**LỜI MỞ ĐẦU**

Càng ngày càng có nhiều doanh nghiệp thấy được các lợi ích của thương mại điện tử (TMĐT) và muốn ứng dụng để tiếp thị hay quảng bá thương hiệu của mình. Lợi ích lớn nhất mà TMĐT đem lại chính là sự tiết kiệm chi phí và tạo thuận lợi cho các bên giao dịch. Giao dịch bằng phương tiện điện tử nhanh hơn so với giao dịch truyền thống và có thể tiến hành giao dịch khi ở cách xa nhau, giữa thành phố với nông thôn, từ nước này sang nước kia, hay nói cách khác là không bị giới hạn bởi không gian địa lý.

Thương mại điện tử là lĩnh vực hoạt động kinh tế không còn xa lạ với nhiều quốc gia. Người ta không còn phải mất nhiều thời gian, công sức, tiền bạc...cho những giao dịch kinh tế. Việc áp dụng thương mại điện tử trong hoạt động kinh doanh là một xu thế tất yếu của thời đại. Sự phát triển mạnh mẽ của TMĐT góp phần thúc đẩy sự cạnh tranh giữa các doanh nghiệp để thu được nhiều lợi ích nhất. Điều này đặc biệt quan trọng trong bối cảnh hội nhập kinh tế quốc tế, khi các doanh nghiệp trong nước phải cạnh tranh một cách bình đẳng với các doanh nghiệp nước ngoài. Và Việt Nam – trong quá trình hội nhập không nằm ngoài xu hướng phát triển đó.

Tuy nhiên, trong giai đoạn suy thoái kinh tế như hiện nay, thì việc ứng dụng một công nghệ hay một dịch vụ Công Nghệ Thông Tin(CNTT) đáp ứng việc quản lý tốt, hiệu quả dữ liệu của riêng công ty cũng như dữ liệu khách hàng, đối tác là một trong những bài toán được ưu tiên hàng đầu và đang không ngừng gây khó khăn cho doanh nghiệp. Để có thể quản lý được nguồn dữ liệu đó, ban đầu các doanh nghiệp phải đầu tư, tính toán rất nhiều loại chi phí như chi phí cho phần cứng, phần mềm, mạng, chi phí cho quản trị viên, chi phí bảo trì, sửa chữa,…Ngoài ra họ còn phải tính toán khả năng mở rộng, nâng cấp thiết bị; phải kiểm soát việc bảo mật dữ liệu cũng như tính sẵn sàng cao của dữ liệu. Để giải quyết vấn đề trên thì chúng ta thấy rằng nếu có một nơi tin cậy giúp các doanh nghiệp quản lý tốt nguồn dữ liệu thì khi đó họ sẽ không còn quan tâm đến cơ sở hạ tầng, công nghệ mà chỉ tập trung chính vào công việc kinh doanh nhằm mang lại hiệu quả và lợi nhuận ngày càng cao hơn. Kháiniệm “điện toán đám mây” đã ra đời trong hoàn cảnh đó và đang dần trở nên quen thuộc đối với các doanh nghiệp, đặc biệt là đối với các doanh nghiệp vừa và nhỏ.

Điện toán đám mây (Cloud Computing), còn gọi là điện toán máy chủ ảo, là mô hình điện toán sử dụng các công nghệ máy tính và phát triển dựa vào mạng internet (Theo Wikipedia). Với một số lợi ích cơ bản như : sử dụng các tài nguyên tính toán động, giảm chi phí, giảm độ phức tạp trong cơ cấu doanh nghiệp, kiến trúc hướng dịch vụ… điện toán đám mây đang tạo cơ hội cho các doanh nghiệp hoạt động hiệu quả, thông minh và tiết kiệm chi phí hơn.

Google là một trong những nhà cung cấp dịch vụ điện toán đám mây hiện nay với nền tảng App Engine cho phép người dùng xây dựng và triển khai ứng dụng web Java hay Python trên cơ sở hạ tầng của mình. Google cung cấp cho người dùng một dịch vụ đám mây miễn phí như App Engine để người dùng có thể trải nghiệm các tiện ích từ điện toán đám mây mang lại cũng như có nhiều thời gian hơn để tìm hiểu về những rủi ro về tính an toàn dữ liệu, để từ đó họ có thể mạnh dạn hơn để sử dụng các dịch vụ của điện toán đám mây.

Với nhận định như vậy, trong khuôn khổ thực hiện khóa luận tốt nghiệp, chúng em lựa chọn đề tài**“Xây dựng ứng dụng thương mại điện tử trên nền tảng điện toán đám mây và nền tảngAndroid”.**Đề tài xây dựng trên nền tảng điện toán đám mây nhằm tiết kiệm chi phí và có thể triển khai một cách dễ dàng nhất cho doanh nghiệp. Cùng với công nghệ di động Android sẽ giúp việc giao dịch với khách hàng an toàn hơn và thực hiện mọi lúc mọi nơi với một chiếc điện thoại di động nhỏ bé.

**Mục tiêu đề tài**

Nghiên cứu và triển khai ứng dụng web trên nền tảng App Engine của Google, tìm hiểu API của GAE, sự hỗ trợ của GAE đối với ngôn ngữ Python và Java, dựa vào những kiến thức đã tìm hiểu được để xây dựng Website thương mại điện tử cho doanh nghiệp, phù hợp với môi trường kinh doanh tại Việt Nam và đáp ứng với các mục tiêu sau:

* Phù hợp với xu hướng phát triển TMĐT tại Việt Nam.
* Trên nền tảng Google App Engine và Android
* Đáp ứng các hình thức TTTT.
* Giải pháp phù hợp với doanh nghiệp tại Việt Nam.
* Có tính an toàn bảo mật cao.
* Đa dạng với nhiều loại hình thanh toán trực tuyến

**Các giai đoạn thực hiện đề tài**

Quá trình nghiên cứu đề tài được tiến hành qua các bước như sau:

**Đối tượng, phạm vi nghiên cứu đề tài**

* Đối tượng nghiên cứu đề tài: giải pháp thương mại điện tử trong xu thế phát triển mới ở Việt Nam. Luận văn tập trung vào nghiên cứu nguyên lý và thực tiễn về cách thức hoạt động của các website TMĐT, đưa ra hình thức xây dựng website cho các doanh nghiệp vừa và nhỏ tại Việt Nam. Qua đó xây dựng một ứng dụng trên nền tảng điện toán đám mây của Google nhằm mang lại hiệu quả cao và chi phí thấp nhất cho doanh nghiệp.
* Phạm vi nghiên cứu: cách thức hoạt động TMĐT Việt Nam tập trung đi sâu vào các hình thức quảng cáo sản phẩm và thanh toán trực tuyến. Nền tảng điện toán đám mây của Google, tìm hiểu nghiên cứu ứng dụng GAE APIs của Java, dựa vào những kiến thức đã tìm hiểu được sẽ triển khai một ứng dụng web cho doanh nghiệp.

**Ý nghĩa thực tiễn của đề tài**

**Đề tài góp phần làm cho mọi người hiểu rõ hơn về nguyên lý,cách thức hoạt động của một website TMĐT. Website được triển khai trên nền tảng GAE nhằm tận dụng thế mạnh của điện toán đám mây. Hơn nữa chi phí để thực hiện Website trên GAE phù hợp cho các doanh nghiệp vừa và nhỏ, bởi lẽ trong thời gian đầu GAE sẽ miễn phí cho ứng dụng. Chính vì vậy đây sẽ là một cơ hội tốt để các doanh nghiệp có thể kiểm tra tính hiệu quả của một website trước khi nó mang lại lợi ích về mặt kinh tế.**

Đề tài góp phần cho mọi người hiểu rõ hơn về nguyên lý, cách thức hoạt động của một website TMĐT. Nó đáp ứng các nhu cầu các doanh nghiệp vừa và nhỏ khi không có một website TMĐT chính thống để phục vụ quá trình kinh doanh của mình. Với thế mạnh nền tảng điện toán đám mây của Google như cơ chế bảo mật, quyền riêng tư và các chính sách bảo vệ cơ sở dữ liệu. Các ứng dụng cung cấp của Google sẽ giúp ta dễ dàng quản lý các ứng dụng của mình, dành thời gian tập trung phát triển các chức năng cho người dùng. Và vấn đề bảo mật vốn là rào cản lớn nhất khiến người dùng Việt Nam vốn e dè với điện toán đám mây thì giờ đây với GAE đã khắc phục được yếu điểm đó.

**Cấu trúc khóa luận**

**Chương 1 : Cơ sở lý thuyết**

**Chương 2 : Phân tích và đánh giá một số Website thương mại điện tử**

**Chương 3 : Giải pháp xây dựng Website TMĐT tại Việt Nam**

**Chương 4 : Ứng dụng**

**Chương 5 : Kết luận**

**Phụ lục A: Đăng ký Google App Engine**

**Phụ lục B: Cài đặt môi trường lập trình GAE cho java**

**Phụ lục C: Các văn bản pháp quy liên quan đến thương mại điện tử**

**Tài liệu tham khảo**

**LỜI CẢM ƠN**

Chúng em xin chân thành cảm ơn Khoa Mạng Máy Tính và Truyền Thông, trường

Đại Học Công Nghệ Thông Tin, Đại Học Quốc Gia TP.HCM đ. tạo điều kiện cho chúngem thực hiện đề tài tốt nghiệp này.

Chúng em xin chân thành cảm ơn thầy Phạm Thi Vương và anh Lê Ngọc Thạch đã tận tình hướng dẫn, chỉ bảo chúng em trong suốt thời gian thực hiện đề tài.

Chúng em cũng xin chân thành cảm ơn quý Thầy Cô trong Khoa MMT&TT đã tận

tình giảng dạy, trang bị cho chúng em những kiến thức cần thiết trong suốt quá trình học tập tại Khoa.

Mặc dù đã cố gắng hoàn thành luận văn với tất cả sự nỗ lực của bản thân, nhưng luận

văn chắc chắn không tránh khỏi những thiếu sót, kính mong quý Thầy Cô tận tình chỉ

bảo.

Xin trân trọng cảm ơn!

**Chương 1**

**CƠ SỞ LÝ THUYẾT**

**1.1 Tổng quan về ứng dụng thương mại điện tử**

**1.1.1Định nghĩa thương mại điện tử**

Thương mại điện tử (còn gọi là **E-Commerce** hay **E-Business**) là quy trình mua bán hàng hóa và dịch vụ thông qua các phương tiện điện tử và mạng viễn thông, đặc biệt là qua máy tính và mạng Internet. **Thương mại điện tử** (Electronic Commerce), một yếu tố hợp thành của nền "*Kinh tế số hóa*", là hình thái hoạt động thương mại bằng các phương pháp điện tử; là việc trao đổi *thông tin* thương mại thông qua các phương tiện công nghệ điện tử mà nói chung là không cần phải in ra giấy trong bất cứ công đoạn nào của quá trình giao dịch (nên còn gọi là "*Thương mại không có giấy tờ*").

Tại Việt Nam, thương mại điện tử thường được hiểu theo cả nghĩa hẹp và nghĩa rộng. Theo nghĩa hẹp, thương mại điện tử là việc mua bán hàng hóa và dịch vụ thông qua các phương tiện điện tử và mạng viễn thông. Theo nghĩa rộng, thương mại điiện tử là việc ứng dụng các thành tựu của công nghệ thông tin và truyền thông vào các hoạt động quản lý và kinh doanh.

**1.1.2 Ứng dụng thương mại điện tử tại Việt Nam**

**1.1.2.1 Website thương mại điện tử**

Năm 2010, tỷ lệ doanh nghiệp có website riêng duy trì ở mức 38%.Tuy nhiên, có 21 % doanh nghiệp dự kiến sẽ xây dựng website trong tương lai so với 17% của năm 2009.Tuy tỷ lệ này không cao nhưng với tỷ lệ các doanh nghiệp vừa và nhỏ có website riêng là điều đáng mừng trong xu thế mới. Việc xây dựng trang website, các doanh nghiệp Việt nam đã tạo cho mình một văn phòng giao dịch trực tuyến, chuẩn bị sẵn sàng cho các giao dịch thương mại điện tử trong thời gian tới.

Ngoài ra hàng ngàn các công ty, cơ sở sản xuất, kinh doanh cũng đưa thông tin quảng cáo trên các web site của các nhà cung cấp thông tin trên Internet như VDC, FPT, Netnam, Phương Nam... Khi vào bất kì trang web nào của Việt nam hiện nay, chúng ta đều thấy khá nhiều logo, banner quảng cáo của các tổ chức doanh nghiệp thuộc mọi lĩnh vực của nền kinh tế: từ điện tử, viễn thông, tin học, sản xuất chế biến, xuất nhập khẩu, du lịch đến các cửa hàng kinh doanh, các nhà may, thậm chí cả các phòng tranh của các hoạ sĩ. Trang web càng đẹp, hấp dẫn, càng nhiều người truy cập thì càng có nhiều công ty quảng cáo.

Theo lĩnh vực hoạt động, tỷ lệ các doanh nghiệp Tài chính,CNTT và TMĐT có website riêng cao nhất tương ứng là 64% và 63%. Tỷ lệ doanh nghiệp thuộc lĩnh vực Dịch vụ có website đạt 45%.Các lĩnh vực còn lại, tỷ lệ các doanh nghiệp sở hữu website nằm trong khoảng 27%-39%.Qua những số liệu thông kê trên đã khẵng định TMĐT ngày càng đi sâu và rộng vào trong hoạt động kinh doanh của các doanh nghiệp trong thời đại mới này. Thương mại điện tử trở thành một nhu cầu tất yếu hay một thứ không thể thiếu đối với việc kinh doanh của doanh nghiệp hiện nay.

|  |  |  |
| --- | --- | --- |
| Lĩnh vực | Có website | Sẽ xây dựng Website |
| Tài chính | **64%** | **30%** |
| CNTT, TMĐT | **63%** | **16%** |
| Dịch vụ | **45%** | **25%** |
| Sản xuất, công nghiệp, năng lượng | **39%** | **22%** |
| Thương mại, buôn bán, bán lẻ | **36%** | **20%** |
| Nông, lâm, thủy sản | **32%** | **21%** |
| Xây dựng, vận tải | **27%** | **21%** |
| Khác | **42%** | **9%** |

Vào những ngày đầu phát triển website thương mại điện tử, đó là thứ thật xa vời với những doanh nghiệp nhỏ hay ở những vùng có mức độ phát triển CNTT thấp. Nhưng chỉ trong một thời gian ngắn mọi thứ đã thay đổi theo xu hướng mới. Tỷ lệ các doanh nghiệp sở hữu website thương mại điện tử ở những địa phương khác ngày được tăng lên, điều đó thể hiện rõ nhu cầu TMĐT của các doanh nghiệp nhỏ không kém với các doanh nghiệp lớn hay hàng đầu Việt Nam.

