**ĐẠI HỌC QUỐC GIA TP. HỒ CHÍ MINH**

**TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN**

**KHOA KỸ THUẬT PHẦN MỀM**

**PHẠM TẤN LONG**

**KHÓA LUẬN TỐT NGHIỆP**

**NGHIÊN CỨU UNITY VÀ XÂY DỰNG ỨNG DỤNG HỌC LUẬT GIAO THÔNG**

**KỸ SƯ NGÀNH KỸ THUẬT PHẦN MỀM**

**TP. HỒ CHÍ MINH, 2014**

**ĐẠI HỌC QUỐC GIA TP. HỒ CHÍ MINH**

**TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN**

**KHOA KỸ THUẬT PHẦN MỀM**

**PHẠM TẤN LONG – 10520163**

**KHÓA LUẬN TỐT NGHIỆP**

**NGHIÊN CỨU UNITY VÀ XÂY DỰNG ỨNG DỤNG HỌC LUẬT GIAO THÔNG**

**KỸ SƯ KỸ THUẬT PHẦN MỀM**

**GIẢNG VIÊN HƯỚNG DẪN**

**ThS. NGUYỄN TRÁC THỨC**

**TP. HỒ CHÍ MINH, 2014**

DANH SÁCH HỘI ĐỒNG BẢO VỆ KHÓA LUẬN

Hội đồng chấm khóa luận tốt nghiệp, thành lập theo Quyết định số …………………… ngày ………………….. của Hiệu trưởng Trường Đại học Công nghệ Thông tin.

* 1. …………………………………………. – Chủ tịch.
  2. …………………………………………. – Thư ký.
  3. …………………………………………. – Ủy viên.
  4. …………………………………………. – Ủy viên.

# LỜI CẢM ƠN

**MỤC LỤC**

**DANH MỤC HÌNH ẢNH**

# 

**DANH MỤC BẢNG**

# TÓM TẮT LUẬN VĂN

Với sự phát triển mạnh mẽ của đất nước trong những năm gần đây, ngoài những thay đổi tích cực về đời sống con người, Việt Nam cũng đang phải đối mặt với nhiều vấn đề lớn, trong đó có vấn đề giao thông.

Với mong muốn được sử dụng công nghệ thông tin để góp phần làm thay đổi đất nước, khoá luận đã được hình thành ý tưởng như vậy.

Từ kết quả nghiên cứu, tìm hiểu về Unity Engine, nhóm đã sử dụng công cụ này để xây dựng nên phần mềm học luật giao thông nhằm hỗ trợ thêm một công cụ học tập trực quan sinh động hơn, nhằm nâng cao ý thức người dân khi tham gia giao thông, từ đó hạn chế tai nạn giao thông, cải thiện tình trạng giao thông Việt Nam.

Trước hết, quá trình nghiên cứu Unity Engine bao gồm tìm hiểu về các vấn đề cơ bản: game engine là gì, các loại game engine, các engine phổ biến, so sánh các engine đó, và lý do lựa chọn Unity. Tiếp theo là các vấn đề chuyên sâu của engine này, bao gồm: kiến trúc component, so sánh với kiến trúc OOP, particle system, animation system, physic system và terrain.

Cuối cùng là việc áp dụng Unity Engine vào việc xây dựng phần mềm học luật giao thông chạy trên nền web, thông qua Unity Web Player. Bên cạnh việc sử dụng Unity, để có thể hoàn thành hệ thống, nhóm đã tìm hiểu thêm về Laravel Framework để xây dựng nên trang quản trị nội dung cho hệ thống.

Kết quả cuối cùng, khóa luận tạo ra một hệ thống bao gồm 3 sản phẩm: Map Editor, Game và CMS.

Map Editor lần một phần mềm standalone, cho phép người quản trị xây dựng tùy biến một khu vực giao thông có hầu hết các thành phần giao thông trong thực tế gồm: làn đường, vỉa hè, vạch kẻ đường, biển báo, đèn giao thông, trạm xe buýt... Bên cạnh là các thành phần phối cảnh: nhà cửa, cây cối, người qua đường, các xe tham gia giao thông.

Game là phần mềm chạy trên web, thông qua Unity Web Player. Đây là nơi người chơi có thể tham gia chơi và học luật giao thông.

CMS là trang web cho phép nhà quản trị quản lý nội dung của game, quản lý người chơi, xem kết quả .v.v.

# MỞ ĐẦU

Trong chín tháng đầu năm 2014, cả nước đã xảy ra 18.697 vụ, làm chết 6.758, làm bị thương 17.835 người. Cũng trong chín tháng qua, lực lượng cảnh sát giao thông toàn quốc đã kiểm tra, lập biên bản hơn 3,37 triệu trường hợp vi phạm trật tự an toàn giao thông đường bộ; phạt tiền 1.943 tỷ đồng; tạm giữ 441.020 phương tiện các loại; tước 275.153 giấy phép lái xe. Với số liệu trên đã cho thấy, bên cạnh những lý do về cơ sở hạ tầng thì ý thức, kiến thức của người tham gia giao thông cũng là nguyên nhân quan trọng.

Trên thực tế, đã có rất nhiều nỗ lực tuyên truyền của nhà nước trên báo chí, truyền hình hay trực tiếp tại các trường học, tổ dân phố. Bên cạnh đó cũng đã có nhiều ứng dụng học luật giao thông, kiểm tra luật giao thông trên PC, mobile để mọi người dễ dàng tiếp cận. Tuy nhiên, tất cả các biện pháp này chỉ cung cấp thông tin về luật giao thông, các quy định xử phạt, v.v.. Nhược điểm của những biện pháp này là chỉ mang tính đối phó, thiếu tính trực quan, không thu hút người dùng và nội dung không được cập nhật, đổi mới, do đó không thể đem lại những thay đổi tích cực trong ý thức cũng như kiến thức người dân.

Như vậy, cần có một hệ thống giáo dục luật giao thông mới trực quan, sinh động hơn, cho phép người học tham gia trực tiếp vào các tình huống giao thông, tương tác như trong thực tế, đồng thời có thể dễ dàng thay đổi, cập nhật nội dung học tập, và hơn nữa là đa nền tảng để có thể đến được với mọi đối tượng người học.

Một game 3D là một ý tưởng tuyệt vời để thực hiện điều này. Các tình huống giao thông sẽ trở nên thực tế hơn, sinh động hơn trong môi trường 3D. Người tham gia có thể tương tác trực tiếp, tạo ra các tình huống vi phạm và bị xử phạt. Những kiến thức giao thông sẽ dễ dàng tiếp cận hơn khi việc học trở thành một trò chơi.

Unity đang là một game engine mạnh, phổ biến và có nhiều ưu điểm hơn hẳn các engine khác, đặc biệt khi Unity trở nên miễn phí cho cả các sản phẩm thương mại kể từ phiên bản 4.3. Người dùng cá nhân hoặc các tổ chức phi lợi nhuận có thể download, sử dụng và phát hành sản phẩm hoàn toàn miễn phí.

Với tất cả những khảo sát đó, em xin lựa chọn đề tài "Nghiên cứu Unity và xây dựng ứng dụng học luật giao thông"

# TỔNG QUAN GAME ENGINE

## Game Engine là gì?

Một game engine là một phần mềm được viết để thiết kế và phát triển video game, hiểu đơn giản nó là loại phần mềm trung gian kết nối tương tác của nhiều ứng dụng trong cùng 1 hệ thống với nhau. Chức năng cốt lõi của game engine phần lớn nằm trong công cụ dựng hình (kết xuất đồ họa) cho các hình ảnh 2 chiều (2D) hay 3 chiều(3D), công cụ vật lý (hay công cụ tính toán và phát hiện va chạm), âm thanh, mã nguồn, hình ảnh động (animation), trí tuệ nhân tạo, phân luồng, tạo dòng dữ liệu xử lý, quản lý bộ nhớ, dựng ảnh đồ thị, và kết nối mạng. Quá trình phát triển game tiết kiệm được rất nhiều thời gian và kinh phí vào việc tái sử dụng và tái thích ứng một engine để tạo nhiều game khác nhau.

## Nguồn gốc Game Engine

Vào lúc trước, rất nhiều công ty phát triển engine của chính họ để giữ bí mật công nghệ. Tuy nhiên thời gian trôi qua, công nghệ phát triển như vũ bão và nhu cầu ngày càng nâng cao khiến những engine mạnh dần phổ biến. Những engine như SCUMM (LucasArts) và SCI (Serria) đã được dùng để thiết kế rất nhiều game phiêu lưu ra mắt vào khoảng cuối thập niên 80 đến giữa thập niên 90. Hoặc những engine như idTech (tạo nên serie game bắn súng Quake) hoặc Unreal Engine (mới đầu được giữ bí mật) được phát triển trở thành những middleware.

Nhiều năm sau, chi phí để tự xây dựng một engine trở nên quá đắt đỏ, và ngày có càng nhiều công ty thiết kế nên những engine hoặc những phần của engine để bán cho những công ty khác thay vì phát triển game. Họ được gọi là những nhà cung cấp middleware. Họ có thể cung cấp những sản phầm với cái giá chấp nhận được, và với hầu hết những studio phát triển game, những sản phầm này rất đáng đồng tiền. Tại sao phải trả tiền thuê 6 lập trình viên trong vòng một năm để phát triển một engine trong khi bạn có thể mua nó với cái giá rẻ hơn nhiều và có liền. Kết quả hầu hết mọi thành phần của game engine đều được bán với nhiều giá khác nhau, hoặc có thể download tại những website cung cấp phần mềm mã nguồn mở.

## Các loại Game Engine

Game engine có nhiều loại khác nhau chia thành từng mức chuyên biệt khác nhau. Để dễ hiểu, tạm chia ra thành 3 loại: roll-your-own, mostly-ready, và point-and-click engine.

### Roll-Your-Own

Roll-your-own engine có thể được xem như những engine ở mức thấp nhất. Ngày nay rất nhiều công ty tự tạo ra engine cho riêng họ với những thứ có thể. Điều này có nghĩa họ dùng những ứng dụng giao diện được công bố rộng rãi, các API như XNA, DirectX, OpenGL, các API và SDL của Windows và Linux. Thêm vào đó, họ có thể dùng những thư viện từ nhiều nguồn khác nhau, được mua hoặc open-source. Những thư viện đó có thể bao gồm cả những thư viện vật lý như Havok và ODE, những thư viện scene-graph như OpenSceneGraph và các thư viện GUI như AntTweakBar.

Thực tế, những hệ thống của Roll-your-own engine được đó cho phép những lập trình viên dễ tương tác hơn rất nhiều vì họ biết cần gì và chọn những thành phần phù hợp từ đó tạo nên chính xác thứ cần thiết. Nhưng cũng vì vậy mà tốn rất nhiều thời gian. Thêm vào đó, những lập trình viên thường gặp cả đống lỗi lăn nhăn khi các thư viện từ nhiều nguồn đụng chạm với nhau. Điều này đã góp phần làm cho những engine dạng này kém hiệu quả và không được ưa chuộng cho lắm ngay cả với engine được thiết kế một cách chuyên nghiệp.

### Mostly-Ready

Những engine nằm ở tầng trung. Các nhà thiết kế game nghiệp dư rất thích những engine dạng này. Chúng được thiết kế rất đầy đủ với mọi tính năng cần thiết, vẽ, tương tác, GUI, vật lý… ngoài ra chúng còn kèm theo cả lô tool mạnh cho bạn tha hồ sử dụng. Những engine tầng này khá phong phú từ OGRE và Genesis3D (open-source), rẻ tiền như Torge hoặc đại gia như Unreal, id Tech và Gamebryo.

Về mức độ, những engine kể trên đều cần lập trình thêm để gắn kết mọi thứ với nhau nhằm tạo nên một game hoàn chỉnh. Hầu như mọi Mostly-ready game engine đều có phần hạn chế một chút so với roll-your-own engine. Cũng đúng thôi vì mặc dù nó là sản phầm của rất nhiều người tham gia phát triển trải qua thời gian dài nhằm mang đến cho khách hàng những “tiện nghi”tốt nhất nhưng đâu phải khách hàng nào cũng giống nhau và phát triển những game tương tự nhau.

### Point-And-Click

Point-and-click engine được xếp ở tầng cao nhất. Những engine này ngày càng trở nên phổ biến. Nó có tất cả mọi tool với đầy đủ chức năng cho phép bạn chỉ việc chọn, click và chọn để tạo ra game. Những engine như vậy, kể cả GameMaker, Torque Game Builder, và Unity3D được thiết kế để càng thân thiện với người dùng càng tốt. Nó cũng yêu cầu ít lập trình hơn. Điều này không có nghĩa việc lập trình không cần, nhưng nếu so với 2 loại engine kể trên trên, nó cần ít nhất.

Nhược điểm chí mạng của những engine point-and-click này đến từ việc những nhà phát triển bị hạn chế rất nhiều vì với loại engine này, bạn chỉ có thể cho ra một loại giao diện như vầy hoặc khác một chút. Nhưng điều này không có nghĩa chúng vô dụng. Vì khi đối mặt với những hạn chế đó, những sáng tạo của con người sẽ càng được kích thích cao. Ngoài ra, engine này cho phép bạn có thể làm việc nhanh hơn, làm ra game nhanh hơn mà không phải làm quá nhiều.

## Một số Engine trên thị trường

### CryEngine

CryEngine 3 là game engine được Crytek phát triển và phá hành vào 4 tháng 10 năm 2009. Là một bướt phát triển tiếp nối theo phiên bản CryEngine2 khá thành công của hãng. Theo Crytek, CryEngine 3 được tao ra với tham vọng trở thành công cụ phát triển game tất cả trong một, ứng dụng trên cả Microsoft windows, PlayStation 3 và Xbox 360. Riêng đối với nền PC, CryEngine 3 sẽ hỗ trợ tốt cả DirectX 9,10,11.

Điểm nổi trội so với các engine đương thời là tập trung vào mặt xử lý hình ảnh, ngoài ra hiệu ứng âm thanh và chuyển động cũng được mô phỏng xuất sắc. Những tựa game như Far Cry, Crysis, Crysis Warhead, Crysis 2 và Aion: Tower of Eternity là những các tên tiêu biểu sử dụng Cry Engine.

### OGRE Engine

OGRE (Object-Oriented Graphics Rendering Engine) là một Engine dựng hình 3D linh hoạt và tập trung vào khung cảnh ( hơn là một công của tạo game ) được viết bằng C++, được thiết kế để mang lại sự đơn giản và trực quan hơn cho nhà phát triển trong công việc tạo ra những ứng dụng có khai thác phần cứng tăng tốc đồ họa 3D. Bộ thư viện của OGRE trựu tượng hóa các hóa các thư viện mức hệ thống như Direct3D và OpenGL để cung cấp một giao diện lập trình dựa trên các đối tượng thể giới thực và các lớp cấp cao.

Như đã nói ở trên, OGRE cũng đã nói lên được chức năng chính của Engine này đó là dựng hình, do mục đích ban đầu của nhà phát triển là cung cấp một giải pháp chung hỗ trợ cho công việc kết xuất hình ảnh. Mặc dù OGRE cũng được trang bị nhiều tính năng khác (như các lớp chuyên về ma trận và vector, công cụ quản lý bộ nhớ,…) nhưng đó chỉ là những tính năng phụ thêm. Vì thế, OGRE không phải là một lựa chọn hoàn hảo cho công việc phát triển game do nó không cung cấp các thư viện hỗ trợ về âm thanh, hệ thống vật lý,… Mặc dù có vẻ như đây là một hạn chế lớn của OGRE, song đó cũng là một đặc điểm khác hay. Việc lựa chọn OGRE như là một Engine chuyên về đồ họa cho phép những nhà phát triển Game tự do sử dụng bất cứ hệ thống vật lý, nhập xuất, âm thanh hay bộ thư viện nào mà họ muốn, đồng thời cũng giúp cho nhóm phát triển OGRE tập trung vào mảng đồ họa thay vì phải phân tán công sức cho những công việc khác. OGRE hoàn toàn hỗ trợ các bộ thư viện OIS,SDL và CEGUI và cả bô công cụ Cg.

### Panda3D

Là một Engine Game bao gồm các chức năng về đồ họa, âm thanh, quản lý nhập xuất, hệ thống phát hiện va chạm và những chức năng khác phù hợp với nhiệm vụ phát triển Game 3D.

Panda3D là engine mã nguồn mỡ theo giấy phép sử dụng BSD được cấp ngày 28/5/2008. Thực tế thì những phiên bản được phát hành trước 28/5/2008 không được xem là miễn phí vì một số sai sót trong giấy phép cũ của Panda3D. Mặc dù vậy những phiên bản Pande3D phát hành sau này đều có thể được sử ụng để phát triển game miễn phí hoặc mang tính thương mại mà không phải trả phí.

Ngôn ngữ kịch bản dùng trong Panda3D là python. Bản thân Engine được viết bằng C++ và sử dụng một bộ sinh mã tự động để tạo thành các chức năng hoàn chỉnh cho Engine thông qua giao diện của Python. Cách tiếp này giúp lập trình tận dụng được ưu điểm của python như hỗ trợ lập trình nhanh và quản lý bộ nhớ tốt hơn, nhưng vẫn giữ lại được ưu điểm về khả năng biên dịch của C++ trong nhân của engine. Cụ thể, Panda3D tích hợp bộ dọn rác tự đọng của Python và công cụ quản lý cấu trúc tự động.

Thường người lập trình sử dụng python làm việc trên panda nhưng vẫn có thể lập trình trực tiếp bằng C++.Panda3D thường dùng cho các game thượng mai cỡ lớn, một số dự án mã nguồn mở và vài khóa học ở trường đại học

### UnrealEngine

Unreal engine là Engine Game được phát triển bởi Epic Games. Từ lần ra mắt đầu tiên vào năm 1998 với game Unreal, nó đã đặt nền móng cho những game sau này như Unreal Tournament, Deus Ex, Turok, Tom.

Mặc dù Engine nà được nhắm đến thể loại FPS nhưng nó cũng tỏ ra khá thành công khi dùng cho nhiều thể loại game khác như stealth, MMORPG cũng như RPG với các game như Mass Effect, The last remmant, killing Floor và Harry Porter.

Công nghệ của Epic game được coi là đối thủ cạnh tranh trực tiếp của CryEngine và cũng được áp dụng phổ biến nhất trong các trò chơi hiện nay. Khởi đầu không thuận lợi do gạp một số vấn đề tương thích với nền máy PS3, Unreal Engine đã chứng kiến sự chuyển mình mạnh mẽ, đặc biệt là hình ảnh đầu ra, để mang đến cho game thủ hàng loạt trải nghiệm thú vị.

Do được viết bằng C++ nên Unreal Engine có tính cơ động cao và đồng thời là một công cụ được rất nhiều nhà phát triển game ngày nay sử dụng trên hầu hết các nền tảng hiện có như Windows, Linux, Android, iOS, Mac OS, trên máy tính cá nhân cũng như trên rất nhiều các thiết bị chơi game chuyên dụng như Dream cast, Xbox, PS.

# Unity3D ENGINE

## Giới thiệu về Unity3D Engine

Unity Engine được phát triển bởi Unity Technologies – là một gói công cụ tích hợp dùng để xây dựng Game 3D hoặc các nội dung có tính tương tác khác như mô hình kiến trúc hay hoạt hình 3D thời gian thực. Unity Engine có thể chạy trên hệ điều hành Windows và Mas OS X. Sản phẩm tạo ra từ Unity Engine có thể chạy được trên các nền Windows, Mac, Linux , Wii,iOS,Android. Bên cạnh đó, Unity Engine có khả năng phát triên Game nền Web hỗ trợ cả Mac và Windows.

Unity hay Unity 3D chắc chắn không phải engine đỉnh cao nhất về mặt đồ họa. Về mặt này, Cry Engine vẫn đang dẫn đầu với ưu thế đồ họa 3D cực kỳ chân thực. Có thể cảm nhận rõ ràng điều này qua chất lượng hình ảnh các tựa game gần đây sử dụng Cry Engine như Far Cry hay Crysis 3. Tuy nhiên, những engine khủng như Cry hay Unreal rất kén chọn và yêu cầu cấu hình cũng khủng không kém. Hơn nữa cái giá để được cấp phép sử dụng Cry Engine chắc chắn cũng không dễ chịu chút nào.

Tuy đồ họa không thể so sánh với Cry Engine nhưng chất lượng hình ảnh Unity 3D mang lại cũng quá đủ để thỏa mãn game thủ hiện nay. Hơn nữa, Unity 3D còn có những ưu điểm mà không phải engine game nào cũng có.

Hỗ trợ đa nền tảng: Một trong các thế mạnh của Unity 3D chính là khả năng hỗ trợ gần như toàn bộ các nền tảng hiện có bao gồm: PlayStation 3, Xbox 360, Wii U, iOS, Android, Windows, Blackberry 10, OS X, Linux, trình duyệt Web và cả Flash. Nói cách khác, chỉ với một gói engine, các studio có thể làm game cho bất kỳ hệ điều hành nào và dễ dàng convert chúng sang những hệ điều hành khác nhau. Đồng thời, đây cũng là giải pháp cho các game online đa nền tảng – có thể chơi đồng thời trên nhiều hệ điều hành, phần cứng khác nhau như Web, PC, Mobile, Tablet….

Dễ sử dụng: Unity 3D được built trong một môi trường phát triển tích hợp, cung cấp một hệ thống toàn diện cho các lập trình viên, từ soạn thảo mã nguồn, xây dựng công cụ tự động hóa đến trình sửa lỗi. Do được hướng đến đồng thời cả lập trình viên không chuyên và studio chuyên nghiệp, nên Unity 3D khá dễ sử dụng. Hơn nữa, đây là một trong những engine phổ biến nhất trên thế giới, người dùng có thể dễ dàng tìm kiếm kinh nghiệm sử dụng của “tiền bối” trên các forum công nghệ.

