

**ĐẠI HỌC QUỐC GIA HÀ NỘI
TRƯỜNG ĐẠI HỌC CÔNG NGHỆ**

Đào Hoàng Sơn

HỆ THỐNG TỔNG HỢP TIN TỨC

KHÓA LUẬN TỐT NGHIỆP ĐẠI HỌC HỆ CHÍNH QUY
Ngành: Công nghệ thông tin

HÀ NỘI - 2012

**ĐẠI HỌC QUỐC GIA HÀ NỘI
TRƯỜNG ĐẠI HỌC CÔNG NGHỆ**

Đào Hoàng Sơn

HỆ THỐNG TỔNG HỢP TIN TỨC

KHÓA LUẬN TỐT NGHIỆP ĐẠI HỌC HỆ CHÍNH QUY

Ngành: Công nghệ thông tin

Cán bộ hướng dẫn: TS. Nguyễn Ngọc Hóa

HÀ NỘI - 2012

**VIETNAM NATIONAL UNIVERSITY, HANOI
UNIVERSITY OF ENGINEERING AND TECHNOLOGY**

Dao Hoang Son

NEWS AGGREGATOR SYSTEM

Major: Information Technology

Supervisor: Dr. Nguyen Ngoc Hoa

HA NOI - 2012

LỜI CẢM ƠN

Trước tiên, tôi xin bày tỏ lòng cảm ơn chân thành đến TS. Nguyễn Ngọc Hóa, người đã hết lòng chỉ bảo, hướng dẫn cho tôi các phương pháp tiếp cận vấn đề, giúp tôi giải đáp những thắc mắc, giải quyết các khó khăn mắc phải để hoàn thành được khóa luận này.

Tôi cũng xin chân thành cảm ơn các thầy cô ở khoa Công nghệ thông tin đã hết lòng truyền đạt cho tôi những kiến thức nền tảng về Công nghệ thông tin, giúp tôi có đủ kiến thức để tiếp cận, hoàn thành được khóa luận này. Bên cạnh đó tôi cũng xin gửi lời cảm ơn đến sự giúp đỡ nhiệt tình của nhà trường, tạo điều kiện tốt nhất cho tôi để hoàn thành khóa luận này.

Tôi xin gửi lời cảm ơn sâu sắc đến những người thân trong gia đình, họ đã động viên rất nhiều về mặt tinh thần để tôi có đủ nghị lực hoàn thành khóa luận này. Tôi cũng xin chân thành cảm ơn bạn bè tôi, những người đã luôn bên cạnh, động viên, tạo mọi điều kiện thuận lợi nhất cho tôi trong quá trình hoàn thành khóa luận.

Hà Nội, tháng 5 năm 2012

Sinh viên

Đào Hoàng Sơn

HỆ THỐNG TỔNG HỢP TIN TỨC

Đào Hoàng Sơn

Khóa QH-2008-I/CQ, ngành Công nghệ thông tin

Tóm tắt Khóa luận tốt nghiệp

Trong thời đại bùng nổ thông tin và các phương tiện thông tin đại chúng, người sử dụng mạng Internet trên thế giới nội chung và ở Việt Nam nói riêng có thể tiếp cận với một lượng thông tin khổng lồ. Các thông tin này rất khó có thể tổng hợp một cách có hệ thống, người đọc khó nắm bắt được hết các tin tức hoặc mất rất nhiều thời gian để đọc hết các tin bài. Do vậy, trong khóa luận này tôi muốn phát triển một hệ thống phân tích các bài báo và đưa ra thông tin tổng hợp giúp người xem nhanh chóng nắm bắt được các vấn đề trong xã hội.

Từ khóa: xử lý text, xu hướng, scaling

LỜI CAM ĐOAN

Tôi xin cam đoan kết quả đạt được trong khoá luận này là sản phẩm của riêng cá nhân tôi. Trong toàn bộ nội dung của khoá luận, những điều được trình bày hoặc là của cá nhân hoặc là được tổng hợp từ nhiều nguồn tài liệu. Tất cả những tham khảo từ các nghiên cứu liên quan đều được chỉ rõ trong trích dẫn và danh mục tài liệu tham khảo một cách rõ ràng.

Tôi xin hoàn toàn chịu trách nhiệm theo quy định cho lời cam đoan của mình.

Hà Nội, tháng 5 năm 2012

Sinh viên

Đào Hoàng Sơn

MỤC LỤC

MỞ ĐẦU	2
CHƯƠNG 1. TỔNG QUAN HỆ THỐNG	3
1.1. Hiện trạng, vấn đề cần giải quyết	3
1.2. Mục tiêu của hệ thống.....	3
1.3. Ý tưởng thiết kế	4
1.3.1. Hệ thống xử lý văn bản tiếng Việt.....	5
1.3.2. Hệ thống tổng hợp tin tức	6
CHƯƠNG 2. CÁC CÔNG CỤ ĐƯỢC SỬ DỤNG	7
2.1. Giới thiệu về các hệ quản trị cơ sở dữ liệu	7
2.1.1. Hệ quản trị cơ sở dữ liệu quan hệ	7
2.1.2. NoSQL	10
2.2. Giới thiệu về RESTful web API	15
2.2.1. Cơ bản.....	15
2.2.2. Sử dụng các phương thức của giao thức HTTP.....	15
2.2.3. Không lưu trạng thái	18
2.2.4. Sử dụng các URI có cấu trúc giống các thư mục	19
2.2.5. Hỗ trợ XML, JSON hoặc cả hai	20
2.3. Giới thiệu về crawler/web crawler.....	20
2.4. Giới thiệu về các ngôn ngữ lập trình web.....	21
2.4.1. PHP	21
2.4.2. Ruby.....	22
2.4.3. Python	23
2.4.4. Node.js	24
CHƯƠNG 3. PHÂN TÍCH THIẾT KẾ HỆ THỐNG	25

3.1. Hệ thống xử lý tiếng Việt	25
3.1.1. Cấu trúc hệ thống.....	25
3.1.2. Cấu trúc cơ sở dữ liệu	27
3.1.3. API cho hệ thống xử lý tiếng Việt	29
3.1.4. Giải thuật xử lý	29
3.2. Crawler (Hệ thống tổng hợp tin tức)	31
3.2.1. Cấu trúc hệ thống.....	31
3.2.2. Cấu trúc cơ sở dữ liệu	32
3.2.3. Giải thuật xử lý	32
3.3. Front-end (Hệ thống tổng hợp tin tức).....	35
4.2.1. Thiết kế giao diện của front-end.....	35
4.2.2. Giải pháp cài đặt	38
CHƯƠNG 4. THỰC NGHIỆM	39
5.1. Kết quả thực nghiệm.....	39
5.1.1. Hệ thống xử lý tiếng Việt	39
5.1.2. Hệ thống tổng hợp tin tức, phần Crawler	40
5.2. Hướng phát triển tiếp theo	42
5.3. Kết luận.....	43
TÀI LIỆU THAM KHẢO	44
Tiếng Việt	44
Tiếng Anh	44
Endnote	44

DANH MỤC TỪ VIẾT TẮT

Từ viết tắt	Tiếng Anh	Tiếng Việt
CPU	Central Processing Unit	Bộ xử lý trung tâm
PaaS	Platform as a Service	Nền tảng như một dịch vụ
RDBMS	Relational Database Management System	Hệ quản trị cơ sở dữ liệu quan hệ
SaaS	Software as a Service	Phần mềm như một dịch vụ

MỞ ĐẦU

Trong thời đại bùng nổ thông tin và các phương tiện thông tin đại chúng, người sử dụng mạng Internet trên thế giới nội chung và ở Việt Nam nói riêng có thể tiếp cận với một lượng thông tin khổng lồ. Một phần lớn lượng thông tin đó được cung cấp bởi các trang báo mạng, trang tin tức hay blog cá nhân của những người nổi tiếng. Điểm chung của các thông tin này là: chủ yếu ở dạng text, có thể có các siêu liên kết đến các thông tin liên quan, trình bày dàn trải, ít có cấu trúc. Với những đặc điểm như vậy, các thông tin này rất khó có thể tổng hợp một cách có hệ thống, người đọc khó nắm bắt được hết các tin tức hoặc mất rất nhiều thời gian để đọc hết các tin bài.

Để giải quyết được vấn đề đó, khóa luận này tìm hiểu một giải pháp tổng hợp và phân tích các bài báo tiếng Việt trên diện rộng, một cách hoàn toàn tự động. Từ đó đưa ra được các thông tin hữu ích, mang tính tổng quát về tình hình xã hội, giúp người đọc nhanh chóng có cái nhìn cụ thể về tin tức trong ngày hoặc một khoảng thời gian nhất định. Với mục tiêu như vậy, có một số khó khăn nhất định về mặt kĩ thuật, khóa luận sẽ lần lượt trình bày các khó khăn đó cũng như giải pháp được lựa chọn để giải quyết.

CHƯƠNG 1. TỔNG QUAN HỆ THỐNG

1.1. Hiện trạng, vấn đề cần giải quyết

Chúng ta đang sống trong thế kỉ 21, công nghệ thông tin đang có những bước phát triển vượt bậc, kéo theo đó là lượng thông tin lớn liên tục được cung cấp bởi các trang tin, báo và tạp chí. Khi có quá nhiều nguồn thông tin như vậy thì việc đọc và nắm bắt được hết các vấn đề là chuyện không hề đơn giản.

Nhu cầu này không phải chỉ từ những cá nhân muốn tìm hiểu tin tức của xã hội mà còn là nhu cầu của các tập thể, công ty kinh doanh, điều tra thị trường. Ví dụ: công ty Ericsson Việt Nam hàng tháng luôn phải bỏ chi phí để thuê nhân lực bên ngoài thu thập các bài viết trong lĩnh vực viễn thông (lĩnh vực hoạt động chính của công ty) để có thể nắm bắt tình hình, xu thế của thị trường.

Với nhu cầu đó của thị trường, khóa luận này mong muốn xây dựng một hệ thống thu nhập bài viết từ các trang tin, báo và tạp chí của Việt Nam sau đó xử lý và cung cấp thông tin một cách có cấu trúc kèm theo số liệu thống kê cụ thể, dễ dàng tùy biến.

Hiện trên thị trường không có nhiều các công ty cung cấp sản phẩm tương tự dành cho nội dung tiếng Việt tuy nhiên cũng có một dịch vụ mạnh có hoạt động liên quan đó là BáoMới¹. Dịch vụ này tổng hợp và cung cấp bài viết từ nhiều nguồn khác nhau, với lợi thế của một trang tổng hợp tin tức, BáoMới có thể nhanh chóng chỉ ra các bài viết đang được nhiều người quan tâm (bấm vào đọc) cũng như tìm được các bài viết tương tự nhau (mục “Tin đăng lại”).

1.2. Mục tiêu của hệ thống

Khóa luận tốt nghiệp này mong muốn tìm ra được giải pháp đối với một số vấn đề thuần kĩ thuật

- Ứng dụng mô hình lưu trữ và xử lý phân tán để dễ dàng cấu hình hệ thống cho phù hợp khi nhu cầu xử lý tăng hoặc giảm.
- Xử lý ngôn ngữ tiếng Việt một cách cơ bản. Xử lý ngôn ngữ là một bài toán lớn. Xử lý tiếng Việt là bài toán còn lớn hơn và hiện có rất nhiều hướng nghiên cứu về vấn đề này của nhiều tác giả khác nhau có thể kể đến như Lê Hồng Phương², nhóm tác giả Phan Xuân Hiếu – Nguyễn Cẩm Tú³. Trong

phạm vi của khóa luận sẽ không đi sâu vào vấn đề này mà chỉ thực hiện đủ để thu được thông tin cần thiết phục vụ cho hệ thống như:

- Tên người, tên địa danh
- Các cụm từ nổi bật

1.3. Ý tưởng thiết kế

Sau khi tìm hiểu các hệ thống liên quan đang có trên thị trường như:

1. BáoMới (miễn phí)

- Tổng hợp thông tin tự động từ 60 nguồn khác nhau, với số lượng 3500 tin tức mỗi ngày. Các thao tác xử lý bao gồm: tìm kiếm các bài trùng lặp (các bài được nhiều trang tin đăng lại của nhau), nhóm các bài viết liên quan theo từng chuyên mục.
- Cung cấp thống kê sơ lược về số lượt đọc và số bài viết theo từng nguồn tin⁴.
- Cho phép người dùng tạo bộ lọc theo chuyên mục, từ khóa để theo dõi tin tức (từ khóa được xử lý theo kiểu FULLTEXT SEARCH).

2. Cubit NewsDesk⁵ (dịch vụ trả phí)

- Tổng hợp thông tin tự động từ nhiều nguồn khác nhau trên Thế giới (bao gồm cả tiếng Việt, tiếng Anh và nhiều ngôn ngữ khác)
- Cho phép người dùng tạo bộ lọc theo từ khóa, thời gian đăng tải, nguồn tin, chủ đề, tên người, tên công ty, tên sản phẩm,...

Có thể thấy nhu cầu xử lý của các hệ thống này về cơ bản là giống nhau: xử lý dữ liệu ở dạng văn bản, phân lớp (classify) thành các nhóm nhỏ theo chủ đề, phân tích chọn lọc ra các từ khóa quan trọng, tên người hoặc danh từ riêng sau đó cung cấp công cụ để trích xuất, lọc, tìm kiếm dựa trên các thông tin đã thu được. Về xử lý tin tức có điểm đặc thù là cần xác định các tin liên quan, các tin trùng lặp được các nguồn đăng lại của nhau để xử lý cho phù hợp. Qua khảo sát cũng thấy nhu cầu xử lý văn bản tiếng Việt hiện nay còn có thể được sử dụng trong nhiều lĩnh vực khác nữa. Ví dụ như trong trường Đại học hoặc các tổ chức giáo dục luôn có nhu cầu xử lý bài viết của người học để xác định các trường hợp sao chép không được phép. Trên thế giới có nhiều dịch vụ, phần mềm hỗ trợ

cho việc xử lý này tuy nhiên việc hỗ trợ tiếng Việt cũng chưa tốt ví dụ như Copyscape⁶ (miễn phí), turnitin⁷ (trả phí), SafeAssign⁸ (trả phí).

Chính vì vậy, khóa luận này tìm hiểu xây dựng hai hệ thống riêng biệt sẵn sàng có thể tiếp tục phát triển hơn nữa trong tương lai.

1.3.1. Hệ thống xử lý văn bản tiếng Việt

Đây là hệ thống cốt lõi được xây dựng làm nền tảng để phục vụ các hệ thống khác. Các hệ thống bên ngoài này (ví dụ như hệ thống tổng hợp tin tức) sẽ chạy độc lập với hệ thống xử lý văn bản tiếng Việt và được coi là một “ứng dụng” sử dụng API của hệ thống này. Cụ thể hơn, hệ thống sẽ được cài đặt là một RESTful web API, hoạt động trên giao thức HTTP hoặc HTTPS qua mạng Internet. Trong điều kiện kết nối như hiện nay, việc phát triển một hệ thống như vậy có thể mở ra rất nhiều khả năng và ứng dụng mới. Tuy vậy, hệ thống xử lý văn bản tiếng Việt sẽ chỉ cung cấp những dịch vụ cơ bản, bao gồm:

- Phân tích và lưu trữ văn bản thành các từ khóa.
- Tìm kiếm, lấy thông kê theo từ khóa.
- Tìm kiếm văn bản tương tự với một văn bản đầu vào.
- Phân lớp một văn bản đầu vào dựa trên dữ liệu và tập các lớp đã có.

