TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN

KHOA CÔNG NGHỆ THÔNG TIN

LỚP CỬ NHÂN TÀI NĂNG

HOÀNG VĂN NGỌC – NGUYỄN VĂN TƯỜNG

MẠNG XÃ HỘI CHIA SẺ ĐỊA ĐIỂM

TRÊN THIẾT BỊ ANDROID

KHOÁ LUẬN TỐT NGHIỆP CỬ NHÂN CNTT

TP.HCM, 2016

TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN

KHOA CÔNG NGHỆ THÔNG TIN

LỚP CỬ NHÂN TÀI NĂNG

HOÀNG VĂN NGỌC 1212264

NGUYỄN VĂN TƯỜNG 1212501

MẠNG XÃ HỘI CHIA SẺ ĐỊA ĐIỂM

tRÊN THIẾT BỊ ANDROID

KHÓA LUẬN TỐT NGHIỆP CỬ NHÂN TIN HỌC

GIÁO VIÊN HƯỚNG DẪN

TS. ĐINH BÁ TIẾN

NIÊN KHÓA 2012 – 2016

|  |
| --- |
| **NHẬN XÉT CỦA GIÁO VIÊN HƯỚNG DẪN**  ………………………………………………………………………………  ………………………………………………………………………………  ………………………………………………………………………………  ………………………………………………………………………………  ………………………………………………………………………………  ………………………………………………………………………………  ………………………………………………………………………………  ………………………………………………………………………………  ………………………………………………………………………………  ………………………………………………………………………………  ………………………………………………………………………………  ………………………………………………………………………………  ………………………………………………………………………………  ………………………………………………………………………………  ………………………………………………………………………………  Khóa luận đáp ứng yêu cầu của LV cử nhân tin học.  TP.HCM, ngày …… tháng …… năm 2016  Giáo viên hướng dẫn |
| **NHẬN XÉT CỦA GIÁO VIÊN PHẢN BIỆN**  ………………………………………………………………………………  ………………………………………………………………………………  ………………………………………………………………………………  ………………………………………………………………………………  ………………………………………………………………………………  ………………………………………………………………………………  ………………………………………………………………………………  ………………………………………………………………………………  ………………………………………………………………………………  ………………………………………………………………………………  ………………………………………………………………………………  ………………………………………………………………………………  ………………………………………………………………………………  ………………………………………………………………………………  ………………………………………………………………………………  Khóa luận đáp ứng yêu cầu của LV cử nhân tin học.  TP.HCM, ngày …… tháng …… năm 2016  Giáo viên phản biện |

LỜI CẢM ƠN

Chúng em xin chân thành cảm ơn Khoa Công Nghệ Thông Tin, trường Đại Học Khoa Học Tự Nhiên, Tp.HCM đã tạo điều kiện tốt cho chúng em thực hiện đề tài này.

Chúng em xin chân thành cảm ơn Thầy Đinh Bá Tiến đã tận tình hướng dẫn, chỉ bảo chúng em trong suốt thời gian thực hiện đề tài. Đồng thời quan tâm, khích lệ chúng em mỗi khi gặp khó khăn và luôn tạo mọi điều kiện thuận lợi nhất để chúng em hoàn thành đề tài đúng thời hạn.

Chúng em cũng xin gửi lời cảm ơn sâu sắc đến quý Thầy Cô trong Khoa đã tận tình giảng dạy, trang bị cho chúng em những kiến thức, kinh nghiệm quý báu trong những năm học vừa qua.

Cuối cùng, Chúng em xin gửi lòng biết ơn sâu sắc đến gia đình, các anh chị và bạn bè đã ủng hộ, giúp đỡ và động viên chúng em trong những lúc khó khăn cũng như trong suốt thời gian học tập và nghiên cứu.

Mặc dù chúng em đã cố gắng hoàn thành luận văn trong phạm vi và khả năng cho phép, nhưng chắc chắn sẽ không tránh khỏi những thiếu sót, kính mong sự cảm thông và tận tình chỉ bảo của quý Thầy Cô và các bạn.

Nhóm thực hiện.

ĐỀ CƯƠNG CHI TIẾT

|  |
| --- |
| **Tên Đề Tài**: Mạng xã hội chia sẻ địa điểm trên thiết bị Android. |
| **Giáo viên hướng dẫn**: TS. Đinh Bá Tiến. |
| **Thời gian thực hiện:** từ ngày 09/11/2015 đến ngày 30/06/2016 |
| **Sinh viên thực hiện:**  Hoàng Văn Ngọc (1212264) – Nguyễn Văn Tường (1212501) |
| **Loại đề tài:** Tìm hiểu công nghệ và xây dựng ứng dụng. |
| **Nội Dung Đề Tài**: Mục tiêu của đề tài được xác định là *tìm hiểu* và *đề xuất giải pháp công nghệ* để *xây dựng mạng xã hội nhằm chia sẻ địa điểm, khám phá những trải nghiệm cuộc sống* và tạo ra *môi trường tương tác thân thiện* cho người sử dụng.  Nội dung chi tiết của luận văn bao gồm :   * Tìm hiểu các giải pháp đang được sử dụng để xây dựng và thực hiện tương tác mạng xã hội trên thế giới. * Tìm hiểu các công nghệ được các công ty liên quan đến mạng xã hội và các ứng dụng đang dùng hoặc đã phát triển thành công. * Tìm hiểu các vấn đề liên quan đến cơ sở dữ liệu dạng đồ thị: cấu tạo, nguyên tắc hoạt động, cách thức tương tác và lập trình. * Tìm hiểu các vấn đề liên quan cũng như cách sử dụng Neo4j và ngôn ngữ truy vấn cơ sở dữ liệu Cypher. * Tìm hiểu xây dựng đa tiểu trình trên web service với Nodejs. * Xác định, phân tích các vấn đề khi xây dựng mạng xã hội và tương tác người dùng trên thiết bị di động. * Đề xuất phương pháp xây dựng mạng xã hội song song với tiến thành cài đặt thử nghiệm và khảo sát đánh giá từ người sử dụng. |

|  |  |
| --- | --- |
| **Kế Hoạch Thực Hiện:**   * 09/11/2015-30/11/2015: Khảo sát các mạng xã hội đã và đang phát triển trên thế giới – chủ yếu tập trung vào chức năng, cách tiếp cận và các luồng thực thi của ứng dụng. * 01/12/2015-15/12/2015: Xác định yêu cầu bài toán hiện tại, tiến hành thiết kế giao diện tương tác người dùng. * 16/12/2015-31/12/2015: Mô tả và phân tích các đặc trưng chính của ứng dụng sẽ xây dựng, tiến hành lựa chọn các công nghệ phù hợp nhất. * 01/01/2016-07/01/2016: Tìm hiểu cấu trúc lưu trữ trên cơ sở dữ liệu dạng đồ thị Neo4j và ngôn ngữ truy vấn Cypher. * 08/01/2016-15/01/2016: Tìm hiểu ngôn ngữ lập trình Nodejs và tiến hành cài đặt Web service liên kết ứng dụng với cơ sở dữ liệu. * 16/01/2016-23/01/2016: Xây dựng mô hình lưu trữ cơ sở dữ liệu, xác định các thực thể và liên kết theo đồ thị. * 24/01/2016-31/01/2016: Xây dựng giao diện người dùng trên thiết bị di động Android thể hiện được các chức năng cơ bản của ứng dụng. * 01/02/2016-15/02/2016: Viết các truy vấn và chuyển ứng dụng sang phụ thuộc hoàn toàn vào Web service. * 16/02/2016-29/02/2016: Tìm hiểu Google Map API ver 2 trên thiết bị Android và tiến hành cài đặt hệ thống bản đồ cho ứng dụng. * 01/03/2016-31/03/2016: Tìm hiểu Google Cloud Message và hoàn thành xây dựng các chức năng cơ bản của mạng xã hội. * 01/04/2016-30/04/2016: Chỉnh sửa giao diện tương tác với người dùng với phong cách thiết kế Material Design đặc trưng được Google giới thiệu cùng với hệ điều hành Android 5.0. * 01/05/2016-31/05/2016: Nâng cấp, tối ưu, hoàn thiện ứng dụng và các chức năng đã xây dựng. * 01/06/2016-30/06/2016: Khảo sát người dùng song song với chỉnh sửa tương tác phù hợp cho ứng dụng. | |
| **Xác nhận của GVHD** | **Ngày 31 tháng 06 năm 2016**  **Nhóm SV Thực hiện**  Hoàng Văn Ngọc – Nguyễn Văn Tường |

MỤC LỤC

[LỜI CẢM ƠN iii](#_Toc454198814)

[ĐỀ CƯƠNG CHI TIẾT iv](#_Toc454198815)

[MỤC LỤC vii](#_Toc454198816)

[DANH MỤC HÌNH ẢNH xi](#_Toc454198817)

[DANH MỤC BẢNG xii](#_Toc454198818)

[DANH MỤC THUẬT NGỮ, TỪ VIẾT TẮT xiii](#_Toc454198819)

[TÓM TẮT KHÓA LUẬN xiv](#_Toc454198820)

[Chương 1 Mạng xã hội – Một xu hướng kết nối toàn cầu 1](#_Toc454198821)

[1.1. Giới thiệu chung 1](#_Toc454198822)

[1.1.1. Lịch sử hình thành và phát triển của mạng xã hội 1](#_Toc454198823)

[1.1.2. Khảo sát hiện trạng các mạng xã hội đã và đang phát triển 4](#_Toc454198824)

[1.2. Lợi ích và tiềm năng của mạng xã hội 5](#_Toc454198825)

[1.3. Mục tiêu đề tài 6](#_Toc454198826)

[1.4. Nội dung luận văn 7](#_Toc454198827)

[Chương 2 Các giải pháp xây dựng mạng xã hội và cách tiếp cận 8](#_Toc454198828)

[2.1. Mạng xã hội Facebook 8](#_Toc454198829)

[2.1.1. Tổng quan 8](#_Toc454198830)

[2.1.2. Cách tiếp cận 9](#_Toc454198831)

[2.2. Mạng xã hội Youtube 12](#_Toc454198832)

[2.2.1. Tổng quan 12](#_Toc454198833)

[2.2.2. Cách tiếp cận 12](#_Toc454198834)

[2.3. Mạng xã hội Zalo 14](#_Toc454198835)

[2.3.1. Tổng quan 14](#_Toc454198836)

[2.3.2. Cách tiếp cận 14](#_Toc454198837)

[2.4. Kết luận 16](#_Toc454198838)

[Chương 3 Giải pháp đề xuất 17](#_Toc454198839)

[3.1. Cở sở dữ liệu đồ thị Neo4j 17](#_Toc454198840)

[3.1.1. Tổng quan 17](#_Toc454198841)

[3.1.2. Mô hình lưu trữ và tổ chức dữ liệu 18](#_Toc454198842)

[3.1.3. Ngôn ngữ truy vấn Cypher 20](#_Toc454198843)

[3.1.4. Sức mạnh của Neo4j 22](#_Toc454198844)

[3.2. Cấu trúc lưu trữ Quadtree 23](#_Toc454198845)

[3.3. Tìm kiếm toàn văn Lucene 26](#_Toc454198846)

[3.4. Hệ thống định vị toàn cầu GPS 28](#_Toc454198847)

[3.4.1. Tổng quan 28](#_Toc454198848)

[3.4.2. Sử dụng GPS trên Android 28](#_Toc454198849)

[3.5. Google services API 30](#_Toc454198850)

[3.5.1. Google Maps API 30](#_Toc454198851)

[3.5.2. Google Places API 32](#_Toc454198852)

[3.5.3. Google Cloud Messaging 34](#_Toc454198853)

[3.5.4. Google Play Services 36](#_Toc454198854)

[3.6. Xây dựng máy chủ với công nghệ NodeJS 37](#_Toc454198855)

[3.6.1. Tổng Quan 37](#_Toc454198856)

[3.6.2. Đặc tính của NodeJS 37](#_Toc454198857)

[3.6.3. Sử dụng NodeJS. 38](#_Toc454198858)

[3.7. Phong cách thiết kế Material Design 38](#_Toc454198859)

[3.8. Kết luận 40](#_Toc454198860)

[Chương 4 Ứng dụng HOMFY 41](#_Toc454198861)

[4.1. Giới thiệu tổng quan 41](#_Toc454198862)

[4.2. Mô hình use-case 42](#_Toc454198863)

[4.2.1. Sơ đồ use-case 42](#_Toc454198864)

[4.2.2. Đặc tả use-case 43](#_Toc454198865)

[4.3. Mô hình dữ liệu 45](#_Toc454198866)

[4.3.1. Sơ đồ dữ liệu 45](#_Toc454198867)

[4.3.2. Đặc tả cơ sở dữ liệu 46](#_Toc454198868)

[4.4. Thiết kế kiến trúc 47](#_Toc454198869)

[4.5. Các chức năng của ứng dụng 47](#_Toc454198870)

[4.6. Kịch bản sử dụng ứng dụng 47](#_Toc454198871)

[4.7. Kết luận 47](#_Toc454198872)

[Chương 5 Thực nghiệm và đánh giá 48](#_Toc454198873)

[5.1. Kết quả thực nghiệm 48](#_Toc454198874)

[5.2. Tổng kết đánh giá 48](#_Toc454198875)

[5.3. Kết luận 48](#_Toc454198876)

[Chương 6 Kết luận và hướng phát triển 49](#_Toc454198877)

[6.1. Các kết quả đạt được 49](#_Toc454198878)

[6.2. Hướng phát triển của đề tài 49](#_Toc454198879)

[TÀI LIỆU THAM KHẢO 50](#_Toc454198880)

[PHỤ LỤC 51](#_Toc454198881)

DANH MỤC HÌNH ẢNH

DANH MỤC BẢNG

DANH MỤC THUẬT NGỮ, TỪ VIẾT TẮT

TÓM TẮT KHÓA LUẬN

Ngày nay, trong bối cảnh điều kiện kinh tế và đời sống của cộng đồng ngày càng phát triển, nhu cầu liên lạc với bạn bè, chia sẻ và cập nhật thông tin đang dần trở nên thiết yếu đối với mọi người. Chúng ta có thể liên lạc với mọi người gần như ở bất cứ nơi đâu, nắm được các thông tin cần thiết, thú vị và tạo mối quan hệ gần gũi góp phần nâng cao chất lượng đời sống tinh thần cũng duy trì quan hệ tốt đẹp với mọi người.

Cùng lúc đó, sự phát triển của Công nghệ thông tin và đặc biệt là quá trình bùng nổ của mạng xã hội ảo đã mang lại nhiều sự thay đổi lớn và sâu sắc cho cuộc sống con người. Với sự hỗ trợ đắc lực của những thiết bị di động, thông qua các ứng dụng mạng xã hội, chúng ta dễ dàng liên lạc với bạn bè, người thân. Bên cạnh đó, nhu cầu chia sẻ và tìm kiếm những địa điểm thú vị cũng như lưu giữ lại cho bản thân những khoảnh khắc tuyệt vời đang dần trở nên phổ biến và cần thiết với tất cả mọi người. Chúng ta có những album, bộ sưu tập, những hình ảnh kỷ niệm khó phai về một nơi nào đó, vậy thì tại sao không chia sẻ với mọi người cũng như giúp kỷ niệm ùa về mỗi lần vô tình đi qua nơi đó.

Nội dung đề tài chúng em tập trung vào việc tìm hiểu và xây dựng một mạng xã hội ảo phục vụ cho việc kết nối bạn bè, chia sẻ và khám phá những địa điểm thú vị mà bạn bè đã trải nghiệm. Cụ thể là ứng dụng cho phép người dùng kết nối với nhau và chia sẻ ảnh về địa điểm, thể hiện góc nhìn trực quan trên bản đồ, tìm kiếm những địa điểm khác quanh vị trí hiện tại cũng như tất cả mọi vị trí trên thế giới. Không chỉ dừng lại ở đó, chúng em còn tìm hiểu cách phát triển hệ thống tối ưu với bài toán hiện tại, ứng dụng thân thiện, dễ sử dụng, dễ dàng nâng cấp, thay đổi để trở nên hấp dẫn và phong phú hơn.

# Mạng xã hội – Một xu hướng kết nối toàn cầu

* Nội dung Chương 1 trình bày tổng quan về mạng xã hội cũng như lợi ích và tiềm năng khi xây dựng và phát triển mạng xã hội, từ đó nêu rõ mục tiêu đề tài của khóa luận. Nội dung tóm tắt của từng chương trong khóa luận được trình bày ở cuối chương này.

## Giới thiệu chung

### Lịch sử hình thành và phát triển của mạng xã hội

Trong xã hội ngày nay, giao tiếp là một nhu cầu cơ bản và cần thiết. Mọi người mong muốn liên lạc, tìm kiếm bạn bè để chia sẻ và trao đổi thông tin. Trước khi internet ra đời thì việc giao tiếp giữa con người với con người còn rất hạn chế vì những lý do như không gian và thời gian. Hầu hết mọi người liên lạc với nhau thông qua thư từ theo cách cổ điển và hoàn toàn chưa có khái niệm mạng xã hội, nơi kết nối mọi người trên khắp thế giới.

Sự ra đời phát triển của internet là một nguồn động lực to lớn đã thúc đẩy sự hình thành của một cách thức, phương tiện giao tiếp mới, đó là mạng xã hội. Với mạng xã hội, mọi người có thể giao lưu, chia sẻ cảm xúc và suy nghĩ cá nhân đến bạn bè, người thân hay là kết bạn với những người có khoảng cách về địa lý.

Năm 1971, email đầu tiên được gửi với nội dung “qwertyuiop” giữa 2 máy tính đánh dấu sự xuất hiện của mạng xã hội trong thế giới thực [1]. Năm 1978, Bulletin Board System (BBS) là mạng xã hội đầu tiên cho phép mọi người giao tiếp với một người khác thông qua internet để chia sẻ dữ liệu [1]. Năm 1994, cùng với khái niệm World Wide Web (www), trang mạng xã hội đầu tiên ra đời là Geocities [1]. Geoticities cho phép người dùng có thể tự do chỉnh sửa thông tin, trang cá nhân của mình.