Tính kinh tế cao: Unity Technologies hiện cung cấp bản miễn phí engine Unity 3D cho người dùng cá nhân và các doanh nghiệp có doanh thu dưới 100.000 USD/năm. Với bản Pro, người dùng phải trả 1.500 USD/năm – một con số rất khiêm tốn so với những gì engine này mang lại.

Có thể thấy, tuy không phải engine “khủng” nhất, nhưng nếu xét toàn diện, Unity 3D đích thực là một trong những engine game đa nền tảng tốt nhất hiện nay.

Unity hay Unity 3D là một engine game đa nền tảng được phát triển bởi Unity Technologies. Engine này được phát triển bằng C/C++ và có khả năng hỗ trợ mã viết bằng C#, JavaScript hoặc Boo. Đây là một trong 4 engine game phổ biến nhất thế giới, hiện đã phát hành bản Unity 3D 4.5.5

## Các tính năng tiêu biểu của Unity3D Engine

Được viết dựa trên ngôn ngữ C# và JavaScript: Thiết kế theo hướng component. Là một môi trường phát triển tích hợp mạnh mẽ, có khả năng xem trước kết quả, sản phẩm có thể triển khai trên nhiều nền tảng:

+ Sử dụng được trên Windows và Mac OSX.

+ Ứng dụng chạy trên nền web (thông qua plugin Unity Web Player) trên Windows và Mac OSX.

+ Chức năng dashboard widget trên Mac OSX

+ Ứng dụng chạy trên Nintendo Wii, iOS, Android, WP 8, PS

Unity cho phép nạp các model từ 3ds Max, Maya, Blender, Mode, Zbrush, Cineme 4D, Cheetah3D.

Engine đồ họa sử dung Direct3D (Windows), OpenGL, OpenGLES và prorietary.

Về mặt dụng hình, Unity engine có những đặc điểm nổi bật sau:

• Deferred Lightning: Cho khả năng thể hiện ánh sáng vào bóng đổ cực kỳ trung thực, mọi đối tượng đều thể hiện được hiệu ứng ánh sáng cũng như bóng tối, mức tác động ở cấp từng điểm ảnh.

• 100 Kiểu tạo bóng tích hợp: từ đơn giản (Diffuse, Glossy,…) đến phức tạp (self Illuminated Bumped Specular….).

• Surface shaders: bộ phát sinh mã tự động nhằm đem lại sự đơn giản trong việc lập trình Vertex Shader và Pixel Shader. Chúng ta đơn giản chỉ cần viết mã Cg và Unity sẽ biên dịch nó thành mã dựng hình, thậm chí cũng có thể biên dịch sang GLSL để chạy trên các thiết bị đi động.

• Scalability: mở rộng khả năng tương thích với phần cứng. Unity sẽ phân tích phần cứng hiện có để bảo đảm người dùng có thể trải nghiệm được những hiệu ứng đồ họa tốt nhất.

• Full Screen Post-Processing Effects: Unity tích hợp rất nhiều hiệu ứng hậu xủa lý hình ảnh toàn màn hình như: Sun shaft, hight quality depth-of-field, lens effects, chromatic abberation, curve-driven color correction…

Nhằm tăng hiệu năng, Unity áp dụng một số kỹ thuật như sau:

• Batching: nhằm hạn chế số lệnh gọi hàm vẽ, Unity tự động kết hợp các đối tượng hình học lại thành các tổ hợp, cách làm này sẽ giúp giảm đáng kể chi phí nhưng đồng thời vẫn giữ lại được toàn bộ hiệu ứng hình ảnh.

• Rewritten Renderring Engine: Tăng sức mạnh và tính linh động của bộ kết xuất hình ảnh lên đến 50% so với các phiên bản trước.

• Occlusion Culling: Giải pháp giúp giảm chi phí cho các sản phẩm chạy trên nền console, di động và web bằng cách giảm số lượng đối tượng cần kết xuất xuống mức tối thiểu.

• GLSL Optimizer: mặc dù OpenGL ES cho phép sử dụng hiệu ứng đổ bóng trên nền di động song nó vẫn tỏ ra quá nặng so với khả năng của thiết bị. Vì vậy, Unity được trang bị bộ tối ưu GLSL Shader với khả năng tăng hiệu năng lên 2-3 lần.

Về phần chiếu sáng, Unity có những đặc điểm khá ưu việt:

• Deferred Renderer: kỹ thuật chiếu sáng hiện đại cho khả năng bao phủ toàn bản đồ, mọi nguồn sáng đều tạo ra hiệu ứng chiếu sáng thậm chí là tia sáng phát ra từ một vụ nổ.

• Realtime Shadows: Hệ thống tạo bóng đổ tiên tiến có khả năng tạo bóng từ mọi nguồn sáng, đông thời sử dụng ít tài nguyên phần cứng.

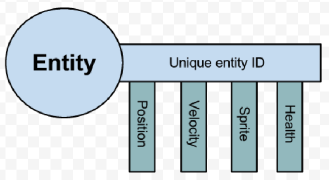
• Screen Space Ambient Occlusin (SSAO): Unity pro sử dụng công nghệ SSAO như là một công cụ tạo hiệu ứng ảnh chuẩn. Đơn giản chỉ cần thêm thành phần này vào bất kỳ góc nhìn nào và chúng ta sẽ đạt được hiệu quả tuyệt vời.

• Sun shafts & Lens Effects : Unity Pro cũng hỗ trợ thêm nhiều hiệu ứng tia sáng mặt trời, đồng thời mô phỏng hiểu ứng phản chiếu thấu kính nội và hiệu ứng lóe sáng nhằm mang lại một trải nghiệm mang tính nghê thuật cao.

• Lighting mapping: Là công nghệ duy nhất cho phếp chúng ta điều khiển không gian toàn bộ game một cách chính xác. Unity tích hợp công cụ lightmapper tốt nhất hiện nay là Beast, giúp tạo ra không gian game hoàn hảo nhất.

## Kiến trúc Component (Component Architecture)

Component Architecture là hướng lập trình hướng dữ liệu. Trong đó một đối tượng thường được gọi là entity ( thực thể ). Tất cả các entity đều giống nhau không chứa bất kỳ đặc tính riêng nào. Một entity có thể add thêm các component. Mỗi thực thể sẽ mang nhưng tính năng tính chất của component và behavior mà được thêm vào.

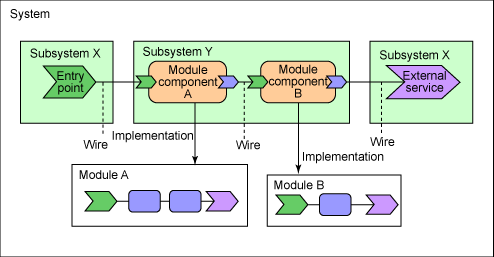


Mỗi component sẽ độc lập với nhau chỉ ảnh hướng tới nhau theo từng liên kết nhất định. Mỗi component và behavior sẽ có một hệ thống xử lý riêng. Chạy trên những nhánh độc lập với nhau.

Mỗi component thông thường sẽ có hai thành phần là

+ Attributes (data) : danh sách các dữ liệu cần thiết.

+ Behaviours (logic): các hàm xử lý dữ liệu



Có thể sử dụng các đặc điểm của OOP nhưng sẽ không phù hợp với kiến trúc của Component.

Trong Component Architecture hầu như không tạo ra một đối tượng nhất định giống như OOP mà tất cả các Object trong component đều là thực thể. Không có kiến trúc theo cây thừa kế trong Component Architecture. Tốc độ xử lý theo kiến trúc Component Architecture nhanh hơn OOP rất nhiều.

### Ưu điểm

Tốc độ xử lý cực nhanh vì mỗi component sẽ chạy trên một hệ thống system riêng biệt nhau.

Reuse code tốt. Một component có thể được sử dụng cho nhiều đối tượng khác nhau nhưng có chung một số đặc điểm về component đó. Giảm độ phức tạp của dự án.

Dễ vận hành và bão trì game: mỗi component hoàn toàn gần như là độc lập với nhau nên dựa theo kiến trúc của component thì rất dễ thích nghi với thây đổi trong game, tránh và hạn chế ảnh hưởng tới các component còn lại.

Hầu như không có cấp độ thừa kế theo cây giống cấu trúc của OOP.

### Nhược điểm

Không thích hợp với các project nhỏ.

## Component trong Unity

Unity được viết thuần theo kiến trúc Component Architecture.  
Trong Unity chỉ có một loại đối tượng duy nhất là GameObject. GameObject có khả năng được gắn thêm vào nó các component. Những component khác nhau được gắn vào với các giá trị khác nhau sẽ tạo ra những đối tượng khác nhau cần thiết cho game.

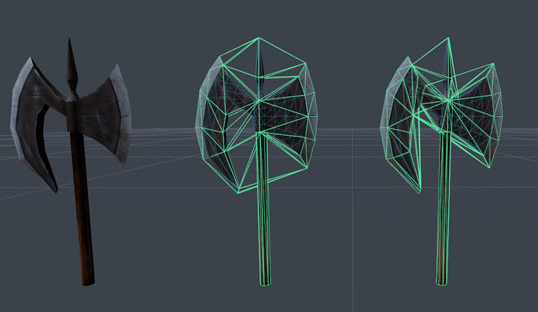
GameObject như một container dùng để chứa các component. Và không có gì khác nhau giữa các đối tượng, điểm khác nhau là các component được gắn vào với nó.

Mặc định, một GameObject mới tạo ra đã chứa một component là Transform, dùng để quản lý vị trí của GameObject đó trong không gian và quan hệ child-parent với các GameObject khác.

## Những component mặc định trong Unity

### Nhóm Mesh

Dùng để hiển thị các model 3d vào Unity.



Bao gồm các component:

1. **Mesh**

Các model 3d được import từ asset

1. **Mesh Filter**

Truyền một mesh từ asset vào cho Mesh Renderer hiển thị

1. **Text Mesh**

Một mesh đặc biệt dùng để hiển thị text trong không gian 3d

1. **Mesh Render**

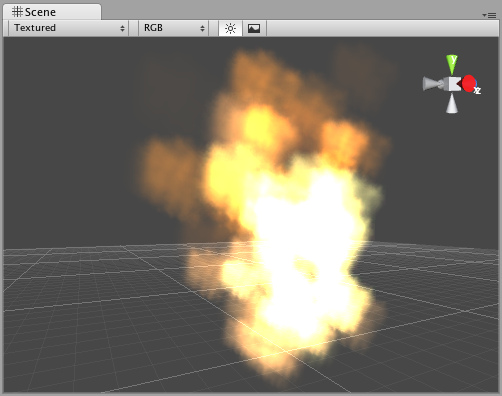
Hiển thị mesh lên không gian 3d

Các định dạng model 3d được Unity hỗ trợ gồm 2 loại:

* Các định dạng đã kết xuất: .FBX, .OBJ
* Các định dạng chưa kết xuất: .MAX, .BLEND nhận từ các phần mềm 3D Studio Max hoặc Blender

### Nhóm Effect

Tạo ra các effect trong game như: khói, hơi nước, hiệu ứng khí quyển.



Bao gồm:

1. **Particle System**

Dùng để hiện thực một hiệu ứng particle nào đó.

1. **Trail Renderer**

Tạo hiệu ứng lưu vết phía sau (đuôi) cho các đối tượng di chuyển trong không gian. Tương tự như Line Renderer sẽ nói ở sau.

1. **Line Renderer**

Tạo ra các đường trong không gian, dựa vào các điểm được xác định. Đường được vẽ không phải là các điểm mà là các texture có độ dày nhất định.

1. **Lens Flare**

Được dùng để thêm vào một nguồn sáng (Light) để tạo hiệu ứng cho nguồn sáng đó. Hiệu ứng được tạo ra giống như khi một camera quay trực tiếp vào một nguồn sáng mạnh.

1. **Halo**

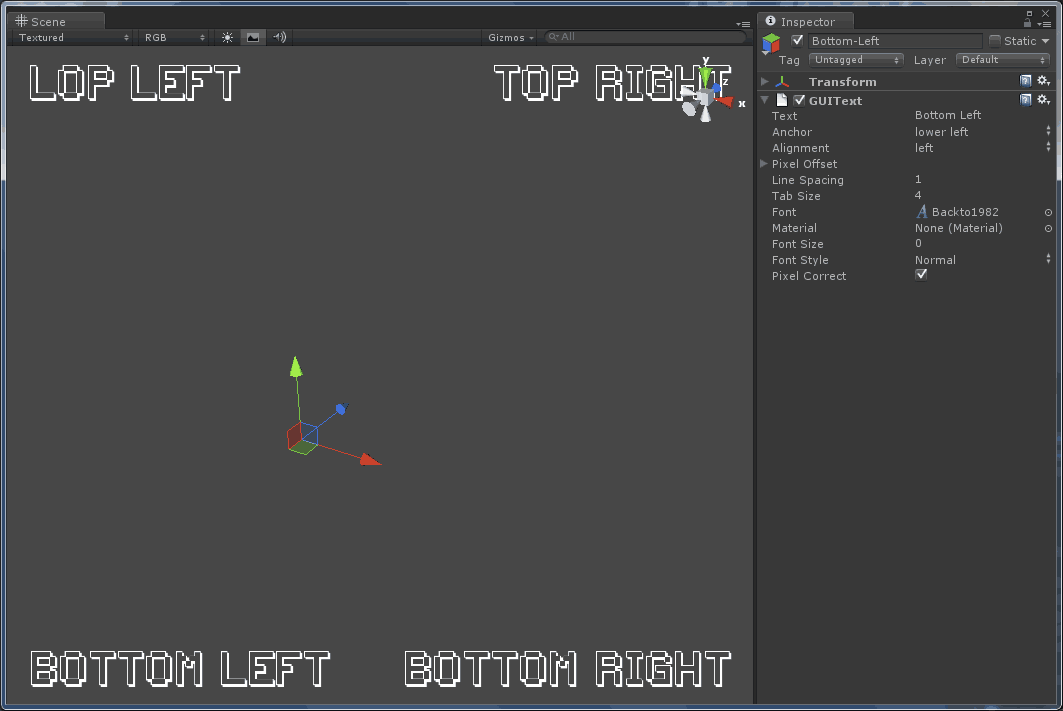
Tạo hiệu ứng vầng sáng xung quanh một đối tượng GameObject. Halo được dùng chủ yếu cho hiệu ứng của một điểm sáng (Point Light).

1. **Projector**

Tạo ra hiệu ứng đổ bóng lên một mặt phẳng.

### Nhóm Renderring

Tạo các thành phần có hiển thị trong không gian, hoặc trong giao diện người dùng.



Bao gồm:

1. **Camera**

Dùng để quay lại các GameObject có trong scene. Đây là component không thể thiếu trong game. Nếu một scene không có GameObject nào chứa component này, sẽ không có gì trong scene được thể hiện lên màn hình.

1. **Flare Layer**

Được gắn vào Camera để tạo ra hiệu ứng Lens Flare

1. **GUI Layer**

Tương tự **Flare Layer**, cần được gắn vào Camera để thể hiện các GUI Text và GUI Texture.

1. **GUI Layer**

Dùng để hiển thị Text lên màn hình 2D.

1. **GUI Texture**

Dùng để hiển thị Texture lên màn hình 2D

1. **Light**

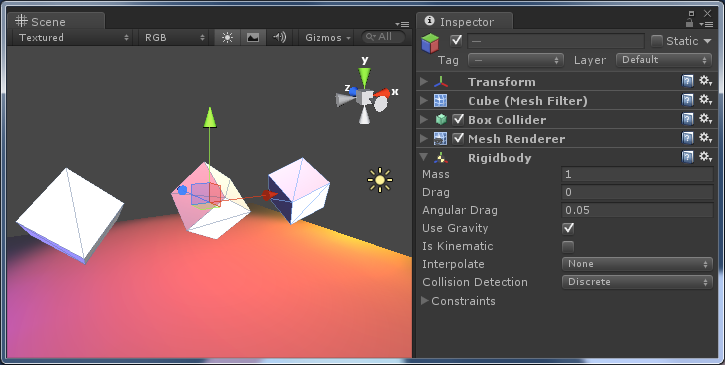
Tạo ra một nguồn sáng, chiếu sáng các thành phần hoặc toàn bộ các GameObject có trong scene.

1. **Skybox**

Dùng để giả lập một không gian vô hạn bao quanh toàn bộ scene.

### Nhóm Physics

Unity sử dụng NVIDIA PhysX để giả lập các tương tác vật lý trong game. Để tạo các chuyển động vật lý, va chạm, phản hồi, ... cần gắn một component physic vào GameObject.



Bao gồm:

1. **Rigidbody**

Cho phép các GameObject có thể chịu tác động lực, bao gồm cả trọng lực trái đất, lực cản không khí… Đối với các đối tượng tĩnh, không cần phản ứng với các tác động của lực thì không cần tới Component này.

1. **Character Controller**

Được dùng để gắn vào các đối tượng mà người chơi điều khiển trong game như: nhân vật góc nhìn thứ nhất, góc nhìn thứ 3. Đối tượng này không chịu sự tác động của các lực vật lý để đảm bảo người chơi có thể di chuyển giống thực, nhưng nó có khả năng gây ra lực lên các đối tượng khác khi va chạm.

1. **Constant Force**

Tạo ra một lực lập tức tác động liên tục lên vật ở mọi frame thay vì chỉ một frame nếu sử dụng hàm *Rigidbody.AddForce*. Được dùng nhiều cho các loại tên lửa bay, hay viên đạn bay.

1. **Collider**

Cho phép các đối tượng có thể va chạm với nhau. Nếu không có một Collider nào được gắn vào GameObject, thì các GameObject sẽ di chuyển xuyên qua nhau.

Bao gồm:

Box Collider: Khối lập phương

Sphere Collider: Khối cầu

Capsule Collider: Khối trụ có 2 đầu tròn.

Mesh Collider: Khối được tạo nên từ một Mesh.

Wheel Collider: Một dạng đặc biệt, dùng cho chiếc xe

### Nhóm Audio

Âm thanh trong game

Bao gồm:

1. **Audio Listener**

Đại diện cho một nơi để lắng nghe các âm thanh có trong không gian 3D và thường được gắn vào Camera chính. Ở một scene, chỉ được phép tồn tại một Audio Listener. Trong môi trường 2D thì có thể gắn ở bấy kỳ đâu cũng được.

1. **Audio Source**

Đại diện cho một đối tượng phát ra âm thanh trong không gian. Trong môi trường 3D, nếu một Audio Source có vị trí càng gần Audio Listener thì âm thanh sẽ càng to hơn.

### Nhóm Terrain



Tạo ra địa hình cho game như mặt đất, đường, đồi núi, cây cối, biển, sông suối,...

# TỔNG QUAN ĐỀ TÀI

## Tổng quan chức năng

Hệ thống lắng nghe mạng xã hội có thể chia ra làm hai phần, một chỉ lo nhiệm vụ thu thập, một chỉ lo nhiệm vụ truy vấn. Hai hệ thống sẽ có thể hoạt động hoàn toàn độc lập với nhau và không gây ảnh hưởng lẫn nhau khi một trong hai hệ thống có sai sót. Hơn nữa, theo nhu cầu ngày càng cao, chúng ta cũng có thể dễ dàng phân tán một phần hệ thống (thu thập hoặc truy vấn) mà không cần tốn chi phí cho cả hai.

### Hệ thống thu thập

Một hệ thống thu thập cần phải có các chức năng sau:

* Hiểu được các chuẩn kết nối của Word Wide Web.
* Đọc hiểu HTML.
* Lưu trữ dữ liệu một cách có cấu trúc.
* Mô phỏng hành động của người dùng thật xuất sắc.

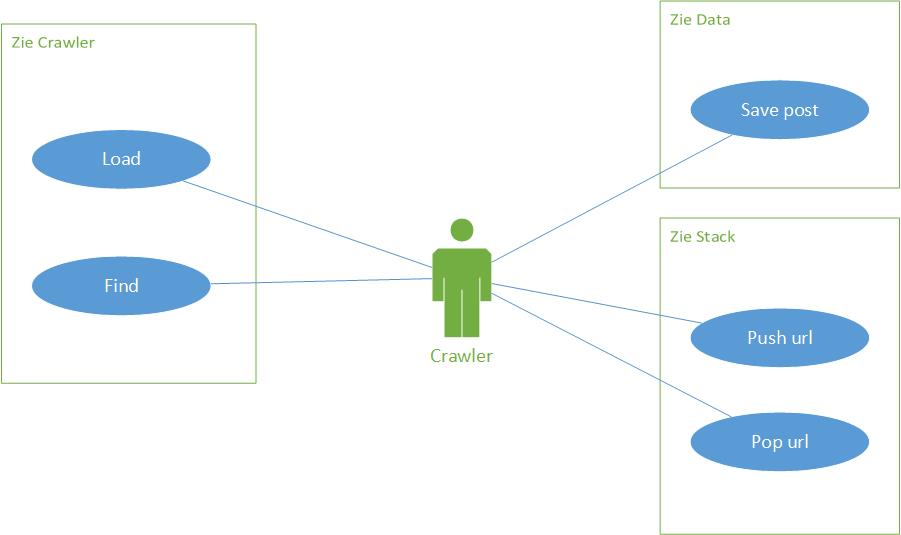
### Hệ thống báo cáo

Dựa trên đặc điểm nghiệp vụ, hệ thống báo cáo phục vụ công tác nghiên cứu thị trường cần phải đạt được các chức năng sau:

* Truy vấn dữ liệu bằng query.
* Thống kê dữ liệu theo ngày (Trend[[1]](#footnote-1)).
* Thống kê dữ liệu theo theo luồng thảo luận (Thread[[2]](#footnote-2)).
* Thống kê dữ liệu theo nguồn (Source).
* Thống kê dữ liệu theo loại mạng xã hội (Social Media Type[[3]](#footnote-3) hay SMT).