Hệ thống này phải được thiết kế để đáp ứng các yêu cầu như sau:

- Có khả năng hỗ trợ nhiều ứng dụng một lúc, đảm bảo an toàn thông tin. Ví dụ như cùng một lúc có thể vừa phục vụ ứng dụng tổng hợp tin tức vừa phục vụ ứng dụng tìm khóa luận bị sao chép của Đại học Quốc gia Hà Nội.
- Có khả năng mở rộng, thu hẹp khả năng xử lý của toàn hệ thống tùy theo nhu cầu sử dụng. Ví dụ với ứng dụng tìm khóa luận bị sao chép của Đại học Quốc gia Hà Nội, cả năm sẽ không có nhiều yêu cầu gửi tới hệ thống tuy nhiên trong thời gian ngắn từ tháng Năm đến tháng Sáu sẽ có lượng dữ liệu lớn đổ về từ tất cả các khoa thuộc tất cả các trường thành viên.

Trong tương lai xa hơn, hệ thống cần trang bị thêm chức năng nâng cao như:

- Quản lý số yêu cầu gửi đến từ một ứng dụng để đảm bảo một ứng dụng không sử dụng quá nhiều tài nguyên (CPU, dung lượng lưu trữ, v.v.) ảnh hưởng đến các ứng dụng khác trong hệ thống. Với chức năng này, hệ thống

có thể cung cấp gói dịch vụ miễn phí với giới hạn về số yêu cầu được gửi đến trong 24 giờ để hỗ trợ các lập trình viên muốn làm quen với hệ thống. Việc này tương tự như các hệ thống PaaS hay SaaS hiện nay với mô hình Freemium.

- Phân tích từ khóa theo văn cảnh, sử dụng các thuật toán tagging, học máy để lọc tách thông tin thay vì chỉ dùng thống kê đơn thuần. Việc này có thể cải thiện độ chính xác và tính đúng đắn của hệ thống một cách đáng kể. Với các thông tin cụ thể hơn về mỗi từ khóa, hệ thống có thể trích xuất được các từ khóa liên quan, cùng chỉ đến một nội dung, từ đó đưa ra được những thống kê hữu ích hơn.

1.3.2. Hệ thống tổng hợp tin tức

Đây là hệ thống thực hiện nhiệm vụ trực tiếp lấy các bài báo từ các nguồn khác nhau, chuyển đến cho hệ thống xử lý tiếng Việt, lưu lại kết quả và hiển thị cho người sử dụng. Để thực hiện công việc của mình, hệ thống tổng hợp tin tức sẽ có hai thành phần: Crawler và Front-end.

Chức năng của Crawler:

- Chạy liên tục 24/7.
- Thu thập đường dẫn của tin, bài từ các nguồn khác nhau. Truy xuất đến các đường dẫn đã có và lưu lại nội dung HTML của tin, bài. Trích xuất nội dung tin, bài từ HTML của trang.
- Chuyển nội dung tin, bài cho hệ thống xử lý tiếng Việt. Lưu lại kết quả sau khi xử lý cùng thông tin liên quan đến tin, bài vào cơ sở dữ liệu của hệ thống.

Chức năng của Front-end:

- Trang chủ hiển thị các tin, bài mới nhất, đáng quan tâm nhất, các từ khóa đang là xu hướng của ngày.
- Xem tin, bài cùng các từ khóa liên quan. Xem từ khóa với đồ thị xu hướng thay đổi theo thời gian. Xem tin, bài theo chủ đề, từ khóa.
- Hỗ trợ đăng nhập với tài khoản cá nhân. Tùy biến nội dung trên trang chủ theo chủ đề, từ khóa.

CHƯƠNG 2. CÁC CÔNG CỤ ĐƯỢC SỬ DỤNG

2.1. Giới thiệu về các hệ quản trị cơ sở dữ liệu

Trong suốt quá trình phát triển của công nghệ thông tin, cơ sở dữ liệu luôn là một trong những vấn đề mấu chốt được đặc biệt quan tâm phát triển. Chính vì vậy nên đã hình thành nhiều mô hình cơ sở dữ liệu khác nhau, dẫn tới các hệ quản trị cơ sở dữ liệu khác nhau. Hiện nay, hệ quản trị cơ sở dữ liệu được sử dụng nhiều nhất là các hệ quản trị cơ sở dữ liệu quan hệ (RDBMS, ví dụ như MySQL, Microsoft SQL Server), được sử dụng rộng rãi trong tất cả các lĩnh vực từ tài chính ngân hàng cho tới trò chơi trực tuyến, mạng xã hội. Trong thời gian gần đây, với sự phát triển của kết nối Internet đi kèm theo nó là sự bùng nổ về lượng thông tin được tạo ra, đã xuất hiện một loại cơ sở dữ liệu mới khác hoàn toàn các loại trước đây: NoSQL. Khóa luận sẽ lần lượt trình bày về các hệ thống này từ đó lựa chọn hệ thống phù hợp nhất do bài toán đặt ra.

2.1.1. Hệ quản trị cơ sở dữ liệu quan hệ

Edgar Codd trong thời gian làm việc tại IBM đã viết một số nghiên cứu về các hướng nghiên cứu mới cho việc xây dựng cơ sở dữ liệu chủ yếu do các mô hình hiện thời đều thiếu khả năng tìm kiếm. Trong các nghiên cứu của mình, nổi bật nhất là tài liệu *A Relational Model Of Data for Large Shared Data Banks*⁹, ông đã mô tả một hệ thống mới để lưu trữ và làm việc với các cơ sở dữ liệu lớn. Thay vì lưu các bản ghi dưới kiểu danh sách liên kết (linked list), ý tưởng của Codd là sử dụng một bảng các bản ghi có kích thước cố định. Một hệ thống danh sách liên kết sẽ rất không hiệu quả khi lưu trữ các cơ sở dữ liệu trống với dữ liệu cho một bản ghi nào đó có thể bị trống. Mô hình quan hệ giải quyết vấn đề này bằng cách chia dữ liệu ra nhiều bảng khác nhau, với các thành phần phụ chuyển ra ngoài bảng chính và chỉ tồn chỗ khi cần thiết. Bắt đầu được phát triển từ những năm 1970, đến nay các hệ quản trị cơ sở dữ liệu quan hệ nổi bật và được sử dụng nhiều trong phát triển web có thể kể tới: MySQL (mã nguồn mở), PostgreSQL (mã nguồn mở), Microsoft SQL Server (thương mại).

2.1.1.1. MySQL

MySQL là hệ quản trị cơ sở dữ liệu quan hệ mã nguồn mở phổ biến nhất thế giới. Các dự án phần mềm mã nguồn mở khi cần một hệ quản trị cơ sở dữ liệu đầy đủ thường sử dụng MySQL. Ngoài việc cung cấp miễn phí MySQL Community Edition, một số phiên bản thương mại cũng được cung cấp với nhiều tính năng bổ sung. Các phần mềm sử

dụng MySQL bao gồm: Joomla! (hệ quản trị nội dung, WordPress (blog), phpBB (diễn đàn trao đổi). MySQL cũng là một trong các thành phần chính của gói phần mềm mã nguồn mở LAMP chuyên phát triển ứng dụng web: Linux (hệ điều hành), Apache (phần mềm cung cấp dịch vụ web), MySQL, PHP/Python (ngôn ngữ lập trình). Không giới hạn với những ứng dụng nhỏ, MySQL còn được sử dụng trong những hệ thống lớn, cực lớn của các công ty như Google, Facebook và Twitter. Gần đây Google¹⁰ và Twitter¹¹ đều cung cấp cho cộng đồng lập trình viên nhánh phát triển riêng của MySQL được sử dụng nội bộ với nhiều điểm cải tiến cao cấp.

MySQL được phát triển bằng ngôn ngữ C và C++, có thể sử dụng trên rất nhiều hệ thống khác nhau như các dòng Linux, Windows, Mac OS X. Hầu hết các ngôn ngữ lập trình đều có sẵn thư viện để kết nối đến cơ sở dữ liệu MySQL. MySQL có đầy đủ các chức năng của các hệ quản trị cơ sở dữ liệu quan hệ khác: hỗ trợ SQL, stored procedure, trigger, hỗ trợ nhiều kiểu lưu trữ khác nhau, đáp ứng đầy đủ tiêu chuẩn ACID, partition, clustering, .v.v.

Để sử dụng MySQL hết sức đơn giản. Như đã trình bày, gói phần mềm LAMP bao gồm MySQL và Linux nên ở hầu hết các hệ điều hành trên nền Linux, việc cài đặt MySQL đều có thể thực hiện với một dòng lệnh, thậm chí ở một số hệ điều hành MySQL luôn được cài đặt sẵn. Điều này cũng đúng với Mac OS X. Trên Windows, lập trình viên có thể tải về MySQL Community Edition miễn phí từ trang chủ của MySQL. Việc quản trị MySQL thường được thực hiện thông qua dòng lệnh (console) tuy nhiên cũng có nhiều phần mềm hỗ trợ việc quản trị với giao diện dễ sử dụng, tiêu biểu có thể kể đến phpMyAdmin (ứng dụng web, mã nguồn mở), Navicat (Linux, Windows, Mac OS X, thương mại), Sequel Pro (Mac OS X, miễn phí). Thời gian gần đây cũng đã xuất hiện các giải pháp MySQL as a Service (mô hình SaaS) chạy trên cơ sở hạ tầng điện toán đám mây của các nhà cung cấp như Amazon, Rackspace. Việc sử dụng MySQL trên nền điện toán đám mây tiện dụng hơn do có bên thứ ba quản lý bảo trì, sao lưu dữ liệu cũng như thực hiện mở rộng hệ thống khi cần thiết tuy nhiên chi phí là khá cao: Amazon \$75/tháng, Rackspace \$17/tháng.

Mặc dù được sử dụng rộng rãi nhưng MySQL cũng có những điểm yếu nhất định. Đơn cử là việc hỗ trợ không tốt với cơ sở dữ liệu lớn, đây là lý do vì sao các công ty như Google hoặc Twitter phải phát triển phiên bản MySQL riêng. Một điểm quan trọng nữa là

sử dụng MySQL để tăng dung lượng chủ yếu phải sử dụng phương pháp vertical scaling có nghĩa là nâng cấp máy chủ thêm nhiều RAM, nhiều dung lượng đĩa cứng, thay CPU tốc độ cao hơn. Mà vertical scaling thì sẽ luôn có giới hạn của nó, chúng ta không thể lắp thêm RAM mãi được do giới hạn về phần cứng cũng như phần mềm. Điểm này khiến cho khóa luận khó có thể ứng dụng MySQL một cách hiệu quả vào trong hệ thống xử lý tiếng Việt.

2.1.1.2. PostgreSQL

PostgreSQL là hệ quản trị cơ sở dữ liệu quan hệ mã nguồn mở cao cấp nhất hiện nay với nhiều hệ thống tích hợp, mở rộng, có thể đáp ứng các yêu cầu khác nhau về dữ liệu, khả năng truy xuất.

Tương tự như MySQL, PostgreSQL được phát triển bằng ngôn ngữ C, nhờ vậy mà nó có thể chạy trên hầu hết các kiến trúc máy tính cũng như các hệ điều hành khác nhau. Đa số các ngôn ngữ lập trình hiện nay đều có sẵn thư viện truy xuất vào cơ sở dữ liệu PostgreSQL. Về chức năng, PostgreSQL có rất nhiều chức năng đặc biệt và mạnh mẽ mà ít có hệ quản trị cơ sở dữ liệu nào khác có thể so sánh được. PostgreSQL cũng hỗ trợ nhiều kiểu dữ liệu khác nhau thậm chí lập trình viên có thể định nghĩa các kiểu dữ liệu mới để sử dụng. PostgreSQL đáp ứng đầy đủ tiêu chuẩn ACID. Ngoài các tính năng sẵn có, với từng cơ sở dữ liệu, lập trình viên có thể quyết định cài đặt thêm các thành phần bổ sung gọi là module để được hỗ trợ nhiều hơn nữa trong một tác vụ cụ thể nào đó, ví dụ như PostGIS¹² là một module rất mạnh hỗ trợ các cơ sở dữ liệu cần xử lý dữ liệu địa lý.

Để sử dụng PostgreSQL cũng khá đơn giản. Đa số các hệ điều hành trên nền Linux đều hỗ trợ gói cài đặt PostgreSQL. Trên hệ điều hành Mac OS X từ phiên bản 10.7 cũng được cài đặt sẵn PostgreSQL (song song với MySQL). Trên Windows, lập trình viên có thể tải về bộ cài đặt pgInstaller miễn phí từ trang chủ của PostgreSQL. Quản trị hệ thống PostgreSQL có nhiều điểm tương đồng so với MySQL khi chủ yếu là sử dụng dòng lệnh. Bên cạnh đó người quản trị có thể cài đặt các công cụ khác hỗ trợ thêm qua giao diện web (phpPgAdmin) hoặc giao diện đồ họa sử dụng pgAdmin (mã nguồn mở, hỗ trợ Linux, Windows, Mac OS X). Ngoài việc tự cài đặt và sử dụng, PostgreSQL cũng có thể được cài đặt trên các nền tảng điện toán đám mây ví dụ như heroku¹³. heroku là nền tảng phát triển hết sức mạnh mẽ, ban đầu chỉ hỗ trợ Ruby (ngôn ngữ lập trình) sử dụng PostgreSQL nhưng sau 5 năm phát triển, hiện nay nền tảng này hỗ trợ cả PHP, JavaScript (Node.js),

Python (Clojure) và Java (Scala) với giá thành không quá đắt: miễn phí sử dụng với cơ sở dữ liệu nhỏ hơn 5MB, \$5/tháng với 20GB và nhiều gói cao cấp hơn nữa.

PostgreSQL là một hệ quản trị cơ sở dữ liệu mạnh với nhiều tính năng nổi bật trong đó, các chức năng liên quan đến scaling như replication, clustering cũng rất tốt. Tuy nhiên với mô hình vốn có của một hệ quản trị cơ sở dữ liệu quan hệ, việc cấu hình một hệ thống nhiều máy chủ chạy PostgreSQL là một điều không hề dễ dàng. Một mô hình hoàn toàn khác sẽ phù hợp hơn với phạm vi của khóa luận này.

2.1.2. NoSQL

2.1.2.1. NoSQL là gì?

Có một số điểm quan trọng về các hệ thống NoSQL:

- NoSQL không sử dụng ngôn ngữ truy vấn là SQL. Các hệ thống NoSQL được thiết kế đặc biệt cho những yêu cầu rất khác so với các hệ thống cơ sở dữ liệu quan hệ truyền thống. NoSQL có thể xử lý được một lượng vô cùng lớn dữ liệu mà các hệ thống cũ không thể tiếp cận. Thường các dữ liệu này không có cấu trúc nhất định và do lượng dữ liệu quá lớn nên phải phân chia ra nhiều máy tính vật lý khác nhau nên các phép JOIN là không thể sử dụng được.
- Hệ thống NoSQL có thể không đáp ứng hết các tiêu chuẩn ACID.
- Kiến trúc của hệ thống NoSQL là kiến trúc phân tán, sẵn sàng xử lý khi có tình huống mất mát dữ liệu logic hay các thiết bị phần cứng. Thường thì dữ liệu được chia nhỏ để lưu trữ ở nhiều các máy tính vật lý khác nhau, mỗi phần dữ liệu bản thân nó cũng được nhân làm nhiều bản để lưu ở các vị trí khác nhau. Hệ thống có thể dễ dàng được mở rộng bằng cách kết nối thêm máy tính vào mạng lưới, sự hỏng hóc của một hay nhiều máy tính trong hệ thống không làm ảnh hưởng đến toàn bộ hệ thống.