Hình 1 Mạng xã hội Geocities [1]

Instant Messager và SixDegree.com xuất hiện năm 1997. Với Instant Messager, người dùng có thể tự do gửi tin nhắn đến bạn của mình hay là tạo lập trang thông tin cá nhân, viết nhật ký và chia sẻ đến mọi người. Do đó, có thể nói rằng Intant Messager là tiền thân của những mạng xã hội hiện nay. SixDegree.com có cùng tư tưởng với Instant Messager, ý tưởng cơ bản dựa trên nguyên lý “six degree of separation” [1]. Tuy nhiên SixDegree.com không gặt hái được nhiều thành công như Instant Messager. Một mạng xã hội tiêu biểu ra đời sau đó là Classmates, cho phép người dùng tìm kiếm và kết nối đối những bạn bè cũ. Cho đến ngày nay, Classmates vẫn tồn tại và có gần 60 triệu người dùng.

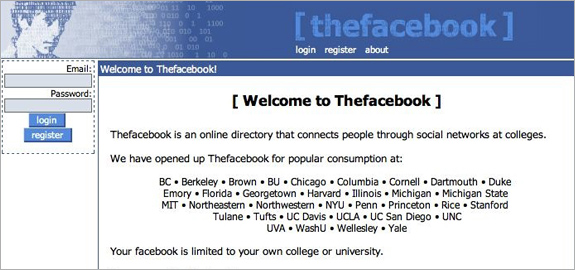
Vào những năm đầu thế kỉ 21, những mạng xã hội mới, hiện đại hơn đã xuất hiện. Đầu tiên đó là Friendster [2], một trang mạng xã hội mang thiên hướng về hẹn hò và là cải tiến của SixDegree. Trong 3 tháng đầu tiên, Friendster đã có đến 3 triệu người dùng. Điều đó có thể cho thấy sức hấp dẫn của mạng xã hội và sẽ sớm trở thành một xu thế mới.



(manjg xaxa hoij friendster [2])

Ngay sau đó, mạng xã hội MySpace ra đời. Chỉ trong một thời gian ngắn, MySpace trở nên phổ biến hơn cả Friendster. Cụ thể MySpace đã đạt gần 300 triệu người đăng ký sử dụng và là mạng xã hội được yêu thích nhất tại Mỹ mặc dù không còn là mạng xã hội dẫn đầu về người dùng [1]. Cùng thời điểm đó, một mạng xã hội về nghề nghiệp xuất hiện đó là Linkedin. Linkedin mang tới một diện mạo mạng xã hội mới, chuyên nghiệp hơn. Không còn chú trọng vào giao tiếp, kết bạn hay giải trí, Linkedin dành cho những có người mong muốn liên kết với những người khác với mục đích công việc, kinh doanh. Cho đến ngày nay, Linkedin vẫn đang là một trong những mạng xã hội lớn và đang tiếp tục phát triển với hơn 400 triệu người dùng [1].

Năm 2004, Facebook ra đời và đang là mạng xã hội dẫn đầu về người dùng trên thế giới hiện nay. Facebook bắt đầu với ý tưởng đơn giản là kết nối mọi người trong trường đại học Harvard với nhau. Trong tháng đầu tiên, có 19.500 sinh viên Harvard đăng ký sử dụng và hiện nay là 1.65 tỉ người dùng [3]. Ngoài những chức năng của một mạng xã hội thông thường, Facebook đã vượt ra những giới hạn cổ điển để cung cấp cho người dùng những tính năng tốt nhất như: chia sẻ ảnh, trực tiếp video, nhắc nhở, hỗ trợ lập trình ứng dụng và thậm chí là kinh doanh, marketing.



(fasfadsfsf asdf a sdfa sdf [2])

Từ lúc hình thành cho đến hiện tại, mạng xã hội đã trải qua nhiều giai đoạn phát triển với các hình thái khác nhau. Từ đơn giản về chức năng, giao diện cho đến phức tạp. Sẽ có nhiều mạng xã hội khác ra đời tùy thuộc vào sở thích, sự quan tâm của người dùng. Nhưng những hiểu biết của chúng ta về mạng xã hội vẫn còn rất hạn chế. Tuy nhiên, có thể khẳng định rằng mạng xã hội sẽ tiếp tục phát triển và là một đề tài thu hút nhiều sự quan tâm, nghiên cứu trong tương lai.

### Khảo sát hiện trạng các mạng xã hội đã và đang phát triển

Hiện tại, Facebook đang là mạng xã hội thu hút mọi sự chú ý và được sử dụng nhiều nhất. Facebook hướng đến mục tiêu kết nối mọi người trên thế giới lại với nhau. Chỉ bằng một vài thao tác đơn giản, chúng ta đã có thể giao tiếp với một người khác trên thế giới. Bên cạnh những tính năng cơ bản như tạo trang cá nhân, chia sẻ trạng thái, bình luận, nhắn tin thì Facebook còn hỗ trợ các tính năng nâng cao như: trực tiếp video, tìm kiếm bạn, gợi ý kết bạn hay nhắc nhở, … Ngoài ra, Facebook còn hỗ trợ một số tính năng giải trí và phục vụ marketing.

Khác biệt hầu hết các mạng xã hội mới, thì Youtube mang thiên hướng về chia sẻ video. Youtube được mua lại và phát triển bởi Google vào năm 2006. Người dùng có thể tìm kiếm những video mà mình thích và quan tâm hoặc là chia sẻ video cá nhân đến mọi người xung quanh.

So với các mạng xã hội trên thế giới thì Zalo là một mạng xã hội phiên bản thuần Việt còn mới, trẻ. Zalo được phát triển bởi tập đoàn VNG năm 2012, đi theo mô hình mobile-first và nhanh chóng thu hút được một lượng lớn người dùng ở Việt Nam. Cho đến tháng 4, 2016 thì Zalo đã đạt ngưỡng 50 triệu người dùng [4].

## Lợi ích và tiềm năng của mạng xã hội

Thực tế trong đời sống để giao tiếp với một khác là rất khó khăn vì những điều kiện khách quan như không gian, thời gian. Nhưng với mạng xã hội, tất cả trở nên đơn giản và thế giới xung quanh không còn bị giới hạn bởi địa lý. Chỉ với một vài thao tác, chúng ta đã có thể giao tiếp, liên lạc với mọi người trên khắp thế giới, tiết kiệm chi phí và thời gian. Bên cạnh đó, chúng ta có thể theo dõi và cập nhật thông tin về bạn bè, người thân hay một chủ đề tin tức nào đó mà chúng ta quan tâm. Với tốc độc lan truyền trên mạng xã hội, mọi tin tức có thể được cập nhật tức thời.

Giáo dục cũng là một lợi ích mà mạng xã hội mang lại. Một vài thống kê cho thấy hầu hết người sử dụng mạng xã hội là những người trẻ và nhu cầu chính của học là trao đổi với bạn bè. Các mạng xã hội như Facebook đều có hỗ trợ tính năng tạo lập nhóm. Qua đó, có thể giao tiếp và trao đổi thông tin học tập. Những hình thức như vậy hiện đang được sử dụng rất nhiều ở Việt Nam.

Ngoài ra mạng xã hội còn có thể dự đoán hành vi của con người thông qua dữ liệu. Người dùng thường có xu hướng ghi lại những hoạt động, cảm xúc của mình trên mạng xã hội. Thông qua đó, mạng xã hội có thể quan sát dữ liệu, phân tích và đưa ra dự đoán. Hướng nghiên cứu khai thác dữ liệu trên mạng xã hội cũng đang là một chủ đề thu hút nhiều sự quan tâm bởi những ứng dụng của nó trong đời sống thực tế. Chẳng hạn, trong lĩnh vực tìm kiếm tội phạm hay là quản lý con người.

Một lợi ích to lớn khác mà mạng xã hội mang lại đó là lợi ích về kinh doanh. Như chúng ta đã biết thì số lượng người sử dụng mạng xã hội rất lớn. Do đó, những hình thức kinh doanh hay marketing trên mạng xã hội sẽ rất hiệu quả bởi tốc độ lan truyền, chia sẻ nhanh. Chúng ta có thể thấy một số hình thức kinh doanh trực tuyến đã xuất hiện trên mạng xã hội, đang dần thay thế hình thức kinh doanh truyền thống và mang lại lợi ích to lớn.

Cùng với sự phát triển của công nghệ và nhu cầu của con người. Mạng xã hội cũng sẽ phát triển và thay đổi theo. Điển hình ở đây là nhu cầu du lịch và chia sẻ địa điểm. Đó chính là động lực thúc đẩy nhóm chúng em thực hiện đề tài xây dựng một mạng xã hội chia sẻ địa điểm kết hợp với mô hình mobile-first đang phổ biến hiện nay.

## Mục tiêu đề tài

Đề tài chúng em được thực hiện theo hướng tìm hiểu, nghiên cứu và áp dụng những công nghệ mới. Đề xuất giải pháp cho hệ thống, để từ đó có thể xây dựng một ứng dụng hoàn chỉnh, tối ưu.

Mục tiêu đề tài nhằm khảo sát hiện trạng, phân tích một số mạng xã hội đang nổi tiếng. Những ưu điểm, hạn chế của mạng xã hội đó. Tìm hiểu các công nghệ mới, đang được sử dụng nhiều hiện nay. Qua đó, đề xuất giải pháp xây dựng một mạng xã hội mới về chia sẻ địa điểm trên thiết bị Andoird, phần nào đáp ứng được nhu cầu du lịch của người dùng.

Nội dung thực hiện của đề tài bao gồm:

Khảo sát và phân tích hiện trạng các mạng xã hội:

* + - Khảo sát và phân tích mạng xã hội Facebook.
    - Khảo sát và phân tích mạng xã hội Youtube.
    - Khảo sát và phân tích mạng xã hội Zalo.
    - Tìm hiểu các công nghệ mà các mạng xã hội đã sử dụng.

Tìm hiểu công nghệ mới và đề xuất giải pháp:

* + - Giải pháp lưu trữ dữ liệu đồ thị với hệ quản trị cơ sở dữ liệu Neo4j.
    - Cách thức tổ chức và quản lý cơ sở dữ liệu khi dữ liệu lớn.
    - Cách thức tìm kiếm nhanh trên dữ liệu lớn.
    - Xu hướng mới xây dựng web server với NodeJS.
    - Chuẩn thiết kế giao diện theo phong cách Material Design của Google.
    - Công nghệ Push Notification với Google Cloud Messaging.
    - Các công nghệ định vị và truy xuất dữ liệu trên Google Map.

## Nội dung luận văn

Luận văn bao gồm 6 chương, sau đây là nội dung chính của từng chương.

**Chương 1:** giới thiệu tổng quan về mạng xã hội, khảo sát một số ứng dụng mạng xã hội đã và đang phát triển. Tiếp theo chúng em nêu lên động lực thực hiện đề tài, từ đó đề ra mục tiêu hướng tới của đề tài.

**Chương 2:** bình luận sâu về một số ứng dụng mạng xã hội, các giải pháp tiếp cận được các công ty mạng xã hội sử dụng cho ứng dụng của mình. Từ đó, phân tích đánh giá dựa trên mục tiêu bài toán và đề ra giải pháp riêng sử dụng phù hợp cho đề tài.

**Chương 3:** trình bày các công nghệ, các giải pháp tin học sẽ được sử dụng khi xây dựng ứng dụng. Song song với đó là phân tích đánh giá các ưu nhược điểm, tính phù hợp với bài toán hiện tại.

**Chương 4:** trình bày báo cáo chi tiết kỹ thuật về ứng dụng. Bao gồm bản vẽ kỹ thuật, các mô hình, lược đồ, quá trình phân tích thiết kế kiến trúc và hướng dẫn sử dụng ứng dụng.

**Chương 5:** trình bày kết quả thực nghiệm khi triển khai ứng dụng từ đó phân tích đánh giá và thể hiện góc nhìn trực quan hơn về người sử dụng với ứng dụng, rút ra các điểm yếu, điểm mạnh và hướng cải thiện ứng dụng cho phù hợp.

**Chương 6:** trình bày kết quả đạt được trong toàn quá trình xây dựng và phát triển ứng dụng. Bên cạnh đó là nhận ra các vấn đề phát sinh khi triển khai ứng dụng và đề xuất các giải pháp phát triển ứng dụng trong tương lai.

# Các giải pháp xây dựng mạng xã hội và cách tiếp cận

* Nội dung Chương 2 phân tích, bình luận về các ứng dụng mạng xã hội đã và đang phát triển dựa trên chức năng, giao diện và cách tiếp cận cũng như hướng giải quyết. Từ đó, chúng em áp dụng cho ứng dụng của đề tài, xác định các chức năng mới, đề ra các giải pháp phù hợp với yêu cầu hiện tại.

## Mạng xã hội Facebook

### Tổng quan

Facebook là một mạng xã hội lớn nhất hiện nay với chức năng chính là chia sẻ thông tin cá nhân, ảnh, video… Số lượng người sử dụng Facebook hiện tại khoảng 1.65 tỉ người dùng. Với lượng người sử dụng khổng lồ như vậy, Facebook đang phục vụ 350 triệu ảnh được đăng tải mỗi ngày [3]. Hơn 150 nghìn tin nhắn được gửi mỗi phút và xử lý 1.2 tỉ lượt truy cập ảnh mỗi giây [3]. Vấn đề được đặt ra là làm sao hệ thống có thể đáp ứng được một lượng lớn truy cập như vậy? Để giải quyết vấn đề đó, Facebook đã áp dụng một số công nghệ hỗ trợ xử lý khác bên cạnh việc tối ưu hóa về mặt thiết kế kiến trúc hệ thống.



(http://quangcaoadword.vn/wp-content/uploads/2015/03/thuc-day-bai-viet-cua-ban-voi-quang-cao-tren-facebook-1024x385.png)

### Cách tiếp cận

Về mặt kĩ thuật, trang mạng xã hội Facebook được phát triển dựa trên mô hình LAMP (Linux, Apache, MySQL, PHP) [5], một mô hình kĩ thuật phát triển phần mềm nổi tiếng lúc đó. Bên cạnh đó, Facebook cũng đã tích hợp thêm nhiều kĩ thuật, phần mềm khác để hệ thống vận hành một cách hiệu quả như ngày hôm nay. Chẳng hạn: Memcahed, Hiphop, BigPipe, Cassandra, Hadoop,… những kĩ thuật này vẫn còn đang được sử dụng rỗng rãi cho đến ngày hôm nay [5].

Ngôn ngữ lập trình được chọn để viết server đó là PHP. Một ngôn ngữ nổi tiếng được dùng cho việc viết web lúc bấy giờ và vẫn còn phổ biến ở hiện tại. PHP được tạo ra bởi Rasmus Lerdorf năm 1994 và nhanh chóng thu hút nhiều sự quan tính bởi những ưu điểm của nó trong lập trình web. Ưu điểm nổi bật là có thể phát triển ứng dụng web một cách nhanh chóng và dễ dàng với chi phí thấp. Với đặc thù tích hợp mã HTML vào trong ngôn ngữ, PHP cho phép các lập trình viên có thể tạo ra các trang thuận tiện hơn. Bên cạnh đó, PHP cũng hỗ trợ kết nối đến nhiều hệ quản trị cơ sở dữ liệu, đặc biệt là MySQL. Ngoài ra, PHP còn có thể thực thi trên nhiều máy chủ khác nhau mà không gặp vấn đề lỗi môi trường. Với những đặc tính đó, PHP sớm trở thành một ngôn ngữ được sử dụng rộng rãi với một cộng đồng người sử dụng đông đảo.

Vấn đề đặt ra đối với hầu hết các ứng dụng mạng xã hội đó là giải quyết việc có hàng triệu kết nối đến cùng một thời điểm. Tổ chức hệ thống như thế nào để ứng dụng có thể phản hồi trong một thời gian ngắn, không bị quá tải. Facebook đã sử dụng Memcahed như là một giải pháp để tăng tốc độ xử lý cho website [5]. Memcahed là một hệ thống phân phối và lưu trữ dữ liệu trên bộ nhớ đệm với hiệu suất cao. Cụ thể, Memcahed sử dụng RAM là nơi lưu trữ và dữ liệu được tổ chức theo key-value nên tốc độ truy xuất nhanh [6]. Ngoài ra, Memcahed còn có khả năng liên kết nhiều máy chủ thành cụm, hỗ trợ tối đa cho những hệ thống phân tán. Với những ưu điểm đó, Memcahed hiện đang được sử dụng nhiều trong các ứng dụng, hệ thống lớn khác như Flickr, Twitter, Youtube, WordPress, …



Hình 2 Lưu trữ dữ liệu trên bộ nhớ đệm với Memcahed(https://memcached.org/)

Như đã đề cập thì Facebook sử dụng ngôn ngữ PHP để lập trình cho server. Tuy nhiên, việc sử dụng PHP cho server cũng gây ra những hạn chế nhất định. Bởi vì PHP là một ngôn ngữ bậc cao và có tốc độ thực thi chậm. Do đó, Facebook đã đề xuất một nền tảng hỗ trợ mới là HipHop [5]. HipHop sử dụng một trình biên dịch JIT (Just In Time) có nhiệm vụ chuyển mã nguồn PHP sang mã nguồn C++ nhằm mục đích tận dụng tốc độ thực thi của ngôn ngữ này. Qua đó, cải thiện đáng kể tốc độ xử lý của website.