## Sơ đồ Use case

### Use case Crawler



Hình ‑ – Use case Crawler

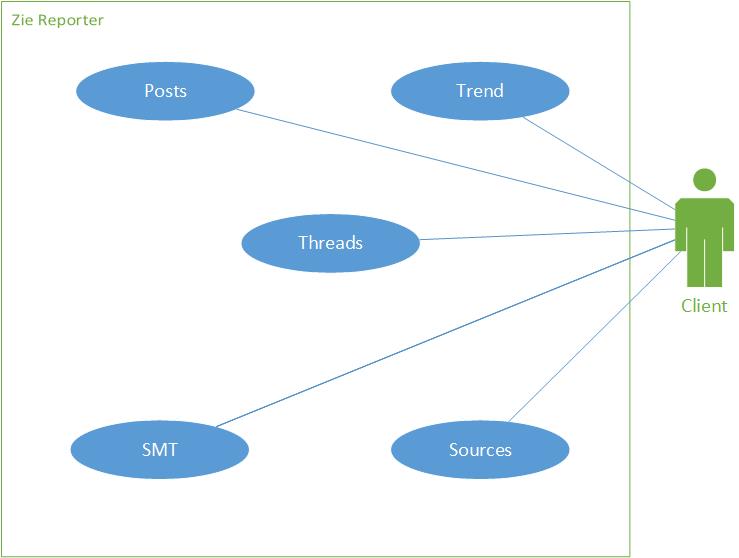
Danh sách Actor

|  |  |  |
| --- | --- | --- |
| STT | Tên Actor | Ý nghĩa/Ghi chú |
| 1 | Crawler | Crawler thực hiện thu thập dữ liệu. |

Danh sách Use case

| STT | Tên Use case | Ý nghĩa/Ghi chú |
| --- | --- | --- |
| 1 | Load | Tải một trang web từ địa chỉ url và thực hiện đọc cấu trúc HTML trả về từ trang web. |
| 2 | Find | Tìm một node HTML được trả về từ trang web thông qua hàm load. |
| 3 | Save post | Lưu một thảo luận tìm được từ việc truy cập trang web. |
| 4 | Push url | Nạp một liên kết tìm được vào hệ thống. |
| 5 | Pop url | Lấy liên kết cần xử lý tiếp theo từ hệ thống. |

### Use case Client



Hình ‑ - Use case Client

Danh sách Actor

|  |  |  |
| --- | --- | --- |
| STT | Tên Actor | Ý nghĩa/Ghi chú |
| 1 | Client | Người dùng đầu cuối hoặc một hệ thống khác truy cập tới Zie (có thể là Front-end) |

Danh sách Use case

| STT | Tên Use case | Ý nghĩa/Ghi chú |
| --- | --- | --- |
| 1 | Posts | Lấy danh sách các thảo luận. |
| 2 | Trend | Lấy danh sách số lượng các thảo luận tính theo ngày. |
| 3 | Sources | Lấy danh sách các nguồn có thảo luận đông nhất. |
| 4 | Threads | Lấy danh sách các luồng thảo luận đông nhất. |
| 5 | SMT | Lấy danh sách các loại mạng xã hội có số lượt thảo luận đông nhất. |

Actor Client ở đây có thể là người dùng thực sự, thông thường là admin hoặc người có khả năng quản trị hệ thống thao tác trực tiếp với Zie. Hoặc trong trường hợp cá biệt, một người dùng đầu cuối có thể được cấp quyền sử dụng Zie.

Trong trường hợp thông thường, Zie sẽ là một hệ thống ngầm và Actor Client là một hệ thống khác giao tiếp trực tiếp với User và nhận dữ liệu xử lý từ Zie. Client không nhất thiết phải cài đặt trên cùng một máy với Zie.

### Use case Save post

| Tên Use case | Save post | |
| --- | --- | --- |
| Mục đích sử dụng | Lưu một thảo luận vừa được thu thập | |
| Mô tả | Crawler thực hiện lưu trữ một thảo luận vừa thu thập được hoặc thông qua giao diện, người dùng thực hiện lưu trữ một thảo luận. | |
| Tác nhân chính | Crawler. | |
| Các tác nhân khác | Người dùng đầu cuối | |
| Điều kiện/trạng thái hệ thống khi bắt đầu Use case | Không có. | |
| Kết quả mong đợi | Id của thảo luận là một số lớn hơn 0 vừa được lưu trữ vào hệ thống. | |
| Kết quả khác | Kết quả | Nguyên nhân |
| -1 | Lệnh gửi tới không đúng cú pháp. |
| -2 | Không thể truy cập tới cơ sở dữ liệu. |
| -3 | Không đủ tham số. |
| -4 | Định dạng ngày tháng không chính xác. |
| Dòng sự kiện chính | Crawler truy cập vào website cần thu thập dữ liệu.  Crawler phân tích dữ liệu và nạp vào hệ thống các thảo luận vừa nhận diện được. | |
| Dòng sự kiện khác | Không có. | |
| Yêu cầu đặc biệt | Không có. | |
| Mức độ ưu tiên | 1 | |

### Use case Push url

| Tên Use case | Push url | |
| --- | --- | --- |
| Mục đích sử dụng | Nạp một url vừa phát sinh vào stack | |
| Mô tả | Crawler xử lý một trang trong website và phát hiện được các liên kết tới những trang khác trong website và thực hiện nạp các liên kết đó vào hệ thống. | |
| Tác nhân chính | Crawler. | |
| Các tác nhân khác | Không có. | |
| Điều kiện/trạng thái hệ thống khi bắt đầu Use case | Không có. | |
| Kết quả mong đợi | Id của url vừa được nạp vào hệ thống là một số lớn hơn 0. | |
| Kết quả khác | Kết quả | Nguyên nhân |
| -1 | Lệnh gửi tới không đúng cú pháp. |
| -2 | Không thể truy cập tới cơ sở dữ liệu. |
| -3 | Không đủ tham số. |
| Dòng sự kiện chính | Crawler truy cập vào website cần thu thập dữ liệu.  Crawler phân tích dữ liệu và nạp vào hệ thống các liên kết tới các website khác. | |
| Dòng sự kiện khác | Người dùng nạp vào hệ thống một đường dẫn chưa được phát hiện bởi Crawler.  Đường dẫn ngày lập tức được nạo vào hệ thống và được xử lý ngay bởi Crawler. | |
| Yêu cầu đặc biệt | Không có. | |
| Mức độ ưu tiên | 1 | |

### Use case Pop url

| Tên Use case | Pop url | |
| --- | --- | --- |
| Mục đích sử dụng | Lấy url tiếp theo sẽ xử lý theo thứ tự ưu tiên của Zie Stack. | |
| Mô tả | Crawler hoàn tất xử lý một trang web và gửi yêu cầu lên hệ thống hỏi địa chỉ trang web tiếp theo cần xử lý. | |
| Tác nhân chính | Crawler. | |
| Các tác nhân khác | Không có. | |
| Điều kiện/trạng thái hệ thống khi bắt đầu Use case | Không có. | |
| Kết quả mong đợi | Id của url, url và referer của url tiếp theo cần được xử lý. | |
| Kết quả khác | Kết quả | Nguyên nhân |
| 0 | Tạm thời chưa có url mới cần được xử lý |
| Dòng sự kiện chính | Crawler khởi động hoặc sau khi hoàn tất xử lý một trang web. Crawler tiếp tục hỏi hệ thống địa chỉ trang web kế tiếp cần xử lý là gì. | |
| Dòng sự kiện khác | Không có. | |
| Yêu cầu đặc biệt | Không có. | |
| Mức độ ưu tiên | 1 | |

### Use case Posts

| Tên Use case | Posts | |
| --- | --- | --- |
| Mục đích sử dụng | Xuất dữ liệu là các thảo luận trong một khoảng thời gian quy định và thỏa một query do người dùng nhập vào. | |
| Mô tả | Người dùng là nhà nghiên cứu sử dụng hệ thống Front-end để nhập query, chọn thời gian cần truy xuất dữ liệu và tạo một báo cáo dữ liệu là những thảo luận trong khoảng thời gian được chọn thỏa query của người dùng. | |
| Tác nhân chính | Người dùng. | |
| Các tác nhân khác | Không có. | |
| Điều kiện/trạng thái hệ thống khi bắt đầu Use case | Không có. | |
| Kết quả mong đợi | Danh sách các thảo luận thỏa điều kiện | |
| Kết quả khác | Kết quả | Nguyên nhân |
| 1 | Search engine không trả về kết quả và gửi kèm lỗi. |
| Kết quả rỗng | Không có thảo luận phù hợp trong khoảng thời gian lựa chọn. |
| Dòng sự kiện chính | Người dùng truy xuất vào front-end và thực hiện tạo một báo cáo mới có kiểu báo cáo là Message. | |
| Dòng sự kiện khác | Không có. | |
| Yêu cầu đặc biệt | Không có. | |
| Mức độ ưu tiên | 1 | |

### Use case Trend

| Tên Use case | Trend | |
| --- | --- | --- |
| Mục đích sử dụng | Xuất dữ liệu là số lượng thảo luận trong một khoảng thời gian quy định tính theo ngày và thỏa một query do người dùng nhập vào. | |
| Mô tả | Người dùng là nhà nghiên cứu sử dụng hệ thống Front-end để nhập query, chọn thời gian cần truy xuất dữ liệu và tạo một báo cáo dữ liệu là số lượng những thảo luận trong khoảng thời gian được chọn tính theo ngày thỏa query của người dùng. | |
| Tác nhân chính | Người dùng. | |
| Các tác nhân khác | Không có. | |
| Điều kiện/trạng thái hệ thống khi bắt đầu Use case | Không có. | |
| Kết quả mong đợi | Danh sách các các ngày và số lượng thảo luận thỏa điều kiện trong các ngày đó. | |
| Kết quả khác | Kết quả | Nguyên nhân |
| 1 | Search engine không trả về kết quả và gửi kèm lỗi. |
| Kết quả rỗng | Không có thảo luận nào trong thời gian phù hợp để thực hiện thống kê. |
| Dòng sự kiện chính | Người dùng truy xuất vào front-end và thực hiện tạo một báo cáo mới có kiểu báo cáo là Trend. | |
| Dòng sự kiện khác | Không có. | |
| Yêu cầu đặc biệt | Không có. | |
| Mức độ ưu tiên | 1 | |

### Use case Sources

| Tên Use case | Sources | |
| --- | --- | --- |
| Mục đích sử dụng | Xuất dữ liệu là top 10 website có số lượng thảo luận trong một khoảng thời gian quy định và thỏa một query do người dùng nhập vào cao nhất. | |
| Mô tả | Người dùng là nhà nghiên cứu sử dụng hệ thống Front-end để nhập query, chọn thời gian cần truy xuất dữ liệu và tạo một báo cáo dữ liệu là những website có số lượt thảo luận cao nhất. | |
| Tác nhân chính | Người dùng. | |
| Các tác nhân khác | Không có. | |
| Điều kiện/trạng thái hệ thống khi bắt đầu Use case | Không có. | |
| Kết quả mong đợi | Danh sách các website có các thảo luận thỏa điều kiện nhiều nhất | |
| Kết quả khác | Kết quả | Nguyên nhân |
| 1 | Search engine không trả về kết quả và gửi kèm lỗi. |
|  | Kết quả rỗng | Không có thảo luận nào trong thời gian phù hợp để thực hiện thống kê. |
| Dòng sự kiện chính | Người dùng truy xuất vào front-end và thực hiện tạo một báo cáo mới có kiểu báo cáo là Source. | |
| Dòng sự kiện khác | Không có | |
| Yêu cầu đặc biệt | Không có. | |
| Mức độ ưu tiên | 1 | |

### Use case Threads

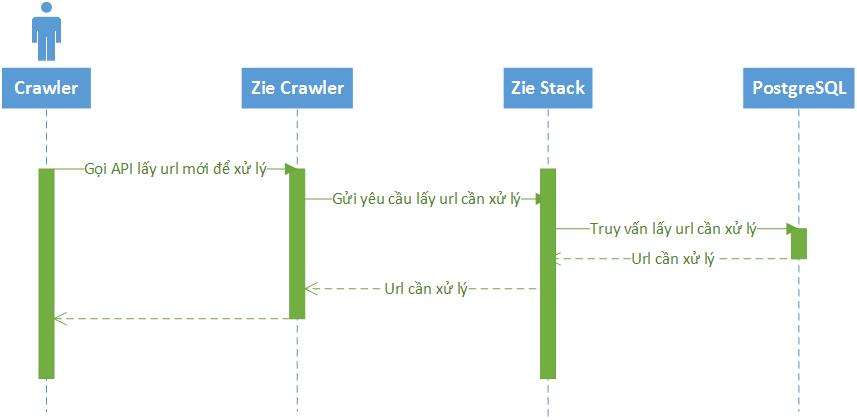
| Tên Use case | Threads | |
| --- | --- | --- |
| Mục đích sử dụng | Xuất dữ liệu là các luồng có số lượng thảo luận cao nhất trong một khoảng thời gian quy định và thỏa một query do người dùng nhập vào. | |
| Mô tả | Người dùng là nhà nghiên cứu sử dụng hệ thống Front-end để nhập query, chọn thời gian cần truy xuất dữ liệu và tạo một báo cáo dữ liệu là những luồng có số lượng thảo luận cao nhất trong khoảng thời gian được chọn thỏa query của người dùng. | |
| Tác nhân chính | Người dùng. | |
| Các tác nhân khác | Không có. | |
| Điều kiện/trạng thái hệ thống khi bắt đầu Use case | Không có. | |
| Kết quả mong đợi | Danh sách các luồng thỏa điều kiện | |
| Kết quả khác | Kết quả | Nguyên nhân |
| 1 | Search engine không trả về kết quả và gửi kèm lỗi. |
| Kết quả rỗng | Không có thảo luận phù hợp trong thời gian được chọn để thực hiện thống kê. |
| Dòng sự kiện chính | Người dùng truy xuất vào front-end và thực hiện tạo một báo cáo mới có kiểu báo cáo là Thread. | |
| Dòng sự kiện khác | Không có. | |
| Yêu cầu đặc biệt | Không có. | |
| Mức độ ưu tiên | 1 | |

### Use case SMT (Social Media Type)

| Tên Use case | SMT | |
| --- | --- | --- |
| Mục đích sử dụng | Xuất dữ liệu là các loại mạng xã hội có số lượng thảo luận cao nhất trong một khoảng thời gian quy định và thỏa một query do người dùng nhập vào. | |
| Mô tả | Người dùng là nhà nghiên cứu sử dụng hệ thống Front-end để nhập query, chọn thời gian cần truy xuất dữ liệu và tạo một báo cáo dữ liệu là những loại mạng xã hội có số lượng thảo luận cao nhất trong khoảng thời gian được chọn thỏa query của người dùng. | |
| Tác nhân chính | Người dùng. | |
| Các tác nhân khác | Không có. | |
| Điều kiện/trạng thái hệ thống khi bắt đầu Use case | Không có. | |
| Kết quả mong đợi | Danh sách các loại mạng xã hội thỏa điều kiện | |
| Kết quả khác | Kết quả | Nguyên nhân |
| 1 | Search engine không trả về kết quả và gửi kèm lỗi. |
| Kết quả rỗng | Không có thảo luận phù hợp trong thời gian được chọn để thực hiện thống kê. |
| Dòng sự kiện chính | Người dùng truy xuất vào front-end và thực hiện tạo một báo cáo mới có kiểu báo cáo là SMT. | |
| Dòng sự kiện khác | Không có. | |
| Yêu cầu đặc biệt | Không có. | |
| Mức độ ưu tiên | 1 | |

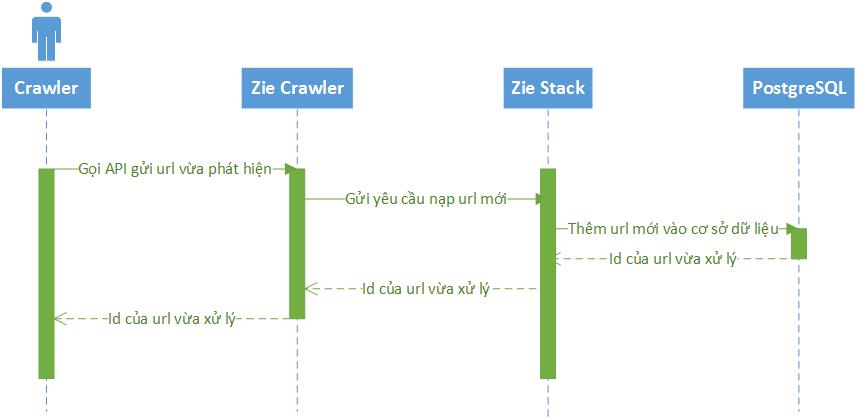
## Sơ đồ tuần tự

### Sơ đồ tuần tự Use case Pop url



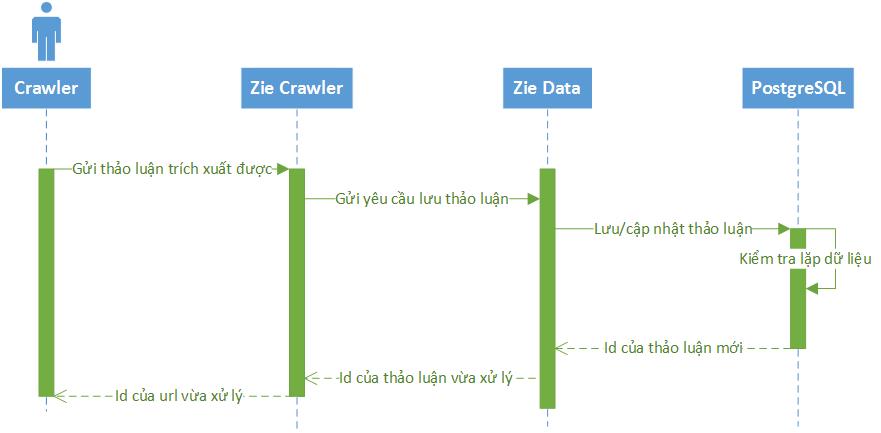
Hình ‑ - Sơ đồ tuần tự Use case Pop url

### Sơ đồ tuần tự Use case Push url



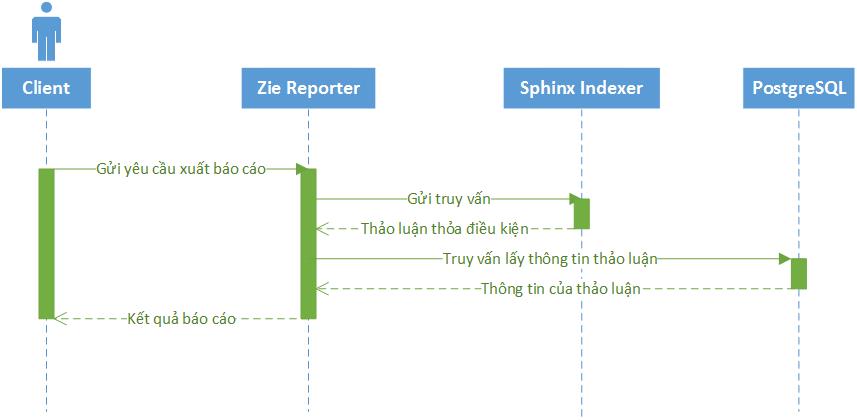
Hình ‑ - Sơ đồ tuần tự Use case Push url