**1.1.2.2 Phần mềm**

Ngày càng nhiểu doanh nghiệp quan tâm, ứng dụng các phẩn mềm TMĐT vào trong việc quản lý quan hệ khách hàng, quản lý sản phẩm, kế toán, nhân sự …Chênh lệch giữa doanh nghiệp nhỏ và vừa và doanh nghiệp lớn trong việc ứng dụng các phẩn mềm phổ thông như phần mềm văn phòng hay phần mềm kế toán không cao. Mặc dù ngày càng nhiều doanh nghiệp nhỏ và vừa quan tâm và bắt đầu ứng dụng các phần mềm chuyên dụng, phần mềm TMĐT, song so sánh với các doanh nghiệp lớn thì tỷ lệ này còn chênh lệch lớn

**1.1.2.3 Sàn giao dịch thương mại điện tử**

Năm 2011 là năm đánh dấu nhiều của sự phát triển của sàn giao dịch thương mại điện tử, đặc biệt đáng kể là một số trang như vatgia.com, nhommua.com, 5giay.com … Tất cả đã tạo nên một khu chợ ảo mà người dùng có thể mua bất kỳ thứ gì mà không cần quan tâm mình đang ở đâu. Mọi thứ thật dễ dàng với người sử dụng với những thao tác đơn giản là có thể mua hàng bằng những tài khoản trực tuyến thông dụng như NganLuong, Paypal … Và tỷ lệ các doanh nghiệp xây dựng sàn giao dịch thương mại điện tử ngày càng gia tăng nhanh đáp ứng xu thế mua hàng và thanh toán trong thời đại số ngày nay. Sau đây là biểu đồ các doanh nghiệp tham gia sàn giao dịch ảo:

**1.1.2.4 Đặt hàng trực tuyến**

Các doanh nghiệp ngày càng chủ động phương tiện CNTT trong việc đặt hàng của mình.Điều đó mang lại sự tiện dụng cho người sử dụng mà còn nhẹ bớt các khâu quản lý, kế toán trong qua trình kinh doanh của doanh nghiệp.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Phương tiện | Điện thoại | Fax | Email | Website |
| Tỷ lệ | **99%** | **90%** | **53%** | **21%** |

Với những cách thức tiếp thị trong đặt hàng trực tuyến đã làm thay đổi dần dần thói quen tiêu dùng của người Việt Nam.Điều đó giúp ích rất nhiều cho chính doanh nghiệp của họ và tạo điều kiện thuận lợi nhất cho khách hàng. Mọi chi phí đi lại, trưng bày sản phẩm, văn phòng đại điện giờ đây được thay thế bằng những website thương mại điện tử với đầy đủ các tính năng như quảng cáo sản phẩm, đặt hàng trực truyến hay cả việc thanh toán trực tuyến. Tất cả mọi thứ tạo nên một động lực tốt cho chúng ta phát triễn TMĐT trong thời kỳ hội nhập ngày nay.

**1.3Maketing trực tuyến**

**1.3.1 Nhu cầu tiếp thị trực tuyến**

Xu hướng truy cập Internet tại gia đình ngày càng tăng cao, cùng với việc bùng nổ công nghệ 3G tại thì trường Việt Nam đã làm quảng cáo ngon trở thành một chiến lược hàng đầu đối doanh nghiệp. Số lượng người mua hàng qua mạng cũng tăng đáng kể và các trang mạng xã hội đang trở thành một hình thức hoạt động phổ biến.Những điều đó cho thấy tiếp thị trực tuyến sẻ trở thành một trong những xu hướng chiến lược trong việc quảng bá thương hiệu, sản phẩm để các doanh nghiệp cạnh tranh với đối thủ, duy trì, mở rộng và phát triển thị trường. Trong thời gian gần đây, nhiều doanh nghiệp đã triễn khai khá thành công các chương trình tiếp thị trực tuyến như Coca-Cola, Zing Me... Những thành công đã có tác dụng khuyến khích doanh nghiệp đầu tư và phát triển các loại hình tiếp thị hay quảng cáo trực tuyến của mình.Sự phát triển của hạ tầng công nghệ và thương mại điện tử, sự hoàn thiện của hệ thống thanh toán và sự gia tăng của hoạt động mua bán trên mạng sẽ tạo nền tảng thúc đẩy sự phát triễn của quảng cáo trực tuyến. Do đó trong thời gian tới nhu cầu quang cáo trực tuyến sẽ ngày càng tăng cao. Để đáp ứng tốt nhu cầu này, doanh nghiệp cung cấp các dịch vụ quảng cáo trực tuyến cần nâng cao chất lượng, đầu tư công nghệ mới để khách hàng được theo dõi hiệu quả của hình thức dịch vụ thông tin này.

**1.3.2 Các loại hình cung cấp tiếp thị trực tuyến**

Thiết kế website cho doanh nghiệp đã trở thành một ngành kinh doanh khá sôi động và thu hút nhiều đối tượng doanh nghiệp tham gia cung cấp dịch vụ. Điều này cho thấy các doanh nghiệp đã nhận thức rõ hiệu quả của việc quảng bá hình ảnh, sản phẩm qua website. Cùng với sự ra đời của phiên bản Google tiếng Việt và dịch vụ Yahoo Search làm cho các doanh nghiệp công nghệ thông tin và thương mại điện tử đã nghiên cứu nghiêm túc để triễn khai them tính năng SEO tại website cho doanh nghiệp nhằm tăng khả năng tìm kiếm, kết nối giữa doanh nghiệp và khách hàng.Ở Việt Nam với sự ra đời hàng loạt các website thương mại điện tử làm cho thị trường quảng cáo trực tuyến ngày càng sôi động hơn và tạo ra nhiều hình thức tiếp thị quảng cáo mới hơn trên thị trường.

Và bên cạnh những website tên tuổi thì thị trường quảng cáo tiếp thị trên mạng xã hội cũng được sự thu hút không ít các doanh nghiệp. Với các ưu điểm của quảng cáo trực tuyến như không bị giới hạn về không gian và thời gian , chi phí hợp lý, dễ xác định phân khúc khách hàng mục tiêu… cộng thêm các ưu điểm đặc biệt của mạng xã hội về tình tương tác cao, hiệu ứng lan truyền theo cấp số nhân, quảng cáo trên mạng xã hội là bước đi đúng cho các doanh nghiệp. Quảng cáo trên mạng xã hội là việc tiến hành một phần hay toàn bộ quy trình tuyên truyền, giới thiệu thông tin về sản phẩm, dịch vụ, doanh nghiệp hay ý tưởng… của doanh nghiệp tới người tiêu dùng thông qua môi trường mạng xã hội. Sau đây là bảng thông kê các mạng xã hội được yêu thích nhất Việt Nam do FTA Viettrack 09/2009 khảo sát:

Quảng cáo trên mạng xã hội là một hình thức mới mẻ và còn chiếm thị phần nhỏ tại thị trường Việt Nam. Tuy nhiên, những ưu điểm của quảng cáo trên mạng xã hội cùng những thành công của nhiều doanh nghiệp khi thử nghiệm hình thức quảng cáo này đã khiến cho quảng cáo trên mạng xã hội trở thành một hiện tượng được chú ý. Khi các mạng xã hội đa trở nên phổ biến, xu hướng quảng cáo trên mạng xã hội tại Việt Nam sẽ còn phát triển mạnh mẽhơn nữa, đòi hỏi sự nghiên cứu và đầu tư nghiêm túc của doanh nghiệp để tối ưu hóa chiến lược quảng cáo của mình. Quảng cáo trên mạng xã hội sẽ góp phần thúc đẩy sự phát triển của quảng cáo trực tuyến tại Việt Nam trong tương lai.

Thế giới đã và đang chứng kiến sự phát triển nhanh chóng và mạnh mẽ của báo điện tử nói riêng và các phương tiện truyền thông trực tuyến nói chung so với các phương tiện báo chí truyền thống khác như báo in, phát thanh, truyền hình. Báo điện tử có sự tổng hợp của công nghệ đa phương tiện, nghĩa là không chỉ văn bản, hình ảnh mà cả âm thanh, hình ảnh động và các tính năng tương tác khác. Đồng thời, nhờ tích hợp công cụ tìm kiếm thông tin khoa học và hiệu quả, người đọc dễ dàng tìm kiếm thông tin. Với những ưu thế này, báo điện tử đang trở thành kênh truyền thông được nhiều người lựa chọn.



Với sự phát triển nhanh chóng của loại hình báo điện tử và số lượng người truy cập Internet để đọc tin tức ngày càng lớn, quảng cáo trực tuyến trên các báo điện tử ở Việt Nam trong thời gian qua cũng có bước phát triển tích cực. Tuy nhiên, phần lớn thị phần quảng cáo trực tuyến ở Việt Nam đều thuộc về một vài tờ báo điện tử có đông người truy cập như VnExpress, VietnamNet, Dantri. Với các báo in có trang tin điện tử thì quảng cáo trên báo điện tử vẫn còn hạn chế.

Tại Việt Nam, hình thức quảng cáo phổ biến trên báo điện tử là đặt banner, logo, hoặc pop-up. Các dạng quảng cáo như qua từ khóa, quảng cáo theo ngữ cảnh, theo hành vi... còn là những khái niệm mới mẻ. Hiện nay cũng chưa có một chuẩn nào đối với các mẫu thiết kế cho quảng cáo trực tuyến (kích thước, vị trí, v.v...). Điều này khiến doanh nghiệp mất thêm nhiều thời gian và chi phí khi tiến hành quảng cáo tại các website khác nhau. Điển hình như vnexpress.net, dantri.com, nếu tính sơ bộ thì chỉ riêng trang chủ mỗi ngày có hơn 20 banner được đặt dưới đủ mọi hình thức. Các cách thức quảng cáo để lôi kéo người xem click vào các banner cũng ngày một đa dạng hơn.

Tuy nhiên, tương tự hình thức sử dụng banner quảng cáo trên mạng xã hội, hiện nay trên các trang báo xuất hiện quá nhiều banner gây tác dụng phản cảm đối với độc giả. Việc sắp đặt các banner nhiều màu sắc cạnh nhau phá hỏng sự hài hòa và mang lại cảm giác rối mắt cho người xem.

Những nguyên nhân này không chỉ ảnh hưởng đến các tờ báo điện tử về nguồn thu, mà còn ảnh hướng đến chính các doanh nghiệp đi quảng cáo vì không tận dụng kênh tiếp thị hiệu quả để giúp phát triển doanh nghiệp. Đã đến lúc lĩnh vực này cần đến một đơn vị trung gian đo kiểm kết quả truy cập và hiệu quả thông tin trên từng trang quảng cáo trực tuyến một cách chính xác và kịp thời để hỗ trợ thông tin cho bạn đọc và giúp các doanh nghiệp dễ dàng ra quyết định khi đặt quảng cáo.

**1.3.3 Thực trạng ứng dụng quảng cáo trực tuyến**

Các doanh nghiệp, cá nhân cũng quan tâm và đẩy mạnh việc quảng bá thương hiệu thông qua mạng xã hội. Quảng cáo hiển thị (đặc biệt là hiển thị banner) là hình thức quảng cáo được sử dụng rộng rãi nhất trên các mạng xã hội ở Việt Nam hiện nay. Đây là hình thức quảng cáo đơn giản, tiết kiệm thời gian quản lý và mang lại hiệu quả cao nhờ khả năng hiển thị thông điệp quảng cáo khá tốt. Điển hình, Samsung đã thực hiện các chiến dịch quảng cáo nhắm vào đối tượng là những người trẻ sử dụng mạng xã hội Facebook. Thương hiệu này đã liên tục xuất hiện trên Facebook cho mỗi chiến dịch quảng cáo sản phẩm mới cùng nhiều hình thức khuyến mãi hấp dẫn.

Diễn đàn là nơi chứng kiến mô hình quảng cáo tương tác đầu tiên ở Việt Nam, với một số diễn đàn nổi bật như www.muare.vn, www.5giay.vn, webtretho.com, enbac.com, v.v… Doanh nghiệp hoặc cá nhân có thể lập chủ đề (topic) để bán hàng hoặc quảng cáo dịch vụ của mình và trực tiếp trao đổi, giao lưu với khách hàng trên topic. Việc lập các topic để quảng cáo hay giới thiệu sản phẩm hoàn toàn miễn phí, nhưng các topic này sẽ bị đẩy lùi xuống các trang sau mỗi khi có các chủ đề mới được lập ra hoặc khi các chủ đề cũ có những bài viết mới. Từ đó nảy sinh nhu cầu sở hữu các “topic dính”. Đó là topic luôn nằm cố định ở khu vực riêng đầu trang 1 của các chuyên mục nên hiệu quả quảng cáo cũng lớn hơn rất nhiều. Hiện nay, nhu cầu lưu thông tin quảng cáo sử dụng topic dính tại các diễn đàn đang rất cao, nhất là các diễn đàn lớn có số lượng truy cập cao.

Muare.vn là một trong những website có hoạt động đấu giá “topic dính” nhộn nhịp nhất vì số lượng thành viên cũng như khách truy cập luôn lên đến con số hàng nghìn. Bắt đầu từ tháng 8/2008, khi nhu cầu sở hữu topic dính tăng cao, ban quản trị website muare.vn quyết định tổ chức đấu giá công khai các topic dính trên diễn đàn này. Hoạt động đấu giá này diễn ra hai tháng một lần, mỗi lần kéo dài 24h, từ 12h trưa hôm trước đến 12h trưa hôm sau, với thủ tục để tham gia đấu giá tương đối đơn giản. Một số diễn đàn như 5giay.vn, webtretho.com, v.v… không áp dụng hình thức đấu giá như muare.vn mà có mức giá cố định cho “topic dính” được gắn cố định ở vị trí đầu tiên của mỗi chuyên trang với mức giá đa dạng tùy theo từng chuyên mục. Các diễn đàn này cũng đang thu hút một lượng khách hàng lớn có nhu cầu quảng bá thương hiệu.

Tiếp thị trực tuyến đang phát triển một cách nhanh chóng và đa dạng về cả hình thức cũng như phương pháp ứng dụng và triển khai. Cùng với sự tiến bộ ngày càng nhanh của công nghệ thông tin và thương mại điện tử, các hình thức tiếp thị trực tuyến sẽ trở thành xu hướng tất yếu trong tương lai với các doanh nghiệp muốn gây dựng thương hiệu, quảng bá hàng hóa và dịch vụ. Tuy nhiên, để đạt được hiệu quả cao nhất doanh nghiệp cần phối hợp chặt chẽ giữa tiếp thị truyền thống và tiếp thị trực tuyến nhằm khai thác được thế mạnh riêng của từng hình thức.

**1.4 An toàn thông tin trong giao dịch TMĐT**

**1.4.1 Thực trạng an toàn thông tin trong giao dịch TMĐT tại Việt Nam**

Hạ tầng công nghệ thông tin và truyền thông đã và đang phát triển mạnh mẽ tại Việt Nam. Theo thống kê của Trung tâm Internet Việt Nam - VNNIC, tính đến tháng 12 năm 2010 số người sử dụng Internet tại Việt Nam đạt mức 26,7 triệu, chiếm khoảng 31,1% dân số cả nước. Tổng dung lượng kênh kết nối quốc tế của Việt Nam là 129.877 Mbps, tổng dung lượng kênh kết nối trong nước là 245.857 Mbps.

