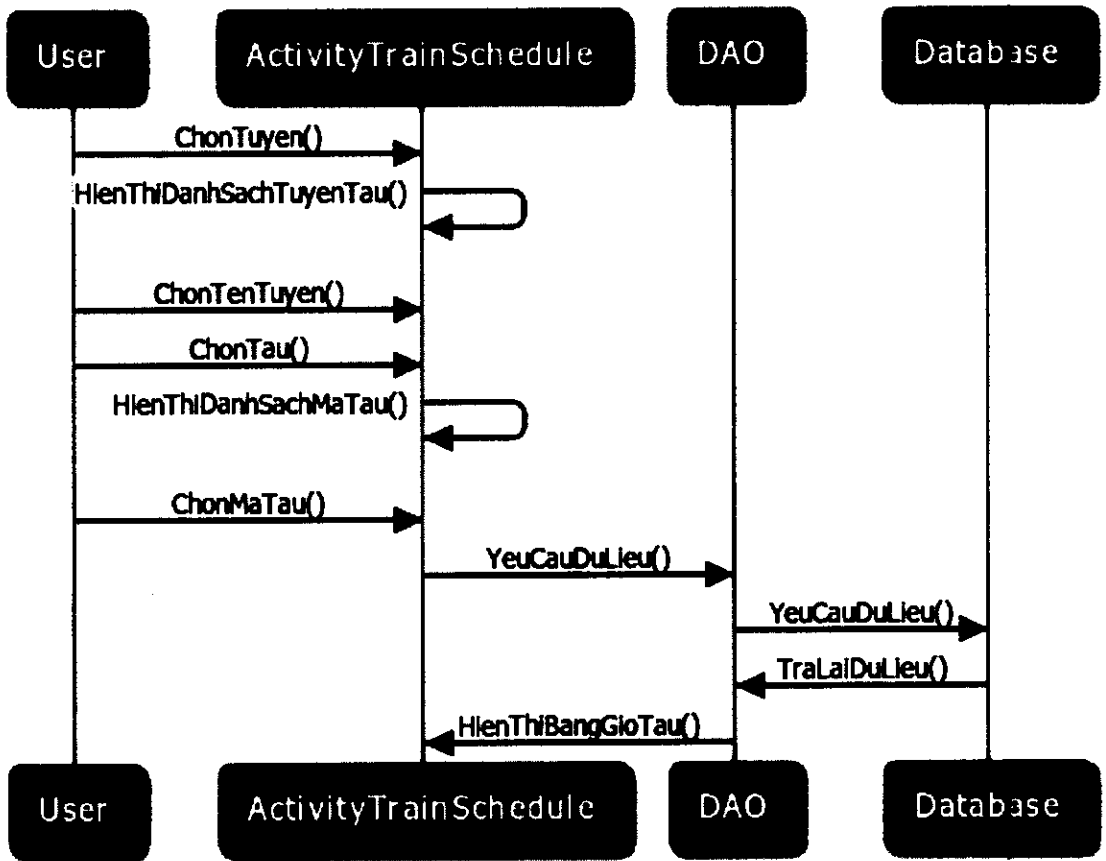
**Chức năng Giờ tàu**

<b>UC #001</b>		<b>Giờ tàu</b>	<b>Độ phức tạp: thấp</b>
<b>Mô tả</b>		Chức năng này cho phép người dùng theo dõi bảng giờ tàu của một tuyến nhất định, bao gồm giờ tàu xuất bến hoặc đến bến.	
<b>Tác nhân</b>	<b>Chính</b>	Người dùng	
	<b>Phụ</b>	Không có	
<b>Tiền điều kiện</b>		Không có	
<b>Hậu điều kiện</b>	<b>Thành công</b>	Hiển thị danh sách bến tàu và giờ tàu tương ứng	
	<b>Lỗi</b>	Thông báo lỗi và giữ nguyên giao diện Giờ tàu	
<b>ĐẶC TẢ CHỨC NĂNG</b>			
<b>Luồng sự kiện chính/Kịch bản chính</b>			
<p>User case này bắt đầu khi người dùng muốn xem giờ tàu rời/xuất bến của các bến thuộc một tuyến cụ thể.</p> <ol style="list-style-type: none"> <li>1. Người dùng chọn chức năng <b>Giờ tàu</b>.</li> <li>2. Ứng dụng sẽ hiển thị giao diện cho phép người dùng chọn <b>Tuyến tàu, Mã tàu</b> cần tra cứu .</li> <li>3. Người dùng chọn tên tuyến tàu cần tra cứu.</li> <li>4. Ứng dụng lấy dữ liệu giờ tàu của mã tàu đầu tiên thuộc tuyến từ Cơ sở dữ liệu sau đó, hiển thị danh sách bến tàu kèm theo giờ tương ứng.</li> <li>5. Người dùng chọn <b>Mã tàu</b> mong muốn trong danh sách giờ tàu.</li> <li>6. Ứng dụng lấy dữ liệu giờ tàu của mã tàu được chọn từ Cơ sở dữ liệu sau đó, hiển thị danh sách bến tàu kèm theo giờ tương ứng.</li> </ol>			
<b>Luồng sự kiện phát sinh/ Kịch bản phát sinh</b>			
<b>1. Ngoại lệ không tồn tại Cơ sở dữ liệu Giờ tàu</b>			
1.1. Ứng dụng thông báo “Không có cơ sở dữ liệu Giờ tàu”.			

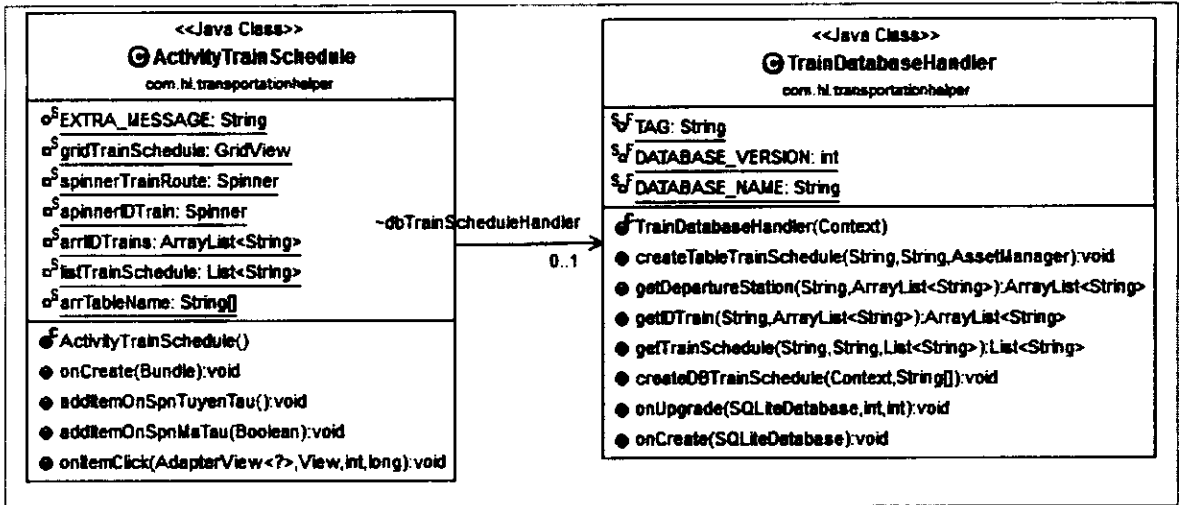
Giao diện minh họa



Sơ đồ trình tự (Sequence diagram)



Sơ đồ lớp chi tiết (Class diagram)



**Chức năng Tuyến buýt**

<b>UC #002</b>		<b>Tuyến buýt</b>	<b>Độ phức tạp: trung bình</b>
<b>Mô tả</b>		Chức năng này giúp người dùng xem thông tin chi tiết của một tuyến buýt, bao gồm số hiệu tuyến, tên tuyến, liệt kê các bến theo trình tự chiều đi/về.	
<b>Tác nhân</b>	<b>Chính</b>	Người dùng	
	<b>Phụ</b>	Không có	
<b>Tiền điều kiện</b>		Không có	
<b>Hậu điều kiện</b>	<b>Thành công</b>	Hiển thị danh sách bao gồm mã tuyến và tên tuyến buýt	
	<b>Lỗi</b>	Thông báo lỗi và giữ nguyên giao diện Tuyến buýt	
<b>ĐẶC TẢ CHỨC NĂNG</b>			
<b>Luồng sự kiện chính/Kịch bản chính</b>			
User case này bắt đầu khi người dùng muốn xem thông tin chi tiết các tuyến buýt có trong cơ sở dữ liệu (Tên tuyến, mã tuyến, các bến chiều đi/về).			
<ol style="list-style-type: none"> <li>1. Người dùng chọn chức năng <b>Tuyến buýt</b>.</li> <li>2. Ứng dụng sẽ hiển thị danh sách các tuyến buýt</li> <li>3. Người dùng chọn một tuyến cần xem thông tin.</li> <li>4. Ứng dụng lấy dữ liệu từ Cơ sở dữ liệu sau đó, hiển thị danh sách các bến xe buýt theo trình tự chiều đi của tuyến.</li> </ol>			

5. Người dùng bấm nút **Chiều đi**.


















6. Ứng dụng lấy dữ liệu từ Cơ sở dữ liệu sau đó, hiển thị danh sách các bến xe buýt theo trình tự chiều về của tuyến.

**Luồng sự kiện phát sinh/ Kịch bản phát sinh**

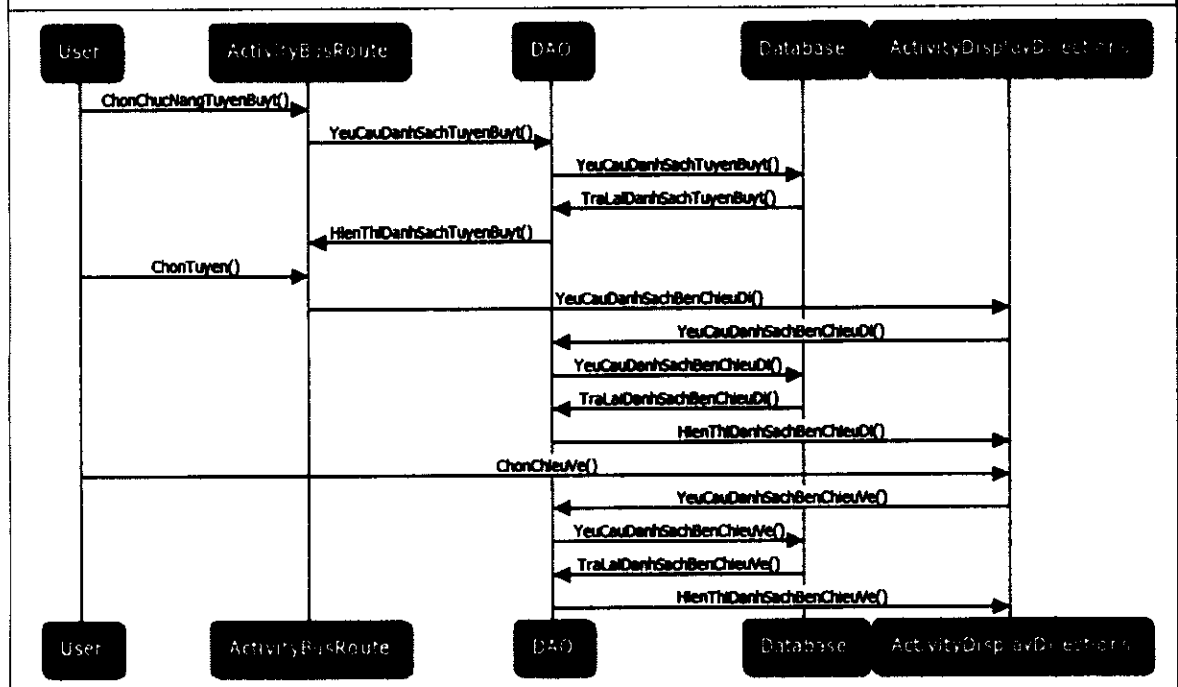
**1. Ngoại lệ không tồn tại Cơ sở dữ liệu Tuyến buýt**

1.1. Ứng dụng thông báo “Không có cơ sở dữ liệu Tuyến buýt”.