### Sơ đồ tuần tự Use case Save post



Hình ‑ - Sơ đồ tuần tự Use case Save Post

### Sơ đồ tuần tự Use case Posts



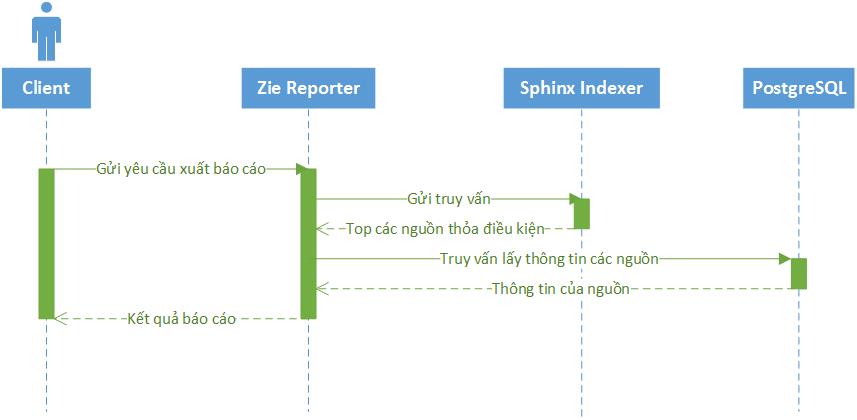
Hình ‑ - Sơ đồ tuần tự Use case Post

### Sơ đồ tuần tự Use case Trend



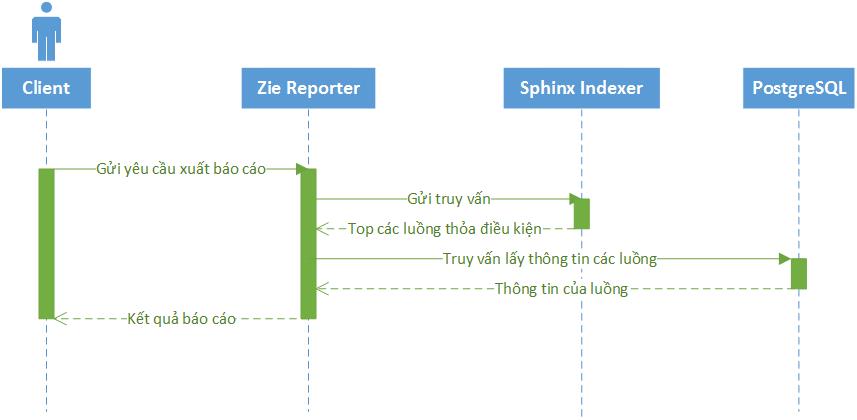
Hình ‑ - Sơ đồ tuần tự Use case Trend

### Sơ đồ tuần tự Use case Sources



Hình ‑ - Sơ đồ tuần tự Use case Sources

### Sơ đồ tuần tự Use case Threads



Hình ‑ - Sơ đồ tuần tự Use case Threads

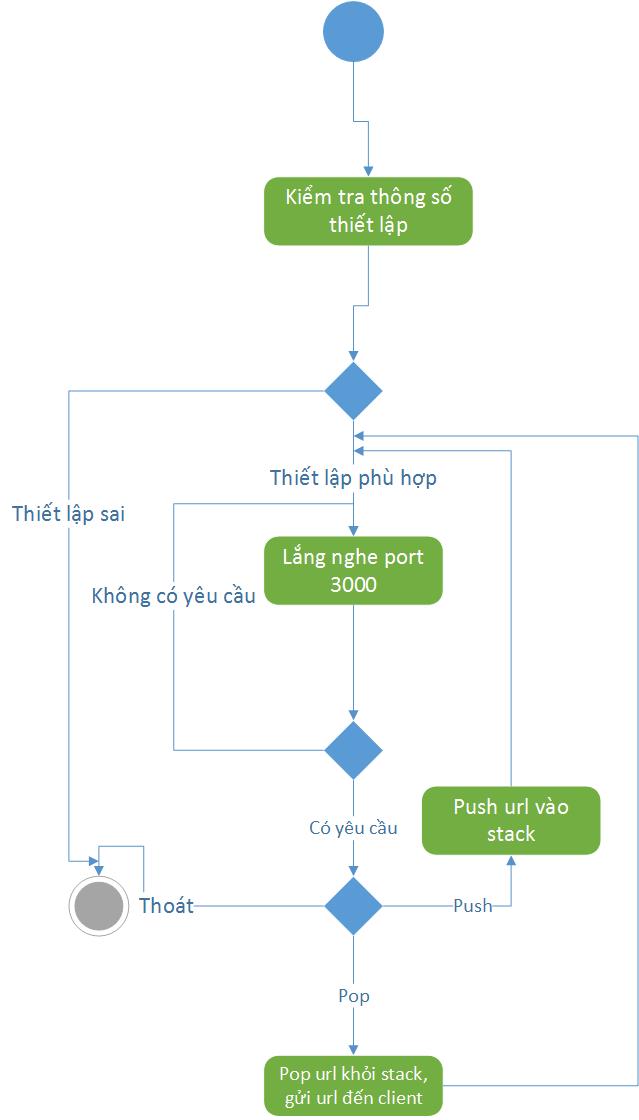
### Sơ đồ tuần tự Use case SMT



Hình ‑ - Sơ đồ tuần tự Use case SMT

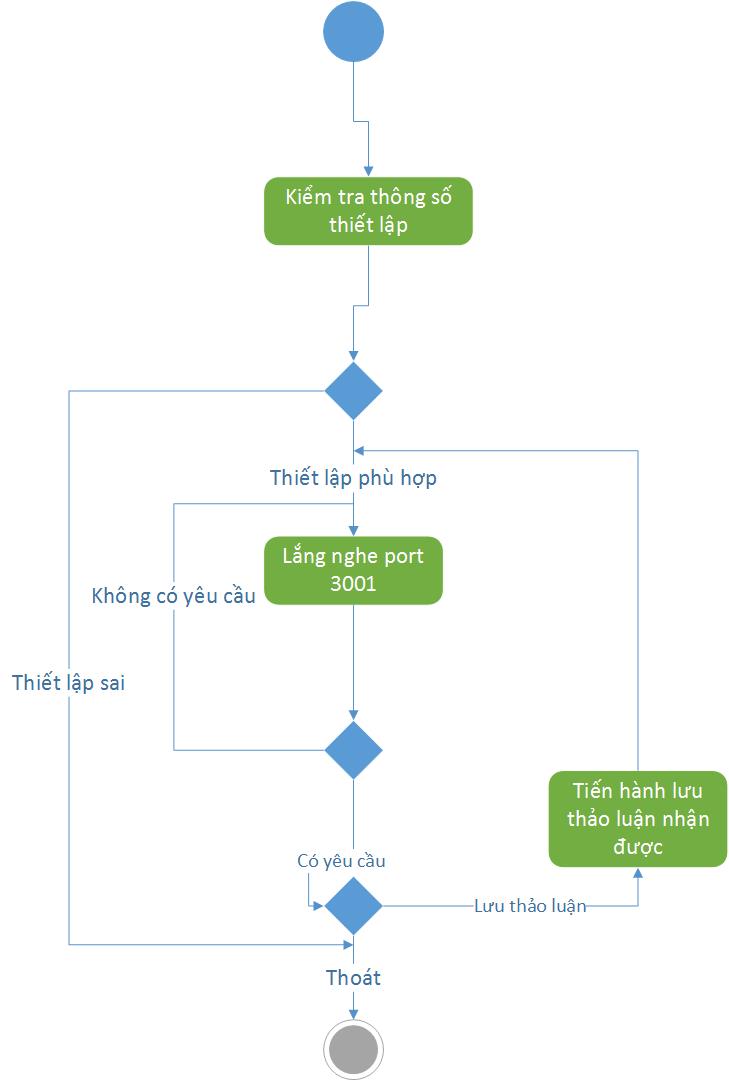
## Sơ đồ hoạt động

### Zie Stack



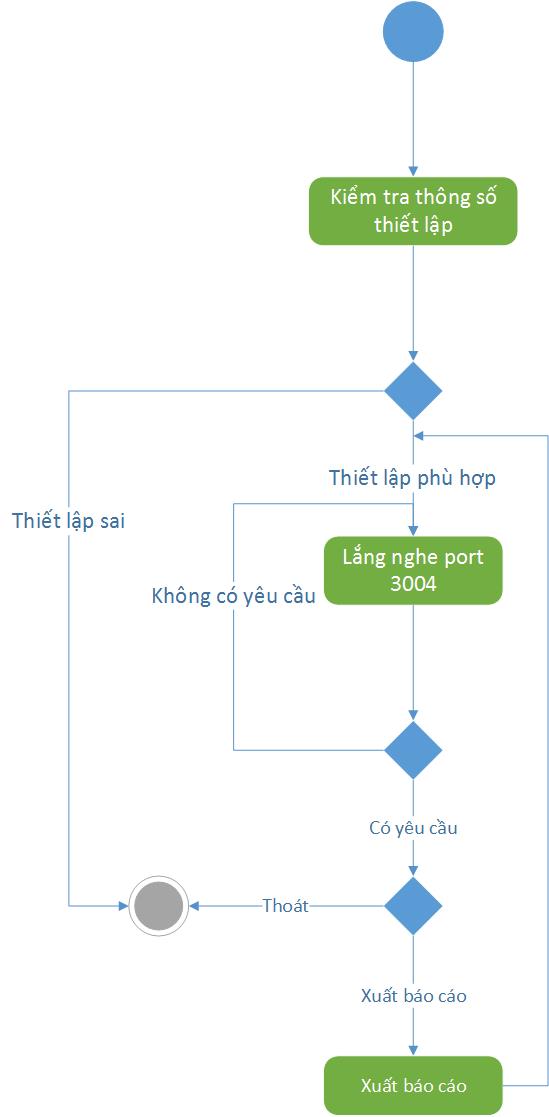
Hình ‑ - Sơ đồ hoạt động Zie Stack

### Zie Data



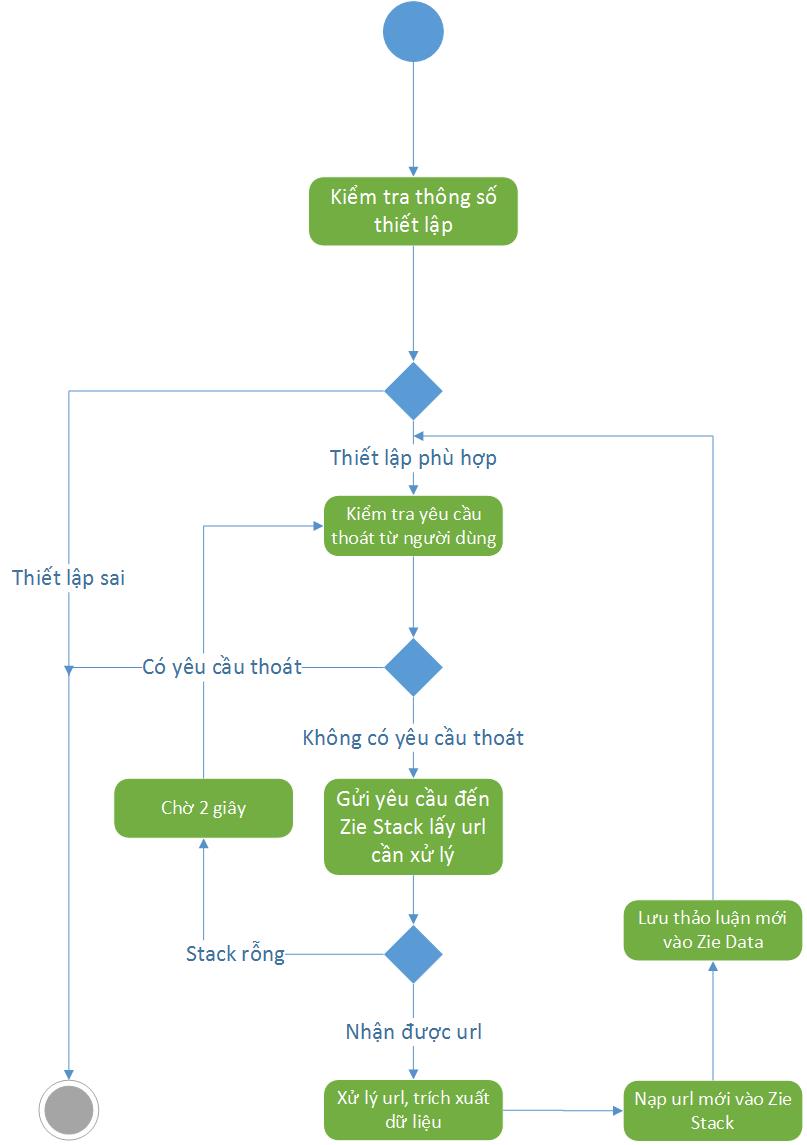
Hình ‑ - Sơ đồ hoạt động Zie Data

### Sơ đồ hoạt động Zie Reporter



Hình ‑ - Sơ đồ hoạt động Zie Reporter

### Sơ đồ hoạt động Zie Crawler



Hình ‑ - Sơ đồ hoạt động Zie Crawler

# HƯỚNG TIẾP CẬN

## Một hệ thống duy nhất vừa đảm nhiệm lưu trữ, vừa đảm nhiệm truy xuất

Phương pháp sử dụng một hệ thống duy nhất dùng cho cả việc lưu trữ và truy xuất dữ liệu có điểm lợi là tiết kiệm được chi phí cho thiết bị lưu trữ, mỗi thể hiện chỉ được lưu trữ một lần duy nhất, có khả năng hoạt động theo thời gian thực.

Tuy nhiên, hệ thống có điểm yếu là hiệu suất của cả quá trình lưu trữ và truy xuất sẽ bị ảnh hưởng. Hệ thống lưu trữ phải thường xuyên index lại dữ liệu dành cho hệ thống truy xuất. Hệ thống truy xuất cần phải kết nối dữ liệu dạng thô do cần cho việc cập nhật lại thông tin ở hệ thống lưu trữ.

Để đáp ứng được nhu cầu cần có một hệ thống quản trị cơ sở dữ liệu quan hệ mạnh, hỗ trợ Full-text nâng cao, hỗ trợ lưu trữ dữ liệu thu thập khổng lồ.

Hệ quản trị cơ sở dữ liệu đề xuất: Oracle[[4]](#footnote-4), PostgreSQL[[5]](#footnote-5).

Hệ quản trị cơ sở dữ liệu Oracle hỗ trợ Oracle Text tương tự Full-text search, hỗ trợ các toán tử boolean, hàm tìm kiếm NEAR[[6]](#footnote-6) và khả năng tìm kiếm dựa trên từ điển đồng nghĩa (tuy nhiên chưa hỗ trợ tiếng Việt). Ngoài ra Oracle bản Enterprise hỗ trợ chức năng Partitioning phù hợp để lưu trữ dữ liệu với dung lượng lớn.

Hệ quản trị cơ sở dữ liệu PostgreSQL hỗ trợ Full-text search, tuy nhiên tại version hiện tại (9.3) PostgreSQL mới chỉ hỗ trợ các toán tử boolean. Chưa đáp ứng được nhu cầu truy vấn dữ liệu nâng cao. Tương tư Oracle, PostgreSQL hỗ trợ Partitioning tuy nhiên phức tạp hơn trong cài đặt và có khả năng tùy biến cao hơn.

## Hai hệ thống, một đảm nhiệm lưu trữ, một đảm nhiệm truy xuất

Sử dụng hai hệ thống một cho truy xuất và còn lại cho lưu trữ sẽ có thể tối ưu hiệu suất cho từng hệ thống. Một hệ thống lưu trữ không cần phải index dữ liệu như một hệ thống chỉ truy xuất. Một hệ thống chỉ truy xuất không cần phải lưu trữ dữ liệu dạng thô như hệ thống chỉ lưu trữ.

Hướng tiếp cận hai hệ thống có điểm mạnh là tăng cường hiệu suất nhưng có điểm yếu là cần gấp đôi chi phí cho thiết bị lưu trữ. Sẽ có một độ trễ nhất định giữa hệ thống lưu trữ và hệ thống truy xuất trong quá trình đồng bộ. Độ phức tạp cao.

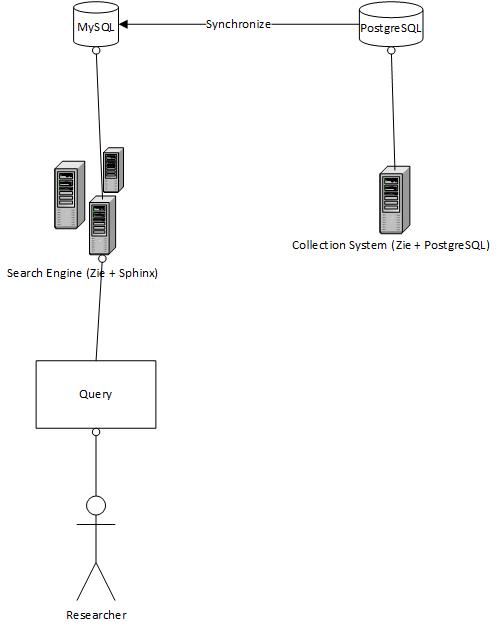
Hệ thống lưu trữ đề xuất: Oracle, PostgreSQL.

Hệ thống truy xuất đề xuất: Sphinx, Lucence.

Trong mô hình hai hệ thống, điểm mạnh hỗ trợ Full-text search[[7]](#footnote-7) nâng cao của Oracle không còn giá trị nữa, thay vào đó là chi phí cao cho bản Enterprise để có được chức năng Partitioning. Trong khi đó, với PostgreSQL chi phí là bằng không.

Sphinx là hệ thống tìm kiếm mạnh, được viết bằng C++ và hỗ trợ trực tiếp MySQL[[8]](#footnote-8), PostgreSQL. Hỗ trợ các toán tử boolean, hàm NEAR và các toán tử phức tạp khác. Có tốc độ truy xuất và index dữ liệu cực nhanh.

Lucence là hệ thống tìm kiếm được viết bằng Java (đã được port qua các ngôn ngữ khác). Lucence hỗ trợ các toán tử tìm kiếm phức tạp. Tuy nhiên, việc kết hợp giữa Lucence và PostgreSQL không đơn giản như giữa Sphinx và PostgreSQL.



Hình ‑ - Sơ đồ mô tả hai hệ thống tách rời (Sphinx + Postgres)

## Đánh giá liên kết[[9]](#footnote-9) dựa trên liên kết đến chính nó

Các liên kết hay siêu liên kết trong một website sẽ dẫn đến một trang chứa nội dung hoặc đến gần trang chứa nội dung mà website muốn gửi đến người đọc. Thông qua liên kết, người dùng website có thể được dẫn đến nơi chứa nội dung bài viết mà người dùng cần. Chính vì vậy, việc đi theo liên kết và đánh giá một liên kết dựa trên các liên kết khác đến nó là có cơ sở.

Ví dụ: Google đánh giá một website có thứ hạng cao dựa vào số lượng website liên kết đến website đó.

Nếu đánh giá liên kết dựa trên số lượng liên kết đến chính nó, các liên kết được dẫn đến nhiều sẽ được ưu tiên xử lý trước và các liên kết ít được dẫn đến hơn sẽ ít được xử lý hơn. Cách đánh giá này tuy có cơ sở nhưng dùng đến đánh giá từng trang trong website thì không chính xác. Trong một website, trang chủ thường được dẫn đến bởi tất cả các trang thành phần, chính vì vậy, trang chủ sẽ có thứ hạng cao nhất, tiếp đến là các trang chuyên mục rồi mới đến các trang nội dung. Theo thứ tự như vậy, hệ thống sẽ đọc được dữ liệu cuối cùng, và sẽ ưu tiên phát tán (đi vào các liên kết tìm được) hơn.

## Đánh giá liên kết dựa trên độ mới của nội dung

Các liên kết chứa nội dung mới nhất sẽ được ưu tiên đưa lên hàng đầu và được ưu tiên xử lý trước. Để đánh giá được chính xác, các liên kết đó cần phải được xử lý và có dữ liệu chứa trong liên kết hoặc phải tìm được cơ sở xác định độ mới của liên kết.

Đánh giá dựa trên độ mới sẽ sinh ra hai trường hợp: một là liên kết chứa dữ liệu được đánh giá chính xác bằng thời gian của dữ liệu đó (thường là thảo luận trong trang web đó), hai là liên kết chưa từng được truy xuất và không xác định được chính xác độ mới của dữ liệu chứa trong liên kết.

Liên kết có thể được đánh giá chính xác nếu có dữ liệu, và có thể được dự đoán nếu chưa có dữ liệu. Giải thuật dự đoán sẽ ảnh hưởng trực tiếp đến chất lượng hệ thống, một hệ thống sử dụng tốt tài nguyên sẽ truy cập tối đa đến những trang mang lại dữ liệu có ích.

Việc đánh giá liên kết có thể dựa trên độ mới của liên kết sinh ra liên kết đó hoặc độ mới của liên kết được sinh ra bởi liên kết đó hoặc chính một thông tin nào đó trong liên kết.

## Thu thập dữ liệu theo chiều sâu

Thu thập dữ liệu là bài toán phức tạp vì cần đảm bảo được dữ liệu là độc lập, không trùng lắp, dữ liệu phải đủ đáp ứng được yêu cầu của chiến dịch nghiên cứu.

Đối với thu thập dữ liệu theo chiều sâu, hệ thống sẽ thu thập dữ liệu của một website từ mới nhất tới cũ nhất, sau đó mới tiến hành thu thập trên website khác, cũng như cập nhật lại.

Thu thập dữ liệu theo chiều sâu giúp hệ thống thu thập đầy đủ dữ liệu của cả website nguồn từ khi website phát triển, khả năng bỏ sót dữ liệu là rất thấp. Tuy nhiên, để có được một hệ thống đủ mạnh để bao phủ 100% hàng loạt các website lớn khác khiến chi phí vượt lên rất cao. Tính đến tháng 12 năm 2013, website hdvietnam.com có hơn 4 triệu bài viết, con số đó ở tinhte.vn và vozforums.com là 18 triệu và 30 triệu bài viết.

## Thu thập dữ liệu theo chiều rộng

Thu thập dữ liệu theo chiều rộng đòi hỏi phải có một đánh giá các liên kết trước khi quyết định thu thập (liên kết nào cần trước, liên kết nào cần sau), đòi hỏi hệ thống phải có sự phân phối dữ liệu cần thu thập chính xác nếu không muốn hệ thống chỉ thu thập được những dữ liệu vô ích.

Thu thập dữ liệu theo chiều rộng giúp hệ thống bao phủ đều các website nguồn. Sức mạnh của hệ thống yêu cầu là không quá cao, chỉ cần đáp ứng được số lượng bài viết mới xuất hiện hàng ngày trên các website nguồn. Tuy nhiên, thu thập theo chiều rộng sẽ đánh đồng tất cả các website nguồn, khó thu thập tập trung trên một website nào đó theo nhu cầu của chiến dịch nghiên cứu thị trường, đòi hỏi chi phí cho giải thuật đánh giá liên kết hay nguồn dữ liệu, hơn nữa, nhu cầu dữ liệu cho các chiến dịch nghiên cứu là khác nhau, thu thập theo chiều rộng đòi hỏi phải xem các chiến dịch nghiên cứu có nhu cầu dữ liệu như nhau.

## Kết luận

* Sử dụng một hệ thống duy nhất đơn giản hóa quy trình xây dựng hệ thống, tuy nhiên sẽ làm giảm hiệu suất hệ thống, chi phí tăng cường cho bản quyền Oracle Enterprise[[10]](#footnote-10).
* Sử dụng hai hệ thống có độ phức tạp nhất định, hiệu suất hệ thống được đảm bảo. Chi phí thừa dành cho hệ thống chỉ là cần thêm thiết bị lưu trữ so với một hệ thống.
* Đánh giá liên kết dựa trên độ mới của liên kết sẽ có thể dẫn hệ thống tới những nơi chứa dữ liệu có ích cao hơn dựa trên liên kết đến chính nó.
* Thu thập dữ liệu kết hợp cả hai hướng theo chiều rộng và theo chiều sâu. Trong đó, chiều sâu cần được giới hạn và đánh giá dựa trên độ phủ của chiều rộng, việc thu thập theo chiều sâu khi chiều rộng đã đảm bảo độ phủ sẽ tối ưu hóa hiệu suất của hệ thống mà vẫn đảm bảo nhu cầu dữ liệu mới nhất.

