****ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH**

**TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN**

**KHOA MẠNG MÁY TÍNH VÀ TRUYỀN THÔNG**

🙚🙚🙚🕮🙘🙘🙘

******

**KHÓA LUẬN TỐT NGHIỆP**

**XÂY DỰNG HỆ THỐNG ĐẶT HÀNG QUA ĐIỆN THOẠI DI ĐỘNG**

**ORDER-SYSTEM**

Giảng viên hướng dẫn: **TS. ĐÀM QUANG HỒNG HẢI**

Sinh viên thực hiện: **NGUYỄN CHÍ HIẾU 10520490**

Lớp : **MMT05**

Khóa : **2010-2015**

***TP. Hồ Chí Minh, tháng 06*** ***năm 2015***

**MỞ ĐẦU**

Ngày nay với sự tiến bộ vượt bậc của khoa học công nghệ, cuộc sống của chúng ta ngày càng được thoải mái và thuận tiện hơn. Một trong những công nghệ phổ biến hiện nay đó là công nghệ định vị thông qua hệ thống định vị toàn cầu. Hiện nay có rất nhiều tổ chức quốc gia phát triển hệ thống này mà tiêu biểu như: Hệ thống Beidou của Cộng Hòa Dân Chủ Nhân Dân Trung Hoa, GLONASS của Nga, QZSS của Nhật, Galileo của liên minh châu Âu EU, nhưng đặc biệt nhất là GPS (Global Positioning System – Hệ thống định vị toàn cầu) do bộ quốc phòng Mỹ thiết kế, xây dựng và được triển khai rộng rãi vào năm 1980. Hệ thống GPS đang được sử dụng rất phổ biến trên toàn thế giới nhờ khả năng định vị chính xác và được cung cấp miễn phí hoàn toàn. Ở Việt Nam, hệ thống GPS không còn quá xa lạ. Nó đang được sử dụng để phục vụ trong một số lĩnh vực như: quản lý điều hành xe, tìm đường, du lịch, thám hiểm, thiết lập bản đồ …

Kể từ khi ra đời màn hình cảm ứng đầu tiên – IBM Simon được ra mắt bởi Bell South vào năm 1993, nhưng thời đại của smartphone tại Mỹ thực sự bắt đầu vào năm 2002 khi công nghệ CPA[1] giúp thực hiện cuộc gọi trên smartphone xuất hiện. Năm 2006, smartphone chỉ chiếm 6% thị phần của Mỹ. Tại thời điểm này, so với tốc độ phát triển của PC, tốc độ "phủ sóng" của smartphone ngày càng rộng lớn hơn. Và cho đến nay, Nielsen cho biết smartphone đã chiếm tới 2/3 số lượng điện thoại di động bán ra, dù đang diễn ra cuộc khủng hoảng kinh tế toàn cầu[[1]](#footnote-1). Với sự phát triển mạnh mẽ như vậy, chip xử lý trong smartphone ngày càng được thu nhỏ, rẻ và có hiệu xuất cao giúp giảm giá thành tạo điều kiện cho smartphone được sử dụng rộng rãi hơn nữa ở các nước đang phát triển như Việt Nam.

Tận dụng khả năng định vị của một số thiết bị GPS. Chúng ta có thể lấy thông tin tức thời và chính xác của thiết bị về vị trí, tốc độ di chuyển. Nhờ vào khả năng này, nó có thể được sử dụng vào trong việc đặt và giao hàng. Chẳng hạn như đề án “Giải pháp đặt hàng, giám sát qua công nghệ GPS” giành giải thưởng Nhân tài Đất Việt 2013 [[2]](#footnote-2) ứng dụng GPS để giúp các doanh nghiệp có thể giám sát toàn bộ quá trình INBOUND - OUTBOUND trong chuỗi cung ứng..

Từ thực tế trên, em đã đưa ra giải pháp là tận dụng khả năng định vị GPS được tích hợp trên hầu hết những smartphone vào việc đặt và giao hàng để. Nội dung chính của giải pháp này là kết nối smartphone đến máy chủ website đặt hàng để đặt hàng bằng ứng dụng đồng thời truyền tải thông tin về vị trí dung cho việc giao hàng. Về phía cửa hàng việc sữ dụng giả pháp này để giúp cho việc giao hàng trở nên tiện .

Hệ thống đặt hàng qua điện thoại di độngđược thực hiện trên nền tảng Android, một hệ điều hành dành cho các thiết bị di động được phát triển bởi Google, bên cạnh đó cũng xây dựng một Web Application thực hiện chức năng điều phối, quản lý đơn hàng và web đặt hàng bằng ngôn ngữ PHP.

Luận văn được cấu trúc như sau. Chương 2 nêu khái quát các kiến thức có liên quan được dùng để phục vụ đề tài, đồng thời khảo sát các công việc có liên quan đến đề tài. Chương 3 giới thiệu cách tiếp cận của chúng em để giải quyết vấn đề. Phần hiện thực hệ thống quản lý taxi cũng như đánh giá được trình bày trong chương 4. Trong chương 5, chúng em tổng kết đề tài và nêu ra một số hướng phát triển tiếp theo của đề tài.

TP. Hồ Chí Minh, ngày 17 tháng 06 năm 2015

Sinh viên

Nguyễn Chí Hiếu

**LỜI CẢM ƠN**

Em xin bày tỏ lòng biết ơn chân thành và sâu sắc nhất đến   
TS. Đàm Quang Hồng Hải, người đã tận tình hướng dẫn và tạo mọi điều kiện tốt nhất cho em trong quá trình định hướng, nghiên cứu và xây dựng đề tài này.

Chúng em xin chân thành cảm ơn các thầy cô Khoa Mạng Máy Tính Và Truyền Thông cũng như tất cả các thầy cô, cán bộ nhân viên trường Đại Học Công Nghệ Thông Tin – Đại Học Quốc Gia Thành Phố Hồ Chí Minh đã tận tình giảng dạy, tạo điều kiện tốt nhất cho chúng em trong suốt quá trình học tập tại trường, giúp chúng em học hỏi và tiếp thu được kiến thức một cách vững vàng.

Cuối cùng chúng em xin bày tỏ lòng biết ơn chân thành đến gia đình, người thân và bạn bè đã giúp đỡ, động viên, khích lệ chúng em trong quá trình hoàn thành đề tài này.

TP. Hồ Chí Minh, ngày 17 tháng 06 năm 2015

Sinh viên

Nguyễn Chí Hiếu

**LỜI CAM ĐOAN**

Em xin cam đoan tất cả nội dung trình bày trong khóa luận tốt nghiệp này có được dựa trên việc nghiên cứu, tổng hợp kiến thức lý thuyết và làm việc thực tế của em. Mọi thông tin trích dẫn đều được chú thích và liệt kê rõ ràng trong danh mục các tài liệu tham khảo.

Em xác nhận khóa luận tốt nghiệp này là sản phẩm của chúng em xây dựng dưới sự hướng dẫn của TS. Đàm Quang Hồng Hải.

TP. Hồ Chí Minh, ngày 17 tháng 06 năm 2015

Sinh viên

Nguyễn Chí Hiếu

**NHẬN XÉT**

**CỦA GIẢNG VIÊN HƯỚNG DẪN**

………………………………………………………………………………………

………………………………………………………………………………………

………………………………………………………………………………………

………………………………………………………………………………………

………………………………………………………………………………………

………………………………………………………………………………………

………………………………………………………………………………………

………………………………………………………………………………………

………………………………………………………………………………………

………………………………………………………………………………………

………………………………………………………………………………………

………………………………………………………………………………………

………………………………………………………………………………………

………………………………………………………………………………………

………………………………………………………………………………………

………………………………………………………………………………………

………………………………………………………………………………………

………………………………………………………………………………………

………………………………………………………………………………………

………………………………………………………………………………………

………………………………………………………………………………………

………………………………………………………………………………………

………………………………………………………………………………………

………………………………………………………………………………………

………………………………………………………………………………………

………………………………………………………………………………………

………………………………………………………………………………………

…………………………………………………………………………………………

**NHẬN XÉT**

**CỦA GIẢNG VIÊN PHẢN BIỆN**

………………………………………………………………………………………

………………………………………………………………………………………

………………………………………………………………………………………

………………………………………………………………………………………

………………………………………………………………………………………

………………………………………………………………………………………

………………………………………………………………………………………

………………………………………………………………………………………

………………………………………………………………………………………

………………………………………………………………………………………

………………………………………………………………………………………

………………………………………………………………………………………

………………………………………………………………………………………

………………………………………………………………………………………

………………………………………………………………………………………

………………………………………………………………………………………

………………………………………………………………………………………

………………………………………………………………………………………

………………………………………………………………………………………

………………………………………………………………………………………

………………………………………………………………………………………

………………………………………………………………………………………

………………………………………………………………………………………

………………………………………………………………………………………

………………………………………………………………………………………

………………………………………………………………………………………

………………………………………………………………………………………

………………………………………………………………………………………

**MỤC LỤC**

[CHƯƠNG 1. GIỚI THIỆU 1](#_Toc422666550)

[1.1 Tên đề tài 1](#_Toc422666551)

[1.2 Nội dung và giới hạn của đề tài 1](#_Toc422666552)

[1.2.1 Nội dung 1](#_Toc422666553)

[1.2.2 Giới hạn 2](#_Toc422666554)

[1.3 Đối tượng, phạm vi và phương pháp nghiên cứu của đề tài 2](#_Toc422666555)

[1.3.1 Đối tượng nghiên cứu 2](#_Toc422666556)

[1.3.2 Phạm vi nghiên cứu 2](#_Toc422666557)

[1.3.3 Phương pháp nghiên cứu 3](#_Toc422666558)

[1.4 Thời gian nghiên cứu 3](#_Toc422666559)

[1.5 Cấu trúc luận văn 3](#_Toc422666560)

[CHƯƠNG 2. CÔNG NGHỆ GPS, HỆ ĐIỀU HÀNH ANDROID VÀ NHỮNG CÔNG NGHỆ SỬ DỤNG TRONG QUÁ TRÌNH XÂY DỰNG HỆ THỐNG 4](#_Toc422666561)

[2.1 Map API 4](#_Toc422666562)

[2.1.1 Google Map API 4](#_Toc422666563)

[2.1.2 Geolocation (geoplugin API) 7](#_Toc422666564)

[2.2 Hệ điều hành Android 8](#_Toc422666565)

[2.2.1 Lịch sử phát triển Android 8](#_Toc422666566)

[2.2.2 Sự phát triển của Android 8](#_Toc422666567)

[2.2.3 Cấu trúc của Android 11](#_Toc422666568)

[2.2.4 Các thành phần chính của Android 14](#_Toc422666569)

[2.2.5 Google Map API Android 16](#_Toc422666570)

[2.2.6 Google Place API Android 26](#_Toc422666571)

[2.2.7 Google Direction API Android 28](#_Toc422666572)

[2.3 Công nghệ GPS 29](#_Toc422666573)

[2.3.1 Tổng quan về GPS 30](#_Toc422666574)

[2.3.2 Những phân đoạn GPS 30](#_Toc422666575)

[2.3.3 Những thế hệ của vệ tinh GPS 31](#_Toc422666576)

[2.3.4 Những vị trí điều khiển 32](#_Toc422666577)

[2.3.5 GPS-một vài khái niệm cơ bản 33](#_Toc422666578)

[2.3.6 Nguyên lý hoạt động của GPS 34](#_Toc422666579)

[2.4 PHP, MySQL, APACHE 35](#_Toc422666580)

[2.4.1 PHP và Yii Framework 35](#_Toc422666581)

[2.4.2 MYSQL 41](#_Toc422666582)

[2.4.3 Apache HTTP server 42](#_Toc422666583)

[2.5 JSON và thư viện mã nguồn mở GSON 45](#_Toc422666584)

[2.5.1 Định nghĩa JSON 45](#_Toc422666585)

[2.5.2 Thư viện mã nguồn mở GSON 46](#_Toc422666586)

[2.6 Kết luận 47](#_Toc422666587)

[CHƯƠNG 3. PHÂN TÍCH VÀ THIẾT KẾ HỆ THỐNG ORDER-SYSTEM 48](#_Toc422666588)

[3.1 Xác định yêu cầu và các đối tượng trong hệ thống 48](#_Toc422666589)

[3.1.1 Yêu cầu đối với hệ thống 48](#_Toc422666590)

[3.1.2 Các đối tượng trong hệ thống 48](#_Toc422666591)

[3.2 Thiết kế hệ thống 50](#_Toc422666592)

[3.3 Mô hình xử lý của hệ thống 52](#_Toc422666593)

[3.4 Giải pháp hiển thị bản đồ 53](#_Toc422666594)

[3.5 Tổng kết 54](#_Toc422666595)

[CHƯƠNG 4. HIỆN THỰC HỆ THỐNG ORDER - SYSTEM 55](#_Toc422666596)

[4.1 Xây dựng cơ sở dữ liệu 55](#_Toc422666597)

[4.1.1 Mô hình dữ liệu 55](#_Toc422666598)

[4.1.2 Chi tiết cơ sở dữ liệu 56](#_Toc422666599)

[4.2 Xây dựng Web Service 61](#_Toc422666600)

[4.2.1 Web Service hỗ trợ truy vấn dữ liệu 61](#_Toc422666601)

[4.2.2 Web Service lựa chọn xe phù hợp 62](#_Toc422666602)

[4.2.3 Web Service tương tác với Client 65](#_Toc422666603)

[4.3 Xây dựng ứng dụng web quản lý và điều hành taxi 69](#_Toc422666604)

[4.3.1 Tài nguyên hỗ trợ 69](#_Toc422666605)

[4.3.2 Cấu trúc ứng dụng 70](#_Toc422666606)

[4.3.3 Xây dựng các chức năng cho các đối tượng sử dụng hệ thống 71](#_Toc422666607)

[4.3.4 Xây dựng chức năng điều phối 71](#_Toc422666608)

[4.4 Xây dựng hệ thống bản đồ 74](#_Toc422666609)

[4.4.1 Phương pháp thực hiện 74](#_Toc422666610)

[4.4.2 Triển khai các bản đồ trong hệ thống 75](#_Toc422666611)

[4.5 Xây dựng ứng dụng Client 81](#_Toc422666612)

[4.5.1 Các xử lý về thông tin GPS 81](#_Toc422666613)

[4.5.2 Các xử lý và triển khai Google Direction API 82](#_Toc422666614)

[4.5.3 Các xử lý cơ bản trên Google Map API v2 83](#_Toc422666615)

[4.5.4 Truy vấn dữ liệu từ Web Service 89](#_Toc422666616)

[4.5.5 Các bước để thiết lập một phiên truyền thông bằng giao thức SIP 89](#_Toc422666617)

[4.5.6 Giao diện chương trình 93](#_Toc422666618)

[4.6 Triển khai tổng đài Trixbox 96](#_Toc422666619)

[4.6.1 Triển khai tổng đài Tribox 96](#_Toc422666620)

[4.6.2 Xây dựng thành phần quản lý phòng hội thoại 98](#_Toc422666621)

[4.7 Tổng kết 99](#_Toc422666622)

[CHƯƠNG 5. KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN 100](#_Toc422666623)

[5.1 Tổng kết đề tài 100](#_Toc422666624)

[5.2 Những khó khăn và hạn chế 101](#_Toc422666625)

[5.3 Hướng phát triển đề tài 101](#_Toc422666626)

[DANH MỤC TÀI LIỆU THAM KHẢO 102](#_Toc422666627)

**DANH MỤC HÌNH ẢNH**

[Hình 2‑1: Thống kê tỉ lệ sử dụng các Map API 5](file:///C:\Users\Nguyen%20Trung\Desktop\Quan%20ly%20taxi_%203-2.docx#_Toc349986848)

[Hình 2‑2: Thống kê thị phần các hệ điều hành di động 10](#_Toc349986849)

[Hình 2‑3: Các tầng trong một hệ thống Android 14](file:///C:\Users\Nguyen%20Trung\Desktop\Quan%20ly%20taxi_%203-2.docx#_Toc349986850)

[Hình 2‑4: Linux Kernel 2.6 15](file:///C:\Users\Nguyen%20Trung\Desktop\Quan%20ly%20taxi_%203-2.docx#_Toc349986851)

[Hình 2‑5: Android Framework 15](file:///C:\Users\Nguyen%20Trung\Desktop\Quan%20ly%20taxi_%203-2.docx#_Toc349986852)

[Hình 2‑6: Application Framework 16](file:///C:\Users\Nguyen%20Trung\Desktop\Quan%20ly%20taxi_%203-2.docx#_Toc349986853)

[Hình 2‑7: Application 16](file:///C:\Users\Nguyen%20Trung\Desktop\Quan%20ly%20taxi_%203-2.docx#_Toc349986854)

[Hình 2‑8: GPS 32](file:///C:\Users\Nguyen%20Trung\Desktop\Quan%20ly%20taxi_%203-2.docx#_Toc349986855)

[Hình 2‑9: 3 phân đoạn chính của GPS 33](#_Toc349986856)

[Hình 2‑10: Phân đoạn điều khiển của GPS 35](#_Toc349986857)

[Hình 2‑11: Qui trình thực thi một trang JSP 37](#_Toc349986858)

[Hình 2‑12: Servlet 38](#_Toc349986859)

[Hình 2‑13: Hoạt động của các servlet 39](#_Toc349986860)

[Hình 2‑14: HttpServlet 40](file:///C:\Users\Nguyen%20Trung\Desktop\Quan%20ly%20taxi_%203-2.docx#_Toc349986861)

[Hình 2‑15: Kiến trúc Tomcat 46](#_Toc349986862)

[Hình 2‑16: Biểu diễn json của một đối tượng 48](#_Toc349986863)

[Hình 2‑17: Biểu diễn json của một mảng 48](#_Toc349986864)

[Hình 2‑18: Sip cơ bản 54](#_Toc349986865)

[Hình 2‑19: Sip qua Registration 54](#_Toc349986866)

[Hình 2‑20: Sip qua Proxy server 55](#_Toc349986867)

[Hình 2‑21: Sip qua redirect serve 55](file:///C:\Users\Nguyen%20Trung\Desktop\Quan%20ly%20taxi_%203-2.docx#_Toc349986868)

[Hình 3‑1: Shadow Tracker® 58](#_Toc349986869)

[Hình 3‑2: GPS Vehicle Tracking Systems 59](#_Toc349986870)

[Hình 3‑3: Open GPS Tracking System 59](file:///C:\Users\Nguyen%20Trung\Desktop\Quan%20ly%20taxi_%203-2.docx#_Toc349986871)

[Hình 3‑4: Mô tả thiết kế hệ thống 66](#_Toc349986872)

[Hình 3‑5: Mô hình thực thể kết hợp 67](#_Toc349986873)

[Hình 3‑6: Mô hình xử lý của hệ thống 68](file:///C:\Users\Nguyen%20Trung\Desktop\Quan%20ly%20taxi_%203-2.docx#_Toc349986874)

[Hình 3‑7: Word Geodetic System 84 71](#_Toc349986875)

[Hình 3‑8: Giao diện tương tác người dùng 72](#_Toc349986876)

[Hình 3‑9: Mô hình giao tiếp giữa ứng dụng và Google Map Service. 73](#_Toc349986877)

[Hình 3‑10: Mô hình giao tiếp giữa ứng dụng và Taxi Manage Web Service 74](file:///C:\Users\Nguyen%20Trung\Desktop\Quan%20ly%20taxi_%203-2.docx#_Toc349986878)

[Hình 4‑1: Cơ sở dữ liệu 76](#_Toc349986879)

[Hình 4‑2: Web Service lựa chọn xe phù hợp 85](file:///C:\Users\Nguyen%20Trung\Desktop\Quan%20ly%20taxi_%203-2.docx#_Toc349986880)

[Hình 4‑3: Chức năng đăng nhập 86](file:///C:\Users\Nguyen%20Trung\Desktop\Quan%20ly%20taxi_%203-2.docx#_Toc349986881)

[Hình 4‑4: Chức năng đăng xuất 87](file:///C:\Users\Nguyen%20Trung\Desktop\Quan%20ly%20taxi_%203-2.docx#_Toc349986882)

[Hình 4‑5: Giao tiếp, cập nhật trạng thái và trao đổi dữ liệu với trung tâm điều khiển 89](#_Toc349986883)

[Hình 4‑6: Cách thức hoạt động của ứng dụng 91](file:///C:\Users\Nguyen%20Trung\Desktop\Quan%20ly%20taxi_%203-2.docx#_Toc349986884)

[Hình 4‑7: Ứng dụng web quản lý và điều hành taxi 92](#_Toc349986885)

[Hình 4‑8: Thành phần điều phối xe 93](file:///C:\Users\Nguyen%20Trung\Desktop\Quan%20ly%20taxi_%203-2.docx#_Toc349986886)

[Hình 4‑9: Hiển thị khách trên bản đồ 94](file:///C:\Users\Nguyen%20Trung\Desktop\Quan%20ly%20taxi_%203-2.docx#_Toc349986887)

[Hình 4‑10: Trạng thái khách hàng 94](file:///C:\Users\Nguyen%20Trung\Desktop\Quan%20ly%20taxi_%203-2.docx#_Toc349986888)

[Hình 4‑11: Danh sách khách hàng 95](file:///C:\Users\Nguyen%20Trung\Desktop\Quan%20ly%20taxi_%203-2.docx#_Toc349986889)

[Hình 4‑12: Mô hình xử lý hiển thị bản đồ 95](file:///C:\Users\Nguyen%20Trung\Desktop\Quan%20ly%20taxi_%203-2.docx#_Toc349986890)

[Hình 4‑13: Hiển thị vị trí xe 98](file:///C:\Users\Nguyen%20Trung\Desktop\Quan%20ly%20taxi_%203-2.docx#_Toc349986891)

[Hình 4‑14: Bản đồ trạng thái của hệ thống 99](#_Toc349986892)

[Hình 4‑15: Hiển thị thông tin một xe 99](#_Toc349986893)

[Hình 4‑16: Số lượng xe trong hệ thống 100](#_Toc349986894)

[Hình 4‑17: Hiển thị theo phân loại trạng thái xe 100](#_Toc349986895)

[Hình 4‑18: Giám sát hành trình của một xe 101](#_Toc349986896)

[Hình 4‑19: Lịch sử hoạt động của một xe 101](#_Toc349986897)

[Hình 4‑20: Vị trí trước 108](#_Toc349986898)

[Hình 4‑21: Vị trí sau 108](file:///C:\Users\Nguyen%20Trung\Desktop\Quan%20ly%20taxi_%203-2.docx#_Toc349986899)

[Hình 4‑22: Màn hình đăng nhập 114](#_Toc349986900)

[Hình 4‑23: Màn hình chính 115](#_Toc349986901)

[Hình 4‑24: Màn hình chỉ đường và tìm địa điểm 115](#_Toc349986902)

[Hình 4‑25: Màn hình nhận khách từ server 116](#_Toc349986903)

[Hình 4‑26: Màn hình khi có cuộc gọi tới và đi 116](#_Toc349986904)

[Hình 4‑27: Mô hình triển khai tổng đài trixbox 117](file:///C:\Users\Nguyen%20Trung\Desktop\Quan%20ly%20taxi_%203-2.docx#_Toc349986905)

[Hình 4‑28: Zoiper Wep 118](#_Toc349986906)

[Hình 4‑29: Quản lý phòng hội thoại 119](file:///C:\Users\Nguyen%20Trung\Desktop\Quan%20ly%20taxi_%203-2.docx#_Toc349986907)

[Hình 4‑30: Giao diện quản lý phòng hội thoại 120](#_Toc349986908)

**DANH MỤC BẢNG**

[Bảng 1‑1: Kế hoạch thực hiện đề tài 4](#_Toc349986932)

[Bảng 2‑1: Yêu cầu cấu hình tối thiểu cho Android 13](#_Toc349986933)

[Bảng 2‑2: Các Class trong GooleMap 26](#_Toc349986934)

[Bảng 2‑3: Các Class cơ bản trong SIP API 31](#_Toc349986935)

[Bảng 2‑4: Các phương thức của HTTP servlet tương ứng với các yêu cầu HTTP 40](#_Toc349986936)

[Bảng 2‑5: So sánh JSP và Servlet 42](#_Toc349986937)

[Bảng 2‑6: Một số định dạng được quy định sử dụng trong Web Service 44](#_Toc349986938)

[Bảng 2‑7: Các phiên bản Tomcat Server 45](#_Toc349986939)

[Bảng 4‑1: Quan hệ ADMIN 77](#_Toc349986940)

[Bảng 4‑2: Quan hệ BRANCH 77](#_Toc349986941)

[Bảng 4‑3: Quan hệ TEAM 77](#_Toc349986942)

[Bảng 4‑4: Quan hệ DRIVER 78](#_Toc349986943)

[Bảng 4‑5: Quan hệ DRIVERDETAIL 78](#_Toc349986944)

[Bảng 4‑6: Quan hệ OPERATOR 79](#_Toc349986945)

[Bảng 4‑7: Quan hệ TAXI 79](#_Toc349986946)

[Bảng 4‑8: Quan hệ CUSTOMER 80](#_Toc349986947)

[Bảng 4‑9: Quan hệ NOTIFICATION 81](#_Toc349986948)

[Bảng 4‑10: Quan hệ EXTENSION 81](#_Toc349986949)

[Bảng 4‑11: Quan hệ CONFERENCES 81](#_Toc349986950)

[Bảng 4‑12: Quan hệ CONFERENCEDETAIL 82](#_Toc349986951)

[Bảng 4‑13: Quan hệ UPDATETIME 82](#_Toc349986952)

[Bảng 4‑14: Web Service hỗ trợ truy vấn dữ liệu 83](#_Toc349986953)

[Bảng 4‑15: Trạng thái xe trong hệ thống 97](#_Toc349986954)

**DANH MỤC CÁC CHỮ VIẾT TẮT**

|  |  |
| --- | --- |
| 3G | Third-generation technology |
| 3GPP | 3rd Generation Partnership Project |
| ADT | Android Development Tools |
| AJAX | Asynchronous JavaScript and XML |
| API | Application Programming Interface |
| CE | Community Edition |
| CPA | Call Progress Analysis |
| CRUD | Create, Read, Update và Delete |
| CSDL | Cơ sở dữ liệu |
| CSS | Cascading Style Sheets |
| ERP | Enterprise resource planning |
| FOC | Full Operational Capability |
| GPL | General Public License |
| GPRS | General Packet Radio Service |
| GPS | Global Positioning System |
| GSM | Global System for Mobile Communications |
| GUI | Graphical user interface |
| HSDPA | High-Speed Downlink Packet Access |
| HTML | HyperText Markup Language |
| HTTP | Hypertext Transfer Protocol Secure |
| IDC | International Data Corporation |
| IMS | IP Multimedia Subsystem |
| IOC | initial operational capability |
| Json | JavaScript Object Notation |
| JSP | Java Server Page |
| MCS | master control station |
| MVC | Model–view–controller |
| OHA | Open Handset Alliance |
| PBX | Private Branch Exchange |
| PC | Personal Computer |
| RAS | Remote Access Services |
| REST | Representational State Transfer |
| RFC | Request for Comments |
| SCTP | Stream Control Transmission Protocol |
| SDP | Session Description Protocol |
| SIP | Session Initation Protocol |
| SMTP | Simple Mail Transfer Protocol |
| SOAP | Simple Object Access Protocol |
| TCP/IP | Transmission Control Protocol /Internet Protocol |
| UA | User Agents |
| UAC | User Agent Client |
| UAS | User Agent Server |
| UDP | User Datagram Protocol |
| URI | Uniform Resource Identifier |
| URL | Uniform Resource Locator |
| VOIP | Voice over Internet Protocol |
| WAAS | Wide Area Augmentation System |
| WGS 84 | Geodetic System 84 |
| WSDL | Web Services Description Language |
| XHTML | Extensible HyperText Markup Language |
| XML | Extensible Markup Language |

**DANH MỤC CÁC THUẬT NGỮ ANH VIỆT**

|  |  |
| --- | --- |
| 3G | Là công nghệ truyền thông thế hệ thứ ba. |
| Activity | Cửa sổ chứa giao diện ứng dụng trong Android. |
| Block | Vệ tinh khối |
| Framework | Chương trình khung. |
| Hash | Chuỗi băm. |
| Hexa | Hệ thập lục phân. |
| Latitude | Vĩ độ. |
| Longitude | Kinh độ. |
| Mashup | Là thuật ngữ dùng để chỉ các ứng dụng và web sites có sự kết hợp dữ liệu hoặc chức năng từ hai hay nhiều nguồn khác nhau. |
| Package | Gói chương trình. |
| Smartphone | Điện thoại thông minh. |
| Web Application | Là một ứng dụng được người dùng truy cập thông qua mạng. |
| Web-based | Là những ứng dụng sử dụng trên nền tảng Web. |
| Widget | Ô điều khiển trên màn hình. |
| Window Phone | Là một hệ điều hành dành cho các thiết bị di động được phát triển bởi Microsoft. |

# 

# GIỚI THIỆU

## Tên đề tài

Xây dựng hệ thống đặt hàng qua điện thoại di động – “ORDER-SYSTEM”.

## Nội dung và giới hạn của đề tài

### Nội dung

Cơ sở hạ tầng và ở Việt Nam ngày càng phát triển. Trong đó hệ thống xe taxi phục vụ khách hàng đã trở nên phổ biến và tiện dụng. Việc xe taxi phát triển mạnh mẽ giúp phục vụ việc đi lại của người dùng trở nên dễ dàng.Ta có thể thấy một vài đặc điểm trong việc điều phối taxi hiện nay như nhân viên điều phối liên lạc với các tài xế taxi thông qua hệ thống bộ đàm, các xe được trang bị thiết bị GPS chuyên dụng để định vị vị trí xe tuy nhiên việc trang bị thiết bị này tại Việt Nam hiện nay mất chi phí cao, triển khai phức tạp và chưa phổ biến. Chính điều này đã gây ra một số hạn chế cho nhân viên trong việc điều phối như việc thông báo, cung cấp thông tin của khách hàng cho taxi được thực hiện thông qua hệ thống bộ đàm đến tất cả các taxi trong hệ thống mà không thể lựa chọn được taxi hợp lý nhất để phục vụ khách hàng, hệ thống không trang bị thiết bị GPS chuyên dụng nhân viên không thể giám sát được trạng thái và vị trí của các taxi trong hệ thống… Để khắc phục những khó khăn này luận văn này đưa ra giải pháp xây dựng hệ thống quản lý taxi trong đó Smartphone nền tảng Android được sử dụng giúp cho tài xế có thể gửi thông tin trạng thái và vị trí về trung tâm điều khiển thay thế cho thiết bị GPS chuyên dụng trên xe ô tô, nhận thông báo đón khách từ trung tâm điều khiển, thực hiện việc liên lạc thông qua tổng đài VOIP của trung tâm trên nền mạng internet 3G thay thế cho hệ thống liên lạc bằng bộ đàm. Với sự phát triển mạnh mẽ của Smartphone, cơ sở hạ tầng mạng viễn thông hiện nay sẽ là nền tảng giúp cho hệ thống vận hành và đáp ứng tốt những yêu cầu của người sử dụng.

Đề tài này sẽ thực hiện những công việc sau:

Phía quản lí cửa hàng:

Cho phép khách hàng sữ dụng mobile application hoặc web đặt hàng các sản phẩm của mình:

* Sản phẩm và các thông số (như tên, giá cả, miêu tả) của các sản phẩm được bán.
* Cho phép khách hàng đặt hàng các sản phẩm.
* Quản lý, điều hành việc giao hàng dựa trên nền tảng Web.
* Hệ thống tổng đài Trixbox quản lý việc liên lạc nội bộ giữa tài xế và nhân viên điều phối.

Phía người giao hàng:

* Smartphone chạy hệ điều hành Android có cài đặt ứng dụng “Post Order” của hệ thống với các chức năng như sau:
* Giao tiếp với web server quản lý điều hành thông qua Web application.
* Cập nhật danh sách của các đơn hàng cần giao.
* Cập nhật bản đồ chứa vị trí của các đơn hàng.
* Nhận thông tin đường đi đế đơn hàng cần giao.
* Thực hiện xác nhận giao hàng.
* Thông báo có sự cố xãy ra với khách hàng trực tiếp thông qua tin nhắn

Phía người giao hàng:

* Đặt hàng thông qua web hoặc application đặt hàng trên smartphone chạy hệ điều hành android.
* Xác nhận các thông tin cần thiết để đặt hàng (tên, số điệ thoại địa chỉ giao hàng).

### Giới hạn

Lĩnh vực kinh doanh, đặc biệt là khinh doanh dịch vụ muốn vận hành được thì hệ thống cần phải có sự kết hợp chặt chẽ của nhiều thành phần khác nhau. Ở đây em chỉ tập trung vào việc xây dựng hệ thống đặt hàng qua điện thoại đối với của hàng thức ăn nhanh. Các vấn đề khác như quản lý hành chính, quản lý nhân sự, quản lý tài sản, quản lý kinh doanh… nằm ngoài tầm vóc của luận văn này.

## Đối tượng, phạm vi và phương pháp nghiên cứu của đề tài

### Đối tượng nghiên cứu

* Các yêu cầu kỹ thuật và chức năng của một hệ thống đặt hàng qua điện thoại.
* Tìm hiểu kỹ thuật lập trình trên thiết bị di động nền tảng Android.
* Kỹ thuật định vị người dùng.
* Tìm hiểu về kỹ thuật xây dựng Web Application với PHP, MYSQL và JAVASCRIPT bằng Yii Framework.
* Nghiên cứu và ứng dụng Google Maps API.