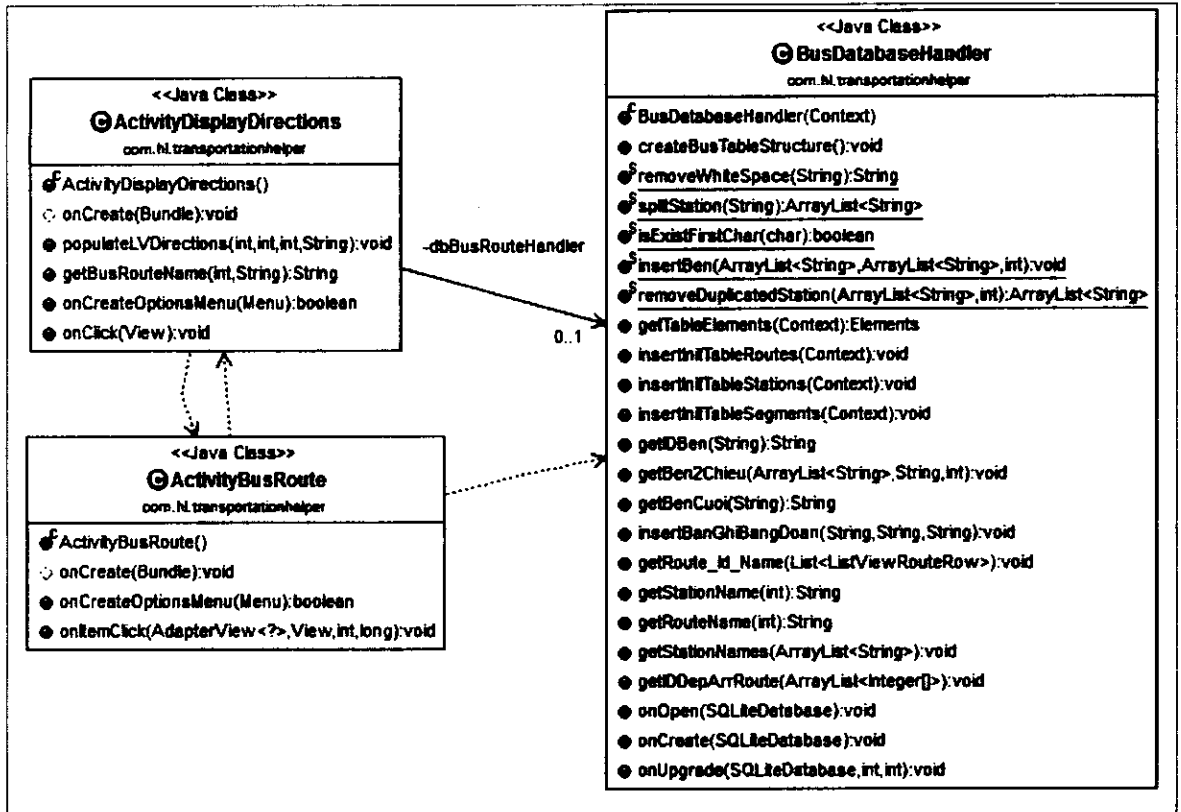
Giao diện minh họa

Hình ảnh giao diện	Chiều đi	Chiều về
 01 Long Biên - Đến xe Yên Nghĩa	 Bắc Cỗ (Bãi đỗ xe Trần Khánh Dư)	 Bến xe Yên Nghĩa
 02 Bắc Cỗ - Bến xe Yên Nghĩa	 Trần Khánh Dư (đường dưới)	 Quốc lộ 6
 03 Đến xe Giáp Bát - Đến xe Gia Lâm	 Trần Hưng Đạo	 Ba La
 04 Long Biên - Yên Sở	 Lê Thánh Tông	 Quang Trung (Hà Đông)
 05 Khu đô thị Linh Đàm - Phú Diễn	 Tráng Tiên	 Trần Phú (Hà Đông)
 06 Đến xe Giáp Bát - Cầu Giấy		 Nguyễn Trãi

Sơ đồ trình tự (Sequence diagram)



Sơ đồ lớp chi tiết (Class diagram)



**Chức năng Chuyển tuyến**

UC #003		Chuyển tuyến	Độ phức tạp: cao
Mô tả		Chức năng này sẽ hỗ trợ người dùng tìm ra một phương án chuyển tuyến xe buýt ngắn nhất giữa bến đầu và bến cuối do người dùng nhập vào.	
Tác nhân	Chính	Người dùng	
	Phụ	Không có	
Tiền điều kiện		Không có	
Hậu điều kiện	Thành công	Hiện thị danh sách hướng dẫn cách chuyển tuyến ngắn nhất để đi từ bến đầu đến bến cuối	
	Lỗi	Thông báo lỗi và giữ nguyên giao diện Chuyển tuyến	
<b>ĐẶC TẢ CHỨC NĂNG</b>			
<b>Luồng sự kiện chính/Kịch bản chính</b>			
User case này bắt đầu khi người dùng muốn xem hướng dẫn di chuyển từ bến đầu đến bến cuối sao cho số tuyến phải chuyển là ít nhất.			



1. Người dùng chọn chức năng **Chuyển tuyến**.
2. Ứng dụng sẽ hiện thị 2 ô để người dùng nhập vào bến đầu, bến cuối.
3. Người dùng nhập vào bến đầu, bến cuối.
4. Ứng dụng hiển thị gợi ý tên các bến ứng với chuỗi người dùng nhập vào.
5. Người dùng chọn tên bến đầu, cuối mong muốn.
6. Người dùng chọn nút **Xem**
7. Ứng dụng lấy giá trị bến đầu/cuối nhập vào sử dụng thuật toán Dijkstra kết hợp với Cơ sở dữ liệu để đưa ra phương án chuyển tuyến.
8. Hiển thị danh sách hướng dẫn chuyển tuyến để đi từ bến đầu đến bến cuối.

**Luồng sự kiện phát sinh/ Kịch bản phát sinh**

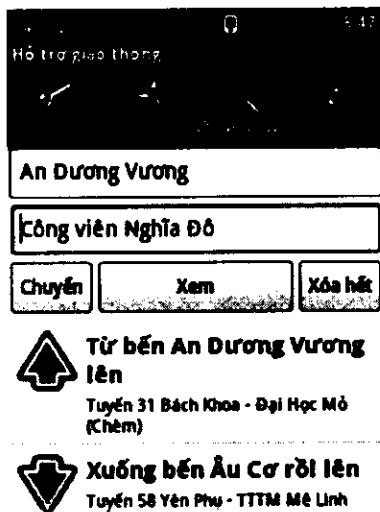
**1. Ngoại lệ không tồn tại Cơ sở dữ liệu Tuyến buýt**

1.1. Ứng dụng thông báo “Không có cơ sở dữ liệu Tuyến buýt”.

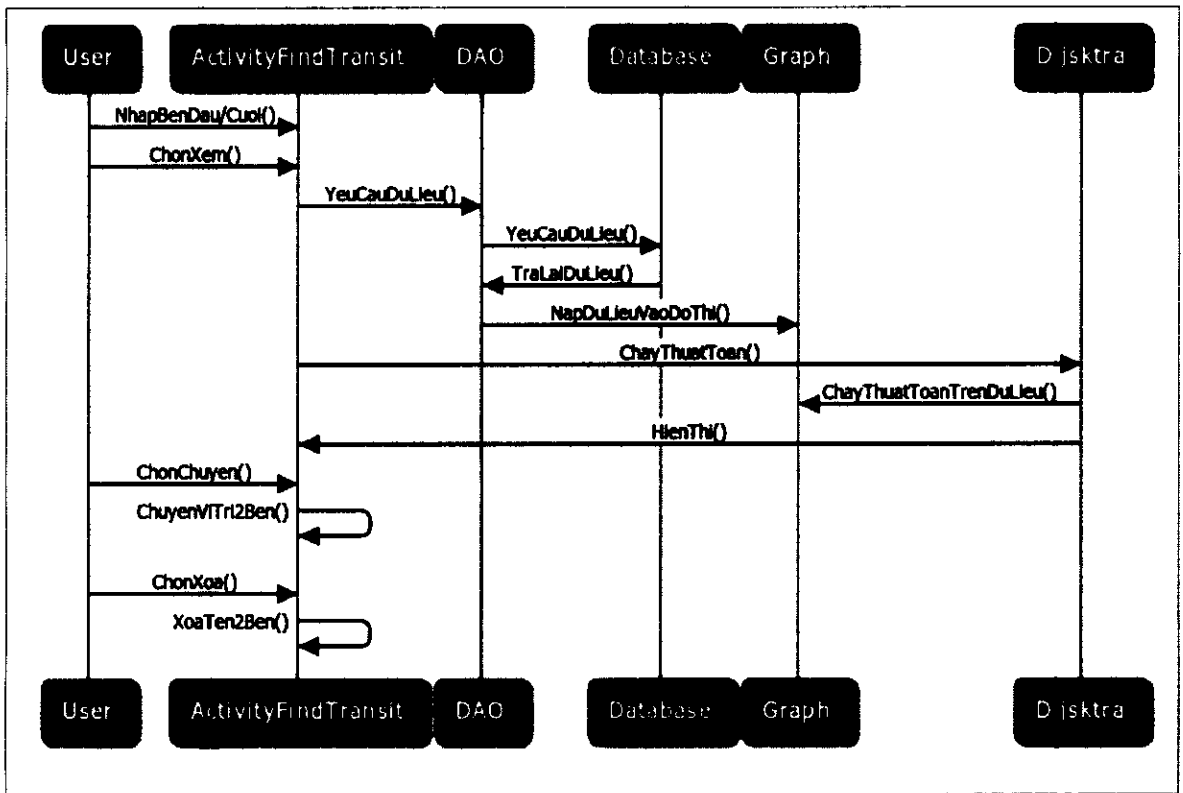
**2. Ngoại lệ không tồn tại phương án chuyển tuyến**

2.1. Ứng dụng thông báo “Không có phương án chuyển tuyến”.

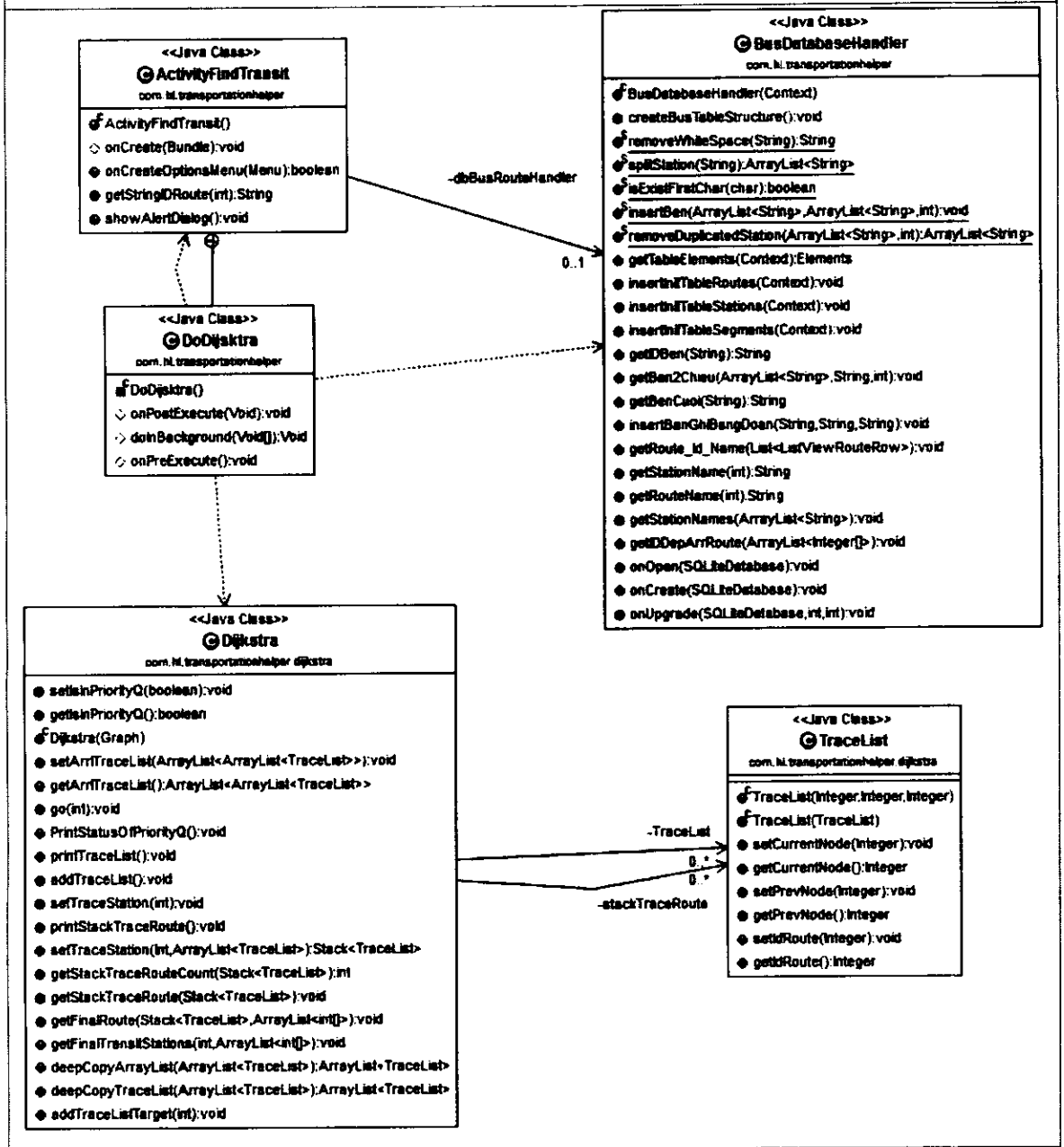
**Giao diện minh họa**



**Sơ đồ trình tự (Sequence diagram)**



Sơ đồ lớp chi tiết (Class diagram)

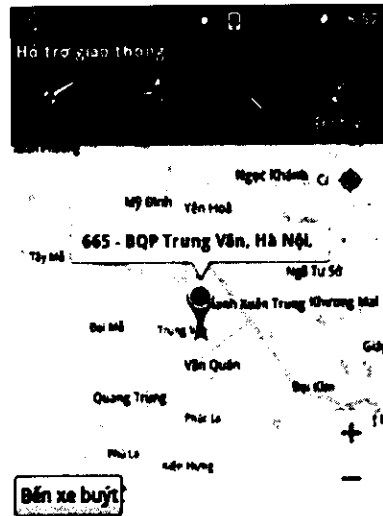


Chức năng Định vị

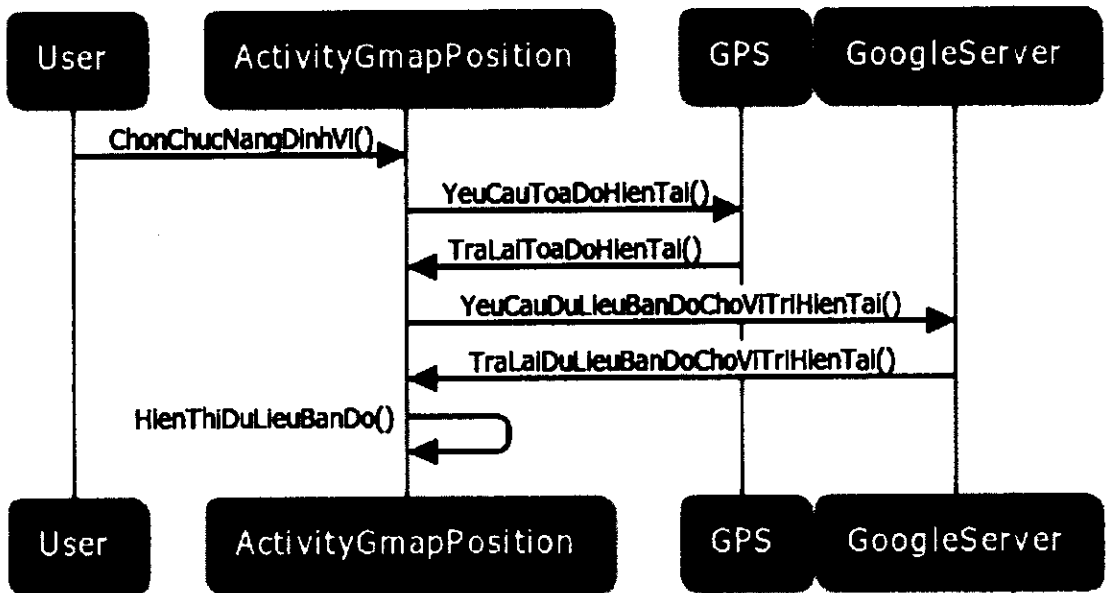
UC #004	Định vị	Độ phức tạp: trung bình
Mô tả	Chức năng này cho biết vị trí hiện tại của người dùng trên bản đồ số (Google Map) và địa chỉ của vị trí đó.	
Tác nhân	Chính	Người dùng
	Phụ	Không có
Tiền điều kiện	Không có	