Từ những tiếp cận trên, nhóm đã đi đến quyết định sử dụng mô hình Sphinx-PostgreSQL làm cơ sở cho hệ thống Zie, đánh giá liên kết bằng độ mới dữ liệu mà liên kết đó đem lại kết hợp dự đoán độ mới của liên kết, đồng thời kết hợp thu thập theo chiều sâu và chiều rộng.

# NHỮNG VẤN ĐỀ GẶP PHẢI KHI HIỆN THỰC HỆ THỐNG ZIE

Để dễ dàng trong việc trình bày những vấn đề có thể gặp phải khi hiện thực hệ thống Zie, đề tài cần thống nhất một số khái niệm sau để tránh gây nhầm lẫn.:

* Gọi các website nguồn là Source[[11]](#footnote-11).
* Những thành phần truy xuất, xử lý website và chuyển thành dữ liệu thô là Crawler.
* Hệ thống lưu trử, xử lý dữ liệu thô đưa thành dữ liệu trực quan là Zie.
* Mỗi đơn vị bài viết của một người dùng vào một thời gian nhất định và phân biệt hoàn toàn với các bài viết khác mà hệ thống thu thập sẽ được gọi là Post[[12]](#footnote-12).

### Lấy dữ liệu từ Source

#### Đặt vấn đề

Mỗi website sẽ có hàng loạt những thông tin, những bài viết của người dùng, chúng ta có thể chia nhỏ thông tin ra theo URL (Uniform Resource Locator). Mỗi URL có thể chứa một hay nhiều bài viết mà chúng ta cần lấy. Đơn vị nhỏ nhất trong Zie phải là Post chứ không thể là URL. Vấn đề này đòi hỏi hệ thống phải có khả năng tạo những kết nối tới các giao thức thông dụng được dùng trong Word Wide Web. Đồng thời, phải có khả năng biên dịch các mã web (HTML, XML) để phân tích được dữ liệu.

Mỗi URL sau một thời gian nhất định sẽ có thể bị thay đổi dữ liệu, dữ liệu bị thay đổi có thể là thêm mới hoặc cập nhật lại, ví dụ cùng một URL trong forum, có thể hôm nay có 3 bài viết, nhưng 5 phút sau sẽ có thêm người viết bài dẫn đến có 4 bài viết. Trong trường hợp này, Zie cũng cần phải nhận biết và có khả năng cập nhật dữ liệu mới.

#### Hướng giải quyết

Từ yêu cầu xử lý dữ liệu dẫn đến chúng ta phải hình thành một thành phần là Crawler, Crawler sẽ biên dịch được các trang web. Vấn đề biên dịch trang web là hoàn toàn khả thi, chúng ta chỉ cần lưu trữ được trang web ở dạng DOM[[13]](#footnote-13) (Document Object Model). Có rất nhiều hệ thống, thư viện hỗ trợ việc này, ngoài ra, dành cho những trường hợp đặc biệt, ta có thể đưa về xử lý dạng chuỗi với dữ liệu trả về từ Source vốn chỉ là một chuỗi các mã HTML có thể đọc được dễ dàng và xử lý rất tốt bằng Regular Expresion[[14]](#footnote-14).

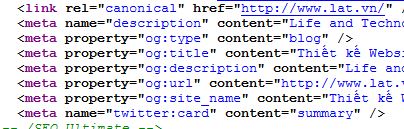
Sau một thời gian phù hợp Zie cần phải duyệt lại các URL đã từng xử lý, để cập nhật mới. Thời gian này phải được tính toán phù hợp để cân bằng giữa việc thu thập các URL mới và cập nhật các URL đã có.

### Nhận diện Post

#### Đặt vấn đề

Khi lưu trữ được một Post, sẽ có trường hợp hệ thống sẽ truy xuất lại và lại thu thập cùng Post đó. Nguyên nhân có thể do Source thay đổi cấu trúc URL hoặc cập nhật lại dữ liệu.

Post không thể được nhận diện qua URL vì nhiều URL có thể đưa đến cùng một nội dung (URL redirection). Tuy Word Wide Web đã đưa ra chuẩn các website có nhiều hơn một URL dẫn đến một nội dung cần có tham số meta canonical[[15]](#footnote-15), nhưng không phải tất cả đều tuân thủ nguyên tác này.



Hình ‑ - Chuẩn canonical meta tag

Các Post có thể có quan hệ với nhau, Post này sẽ là reply của Post khác. Cần tổ chức lưu trữ hiệu quả.

#### Hướng giải quyết

Post có thể được nhận diễn qua mã hash của nội dung Post, và một tham số nữa như chuyên mục của Post. Sẽ không có hai bài viết cùng một nội dung trong một chuyên mục. Trong trường hợp này, khả năng nhận dạng có thể giúp loại bỏ những bài viết lặp nội dung (spam) từ các website.

Việc lưu trữ quan hệ giữa các Post có thể sử dụng mô hình Nested Set Model[[16]](#footnote-16) đem lại hiệu quả truy xuất rất cao, tuy nhiên, trong thực tế chứng minh dữ liệu nghiên cứu không quá quan tâm về vấn đề này mà chủ yếu chỉ cần thông tin về bài viết sinh ra chuyên mục (Original Post) và các bài viết thành phần của chuyên mục đó.

### Nhận diện ngôn ngữ

#### Đặt vấn đề

Các website lớn hoặc các hệ thống mạng xã hội có thể chứa nhiều Post ở nhiều ngôn ngữ khác nhau. Hệ thống cần phải nhận diện được để đưa ra những thống kê theo một quốc gia, hoặc vùng lãnh thổ hợp lý.

Tuy nhiên, thói quen của người dùng không phải chỉ dùng một ngôn ngữ trong lúc viết Post, nên việc nhận diện ngôn ngữ cần phải có khả năng phát hiện được tất cả các ngôn ngữ người dùng sử dụng.



Hình ‑ - Người dùng viết lẫn lộn các ngôn ngữ

#### Hướng giải quyết

Trường hợp một: Source hỗ trợ nhiều ngôn ngữ:

Cần phải có một bộ nhận diện ngôn ngữ, có thể thông qua từ điển, hoặc bảng mã các ký tự của ngôn ngữ đó, những đặc trưng của ngôn ngữ. Về việc này, đã có các bộ ứng dụng phát hiện ngôn ngữ miễn phí được phét triển cho người dùng sử dụng, chúng ta có thể tận dụng lại, và chỉnh sửa những thứ cần thiết. Hoặc sử dụng những API chất lượng nhưng giá cả phải chăng như <http://detectlanguage.com/>

Theo tự nhiên, các bài post thường chỉ chứa cao nhất hai hoặc ba ngôn ngữ khác nhau, các trường hợp nhiều hơn ba ngôn ngữ là rất hiếm và phần lớn là lỗi của bộ nhận diện ngôn ngữ chưa được chính xác. Chính vì thế, chúng ta chỉ cần lưu trữ lại ba ngôn ngữ có tỷ lệ nhận diện cao nhất trong Post.

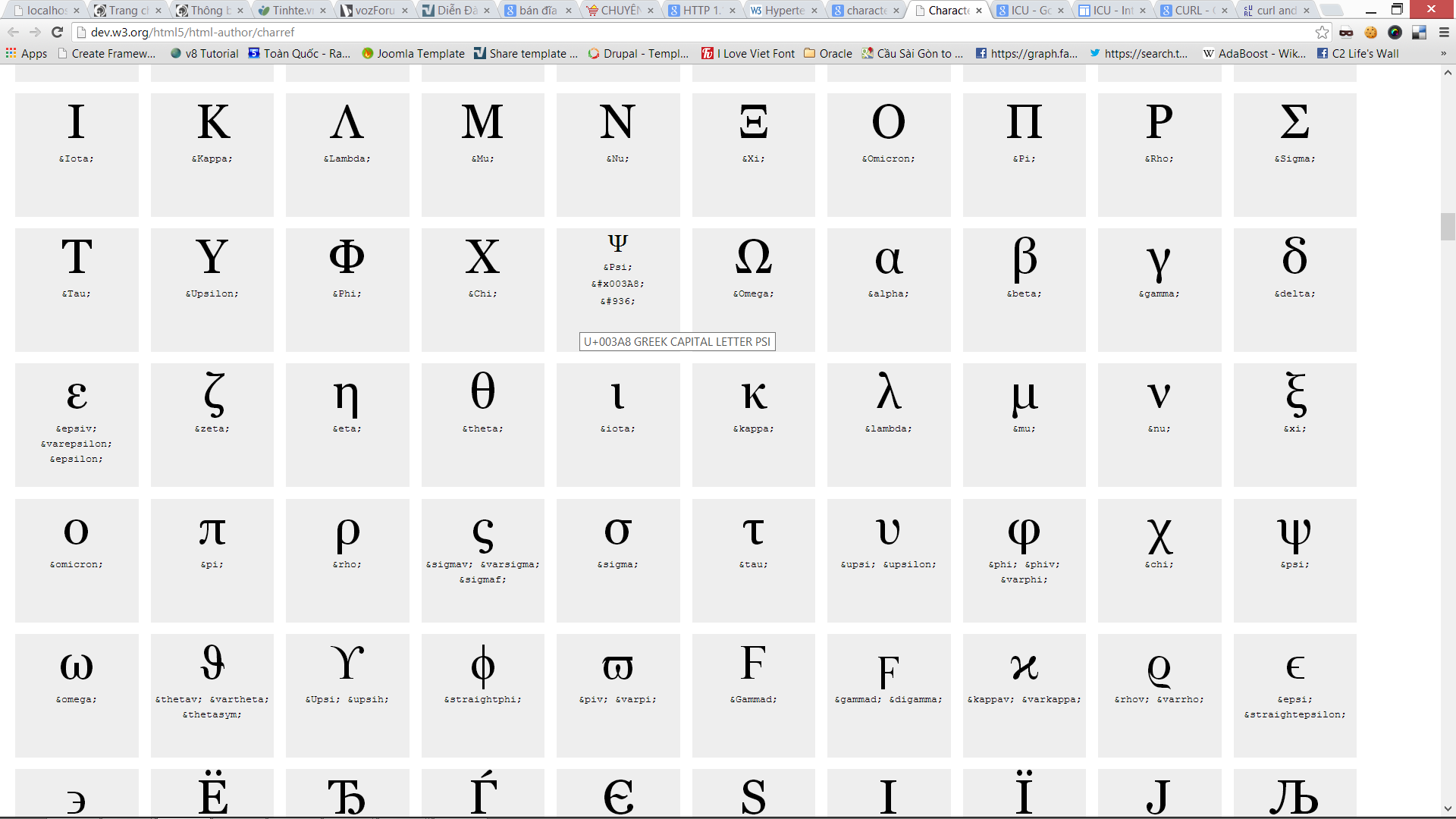
Trường hợp hai: Source chỉ chứa một ngôn ngữ

Trường hợp này có thể xử lý đơn giản là gán ngôn ngữ duy nhất cho Source, tuy nhiên, vẫn có thể có trường hợp ngoại lệ là Source sẽ phát triển thêm một ngôn ngữ mới, hoặc thói quen người dùng sẽ viết bài trên nhiều ngôn ngữ. Chúng ta có thể an toàn hơn bằng cách đặt ưu tiên cho một ngôn ngữ nhất định.

### Tính tương thích của hệ thống

#### Đặt vấn đề

Hiện nay, Word Wide Web đã phát triển rất xa và có rất nhiều nguyên tắc được lập ra trong việc phát triển và sử dụng Word Wide Web. Chính vì thế, tính tương thích của hệ thống là điều rất quan trọng, hệ thống cần phải đọc hiểu được các chuẩn Word Wide Web hiện đại (HTTP/1.0, HTTP/1.1), có khả năng giải mã các chuẩn encode phổ biến (UTF-8, ISO-8859-1, Windows-1252…), có khả năng kết nối tới các giao thức mã hóa (HTTPS, SFTP, FTPS…) và phải giải mã được các character entity[[17]](#footnote-17) (&nbsp;, &#222…) trong HTML.



Hình ‑ - Một phần bảng mã character entity từ w3.org

#### Hướng giải quyết

Trường hợp: đọc hiểu các chuẩn encode phổ biến

Tính đến năm 2013 chuẩn encode phổ biến nhất trong tiếng Việt là UTF-8 nên có thể ưu tiên giải mã chuẩn encode này trước, ngoài ra trong cộng đồng mã nguồn mở cũng đã có phát triển bộ thư viện ICU (International Components for Unicode) đến version 52.1 đã hỗ trợ rất nhiều chuẩn encode khác nhau.

Trường hợp: hỗ trợ nhiều giao thức kết nối

Có thể sử dụng CURL là giải pháp tối ưu, CURL là bộ thư việc mã nguồn mở được phát triển và tin dùng bởi rất nhiều ứng dụng lớn trên thị trường hiện nay. CURL hỗ trợ hầu hết các giao thức cần thiết trong Word Wide Web như HTTP, HTTPS, IMAP, IMAPS….

Trường hợp: giải mã các character entity

Hiện nay, trên thế giới có khá nhiều thư viện hỗ trợ việc giải mã các character entity, nhưng chưa có một thư viện nào hỗ trợ thực sự đầy đủ. Giải mã các chracter entity có thể xây dựng lại dựa trên nguyên tắc của character entity đó và bộ từ điển phù hợp cho character entity tương ứng. Các character entity đã được liệt kê đầy đủ tại <http://w3.org> nên việc thực hiện cũng không chiếm quá nhiều thời gian.

### Xử lý dữ liệu lớn

#### Đặt vấn đề

Dữ liệu thu thập được từ hệ thống theo thời gian sẽ là hàng triệu, chục triệu, thậm chí là trăm triệu Post (website tinhte.vn có hơn 18 triệu bài viết từ năm 2006 đến nay, vozforums.com là hơn 30 triệu bài). Với một nguồn dữ liệu khổng lồ sẽ làm cho hệ thống dễ dàng quá tải và mất khả năng kiểm soát. Xóa dữ liệu cũ cũng không phải là một biện pháp hay vì dữ liệu cũ vẫn là dữ liệu giá trị và có thể nói là cực kỳ giá trị trong phân tích.

#### Hướng giải quyết

Biện pháp hay và hợp lý nhất đó chính là chia nhỏ dữ liệu theo thời gian đồng thời phân tán hệ thống, những dữ liệu càng gần thời gian hiện tại sẽ có tốc độ cập nhật cao hơn và được ưu tiên xử lý nhiều hơn, các dữ liệu đã lâu cũng sẽ có ít Post mới, ít được chỉnh sửa từ Source nên chúng ta cũng có thể bỏ qua các xử lý cập nhật, trích xuất để tối ưu tài nguyên cho việc cập nhật dữ liệu mới nhất.

Tố chức Zie có khả năng phân tán để chia nhỏ xử lý, chia nhỏ dữ liệu. Trong thực tế chứng minh, mỗi Post thường có dung lượng trunh bình là 1 KB, không quá 1000 từ, tuy nhiên vẫn có những bài Post lên tới 100KB, tương đương khoảng 100 nghìn từ đối với dạng mã hóa là UTF-8. Thì phải với 10 triệu Post, chúng ta chỉ tốn khoảng 10 GB lưu trữ, chủ yếu là phải lưu trữ các quan hệ thật tốt để đảm bảo tốc độ truy xuất.

Hướng các Crawler phát tán URL của một website một cách có chọn lọc, không phán tán bừa bãi sẽ hướng được hệ thống tới những URL chất lượng chỉ chứa dữ liệu cần thiết, giảm thiểu dữ liệu thừa, giảm thiểu tiêu tốn tài nguyên hệ thống.

### Nhiều Source dẫn đến nhiều Crawler

#### Đặt vấn đề

Trong thực tế, số lượng Source cần phải lắng nghe có thể rất lớn, hàng nghìn, chục nghìn, việc cần phải cung cấp mỗi Source một Crawler khác nhau cũng làm cho số lượng Crawler phải tăng lên, hơn nữa, khi các Source có thay đổi cấu trúc website như giao diện, URL, bảo mật, Crawler cũng cần phải được cập nhật.

#### Hướng giải quyết

Trước hết cần có một bộ phận có khả năng phát hiện vấn đề về Crawler bằng cách truy vấn lại cùng một đường dẫn, nhưng phát hiện số lượng Post khác quá nhiều so với ban đầu.

Việc lập trình, hay tạo Crawler cần phải được đơn giản hóa, nếu có thể chỉ cần dùng giao diện đồ họa để tạo Crawler sẽ là một bước tiến vượt bậc. Trong thực tế, Crawler chỉ cần nhận diện những vị trí có dữ liệu, tuy nhiên, có thể Crawler sẽ phải xử lý các vấn đề như chuyển đổi định dạng ngày tháng, xác định hướng phát tán URL, nên thực tế vẫn cần một ngôn ngữ lập trình làm nền hỗ trợ. Ở đây, chúng ta có thể sử dụng các ngôn ngữ Script không cần biên dịch và đính kèm bộ biên dịch vào Zie như (Javascript, Python...).

Crawler cần phải được hỗ trợ khả năng giả dạng người dùng một cách hoàn hảo, vì hiện tại, với các website lớn, việc họ xây dựng các tường lửa để ngăn chặn những kết nối không phải là người dùng nhằm bảo vệ website (ngăn chặn DDOS) là rất dễ hiểu. Tránh tình trạng Source sẽ phát hiện và chặn Crawler.



Hình ‑ - Một tường lửa đơn giản để nhận diện người dùng

# THIẾT KẾ HỆ THỐNG ZIE

Hình ‑ - Mô hình đơn giản Zie

## Zie

Zie là thành phần cốt lõi của hệ thống và có tên trùng với tên hệ thống (Zie). Nhìn tổng quan, Zie là bộ não điều hành mọi hoạt động, bao gồm các hoạt động như chỉ đạo Crawler, thông báo lỗi, lưu trữ dữ liệu, chuẩn hóa dữ liệu. Nói cách khác, Zie là thành phần duy nhất trong hệ thống có khả năng hoạt động độc lập và là thành phần cực kỳ quan trọng. Zie được chia làm ba phần nhỏ như sau:

### Zie Stack

#### Ý tưởng:

Một trong những thành phần quan trọng trong website đó chính là liên kết (link). Khi xây dựng website, thông thường trang chủ sẽ chứa liên kết dẫn người dùng đến các trang thông tin chưa dữ liệu, các trang chứa dữ liệu hoặc sẽ liên kết với trang thông tin khác, hoặc là với trang chủ.

Hình ‑ - Hướng di chuyển của người dùng qua liên kết

Đi theo các liên kết trên website, người dùng sẽ truy cập được vào các trang nội dung mà website muốn gửi đến người dùng. Chính vì vậy, hệ thống sẽ cần một bộ nhớ stack để lưu vết liên kết và xử lý liên kết theo thứ tự LIFO[[18]](#footnote-18) tương tự như trình tự một người dùng thông thường truy cập vào website.

#### Khái niệm

Zie Stack hay gọi tắt là Stack có nhiệm vụ phân phối các liên kết của các trang web cần thu thập đến Crawler, đồng thời Stack sẽ nhận lại các liên kết mới từ Crawler sau khi Crawler hoàn tất xử lý một trang web, các liên kết mới sẽ được nạp vào Stack và sẽ được xử lý theo trình tự LIFO. Quá trình sản sinh ra liên kết mới từ Crawler gọi là Spreading.

Từ những dữ liệu được thu thập bởi Crawler (thời gian, mức độ quan trọng, khả năng sinh mới liên kết, khả năng sinh mới dữ liệu…), Stack sẽ sắp xếp các liên kết theo một thứ tự nhất định, phân loại các liên kết và sẽ gửi lại cho Crawler xử lý các liên kết kế tiếp.

Stack không chỉ đơn giản hoạt động theo nguyên tắc LIFO mà Stack ở đây có một sự thông minh, từ những thông tin được nạp vào cuối cùng có thể sẽ thay đổi thứ tự của những liên kết đã được nạp trước. Các liên kết dẫn tới dữ liệu không cần thiết, hoặc được dự đoán là chứa dữ liệu quá cũ kỹ sẽ được Stack đẩy càng về sau.

Hình ‑ - Quy trình hoạt động của Zie Stack

### Zie Crawler

#### Ý tưởng:

Mỗi website hệ thống sẽ cần có một Crawler khác nhau để thu thập dữ liệu, vì cấu trúc của các website khác nhau là khác nhau. Để có khả năng thu thập được cả thị trường, Zie cần đạt được một số lượng Crawler đáng kể và sẽ sinh mới hoặc cập nhật lại liên tục.

Chính vì thế, công việc lập trình Crawler là lâu dài và cần nhu cầu số lượng, đơn giản hóa công việc lập trình Crawler sẽ giảm bớt chi phí đào tạo, chi phí lập trình, giảm bớt lỗi phát sinh bởi lập trình viên. Zie Crawler ra mắt như một bộ biên dịch cung cấp môi trường lập trình hoàn toàn mới mà ở đó chỉ dành cho Crawler, trong đó rất nhiều thứ đã được tự động hóa để đơn giản Crawler tới mức tuyệt đối.

#### Khái niệm:

Zie Crawler phân biệt hoàn toàn với Crawler. Crawler nằm tách rời bên ngoài Zie trong khi Zie Crawler là một thành phần thuộc Zie. Zie Crawler là bộ biên dịch các Crawler, cung cấp cho Crawler một bộ API giao tiếp với Zie và là thành phần duy nhất giúp cho Crawler có thể liên lạc với Zie (Zie Stack, Zie Data).