Tuy nhiên, tại Việt Nam sự phát triển nhanh chóng của Internet chưa kết hợp với các giải pháp an toàn thông tin tương ứng. Các sự cố lớn gần đây với mạng Internet như sự cố phân giải tên miền, lan truyền mã độc, xâm nhập hệ thống, từ chối dịch vụ, tấn công website là minh chứng cho điều này và đang trở thành vấn đề lớn cho an toàn thông tin tại Việt Nam. Cùng với sự phát triển của Internet và các phương tiện điện tử, giao dịch điện tử ngày càng phát triển và trở nên phổ biến. Cùng với sự phát triển của hình thức giao dịch điện tử, vấn đề an toàn an ninh mạng đang trở nên ngày càng cấp bách.

Nguồn nhân lực công nghệ thông tin bao gồm nhân lực làm việc trong các doanh nghiệp viễn thông, doanh nghiệp công nghiệp công nghệ thông tin; nhân lực cho ứng dụng công nghệ thông tin; nhân lực cho đào tạo công nghệ thông tin, điện tử, viễn thông và người dân sử dụng các ứng dụng công nghệ thông tin. Phát triển nguồn nhân lực công nghệ thông tin là yếu tố then chốt có ý nghĩa quyết định đối với việc ứng dụng và phát triển công nghệ thông tin do đó phải được đảm bảo về chất lượng, đồng bộ, chuyển dịch nhanh về cơ cấu theo hướng tăng nhanh tỷ lệ nguồn nhân lực có trình độ cao, tăng cường năng lực công nghệ thông tin quốc gia.

**1.4.2Những mối đe dọa an toàn thông tin trong giao dịch TMĐT**

**Tội phạm sử dụng công nghệ cao**

Trong thời gian vừa qua đã xảy ra nhiều vụ tấn công với mục tiêu chủ yếu là mạng máy tính của các cơ quan, tổ chức truyền thông báo chí, làm gián đoạn thông tin và gây tác hại không nhỏ cho các tổ chức này. Thủ đoạn phá hoại chủ yếu thông qua việc phát tán virus, phần mềm gián điệp, điều khiển từ xa, sâu độc, v.v… Phổ biến nhất là cài phần mềm gián điệp vào các máy tính trong một hệ thống để lấy thông tin mật khẩu, nick chat,v.v... Ngoài ra, các tin tặc cũng lợi dụng các trang mạng xã hội để mở rộng phạm vi phá hoại.

Một hình thức tấn công khác là truy cập, sử dụng trái phép dữ liệu và đưa thông tin trái phép lên mạng. Do hệ thống mạng của một số cơ quan, tổ chức chưa đạt yêu cầu về an toàn thông tin, website do các tổ chức này quản lý đã bị tin tặc tấn công, gây ảnh hưởng khá nghiêm trọng đến uy tín và hoạt động. Ngoài ra, các website, diễn đàn do các nhóm cá nhân quản lý cũng thường xuyên bị tấn công và đăng tải những nội dung mang tính đồi trụy, hăm dọa người khác, vi phạm pháp luật hoặc dẫn tới hành vi phạm pháp.

**Lừa đảo trong thương mại điện tử**

Một trong những rào cản khiến người dùng e ngại khi tiến hành giao dịch TMĐT, đó là nguy cơbị tổn hại lợi ích do các hành vi lừa đảo trên môi trường mạng. Một số loại hình tội phạm phổbiến ghi nhận được trong thời gian qua là lừa đảo qua quảng cáo, bán hàng trực tuyến trênmạng Internet, trong mua bán ngoại tệ, vàng, huy động vốn tín dụng; gửi email thông báo trúngthưởng xổ số lớn, đề nghị tham gia rửa tiền, đề nghị nhận hộ tiền thừa kế, v.v…

Với sự hỗ trợ của công nghệ hiện đại và các thủ tục đăng ký tên miền khá đơn giản, việc giao dịchmua bán qua website đã trở nên thuận tiện hơn trước. Tuy nhiên, đây cũng là kẽ hở để kẻ gian lợidụng, tiến hành những thủ đoạn lừa đảo, xâm hại tới quyền lợi của các bên tham gia, đặc biệt là khách hàng. Thủ đoạn của bọn tội phạm thường là tạo trang web bán hàng giả bằng cách đăng ký tên miền và mua tên miền, tạo trang web giống trang web bán hàng thật, trong đó mọi mặt hàng đều có giá bán rẻ hơn trang web bán hàng thật. Trên trang thanh toán, khách hàng điềnthông tin cá nhân, thông tin thẻ tín dụng và được chuyển trực tiếp về email của tội phạm.

**Hành vi gây rối, cạnh tranh thiếu lành mạnh**

Bên cạnh những hành vi tội phạm với mục tiêu chiếm đoạt tài sản thông qua sử dụng công nghệ cao hoặc lửa đảo trong giao dịch, còn tồn tại những hành vi gây rối gây ảnh hưởng trực tiếp và gián tiếp tới môi trường kinh doanh trực tuyến. Một số hành vi mà tội phạm thường sử dụng là thâm nhập trái phép hệ thống thông tin của doanh nghiệp, tấn công từ chối dịch vụ DDOS, phát tán virus và các mã độc hại, v.v... Những hành vi này đã gây trở ngại tới việc tiếp cận thông tin của người dùng, việc phổ biến thông tin của quản trị website và có thể làm tê liệt hoạt động trao đổi thông tin hay tác nghiệp của các cơ quan, tổ chức, cá nhân do bị virus phá hủy tài liệu lưu trữ.

**1.4.3 Một số giải pháp đẩy mạnhan toàn thông**

**Nâng cao nhận thức và đẩy mạnh tuyên truyền về an toàn thông tin**

Sử dụng các phương tiện thông tin đại chúng, tổ chức các sự kiện, hội nghị, hội thảo để tuyêntruyền nâng cao nhận thức cho cơ quan nhà nước, doanh nghiệp và người dân về an toàn thông tin.

**Hoàn thiện các cơ chế, chính sách về an toàn thông tin**

Rà soát và hoàn thiện các văn bản quy phạm pháp luật, cơ chế và chính sách của nhà nước, tạo môi trường thuận lợi để đảm bảo an toàn thông tin, đáp ứng các yêu cầu về hội nhập toàn diệnkinh tế quốc tế, thúc đẩy hợp tác và cạnh tranh lành mạnh giữa các doanh nghiệp. Tăng cườngcác khung hình phạt xử lý mạnh và kiên quyết khi có vi phạm về an toàn thông tin. Đây là nhữngcăn cứ pháp lý quan trọng để đấu tranh với các hành vi tấn công bất hợp pháp vào các hệ thốngmạng máy tính, cơ sở dữ liệu.

**Phát triển nguồn nhân lực về an toàn thông tin**

Đào tạo nguồn nhân lực riêng về lĩnh vực an toàn thông tin thông qua việc xây dựng hệ thốngtiêu chí kỹ năng cần thiết đối với các chuyên gia an toàn thông tin. Xây dựng chương trình và tổchức đào tạo đội ngũ chuyên gia trong lĩnh vực đảm bảo an toàn thông tin phù hợp với yêu cầucủa giai đoạn cạnh tranh và hội nhập. Phát triển nguồn nhân lực có khả năng tiếp nhận chuyểngiao công nghệ, tránh bị lệ thuộc vào nước ngoài.

**Tăng cường hợp tác quốc tế về an toàn thông tin**

Tăng cường hợp tác phòng chống tấn công mạng thông qua việc chia sẻ, trao đổi thông tin giữacác quốc gia trong khu vực và trên thế giới. Đẩy mạnh hợp tác với các tổ chức quốc tế trong lĩnhvực an toàn thông tin, phối hợp trao đổi, đào tạo chuyên gia trong lĩnh vực kỹ thuật và quản lýan toàn thông tin. Phối hợp giữa các đơn vị tư vấn, chuyên gia an ninh thông tin sẵn sàng ứngphó với những sự cố liên quan tới mất an toàn thông tin. Thu hút các dự án đầu tư nước ngoàidựa trên cơ sở chuyển giao công nghệ, từng bước tiến tới làm chủ công nghệ và phát triển cácsản phẩm an toàn thông tin đặc thù của Việt Nam.

**1.5Yêu cầu đối với thương mại điện tử**

**Hạ tầng cơ sở công nghệ**

Chỉ có thể tiến hành thực tế và một cách có hiệu quả thương mại điện tử khi đã có một hạ tầng cơ sở công nghệ thông tin đủ năng lực, bao gồm hai nhánh: tính toán (computing) và [truyền thông](http://vi.wikipedia.org/wiki/Truy%E1%BB%81n_th%C3%B4ng) (communications), hai nhánh này ngoài công nghệ thiết bị còn cần phải có một nền công nghiệp điện tử vững mạnh làm nền; và hiện nay đang có xu hướng đưa cả công nghệ [bảo mật](http://vi.wikipedia.org/wiki/B%E1%BA%A3o_m%E1%BA%ADt) và an toàn vào cơ sở hạ tầng công nghệ của thương mại điện tử. Đòi hỏi về hạ tầng cơ sở công nghệ bao gồm hai mặt: một là tính tiên tiến, hiện đại về công nghệ và thiết bị, hai là tính phổ cập về kinh tế (đủ rẻ tiền để đông đảo con người có thể thực tế tiếp cận được).

**Hạ tầng cơ sở nhân lực**

Hoạt động [thương mại](http://vi.wikipedia.org/wiki/Th%C6%B0%C6%A1ng_m%E1%BA%A1i), theo đúng nghĩa của chữ *"thương mại"* trong *"thương mại điện tử"*, liên quan tới mọi người, từ [người tiêu dùng](http://vi.wikipedia.org/wiki/Ng%C6%B0%E1%BB%9Di_ti%C3%AAu_d%C3%B9ng) đến [doanh nghiệp](http://vi.wikipedia.org/wiki/Doanh_nghi%E1%BB%87p), phân phối, [Chính phủ](http://vi.wikipedia.org/wiki/Ch%C3%ADnh_ph%E1%BB%A7), các nhà công nghệ, nên việc áp dụng thương mại điện tử tất yếu đòi hỏi đa số con người phải có kỹ năng thực tế ứng dụng công nghệ thông tin một cách có hiệu quả, có thói quen làm việc trên [máy tính](http://vi.wikipedia.org/wiki/M%C3%A1y_t%C3%ADnh), trên [mạng máy tính](http://vi.wikipedia.org/wiki/M%E1%BA%A1ng_m%C3%A1y_t%C3%ADnh), và cần phải có một đội ngũ chuyên gia thông tin đủ mạnh.

**Bảo mật, an toàn**

Giao dịch thương mại qua các phương tiện điện tử, trong đó mọi [dữ liệu](http://vi.wikipedia.org/wiki/D%E1%BB%AF_li%E1%BB%87u) (kể cả [chữ ký](http://vi.wikipedia.org/wiki/Ch%E1%BB%AF_k%C3%BD_%C4%91i%E1%BB%87n_t%E1%BB%AD)) đều ở dạng số hoá, đặt ra yêu cầu nghiêm ngặt về tính [bảo mật](http://vi.wikipedia.org/wiki/B%E1%BA%A3o_m%E1%BA%ADt), an toàn, tránh mất tiền, lừa gạt, thay đổi [thông tin](http://vi.wikipedia.org/wiki/Th%C3%B4ng_tin), xâm nhập dữ liệu, v.v... là các [rủi ro](http://vi.wikipedia.org/wiki/R%E1%BB%A7i_ro) ngày một lớn, không chỉ với người buôn bán, mà cả với người quản lý, với từng quốc gia, vì các hệ thống điện tử có thể bị kẻ xấu (thường gọi là *"hacker"*) xâm nhập, đòi hỏi phải có các hệ thống bảo mật, an toàn được thiết kế trên cơ sở kỹ thuật mã hóa hiện đại, và một cơ chế an ninh hữu hiệu (nhất là đối với các hệ thống có liên quan tới an ninh quốc gia). Ngoài ra, còn có nhu cầu ngày càng tăng vì giữ gìn bí mật riêng tư.

**Hệ thống thanh toán tự động**

Thương mại điện tử chỉ có thể thực hiện thực tế và có hiệu quả khi đã tồn tại một hệ thống thanh toán tài chính ở mức độ phát triển đủ cao, cho phép tiến hành thanh toán tự động (trong đó *"thẻ thông minh"*) có vai trò đặc biệt quan trọng trong kinh doanh bán lẻ; khi chưa có hệ thống này, thì thương mại điện tử chỉ giới hạn trong khâu trao đổi tin tức, còn việc buôn bán hàng hóa và dịch vụ vẫn phải kết thúc bằng trả tiền trực tiếp hoặc qua các phương tiện thanh toán truyền thông; hiệu quả sẽ thấp, rất có thể không đủ bù đắp chi phí trang bị phương tiện thương mại điện tử.

**Bảo vệ sở hữu trí tuệ**

Giá trị của sản phẩm ngày nay tập trung ở *"chất xám"*; tài sản của con người, của quốc gia, đang quy dần về *"tài sản chất xám"*, thông tin trở thành tài sản, và bảo vệ tài sản cuối cùng sẽ trở thành bảo vệ sở hữu trí tuệ. Vì thế trong việc truyền gửi các dữ liệu qua mạng nổi lên vấn đề bảo vệ sở hữu chất xám và bản quyền của các thông tin (hình thức quảng cáo, nhãn hiệu thương mại, cấu trúc cơ sở dữ liệu, các nội dung truyền gửi), ở các khía cạnh phức tạp hơn nhiều so với việc bảo vệ sở hữu trí tuệ trong nền kinh tế vật thể.

**Bảo vệ người tiêu dùng**

Bảo vệ người tiêu dùng là mục tiêu ngày càng được đề cao trong thương mại. Quy cách phẩm chất hàng hóa, và các thông tin liên quan trong thương mại điện tử đều ở dạng số hóa, nên người mua chịu rủi ro lớn hơn so với giao dịch thương mại vật thể; để bổ cứu, phải có cơ chế trung gian đảm bảo chất lượng, là một khía cạnh đang nổi lên trước thực tiễn rủi ro đang ngày càng gia tăng trong giao dịch thương mại điện tử, xâm phạm vào quyền lợi của người tiêu dùng.