Chia sẻ ảnh là một tính năng nổi bật nhất của Facebook. Mỗi ngày Facebook có khoảng vài trăm triệu ảnh được đăng tải và được lưu trữ tương ứng với 4 kích thước ảnh khác nhau [7]. Yêu cầu đó đòi hỏi phải có một hệ thống lưu trữ ảnh hoạt động thật hiệu quả và chính xác. Và HayStack là đã được Facebook phát triển cho mục đích đó. HayStack hoạt động hiệu quả hơn so với hệ thống lưu trữ ảnh cũ như NFS. Về mặt kiến trúc, HayStack không còn tách biệt tầng truy xuất ảnh và tầng lưu trữ ảnh thành 2 phần riêng biệt, cũng như lưu từng ảnh sang từng file riêng biệt [5]. Một tầng server mới được hình thành để lưu trữ thông tin tổng quát về một đối tượng. Đối tượng ở đây có thể là nhiều ảnh. Điều đó làm tăng tốc độ truy xuất ảnh và giảm thiểu việc đọc các thông tin không cần thiết.

Bên cạnh việc lưu trữ dữ liệu trên hệ quản trị cơ sở dữ liệu MySQL truyền thống thì Facebook còn sử dụng thêm một số hệ quản trị cơ sử dữ liệu khác. Một trong số đó là Cassandra [5], một loại thuộc hệ NoSQL.



Hình 3 Hệ quản trị cơ sở dữ liệu Cassandra(https://en.wikipedia.org/wiki/Apache\_Cassandra)

Cassandra là một hệ quản trị cơ sở dữ liệu phân tán mã nguồn mở được xây dựng và phát triển bởi Facebook. Điểm mạnh của Cassandra đó là khả năng phân tán dữ liệu. Dữ liệu có thể được chia thành nhiều thành phần và lưu trữ ở nhiều nơi. Trong khi việc tương tác vẫn đơn giản, giống như làm việc với một khối dữ liệu thống nhất. Với tính năng đó thì việc mở rộng, sao lưu dữ liệu không gặp nhiều khó khăn. Một ưu điểm cở bản khác của hầu hết các loại hệ quản trị cơ sở dữ liệu NoSQL đó là tính hướng cột và dữ liệu được lưu trữ theo key-value. Hỗ trợ việc truy xuất nhanh và dễ dàng. Ngoài ra, Cassandra còn hỗ trợ việc chạy đa luồng với hiệu suất cao hay là tính chịu tải nặng, ổn định với dữ liệu lớn.

Ngoài Cassandra thì Facebook còn sử dụng hệ quản trị cơ sở dữ liệu trên hệ sinh thái đang nổi tiếng hiện nay đó là Hadoop [5]. Ưu điểm của Hadoop đó là việc lưu trữ và xử lý trên dữ liệu lớn. Hadoop gồm các thành phần xử lý chính: Map-Reduce, HDFS (Hadoop Distributed File System), YARN. Bên cạnh đó Hadoop cũng hỗ trợ nhiều framework khác để thuận tiện cho việc truy xuất dữ liệu. Với khối lượng dữ liệu khổng lồ hiện nay của Facebook thì hiển nhiên Hadoop là một lựa chọn tối ưu cho việc xử lý thông tin. Map-Reduce hỗ trợ việc khai thác dữ liệu bằng cách chia nhỏ công việc và tiền xử lý dữ liệu trước khi chạy thuật toán. HDFS là một hệ thống lưu trữ tập tin phân tán dựa trên hệ thống lưu trữ file của Google. Cung cấp cho Facebook một hệ thống lưu trữ dữ liệu ổn định và hiệu quả.

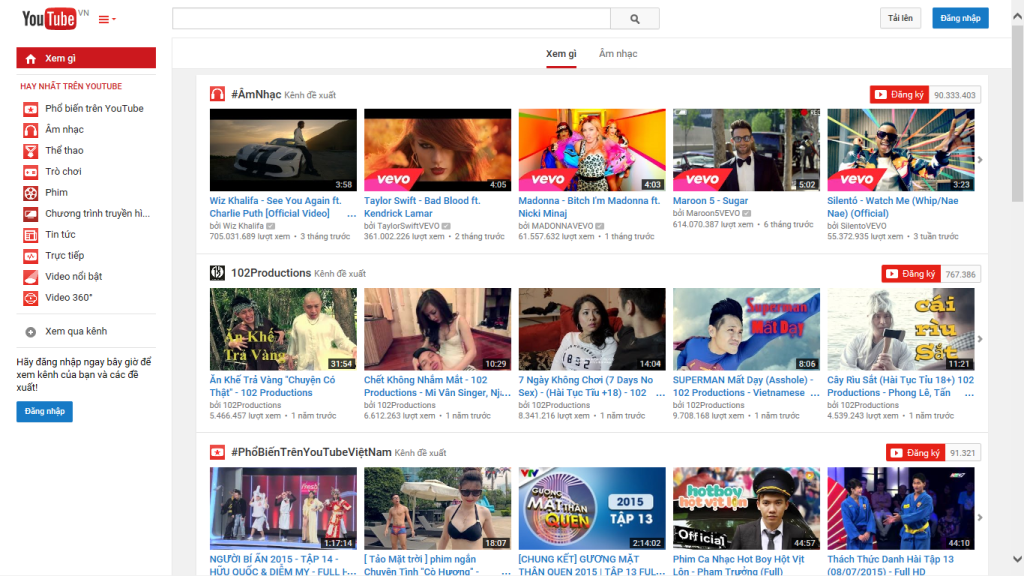


Hình 4 Hệ sinh thái Hadoop(https://en.wikipedia.org/wiki/Apache\_Hadoop)

## Mạng xã hội Youtube

### Tổng quan

Youbute cũng đang là một mạng xã hội chia sẻ video lớn hiện nay. Với hơn 1 tỉ người dùng, khoảng 4 tỉ lượt xem video mỗi ngày và hơn 500 giờ video được đăng tải mỗi phút [8].



(https://upload.wikimedia.org/wikipedia/vi/c/c0/YouTube.png)

Khác với các mạng xã hội khác, Youtube tập trung vào mảng xử lý, truyền tải dữ liệu dạng video. Do đó, vấn đề được đặt ra là làm sao phải đảm được bảo chất lượng của video khi có nhiều lượt truy cập. Việc đó đòi hỏi phải xây dựng một hệ thống tối ưu về việc lưu trữ video cũng như một server mạnh trong xử lý dữ liệu. Các giải pháp công nghệ đã được Google sử dụng để giải quyết các yêu cầu trên đó là Python, MySQL, Apache, Vitess, Zookeeper,… [9]

### Cách tiếp cận

Khác với hầu hết các ứng dụng khác, đó là lựa chọn những ngôn ngữ cổ điển như PHP, Java thì Google đã chọn Python, một ngôn ngữ mới để phát triển Youtube [9]. Python ra đời vào năm 1991 và nhanh chóng được sử dụng rộng rãi trong những ứng dụng cần nhiều tính toán và xử lý dữ liệu [10].



Hình 5 Ngôn ngữ lập trình Python ( [10])

Với mạng xã hội chia sẻ video như Youtube thì việc phân tích và xử lý dữ liệu là rất quan trọng, không phải ngôn ngữ nào cũng đáp ứng được điều đó. PHP không mạnh trong việc xử lý thông tin, Java có tốc độ tương đối chậm. Do đó, Python là một lựa chọn phù hợp, hiệu quả đối với một ứng dụng cần nhiều xử lý, tính toán như Youtube.

Cũng tương tự như Facebook, Youtube sử dụng hệ quản trị cơ sở dữ liệu MySQL. Nhưng để giải quyết vấn đề lưu trữ khi kích thước dữ liệu tăng nhanh, Youtue đã sử dụng Vitess, một công nghệ mã nguồn mở được phát triển bởi Google [9]. Vitess là giải pháp cho việc mở rộng hệ quản trị cơ sở dữ liệu MySQL. Vitess hỗ trợ việc phân tán dữ liệu trên những dịch vụ lưu trữ điện toán đám mây và vận hành hiệu quả không khác gì như trên thiết bị phần cứng. Bên cạnh đó, Vitess còn kế thừa một số đặc trưng của hệ quản trị cơ sở dữ liệu NoSQL như lưu trữ key-value, giúp cho việc mở rộng dữ liệu dễ dàng hơn. Một ưu điểm khác của Vitess đó là có thể dễ dàng quản lí hàng nghìn kết nối đến cơ sở dữ liệu mà không cần quan tâm đến bộ nhớ. Vì giao thức gRPC của Vitess cho phép tạo ra một kết nối đến MySQL mà không cần nhiều bộ nhớ như cách thông thường [11].

Python ra đời với mục tiêu đơn giản hóa tính toán, phân tích dữ liệu. Nhưng các ngôn ngữ hiện đại như Python thường có đặc điểm là thực thi chậm. Cho nên Python chỉ phù hợp cho những ứng dụng cần nhiều xử lý, không quan trọng về tốc độ. Đó chính là vấn đề được đặt ra đối với Youtube. Bởi vì một ứng dụng xem video trực tuyến thì yêu cầu tốc độ rất cao và thời gian phản hồi nhanh. Do đó, Google đã sử dụng Psyco là giải pháp cho việc nâng cao tốc độ thực thi của Python. Psyco là một loại trình biên dịch JIT (Just-In-Time), được viết bằng ngôn ngữ C và chuyên được sử dụng cho Python. Khi sử dụng Psyco, tốc độ thực thi của Python có thể tăng từ 2 lần đến 100 lần mà không có bất cứ sự chỉnh sửa mã nguồn nào [12].

## Mạng xã hội Zalo

### Tổng quan

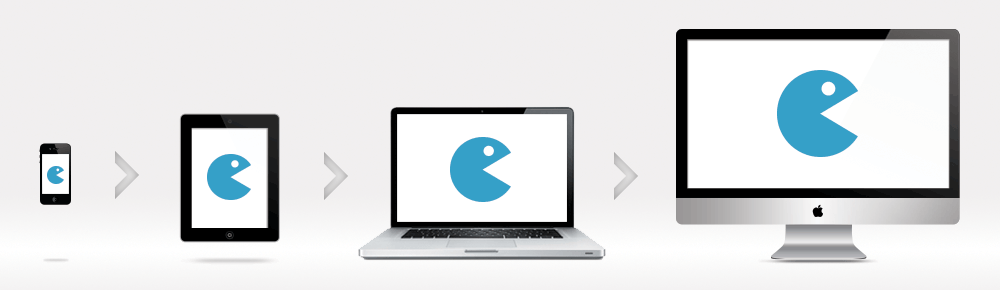


Hình 6 Mạng xã hội Zalo (https://vi.wikipedia.org/wiki/Zalo)

Zalo là một mạng xã hội mới ở Việt Nam, được phát triển bởi công ty VNG. Ra mắt vào năm 2012, sau khởi đầu khó khăn, Zalo đã vượt qua các ứng dụng lớn khác như Line, WeChat, Kakao Talk để thống lĩnh thị trường Việt. Không giống như các mạng xã hội khác, Zalo áp dụng mô hình Mobile-First, ưu tiên phát triển ứng dụng trên điện thoại di dộng [13]. Với mục đích ban đầu chỉ là xây dựng một ứng dụng nhắn tin trên điện thoại nhưng Zalo đã hòa nhập xu thế và phát triển thành một mạng xã hội với đông đảo người sử dụng. Về chức năng, Zalo từ bỏ việc đăng ký, đăng nhập tài khoản bằng ZingMe mạng xã hội anh em với Zalo. Bằng việc sử dụng số điện thoại để thay thế, Zalo đã loại bỏ những điều không cần thiết và cho người cảm thấy sự tiện lợi của ứng dụng. Ngoài chức năng chính là nhắn tin, gọi trực tuyến, Zalo còn cung cấp đầy đủ các chức năng cơ bản của một mạng xã hội giống như Facebook. Với những lý do đó, có thể trong tương lai Zalo sẽ vượt qua cả Facebook để trở thành ứng dụng mạng xã hội được sử dụng nhiều nhất ở Việt Nam.

### Cách tiếp cận

Theo thống kê thì có hàng tỉ thiết bị điện thoại truy cập vào hàng triệu trang web mỗi giây. Và con số đó sẽ còn tăng nhanh vì nhu cầu người dùng và thị trường thiết bị di động đang phát triển. Do đó trong tương lai người dùng sẽ có xu hướng sử dụng những thiết bị này để truy cập vào các trang web và mạng xã hội thay vì máy tính. Đón đầu xu thế đó, Zalo đã lựa chọn mô hình Mobile-First để xây dựng mạng xã hội. Với cách cổ điển, ứng dụng sẽ được phát triển trên web, sau đó sẽ được chuyển đổi để tương thích với điện thoại nhưng thường gây ra khó khăn trong việc thay đổi giao diện. Với Mobile-First, ứng dụng sẽ được ưu tiên thiết kế cho điện thoại trước. Bên cạnh đó, với chức năng chính là nhắn tin và gọi trực tuyến thì Mobile-First là lựa chọn chính đáng của Zalo.



Hình 7 Hướng tiếp cận thiết kế ứng dụng của Zalo

(http://chinrong.com/wp-content/uploads/original-ap.png)

Về mặt lưu trữ, VNG chọn hệ quản trị cơ sở dữ liệu PostgreSQL cho Zalo thay vì MySQL như những mạng xã hội khác [14]. Để so sánh giữa PostgreSQL và MySQL thì không có sự nổi trội nào cả. Nhưng mỗi cái có những ưu điểm và hạn chế nhất định. MySQL là lựa chọn tối ưu cho những ứng dụng lớn, muốn tối đa hóa khả năng lưu trữ, không quan trọng về vấn đề chi phí hay bản quyền. Trong khi PostgreSQL lại nổi bật trong việc thiết kế lưu trữ, hỗ trợ nhiều chức năng khác như người dùng có thể tùy biến kiểu dữ liệu, kế thừa trong bảng và được sử dụng hoàn toàn miễn phí [15].



Hình 8 Hệ quản trị cơ sở dữ liệu PostgreSQL

(https://madlab.org.uk/wp-content/uploads/2015/03/postgresql-logo.png)

Bên cạnh các yêu cầu khác thì tìm kiếm là một nhu cầu cơ bản của mọi ứng dụng. Do đó, Zalo đã chọn Elastic Search để nâng cao khả năng tìm kiếm của mình. Elastic Search được phát triển bởi Shay Banon, kế thừa từ Apache Lucene và là một giải pháp tìm kiếm thời gian thực mà không cần có cấu hình đặc biệt [14]. Elastic có thể hỗ trợ nhiều ngôn ngữ khác nhau như Java, JavaScirpt, PHP, Perl và được nhiều ứng dụng nổi tiếng sử dụng như Mozilla, Quora, Github, …

## Kết luận

Qua quá trình tìm hiểu một số công nghệ mà các mạng xã hội hiện có đã sử dụng, chúng ta có thể thấy rằng các công nghệ đã được dùng rất đa dạng. Tùy thuộc vào yêu cầu chức năng mà một số công nghệ sẽ được sử dụng sao cho phù hợp, tối ưu hóa hiệu quả. Không có công nghệ nào là nổi bật hơn hết. Mỗi công nghệ, kĩ thuật đều có những ưu điểm và khuyết điểm riêng, phụ thuộc vào mục tiêu mà ứng dụng hướng đến, nhà phát triển sẽ lựa chọn công nghệ tốt nhất cho ứng dụng.

# Giải pháp đề xuất

* Nội dung chương 3 trình bày các công nghệ tin học, các giải pháp được thực hiện khi xây dựng đề tài. Bên cạnh đó, chúng em còn phân tích tính phù hợp, các ưu nhược điểm, lý do chọn giải pháp đó và khả năng mở rộng khi phát triển ứng dụng.

## Cở sở dữ liệu đồ thị Neo4j

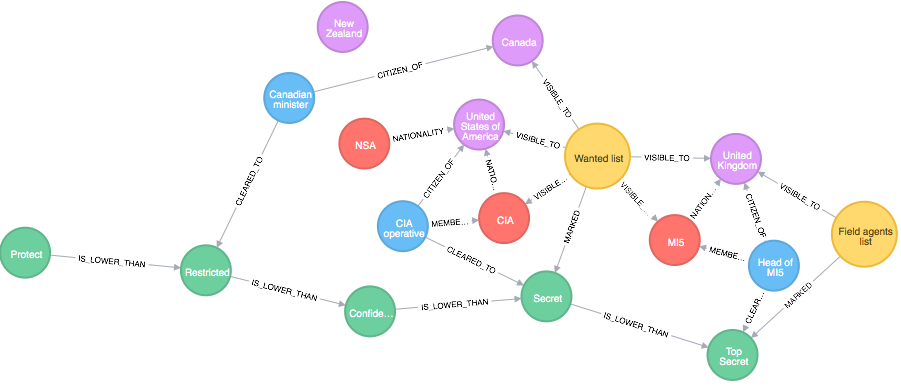
### Tổng quan

Trong toán học và tin học, đồ thị là đối tượng nghiên cứu cơ bản trong lí thuyết. Một cách không chính thức, có thể nói đồ thị là một tập các đối tượng gọi là đỉnh được nối lại với nhau bởi các cạnh. Thông thường, đồ thị được thể hiện dưới dạng một tập các điểm nối với nhau bằng các đoạn thẳng. Tùy theo nhu cầu sử dụng mà các cạnh có thể vô hướng hoặc có hướng. Cùng với đồ thị, nhiều bài toán thực tế xuất hiện và vẫn đang còn là thách thức với các nhà khoa học cho đến tận bây giờ.

Cở sở dữ liệu đồ thị là một cách tổ chức và lưu trữ dữ liệu dựa trên cấu trúc của đồ thị. Khi đó, các thông tin cần thiết được lưu trữ như các thuộc tính của các đỉnh, các cạnh. Các đỉnh có quan hệ trực tiếp với các đỉnh khác dựa trên cạnh nối. Có thể nói, đồ thị cung cấp cái nhìn trực quan về các mối liên hệ giữa các đối tượng với nhau. Cơ sở dữ liệu đồ thị cung cấp các phương thức truy vấn không chỉ truy cập vào dữ liệu được lưu trữ bên trong các đỉnh mà còn thể hiện được cấu trúc đồ thị của đỉnh đó [16].