Zie Crawler hoạt động như một trình duyệt web. Zie Crawler có khả năng biên dịch một trang web, phân tích các thành phần của trang web và cho phép Crawler dễ dàng lấy thông tin các thành phần của một trang web hoặc gửi các gói tin đặc biệt đến website chủ nhằm mục đích vượt qua các rào cản (firewall) từ website chủ. Ngoài ra, Zie Crawler chỉ cho phép lập trình viên viết Crawler làm nhiệm vụ của Crawler, đảm bảo tách rời với hệ thống, sơ suất từ Crawler sẽ không ảnh hưởng tới hệ thống và bất cứ sự xâm nhập nào từ Crawler đều là không tưởng.

Zie Crawler còn đảm nhận phần lớn các xử lý dữ liệu như:

* Kiểm tra khả năng lỗi dữ liệu.
* Xác định sự tương quan giữa các dữ liệu thu thập được (xác định một bài viết thuộc chủ đề nào, website nào, là bài viết trả lời cho bài viết nào…)
* Xác định ngôn ngữ được sử dụng trong website. Đây là thành phần có chức năng mạnh mẽ và mang lại lợi ích cực kỳ to lớn cho hệ thống.

### Zie Data

#### Ý tưởng:

Ý tưởng duy nhất đưa đến sự ra đời của Zie Data là phân chia xử lý, module hóa chức năng hệ thống.

#### Khái niệm:

Zie Data hay gọi tắt là Data là thành phần tương tự như Zie Stack nhưng chỉ có duy nhất một chức năng là nhận các dữ liệu về bài viết thu thập được bởi Crawler.

Các dữ liệu nhận được đều đã qua xử lý bởi Crawler nên nhiệm vụ của Data là lưu trữ, phân phối đến các cơ sở dữ liệu phân tán. Ngoài ra Data là thành phần đảm bảo tính nhất quán của dữ liệu, Data sẽ đảm bảo không có bất kỳ dữ liệu trùng lắp nào được lưu trữ lại hai lần.

### Zie Reporter

#### Ý tưởng:

Thay cho việc để hệ thống Front-end truy cập trực tiếp vào dữ liệu, việc xây dựng một module trong suốt giữa Front-end và dữ liệu là cần thiết không chỉ nhằm mục đích bảo mật mà còn nhằm mục đích giảm tải xử lý cho Front-end, giảm chi phí xây dựng Front-end và tăng cường sự chủ động của hệ thống trong việc trích xuất dữ liệu.

#### Khái niệm

Zie Reporter hay gọi tắt là Reporter là trung gian giữa Zie và Front-end, Reporter sẽ thể hiện dữ liệu ở dạng Front-end có thể hiểu được và nhận query từ Front-end, biên dịch query từ Front-end để gửi đến hệ thống truy xuất dữ liệu.

### Tổng quan về Zie

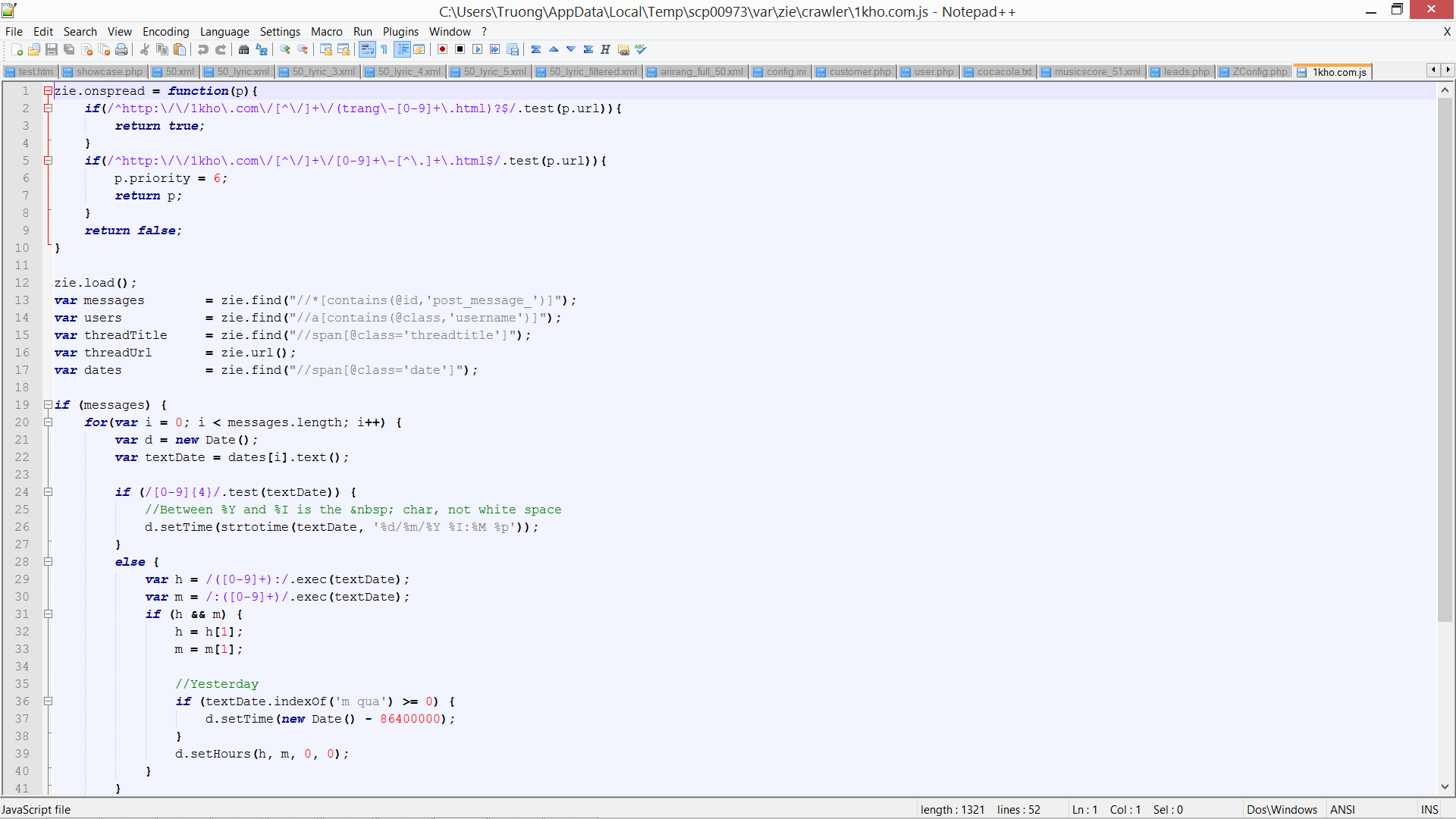
Hình ‑ - Sơ đồ quan hệ đơn giản giữa các thành phần trong Zie

## Crawler

Crawler là tập hợp tất cả các đoạn mã xử lý các website khác nhau để từ liên kết nhận được từ Stack, Crawler sẽ trích xuất được dữ liệu cho Data và nạp lại các liên kết mới cho Stack.

Crawler là thành phần đòi hỏi độ rộng do có rất nhiều, rất nhiều các website cần phải xử lý. Chính vì vậy, Crawler đã được hỗ trợ từ Zie và Crawler sẽ được hỗ trợ một ngôn ngữ lập trình riêng được Zie hiểu mà không cần phải biên dịch, ngôn ngữ lập trình Crawler sẽ đơn giản hóa các thao tác thông thường của một Crawler. Theo các tiêu chí trên, ngôn ngữ lập trình cho Crawler nên là một ngôn ngữ dạng Script, có tính tùy biến cao, uyển chuyển, phù hợp với môi trường xử lý HTML, JavaScript là những ngôn ngữ phổ biến để thiết kế website. Trong mô hình của hệ thống hiện tại, nhóm sử dụng ngôn ngữ lập trình JavaScript làm cơ sở phát triển Crawler.

Hiện tại, Crawler là thụ động, chỉ được gọi bởi Zie khi cần thiết và sau khi hoàn thành nhiệm vụ, Crawler cũng sẽ được tắt đi và ở trang thái chờ.



Hình ‑ – Một Crawler được viết bằng API của Zie

## Front-end

Front-end là thành phần duy nhất có giao diện cho phép người dùng thực hiện trích xuất, thống kê lại dữ liệu có được từ quá trình thu thập của Zie.

## Thiết kế dữ liệu

### Table authors

Lưu trữ thông tin người gửi một bài viết trên internet.

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Data type | Not Null? | Primary key? |
| id | bigint | Yes | No |
| fullname | character varying(500) | No | No |
| url | bigint | No | No |
| min\_date | date | Yes | No |
| max\_date | date | Yes | No |
| source | bigint | Yes | No |

Bảng ‑ – Cấu trúc table authors

### Table contents

Lưu trữ lại nội dung bài viết trên internet, ngôn ngữ sử dụng, tỷ lệ các ngôn ngữ sử dụng. Đồng thời lưu trữ lại mã hash để đối chiếu.

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Data type | Not Null? | Primary key? |
| id | bigint | Yes | No |
| max\_date | date | No | No |
| min\_date | date | No | No |
| content | text | No | No |
| lang | smallint[] | Yes | No |
| percent | smallint[] | Yes | No |
| hash | bytea | No | No |
| primitive | bytea | No | No |
| score | real[] | Yes | No |

Bảng ‑ - Cấu trúc table contents

### Table languages

Danh sách các ngôn ngữ hỗ trợ bởi Zie.

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Data type | Not Null? | Primary key? |
| id | integer | Yes | Yes |
| code | character varying(10) | Yes | No |

Bảng ‑ - Cấu trúc table languages

### Table posts

Danh sách các bài viết, bao gồm thông tin về nội dung, ngày xử lý, ngày viết bài, tác giả viết bài. Đây là bảng kết nối tất cả các bảng thông tin khác. Và cũng là thành phần quan trọng để tính toán báo cáo gửi đến cho Front-end.

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Data type | Not Null? | Primary key? |
| id | bigint | Yes | No |
| liked | smallint | Yes | No |
| title | character varying(255) | No | No |
| author | bigint | Yes | No |
| date | timestamp without time zone | Yes | No |
| content | bigint | Yes | No |
| thread | bigint | Yes | No |
| url | bigint | Yes | No |
| crawl | timestamp with time zone | No | No |

Bảng ‑ - Cấu trúc table posts

### Table sources

Danh sách các nguồn website được hỗ trợ bởi hệ thống. Đồng thời, bảng còn có trách nhiệm lưu trữ lại những lần xử lý với các website này để thực hiện những xử lý xoay vòng, tránh làm quá tải website nguồn.

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Data type | Not Null? | Primary key? |
| id | bigint | Yes | Yes |
| domain | character varying | Yes | No |
| state | smallint | Yes | No |
| crawl | timestamp without time zone | Yes | No |
| update | timestamp without time zone | Yes | No |
| updatable | boolean | Yes | No |
| type | smallint | Yes | No |
| out | boolean | Yes | No |
| ccycle | interval | No | No |
| ucycle | interval | No | No |
| cycle | smallint | Yes | No |
| cpop | smallint | No | No |
| upop | smallint | No | No |

Bảng ‑ - Cấu trúc table sources

### Table stacks

Bảng stacks là dữ liệu được xây dựng và sắp xếp bởi Zie Stack. Lưu trữ các thông tin về thời gian dữ liệu của liên kết, nguồn gốc sản sinh ra liên kết. Số lần xử lý liên kết và lỗi xử lý gửi đến từ Crawler.

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Data type | Not Null? | Primary key? |
| id | bigint | Yes | No |
| push | timestamp without time zone | No | No |
| pop | timestamp without time zone | No | No |
| popped | integer | Yes | No |
| priority | smallint | Yes | No |
| err | boolean | Yes | No |
| err\_message | text | No | No |
| url | bigint | Yes | No |
| parent | bigint | No | No |
| data | timestamp without time zone | Yes | No |
| fresh | interval | Yes | No |
| updatable | boolean | Yes | No |

Bảng ‑ - Cấu trúc table stacks

### Table threads

Mỗi bài viết được phân định thuộc vào một thread hay còn gọi là một luồng. Mỗi luồng sẽ có một bài viết sinh ra luồng gọi là Lead (Original Post).

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Data type | Not Null? | Primary key? |
| id | bigint | Yes | No |
| title | character varying(1000) | No | No |
| url | bigint | No | No |
| lead | bigint | No | No |
| min\_date | date | Yes | No |
| max\_date | date | Yes | No |
| lang | smallint[] | Yes | No |

Bảng ‑ - Cấu trúc table threads

### Table urls

Lưu trữ các liên kết được sử dụng, bao gồm các liên kết của bài viết, của thread và cả của stack. Mỗi liên kết sẽ được xác định bằng một mã hash đảm bảo không trùng lấp.

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Data type | Not Null? | Primary key? |
| id | bigint | Yes | No |
| url | character varying(3000) | No | No |
| min\_date | date | No | No |
| max\_date | date | No | No |
| hash | bytea | Yes | No |
| source | bigint | Yes | No |
| primitive | bytea | No | No |

Bảng ‑ - Cấu trúc table urls

## Kỹ thuật phân chia dữ liệu

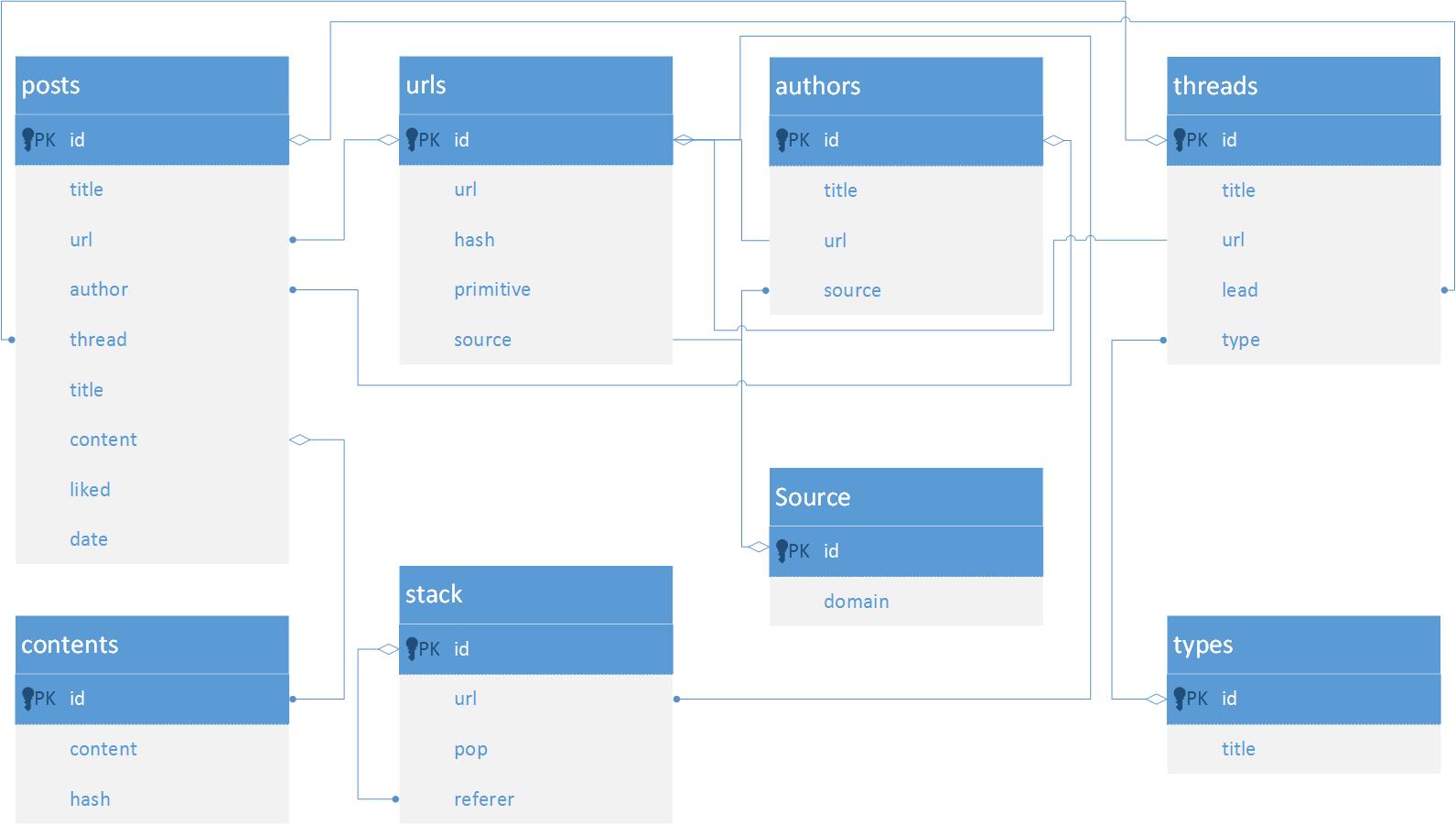
Ứng dụng chức năng Partitioning[[19]](#footnote-19) trong cơ sở dữ liệu. Ở mỗi bảng dữ liệu đều có chứa hai cột min\_date và max\_date lưu trữ thời gian lâu nhất và mới nhất mà thành phần đó được sử dụng. Các bảng dữ liệu sẽ được phân chia ra theo max\_date để giảm kích thược các bảng, đồng thời tăng tốc độ xử lý vì phải truy vấn trên một phân vùng của dữ liệu thay cho toàn bộ dữ liệu.

Ví dụ: table posts sẽ được chia ra làm 3 table

* posts\_2011: lưu trữ những thảo luận có max\_date nhỏ hơn 01-01-2012.
* posts\_2012: lưu trữ những thảo luận có max\_date nhỏ hơn 01-01-2013.
* posts\_2013: lưu trữ những thảo luận có max\_date nhỏ hơn 01-01-2014.

Khi truy xuất tới một loạt các thảo luận trong một khoảng thời gian, hệ thống sẽ có thể nhanh chóng xác định phân vùng của dữ liệu cần thiết và truy xuất trực tiếp trên đó thay cho việc phải truy xuất tất cả dữ liệu.

## Sơ đồ quan hệ



Hình ‑ - Sơ đồ quan hệ

# KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

## Kết luận

Từ những cấu trúc, mô hình và phương pháp tiếp cận nêu trên, nhóm đã xây dựng thành công hệ thống thu thập dữ liệu với độ phủ đạt khoảng 200 website vừa và lớn tại Việt Nam. Hệ thống có khả năng thu thập dữ liệu từ các kênh mạng xã hội lớn như Facebook, Twitter…

Hiện nay, với cấu hình phần cứng:

* CPU: 2 × Intel Xeon E5-2620 (2 Ghz, 6 cores).
* RAM: 4 × 8GB RAM.
* Hardisk: 4 × SSD Intel 335 180 GB – Raid 10.
* Internet Bandwidth: 20 Mbps Up/20 Mbps Down.

Hệ thống có thể thu thập 100.000 thảo luận mới/phút (chưa bao gồm cập nhật lại dữ liệu đã được thu thập). Tốc độ truy xuất dữ liệu đạt dưới 10 giây cho một báo cáo lớn và dưới 1 giây cho một báo cáo trung bình. Tính đến tháng 1 năm 2014, sau 8 tháng xây dựng và thử nghiệm, hệ thống hiện tại đang lưu trữ dữ liệu của hơn 100.000.000 bài viết và tập trung chủ yếu là dữ liệu trong năm 2013.

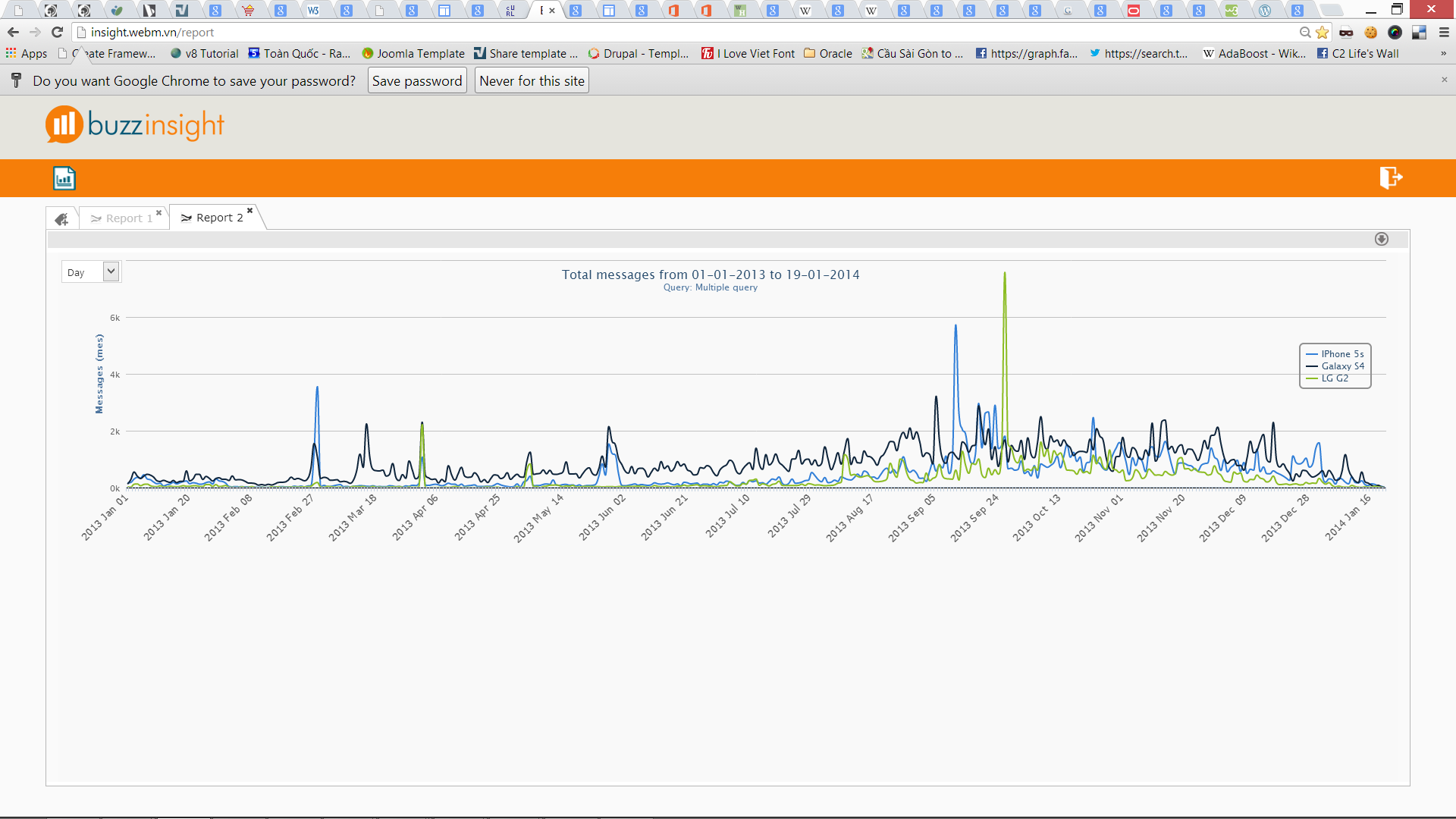
Hệ thống không có tích hợp bất kỳ ứng dụng thương mại nào nên chi phí phát triển là rất thấp. Hoàn toàn có lợi thế cạnh tranh với các đối thủ trên thị trường và hoàn toàn nằm trong khả năng của các nhà nghiên cứu thị trường thậm chí là cá nhân.