Như vậy, có thể thấy NoSQL là một hệ thống quản trị cơ sở dữ liệu đã đánh đổi các tính năng mở rộng để có được sự tối ưu trong việc quan trọng nhất của cơ sở dữ liệu: lưu trữ dữ liệu. Chính vì vậy, cơ sở dữ liệu NoSQL rất hữu ích khi làm việc với một lượng lớn dữ liệu không có nhiều mối quan hệ lẫn nhau, hỗ trợ tốt cho các nhu cầu thống kê hoặc phân tích thời gian thực. Đặc điểm này của hệ thống NoSQL khiến nó rất phù hợp

để sử dụng làm hệ quản trị cơ sở dữ liệu chính cho hệ thống xử lý tiếng Việt. Sau đây là một số hệ thống NoSQL tiêu biểu đã được nghiên cứu tìm hiểu trong quá trình làm khóa luận.

2.1.2.2. Apache Hadoop

Apache Hadoop vốn là một nền tảng phần mềm hỗ trợ xử lý dữ liệu phân tán cho các ứng dụng cần xử lý một lượng lớn dữ liệu bằng một mạng lưới nhiều máy tính (dữ liệu lên tới hàng petabyte, số lượng máy tính lên cỡ hàng nghìn). Apache Hadoop bao gồm hai thành phần chính là hệ thống Hadoop Distributed File System và hệ thống MapReduce vốn được phát triển từ tài liệu nghiên cứu của Google về Google File System và Google MapReduce. Như ta đã biết, Google là công ty cung cấp dịch vụ tìm kiếm số một Thế giới, hàng ngày các máy chủ của Google phải xử lý một lượng dữ liệu khổng lồ đồng thời phải truy xuất dữ liệu đó để phục vụ 90 triệu tác vụ tìm kiếm một ngày nên khi các tài liệu nghiên cứu này được công bố, cộng đồng lập trình viên đã rất quan tâm và nhiều cá nhân cũng như tổ chức đã đóng góp trong sự phát triển của dự án Apache Hadoop. Giờ đây hệ thống này không chỉ được sử dụng ở Google mà còn ở nhiều các công ty lớn khác như Yahoo! (với 10000 máy) hay Facebook (với 30 PB dữ liệu) và nhiều công ty khác nữa.

Điểm đặc biệt nhất trong Apache Hadoop là hệ thống MapReduce với ý tưởng phân chia một tác vụ ra làm hai giai đoạn: giai đoạn Map và giai đoạn Reduce. Trong đó giai đoạn Map có thể được thực hiện ngay tại các máy tính lưu trữ dữ liệu. Giai đoạn Map sẽ xử lý dữ liệu và lấy ra kết quả dưới dạng (key, value). Việc này đem lại hai ưu điểm lớn:

- Tận dụng được khả năng xử lý của nhiều máy tính (nhiều nhân CPU) để thực hiện cùng một công việc. Thay vì chỉ có một máy tính (với 4 nhân CPU) thực hiện việc xử lý thì Hadoop cho phép phân tán việc xử lý này ra hàng trăm máy tính (với hàng nghìn nhân CPU), việc này giúp tăng tốc độ xử lý lên gấp nhiều lần.
- Giảm gánh nặng băng thông lên đường truyền của hệ thống. Theo mô hình cũ, dữ liệu phải được chuyển từ cơ sở dữ liệu về máy tính xử lý tuy nhiên với việc xử lý ngay tại nơi lưu trữ, đường truyền sẽ không bị chiếm dụng cho các dữ liệu không cần thiết, vừa tiết kiệm tài nguyên hệ thống, vừa tiết kiệm thời gian chạy tác vụ.

Sau khi giai đoạn Map được thực thi xong, kết quả này được sử dụng để chạy tiếp giai đoạn Reduce, các kết quả của Map có cùng key sẽ được xử lý cùng nhau để tổng hợp lại thành kết quả cuối cùng của key đó. Sau cùng, một tập hợp các cặp (key, value) sẽ là kết quả cuối cùng của tác vụ MapReduce.

Xét ví dụ tác vụ MapReduce đơn giản như sau:

- Có các tin bài tiếng Việt.
- Giai đoạn Map: Đếm các từ trong một bài và trả lại kết quả dưới dạng (từ, số lần xuất hiện).
- Giai đoạn Reduce: Tính tổng số lần xuất hiện của một từ và trả lại kết quả dưới dạng (từ, tổng số lần xuất hiện).

Như vậy, với một tập các tin bài, ta sẽ thu được danh sách các từ xuất hiện và tần suất của chúng. Xét hai câu kí tự “tôi yêu Việt Nam” và “đường sắt Bắc Nam”:

- Các kết quả trong giai đoạn Map:
 - (tôi, 1)
 - (yêu, 1)
 - (việt, 1)
 - (nam, 1)
 - (đường, 1)
 - (sắt, 1)
 - (bắc, 1)
 - (nam, 1)
- Trong các kết quả này có hai kết quả có trùng key là “nam” vì vậy giai đoạn Reduce sẽ thực hiện trên hai kết quả này và trả lại (nam, 2).
- Cuối cùng, ta có danh sách các từ xuất hiện trong hai câu kí tự ban đầu là: tôi 1, yêu 1, việt 1, đường 1, sắt 1, bắc 1, nam 2.

Như đã trình bày ở trên, Apache Hadoop không thực sự được coi là một hệ quản trị cơ sở dữ liệu nhưng ý tưởng về lưu trữ dữ liệu phân tán và xử lý phân tán thông qua tác

vụ MapReduce từ khi xuất hiện đã tạo được nhiều bước tiến lớn trong việc phát triển các hệ thống xử lý dữ liệu lớn (Big Data) nói chung và các hệ thống quản trị cơ sở dữ liệu nói riêng. Hầu hết các hệ thống NoSQL hiện nay đều hỗ trợ tác vụ MapReduce.

2.1.2.2. MongoDB

MongoDB là một hệ quản trị cơ sở dữ liệu NoSQL hướng văn bản, có nghĩa là thay vì lưu trữ dữ liệu vào các bảng theo cách truyền thống, MongoDB có khái niệm tập (collection), mỗi tập chứa nhiều văn bản (document) được lưu trữ dưới dạng BSON (một dạng JSON, JavaScript Object Notation). MongoDB có quá trình phát triển có phần khác biệt so với các phần mềm nguồn mở khác khi nó đã và vẫn đang được phát triển bởi một công ty thương mại (10gen), ngoài phiên bản miễn phí, 10gen cũng cung cấp các phiên bản thương mại cho MongoDB.

MongoDB được phát triển bằng ngôn ngữ C++, hỗ trợ bản cài đặt chạy được trên Linux, Windows, Mac OS X và Solaris. Về chức năng, MongoDB không đáp ứng đủ các tiêu chuẩn ACID nhưng bù lại, hỗ trợ mạnh mẽ cho việc horizontally scaling bằng sharding nên hệ thống MongoDB có thể chạy trên nhiều máy tính, tự động cân bằng tải (load balancing), tự động nhân bản dữ liệu để đảm bảo hệ thống vẫn chạy tốt trong trường hợp có lỗi phần cứng. Đặc biệt, MongoDB hỗ trợ sử dụng mô hình MapReduce để xử lý dữ liệu sử dụng ngôn ngữ JavaScript.

Để sử dụng MongoDB, lập trình viên phải tải về gói cài đặt trên trang chủ. Việc quản trị hệ thống thực hiện qua dòng lệnh. Là một hệ thống mới nên các thư viện cho MongoDB không được nhiều bằng các hệ thống khác, gen10 hỗ trợ chính thức thư viện cho các ngôn ngữ sau: C, C++, Erlang, Haskell, Java, JavaScript, .NET, Node.js, Perl, PHP, Python, Ruby, Scala. Ngoài ra cộng đồng lập trình viên sử dụng MongoDB cũng có nhiều dự án thiết kế thư viện ở các ngôn ngữ khác, chi tiết có thể xem trên trang chủ của MongoDB.

Là một hệ thống tốt nhưng MongoDB liên tục là tâm điểm của cộng đồng lập trình viên khi có một số công ty dạng khởi nghiệp (startup) viết những bài viết nói về việc chuyển từ MongoDB sang một hệ thống khác do các điểm yếu như:

- MongoDB sử dụng khóa toàn cục, khi đang ghi dữ liệu mới hoặc cập nhật tài liệu thì MongoDB sử dụng một khóa toàn cục trên toàn hệ thống. Điều này có thể khiến việc đọc dữ liệu ở tập B phải chờ trong khi đang cập nhật dữ liệu ở

tập A. Mặc dù vậy, việc sử dụng khóa này được thực hiện rất nhanh ở trong bộ nhớ RAM và các phiên bản mới gần đây như 2.0 và trong tương lai là 2.2 sẽ tiếp tục có những cải tiến trong vấn đề này.

- Tính năng sharding không thực sự ổn định. Vấn đề này tồn tại trong bản 1.6 nhưng đã được gen10 thông báo khắc phục thành công ở 1.8 và cải thiện hơn nữa ở 2.0 với nhiều tùy chọn để kiểm tra dữ liệu đã được lưu đầy đủ và chính xác.

Với hỗ trợ tốt cho việc phát triển hệ thống và MapReduce, MongoDB rất phù hợp để làm thành phần lưu trữ dữ liệu chính cho hệ thống xử lý tiếng Việt.

2.1.2.3. Redis

Redis là một hệ thống lưu trữ dữ liệu dưới dạng (key, value) mã nguồn mở. Redis được phát triển bằng ngôn ngữ C, hỗ trợ hầu hết các kiến trúc máy tính và hệ điều hành tuy nhiên trên trang chủ, Redis không được cung cấp dưới dạng bộ cài đặt mà lập trình viên muốn sử dụng phải tải về mã nguồn để tự biên dịch. Với các hệ thống nền Linux hay Mac OS X, việc này không quá phức tạp. Thư viện Redis được hỗ trợ ở hầu hết các ngôn ngữ lập trình thông dụng hiện nay.

Redis được thiết kế như một từ điển với các key được trỏ đến một giá trị nhất định. Các kiểu dữ liệu của Redis bao gồm: Giá trị đơn (xâu kí tự, số), Danh sách, Tập hợp, Tập hợp có sắp xếp, Mảng băm. Về cơ bản, Redis lưu toàn bộ dữ liệu của nó ở trong RAM và ghi xuống đĩa cứng thường xuyên để đảm bảo tính toàn vẹn của dữ liệu. Việc này vừa là điểm mạnh vừa là điểm yếu của hệ thống Redis.

Với việc toàn bộ dữ liệu được lưu trong RAM, các tác vụ thực hiện rất nhanh, chủ yếu với thời gian $O(1)$. Tuy nhiên việc này khiến cho Redis không thể sử dụng để chứa một lượng dữ liệu lớn hơn RAM của máy tính, vấn đề này có thể khắc phục bằng cách nâng dung lượng RAM nhưng cũng chỉ đến một giới hạn mà thôi. Người quản trị hệ thống có thể cấu hình nhiều máy tính chạy Redis nhưng với thiết lập nhân bản (replication) thì việc này cũng không tăng dung lượng của hệ thống.

Như vậy ta có thể thấy Redis đã đánh đổi khả năng lưu trữ để có hiệu năng cao, cung cấp tốc độ truy xuất dữ liệu lớn thay vì lưu được nhiều dữ liệu. Đây là lý do giúp

cho Redis trở thành công cụ tốt cho việc tổng hợp, thống kê với lượng lớn dữ liệu hoặc có yêu cầu xử lý thời gian thực.

2.2. Giới thiệu về RESTful web API

REST đang ngày càng được sử dụng nhiều hơn trên mạng Internet thay thế cho các chuẩn cũ hơn như SOAP hay WSDL vì tính đơn giản của nó. Minh chứng cho việc này là việc sử dụng RESTful web API của các nhà cung cấp dịch vụ lớn như Facebook, Google, Yahoo!.

2.2.1. Cơ bản

REST đưa ra một số quan điểm về cấu trúc, dựa vào đó để thiết kế các dịch vụ web cung cấp các tài nguyên của hệ thống, bao gồm việc trình bày các trạng thái của tài nguyên như thế nào hay truyền tải như thế nào qua giao thức HTTP cho các ứng dụng khác nhau chạy trên các nền tảng và sử dụng các ngôn ngữ lập trình khác nhau. Khi được giới thiệu bởi Roy Fielding vào năm 2000, REST không thu hút được nhiều sự chú ý nhưng hiện nay, các hệ thống hỗ trợ REST đã xuất hiện nhiều và vẫn đang tiếp tục được phát triển.

Có bốn quan điểm chính cần lưu ý trong dịch vụ web REST:

- Sử dụng các phương thức của giao thức HTTP (GET, POST, PUT, DELETE).
- Không lưu lại trạng thái.
- Sử dụng các URI có cấu trúc giống các thư mục.
- Hỗ trợ XML, JSON hoặc cả hai.

2.2.2. Sử dụng các phương thức của giao thức HTTP

Một trong những đặc điểm quan trọng của một dịch vụ REST là việc sử dụng các phương thức của giao thức HTTP đúng như cách nó được định nghĩa trong RFC 2616. Ví dụ như phương thức GET là phương thức mà ứng dụng cần sử dụng để lấy dữ liệu về một đối tượng nào đó từ dịch vụ web. Cụ thể hơn, REST yêu cầu người thiết kế phải sử dụng các phương thức như sau:

- POST: Tạo một tài nguyên trên hệ thống.
- GET: Để lấy một tài nguyên trên hệ thống.

- PUT: Để cập nhật một tài nguyên trên hệ thống.
- DELETE: Để xóa một tài nguyên trên hệ thống.