**Môi trường kinh tế và pháp lý**

Mỗi một quốc gia, thương mại điện tử chỉ có thể tiến hành khi tính pháp lý của nó được thừa nhận (biểu hiện cụ thể bằng sự thừa nhận pháp lý giá trị của các giao dịch điện tử, [chữ ký điện tử](http://vi.wikipedia.org/wiki/Ch%E1%BB%AF_k%C3%BD_%C4%91i%E1%BB%87n_t%E1%BB%AD), chữ ký số hóa, các thanh toán điện tử, các dữ liệu có xuất xứ từ các cơ quan nhà nước, sở hữu trí tuệ hàm chứa trong thông tin trên Website, bí mật đời tư, và bảo vệ pháp lý đối với mạng thông tin chống tội phạm xâm nhập), và có các cơ quan xác thực hoặc chứng nhận chữ ký điện tử, v.v...; Ngoài ra, còn đòi hỏi mọi doanh nghiệp, hàng hóa và dịch vụ đều đã được mã hóa thống nhất; một hệ thống thuế thích hợp để xử lý các dữ liệu và các dịch vụ mua bán qua mạng; nói cách khác, đòi hỏi phải có một môi trường kinh tế đã tiêu chuẩn hóa ở mức cao, với các khía cạnh của thương mại điện tử được phản ánh đầy đủ trong quan hệ nội luật. Trên bình diện quốc tế, vấn đề môi trường pháp lý còn phức tạp hơn nữa, vì các trao đổi là xuyên quốc gia, đòi hỏi phải có sự hài hòa giữa các hệ thống pháp luật và hệ thống chính trị khác nhau.

**Tác động văn hoá xã hội**

Tác động văn hóa xã hội của thương mại điện tử xuất hiện khi sử dụng Internet làm công cụ giao tiếp, như khi tiến hành thương mại điện tử qua biên giới (với nước khác), hoặc nếu trong một quốc gia nhưng sử dụng Iternet, Web làm công cụ mạng. Internet có thể trở thành *"hộp thư"* giao dịch mua bán dâm, ma tuý, và buôn lậu. Internet cũng có thể trở thành một phương tiện thuận lợi cho các lực lượng chống đối sử dụng để tuyên truyền, kích động lật đổ Chính phủ và hoặc gây rối làm loạn trật tự xã hội.

**Lệ thuộc công nghệ**

[Hoa Kỳ](http://vi.wikipedia.org/wiki/Hoa_K%E1%BB%B3) đang khống chế toàn bộ công nghệ thông tin trên thế giới, cả phần cứng cũng như phần mềm (bao gồm phần mềm hệ thống và phần mềm ứng dụng), chuẩn công nghệ Internet, các phần mềm tìm kiếm và trình duyệt chủ yếu cũng là của Hoa Kỳ, Hoa Kỳ đã đi đầu và có nhiều thực tiễn về kinh tế số hóa và thương mại điện tử. Một khi thương mại được số hóa thì toàn thế giới sẽ chịu sự khống chế công nghệ Hoa Kỳ và các nước tiên tiến gần với Hoa Kỳ, là điều có thể đưa tới nhiều hệ quả.

**1.6Lợi ích của ứng dụng TMĐT**

**Quảng bá thông tin và tiếp thị cho một thị trường toàn cầu với chi phí cực thấp:** chỉ với vài chục đô-la Mỹ mỗi tháng, bạn đã có thể đưa thông tin quảng cáo của bạn đến với vài trăm triệu người xem từ các nơi trên thế giới. Đây là điều mà chỉ có Thương Mại Điện Tử làm được cho doanh nghiệp. Thử so sánh với một quảng cáo trên báo Tuổi Trẻ với vài triệu độc giả, mỗi lần quảng cáo bạn phải trả ít nhất 50 đô-la Mỹ, còn nếu bạn có một website của mình, bạn có thể quảng cáo thông tin 24 giờ mỗi ngày, 7 ngày mỗi tuần, và lượng độc giả của bạn là hàng trăm triệu người từ mọi nơi trên thế giới. Chi phí cho website của bạn mỗi tháng ước tính (kinh tế nhất) là: 5 đô-la Mỹ chi phí lưu trữ trực tuyến (hosting), 10-20 đô-la Mỹ trả cho chi phí quảng cáo (liệt kê địa chỉ web của bạn trên một dạng danh bạ doanh nghiệp điện tử.

**Dịch vụ tốt hơn cho khách hàng:** với Thương Mại Điện Tử, bạn có thể cung cấp catalogue, brochure, thông tin, bảng báo giá cho đối tượng khách hàng một cách cực kỳ nhanh chóng, bạn có thể tạo điều kiện cho khách hàng mua hàng trực tiếp từ trên mạng v.v… Nói tóm lại, Thương Mại Điện Tử mang lại cho bạn các công cụ để làm hài lòng khách hàng, bởi trong thời đại ngày nay, yếu tố thời gian thực sự là vàng bạc, không ai có đủ kiên nhẫn phải chờ đợi thông tin trong vài ngày. Hơn nữa, ngày nay chất lượng dịch vụ và thái độ phục vụ là những yếu tố rất quan trọng trong việc tìm và giữ khách hàng. Nếu bạn không xử lý yêu cầu thông tin của đối tượng quan tâm một cách nhanh chóng, họ sẽ không kiên nhẫn mà chờ bạn, trong khi đó có biết bao đối thủ cạnh tranh đang săn đón họ.

**Tăng doanh thu:** với Thương Mại Điện Tử, đối tượng khách hàng của bạn giờ đây đã không còn bị giới hạn về mặt địa lý, hay thời gian làm việc. Bạn không chỉ có thể bán hàng cho cư dân trong thành phố của bạn, mà bạn còn có thể bán hàng trong toàn bộ Viêt Nam hoặc các nước khác.Bạn không ngồi chờ khách hàng tự tìm đến với bạn mà bạn đang tích cực và chủ động đi tìm khách hàng cho mình. Vì thế, chắc chắn rằng số lượng khách hàng của bạn sẽ tăng lên đáng kể dẫn đến tăng doanh thu. Đó là điều mà doanh nghiệp nào cũng mơ ước.

**Giảm chi phí hoạt động:** với Thương mại điện tử, bạn không phải tốn kém nhiều cho việc thuê cửa hàng, mặt bằng, đông đảo nhân viên phục vụ, bạn cũng không cần phải đầu tư nhiều cho kho chứa... Chỉ cần khoảng 10 triệu đồng xây dựng một website bán hàng qua mạng, sau đó chi phí vận hành website mỗi tháng không quá một triệu đồng. Nếu website của bạn chỉ là trưng bày thông tin, hình ảnh sản phẩm, bạn tiết kiệm được chi phí in ấn brochure, catalogue và cả chi phí gửi bưu điện những ấn phẩm này. Và đặc biệt nếu doanh nghiệp bạn làm hàng xuất khẩu, bạn có thể ngồi ở nhà và tìm kiếm khách hàng qua mạng, không cần phải tốn kém nhiều cho những chuyến đích thân “xuất ngoại”.

**Lợi thế cạnh tranh:** việc kinh doanh trên mạng là một “sân chơi” cho sự sáng tạo, nơi đây, bạn tha hồ áp dụng những ý tưởng hay nhất, mới nhất về dịch vụ hỗ trợ, chiến lược tiếp thị v.v… Và một khi tất cả các đối thủ cạnh tranh của bạn đều áp dụng Thương Mại Điện Tử, thì phần thắng sẽ thuộc về ai sáng tạo hay nhất để tạo ra nét đặc trưng cho doanh nghiệp, sản phẩm, dịch vụ của mình để có thể thu hút và giữ được khách hàng.

**An toàn tuyệt đối:** người dùng sẽ không còn lo ngại của việc lừa đảo trên mạng nữa thay vào đó họ sẻ được dùng một ví điện tử để mua hàng. Nó được đảm bảo rất khắc khe, an toàn cao với những công nghệ mới hiện nay.

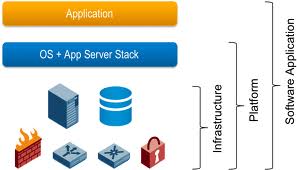
**Chương 3**

**GIẢI PHÁP XÂY DỰNG WEBSITE THƯƠNG MẠI ĐIỆN TỬ TẠI VIỆT NAM**

**3.1 Giải pháp xây dựng website thương mại điện tử tại Việt Nam**

**3.1.1 Cloud Computinglàcơ sở hạ tầng phát triễn ứng dụng thương mại điện tử**

Trong giai đoạn suy thoái kinh tế như hiện nay, việc ứng dụng một công nghệ hay một dịch vụ CNTT đáp ứng việc quản lý tốt, hiệu quả dữ liệu của riêng công ty cũng như dữ liệu khách hàng, đối tác là một trong những bài toán được ưu tiên hàng đầu và đang không ngừng gây khó khăn cho doanh nghiệp. Để có thể quản lý được nguồn dữ liệu đó, ban đầu các doanh nghiệp phải đầu tư, tính toán rất nhiều loại chi phí như chi phí cho phần cứng, phần mềm, mạng, chi phí cho quản trị viên, chi phí bảo trì, sửa chữa,…Ngoài ra họ còn phải tính toán khả năng mở rộng, nâng cấp thiết bị; phải kiểm soát việc bảo mật dữ liệu cũng như tính sẵn sàng cao của dữ liệu. Để giải quyết vấn đề trên thì chúng ta thấy được rằng nếu có một nơi tin cậy giúp các doanh nghiệp quản lý tốt nguồn dữ liệu đó, các doanh nghiệp sẽ không còn quan tâm đến cơ sở hạ tầng, công nghệ mà chỉ tập trung chính vào công việc kinh doanh của họ thì sẽ mang lại cho họ hiệu quả và lợi nhuận ngày càng cao hơn. Khái niệm “**điện toán đám mây**” đã ra đời trong hoàn cảnh đó và đang dần trở nên quen thuộc đối với các doanh nghiệp, đặc biệt là đối với các doanh nghiệp vừa và nhỏ.



**Mô hình dịch vụ điện toán đám mây**

**Với điện toán đám mây doanh nghiệp được nhiều lợi ích và cụ thể như sau:**

* *Tự phục vụ theo nhu cầu (On-demand self-service)*: Mọi yêu cầu đều được thực hiện thông qua việc gửi yêu cầu qua môi trường internet, không cần tương tác trực tiếp với người cung cấp dịch vụ.
* *Truy xuất diện rộng (Broad network access)*: Cloud Computing cung cấp mọi thứ trên nền tảng internet ( mọi thứ đều nằm trên mạng, người dùng chỉ cần một thiết bị đầu cuối có khả năng truy cập mạng và sử dụng). Chính vì thế với thiết bị cầm tay, có khả năng di dộng cao…. Vẫn có thể sử dụng dịch vụ Cloud Computing, điều đó dẫn đến tiềm năng mở rộng cao.
* *Dùng chung tài nguyên (Resource pooling)*: Tài nguyên nhà cung cấp dịch vụ được sử dụng để cung cấp cho nhiều người.Nhờ công nghệ ảo hóa,tài nguyên sẽ được phân phát theo nhu cầu người dùng theo từng thời điểm xác định.Nhờ đó mà nâng cao khả năng phục vụ cho nhiều người dùng trên một đơn vị tài nguyên hiện có.
* *Khả năng co giãn (Rapid elasticity):* Công nghệ Cloud Computing cho phép khả năng tự động mở rộng hoặc thu nhỏ hệ thống tùy theo nhu cầu của người dùng. Khi nhu cầu tăng cao, hệ thống sẽ tự mở rộng bằng cách thêm tài nguyên vào. Khi nhu cầu giảm xuống, hệ thống sẽ tự giảm bớt tài nguyên. Mọi thứ đều được thực hiện một cách linh động và mềm dẻo,hoàn toàn không thay đổi chất lượng dịch vụ với người dùng. Nhờ đó mà nhà cung cấp giảm thiểu được tài nguyên dư thừa,đồng thời khách hàng cũng được hưởng lợi từ việc giảm thiểu chi phí không cần thiết.
* *Điều tiết dịch vụ (Measured service)*: Nhờ khả năng co giãn mà hệ thống có thể tự động kiểm soát và tối ưu hóa việc sử dụng tài nguyên (dung lượng lưu trữ, đơn vị xử lý, băng thông…). Đảm bảo lợi ích cho cả người dùng và nhà cung cấp một cách minh bạch, rõ ràng.

Với điện toán đám mây chúng ta có thể dễ dàng lựa chọn các hình thức dịch vụ khác nhau phục vụ cho ứng dụng thương mại điện tử của mình.Dịch vụ nền tảng **PaaS** (Platform as a Service) là lựa chọn tốt nhất cho các doanh nghiệp muốn phát triển các phần mềm, phục vụ nhu cầu tính toán hoặc xây dựng thành dịch vụ website thương mại điện tử, mà không quan tâm đến chi phí hay sự phức tạp của việc trang bị và quản lý các lớp phần cứng và phần mềm bên dưới.Dịch vụ App Engine của Google là một dịch vụ PaaS điển hình, cho phép khách hàng xây dựng các ứng dụng web với môi trường chạy ứng dụng và phát triển dựa trên các ngôn ngữ lập trình chạy trên máy ảoJava hoặc Python.

**3.1.2 Lựa chọn nền tảng xây dựng website thương mại điện tử**

**Google App Engine (GAE)** là một nền tảng để phát triển và lưu trữ những ứng dụng web trong những trung tâm dữ liệu do Google quản lý (Google-managed data centers). Và nó là lựa chọn sáng giá nhất cho các doanh nghiệp muốn xây dựng những ứng dụng với “chi phí thấp” mà vẫn đảm bảo “chất lượng tốt” như các nền tảng khác trong thời kỳ kinh tế đang xuống dốc.

GAE là 1 dịch vụ lưu trữ các ứng dụng web. Khi nói đến “ứng dụng web”, chúng ta muốn nói tới các ứng dụng hay dịch vụ được truy cập trên các trang web, thường là thông qua trình duyệt web: những trang web mua bán, mạng xã hội … App Engine cũng có thể phục vụ các trang web truyền thống như xử lý văn bản hay hình ảnh nhưng được thiết kế dành cho thời gian thực.

Thực ra, GAE được thiết kế để lưu trữ những ứng dụng và phục vụ nhiều người dùng một cách đồng thời. Khi một ứng dụng có thể phục vụ nhiều người dùng một cách đồng thời mà không làm giảm hiệu suất, chúng ta gọi đó là sự co giãn (scales). Những ứng dụng được viết cho App Engine sẽ được co giãn một cách tự động. Càng nhiều người sử dụng chương trình, App Engine sẽ tạo ra càng nhiều tài nguyên cho ứng dụng đó và quản lý chúng. Chính bản thân ứng dụng cũng không cần phải biết đến các tài nguyên mà nó đang sử dụng.

Không như những server cung cấp các dịch vụ lưu trữ thông thường hay các server có chức năng tự quản lý, với Google App Engine, chúng ta chỉ phải trả tiền cho những tài nguyên mà chúng ta sử dụng. Những tài nguyên này được đo bằng gigabyte và không có bất kì lệ phí hàng tháng nào hay lệ phí để chúng ta thay đổi diện mạo trang web.Hóa đơn thanh toán những nguồn tài nguyên này bao gồm CPU chúng ta sử dụng, lưu trữ hàng tháng, băng thông vào ra (incoming and outgoing bandwidth), và một số các tài nguyên khác của dịch vụ App Engine. Để giúp chúng ta có thể làm quen với GAE, mỗi lập trình viên sẽ có một lượng tài nguyên miễn phí đủ để chạy những ứng dụng nhỏ với băng thông thấp.