<b>Hậu điều kiện</b>	<b>Thành công</b>	Hiển thị đánh dấu vị trí người dùng trên bản đồ số (Google Map) và địa chỉ của vị trí đó.
	<b>Lỗi</b>	Thông báo lỗi.
<b>ĐẶC TẢ CHỨC NĂNG</b>		
<b>Luồng sự kiện chính/Kịch bản chính</b>		
<p>User case này bắt đầu khi người dùng muốn biết địa chỉ của vị trí hiện tại và nó nằm ở đâu trên bản đồ số (Google Map).</p> <ol style="list-style-type: none"> <li>1. Người dùng chọn chức năng <b>Định vị</b>.</li> <li>2. Ứng dụng tự động lấy dữ liệu vị trí hiện tại thông qua chức năng GPS sau đó, gửi yêu cầu về dữ liệu bản đồ cho vị trí này đến Google server.</li> <li>3. Ứng dụng nhận dữ liệu bản đồ trả về từ Google server rồi hiển thị đánh dấu và thông tin của vị trí trên bản đồ.</li> <li>4. Ứng dụng hiển thị gợi ý tên các bến ứng với chuỗi người dùng nhập vào.</li> <li>5. Khi người dùng di chuyển, ứng dụng lặp lại các bước 3, 4.</li> </ol>		
<b>Luồng sự kiện phát sinh/ Kịch bản phát sinh</b>		
<b>1. Ngoại lệ chưa bật chức năng GPS</b>		
<ol style="list-style-type: none"> <li>1.1. Ứng dụng thông báo “Bạn có muốn bật GPS?”.</li> <li>1.2. Người dùng chọn “Có”.</li> <li>1.3. Chuyển đến mục bật chức năng GPS của thiết bị.</li> </ol>		
<b>2. Ngoại lệ không tìm được địa chỉ của vị trí hiện tại</b>		
<ol style="list-style-type: none"> <li>2.1. Ứng dụng hiển thị vị trí hiện tại nhưng không kèm thông tin địa chỉ.</li> </ol>		

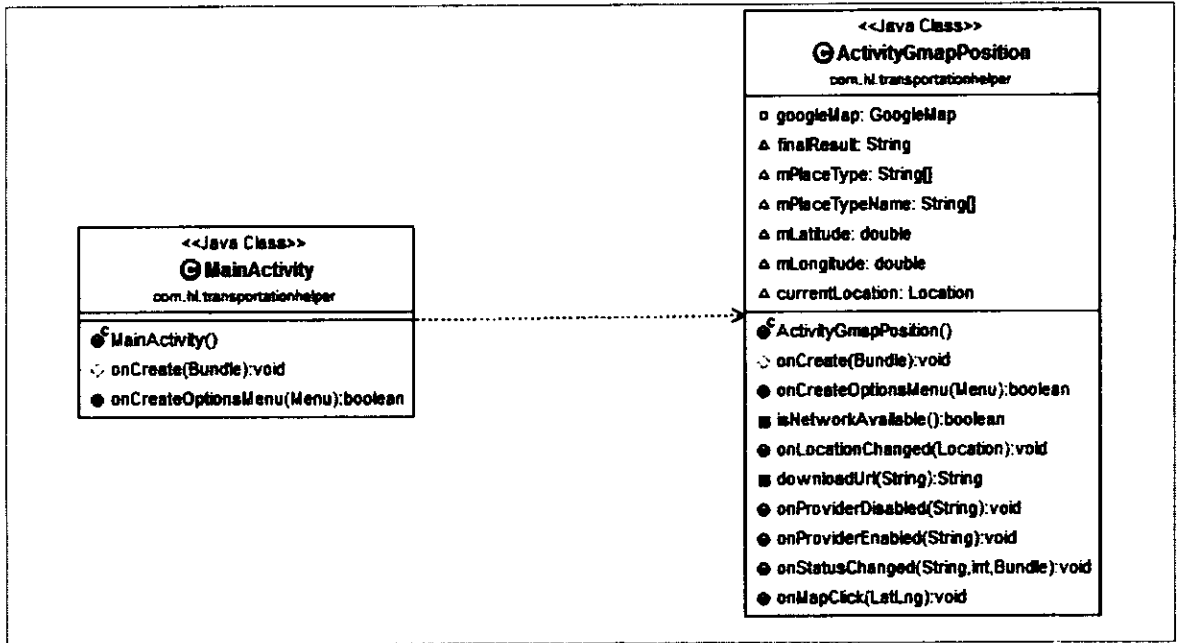
Giao diện minh họa



Sơ đồ trình tự (Sequence diagram)



Sơ đồ lớp chi tiết (Class diagram)



**Chức năng Bến xe buýt**

<b>UC #005</b>		<b>Bến xe buýt</b>	<b>Độ phức tạp: cao</b>
<b>Mô tả</b>		Chức năng này sẽ đánh dấu vị trí các bến xe buýt gắn với vị trí hiện tại của người dùng kèm theo địa chỉ của các bến đó trên bản đồ số (Google Map).	
<b>Tác nhân</b>	<b>Chính</b>	Người dùng	
	<b>Phụ</b>	Không có	
<b>Tiền điều kiện</b>		Không có	
<b>Hậu điều kiện</b>	<b>Thành công</b>	Hiển thị đánh dấu các bến xe buýt gắn với vị trí hiện tại của người dùng trên bản đồ số (Google Map) và địa chỉ của các bến đó.	
	<b>Lỗi</b>	Thông báo lỗi.	
<b>ĐẶC TẢ CHỨC NĂNG</b>			
<b>Luồng sự kiện chính/Kịch bản chính</b>			
User case này bắt đầu khi người dùng muốn tìm các bến xe buýt quanh vị trí hiện tại.			
<ol style="list-style-type: none"> <li>1. Người dùng chọn nút <b>Bến xe buýt</b> trong chức năng <b>Định vị</b>.</li> <li>2. Ứng dụng tự động lấy dữ liệu vị trí hiện tại thông qua chức năng GPS sau</li> </ol>			

đó, gửi yêu cầu về dữ liệu bản đồ, thông tin về các bến xe buýt gần vị trí này đến Google server.

3. Ứng dụng nhận dữ liệu trả về từ Google server rồi hiển thị đánh dấu và thông tin của các bến xe buýt gần vị trí hiện tại trên bản đồ.

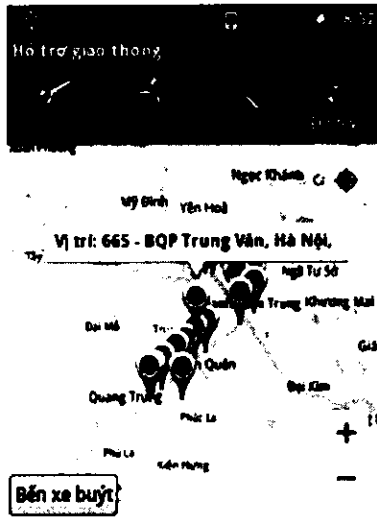
**Luồng sự kiện phát sinh/ Kịch bản phát sinh****1. Ngoại lệ chưa bật chức năng GPS**

- 1.1. Ứng dụng thông báo “Bạn có muốn bật GPS?”.
- 1.2. Người dùng chọn “Có”.
- 1.3. Chuyển đến mục bật chức năng GPS của thiết bị.

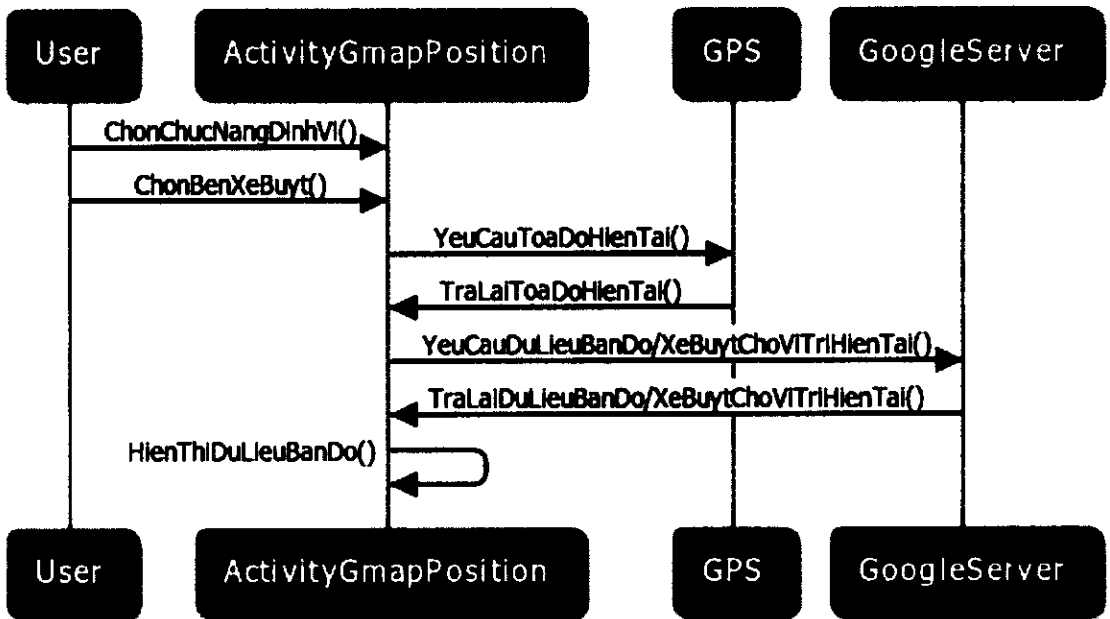
**2. Ngoại lệ không tìm được địa chỉ của vị trí hiện tại**

- 2.1. Ứng dụng hiển thị vị trí hiện tại nhưng không kèm thông tin địa chỉ.

Giao diện minh họa

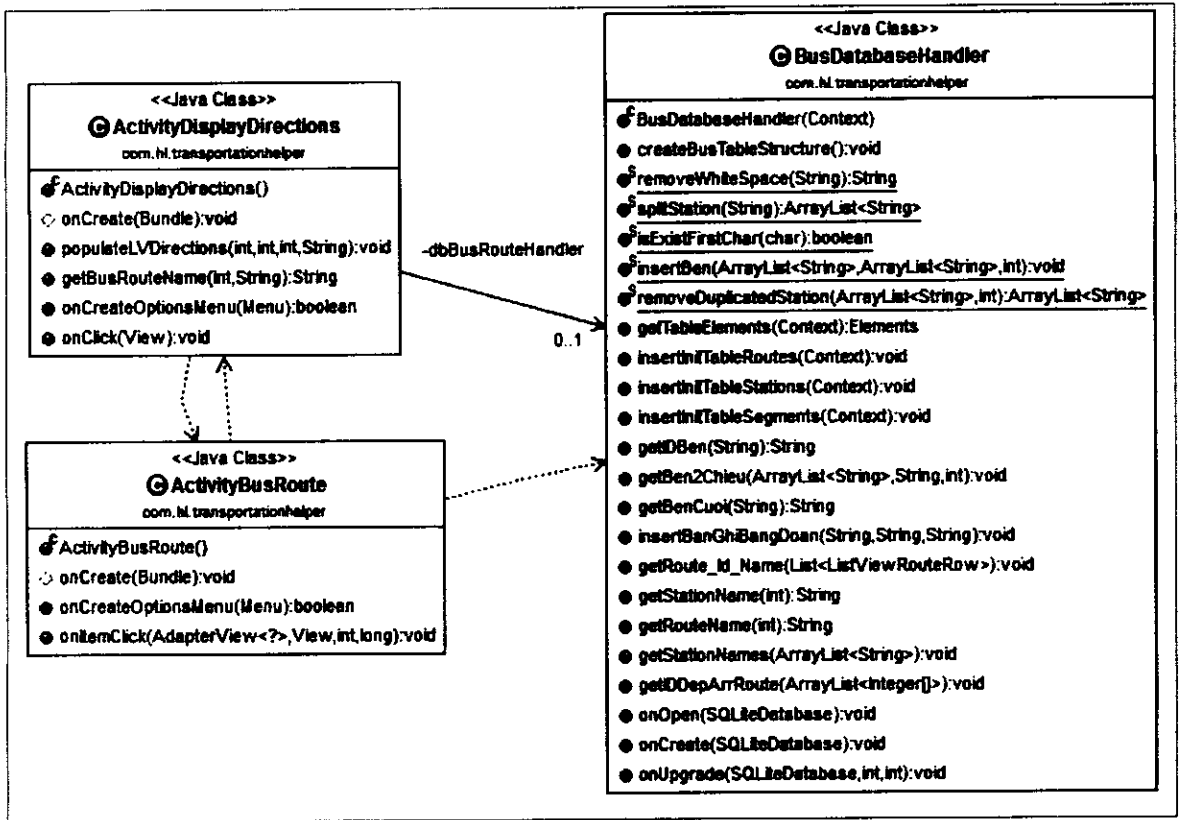


Sơ đồ trình tự (Sequence diagram)



Sơ đồ lớp chi tiết (Class diagram)





**CHƯƠNG 3. THIẾT KẾ VÀ TRIỂN KHAI**

Như đã phân tích ở các chương trước việc lựa chọn phát triển ứng dụng chạy trên hệ điều hành Android đạt được nhiều ưu thế hơn so với các nền tảng khác, nhằm tìm hiểu chi tiết hơn về các ưu thế này, chương này sẽ phân tích sơ lược về kiến trúc của hệ điều hành Android và kiến trúc thành phần của ứng dụng.