Nhóm đã cài đặt hệ thống tại địa chỉ <http://insight.webm.vn> với giao diện front-end đơn giản nhằm mục đích thử nghiệm hệ thống và đặc biệt hiện nay đang được sử dụng bởi công ty phân tích thị trường WebM Corp, đồng thời, hệ thống cũng đã nhận được một số thành tích:

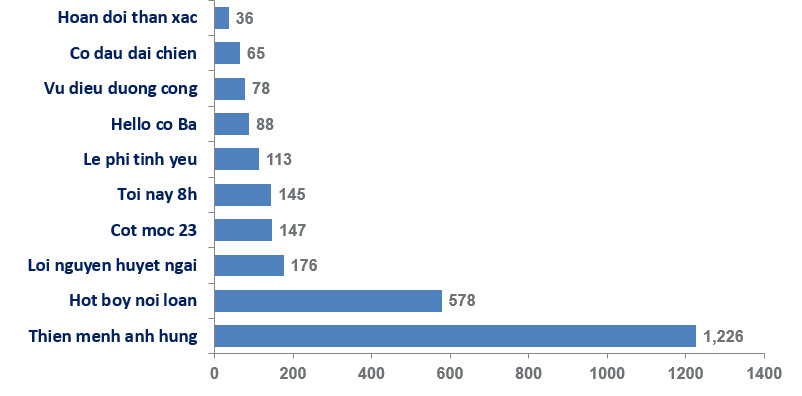
* Giải nhì trong hội nghị khoa học trẻ 2013 tại trường Đại Học Công Nghệ Thông Tin – ĐH Quốc Gia TP Hồ Chí Minh.
* Giải ba cuộc thi sinh viên nghiên cứu khoa học Eureka 2013 của thành phố Hồ Chí Minh.

Về kỹ thuật, hệ thống tính đến tháng 1 năm 2014 đang sử dụng hệ quản trị cơ sở dữ liệu mở PostgreSQL 9.3, hệ thống tìm kiếm mở Sphinx 2.1.4, xây dựng bằng ngôn ngữ C++ trên hệ điều hành CentOS 6.4. Hệ thống sử dụng 100% thư viện nguồn mở.

Dưới đây là hai mẫu báo cáo được xây dựng nhờ dữ liệu phân tích từ hệ thống:



Hình ‑ - Mức độ quan tâm của người dùng với 3 sản phẩm Iphone 5S, Galaxy S4 và LG G2.



Hình ‑ - Mức độ quan tâm của người dùng internet đến phim tết 2013

## Hướng phát triển

Trong tương lai, hệ thống sẽ được tập trung cái thiện hiệu suất ngày một tốt hơn và phát triển các module nâng cao như:

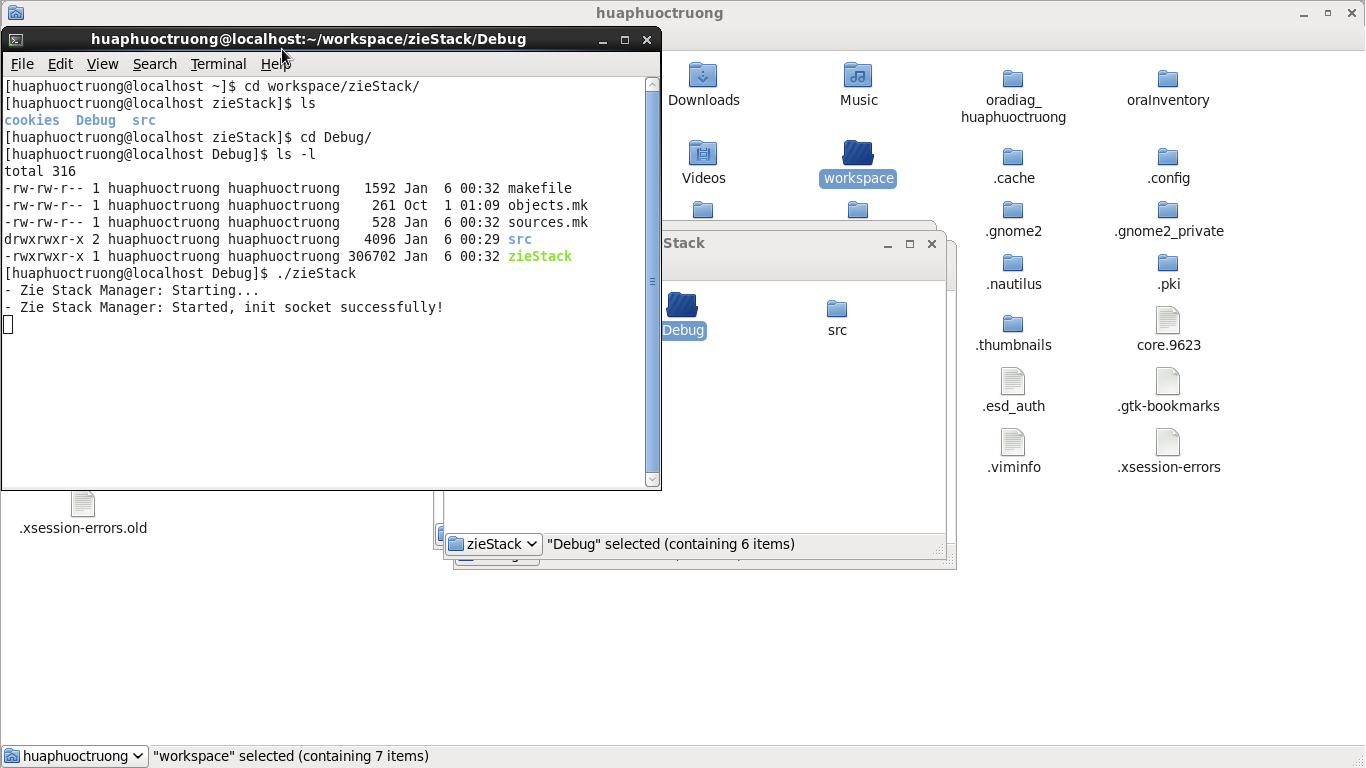
* Phân loại thảo luận dựa trên các dữ liệu thu thập được, các nhà nghiên cứu sẽ có thể truy xuất được những dữ liệu có giá trị cao nhất.
* Nhận diện cảm xúc của thảo luận.
* Thu thập thông tin người dùng.

## 

# CÀI ĐẶT MINH HỌA

## Zie Stack

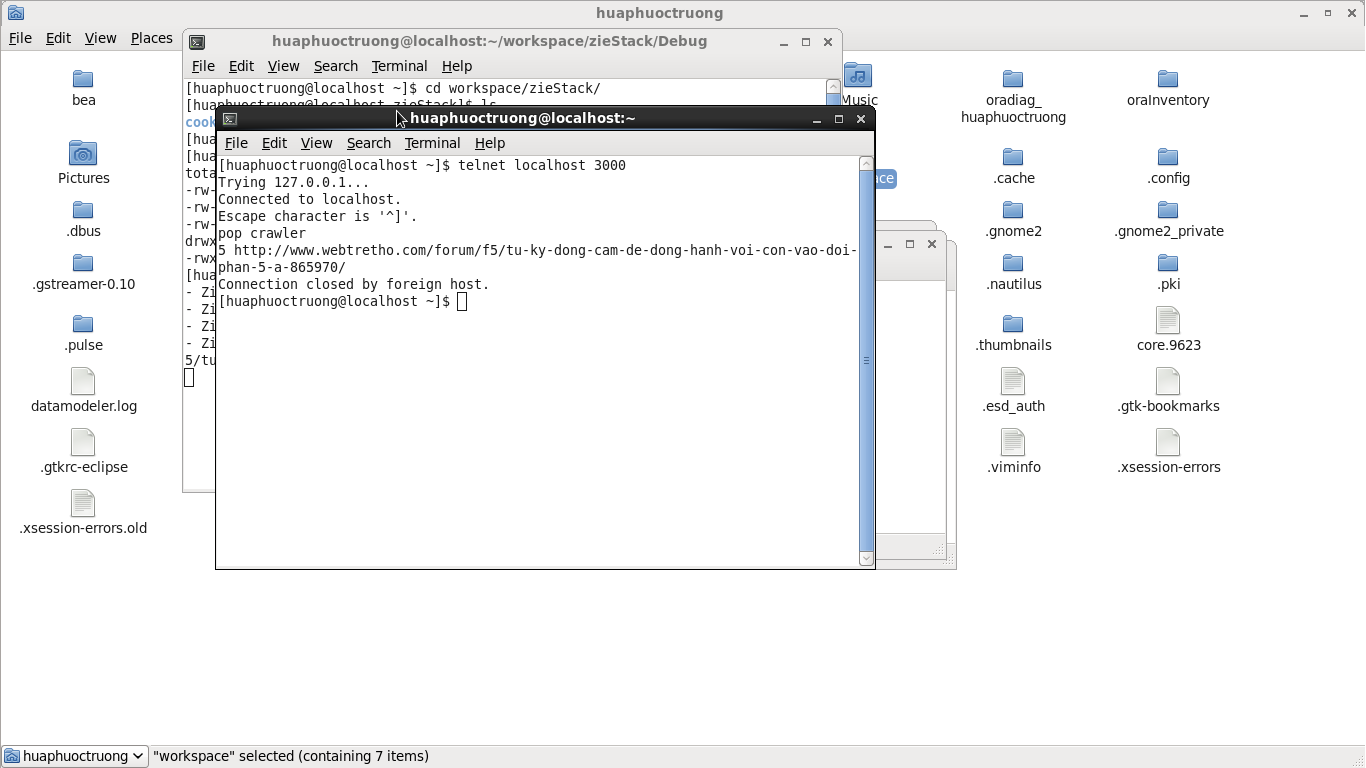
Khởi động Zie Stack bằng cách chạy lệnh ZieStack từ terminal.



Hình ‑ - Khởi động Stack

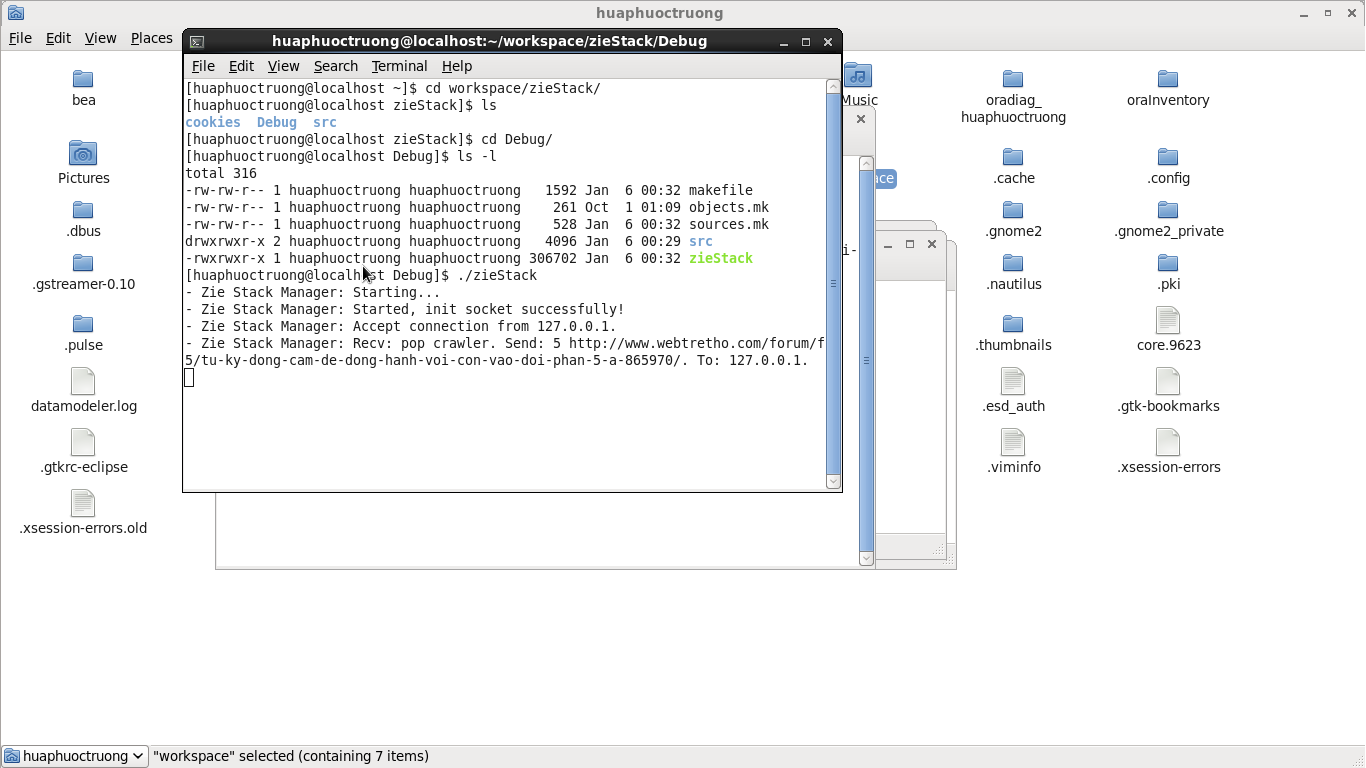
Zie Stack sẽ hiển thị thông báo hoàn tất nếu khởi động thành công, ngược lại, sẽ có thông báo lỗi tùy theo từng trường hợp.

Lấy một url từ stack bằng Zie Stack bằng cách dùng lệnh telnet, kết nối tới port của Zie Stack, mặc định là 3000.



Hình ‑ - Pop Zie Stack

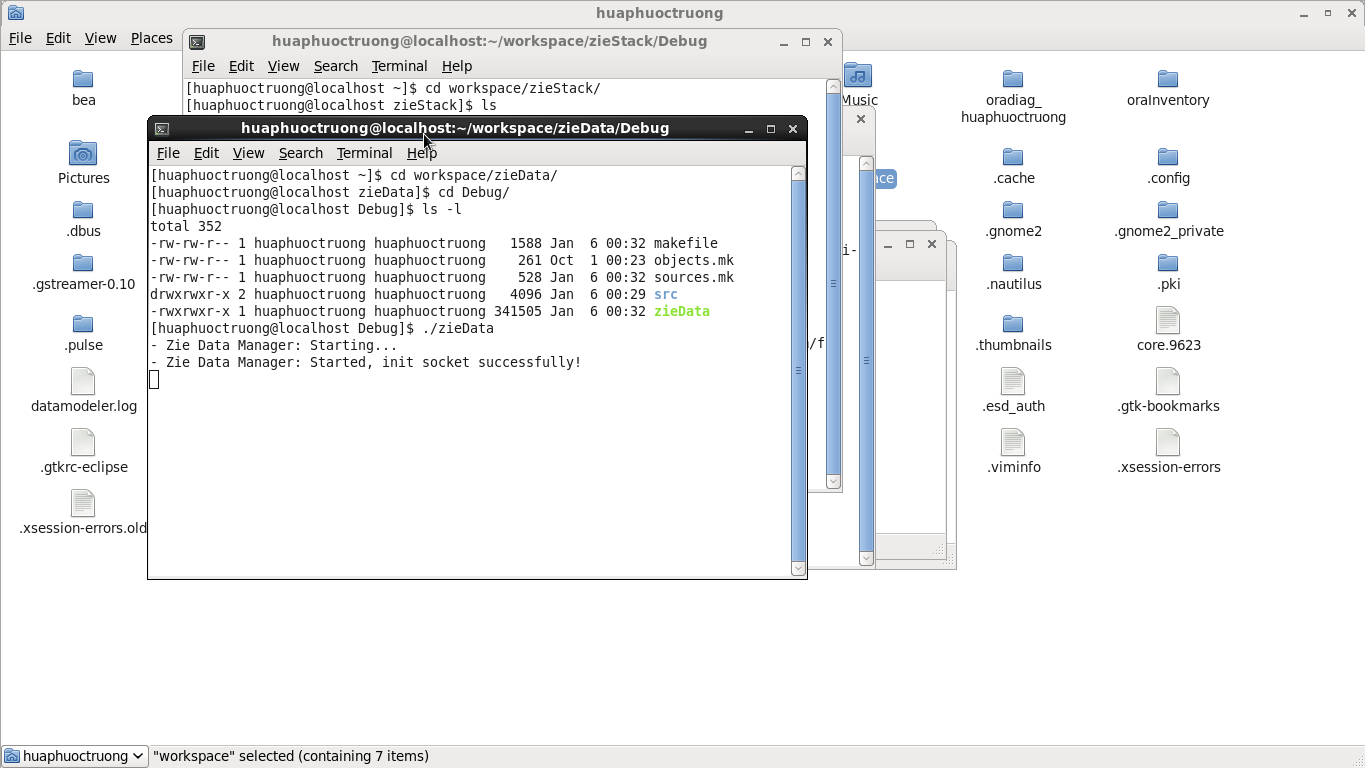
Ngoài ra, khi có truy xuất từ Client hoặc người dùng, Zie Stack cũng sẽ dùng cơ chế Log để thông báo lại tiến trình cho người quản trị:



Hình ‑ - Cơ chế Log trong Zie Stack

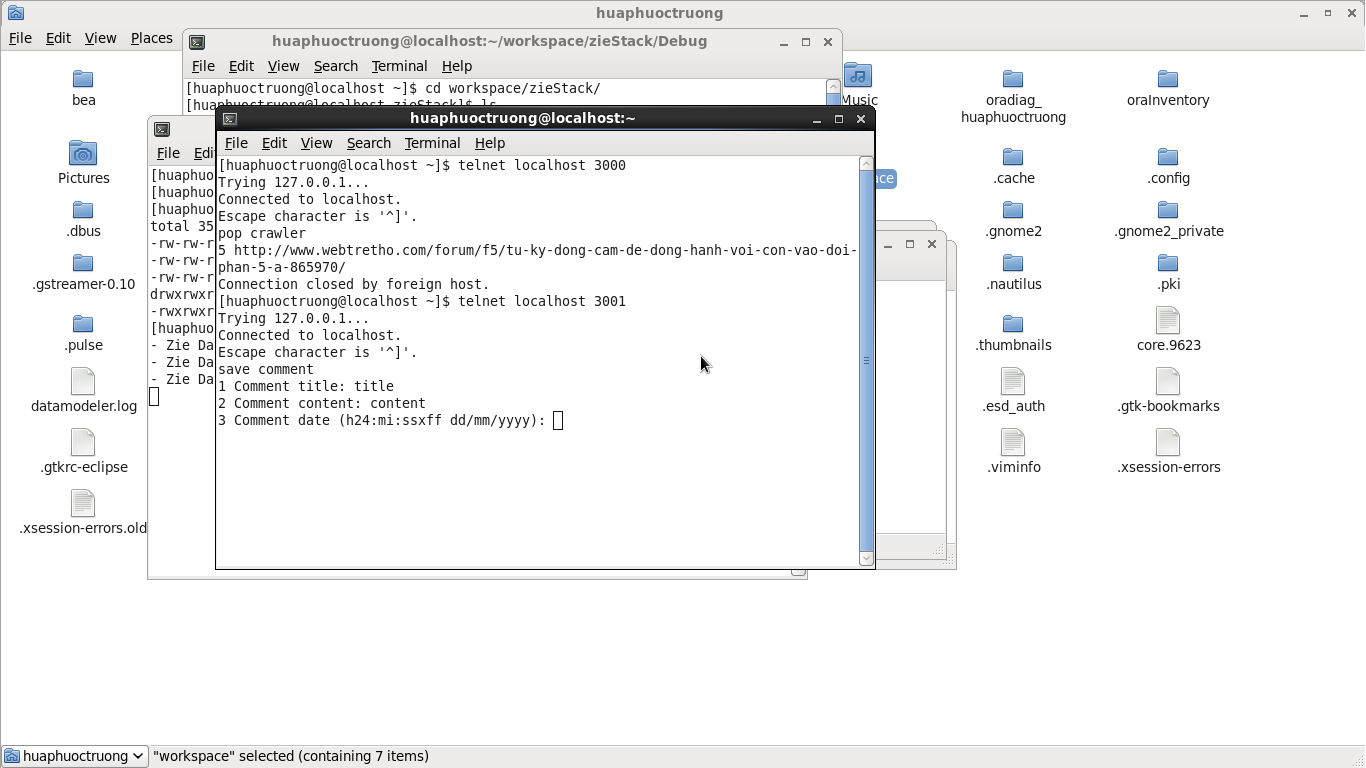
## Zie Data

Tương tự Zie Stack, khởi động Zie Data thông qua lệnh ZieData. Mặc định, Zie Data sẽ dùng port 3001.