Trước đây, nhiều dịch vụ web được thiết kế sử dụng các phương thức này không đúng mục đích, đặc biệt là phương thức GET. Hầu hết các tác vụ thực thi đều có thể được yêu cầu bằng phương thức GET thông qua các tham số ví dụ như:

```
GET /adduser?name=A HTTP/1.1
```

Thực sự thì việc này hoàn toàn không gây khó khăn gì về mặt kỹ thuật nhưng về ngữ nghĩa thì không đúng. Yêu cầu GET ở trên thay vì chỉ lấy về thông tin thì lại làm thay đổi hệ thống theo một cách nào đó, cụ thể ở đây là tạo thêm một người dùng mới có tên là A. Ngoài vấn đề về ngữ nghĩa, việc cho phép các yêu cầu GET thay đổi thông tin trên hệ thống có thể dẫn tới việc thay đổi không dự đoán trước được khi các công cụ crawler vô tình gửi các yêu cầu tương tự tới hệ thống. Để giải quyết vấn đề này, đơn giản là sử dụng một phương thức khác phù hợp hơn cho việc tạo người dùng mới, POST:

```
POST /users HTTP/1.1
```

```
Content-Type: application/javascript
```

```
{
    name: "A"
}
```

Với thiết kế mới này, hệ thống đã sử dụng đúng phương thức POST của HTTP. Khi dịch vụ web nhận được yêu cầu POST này thì một người dùng mới có thể được tạo ra, sau đó người dùng này nên là một tài nguyên con của `/users`. Như vậy, để lấy được thông tin về người dùng mới này, ứng dụng có thể gửi một yêu cầu GET như sau:

```
GET /users/A HTTP/1.1
```

Tương tự như vậy, trong các thiết kế dịch vụ web trước đây, các yêu cầu GET cũng được chấp nhận để cập nhật một tài nguyên nào đó, ví dụ:

```
GET /updateuser?name=A&newname=B HTTP/1.1
```


Một cách làm tốt hơn cho việc cập nhật này theo các quy định của REST đó là sử dụng phương thức PUT. Như vậy, để cập nhật đối tượng tính `name` cho A, có thể gửi yêu cầu PUT như sau:

```
PUT /users/A HTTP/1.1

Content-Type: application/javascript

{
    name: "B"
}
```

Ta có thể thấy yêu cầu PUT này trỏ thẳng đến tài nguyên `/users/A` chính là người dùng mà nó muốn cập nhật thông tin, sau đó thông tin mới được gửi lên tương tự như khi tạo một người dùng mới ở đây là giá trị mới cho thuộc tính `name`.

Theo thiết kế như trên, các URI của một dịch vụ REST thường chỉ sử dụng các danh từ (`/users`, `/documents`, `/words`) chứ không chứa các động từ (dạng `/adduser` hay `/updateuser`). Các động từ/phương thức GET, POST, PUT, DELETE đều đã được quy chuẩn trong giao thức HTTP. Một số ví dụ về URI của các dịch vụ web lớn hiện nay:

- Facebook
 - Để lấy thông tin về một người sử dụng có thể gửi yêu cầu GET như sau: GET `https://graph.facebook.com/sondh` HTTP/1.1
 - Để lấy ảnh của một người sử dụng có thể gửi yêu cầu GET như sau: GET `https://graph.facebook.com/sondh/picture` HTTP/1.1
 - Để đăng bài lên trang cá nhân của một người sử dụng có thể gửi yêu cầu POST như sau: POST `https://graph.facebook.com/sondh/feed` HTTP/1.1 `message=Hello+world!`
- Twitter

- Để lấy về các tweet của một người sử dụng, có thể gửi yêu cầu GET như sau: GET
`https://userstream.twitter.com/2/mrpaint.json`
HTTP/1.1
- Để đăng tweet có thể gửi yêu cầu POST như sau: POST
`https://api.twitter.com/1/statuses/update.json`
HTTP/1.1 status=Hello+world!

2.2.3. Không lưu trạng thái

Các dịch vụ web cần mở rộng dung lượng thường xuyên để đáp ứng nhu cầu của các ứng dụng. Các hệ thống máy tính với chức năng cân bằng tải cũng như khả năng hoạt động bình thường trong trường hợp có máy tính bị lỗi trong hệ thống sẽ phải được thiết kế sao cho yêu cầu từ ứng dụng gửi đến có thể được phục vụ nhanh nhất, hiệu quả nhất bởi bất cứ máy tính nào đang rỗi. Chính vì vậy, các kết quả trả về cho các yêu cầu gửi đến phải là những kết quả toàn vẹn, độc lập với nhau. Có như vậy thì hệ thống dẫn đường trung gian mới có thể chuyển tiếp, dẫn đường không lo ảnh hưởng đến kết quả trả về.

Với thiết kế không lưu trạng thái, máy tính của hệ thống dịch vụ web khi xử lý yêu cầu sẽ không cần phải truy xuất một biến trạng thái nào trong hệ thống mà toàn bộ các thông số cần thiết đều có trong yêu cầu gửi đến. Cách xử lý như thế này nâng cao hiệu suất cho dịch vụ web đồng thời cũng đơn giản hóa thiết kế đi nhiều lần, tránh việc đồng bộ hóa dữ liệu giữa các tiến trình trong một máy tính hay giữa các máy tính trong hệ thống vốn vô cùng phức tạp.

Trong các dịch vụ web cũ, thường thấy gửi đến các yêu cầu GET tương tự như sau:

```
GET /posts/getNextPage HTTP/1.1
```

Với yêu cầu như thế này, dịch vụ web sẽ phải xử lý để xác định xem ứng dụng nào đang yêu cầu danh sách các bài viết. Sau đó kiểm tra xem đã yêu cầu trang nào rồi để phục vụ trang tiếp theo. Trong trường hợp hệ thống dịch vụ web chạy trên nhiều máy tính, các thông tin như vậy bắt buộc phải lưu trong cơ sở dữ liệu để tất cả các máy tính có thể truy xuất được, việc này đã tạo thêm gánh nặng xử lý cũng như tiêu tốn nhiều tài nguyên hệ thống. Thay vào đó, trong một dịch vụ REST, việc yêu cầu các trang bài viết phải được thực hiện như sau

GET /posts?page=1 HTTP/1.1

GET /posts?page=2 HTTP/1.1

...

Về tổng số lượng trang của bài viết có thể được trả lại trong kết quả của mỗi yêu cầu này để ứng dụng quyết định xem có cần thiết phải yêu cầu trang nữa hay không. Với thiết kế mới này, tài nguyên phía dịch vụ web tiết kiệm được đáng kể và có thể sử dụng để phục vụ nhiều yêu cầu hơn.

2.2.4. Sử dụng các URI có cấu trúc giống các thư mục

Từ phía ứng dụng sử dụng dịch vụ web, các URI là thành phần đầu tiên mà nó quan tâm vì vậy các URI này phải được thiết kế sao cho ứng dụng có thể phán đoán dễ dàng URI cho một tài nguyên nào đó. Để giúp ứng dụng dễ hiểu và dễ nhớ các URI trong dịch vụ web, cách thực hiện là sử dụng các URI có cấu trúc giống các thư mục có nghĩa là URI không phải là một chuỗi kí tự với những kí tự gạch chéo mà nó phải được hiểu như một cây tài nguyên với mối quan hệ phù hợp giữa tài nguyên cha và con.

Như đã trình bày ở trước, URI có thể được xác định thông qua một thuộc tính duy nhất nào đó, ví dụ như thuộc tính name cho người sử dụng. Như vậy URI đến người sử dụng có tên là A sẽ là:

`http://api.koluto.com/users/A`

Ngoài ra có thể hỗ trợ các URI phân chia theo chủ đề bài viết, hoặc theo ngày tháng ví dụ:

`http://api.koluto.com/documents/{năm}/{tháng}/{ngày}`

Cũng có một số điểm lưu ý khác khi thiết kế các URI cho dịch vụ web, đó là:

- Nên ẩn các thông tin về mã cài đặt của hệ thống ví dụ như phần mở rộng của các file mã nguồn (.php, .asp) để có thể thay đổi công nghệ phát triển hệ thống mà không phải thay đổi URI tới các tài nguyên.
- Sử dụng toàn bộ bằng các chữ tiếng Anh, chữ thường (không dùng chữ in hoa).
- Không sử dụng kí tự trống (space), thay vào đó sử dụng dấu gạch ngang hoặc gạch dưới.

- Hạn chế sử dụng các tham số ngoài (phần phía sau dấu hỏi chấm trong yêu cầu GET).
- Nếu có yêu cầu gửi đến URI chưa đầy đủ, thay vì thông báo 404 Not Found, nên có hướng dẫn làm thế nào để có URI đầy đủ. Ví dụ nếu lập trình viên mới gửi yêu cầu GET `/users HTTP/1.1` thì nên in ra dòng hướng dẫn phải truy xuất đến `/users/{name}`.

Về URI, điểm quan trọng nhất là URI cho một tài nguyên **không được** thay đổi dù hệ thống cài đặt thay đổi hoặc nâng cấp hay bất cứ lý do gì khác.

2.2.5. Hỗ trợ XML, JSON hoặc cả hai

Kết quả trả lại của dịch vụ web trình bày tài nguyên dưới một dạng nào đó tại thời điểm yêu cầu. Chính vì vậy việc trình bày làm sao cho đơn giản, dễ đọc, dễ sử dụng là vấn đề cần được quan tâm.

Trước đây SOAP là giao thức được các dịch vụ web sử dụng nhiều với các tính năng mạnh mẽ, hỗ trợ nhiều mô hình dịch vụ khác nhau. Chính vì vậy, để thiết lập SOAP với bất cứ ứng dụng nào cũng là một công việc phức tạp và dễ dàng gặp phải lỗi không tương thích giữa ứng dụng và dịch vụ web. Hiện nay với sự xuất hiện của XML và JSON, việc trao đổi giữa ứng dụng và dịch vụ web đã trở nên dễ dàng hơn nhiều lần. Đặc biệt với JSON, lập trình viên có thể dễ dàng trực tiếp đọc kết quả trả về để kiểm tra tính đúng đắn của ứng dụng/dịch vụ web.

Chính vì vậy, dịch vụ web nên hỗ trợ XML hoặc JSON hoặc cả hai trong việc trình bày các tài nguyên của mình. Trong trường hợp hỗ trợ nhiều cách trình bày khác nhau, dịch vụ web có thể dựa vào HTTP header “Accept” để xác định ứng dụng có thể hiểu cách trình bày nào và trả lại kết quả tương ứng. Việc này hỗ trợ các lập trình viên làm việc trên nhiều nền tảng khác nhau dễ dàng tiếp cận với dịch vụ web bằng công cụ mà nền tảng đó có sẵn.

2.3. Giới thiệu về crawler/web crawler

Theo Wikipedia¹⁴, crawler là một chương trình tự động truy cập đến các trang web trên mạng Internet để lấy dữ liệu. Các crawler chủ yếu được các hệ thống tìm kiếm (search engine) sử dụng để đánh chỉ mục các trang web nhằm cung cấp cho người tìm kiếm các kết quả tốt nhất, cập nhật nhất. Crawler hoạt động theo nhiều cơ chế khác nhau

tùy theo nhu cầu sử dụng nhưng thường thì sẽ có một số địa chỉ web ban đầu, crawler tải về các trang web tại các địa chỉ này, dựa trên các siêu liên kết mà nó tìm thấy, crawler cập nhật danh sách các địa chỉ quan tâm và tiếp tục lặp lại quá trình.

Khi hoạt động, crawler cũng có một số vấn đề như sau:

- Cùng một nội dung có thể có nhiều đường dẫn để truy xuất nó. Có thể là do lỗi vô tình của người phát triển trang web hoặc có thể do trang web hỗ trợ nhiều cách trình bày cho một nội dung (xem đầy đủ, xem trên di động, xem bản để in,.v.v.). Với ví dụ một nội dung có 3 phiên bản khác nhau, nếu crawler cần xử lý 3500 nội dung thì số lượng phải xử lý đã vượt quá 10000 đường dẫn. Đó là chưa kể đến việc phải cập nhật lại nội dung của các đường dẫn đã xử lý để đảm bảo kết quả lưu luôn cập nhật. Chính vì vậy lượng công việc của crawler sẽ cộng dồn và ngày một nhiều nếu không có cơ chế lựa chọn, ưu tiên thích hợp.
- Các nội dung trên mạng Internet hiện hầu hết ở dưới dạng chữ, trình bày trong một văn bản siêu đánh dấu (HTML) tuy nhiên nội dung chính của trang luôn đặt ở giữa một loại các đoạn mã hỗ trợ khác, có thể là đoạn mã hiển thị trên nhiều trình duyệt khác nhau, có thể là các đoạn mã hiển thị ảnh, phim, nội dung đa phương tiện nhằm mục đích quảng cáo,.v.v. Crawler sẽ phải xử lý để lọc bỏ thông tin thừa, giữ lại nội dung chính của trang. Chưa kể trong các trang web hiện đại, nội dung thường được xử lý JavaScript trước khi đưa vào trang, hoặc có thể trình duyệt phải thực hiện các tác vụ AJAX để cập nhật nội dung trang, như vậy trong đoạn HTML mà crawler tải về được hầu như không có nội dung nào có giá trị. Gần đây nhiều người nhận thấy crawler của Google (công cụ tìm kiếm lớn nhất hiện nay) ngoài việc tải về HTML còn chạy cả các đoạn mã JavaScript nhưng trong đó, đây quả là một bước tiến lớn của Google mà không phải tất cả các crawler đều làm được.

2.4. Giới thiệu về các ngôn ngữ lập trình web

2.4.1. PHP

PHP là một ngôn ngữ lập trình mã nguồn mở phổ biến, chủ yếu được sử dụng để lập trình web và là một thành phần trong gói phần mềm LAMP đã đề cập trong khóa luận.

Hầu hết các môi trường đều hỗ trợ sử dụng PHP rất mạnh: trên các hệ điều hành dòng Linux, PHP thường được đóng gói sẵn cùng Apache HTTP Server, Mac OS X cũng vậy. Trên Windows, gói cài đặt Apache HTTP Server và PHP có thể được tải về từ trang chủ hoặc sử dụng các bộ cài đặt all-in-one như XAMPP.

Được bắt đầu phát triển từ năm 1995 với sự phát triển không ngừng, PHP nhận được nhiều sự đóng góp của cộng đồng lập trình viên, từ đó có nhiều hệ thống nền được viết để hỗ trợ xây dựng các trang web sử dụng PHP. Trong số các hệ thống này, chưa có nhiều hệ thống dành sự quan tâm tới việc phát triển một hệ thống API nên không có nhiều lựa chọn tuy nhiên nổi bật nhất là hệ thống nền CodeIgniter với bản viết riêng hỗ trợ cho việc xây dựng dịch vụ web codeigniter-restserver¹⁵.

2.4.2. Ruby

Ruby là một ngôn ngữ lập trình không mới (xuất hiện cùng thời gian với PHP) tuy nhiên tới 5 năm gần đây mới bắt đầu có sự phát triển mạnh mẽ, chủ yếu nhờ sự phát triển của hệ thống nền Ruby on Rails được phát triển bởi một bên thứ ba. Ý tưởng khi thiết kế Ruby tập trung vào lập trình viên, làm sao để công việc lập trình trở nên thú vị và hiệu quả. Chính vì vậy các đoạn mã Ruby khi được đưa ra rất dễ hiểu cho dù người đọc không biết tới ngôn ngữ Ruby từ trước hoặc thậm chí chưa từng lập trình. Với ý tưởng thiết kế này, Ruby đưa ra nhiều khái niệm mới, ít thấy ở các ngôn ngữ lập trình phổ biến khác. Để sử dụng Ruby, lập trình viên chủ yếu tự biên dịch với mã nguồn được cung cấp, Ruby chạy được ở hầu hết các hệ thống hiện có trên Thế giới. Trên các hệ thống thông dụng dòng Linux, Mac OS X hoặc Windows cũng có sẵn các gói cài đặt dành cho các lập trình viên không muốn mất thời gian biên dịch lại từ đầu. Từ phiên bản Mac OS X 10.5, Ruby còn được cài đặt sẵn trong hệ thống Mac OS X.