**3.1. Hệ điều hành Android**

**Giới thiệu:**

Android là hệ điều hành dành cho các thiết bị di động dựa trên nền tảng Linux. Ban đầu, Android được phát triển bởi Android, Inc. Năm 2005 Google mua lại công ty này.

Google cùng các đối tác gồm những nhà mạng viễn thông, các hãng sản xuất thiết bị di động đã cùng sáng lập Open Handset Alliance (tạm dịch: "Liên minh Thiết bị cầm tay Mở") và phát triển Android.

Tháng 11-2007, Google chính thức giới thiệu phiên bản đầu tiên đến với thị trường điện thoại di động thông minh.

Lịch sử phát triển của Android được tóm tắt trong bảng sau:

1.0	23 tháng Chín 2008	
1.1	9 tháng Hai 2009	
1.5	30 tháng Tư 2009	Cupcake
1.6	15 tháng Chín 2009	Donut
2.0/2.1	26 tháng Mười 2009	Eclair
2.2	20 tháng Năm 2010	Froyo
2.3	6 tháng Mười hai 2010	Gingerbread
3.0	22 tháng Hai 2011	Honeycomb
4.0.1	19 tháng Mười 2011	Ice Cream Sandwich
4.1	27 tháng Sáu 2012	Jelly Bean

Bảng 3.1. Lịch sử phát triển hệ điều hành Android

Android 1.0 và 1.1 không được đặt tên mã. Phiên bản 1.5 có tên Cupcake và theo đó, mỗi phiên bản đều mang một tên gọi "dễ thương" đi kèm, như Android 2.1 Eclair hay 2.2 Froyo, 2.3 Gingerbread ...

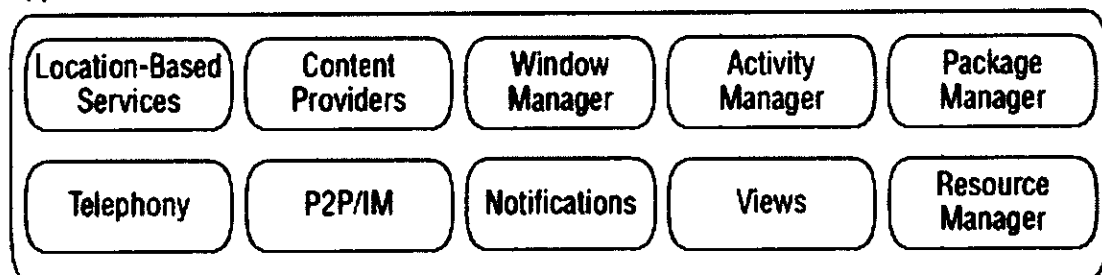
**Kiến trúc hệ điều hành Android:**

Các thành phần cơ bản của Android được mô tả ở hình sau:

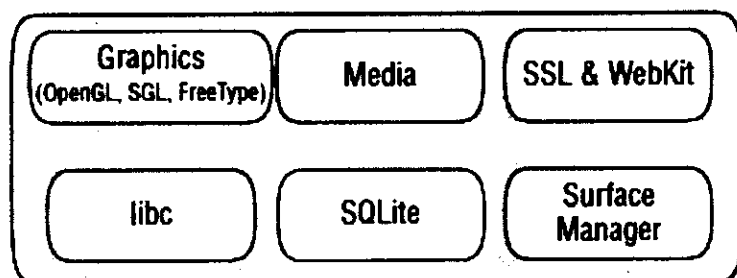
**Application Layer**



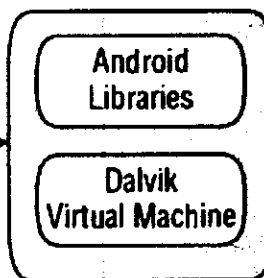
**Application Framework**



**Libraries**



**Android Runtime**



**Linux Kernel**



Hình 3.1. Kiến trúc hệ điều hành Android

- + Linux kernel – các dịch vụ cơ bản được điều khiển bởi nhân Linux 2.6 (trình điều khiển phần cứng, quản lý tiến trình và bộ nhớ, quản lý an ninh, quản lý mạng và nguồn điện).
- + Libraries – các thư viện C/C++ cơ bản như libc và SSL, bên cạnh đó là:
  - Thư viện cho xử lý nhạc và video
  - Thành phần hỗ trợ hiển thị

- Các thư viện SGL và OpenGL cho đồ họa 2D và 3D
- SQLite cho cơ sở dữ liệu
- SSL và WebKit cho trình duyệt web tích hợp và an ninh mạng
- + Android runtime – gồm các thư viện cơ bản giúp lập trình cho Android bằng ngôn ngữ Java và máy ảo Dalvik, giúp lập trình viên chạy thử các ứng dụng.
- + Application Framework – cung cấp các lớp để tạo nên các ứng dụng Android.
- + Application Layer – các ứng dụng sẽ hoạt động ở lớp này sử dụng các lớp và dịch vụ được hỗ trợ ở tầng application framework.

### 3.2. Chu trình sống của một ứng dụng

#### Activity là gì?

Một Activity là một thành phần của ứng dụng tương tự như một màn hình cho phép người dùng thực hiện một yêu cầu nào đó, như gọi điện, chụp ảnh, gửi email, hay xem bản đồ...

Một ứng dụng thường bao gồm nhiều activity được liên kết với nhau. Khi ứng dụng được khởi động, một activity có thể gọi đến các activity khác để thực hiện nhiều hành động khác nhau. Mỗi khi activity mới được gọi thì activity trước đó sẽ dừng lại, nhưng hệ thống vẫn lưu nó lại trong một ngăn xếp gọi là "back stack". Khi một activity mới khởi động nó cũng được đưa vào back stack. Cấu trúc này có cơ chế hoạt động giống như ngăn xếp, "vào sau, ra trước". Theo đó, khi người dùng làm việc xong với activity hiện tại và bấm nút Back, nó sẽ bị đẩy khỏi ngăn xếp (đồng thời bị hủy) và các activity trước nó sẽ được khôi phục lại.

#### Chu trình sống của một activity (Activity lifecycle)

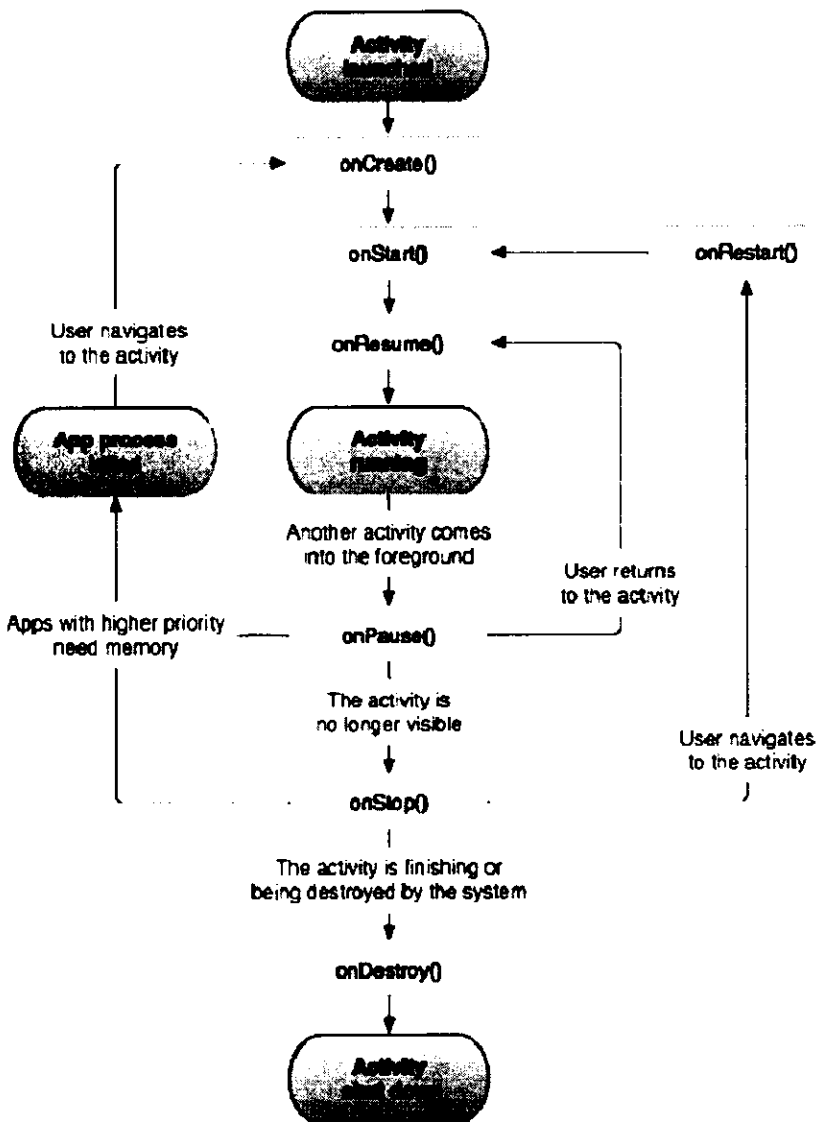
Chuỗi các "hành động" một activity sẽ thực hiện từ lúc nó được bắt đầu đến khi kết thúc được gọi là chu trình sống. Một activity được khởi động, nó sẽ được đặt ở đầu ngăn xếp và trở thành activity đang chạy (running activity), activity trước nó sẽ nằm phía dưới trong ngăn xếp, và sẽ không được gọi một khi activity trên nó vẫn còn tồn tại.

Một activity sẽ có bốn trạng thái sau:

- + Nếu một activity hiện đang hoạt động (ở đầu ngăn xếp), nó được gọi là đang hoạt động hoặc đang chạy (active hay running).
- + Nếu một activity không thể tương tác được nhưng vẫn hiện, nó được chuyển về trạng thái tạm dừng (pause), nhưng trong trường hợp bộ nhớ cạn kiệt, hệ thống sẽ kết thúc activity này.

- + Nếu một activity bị che hoàn toàn bởi một activity khác, nó chuyển về trạng thái dừng (stop). Lúc này nó có thể bị kết thúc bởi hệ thống nếu có một ứng dụng nào đó cần bộ nhớ.
- + Nếu một activity ở trạng thái dừng hoặc tạm dừng, hệ thống sẽ hỏi hoặc tự động kết thúc tiến trình mà activity này nắm giữ để xóa nó khỏi bộ nhớ. Khi hiển thị lại cho người dùng thì activity này được khởi động và khôi phục lại trạng thái của nó trước.

Sơ đồ dưới đây mô tả chu trình sống của một activity:



Hình 3.2. Chu trình sống của một Activity

Các phương thức được gọi trong chu trình sống của một activity:

- + onCreate ()

Được gọi khi activity được khởi tạo. Các thành phần giao diện và khởi tạo dữ liệu được thực hiện ở đây. Phương thức này cũng đi kèm với một đối tượng lưu trữ trạng thái trước đó của activity.

+ onRestart ()

Được gọi sau khi activity dừng.

+ onStart ()

Được gọi khi activity hiển thị với người dùng.

+ onResume ()

Được gọi khi activity bắt đầu tương tác với người dùng.

+ onPause ()

Được gọi khi hệ thống muốn khôi phục trạng thái của activity trước đó. Phương thức này thường dùng để lưu lại dữ liệu, hay dừng bất cứ hoạt động nào tiêu tốn CPU.

+ onStop ()

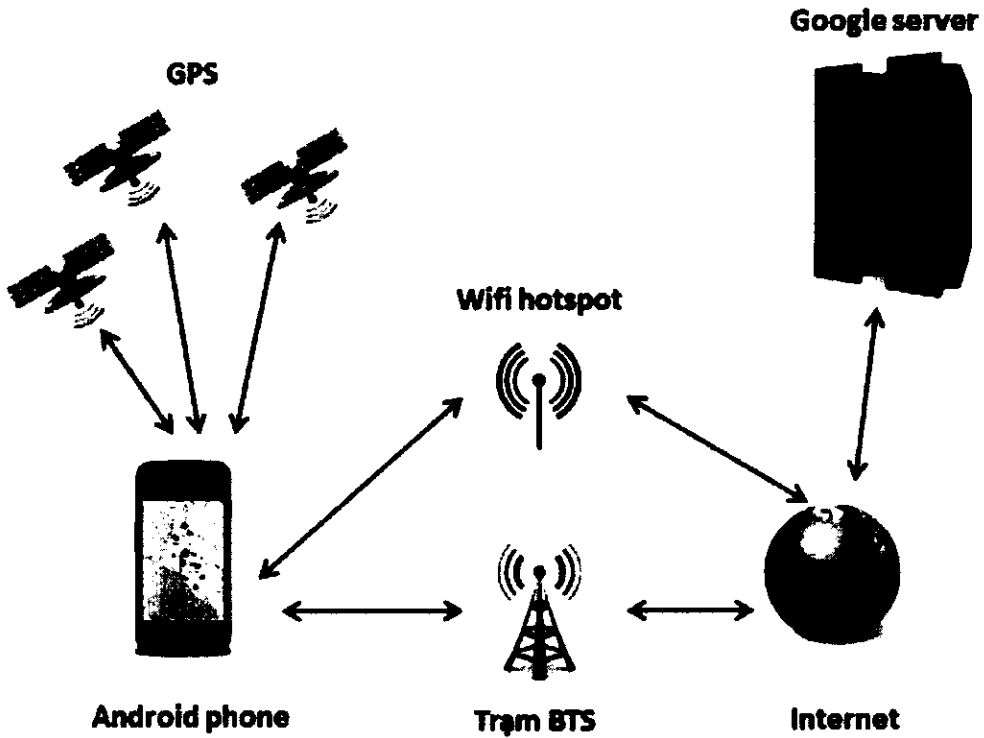
Được gọi khi activity không còn thấy được, vì một activity khác đã được khôi phục và che phủ lên activity này.

+ onDestroy ()

Được gọi khi kết thúc activity hay hệ thống tạm thời hủy nó để lấy không gian bộ nhớ.