Cùng với sự phát triển thịnh hành của NoSQL, cơ sở dữ liệu đồ thị là một cách tiếp cận tối ưu cho rất nhiều bài toán mạng xã hội hiện nay [17]. Hầu hết các mạng xã hội lớn hiện nay như Facebook, Yahoo, LinkedIn… đều sử dụng cở sở dữ liệu đồ thị để lưu trữ khối lượng dữ liệu khổng lồ của hệ thống [17]. Với cơ sở dữ liệu đồ thị, người quản trị hoàn toàn có thể cài đặt các thuật toán tìm duyệt tối ưu phù hợp để nâng cao hiệu suất sử dụng [18].

Neo4j là một cơ sở dữ liệu đồ thị được thiết kế trên ngôn ngữ Java và Scala có thể dùng để nhúng vào mã nguồn các ứng dụng cũng như liên kết client/server dựa trên REST API [19]. Với Neo4j, đỉnh và cạnh là các đối tượng được sử dụng như mô hình dữ liệu với khả năng thêm thuộc tính bất cứ thời điểm nào.



(<https://leanjavaengineering.files.wordpress.com/2015/04/graph.png>)

### Mô hình lưu trữ và tổ chức dữ liệu

* Đỉnh

Đỉnh là một đơn vị cơ bản trong đồ thị. Với Neo4j, đỉnh được dùng để đại diện cho các thực thể lưu trữ. Đỉnh chứa các thuộc tính là các trường lưu trữ của dữ liệu.

User

Person

Name = ‘Ngoc’

Age = 22

Nhãn

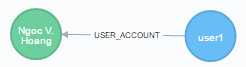
Nhãn

Thuộc tính

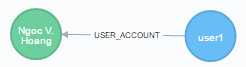
Đỉnh

Có thể phân biệt các đỉnh với nhau bằng các gắn nhãn cho đỉnh. Một đỉnh có thể có nhiều nhãn hoặc ko có nhãn.

* Cạnh

Cạnh là một đơn vị cơ bản trong đồ thị. Với Neo4j, cạnh được dùng để thể hiện mối quan hệ giữa các đỉnh với nhau. Cạnh cũng có thể có các thuộc tính, dựa vào đó có thể tìm kiếm dữ liệu có liên quan.

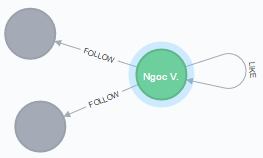
Cạnh luôn có hướng, được xác định theo hướng đi vào và hướng đi ra của một đỉnh. Một cạnh kết nối 2 đỉnh lại với nhau được đảm bảo hợp lệ từ đỉnh bắt đầu đến đỉnh kết thúc. Đây là một yếu tố quan trong được sử dụng khi sử dụng đồ thị.



Start node

End node

Mặc dù cần phải xác định rõ hướng giữa 2 đỉnh, nhưng cạnh là như nhau khi đi qua một trong hai hướng. Điều này có nghĩ là không cần phải có thêm các cạnh trùng lặp theo hướng ngược lại mà vẫn có thể duyệt đồ thị một cách bình thường. Khi duyệt đồ thị có thể bỏ qua hướng nếu không cần sử dụng đến.

Một đỉnh cũng có thể có cạnh nối với chính nó.

* Thuộc tính

Mỗi đỉnh và cạnh đều có thể có các thuộc tính. Thuộc tính là các cặp khóa- giá trị mà khóa chính là một chuỗi. Giá trị của thuộc tính có thể là một giá trị kiểu nguyên thủy (String, Interger, Boolean,…) hoặc mảng của các giá trị nguyên thủy đó.

Với Neo4j, không cho phép có thuộc tính null như Java, C#, C++,… Trong trường hợp cần gán giá trị null cho thuộc tính nào đó thì có thể xóa thuộc tính đó. Dĩ nhiên khi thuộc tính không tồn tại thì mặc định sẽ được gán giá tị null.

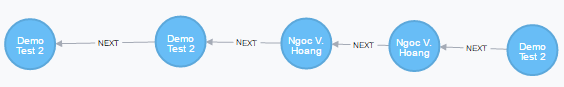
* Nhãn

Nhãn được sử dụng để gom nhóm các đỉnh, các cạnh lại với nhau. Tất cả các đỉnh, các cạnh có chung một nhãn sẽ được chung nhóm. Khi đó những câu truy vấn chỉ cần truy cập vào các nhóm này, thay vì toàn bộ đồ thị, điều đó giúp cho các câu truy vấn trở nên dễ hiểu và thực thi hiệu quả hơn.

Một đỉnh hoặc một cạnh có thể có được gắn nhiều nhãn. Tên của nhãn có thể là một chuỗi Unicode với độ dài tối đa là xấp xỉ 2 tỷ ký tự.

* Đường dẫn

Đường dẫn như là một đồ thị con của cơ sở dữ liệu đồ thị, bao gồm một hay nhiều đỉnh kết nối với nhau bởi các cạnh.



Đường dẫn ngắn nhất có thể là có độ dài bằng 0, trong trường hợp đó, đường dẫn chỉ bao gồm 1 đỉnh và không có cạnh nào nối với chính bản thân nó.

### Ngôn ngữ truy vấn Cypher

Cypher là ngôn ngữ truy vấn cho cơ sở dữ liệu đồ thị, cho phép tìm kiếm thông tin theo một mô hình nào đó. Cypher cung cấp cú pháp khai báo cụ thể và được xem là ngôn ngữ truy vấn trên đồ thị dễ tìm hiểu nhất và là một cơ sở tuyệt vời để tìm hiểu về đồ thị [19]. Truy vấn Cypher giúp mô phỏng lại được hoàn toàn về đồ thị, thể hiện một cái nhìn trực quan và chính xác trên chính cơ sở dữ liệu mà người dùng tạo ra.

Cũng giống như các ngôn ngữ khác, Cypher cung cấp các câu lệnh, từ khóa, cú pháp riêng để hình thành lên câu truy vấn.

* Cú pháp:

Cypher sử dụng các cặp ký hiệu đặc biệt tương phản nhau để thể hiện cho các thực thể. Cặp dấu ngoặc đơn () được sử dụng đại diện cho 1 đỉnh, cặp dấu ngoặc nhọn {} thể hiện cho các thuộc tính, cặp dấu -- thể hiện cho 1 cạnh với [] là chi tiết của cạnh đó.

(user: User{name = ‘Ngoc’})

-[action: Action{type = ‘post’}]->

(post: Post{title = ‘hello world’})

Đỉnh

Cạnh

Biến

Nhãn

Thuộc tính

Khi đó, đồ thị sẽ được duyệt theo thứ tự từ đỉnh User đến đỉnh Post thông qua kết nối cạnh Action. Các giá trị thuộc tính thêm vào dùng để lọc trực tiếp trên các thực thể tương ứng, các biến được sử dụng với mục đích ghi nhận giá trị hiện tại và có thể dùng cho phần sau của câu truy vấn khi cần.

Các cặp dấu mà rỗng là có nghĩ sẽ đại diện cho tất cả các thực thể tương ứng trong toàn đồ thị, do đó, khi thực thi 1 câu truy vấn cần phải sử dụng theo nhãn để tối ưu tốc độ thực thi.

(object)-[action: Action{type = ‘post’}]->

(post: Post{title = ‘hello world’})

Khi đó, đồ thị sẽ duyệt đến đỉnh Post thông qua kết nối cạnh Action với bất cứ đỉnh nào thỏa điều kiện của post và action. Biến object ở trên đại diện cho tất cả các đỉnh có trong đồ thị.

* Từ khóa:

Cypher sử dụng các từ khóa riêng để hình thành lên câu truy vấn của mình, trong truy vấn thì các từ khóa không phân biệt chữ hoa hay chữ thường.

START user = **node**:User(name = ‘Ngoc’)

MATCH (user)-[action: Action{type = ‘post’}]->

(post: Post{title = ‘hello world’})

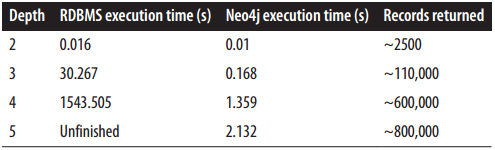
RETURN post.content

* + - START mô tả một hoặc nhiều điểm bắt đầu, có thể là tập các đỉnh hoặc tập các cạnh. Những thực thể này sẽ được chọn dựa trên các chỉ mục hoặc tìm kiếm dựa trên thuộc tính tương ứng.
    - MATCH thể hiện rõ luồng duyệt đồ thị từ đỉnh bắt đầu đến đỉnh kết thúc. Có thể thấy rõ, lệnh match đã tạo ra một mô hình đồ thị con. Cypher sẽ so khớp mô hình này với các đỉnh xung quanh từ đó tìm ra các đồ thị con thỏa yêu cầu [19].
    - RETURN quyết định các đỉnh, các cạnh, các thuộc tính nào được trả về cho câu truy vấn.
    - Ngoài các từ khóa quan trọng khác như WHERE dùng để lọc kết quả dựa theo các tiêu chí, CREATE để tạo đỉnh hoặc cạnh, SET dùng để thiết lập giá trị cho thuộc tính, và nhiều từ khóa khác như DELETE, WITH, MERGE,…

### Sức mạnh của Neo4j

* Hiệu suất

Đối với cơ sở dữ liệu quan hệ, khi tập dữ liệu ngày càng nhiều thì thời gian để thực hiện các truy vấn sâu và kết hợp nhiều bảng lại với nhau sẽ tốn rất nhiều thời gian và kém hiệu quả bởi các phép kết, rồi lọc. Nhưng với cơ sở dữ liệu đồ thị nói chung thì các truy vấn đã được nội bộ hóa thành một phần đồ thị con của toàn bộ đồ thị. Thời gian thực thi truy vấn tỉ lệ với kích thước của một phần đồ thị con liên quan, hoàn toàn không phụ thuộc vào toàn bộ kích thước lưu trữ của hệ thống. Nói cách khác, cho đù kích thước lưu trữ của hệ thống ngày càng tăng cao nhưng thời gian truy vấn dữ liệu vẫn luôn được đảm bảo [16].

Tùy theo tính chất của dữ liệu mà cơ sở dữ liệu đồ thị thể hiện được sự hiệu quả cao của mình. Khi cần xử lý dữ liệu qua nhiều lần truy vấn và mức độ duyệt sâu lên cao cũng như dữ liệu kết nối với nhau phức tạp thì cơ sở dữ liệu đồ thị là giải pháp tối ưu.

Sso sánh …. [19]

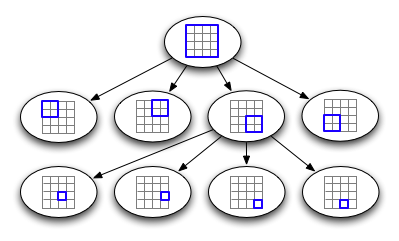
* Linh động

Bản thân cơ sở dữ liệu đồ thị là một dạng đặc biệt của NoSQL, do đó khi xây dựng không cần phải xác định trước cấu trúc hay mô hình của cở sở dữ liệu [18]. Việc thêm vào một đỉnh, một cạnh hay một thuộc tính bên trong hoàn toàn không ảnh hưởng đến các thực thể khác, cũng không làm thay đổi kết quả của các truy vấn đang thực hiện. Do đó, tính linh động của Neo4j rất cao, phù hợp với công việc có tính chất thay đổi, không ổn định và hoàn toàn có thể sửa đổi theo thời gian mà không cần phải xây dựng lại.

* Tối ưu

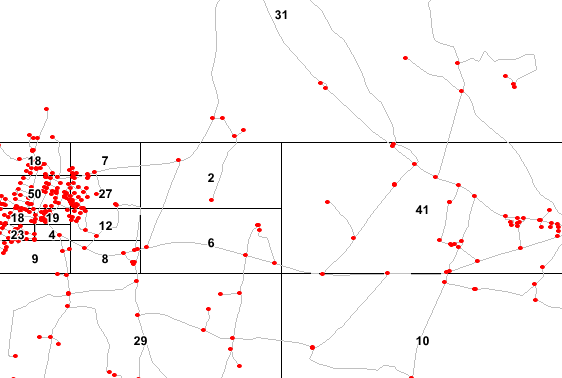
Neo4j cung cấp các API để duyệt đồ thị một cách tối ưu dựa trên cài đặt của ngôn ngữ Java. Các thuật toán tìm đường đi ngắn nhất, duyệt đồ thị sâu đều được tối ưu hoàn toàn. Bên cạnh đó các kỹ thuật đánh chỉ mục trên bảng cũng được cài đặt bên trong [19]. Không những thế, người sử dụng hoàn toàn có thể cài đặt lại các thuật toán này, vận dụng, thí nghiệm các xu hướng, thuật toán mới để đưa ra giải pháp phù hợp với nhu cầu hiện tại và phát triển ứng dụng một cách hoàn hảo nhất.

## Cấu trúc lưu trữ Quadtree

Giống như tên gọi, Quadtree là một trong những cấu trúc dữ liệu dạng cây. Được dùng để phân hoạch vùng không gian hai chiều hiện tại thành các vùng không gian nhỏ hơn và dễ quản lý hơn [20]. Khác với khác cấu trúc cây khác, mỗi lần phân hoạch, Quadtree sẽ chia vùng hiện tại thành 4 vùng. Do đó, mỗi node trong Quadtree sẽ có tối đa là 4 node con.

(http://static.notdot.net/uploads/quadtree.png)

Với cơ sở dữ liệu gắn liền với tọa độ địa lý, thì việc lưu trữ theo cấu trúc Quadtree sẽ tối ưu hoàn toàn câu truy vấn [21]. Có thể thấy rõ, các lệnh truy vấn sẽ tập trung vào một số vùng nhỏ trên toàn bộ bản đồ, không phải tốn nhiều thời gian chi phí duyệt qua các dữ liệu không cần thiết. Do đó, với bài toán hiện tại, việc tổ chức cơ sở dữ liệu theo cấu trúc Quadtree sẽ giúp cải thiện tốc độ tìm kiếm một địa điểm trên bản đồ.

Tuy nhiên, khi tìm kiếm trên bản đồ với phạm vi nhỏ, thì việc tổ chức theo Quadtree có phần không hiệu quả. Bởi mỗi lần phân hoạch chỉ được 4 node con, trong khi kinh độ, vĩ độ địa lý là có giới hạn và các khoảng cách trong phạm vi nhỏ chỉ cần tính bằng đơn vị phút là đủ. Do đó, giải pháp được đề ra là sử dụng một biến thể của Quadtree theo cấu trúc lưới để tổ chức lại các chỉ mục không gian lưu trữ. Cụ thể, với giá trị vĩ độ nằm trong khoảng từ -90 đến 90 và giá trị kinh độ từ -189 đến 190, Quadtree sẽ được tổ chức như một ma trận có kích thước 180x360 với mỗi điểm trên ma trận đó sẽ đại diện cho một điểm tương ứng nằm trên bản đồ địa lý. Trong mỗi điểm con sẽ được tổ chức thành một ma trận với kích thước 100x100 với mỗi điểm trên ma trận con này sẽ đại diện cho đơn vị phút trên tọa độ địa lý.

(http://www.routeware.dk/wordpress/wp/content/uploads/2013/03/spatial\_index.png)

Khi đó, kết hợp với chỉ mục, chi phí tối đa có thể để truy cập vào một vùng lưu trữ theo đơn vị phút của tọa độ địa lý là:

+ + + = 31

* + - chi phí tìm kiếm vĩ độ đơn vị độ.
    - chi phí tìm kiếm kinh độ đơn vị độ.
    - chi phí tìm kiếm kinh độ và vĩ độ theo đơn vị phút trên ma trận nhỏ.

Do đó, khi cần duyệt tìm một thực thể không gian theo một khoảng cách nhỏ, thì chỉ cần truy vấn nào một vùng nhỏ tương ứng trên Quadtree và sẽ không tốn nhiều chi phí khi phải duyệt hết toàn bộ dữ liệu, tăng tốc sử dụng cho hệ thống.

## Tìm kiếm toàn văn Lucene

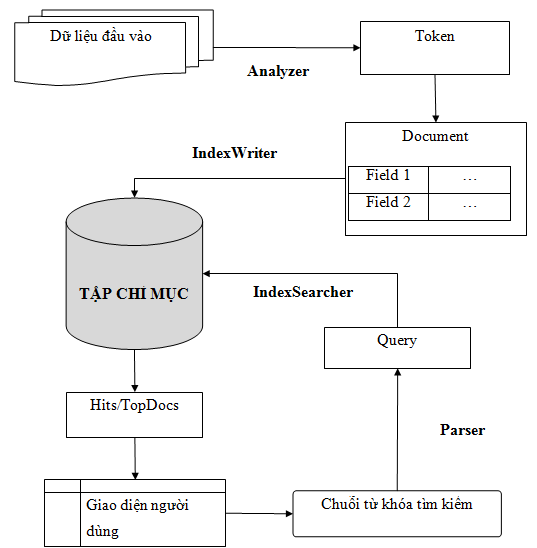
Trong một hệ thống mạng xã hội, chức năng tìm kiếm đóng vai trò cực kỳ quan trọng, quyết định sự thành công hay thất bại của hệ thống đó. Hệ thống ứng dụng phải đáp ứng được nhu cầu tìm kiếm bạn bè, tìm kiếm thông tin một cách chính xác, nhanh chóng và tiện lợi.