Như đã trình bày, Ruby trở nên phổ biến phần nhiều dựa vào Ruby on Rails. Đây là một hệ thống nền hỗ trợ đa dạng, rất nhiều tính năng cho các trang web. Có thể nói Ruby on Rails là một trong những hệ thống nền đầu tiên đưa được khái niệm ORM (object relational mapping) vào sử dụng với hệ thống Active Record. Sau Ruby on Rails, một loạt các hệ thống nền khác cũng lần lượt cài đặt chức năng tương tự này.

Tuy Ruby on Rails hỗ trợ nhiều chức năng nhưng hầu hết đều không sử dụng nhiều trong một hệ thống API như hệ thống xử lý tiếng Việt với các đặc điểm về dòng xử lý không quá phức tạp, không có tương tác nhiều với người dùng, không sử dụng

HTML, v.v. Ruby có một hệ thống nền khác phù hợp hơn đó là Sinatra. Sinatra là một hệ thống nền mã nguồn mở nhỏ gọn không có nhiều chức năng như Ruby on Rails, nó cũng không tuân thủ mô hình MVC (model-view-controller). Một ứng dụng web sử dụng Sinatra có thể chỉ có vài dòng như sau:

```
#!/usr/bin/env ruby

require 'sinatra'

get '/' do

  redirect to('/hello/World')

end

get '/hello/:name' do

  "Hello #{params[:name]}!"

end
```

Khi chạy ứng dụng này ở máy localhost, cổng 80. Truy cập vào địa chỉ <http://localhost/> sẽ được tự động chuyển tới <http://localhost/hello/World> và lúc đó sẽ hiển thị một dòng “Hello World”. Rất đơn giản và hiệu quả. Có thể thấy một hệ thống như thế này phù hợp để phát triển RESTful web API.

2.4.3. Python

Python có thể mạnh trong việc xử lý dữ liệu text đồng thời có sẵn một số thư viện riêng chuyên xử lý HTML dành cho Python. Về khả năng này, Perl cũng là một ngôn ngữ tương tự tuy nhiên Perl là một ngôn ngữ khá cũ và cú pháp không thân thiện như Python.

Python là một trong những ngôn ngữ lập trình phổ biến nhất thế giới (đứng thứ 8 trong danh sách xếp hạng của TIOBE trong tháng Năm năm 2012¹⁶). Hầu hết các hệ điều hành dòng Linux và Mac OS X được cài đặt sẵn Python. Các phiên bản của hệ điều hành Windows có thể dễ dàng tải và cài đặt từ trang chủ của Python. Python có rất nhiều thư viện hỗ trợ đầy đủ các yêu cầu khác nhau của lập trình viên. Nhiều thư viện của Python

được đánh giá cao và được lấy làm ý tưởng để xây dựng các thư viện tương tự ở các ngôn ngữ khác. Thư viện của Python đa dạng, từ tính toán phức tạp (SciPy, NumPy) cho đến xử lý ngôn ngữ tự nhiên (NLTK) hoặc thậm chí là hỗ trợ xây dựng ứng dụng với giao diện đồ họa đầy đủ (PyGTK, hỗ trợ đa nền tảng). Python được sử dụng ở nhiều công ty, tổ chức lớn trên Thế giới như Google (Google Crawler được viết một phần bằng Python), Yahoo!, NASA, v.v. Về hiệu năng, Python sử dụng mô hình máy ảo tương tự Java, mã nguồn (.py) được biên dịch thành dạng bytecode (.pyc) trước khi được chạy trong máy ảo, điều này giúp mã Python có được hiệu năng cao.

2.4.4. Node.js

Node.js không phải là một ngôn ngữ lập trình như PHP hay Ruby. Node.js là một hệ thống phần mềm hỗ trợ chạy các chương trình viết bằng ngôn ngữ JavaScript với sự tập trung cho các chương trình chạy trên nền Internet, dịch vụ web. Nhân tố chính và quan trọng nhất của Node.js là bộ xử lý JavaScript V8 mã nguồn mở của Google (đây chính là bộ xử lý trong trình duyệt Chromium và Google Chrome) với hiệu suất rất cao giúp mã JavaScript có thể chạy ngang ngửa về tốc độ so với mã C/C++. Node.js là một hệ thống còn non trẻ so với lịch sử phát triển lâu dài của PHP hay Ruby (bắt đầu phát triển vào năm 2009) nhưng đã có những bước nhảy vọt trong thiết kế để phù hợp với các nhu cầu hiện đại. Điều này khiến sự quan tâm của cộng đồng lập trình viên dành cho Node.js càng ngày càng lớn. Đặc biệt, Node.js đã được bầu chọn bởi tạp chí InfoWorld là “Công nghệ của năm 2012”.

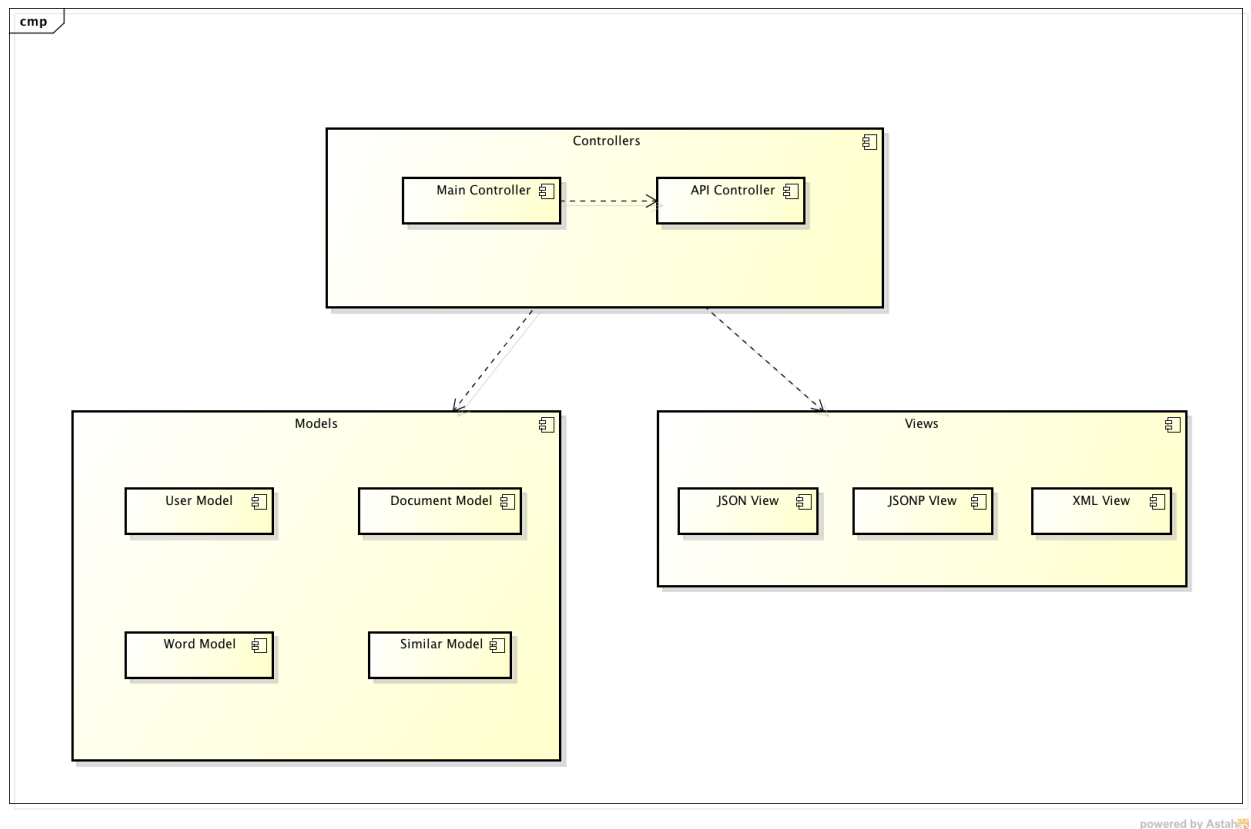
Với thiết kế hướng sự kiện sử dụng các hàm gọi lại (callback) và giao tiếp vào ra không đồng bộ (non-blocking IO), Node.js đem đến một hệ thống có hiệu năng cao. Cùng một lúc phục vụ được nhiều người sử dụng, đặc biệt với các hệ thống có hiện tượng nghẽn cổ chai (bottleneck) ở cơ sở dữ liệu. Tuy vậy, Node.js cũng có điểm hạn chế là không hỗ trợ multi-thread, vấn đề này có thể được giải quyết bằng cách chạy nhiều tiến trình Node.js rồi đặt sau một hệ thống cân bằng tải. Với mô hình như vậy, lập trình viên cũng có thể dễ dàng mở rộng dung lượng của hệ thống bằng cách cắm thêm máy tính vào mạng và tiếp tục cân bằng tải giữa các tiến trình trên các máy tính khác nhau.

CHƯƠNG 3. PHÂN TÍCH THIẾT KẾ HỆ THỐNG

3.1. Hệ thống xử lý tiếng Việt

3.1.1. Cấu trúc hệ thống

Hệ thống xử lý tiếng Việt được thiết kế theo 3 lớp theo mô hình MVC (Model View Controller):



3.1.1.1. Controllers

Hệ thống xử lý tiếng Việt có 2 controllers. Trong đó Main Controller đóng vai trò chủ chốt trong việc xử lý các yêu cầu gửi đến. API Controller được đưa vào để hỗ trợ trong việc xử lý xác minh người dùng đồng thời xác định view cần sử dụng để trình bày kết quả.

3.1.1.2. Models

Hiện tại, hệ thống có 4 models:

1. User Model: xử lý các tác vụ liên quan đến người sử dụng (người quản trị của các ứng dụng). Model này có các phương thức là:
 - `getUserByUserName`: tìm người dùng với tên đăng nhập được chỉ định. Input: `userName`.
 - `insertUser`: thêm một người dùng mới vào hệ thống. Input: `newUser` trong đó `newUser` chứa các thông tin về người dùng mới: `username`, `password`, `email`.
2. Document Model: xử lý các tác vụ liên quan đến tài liệu được lưu trữ trong hệ thống. Model này có các phương thức là:
 - `getDocuments`: tìm các tài liệu thuộc một ứng dụng được yêu cầu. Input: `appId`.
 - `getDocument`: tìm một tài liệu được chỉ định. Input: `appId`, `documentId`.
 - `insertDocument`: thêm một tài liệu mới vào hệ thống. Input: `newDocument` trong đó `newDocument` chứa các thông tin về tài liệu mới: `appId`, `text`, `sections`.
 - `deleteDocument`: xóa tài liệu được chỉ định khỏi hệ thống. Input: `document` trong đó `document` chứa `_id` để xác định tài liệu trong hệ thống.
 - `parseText`: xử lý văn bản để lấy các từ khóa. Input: `text`. Output: danh sách các từ khóa.
 - `countTokens`: đếm trong danh sách các từ khóa để tổng hợp số lượng của mỗi từ khóa. Input: danh sách các từ khóa `tokens`. Output: mảng chứa các từ khóa và số lần xuất hiện.
3. Word Model: xử lý các tác vụ liên quan đến từ khóa. Model này có các phương thức là:

- `getAppWords`: tìm các từ khóa phổ biến nhất của một ứng dụng. Input: `appId`.
- `getAppWord`: tìm một từ khóa được chỉ định trong một ứng dụng. Input: `appId`, `word`.
- `incrWord`: tăng bộ đếm cho từ khóa trong một ứng dụng. Input: `appId`, `word`, `sections`, `count`. Trong đó `sections` là các mục mà tài liệu chứa từ khóa được xếp vào, `count` là số lần xuất hiện của từ khóa trong tài liệu đó.
- `getAppSectionWords`: tìm các từ khóa phổ biến nhất trong một mục của một ứng dụng. Input: `appId`, `section`.

4. `Similar Model`: có một phương thức duy nhất là `findSimilarDocuments` tìm trong hệ thống các tài liệu tương tự với đoạn văn bản được chỉ định. Input: `appId`, `text`.

3.1.1.3 Views

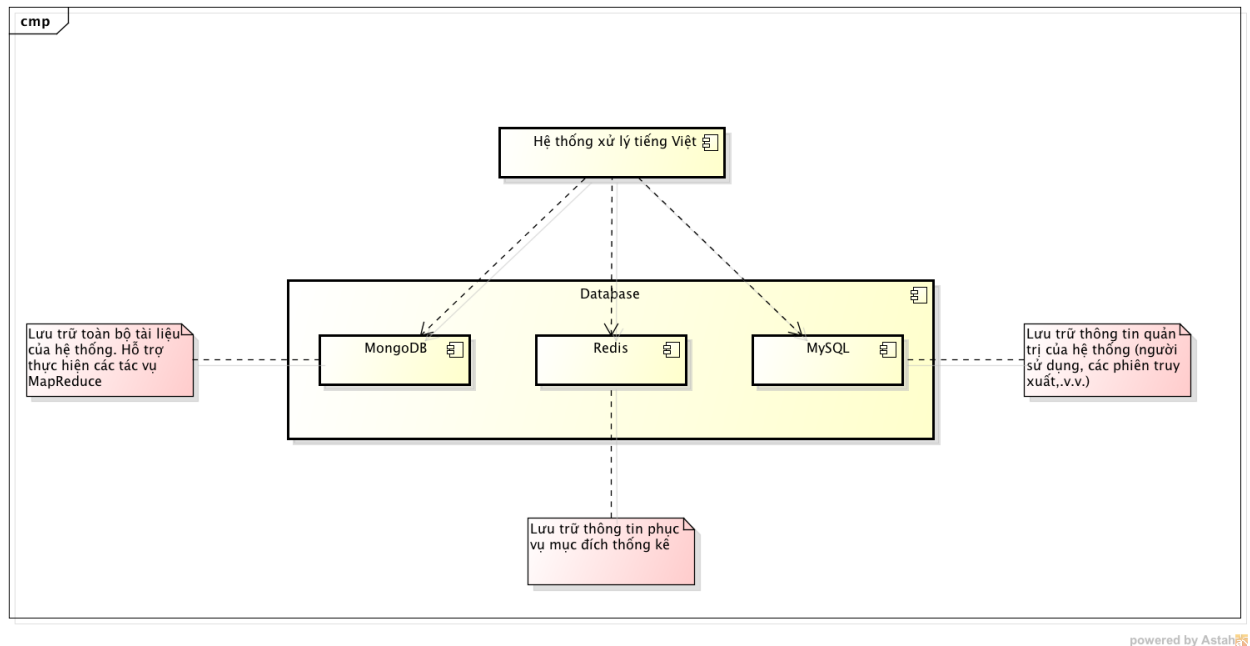
Tùy theo yêu cầu được gửi đến (dựa trên User Agent và Accept header), API Controller sẽ xác định định dạng trả lại cho yêu cầu đó. Có 3 loại định dạng:

1. `JSON`: là cách trình bày được sử dụng đầu tiên ở ngôn ngữ JavaScript tuy nhiên do nhiều ưu điểm đã phát triển và mở rộng, được hỗ trợ rộng rãi.
2. `JSONP`: là cách trình bày đặc biệt cho JavaScript, thay vì trả lại kết quả là một chuỗi JSON, `JSONP` bao bên ngoài kết quả đó một lời gọi hàm JavaScript. `JSONP` thường được sử dụng để vượt qua các rào cản khi sử dụng `XMLHttpRequest` giữa các tên miền khác nhau.
3. `XML`: là cách trình bày phổ biến được hỗ trợ trên rất nhiều ngôn ngữ.