### **3.3. Kiến trúc mạng**

Chức năng “Định vị” của ứng dụng có kết nối với mạng Internet để lấy thông tin, hình dưới đây mô tả cách tính năng này làm việc với mạng:

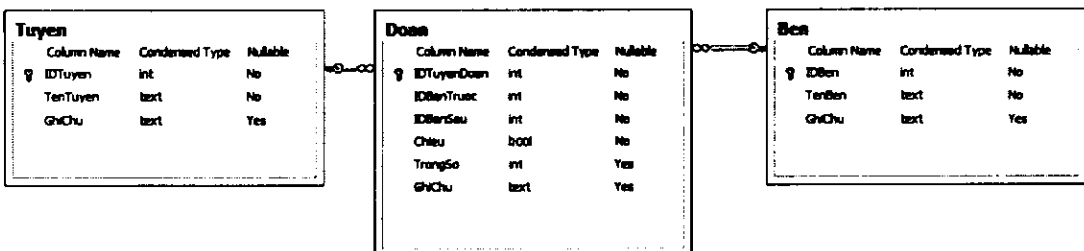


Hình 3.3. Kiến trúc mạng

Điện thoại thông qua chức năng kết nối GPS (Global Positioning System) sẽ lấy thông tin về tọa độ của vị trí hiện tại gồm có longitude (kinh độ) và latitude (vĩ độ). Sau đó điện thoại sẽ kết nối với mạng Internet thông qua Wifi hotspot hoặc trạm viễn thông (BTS) rồi gửi dữ liệu tới Google server, thông tin trả về là các Map Fragment chứa dữ liệu hình ảnh về vị trí hiện tại trên bản đồ Google map.

### 3.4. Mô hình hóa dữ liệu

#### Sơ đồ cơ sở dữ liệu



Hình 3.4. Sơ đồ cơ sở dữ liệu

**Cấu trúc các bảng dữ liệu**

Bảng Tuyen

1. Số hiệu: 1		2. Tên bảng: Tuyen		3. Bí danh	
4. Mô tả: Lưu thông tin của các tuyến buýt					
5. Mô tả chi tiết các cột					
Số thứ tự	Tên cột	Mô tả	Kiểu dữ liệu	Null	
1	IDTuyen	Trường khóa	INTEGER	NOT NULL	
2	TenTuyen	Tên của tuyến	TEXT	NOT NULL	
3	GhiChu	Ghi chú cho mỗi tuyến	TEXT	ALLOW NULL	

Bảng 3. 2. Bảng dữ liệu tuyến xe buýt

Bảng Doan

1. Số hiệu: 2		2. Tên bảng: Doan		3. Bí danh	
4. Mô tả: Lưu thông tin về các đoạn của một tuyến					
5. Mô tả chi tiết các cột					
Số thứ tự	Tên cột	Mô tả	Kiểu dữ liệu	Null	
1	IDTuyenDoan	Trường khóa	INTEGER	NOT NULL	
2	IDBenTruoc	Khóa của bảng Bến	INTEGER	NOT NULL	
3	IDBenSau	Khóa của bảng Bến	INTEGER	NOT NULL	
4	Chieu	Chỉ định chiều đi hoặc về	BOOL	NOT NULL	
5	TrongSo	Khoảng cách giữa 2 bến	INTEGER	ALLOW NULL	
6	GhiChu	Ghi chú cho mỗi đoạn	TEXT	ALLOW NULL	

Bảng 3.3. Bảng dữ liệu các đoạn của tuyến xe buýt

Bảng Ben

1. Số hiệu: 3		2. Tên bảng: Ben		3. Bí danh	
4. Mô tả: Lưu thông tin của các bến					



5. Mô tả chi tiết các cột				
Số thứ tự	Tên cột	Mô tả	Kiểu dữ liệu	Null
1	IDBen	Trường khóa	INTEGER	NOT NULL
2	TenBen	Tên của bến	TEXT	NOT NULL
3	GhiChu	Ghi chú cho mỗi bến	TEXT	NOT NULL

**Bảng 3.4. Bảng dữ liệu các bến xe buýt**

### 3.5. Triển khai

Để xây dựng được một ứng dụng Android thì trước hết ta phải hiểu về cấu trúc của một project Android, cách nó sắp xếp và quản lý file thế nào. Một ứng dụng gồm những phần gì và nó hoạt động ra sao và cách ứng dụng để xây dựng đề tài. Chương này sẽ trình bày khá chi tiết các nội dung kể trên.

### Cấu trúc của một project Android

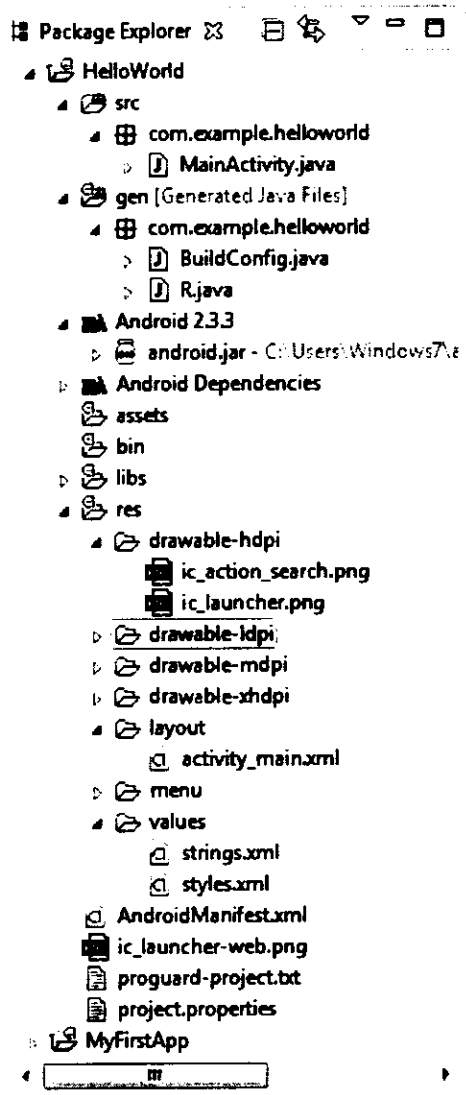
Các thành phần cơ bản của một project Android bao gồm:

- src – chứa file mã nguồn có đuôi \*.java. Ở ví dụ này chỉ có 1 file là, MainActivity.java. MainActivity.java là file mã nguồn cho activity này.
- Android 2.3.3 – thành phần này chỉ chứa 1 file, android.jar, bao gồm các thư viện cần thiết cho một ứng dụng.
- gen – chứa file R.java, file này được trình biên dịch tự động sinh ra, chỉ ra liên kết đến các tài nguyên có trong project (hình ảnh, các chuỗi, nút bấm...) Không nên thay đổi file này.
- assets – chứa các tài nguyên mà ứng dụng cần để hoạt động, như file HTML, file văn bản, cơ sở dữ liệu...
- res -chứa các tài nguyên bên ngoài như: hình ảnh, các file dữ liệu... được ứng dụng Android sử dụng.

AndroidManifest.xml -chứa thông tin về các activity, view, dịch vụ... Nó cũng liệt kê ra các quyền cần thiết để một ứng dụng Android hoạt động.

### Cấu trúc của file AndroidManifest.xml

```
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
package="com.example.helloworld"
android:versionCode="1"
android:versionName="1.0" >
<uses-sdk
android:minSdkVersion="8"
```



Hình 3. 5. Cấu trúc một project Android

```

android:targetSdkVersion="15" />

<application
    android:icon="@drawable/ic_launcher"
    android:label="@string/app_name"
    android:theme="@style/AppTheme" >
    <activity
        android:name=".MainActivity"
        android:label="@string/title_activity_main" >
        <intent-filter>
            <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER" />
        </intent-filter>
    </activity>
</application>
</manifest>

```

File AndroidManifest.xml chứa thông tin chi tiết về ứng dụng:

- Chỉ ra tên package của ứng dụng, ở ví dụ này là “com.example.helloworld”
- android:versionCode: chỉ ra phiên bản của ứng dụng. Giá trị này cần thiết khi nâng cấp ứng dụng.
- android:versionName: giá trị ở dòng này sẽ dùng để hiển thị phiên bản cho người dùng.
- android:minSdkVersion: chỉ ra phiên bản nhỏ nhất mà ứng dụng sẽ hoạt động trên đó.
- android:targetSdkVersion: chỉ ra phiên bản cao nhất mà ứng dụng có thể hoạt động trên đó.
- android:icon: ứng dụng sẽ sử dụng ic\_launcher.png nằm trong thư mục drawable.
- android:label: tên của ứng dụng là giá trị app\_name được định nghĩa trong file strings.xml.
- Chỉ có một activity đại diện bởi file MainActivity.java.
  - Phần tử<intent-filter>: đăng kí các activity, dịch vụ và Broadcast Receiver để thao tác trên bộ dữ liệu.

**File R.java**

Nội dung đại diện của một file R.java:

```
package com.example.helloworld;

public final class R {
    public static final class attr {
    }

    public static final class drawable {
        public static final int ic_action_search=0x7f020000;
        public static final int ic_launcher=0x7f020001;
    }

    public static final class id {
        public static final int menu_settings=0x7f070000;
    }

    public static final class layout {
        public static final int activity_main=0x7f030000;
    }

    public static final class menu {
        public static final int activity_main=0x7f060000;
    }

    public static final class string {
        public static final int app_name=0x7f040000;
        public static final int hello_world=0x7f040001;
        public static final int menu_settings=0x7f040002;
        public static final int title_activity_main=0x7f040003;
    }

    public static final class style {
        public static final int AppTheme=0x7f050000;
    }
}
```

File R.java được trình biên dịch tự động sinh ra, file này được sử dụng để quản lý các thuộc tính được khai báo trong file XML của ứng dụng. Mã nguồn của file R.java được tự động sinh khi có bất kỳ một sự kiện nào xảy ra làm thay đổi các thuộc tính trong ứng dụng. Ví dụ như, thêm file hình ảnh từ bên ngoài vào project thì đường dẫn đến file

đó cũng có trong file R.java hoặc xoá một file hình ảnh thì đường dẫn tương ứng đến hình ảnh đó cũng tự động bị xoá.

### **3.6. Xử lý dữ liệu đầu vào**

Để có được thông tin phục vụ cho việc xây dựng cơ sở dữ liệu một cách chính xác và tự động, thì trước hết, thông tin đầu vào cần phải được xử lý để loại bỏ các yếu tố dư thừa đồng thời phải được tổ chức sao cho việc lấy dữ liệu đạt được hiệu quả cao nhất.

#### **Xử lý thông tin giờ tàu**

Vì dữ liệu tại địa chỉ <http://www.vr.com.vn/gio-tau.html> được bảo vệ bằng SSL (Secure Socket Layer) nên không thể lấy về tự động do đó các trang thông tin này đều phải lưu lại thủ công để sử dụng ở dạng offline. Tức là có 10 tuyến bao gồm:

"Hà Nội - Sài Gòn"

"Sài Gòn - Hà Nội"

"Hà Nội - Lào Cai"

"Lào Cai - Hà Nội"

"Hà Nội - Hải Phòng"

"Hải Phòng - Hà Nội"

"Hà Nội - Đồng Đăng"

"Đồng Đăng - Hà Nội"

"Hà Nội - Quán Triều"

"Quán Triều - Hà Nội"

sẽ có tương ứng 10 file HTML

"gt\_Hanoi\_Saigon.html"

"gt\_Saigon\_Hanoi.html"

"gt\_Hanoi\_Laocai.html"

"gt\_Laocai\_Hanoi.html"

"gt\_Hanoi\_Haiphong.html"

"gt\_Haiphong\_Hanoi.html"

"gt\_Hanoi\_Dongdang.html"

"gt\_Dongdang\_Hanoi.html"

"gt\_Hanoi\_Quantrieu.html"

"gt\_Quantrieu\_Hanoi.html"

Tuyến	Sài Gòn - Hà Nội	Mã tàu	- Tất cả -
Ga đi	Sài Gòn	Ga đến	Hà Nội

TÊN GA \ MÃC TAG	SE2	SE4	SE6	SE8	SE10	SE12	SE14	SE16	TN
Sài Gòn	19.00	23.00	15.45	6.25	22.10	13.25	8.35	10.55	14.00
Biển Hòa	19.39 - 19.42		16.24 - 16.27	7.04 - 7.07					
Long Khánh									
Tháp Chàm			21.20 - 21.26						

```

<div id="inner" class="box">
  <table id="grv" class="tbl_search" width="100%" cellpadding="3" cellspacing="3" border="0" style="border-width: 0px;">
    <tbody>
      <tr class="title" valign="middle">
        <th width="150" scope="col">TÊN GA \ MÃC TAG</th>
        <th scope="col">SE2</th>
        <th scope="col">SE4</th>
        <th scope="col">SE6</th>
        <th scope="col">SE8</th>
        <th scope="col">SE10</th>
        <th scope="col">SE12</th>
        <th scope="col">SE14</th>
        <th scope="col">SE16</th>
        <th scope="col">TN</th>
      </tr>
      <tr class="oddrow">
        <td style="text-align: left; font-weight: bold;">Sài Gòn</td>

```

Hình 3.6. Minh họa cấu trúc file đầu vào

Các file HTML này sẽ được lưu tại mục "assets" của project để phục vụ cho việc lấy dữ liệu. Sau đó ta sẽ sử dụng một thư viện Java tên là "Jsoup" để đọc các file HTML theo cấu trúc DOM nhằm tìm ra thẻ chứa dữ liệu mong muốn. Ví dụ, ta muốn lấy ra tập hợp các dòng của bảng trong hình minh họa trên có thể dùng đoạn code sau:

```

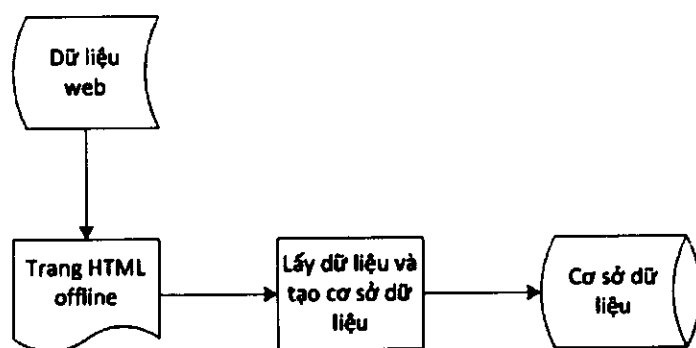
Document doc = Jsoup.parse (outStream.toString ());
Element tblContent = doc.getElementById ("grv");
Elements tblRows = tblContent.getElementsByTag ("tr");

```

Với Document, Element, Elements là các đối tượng được định nghĩa trong thư viện Jsoup, outputStream là file HTML được đọc vào thành luồng dữ liệu. Sau khi lấy được dữ liệu cần thiết thì việc tiếp theo là tạo cơ sở dữ liệu và insert dữ liệu này vào các trường. Để thực hiện các việc này lớp TrainDatabaseHandler cung cấp 2 phương thức createDBTrainSchedule (Context c, String[] arrTableName) để tạo cơ sở dữ liệu và createTableTrainSchedule (String tbName, String link, AssetManager am) có đầu vào lần lượt là:

- tbName - tên bảng cần tạo
- link - tên file HTML tại thư mục assets
- am - đối tượng quản lý assets, cho kết quả là một bảng dữ liệu.