Lucene là một thư viện mã nguồn mở dùng để phân tích, đánh chỉ mục và tìm kiếm thông tin với hiệu suất cao dựa trên ngôn ngữ Java [22]. Lucene có thể được tùy chỉnh cách phân tích dữ liệu, xác định độ ưu tiên, cách đánh chỉ mục sao cho phù hợp với yêu cầu của người sử dụng. Bên cạnh đó, Lucene có thể thực hiện đánh chỉ mục và tìm kiếm trên nhiều định dạng dữ liệu khác nhau như một trường dữ liệu trong một cơ sở dữ liệu, tập tin HTML, PDF, Microsoft Word,… Lucene hỗ trợ nhiều tiêu chí tìm kiếm phổ biến như các kiểu Boolean (AND, OR, NOT), tìm kiếm mờ, tìm kiếm xấp xỉ, tìm kiếm ký tự đại diện hay tìm kiếm dữ liệu trong một khoảng nhất định [22]…

(https://lucene.apache.org/core/)

Với Neo4j, thì đây là một thư viện mạnh mẽ cho phép tìm kiếm toàn văn với độ chính xác cao, khả năng tùy biến lớn và tốc độ nhanh. Vì Neo4 và Lucene đều được phát triển bởi ngôn ngữ Java, nên việc sử dụng Lucene trên Neo4j là rất dễ dàng. Các phiên bản của Neo4j đều được tích hợp sẳn cơ chế tìm kiếm dựa trên Lucene, người sử dụng chỉ cần tùy chỉnh cơ chế và xác định các trường dữ liệu liên quan khi sử dụng.

Cơ chế thực thi của Lucene bao gồm hai thao tác. Một là phân tích dữ liệu, xác định độ ưu tiên và đánh chỉ mục cho trường dữ liệu cần tìm kiếm mà người dùng đã tùy chỉnh. Hai là phân tích câu truy vấn mà người dùng yêu cầu, sau đó tạo ra các câu truy vấn mới và thực hiện quá trình tìm kiếm dựa trên tập chỉ mục đã lưu.

Quá trình Lucene thực thi có thể được biểu diễn như sau:

(<https://lucene.apache.org/core/documentation.html>)

* *Directory:* nơi lưu trữ tập chỉ mục.
* *Document:* các trường dữ liệu và các tài liệu dùng để lưu trữ dữ liệu chỉ mục, phục vụ cho việc tìm kiếm.
* *Analyzer:* cơ chế phân tích dữ liệu đầu vào, chuẩn hóa dữ liệu chuẩn bị cho việc tạo chỉ mục.
* *IndexWriter:* cơ chế lưu trữ, tạo mới hoặc cập nhật nội dung tập chỉ mục.
* *Query:* chuỗi văn bản chứa câu lệnh truy vấn, được tạo thành sau khi parser phân tích chuỗi văn bản người dùng nhập vào.
* *IndexSearcher:* thực hiện việc tìm kiếm trên tập chỉ mục.
* *Hit/TopDocs:* chứa các tài liệu phù hợp với kết quả truy vấn.

Với các chức năng được tối ưu, khả năng tương thích, dễ sử dụng, Lucene luôn luôn là sự lựa chọn hoàn hảo cho quá trình tìm kiếm toàn văn thay vì sử dụng cơ chế của hệ quản trị cơ sở dữ liệu. Và Neo4j Lucene một giải pháp toàn hoàn phù hợp với yêu cầu tìm kiếm của bài toán.

## Hệ thống định vị toàn cầu GPS

### Tổng quan

GPS là hệ thống xác định vị trí dựa trên các vệ tinh nhân tạo do bộ quốc phòng Hoa Kỳ thiết kế, xây dựng, vận hành và quản lý [23]. Ban đầu GPS chỉ dùng cho mục đích quân sự, nhưng từ năm 1980 GPS được phép sử dựng cho dân sự và trở thành một công cụ hỗ trợ đắc lực cho nhiều lĩnh vực trong cuộc sống.

Ngày nay, các thiết bị di động hầu như đều được tích hợp sẳn GPS, nhu cầu khai thác và sử dụng GPS của người dùng càng trở nên phổ biến và thiết yếu. Bên cạnh đó, công nghệ bản đồ trực truyến ra đời và phát triển mạnh đã giúp GPS thể hiện được các đặc điểm tiềm ẩn [24]. Việc tìm kiếm đường đi, xác định vị trí của người sử dụng trở nên dễ dàng hơn, trực quan hơn. Các hệ thống bán hàng, maketing, theo dõi hành trình,… sử dụng tọa độ GPS đang dần phát triển và tham vọng thay thế các phương thức truyền thống.

### Sử dụng GPS trên Android

Với các thiết bị di động hệ điều hành Android, khả năng sử dụng GPS đã được cài đặt sẳn. Tuy độ chính xác không phải tuyệt đối, nhưng cũng đáp ứng được phần nào các nhu cầu hằng ngày của người sử dụng.

Về mặt lập trình, Android có hai cơ chế cung cấp tọa độ của thiết bị là *GPS Location Provider* và *Network Location Provider*. Chỉ cần sử dụng một trong hai cơ chế này là có thể lấy được tọa độ của thiết bị. Tuy nhiên, nên sử dụng cả hai vì chúng có một số đặc điểm như sau:

* + - *GPS Location Provider* có thể tốn nhiều thời gian khi thiết bị đang ở trong bên trong nhà kín và sẽ gây tốn nhiều năng lượng.
    - *GPS Location Provider* cung cấp tọa độ với sai số so với thực tế rất thấp.
    - *Network Location Provider* khi thiết bị không thể kết nối internet hoặc đường truyền bị yếu.
    - *Network Location Provider* cung cấp tọa độ dựa vào các tòa nhà lớn, hệ thống wifi gần đó nên sai số có thể sẽ sai khác nhiều so với thực tế [25].

Để sử dụng được hai cơ chế trên thì ứng dụng android phải được quyền truy cập vào hệ thống GPS của thiết bị.

<uses-permission android:name="android.permission.ACCESS\_FINE\_LOCATION" />

<uses-permission android:name="android.permission.ACCESS\_COARSE\_LOCATION" />

<uses-permission android:name="android.permission.INTERNET" />

* + - ACCESS\_FINE\_LOCATION cho phép sử dụng cả hai cơ chế cung cấp.
    - ACCESS\_COARSE\_LOCATION cho phép sử dụng *Network Location Provider* để cung cấp tọa độ cho ứng dụng.
    - INTERNET là yêu cầu bắt buộc khi muốn sử dụng *Network Location Provider*.

Android cung cấp sẳn các đối tượng trong trong thư mục *android.location* với nhiều phương thức giúp dễ dàng xác định vị trí, cũng như quản lý các kết nối của thiết bị khi sử dụng. Bao gồm nhiều đối tượng như:

|  |  |
| --- | --- |
| *Address* | Đối tượng thể hiện địa chỉ mô tả theo vị trí |
| *Criteria* | Đối tượng giúp ứng dụng xác định được cơ chế phù hợp |
| *Geocoder* | Đồi tượng ghi nhận các thông tin địa lý |
| *GpsSatellite* | Đối tượng thể hiện trạng thái của các vệ tinh GPS |
| *GpsStatus* | Đối tượng quản lý trạng thái của GPS trên thiết bị |
| *Location* | Đối tượng thể hiện tọa độ địa lý |
| *LocationManager* | Đối tượng quản lý truy cập vào hệ thống tọa độ của thiết bị |
| *LocationProvider* | Đối tượng thể hiện cơ chế truy cập GPS |

Khi đó để truy cập vào hệ thống GPS của thiết bị, cần phải sử dụng một đối tượng *LocationManager* để quản lý các kết nối thông qua phương thức đã được định nghĩa sẳn *getSystemService()*.

locationManager = (LocationManager)getSystemService(Context.LOCATION\_SERVICE);

Tiếp theo đó là xác định cơ chế cung cấp tọa độ theo các cơ chế đã trình bày bên trên. Để thực hiện điều này, lớp *LocationManager* cung cấp một phương thức để khởi tạo yều cầu truy cập *requestLocationUpdates()*. Phương thức này yêu cầu cung cấp cơ chế hoạt động, thời gian, khoảng cách và một đối tượng *LocationListener* để ghi nhận khi tọa độ thay đổi.

locationManager.requestLocationUpdates(LocationManager.GPS\_PROVIDER, TIME, DISTANCE, locationListener);

Ngoài các cơ chế có sẳn của thiết bị Android, để biết được thông tin tọa độ của thiết bị, còn có thể sử dụng các API của bên thứ ba như Google Play Services. Tuy nhiên, thiết bị phải cho phép truy cập hệ thống hoặc kết nối internet thì mới có thể xác định được vị trí hiện tại. Tính riêng tư cá nhân của người sử dụng luôn được đảm bảo.

## Google services API

Android là một hệ điều hành dựa trên nền tảng Linux được thiết kế dành cho các thiết bị di động, được Google phát triển và cập nhật hằng ngày [26]. Các hệ thống hỗ trợ của Google đã được cài đặt sẳn và trở thành một phần không thể thiếu trên mỗi thiết bị Android. Do đó, việc tích hợp hệ thống hỗ trợ này vào ứng dụng sẽ không tốn quá nhiều chi phí, cũng như không đòi hỏi thiết bị phải cài đặt thêm nhiều chức năng khác. Ứng dụng sẽ hiệu quả hơn, tối ưu hơn và dễ sử dụng hơn cho người dùng.

### Google Maps API

Google Maps là một ứng dụng dịch vụ và công nghệ bản đồ trực tuyến trên web miễn phí được cung cấp bởi tập đoàn Google, hỗ trợ nhiều dịch vụ khác của Google đặc biệt là tìm đường và dẫn đường. Với tính năng mở và hoàn toàn miễn phí, người dùng có thể cập nhật thêm thông tin về địa điểm, đường đi,… giúp cho kho dữ liệu về địa điểm ngày càng đa dạng, chi tiết và sát với thực tế. Không dừng lại ở đó, để tăng tính tiện dụng và tương tác dễ dàng hơn cho người dùng, Google đã đưa ra dịch vụ hỗ trợ đắc lực đó chính là Google Maps API. Đây là một dịch vụ miễn phí của Google cho phép người dùng nhúng Google Maps và các ứng dụng tiện vào trong ứng dụng của mình.

Google Maps API hiện tại đã được phát triển đến phiên bản thứ 3, không chỉ cho phép sử dụng trên nền tảng web mà còn giúp dễ dàng nhúng vào trong các ứng dụng di động. Khi đó, lập trình viên dễ dàng tùy biến, thay đổi, tạo hiệu ứng và xây dựng các chức năng đặc trưng riêng cho ứng dụng của mình. Google Maps API cho phép nhiều chức năng cơ bản thao tác trên bản đồ như đánh dấu các địa điểm kèm theo thông tin, chỉ dẫn đường giao thông công cộng, khoanh vùng khu vực,… [25]

Để nhúng Google Maps vào trong ứng dụng Android, trước tiên là phải cài đặt thư viện “*google-play-services*” và sử dụng cho ứng dụng. Sau đó, tiến hành tạo một khóa quản lý bởi Google Maps API để kết nối ứng dụng với các hệ thống chức năng của Google Maps. Cụ thể gồm các bước như sau:

1. Cài đặt và cấu hình thư viện “*google-play-services*” từ android SDK manager.
2. Thêm thư viện “*google-play-services*” và biên dịch trong ứng dụng.
3. Yêu cầu chứng chỉ từ Google để xác nhận đăng ký sử dụng Google Maps trên ứng dụng.
4. Tạo một ứng dụng trên trang quản lý Google APIs.
5. Tạo một khóa sử dụng cho Google Maps API.
6. Sử dụng khóa đó để nhúng Goolge Maps vào ứng dụng.

Khóa được tạo ra sẽ được mã hóa với thuật toán SHA-1 và lưu trên máy dưới dạng vùng nhớ đệm để sử dụng cho quá trình biên dịch cũng như triển khai ứng dụng. Khóa này sẽ được gắn riêng cho mỗi ứng dụng, tức là chỉ sử dụng được riêng cho ứng dụng có đường dẫn quản lý đó.

Khi đó, để khởi tạo giao diện cho Google Maps trên ứng dụng android, có thể sử dụng phần tử *<fragment>* trên tập tin đặc tả giao diện của một màn hình trong ứng dụng và sử dụng nó như một đối tượng *MapFragment* bằng các ép kiểu thông qua thuộc tính *name*.

<fragment xmlns:android="http://schemas.android.com/apk/res/android"  
    android:name="com.google.android.gms.maps.MapFragment"  
    android:id="@+id/map"  
    android:layout\_width="match\_parent"  
    android:layout\_height="match\_parent"/>

Bên cạnh đó, cũng có thể sử dụng trực tiếp đối tượng *MapFragment* thông qua phương thức tĩnh *newInstance()*.

mMapFragment = MapFragment.newInstance();

FragmentTransaction fragmentTransaction =

getFragmentManager().beginTransaction();

fragmentTransaction.add(R.id.my\_container, mMapFragment);

fragmentTransaction.commit();

Khi đó, người sử dụng hoàn làm chủ được Google Maps. Có nghĩa là được phép tùy chỉnh, xây dựng thêm các chức năng, thể hiện giao diện phù hợp,… trên Google Maps cho ứng dụng của mình.

### Google Places API

Google Places API là một tiện ích được phát triển song song với Google Maps, định nghĩa các thông tin của địa điểm trên bản đồ bao gồm vị trí địa lý, tên gọi vùng miền và chức năng mô tả nổi bật. API này được sử dụng thông qua giao thức kết nối HTTPS [25]. Bên cạnh đó, Google Places API còn cho phép lập trình viên truy cập vào cở sở dữ liệu của một vị trí được lưu trên máy chủ của Google, bao gồm thông tin chi tiết về địa điểm cũng như các hạng mục trong cơ sở dữ liệu.

Với bất kỳ hệ thống ứng dụng có sử dụng bản đồ, thì việc tìm kiếm thông tin dựa trên địa điểm địa lý là rất cần thiết. Người dùng chỉ cần nhớ tên địa điểm đó, hoặc một số thông tin liên quan là có thể tìm được thông tin cụ thể, cũng như giúp ứng dụng trở nên tiện ích hơn, công cụ tìm kiếm trở nên đa dạng phù hợp hơn với yêu cầu của người dùng.

Google Places API cung cấp nhiều chức năng tiện ích hỗ trợ cho người dùng và lập trình viên như:

* + - *Place Searchs*: trả về danh sách các địa điểm dựa trên tọa độ địa lý hoặc chuỗi có thông tin tóm tắt về địa điểm đó.
    - *Place Details*: trả về chi tiết của một địa điểm, bao gồm cả đánh giá của người dùng.
    - *Place Action*: cho phép thêm chi tiết của một địa điểm vào trong cơ sở dữ liệu của Google Place.
    - *Place Photo*: cho phép truy cập vào các địa điểm có liên quan đến hình ảnh.
    - *Place Autocomplete*: chức năng tự động hoàn tất tên hoặc địa điểm khi người dùng tìm kiếm.

Cũng như Google Maps API, để sử dụng Google Places API cũng cần phải sử dụng một khóa để xác nhận truy cập trên dữ liệu của Google. Có thể sử dụng chung khóa với Google Maps API và lưu ý bật tính năng sử dụng Google Places API cho toàn bộ kết nối.

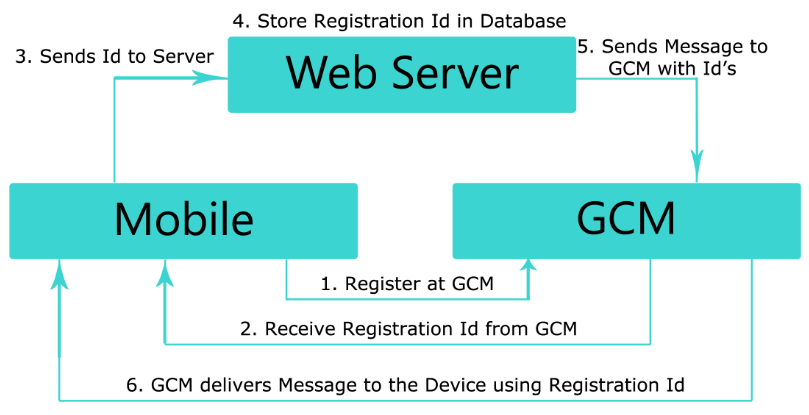
Để sử dụng được Google Places API trên ứng dụng Android, lập trình viên không phải thực hiện quá nhiều thao tác, cũng như quá trình xử lý không quá phức tạp. Ví dụ như để sử *PlaceAutocomplete* có thể sử dụng phần tử *<fragment>* như Google Maps API với thuộc tính *name* là đối tượng *PlaceAutocompleteFragment*.

<fragment  
  android:id="@+id/place\_autocomplete\_fragment"  
  android:layout\_width="match\_parent"  
  android:layout\_height="wrap\_content"  
  android:name=

"com.google.android.gms.location.places.ui.PlaceAutocompleteFragment"  
  />

Khi đó, quá trình xử lý chỉ đơn giản là nhận kết quả trả về khi người dùng chọn vào một địa điểm thông qua lớp *PlaceSelectionListener*. Lớp này sẽ được cài đặt thay thế cơ chế mặc định của *PlaceAutocompleteFragment* và lập trình viên sẽ có được thông tin mong muốn, cũng như xử lý được thông tin của người sử dụng khi chức năng hoạt động.

### Google Cloud Messaging

Google Cloud Messaging (GCM) là một dịch vụ giúp gửi thông điệp đến các thiết bị người dùng. Bên cạnh đó, GCM còn cho phép gửi thông điệp từ thiết bị người dùng đến hệ thống của Google và hỗ trợ việc truyền dữ liệu từ server đến cả ứng dụng Android và iOS [27] [28]. Với thiết bị Android thì giải pháp gửi thông điệp mặc định chính là GCM [27].

(sending message through gcm [28])

Với mạng xã hội, quá trình truyền gửi thông điệp và chia sẻ thông tin phải được thực hiện một cách tối ưu. Bởi đây luôn là vấn đề không thể thiếu và quyết định sử thành công hay thất bại của mạng xã hội. Người sử dụng luôn có nhu cầu cập nhật các lượt bình luận, các tin nhắn một cách cụ thể, nhanh chóng và chính xác. Do đó, hệ thống truyền gửi thông điệp cần phải vững chắc, ổn định và có khả năng xử lý lượng thông tin cho nhiều thiết bị một cách đồng thời. GCM là một giải pháp tốt bởi vì có khả năng truyền thông điệp đến hàng nghìn thiết bị trong một lần [27] và tương thích sẳn với hệ điều hành Android.