Google App Engine cung cấp hai môi trường thực thi tốt cho các ứng dụng. Đó là Java và Python. Môi trường chúng ta chọn sẽ phụ thuộc vào ngôn ngữ và những công nghệ liên quan khi chúng ta dùng để phát triển ứng dụng.Cả hai môi trường Java và Python đều sử dụng chung một mô hình: một yêu cầu gửi đến ứng dụng trên server, ứng dụng được kích hoạt (nếu cần thiết), gọi bộ phận xử lý yêu cầu và trả về kết quả cho client. Mỗi môi trường sử dụng bộ tiền xử lý (interpreter) cho riêng mình (JVM hay Python).

* Môi trường Java thực thi các ứng dụng được viết cho JVM6. Ứng dụng có thể được phát triển dựa vào ngôn ngữ lập trình Java hoặc hầu hết các ngôn ngữ có thể biên dịch và chạy trên JVM: ví dụ PHP (dùng Quercus), Ruby (dùng JRuby), Javascript (dùng Rhino), Scala, Groovy. App Engine cũng hỗ trợ Google Web Tootkit (GWT).
* Môi trường Python thực thi các ứng dụng được viết dựa vào ngôn ngữ lập trình Python bản 2.5. App Engine gọi các ứng dụng Python nhờ vào CGI. Ứng dụng có thể dùng hầu hết các thư viện của Python, các framework của Python như Django, web2py, Pylons.

**3.1.3 Lựa chọn hình thức thanh toán trực tuyến**

**Khi các website thương mại điện tử ngày càng nâng cao chất lượng lẫn hình thức thể hiện thì một điều không thể thiếu là thanh toán trực tuyến.Với mục tiêu phục vụ khách hàng có nhiều sự chọn lựa phương thức thanh toán trong TTTT, website cung cấp nhiều hình thức thanh toán khác nhau như qua tài khoản NganLuong, thẻ cào hay qua điện thoại di động.**

**3.1.3.1 Thanh toán trên điện thoại di động**

mXu là ứng dụng trên điện thoại di động hỗ trợ Android cho phép người sử dụng đăng ký một tài khoản tại website có thể nạp , gửi tiền vào tải khoản ảo của mình. Và sử dụng điện thoại để thanh toán các giao dịch trực tuyến tại website hay hình thức thanh toán qua tin nhắn SMS.  
**Cách thức hoạt động:**  
  
  
Ghi chú luồng thông tin:

Khách hàng lựa chọn sản phẩm trên website qua ứng dụng client.

User gửi yêu cầu giao dịch từ ĐTDĐ lên hệ thống website thông qua giao diện chương trình hoặc GPRS/Wifi/3G

Server xử lý thông tin tiếp nhận và gửi kết quả về cho khách hàng

Khách hàng nhận được mã Key để thanh toán khi giao hàng.

**Lợi ích mang lại từ hình thức thanh toán này:**  
  
- Đăng ký tài khoản mXU bằng chính số điện thoại di động gán với một tài khoản tại website.  
- Thay đổi đăng ký tài khoản trên mXu khi có nhu cầu thay đổi máy điện thoại, số điện thoại hoặc tài khoản sử dụng mXu, bạn chọn chức năng thay đổi đăng ký và nhập các thông tin số thẻ, số PIN và số điện thoại mới   
  
- Kiểm tratài khoản: Kiểm tra số dư hiện tại của tài khoản của bạn.   
  
- Mua hàng hoá dịch vụ: bạn có thể dễ dàng lựa chọn sản phẩm trên website và thanh toán an toàn nhất qua chương trình mXu.  
  
- Chuyển Xu: Cho phép bạn chuyển khoản cho người khác thông qua số tài khoản hoặc số điện thoại đã đăng ký sử dụng mXu của người đó, sau đó bạn nhập số tiền cần chuyển để thực hiện chuyển khoản.   
  
- Liệt kê Giao dịch: Cho phép bạn kiểm tra thông tin 5 giao dịch gần nhất của bạn. Mỗi giao dịch gồm có các thông tin sau: Ngày giao dịch, số tiền, loại giao dịch.  
- Quản lý tài khoản dễ dàng, tính bảo mật cao: có thể kiểm tra mọi giao dịch của tài khoản bằng chính ứng dụng trên điện thoại di động hoặc qua trang web tại mọi thời điểm với sự bảo mật tuyệt đối cho mọi thông tin.

-Không giới hạn về không gian và thời gian: có thể bán hàng tại bất cứ đâu, bất cứ nơi nào, bất kể lúc nào nếu nơi đó có sóng GPRS hoặc kết nối Internet và tài khoản có đủ số XU cho giao dịch.

**3.1.3.2 Thanh toán qua cổng thanh toán trực tuyến**  
**3.1.3.3 Thanh toán qua thẻ trả trước**

Website phát hành nhiều loại thẻ trả trước khác nhau để khách hàng có thể dễ dàng lựa chọn. Khách hàng chỉ việc điền mã số thẻ để nạp tiền qua giao diện website hay điện thoại di động và nhắn tin SMS theo cú pháp quy định. Tất cả quy trình thực hiện được đảm bảo an toàn nhất và cụ thể như sau:  
  
Bước 1: Khánh hàng tiếp cận đại lý của website, chọn loại thẻ cần mua với nhiều mệnh giá khác nhau.  
  
Bước 2: Khách hàng vào website để đăng ký tài khoản mới hoặc bỗ sung các thông tin tài khoản của mình.  
  
Bước 3: khách hàng nạp tiền qua giao diện website hoặc là nhắn tin SMS theo cú pháp quy định.  
Bước 4: server kiểm tra tính hợp lệ của mã số thẻ cào và gửi thông báo về cho khách hàng.  
**Lợi ích mang lại từ hình thức thanh toán này:**

**- Khách hàng** có thể thanh toán vào bất cứ lúc, không mất thời gian để tìm, đến và chờ đợi thanh toán tại một vài điểm thanh toán của nhà cung cấp dịch vụ.  
- Nhanh chóng, tiện lợi và thoải mái  
  
- Nhà cung cấp dịch vụ không phải tốn chi phí duy trì một hệ thống đại lý đi thu tiền vào cuối tháng.  
  
- Nhà cung cấp không lo những rủi ro về tiền khi bị chiếm dụng vốn, khả năng khó đòi từ người thu tiền, tiền giả, …  
  
- Nhà cung cấp không tốn kém trong việc in ấn hoá đơn.

**3.1.3.4 Thanh toán qua tổng đài di động**

**3.2 Đẩy mạnh việc phát triển TMĐT**

Sau bốn năm triển khai kế hoạch tổng thể phát triển TMĐT giai đoạn 2006-2010,

TMĐT Việt Nam đ. có những bước phát triển mạnh và vững chắc, dần đi vào cuộc sống

và góp phần tích cực vào sự phát triển kinh tế của đất nước.

Để TMĐT tiếp tục phát triển mạnh mẽ hơn nữa, giúp nâng cao năng lực cạnh tranh

của doanh nghiệp và cả nền kinh kế, trong tương lai cần tiếp tục phát huy những thành

quả đ. đạt được, đồng thời khắc phục những khó khăn trở ngại đ. bộc lộ r. hoặc có khả

năng sẽ xảy ra trong tương lai. Để giải quyết những khó khăn trở ngại này cần có sự tham

gia tích cực, chặt chẽ của các cơ quan quản l. nhà nước, doanh nghiệp, các tổ chức x.

hội nghề nghiệp và người tiêu dùng. Cần có một số biện pháp tập trung thực hiện như

sau [1]:

GVHD: Th.S Vũ Trí Dũng SVTH: Trần Hồng Nghi và Tôn Thị Kim Loan

60

**3.2.1 Đối với cơ quan quản l. nhà nước**

**3.2.1.1 Hỗ trợ mạnh mẽ các doanh nghiệp ứng dụng TMĐT**

Kết quả điều tra của Bộ Công Thương trong năm 2009 và các năm gần đây cho thấy,

đến nay hầu hết các doanh nghiệp trong cả nước đ. triển khai ứng dụng TMĐT ở các

mức độ khác nhau. Từ việc chỉ trang bị máy tính có kết nối Internet và sử dụng một số

phần mềm văn ph.ng, đến nay nhiều doanh nghiệp đ. chuyển sang ứng dụng các phần

mềm chuyên dụng, sử dụng các h.nh thức giao dịch, mua bán qua mạng, v.v. . . Việc ứng

dụng TMĐT trong doanh nghiệp đ. thu được những kết quả rất r. ràng.

Với mục tiêu giúp doanh nghiệp có thêm kênh thông tin t.m kiếm khách hàng, thị

trường, từ năm 2006 đến nay, Bộ Công Thương đ. triển khai xây dựng một số cổng

TMĐT và cổng thông tin trên Internet, bao gồm: Cổng TMĐT quốc gia tại địa chỉ

www.ecvn.com, Cổng Thông tin thị trường nước ngoài tại địa chỉ www.ttnn.com.vn và

Cổng Thông tin xuất khẩu www.vnex.com.vn. Trong năm 2010, Bộ Công Thương cần

đẩy mạnh hơn nữa hoạt động của các cổng này, góp phần giúp doanh nghiệp cắt giảm

chi phí và nâng cao hiệu quả trong các hoạt động t.m kiếm thị trường và khách hàng xuất

khẩu.

Việc cung cấp trực tuyến các dịch vụ công cũng góp phần tích cực giúp doanh nghiệp

giảm bớt thời gian và chi phí liên quan tới thủ tục đăng k. kinh doanh, thủ tục xuất nhập

khẩu, v.v. . . Trong các năm vừa qua, việc cung cấp trực tuyến dịch vụ công liên quan

đến thương mại cũng đ. được một số Bộ, ngành và địa phương triển khai khá tốt. Trong

thời gian tới cần đẩy mạnh hơn nữa các hoạt động này.

Đối với việc hỗ trợ cụ thể cho doanh nghiệp, cơ quan quản l. chuyên ngành về TMĐT

cần nghiên cứu xây dựng những chương tr.nh riêng biệt để hỗ trợ cho doanh nghiệp trong

các lĩnh vực sản xuất kinh doanh khác nhau, triển khai thí điểm và nhân rộng mô h.nh

trao đổi dữ liệu kinh doanh điện tử giữa các doanh nghiệp.

**3.2.1.2 Tiếp tục đẩy mạnh tuyên truyền phổ biến về TMĐT**

Là một nhiệm vụ quan trọng của Kế hoạch tổng thể phát triển TMĐT giai đoạn 2006-

2010, trong thời gian qua, hoạt động tuyên truyền, phổ biến về TMĐT đ. được các cơ

quan quản l. nhà nước, cơ quan thông tin đại chúng tổ chức triển khai mạnh mẽ. Với

thực tế phát triển TMĐT như hiện nay, có thể nói hoạt động tuyên truyền phổ biến về

GVHD: Th.S Vũ Trí Dũng SVTH: Trần Hồng Nghi và Tôn Thị Kim Loan

61

TMĐT đ. đạt được mục tiêu đề ra.

Tuy nhiên, TMĐT là một lĩnh vực mới và phát triển nhanh chóng, nên trong thời gian

tới cần tiếp tục đẩy mạnh hoạt động tuyên truyền phổ biến về lĩnh vực này. Việc tuyên

truyền phổ biến cần đi sâu vào các nội dung cụ thể như: giới thiệu các mô h.nh ứng dụng

TMĐT hiệu quả, bảo đảm an toàn an ninh trong giao dịch TMĐT, bảo vệ thông tin cá

nhân trong TMĐT, lợi ích của việc mua sắm trên mạng và thanh toán điện tử.

**3.2.1.3 Tăng cường thực thi pháp luật về TMĐT**

Sau khi Quốc hội thông qua Luật Giao dịch điện tử vào năm 2005 và Luật Công nghệ

thông tin năm 2006, từ năm 2006-2008 Chính phủ đ. ban hành bảy Nghị định hướng dẫn

hai Luật này, h.nh thành khung pháp l. cơ bản cho các hoạt động trong lĩnh vực TMĐT.

Trong thời gian vừa qua, các Bộ, ngành hữu quan cũng đ. ban hành nhiều văn bản hướng

dẫn thực hiện các Nghị định trong hệ thống Luật Giao dịch điện tử và Luật Công nghệ

thông tin.

Để nội dung Luật và các văn bản dưới luật thực sự đi vào cuộc sống, trong thời gian

tới các cơ quan quản l. nhà nước cần đẩy mạnh việc triển khai thực hiện các quy định tại

văn bản pháp luật về TMĐT đ. được ban hành, bảo đảm việc thực hiện nghiêm túc của

các cơ quan quản l. nhà nước, doanh nghiệp và mỗi cá nhân trong x. hội. Trong triển

khai cần chú trọng tới hoạt động hướng dẫn, phổ biến nội dung của văn bản pháp luật để

doanh nghiệp, nhân dân hiểu và thực hiện đúng các quy định đ. ban hành.

Để thực thi tốt pháp luật về TMĐT, các cơ quan quản l. nhà nước hữu quan cần tăng

cường bộ máy tổ chức, nâng cao năng lực cán bộ và triển khai mạnh mẽ các hoạt động

thực thi pháp luật. Chủ động phát hiện các hành vi vi phạm pháp luật và tiến hành xử l.

nghiêm minh.

Tăng cường các biện pháp bảo vệ người tiêu dùng trong TMĐT nhằm thúc đẩy sự

tham gia của nhiều tầng lớp nhân dân vào các h.nh thức giao dịch mua bán trên mạng.

**3.2.2 Đối với các doanh nghiệp**

**3.2.2.1 Đẩy mạnh hơn nữa ứng dụng TMĐT nhằm nâng cao năng lực cạnh tranh**

Thực tiễn phát triển trong những năm vừa qua cho thấy kết quả của việc ứng dụng

TMĐT đối với hoạt động sản xuất kinh doanh của các doanh nghiệp là rất r. ràng. Nhiều

GVHD: Th.S Vũ Trí Dũng SVTH: Trần Hồng Nghi và Tôn Thị Kim Loan

62

doanh nghiệp cũng đ. mạnh dạn đầu tư ứng dụng các phần mềm quản l. chuyên dụng,

bố trí cán bộ chuyên trách về TMĐT, sử dụng dịch vụ công do các cơ quan quản l. nhà

nước cung cấp. . . để nâng cao hiệu quả kinh doanh. Tuy nhiên, tỷ lệ các doanh nghiệp

ứng dụng sâu TMĐT vẫn c.n ở mức khiêm tốn.

Trong thời gian tới, các doanh nghiệp cần đẩy mạnh đầu tư ứng dụng TMĐT để tối

ưu hóa quy tr.nh sản xuất, kinh doanh, đặc biệt là cần đầu tư cho các phần mềm chuyên

dụng như quản l. tài nguyên doanh nghiệp, quản trị chuỗi cung ứng, quản l. khách hàng,

v.v. . .

Tham gia sàn giao dịch TMĐT là một h.nh thức xúc tiến thương mại, t.m kiếm thị

trường có hiệu quả với chi phí rất thấp. Tuy nhiên, v. nhiều l. do khác nhau, việc tham

gia sàn TMĐT của các doanh nghiệp có xu hướng chững lại trong hai năm 2008-2009.