Toàn bộ quá trình trên được mô tả bằng hình sau:



Hình 3.7. Quy trình xử lý thông tin giờ tàu

- Hàm `createDBTrainSchedule(Context c, String[] arrTableName)`:

Như hình 3.6, dữ liệu mà ta cần là mã tàu, chính là các chuỗi SE2, SE4... nằm trong thẻ "th". Hàm này sẽ lấy ra các chuỗi sau đó kết hợp để tạo ra một câu lệnh tạo bảng theo đúng cú pháp SQLite rồi thực thi câu lệnh này. Sau khi cấu trúc bảng dữ liệu của một tuyến được tạo ra như trên, hàm này tiếp tục lấy các chuỗi tiếp theo nằm trong thẻ "td" bao gồm tên ga tàu, giờ tàu xuất/vào bến. Cuối cùng nó kết hợp các chuỗi lấy được rồi thực thi câu lệnh insert dữ liệu vào bảng.

- Hàm `createTableTrainSchedule (String tbName,String link, AssetManager am)`:

Hàm này sẽ được gọi trong activity `ActivityTrainSchedule` để tạo ra toàn bộ cơ sở dữ liệu giờ tàu. Khi hoạt động, nó sẽ gọi tới hàm `createDBTrainSchedule` nói trên để tạo ra bảng dữ liệu ứng với mỗi tuyến. Hai thông tin quan trọng cần cho hàm này là:

- + `am`: một đối tượng `AssetManager` quản lý tài nguyên mà chương trình cần sử dụng.
- + `link`: đường dẫn đến file html lưu tại thư mục "assets" do `AssetManager` quản lý.

Đối tượng `AssetManager` căn cứ theo đường dẫn được chỉ ra ở tham số "link" sẽ tìm đến đúng file html có tên tương ứng để hỗ trợ cho hàm thao tác trên các file này.

### Xử lý thông tin tuyến buýt

Bước đầu việc lấy thông tin xe buýt cũng được thực hiện tương tự như phần trước, tức là trang web được lưu trữ offline và lưu tại thư mục "assets" của project.

So với dữ liệu giờ tàu giờ tàu thì việc lấy dữ liệu về tuyến buýt gặp nhiều khó khăn hơn. Khó khăn chính ở đây là dữ liệu có định dạng không thống nhất và được mô tả ở dưới đây:

1. Vì ta cần lấy ra tên bến duy nhất để lưu vào cơ sở dữ liệu, nên các bến giống nhau nhưng có kèm thông tin phía sau sẽ trở thành “nhiều”, tức là chúng sẽ được coi là các bến khác nhau trong khi đó vẫn chỉ là một bến. Ví dụ:

ngoài bến Long Biên còn có Long Biên (Yên Phụ - Khoang 2), Long Biên (Yên Phụ - Khoang 1). Ta chỉ cần lấy chuỗi “Long Biên” để chỉ tên bến, nhưng những chuỗi còn lại cũng chỉ bến Long Biên nhưng sẽ trở thành các bến khác nhau nếu chúng được lưu vào cơ sở dữ liệu. Hơn nữa, ta dùng dấu “-” (gạch ngang) để xác định kí tự phân tách giữa các chuỗi nên khi tách ra ví dụ với chuỗi “Long Biên (Yên Phụ - Khoang 2)”, ta sẽ được 2 chuỗi “Long Biên (Yên Phụ” và “ Khoang 2)”. Hai chuỗi này trở thành nhiều.

2. Tương tự như vậy, chuỗi “Quay đầu tại...” cũng trở thành nhiều vì chuỗi này nằm giữa hai dấu “-” (gạch ngang), kí hiệu để nhận diện và tách chuỗi. Ví dụ:

Xã Đàn - Quay đầu tại đối diện ngõ Xã Đàn 2 - Xã Đàn - Khâm Thiên  
 Khi tách chuỗi, ta sẽ dùng dấu “-” (gạch ngang) để nhận biết kí tự phân cách chuỗi. Nếu không loại bỏ chuỗi “Quay đầu tại...” thì kết quả sau khi tách chuỗi sẽ là, Xã Đàn, Quay đầu tại đối diện ngõ Xã Đàn 2, Xã Đàn, Khâm Thiên. Khi đó chuỗi “Quay đầu tại đối diện ngõ Xã Đàn 2” sẽ được coi là một bến, nhưng rõ ràng là không tồn tại bến này.

3. Riêng với bến Yên Phụ, còn xuất hiện giá trị Yên Phụ (Khoang 1), Yên Phụ (Khoang 2) trong khi vẫn chỉ là một bến. Vấn đề này tương tự như đã nêu ở mục 1 nhưng khác ở điểm là không xuất hiện dấu gạch ngang trong chuỗi cần tách.

4. Để ngăn cách giữa các chữ ngoài dấu cách kí tự Non-breaking space cũng được sử dụng. Vì dấu cách (mã ASCII hệ thập phân có giá trị: 32) và Non-breaking space (mã ASCII hệ thập phân có giá trị: 160) đều là kí tự trắng nên nếu không loại bỏ kí tự này khi lấy dữ liệu đầu vào sẽ dẫn đến việc so sánh sai khi ứng dụng hoạt động. Ví dụ khi insert tên bến Long Biên vào cơ sở dữ liệu mặc dù đã sử dụng hàm trim() để xóa kí tự trắng ở đầu và cuối chuỗi nhưng hàm này chỉ xóa kí tự space (dấu cách) chứ không xóa Non-breaking space do đó chuỗi Long Biên và [Non-breaking space]Long Biên được coi là hai chuỗi khác nhau dẫn tới sai sót cho kết quả trả về, tương tự như vậy, nếu chuỗi xuất hiện dấu chấm câu (“.”) cũng cần phải loại bỏ.

Giải pháp cho vấn đề này như sau, ta sẽ tạo ra các pattern để bắt các chuỗi gây nhiễu trên, sau đó dùng cặp Pattern- Matcher để xử lý loại bỏ. Các mẫu trên bao gồm:

1. Bắt các cụm “ (Yên Phụ - Khoang 1)”, “ (Yên Phụ - Khoang 2)”

```
public final static String patternYenPhuKhoang
    = "\\ (Yên Phụ - Khoang [1-2])\\)";
```

2. Bắt các cụm “Quay đầu tại...”



```
public final static String patternQuayDau
= "(\\-{1}\\s*+Quay đầu\\s*( ([\\p{L}])\\s*+[0-9]*)+\\s* ( ([\\p{L}])\\s*))";
```

3. Bắt các chuỗi “ (Khoang 1)”, “ (Khoang 2)”

```
public final static String patternKhoang
= "\\ (Khoang [1-2]\\)";
```

Riêng lỗi 4 được ghép với các mẫu khác tạo thành một mẫu hoàn chỉnh để bắt được tất cả các chuỗi đã nêu ở trên và dưới đây là mẫu tổng hợp:

```
public static Pattern ptnTongHop =
Pattern.compile (patternKhoang + "|"
+ patternQuayDau + "|" + patternYenPhuKhoang
+ "|" + "\\." + "|"
+ String.valueOf( (char) 160));
```

Nói thêm về Pattern, Matcher:

Pattern và Matcher là hai lớp thuộc package java.util.regex làm việc trên chuỗi (string). Lớp Pattern: một đối tượng pattern khi gọi lệnh compile sẽ biên dịch chuỗi biểu thức chính quy để tăng hiệu suất cho quá trình tìm kiếm.

Lớp Matcher: đối tượng matcher sẽ dịch pattern được biên dịch ở trên để thực hiện việc tìm kiếm chuỗi theo mẫu.

Sau khi tìm thấy các phân tử gây nhiễu, ta sẽ loại bỏ chúng bằng cách thay thế các phân tử đó bằng dấu cách. Sau bước này ta sẽ thu được một chuỗi đã loại bỏ nhiễu và chỉ còn chuỗi đại diện tên các bến và cách nhau bằng dấu “-” gạch ngang. Thông thường, ở đây ta sẽ gọi hàm String.split() hoặc Regex.Split() với tham số là dấu gạch ngang “-”, nhưng trên chương trình mô phỏng Android (Android emulator) hai hàm này gây ảnh hưởng lớn tới tốc độ thực thi của chương trình. Nguyên nhân chính là do mỗi lần gọi String.split() thì biểu thức chính quy đầu vào sẽ được biên dịch lại do đó làm chậm quá trình tìm kiếm chuỗi. Giải pháp cho vấn đề này là hàm dưới đây:

```
public static ArrayList<String>
splitStation (String strArrLColumn)
```

đầu vào của hàm là chuỗi các bến phân tách bằng dấu cách, đầu ra là một ArrayList kiểu string chứa tên bến đã tách. Hàm này làm việc như sau, với mỗi chuỗi đầu vào, nó sẽ kiểm tra từ vị trí đầu (begin) đến vị trí cuối (end) nếu xuất hiện kí tự “-” thì cắt lấy xâu trước dấu “-” rồi đưa vào ArrayList. Chuỗi sau dấu gạch ngang sẽ được gán thành

chuỗi hiện tại. Tốc độ thực thi hàm này sẽ càng lúc càng nhanh vì sau mỗi lần lặp, độ dài của chuỗi cần tìm lại giảm xuống.

Đến bước này, để chuẩn bị cho việc insert vào cơ sở dữ liệu, ta sẽ định nghĩa một lớp mới tên là LookUpTable

```
public class LookUpTable {
    private char element;
    private int begin;
    private int end;
    //Cấu tử và các phương thức set, get
    ...
}
```

Mỗi đối tượng thuộc lớp này sẽ có dữ liệu là

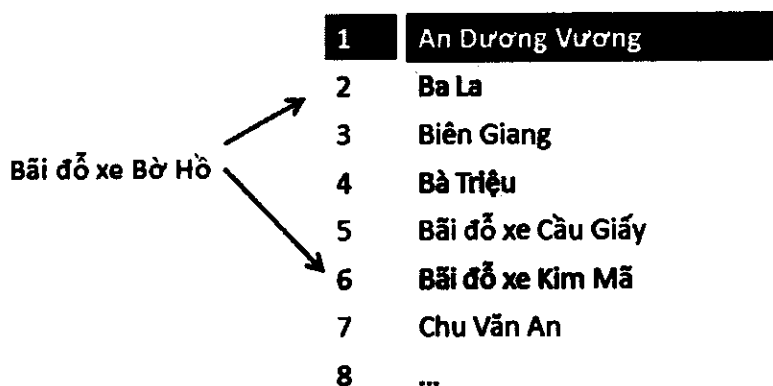
element: chữ cái đầu tiên của chuỗi

begin: vị trí đầu tiên xuất hiện chữ cái này trong ArrayList

end: vị trí cuối cùng xuất hiện chữ cái này trong ArrayList

Khi insert tên bến vào ArrayList ta chỉ cần so sánh chuỗi đó trong khoảng vị trí bắt đầu từ begin và kết thúc tại vị trí end của ArrayList. Nếu tên bến đã có trong ArrayList thì không cần lưu, ngược lại thì nó sẽ được lưu vào ArrayList .

Hình dưới đây sẽ minh họa tác dụng của lớp này:



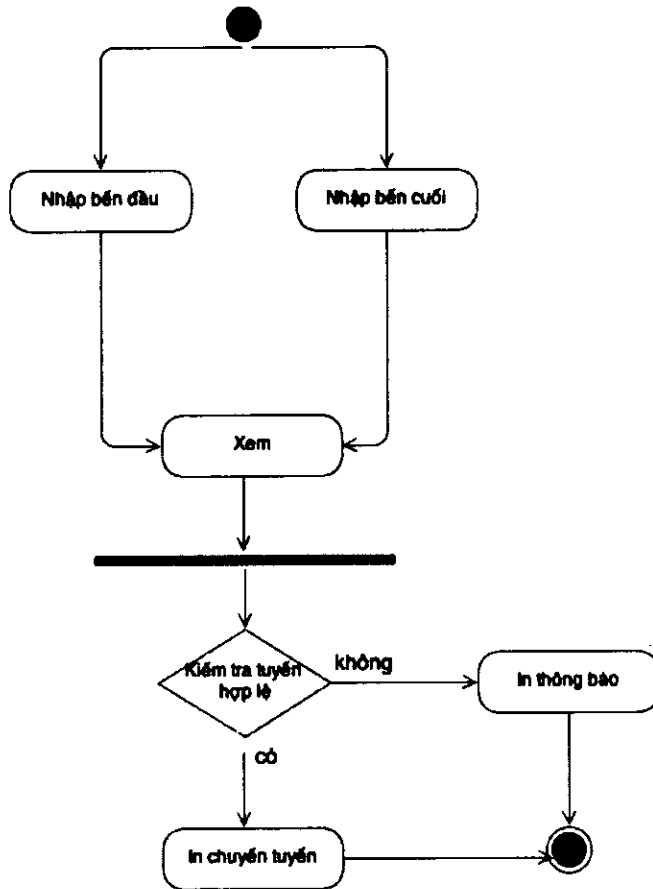
Hình 3.8. Minh họa LookUpTable

Giả sử ArrayList đã có các tên bến như trên, trong quá trình tách chuỗi ta lấy chuỗi “Bãi đỗ xe Bờ Hồ” đem so sánh với giá trị các bến này. Lúc này, đối tượng thuộc lớp LookUpTable có giá trị element = ‘B’ sẽ có các giá trị begin, end tương ứng là 2 và 6. Chuỗi “Bãi đỗ xe Bờ Hồ” sẽ được so sánh từ vị trí thứ 2 đến vị trí thứ 6 mà không cần kiểm tra đến hết ArrayList.