### Phạm vi nghiên cứu

Phạm vi nghiên cứu của đề tài được tập trung vào những phần sau:

* Tập trung vào kỹ thuật phát triển ứng dụng trên Google Maps API trên nền tảng Android kết hợp với định vị ngừời dùng bằng hệ thống GPS.
* Phát triển Web Application với PHP, MYSQL và JAVASCRIPT bằng Yii Framework.
* Ứng dụng Google Map JavaScript API trong việc hiển thị dữ liệu đơn hàng dùng trong phân phối.

### Phương pháp nghiên cứu

Đề tài được thực hiện dựa trên phương pháp khảo sát, phân tích, ứng dụng các đối tượng nghiên cứu để xây dựng hệ thống.

## Thời gian nghiên cứu

Thời gian bắt đầu: 15/04/2012

Thời gian hoàn thành: 17/06/2015

## Cấu trúc luận văn

Luận văn được cấu trúc như sau.

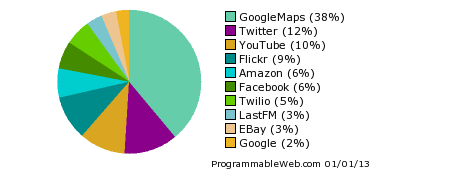
* Chương 1: Giới thiệu
* Chương 2: Công nghệ GPS, Hệ điều hành Android và những công nghệ sử dụng trong việc xây dựng hệ thống
* Chương 3: Phân tích và thiết kế hệ thống ORDER-SYSTEM
* Chương 4: Hiện thực hệ thống ORDER-SYSTEM
* Chương 5: Kết luận và hướng phát triển

# CÔNG NGHỆ GPS, HỆ ĐIỀU HÀNH ANDROID VÀ NHỮNG CÔNG NGHỆ SỬ DỤNG TRONG QUÁ TRÌNH XÂY DỰNG HỆ THỐNG

Chương này nhóm chúng em sẽ trình bày những nền tảng lý thuyết chính được sử dụng để làm cơ sở xây dựng hệ thống quản lý taxi bao gồm: Map API, Hệ điều hành Android, Công nghệ GPS, PHP, MySQL, JSON và thư viện mã nguồn mở GSON.

## Map API

Ngày nay trên các hệ thống websites các bản đồ là một thành phần xuất hiện khá phổ biến. Các bản đồ này được sử dụng để giúp hiển thị vị trí một địa điểm cụ thể lên bản đồ, tìm kiếm một địa chỉ, tìm hướng dẫn chỉ đường đến một địa chỉ và còn nhiều tiện ích khác nữa được các bản đồ hỗ trợ. Một sự vật bất kì chỉ cần nó có thông tin về vị trí thì đều có thể được hiển thị lên bản đồ.

Hiện nay có rất nhiều nhà cung cấp giải pháp về bản đồ như Yahoo! Maps, Bing Maps … nhưng được sử dụng phổ biến nhất vẫn là Google Maps. Theo thống kê ngày 1/1/2013 của website [www.programmableweb.com/apis](http://www.programmableweb.com/apis) Google Maps chiếm tới 38% các nội dung được sử dụng trong các Mashup[[3]](#footnote-3). Mashup là thuật ngữ dùng để chỉ các ứng dụng và web site có sự kết hợp dữ liệu hoặc chức năng từ hai hay nhiều nguồn khác nhau, cách thức này đang ngày càng trở nên phổ biến trong các ứng dụng web.

Hình ‑: Thống kê tỉ lệ sử dụng các Map API

### Google Map API

Google Map API tập hợp các API dành cho các nhà phát triển ứng dụng có thể tương tác, cũng như sử dụng các dịch vụ được Google và được cung cấp một cách miễn phí và có phí. Nền tảng của các API này là bản đồ của Google ở những dạng khác nhau giúp dễ dàng tích hợp nó vào trong các ứng dụng khác nhau như Website, smartphone… Ngoài ra, với Google Map cũng như các dịch vụ khác, chúng ta còn có thể tạo ra một bản đồ chỉ hiển thị một khu vực được định sẵn của riêng mình.

#### Google Maps Javascript API

Google Maps Javascript API được cung cấp dành riêng cho việc sử dụng Google Map trên các website. API này giúp hiển thị cũng như tương tác với bản đồ, kèm theo đó là các dịch vụ như tìm kiếm, chỉ đường… được cung cấp một cách miễn phí. Kết hợp với tài khoản Google, chúng ta có thể đánh dấu các đia điểm của riêng mình kèm theo hình ảnh và nội dung. Website sử dụng API này do Google tạo ra để cung cấp cho người sử dụng là <http://maps.google.com>.

#### Google Maps API Flash

Có chức năng tương tự như Google Maps Javascript API, nhưng sử dụng Flash để hiển thị thay vì Javascript. Google Maps API Flash dùng để phát triển các ứng dụng sử dụng Flash.

#### Google Earth API

Tương tự như Google Map Javascript API nhưng cung cấp thêm tính năng hiển thị hình ảnh 3D kỹ thuật số toàn cầu. Sử dụng API này chúng ta có thể đánh dấu, vẽ đường đi, hiển thị hình ảnh trên bề mặt bản đồ. Ngoài ra với API này, chúng ta còn có thể xây dựng ứng dụng bản đồ hiển thị 3D phức tạp để thêm vào website. Để sử dụng đươc Google Earth API thì yêu cầu trình duyệt phải được cài đặt thêm một số plugin.

#### Google Maps Image API

Là API dùng để cung cấp hình ảnh về một địa điểm cụ thể cho người dùng. Tham số đưa vào là tọa độ của điểm cần hiển thị và kích thước của hình ảnh tạo ra. Google Map Image API gồm hai loại chính là StaticMap và StreetView:

* Static Map: cung cấp cho người dùng hình ảnh tĩnh của bản đồ để hiển thị vào trong website.
* StreetView: cung cấp cho người dùng hình ảnh đường phố tại một địa điểm bất kỳ. Nhưng dịch vụ này chỉ cung cấp trong một số nước nhất định.

Google Maps Image API được cung cấp hoàn toàn miễn phí và không giới hạn số lượt sử dụng.

#### Maps API Web Service

Map API Web Service là tập hợp các dịch vụ về bản đồ đươc Google cung cấp thông qua giao thức HTTP, người dùng sử dụng các định dạng URL cùng những tham số khác nhau để truy vấn thông tin từ Google. Kết quả trả về là thông tin được lưu theo định dạng Json hay XML. Maps API Web Service gồm các dịch vụ sau đây:

* Direction API: cho phép người dùng tìm kiếm đường đi ngắn nhất giữa hai điểm trên bản đồ tùy thuộc vào loại phương tiện di chuyển. Phương tiện di chuyển được cung cấp gồm có 3 loại là xe ôtô, xe buýt và đi bộ. Kết quả trả về là một tập tin Json lưu trữ thông tin hướng dẫn đường đi, khoảng cách, danh sách các điểm để có thể vẽ đường đi trên Google Map. Đây là một dịch vụ hoàn toàn miễn phí, nhưng ở một số nước trong đó có Việt Nam vẫn chưa cung cấp toàn bộ các dịch vụ này.

Định dạng truy vấn của Direction API là:

http://maps.googleapis.com/maps/api/directions/output?parameters

* Distance Matrix API: là một dịch vụ của google giúp tính toán khoảng cách và thời gian giữa hai hay nhiều điểm trên bản đồ khi di chuyển bằng các phương tiện khác nhau. Dịch vụ này chỉ cho phép tính toán tối đa 100 địa điểm trên một lần truy vấn, 100 địa điểm trong 10 giây, 2500 địa điểm trong một ngày. Đối với phiên bản thương mại thì được sử dụng: 625 địa điểm trong một yêu cầu, 1000 địa điểm trong 10 giây và 100000 địa điểm trong 1 ngày.

Định dạng truy vấn của Distance Matrix API là:

http://maps.googleapis.com/maps/api/distancematrix/output?parameters

* Elevation API: là một dịch vụ của Google dùng để truy vấn thông tin về độ cao, độ sâu của bất kỳ điểm nào trên bản đồ. Giá trị này được tính toán thông qua độ cao trung bình của 4 điểm gần đó nhất. Cũng như Distance Matrix API, Elevation API cũng chỉ cho phép thực hiện tối đa 512 địa điểm trong một truy vấn, 2500 truy vấn trong một ngày. Với phiên bản thương mại có thể thực hiện tối ta 100.000 truy vấn trong một ngày và mỗi truy vấn chứa tối đa 1.000.000 địa điểm.

Định dạng truy vấn của Distance Matrix API là:

http://maps.googleapis.com/maps/api/elevation/outputFormat?parameters

* Geocoding API: là một dịch vụ của Google cho phép chuyển đổi qua lại các địa chỉ cụ thể thực tế thành tọa độ vật lý. Nhờ đó chúng ta có thể tìm kiếm chính xác vị trí trên bản đồ thông qua địa chỉ. Nó được cung cấp miễn phí nhưng cũng rất hạn chế trong việc truy xuất. Mỗi ngày đối với một người dùng chỉ được sử dụng 2500 truy vấn và 10000 truy vấn đối với phiên bản thương mại.

Định dạng truy vấn của Geocoding API theo dạng:

http://maps.googleapis.com/maps/api/geocode/output?parameters

* Place API: là dịch vụ của Google cho phép truy vấn để tìm kiếm thông tin về địa điểm, thông tin được cung cấp bao gồm địa chỉ, phân loại, điểm đánh giá… Nhưng Place API chỉ cho phép người dùng thực hiện 1000 truy vấn trong một ngày.

Định dạng truy vấn của Geocoding API theo dạng:

https://maps.googleapis.com/maps/api/place/search/output?parameters

* Time Zone API: là dịch vụ của Goole cho phép truy vấn múi giờ cho một vị trí trên trái đất. Tham số đầu vào là vị trí cần truy vấn thời gian, API sẽ trả về tên và id của múi giờ và thời gian lệch so mới múi giờ gốc.

Định dạng truy vấn của Geocoding API theo dạng:

<https://maps.googleapis.com/maps/api/timezone/outputFormat?parameters>

### Geolocation (geoplugin API)

Geolocation là công nghệ định vị trên web dựa trên IP nên còn được biết đến với tên IP Location. GeoPlugin là cách dễ nhất để bạn biết nơi truy cập của bạn là từ đó bạn có thể cung cấp cho họ với nội dung phù hợp hơn với vị trí địa lý của họ.

GeoPlugin độc đáo ở chỗ nó cho phép bạn làm điều này một đơn giản với nhiều ngôn ngữ lập trình khác nhau. Không cần cài đặt phần mềm cần thiết, và cho dù ngôn ngữ lập trình của bạn lựa chọn là javascript, php, xml hoặc bất cứ điều gì, geoPlugin có thể tìm ra vùng truy cập của bạn.

**PHP web service**

Để kết nối với dịch vụ đơn giản chỉ cần thực hiện tru vấn đến địa chỉ:

http://www.geoplugin.net/php.gp?ip=xx.xx.xx.xx

Ví dụ thực hiện truy vấn sau:

var\_dump(unserialize(file\_get\_contents('http://www.geoplugin.net/php.gp?ip='.$\_SERVER['REMOTE\_ADDR'])))

Kết quả nhận được sẽ tương tự như sau:

array (

'geoplugin\_request' => '58.186.19.182',

'geoplugin\_status' => 200,

'geoplugin\_credit' => 'Some of the returned data includes GeoLite data created by MaxMind, available from [http://www.maxmind.com](http://www.geoplugin.com/webservices/%5C%5C%5C%27http:/www.maxmind.com%5C%5C%5C%27).',

'geoplugin\_city' => 'Ho Chi Minh City',

'geoplugin\_region' => 'Hồ Chí Minh',

'geoplugin\_areaCode' => '0',

'geoplugin\_dmaCode' => '0',

'geoplugin\_countryCode' => 'VN',

'geoplugin\_countryName' => 'Vietnam',

'geoplugin\_continentCode' => 'AS',

'geoplugin\_latitude' => '10.8142',

'geoplugin\_longitude' => '106.643799',

'geoplugin\_regionCode' => '20',

'geoplugin\_regionName' => 'Hồ Chí Minh',

'geoplugin\_currencyCode' => 'VND',

'geoplugin\_currencySymbol' => '₫',

'geoplugin\_currencySymbol\_UTF8' => '₫',

'geoplugin\_currencyConverter' => '21805.495',

)

Trong hệ thống ORDER-SYSTEM sẽ dụng geoplugin để xác định vị trí khi khách hàng thực hiện đặt hàng trên web site.

## Hệ điều hành Android

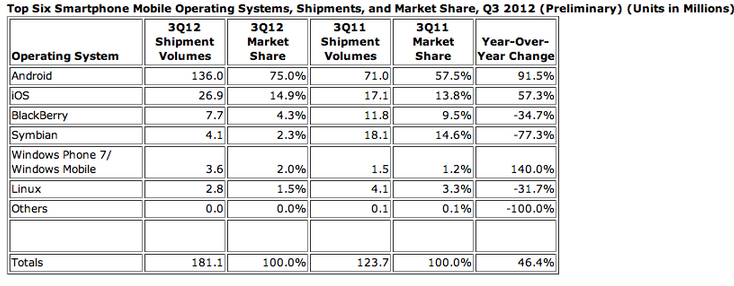
### Lịch sử phát triển Android

Vào khoảng những năm đầu của thế kỷ 21, Andy Rubin[[4]](#footnote-4) đã tạo ra Android như một hệ điều hành mở dành cho thiết bị di động. Năm 2005, Google đã mua sản phẩm này từ tay Andy Rubin và cử ông làm giám đốc phát triển nền tảng di động cho Google. Đây thực sự là một quyết định mang tính đột phá của Google, một mặt giúp Google khẳng định mạnh mẽ hơn vị trí của mình trong lĩnh vực mới được xem là Internet 2.0, nhưng mặt khác cũng tạo ra cuộc cạnh tranh khốc liệt giữa Google và đối thủ nặng ký Apple Iphone, đang là nhà cung cấp được ưu chuộng nhất thời gian này.

Tháng 11 năm 2007, Android chính thức ra mắt phiên bản 1.0 ra đời cùng với sự thành lập tổ chức OHA[[5]](#footnote-5) gồm nhiều công ty phần cứng, phần mềm. Mục tiêu chính của tổ chức này là cùng nhau nghiên cứu phát triển tiêu chuẩn mở cho các thiết bị di động, để giảm giá thành sản phẩm và nâng cao chất lượng dịch vụ. Android là hệ điều hành mở dành cho thiết bị di động đầu tiên, ra đời theo mục tiêu này. Nó dựa trên nền tảng mở Linux và ngôn ngữ lập trình Java của Oracle. Nền tảng Android phải hoạt động như một máy tính chạy trên chip, nghĩa là đủ nhỏ để vừa với thiết bị và đủ mạnh để chạy các ứng dụng. Ngoài ra, Android còn cung cấp các công cụ, tài liệu, thư viện hỗ trợ những nhà phát triển phần mềm và phần cứng phát triển ứng dụng cho thiết bị di động một cách dễ dàng. Trải qua nhiều năm, Android đã phát triển và nâng cấp lên các phiên bản khác nhau và phiên bản mới nhất hiện nay là 4.2. Thiết bị cầm tay cài đặt Android trên thị trường là thiết bị T-Mobile G1 do HTC sản xuất và bán ra thị trường Mỹ. Ngày nay, rất nhiều nhà sản xuất điện thoại di động như HTC, Samsung, LG Electronic, T-Mobile sử dụng Android làm hệ điều hành cho các dòng điện thoại mới của họ.

### Sự phát triển của Android

Có đến 136 triệu điện thoại Android được xuất xưởng trong quý 3 năm nay, tăng 91,5% so với cùng kỳ năm ngoái. Trong quý 3/2011, hệ điều hành Android có mặt trong 71 triệu điện thoại được phát hành và nắm trong tay khoảng 57,5% thị phần[[6]](#footnote-6).



Hình ‑: Thống kê thị phần các hệ điều hành di động

Bên cạnh sư phát triển chóng mặt của Android, có thêm hai hệ điều hành khác đang có đà tăng trưởng nhanh. Đầu tiên phải nhắc đến nền tảng iOS, một nền tảng từng thống trị thị trường smartphone. Theo IDC[[7]](#footnote-7), số lượng điện thoại iOS được xuất xưởng tăng tới 57,3%, trong quý 3 xuất xưởng trên 26.9 triệu chiếc iPhone chiếm 14,9% thị trường, tăng 17,1 triệu chiếc so với cùng kỳ năm ngoái. Tiếp tới là nền tảng Windows Phone cùa Microsoft. Hệ điều hành di động này dù ra đời sau nhưng cũng có mức tăng trưởng thật sự ấn tượng. Thống kê của IDC cho thấy đã có khoảng 3,6 triệu điện thoại Windows Phone được phát hành trong quý 3 nhưng con số này vẫn còn đứng sau 7,7 triệu điện thoại BlackBerry của RIM và 4,1 triệu điện thoại Nokia chạy trên nền tảng Symbian.

Sự phát triển của Android không chỉ thể hiện trên thị trường sản phẩm mà còn trên cả thị trường ứng dụng. Đến tháng 10 năm 2012, đã có khoảng 700.000 ứng dụng được viết cho Android, và số lượng tải về ứng dụng từ Google Play - cửa hàng ứng dụng chính của Android - được ước tính là 25 lượt[[8]](#footnote-8).

Trong lần đầu tiên ra mắt vào ngày 22 tháng 10 năm 2008, hệ điều hành Android bao gồm 12.000.000 dòng mã lệnh trong đó có 3.000.000 dòng XML, 2.800.000 dòng C, 2.100.000 dòng Java và 1.750.000 dòng C++.Từ lúc xuất hiện đến nay thì Android đã cho ra mặt những phiên bản[[9]](#footnote-9):

* Android 1.0 Donut (API level 1): Ngày 15 tháng 9 năm 2008, Android 1.6 SDK ra đời với tên gọi là dubbed Donut dựa trên nền tảng Linux 2.6.29 bao gồm rất nhiều những cập nhật mới. Thiết bị đầu tiên được sử dụng là HTC Dream.
* Android 1.1 (API level 2): Ngày 9 tháng 2 năm 2009, phiên bản cập nhật Android 1.1 ra đời. Thiết bị được sử dụng đầu tiên cũng là HTC Dream. Phiên bản này không có đột phá về mặt ý tưởng, mà chỉ là phiên bản sửa lỗi của phiên bản 1.0.
* Android 1.5 Cupcake (API level 3): Ra đời ngày 30 tháng 4 năm 2009, Android 1.5 được chính thức ra đời dựa trên nền tảng linx 2.6.27. Đây là phiên bản đầu tiên được đặt tên dựa theo những món tráng miệng. Phiên bản này cung cấp những cải tiến mới như Android 1.5 hỗ trợ hiễn thị bàn phím trên màn hình, Widget, ghi âm và hình ảnh.
* Android 1.6 Donut (API level 4): Ra đời ngày 15 tháng 9 năm 2009, Android 1.6 SDK dubbed Donut được ra đời dựa trên nền tảng Linux 2.6.29 bao gồm những cập nhật mới như: chức năng tìm kiếm nhanh, đọc văn bản, xóa nhiều tập tin hình ảnh củng lúc, hỗ trợ màn hình có độ phân giải WVGA…
* Android 2.0 Éclair (API level 5): được chính thức ra đời vào ngày 26 tháng 10 năm 2009, Android 2.0 với tên gọi là Eclair chạy trên nền tảng Linux 2.6.29, với những tính năng ngày càng hoàn thiện và tối ưu hơn chẳng hạn như: cho phép quản lý nhiều tài khoảng Email, tìm kiếm tin nhắn, Google Map 3.1.2, hỗ trợ nhiều kích thước và độ phân giải của màn hình…
* Android 2.01 Éclair (API level 6): Ra đời ngày 3 tháng 12 năm 2009, đây chỉ là phiên bản sửa lỗi cho phiên bản trước đó là Android 2.0
* Android 2.1 Eclair (API level 7): Phiên bản ra đời ngày 12 tháng 1 năm 2010, không có nâng cấp đáng kể trong phiên bản này.
* Android 2.2 Proyo (API level 8): Vào ngày 20 tháng 5 năm 2010, Android 2.2 SDK được ra đời dựa trên nền tảng Linux 2.6.32, với những tính năng bổ sung như: tối ưu hóa tốc độ và sử dụng bộ nhớ, tích hợp Javascript version 8 của Chrome vào trong trình duyệt web, hỗ trợ dịch vụ C2DM, hỗ trợ Adobe Flash…
* Android 2.3 Gingerbread (API level 9): Ra đời vào ngày 6 tháng 12 năm 2010 dựa trên nền tảng Linux 2.6.35 với những thay đổi như: hỗ trợ thiết kế giao diện đơn giản và hiệu quả, hỗ trợ hiển thị trên màn hình có độ phân giải lớn, nhập văn bản thông minh và nhanh hơn, hỗ trợ chức năng sao chép và dán, hỗ trợ tính năng dọn rác để tăng khả năng xử lý cho hệ điều hành, …..
* Android 2.3.3 Gingerbread(API level 10): Ra đời vào ngày 9 tháng 2 năm 2011 được sử dụng để sửa lỗi cho phiên bản trước.
* Android 3.0 Honeycomb (API level 11): Ngày 22 tháng 2 năm 2011 phiên bản Android 3.0 SDK lần đầu tiên xuất hiện trên máy tính bảng chạy hệ điều hành Android dựa trên nền tảng Linux 2.6.36 với những thay đổi lớn như: thêm System Bar và Action Bar, hỗ trơ đa tác vụ, bàn phím ảo được thiết kế để nhập văn bản nhanh hơn, trình duyệt với nhiều tab, hỗ trợ những bộ vi xử lý đa lõi, có khả năng mã hóa tất cả thông tin người dùng, Google Talk hỗ trợ chức năng Video Chat.
* Android 3.1 Honeycomb (API level 12): Ra đời ngày 10 tháng năm 2011 nhằm mục đích cung cấp thêm một số tính năng so với phiên bản trước như: hỗ trợ kết nối USB, hỗ trợ kết nối tay cầm chơi game, hỗ trợ HTTP proxy[[10]](#footnote-10) cho mỗi kết nối Wifi…
* Android 3.2 Honeycomb (API level 13): Ra đời ngày 15 tháng 7 năm 2011 nhưng không có sự thay đổi gì nổi bật ngoại trừ một số cải tiến như: khả năng truy cập vào các tập tin nằm trong thẻ nhớ.
* Android 4.0 Ice Cream Sandwich (API level 14): Được xây dựng trên nền tảng Linux 3.0.1 chính thức công bố vào ngày 19 tháng 10 năm 2011. Phiên bản này kết hợp thiết kế của cả 2 phiên bản là 3.x và 2.x vì vậy nó có thể chạy trên bất cứ thiết bị di động nào dù là điện thoại di động hay máy tính bảng, cùng với đó là những tính năng ưu việc hơn như: giao diện đẹp, Widget có thể thay đổi kích thước, cho phép khóa màn hình …
* Andoid 4.1 Jelly Bean (API level 16): Google công bố Android 4.1 ở hội nghị Google I/O vào ngày 27 tháng 6 năm 2012. Dựa trên nền tảng linux 3.0.31, JellyBean được xem như là một bản cập nhật với mục đích chính là cải thiện chức năng và hiệu xuất của giao diện, tăng khả năng tương tác với người dùng, máy tính bảng Nexus 7 là thiết bị đầu tiên được nâng cấp lên phiên bản này vào ngày 13 tháng 7 năm 2012.
* Android 4.2 Jelly Bean (API level 17): Google dự định công bố phiên bản Android 4.2 vào sự kiện ở thành phố New York nhưng bị hủy và dời vào ngày 13 tháng 12 năm 2012. LG Nexus 4 và SamSung’s Nexus 10 là hai sản phẩm đầu tiên được cập nhật phiên bản này. Các tính năng mới trong phiên bản này như: thêm vào tính năng “Photo Sphere”, bàn phím nhập hiệu quả hơn, màn hình khóa hỗ trợ thêm việc hiển thị Widget và chuyển trực tiếp sang camera, hỗ trợ đa người dùng trên máy tính bảng, hiển thị mạng wireless…

### Cấu trúc của Android

#### Phần cứng

Android là một nền tảng phần mềm được chạy trên các thiết bị di động vì vậy những yêu cầu về phần cứng của Android tương đối thấp, sau đây là những yêu cầu về cấu hình tối thiểu mà Android yêu cầu:

|  |  |
| --- | --- |
| Phần cứng | Những yêu cầu tối thiểu |
| Chip set | ARM-based |
| Ram | 128MB Ram. 256MB Flash External |
| Ổ cứng | Minu hay Micro SD |
| Màn hình chính | QVGA TFT LCD hoặc lớn hơn. |
| Navigation Key | 2MP CMOS |
| Bluetooth | 1.2 hoặc 2.0 |

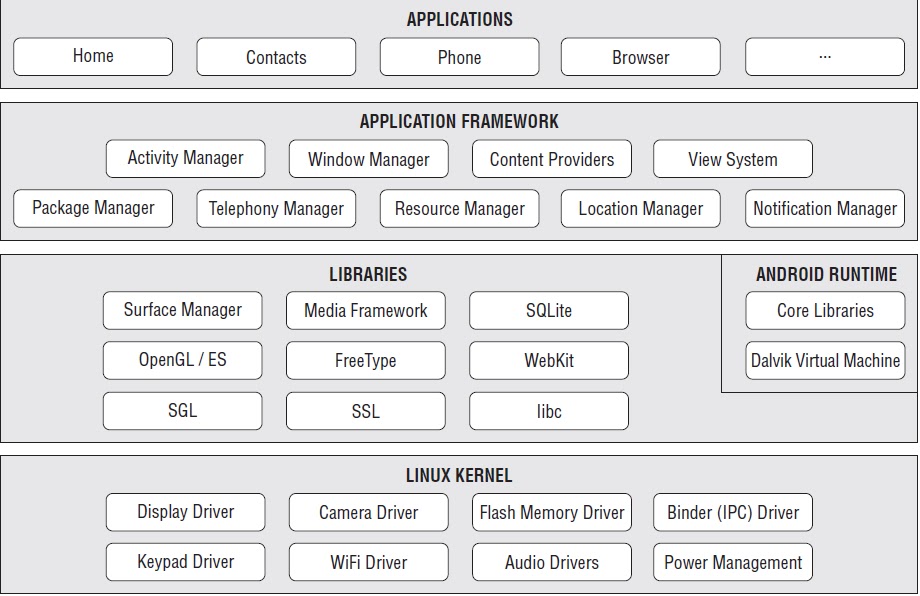
Bảng ‑: Yêu cầu cấu hình tối thiểu cho Android

#### Hệ điều hành Android

Có 5 tầng phân biệt trong một hệ thống Android:

* Tầng lõi ARM Linux: là tầng thấp nhất được xây dựng từ hai thành phần chính là Linux kernel và bộ vi xử lý ARM[[11]](#footnote-11). Bộ vi xử lý ARM hoạt động với hiệu suất cao nhưng tiêu thụ năng lượng rất ít. Còn Linux kernel phiên bản 2.6 được chứng minh là phiên bản có tình ổn định cao. Sự kết hợp giữa hai thành phần này, nhằm giải quyết phần nào giới hạn về nguồn năng lượng sử dụng và tối ưu việc sử dụng các tài nguyên của thiết bị.
* Tầng thư viện: chứa các mã nguồn cấp thấp được sử dụng cho các chức năng cơ bản như: mã hóa và giải mã âm thanh, hình ảnh kĩ thuật số, trình bày các giao diện đồ họa, bảo mật lưu lượng TCP/IP cũng như các thành phần cho trình duyệt Web, hỗ trợ xuất cơ sở dữ liệu (SQLite)…
* Tầng máy ảo Android: đây là một phần mềm dùng để chạy các ứng dụng trên thiết bị Android, bao gồm một tập các thư viện lõi cung cấp hầu hết các chức năng trong thư viện lõi của ngôn ngữ lập trình Java và máy ảo Dalvik[[12]](#footnote-12). Mỗi ứng dụng Android sẽ chạy trong một tiến trình riêng dành cho nó, với một máy ảo Dalvik riêng cũng được thiết lập dành riêng cho mỗi ứng dụng. Dalvik được viết để Android có thể chạy nhiều máy ảo cùng lúc nhằm tăng hiệu xuất.
* Tầng khung ứng dụng Android: tầng này có chức năng định nghĩa các Android API, cung cấp bộ công cụ ở mức cao để lập trình viên Android có thể dễ dàng xây dựng ứng dụng.
* Tầng ứng dụng lõi Android: bao gồm các ứng dụng cơ bản hỗ trợ thêm cho người lập trình như: Webkit brower, Google Calendar, Gmail, Maps.

Hình ‑: Các tầng trong một hệ thống Android

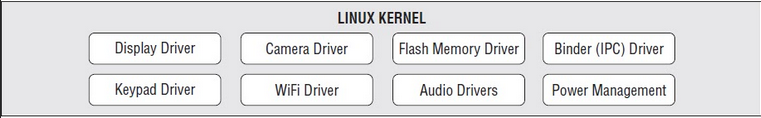
* Application…Chúng ta có thể viết thêm các ứng dụng cho tầng này.

**Linux Kernel 2.6**

Android được xây dựng trên nền tảng Linux kernel 2.6 mã nguồn mở, chứa những trình điều khiển dùng để điều khiển phần cứng như Keypad, Wifi, Camera, Audio, màn hình… được xem là trung tâm chỉ huy của nền tảng Android. Các giải thuật quản lý tài nguyên, chuyển đổi qua lại giữa các tác vụ qua lại, phân chia quyền, giải quyết tranh chấp được cài đặt trong linux kernel đảm nhận trách nhiệm tương tác với các thiết bị ngoại vi. Tuy nhiên người lập trình không thể lập trình trực tiếp trên tầng này.

Những tính năng nổi bật của Linux Kernel:

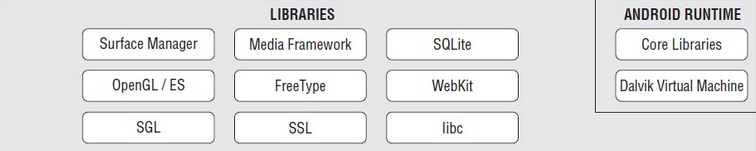
* Mô hình bảo mật tốt.
* Quản lý bộ nhớ tối ưu.
* Quản lý tiến trình hiệu quả.
* Quản lý các kênh kết nối mạng tốt.



Hình ‑: Linux Kernel 2.6

**Android Framework**

Được phát triển trên nền của Linux kernel, tầng này được cấu thành từ hai hay nhiều tầng con, cung cấp những tính năng cho Android:

* Android native libraries: đây là tập thư viện C/C++ và bạn sẽ gọi thư viện này thông qua giao diện Java. Phần này gồm thư viện:
  + Surface Manager dùng đễ tạo ra các cữa sổ giao diện.
  + OpenGL dùng để hỗ trợ xây dựng ứng dụng đồ họa 2D và 3D.
  + Media Framework dùng để xây dựng các ứng dụng âm thanh và hình ảnh.
  + SSL cung cấp giao thức truyền bảo mật.
  + SQLiter cung cấp khả năng quản lý cơ sở dữ liệu.
* Android Runtime: gồm các thư viện lõi của Java và máy ảo Dalvik. Các ứng dụng Android đươc viết bằng ngôn ngữ Java, nên khi biên dịch sẽ thành các tập tin \*.class. Máy ảo Dalvik sẽ làm chức năng đóng gói các tập tin này thành các tập tin nhỏ gọn \*.dex có thể dùng để chạy trên thiết bị, giúp tối ưu bộ nhớ và nguồn năng lượng giới hạn.

**Application Framework**

Hình ‑: Android Framework

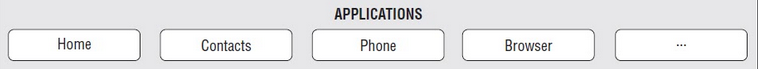
Tầng này chứa các thư viện Java hỗ trợ người dùng giao tiếp với tầng Android framework. Một phần của tầng này là do Google cung cấp sẵn cho chúng ta, một phần là do chúng ta tạo ra. Trong tầng này thành phần quan trọng nhất là Activity Manager dùng để quản lý chu trình sống của Activity.

* Activity Manager: quản lý chu trình sống của Activity trong ứng dụng Android.
* Telephony Manager: cung cấp thư viện để truy xuất đến các dịch vụ điện thoại cũng như thuê bao.
* View System: xử lý giao diện trong Android của bạn.
* Location Manager: cung cấp thư viện hỗ trợ xác định vị trí của chúng ta.



**Application**

Hình ‑: Application Framework

Đây là tầng cao nhất và ứng dụng của bạn sẽ nằm trong tầng này. Tuy nhiên, để tiện lợi cho việc truy cập một số thông tin ở các tầng dưới, Google đã viết một số thư viện tiện ích trong tầng này giúp chúng ta có thể truy xuất thông tin các tầng ở dưới linh hoạt và hiệu quả nhất.