3.1.2. Cấu trúc cơ sở dữ liệu

Sau khi tìm hiểu về hệ thống quản trị cơ sở dữ liệu hiện có, nhận thấy rằng mỗi hệ thống đều có những ưu nhược điểm riêng chứ không có một giải pháp nào có thể áp dụng cho mọi bài toán. Chính vì vậy, hệ thống xử lý tiếng Việt sử dụng kết hợp nhiều hệ thống cơ sở dữ liệu như sau:

- MongoDB: Lưu trữ các văn bản được đăng lên hệ thống
- Redis: Chuyên sử dụng để tính toán thống kê
- MySQL: Lưu trữ các thành phần khác của hệ thống như thông tin người sử dụng (tên đăng nhập, mật khẩu), thông tin các ứng dụng đã đăng ký, v.v.



Cấu trúc chi tiết:

- MongoDB: có một collection duy nhất để chứa tài liệu của hệ thống (documents). Mặc dù MongoDB không hỗ trợ schema nhưng hệ thống tự quy ước mỗi tài liệu phải có các thông tin là: appId, text, extraData, sections.
- Redis: hệ thống sử dụng 2 loại dữ liệu của Redis là set và hash. Trong đó cụ thể là:
 - aws_{appId}: là set chứa các từ khóa của một ứng dụng.
 - astws_{appId}_{section}: là set chứa các từ khóa của một mục của một ứng dụng.
 - awh_{appId}_{word}: là hash chứa bộ đếm của một từ khóa của một ứng dụng. Trong đó các hash key là _app nếu là bộ đếm của

toàn ứng dụng và là `s_{section}` nếu là bộ đếm của một mục của ứng dụng.

- MySQL: hiện tại hệ thống chỉ sử dụng một bảng trong MySQL là bảng `user` với các cột như sau:
 - `app_id`: UNSIGNED INT AUTO_INCREMENT
 - `username`: VARCHAR(255) NOT NULL
 - `password`: VARCHAR(255) NOT NULL
 - `email`: TEXT
 - `created`: UNSIGNED INT NOT NULL

3.1.3. API cho hệ thống xử lý tiếng Việt

Sau khi tìm hiểu kỹ càng các yêu tố cần cân nhắc để thiết kế RESTful web API, hệ thống xử lý tiếng Việt sẽ được thiết kế với các URI như sau:

URI	Phương thức	Ý nghĩa
/users	POST	Tạo người sử dụng mới, đồng thời tạo ứng dụng mới với người sử dụng này là người quản trị.
/documents	GET	Lấy danh sách các tài liệu của một ứng dụng.
/documents/:id	GET	Lấy thông tin về một tài liệu với id được chỉ định.
/documents	POST	Đăng một tài liệu mới lên hệ thống xử lý tiếng Việt.
/similar	POST	Tìm các tài liệu tương tự đoạn text được yêu cầu.
/search	POST	Tìm các tài liệu dựa trên từ khóa được chỉ định.
/words	GET	Lấy danh sách các từ khóa trong các tài liệu đã được xử lý.
/words/:word	GET	Lấy thông tin về một từ khóa được chỉ định.
/sections/:section	GET	Lấy thông tin về một chủ đề được chỉ định.

3.1.4. Giải thuật xử lý

Hệ thống xử lý tiếng Việt có hai xử lý đòi hỏi tính toán lớn đó là việc trích xuất từ khóa từ một đoạn văn bản và việc tìm kiếm văn bản tương tự.

3.1.4.1. Trích xuất từ khóa

Việc trích xuất từ khóa được thực hiện trong Document Model với input là một đoạn văn bản và output là danh sách các từ khóa trong đoạn văn bản đó. Các bước thực hiện trích xuất:

1. Xử lý trích xuất từng từ trong văn bản sử dụng các dấu cách, kí tự phân chia câu và các kí tự đặc biệt khác.
2. Xử lý loại bỏ các từ không được xử lý (stop word) từ một danh sách có trước. Các từ này là: của, là, và, có, được, người, đã, không, trong, cho, những, các, với, này, để, ông, ra, khi, lại, đến, về, nhà, vào, cũng, làm, từ, đó, như, nhiều, sau, bị, tại, anh, phải, con, sự, tôi, việc, thì, chỉ, trên, còn, mà, thế, ngày, đi, nhưng, hiện, hàng, nhất, số, điều, theo, sẽ, bà, đang, cả, biết, mình, trước, vì, chỉ, rất, lên, rằng, nói, hơn, một, khác, chúng, đây, do, khá.
3. Xử lý gộp các từ đơn thành các cặp từ ghép và bộ ba. Ví dụ với 4 từ đơn là A B C D. Ghép thành các bộ từ: AB, BC, CD, ABC, BCD.
4. Xử lý loại bỏ các từ đơn hoặc từ ghép xuất hiện trọn vẹn ở trong một bộ ba nào đó. Ví dụ trong văn bản có 3 lần xuất hiện của từ ghép AB nhưng cũng có 3 lần xuất hiện của bộ ba ABC, như vậy từ ghép AB sẽ bị loại bỏ vì nó đã nằm trọn vẹn trong bộ ba ABC.

3.1.4.2. Tìm kiếm văn bản tương tự

Việc xử lý tìm kiếm văn bản tương tự được thực hiện trong Similar Model với input là một đoạn văn bản và output là danh sách các tài liệu tương tự với văn bản đó. Việc xử lý được thực hiện thông qua tác vụ MapReduce của MongoDB với hai giai đoạn là Map và Reduce.

Các bước xử lý trong giai đoạn Map:

1. Thực hiện phân tích n-gram với đoạn văn bản và tài liệu đang được xử lý
 - a. Phân tích n-gram với $n = 5$, $window = 50$, lấy từ có giá trị cao nhất.

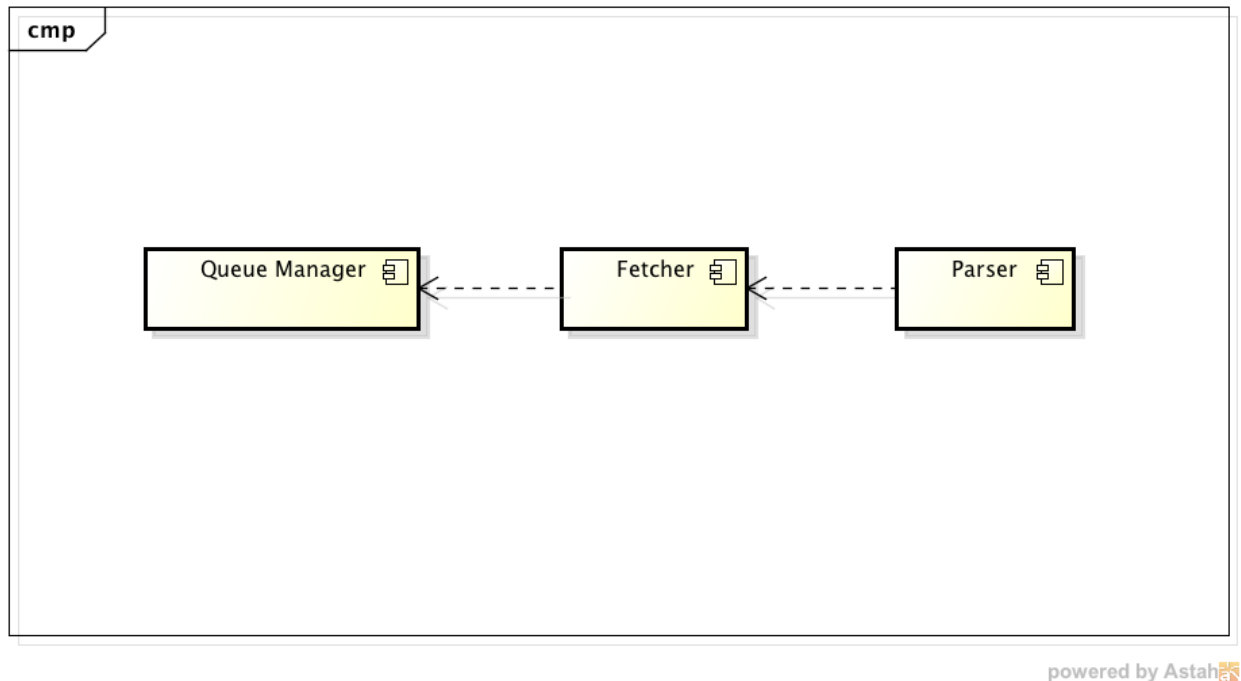
- b. Phân tích với $n = 5$ nghĩa là với đoạn văn bản “abcdefghi” sẽ phân tích thành các đoạn 5 chữ cái “abcde”, “bcdef”, “cdefg”, “defgh”, “efghi”.
 - c. Phân tích với $window = 50$ nghĩa là cứ mỗi đoạn văn bản 50 chữ lại giữ 1 đoạn 5 chữ để xử lý
 - d. Giá trị của đoạn 5 chữ được tính là tổng các giá trị trong bảng mã Unicode của từng kí tự.
2. Sau khi có kết quả phân tích n-gram của hai văn bản (văn bản được yêu cầu tìm và văn bản của tài liệu trong hệ thống), thực hiện so sánh các đoạn 5 chữ trùng khớp giữa hai kết quả này. Tính ra giá trị so sánh là số đoạn trùng khớp chia cho số đoạn 5 chữ. Giá trị này sẽ trong khoảng 0..1.
 3. Nếu giá trị so sánh lớn hơn 0.5 (giá trị này đang được hard code trong hệ thống) thì thực hiện emit với key là `_id` và value là giá trị so sánh. emit là một phương thức cung cấp bởi hệ thống MapReduce, dùng trong giai đoạn Map để đưa kết quả cho giai đoạn Reduce.

Xử lý trong giai đoạn Reduce đơn giản hơn, chỉ bao gồm việc trả lại value đầu tiên trong danh sách các value. Giai đoạn này chỉ phải xử lý như vậy vì key được emit trong giai đoạn Map là `_id` tức được đảm bảo là duy nhất với mỗi tài liệu rồi.

3.2. Crawler (Hệ thống tổng hợp tin tức)

3.2.1. Cấu trúc hệ thống

Crawler được thiết kế với 3 đoạn mã Python chạy độc lập với nhau tuy nhiên mỗi giai đoạn đều sử dụng kết quả của giai đoạn trước với mô hình như sau:



Mỗi giai đoạn có những công việc cụ thể là:

1. Queue Manager: lấy đường dẫn tin bài từ RSS feed
2. Fetcher: tải tin bài về từ các đường dẫn lấy được ở bước 1
3. Parser: xử lý các tin bài đã tải về, trích xuất nội dung và chuyển cho hệ thống xử lý tiếng Việt

3.2.2. Cấu trúc cơ sở dữ liệu

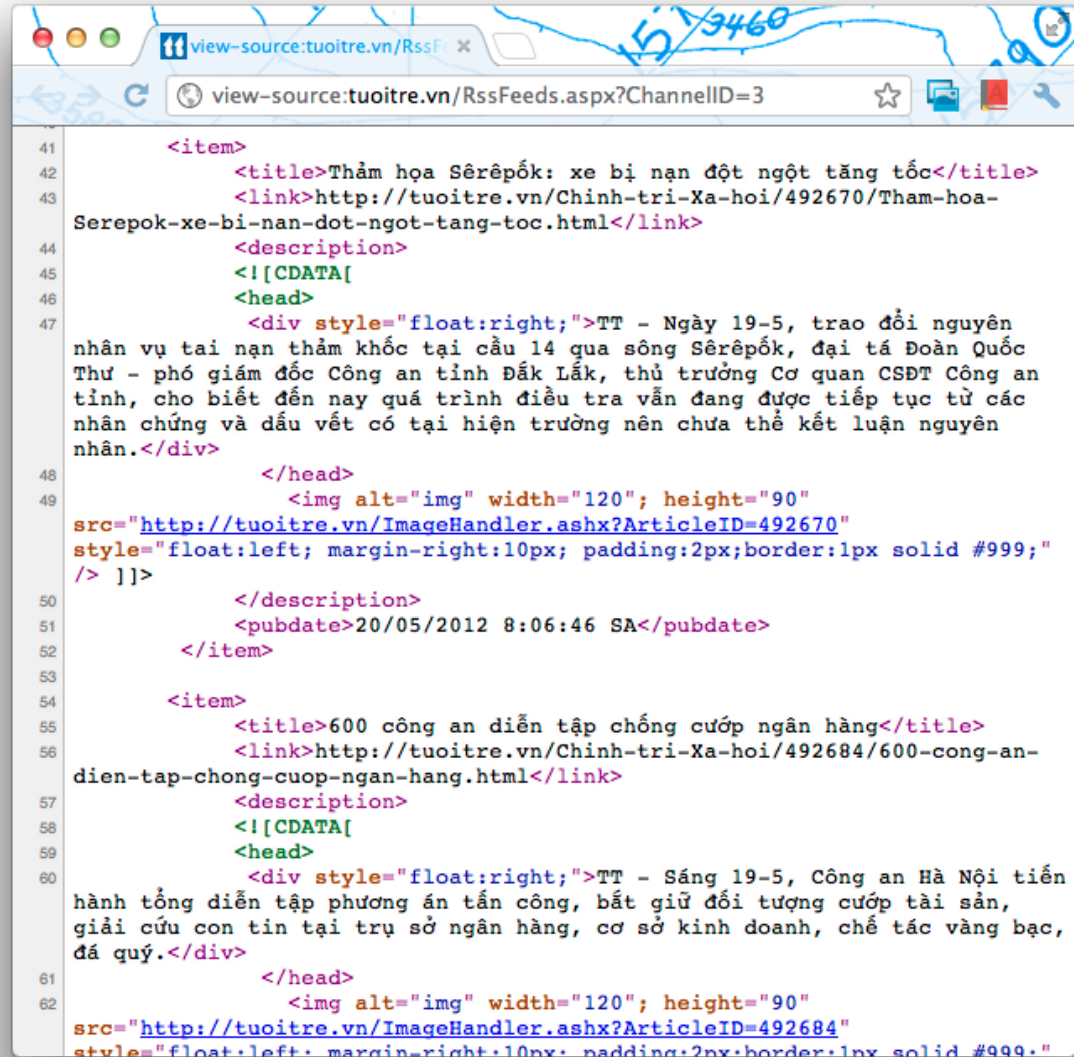
Toàn bộ Crawler sử dụng hệ thống file để lưu danh sách các RSS feed cũng như danh sách các đường dẫn chờ tải về. Riêng ở bước Parse, Crawler sẽ truy xuất vào cơ sở dữ liệu của Front-end để lưu trữ dữ liệu.

3.2.3. Giải thuật xử lý

3.2.3.1. Queue

Để cài đặt crawler của hệ thống tổng hợp tin tức, do có mục tiêu rõ ràng là các tin bài từ các trang báo, trang tin, trang tổng hợp của Việt Nam nên khóa luận không thực hiện theo mô hình thông thường của crawler (tải về, tìm siêu liên kết, lưu lại) mà sẽ sử dụng các trang RSS được cung cấp để thu thập đường dẫn đến các trang tin bài.