Nhằm tận dụng ưu điểm của phương thức này trong việc t.m kiếm khách hàng, mở rộng

thị trường, doanh nghiệp cần tích cực hơn nữa trong việc tham gia vào các sàn giao dịch

TMĐT có uy tín trong nước cũng như trên thế giới.

Để ứng dụng TMĐT có kết quả tốt, các doanh nghiệp cần chú trọng tới chiến lược

kinh doanh của m.nh để xây dựng chiến lược ứng dụng TMĐT phù hợp. Đồng thời, chiến

lược ứng dụng TMĐT của doanh nghiệp cũng phải phù hợp với tr.nh độ phát triển về hạ

tầng công nghệ thông tin và TMĐT tại địa phương.

**3.2.2.2 Nâng cao nhận thức về TMĐT và .thức tuân thủ pháp luật**

Trong thời gian vừa qua, các cơ quan quản l. nhà nước đ. ban hành nhiều văn bản

quy phạm pháp luật về TMĐT. Đến nay, hệ thống pháp luật liên quan tới TMĐT đ.

tương đối hoàn thiện. Bên cạnh đó, các doanh nghiệp cũng đ. nhận thức được tầm quan

trọng và tổ chức ứng dụng TMĐT ở nhiều mức độ khác nhau.

Tuy nhiên, sự quan tâm của doanh nghiệp đối với các quy định pháp luật về TMĐT

c.n ở mức độ thấp. Trong thời gian tới các doanh nghiệp cần tích cực t.m hiểu, nghiên

cứu các văn bản liên quan đến TMĐT như Luật Giao dịch điện tử, Luật Công nghệ thông

tin, các nghị định quy định chi tiết thi hai Luật này và hệ thống văn bản hướng dẫn liên

quan đến chứng từ điện tử, giao kết và thực hiện hợp đồng trên website TMĐT, thư rác,

tên miền, xử phạt hành chính, v.v... Trong quá tr.nh hoạt động, doanh nghiệp nên t.m

hiểu, phát hiện những khó khăn, bất cập trong việc thực hiện các quy định pháp luật về

TMĐT, chủ động đề xuất bổ sung, sửa đổi. Doanh nghiệp có thể trực tiếp, hoặc thông

GVHD: Th.S Vũ Trí Dũng SVTH: Trần Hồng Nghi và Tôn Thị Kim Loan

63

qua các tổ chức x. hội nghề nghiệp như Hiệp hội TMĐT, Ph.ng Thương mại và Công

nghiệp Việt Nam, để phản ánh, đóng góp . kiến của m.nh đến các cơ quan quản l. nhà

nước.

**3.2.3 Đối với người tiêu dùng**

Từ năm 2007 đến nay, hoạt động mua bán trực tuyến đ. có bước phát triển khá mạnh

mẽ tại các thành phố lớn. Việc bán hàng hóa và dịch vụ qua mạng Internet đ. được nhiều

doanh nghiệp uy tín thuộc nhiều lĩnh vực khác nhau triển khai và đạt kết quả khả quan,

góp phần nâng cao sức cạnh tranh của doanh nghiệp, đồng thời mang lại sự tiện lợi, thoải

mái cho người tiêu dùng.

Để giúp hoạt động mua bán trực tuyến phát triển mạnh hơn nữa, góp phần h.nh thành

môi trường mua sắm hiện đại, người tiêu dùng cần mạnh dạn thực hiện việc mua sắm

trên mạng, có thể khởi đầu bằng việc thử nghiệm mua hàng hóa tại một website TMĐT

có uy tín. Bên cạnh đó, những cá nhân đ. có kinh nghiệm mua sắm trực tuyến cần tích

cực tuyên truyền, cổ động, vận động người thân, bạn bè tham gia h.nh thức mua bán tiện

lợi và an toàn này.

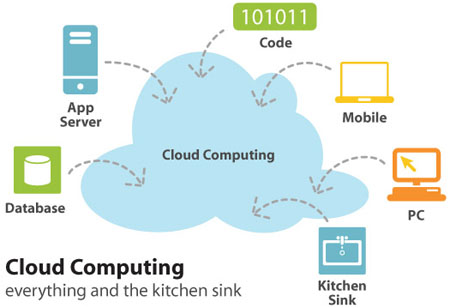
Ngoài việc tích cực tham gia mua sắm trực tuyến, để h.nh thành môi trường TMĐT

an toàn, người tiêu dùng cũng cần trang bị cho m.nh những kiến thức cơ bản đối với việc

sử dụng mạng Internet, tránh bị lừa đảo làm mất thông tin cá nhân, phát tán virus...

**3.3 Công nghệ**

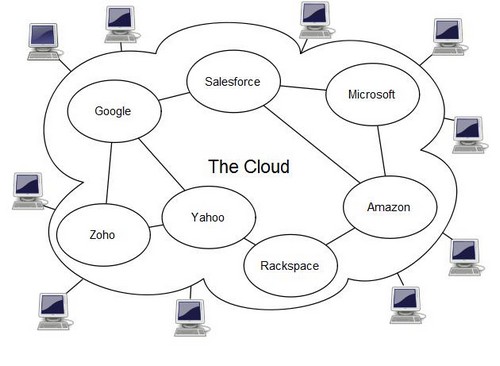
**3.3.1 Điện toán đám mây**  
 Đám mây (cloud) là biểu tượng tượng trưng cho Internet và thường được sử dụng trong các mô hình, sơ đồ mạng máy tính. Do đó, điện toán đám mây được hiểu như cách thức áp dụng các kỹ thuật điện toán dựa trên Internet. Cụ thể hơn, đó là một mô hình mà mọi thông tin đều được lưu trữ, tính toán, xử lý trong các máy chủ đặt trên Internet. Người sử dụng có thể làm việc với các thông tin đó mà không cần phải hiểu biết về công nghệ, kỹ thuật và hạ tầng cơ sở của đám mây.



**Hình 1: Mọi thứ đều tập trung vào đám mây**

Với cách thức lưu trữ và xử lý thông tin như vậy, người sử dụng có thể dễ dàng truy cập vào đám mây chỉ với một ứng dụng có khả năng truy nhập Internet và từ bất kỳ thiết bị nào, bao gồm máy tính, thiết bị cầm tay, di động, thiết bị giải trí... Ví dụ rõ ràng nhất về ứng dụng thực tế đang được sử dụng rộng rãi, đại diện cho một ứng dụng trong mô hình điện toán đám mây, là dịch vụ thư điện tử trực tuyến được cung cấp bởi nhiều hãng lớn như Hotmail của Microsoft, Yahoo Mail của Yahoo, Gmail của Google... Với dịch vụ này, người sử dụng chỉ cần trình duyệt web cùng tài khoản cá nhân đã được đăng ký là có thể thực hiện trao đổi, giao dịch thư điện tử mà không cần quan tâm đến vấn đề kỹ thuật, phần mềm, hạ tầng do điềuđó được đảm bảo bởi các nhà cung cấp dịch vụ, mà cụ thể ở đây là Microsoft, Yahoo, Google...

**Sự khác biệt giữa điện toán đám mây và điện toán truyền thống**  
 Trong mô hình điện toán truyền thống, các cá nhân, doanh nghiệp sẽ xây dựng riêng cơ sở hạ tầng kỹ thuật để tự cung cấp các dịch vụ cho các hoạt động thông tin đặc thù của mình. Với mô hình này, mọi thông tin sẽ được lưu trữ, xử lý nội bộ và họ sẽ trả tiền để triển khai, duy trì cơ sở hạ tầng đó (mua thiết bị phần cứng, phần mềm chuyên dụng, trả lương cho bộ phận điều hành...).



**Hình 2 Mô hình tổng quan điện toán đám mây**

Khác với mô hình điện toán truyền thống, điện toán đám mây lưu trữ và xử lý toàn bộ thông tin trong đám mây Internet. Mọi công nghệ, kỹ thuật, cơ sở hạ tầng cũng như chi phí triển khai trong đám mây sẽ do nhà cung cấp đảm bảo xây dựng và duy trì. Do đó, thay vì phải đầu tư từ đầu rất nhiều tiền cho chi phí xây dựng cơ sở hạ tầng riêng, các cá nhân, doanh nghiệp trong quá trình hoạt động sẽ chỉ phải trả số tiền vừa đủ theo nhu cầu sử dụng của mình (pay-for-what-you-use). Như vậy, mô hình này có rất nhiều lợi ích như sử dụng hợp lý nguồn vốn, điều hòa chi tiêu tính toán theo thực tế sử dụng, luôn hưởng năng suất tính toán theo cam kết của nhà cung cấp dịch vụ, tận dụng được sức mạnh của Internet và các siêu máy tính, giảm cơ bản trách nhiệm quản lý hệ thống CNTT nội bộ.

**Cấu trúc phân lớp của mô hình Điện toán đám mây**

Về cơ bản, “điện toán đám mây” được chia ra thành 3 lớp riêng biệt, có tác động qua lại lẫn nhau:



Cấu trúc phân lớp mô hình Điện toán đám mây

Các tầng tạo nên một đám mây bao gồm:

* Các dịch vụ ứng dụng:

Tầng dịch vụ ứng dụng này lưu trữ các ứng dụng phù hợp với mô hình SaaS. Đây là những ứng dụng chạy trong một đám mây và được cung cấp theo yêu cầu về các dịch vụ cho người dùng. Đôi khi các dịch vụ này được cung cấp miễn phí và các nhà cung cấp ứng dụng tạo ra doanh thu trực tiếp từ việc quảng cáo Web hay từ việc sử dụng dịch vụ. Thành phần này được hiện thực bằng qua các dịch vụ rất quen thuộc hàng ngày như Gmail, Yahoo mail…

Ngày nay, các dịch vụ ứng dụng này được chuyển trực tiếp đến cộng đồng doanh nghiệp.Ở đó có lưu trữ các yêu cầu phần mềm có sẵn để xử lý bảng lương, quản lý nguồn nhân lực, cộng tác, quản lý khách hàng, quản lý mối quan hệ đối tác kinh doanh và nhiều hơn nữa. Chúng làm lợi cho người tiêu dùng bằng cách giải phóng cho họ khỏi việc cài đặt và bảo trì phần mềm và các dịch vụ được sử dụng theo nhu cầu.

* Các dịch vụ nền tảng:

Đây là tầng ở đó chúng ta thấy cơ sở hạ tầng ứng dụng nổi lên như là một tập hợp các dịch vụ.Dịch vụ này chạy mà không bị hạn chế và nó dành để hỗ trợ cho các ứng dụng.Để đạt được khả năng mở rộng khi cần thiết, chúng thường được ảo hóa chạy trên trong đám mây hay là trong một trung tâm dữ liệu doanh nghiệp truyền thống.

* Các dịch vụ cơ sở hạ tầng

Đây là tẩng tập hợp các thiết bị vật lý như các máy chủ, các thiết bị mạng và các đĩa lưu trữ được đưa ra như là các dịch vụ được cung cấp cho người tiêu dùng.Các dịch vụ này hỗ trợ cơ sở hạ tầng ứng dụng – bất kể cơ sở hạ tầng đó đang được cung cấp qua một đám mây hay không – và nhiều người dùng hay không. Cũng như với cơ sở các dịch vụ nển tảng, sự ảo hóa là một phương pháp thường được sử dụng để tạo ra chế độ phân phối các nguồn tài nguyên theo yêu cầu.

**Hiện thực của Cloud Computing**

**Infrastructure-as-a-Service (IaaS – Dịch vụ hạ tầng**): Cung cấp cho người dùng hạ tầng thô (thường là dưới hình thức các máy ảo) như là một dịch vụ.Những kiến trúc ảo xếp chồng là một ví dụ của xu hướng mọi thứ là dịch vụ và có cùng những điểm chung. Hơn hẳn một máy chủ cho thuê, không gian luu trử tập trung hay thiết bị mạng, máy trạm thay vì đầu tư mua những nguyên thì có thể thuê đầy đủ dịch vụ bên ngoài.Những dịch vụ này thông thường được tính chi phí trên cơ sở tính toán chức năng và lượng tài nguyên sử dụng (và từ đó ra chi phí) sẽ phản ảnh được mức độ của hoạt động.Đầy lầ một sự phát triển của những giải pháp lưu trữ web và máy chủ cá nhân ảo.

**Platform-as-a-Service (PaaS – Dịch vụ nền tảng):** Cung cấp API cho phát triển ứng dụng trên mộtt nền tảng trừu tượng.Cung cấp nền tảng tinh toán và một tập các giải pháp nhiều lớp. Nó hỗ trợ việc triển khai ứng dụng mà không quan tâm đến chi phí hay sự phức tạp của việc trang bị và quản lý các lớp phần cứng và phần mềm bên dưới, cung cấp tất cả các tính năng cần thiết để hỗ trợ chu trình sống đầy đủ của việc xây dựng và cung cấp một ứng dụng và dịch vụ web sẵn sàng trên Internet mà không cần bất kì thao tác tải hay cài đặt phần mềm cho những người phát triển, quản lý tin học, hay người dùng cuối. Nó còn được biết đến với một tên khác là cloudware.

Cung cấp dịch vụ nền tảng (PaaS) bao gồm những điều kiện cho qui trình thiết kế ứng dụng, phát triển, kiểm thử, triển khai và lưu trữ ứng dụng có giá trị như là dịch vụ ứng dụng như cộng tác nhón, săp xếp và tích hợp dịch vụ web, tích hợp cơ sở dữ liệu, bảo mật, khả năng mở rộng, quản lý trạng thái, phiên bản ứng dụng, các lợi ích cho cộng đồng phát triển và nghiên cứu ứng dụng. Những dịch vụ này được chuẩn bị như là một giải pháp tính hợp trên nền web.

**Software-as-a-Service (SaaS – Dịch vụ phần mềm):** Cung cấp dịch vụ phần mềm thực thi từ xa. Dịch vụ phần mềm (SaaS) là một mô hình triển khai ứng dụng mà ở đó người cung cấp cho phép người dụng sử dụng dịch vụ theo yêu cầu. Những nhà cung cấp SaaS có thể lưu trữ ứng dụng trên máy chủ của họ hoặc tải ứng dụng xuống thiết bị khách hàng, vô hiệu hóa nó sau khi kết thúc thời hạn. Các chức năng theo yêu cầu có thể được kiểm soát bên trong để chia sẻ bản quyền của một nhà cung cấp ứng dụng thứ ba.