Hình ‑: Application

### Các thành phần chính của Android

Những ứng dụng Android được viết bằng ngôn ngữ Java, sử dụng bộ công cụ Android SDK để biên dịch toàn bộ chương trình thành tập tin \*.apk để có thể cài đặt trên thiết bị chạy Android.

Mỗi ứng dụng sau khi được cài đặt trên thiết bị sẽ có những đặc điểm sau:

* Mỗi ứng dụng sẽ được Android gán cho một User Id phân biệt với nhau.
* Hệ thống sẽ thiết lập quyền hạn riêng cho mỗi ứng dụng trong việc truy nhập vào tập tin của hệ thống.
* Mỗi tiểu trình của ứng dụng sẽ chạy trên các máy ảo khác nhau vì vậy những ứng dụng này sẽ hoạt động độc lập với nhau.

Dù khác nhau ở cách thức hoạt động nhưng bất kỳ chương trình Android nào cũng bao gồm 3 thành phần chính là:

* + Application Components.
  + Tập tin Manifest.
  + Application Resources.

#### Application Components

Application Components đóng vai trò là thành phần giúp giao tiếp giữa ứng dụng của chúng ta và hệ thống. Có 4 loại Application Component chính như sau:

* Activities: Đại diện cho một cửa sổ chứa giao diện của chương trình giao tiếp với người dùng. Activities chứa giao diện và các hàm xử lý của chương trình. Mỗi Activity được tạo ra trong ứng dụng sẽ kế thừa một lớp Activity của Android Platform.Có 3 dạng Activity chính là:
  + Floating Activity: là loại Acitivity nổi, có kích thước không phủ lấp toàn bộ màn hình.
  + Activity Group: là một nhóm các Activity nhúng vào trong Activity khác.
  + Full Screen Activity: là dạng Activity phủ kín toàn bộ màn hình.
  + Một vòng đời của Activity gồm 3 trạng thái chính là:
  + Resumed: là trạng thái chạy của Activity, khi Activity chuyển sang trạng thái này, giao diện của nó sẽ được hiển thị trên màn hình.
  + Paused: là khi một Activity đang chạy mà có một Activity khác được khởi động thì trạng thái của Activity cũ sẽ chuyển sang dạng Paused. Khi ở trạng thái này thì Activity sẽ không còn được hiển thị trên màn hình.
  + Stopped: Khi ở trạng thái này thì Activity sẽ bị hủy nếu có một ứng dụng khác yêu cầu thêm bộ nhớ từ hệ thống.
  + Android cũng cấp cấp một số sự kiện để quản lý 3 trạng thái này của Activity:
  + onCreate(): Sự kiện này được gọi khi Activity lần đầu tiên được tạo. Trong sự kiện này chúng ta phải thực hiện một số công việc như: tạo giao diện cho Activity, tải dữ liệu vào danh sách… Sau sự kiện này thì sự kiện onStart() sẽ được gọi.
  + onStart(): Sự kiện này được gọi khi giao diện của ứng dụng được hiển thị lên màn hình. Lúc này tương tác của người dùng vẵn chưa được thiết lập.
  + onRestart(): Sự kiện này được gọi khi một Activity đang ở trạng thái Stopped nhưng muốn hiển thị và dành quyền điều khiển.
  + onResume(): Sự kiện này được gọi khi Activity bắt đầu cho phép tương tác với người dùng.
  + onPause(): Sự kiện này được gọi khi một Activity khác muốn dành quyền hiển thị và điều khiển của Activity hiện hành.
  + onStop(): Sự kiện này được gọi khi một Activity bị che khất hoàn toàn.
  + onDestroy(): Sự kiện này là sự kiện cuối cùng trước khi Acitivity bị hủy.
* Services: đây là Application Component chạy nền để thực hiện những tác vụ ẩn bên dưới mà không cần phải hiển thị hay chiếm quyền điều khiển. Một Services được tạo ra trong ứng dụng sẽ là một lớp con trong lớp Service trong Android Platform và chúng ta có thể dùng Activity để khởi động Service này.
* Content Providers: được sử dụng để quản lý một tập các dữ liệu chia sẻ ở những định dạng khác nhau như dưới dạng tập tin, cơ sở dữ liệu SQLite, trên Website hay bất cứ nơi nào mà chúng ta có thể truy xuất được. Thông qua Content Provider, những ứng dụng khác có thể truy xuất dữ liệu từ ứng dụng của chúng ta nếu được cho phép.
* Broadcast Receivers: là loại Application Component trong ứng dụng dùng để lắng nghe các thông điệp được gửi đi từ hệ thống. Bản thân những ứng dụng cũng có thể gửi đi những những Broadcast Receivers để những ứng dụng khác biết. Một đối tượng BroadcastReceivers được tạo ra từ lớp cha BroadcastReceiver trong Android Platform.

#### Tập tin Manifest

Được sử dụng để khai báo các Application Component, tập tin Manifest sẽ bao gồm những nội dung sau:

* Quyền hạn của ứng dụng:
* Khai báo các API được sử dụng trong chương trình.
* Khai báo những tính năng phần cứng mà ứng dụng có sử dụng.
* Các thư viện.
* Danh sách các Application Component.

#### Application Resources

Là những tài nguyên được sử dụng cho việc thiết kế giao diện cho chương trình bao gồm hình ảnh, âm thanh, giao diện… Tất cả những Application Resources này được lưu trữ trong thư mục res/. Mỗi thành phần trong Application Resources đều được hệ điều hành Android gắn cho một ID tương ứng và lưu trữ trong tập tin R.java trong thư muc gen/ của ứng dụng.

### Google Map API Android

Google Maps API là thư viện lập trình miễn phí được cung cấp cho người phát triển ứng dụng cùng với đó là các dịch vụ đi kèm như tìm kiếm địa điểm, tìm đường… Mặc dù miễn phí nhưng để có thể sử dụng các dịch vụ này chúng ta cần phải đăng ký và chứng thực với Google.

Trước khi làm việc với Google Map API, chúng ta cần tải API này và chắc chắn rằng mình đã có một Google Map Android API v2 key. Cả API và key đều được cung cấp miễn phí.

Để tạo một ứng dụng android mới sử dụng Google Map Android API v2 chúng ta phải thực hiện theo một số bước khác nhau. Nhiều bước chúng ta chỉ thực hiện một lần nhưng thông tin thu được sẽ hữu ích cho các ứng dụng tương lai sau này. Toàn bộ quá trình thêm một bản đồ đươc cung cấp bởi Google vào trong một ứng dụng Android được thực hiện như sau:

1. Tải và cấu hình Google Play service SDK[[13]](#footnote-13).Google Map Android API là một thành phần của SDK này.
2. Lấy được một API key: Để thực hiện được điều này chúng ta cần phải đăng ký một project trong Google APIs Console và nhận một chữ ký chứng thực cho chương trình của chúng ta:
3. Ghi rõ các cài đặt trong Application Manifest.
4. Thêm một Google Map tới một Android project mới hay đã tồn tại.
5. Xuất bản ứng dụng của chúng ta để có thể được sử dụng rộng rãi.

Google Map Android API v2 là một bộ phận của Google Play Service SDK, chúng ta có thể tải nó từ Android SDK manager. Để sử dụng google Maps android apiv2 trong chương trình, chúng ta phải cài đặt Google Play service SDK trước.

#### Google Map API Android v2 Key

Google maps android api v2 sử dụng một hệ thống quản lý Key mới. Những Key từ phiên bản Google Map Android v1 thông thường được biết với tên gọi là MapView sẽ không hoạt động trên phiên bản v2.

Để truy cập vào dịch vụ Google Maps với Maps API, chúng ta phải thêm một Maps API Key vào chương trình. Key này được cung cấp miễn phí cho tất cả mọi người. Chúng ta có thể dùng nó cho bất kỳ chương trình nào gọi tới Maps API và nó hỗ trợ không giới hạn số người dùng. Chúng ta thu được một Maps API Key từ Google APIs Console bằng việc cung cấp chữ ký chứng thực của chương trình chúng ta và tên của package. Một khi đã có được Key, chúng ta thêm nó vào ứng dụng của chúng ta bằng cách khai báo thêm một thành phần trong file AndroidManifest.xml của chương trình.

Để hiểu được quá trình đăng ký chương trình của chúng ta và thu được một Key yêu cầu một vài kiến thức về quá trình xuất bản một ứng dụng Android và những yêu cầu của nó. Tóm lại, tất cả các ứng dụng android phải đăng ký một chữ ký số mà bạn giữ Key riêng. Bởi vì chứng thực số chỉ có duy nhất, họ cung cấp một cách đơn giản để nhận biết chương trình của chúng ta. Nó rất hiệu quả trong việc tìm kiếm chương trình của chúng ta trong hệ thống cũng như Google Play Store, cũng như dò tìm những tài nguyên mà chương trình của chúng ta sử dụng như Google Maps servers.

Maps API Key được liên kết rõ ràng với cặp chứng thực/package, thay vì cho người sử dụng hay chương trình. Chúng ta chỉ cần duy nhất một Key cho mỗi chứng thực, bất kể bao nhiêu người dùng ứng dụng. Những ứng dụng mà sử dụng chung chứng thực có thể sử dụng chung API Key. Tuy nhiên, nên đăng ký cho mỗi chương trình một chứng thực khác nhau và nhận một khóa khác nhau cho mỗi chương trình.

Để nhận một Key cho một chương trình chúng ta cần thực hiện qua một số bước, những bước thực hiện được nêu ở đây và mô tả cụ thể trong chi tiết của mỗi phần:

1. Lấy lại thông tin về chứng thực của chương trình.
2. Đăng ký một project trong Google APIs Console[[14]](#footnote-14) và thêm vào Maps API như là một dịch vụ của project.
3. Một khi chúng ta có một project, chúng ta có thể yêu cầu một hay nhiều Key.
4. Cuối cùng, chúng ta có thể thêm một hay nhiều Key vào chương trình.

#### Hiển thị thông tin chứng thực

Maps API Key dựa trên một hình thức ngắn của chứng thực số, được biết đến với tên gọi SHA-1 fingerprint. Fingerprint là một chuỗi ký tự duy nhất được tạo ra từ thuật toán Hash[[15]](#footnote-15) của SHA-1[[16]](#footnote-16). Bởi vì fingerprint là một thuật toán đặc biệt nên Google Map sử dụng nó như một cách để chứng thực cho chương trình.

Để hiển thị SHA-1 fingerprint cho chứng thực của chúng ta thì trước tiên chúng ta phải đảm bảo chúng ta đã có chứng thực của mình. Chúng ta có hai cách để nhận chứng thực này:

* Debug certificate: bộ công cụ android SDK tạo ra chứng thực này một cách tự động khi chúng ta thực hiện debug chương trình từ dòng lệnh hay khi chúng ta build hay run chương trình từ eclipse mà không xuất bản chương trình. Chứng thực chỉ được dùng cho một chương trình mà chúng ta đang thử nghiệm. Chúng ta không thể xuất bản chương trình với debug certificate. Chúng ta có thể tạo ra một API Key từ chứng thực này nhưng Key này chỉ được sử dụng cho chương trình thử nghiệm.
* Release certificate: Bộ công cụ Android SDK tạo ra chứng thực này khi chúng ta sử dụng chức năng “release” của IDE. Chúng ta cũng có thể tạo ra chứng thực này thông qua chương trình keytool. Chứng thực này có thể được sử dụng cho một chương trình mà bạn muốn xuất bản. Một khi chúng ta có chứng thực

chính xác mà chúng ta cần, chúng ta có thể hiển thị SHA-1 fingerprint sử dụng chương trình keytool[[17]](#footnote-17).

Để có thể nhận được chính xác loại chứng thực cần thiết thì chúng ta phải thực hiện đẩy đủ một số bước sau đây:

*Đối với Debug certificate*

1. Xác định chính xác vị trí của tập tin “debug.keystore”, mặc định nó nằm chung thư mục với thư mục Android Virtual Device của bạn:
   * OS X and Linux: ~/.android/
   * Windows Vista và Windows 7: C:\Users\your\_username\.android\

Nếu chúng ta sử dụng eclipse với ADT và chúng ta không chắc tập tin này nằm ở đâu thì bạn có thể kiểm tra đường dẫn tại: Windows>Prefs>Android>Build.

1. Hiển thị thông tin SHA-1 fingerprint
   * Đối với Linux hay OS X, mở giao diện dòng lệnh và nhập vào dòng lệnh sau:

keytool -list -v -keystore ~/.android/debug.keystore -alias androiddebugkey -storepass android -keypass android

* + Đối với Vista hay Win 7, chạy dòng lệnh sau:

keytool -list -v -keystore "C:\Users\your\_user\_name\.android\debug.keystore" -alias androiddebugkey -storepass android -keypass android

1. Chúng ta sẽ nhìn thấy kết quả như bên dưới đây:

 Alias name: androiddebugkey  
 Creation date: Jan 01, 2013  
 Entry type: PrivateKeyEntry  
 Certificate chain length: 1  
 Certificate[1]:  
 Owner: CN=Android Debug, O=Android, C=US  
 Issuer: CN=Android Debug, O=Android, C=US  
 Serial number: 4aa9b300  
 Valid from: Mon Jan 01 08:04:04 UTC 2013 until: Mon Jan 01 18:04:04 PST 2033  
 Certificate fingerprints:  
      MD5:  AE:9F:95:D0:A6:86:89:BC:A8:70:BA:34:FF:6A:AC:F9  
      SHA1: BB:0D:AC:74:D3:21:E1:43:07:71:9B:62:90:AF:A1:66:6E:44:5D:75  
      Signature algorithm name: SHA1withRSA  
      Version: 3

Dòng bắt đầu bằng chuỗi SHA1 sẽ chứa mã chứng thực SHA-1 fingerprint. Chuỗi chứng thực fingerprint chứa 20 ký tự hexa[[18]](#footnote-18) và được ngăn cách với nhau bởi dấu hai chấm.

*Đối với release fingerprint*

1. Không có vị trí mặc định cho tập tin release keystore, nếu bạn không chỉ định nó khi xuất bản chương trình thì nó sẽ cho ra một tập tin apk không được đăng ký. Bạn phải đăng ký nó trước khi đưa ra cộng đồng sử dụng. Đối với release certificate, chúng ta cần biết về tên, mật khẩu, của keystore và chứng thực của nó. Có thể hiện thị tất cả các tên của khóa trong keystore[[19]](#footnote-19) bằng dòng lệnh:

keytool -list -keystore your\_keystore\_name

1. Thay thế “your\_keystore\_name” bằng đường dẫn và tên của keystore của chúng ta bao gồm cả đuôi .keystore, cần cung cấp mật khẩu của keystore để có thể hiển thị được những thông tin này.

Nhập vào dòng lệnh sau trong giao diện dòng lệnh:

keytool -list -v -keystore your\_keystore\_name -alias your\_alias\_name

1. Thay thế “your\_keystore\_name” bằng đường dẫn và tên của keystore của chúng ta bao gồm cả đuôi .keystore. Thay thế “your\_alias\_name”bằng tên mà chúng ta đã đăng ký khi khởi tạo.

Kết quả hiển thị ra sẽ tương tự như bên dưới:

Alias name: <alias\_name>  
 Creation date: Feb 02, 2013  
 Entry type: PrivateKeyEntry  
 Certificate chain length: 1  
 Certificate[1]:  
 Owner: CN=Android Debug, O=Android, C=US  
 Issuer: CN=Android Debug, O=Android, C=US  
 Serial number: 4cc9b300  
 Valid from: Mon Feb 02 08:01:04 UTC 2013 until: Mon Feb 02 18:05:04 PST 2033  
 Certificate fingerprints:  
      MD5:  AE:9F:95:D0:A6:86:89:BC:A8:70:BA:34:FF:6B:AC:F9  
      SHA1: BB:0D:AC:74:D3:21:E1:43:67:71:9B:62:90:AF:A1:66:6E:44:5D:75  
      Signature algorithm name: SHA1withRSA  
      Version: 3

Dòng bắt đầu bằng chuỗi SHA1 sẽ chứa mã chứng thực SHA-1 fingerprint. Chuỗi chứng thực fingerprint chứa 20 ký tự hexa và được ngăn cách với nhau bởi dấu hai chấm.

#### Tạo một API project

Một khi chúng ta đã đăng ký thành công chứng thực fingerprint, tạo và chỉnh sửa một project cho chương trình của chúng ta trên Google APIs Console và đăng ký Maps API.

Để nhận đươc một project và đăng ký với API chúng ta cần thực hiện một số bước như sau:

1. Trong trình duyệt, chúng ta chuyển hướng tới trang https://code.google.com/apis/console/

* Nếu chúng ta chưa từng sử dụng Google API Console trước đây thì cẩn phải tạo mới một project thông qua nhấp chuột vào nút “Create Project”. Console sẽ tạo cho chúng ta một project có tên là API Project. Ở trang tiếp theo, tên này sẽ được hiện ra phía bên trái, tại đây chúng ta sẽ có thể đổi tên cũng như quản lý Project.
* Nếu trước đây chúng ta đã sử dụng Google API Console thì sẽ thấy ngay danh sách cái project và các dịch vụ. Lúc này chúng ta có thể tạo một project mới hoặc đổi tên project cũ lại để sử dụng

1. Chúng ta sẽ thấy danh sách các dịch vụ nằm ở trung tâm của trang. Nếu vẫn chưa thấy, chọn “service” bên thanh định hướng phía bên trái.
2. Trong danh sách các dịnh vụ này, kéo xuống cho đến khi bạn thấy Google Maps Android API v2. Bật dịch vụ này.
3. Màn hình tiếp theo hiển thị các điều khoảng của dịch vụ. Nếu đồng ý, nhấp vào “Accept” để đồng ý. Sau đó nó sẽ trả lại màn hình chính.

Bây giờ chúng ta đã có thể nhận khóa Maps API.

#### Nhận một API Key

Nếu chương trình của chúng ta đã đăng ký sử dụng dịch vụ Google Android Map v2, thì chúng ta có thể yêu cầu nhận một API Key, có thể đăng ký nhiều khóa trên một project.

Để nhận được Key chúng ta phải thể thực hiện các bước sau đây:

1. Trên thanh định hướng phía bên trái, nhấp chuột vào API Access.
2. Trên trang kết quả, nhấp chuột vào Create New Android Key.
3. Sau đó nhập vào SHA-1 fingerprint và tên của Packet của bạn. ví dụ như:

BB:0D:AC:74:D3:21:E1:43:67:71:9B:62:91:AF:A1:66:6E:44:5D:75;com.example.android.mapexample

1. Google API console trả lại chuỗi gồm 40 ký tự đó là Key của chương trình Android.

AIzaSyBdVl-cTICSwYKrZ95SuvNw7dbMuDt1KG0

1. Lưu Key này để sử dụng cho các chương trình sử dụng dịch vụ Google map API v2.

#### Một số class cơ bản dùng để hiển thị và quản lý trên android

Theo [[1](#Placeholder1)] thì ta có thể tóm tắt các class trong GooleMap như sau:

|  |  |
| --- | --- |
| Tên Class | Chức năng |
| BitmapDescriptor | Dùng để tạo ra một bức hình, đối với marker có thể dùng để tạo ra marker icon, đối với Ground Overlay nó có thể thiết lập một hình ảnh tại một địa điểm trên bề mặt bản đồ. Để có được một Bitmapdescriptor ta sử dụng class BitmapDescriptorFactory. |
| BitmapDescriptorFactory | Dùng để tạo ra BitmapDescriptor |
| CameraPosition | Một class dùng để tập hợp tất cả các thuộc tính của việc hiển thị một điểm trên bản đồ. |
| CameraPosition.Builder | Dùng để xây dựng một camera position. |
| GroundOverlay | Dùng để hiện thị một hình ảnh trên bề mặt bản đồ tại một vị trí xác định. |
| GroundOverlayOptions | Dùng để tập hợp các thuộc tính của GroundOverlay với các thuộc tính cơ bản như:   * Anchor(float u, float v): Xác định điểm neo của hình. * Position(LatLng location, float width, float height): Chỉ ra vị trí, chiều dài và chiều rộng của khung hình. * Image(BitmapDescriptor image): thiết lập hình ảnh hiển thị trong GroundOverlay * Bearing(float bearing): thiết lập góc nhìn nghiêng với tham số là độ lớn của góc, mặc định bằng 0 |
| LatLng | Là class đai diện cho cặp tọa độ latitude/longitude dùng để xác định vị trí của một điểm bất kỳ trên bản đồ trước khi thể hiện nó. Tham số đầu vào là cặp giá trị kiểu double đại diện cho latitude/longitude. |
| LatLngBound | Là class đại diện cho những cặp tọa độ LatLng để tạo thành hình chữ nhật. |
| Marker | Là class dùng để đánh dấu một điểm trên bản đồ bằng một icon. Marker được vẽ theo định hướng màn hình của thiết bị hơn là so với bề mặt bản đồ. Một Marker gồm có bốn thuộc tính cơ bản là:   * Anchor: là điểm trên hình mà sẽ đặt vào vị trí trên bản đồ. Mặc định là 50% về phía bên trái của hình và bên cạnh trên của hình. |
|  | * Position: là giá trị LatLng cho vị trí của marker trên bản đồ. Chúng ta có thể thay đổi vị trí này bất kể khi nào chúng ta muốn di chuyển marker. * Title: Là chuỗi ký tự sẽ hiển thị trên cữa sổ khi chúng ta nhấp vào marker. * Snippet: là chuỗi ký tự hiển thị bên dưới Title. * Icon: là ảnh sẽ hiển thị đại diện cho marker. Nếu không được thiết lập thì marker sẽ sử dụng giá trị mặc định. Chúng ta không thể thay đổi thuộc tính này của marker một khi chúng ta đã tạo ra nó. * Drag status: Nếu chúng ta muốn kéo thả Marker thì thiết lập giá trị này là true.   Visibility: Mặc định giá trị này sẽ thiết lập là true, nếu muốn ẩn marker thì chuyển thành false. |
| MarkerOptions | Là class tập hợp cho các thuộc tính của Marker. |
| Polygon | Là class dùng để vẽ một hình nhiều cạnh trên bề mặt trái đất. Hình này có thể lồi hoặc lõm gồm các thuộc tính sau đây:   * Outline: là danh sách các đỉnh của hình sắp xếp theo chiều ngược của kim đồng hồ. * Holes: là vùng nằm bên trong Polygon không được lấp kín. Holes được xác đinh tương tự theo cách của Outline. Một Outline có thể chứa nhiều holes nhưng các holes này không được chồng lên nhau. * Stroke Width: độ dày của đường nối giữa các đỉnh của polygon. * Stroke color: màu của đường nối giữa các đỉnh của polygon. * Fill color: màu nền của polygon. * Visibility: dùng để thiết lập thuộc tính ẩn hiện cho polygon. * Geodesis status: xác định đường nối các đỉnh của polygon có vẽ theo địa hình hay không. |
| PolygonOptions | Là class tập hợp các thuộc tính của polygon |
| Polyline | Là class đại diện cho danh sách các điểm đươc nối với nhau bằng các đường thẳng. Các thuộc tính cơ bản của polyline bao gồm:   * Points: là danh sách các điểm được nối liên tiếp với nhau. Mặc định polyline không khép kín trừ khi điểm đầu trùng với điểm cuối. * Width: thể hiện chiều rộng của đường nối các điểm. * Color: thể hiện màu sắc của các đường nối các điểm. * Visibility: dùng để thiết lập thuộc tính ẩn hiện cho polyline. * Geodesis status: xác định đường nối các đỉnh của polyline có vẽ theo địa hình hay không. |
| PolylineOptions | Là class thể hiện các thuộc tính của polyline. |
| Tile | Là class thể hiện các thuộc tính của polyline. |
| TileOverlay | Là class dùng để hiện thị hình ảnh trên đỉnh của bản đồ. Những hình ảnh này có thể trong suốt nên cho phép chúng ta có thể thêm những tính năng vào trong bản đồ. |
| TileOverlayOptions | Là class tập hợp các thuộc tính của TileOverlay. |
| TileProvider | Là class dùng để dùng cho việc hiển thị hình ảnh trong TileOverlay. |
| UriTileProvider | Là một bộ phận của TileProvider nhưng nó chỉ yêu cầu một URL trỏ đến hình ảnh được cung cấp. |
| VisibleRegion | Là class dùng để vẽ một polyline bằng việc xác định bốn điểm. |
| CameraUpdate | Xác định sự di chuyển của camera giúp thay đổi góc nhìn của người dùng bằng cách sử dụng các phương thức như animateCamera(CameraUpdate), animateCamera(CameraUpdate,GoogleMap.CancelableCallback) hay moveCamera(CameraUpdate). |
| CameraUpdateFactory | Là class dùng để chứa các phương thức để khởi tạo một đối tượng CameraUpdate. Để thay đổi đối tượng góc nhìn của người sử dụng trong bản đồ chúng ta dùng các phương thức như: animateCamera(CameraUpdate),animateCamera(CameraUpdate, GoogleMap.CancelableCallback) hay moveCamera(CameraUpdate) |
| GoogleMap | Là một interface mà một đối tượng nào đó phải kế thừa khi muốn hiển thị và quản lý bản đồ. |
| GoogleMapOptions | Là class chứa các thiết lập để khởi tạo một đối tượng GoogleMap. |
| MapView | Dùng để hiển thị bản đồ. |
| Projection | Được sử dụng để chuyển đổi giữa vị trí trên màn hình thành tọa độ trên bản đồ. Class này chỉ gồm hai phương thức chính là:   * public LatLng fromScreenLocation (Point point): giúp chuyển một điểm trên màn hình về tọa độ vật lý trên bản đồ. * public Point toScreenLocation (LatLng location): giúp chuyển một tọa độ vật lý trên bản đồ thành tọa độ điểm ảnh trên màn hình. |
| UiSettings | Là class chứa các thiết lập về việc hiển thị các thành phần hỗ trợ trên bản đồ. |

Bảng ‑: Các Class trong GooleMap

### Google Place API Android

Google Places API cho phép chúng ta thực hiện các truy vấn thông tin về địa điểm bao gồm các địa điểm lân cận cũng như chính xác các địa điểm. Google Place API trả về danh sách các địa điểm cùng thông tin tóm gọn về mỗi địa điểm, còn thông tin chi tiết sẽ được cung cấp thông qua truy vấn Place Details. Theo [[2](#Placeholder2)] ta có thể tóm tắt các loại truy vần của Google Place API Android như sau:

#### Truy vấn thông tin gần chúng ta

Cung cấp cho chúng ta thông tin về địa điểm bên trong một khu vực được chỉ định sẵn. Chúng ta có thể chọn lọc kết quả thông qua việc cung cấp thêm các từ khóa hay loại của địa điểm mà chúng ta đang muốn tìm kiếm.

HTTP URL theo dạng:

https://maps.googleapis.com/maps/api/place/nearbysearch/*output*?*parameters*

Giá trị tham số output là định dạng văn bản của Google sẽ trả về khi chúng ta gửi yêu cầu lên server bao gồm:

* + Json: chỉ ra giá trị trả về là JavaScript Object Notation(Json)
  + Xml: chỉ ra giá trị trả về là XML.

Chúng ta cần chắc chắn cung cấp đầy đủ giá trị của các tham số quan trọng trong yêu cầu để có thể nhận được chính xác thông tin về địa điểm cần tìm, giống như URLs chuẩn thì các tham số sẽ được cách nhau bằng dấu “&”.

Những tham số yêu cầu cơ bản bao gồm:

* + Key: giá trị của tham số là API Key mà chúng ta được cung cấp thông qua việc đăng ký với google.
  + Language: giá trị của tham số là ngôn ngữ mà Google sẽ trả về.
  + Radius: giá trị của tham số là bán kính tính từ tâm của điểm mà chúng ta muốn truy vấn thông tin.
  + Sensor: giá trị của tham số này là true hay false dùng để xác định có hay không sử dụng những bộ cảm biến như GPS … để xác định vị trí.

Maps API sử dụng cho khách hàng là doanh nghiệp không bao gồm tham số chữ ký trong các truy vấn.

Ví dụ cần truy vấn các địa điểm về ẩm thực tại Sydney trong bán kính 500m kể từ vị trí hiện tại chúng ta sẽ sử dụng câu truy vấn sau:

https://maps.googleapis.com/maps/api/place/nearbysearch/json?location=33.8670522,151.1957362&radius=500&types=food&name=harbour&sensor=false&key=*AddYourOwnKeyHere*

#### Text Search Request

Google Places API Text Search Service là một dịch vụ web dùng để trả lại thông tin về thông tin của một địa điểm cụ thể. Dịch vụ sẽ trả lại danh sách địa điểm phù hợp với truy vấn. Chúng ta có thể sử dụng Google Detail để có thêm được những thông tin cụ thể về các địa điểm được trả về.

* Truy vấn của Place Text Search là một HTTP URL theo dạng:

https://maps.googleapis.com/maps/api/place/textsearch/*output*?*parameters*

* Giá trị của tham số output xác định kiểu của kết quả trả về là Json hay xml.
* Các tham số yêu cầu cơ bản bao gồm:
  + Query: là một chuỗi truy vấn theo kiểu String dùng để tìm kiếm địa điểm.
  + Key và sensor: là những tham số tương tự với Nearby Search Request.
* Ví dụ chúng ta muốn tìm khách sạn tại Sydney thì chuỗi truy vấn sẽ như sau:

https://maps.googleapis.com/maps/api/place/textsearch/xml?query=restaurants+in+Sydney&sensor=true&key=*AddYourOwnKeyHere*

#### Radar Search Request

Google Places API Radar Search Service cho phép chúng ta tìm trên 200 địa điểm một lần, tuy nhiên thông tin về những địa điểm này sẽ ít hơn bình thường so với sử dụng Text Search hay Nearby Search. Với Radar Search giúp người sử dụng có thể xác định khu vực yêu thích của mình trên bản đồ. Để có thêm thông tin cụ thể về các địa điểm trong danh sách trả về chúng ta có thể sử dụng Google Detail.

* Truy vấn của Radar Search Request là một HTTP URL theo dạng:

https://maps.googleapis.com/maps/api/place/radarsearch/*output*?*parameters*

* Giá trị của tham số output xác định kiểu của kết quả trả về là Json hay xml.
* Các tham số cơ bản bao của Radar Search tương tự như Nearby Search.
* Ví dụ chúng ta muốn tìm kiếm những quán cà phê có tên là vegetarian trong khu vực có bán kính 5000m thì chuỗi truy vấn sẽ như sau:

https://maps.googleapis.com/maps/api/place/radarsearch/json?location=48.859294,2.347589&radius=5000&types=food|cafe&sensor=false&keyword=vegetarian&key=*AddYourOwnKeyHere*

#### Kết quả truy vấn được trả về từ Google Search

* Định dạng của kết quả trả về được chỉ định thông qua tham số output trong truy vấn đến Google Search.
* Kết quả trả về của TextSearch tương tự như NearbySearch ngoại trừ thuộc tính formatted\_address thay thế bằng vicinity.

Một Json trả về thì gồm 4 thành phần chính là:

* “status” chứ metadata của yêu cầu.
* “results” chứa mảng cách địa điểm tra về với thông tin cụ thể của từng địa điểm.
* “http\_attributions” chứa các thiết lập về thẩm quyền của danh sách các địa điểm đang được hiển thị.
* “next\_page\_token” chứa token dùng để hiển thị cho 20 kết quả tiếp theo. Để hiển thị thêm các kết quả này thì chúng ta chỉ cần truyền giá trị này vào tham số pagetoke. Tối đa cho một kết quả trả về từ Goole là 60 địa điểm và đươc chia thành 3 trang khác nhau.

### Google Direction API Android

Google Directions API là một dịch vụ dùng để tính toán đường đi giữa hai điểm trên bản đồ sử dụng các truy vấn HTTP. Chúng ta có thể tìm kiếm đường đi theo các chế độ khác nhau như: xe buýt, đi bộ, xe ôtô. Đường đi được xác định thông qua tọa độ của điểm đầu và điểm cuối được nhập vào thông qua một chuỗi ký tự hay tọa độ latitude/longitude. Google Direction API có thể trả lại nhiều đường đi khác nhau dựa trên danh sách các điểm mốc. Dù là một dịch vụ miễn phí nhưng đối với một người dùng thông thường thì chỉ có thể sử dụng 2500 truy vấn trong một ngày, còn đối với người dùng trong kinh doanh thì có thể sử dụng 100000 truy vấn trong một ngày.

Cấu trúc truy vấn của Google Directions API theo dạng:

http://maps.googleapis.com/maps/api/directions/*output*?*parameters*

Giá trị của biến output là Json hay XML.

Các tham số yêu cầu bao gồm:

* Origin: Vị trí điểm đầu của đường đi. Giá trị tham số có thể là tọa độ latitude/longitude hoặc địa chỉ. Chúng ta cần chú ý rằng trong chuỗi giá trị của tham số này không được sử dụng khoảng trắng.
* Destination: Vị trí điểm cuối của đường đi. Giá trị tham số có thể là tọa độ latitude/longitude hoặc địa chỉ. Chúng ta cần chú ý rằng trong chuỗi giá trị của tham số này không được sử dụng khoảng trắng.
* Sensor: Xác định có hay không sử dụng bộ cảm biến cho việc tính toán đường đi.