3.7. Cài đặt các chức năng chính

**Chức năng Chuyển tuyến**

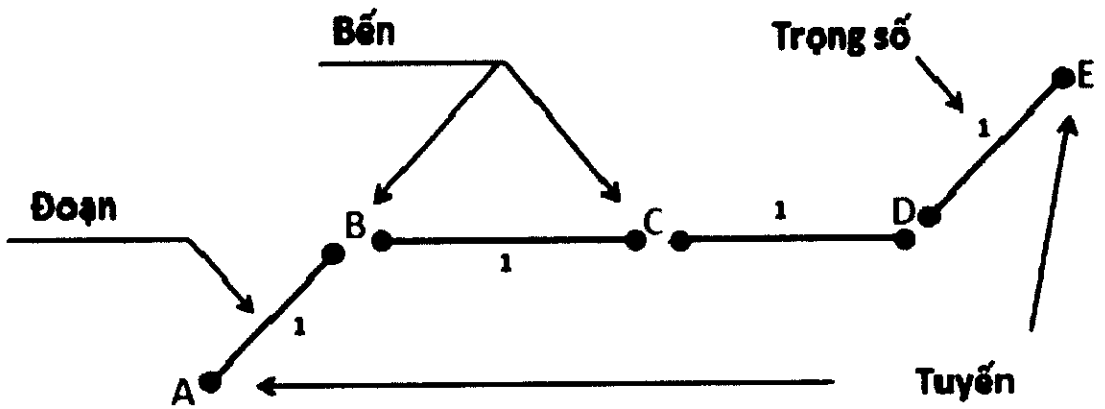
Sơ đồ hành động của chức năng này được minh họa dưới đây



Hình 3.9. Sơ đồ hành động chức năng chuyển tuyến

Khi người dùng chọn nhập bến đầu hoặc bến cuối, ứng dụng đều đưa ra gợi ý cho các chuỗi nhập vào nếu có chuỗi hợp lệ. Tính năng này có hỗ trợ tiếng Việt nên người dùng có thể gõ những chữ không dấu hoặc có dấu có trong tên bến sau đó dựa vào gợi ý để chọn ra bến theo yêu cầu. Ví dụ, nếu bến khởi hành là Bà Triệu, người dùng gõ “B” và chọn trong danh sách các bến có chữ “b” như Ba La, Bà Triệu,...

Khi người dùng bấm nút “Xem”, bến đi sẽ trở thành điểm đầu, thông qua thuật toán Dijkstra sẽ tìm được đường đi ngắn nhất tới điểm cuối tức bến đến, nếu tồn tại đường đi giữa 2 điểm này. Dữ liệu đầu vào cho thuật toán Dijkstra được mô hình hóa dưới dạng cơ sở dữ liệu có thể được mô tả tổng quát bằng hình minh họa dưới đây.



Hình 3.10. Minh họa mô hình hóa dữ liệu

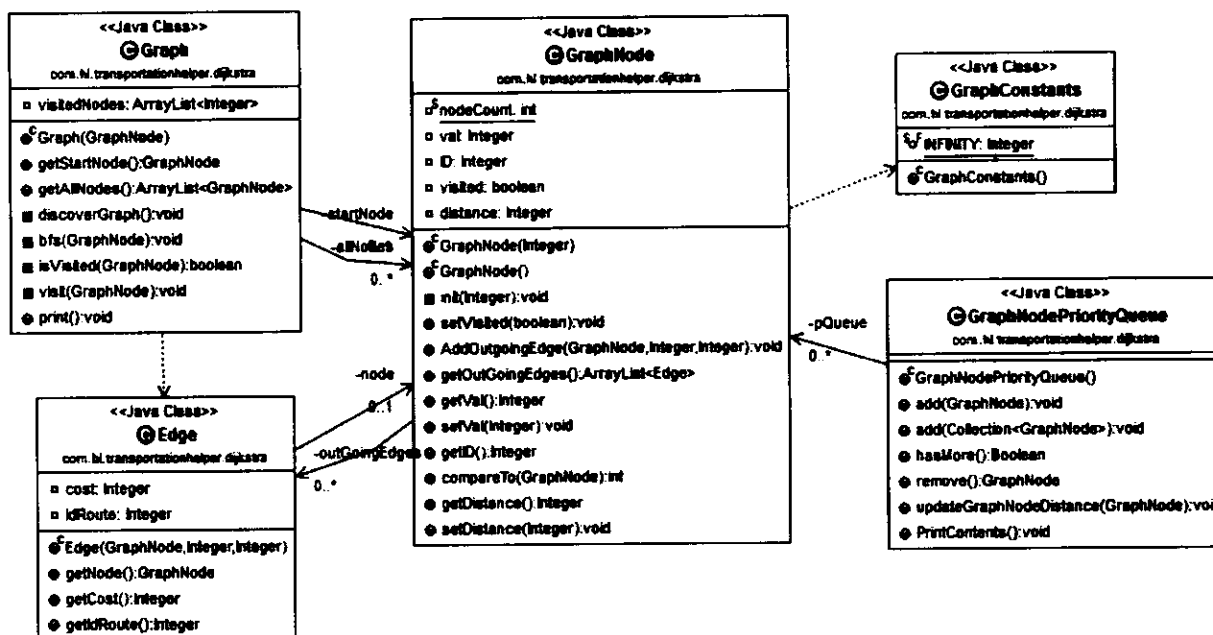
Ta coi các bến là các đỉnh của đồ thị, thông tin về bến được lưu tại bảng Bến. Hai bến tạo nên 1 đoạn tương ứng với cạnh của đồ thị, thông tin này được lưu tại bảng Đoạn, bao gồm giá trị khóa của bến trước, giá trị khóa của bến sau, trọng số, chiều... Ở đây, trọng số là khoảng cách giữa 2 bến, chiều đặc trưng cho hướng đi hay về của đoạn. Ví dụ, Tuyến A đến E có chiều đi là A – B – C – D – E thì thông tin về đoạn A – B trong cơ sở dữ liệu sẽ được lưu như sau (giả sử khóa của A là 1, B là 2, trọng số là 1):

```
IDBenTruoc: 1
IDBenSau: 2
Chiều: 0
TrongSo: 1
```

...

Khi có người dùng nhập vào bến đi và bến đến, dữ liệu về phần đồ thị gồm các đoạn kết nối giữa hai đỉnh này sẽ được lưu trữ trong cấu trúc Priority Queue (hàng đợi ưu tiên). Việc này nhằm chuẩn bị cho thuật toán Dijkstra hoạt động theo kiểu “Giảm khóa” (decrease key). Có nghĩa là, với mỗi đỉnh nằm ở đầu hàng đợi ưu tiên, sau khi cập nhật giá trị khoảng cách tới các đỉnh kề nó, nó sẽ bị đẩy ra khỏi hàng đợi.

Để tìm được trình tự chuyển tuyến, ta sẽ lưu vết đường đi và lấy ra được các đoạn nối giữa hai đỉnh đầu/cuối đảm bảo yêu cầu với số lần chuyển tuyến ít nhất. Cách thức làm việc chi tiết của chức năng này được trình bày cụ thể sau đây.



Hình 3.11. Các lớp hỗ trợ thuật toán Dijkstra

Để có thể áp dụng thuật toán Dijkstra vào bài toán ta cần xây dựng các lớp hỗ trợ sau:

- Graph: lưu trữ thông tin và các phương thức tương tác với danh sách các node thuộc cùng một đồ thị(allNodes), các node đã được duyệt qua(visitedNodes) và node khởi đầu của quá trình tìm đường đi ngắn nhất(startNode).
- GraphConstants: chứa hằng số khi khởi tạo đồ thị ở đây là INFINITY =999999, tức là tương đương với khoảng cách giữa 2 node là vô cùng trên lý thuyết.
- GraphNode: chứa toàn bộ thông tin của một node trong đồ thị bao gồm, tên, ID, tình trạng đã được duyệt hay chưa, danh sách các đỉnh kề. Ngoài ra lớp này còn chứa định nghĩa của lớp Edge(cạnh). Lớp này chứa thông tin về node, trọng số hay “giá”, và ID của một cạnh(tức là đoạn nối giữa hai bên).
- GraphNodePriorityQueue: chứa đồ thị để phục vụ việc tìm đường đi ngắn nhất giữa 2 node theo kiểu “Giảm khóa” (decrease key).

Sau khi có các lớp hỗ trợ, ta xây dựng lớp Dijkstra để kết hợp các lớp này để xử lý thông tin và đưa ra phương án chuyển tuyến ngắn nhất. Ta cần có một đối tượng có thể lưu lại vết đường đi của một node để có thể in ra các thông số cần thiết về tuyến đường(TraceList). Các giá trị này bao gồm ID của node hiện tại(currentNode), ID của node trước nó(prevNode), ID của cạnh nối hai node này(idRoute). Đến bước này, để tìm ra phương án chuyển tuyến ngắn nhất ta tiến hành theo trình tự sau:

1. Chạy thuật toán Dijkstra để tìm ra khoảng cách ngắn nhất giữa 2 bến đã chọn.
2. Chạy lại thuật toán Dijkstra rồi kiểm tra tuyến đường với khoảng cách đã tìm được, đồng thời lưu lại vết của mỗi tuyến.
3. Tìm ra phương án có số lần chuyển tuyến ngắn nhất trong danh sách được tìm ra ở bước 2.

Ở đây cần giải thích kỹ hơn bước 2, vì sao lại phải chạy lại thuật toán Dijkstra 1 lần nữa? Nguyên nhân là do có thể tồn tại nhiều phương án chuyển tuyến có khoảng cách ngắn nhất nhưng số lần chuyển tuyến lại không ngắn nhất. Ta sẽ thấy rõ hơn qua ví dụ sau: Giả sử ta muốn đi chuyển từ bến A đến bến F, giữa A và F có ba phương án chuyển tuyến ngắn nhất như sau

- + A - E - F
- + A - B - C - F
- + A - C - D - C - F

Mặc dù 3 lựa chọn này đều đạt tiêu chí có tổng quãng đường là ngắn nhất nhưng số lần chuyển tuyến thì lại khác nhau. Vì vậy ta cần phải chạy lại thuật toán một lần nữa để lưu lại tất cả các danh sách vết đường đi rồi sau đó kiểm tra để tìm ra được phương án có số lần chuyển bến là ít nhất.

### **Chức năng Định vị**

Chức năng này sẽ hiển thị vị trí hiện tại của người dùng thông qua GPS và Google Map đồng thời cho phép tìm kiếm các bến xe buýt gần vị trí hiện tại. Để làm được việc này, ta sẽ sử dụng Google Map API for Android và Google Place API. Đây là 2 API (Application Programming Interface – giao diện lập trình ứng dụng) mạnh mẽ và hỗ trợ rất tốt cho lập trình viên, trong đó Google Map API cung cấp thông tin về các loại bản đồ (giao thông, vệ tinh...) còn Google Place API cho thông tin về địa điểm (bến xe, quán ăn...). Vì phiên bản Google Map hiện tại là V2 (version 2) không cho phép chạy trên máy mô phỏng (Emulator) nên ta phải kiểm tra hoạt động trên thiết bị thật. Chức năng này hoạt động như sau: điện thoại thông qua tính năng GPS sẽ kết nối với vệ tinh GPS để lấy ra 2 giá trị là longitude – kinh độ và latitude – vĩ độ rồi kết nối với Internet, thông qua Google Map API sẽ gửi dữ liệu này đến máy chủ của Google, sau đó nó sẽ gửi lại dữ liệu là một MapFragment. Đây là một phần hình ảnh chứa thông tin về vị trí hiện tại của người sử dụng, đường sá khu vực xung quanh... Khi người dùng bấm vào nút “Bến xe buýt”, quá trình gửi và nhận dữ liệu cũng tương tự như trên để lấy thông tin về bản đồ, đồng thời thực hiện thêm một bước nữa là thông qua Google Place API để lấy dữ liệu về

địa điểm. Ở đây là các bến xe buýt trong phạm vi định sẵn gắn với vị trí hiện tại của người dùng.

Để sử dụng được Google Map API và Google Place API, ta phải đăng kí sử dụng dịch vụ với Google theo các bước sau:



Hình 3.12. Lấy debug keystore

Debug keystore là một mã SHA-1 sinh ra từ file debug.keystore.

**Simple API Access**

Use API keys to identify your project when you do not need to access user data. [Learn more](#)

**Key for Android apps (with certificates)**

API key: AIzaSyB0K3Xj6t2rFDhwLSKdZuTG15DIttSaUk  
 Android apps: 19:B2:28:D1:29:2B:9B: - - - - - 67:DB:2F:com.hl.transportationhelper  
 Activated on: May 6, 2013 3:26 AM  
 Activated by: wasabi.love.sashimi@gmail.com - you

Hình 3.13. Lấy API key

Sau khi nhập Debug keystore và tên project(cách nhau bởi dấu “;”) ta sẽ thu được API key như hình trên. Cuối cùng, ta đưa khóa nhận được vào file AndroidManifest.xml như sau:

```
<meta-data
    android:name="com.google.android.maps.v2.API_KEY"
    android:value="@string/myGGAPIKey" />
```

Với myGGAPIKey là API key được định nghĩa ở file res/value/strings.xml.

Sau khi thực hiện thành công các bước trên, ta đã sẵn sàng cho việc sử dụng Google Map API và Google Place API.

Ở chức năng Định vị, để lấy được thông tin về địa chỉ của vị trí hiện tại, ta sẽ áp dụng kỹ thuật Reverse geocoding. Đó là quá trình ước lượng và trả về thông tin địa chỉ gắn với tọa độ cho trước. Do đây là quá trình ước lượng, dựa trên khoảng cách đến một tọa độ nhất định nên, kết quả trả về có thể chỉ là gần đúng(gần với vị trí cho trước) hoặc không tìm được địa chỉ nào.