Để sử dụng dịch vụ này, mỗi thiết bị cần phải đăng ký với GCM server và những thiết bị muốn nhận được thông điệp cũng phải được đăng ký với GCM server. Với thiết Android, cần thêm sự trợ giúp của Google Play Services.

Để sử dụng được GCM trên thiết bị Android, lập trình viên cần thực hiện một số thao tác như sau:

* Đăng ký sử dụng GCM

Trước tiên phải đăng ký sử dụng GCM trên máy chủ Google giống như Google Maps API và Google Places API. Khi đăng ký thành công, máy chủ GCM sẽ cung cấp một khóa cho người sử dụng, khóa này quản lý cơ chế xác nhận gửi thông điệp từ ứng dụng. Lưu ý là phải bật dịch vụ GCM cho thiết bị Android.

* Xây dựng GCM cho ứng dụng

GCM cho ứng dụng là cơ chế giúp nhận các thông điệp từ server và gửi các thông điệp khác lên máy chủ của GCM. Hệ điều hành Android cung cấp cơ chế để xây dựng GCM với những thiết bị phiên bản Android 2.2 trở lên và phải cài đặt ứng dụng quản lý dịch vụ *google play store*. Khi đó, để có thể sử dụng được dịch vụ GCM, ứng dụng cụ thể phải được cấp phép những yêu cầu sau đây:

* + - *com.google.android.c2dm.permission.RECEIVE*: giúp thiết bị có thể đăng ký và nhận thông điệp từ hệ thống server của GCM.
    - *com.google.android.c2dm.permission.SEND*: giúp GCM có thể gửi thông điệp đến ứng dụng.
    - *android.permission.INTERNET*: cho phép sử dụng internet để để gửi những mã đăng ký thiết bị đến server.
    - *android.permission.WAKE\_LOCK*: giúp ứng dụng duy trì hoạt động, không bị treo khi có thông điệp gửi đến.

Để giúp quá trình gửi nhận thông điệp một cách hiệu quả và không ảnh hưởng đến luồng thực thi của ứng dụng, GCM phải được cài đặt dưới dạng một dịch vụ chạy ngầm. Android có hai cách để tạo một dịch vụ chạy song song với ứng dụng là *Service* và *IntentService*, cùng hai cơ chế truyền dữ liệu từ dịch vụ đến luồng thực thi chính của ứng dụng *BroadcastReceiver* và *WakefulBroadcastReceiver*.

* Quản lý thông tin GCM trên web services

Khi GCM cung cấp các định danh cũng như các tài khoản liên quan cho thiết bị, hệ thống cần lưu lại các thông tin này để sử dụng. Khi đó, muốn gửi một thông điệp đến thiết bị, hệ thống chỉ cần gửi đến máy chủ GCM thông tin định danh đã lưu và thông điệp tương ứng. Trong trường hợp thiết bị nhận đã tắt hoặc không thể kết nối internet, thì các thông điệp sẽ được tổ chức theo hàng đợi trên máy chủ GCM và sẽ được gửi đi khi thiết bị hoạt động trở lại.

Bên cạnh những thuận lợi đó, GCM cũng tồn tại một số hạn chế như quá trình truyền gửi thông điệp là không thể dự đoán được, có nghĩa là thời gian để thực hiện có thể bị trì hoãn, ứng dụng có thể sẽ không cập nhật thông tin một cách nhanh chóng.

Hệ thống GCM có thể được xây dựng với nhiều công nghệ phổ biến như Java, php, Asp.net, Python,… và cần sử dụng khóa dịch vụ để cung cấp cơ chế xác thực quản lý trên máy chủ GCM. Trong nội dung đề tài này, hệ thống GCM được xây dựng sử dụng công nghệ Node.js và sẽ được bình luận chi tiết ở phần sau.

### Google Play Services

Google Play Services là tập hợp các tính năng của Google để xây dựng các ứng dụng có thể kết nối với các sản phẩm của Google như Maps, Drive, Gmail,... Trên thiết bị Android, Google Play Services giúp quá trình trải nghiệm ứng dụng được cải thiện, tăng tốc khả năng tìm kiếm ngoại tuyến. Khi đó, các ứng dụng của Google sẽ được tận dụng, cập nhật và dễ dàng tích hợp những tính năng mới nhất.

Google Play Services giúp người dùng dễ dàng truy cập vào các dịch vụ của Google và được tích hợp chặt chẽ với hệ điều hành Android, cho phép người dùng triển khai các chức năng mong muốn một cách nhanh chóng, không phải lo lắng về vấn đề thiết bị có hỗ trợ hay không.

Thư viện ứng dụng của Google Play Services được cung cấp thông qua Android SDK Manager, và luôn được tự động cập nhật. Điều này giúp người dùng dễ dàng tập trung vào các chức năng quan trọng hơn. Để sử dụng được các dịch vụ hỗ trợ được giới thiệu bên trên, lập trình viên phải sử dụng thư viện này như một yếu tố bắt buộc cho ứng dụng của mình.

## Xây dựng máy chủ với công nghệ NodeJS

### Tổng Quan

NodeJS là một nền tảng mã nguồn mở được viết bởi Ryan Dahl vào năm 2009 dựa trên Javascript engine V8 của Google [29]. NodeJS có thể chạy trên nền tảng nhiều hệ điều hành khác nhau và cung cấp các thư viện dưới dạng Javascript giúp đơn giản hóa việc lập trình và tối ưu thời gian thực thi ở mức thấp nhất. Trước khi NodeJS ra đời, Javascript chủ yếu được dùng để nhúng vào mã nguồn HTML sử dụng trên các website cho phép người dùng tương tác với trình duyệt. Với NodeJS, lập trình viên có thể sử dụng Javascript để phát triển các ứng dụng trên máy chủ hoặc các ứng dụng về mạng máy tính [30].

Node.js là một phần mềm mã nguồn mở được viết dựa trên ngôn ngữ Javascript cho phép lập trình viên có thể xây dựng các ứng dụng chạy trên máy chủ. Node.js chứa một thư viện cho phép các ứng dụng hoạt động như một máy chủ mà không cần phải sử dụng thêm các phần mềm khác như IIS, Apache HTTP Server.

### Đặc tính của NodeJS

Mặc dù có rất nhiều công nghệ để xây dựng máy chủ lâu đời như PHP, JavaEE, .NET hay mới xuất hiện thời gian gần đây như Python, Ruby,… NodeJS vẫn khẳng định được vị trí của mình bởi đặc tính thời gian thực và cơ chế không đồng bộ [30]. Có nghĩa là quá trình truyền xử lý các yêu cầu một cách đồng thời và thời gian phản hồi nhanh chóng, tất cả các dịch vụ được xây dựng chỉ chờ kết quả trả về từ máy chủ và không bị ảnh hưởng bởi các luồng nhập xuất khác. Bên cạnh đó, NodeJs nhận và xử lý nhiều kết nối chỉ với một luồng duy nhất với sự kiện lặp giúp hệ thống tốn rất ít RAM, tận dụng tối đa tài nguyên và không tạo ra độ trễ [29]. Ngoài ra, NodeJS còn có một số đặc tính nổi bật như :

* + - Khi cần xử lý một luồng dữ liệu cực lớn, NodeJS sẽ dựng các Proxy phân vùng các luồng dữ liệu để đảm bảo hoạt động tối đa cho các luồng dữ liệu khác.
    - Xử lý dịch vụ RESTful/JSON với cơ chế event-driven [30].

### Sử dụng NodeJS.

Với công nghệ NodeJS, lập trình viên có thể xây dựng máy chủ websocket, hệ thông truyền gửi thông điệp, mảy chủ quảng cáo, ứng dụng truyền tải tập tin từ thiết bị người dùng [29],… Vì bản thân NodeJS được phát triển từ Javascript nên cú pháp lập trình cũng tương đồng nhau.

Để cài đặt một thư viện hỗ trợ trên Node.js chỉ cần sử dụng lệnh *npm install* cùng với tên thư viện đó. Ví dụ cài đặt thư viện *express*.

npm install express

Khi đó thư viện sẽ được tải về trên ứng dụng dưới dạng một thư mục và để sử dụng chỉ cần dùng lệnh *require*.

var express = require('express');

var app = express();

app.get('/', function (req, res) {

res.send('Hello World');

})

var server = app.listen(8081, function () {

var host = server.address().address

var port = server.address().port

console.log("Example app listening at http://%s:%s", host, port)

})

Sử dụng NodeJS kết hợp với NoSQL là một sự lựa chọn hoàn hảo, bởi quá trình xử lý JSON của 2 công nghệ này là rất tốt. Dữ liệu trả về từ cơ sở dữ liệu là JSON và quá trình xử lý truyền dữ liệu từ máy chủ NodeJS cũng là JSON nên không tốn thêm nhiều chi phí.

Trong hệ thống ứng dụng của mình, chúng em sử dụng công nghệ NodeJS để xây dựng một máy chủ quản lý các dịch vụ ứng dụng, giúp tăng tốc xử lý, đảm bảo thời gian thực thi cho người dùng.

## Phong cách thiết kế Material Design

Material Design là một phong cách thiết kế mới được Google giới thiệu cùng lúc với phiên bản hệ điều hành Android 5.0 lollipop [31], đánh dấu sự khởi động của một cuộc cách mạng về mặt giao diện trên các thiết bị Android. Material Design ra đời với nhiệm vụ cực kỳ quan trọng đó là hợp nhất thiết kế giao diện ứng dụng xuyên suốt nhiều loại thiết bị khác nhau – điện thoại, máy tính bảng, máy tính, TV, thiết bị đeo được và tham vọng cho tất cả mọi thứ.

Material Design gây ấn tượng với người dùng bằng cách nhấn mạnh vào các mảng màu đậm táo bạo, những khối hình phẳng, trôi nổi và kết hợp với hiệu ứng chuyển động tự nhiên [31]. Phong cách thiết kế Material Design nhắm đến sự đơn giản, thân thiện, tạo ấn tượng với người dùng khi bắt đầu trải nghiệm ứng dụng. Material Design được coi như bộ chuẩn chung về thiết kế giao diện ứng dụng từ các bên thứ ba, lẫn các đối tác tham gia vào hệ sinh thái Google [31].

Một số đặc điểm có thể nhận biết trên những ứng dụng Material Desgin như sau:

* Sử dụng màu nổi bật, luôn có một màu chủ đạo ở cạnh trên ứng dụng.
* Các biểu tượng phẳng, đơn giản, dễ hiểu.
* Những ứng dụng sẽ có những nút to tròn nằm ở góc dưới bên phải dùng để sử dụng chức năng tạo mới.
* Giao diện phẳng, không hiệu ứng chuyển màu, đổ bóng đen.
* Các hiệu ứng chuyển động tự nhiên, dễ hiểu, có thể gợi ý cho một tính năng nào đó.
* Các công cụ, nút nhấn, chữ viết,… có nhiều khoảng trắng thân thiện.

Với nhiều ưu điểm vượt trội, thân thiện hơn với nhà phát triển, giao diện đẹp mắt, xuyên suốt, tạo thiện cảm với người sử dụng cùng với sự hỗ trợ mạnh mẽ từ phía nhà phát triển Google, Material Design trong thiết kế sản phẩm hứa hẹn sẽ xuất hiện nhiều hơn trong tương lai [31].

Với những thiết bị Android, thì Material Design giống như một hơi thở mới cho thiết kế trên nền tảng hệ điều hành này. Nhờ vậy, chúng ta sẽ cảm thấy như đang sử dụng một chiếc điện thoại mới, tránh đi sự nhàm chán mà chúng ta đã nhìn thấy trong nhiều năm qua.

Hiện tại, đã có nhiều ứng dụng sử dụng phong cách này trong thiết kế giao diện. Phải nhắc đến đầu tiên đó chính là các ứng dụng của Google như Inbox, Youtube, kho ứng dụng Google Play với giao diện nhẹ nhàng, hiệu ứng tự nhiên, sắp xếp thông minh, và nhiều thành phần đồ họa đơn giản khác. Bên cạnh đó, các ứng dụng bên thứ ba như Facebook, Evernote, Wave Calender,… cũng đang dần chuyển sang phong cách thiết kế này. Tại Việt Nam, cũng có nhiều ứng dụng như Laban Key, ePlay cũng sử dụng Material Desgin xuyên suốt giao diện người dùng từ điện thoại, máy tính bảng, website.

Nắm được các đặc điểm ưu Việt đó, nhóm chúng em sẽ sử dụng phong cách thiết kế này cho ứng dụng của mình, tạo cảm giác thân thiện, tự nhiên, dễ sử dụng cho tất cả mọi người.

## Kết luận

Trong nội dung khóa luận, chúng em sẽ xây dựng một ứng dụng trên nền tảng hệ điều hành Android, thiết kế bằng phong cách Material Design. Ứng dụng sẽ được tích hợp các dịch vụ của Google như Google Maps API, Places API. Bên cạnh đó, hệ thống xác định tọa độ địa lý và chạy ngầm hợp lý, không tốn nhiều chi phí duy trì hoạt động của thiết bị.

Hệ thống máy chủ được xây dựng trên công nghệ NodeJS và cơ sở dữ liệu đồ thị Neo4j. Ngoài ra, cấu trúc lưu trữ Quadtree được xây dựng để tổ chức dữ liệu, quản lý và tối ưu truy vấn tìm kiếm với Lucene. Hệ thống truyền gửi thông điệp được xây dựng dựa trên Google Cloud Messaging. Hy vọng, sẽ tạo được trải nghiệm tốt cho mọi người.

# Ứng dụng HOMFY

* Chương 4 trình bày báo cáo chi tiết kỹ thuật về ứng dụng. Bao gồm bản vẽ kỹ thuật, các mô hình, lược đồ, quá trình phân tích thiết kế kiến trúc và hướng dẫn sử dụng ứng dụng.

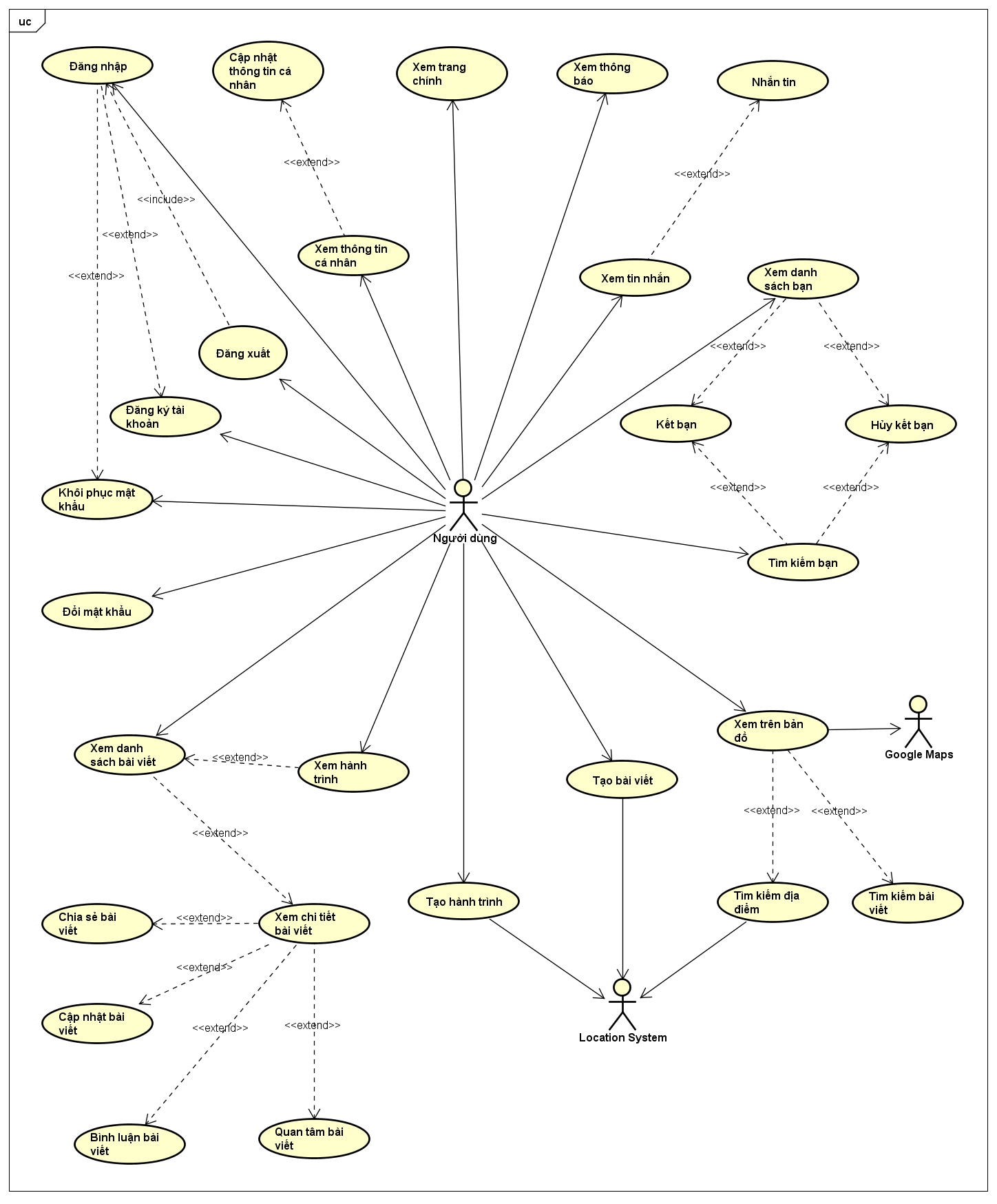
## Giới thiệu tổng quan

Ngày nay, dịch vụ du lịch ngày càng phát triển mạnh mẽ với nhiều hình dạng, biến thể và chiếm lĩnh thị phần lớn trong nền kinh tế. Mọi người có nhu cầu tự tìm hiểu các địa điểm nổi tiếng, những vùng đất lạ mà mỗi cá nhân đó chưa từng đặt chân đến. Cụ thể là hình thức du lịch “phượt”, một hình thức du lịch đang rất được yêu thích, phổ biến hiện nay. Nhưng người du lịch thường gặp khó khăn trong việc tìm kiếm địa điểm, không biết nên đi đâu hay cần sự giới thiệu, lời khuyên của người khác. Cách phổ biến nhất là tham khảo các trang mạng hoặc hỏi người dân địa phương. Tuy nhiên, những cách đó cũng không tối ưu. Bởi vì ngay cả những dân bản địa cũng không hẳn là biết hết những địa điểm đẹp. Và trên thục tế, cũng chưa có một ứng dụng hay trang mạng chính thức nào chuyên giới thiệu các địa điểm du lịch. Với động lực chính đáng đó, nhóm chúng em đề xuất xây dựng một ứng dụng mạng xã hội chia sẻ địa điểm trên thiết bị Android phục vụ nhu cầu du lịch của mọi người.