**3.3.2 Google App Engine (GAE)**



Google App Engine

App Engine của Google là một nền tảng thực sự để xây dựng và triển khai ứng dụng Web Java (và Python) trên cơ sở hạ tầng mở rộng của Google. Nó không yêu cầu lệ phí cấp phép sử dụng (tất nhiên, trừ ra một số thư viện phần mềm mà bạn chọn để sử dụng trên cơ sở hạ tầng ấy yêu cầu giấy phép) và không cần chi phí trả trước cho băng thông hoặc không gian lưu trữ. Cơ sở hạ tầng của App Engine là hoàn toàn miễn phí cho đến khi bạn đạt đến ngưỡng về mức sử dụng — dung lượng lưu trữ 500MB và, theo như trích dẫn nguyên văn từ Google, "đủ CPU và băng thông cho khoảng 5 triệu lượt xem trang mỗi tháng".Không như những server cung cấp các dịch vụ lưu trữ thông thường hay các server có chức năng tự quản lý, với Google App Engine, chúng ta chỉ phải trả tiền cho những tài nguyên mà chúng ta sử dụng. Những tài nguyên này được đo bằng gigabyte và không có bất kì lệ phí hàng tháng nào hay lệ phí để chúng ta thay đổi diện mạo trang web.Hóa đơn thanh toán những nguồn tài nguyên này bao gồm CPU chúng ta sử dụng, lưu trữ hàng tháng, băng thông vào ra (incoming and outgoing bandwidth), và một số các tài nguyên khác của dịch vụ App Engine. Để giúp chúng ta có thể làm quen với GAE, mỗi lập trình viên sẽ có một lượng tài nguyên miễn phí đủ để chạy những ứng dụng nhỏ với băng thông thấp.

**3.3.2.1 Thành phần Google App Engine:**

### MÔI TRƯỜNG THỰC THI (THE RUNTIME ENVIRONMENT)

Một ứng dụng App Engine đáp ứng các yêu cầu web. Một yêu cầu web sẽ bắt đầu khi có một người dùng hay điển hình là các trình duyệt web của người dùng gửi một yêu cầu truy cập vào ứng dụng thông qua giao thức HTTP. Khi App Engine nhận được yêu cầu, nó sẽ xác định ứng dụng dựa vào tên miền, hoặc tên miền con của .appspot.com (cung cấp miễn phí mỗi ứng dụng) hoặc là một tên miền riêng của chúng ta đã được đăng kí và thiết lập với Google Apps. App Engine lựa chọn một máy chủ từ nhiều máy chủ để xử lý các yêu cầu đó. Sau đó, App Engine sẽ gửi các yêu cầu đã nhận được từ người dùng đến ứng dụng phù hợp để xử lý, sau khi đã xử lý xong các ứng dụng này sẽ gửi dữ liệu trả về cho App Engine, App Engine sẽ nhận dữ liệu phản hồi từ các ứng dụng và trả về cho người dùng thông qua trình duyệt web.

Theo góc nhìn của ứng dụng, môi trường thực thi chỉ xuất hiện và tồn tại khi bắt đầu một yêu cầu và sẽ biến mất khi yêu cầu đó được đáp ứng xong. App Engine cung cấp tối thiểu 2 cách thức lưu trữ dữ liệu tồn tại giữa các yêu cầu , nhưng các cơ chế này tồn tại bên ngoài môi trường thực thi. Môi trường thực thi sẽ không duy trì trạng thái giữa các yêu cầu, hoặc ít nhất không mong muốn các trạng thái sẽ được duy trì giữa các yêu cầu. App Engine có thể phân phát lưu lượng truy cập trong nhiều server, vì nó cần phải đáp ứng cho nhiều yêu cầu xử lý như nhau, bất kể có bao nhiêu lưu lượng truy cập nó sẽ xử lý cùng một lúc.

Các đoạn code ứng dụng không thể truy cập vào máy chủ mà nó đang chạy trong các kiến trúc truyền thống. Một ứng dụng có thể đọc các tập tin từ hệ thống tập tin của nó, nhưng nó không thể ghi vào các tập tin này, cũng như không thể đọc các tập tin thuộc các ứng dụng khác. Một ứng dụng có thể nhìn thấy các biến môi trường được thiết lập bởi App Engine, nhưng những thao tác của các biến này không nhất thiết phải tồn tại giữa các yêu cầu. Một ứng dụng không thể truy cập vào cơ sở mạng của phần cứng máy chủ, mặc dù nó có thể thực hiện các thao tác sử dụng các dịch vụ mạng.

Bản thân GAE có cơ chế để quản lý các trạng thái của từng yêu cầu trong mỗi ứng dụng dưới dạng hộp đen của nó (người phát triển không biết rõ cơ chế bên trong nhưng hỗ trợ người phát triển những dịch vụ cần thiết). Điều này cho phép App Engine xử lý một yêu cầu với một máy chủ mà nó mong muốn trong ước tính của nó để trả về phản hồi nhanh nhất. Không có cách nào để đảm bảo rằng phần cứng trên cùng một máy chủ sẽ xử lý hai lần yêu cầu, ngay cả khi các yêu cầu đến cùng từ một client, và đến khá nhanh chóng.

Hộp đen này cho phép App Engine chạy nhiều ứng dụng trên cùng một máy chủ, trong đó hành vi của một ứng dụng không làm ảnh hưởng đến các ứng dụng khác. Ngoài ra để giới hạn quyền truy cập đến hệ điều hành, môi trường thực thi cũng giới hạn việc sử dụng CPU và bộ nhớ . App Engine giữ các giới hạn này linh hoạt và chặt chẽ hơn các giới hạn này để các ứng dụng sử dụng nhiều tài nguyên hơn để bảo vệ tài nguyên được chia sẻ từ những ứng dụng “không mong muốn”.

Mỗi yêu cầu có tối đa 30s để trả về phản hồi cho client. Mặc dù thời gian này có vẻ đáp ứng tốt cho một ứng dụng web, nhưng App Engine được tối ưu hóa cho các ứng dụng đáp ứng chưa đầy một giây. Ngoài ra nếu một ứng dụng sử dụng nhiều CPU, App Engine có thể làm chậm nó xuống, nên các ứng dụng không trì hoãn bộ vi xử lý trên một máy phục vụ đa ứng dụng. Một CPU tập trung xử lý yêu cầu có thể mất nhiều thời gian để hoàn thành, khi App Engine dò tìm các mô hình theo cách sử dụng CPU và phân bổ cho phù hợp.

Google App Engine cung cấp hai môi trường thực thi tốt cho các ứng dụng. Đó là Java và Python. Môi trường chúng ta chọn sẽ phụ thuộc vào ngôn ngữ và những công nghệ liên quan khi chúng ta dùng để phát triển ứng dụng.

Môi trường Java thực thi các ứng dụng được viết cho JVM6. Ứng dụng có thể được phát triển dựa vào ngôn ngữ lập trình Java hoặc hầu hết các ngôn ngữ có thể biên dịch và chạy trên JVM: ví dụ PHP (dùng Quercus), Ruby (dùng JRuby), Javascript (dùng Rhino), Scala, Groovy. App Engine cũng hỗ trợ Google Web Tootkit (GWT).

Môi trường Python thực thi các ứng dụng được viết dựa vào ngôn ngữ lập trình Python bản 2.5. App Engine gọi các ứng dụng Python nhờ vào CGI. Ứng dụng có thể dùng hầu hết các thư viện của Python, các framework của Python như Django, web2py, Pylons.

Cả hai môi trường Java và Python đều sử dụng chung một mô hình: một yêu cầu gửi đến ứng dụng trên server, ứng dụng được kích hoạt (nếu cần thiết), gọi bộ phận xử lý yêu cầu và trả về kết quả cho client. Mỗi môi trường sử dụng bộ tiền xử lý (interpreter) cho riêng mình (JVM hay Python).

### CÁC FILE SERVER TĨNH (STATIC FILE SERVERS)

Hầu hết các website có một số tài nguyên mà chúng chuyển đến các trình duyệt không thay đổi trong suốt hoạt động của site. Ví dụ như các hình ảnh và các file Css hỗ trợ hiển thị của site, các đoạn mã Javascript chạy trên trình duyện, và các file HTML tĩnh. Vì việc gửi những file này không cần đến việc gọi code của ứng dụng, nói cách khác, việc này không cần thiết, làm giảm hiện quả làm việc của các server ứng dụng.

Thay vì thế, App Engine cung cấp một tập các server tách biệt chuyên làm nhiệm vụ trao đổi các file tĩnh này. Những server nói trên tối ưu hóa cho kiến trúc bên trong và hạ tầng mạng để xử lý các requests cho các tài nguyên tĩnh. Đối với client, các file tĩnh này cũng giống như các tài nguyên khác được cung cấp bởi ứng dụng của chúng ta.

Chúng ta upload các file tĩnh cùng với code ứng dụng. Chúng ta có thể cấu hình vài cách hiển thị nhất định cho các file này, bao gồm các URL cho file tĩnh, loại nội dung, và các hướng dẫn cho trình duyệt để lưu các bản sao file này trong bộ nhớ cache để giảm lưu lượng và tăng tốc cho trang web.

### KHO DỮ LIỆU (DATASTORE)

Hầu hết mọi ứng dụng web đều cần một nơi để chứa thông tin khi xử lý yêu cầu từ phía client và lấy thông tin cho những lần yêu cầu sau này. Việc sử dụng một server làm một trung tâm dữ liệu là một ý tưởng rất hay nhưng có vấn đề xảy ra khi quá nhiều kết nối được gửi tới server này.

Hệ thống database của Google App Engine gần giống như một đối tượng database. Thiết kế của App Engine datastore chỉ là trừu tượng để cho App Engine có thể xử lý việc phân tán và co giãn các ứng dụng. Chính vì thế trong các đoạn mã của chúng ta sẽ quan tâm đến những thứ khác được nêu sau đây.

#### THỰC THỂ (ENTITIES) VÀ THUỘC TÍNH (PROPERTIES)

Một ứng dụng App Engine có thể chứa dữ liệu trong một hay nhiều thực thể datastore. Mỗi thực thể có một hay nhiều thuộc tính, mỗi thuộc tính đều có tên và giá trị. Mỗi thực thể đều được phân loại để tiện cho việc truy vấn.

Ở cái nhìn đầu tiên, chúng ta thấy thực thể có thể gần giống các hàng (row) trong một bảng trong cơ sở dữ liệu quan hệ. Và mỗi thuộc tính có thể giống với một cột. Tuy nhiên đối với các thực thể cùng loại với nhau có thể có các thuộc tính khác nhau. Thứ 2, các thực thể có thể có cùng thuộc tính với các thực thể cùng loại khác nhưng khác về kiểu dữ liệu. Một điểm khác nhau nữa giữa thực thể và các hàng (row) là các thực thể có thể có nhiều giá trị cho một thuộc tính đơn lẻ.

Mỗi thực thể có một khóa riêng (key) phân biệt lẫn nhau được cung cấp bởi ứng dụng hoặc do App Engine. Khác với CSDL quan hệ, khoá của thực thể không phải là thuộc tính, nó tồn tại độc lập với thực thể. Khoá của thực thể không được thay đổi khi thực thể đã được tạo ra.

#### TRUY VẤN (QUERIES) VÀ CHỈ MỤC (INDEXES)

Một câu truy vấn trên datastore trả về không hoặc nhiều thực thể cùng loại với nhau. Nó cũng có thể trả về các khóa của thực thể. Câu truy vấn có thể dựa vào các giá trị thuộc tính của thực thể và được sắp xếp theo giá trị của thuộc tính. Câu truy vấn cũng có thể làm việc với các khóa của thực thể.

Với App Engine, mỗi câu truy vấn sẽ có một chỉ mục trong datastore. Khi ứng dụng cần thực hiên một câu truy vấn, thì datastore sẽ tìm chỉ mục của câu truy vấn đó. Khi chúng ta tạo mới một thực thể và cập nhật cái cũ thì datastore cũng sẽ cập nhật lại chỉ mục. Điều này giúp cho câu truy vấn được nhanh hơn.

#### PHIÊN GIAO DỊCH (TRANSACTION)

Khi một ứng dụng có quá nhiều client liên tục đọc hay ghi cùng một dữ liệu ở cùng một thời điểm, thì phiên giao dịch rất cần thiết để dữ liệu không bị đọc sai. Mỗi phiên giao dịchlà đơn vị nhỏ nhất và chỉ có hai trạng thái là thành công hoặc thất bại.

Một ứng dụng đọc hay cập nhật nhiều thực thể trong một phiên giao dịch, nhưng nó phải nói cho App Engine biết những thực thể nào sẽ được cập nhật khi nó tạo ra nhiều thực thể. Ứng dụng làm được điều này bởi việc tạo ra nhóm thực thể. Nhờ nhóm thực thể, App Engine sẽ biết được các thực thể sẽ phân tán như thế nào qua các server, vì thế nó có thể khẳng định chắc chắn là phiên giao dịch thành công hay thất bại. App Engine cũng hỗ trợ những phiên giao dịch nội bộ (local transaction).

Nếu một người dùng cố gắng cập nhật một thực thể trong khi người khác đang cập nhật thực thể đó thì datastore sẽ ngay lập tức trả về một biệt lệ báo lỗi. Trong thuật ngữ database, thì ta nói App Engine sử dụng “optimistic concurrency control”. Với chỉ mục và “optimistic concurrency control”, App Engine được thiết kế giúp cho ứng dụng có thể đọc dữ liệu nhanh hơn và đảm bảo hơn.

#### CÁC DỊCH VỤ (THE SERVICES)

Dịch vụ chính là mối quan hệ giữa datastore với môi trường thực thi. GAE bao gồm một số các dịch vụ hữu ích cho các ứng dụng web.

Dịch vụ memcache là dịch vụ lưu trữ theo khóa - giá trị. Thuận lợi chính của dịch vụ này trên datastore là tốc độ nhanh, rất nhanh so với việc lưu trữ và lấy dữ liệu một cách bình thường trên datastore. Memcache lưu trữ dữ liệu trên bộ nhớ thay vì trên ổ đĩa để tăng tốc độ truy cập. Nó cũng phân tán như datastore tuy nhiên không lưu trữ, vì thế nếu mất điện thì dữ liệu trên memcache cũng mất. Và nó cũng có nhiều giới hạn sử dụng hơn datastore. Việc sử dụng memcache tốt nhất là lưu lại các kết quả của các câu query hay các tính toán trước đó. Ứng dụng sẽ kiểm tra trong memcache, nếu không có dữ liệu thì sẽ tiến hành query trên datastore.

Ứng dụng App Engine có thể truy cập các tài nguyên web khác nhờ vào dịch vụ URL Fetch. Dịch vụ này tạo ra các yêu cầu theo dạng HTTP gửi đến các server khác trên Internet như là việc tương tác với các web service khác. Vì khi ta truy cập đến server khác thời gian sẽ lâu nên URL Fetch hỗ trợ chạy ngầm bên dưới trong khi xử lý các yêu cầu khác.

Ứng dụng App Engine có thể gửi mail dựa vào dịch vụ Mail, hoặc nếu được cấu hình, nó có thể nhận được email.

Ứng dụng App Engine có thể gửi và nhận những tin nhắn đến các dịch vụ chat có sử dụng giao thức XMPP bao gồm Google Talk.

#### TÀI KHOẢN GOOGLE (GOOGLE ACCOUNTS)

Các chức năng của App Engine tích hợp trong các tài khoản của Google như Google Mail, Google Docs và Google Calendar. Chúng ta có thể sử dụng tài khoản trên Google như cho các ứng dụng của chúng ta, do đó không cần thiết lập thêm. Và nếu các người dùng của chúng ta đã có tài khoản trên Google, họ có thể đăng nhập vào ứng dụng của chúng ta với các tài khoản đó, mà không cần tạo tài khoản riêng trong ứng dụng. Tất nhiên, điều đó không bắt buộc. Chúng ta luôn có thể xây dựng hệ thống tài khoản riêng của mình, hoặc sử dụng một provider OpenID.