Các tham số tùy chọn cơ bản bao gồm:

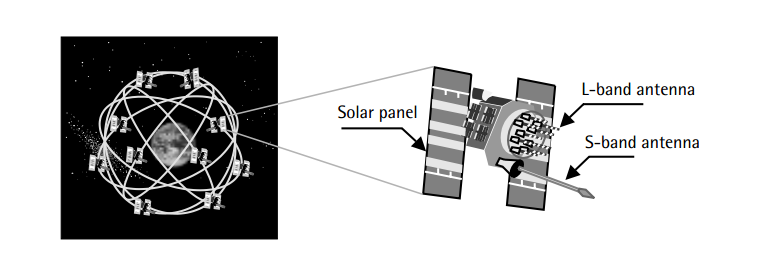
* Mode: dùng để xác định loại của tuyến đường cần tin. Google cung cấp 4 loại chính bao gồm:
  + Driving: loại dùng để tìm đường cho xe ô tô hay các phương tiện lớn.
  + Walking: loại dùng để tìm đường cho người đi bộ.
  + Bicycling: loại dùng để tìm đường cho xe đạp.
  + Transit: loại dùng để tìm đường cho các phương tiện công cộng.
* Waypoints: là danh sách các điểm cố định mà tuyến đường cần phải đi qua.
* Avoid: tham số đưa vào khi tìm đường sẽ tránh bao gồm 2 giá trị là:
  + Tolls: tránh những chỗ có thu phí.
  + Highways: tránh đường cao tốc.
* Alternatives: tham số này xác định có hay không việc hiển thị nhiều hơn một tuyến đường cho người dùng lựa chọn.
* Language: xác định ngôn ngữ của hướng dẫn chỉ đường được trả về. Mặc định sẽ là tiếng Anh.

## Công nghệ GPS

Hệ thống định vị toàn cầu (GPS) là một hệ thống vệ tinh điều hướng được phát triển bởi bộ quốc phòng mỹ trong những năm đầu của thập niên 70. Ban đầu, GPS được phát triển như là một hệ thống được sử dụng trong mục đích quân sự. Tuy nhiên, thời gian sau nó được sử dụng thêm vào các mục đích dân sự.

GPS cung cấp liên tục thông tin vị trí và thời gian bất kỳ đâu trên thế giới trong bất kỳ điều kiện thời tiết nào. Nó phục vụ một không hạn chế người dùng cũng như được sử dụng cho các lý do an ninh. GPS là hệ thống một chiều thụ động có nghĩa là những người dùng chỉ có thể nhận tín hiệu từ vệ tinh mà không thể gửi trả. Để sử dụng GPS người dùng chỉ cần có thiết bị thu sóng và không phải trả bất kỳ chi phí nào khác. Sau khi tổng hợp từ [[3](#Placeholder3)] chúng ta có thể đưa ra một số khái niệm cơ bản vể GPS như dưới đâu.

### Tổng quan về GPS

GPS bao gồm một 24 vệ tinh được liên kết với nhau. Hệ thống này được hoàn thành vào tháng 7 năm 1993 và công bố chính thức vào ngày 8 tháng 12 năm 1993. Nó được xem là cơ sở quan trọng của hệ thống GPS với tên gọi “Năng lực hoạt động ban đầu” gọi tắt là IOC(initial operational capability). Để đảm bảo hệ thống vệ tinh này bao phủ toàn bộ trái đất một cách liên tục thì 24 vệ tinh được sắp xếp sao cho cứ mỗi 4 vệ tinh được đặt trong 6 mặt phẳng quỹ đạo (Hình 2-8). Với sự bố trí này sẽ có trên 4 vệ tinh luôn hiện hữu ở bất kỳ vị trí nào trên trái đất, nếu góc ngẩng là 10o. Để định vị tại một vị trí ta cần tối thiểu 4 vệ tinh.

Hình ‑: GPS

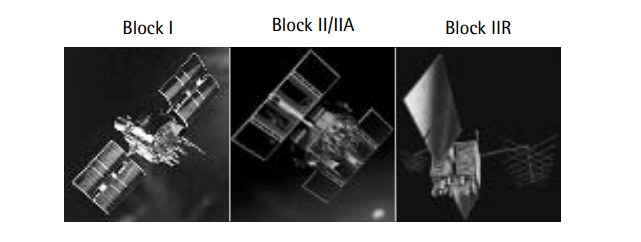
Quỹ đạo của các vệ tinh GPS gần như là hình tròn (Một hình elipse với tâm sai cực đại khoảng 0.01) với độ nghiêng khoảng 550 so với xích đạo. Nửa trục lớn của quỹ đạo GPS có độ dài khoảng 26560 km, tức là vệ tinh có độ cao khoảng 20200km bên trên bề mặt trái đất. Chu kỳ quỹ đạo vệ tinh tương ứng khoảng 12 giờ thiên văn.

Hệ thống GPS đạt được khả năng hoạt động đầy đủ gọi tắt là FOC(full operational capability) vào ngày 17 tháng 7 năm 1995 đảm bảo khả năng hoạt động thực tế tối thiểu 24 vệ tinh GPS. Thực tế thì khi GPS đạt FOC thì trên quỹ đạo hoạt động thường có nhiều hơn 24 vệ tinh.

### Những phân đoạn GPS

GPS bao gồm 3 phân đoạn chính là: phân đoạn không gian, phân đoạn điều khiển, phân đoạn người dùng.

* Phân đoạn không gian chính là nhóm gồm 24 vệ tinh, mỗi vệ tinh GPS phát đi tín hiệu bao gồm những thành phần sau: hai sóng sin là thành phần sóng mang, hai chuỗi dữ liệu số, và một thong điệp điều hướng, dữ liệu số và thông điệp điều hướng. Sóng mang và chuỗi dữ liệu số chủ yếu dùng để xác định khoảng cách từ máy thu của người sử dụng đến những vệ tinh GPS. Thông điệp điều hướng bao gồm tọa độ vệ tinh được biểu diễn dưới hàm biến đổi theo thời gian và các thông tin cần thiết khác. Những tín hiệu này được phát đi bởi các đồng hồ nguyên tử có độ chính xác cao được gắn trên những vệ tinh.
* Phân đoạn điều khiển hệ thống GPS bao gồm một mạng lưới rộng khắp những trạm theo dõi và một trạm điều khiển đặt tại bang Colorado của Mỹ. Nhiệm vụ của phân đoạn này là theo dõi, quản lý các vệ tinh GPS bao gồm vị trí của vệ tinh, tình trạng hệ thống, hoạt động của các đồng hồ nguyên tử.. và một số vấn đề khác. Những thông tin này được truyền tải trong thông tin về vị trí của vệ tinh và được chứa trong máy thu, khi vệ tinh di chuyển thì các thông tin này liên tục được cập nhật lại. Những thông tin này sau đó được đóng gói lại và gửi lên vệ tinh.
* Phân đoạn người dùng bao gồm tất cả người dân và quân đội. Với một bộ thu được kết nối với anten GPS, một người dùng có thể nhận tín hiệu GPS, tín hiệu này dùng để xác định vị trí mà không cần quan tâm vị trí của người đó đang ở đâu. GPS là một dịch vụ được cung cấp cho tất cả người sử dụng mà không phải trả thêm một chi phí nào.



Hình ‑: 3 phân đoạn chính của GPS

### Những thế hệ của vệ tinh GPS

Trong hệ thống ban đầu, nhóm vệ tinh GPS được xây dựng với chuỗi gồm 11 vệ tinh, gọi là những Block I. Vệ tinh đầu tiên trong chuỗi này bắt đầu hoạt động đầu tiên vào ngày 22 tháng 2 năm 1978, cái cuối cùng hoạt động vào ngày 9 tháng 10 năm 1985. Block I chủ yếu được sử dụng với mục đích chủ yếu là thí nghiệm. Góc nghiêng mặt phẳng quỹ đạo của vệ tinh này so với đường xích đạo là 630, sau đó được sửa đổi ở các thế hệ sau. Block I cuối cùng được rút ra vào ngày 18 tháng 11 năm 1995.

Thế hệ thứ 2 của các vệ tinh GPS được biết tới với tên gọi là Block II/IIA. Block IIA là phiên bản nâng cấp của khối II, với việc tăng khả năng lưu trữ dữ liệu thông điệp điều hướng từ 14 ngày đối với khối II đến 180 ngày đối với khối IIA, điều này giúp tang khả năng hoạt động độc lập của vệ tinh mà không cần sự hỗ trợ từ mặt đất. Có tổng cộng 28 Block II/IIA đã hoạt động trong khoảng thời gian từ tháng 2 năm 1989 đến tháng 11 năm 1997. Mặt phẳng nghiêng của Block II/IIA so với đường xích đạo là 550, thời gian hoạt động trong không trung là 7.5 năm nhưng thường chúng tồn tại trên không gian lâu hơn thế rất nhiều. Block II/IIA được tăng cường khả năng bảo mật hơn so với các phiên bản trước bằng việc thêm vào khả năng lựa chọn người dùng và chống lừa đảo trong cấu trúc tín hiệu truyền về máy thu.

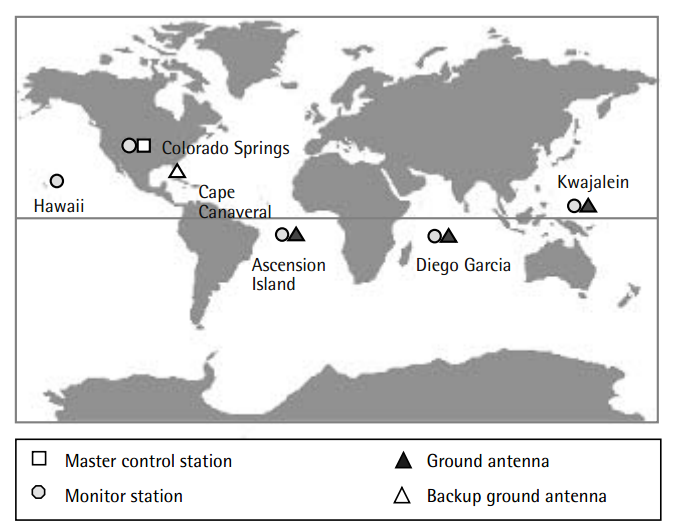
Bản nâng cấp tiếp theo của Block IIA là Block IIR, nó có khả năng tương thích với các Block thế hệ trước đó. Block IIR bao gồm 21 vệ tinh với thời gian sống được thiết kế là 10 năm. Thêm vào đó là độ chính xác cao hơn trước, những Block IIR có khả năng hoạt động độc lập cao nhờ giảm được phương sai của độ chính xác theo thời gian. Khả năng định vị của Block này được nâng cao nhờ chúng có khả năng tự sắp xếp lẫn nhau. Vào tháng 7 năm 2001, 6 vệ tinh thuộc Block IIR đã hoạt động thành công.

Theo sau Block IIR là Block IIF bao gồm 33 vệ tinh, khoảng thời gian sống của vệ tinh là 15 năm. Những vệ tinh thuộc khối này được tăng cường tính chính xác trong định vị.

Hệ thống Block III là hệ thống mới nhất được thiết kế và phát triển bởi tập đoàn Lockheed Martin[[20]](#footnote-20) theo đơn đặt hàng của bộ quốc phòng mỹ, vòng đời của vệ tinh thuộc khối này là 15 năm. Hiện tại đang có 4 vệ tinh đang được phát triển và đến năm 2014 sẽ nâng lên thành 35 vệ tinh. Các vệ tinh này được thiết kế với trọng lượng nhẹ, độ bền cao bằng vật liệu composite[[21]](#footnote-21). Sự thay đổi đầu tiên là trong việc truyền tín hiệu GPS dành cho dân sự sẽ không sử dụng đường truyền có tần số L1 mà thay vào đó bằng tần số L2 (1227,6 MHz), nó được sử dụng cho các Block IIR và các Block sau này. Tín hiệu truyền trên tần số L2 tăng thêm độ chính xác về vị trí cũng như tăng khả năng dẫn đường nhờ vào hệ thống tín hiệu dự phòng giảm khả năng bị nhiễu và mất sóng.

### Những vị trí điều khiển

Phân đoạn điều khiển của GPS bao gồm một trạm điều khiển chính và rộng khắp những trạm giám sát và những trạm điều khiển mặt đất.



Hình ‑: Phân đoạn điều khiển của GPS

Có 5 trạm giám sát và định vị tại Colorado Springs, Hawaii, Kwajalein, Diego Garcia, và đảo Ascension. Vị trí hoặc tọa độ của những trạm giám sát này được xác định chính xác. Những trạm này có mục đích theo dấu vết liên tục tất cả những vệ tinh GPS trong tầm nhìn, upload thông tin, điều chỉnh quản lý hoạt động của các vệ tinh.

### GPS-một vài khái niệm cơ bản

Mỗi vệ tinh GPS phát liên tục một tín hiệu vô tuyến tạo thành từ tổng thể 2 sóng mang[[22]](#footnote-22), 2 mã và một thông điệp điều hướng, bộ thu GPS phân tích thông tin này thu được bao gổm các khoảng cách tới những vệ tinh GPS và tọa độ của vệ tinh xác định qua thông điệp điều hướng.

Theo lý thuyết chỉ duy nhất cần 3 khoảng cách đến 3 vệ tinh mà được theo dấu vết một cách đồng thời. Trong trường hợp này, bộ thu sẽ được định vị tại chỗ giao nhau của ba hình cầu, mỗi hình cầu này có một bán kính tương ứng với khoảng cách vệ tinh – bộ thu và tâm vệ tinh. Tuy nhiên cần đến 4 vệ tinh để mô tả độ lệch giữa đồng hộ bộ thu và đồng hồ vệ tinh. Độ chính xác của phương pháp này là giới hạn trong 100m theo chiều ngang, 156m theo chiều dọc, và 340ns theo thành phần thời gian. Độ chính xác này tương đối thấp vì vậy phương pháp này không còn được sử dụng và được thay thế bằng phương pháp vi sai, trong đó sử dụng hai bộ thu theo dấu vết đồng thời cùng một vệ tinh. Trong trường hợp này, có thể đạt được độ chính xác từ vài centimet đền vài mét.

Lợi ích khác của GPS là khả năng mô tả vận tốc của người sử dụng thông qua một vài phương pháp. Phương pháp được sử dụng rộng rãi nhất là đánh giá tần số Doppler[[23]](#footnote-23) của tín hiệu GPS thu được. Độ dịch Doppler được xem như là chuyển động tương đối giữa bộ thu và vệ tinh.

### Nguyên lý hoạt động của GPS

#### Hoạt động cơ bản

Các vệ tinh GPS bay vòng quanh Trái Đất hai lần trong một ngày theo một quỹ đạo rất chính xác và phát tín hiệu có thông tin xuống Trái Đất. Các máy thu GPS nhận thông tin này và bằng phép tính lượng giác tính được chính xác vị trí của người dùng. Về bản chất máy thu GPS so sánh thời gian tín hiệu được phát đi từ vệ tinh với thời gian nhận được chúng. Sai lệch về thời gian cho biết máy thu GPS ở cách vệ tinh bao xa. Rồi với nhiều quãng cách đo được tới nhiều vệ tinh máy thu có thể tính được vị trí của người dùng và hiển thị lên bản đồ điện tử của máy. Máy thu phải nhận được tín hiệu của ít nhất ba vệ tinh để tính ra vị trí hai chiều (kinh độ và vĩ độ) và để theo dõi được chuyển động. Khi nhận được tín hiệu của ít nhất 4 vệ tinh thì máy thu có thể tính được vị trí ba chiều (kinh độ, vĩ độ và độ cao). Một khi vị trí người dùng đã tính được thì máy thu GPS có thể tính các thông tin khác, như tốc độ, hướng chuyển động, bám sát di chuyển, khoảng hành trình, quãng cách tới điểm đến, thời gian Mặt Trời mọc, lặn và nhiều thứ khác nữa.

#### Độ chính xác của GPS

Độ chính xác của các thiết bị GPS ngày nay được tăng lên đáng kể, nhờ vào thiết kế hoạt động song song nhiều kênh của nó. Các máy thu 12 kênh song song nhanh chóng bám sát các quả vệ tinh khi vừa mới phát tín hiệu và chúng duy trì chắc chắn quan hệ này dù là trong ở rừng rậm rạp hay trong các tòa nhà cao tầng. Độ chính xác của các máy GPS trung bình khoảng 15 mét, các yếu tố ảnh hưởng đến độ chính xác này là trạng thái khí quyển và các yếu tố gây nhiễu trong không khí khác.Các mày thu phát sóng được trang bị khả năng WAAS (Wide Area Augmentation System) có thể tăng độ chính xác trung bình dưới 3m. WAAS là một dịch vụ sửa lỗi để tăng độ chính xác của các thiết bị GPS. Hệ thống bao gồm một mạng các đài thu tín hiệu GPS và phát tín hiệu đã sửa lỗi bằng các máy phát hiệu chỉnh. Để thu được tín hiệu đã sửa lỗi, người dùng phải có máy thu tín hiệu vi sai bao gồm cả ăn-ten để dùng cùng với máy thu GPS của họ.

## PHP, MySQL, APACHE

### PHP và Yii Framework

#### PHP và mô hình MVC

**PHP**

PHP (PHP: Hypertext Preprocessor) là một ngôn ngữ lập trình kịch bản hay một loại mã lệnh chủ yếu được dùng để phát triển các ứng dụng viết cho máy chủ, mã nguồn mở, dùng cho mục đích tổng quát. Nó rất thích hợp với web do được tối ưu hóa cho các ứng dụng web, tốc độ nhanh, nhỏ gọn, cú pháp giống C và Java, dễ học và thời gian xây dựng sản phẩm tương đối ngắn hơn so với các ngôn ngữ khác nên PHP đã nhanh chóng trở thành một ngôn ngữ lập trình web phổ biến nhất thế giới.  
 PHP được phát triển từ một sản phẩm có tên là PHP/FI. PHP/FI do Rasmus Lerdorf tạo ra năm 1995, ban đầu được xem như là một tập con đơn giản của các mã kịch bản Perl để theo dõi tình hình truy cập đến bản sơ yếu lý lịch của ông trên mạng. Ông đã đặt tên cho bộ mã kịch bản này là ‘Personal Home Page Tools’.

**Mô hình MVC**

MVC được khai sinh với framework được phát triển bởi Tryve Reenskaug cho nền tảng Smalltalk vào cuối những năm 70. Từ đó trở đi nó đóng vai trò là một qui tắc quan trọng trong hầu hết các framework giao diện người dùng và trong các ý tưởng thiết kế giao diện người dùng. Một số thông tin cơ bản về MVC:

* Tương tác giữa các thành phần.

Ngoài việc chia ứng dụng thành ba thành phần, thiết kế MVC xác định các tương tác giữa chúng.

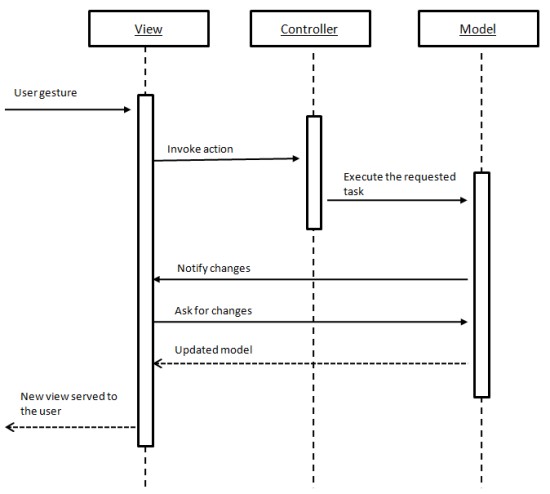
* Một Controller có thể gửi các lệnh đến các view liên kết đến nó để thay đổi cách hiển thị của view. Nó có thể gửi các lệnh đến model để cập nhật trạng thái của model.
* Một model thông báo cho các view và các controller liên quan với nó khi có sự thay đổi trạng thái. Việc thông báo cho phép các view xử lý các output đã thay đổi, và các controller thay đổi các lệnh có thể thực thi. Một sự thực thi MVC thụ động bỏ qua các thông báo này, bởi vì ứng dụng không yêu cầu chúng hoặc nền tảng phần mềm không hỗ trợ.
* Một view thu thập thông tin từ model để tạo ra một giao diện đầu ra.



* Cách sử dụng MVC trong các ứng dụng web.

MVC được xem như nhắm tới để trở thành một kiến trúc cho các ứng dụng World Wide Web. Một vài framework ứng dụng thương mại và phi thương mại đã được tạo ra với việc sử dụng MVC là bắt buộc. Theo cách tiếp cận này, các client gửi các request hyperlink hoặc form input đến controller và sau đó nhận về một trang web đầy đủ và cập nhật (Hoặc một tài liệu khác) từ view; thành phần model tồn tại hoàn toàn trên server. Khi các công nghệ client đã phát triển, các frame workd như JavaScriptMVC và Backbone đã được tạo ra cho phép các thành phần MVC được thực thi một phần trên client.

* MVC làm việc như thế nào.
* Nhìn lại sơ đồ phía trên, ta thấy có mũi tên nét liền và những mũi tên nét đứt. Những mũi tên nét đứt được hình thành trên quan điểm của người dùng mà không phải là của những nhà thiết kế phần mềm thực sự. Do đó chúng ta chỉ quan tâm đến những mũi tên còn lại.  
   Đây là một cách đơn giản để mô tả lại luồng sự kiện được xử lý trong MVC:
* User tương tác với View, bằng cách click vào button, user gửi yêu cầu đi.
* Controller nhận và điều hướng chúng đến đúng phương thức xử lý ở Model.
* Model nhận thông tin và thực thi các yêu cầu.
* Khi Model hoàn tất việc xử lý, View sẽ nhận kết quả từ Model và hiển thị lại cho người dùng.



**MVC và PHP**

Mô hình MVC là một cách chia nhỏ ứng dụng, thậm chí là một phần của giao diện của ứng dụng thành 3 phần: model, view và controller. MVC được phát triển ban đầu để ánh xa các quá trình truyền thống như: nhập, xử lý , xuất thành các lĩnh vực GUI:

Input 🡪 Processing 🡪 Output

Controller 🡪 Model 🡪 View.

* **Model.**

Một model là một đối tượng đại diện cho dữ liệu hoặc một hành động như: một bảng trong CSDL .

Model quản lý hành vi hoặc dữ liệu của miền ứng dụng, phản hồi lại các yêu cầu truy xuất thông tin về trạng thái của nó và phản hồi lại các chỉ thị để thay đổi trạng thái.

Model là một phần đại diện cho trạng thái và các hành vi mức thấp của component. Nó quả lý trạng thái và thực hiện các sự chuyển đổi trạng thái. Model không có tri thức cụ thể về controller hoặc view của nó. View là một phần quản lý sự hiên thị hình ảnh của trạng thái được đưa ra bởi model. Một model có thể có nhiều hơn 1 view.

Chú ý rằng model không cần thiết có một database cố định. Nó có thể truy cập qua một đối tượng truy cập dữ liệu (DAO) riêng biệt.

* **View.**

Một view là một số form hiện thị trạng thái của model.

View quản lý việc xuất ra các dữ liệu đồ họa hay text

View chỉ ra nội dung của một model. Nó truy câp vào CSDL qua model và xác định cách dữ liệu được đưa ra.

* **Controller.**

Một controller cung cấp các cơ sở đề thay đổi trạng thái của model. Controller dịch các input từ chuột và bàn phím của người dung, truyền lệnh đến model hay view để thay đổi cho phù hợp.

Một controller là các phương thức mà người dung tương tác với ứng dụng. Một controller cho phép nhập thông tin từ người dung và chỉ thị cho model và view thực hiện các hành động dựa trên đầu vào. Trong thực tế, controller có thể phản hồi cho việc ánh xạ các action của người cuối đến các phản hồi của ứng dụng.

Controller dịch các tương tác với view thành các hành động để thực thi bởi model. Trong một GUI client đơn lẻ, các tương tác người dung có thể là các click vào button và nhiều sự lựa chọn trong khi một ứng dụng web xuất hiện như một request HTTP GET hoặc POST. Các action được thực hiện bởi model bao gồm việc kích hoạt các quá trình nghiệp vụ hoặc thay đổi trạng thái của model. Dựa trên các tương tác người dung và đầu ra của các tương tác với model, controller phản hồi bằng việc xuất ra một view phù hợp.

Controller là thành phẩn quản lý tương tác người dung với model. Nó cung cấp các cơ chế để các thay đổi trạng thái của model được tạo ra.

#### Yii framework

Yii là một PHP MVC Framework mã nguồn mở hoàn toàn miễn phí, có hiệu năng cao, giúp bạn phát triển tốt các ứng dụng Web. Yii cho phép tái sử dụng tối đa các thành phần (components) của ứng dụng để tăng tốc độ viết ứng dụng.

Yii là từ viết tắt của từ *Yes, it is!* Yii được phát âm là \i:\ , vì thế nó cũng được hiểu là viết ngắn gọn 3 chữ cái đầu của Easy (dễ dàng), Efficient (hiệu suất) và Extensible (khả năng mở rộng).

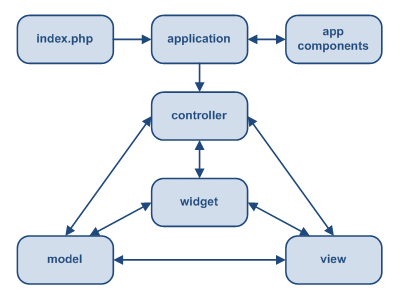
Ưu điểm của Yii là tính đơn giản trong cách viết kết hợp với thế mạnh “scripting” của PHP và cơ chế mảng Index (Associative Array).

Yii là một MVC framework  nhưng có kiến trúc OOP rất tốt và định hướng theo component nên mức độ tái sử dụng giữa các project rất cao. Yii đặc biệt chạy nhanh hơn Zend và Symfony.

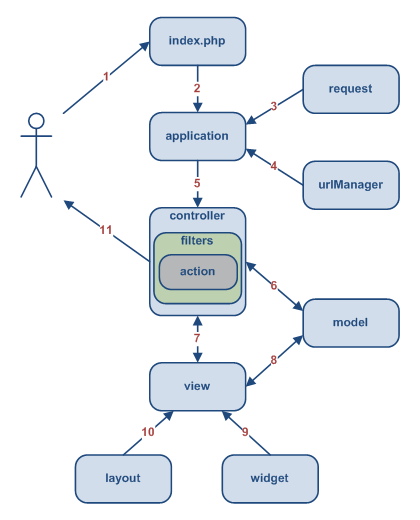
Tiền thân của Yii Framework chính là PRADO Framework. Tác giả của Yii Framework là Qiang Xue, 1 người Trung Quốc. Qiang Xue bắt đầu xây dựng Yii Framework vào khoảng 01/01/2008. Có nghĩa là tính tới nay đã được 6 năm. Qiang Xue cũng chính là người đã từng phát triển và maintain PRADO, tuy nhiên, nhận thấy PRADO có nhiều hạn chế và bản thân Qiang Xue muốn phát triển 1 framework mang những đặc điểm như dễ sử dụng, dễ tiếp cận, bảo mật cao và chuyên nghiệp hơn nên QX đã sử dụng kinh nghiệm của mình trong thời gian phát triển PRADO để viết ra Yii. Sau gần 1 năm phát triển, tới 08/12/2008, bản Yii Framework đầu tiên đã được public và nhanh chóng được mọi người đón nhận.

Đặc điểm của yii:

* Yii triển khai trên mô hình thiết kế MVC, được thực hiện rộng rãi trong lập trinh web. MVC giúp nhà phát triển có thể dễ dàng thay đổi các phần mà không thay đổi phần khác. Trong MVC, model đưa ra thông tin và các luật, view chưa các yếu tố của giao diện người dùng, và controller quản lý tương tác gữa model và view.
* Bên cạnh việc triển khai MVC, Yii cũng giới thiệu 1 front-controller được gọi là Application, nó đóng gói ngữ cảnh thực thi việc xử lý các yêu cầu. Application thu thập một vài thông tin về một yêu cầu người dùng và sau đó gửi nó vào 1 controller phù hợp cho việc trao đổi sau này.
* Sơ đồ sau cho thấy cấu trúc tĩnh của 1 ứng dụng Yii



* Quy trình làm việc cụ thể:



1. Một người sử tạo một request với URL: <http://example.com/index.php?r=post/show&id=1> và web server trao đổi yêu cầu bằng việc thực thi bootstrap script index.php

2. Bootstrap script tạo ra một thực thể Application và chạy chúng.

3. Application lấy thông tin chi tiết về yêu cầu người dùng từ một thanh phần của application được đặt tên là request.

4. Application xác controller được request và action với sự giúp đỡ của một thành phần application tên là urlManager. Trường hợp này controller là post sẽ được thay thế bằng lớp PostController và action là show, phương thức thực tế của nó sẽ được xác đinh bưởi controller.

5. Application tạo ra một thực thể của controller được request để trao đổi request của người dùng sau này. Controller xác định action show tương ứng phương thức actionShow trong lớp controller. Nó sau đó sẽ tạo ra và thực thi bộ lọc liên kết với action này. Action được thực thi nếu nó được cho phép bởi bộ lọc.

6. Action đọc một model có id=1 trong cơ sở dữ liệu.

7. Action trả lại môt view có tên show với model post.

8. View đọc và hiển thị các thuộc tính của model post.

9. View thực thi một vài widget (thành phần được nhúng vào một view script để tạo ra giao diện người dùng phức tạp).

10. View trả lại kết quả được nhúng vào trong một layout (Một view đặc biệt được sử dụng để trang trí các view, thường là 1 phần của giao diện người dùng nằm giữa các view).

11. Action hoàn tất việc trả lại view và hiển thị kết quả tới người dùng.

### MYSQL

MySQL là hệ quản trị cơ sở dữ liệu mã nguồn mở phổ biến nhất trên thế giới. Facebook, Twitter, Wikipedia, YouTube hiện đang sử dụng hệ quản trị cơ sở dữ liệu này. Hầu hết các cơ sở dữ liệu quan hệ một vài thập kỷ trở lại đây đều được thiết kế hướng đến việc xây dựng các ứng dụng ERP phức tạp thì MySQL được thiết kế và tối ưu hóa hỗ trợ cho việc xây dựng những ứng dụng Web. Với hiệu suất khả năng mở rộng, độ tin cậy cao cũng như dễ dàng trong sử dụng, tiết kiệm được chi phí mà MySQL đã trở thành hệ quản trị cơ sở dữ liệu mặc định với các ứng dụng Web-based đối với đa số các lập trình viên.

Một điểm mạnh nữa của MySQL là phần mềm mã nguồn mở miễn phí, rất dễ dàng triển khai trên các hệ điều hành khác nhau như Windows, Linux, Mac OS X, Unix …

Hiện tại MySQL được triển khai với những cải thiện đáng kể như: tăng hiệu suất, khả năng mở rộng, tăng độ khả dụng với người dùng, kiến trúc đa xử lý về phần cứng, phần mềm … MySQL là sự kết hợp giữa cơ sở dữ liệu MySQL và InnoDB storage engine (từ phiên bản MYSQL 5.5) cung cấp cho chúng ta một giải pháp quản lý dữ liệu mạnh mẽ với những ưu điểm sau:

- InnoDB storage engine là mặc định trong MySQL 5.5

- Tăng hiệu suất và khả năng mở rộng trên Windows.

- Tương thích, tận dụng được tối đa hiệu suất với các hệ thống kiến trúc đa xử lý trên các nền tảng khác nhau.

- Tăng khả năng đáp ứng, hiệu quả, khả năng quả lý, dễ dàng hơn cho người sử dụng...

Với InnoDB mặc định trong MySQL, những bảng được tạo mới sẽ có Engine Storage mặc định là InnoDB giúp cho phép lập trình các ứng dụng Transaction mà không cần phải thay đổi lại cấu hình. Ngoài ra InnoDB đã được tái cấu trúc giúp tối ưu hóa hiệu suất bằng cách tận dụng tối đa công suất các hệ thống phần cứng đa lõi, đa luồng. Sau đây là các kết quả có được khi so sánh giữa MySQL5.5 và MySQL 5.1:

- Windows: tăng 1500% hiệu suất khi đọc và ghi dữ liệu, 500% khi chỉ đọc.

- Linux: tăng 360% hiệu suất khi dọc và ghi dữ liệu, 200% khi chỉ đọc.

Chính vì những ưu điểm đó mà chúng em quyết định chon MySQL để triển cơ sở dữ liệu cho hệ thống này.

### Apache HTTP server

**Apache** hay là **chương trình máy chủ HTTP** là một chương trình dành cho máy chủ đối thoại qua giao thức HTTP. Apache chạy trên các hệ điều hành tương tự như Unix, Microsoft Windows, Novell Netware và các hệ điều hành khác. Apache đóng một vai trò quan trọng trong quá trình phát triển của mạng World Wide Web.