Mục tiêu cơ bản của ứng dụng là hướng đến việc giới thiệu địa điểm du lịch cho người sử dụng. Với chức năng cơ bản của một mạng xã hội, ứng dụng còn có thể lưu giữ những kỉ niệm, khoảnh khắc đẹp của người dùng với địa điểm đó và chia sẻ đến bạn bè, mọi người xung quanh. Bên cạnh đó, ứng dụng còn cung cấp một số chức năng khác như tìm kiếm địa điểm theo nhu cầu, theo dõi trực tuyến hành trình. Những địa điểm được giới thiệu, chia sẻ là những địa điểm đã được mọi người đi tới và chứng thực. Do đó, người dùng có thể hoàn toàn tin tưởng và sử dụng ứng dụng.

## Mô hình use-case

### Sơ đồ use-case



### Đặc tả use-case

* Danh sách các Actor

|  |  |  |
| --- | --- | --- |
| STT | Tên Actor | Ý nghĩa/Ghi chú |
| 1 | Người dùng | Người sử dụng ứng dụng |
| 2 | Google Maps | Ứng dụng Google Maps, được sử dụng khi xem bài viết trên bản đồ |
| 3 | Location System | Hệ thống lưu trữ vị trí của ứng dụng |

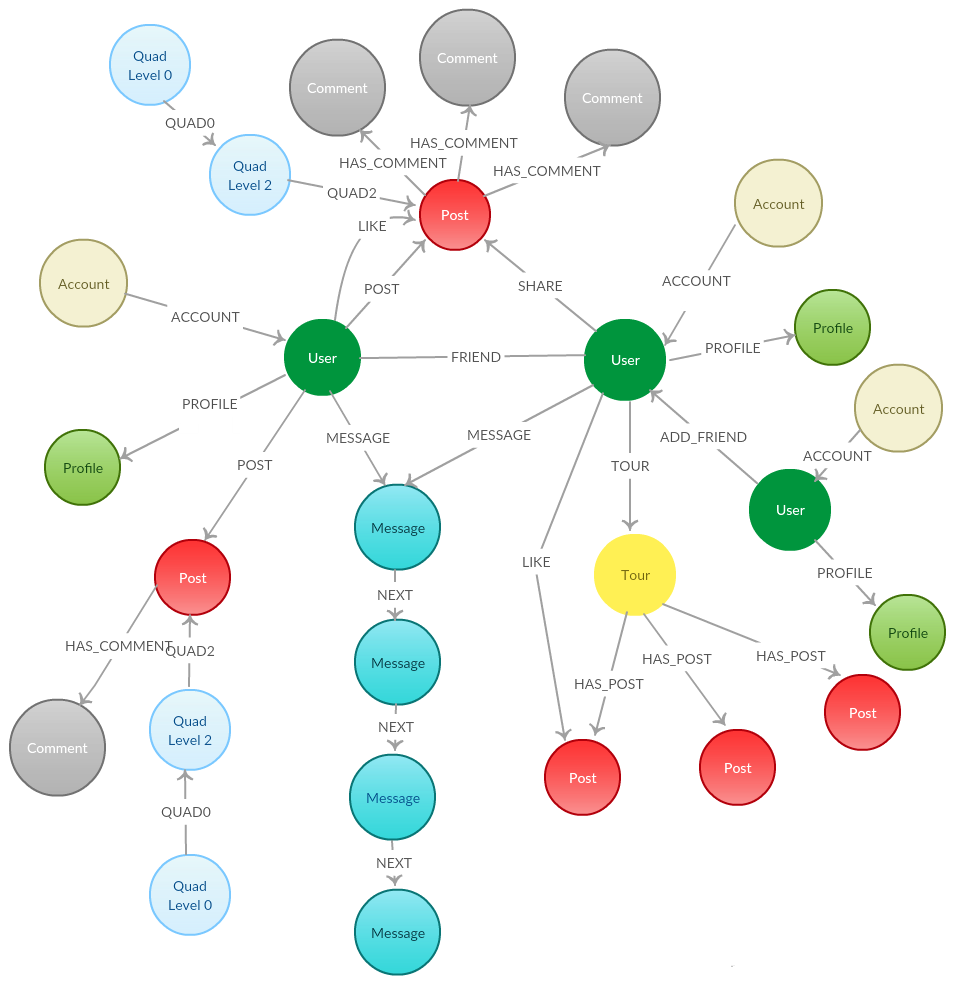
* Danh sách các Use-case

|  |  |  |
| --- | --- | --- |
| STT | Tên Use-case | Ý nghĩa/Ghi chú |
| 1 | Đăng ký tài khoản | Người sử dụng thực hiện chức năng đăng ký tài khoản mới. |
| 2 | Đăng nhập | Người sử dụng thực hiện chức năng đăng nhập vào tài khoản. |
| 3 | Khôi phục mật khẩu | Người sử dụng thực hiện chức năng khôi phục mật khẩu. |
| 4 | Đăng xuất | Người sử dụng thực hiện chức năng đăng xuất khỏi tài khoản. |
| 5 | Đổi mật khẩu | Người sử dụng thực hiện chức năng đổi mật khẩu mới. |
| 6 | Xem thông tin cá nhân | Người sử dụng thực hiện chức năng xem thông tin cá nhân của mình. |
| 7 | Cập nhật thông tin cá nhân | Người sử dụng thực hiện chức năng cập nhật thông tin cá nhân, thực hiện cập nhật bên trong trang xem thông tin cá nhân. |
| 8 | Xem trang chính | Người sử dụng thực hiện chức năng xem trang chính. Trang chính là trang xuất hiện đầu tiên sau khi đăng nhập, chứa danh sách các bài viết mới nhất. |
| 9 | Xem thông báo | Người sử dụng thực hiện chức năng xem thông báo mới. |
| 10 | Xem tin nhắn | Người sử dụng thực hiện chức năng xem tin nhắn. Lịch sử các tin nhắn. |
| 11 | Nhắn tin | Người sử dụng thực hiện chức năng nhắn tin. |
| 12 | Xem danh sách bạn | Người sử dụng thực hiện chức năng xem danh sách bạn bè. |
| 13 | Tìm kiếm bạn | Người sử dụng thực hiện chức năng tìm kiếm bạn đã có hoặc tìm kiếm bạn mới |
| 14 | Kết bạn | Người sử dụng thực hiện chức năng kết bạn, gửi lời mời kết bạn. |
| 15 | Hủy kết bạn | Người sử dụng thực hiện chức năng hủy bỏ quan hệ bạn với người khác |
| 16 | Xem trên bản đồ | Người sử dụng thực hiện chức năng xem trên bản đồ. Chuyển giao diện ứng dụng sang bản đồ. |
| 17 | Tìm kiếm địa điểm | Người sử dụng thực hiện chức năng tìm kiếm một địa điểm trên bản đồ. |
| 18 | Tìm kiếm bài viết | Người sử dụng thực hiện chức năng tìm kiếm một bài viết. Việc tìm kiếm được thực hiện trên bản đồ với nhiều tiêu chí khác nhau. |
| 19 | Tạo bài viết | Người sử dụng thực hiện chức năng tạo một bài viết mới và chia sẻ với mọi người. |
| 20 | Tạo hành trình | Người sử dụng thực hiện chức năng tạo hành trình. Trực tuyến một hành trình, bao gồm nhiều bài viết. Mọi người có thể theo dõi hành trình. |
| 21 | Xem hành trình | Người sử dụng thực hiện chức năng xem hành trình của mình hoặc của người khác. Hành trình bao gồm danh sách các bài viết thuộc hành trình đó. |
| 22 | Xem danh sách bài viết | Người sử dụng thực hiện chức năng xem danh sách các bài viết theo một tiêu chí xác định. |
| 23 | Xem chi tiết bài viết | Người sử dụng thực hiện chức năng xem chi tiết một bài viết cụ thể. |
| 24 | Quan tâm bài viết | Người sử dụng thực hiện chứ năng Like/Thích một bài viết. |
| 25 | Bình luận bài viết | Người sử dụng thực hiện chức năng Comment/Bình luận một bài viết. |
| 26 | Cập nhật bài viết | Người sử dụng thực hiện chức năng Edit/Cập nhật một bài viết. |
| 27 | Chia sẻ bài viết | Người sử dụng thực hiện chức năng Share/Chia sẻ một bài viết. |

## Mô hình dữ liệu

### Sơ đồ dữ liệu

Với cơ sở dữ liệu đồ thị Neo4j, các thực thể lưu trữ dữ liệu được biểu diễn dưới dạng các đỉnh có các khóa để phân biệt với nhau, liên kết với với nhau bằng các cạnh thay cho khóa ngoại như RDBMS. Bên trong các đỉnh, chứa các trường thuộc tính lưu trữ theo yêu cầu.



### Đặc tả cơ sở dữ liệu

|  |  |  |
| --- | --- | --- |
| STT | Tên thực thể | Ý nghĩa/Ghi chú |
| 1 | User | Người sử dụng ứng dụng, bao gồm thông tin về tên, ảnh đại diện. |
| 2 | Account | Thông tin tài khoản của người sử dụng, bao gồm email, mật khẩu. |
| 3 | Profile | Thông tin cá nhân của người sử dụng, bao gồm các chi tiết về địa chỉ, ngày sinh,… |
| 4 | Post | Thông tin chi tiết về một bài viết bao gồm nội dung, các hình ảnh liên quan và vị trí địa lý tương ứng. |
| 5 | Tour | Lưu giữ thông tin của một chuyến hành trình. Một chuyến hành trình có thể có nhiều bài viết liên quan. |
| 6 | Message | Lưu giữ nội dung của một tin nhắn trong cuộc trò chuyện giữa hai người sử dụng. |
| 7 | Comment | Chứa thông tin chi tiết về các bình luận của người sử dụng trên các bài viết. Bao gồm nội dung, thời gian bình luận. |
| 8 | Quad Level 0 | Một đối tượng lưu trữ cấu hình của lần phân hoạch lần thứ nhất trên cấu trúc dữ liệu Quadtree. Đối tượng này chứa thông tin tọa độ của các bài viết trên đơn vị độ. |
| 9 | Quad Level 2 | Tương tự như Quad Level 0, đối tượng này sẽ lưu trữ cấu hình của lần phân hoạch lần thứ hai trên cấu trúc dữ liệu Quadtree nhưng chứa thông tin tọa độ các bài viết trên đơn vị phút. |
| 10 | ACCOUNT | Biểu diễn mối quan hệ giữa thông tin tài khoản với người sử dụng. Cụ thể, người sử dụng sẽ sở hữu một tài khoản và được biểu diễn bởi cạnh ACCOUNT. |
| 11 | PROFILE | Biểu diễn mối quan hệ giữa thông tin cá nhân với người sử dụng. Người sử dụng sẽ có một cạnh nối PROFILE đến thông tin cá nhân của mình. |
| 12 | FRIEND | Biểu diễn mối quan hệ giữa hai người sử dụng. Cụ thể ở đây là mối quan hệ bạn. Cạnh nối FRIEND sẽ được hình thành khi hai người là bạn của nhau. |
| 13 | ADD\_FRIEND | Biểu diễn thông tin yêu cầu kết bạn từ một người sử dụng đến một người khác. Một cạnh nối tên ADD\_FRIEND sẽ xuất hiện để thể hiện sự yêu cầu. |
| 14 | CONFIRM | Biểu diễn thông tin xác nhận yêu cầu kết bạn tương ứng với ADD\_FRIEND. Cạnh nối sẽ xuất hiện khi người sử dụng xác nhận đồng ý thêm bạn. |
| 15 | POST | Biểu diễn mối liên kết giữa người sử dụng với bài viết của mình. Một người sử dụng có thể có nhiều bài post tương ứng. |
| 16 | LIKE | Biểu diễn thông tin ghi nhận người sử dụng quan tâm đến một bài viết. Cạnh nối xuất hiện khi thực hiện chức năng Like/Thích. |
| 17 | SHARE | Biểu diễn mối liên kết chia sẻ bài viết của một người sử dụng. Cạnh nối xuất hiện khi thực hiện chức năng Share/Chia sẻ. |
| 18 | HAS\_COMMENT | Biểu diễn thông tin các lượt bình luận của người sử dụng đến một bài viết. Cạnh nối xuất hiện khi thực hiện chức năng Comment/Bình luận. |
| 19 | TOUR | Biểu diễn thông tin liên kết giữa người dùng với thông tin hành trình của mình. Khi thực hiện chức năng tạo Tour/Hành trình, cạnh nối sẽ xuất hiện. |
| 20 | HAS\_POST | Biểu diễn mối liên kết giữa hành trình với các bài viết bên trong hành trình đó. Khi tạo một bài viết trong một hành trình, cạnh nối sẽ xuất hiện. |
| 21 | MESSAGE | Biểu diễn thông tin liên kết giữa một tin nhắn với một người sử dụng. Cạnh nối xuất hiện khi người sử dụng thực hiện chức năng nhắn tin. |
| 22 | NEXT | Biểu diễn sự liên kết các tin nhắn lại với nhau theo thứ tự. Tin nhắn này sẽ trỏ đến tin nhắn liền kề bằng cạnh NEXT. |
| 23 | QUAD0 | Biểu diễn sự liên kết các đỉnh trong quadtree sau lần phân hoạch đầu tiên. Quad Level 0 liên kết với Quad Level 2 thông qua cạnh QUAD0. |
| 24 | QUAD2 | Biểu diễn sự liên kết các đỉnh trong quadtree sau lần phân hoạch thứ hai. Quad Level 2 liên kết với thực thể khác thông qua cạnh QUAD2. |

## Thiết kế kiến trúc hệ thống

### Sơ đồ kiến trúc

Kiến trúc hệ thống được chia phân chia thành 3 phần: Server, Client và hệ thống ứng dụng bên thứ 3. Các thành phần này sẽ kết nối, liên kết với nhau để truy xuất dữ liệu theo yêu cầu. Minh họa cụ thể như hình bên dưới.

Database

**Server**

Database Connection

API for Client

GCM Connection

Mail Services

**…**

**OutSide**

Gmail Server

GCM Server

**Client**

HOMFY

### Đặc tả kiến trúc

Hệ thống được xây dựng theo mô hình Client-Server. Trong đó, Client là một ứng dụng trên thiết bị Android, cung cấp các chức năng và tương tác với người sử dụng. Server có nhiệm vụ xử lý thông tin do Client gửi lên, kết nối với cơ sở dữ liệu và các hệ thống khác nhằm truy xuất dữ liệu và trả kết quả về.

Server hỗ trợ Client thông qua dịch vụ Web Restful API, một giao thức truyền tải dữ liệu đang rất phổ biến hiện nay. Server cung cấp đầy đủ các API tương ứng với các chức năng của Client. Khi cần truy xuất dữ liệu, Client sẽ gửi truy vấn dưới dạng giao thức HTTP, Server sẽ xử lý và trả kết quả ở dạng JSON. Với định dạng dữ liệu JSON, Client có thể dễ dàng chuyển đổi sang dữ liệu đối tượng và xử lý.

Bên cạnh đó, Client và Server còn kết nối với các hệ thống khác như Google Cloud Messaging hay Gmail Server để tương tác nhằm thực hiện một số chức năng nhất định. Server liên kết đến Google Cloud Messaging nhằm thực hiện chức năng Push Notification đến toàn bộ ứng dụng, kết nối đến Gmail Server để thực hiện gửi email. Client sẽ lắng nghe phản hồi từ Google Cloud Messaging, nhận dữ liệu, xử lý và phản hồi cho người sử dụng.