Ở chức năng Bến xe buýt, khi người dùng kích hoạt chức năng này bằng cách bấm nút “Bến xe buýt”, một lệnh truy vấn dữ liệu được gửi đến Google server dưới dạng HttpURLConnection với đầu vào là một chuỗi được xây dựng như sau:

```
String type = "bus_station";
```

```
StringBuilder sb = new  
StringBuilder("https://maps.googleapis.com/maps/api/place/nearbysearch/json?");  
sb.append("location=" + mLatitude + "," + mLongitude);  
sb.append("&radius=2000");  
sb.append("&types=" + type);  
sb.append("&sensor=true");  
sb.append("&key=AIzaSyB0l_g6vejjoCBqy4Gc43GyCa-  
pm5GkYxU");
```

Trong đó:

- location: là tọa độ của vị trí hiện tại
- radius: là khoảng cách(bán kính) tới vị trí hiện tại tính bằng đơn vị mét(m)
- type: là kiểu vị trí cần tìm kiếm thông tin
- key: là khóa để sử dụng dịch vụ Google Place

Kết quả trả về của câu lệnh này là một đối tượng Json(Json Object). Đối tượng này sẽ chứa một danh sách các địa điểm cùng với thông tin liên quan tới địa điểm đó. Với tiêu chí đặt ra ở đây là tìm các bến xe buýt trong khoảng cách 2000m quanh vị trí hiện tại thì, đối tượng Json lúc này sẽ chứa thông tin về các bến xe buýt xung quanh vị trí đó trong khoảng 2000m. Để lấy được thông tin của đối tượng này, ta cần một lớp hỗ trợ là PlaceJSONParser. Lớp này sẽ đọc vào một đối tượng Json và trả về danh sách các địa điểm và thông tin đi kèm với nó như số nhà, đường phố, quận huyện...



### **3.8. Cấu hình**

#### **1. Cấu hình phát triển**

Eclipse 4.2, Java 1.6 và Android 4.2, Google Map API v2 cho Android, Google Place API

#### **2. Cấu hình tối thiểu:**

- Google Play services 3.0.25 (583950-10)
- Android 2.3.3–2.3.7 Gingerbread (API level 10)
- CPU 830 MHz ARMv6
- Ram 290 MB
- Hỗ trợ GPS
- Có tính năng WIFI hoặc hỗ trợ kết nối Internet

#### **3. Cấu hình đề nghị:**

- Google Play services 3.0.27 (599131-10) hoặc mới hơn
- Android 4.1 Jelly Bean (API level 16)
- CPU từ 1Ghz trở lên
- Ram từ 500 MB trở lên
- Hỗ trợ GPS
- Có tính năng WIFI hoặc hỗ trợ kết nối Internet

**CHƯƠNG 4. KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN**

Với ý tưởng ban đầu là thực hiện khóa luận với đề tài:

*“Xây dựng phần mềm hoạt động trên hệ điều hành Android, hỗ trợ người dùng tìm ra cách di chuyển giữa các bến xe buýt”.*

Mặc dù phải tự tìm hiểu về một khái niệm hoàn toàn mới mẻ là hệ điều hành Android, nhưng tác giả đã cố gắng nghiên cứu và trình bày một cách chân thực nhất những hiểu biết của mình về các vấn đề có thuộc khuôn khổ của khóa luận này. Với tính năng Chuyển tuyến, do lý do khách quan là không thể tìm được bộ dữ liệu chi tiết về khoảng cách giữa các tuyến nhưng tác giả đã hoàn thành phần này theo hướng tìm ra cách di chuyển giữa 2 bến xe buýt mà số lần chuyển bến là ít nhất, đồng thời cơ sở dữ liệu được thiết kế theo hướng mở (có trường Trọng số), trường này sẽ phục vụ cho việc tính toán chính xác cách di chuyển ngắn nhất dựa vào khoảng cách giữa các bến nếu có được bộ dữ liệu chi tiết. Bên cạnh đó, tính năng Định vị cũng đem đến khả năng mở rộng cho chính nó. Vì hoạt động dựa trên Google Map API và Google Place API nên khả năng bổ sung các tính năng mới là rất khả thi và hai API này được cập nhật thường xuyên nên khi được mở rộng tính năng này sẽ có khả năng đáp ứng được yêu cầu của người dùng.

Mặc dù nghiên cứu một lĩnh vực mới, nhưng tác giả đã luôn nhận được sự hỗ trợ về tài liệu và kiến thức từ các thầy cô và sự động viên chia sẻ từ bạn bè để tác giả có thể hoàn thành khóa luận này.

Xin chân thành cảm ơn!

**PHỤ LỤC 1 - CÁC THUẬT NGỮ****1. HTML:**

HTML là chữ viết tắt của Hyper Text Markup Language (Ngôn ngữ hiển thị siêu văn bản).

- Một file HTML là một file text bao gồm những tag nhỏ
- Những tag hiển thị nói cho trình duyệt biết nó phải hiển thị trang đó như thế nào
- Một file HTML phải có phần mở rộng là .htm hoặc .html
- Một file HTML có thể được tạo bởi một trình soạn thảo đơn giản.

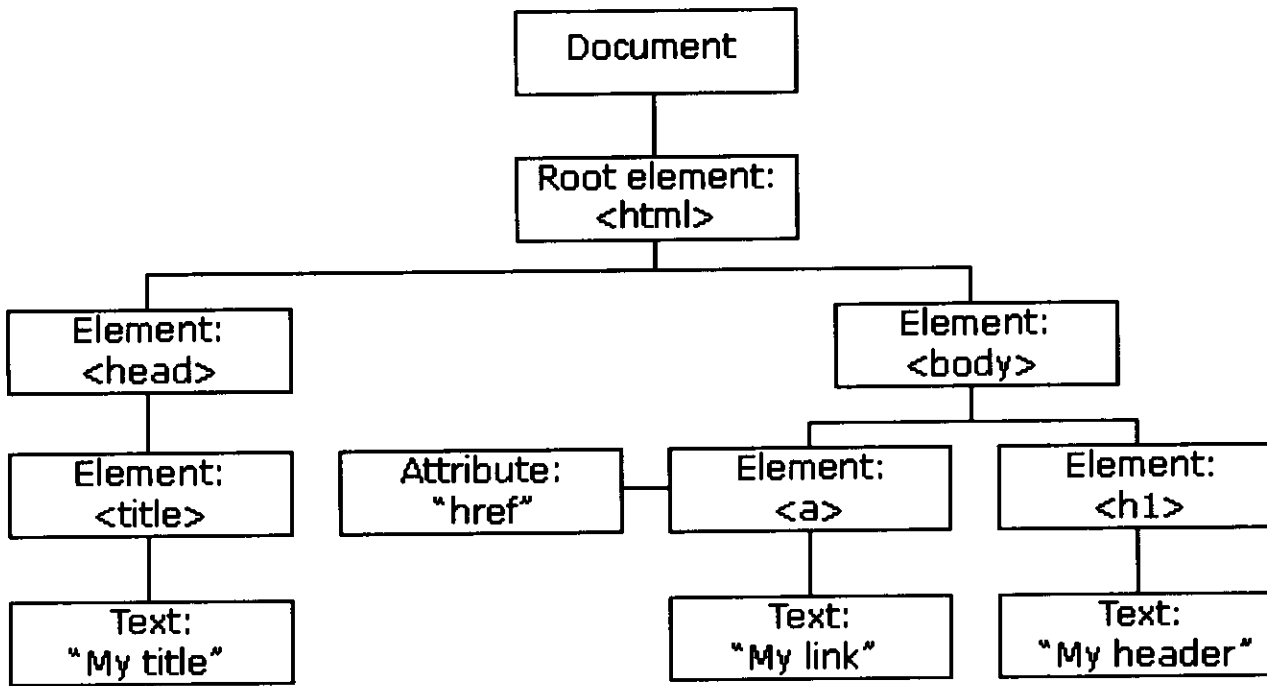
Cấu trúc cơ bản của 1 file HTML có dạng như sau:

```
<HTML>
  <HEAD>
    <TITLE> Tên trang </TITLE>
  </HEAD>
  <BODY>
    ....
    <!-- Văn bản và các thẻ HTML -->
    ....
  </BODY>
</HTML>
```

**2. DOM:**

là một mô hình cho phép truy xuất đến các thành phần của một tài liệu có cấu trúc. Viết tắt của ( Document Object Model ), được hỗ trợ bởi hầu hết các trình duyệt, nó cung cấp các Đối tượng , Phương thức, Thuộc tính để truy xuất các tài liệu HTML, XML , XHTML.

Minh họa DOM của file HTML.



**3. Jsoup:**

là một thư viện Java làm việc với HTML. Nó cung cấp các phương thức tiện lợi để trích xuất và thao tác với dữ liệu được tổ chức theo kiểu DOM. Ví dụ về việc lấy dữ liệu dùng Jsoup :

```

File input = new File ("/tmp/input.html");
Document doc = Jsoup.parse
    (input, "UTF-8", "http://example.com/");
Element content = doc.getElementById ("content");
Elements links = content.getElementsByTag ("a");
for (Element link : links) {
    String linkHref = link.attr ("href");
    String linkText = link.text ();
}
    
```

Trong ví dụ này, file “input.html” sẽ được đọc và lưu vào đối tượng doc kiểu Document, từ đó lấy ra các đường dẫn tới web có trong file này.

**4. SSL (Secure Sockets Layers):**

Việc kết nối giữa một Web browser tới bất kỳ điểm nào trên mạng Internet đi qua rất nhiều các hệ thống độc lập mà không có bất kỳ sự bảo vệ nào với các thông tin trên

đường truyền. Không một ai kể cả người sử dụng lẫn Web server có bất kỳ sự kiểm soát nào đối với đường đi của dữ liệu hay có thể kiểm soát được liệu có ai đó thâm nhập vào thông tin trên đường truyền. Để bảo vệ những thông tin mật trên mạng Internet hay bất kỳ mạng TCP/IP nào, SSL đã kết hợp những yếu tố sau để thiết lập được một giao dịch an toàn:

- Xác thực: đảm bảo tính xác thực của trang mà bạn sẽ làm việc ở đầu kia của kết nối. Cũng như vậy, các trang Web cũng cần phải kiểm tra tính xác thực của người sử dụng.

- Mã hoá: đảm bảo thông tin không thể bị truy cập bởi đối tượng thứ ba. Để loại trừ việc nghe trộm những thông tin “nhạy cảm” khi nó được truyền qua Internet, dữ liệu phải được mã hoá để không thể bị đọc được bởi những người khác ngoài người gửi và người nhận.

- Toàn vẹn dữ liệu: đảm bảo thông tin không bị sai lệch và nó phải thể hiện chính xác thông tin gốc gửi đến.

### **5. Geocoding (tìm kiếm theo địa chỉ)**

Một đối tượng trên bản đồ bao giờ cũng được biểu diễn bằng một kiểu dữ liệu đồ họa. Phần đồ họa này có thể thu được bằng cách số hoá hay quét ảnh bản đồ. Tuy nhiên, khi ta đã có bản đồ (bản đồ số), chúng ta cũng có thể xác định được phần đồ họa biểu diễn đối tượng hay là vị trí, hình dạng của đối tượng thông qua các dữ liệu mô tả vị trí của nó ví dụ: số nhà, tên đường, tên quận...

Geocoding (hay address matching) là một tiến trình nhằm xác định các đối tượng trên cơ sở mô tả vị trí của chúng. Đây là một kỹ thuật rất nổi tiếng, có mặt trong rất nhiều ứng dụng của GIS.

Người ta gọi một geocoding service là quá trình chuyển đổi toàn bộ mô tả thuộc tính về vị trí sang mô tả không gian.

Để tìm được vị trí thông qua địa chỉ, geocoding service phải tham chiếu đến ít nhất một nguồn dữ liệu bao gồm cả thông tin về địa chỉ (thuộc tính) và thông tin không gian (vị trí, hình dạng). Dữ liệu này được gọi là dữ liệu tham chiếu. Các geocoding service có thể thao tác trên nhiều kiểu dữ liệu tham chiếu khác nhau.

Sau khi đã geocoding dữ liệu tham chiếu (tức là ánh xạ mô tả thuộc tính vào mô tả không gian). Ta có thể nhập địa chỉ của đối tượng cần tìm. Quy trình xử lý trải qua các bước sau:

- Chuẩn hoá giá trị địa chỉ vừa nhập vào bằng cách tách nó thành các thành phần địa chỉ nhỏ.

Geocoding service sau đó sẽ tìm trong nguồn dữ liệu tham chiếu để xác định các đối tượng có các thành phần địa chỉ tương ứng với dữ liệu nhập vào. Mỗi kiểu geocoding service sẽ quy định các định dạng của các thành phần địa chỉ này.

Tập kết quả trả về sẽ được gán các trọng số (điểm) để tìm ra kết quả gần đúng nhất.

Geocoding service sẽ đánh dấu đối tượng vừa được tìm thấy trên bản đồ bằng một đối tượng đồ họa.

**PHỤ LỤC 2 – DANH MỤC TÀI LIỆU THAM KHẢO**

**Tài liệu tiếng Việt**

1. Quyết định số 1327/QĐ-TTG, *Quy hoạch phát triển giao thông vận tải đường bộ Việt Nam đến năm 2020 và định hướng đến năm 2030*, Thủ tướng Chính phủ.
2. Nguyễn Việt Cường, *Tuyến bus Hà Nội*, Trường Đại Học Công Nghệ - Đại Học Quốc Gia Hà Nội, 2012.
3. Trần Xuân Bộ, *EBUS 2.2 – Phần mềm trợ giúp đi xe Bus tại Hà Nội và Hồ Chí Minh*, 1/2010.

**Tài liệu tiếng Anh**

1. Wei Meng Lee, *BeginningAndroid™ Application Development*, Wiley Publishing Inc., 10475 Crosspoint Boulevard Indianapolis, 2011
2. Reto Meier, *Professional Android™ Application Development*, Wiley Publishing Inc., 10475 Crosspoint Boulevard Indianapolis, 2009