Với một danh sách địa chỉ các trang RSS có sẵn, ở bước Queue chỉ cần lần lượt tải về các trang RSS này ở dạng XML và xử lý.



```
41     <item>
42         <title>Thảm họa Sêrêpôk: xe bị nạn đột ngột tăng tốc</title>
43         <link>http://tuoitre.vn/Chinh-tri-Xa-hoi/492670/Tham-hoa-
Serepok-xe-bi-nan-dot-ngot-tang-toc.html</link>
44         <description>
45             <![CDATA[
46                 <head>
47                     <div style="float:right;">TT - Ngày 19-5, trao đổi nguyên
nhân vụ tai nạn thảm khốc tại cầu 14 qua sông Sêrêpôk, đại tá Đoàn Quốc
Thư - phó giám đốc Công an tỉnh Đắk Lắk, thủ trưởng Cơ quan CSĐT Công an
tỉnh, cho biết đến nay quá trình điều tra vẫn đang được tiếp tục từ các
nhân chứng và dấu vết có tại hiện trường nên chưa thể kết luận nguyên
nhân.</div>
48                 </head>
49                  ]]>
50             </description>
51             <pubdate>20/05/2012 8:06:46 SA</pubdate>
52         </item>
53
54     <item>
55         <title>600 công an diễn tập chống cướp ngân hàng</title>
56         <link>http://tuoitre.vn/Chinh-tri-Xa-hoi/492684/600-cong-an-
dien-tap-chong-cuop-ngan-hang.html</link>
57         <description>
58             <![CDATA[
59                 <head>
60                     <div style="float:right;">TT - Sáng 19-5, Công an Hà Nội tiến
hành tổng diễn tập phương án tấn công, bắt giữ đối tượng cướp tài sản,
giải cứu con tin tại trụ sở ngân hàng, cơ sở kinh doanh, chế tác vàng bạc,
đá quý.</div>
61                 </head>
62                  rồi tiếp tục tìm tiếp đối tượng <link> và trích xuất đường dẫn để lưu lại.

### 3.2.3.2. Fetch

Sau khi có đường dẫn tin bài ở bước 1, Fetch lấy lần lượt từng đường dẫn từ trong hàng chờ và tải về lưu tạm trong thư mục `articles`. Đường dẫn lưu được tính toán tương ứng theo đường dẫn tới tin bài. Một số ví dụ:

- <http://tuoitre.vn/The-gioi/383531/Bo-anh-hiem-hoi-cua-gia-dinh-Kim-Jong-Il.html> → `articles/tuoitre.vn/The-gioi/383531/Bo-anh-hiem-hoi-cua-gia-dinh-Kim-Jong-Il.html`
- <http://vnexpress.net/gl/van-hoa/2012/01/cac-ngoi-sao-gap-su-co-tren-may-bay> → `articles/vnexpress.net/gl/van-hoa/2012/01/cac-ngoi-sao-gap-su-co-tren-may-bay`

### 3.2.3.3. *Parse*

Sau khi đã có toàn bộ mã HTML của trang tin bài ở bước 2, Parse bắt đầu quá trình phân tích để lọc ra được nội dung tin bài (loại bỏ phần mã HTML).

Có nhiều giải pháp cho việc trích lọc này:

- Lọc bán thủ công: lập trình viên xem và nghiên cứu mã HTML của một trang nào đó (ví dụ một trang tin bài bất kì của Tuổi trẻ). Sau khi tìm hiểu kĩ cấu trúc của trang đó, lập trình viên viết đoạn mã trích xuất nội dung của trang tin bài đó theo một cách tổng quát nhất có thể. Sau đó, bất cứ khi nào có một trang tin bài khác tương tự (ví dụ cùng của Tuổi trẻ) thì hệ thống sử dụng đoạn mã đã có để trích xuất. Nhược điểm lớn của phương pháp này là lập trình viên phải xử lý lần lượt từng hệ thống đường dẫn để đảm bảo có đủ các đoạn mã trích xuất cho tất cả các đường dẫn trong hệ thống. Đó là một việc rất mất thời gian.
- Lọc thông minh: xử lý hiển thị trang web một cách đầy đủ sau đó tìm phần nội dung được hiển thị ở giữa cửa sổ trình duyệt hoặc ở trên đường cắt (above the fold, giới hạn 600px) kết hợp thêm một vài điều kiện khác. Từ đó tìm ra đoạn nội dung quan trọng của trang web để trích xuất. Giải pháp này lọc hiệu quả cao, gần với góc nhìn của người xem tuy nhiên yêu cầu xử lý lớn do để hiển thị trang web hoàn chỉnh cần rất nhiều thành phần: xử lý HTML, xử lý CSS, thậm chí phải xử lý cả JavaScript. Trước đây có giải pháp mã nguồn mở

Readability<sup>17</sup> viết bằng JavaScript ở dạng bookmarklet nên tận dụng được trình duyệt vốn đang chạy sẵn đem lại kết quả rất chính xác.

Bộ lọc của hệ thống sau khi thử nghiệm nhiều phương pháp khác nhau, cuối cùng đã sử dụng thuật toán như sau:

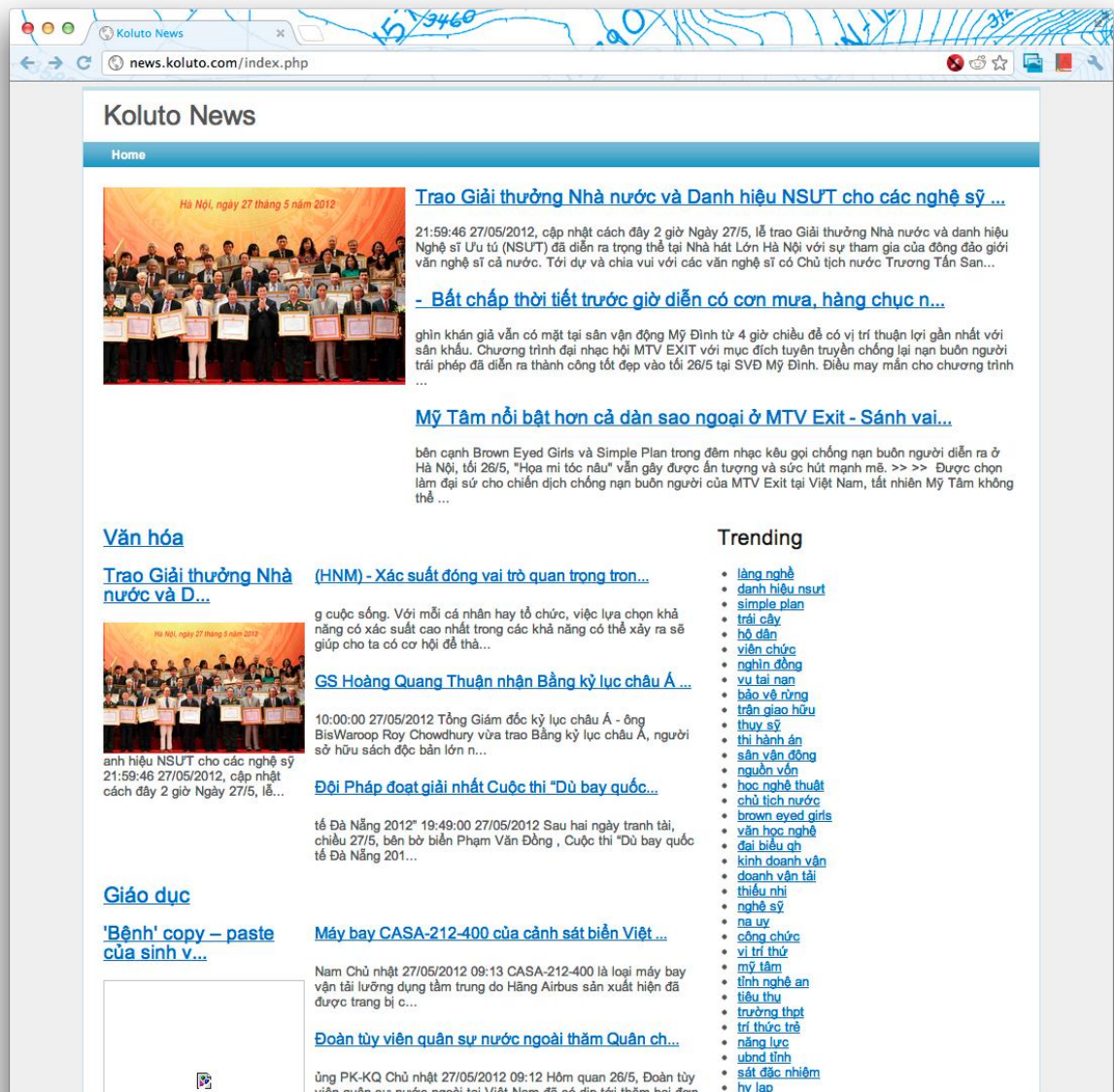
1. Mô hình hóa HTML của trang thành các thành phần với quan hệ cha con
2. Tìm các thẻ HTML có định danh là `<p>`, `<span>` và `<blockquote>`  
Kiểm tra xem nội dung của thẻ có chứa dấu chấm (.) hay không?
  - a. Nếu có, lấy cha của nó nếu cha chỉ chứa một phần tử con.
  - b. Lặp lại (2.b) cho đến khi không lấy được cha thỏa mãn nữa thì lưu lại thẻ cha hiện tại vào danh sách `filtered`.
3. Lấy thẻ cha của các thẻ trong `filtered` đưa vào danh sách `parents`.
4. Kiểm tra số thẻ trong `parents`:
  - a. Nếu có 1 thẻ, trả lại kết quả đã tìm thấy.
  - b. Nếu có 0 thẻ, không tìm thấy.
  - c. Nếu có nhiều thẻ:
    - i. Lấy thẻ chứa nhiều nội dung văn bản nhất.
    - ii. Trả lại kết quả tìm thấy.

Sau khi có kết quả lọc nội dung, nếu tìm thấy nội dung thì gửi nội dung này lên hệ thống xử lý tiếng Việt để xử lý và lưu vào cơ sở dữ liệu của Front-end.

### **3.3. Front-end (Hệ thống tổng hợp tin tức)**

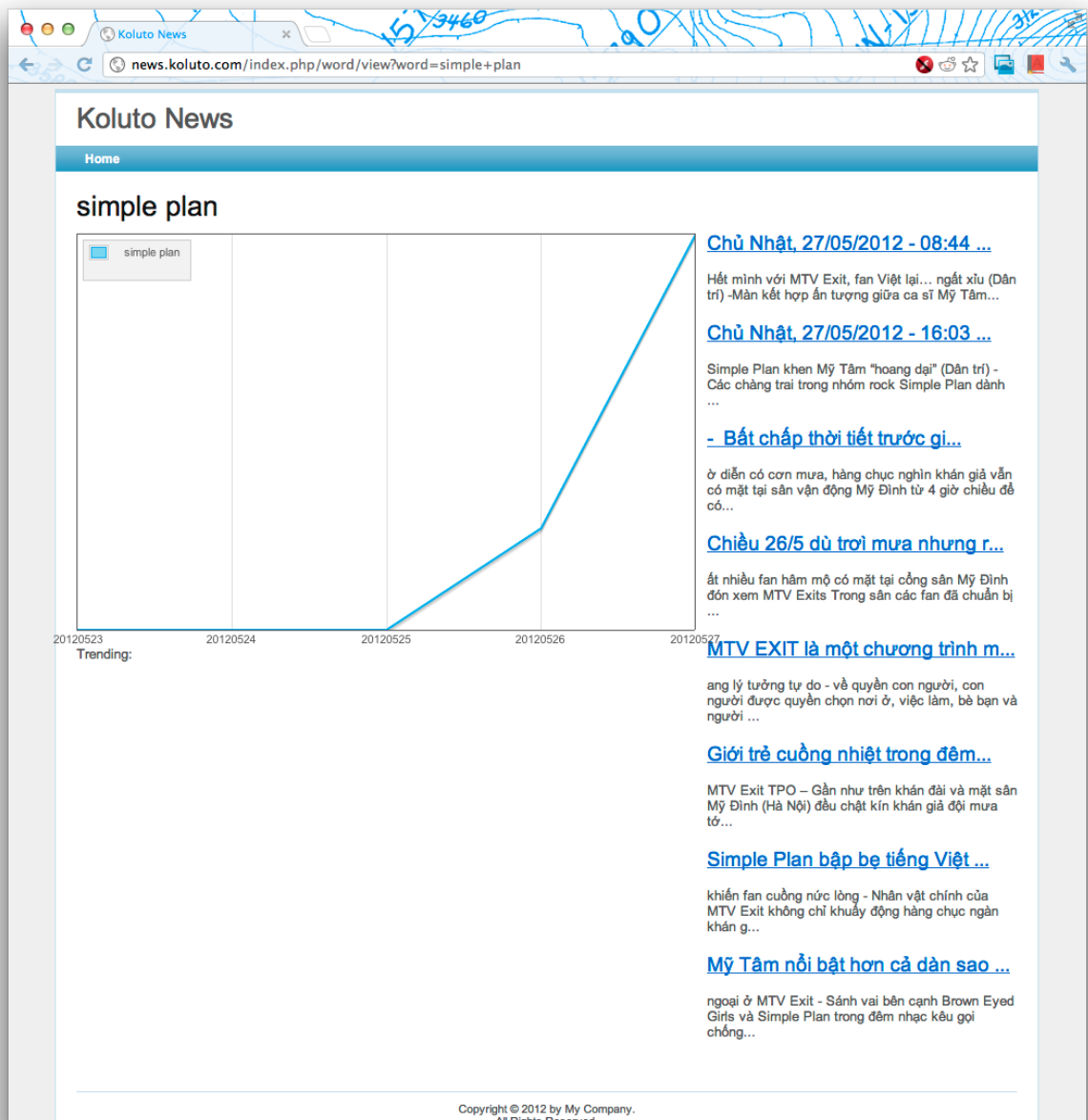
#### **4.2.1. Thiết kế giao diện của front-end**

##### **4.2.1.1. Trang chủ**



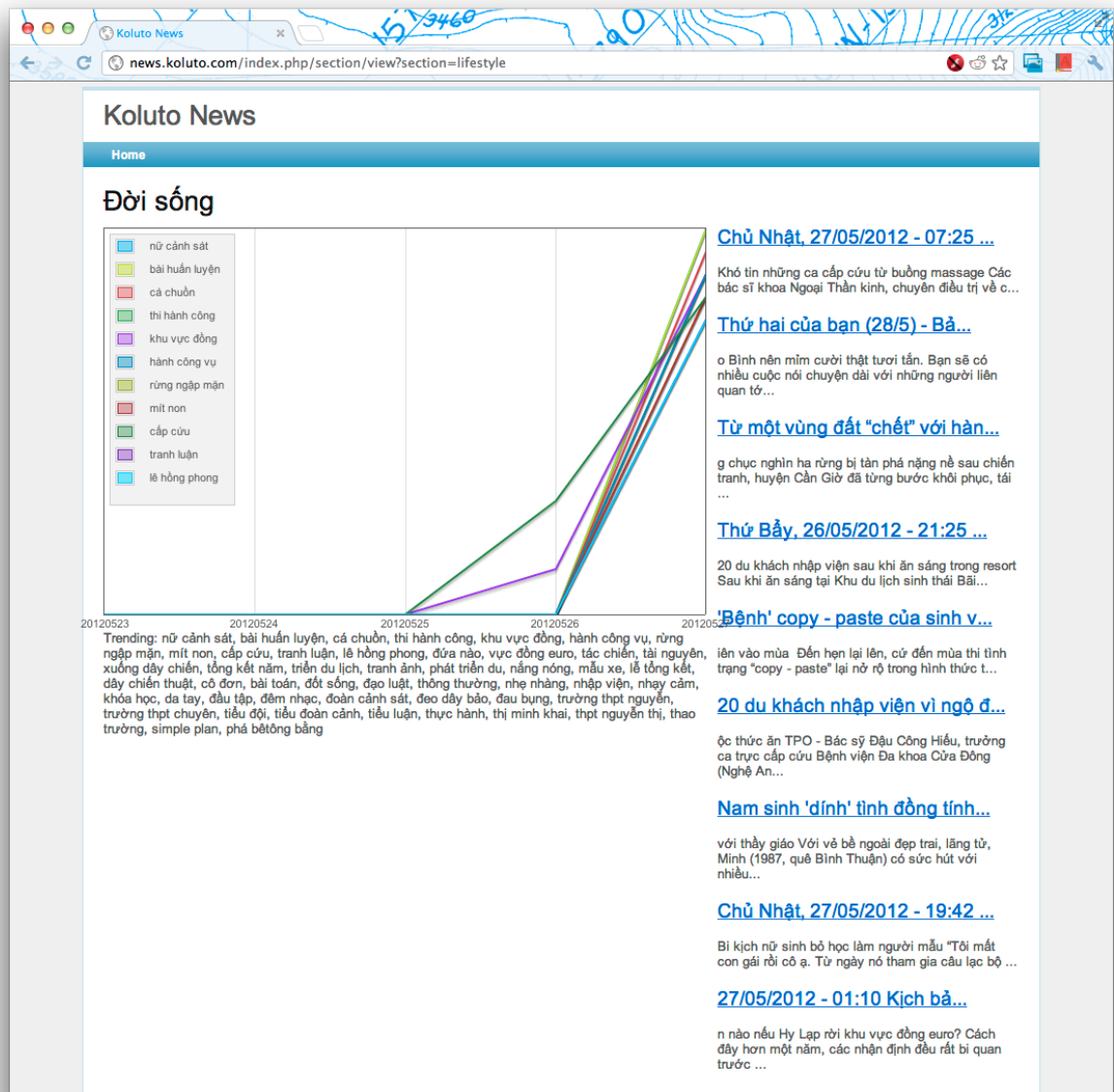
- Feature News: Hiển thị các tin mới nhất, đáng quan tâm nhất trong hệ thống
- Sections: Hiển thị lần lượt các mục và các tin trong mục
- Trending: Hiển thị danh sách các từ khóa đang thu hút sự quan tâm (trên toàn hệ thống)

#### 4.2.1.2. Trang từ khóa



Khi người dùng click vào một từ khóa, hiển thị biểu đồ thay đổi của từ khóa theo ngày. Đồng thời hiển thị danh sách các tin liên quan mới nhất của từ khóa.

#### 4.2.1.3. Trang theo mục



Khi người dùng click vào một mục, hiển thị biểu đồ thay đổi của các từ khóa trong mục này, chỉ hiện thị những từ khóa đáng quan tâm nhất. Đồng thời hiển thị danh sách các bài viết mới nhất, đáng quan tâm nhất ở bên cạnh.

#### 4.2.2. Giải pháp cài đặt

Với sự đơn giản của Front-end của hệ thống, phần này được xây dựng bằng PHP, kết nối với cơ sở dữ liệu MySQL (chung với Crawler). Để tăng tốc độ phát triển, sử dụng hệ thống nền Yii<sup>18</sup>.

## CHƯƠNG 4. THỰC NGHIỆM

### 5.1. Kết quả thực nghiệm

Toàn bộ mã nguồn của hệ thống đã được đăng tải ở các địa chỉ:

- Hệ thống xử lý tiếng Việt
  - <https://github.com/daohoangson/koluto-platform>
- Hệ thống tổng hợp tin tức
  - Crawler
    - <https://github.com/daohoangson/koluto-demo-crawler>
  - Front-end
    - <https://github.com/daohoangson/koluto-demo-front-end>

Hệ thống được chạy thử ở các địa chỉ:

- Hệ thống xử lý tiếng Việt
  - <http://api.koluto.com:29690/>
- Hệ thống tổng hợp tin tức
  - <http://news.koluto.com/>

#### 5.1.1. Hệ thống xử lý tiếng Việt

##### 5.1.1.1. *POST /documents*

Đây là kết quả chạy thử `POST /documents` với nội dung gửi lên là tài liệu với độ dài khác nhau. Thời gian được đo ở phía Node.js sử dụng hàm `Date.now()`. Tập dữ liệu test được lựa chọn ngẫu nhiên trong 191805 file tài liệu.

| Lần chạy | Độ dài tài liệu | Thời gian |
|----------|-----------------|-----------|
| 1        | 4263            | 836ms     |
| 2        | 3115            | 263ms     |
| 3        | 1173            | 38ms      |
| 4        | 10485           | 3174ms    |

| Lần chạy | Độ dài tài liệu | Thời gian |
|----------|-----------------|-----------|
| 5        | 4726            | 1884ms    |
| 6        | 3576            | 304ms     |
| 7        | 9177            | 1683ms    |
| 8        | 11750           | 3395ms    |
| 9        | 6917            | 2449ms    |
| 10       | 516             | 9ms       |

#### 5.1.1.2. *POST /similar*

Đây là kết quả chạy thử `POST /similar` với nội dung gửi lên để kiểm tra và số tài liệu trong cơ sở dữ liệu khác nhau. Thời gian được đo ở phía Node.js sử dụng hàm `Date.now()`. Tập dữ liệu test được lựa chọn ngẫu nhiên trong 191805 file tài liệu.

| Lần chạy | Độ dài tài liệu | Số tài liệu trong db | Thời gian |
|----------|-----------------|----------------------|-----------|