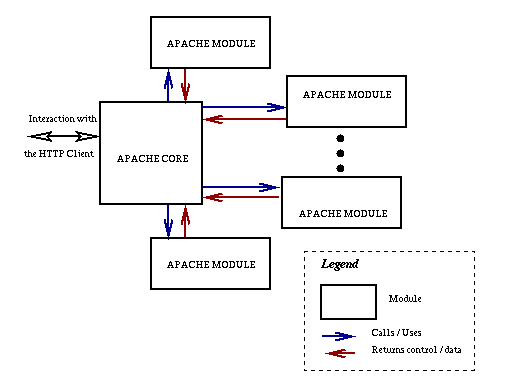
Khi được phát hành lần đầu, Apache là chương trình máy chủ mã nguồn mở duy nhất có khả năng cạnh tranh với chương trình máy chủ tương tự của Netscape Communications Corporation mà ngày nay được biết đến qua tên thương mại Sun Java System Web Server. Từ đó trở đi, Apache đã không ngừng tiến triển và trở thành một phần mềm có sức cạnh tranh mạnh so với các chương trình máy chủ khác về mặt hiệu suất và tính năng phong phú. Từ tháng 4 năm 1996, Apache trở thành một chương trình máy chủ HTTP thông dụng nhất. Hơn nữa, Apache thường được dùng để so sánh với các phần mềm khác có chức năng tương tự. Tính đến tháng 1 năm 2007 thì Apache chiếm đến 60% thị trường các chương trình phân phối trang web (xem bài tiếng Anh)

Apache được phát triển và duy trì bởi một cộng đồng mã nguồn mở dưới sự bảo trợ của Apache Software Foundation. Apache được phát hành với giấy phép Apache License và là một phần mềm tự do và miễn phí.

#### Kiến trúc Apache

Apache là một Web Server được xây dựng theo cấu trúc module, tức là [Apache](http://dinhchithanh.com/tag/apache/) là một tập hợp nhiều thành phần nhỏ được gắn kết với nhau. Các thành phần nhỏ này được phân ra thành 2 loại chính là Apache Core (thành phần lõi) và Apache Modules

Tổng quát, cấu trúc của Apache được trình bày trong sơ đồ dưới đây:



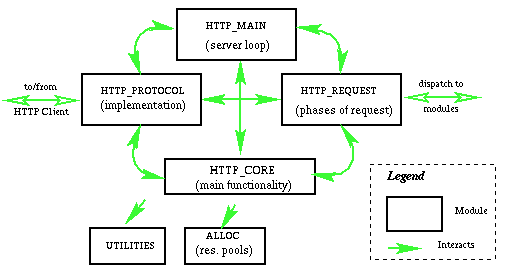
Hình ‑: Kiến trúc Apache

#### Thành phần lõi của apache

Tất cả các chức năng cơ bản nhất của Apache Web Server được cài đặt trong thành phần này. Chính thành phần này sẽ cung cấp các tài nguyên phù hợp cho từng request mà Web Server nhận được. Lõi của Apache được cấu thành từ các thành phần nhỏ sau:

* **http\_protocol.c**: chứa các thủ tục làm nhiệm vụ giao tiếp trực tiếp với các client thông qua socket bằng giao thức HTTP. Tất cả các công việc trao đổi dữ liệu với client đều do thành phần này đảm trách.
* **http\_main.c:** thành phần làm nhiệm vụ khởi động server và tạo vòng lặp chính để đợi và chấp nhận các kết nối. Đồng thời cũng làm nhiệm vụ quản lý các bộ thời gian timeout.
* **http\_request.c:** thành phần này làm nhiệm vụ quản lý các tiến trình xử lý bản tin request, đảm bảo chuyển các bản tin điều khiển tới các module phù hợp theo đúng thứ tự. Ngoài ra, nó còn đảm nhận vai trò quản lý các lỗi xảy ra trên server.
* **http\_core.c:** thành phần này sẽ triển khai các chức năng cơ bản nhất của Apache.
* **Alloc.c:** thành phần kiểm soát việc phân chia tài nguyên và lưu trữ các thông tin về sự phân chia đó.
* **http\_config.c:** đảm nhận vai trò xử lý file cấu hình và hỗ trợ cho các virtual host. Một trong những chức năng quan trọng của thành phần này là liệt kê những module được sử dụng trong Apache.

Kiến trúc chi tiết của lõi Apache được thể hiện trong sơ đồ sau:



Hình ‑16: Kiến trúc chi tiết phần lõi của Apache

#### Các bước xử lý một gói tin request của Apache

Khi một request được gửi đến Web Server, thì nhiệm vụ xử lý gói Request đó sẽ được nhiều modules khác nhau đảm nhận. Mỗi một phản hồi của Web Server là kết quả của một quá trình xử lý trong Apache. Quá trình xử lý này được phân chia thành nhiêu công đoạn nhỏ hơn, mỗi công đoạn do một module đảm nhận. Mỗi module chỉ liên lạc trực tiếp với core để nhận lệnh và xử lý, sau đó gửi trả kết quả lại cho core, và quá trình đó lại lặp lại đối với các modules khác. Lưu ý là, các module không biết tới các module khác.

Quá trình xử lý một gói tin Request của Apache được thông qua các bước sau:

1. Phân giải: Chuyển đổi từ một URI thành một đường dẫn chứa file tương ứng trên server

2. Kiểm tra truy nhập: dựa vào thông tin hostname và các thông tin khác.

3. Kiểm tra User ID từ gói tin HTTP Request và kiểm tra tính hợp lệ của user.

4. Cho user quyền để truy cập đến các tài nguyên cụ thể.

5. Xác định MIME của đối tượng bị truy vấn. Tức là thông tin về kiểu định dạng của tài nguyên trên server được gọi tên trong gói HTTP Request (danh sách các MIME content type: <http://www.iana.org/assignments/media-types/index.html>)

6. Chỉnh sửa lại một số thông tin, ví dụ thay đổi alias thành một đường dẫn thực (định nghĩa alias sẽ được trình bày trong phần về mod\_alias dưới đây)

7. Phản hồi dữ liệu lại cho client.

8. Ghi lại log.

Tiến trình xử lý này sẽ được điều khiển bới thành phần http\_request.c trong core của Apache.

## JSON và thư viện mã nguồn mở GSON

### Định nghĩa JSON[[24]](#footnote-24)

Json là một ngôn ngữ định dạng dữ liệu hoàn toàn độc lập, dùng để lưu trữ và trao đổi các thông tin dạng văn bản. Json giống với Xml nhưng cấu trúc đơn giản hơn, giúp cho việc phân tích, xử lý dữ liệu một cách nhanh hơn và dễ dàng hơn. Json được dựa trên cơ sở ngôn ngữ lập trình JavaScript, chuẩn ECMA-262 3rd tháng 12, 1999. Một thể hiện của json như sau:

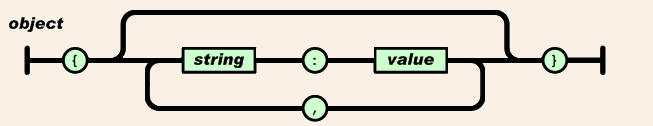
{"employees": [{ "firstName":"John", "lastName":"Doe" },{ "firstName":"Anna", "lastName":"Smith" },{ "firstName":"Peter", "lastName":"Jones" }]}

Json được xây dựng trên hai cấu trúc

* Là một tập hợp các cặp tên – giá trị tương tự như các đối tượng trong các ngôn ngữ lập trình khác.
* Một dãy các giá trị được xắp xếp tương tự như các array, vector, list hay sequence.

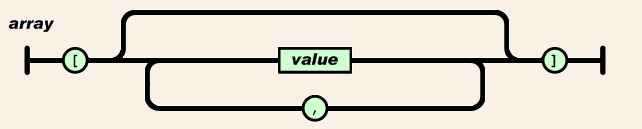
Hai hình thức biểu diễn của Json được sử dụng trong hệ thống này là:

- Object (Đối tượng): Là một dãy các cặp Tên/ Giá trị. Một Object được bắt đầu bằng dấu ngoặc đơn trái ({) và kết thúc bằng dấu ngoặc đơn phải (}). Tên và giá trị ngăn cách bởi dấu hai chấm (:). Mỗi cặp Tên/ Giá trị được ngăn cách bởi dấu phẩy (,).



Hình ‑: Biểu diễn json của một đối tượng

- Array(mảng): Array là một dãy các cặp giá trị, Array được bắt đầu bằng dấu ngoặc vuông trái ([) và kết thúc bằng dấu ngoặc vuông phải (]). Các giá trị được ngăn cách bởi dấu phẩy (,).



Hình ‑: Biểu diễn json của một mảng

Trong hệ thống ORSER-SYSTEM này định dạng dữ liệu sẽ sử dụng cho việc truyền thông giữa server API và android application chính là Json.

### Thư viện mã nguồn mở GSON[[25]](#footnote-25)

Gson là một thư viện Java được sử dụng trong việc chuyển đổi một đối tượng Java bất kỳ sang định dạng Json. Gson cũng hỗ trợ chuyển đổi từ một chuỗi định dạng Json sang đối tượng Java tương ứng. Đây là một thư viện mã nguồn mở được lưu trữ tại <http://code.google.com/p/google-gson>.

Những ưu điểm của Gson:

* Cung cấp các phương thức giúp việc chuyển đổi qua lại giữa đối tượng Java và Json trở nên đơn giản.
* Hỗ trợ chuyển đổi qua lại các đội tượng bất kỳ.
* Cho phép tùy chỉnh các đối tượng.
* Hỗ trợ các đối tượng phức tạp.
* Tạo ra đối tượng Json gọn nhẹ, dễ phân tích.

Lớp chính trong thư viện này là Gson, được khởi tạo đơn giản bằng cách gọi new Gson(). Ngoài ra còn có một lớp GsonBuilder sử dụng để tạo ra một đối tượng Gson với tùy chỉnh thích hợp. Các đối tượng Gson sẽ không duy trì trạng thái khi chuyển đổi qua lại giữa đối tượng Java và Json vì thế ta có thể sử dụng lại đối tượng Gson này trong các lần chuyển đổi tiếp theo. Dưới đây là ví dụ đơn giản cho thấy cách thức sử dụng đối tượng Gson:

* Chuyển đổi từ đối tượng Java sang Json

Gson gson = new Gson();

gson.toJson(1);

gson.toJson("abcd");

gson.toJson(new Long(10));

int[] values = { 1 };

gson.toJson(values);

* Chuyển đổi từ Json sang đối tượng Java

int one = gson.fromJson("1", int.class);

Integer one = gson.fromJson("1", Integer.class);

Long one = gson.fromJson("1", Long.class);

Boolean false = gson.fromJson("false", Boolean.class);

String str = gson.fromJson("\"abc\"", String.class);

String anotherStr = gson.fromJson("[\"abc\"]", String.class);

Trong hệ thống này Gson sẽ được sử dụng để chuyển đổi định dạng dữ liệu truy vấn từ cơ sở dữ liệu sang định dạng là Json trong android application.

Phiên bản Google Gson Version 2.3.1 được phép tải miễn phí tại http://search.maven.org/#artifactdetails|com.google.code.gson|gson|2.3.1|jar

## Kết luận

Qua chương này chúng ta đã nắm vững được những công nghệ để xây dựng hệ thống một cách chi tiết và rõ ràng. Bằng việc nắm vững những kiến thức, công nghệ nền tảng như đã trình bày ở trên, em tiến hành chuyển sang việc phân tích thiết kế hệ thống quản lý taxi, phần này sẽ được trình bày trong chương tiếp theo.

# PHÂN TÍCH VÀ THIẾT KẾ HỆ THỐNG ORDER-SYSTEM

Nội dung chương này sẽ làm rõ những yêu cầu cho hệ thống ORDER-SYSTEM trong thực tế để từ đó có thể tiến hành phân tích những yêu cầu cần phải đáp ứng khi xây dựng hệ thống để từ đó đưa ra phương pháp giải quyết phù hợp trong việc xây dựng hệ thống này.

## Xác định yêu cầu và các đối tượng trong hệ thống

Sau đây chúng ta sẽ tiến hành xác định các đối tượng và yêu cầu cần có với một hệ thống đặt hàng.

### Yêu cầu đối với hệ thống

Đối với một hệ thống đặt hàng luôn có những yêu cầu bắt buộc hệ thống phải đáp ứng được nếu muốn được đưa vào sử dụng trong thực tế.

Hệ thống của chúng em có khả năng đáp ứng được những yêu cầu sau đây:

* Cho phép người dung đầu cuối có thể xem và đặt mua sản phẩm thông qua web hoặc android application.
* Cho phép người quản lí cửa hang có thể phân phối đơn hang cho người giao hàng thông qua bản đồ.
* Cho phép người giao hàng nhìn thấy các đơn hàng cần giao thông qua android application.
* Cho phép người giao có thể tìm đường báo trễ và xác hận hoàn thnhf việc giao hàng bằng android application.

Đối với việc xây dựng client chuyên dụng để phục vụ người dung đầu cuối và người giao hàng trong hệ thống sẽ có đáp ứng những yêu cầu sau:

* Là Smartphone chạy hệ điều hành Android có cài đặt ứng dụng client chuyên dụng của hệ thống với các chức năng như sau:
* Giao tiếp với database quản lý điều hành thông qua Web API server.
* Cập nhật trạng thái vij trí của mình.

### Các đối tượng trong hệ thống

Sau khi nắm bắt được các yêu cầu của hệ thống, chúng ta sẽ phân tích những đối tượng hoạt động chủ yếu trong hệ thống này là khách hàng, người giao hàng và người quản lý hệ thống.

#### Người quản lý hệ thống

Người quản lí hệ thống là người trực tiếp làm việc với hệ thống này thông qua trình duyệt web để thực hiện việc quản lý và điều phối các các nhân viên giao hàng trong hệ thống. Người quản lý có một tài khoản để đăng nhập vào hệ thống thực hiện việc điều phối nhân viên giao hàng thích hợp để giao các đơn hàng. Quá trình diễn ra như sau:

* Người quản lí sẽ kiểm tra các đơn hàng của khách cần giao.
* Thực hiện việc điều phối bằng cách: thông qua bản đồ trạng thái của hệ thống sẽ hiển thị vị trí của các đơn hàng, người quản lý tiến hàng tiến hành tìm các đơn hàng gần nhau và giao cho nhân viên giao hàng thích hợp.
* Ngoài ra, nhân viên quản lý còn có thể thực hiện các tao tác thiết lập cho hệ thống như them đơn sản phầm, tạo ra một sự kiện khuyến mãi và thống kê lại các đơn hàng đã giao.

#### Nhân viên giao hàng

Là người liên hệ trực tiếp phục vụ khách hàng trong hệ thống. Cái nhân viên giao hàng sẽ phải đăng nhập vào hệ thống thông qua ứng dụng client được cài đặt trên Smartphone chạy hệ điều hành Android. Khi nhân viên giao hàng đăng nhập vào hệ thống, client này sẽ có nhiệm vụ tự động cập nhật thông tin về đơn hàng và vị trí các đơn hàng và hiển thị chúng trên bản đồ.

Trong bản đồ này sẽ hiển thị đường đi đến vị trí đơn hàng cần giao mà nhân viên giao hàng đã chọn. Đồng thời cũng gỡi tin nhắn đến khách hàng khi sắp đến vị trí cần giao.

#### Khách hàng

Là đối tượng phục vụ chính. Khách hàng thông qua web site hoặc android application để thực hiện đặt hàng các sản phẩm của hệ thống thống:

Khi đặ hàng khách hàng phải cung cấp các thông tin sau đây:

* Tên khách hàng.
* Địa chỉ email.
* Số điện thoại.
* Vị trí của đơn hàng và địa chỉ cần giao.

Sau khi đơn hàng đã được tạo khách hàng sẽ chờ nhân viên giao hàng liên lạc thông báo việc nhận hàng.

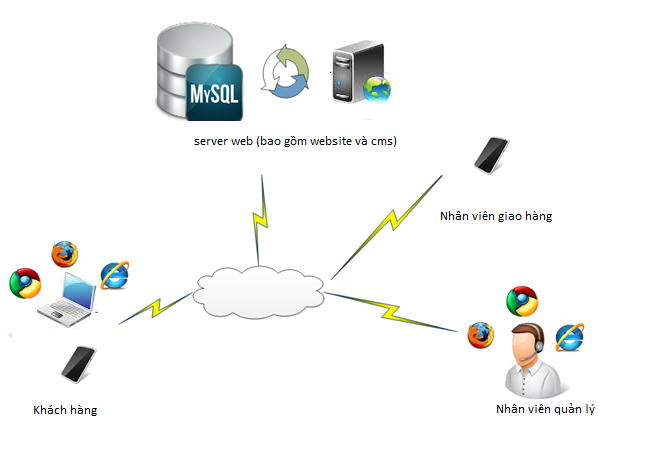
## Thiết kế hệ thống

Hệ thống ORDER-SYSTEM là hệ thống hoạt động theo mô hình Client – Server bao gốm bốn thành phần chính sau đây:

* CMS Web Application là một ứng dụng web cung cấp các chức năng:
  + Web Application cung cấp các chức năng quản lý và điều hành hệ thống sử dụng thông qua trình duyệt web nên rất tiện dụng và nhanh chóng.
  + Cung cấp cho nhân viên quản lý hệ thống có cái nhìn tổng thể về hệ thống với việc hiển thị vị trí, trạng thái của các đơn hàng của khách hàng lên bản đồ số giúp tăng hiệu quả trong việc phân phối đơn hàng.
  + Lưu trữ lịch sử đơn hàng.
  + Thêm mới các sản phẩm để khách đặt hàng.
* Web Application trực tuyến :
  + Cho phép khách hàng đặt hàng các sản phẩm.
  + Xác định vị trí tuong đối của khách hàng thông qua ip internet.
* API Web Application trực tuyến :
  + Cho phép các android application giao tiếp với cơ sở dữ liệu.
  + Là một web application chạy trên nền PHP.

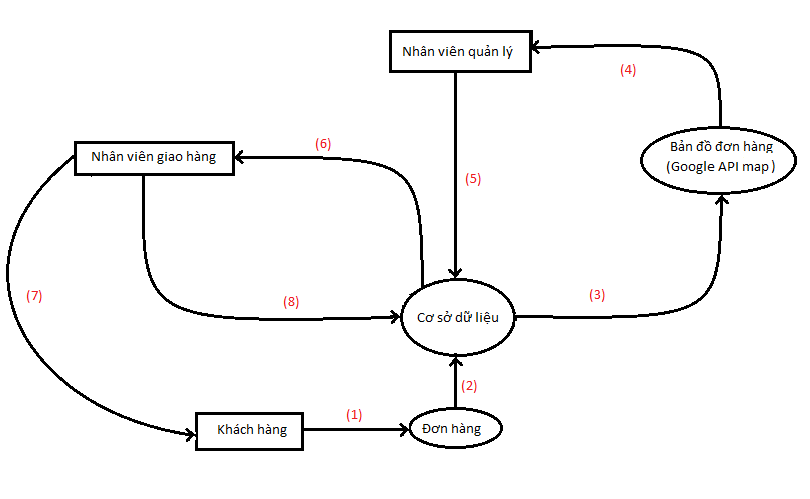
* Android application đặt hàng là ứng dụng được cài đặt trên Smartphone chạy hệ điều hành android với các chức năng:
  + Hiện thị các sản phẩm đang bán của cửa hàng.
  + Cho phép khách hàng đặt hàng thông qua aplication.
  + Cho phép khách hàng xem các đơn hàng đã đặt của mình.
* Android application giao hàng là ứng dụng được cài đặt trên Smartphone chạy hệ điều hành andrhangfdungf cho nhân viên giao hàng với các chức năng:
  + Hiện thị các các đơn hàng cần giao trên google map.
  + Cho phép nhân viên giao hàng xác nhận giao hàng và báo trễ với khách hàng.
  + Cho phép gữi sms tự động khi đến địa điểm giao hàng.

Thiết kế hệ thống được mô tả như hình sau:



Hình ‑: Mô tả thiết kế hệ thống

## Mô hình xử lý của hệ thống

Sau khi đã nắm bắt được những đối tượng và yêu cầu cần có đối với hệ thống này, chúng ta đưa ra mô hình xử lý của hệ thống như được mô tả trong sơ sồ sau:­ **Hình 3‑6: Mô hình xử lý của hệ thống**

Hệ thống hoạt động theo từng bước sau đây:

* (1) Khách hàng sử dụng web site hoặc android application để tạo ra một đơn hàng.
* (2) Đơn hàng được lưu vào database thông qua API web application hoặc website đặt hàng.
* (3) Các đơn hàng chưa được phân phối giao sẽ hiển thị trong bản đồ phân phối.
* (4) Lúc này nhân viên quản lý có nhiệm vụ phân phối các đơn hàng đó cho những nhân viên giao hàng và lưu vào database (5).
* (6) Các nhân viên giao hàng sẽ đăng nhập vào hệ thống bằng android application thông qua API web application để lấy danh sách đơn hàng và thực hiện giao hàng với sự trợ giúp của dịch vụ tìm đường ứng dụng GPS thông qua google map service.
* (7) Nhân viên giao hàng sẽ thực hiện thông báo với khách hàng để giao hàng.
* (8) Sau khi giao hàng thành công nhân viên giao hàng sẽ xác nhận lại trên android application là đã giao thành công, thông tin này sẽ được lưu lại vào databse và kết thúc quá trình giao hàng.

Như vậy chúng ta đã nắm được cơ bản những xử lý mà hệ thống cần phải thực hiện. Tiếp theo chúng ta sẽ tiến hành xây dựng client ở phần tiếp theo.

## Giải pháp hiển thị bản đồ

Giải pháp về bản đồ là một trong những thành phần quan trọng trong hệ thống này. Google Maps API có khả năng hỗ trợ lập trình viên khai thác sức mạnh của Google Maps để sử dụng trong các ứng dụng riêng của mình. Cụ thể đây là công cụ giúp hiển thị dữ liệu cụ thể của từng ứng kết hợp với bản đồ rất hiệu quả. Và trong hệ thống này, chúng em sử dụng Google Maps API để hiển thị dữ liệu về vị trí và trạng thái của các taxi lên trên bản đồ giúp cho người điều hành thuận tiện hơn trong việc quản lý, giám sát và điều hành các xe taxi trong hệ thống.

Google Maps được giới thiệu trên Blog post của google trong tháng 2 năm 2005. Ưu điểm tại thời điểm này của Google Maps là cho phép người dùng kéo chuột để di chuyển bản đồ. Chi phí cho các giải pháp về bản đồ lúc này rất tốn kém và đòi hỏi phải có một máy chủ bản đồ đặc biệt và hầu như chỉ được tạo ra nhằm đáp ứng các yêu cầu riêng của Google do đó nó chứ rất nhiều tính năng không cần thiết làm cho API trên nên cồng kềnh hơn.Từ năm 2005, sự bùng bổ của các thiết bị di động khiến API cũ trở nên chậm chạp và không còn phù hợp cho những thiết bị này nữa. Chính vì vậy các nhà phát triển giải pháp bản đồ đã quyết định xây dưng mới lại API này. Trong quá trình phát triển Google đã cung cấp các Api giúp các lập trình viên dễ dàng tiếp cận và tích hợp bản đồ vào trong ứng dụng của mình. Giúp cho Google Maps trở thành giải pháp bản đồ được sử dụng phổ biến như ngày nay.

Trong hệ thống chúng em sử dụng hai Api cụ thể là Google Maps JavaScript API v3 và Google Maps Android API v2.

Phiên bản 3 của Google Maps JavaScript API là phiên bản được làm mới hoàn toàn so với các phiên bản trước của Google. Api này cho phép nhúng trực tiếp Google Maps vào trang web riêng của chúng ta, nó được thiết kế đặc biệt để hoạt động nhanh và nhiều tính năng hơn cho các thiết bị di động và trình duyệt web. Api này cung cấp các tiện ích giúp cho việc thao tác bản đồ (Giống như trang http://maps.google.com), việc xử lý nội dung trên bản đồ được thực hiện thông qua các Web Service do Google cung cấp giúp chúng ta có thể tạo ra các ứng dụng bản đồ một cách dễ dàng và mạnh mẽ hơn.

Google Maps JavaScript thực chất là sự kết hợp của HTML, CSS và JavaScript. Bản đồ được chia thành từng ô nhỏ là những hình ảnh của bản đồ. Những ảnh này được tải về bằng Ajax sau đó chèn vào một thẻ Div trong trang HTML. Khi người dùng thao tác với bản đồ, API sẽ gửi tọa độ mới và mức độ zoom của bản đồ để gọi Ajax để lấy về những hình ảnh phù hợp với thao tác trên bản đồ.

API này bao gồm các tập tin JavaScript có chứa các lớp với các thuộc tính và phương thức để chúng ta có thể sử dụng khi thao tác với bản đồ. API này được thiết kế chú trọng đến việc tăng hiệu suất cả khi sử dụng trên thiết bị di động và trên các máy tính. Để thực hiện điều này, với phiên bản cũ trình duyệt phải tải và thực thi rất nhiều Script trước khi có thể hiển thị được bản đồ thì phiên bản mới được module hóa sẽ tải về những code cần thiết trước, hiển thị bản đồ sau đó tải tiếp những thứ còn lại sau khi đã hiện thị bản đồ.

Hệ thống tọa độ được sử dụng trong Google Maps là Word Geodetic System 84 (WGS 84), cùng chung hệ thống với Global Postioning System. Vì vậy sẽ rất tiện lợi khi kết hợp bản đồ Google maps với GPS trên Smartphone vì chúng ta không cần tính toán chuyển đổi tọa độ. Để xác định một điểm trong hệ tọa đồ này ta sẽ sử dụng vĩ độ(Latitude) và kinh dộ (Longtitude), hai giá trị này thể hiện bằng số thập phân. Các giá trị vĩ độ luôn đứng trước kinh độ (vĩ độ, kinh độ). Ví dụ như (40.714,-47.005)



Hình ‑: Word Geodetic System 84[[7](#Sve10)]

## Tổng kết

Qua chương này chúng ta đã thấy được chi tiết hơn những yêu cầu và đối tượng trong hệ thống. Bước đầu xác định các thành phần, các xử lý chính của hệ thống. Chúng ta sẽ đi vào phần thiết kế chi tiết của hệ thống ở chương tiếp theo.

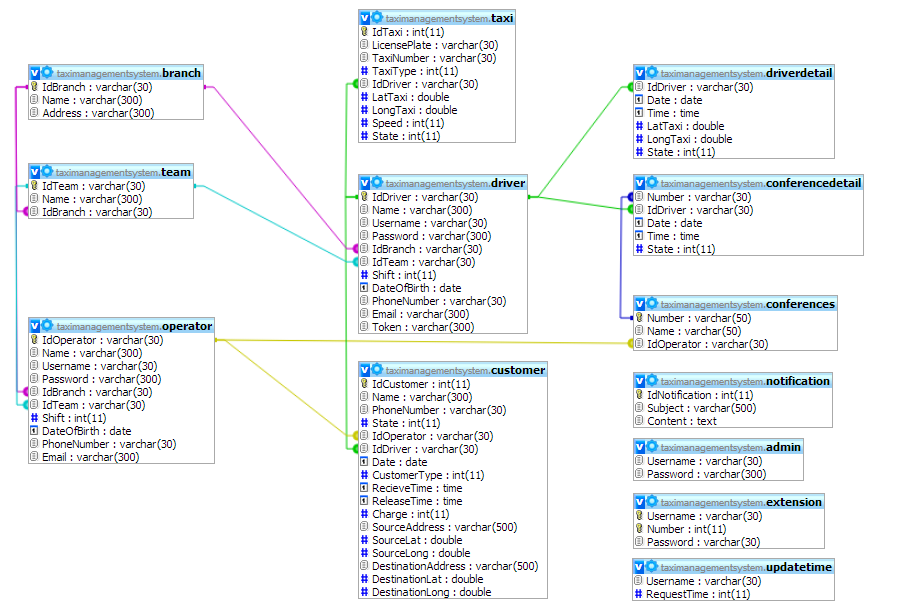
# HIỆN THỰC HỆ THỐNG ORDER - SYSTEM

Sau khi đã nắm vững cơ sở lý thuyết và tiến hành phân tích yêu cầu, đưa ra phương pháp giải quyết bài toán mà đề tài đặt ra, chương này sẽ trình bày chi tiết việc thiết kế, xây dựng và mô hình triển khai hệ thống.

## Xây dựng cơ sở dữ liệu

### Mô hình dữ liệu

Dữ diệu của hệ thống được lưu trữ tại cơ sở dữ liệu taximanagementsystem trên hệ quản trị cơ sở dữ liệu MySQL Server và được thiết kế như sau:



Hình ‑: Cơ sở dữ liệu

### Chi tiết cơ sở dữ liệu

#### Quan hệ ADMIN

Ý nghĩa: lưu giữ thông tin chứng thực đăng nhập của quản trị hệ thống.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Tên Thuộc tính** | **Diễn giải** | **Kiểu dữ liệu** | **Giá trị** | **Ghi chú** |
| **Username** | Tài khoản đăng nhập hệ thống | varchar(30) | notNull |  |
| Password | Mật khẩu đăng nhập | varchar(300) | notNull |  |

Bảng ‑: Quan hệ ADMIN

#### Quan hệ BRANCH

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Tên Thuộc tính** | **Diễn giải** | **Kiểu dữ liệu** | **Giá trị** | **Ghi chú** |
| **IdBranch** | Mã chi nhánh | varchar(30) | notNull |  |
| Name | Tên chi nhánh | varchar(300) | notNull |  |
| Address | Địa chỉ chi nhánh | varchar(300) | notNull |  |

Ý nghĩa: lưu trữ thông tin các chi nhánh trong hệ thống.

Bảng ‑: Quan hệ BRANCH

#### Quan hệ TEAM

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Tên Thuộc tính** | **Diễn giải** | **Kiểu dữ liệu** | **Giá trị** | **Ghi chú** |
| **IdTeam** | Mã đội | varchar(30) | notNull |  |
| Name | Tên đội | varchar(300) | notNull |  |
| IdBranch | Mã chi nhánh của đội | varchar(30) | notNull |  |

Ý nghĩa: lưu trữ thông tin về các đội xe trong hệ thống.

Bảng ‑: Quan hệ TEAM

#### Quan hệ DRIVER

Ý nghĩa: lưu trữ danh sách các tài xế trong hệ thống bao gồm các thông tin cá nhân, thông tin tài khoản đăng nhập hệ thống của tài xế.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Tên Thuộc tính** | **Diễn giải** | **Kiểu dữ liệu** | **Giá trị** | **Ghi chú** |
| **IdDriver** | Mã tài xế | varchar(30) | notNull |  |
| Name | Tên tài xế | varchar(300) | notNull |  |
| Username | Tài khoản đăng nhập hệ thống | varchar(30) | notNull |  |
| Password | Mật khẩu đăng nhập hệ thống | varchar(300) | notNull |  |
| IdBranch | Mã chi nhánh của tài xế | varchar(30) | notNull |  |
| IdTean | Mã đội của tài xế | varchar(30) | notNull |  |
| Shift | Ca làm việc | int | notNull |  |
| DateOfBirth | Ngày sinh | date | notNull |  |
| PhonneNumber | Số điện thoại | varchar(30) | notNull |  |
| Email | Email tài xế | varchar(300) | notNull |  |
| Token | Token kiểm tra trạng thái đăng nhập của tài xế | varchar(300) | notNull |  |

Bảng ‑: Quan hệ DRIVER

#### Quan hệ DRIVERDETAIL

Bảng ‑: Quan hệ DRIVERDETAIL

Ý nghĩa: Lưu trữ thông tin về lịch sử hoạt động của tài xế như ngày, giờ, vị trí, trạng thái của xe taxi do tài xế điều khiển sẽ được tự động cập nhật khi tài xế đăng nhập hệ thống.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Tên Thuộc tính** | **Diễn giải** | **Kiểu dữ liệu** | **Giá trị** | **Ghi chú** |
| **IdDriver** | Mã tài xế | varchar(30) | notNull |  |
| Date | Ngày hoạt động của tài xế | date | notNull |  |
| Time | Thời gian hoạt động của tài xế | time | notNull |  |
| LatTaxi | Tọa độ Latitude của tài xế | double | notNull |  |
| LongTaxi | Tội độ Longitude của tài xế | double | notNull |  |
| State | Trạng thái của tài xế | int | notNull |  |

#### Quan hệ OPERATOR

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Tên Thuộc tính** | **Diễn giải** | **Kiểu dữ liệu** | **Giá trị** | **Ghi chú** |
| **IdOperator** | Mã nhân viên điều phối | varchar(30) | notNull |  |
| Name | Tên nhân viên điều phối | varchar(300) | notNull |  |
| Username | Tài khoản đăng nhập hệ thống | varchar(30) | notNull |  |
| Password | Mật khẩu đăng nhập hệ thống | varchar(300) | notNull |  |
| IdBranch | Mã chi nhánh của nhân viên | varchar(30) | notNull |  |
| IdTean | Mã đội của nhân viên | varchar(30) | notNull |  |
| Shift | Ca làm việc | int | notNull |  |
| DateOfBirth | Ngày sinh | date | notNull |  |
| PhonneNumber | Số điện thoại | varchar(30) | notNull |  |
| Email | Email tài xế | varchar(300) | notNull |  |

Ý nghĩa: Lưu trữ thông tin, tài khoản đăng nhập của các nhân viên điều phối trong hệ thống.