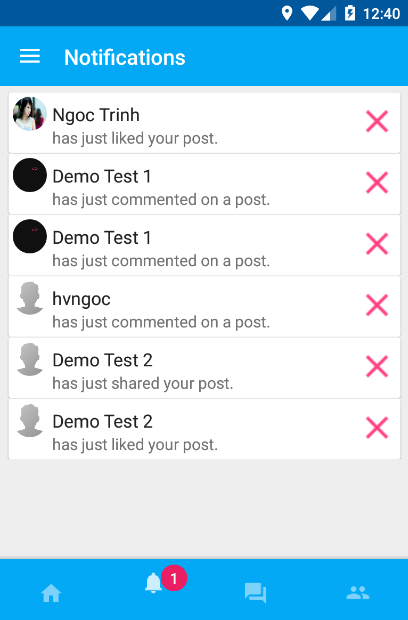
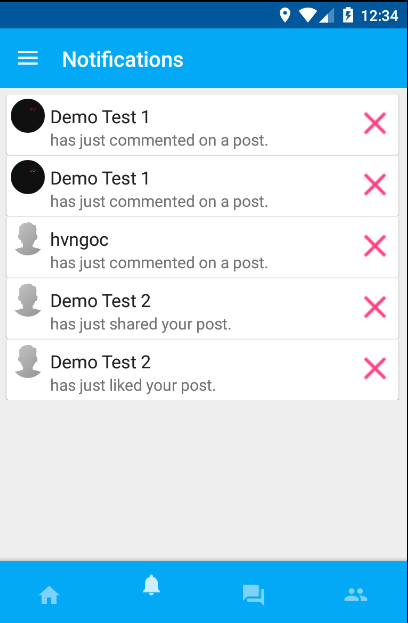
## Các chức năng của ứng dụng

### Đăng ký tài khoản

### Đăng nhập vào tài khoản

### Xem thông báo

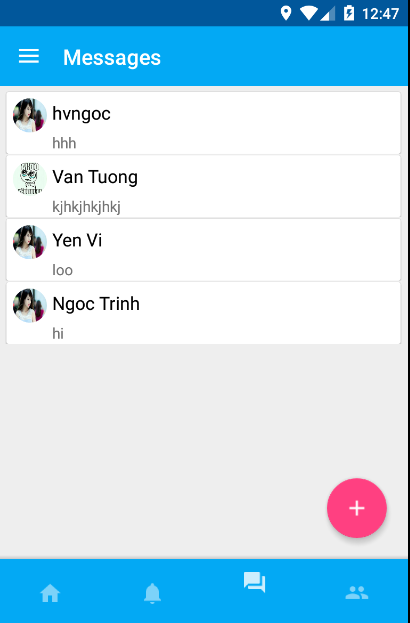
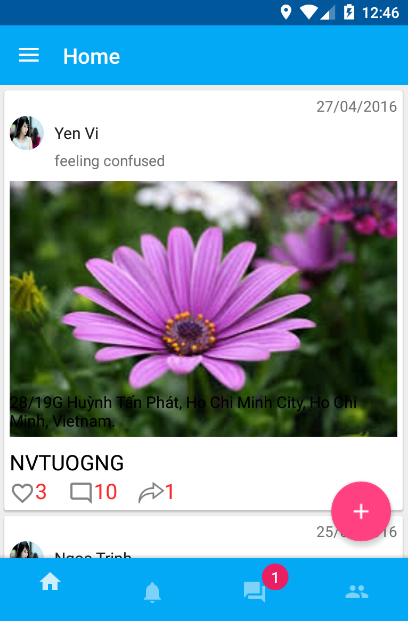
Người sử dụng có thể nhấn vào phần Notification/Thông báo trên thanh công cụ nằm phí dưới của ứng dụng để xem thông báo mới. Thông báo bao gồm các sự kiện như yêu cầu, xác nhận kết bạn, quan tâm, chia sẻ hoặc bình luận một bài viết nào đó.



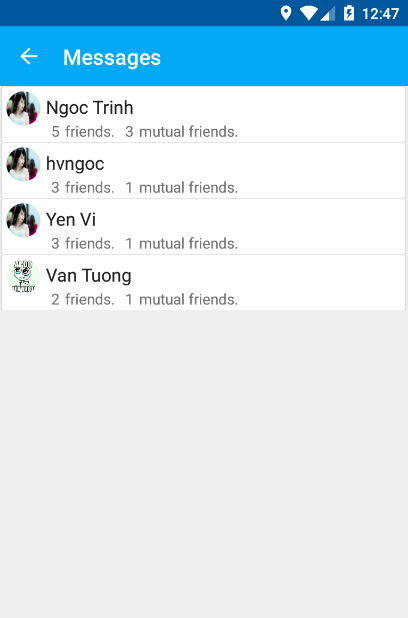
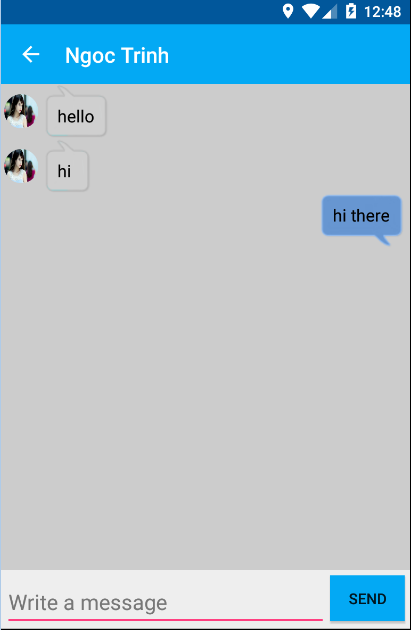
Bên trong chức năng thông báo, người sử dụng có thể xóa thông báo khi đã xem. Khi có một thông báo mới, thanh công cụ sẽ hiển thị thông báo như hình bên trên.

### Xem tin nhắn

Để xem lịch sử tin nhắn hoặc tạo tin nhắn mới, người sử dụng có thể nhấn vào phần Message/Nhắn tin trên thanh công cụ nằm ở phía dưới của ứng dụng. Tương tự như phần thông báo, khi có một tin nhắn mới đến, ứng dụng cũng sẽ thông báo bằng cách hiển thị số ở dưới thanh công cụ. Người dùng nhấn vào để có thể xem tin nhắn đó.



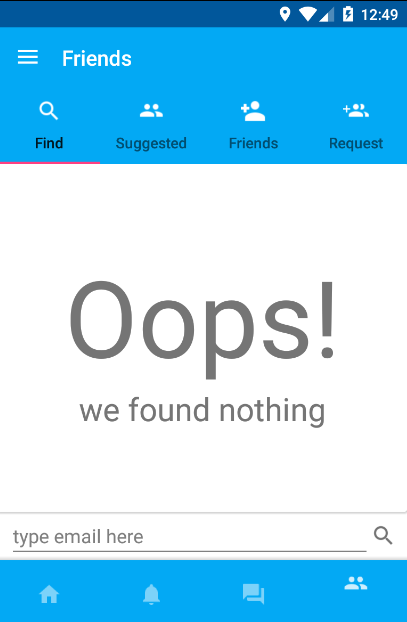
Bên cạnh đó, người sử dụng còn có thể thực hiện tạo tin nhắn mới để gửi đến bạn của mình. Chức năng tin nhắn mới minh họa như hình bên dưới. Người sử dụng nhấn vào biểu tượng tạo tin nhắn, chọn người muốn nhắn tin và thực hiện trò chuyện. Tin nhắn nhận sẽ được hiển thị bên trái cùng với ảnh đại diện. Tin nhắn của chủ thể được hiển thị ở bên phải mới màu nền xanh.

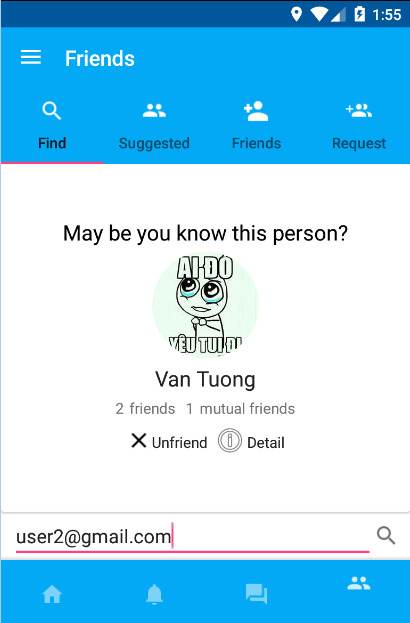
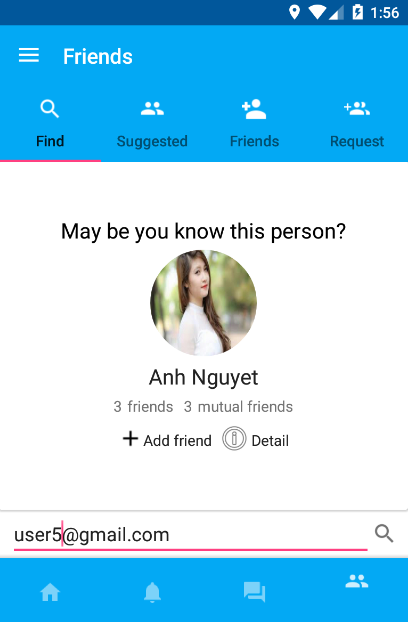
### Xem danh sách bạn

Để xem danh sách bạn, người sử dụng nhấn vào phần Friend/Bạn bè. Giao diện bên trong sẽ được chia thành 4 phần nhỏ hơn tương ứng với 4 chức năng: tìm kiếm bạn, xem gợi ý kết bạn, xem danh sách bạn, xem yêu cầu kết bạn.

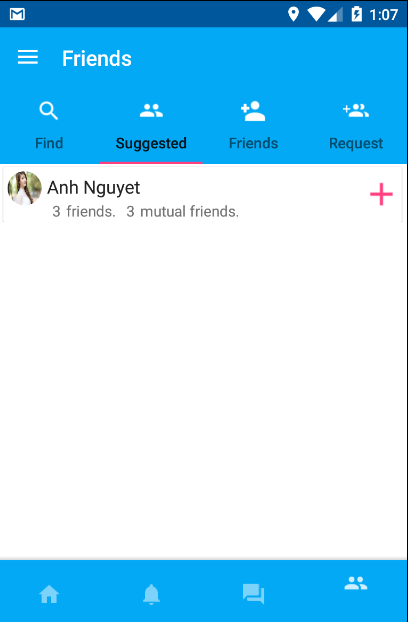
Để tìm kiếm bạn, người sử dụng nhập vào email tương ứng để tìm. Người sử dụng có thể tìm kiếm một người khác đã là bạn hoặc chưa có quan hệ.



Tương ứng với từng nhu cầu, giao diện sẽ thay đổi. Nếu kết quả tìm kiếm đã có mối quan hệ bạn thì giao diện sẽ có thông tin hủy kết bạn. Và ngược lại, nếu kết quả tìm kiếm chưa có quan hệ bạn thì giao diện sẽ có thông tin kết bạn. Minh họa như hình bên dưới.

Để xem gợi ý kết bạn, người sử dụng nhấn vào phần Suggested/Gợi ý. Danh sách các lời gợi ý kết bạn sẽ được hiển thị. Người dùng nhấn vào từng thành phần trong danh sách để thực hiện chức năng kết bạn tương ứng.



### Xem thông tin cá nhân

### Tạo bài viết

### Xem chi tiết bài viết

### Xem trên bản đồ

## Kịch bản sử dụng ứng dụng

## Kết luận

# Thực nghiệm và đánh giá

* Nội dung của chương 5 trình bày về kết quả thực nghiệm khi triển khai ứng dụng. Phân tích các nguyên nhân giải pháp và đánh giá sơ bộ dựa trên các trải nghiệm của kết quả. Từ đó, nhận ra các điểm mạnh cũng như thiếu sót, rút ra kinh nghiệm và hình thành các hướng giải quyết cải thiện ứng dụng.

## Kết quả thực nghiệm

## Tổng kết đánh giá

## Kết luận

# Kết luận và hướng phát triển

✍ Nội dung của chương 6 trình bày tóm lược những nội dung mà nhóm đã tìm hiểu và các kết quả đạt được trong quá trình thực hiện đề tài, qua đó mở ra hướng phát triển mới của đề tài trong tương lai

## Các kết quả đạt được

## Hướng phát triển của đề tài

TÀI LIỆU THAM KHẢO

|  |  |
| --- | --- |
| [1] | “History Of Social Networking: How It All Began,” 1 April 2016. [Trực tuyến]. Available: http://1stwebdesigner.com/history-of-social-networking/. |
| [2] | D. T. Staff, “THE HISTORY OF SOCIAL NETWORKING,” [Trực tuyến]. Available: http://www.digitaltrends.com/features/the-history-of-social-networking/. |
| [3] | “Number of monthly active Facebook users worldwide as of 1st quarter 2016 (in millions),” [Trực tuyến]. Available: http://www.statista.com/statistics/264810/number-of-monthly-active-facebook-users-worldwide/. |
| [4] | Wikipedia, “Zalo,” [Trực tuyến]. Available: https://vi.wikipedia.org/wiki/Zalo. |
| [5] | “Exploring the software behind Facebook, the world’s largest site,” [Trực tuyến]. Available: http://royal.pingdom.com/2010/06/18/the-software-behind-facebook/. |
| [6] | Wikipedia, “Memcached,” [Trực tuyến]. Available: https://en.wikipedia.org/wiki/Memcached. |
| [7] | J. Bullas, “21 Awesome Facebook Facts and Statistics You Need To Check Out,” [Trực tuyến]. Available: http://www.jeffbullas.com/2015/04/17/21-awesome-facebook-facts-and-statistics-you-need-to-check-out/. |
| [8] | C. Smith, “By the number: 125+ Amazing Youtube statistics,” [Trực tuyến]. Available: http://expandedramblings.com/index.php/youtube-statistics/. |
| [9] | T. Hoff, “7 Years Of YouTube Scalability Lessons In 30 Minutes,” [Trực tuyến]. Available: http://highscalability.com/blog/2012/3/26/7-years-of-youtube-scalability-lessons-in-30-minutes.html. |
| [10] | Wikipedia, “Python (programming language),” [Trực tuyến]. Available: https://en.wikipedia.org/wiki/Python\_(programming\_language). |
| [11] | A. Yeh, “Google Cloud Platform Blog,” [Trực tuyến]. Available: https://cloudplatform.googleblog.com/2015/03/scaling-MySQL-in-the-cloud-with-Vitess-and-Kubernetes.html. |
| [12] | Wikipedia, “Psyco,” [Trực tuyến]. Available: https://en.wikipedia.org/wiki/Psyco. |
| [13] | V. C. THANH DAO, “Inside Zalo: Developing a mobile messager for the audiens of milions,” [Trực tuyến]. Available: http://www.slideshare.net/QuangDiuNguyn/vn-mobile-day2013-zalo-sharing-21483998. |
| [14] | V. C. Nguyen Quang Nam, “Zalo real-tine system: Architecture and Experience,” [Trực tuyến]. Available: http://www.slideshare.net/Zalo\_app/experience-lessons-from-architecture-of-zalo-real-time-system. |
| [15] | Wikipedia, “PostgreSQL,” [Trực tuyến]. Available: https://en.wikipedia.org/wiki/PostgreSQL. |
| [16] | Pradeep, Jadhav; Oberoi, Ruhi, “Comparative Analysis of Graph Database Models using Classification and Clustering by using Weka tool,” *International Journal of Advanced Research in Computer Science and Software Engineering ,* tập 5, số 2, pp. 438-445, 2015. |
| [17] | S. Patil, G. Vaswani và A. Bhatia, “Graph Databases - An Overview,” *International Journal of Computer Science and Information Technologies (IJCSIT),* tập 5, số 1, pp. 657-660, 2014. |
| [18] | S. A. T. Mpinda, L. C. Ferreira và M. X. Ribeiro, “Evaluation of Graph Databases performance through indexing techiques,” *International Journal of Artificial Intelligence & Applications (IJAIA),* tập 6, số 5, pp. 87-98, September 2015. |
| [19] | S. A. T. Mpinda và P. A. Bungama, “Graph database application using Neo4j,” *International Journal of Engineering Research & Technology (IJERT),* tập 4, số 4, pp. 999-1002, April 2015. |
| [20] | D. A. Abbas, “Image watermark detection techniques using quatrees,” *Applied Computing and Informatics,* tập 11, số 2, pp. 102-115, July 2015. |
| [21] | D. Funda và O. Oğuzcan, “Dynamic point-region quadtrees for particles simulations,” *Information Sciences, Elsevier,* tập 218, số 1, pp. 133-145, 2013. |
| [22] | M .Mohd, “Development of Search Engines using Lucene: An Experience,” *Procedia Social and Behavioral Sciences,* tập 18, pp. 282-285, 2011. |
| [23] | Wikipedia, “Global Positioning System,” [Trực tuyến]. Available: https://en.wikipedia.org/wiki/Global\_Positioning\_System. |
| [24] | J. C. Herrera, B. W. Daniel, R. Herring, X .Ban, Q. Jacobson và A. M. Bayen, “Evaluation of traffic data obtained via GPS-enabled mobile phones: The Mobile Century field experiment,” *Transportation Research C.: Emerging Technologies,* tập 18, số 4, pp. 568-583, August 2010. |
| [25] | P. Doshi, P. Jain và A. Shakwala, “Location Based Services and Integration of Google Maps in Android,” *International Journal Of Engineering And Computer Science (IJECS),* tập 3, số 3, pp. 5072-5077, March 2014. |
| [26] | Wikipedia, “Android (operating system),” [Trực tuyến]. Available: https://en.wikipedia.org/wiki/Android\_(operating\_system). |
| [27] | Y. A. Yilmaz, B. I. Aydin và D. Murat, “Google Cloud Messaging (GCM): An Evaluation,” *IEEE Global Communications Conference,* tập 14, số 3, pp. 2847-2852, December 2014. |
| [28] | H.Singh, S. Kumar và H. Kaur, “Location Based System Using Google Cloud Messaging,” *National Conference on Innovative Trends in Computer Science Engineering (ITCSE),* tập 13, số 4, pp. 183-186, 2015. |
| [29] | K. Subramani, V. Hemapriya, L. A. Anto và V. R. Yazhini, “NodeJS: Building a High Performance Event Manager in Android Flatform,” *Australian Journal Of Basic And Applied Sciences,* tập 10, số 1, pp. 174-177, January 2016. |
| [30] | S. L. Bangare, S. Gupta, M. Dalal và A. Inamdar, “Using Node.Js to Build High Speed and Scalable Backend Database Server,” *International Journal of Research in Advent Technology,* tập 14, số 6, pp. 61-64, March 2016. |
| [31] | K. Ganwani và K. Jha, “Google Material Design Visual Language Technology By Google,” *Journal of Global Research Computer Science & Technology ,* tập 3, số 3, pp. 12-14, February 2015. |

PHỤ LỤC

Phụ lục 1. Đặc tả cấu trúc file XML lưu trữ các thông tin tăng cường cho video