Hình ‑ - Khởi động Zie Data

Sử dụng lệnh “save comment” để tiến hành lưu thảo luận từ người dùng. Khi nhập thông qua terminal, Zie Data sẽ có thông báo nhắc nhở người dùng cần điền thông tin gì tiếp theo.

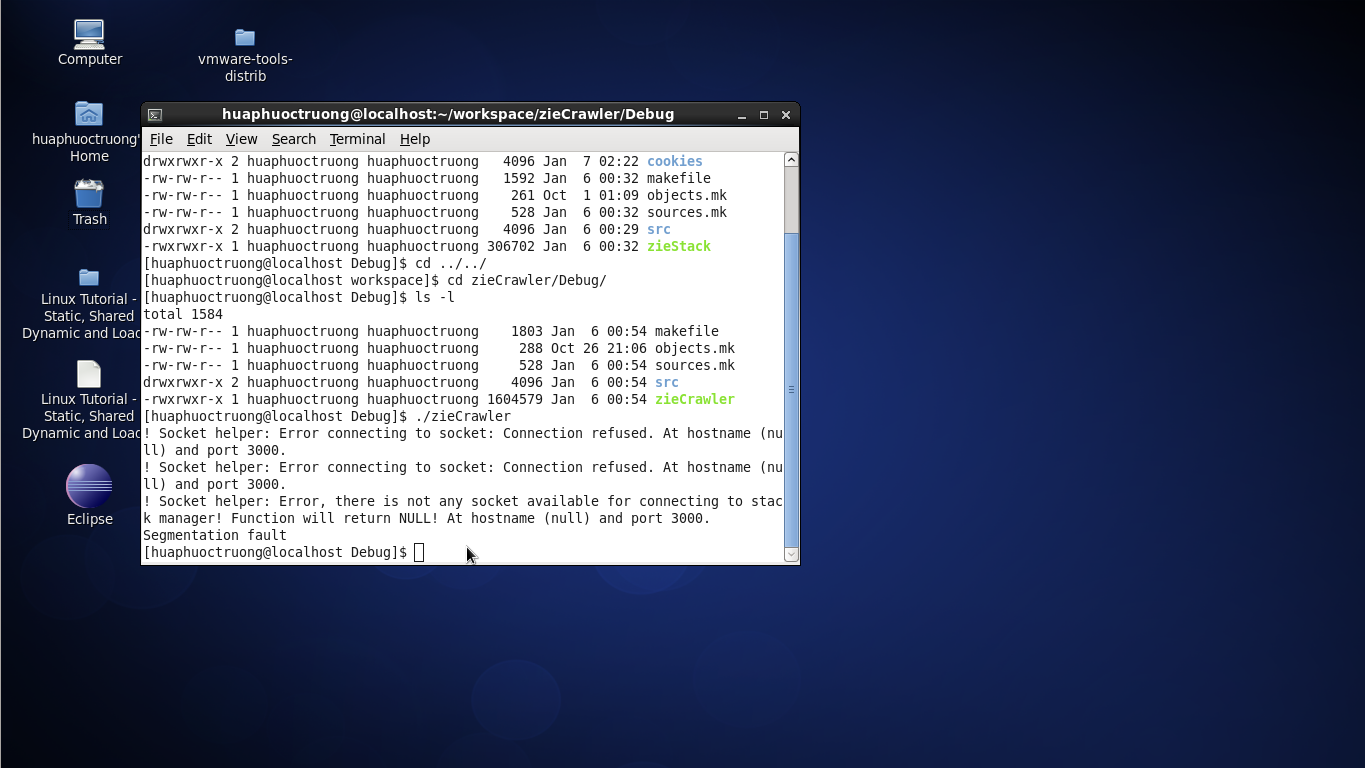


Hình ‑ - Lệnh save của Zie Data

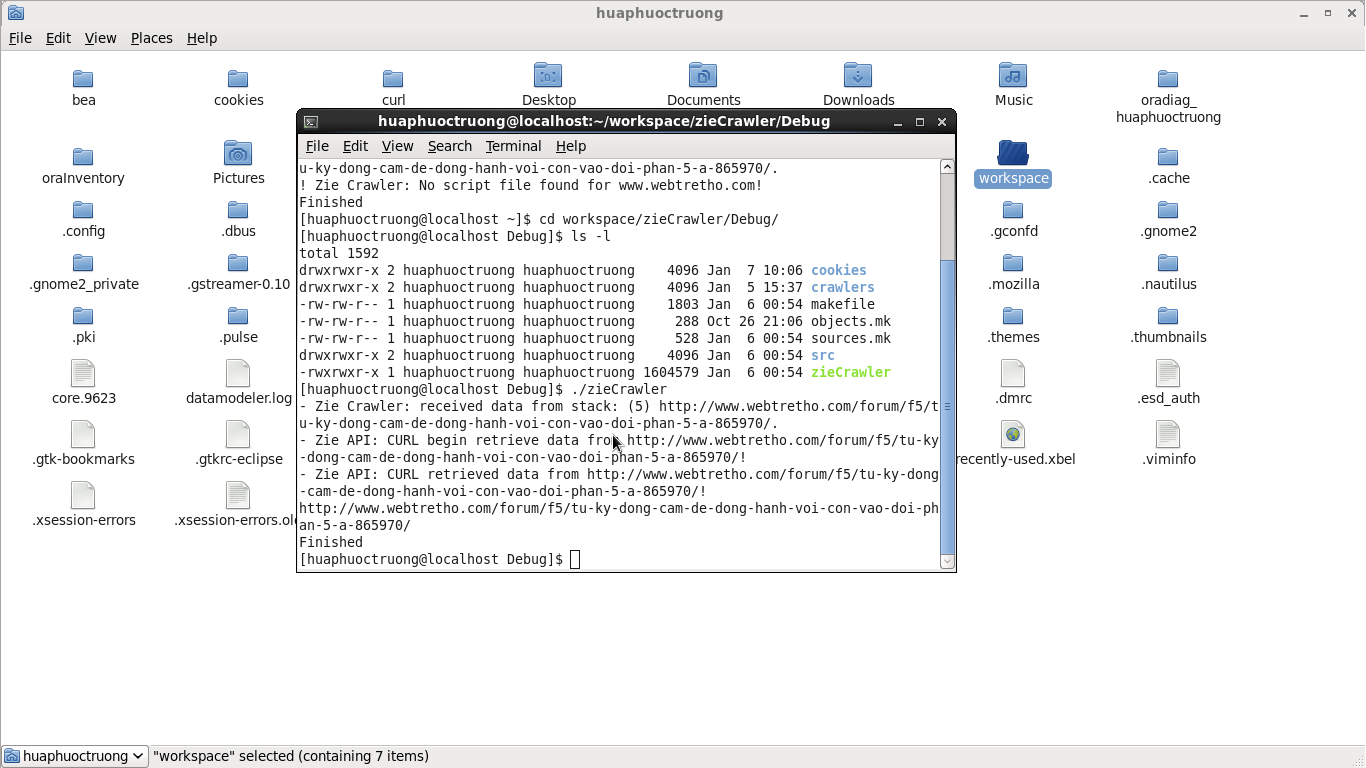
## Zie Crawler

Khởi động Zie Crawler bằng cách dùng lệnh ZieCrawler. Khi khởi động không thành công, Zie Crawler sẽ có thông báo rõ ràng lý do vì sao không khởi động được.

Ví dụ dưới đây là trường hợp Zie Crawler không thể chạy do Zie Data và Zie Stack chưa được khởi động:



Hình ‑ - Zie Crawler thông báo không tìm thấy Stack và Data



Hình ‑ - Crawler chạy thành công

# TÀI LIỆU THAM KHẢO

1. Carlos Castillo, Efective Web Crawling, University of Chile, Chile, 2004.
2. Dr Lyn McCredden, Research and Graduate Studies Convenor, “Influence of the World Wide Web on literature”, School of Literary and Communication Studies - Deakin University, Australia, 1997.
3. Junghoo Cho, Crawling the web: discovery and maintenance of large-scale web data, Stanford University, United State, 2001.
4. Katarzyna Musiał, “Recomemdation system for online social network”, School of Engineering, Sweden, 2006.
5. Sergey Brin and Lawrence Page, “The Anatomy of a Large-Scale Hypertextual Web Search Engine”, Computer Science Department, Stanford University, Stanford, CA 94305.
6. Fork, Exec and Process control

<http://www.yolinux.com/TUTORIALS/ForkExecProcesses.html>

1. Linux Howtos: C/C++ -> Sockets Tutorial

<http://www.linuxhowtos.org/C_C++/socket.htm>

1. PostgreSQL libpq

<http://www.postgresql.org/docs/8.1/static/libpq.html>

1. Sphinx

<http://sphinxsearch.com/docs/current.html>

1. The global structure of an HTML document

<http://www.w3.org/TR/html401/struct/global.html>

1. The global structure of an HTML document

<http://www.w3.org/TR/html401/struct/global.html>

# PHỤ LỤC 1: POSTGRESQL

Zie sử dụng hệ quản trị cơ sở dữ liệu PostgreSQL 9.3 làm nền tảng.



PostgreSQL là hệ quản trị cơ sở dữ liệu nguồn mở, miễn phí và được hỗ trợ bởi nhiều nền tảng khác nhau, đồng thời Postgres còn là hệ quản trị cơ sở dữ liệu được khuyến nghị sử dụng bởi nhà nước và được cộng đồng internet xem là sự thay thế đáng giá cho Oracle.

PostgreSQL có thể sử dụng, cài đặt dễ dàng trên Linux, chiếm ít tài nguyên hệ thống hơn Oracle, tuy nhiên, điểm khiếm khuyết của Postgres là không có tính năng Logging như Oracle có thể bảo vệ dữ liệu an toàn. Nhưng tính năng này hoàn toàn có thể thay thế bằng một chế độ Auto backup.

# PHỤ LỤC 2: COMPACT LANGUAGE DETECTOR



Đây là thành phần phát hiện ngôn ngữ của trình duyệt web Chrome, hoàn toàn mở và có tốc độ xử lý rất nhanh, sai số chấp nhận được. Có tổ chức đơn giản, dễ dàng phát triển.

# PHỤ LỤC 3: PQ

PQ là bộ API viết bằng C hỗ trợ giao tiếp với cơ sở dữ liệu Postgres và hoàn toàn mở, tương đương với OCCI trong Oracle. Tuy không thực sự mạnh mẽ nhưng PQ đáp ứng được các yêu cầu tối thiểu của Zie.

# PHỤ LỤC 4: SPHINX CLIENT



Hệ thống sử dụng một bộ Index riêng tác rời hoàn toàn với dữ liệu là Sphinx. Sphinx mang lại tốc độ truy xuất trên cả tuyệt vời. Trong đó Sphinx Client là thư viện hỗ trợ giao tiếp với Sphinx trên C và hoàn toàn mở.

# PHỤ LỤC 5: ICU



ICU (International Components for Unicode) là thư viện được viết trên C/C++ hỗ trợ phần lớn các chuẩn mã hóa trong Unicode. ICU hiện đang được sử dụng trong các sản phẩm của Apache, IBM, Google… như: DB2, Lotus, Chrome, Android OS, iOS, MacOS, Safari…

# PHỤ LỤC 6: CHUẨN GIAO TIẾP TRONG ZIE

1. **Nguyên tắc giao tiếp trong Zie**

Trong Zie, các module giao tiếp với nhau qua socket. Mỗi module có thể hoạt động trên một PC khác nhau hoặc cùng một PC nhưng trên các Port khác nhau. Trong đó, ngoại trừ đặc biệt trường hợp của Zie Stack không nên khởi động nhiều instance trên các server khác nhau nhằm tránh trùng lắp thì các module khác đều có khả năng hoạt động trên nhiều PC, nhiều instance khác nhau.

Một lệnh trong Zie có cấu trúc như sau:

[Tên lệnh] [Tham số 1] [Tham số 2] […] [Tham số n]

Trường hợp tham số có chứa ký tự khoảng trắng thì cấu trúc lệnh như sau:

[Tên lệnh][CR+LF][Tham số 1][CR+LF][Tham số 2][CR+LF][Tham số 3][CR+LF]…[CR+LF][Tham số n]

Một kết quả trả về trong Zie sẽ có cấu trúc như sau:

[State number] [Thông điệp]

Trong đó:

* State number: là thông số trạng thái của kết quả, thường cho kết quả không âm khi xử lý thành công và cho kết quả âm khi có lỗi xảy ra. Mỗi giá trị State number sẽ tương đương với một kết quả khác nhau và đảm bảo không trùng lắp.
* Thông tin: là một chuỗi báo xử lý thành công, hoặc một chuỗi thông báo lỗi trong trường hợp có lỗi xảy ra trong quá trình xử lý.

5. **Zie Stack**

Mặc định, Zie Stack hoạt động trên port 3000.

Nhằm đảm bảo các Crawler không cùng xử lý một url nên cần thiết phải có một module đóng vai trò như người tổng hợp các Crawler, điều hướng các Crawler. Khi một Crawler được khởi động, Zie Crawler Stack sẽ cung cấp cho Crawler một địa chỉ mà Crawler cần đi đến để tiến hành truy vét dữ liệu.

Trong quá trình truy vét, Crawler có thể sẽ phát hiện những liên kết mới. Ở đây, Zie Crawler Stack đóng một vai trò mới là người tiếp nhận các url từ Crawler, sắp xếp, lưu trữ url vào Stack và trả lại cho Crawler khi cần thiết.

Những lệnh được Zie Crawler Stack hỗ trợ:

| Lệnh | Tham số | Ý nghĩa | Kết quả trả về |
| --- | --- | --- | --- |
| crawl | Không có. | Lấy liên kết kế tiếp cần được xử lý từ Zie Stack. | Một số nguyên báo trạng thái kết quả (state number) trong đó nếu là số dương lớn hơn không là kết quả truy vấn thành công và cũng là giá trị identity của url trong stack.  Đi kèm với state number là chuỗi url phân cách bởi khoảng trắng. |
| push | Liên kết mới cần nạp vào Zie Crawler Stack. | Nạp một liên kết mới vào Zie Stack. | Một state number đi kèm với chuỗi thông báo kết quả. Nếu state number bằng 0 tức là quá trình nạp thành công, ngược lại, nếu state number là số âm, quá trình nạp thất bại. |
| update | Không có | Lấy liên kết kế tiếp cần được cập nhật lại từ Zie Stack. | Một số nguyên báo trạng thái kết quả (state number) trong đó nếu là số dương lớn hơn không là kết quả truy vấn thành công và cũng là giá trị identity của url trong stack.  Đi kèm với state number là chuỗi url phân cách bởi khoảng trắng. |

Mã lỗi trong Zie Stack

|  |  |
| --- | --- |
| Mã lỗi | Thông điệp |
| -1 | Lệnh không đúng |
| -2 | Không thể truy vấn tới cơ sở dữ liệu |
| -3 | Không thể thoát Zie Stack |

Trong quá trình yêu cầu một metric hoặc một thống kê, sẽ có trường hợp thống kê quá lớn dẫn tới quá trình xử lý phải diễn ra trong thời gian dài và đòi hỏi các tiến trình cần phải được xử lý một cách có thứ tự. Zie Statist Stack có nhiệm vụ quản lý các yêu cầu thống kê, điều hướng Zie Statist.

1. **Zie Data**

Zie Data mặc định hoạt động trên port 3001.

Zie Data đóng vai trò là cầu nối trung gian giữa cơ sở dữ liệu và Crawler. Một Crawler khi tìm được dữ liệu phù hợp như thông tin người dùng, ý kiến, lời nhận xét… sẽ gửi tới Zie Data theo một cấu trúc định sẵn, tại đây Zie Data sẽ xử lý lại dữ liệu thô thành dữ liệu có cấu trúc và lưu trữ vào cơ sở dữ liệu.

|  |  |  |  |
| --- | --- | --- | --- |
| Lệnh | Tham số | Ý nghĩa | Kết quả trả về |
| post | Dữ liệu thô một thảo luận bao gồm các thông tin như: tiêu đề, nội dung, ngày đăng thảo luận, liên kết thảo luận, số lượt like, trang thái thảo luận banned/active, username của người đăng tin, liên kết chuyên mục của thảo luận, tiêu đề chuyên mục của thảo luận.  Mỗi thông tin sẽ được truyền cách nhau bởi cặp ký tự Return và Endline. | Thông báo cho Zie Data biết đang có dữ liệu thô của một thảo luận cần được lưu trữ vào cơ sở dữ liệu. | Kết quả cuối cùng với state number có giá trị bằng 0 khi quá trình lưu trữ thành công. Ngược lại state number sẽ mang giá trị âm.  Ngoài ra, trong các quá trình trung gian, có thể Zie Data sẽ đưa ra những thông báo nhắc nhở nếu người dùng bỏ sót dữ liệu. |

Mã lỗi và mã thông báo trong Zie Data

|  |  |
| --- | --- |
| Mã lỗi / mã thông báo | Thông điệp |
| -1 | Lệnh không đúng |
| -2 | Không thể truy cập tới cơ sở dữ liệu |
| -3 | Không đủ tham số |
| -4 | Định dạng ngày tháng năm không chính xác, và hãy nhập lại. |
| -5 | Không thể thoát Zie Data |
| 1 | Hãy nhập tiêu đề thảo luận |
| 2 | Hãy nhập nội dung thảo luận |
| 3 | Hãy nhập ngày đăng của thảo luận |
| 4 | Địa chỉ url của thảo luận |
| 5 | Số lượt người chú ý đến thảo luận này |
| 6 | Thảo luận này có đang bị cấm ở trang nguồn (0/1) |
| 7 | Tên đăng nhập của người đăng thảo luận |
| 8 | Nguồn của thảo luận |
| 9 | Tên luồng của thảo luận |
| 10 | Địa chỉ của luồng của thảo luận |

1. Trend: thuật ngữ chuyên ngành nghiên cứu thị trường, ý chỉ xu hướng phát triển của một chiến dịch. [↑](#footnote-ref-1)
2. Thread: nhiều bài viết trong một diễn dàn có liên quan với nhau được gọi là một thread, trong nghiệp vụ phân tích, thread dùng để chỉ các bài viết đang bàn luận về cũng một chủ đề, không hạn chế là chỉ trong diễn đàn. [↑](#footnote-ref-2)
3. Social Media Type: là thuật ngữ chuyên ngành nghiên ý chỉ các loại mạng xã hội, bao gồm: diễn đàn, facebook, twitter, tin tức, rao vặt, website của các nhà cung cấp. [↑](#footnote-ref-3)
4. Oracle: hệ quản trị cơ sở dữ liệu quan hệ phổ biến trên thế giới được phát triển bởi tập đoàn Oracle với hơn 50 năm kinh nghiệm. [↑](#footnote-ref-4)
5. PostgreSQL: hệ quản trị cơ sở dữ liệu quan hệ và đối tượng nguồn mở được khoa điện toán của đại học California tại Berkeley phát triển. [↑](#footnote-ref-5)
6. NEAR: hàm tìm kiếm một từ gần một từ khác. Ví dụ word1 NEAR/3 word2 nghĩa là giữa word1 và word2 có không quá 3 từ khác. [↑](#footnote-ref-6)
7. Full-text search: là một kỹ thuật tìm kiếm trên cở sở dữ liệu là các bài viết dựa trên dữ liệu đã được index. [↑](#footnote-ref-7)
8. MySQL: là hệ quản trị cơ sở dữ liệu tự do nguồn mở phổ biến nhất thế giới bởi tốc độ truy xuất cực nhanh. [↑](#footnote-ref-8)
9. Liên kết: là cách dịch của URL (Uniform Resource Location) sang tiếng Việt, ý chỉ các anchor trong một website. [↑](#footnote-ref-9)
10. Oracle Enterprise: là một sản phẩm thuộc Oracle Database, có phí bản quyền lên tới 47.500 USD cho một processor. [↑](#footnote-ref-10)
11. Source: tập hợp các website nguồn bao gồm tất cả các loại website như mạng xã hội, diễn đàn, trang tin, trang rao vặt, website của các nhà cung cấp… [↑](#footnote-ref-11)
12. Post: là cách gọi chung của bài báo cáo này đối với các bài viết trong forum, các bài báo, các thảo luận, comment trong các mạng xã hội hay cả những feedback của người dùng. [↑](#footnote-ref-12)
13. DOM (Document Object Model): là một API thường có dạng một cây cấu trúc dữ liệu, được dùng để truy xuất các tài liệu dạng HTML và XML. [↑](#footnote-ref-13)
14. Regular Expresion: là một biểu thức thường được dùng để mô tả một mẫu chuỗi. [↑](#footnote-ref-14)
15. Meta canonical: là một tham số thuộc header của HTML, có tác dụng chỉ định một loạt các URL khách nhau thực ra chỉ là một. [↑](#footnote-ref-15)
16. Nested Set Model là cấu trúc dữ liệu dạng cây nhưng được tổ chức để phù hợp lưu trữ trong cơ sở dữ liệu quan hệ. [↑](#footnote-ref-16)
17. Chracter entity: là những ký tự hoặc nhóm ký tự dùng để thay thế một ký tự đã được sử dụng trong HTML với một ý nghĩa đặc biệt nào đó. Ví dụ: &lt; tương đương "<", &gt; tương đương ">". [↑](#footnote-ref-17)
18. LIFO: last in first out. [↑](#footnote-ref-18)
19. Partitioning: là kỹ thuật phân đoạn dữ liệu được hỗ trợ bởi các hệ quản trị cơ sở dữ liệu phức tạp. [↑](#footnote-ref-19)