Bảng ‑: Quan hệ OPERATOR

#### Quan hệ TAXI

Bảng ‑: Quan hệ TAXI

Ý nghĩa: lưu trữ thông tin của các xe taxi hoạt động trong hệ thống. Vị trí của các xe sẽ được cập nhật khi tài xế điều khiển xe đó đăng nhập vào hệ thống.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Tên Thuộc tính** | **Diễn giải** | **Kiểu dữ liệu** | **Giá trị** | **Ghi chú** |
| **IdTaxi** | Mã xe taxi trong hệ thống | int | notNull |  |
| LicensePlate | Biển số xe taxi | varchar(30) | notNull |  |
| TaxiNumber | Số hiệu taxi | varchar(30) | notNull |  |
| TaxiType | Loại xe taxi | int | notNull |  |
| IdDriver | Mã tài xế của taxi | varchar(30) | notNull |  |
| LatTaxi | Tọa độ Latitude của taxi | double | notNull |  |
| LongTaxi | Tọa độ Longtitue của taxi | double | notNull |  |
| Speed | Tốc độ của xe | int | notNull |  |
| State | Trạng thái của xe | int | notNull |  |

#### Quan hệ CUSTOMER

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Tên Thuộc tính** | **Diễn giải** | **Kiểu dữ liệu** | **Giá trị** | **Ghi chú** |
| **IdCustomer** | Mã số khách hàng | int | notNull |  |
| Name | Tên khách hàng | varchar(300) | notNull |  |
| PhoneNumber | Số điện thoại khách hàng | varchar(30) | notNull |  |
| State | Trạng thái của khách hàng | int | notNull |  |
| IdOperator | Mã nhân viên điều phối | varchar(30) | notNull |  |
| IdDriver | Mã tài xế phục vụ | varchar(30) | notNull |  |
| Date | Ngày phục vụ khách | date | notNull |  |
| CustomerType | Phân loại khách hàng | int | notNull |  |
| RecieveTime | Thời gian nhận khách | time | notNull |  |
| ReleaseTime | Thời gian trả khách | time | notNull |  |
| Charge | Chí phí | int | notNull |  |
| SourceAddress | Địa chỉ đón khách | varchar(500) | notNull |  |
| SourceLat | Tọa độ Latitude nơi đón khách | double | notNull |  |
| SourceLong | Tọa độ Longitude nơi đón khách | double | notNull |  |
| DestinationAddress | Địa chỉ nơi trả khách | varchar(500) | notNull |  |
| DestinationLat | Tọa độ Latitude nơi trả khách | double | notNull |  |
| DestinationLong | Tọa độ Longitude nơi trả khách | double | notNull |  |

Ý nghĩa: lưu trữ thông tin trạng thái, vị trí, thông tin trong quá trình phục vụ khách hàng.

Bảng ‑: Quan hệ CUSTOMER

#### Quan hệ NOTIFICATION

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Tên Thuộc tính** | **Diễn giải** | **Kiểu dữ liệu** | **Giá trị** | **Ghi chú** |
| **IdNotification** | Mã số thông báo | int | notNull |  |
| Subject | Chủ đề thông báo | varchar(500) | notNull |  |
| Content | Nội dung thông báo | text | notNull |  |

Ý nghĩa: lưu trữ nội dung các thông báo trong hệ thống

Bảng ‑: Quan hệ NOTIFICATION

#### Quan hệ EXTENSION

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Tên Thuộc tính** | **Diễn giải** | **Kiểu dữ liệu** | **Giá trị** | **Ghi chú** |
| **Username** | Tài khoản của nhân viên | varchar(30) | notNull |  |
| Number | Số nội bộ của nhân viên | int | notNull |  |
| Password | Mật khẩu xác thực số nội bộ với trixbox | varchar(30) | notNull |  |

Ý nghĩa: lưu trữ thông tin về tài khoản SIP trong hệ thống VOIP nội bộ.

Bảng ‑: Quan hệ EXTENSION

#### Quan hệ CONFERENCES

Ý nghĩa: Lưu trữ thông tin các phòng hội thoại trong hệ thống.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Tên Thuộc tính** | **Diễn giải** | **Kiểu dữ liệu** | **Giá trị** | **Ghi chú** |
| **Number** | Số phòng hội thoại nội bộ | varchar(50) | notNull |  |
| Name | Tên phòng | varchar(50) | notNull |  |
| IdOperator | Id nhân viên tạo phòng | varchar(30) | notNull |  |

Bảng ‑: Quan hệ CONFERENCES

#### Quan hệ CONFERENCEDETAIL

Ý nghĩa: Lưu trữ thông tin các tài xế trong một phòng hội thoại

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Tên Thuộc tính** | **Diễn giải** | **Kiểu dữ liệu** | **Giá trị** | **Ghi chú** |
| Number | Số phòng hội thoại | varchar(30) | notNull |  |
| IdDriver | Mã tài xế thuộc phòng | varchar(30) | notNull |  |
| Date | Ngày tạo phòng | date | notNull |  |
| Time | Thời gian được mời vào phòng | time | notNull |  |
| State | Trạng thái của tài xế đối với phòng | int | notNull |  |

Bảng ‑: Quan hệ CONFERENCEDETAIL

#### Quan hệ UPDATETIME

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Tên Thuộc tính** | **Diễn giải** | **Kiểu dữ liệu** | **Giá trị** | **Ghi chú** |
| **Username** | Tài khoản đăng nhập | varchar(30) | notNull |  |
| RequestTime | Thời gian yêu cầu | int | notNull |  |

Ý nghĩa: Lưu trữ giá trị thời gian yêu cầu tài xế phải cập nhật về trung tâm điều khiển.

Bảng ‑: Quan hệ UPDATETIME

## Xây dựng Web Service

### Web Service hỗ trợ truy vấn dữ liệu

Việc hiển thị dữ liệu của hệ thống lên Google Map đòi hỏi việc phải truy vấn dữ liệu được lưu trữ trong cơ sở dữ liệu MySQL. Dữ liệu cần phải truy vấn ở đây chủ yếu là thông tin về xe taxi trong hệ thống theo từng yêu cầu cụ thể của nhân viên điều phối, thông tin khách hàng phục vụ cho việc thống kê danh sách khách hàng của hệ thống, thông tin về lịch sử hoạt động của taxi trong hệ thống.

Để thực hiện việc truy vấn dữ liệu hệ thống xây dựng RESTful Web Service để thực hiện việc truy vấn dữ liệu từ MySQL theo yêu cầu của từng chức năng trong hệ thống. Dữ liệu truy vấn sẽ được chuyển thành định dạng Json với sự hỗ trợ của thư viện Gson và được phản hồi lại theo từng yêu cầu.

|  |  |  |
| --- | --- | --- |
| Phương thức | URI | Chức năng |
| GET | /resources/query/getAllCustomerInDate | Lấy thông tin tất cả khách hàng trong ngày |
| GET | /resources/query/getAllCustomerInMonth | Lấy thông tin tất cả khách hàng trong tháng |
| GET | /resources/query/ | Lấy thông tin toàn bộ taxi trong hệ thống |
| GET | /resources/query/ getCustomerOfOperatorByDate | Khách hàng của một nhân viên trong ngày |
| GET | /resources/query/ getCustomerOfDriverByDate | Khách hàng của một tài xế trong ngày |
| GET | /resources/query/ getCustomerOfDriverByMonth | Khách hàng của một tài xế trong tháng |
| GET | /resources/query/ getCustomerOfOperatorByMonth | Khách hàng của một nhân viên trong tháng |
| GET | /resources/query/ getTaxi | Thông tin một xe cụ thể |
| GET | /resources/query/ getTimeLineTaxi | Thông tin lịch sử hoạt động của xe |
| GET | /resources/query/ joinConference | Giám sát thông tin tham gia phòng hội thoại |

Bảng ‑: Web Service hỗ trợ truy vấn dữ liệu

### Web Service lựa chọn xe phù hợp

Trong quá trình điều phối của nhân viên tại trung tâm điều khiển thì việc lựa chọn các xe taxi phù hợp và tối ưu nhất để phục vụ khách hàng chiếm một vị trí rất qua trọng. Yêu cầu đối với việc thực hiện lựa chọn taxi là cho phép nhân viên diều phối lựa chọn theo loại taxi, chọn số lượng taxi phụ hợp mà hệ thống có thể gợi ý lựa chọn, cho phép điều một taxi hoặc nhiều taxi cùng một lúc theo thứ tự ưu tiên về khoảng cách từ taxi đến vị trí khách hàng.

Nhân viên điều phối sẽ lựa chọn xe bằng cách kết hợp những tùy chọn sau: chọn loại xe, chọn số lượng xe gợi ý, chọn điều một xe hoặc nhiều xe cùng lúc. Tiếp theo nhân viên điều phối sẽ phải cung cấp thông tin về tọa độ khách hàng. Những thông tin này sẽ được Web Service tiếp nhận và xử lý. Quá trình xử lý được thực hiện theo các bước sau đây:

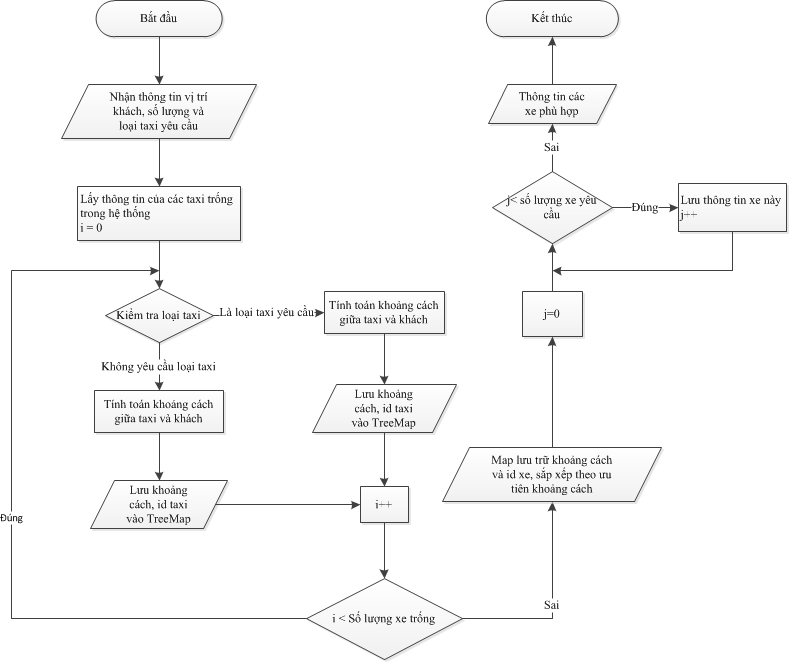
* Tiếp nhận thông tin từ nhân viên điều phối.
* Lấy thông tin về các xe trong trong hệ thống.
* Phân loại theo loại taxi yêu cầu, tính toán khoảng các từ taxi đến vị trí của khách hàng. Xắp xếp ưu tiên theo khoảng cách gần nhất.

Khoảng cách giữa hai tọa độ A(latA,lngA) và B(latB,lngB) với bán kính trái đất R=6371Km được tính theo công thức sau:

d=arccos(sin(latA)\*sin(latB)+cos(latA)\*cos(latB)\*cos(lngA-lngB))\*R

* Dựa theo số lượng taxi trong yêu cầu để lựa chọn số lượng cho phù hợp sau đó gửi thông tin về những taxi phù hợp cho nhân viên điều phối.

Quá trình xử lý được mô tả trong sơ đồ sau:

Chi tiết Web Service:

Hình ‑: Web Service lựa chọn xe phù hợp

* Phương thức: GET
* URI: /resources/query/getNearTaxi
* Danh sách tham số

|  |  |
| --- | --- |
| Tham số | Mô tả |
| lat | Latitute của khách hàng |
| tng | Longitude của khách hàng |
| no | Số lượng xe yêu cầu gợi ý |
| isMultiTaxi | Chọn nhiều xe hay không |
| taxiType | Loại xe yêu cầu gợi ý |

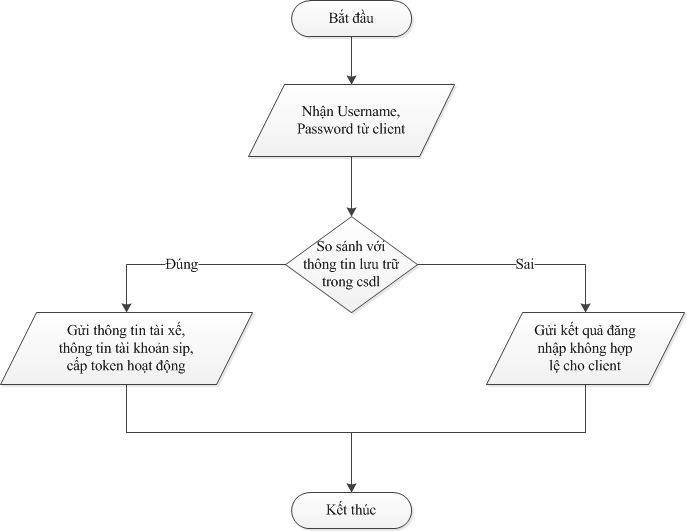
### Web Service tương tác với Client

#### Chức năng đăng nhập, đăng xuất

*Đăng nhập:*

Để nhân viên điều phối có thể quản lý và giám sát các taxi trong hệ thống thì bắt buộc các tài xế taxi phải truy cập vào hệ thống thống bằng cách cung cấp username và password cho ứng dụng client. Client sẽ gửi thông tin này lên server điều khiển để xác thực sự tồn tại của username này. Sau khi đã kiểm tra thông tin là hợp lệ hệ thống sẽ phát sinh ngẫu nhiên và duy nhất một token cho username vừa thực hiện đăng nhập sau đó gửi trả về cho client phía tài xế. Token này được tạo bằng các sử dụng lớp java.util.UUID để tạo ra một token với giá trị có độ dài 128 bit.

Khi nhận được thông tin token này, client sẽ phải sử dụng token này cho việc truy cập các tài nguyên và cập nhật thông tin về server điều khiển vì server sẽ chỉ đáp ứng những yêu cầu có kèm đúng token được cấp phát khi tài xế đăng nhập hệ thống. Ngoài ra, khi đăng nhập thành công hệ thống còn cung cấp thêm thông tin về tài xế, thông tin về tài khoản sip để tài xế sử dụng trong việc liên lạc thông qua tổng đài Trixbox tại trung tâm điều khiển.

Mô hình hoạt động của của Web Service xử lý đăng nhập được thể hiện trong hình sau:

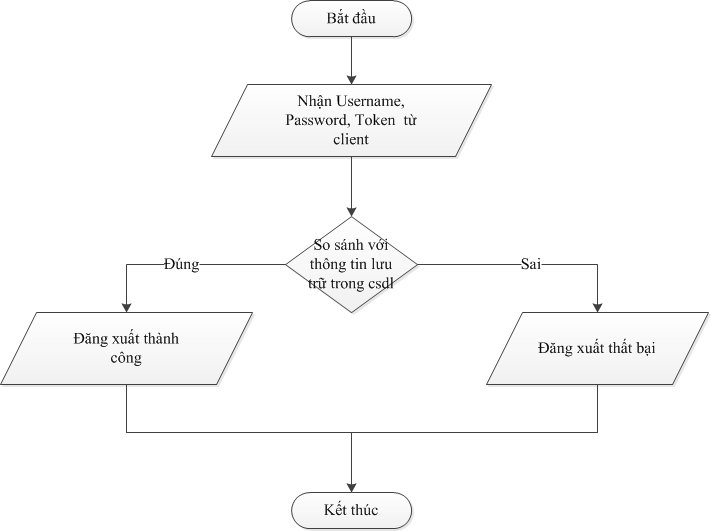
Hình ‑: Chức năng đăng nhập

Chi tiết Web Service:

* Phương thức: GET
* URI: /resources/login
* Danh sách tham số

|  |  |
| --- | --- |
| Tham số | Mô tả |
| username | Tài khoản đăng nhập hệ thống |
| password | Mật khẩu |

*Đăng xuất:*

Việc kết thúc hoạt động, tài xế sẽ phải thực hiện việc đăng xuất khỏi hệ thống để nhân viên điều phối có thể kiểm soát được những taxi nào đang hoạt động và đã ngưng hoạt động. Thông tin chứng thực được gửi lên server điều khiển bao gồm: username, password, và token đã được cấp phát lúc đăng nhập. Quá trình xử lý đăng xuất của tài xế được thực hiện thông qua Web Service được mô tả như hình sau đây.

Hình ‑: Chức năng đăng xuất

Chi tiết Web Service:

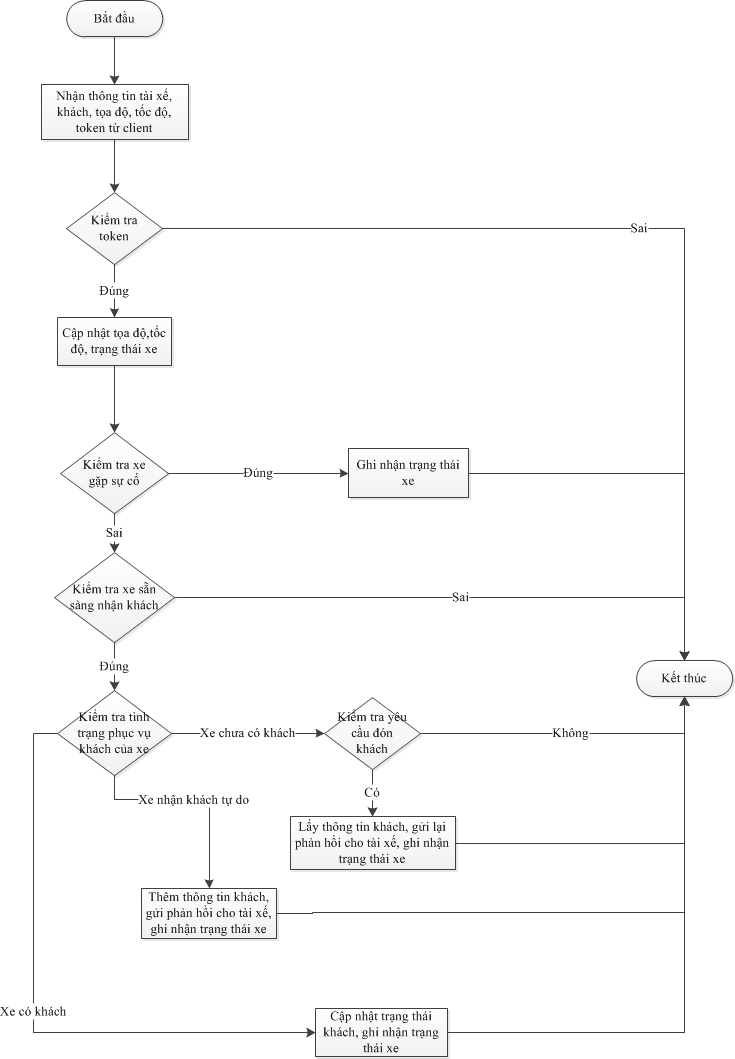
* Phương thức: GET
* URI: /resources/logout
* Danh sách tham số

|  |  |
| --- | --- |
| Tham số | Mô tả |
| username | Tài khoản đăng nhập hệ thống |
| password | Mật khẩu |
| token | Token được cấp phát khi đăng nhập |

#### Giao tiếp, cập nhật trạng thái và trao đổi dữ liệu với trung tâm điều khiển

Trong việc quản lý và giám sát taxi, việc liên lạc trao đổi thông tin giữa trung tâm điều hành và tài xế chiếm vai trọ rất quan trọng. Để đáp ứng yêu cầu này của hệ thống, chúng ta sẽ phải thiết kế một Web Service nhằm thực hiện việc giao tiếp, cập nhật trạng thái và trao đổi dữ liệu với trung tâm điều khiển. Client hoạt động trên mỗi taxi có nhiệm vụ cập nhật các thông tin về vị trí, tốc độ, trạng thái xe cũng như tình trạng của khách nếu xe đang phục vụ khách. Khi nhận được những thông tin cập nhật từ client, web service này sẽ tiến hành xử lý những thông tin này như sau:

* Ghi nhận trạng thái xe, cập nhật tọa độ vị trí, tốc độ của xe vào cơ sở dữ liệu.
* Tùy vào trạng thái của xe để có thể cung cấp thông tin phản hồi đến client cho phù hợp.
* Cung cấp thông tin khách hàng cho tài xế khi xe có yêu cầu đón khách từ nhân viên điều phối.
* Ghi nhận trạng thái và thông tin khách hàng nếu xe đang phục vụ khách.
* Cung cấp thông tin về phòng hội thoại khi tài xế được yêu cầu tham gia phòng từ nhân viên điều phối.
* Cung cấp thông tin về thời gian phải gửi cập nhật về trung tâm điều khiển.



Hình ‑: Giao tiếp, cập nhật trạng thái và trao đổi dữ liệu với trung tâm điều khiển

Chi tiết Web Service:

* Phương thức: GET
* URI: /resources/query/update
* Danh sách tham số

|  |  |
| --- | --- |
| Tham số | Mô tả |
| username | Tài khoản của tài xế cập nhật |
| lat | Latitute của xe |
| tng | Longitude của xe |
| speed | Tốc độ của xe |
| state | Trạng thái của xe |
| idCustomer | Mã khách hàng đang phục vụ |
| token | Token được cấp phát khi đăng nhập |

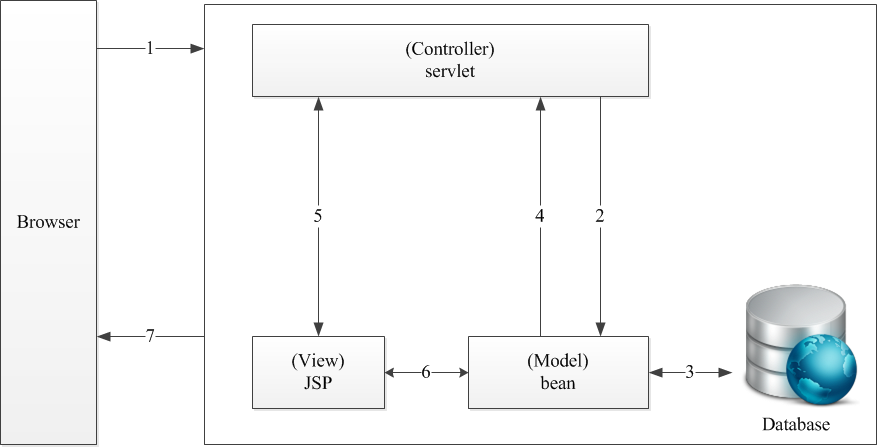
## Xây dựng ứng dụng web quản lý và điều hành taxi

Để giúp nhân viên sử dụng các tài nguyên trong hệ thống vào việc điều phối xe, chúng em xây dựng một ứng dụng web phục vụ việc quản lý, giám sát và điều phối xe taxi trong hệ thống.

### Tài nguyên hỗ trợ

* **IDE:** NetBeans IDE 7.0.1
* **SERVER:** Apache Tomcat 7.0.14.
* **JAVA EE VERSION**: Java EE 6 Web.
* **JAVA PLATFORM:** JDK 1.6 (Java Development Kit) đây là bộ thư viện, công cụ, tài liệu để phát triển ứng dụng viết bằng ngôn ngữ java được cung cấp bởi Oracle. Bao gồm các thành phần chính sau:
  + Development tool: bộ công cụ giúp phát triển, thực thi, debug và tạo java doc cho ứng dụng java
  + Runtime Enviroment: môi trường thực thi ứng dụng java bao gồm máy ảo java, các thư viện hỗ trợ.
  + Source code: bao gồm toàn bộ source code của JDK để ngừời sử dụng có thể nắm bắt đựợc các xử lý của bộ thư viện.
* **JDBC Driver for MySQL (Connector/J)**: MySQL cung cấp một thư viện chuẩn được gọi là driver để các framework như JDBC, ODBC, .NET tạo một ứng dụng có thể thao tác, xử lý trên hệ quản trị cơ sở dữ liệu này. Phiên bản được sử dụng là 5.1.20 download miễn phí tại: <http://www.mysql.com/downloads/connector/j/>.

### Cấu trúc ứng dụng

Ứng dụng hỗ trợ là một Java Web Application với cơ sở dữ liệu là MySQL đươc xây dựng theo mô hình MVC(Model – View – Controller). Hình sau mô tả cách thức hoạt động của các thành phần trong ứng dụng:

Hình ‑: Cách thức hoạt động của ứng dụng

Controller: package servlet chứa các Servlet xử lý và phản hồi lại các yêu cầu được thực hiện bởi người dùng.

Model: package bean bao gồm các lớp java thực hiện việc ánh xạ dữ liệu từ cơ sở dữ liệu MySQL sang các lớp đối tượng java

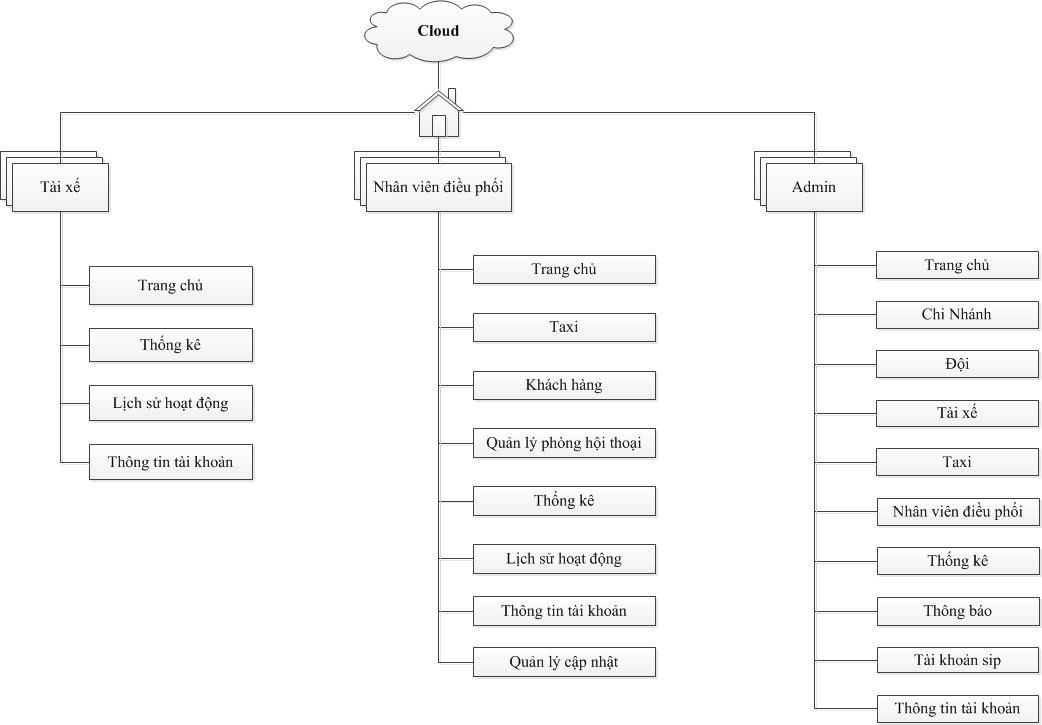
View: pakage chứa các trang JSP có nhiệm vụ trình bày dữ liệu ra màn hình.

Các bước hoạt động của ứng dụng:

* Nhân viên điều phối thực hiện gửi yêu cầu đến server thông qua trình duyệt web.
* Các Servlet (Controller): có nhiệm vụ nhận các yêu cầu, phân tích các giá trị tham số trong yêu cầu để tiến hành xử lý, lựa chọn các Model để thực hiện.
* Model thực hiện các yêu cầu kết nối, truy vấn dữ liệu từ cơ sở dữ liệu MySQL và trả về kết quả cho Controller
* Controller nhận kết quả trả về từ Model thực hiện tạo hoặc lựa chọn View để hiện thị kết quả nhận được.
* Controller gửi phản hồi lại trình duyệt web của nhân viên điều phối để hiện thị kết quả của yêu cầu mà nhân viên điều phối đã thực hiện.

### Xây dựng các chức năng cho các đối tượng sử dụng hệ thống

Ứng dụng web quản lý và điều hành taxi chủ yếu phục vụ ba đối tượng của hệ thống được mô tả như hình sau:



Hình ‑: Ứng dụng web quản lý và điều hành taxi

Mỗi đối tượng sẽ có những chức năng và quyền hạn được quy định cụ thể như sau:

* Tài xế: Có các thành phần cho phép: kiểm tra vị trí, trạng thái xe hiện tại, thống kê khách hàng đã phục vụ của tài xế, xem lại lịch sử hoạt động của taxi theo ngày, chỉnh sửa thông tin tài khoản của tài xế.
* Nhân viên điều phối: có các thành phần cho phép kiểm tra, giám sát vị trí, trạng thái của từng xe hoặc tất cả các xe trong hệ thống, thực hiện việc tiếp nhận, lựa chọn và điều phối xe thích hợp phục vụ khách hàng, thông kê khách hàng phục vụ, quản lý thời gian quy định mỗi tài xế phải thực hiện việc cập nhật thông tin về trung tâm điều khiển, quản lý các phòng hội thoại, việc yêu cầu tham gia phòng hội thoại đến các tài xế
* Quản trị viên: quản lý các đối tượng trong cơ sở dữ liệu, quản lý thông tin tài khoản, tài khoản sip.

### Xây dựng chức năng điều phối

Đây là thành phần chiếm vai trò rất quan trọng trong ứng dụng. Nó giúp cho hoạt động điều phối của nhân viên trở nên tiện lợi, nhanh chóng và chính xác. Nhân viên điều phối sẽ sử dụng các control trên form để cung cấp các thông tin cần thiết đến Web Service lựa chọn xe phù hợp để Web Service này thực hiện việc lựa chọn, sau đó nhận kết quả từ Web Service này để hiện thị cho nhân viên điều phối lựa chọn những xe phù hợp nhất.

Hình dưới đây là giao diện sử dụng của thành phần điều phối xe:



Hình ‑: Thành phần điều phối xe

Từ đó ta có các Control để thực hiện việc điều phối như sau:

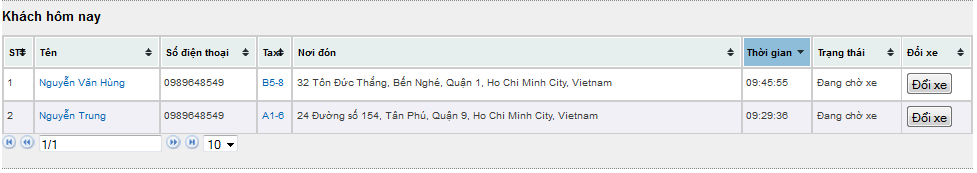
* Loại xe: lựa chọn loại xe điều phối, ở đây ta có thể chọn loại 4 chỗ, 7 chỗ hoặc chọn tất cả các xe.
* Số lượng xe để lựa chọn: đưa ra yêu cầu về số lượng xe mà hệ thống sẽ đưa ra gợi ý để lựa chọn, ví dụ giá trị đây là 5 có nghĩa là hệ thống sẽ chọn ra 5 xe có vị trí gần nhất để cho nhân viên điều phối lựa chọn.
* Điều nhiều xe: tùy chọn cho phép điều một xe hoặc nhiều xe đến một vị trí khách hàng.
* Chọn địa chỉ, Quận: là control giúp cho nhân viên điều phối tìm kiếm địa chỉ, tọa độ của khách hàng trên bản đồ Google Map bằng các sử dụng đối tượng Geocoder của Goolge Map Api.
* Thông tin khách hàng: nhân viên điều phối sẽ nhập tên và số điện thoại của khách hàng. Những thông tin về nơi đón và vị trí khách hàng sẽ được hệ thống tự động điền vào. Những thông tin này sau đó sẽ được gửi đến tài xế được lựa chọn.
* Tài xế: đưa ra một danh sách các tài xế có vị trí gần nhất đối với khách hàng, nhân viên điều phối có thể lựa chọn một hoặc nhiều tài xế để phục vụ khách.

Sau khi đã lựa chọn được xe phù hợp, nhân viên sẽ nhấn chọn nút “Thêm” để thực hiện việc điều phối xe cho khách. Trạng thái khách sẽ được hiện thị ngay trên bản đồ điều phối để giúp nhân viên có thể theo dõi việc phục vụ khách của các xe taxi

Hình ‑: Hiển thị khách trên bản đồ

Các biểu tượng tương ứng với trạng thái của khách hàng được mô tả như hình dưới đây:

Hình ‑: Trạng thái khách hàng

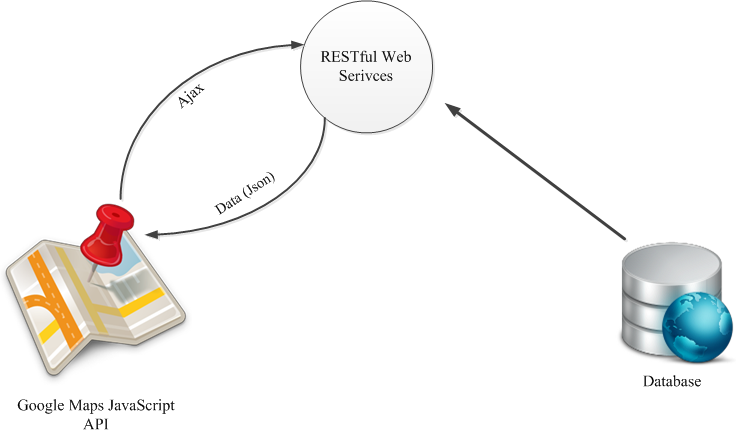
Sẽ có một danh sách khách hàng trong ngày được hiển thị như sau:

Hình ‑: Danh sách khách hàng

Trong trường hợp tài xế sau khi nhận khách không thể thực hiện được việc đón khách sẽ có một bảng thông báo hiện thị trên bản đồ để cho nhân viên điều phối thực hiện việc chuyển đổi tài xế phục vụ cho khách đó.