| 1        | 3142            | 152                  | 1858ms    |
| 2        | 3112            | 152                  | 1854ms    |
| 3        | 3596            | 311                  | 1610ms    |
| 4        | 2765            | 311                  | 1563ms    |
| 5        | 4524            | 311                  | 1832ms    |

### 5.1.2. Hệ thống tổng hợp tin tức, phần Crawler

#### 5.1.2.1. *Queue*

Hiện tại, các địa chỉ RSS đang được sử dụng đến từ các nguồn sau:

| Nguồn            | Số địa chỉ RSS |
|------------------|----------------|
| 24h.com.vn       | 16             |
| antd.cand.com.vn | 5              |



| <b>Nguồn</b>    | <b>Số địa chỉ RSS</b> |
|-----------------|-----------------------|
| cand.com.vn     | 8                     |
| daidoanket.vn   | 4                     |
| dantri.com.vn   | 44                    |
| giaoduc.net.vn  | 7                     |
| hanoimoi.com.vn | 9                     |
| laodong.com.vn  | 17                    |
| nhandan.org.vn  | 11                    |
| phapluattp.vn   | 8                     |
| sgtt.vn         | 11                    |
| tienphong.vn    | 30                    |
| tuoitre.vn      | 23                    |
| vietnamnet.vn   | 7                     |
| vietnamplus.vn  | 11                    |
| vneconomy.vn    | 8                     |
| vnexpress.net   | 8                     |
| zing.vn         | 22                    |
| <b>Tổng số</b>  | <b>249</b>            |

Với 249 đường dẫn RSS, kết quả chạy Queue vào ngày 28-05-2012 trích xuất được 4052 đường dẫn tin bài.

#### **5.1.2.2. Parse**

Đây là kết quả các lần chạy thử Parse với tập dữ liệu đầu vào khác nhau (chọn lựa ngẫu nhiên từ 191805 file dữ liệu).

Cách thức thực hiện test:

- Tạo tập dữ liệu:

- o `python test002_get_articles_to_test.py`

- Xử lý trên tập dữ liệu đã tạo:

- o `python test003_parse_test_articles.py`

Tỉ lệ được tính bằng kết quả của phép chia số tin bài xử lý thành công cho tổng số tin bài trong tập dữ liệu. Thời gian được đo bằng câu lệnh `time` của Unix. Lấy giá trị `real` chia cho số tin bài đã xử lý thành công.

| Lần chạy | Số lượng | Thành công | Tỉ lệ  | Thời gian |
|----------|----------|------------|--------|-----------|
| 1        | 164      | 144        | 87.80% | 111ms     |
| 2        | 248      | 220        | 88.71% | 106ms     |
| 3        | 415      | 380        | 91.57% | 113ms     |
| 4        | 3864     | 3544       | 91.72% | 119ms     |
| 5        | 3823     | 3508       | 91.76% | 126ms     |

Một số trang có tỉ lệ xử lý không thành công cao:

- `batdongsan.vietnamnet.vn` (do một số bài có toàn ảnh)
- `daidoanket.vn` (do nội dung chính để trong thẻ `<div>`)
- `hanoimoi.com.vn` (do nội dung chính để trong thẻ `<div>`)
- `nhandan.com.vn` (do một số đường dẫn trong RSS không phải là tin bài)
- `vef.vn` (do các trang tải về sử dụng JavaScript để chuyển tới một trang khác)

## 5.2. Hướng phát triển tiếp theo

Hầu hết các thành phần của hệ thống đều có thể được cải thiện theo các hướng khác nhau để tăng cường khả năng của toàn hệ thống.

Đối với hệ thống xử lý tiếng Việt, mô hình lưu trữ hiện nay trong thực nghiệm có những thời điểm vẫn còn chậm, gây treo máy nên cần phải thực hiện nhiều kiểm tra hơn nữa để đạt được tính ổn định. Mặc dù thiết kế để thực hiện horizontal scaling tuy nhiên hệ thống chưa được cài đặt trên nhiều hơn hai máy chủ (cho cả cơ sở dữ liệu MongoDB và Node.js). Đúng theo thiết kế thì hiệu năng của hệ thống tăng tuyến tính theo số lượng máy

chủ được sử dụng. Về mặt xử lý, có hai giai đoạn quan trọng là bước phân tích từ khóa và bước tìm kiếm các tài liệu tương tự với một tài liệu cho trước. Việc xử lý trong hai bước này còn thô sơ, chưa áp dụng được các thuật toán mạnh trong xử lý ngôn ngữ tự nhiên. Trong quá trình thực hiện cũng còn nhiều ý tưởng chưa thể thực hiện được ví dụ như việc xử lý giá trị số theo cách riêng bằng cách xác định các chuỗi đặc trưng của tiếng Việt như “nghìn”, “triệu”, “tỉ” (hoặc “tỷ”) để trích xuất nhiều hơn nữa các thông tin có giá trị. Ngoài ra có thể xác định tên riêng của người hay địa danh thông qua cách trình bày với chữ đầu tiên viết hoa và có từ 2 từ như vậy đứng liền nhau (không tính từ đầu tiên của câu).

Đối với hệ thống tổng hợp tin tức, giai đoạn Parse của Crawler là giai đoạn có thể cải thiện nhiều nhất. Mặc dù tỉ lệ lọc bài viết thành công nằm trong khoảng chấp nhận được tuy nhiên vẫn còn chưa cao như mong muốn. Thêm vào đó, kết quả lọc đôi khi còn chứa các thông tin thừa ví dụ như “Lưu để đọc sau” hoặc “Email bài này”, “In trang này”, v.v. Đây là các siêu liên kết đến các chức năng riêng của một số trang tin, do các siêu liên kết này đặt quá gần với nội dung bài nên giai đoạn Parse đã xác định nhầm. Ở trang hiển thị, hiện tại còn thiếu một số tính năng theo dự kiến ban đầu như cho phép người xem đăng ký tài khoản, tạo các bộ lọc, thiết lập thông báo hàng ngày qua email, v.v.

Nhìn chung, hệ thống xử lý tiếng Việt là một hệ thống khá hoàn chỉnh về chức năng và có thể được phát triển để cung cấp dịch vụ mang tính chất thương mại với mô hình Freemium như đã trình bày. Để đạt được mục tiêu đó, cần đặt ưu tiên cao nhất vào việc đảm bảo tính ổn định của hệ thống, chống các truy cập trái phép, đảm bảo an toàn thông tin của người sử dụng, v.v.

### **5.3. Kết luận**

Với những kết quả nghiên cứu và thực nghiệm như đã trình bày, công cụ thu được đã giải quyết được phần nào các vấn đề được đặt ra. Tuy nhiên do còn thiếu kinh nghiệm nên kết quả thu được còn nhiều hạn chế, cần tiếp tục được cải thiện, nâng cấp.

## TÀI LIỆU THAM KHẢO

### Tiếng Việt

[1] TS. Nguyễn Tuệ. Giáo trình cơ sở dữ liệu. Khoa Công Nghệ, ĐHQGHN, 2004

[2] ThS. Đào Kiến Quốc. Giáo trình phân tích thiết kế Hệ thống thông tin - Tin học hoá. Khoa Công Nghệ, ĐHQGHN

### Tiếng Anh

[3] Peter Rob, Carlos Coronel. Database system: Design, implementation and management. Wadsworth Publishing Company, 1993

### Endnote

---

<sup>1</sup> Trang tổng hợp thông tin tự động BáoMới (<http://www.baomoi.com>)

<sup>2</sup> Các công cụ xử lý văn bản tiếng Việt của Lê Hồng Phương (<http://www.loria.fr/~lehong/software.php>)

<sup>3</sup> Các nghiên cứu của Phan Xuân Hiếu và Nguyễn Cẩm Tú (<https://sites.google.com/site/pxhieu/software>)

<sup>4</sup> Thống kê nguồn tin của BáoMới (<http://www.baomoi.com/Statistics/Report.aspx>)

<sup>5</sup> Trang tổng hợp thông tin Cubit NewsDesk (<https://cubit.moreover.com>)

<sup>6</sup> Dịch vụ xác định sao chép trên mạng Copyscape (<http://copyscape.com>)

<sup>7</sup> Dịch vụ xác định sao chép trong văn bản học thuật turnitin (<http://turnitin.com>)

<sup>8</sup> Dịch vụ xác định sao chép trong văn bản học thuật SafeAssign (<http://safeassignment.com>)

<sup>9</sup> Codd, E.F. (1970). “A Relational Model of Data for Large Shared Data Banks”

<sup>10</sup> Nhánh phát triển của MySQL được sử dụng tại Google (<http://code.google.com/p/google-mysql/>)

<sup>11</sup> Nhánh phát triển của MySQL được sử dụng tại Twitter (<https://github.com/twitter/mysql>)

<sup>12</sup> PostGIS (<http://postgis.refractory.net>)

<sup>13</sup> Nền tảng điện toán đám mây heroku (<http://heroku.com>)

<sup>14</sup> Web crawler ở Wikipedia tiếng Anh ([http://en.wikipedia.org/wiki/Web\\_crawler](http://en.wikipedia.org/wiki/Web_crawler))

<sup>15</sup> Cài đặt RESTful web API đầy đủ cho CodeIgniter (<https://github.com/philsturgeon/codeigniter-restserver>)

<sup>16</sup> Bảng xếp hạng TIOBE (<http://www.tiobe.com/index.php/content/paperinfo/tpci/index.html>)

<sup>17</sup> Readability mã nguồn mở (<http://code.google.com/p/arc90labs-readability/>)

<sup>18</sup> Hệ thống nền Yii (<http://www.yiiframework.com/>)