## Xây dựng hệ thống bản đồ

### Phương pháp thực hiện

Trong hệ thống này,để giúp người điều phối có được một cái nhìn tổng quan và dễ dàng trong việc điều phối các xe thì Server phải có nhiệm vụ hiển thị vị trí và trạng thái các xe đang hoạt động trên hệ thống lên bản đồ số. Vị trí và trạng thái của xe được cập nhật tương ứng với hoạt động của xe theo thời gian thực. Cơ chế hoạt động của thành phần hiển thị bản đồ được mô tả như hình sau:

Hình ‑: Mô hình xử lý hiển thị bản đồ

Chức năng các thành phần:

* Google Maps JavaScript API: hiển thị dữ liệu là vị trí và trạng thái của các xe trong hệ thống lên bản đồ. Dữ liệu lấy về theo phương thức Ajax (Asynchronous JavaScript and XML) với định dạng dữ liệu là Json.
* RESTful Web Service: có nhiệm vụ truy vấn dữ liệu từ cơ sở dữ liệu, chuyển dữ liệu về định dạng Json và gửi dữ liệu về theo yêu cầu.
* Cơ sở dữ liệu: lưu trữ vị trí, trạng thái và các thông tin của các xe trong hệ thống.

Theo như hình mô tả ta thấy dữ liệu được hiển thị lên bản đồ sẽ được lấy về theo phương thức Ajax được thực thi với sự hỗ trợ của thư viện Jquery thông qua RESTful Web Service với định dạng Json. Trong một khoảng thời gian nhất định, yêu cầu lấy dữ liệu sẽ được gửi đến RESTful Web Service. Dữ liệu trả về sẽ được xử lý và hiển thị lên bản đồ.

### Triển khai các bản đồ trong hệ thống

#### Hiển thị bản đồ

Google Maps JavaScript API là bản đồ số được nhúng trực tiếp vào trang web cụ thể, để được quyền sử dụng bản đồ này chúng ta phải đăng ký, đồng ý với các quy định của Google để được cấp một key. Chúng ta có sử dụng đã đăng ký này cho tất cả các Google Maps API. Để tạo key chúng ta sẽ thực hiện theo những bước sau:

* Truy cập và đăng nhập bằng tài khoản của google vào APIs Console tại https://code.google.com/apis/console.
* Trong phần Services sẽ liệt kê tất cả các API được cung cấp bởi Google, chúng ta sẽ kích hoạt Google Maps API v3 tại đây.
* Sau khi kích hoạt, trong phần API Access ta sẽ thấy API key đã được tạo ra
* Sau khi có được key. Việc hiển thị bản đồ trên web rất đơn giản như trong ví dụ sau:
* Chèn MAPs API JavaScript bằng các sử dụng thẻ script như ví dụ sau

<script type="text/javascrip> src="https://maps.googleapis.com/maps/api/js?key=YOUR\_API\_KEY&sensor=SET\_TO\_TRUE\_OR\_FALSE">

</script>

Với YOUR\_API\_KEY chính là key chúng ta đã đăng ký ở phần trên.

* Tạo một thẻ div với tên là “map\_canvas”, bản đồ sẽ được hiển thị trên web bên trong thẻ div này.

<div id="map\_canvas" style="width:100%; height:100%"></div>

Chiều dài, rộng của bản đồ sẽ phụ thuộc vào các thuộc tính của div này

* Chúng ta sẽ tạo một JavaScript Object chứa các thuộc tính của bản đồ

var mapOptions = {

center: new google.maps.LatLng(-34.397, 150.644),

zoom: 8,

mapTypeId: google.maps.MapTypeId.ROADMAP

};

* Khởi tạo đối tượng map sẽ được hiển thị trong div “map\_canvas” với các Option được khỏi tạo trong đối tượng mapOptions

var map = new google.maps.Map(document.getElementById("map\_canvas"),

mapOptions);

Như vậy, về cơ bản chúng ta đã hiển thị được bản đồ số của Google lên trên web. Tiếp theo chúng ta sẽ tìm hiểu về phương thức hiển thị xe, vị trí và trạng thái các xe lên trên bản đồ.

#### Hiển thị xe

Việc tương tác với bản đồ, yêu cầu, xử lý, hiển thị dữ liệu sẽ được thực hiện hoàn toàn bằng JavaScript. Để hiển thị vị trí và trạng thái các xe chúng ta sẽ sử dụng Maker(google.maps.Marker) của API. Đối tượng Maker này có chức năng xác định một vị trí trên bản đồ. Vị trí của xe cũng chính là vị trí của Maker trên bản đồ. Trạng thái của xe hiển thị bằng cách thay đổi icon của Maker, được quy định theo bảng sau đây:

|  |  |
| --- | --- |
| Icon | Trạng thái |
| C:\Users\Nguyen Trung\AppData\Local\Microsoft\Windows\Temporary Internet Files\Content.Word\camaro_256.png | Xe có thể phục vụ khách |
| C:\Users\Nguyen Trung\AppData\Local\Microsoft\Windows\Temporary Internet Files\Content.Word\Renault laguna1.png | Xe đang trên đường đón khách |
| C:\Users\Nguyen Trung\AppData\Local\Microsoft\Windows\Temporary Internet Files\Content.Word\red_256.png | Xe đang phục vụ khách |
| C:\Users\Nguyen Trung\AppData\Local\Microsoft\Windows\Temporary Internet Files\Content.Word\aston_martin.png | Xe vượt tốc độ |
| C:\Users\Nguyen Trung\Desktop\IConcar\Car.png | Xe gặp sự cố |
| C:\Users\Nguyen Trung\AppData\Local\Microsoft\Windows\Temporary Internet Files\Content.Word\Actions-im-kick-user-icon.png | Xe mà tài xế chưa đăng nhập vào hệ thống |

Bảng ‑: Trạng thái xe trong hệ thống

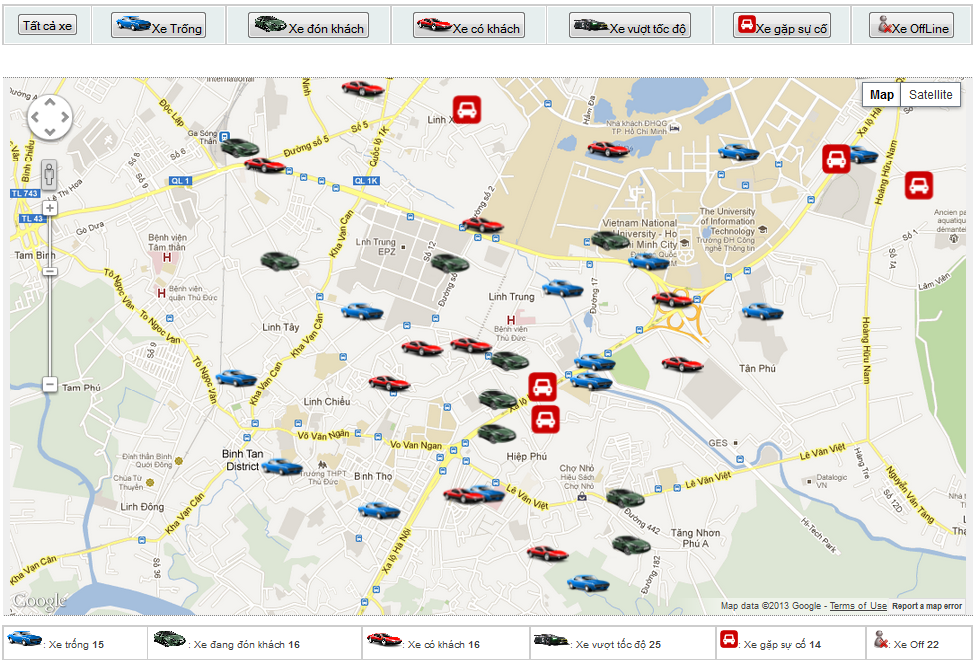
Vị trí của xe sẽ được hiển thị trực tiếp lên bản đồ như hình sau:



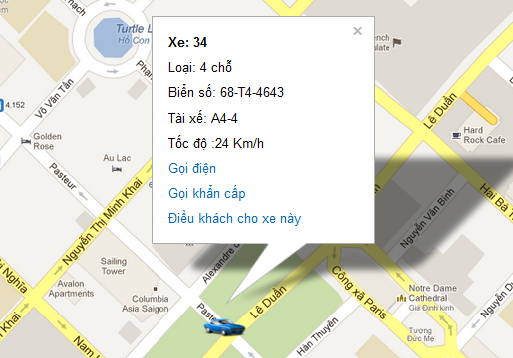
Hình ‑: Hiển thị vị trí xe

Để có thể hiển thị tất cả các xe trong hệ thống lên bản đồ, chúng ta sẽ bằng phương thức Ajax gửi yêu cầu đến các Web Service, Web Service sẽ xử lý yêu cầu, truy suất lấy dữ liệu từ MySQL Server, chuyển dữ liệu về định dạng Json và gửi dữ liệu về lại trình duyệt yêu cầu. Dữ liệu được trả về là một mảng các đối tượng lưu trữ thông tin về vị trí, trạng thái cũng như các thông tin cơ bản của xe taxi.

Tương ứng với mảng các đối tượng xe taxi này chúng ta sẽ tạo một mảng các đối tượng Maker, khởi tạo đối tượng Maker này với thông vị trí và trạng thái được lấy từ mảng dữ liệu thông qua Web Service. Để hiển thị thông tin của mỗi xe, chúng ta sử dụng đối tượng InfoWindow (google.maps.InfoWindow() ) để giúp cung cấp thêm thông tin về xe cho người điều phối. Sau khi đã khởi tạo các đối tượng Maker, InfoWindow, các đối tượng này sẽ được hiển thị lên bản đồ Google Map. Bằng cách này, nhân viên điều phối có thể thấy được vị trí, trạng thái của tất cả các xe trong hệ thống thông qua bản đồ sau:



Hình ‑: Bản đồ trạng thái của hệ thống



Hình ‑: Hiển thị thông tin một xe

Để có thể hiển thị hoạt động các xe một cách liên tục hệ thống sử dụng phương thức setInterval(), phương thức này sẽ gọi thực hiện một phương thức cụ thể lặp lại trong một khoảng thời gian đã được định trước.Chúng ta xẽ sử dụng phương thức này, kết hợp với Ajax để lấy dữ liệu trong cơ sở dữ liệu về trình duyệt. Dữ liệu này sẽ được xử lý để cập nhật vị trí lại cho Maker tương ứng với vị trí do Client gửi lên cho Server.

Ngoài ra, dựa trên trạng thái của xe, hệ thống có khả năng co biết số lượng xe theo trạng thái trong hệ thống, phân loại để hiển thị các xe lên bản đồ theo trạng thái, và một số chức năng có sử dụng bản đồ trong hệ thống như sau:

* Số lượng xe theo trạng thái.



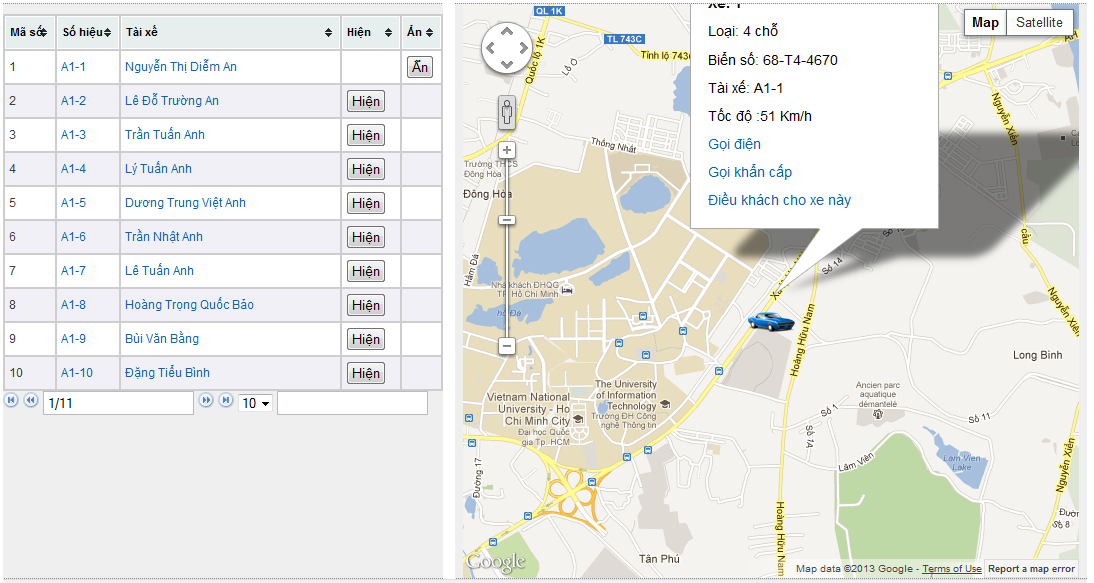
Hình ‑: Số lượng xe trong hệ thống

* Vị trí của các xe trống có khả năng phục vụ khách trong hệ thống.



Hình ‑: Hiển thị theo phân loại trạng thái xe

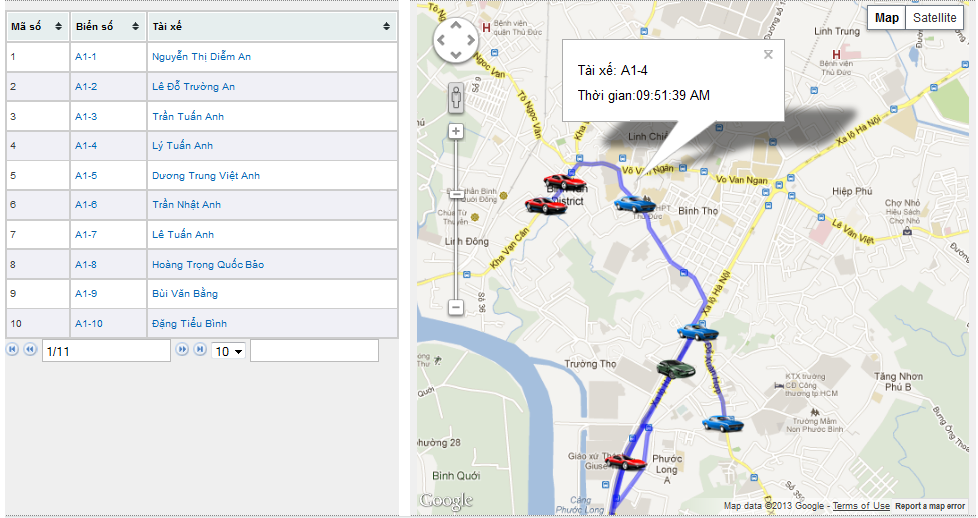
* Hiển thị vị trị và trạng thái của một xe cụ thể: danh sách tất cả các xe được hiện thị trong bảng, chúng ta có thể chọn một xe cụ thể để tiến hành giám sát.



Hình ‑: Giám sát hành trình của một xe

#### Hiển thị lịch sử hoạt động của xe

Trạng thái và vị trí của xe được client cập nhất lên Server sẽ được hệ thống lưu trự lại, thông tin lưu tữ bao gồm mã tài xế, tọa độ xe, ngày, giờ và trạng thái của xe. Dữ liệu được lưu tại MySQL Server và được tải bằng cách sử sụng Ajax kết hợp với Web Service. Trạng thái của xe được thể hiện bằng những Maker với mỗi icon khác nhau cho mỗi trạng thái khác nhau.



Hình ‑: Lịch sử hoạt động của một xe

## Xây dựng ứng dụng Client

### Các xử lý về thông tin GPS

Đa số các điện thoại sử dụng hệ điều hành android đều sử dụng chip xử lý có khả năng nhận tín hiệu GPS từ vệ tinh. Vì vậy android cung cấp một bộ thư viện dùng để giao tiếp và xử lý các thông tin lấy từ GPS nằm trong package android.location. Trong package này gồm có các thành phần như LocationManager, Location… Nhưng quan trọng nhất là lớp location, lớp này đại diện cho tập hợp các thông tin thu từ vệ tinh bao gồm: kinh độ, vỹ độ, các tùy chọn về cao độ, tốc độ, độ chính xác của vị trí….

Cách thức việc lấy thông tin GPS được thực hiện thông qua các nhà cung cấp. Có hai nhà cung cấp chính ở đây là GPS\_PROVIDER và NETWORK\_PROVIDER được đặt trưng bởi lớp provider. Lớp này được khởi tạo thông qua hàm getBestProvider(), Android sẽ tự động chọn nhà cung cấp có độ chính xác cao nhất tại thời điểm hiện tại cho chúng ta. Một số hàm cơ bản để lấy thông tin về Location cơ bản là:

Location mLocation = null;

mLocation.getAccuracy(); // Trả lại thông tin là độ chính xác của GPS.

mLocation.getAltitude(); // Trả lại thông tin là cao độ của vị trí hiện tại.

mLocation.getLatitude(); // Trả lại thông tin là Latitude;

mLocation.getLongitude(); // Trả lại thông tin là Longutude;

mLocation.getSpeed(); // Trả lại thông tin là tốc độ hiện tại;

Để có thể nhận biết sự thay đổi về vị trí khi người dùng di chuyển, chúng ta cần phải thực hiện implement một interface là LocationListener để có được những hàm lắng nghe sự thay đổi về Location. Interface này gồm có 4 hàm chính với chức năng khác nhau nhưng quan trọng nhất là hàm onLocationChanged(Location arg0) dùng để xử lý khi có sự thay đổi về tọa độ.

@Override

**public** **void** onLocationChanged(Location location) {

// **TODO** Auto-generated method stub

}

@Override

**public** **void** onProviderDisabled(String provider) {

// **TODO** Auto-generated method stub

}

@Override

**public** **void** onProviderEnabled(String provider) {

// **TODO** Auto-generated method stub

}

@Override

**public** **void** onStatusChanged(String provider, **int** status, Bundle extras) {

// **TODO** Auto-generated method stub

}

Những quyền cần thiết để có thể truy xuất thông tin về GPS tối thiểu cần có bao gồm:

* Android.permission.ACCESS\_FINE\_LOCATION: quyền này giúp chúng ta có thể sử dụng LocationManager.GPS\_PROVIDER để xác định vị trí thông qua GPS.
* Android.permission.ACCESS\_FINE\_LOCATION hay android.permission.ACCESS\_COARSE\_LOCATION: những quyền này giúp chúng ta có thể sử dụng LocationManager.NETWORK\_PROVIDER để xác định vị trí hiện tại.

### Các xử lý và triển khai Google Direction API

Chức năng tìm đường là một trong các dịch vụ mà google cung cấp cho người sử dụng. Đối với việc phát triển ứng dụng thì google cung cấp Direct API để thực hiện điều này. Hệ thống tìm đường này cung cấp tìm đường cho ba loại phương tiện chính bao gồm: xe hơi, xe buýt và đi bộ. Tuy nhiên tùy thuộc vào quốc gia mà dịch vụ này có được cung cấp đầy đủ và chính xác không. Ở Việt Nam thì chủ yếu sử dụng dịch vụ tìm đường cho xe hơi nhưng độ chính xác vẫn chưa được cao. Direct API đã được giới thiệu cụ thể ở chương trước, vì vậy ở phần này chúng ta chủ yếu tập trung vào các bước thực hiện. Như đã biết, để có thể sử dụng được dịch vụ này chúng ta cần gửi các yêu cầu lên server của Google. Cấu trúc URL như sau:

<http://maps.google.com/maps/api/directions/json?origin=lat1,long1&destination=lat2,long2&language=vi&sensor=true&mode=driving>

Các tham số cần chú ý là:

* Origin= latitude và longitude của điểm bắt đầu.
* Destination= latitude và longitude của điểm kết thúc.
* Language= ngôn ngữ sử dụng cho các chỉ dẫn, google chỉ hỗ trợ một số ngôn ngữ trong đó có tiếng việt.
* Sensor = giá trị này xác định có sử dụng hay không các cảm biến trong việc xác định vị trí và tìm đường.
* Mode= giá trị này chỉ ra chế độ tìm đường đang được sử dụng.

Đối với phiên bản hiện tại mặc định giá trị trả về từ các dịch vụ của Google là tập tim theo dịnh dạng Json. Phân tích thông tin từ tập tin này chúng ta sẽ có được các chỉ dẫn về tìm đường và danh sách các điểm để có thể vẽ đường trên bản đồ. Các thành phần chính từ tập tin trả về bao gồm nhiều thẻ khác nhau, mỗi thẻ chứa một thông tin riêng.

* Thẻ “routes”: là thẻ chính lớn nhất của tập tin Json. Nó chứa toàn bộ thông tin mà chúng ta cần có.
* Thẻ “bounds”: là thẻ chứa tọa độ của hai điểm mút thuộc hướng đông bắc và tây nam.
* Thẻ “copyrights”: là thẻ chứa thông tin về bản quyền thông tin này.
* Thẻ “legs”: là thẻ chứa thông tin về việc tìm đường. Thẻ “legs” bao gồm nhiều thẻ con bên trong:
  + Thẻ “distance”: chứa thông tin về khoảng cách giữa hai điểm.
  + Thẻ “duration”: chứa thông tin về thời gian duy chuyển giữa hai điểm.
  + Thẻ “end\_address”, “end\_location”,”start\_address”,”start\_location”: chứa thông tin về điểm bắt đầu và điểm kết thúc.
  + Thẻ “steps”: chứa thông tin về những đoạn đường nhỏ khác nhau, mỗi đoạn này chứa các thông tin như: khoảng cách, thời gian, điểm bắt đầu, điểm kết thúc, chỉ dẫn, và danh sách các tọa độ dùng để vẽ đoạn đường này trên bản đồ.
  + Thẻ “overview\_polyline”: chứa thông tin về tập hợp tất cả tọa độ của quãng đường, nó đươc chuyển thành mã ascii. Vì vậy muốn sử dụng chúng ta phải giải mã để thu được danh sách các điểm này.

Xử lý các thông tin này chúng ta chúng ta có thể vẽ đoạn đường này trên bản đồ và hiển thị những thông tin cần thiết.

### Các xử lý cơ bản trên Google Map API v2

#### Những cải tiến mới trong Google Map API v2 so với Google Map API v1

Bản đồ được gói trong class MapFragment, và kế thừa từ class Fragment của Android. Với MapFragment chúng ta có thể thêm bản đồ vào một phần của một Activity lớn. Vì vậy chúng ta không những có thể hiển thị bản đồ trong các thiết bị có màn hình nhỏ như thiết bị di động mà còn có thể hiển thị nó trong một giao diện đồ họa phức tạp của các loại thiết bị di động có màn hình lớn như máy tính bảng.

Map API được xây dựng trên nền tảng Vector. Vì vậy khối lượng dữ liệu trả về sẽ ít hơn, việc hiển thị bản đồ trong ứng dụng của chúng ta sẽ nhanh hơn và ít tốn dung lượng hơn.

Bộ đệm dữ liệu được cải thiện, vì vậy người dùng sẽ không còn thấy các khoảng trống khi hiển thị bản đồ.

Bản đồ có khả năng thể hiện 3D. Bằng cách di chuyển điểm nhìn, người sử dụng có thể thấy bản đồ ở các góc nhìn khác nhau theo quy luật xa gần.

API còn bổ sung thêm các đối tượng mới nhằm hỗ trợ tối đa cho người phát triển ứng dụng như Polyline, Polygon, TileOverlay, GroundOverlay…

#### Đánh dấu vị trí trên bản đồ bằng Marker

Marker đại diện cho một vị trí trên bản đồ. Chúng ta có thể thay đổi các tính chất cơ bản của Marker như màu sắc, vị trí, hình đại diện… Marker được hướng dựa vào màn hình của thiết bị hơn là bề mặt bản đồ vì vậy khi ta thực hiện các hành động như xoay, phóng to thu nhỏ, nghiêng thì sẽ không là thay đổi sự định hướng của Marker. Chúng ta có thể tương tác với Marker thông qua một số hành động được cung cấp bởi Google Map API bao gồm click và kéo thả.

Để thêm một marker vào trong bản đồ chúng ta cần phải sử dụng hàm addMarker() của đối tượng GoogleMap với tham số là MarkerOption.

private GoogleMap mMap;  
mMap = ((MapFragment) getFragmentManager().findFragmentById(R.id.map)).getMap();  
mMap.addMarker(new MarkerOptions()  
       .position(new LatLng(0, 0))  
        .title("Hello world"));

Marker Option là class dùng để chứa toàn bộ thông tin về các thuộc tính của Marker, các thuộc tính này bao gồm:

* Position: là thuộc tính cần phải thiết lập đối với bất kỳ marker nào, dùng để xác định vị trí gắn marker.
* Title: chứa thông tin sẽ hiển thị trong tiêu đề của InfoWindow.
* Snippet: chứa thông tin sẽ hiển thị trong nội dung của InfoWindow.
* Draggable: chứa thuộc tính có hay không cho người dùng kéo thả Marker.
* Visible: chứa thuộc tính có hay không hiển thị marker.
* Anchor: chứa điểm trên ảnh được dùng để gắn vào bản đồ, mặc định là giữa ảnh.
* Incon: hình ảnh sẽ được hiển thị tại vị trí đặt marker, chúng ta không thể thay đổi thuộc tính này đối với các marker đã được gắn vào bản đồ.

Tính năng nổi bật trong Google Map API v2 là cung cấp việc hiển thị thông tin của Marker bằng InfoWindow. InfoWindow sẽ được hiển thị thông qua việc nhấn nhẹ vào Marker. Trên bản đồ cùng lúc chỉ có thể hiển thị được một window duy nhất, window hiện tại sẽ ẩn đi nếu một window khác được hiển thị. Chúng ta cũng có thể hiện thị cửa sổ này thông qua hàm showInfoWindow() trên Marker và ngược lại sử dụng hàm hideInfoWindow() để ẩn cửa sổ.

Chúng ta cũng có thể thay đổi nội dung và thiết kế của InfoWindow bằng cách tạo ra một lớp kế thừa của InfoWindowAdapter() interface sau đó gọi hàm GoogleMap.setInfoWindowAdapter() để thiết lập InfoWindow mới tạo thành mặc định. Interface này có hai phương thức chính mà chúng ta phải kế thừa là getInfoWindow(Marker) và getInfoContents(Marker). API sẽ gọi hàm getInfoWindow(Marker) trước, nếu trả lại giá trị null nó sẽ gọi hàm getInforContents(Marker) tiếp theo. Nếu giá trị trả về vẫn là null thì giao diện cửa sổ mặc định sẽ được hiển thị.

Hàm getInfoWindow() cho phép chúng ta thay đổi giao diện của toàn bộ infoWindow. Hàm getInfoContents() cho phép chúng ta thay đổi nội dung của infoWindow còn khung và hình nền vẫn được giữ nguyên.

**class** CustomInfoWindow **implements** InfoWindowAdapter

{

@Override

**public** View getInfoContents(Marker arg0) {

// Các hàm xử lý

**return** **null**;

}

@Override

**public** View getInfoWindow(Marker arg0) {

// Các hàm xử lý

**return** **null**;

}

}

Map API cho phép chúng ta bắt các sự kiện click vào marker thông qua phương thức OnMarkerClickListener, để thiết lập phương thức này trên bản đồ ta gọi hàm GoogleMap.setOnMarkerClickListener(OnMarkerClickListener). Khi người dùng click vào một marker thì mặc định InfoWindow sẽ hiện ra và di chuyển camera bản đồ vào vị trí của marker vừa được click. Phương thức này trả về một giá trị Boolean dùng để xác định người dùng có sử dụng event này không. Ngoài ra, API còn cho phép bắt sự kiện click vào InfoWindow thông qua phương thức OnInfoWindowClickListener. Để thiết lập phương thức này chúng ta gọi hàm GoogleMap.setOnInfoWindowClick(OnInfoWindowClick), khi chúng ta click vào InfoWindow thì hàm onInfoWindowClick(Marker) sẽ được gọi, và info window được click sẽ hiện sáng hơn so với bình thường.

#### Vẽ đường thẳng trên bản đồ bằng Polyline

Polyline là một đối tượng mới trong Google Map API v2 được sử dụng để vẽ hình trên bản đồ bằng các đường thẳng nối với nhau. Chúng ta có thể thay đổi các thuộc tính của Polyline như màu sắc, chiều rộng của nét vẽ, thuộc tính ẩn. Đối tượng Polyline bao gồm danh sách các điểm trên bản đồ, sau đó các điểm này sẽ được nối với nhau bằng những đường thẳng.

Để tạo ra một đối tượng Polyline trước tiên chúng ta cần khởi tạo một đối tượng PolylineOptions dùng để chứa các thuộc tính của Polyline và thêm danh sách các điểm vào nó, những điểm này được xác định trên bản đồ vì vậy chúng ta khởi tạo chúng như những đối tượng LatLng. Đường thẳng nối các điểm thực hiện theo thứ tự của cách điểm trong danh sách lưu trong đối tượng PolylineOptions. Để vào PolylineOption chúng ta sử dụng phương thức PolylineOption.add().

// Khởi tạo đối tượng PolylineOption và thêm danh sách các điểm vào đối tượng vừa khởi tạo.  
PolylineOptions rectOptions = new PolylineOptions()  
        .add(new LatLng(37.35, -122.0))  
        .add(new LatLng(37.45, -122.0))     
        .add(new LatLng(37.45, -122.2))     
        .add(new LatLng(37.35, -122.2))     
        .add(new LatLng(37.35, -122.0));   
  
// thiết lập màu của đường thẳng nối các điểm  
rectOptions.color(Color.RED);  
  
// Thêm Polyline vào trong bản đồ  
Polyline polyline = myMap.addPolyline(rectOptions);

Các thuộc tính cơ bản của Polyline bao gồm:

* Color: dùng để thiết lập màu của đường nối các điểm, giá trị này được thiết lập thông qua phương thức PolylineOptions.color(), chúng ta cũng có thể thay đổi được thuộc tính này sau khi đã thêm Polyline vào bản đồ bằng phương thức Polyline.setcolor().
* Width: dùng để thiết lập bề ngang của đường nối các điểm tính bằng pixel(px) thông qua phương thức PolylineOptions.width().
* Visibility: là giá trị boolea dùng để xác định có hay không việc hiển thị Polyne bằng phương thức PolylineOptions.visible().

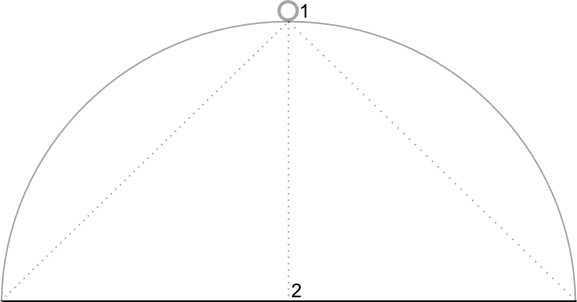
#### Thay đổi, di chuyển góc nhìn trong bản đồ

Google Map API v2 cung cấp thêm một đối tượng hoàn toàn mới là camera được sử dụng để thay đổi góc nhìn của người dùng trong bản đồ. Google Map v2 hỗ trợ hiển thị 3D vì vậy người dùng có thể xoay và nghiêng bản đồ bằng các thao tác đơn giản làm tăng khả năng tương tác của người dùng và bản đồ cũng như khả năng định hướng. Vì vậy đối tượng camera hết sức quan trọng trong Google Map API.

Việc thay đổi góc nhìn của người dùng thông qua đối tượng camera không làm thay đổi bất cứ marker, overlay hay các đối tượng đồ họa khác mà chúng ta đã thêm vào bản đồ trước đó.

CameraPositon là đối tượng dùng để tập hợp các thuộc tính dùng để thiết lập cho Camera, bao gồm các thuộc tính cơ bản như sau:

* Target(Location): dùng để thiết lập tọa độ để đặt góc nhìn, xác định thông qua latitude và longitude.
* Zoom: dùng để thiết lập độ phóng đại, độ phóng đại càng cao thì càng hiển thị rõ chi tiết của bản đồ.
* Tilt: dùng để thiết lập góc nhìn nghiêng trên bản đồ, điểm nhìn sẽ được đặt trên một cung tròn trên bản đồ, Khi sử dụng góc nhìn nghiêng thì bản đồ sẽ tự động xuất hiện la bàn trên góc trên bên trái để khôi phục lại góc nhìn thẳng như mặc định.



Hình ‑: Vị trí trước



Hình ‑: Vị trí sau

Muốn thay đổi vị trí của camera trước tiên chúng ta cần phải xác định chúng ta cần di chuyển camera đến đâu bằng cách sử dụng đối tượng CameraUpdate. Map API cho phép chúng ta tạo ra rất nhiều loại CameraUpdate khác nhau thông qua sử dụng CameraUpdateFactory. Các thuộc tính cơ bản bao gồm:

* Thay đổi độ phóng đại: chúng ta có thể sử dụng hàm CameraUpdateFactory.zoomIn() và CameraUpdateFactory.zoomOut() để thay đổi độ phóng đại của một đối tượng CameraUpdate theo từng cấp độ hay có thể sử dụng hàm CameraUpdateFactory.zoomTo(float) để thay đổi độ phóng đại đến một giá trị xác định.
* Thay đổi vị trí: Có hai phương thức phổ biến để thay đổi vị trí là:
  + CameraUpdateFactory.newLatLng(LatLng): thay đổi vị trí góc nhìn đến một vị trí xác định trên bản đồ bằng giá trị tọa độ Latitude và Longitude. Trong khi giữ nguyên các thuộc tính khác.
  + CameraUpdateFactory.newLatLngZoom(LatLng, float): Kết hợp thay đổi vị trí và độ phóng đại.
    - Phương thức đầy đủ nhất để thay đổi vị trí của camera là sử dụng phương thức:

CameraUpdateFactory.newCameraPosition(CameraPosition) trong đó CameraPosition được khởi tạo thông qua việc sử dụng phương thức new CameraPosition hay CameraPosition.Builder().

Sau khi đã có được đối tương CameraUpdate, Map API cung cấp cho chúng ta hai phương thức để di chuyển góc nhìn trên bản đồ đó là:

* GoogleMap.moveCamera(CameraUpdate): phương thức này thì góc nhìn sẽ được duy chuyển nhanh đến vị trí mà người dùng không cảm thấy được sự di chuyển.
* GoogleMap.animateCamera(cameraUpdate, callback, duration): phương thức này thì góc nhìn sẽ di chuyển mượt hơn, tạo cho người dùng cảm thấy góc nhìn đang di chuyển trên bản đồ đến vị trí cần thiết.Các tham số cần thiết ở đây bao gồm:
  + cameraUpdate: là đối tượng cameraUpdate để xác định vị trí mà camera sẽ di chuyển tới.
  + callback: là phương thức được gọi khi thực hiện việc di chuyển.
  + duration: là thời gian di chuyển từ vị trí ban đầu đến vị trí mới.

Việc Google Map API v2 cung cấp một phương thức giúp thay đổi góc nhìn người dùng giúp tăng khả năng tương tác của người dùng, tạo sự thoải mái và tiện dụng khi sử dụng.

### Truy vấn dữ liệu từ Web Service

Việc truy vấn dữ liệu từ server được sử dụng thông qua phương thức GET của giao thức HTTP, kết quả trả về là tập tin Json. Các truy vấn này giúp Server có thể biết được trạng thái cũng như vị trí hiện tại của taxi. Sau đây là một số truy vấn được sử dụng trong chương trình:

* Khi xe không có khách và ở trạng thái nhận khách từ Server thì truy vấn gửi lên Server sẽ là:

[http://localhost:8084/TaxiManagementSystem/resources/update?username={user}&lat={lat}&long={long}&speed={speed}&state=0&idCustomer=null&token={token}](http://localhost:8084/TaxiManagementSystem/resources/update?username=%7buser%7d&lat=%7blat%7d&long=%7blong%7d&speed=%7bspeed%7d&state=0&idCustomer=null&token=%7btoken%7d)

* Khi xe ở trạng thái đang đi đón khách sau khi nhận khách từ Server:

[http://localhost:8084/TaxiManagementSystem/resources/update?username={user}&lat={lat}&long={long}&speed={speed}&state=3&idCustomer={id}](http://localhost:8084/TaxiManagementSystem/resources/update?username=%7buser%7d&lat=%7blat%7d&long=%7blong%7d&speed=%7bspeed%7d&state=3&idCustomer=%7bid%7d)

* Khi xe ở trong trạng thái có khách thì truy vấn gửi lên server sẽ là:

[http://localhost:8084/TaxiManagementSystem/resources/update?username={user}&lat={lat}&long={long}&speed={speed}&state=1&idCustomer={id}](http://localhost:8084/TaxiManagementSystem/resources/update?username=%7buser%7d&lat=%7blat%7d&long=%7blong%7d&speed=%7bspeed%7d&state=1&idCustomer=%7bid%7d)

* Khi xe trả khách thì truy vấn gửi lên server sẽ là:

[http://localhost:8084/TaxiManagementSystem/resources/update?username={user}&lat={lat}&long={long}&speed={speed}&state=1&idCustomer={id}](http://localhost:8084/TaxiManagementSystem/resources/update?username=%7buser%7d&lat=%7blat%7d&long=%7blong%7d&speed=%7bspeed%7d&state=1&idCustomer=%7bid%7d)

* Khi xe đón khách dọc đường:

[http://localhost:8084/TaxiManagementSystem/resources/update?username={user}&lat={lat}&long={long}&speed={speed}&state=2&idCustomer=unknow&token={token}](http://localhost:8084/TaxiManagementSystem/resources/update?username=%7buser%7d&lat=%7blat%7d&long=%7blong%7d&speed=%7bspeed%7d&state=2&idCustomer=unknow&token=%7btoken%7d)

Khi có lỗi xảy ra trong quá trình truy vấn lên Server sẽ thông báo cho người sử dụng biết và sẽ thực hiện lại truy vấn vừa xảy ra lỗi.

### Các bước để thiết lập một phiên truyền thông bằng giao thức SIP

#### Thiết lập các thông số trong tập tin Manifest

Nếu muốn xây dựng chương trình sử dụng SIP API thì chúng ta cần phải đảm bảo điều kiện là thiết bị phải chạy trên Android phiên bản 2.3 trở lên.

Để có thể sử dụng SIP API chúng ta cần thêm vào hai sự cho phép trong tập tin manifest đó là: android.permission.USE\_SIP và android.permission.INTERNET. Để đảm bảo rằng ứng dụng của chúng ta chỉ có thể được cài đặt trên các thiết bị Android 2.3 trở lên thì cần thêm vào một thiết lập nữa là:

* <uses-sdk android:minSdkVersion="9" />
* <uses-deature android:name=”android.hardware.sip.voip”>Thiết lập này dùng để điều khiển làm sao chương trình có thể lọc được các thiết bị không hỗ trợ SIP API.
* Để có thể nhận được cuộc gọi từ một tài khoảng SIP khác thì chúng ta cần phải chỉ định một lớp con của BroadcastReceiver trong tập tin manifest:
* <receiver android:name=".IncomingCallReceiver" android:label="Call Receiver"/>

#### Thiết lập các thành phần cơ bản trong SIP API

1. Tạo ra một SipManager

Để sử dụng SIP API chúng ta bắt buộc phải tạo ra một đối tượng SipManager. Đối tượng này sẽ làm nhiệm vụ quản lý ứng dụng của chúng ta bao gồm:

* Thiết lập một phiên truyền thông SIP.
* Thiết lập và nhận cuộc gọi.
* Đăng ký và hủy đăng ký với nhà cung cấp dịch vụ SIP.
* Kiểm tra trạng thái phiên kết nối.

Chúng ta sử dụng phương thức sau để tạo ra một đối tượng SipManager:

public SipManager mSipManager = null;

if(mSipManager == null) {

mSipManager = SipManager.newInstance(this);

}

1. Đăng ký với một Sip Server

Một ứng dụng SIP thông thường bao gồm một hay nhiều người dùng, mỗi người dùng sử dụng một tài khoảng SIP. Mỗi tài khoảng SIP được đại diện bởi một đối tượng SipProfile. Tài khoảng được đăng ký trên thiết bị được gọi là local profile. Tài khoản Sip được kết nối tới gọi là peer profile. SipProfile được tạo ra thông qua phương thức SipProfile.Builder:

public SipProfile mSipProfile = null;

...

SipProfile.Builder builder = new SipProfile.Builder(username, domain);

builder.setPassword(password);

mSipProfile = builder.build();

Để có thể nhận cuộc gọi từ bên ngoài chúng ta phải thiết lập một hành động android.Sipdemo.INCOMING\_CALL, được sử dụng để tạo ra bộ lọc các intend khi thiết bị nhận cuộc gọi từ bên ngoài.

Intent intent = new Intent();

intent.setAction("android.SipDemo.INCOMING\_CALL");

PendingIntent pendingIntent = PendingIntent.getBroadcast(this, 0, intent, Intent.FILL\_IN\_DATA);

mSipManager.open(mSipProfile, pendingIntent, null);

Cuối cùng thiết lập một SipRegistrationListener trên SipManager được dùng để xác định tài khoảng Sip có đăng ký thành công với nhà cung cấp dịch vụ sip không.

mSipManager.setRegistrationListener(mSipProfile.getUriString(), new SipRegistrationListener() {

public void onRegistering(String localProfileUri) {

updateStatus("Registering with SIP Server...");

}

public void onRegistrationDone(String localProfileUri, long expiryTime) {

updateStatus("Ready");

}

public void onRegistrationFailed(String localProfileUri, int errorCode,

String errorMessage) {

updateStatus("Registration failed. Please check settings.");

}

Sau khi chương trình sử dụng xong một SipProfile, chúng ta cần đóng nó để giải phóng bộ nhớ và ngắt đăng ký với Server.

public void closeLocalProfile() {

if (mSipManager == null) {

return;

}

try {

if (mSipProfile != null) {

mSipManager.close(mSipProfile.getUriString());

}

} catch (Exception ee) {

Log.d("WalkieTalkieActivity/onDestroy", "Failed to close local profile.", ee);

}

}

1. Thực hiện một cuộc gọi thoại

Để có thể thực hiện được một cuộc gọi thoại trước tiên chúng ta cấn các điểu kiện sau:

* SipProfile thực hiện cuộc gọi và địa chỉ Sip được gọi phải tốn tại và hoàn thành đăng ký với Server.
* Khởi tạo thành công đối tượng SipManager.

Bước tiếp theo chúng ta thiết lập phương thức SipAudioCall.Listener để có thể nhận biết được các sự kiện tương tác với nhau trong quá trình thiết lập cuộc gọi.

SipAudioCall.Listener listener = new SipAudioCall.Listener() {

@Override

public void onCallEstablished(SipAudioCall call) {

call.startAudio();

call.setSpeakerMode(true);

call.toggleMute();

}

@Override

public void onCallEnded(SipAudioCall call) {

}

};

Một khi đã thiết lập xong phương thức SipAudioCall.Listener, chúng ta có thể thiết lập cuộc gọi thông qua phương thức makeAudioCall của đối tượng SipManager. Các tham số bao gồm:

* Địa chỉ Sip dùng đễ gọi.
* Địa chỉ Sip được gọi.
* Một phương thức SipAudioCall.Listener để lắng nghe các sự kiện từ SipAudioCall.
* Thời gian timeout.

call = mSipManager.makeAudioCall(mSipProfile.getUriString(), sipAddress, listener, 30);

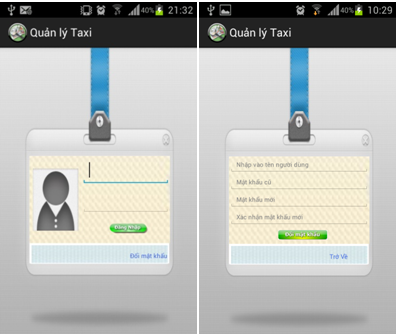
1. Nhận cuộc gọi

Để nhận cuộc gọi, chúng ta phải khai báo một subclass của BroadcastReceiver có khả năng trả lời một intend khi nhận ra nó là một cuộc gọi tới. Sau đây là những thông báo cần thiết đễ thực hiện điều này:

* Trong tập tin Manifest khai báo một thẻ <receiver>.
* Tạo là một class kế thừa từ BroadcastReceiver.
* Thiết lập một tài khoảng SIP kết nối đền intend để có thể nhận cuộc gọi từ bên ngoài.
* Thiết lập một bộ lọc intend để có thể nhận biết được cuộc gọi đến.

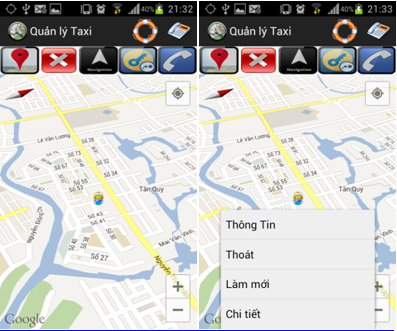
### Giao diện chương trình

Màn hình đăng nhập: được sử dụng để tài xế đăng nhập tài khoản, sau khi chứng thực thành công trên server thì tài xế mới có thể sử dụng được chương trình.



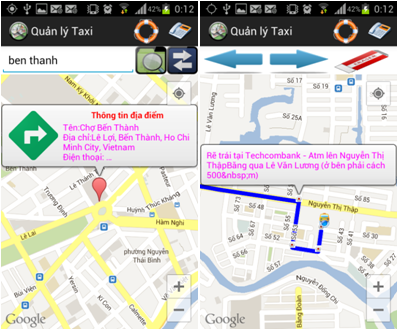
Hình ‑: Màn hình đăng nhập

Màn hình chính: là giao diện của Google Map v2, ngoài ra còn có những nút trên thanh điều hướng để thông báo trạng thái của taxi về trung tâm điều khiển, các tính năng của từng nút đã được trình bày trong hình 3-8.



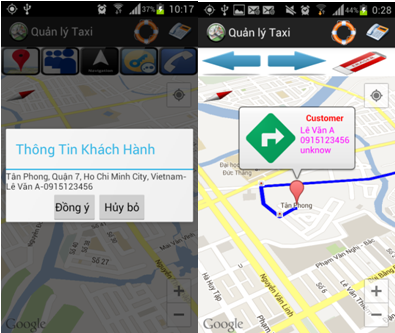
Hình ‑: Màn hình chính

Màn hình chỉ đường và tìm địa điểm: Sau khi tìm kiếm địa điểm bằng cách nhập vào tên địa điểm sau đó bấm nút tìm kiếm. Để có thông tin cụ thể về địa điểm ta tài xế nhấn vào điểm đánh dấu. Khi muốn tìm đường thì người dùng nhấn vào thông tin vừa hiển thị. Giao diện tìm đường sẽ hiện ra, người dùng sử dụng những nút tới lui để xem chỉ dẫn hoặc vào mục chi tiết trong menu.



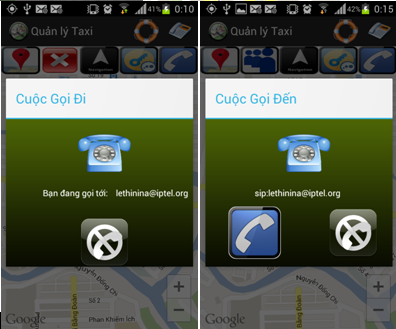
Hình ‑: Màn hình chỉ đường và tìm địa điểm

Màn hình nhận khách từ server: Khi có khách gửi từ server xuống thì có 2 lựa chọn là đồng ý và hủy bỏ. Nếu đồng ý thì chương trình sẽ tự động hiển thị đường đi tới vị trí của khách.



Hình ‑: Màn hình nhận khách từ server

Màn hình khi có cuộc gọi tới và đi: Chức năng này tương tự như thoại thông thường, màn hình giao diện bao gồm 2 nút chính dùng để bắt máy và tắt cuộc gọi.



Hình ‑: Màn hình khi có cuộc gọi tới và đi

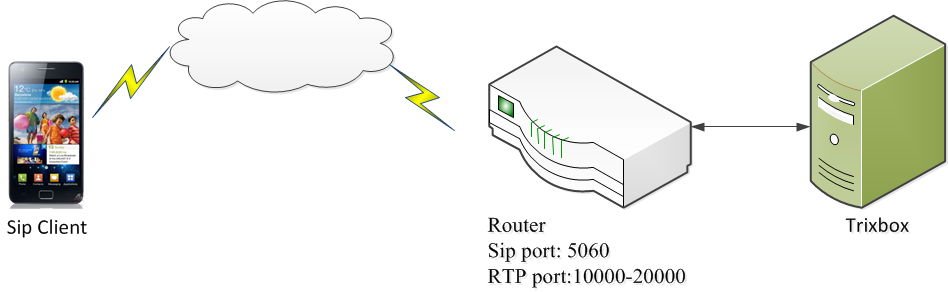
## Triển khai tổng đài Trixbox

### Triển khai tổng đài Tribox

Tổng đài phục vụ trong việc liên lạc giữa tài xế và nhân viên điều phối trong hệ thống được triển khai là tổng đài mã ngồn mở Trixbox CE 2.6 được cài đặt trên hệ thống máy ảo Vmware .Hệ thống này sẽ thay thế cho việc liên lạc bằng bộ đàm truyền thống đang được những nhà cung cấp dịch vụ taxi sử dụng. Giúp cho việc liên lạc trở nên thuận tiện và tiết kiệm chi phí đầu tư cơ sở hạ tầng.

Để tổng đài có thể phục vụ việc liên lạc thông qua mạng dữ liệu 3G chúng ta phải tiến hành public tổng đài này để cho các Sip Client có thể giao tiếp được với tổng đài thông qua mạng internet. Trong hệ thống này tổng đài sẽ được public bằng cách thực hiện phương thức NAT trên Router. Cụ thể chúng ta sẽ Nat các port tương ứng với các giao thức sau:

* Giao thức SIP: port 5060 sử dụng trong việc thiết lập, quản lý và kết thúc cuộc gọi.
* Giao thức RTP: port 10000-20000 sử dụng trong việc truyền dữ liệu trong cuộc gọi.



Hình ‑: Mô hình triển khai tổng đài trixbox

Hình vẽ trên mô tả hệ thống liên lạc thông qua tổng đài Trixbox. Cũng dựa trên tổng đài này hệ thống cung cấp cho nhân viên điều phối có khả năng tạo ra những phòng hội thoại giúp có thể liên lạc theo nhóm các tài xế và nhân viên điều phối.

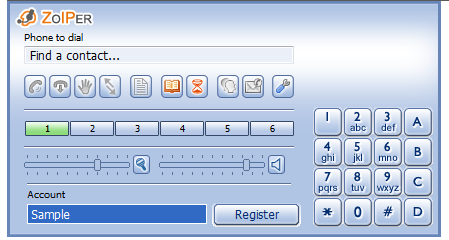
Mỗi nhân viên điều phối và tài xế taxi đều được cung cấp một tài khoản Sip quản lý bởi Trixbox. Thông tin tài khoản này sẽ được cung cấp cho tài xế khi đăng nhập, tài xế sẽ sử dụng chức năng gọi VOIP của client để thực hiện các cuộc gọi. Về phía nhân viên điều phối, sẽ thực hiện các cuộc gọi đến tài xế thông qua Zoiper Web được nhúng trong trình duyệt. Khi cần gọi tài xế nào chỉ cần nhấp chọn Maker của tài xế đó sau đó chọn chức năng “Gọi” hệ thống sẽ tự động cấu hình và thực hiện cuộc gọi đến tài xế. Ngoài lựa chọn trên nhân viên điều phối có thể sử dụng bất kỳ loại soft phone nào trên máy tính có hỗ trợ giao thức Sip để thực hiện việc liên lạc thông qua tổng đài này.

Để thực hiện việc tích hợp Softphone trên trình duyệt ta sẽ sử dụng Zoiper Web[[26]](#footnote-26). Zoiper Web được thiết kế cho phép nhúng trực trực tiếp vào các trang web và có khả năng tương thích với các trình duyệt phổ biến hiện nay. Zoiper Web cho phép người dùng sử dụng trình duyệt web để thực hiện và nhận các cuộc gọi VOIP sử dụng các giao thức SIP hoặc IAX2. API này cho phép lập trình viên kiểm soát Zoiper Web bằng cách sử dụng ngôn ngữ JavaScript.

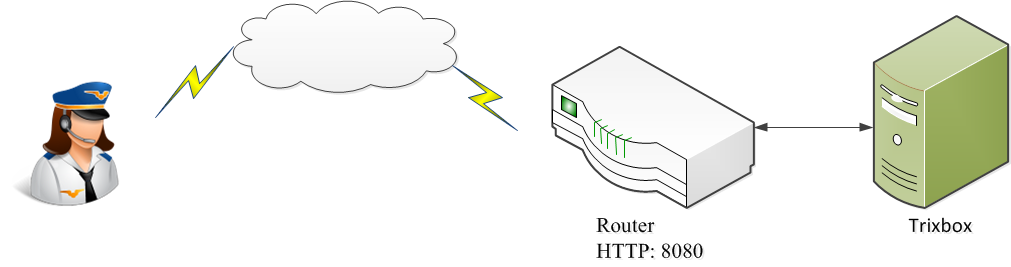
API này cung cấp các lớp với chức năng tạo, quản lý cấu hình, tài khoản, danh bạ và các cuộc gọi. Cung cấp các thông tin phản hồi từ các sự kiện của Zoiper Web đến trình duyệt. Tất cả các thuộc tính, phương thức và các tham số đều sử dụng dữ liệu là “string”.

API được xây dựng trên cơ sở lớp “Phone” cho phép quản lý các đối tượng “Config”, “Account”, “Call” and “Contact”.

Chúng ta sẽ sử dụng JavaScript khởi tạo các đối tượng này để thực hiện các cuộc gọi với Zoiper Web được nhúng vào trình duyệt.



Hình ‑: Zoiper Wep

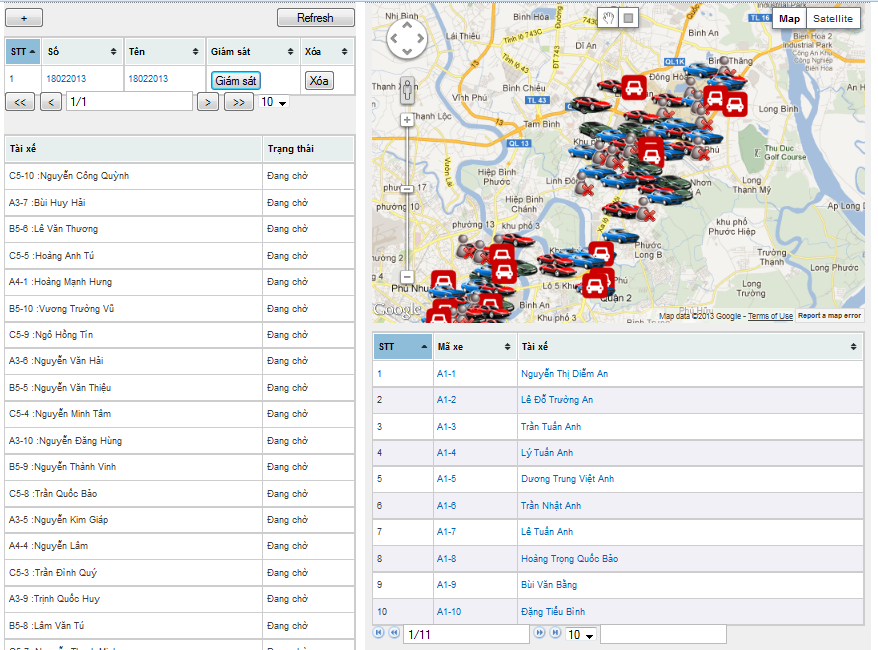
Tất cả các cuộc gọi giữa tài xế và nhân viên điều phối sẽ hoạt động trên nền mạng internet 3G.

Hình ‑: Quản lý phòng hội thoại

### Xây dựng thành phần quản lý phòng hội thoại

Việc sử dụng các phòng hội thoại giúp cho sự liên lạc giữa nhân viên điều phối và tài xế trở nên linh hoạt hơn, nhân viên điều phối có thể chọn liên lạc với các tài xế theo nhóm, theo từng khu vực cụ thể được hiển thị một cách trực quan trên bản đồ giúp hạn chế việc tài xế phải nhận thông báo không phù hợp với vị trí của mình như khi sử dụng bộ đàm. Để thực hiện chức năng cho phép nhân viên điều phối tạo các phòng hội thoại, chúng ta sẽ sử dụng việc khởi tạo và quản lý các conference trong tổng đài Trixbox thông qua giao thức HTTP.

Nhân viên điều phối sẽ sử dụng giao diện web của thành phần quản lý, cung cấp các thông số để khởi tạo conference. Ở đây nhân viên chỉ cần cung cấp số phòng và tên phòng, sau đó server Tomcat sẽ khởi tạo một yêu cầu HTTP trên cổng 8080 đến tổng đài Trixbox để yêu cầu khởi tạo một conference. Sau đó nhân viên điều phối sẽ thực hiện việc lựa chọn nhóm tài xế để gửi yêu cầu tham gia vào conference, nhân viên điều phối cũng có thể giám sát việc tham gia và rời khỏi phòng của tài xế.



Hình ‑: Giao diện quản lý phòng hội thoại

## Tổng kết

Chúng ta đã đi vào thiết kế chi tiết từng thành phần trong hệ thống đảm bảo hệ thống có thể đáp ứng đầy đủ những yêu cầu đã đề ra và có khả năng triển khai hoạt động trong thực tế.

# KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

## Tổng kết đề tài

Khóa luận đã tổng kết và trình bày một cách hệ thống và cụ thể về kiến thức nền tảng, kiến thức về công nghệ để xây dựng hệ thống ORDER-SYSTEM dựa trên nền tảng Google Map Service và một số công nghệ khác. Đề tài đã xây dựng được một ứng dụng hoàn chỉnh trên cả client và server bao gồm các chức năng chính như sau:

Trung tâm điều hành với các chức năng:

* Web Application cung cấp các chức năng quản lý và điều hành hệ thống sử dụng thông qua trình duyệt web nên rất tiện dụng và nhanh chóng.
* Lưu trữ dữ liệu đơn hàng có thể được sữ dụng bằng các phần mềm kế toán tự xây dựng.
* Tiếp nhận, xử lý thông tin yêu cầu phục vụ của khách hàng, tiến hdungjwphaan phối nhân viên giao hàng.
* Cho phép thêm các sản phẩm để người dùng có thể đặt hàng.

Ứng dụng client android application dùng cho nhân viên giao hàng:

* Hiện thị vị trí các đơn hàng được phân phối , chức năng tìm đường dựa trên Google Map.
* Cung cấp việc thông tin với khách hàng qua sms của điện thoạọa

Tóm lại, đề tài “XÂY DỰNG HỆ THỐNG ORDER-SYSTEM” đã hoàn thành các yêu cầu đặt ra và có khả năng triển khai hoạt động trong thực tế. Việc quản lý nhân viên giao hàng cũng được thực hiện dễ dàng hơn. Hệ thống được triển khai giúp có thêm một hướng tiếp cận mới trong việc quản lý và điều hành việc giao hàng trong bối cảnh hiện nay bằng cách tận dụng sự phát triển mạnh mẽ của các công nghệ và thiết bị di động những năm gần đây mà điển hình trong hệ thống này là Smartphone Android và công nghệ truyền tải thông tin thông qua mạng 3G. Đề tài được hoàn thành dựa trên nền tảng nghiên cứu lý thuyết, khảo sát yêu cầu đối với một hệ thống giao hàng trên thực tế giúp khắc phục những hạn chế của hệ thống truyền thống, tăng hiệu suất trong quá trình quản lý, điều hành hệ thống tại trung tâm điều khiển.

## Những khó khăn và hạn chế

Trong quá trình thực hiện nhóm tác giả đã gặp một số khó khăn khác nhau về công nghệ, thiết bị và việc thử nghiệm đề tài trong thực tế. Cụ thể những khó khăn hạn chế gặp phải như sau:

* Những công nghệ áp dụng đòi hỏi tính thực tế cao chưa có tài liệu tiếng việt nên tốn thời gian đầu tư nghiên cứu nhiều, đặc biệt là Google Map API v2.
* Những hạn chế về mặt dịch vụ cũng như độ chính xác của Google Map ở Việt Nam gây ra một số sai lệch trong việc tìm kiếm và phân giải địa chỉ.
* Trong quá trình thực hiện nhiều lần tuyến cáp quang AAG xãy ra sự cố ảnh hưởng đến quá trình thực hiện.

## Hướng phát triển đề tài

Đề tài bước đầu đã xây dựng được hệ thống ORDER-SYSTEM thông qua thiết bị di động nền tảng Android. Nếu có thêm thời gian và kinh phí đầu tư thêm từ nhà trường và các công ty thì tương lai có thể phát triển hệ thống một cách hoàn chỉnh hơn, không chỉ có chức năng quản lý mà còn hỗ trợ tối đa cho khách hàng và nhân viên giao hàng. Một số hướng phát triển như:

* Mở rộng hệ thống để sử dụng cho chuổi cửa hàng.
* Thêm khả năng quản lý thông qua việc sử dụng tổng đài VIOP để thông tin với khách hàng và nhân viên giao hàng.
* Xây dựng một bản đồ riêng cho khu vực Hồ Chí Minh để tăng tốc độ xử lý và giảm sai sót trong quá trình phân giải địa chỉ thành tọa độ trên bản đồ.
* Áp dụng các thuật toán để tính tự động hóa việc phân phối đơn hàng cho nhân viên giao hàng đồng thời tìm ra đường đi tối ưu cho nhân viên giao hàng .
* Áp dụng bản đồ lưu lượng giao thông để tránh các tình huống trễ do giao thông tắc nghẽn.

Màn hình tìm kiếm địa điểm

Màn hình chính

Màn hình đăng nhập

# DANH MỤC TÀI LIỆU THAM KHẢO

|  |  |
| --- | --- |
| [1] | (2012, Dec.) Android Map API. [Online]. https://developers.google.com/maps/documentation/android/reference/com/google/android/gms/maps/package-summary |
| [2] | Google Place API. [Online]. https://developers.google.com/places/documentation/ |
| [3] | A. El-Rabbany, "Chaper 1: Introduction to GPS," in *Introduction to GPS - The Global Positioning System*. Boston, London: Artech House, 2002, p. 1. |
| [4] | T. D. Le, *Web Programming*. UIT, 2010. |
| [5] | S. Jose, *RESTful Java Web Services*. Birmingham: Packt Publishing Ltd, 2009. |
| [6] | A. Vukotic and J. Goodwill, *Apache Tomcat 7*. Apress, 2011. |
| [7] | G. Svennerberg, *Beginning Google Maps API 3*. Apress, 2010. |
| [8] | (2012, Dec.) Android Developer. [Online]. http://www.developer.android.com |
| [9] | S. Allamaraju, *RESTful Web Services Cookbook*. O’Reilly Media, Inc, 2010. |
| [10] | B. Burke, *RESTful Java with JAX-RS*. O’Reilly Media, Inc, 2010. |
| [11] | Google. (2012, Dec.) Google Map API v2. [Online]. https://developers.google.com/maps/documentation/android/ |
| [12] | M. Kalin, *Java Web Services: Up and Running*. O’Reilly Media, Inc, 2009. |
| [13] | G. Kerry, *trixbox CE 2.6*. Birmingham: Packt Publishing Ltd, 2009. |
| [14] | R. Leonard and R. Sam, *RESTful Web Services*. O’Reilly Media, Inc, 2007. |
| [15] | T. A. Nguyen, L. T. K. Ton, and N. H. Tran, *Android programming tutorial*. UIT, 2011. |

1. http://www.thongtincongnghe.com/article/35078 [↑](#footnote-ref-1)
2. http://gps.vietmap.vn/giai-phap/taxi [↑](#footnote-ref-2)
3. http://vietbao.vn/Vi-tinh-Vien-thong/Mashup-mon-qua-thu-vi-cua-trao-luu-Web-20/11054949/217/ [↑](#footnote-ref-3)
4. Andy Rubin: http://en.wikipedia.org/wiki/Andy\_Rubin [↑](#footnote-ref-4)
5. Open Handset Alliance: http://www.openhandsetalliance.com/ [↑](#footnote-ref-5)
6. Thị phần các hệ điền hành trên thiết bị di động: http://dantri.com.vn/suc-manh-so/android-dang-thong-tri-thi-truong-smartphone-658334.htm [↑](#footnote-ref-6)
7. IDC : http://en.wikipedia.org/wiki/International\_Data\_Corporation [↑](#footnote-ref-7)
8. Thống kê trên Google Market: http://vi.wikipedia.org/wiki/Android [↑](#footnote-ref-8)
9. Android version history: http://en.wikipedia.org/wiki/Android\_version\_history [↑](#footnote-ref-9)
10. HTTP proxy: http://vi.wikipedia.org/wiki/Proxy\_server [↑](#footnote-ref-10)
11. http://vi.wikipedia.org/wiki/C%E1%BA%A5u\_tr%C3%BAc\_ARM. [↑](#footnote-ref-11)
12. Dalvik: http://en.wikipedia.org/wiki/Dalvik\_(software) [↑](#footnote-ref-12)
13. Google Play service SDK: http://developer.android.com/google/play-services/index.html [↑](#footnote-ref-13)
14. Google APIs Console: https://code.google.com/apis/console/ [↑](#footnote-ref-14)
15. Hash: http://en.wikipedia.org/wiki/Hash\_function [↑](#footnote-ref-15)
16. SHA-1: http://en.wikipedia.org/wiki/SHA-1 [↑](#footnote-ref-16)
17. Keytool: http://docs.oracle.com/javase/1.4.2/docs/tooldocs/windows/keytool.html [↑](#footnote-ref-17)
18. Hexa: http://en.wikipedia.org/wiki/Hexadecimal [↑](#footnote-ref-18)
19. Keystore: http://docs.oracle.com/javase/1.4.2/docs/api/java/security/KeyStore.html [↑](#footnote-ref-19)
20. Lockheed Martin: http://vi.wikipedia.org/wiki/Lockheed\_Martin [↑](#footnote-ref-20)
21. Composite: http://vi.wikipedia.org/wiki/V%E1%BA%ADt\_li%E1%BB%87u\_composite [↑](#footnote-ref-21)
22. Sóng mang trong điều chế tín hiệu: http://vi.wikipedia.org/wiki/%C4%90i%E1%BB%81u\_ch%E1%BA%BF\_t%C3%ADn\_hi%E1%BB%87u [↑](#footnote-ref-22)
23. Doppler: http://vi.wikipedia.org/wiki/Hi%E1%BB%87u\_%E1%BB%A9ng\_Doppler [↑](#footnote-ref-23)
24. http://www.json.org/index.html [↑](#footnote-ref-24)
25. https://sites.google.com/site/gson/gson-user-guide [↑](#footnote-ref-25)
26. Trích tài liệu ZOIPER WEB API [↑](#footnote-ref-26)