#### HÀNG ĐỢI CÔNG VIỆC VÀ LẬP LỊCH (TASK QUEUES VÀ CRON JOBS)

Một ứng dụng web phải đáp ứng các request trên web một cách nhanh chóng, thông thường không quá 1 giây và tốt nhất là khoảng vài chục milli giây, để giúp cho người dùng chạy hệ thống mượt hơn. Điều đó buộc hệ thống phải xử lý số lượng công việc lớn trong thời gian sớm nhất. Và người dùng cũng cần được đảm bảo là yêu cầu sẽ được đáp ứng.

Để làm được việc này, App Engine sử dụng task queues. Task queues cho phép các bộ xử lý request xác định công việc sẽ làm trong thời gian tiếp theo. Queues bảo đảm các task đều sẽ được hoàn tất. Nếu như một task lỗi, queue sẽ cố gắng làm cho đến khi hoàn tất thành công. Chúng ta có thể cấu hình tốc độ mà các queues được xử lý để dàn trải khối lượng công việc trong 1 ngày.

Một queue thể hiện 1 task bằng cách gọi một bộ xử lý request. Mã nguồn chạy, tải dữ liệu, tạo ra task, chuyển đến bộ xử lý task như một request HTTP.

App Engine có một service khác cho việc thực hiện các tasks vào các thời điểm được xác định trong một ngày. Các task được lên lịch còn gọi là “cron jobs”, tên này được mượn từ một chức năng tương tự trong hệ đều hành Unix. Dịch vụ lập lịch cho tasks có thể gọi bộ xử lý request vào một thời gian nhất định trong ngày, trong tuần, hoặc trong tháng, dựa trên thời gian biểu mà chúng ta lập ra khi upload ứng dụng. Các tasks như thế hữu dụng trong việc bảo trì hoặc gửi các thông báo định kì.

#### CÁC TOOL LẬP TRÌNH

Google cung cấp một số tool miễn phí cho việc phá triển ứng dụng App Engine bằng ngôn ngữ Java và Python. Chúng ta có thể download software development kit (SDK) cho ngôn ngữ tương ứng và hệ điều hành của chúng ta từ website Google. Người dùng Java có thể downSDK cho Java trong form plug-in cho Eclipse. Người dùng Python sử dụng Windows hoặc Mac OS X có thể tải SDK cho Python từ form của giao diện ứng dụng. Các SDK đó đều có file zip, sử dụng command-lines để trực tiếp tích hợp vào môi trường phát triển hoặc xây dựng hệ thống.

Mỗi SDK bao gồm một server web chạy ứng dụng của chúng ta tại máy cá nhân, đóng vai trò môi trường thực thi,kho dữ liệu và các service. Server này tự động phát hiện những thay đổi trong file source và load lại chúng nếu cần, vì thế chúng ta có thể luôn mở server trong khi đang lập trình cho ứng dụng.

Nếu chúng ta sử dụng Eclipse, chúng ta có thể chạy server cho Java trong bộ tích hợp debugger, và có thể đặt các breakpoint trong code. Chúng ta cũng có thể sử dụng Eclipse cho ứng dụng phát triển Python dùng PyDev, một nhánh mở rộng của Eclipse, bao gồm bộ debugger cho Python.

Mỗi SDK cũng bao gồm một công cụ cho việc tương tác với các ứng dụng chạy trên App Engine. Chủ yếu là sử dụng cho việc upload code lên App Engine. Chúng ta cũng có thể sử dụng tool để download các file log từ ứng dụng đang chạy, hoặc quản lý danh mục ứng dụng.

SDK Python và Java gồm một chức năng mà chúng ta có thể cài đặt trong ứng dụng cho việc truy cập có thứ tự từ xa một cách an toàn đến ứng dụng. SDK Python bao gồm các công cụ dùng cho việc xử lý dữ liệu lớn, như upload dữ liệu từ file text, và tải lượng lớn dữ liệu, hoặc sao lưu cho mục đích lưu trữ. SDK cũng có command-line shell của Python cho việc test, kiểm lỗi và thao tác với dữ liệu. (Các tool này có trong SDK cho Python, nhưng cũng làm việc với các ứng dụng Java sử dụng phiên bản Java có chức năng truy cập từ xa). Chúng ta có thể viết đoạn mã script và chương trình sử dụng chức năng truy cập từ xa cho việc vận chuyển lượng lớn dữ liệu và các biện pháp bảo trì khác.

.

**Python Runtime:**

AppEngine hỗ trợ Python Runtime phiên bản 2.5.2. Hầu hết các lib của Python Standard Library đều được hỗ trợ. Tuy nhiên do vấn đề về security nên các extensions viết bằng C sẽ không được hỗ trợ. Điều này có nghĩa rằng bạn không thể nhúng C extensions  vào app của bạn khi sử dụng App Engine.

Python Runtime cung cấp APIs cho datastore, Google Accounts, App Engine services nói ở phần sau. Để tiện lợi cho việc phát triển web AppEngine cũng cung cấp 1 web framework đơn giản là webapp. Do viết bằng ngôn ngữ Python nên AppEngine hỗ trợ hầu hết các Python framework như Django, CherryPy, Pylons, web.py với một ít thay đổi nhất định.

**Datastore:**

Datastore là cơ chế để thao tác với dữ liệu trên hệ thống dữ liệu phân tán của Google. APIs của Datastore sẽ cung cấp cho bạn Interface để có thể thao tác với các dữ liệu phía dưới.

**Google Accounts:**

AppEngine liên kết mật thiết với tài khoản Google. Bạn có thể cho user login vào ứng dụng của mình bằng tài khoản Google của họ và còn nhiều thứ khác. Users API sẽ xử lý vấn đề này.

**App Engine Services:**

Cung cấp nhiều dịch vụ để bạn có thể sử dụng cho ứng dụng của mình. Những dịch vụ có thể liệt kê ở đây là: URL Fetch, Mail, Memcache, Image Manipulation.

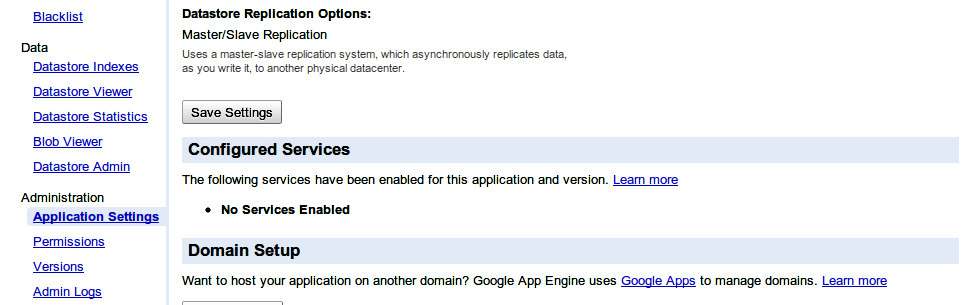
**Sức mạnh của google app engine**

**Datastore**

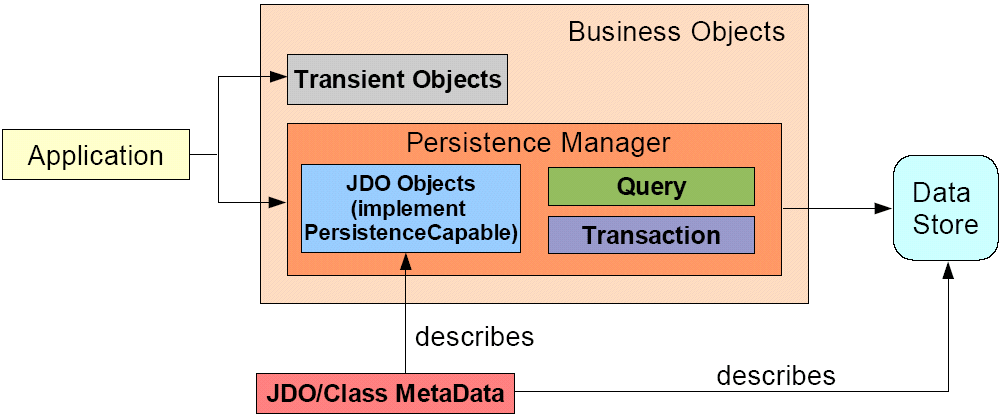
Kho dữ liệu của App Engine đem lại khả năng lưu trữ mạnh mẽ cho ứng dụng web, và tốc độ khi truy cập dữ liệu. Ứng dụng tạo ra những thực thể (entity) và dữ liệu được chứa trong thuộc tính (property) của mỗi thực thể. Tất cả các câu truy vấn dữ liệu đều đã được đánh chỉ mục từ trước để đem lại sự nhanh nhất khi truy vấn số lượng lớn dữ liệu.

App Engine cung cấp 2 sự lựa chọn cho chúng ta về kho dữ liệu, bạn chỉ có thể chọn lựa khi bạn là admin của hệ thống. Khi đó hãy vào Admin Console để tùy chỉnh:

* **Kho dữ liệu Master/Slave**: dựa theo mô hình hệ thống master-slave, hệ thống sẽ ghi lại dữ liệu một cách bất đồng bộ đối với trung tâm dữ liệu vật lý. Ở bất cứ thời điểm nào cũng chỉ có 1 trung tâm dữ liệu là chủ (master) mà thôi. Đảm bảo tính sẵn sàng cho việc đọc và truy vấn dữ liệu, nhưng phải mất một ít thời gian khi trung tâm dữ liệu gặp sự cố hoặc thời gian chết. Tùy chọn này cần lượng lưu trữ và CPU thấp nhất. Đây là tùy chọn mặc định.
* **Kho dữ liệu High Replication**: dựa theo giải thuật Paxos, cung cấp tính sẵn sàng cho việc đọc và ghi dữ liệu rất cao. Nhưng tùy chọn này cần lượng lưu trữ và CPU cao gấp 3 lần so với tùy chọn Master/Slave.

Hình 4.1: Màn hình thiết lập tuỳ chọn các lưu trữ của datastore

Google App Engine cung cấp khả năng để lưu giữ lâu bền dữ liệu khi sử dụng JDO, là một tiêu chuẩn Java cho lưu giữ lâu bền. Đối với hầu hết các nhà phát triển Java, lưu giữ lâu bền dữ liệu thường ngụ ý việc ghi lưu thông tin vào một cơ sở dữ liệu quan hệ; tuy nhiên, trong trường hợp của Google, cơ chế lưu trữ nằm dưới là sở hữu độc quyền của Big Table, không phải là cơ sở dữ liệu quan hệ. Điều đó không thực sự quan trọng: các chi tiết về cách Google lưu giữ lâu bền các thuộc tính cụ thể phần lớn được ẩn dấu với người sử dụng. Chỉ cần nói rằng, bạn có thể sử dụng các đối tượng Java bình thường (hoặc các đối tượng Groovy, cho vấn đề này) để xây dựng một ứng dụng có thể lưu trữ thông tin như bạn sẽ làm trong bất kỳ ứng dụng nào khác.



JDO là khá đơn giản. Chúng ta tạo các POJO — Plain Old Java Objects (các đối tượng thuần Java) (có thể các có mối quan hệ với các đối tượng Java khác) — mà bạn khai báo có khả năng lưu giữ lâu bền thông qua chú giải mức lớp **@PersistenceCapable**. Bạn xác định các thuộc tính nào của đối tượng được lưu giữ lâu bền qua các chú giải **@Persistent**. Ví dụ, tôi muốn lưu trữ các sự kiện thi đấu ba môn phối hợp (lúc này tôi sẽ tập trung vào sự kiện này mà không phải là nhiều kết quả khác có liên quan đến một cuộc thi đấu ba môn phối hợp) có nghĩa là, một sự kiện có một tên (tên của cuộc thi đấu ba môn phối hợp), có thể là có một mô tả (cuộc thi đấu ba môn phối hợp kiểu nào) và một ngày tháng.

|  |
| --- |
| import java.util.Date;  import javax.jdo.annotations.PersistenceCapable;  import javax.jdo.annotations.Persistent;  import javax.jdo.annotations.IdentityType;  @PersistenceCapable(identityType = IdentityType.APPLICATION)  public class Triathlon {  @Persistent  private Date date;  @Persistent  private String name;  @Persistent  private String description;  } |

Lưu trữ lâu bền dữ liệu, bất kể cơ chế của lớp dưới (đó là, cơ sở dữ liệu quan hệ hoặc Big Table của Google) vẫn đòi hỏi khái niệm về khóa(Primary key): một cách để bảo đảm tính độc nhất của các khía cạnh dữ liệu khác nhau để tránh những xung đột dữ liệu. Ví dụ, trong trường hợp của một cuộc thi đấu ba môn phối hợp, khóa có thể là tên của cuộc thi đấu ba môn phối hợp này.Nếu hai cuộc thi đấu ba môn phối hợp có cùng tên, thì khóa có thể là tổ hợp gồm tên và ngày tháng. Bất kể người dùng chọn để biểu diễn một khoá với App Engine của Google và JDO như thế nào, người dùng phải chỉ định một khóa trong đối tượng JDO của bạn thông qua chú giải @PrimaryKey. Người dùng cũng có thể chọn các chiến lược để sinh ra một khóa như thế nào — hoặc do người dùng làm hoặc do cơ sở hạ tầng của Google.

Kho dữ liệu App Engine lưu trữ các đối tượng dữ liệu là thực thể (entity). Mỗi thực thể có 1 hay nhiều thuộc tính. Ta có thể hình dung đối với CSDL quan hệ thì 1 entity được ví như là 1 dòng record trong một bảng, tuy nhiên có một sự khác biệt đôi chút. Kho dữ liệu của App Engine được xây dựng theo hướng CSDL hướng đối tượng (object database), đây là 1 loại CSDL vẫn còn mới và vẫn chưa được thông dụng. CSDL quan hệ mới là loại đang được sử dụng rộng rãi và phổ biến. Vì thế kho dữ liệu của App Engine đã cố gắng xây dựng hết tất cả những đặc điểm vốn có của CSDL quan hệ để đem lại cho người phát triển sự quen thuộc. Chúng ta vẫn có những câu truy vấn, phiên làm việc (transaction), khóa … như chúng ta từng làm việc với CSDL quan hệ.

Đầu tiên là entity, mỗi entity sẽ có 1 khóa để phân biệt mình với tất cả các entity khác trong datastore. Chúng ta lưu ý là trong datastore không hề có bảng như trong CSDL quan hệ thông thường. Đối với datastore, ta nên hiểu nó chỉ là 1 bảng và là 1 cái bảng rất lớn. Đó là lý do mà nó có tên là BigTable. Vì thế khóa để phân biệt entity này với tất cả các entity khác trong cái bảng to lớn ấy. Khóa có 2 dạng, chúng ta có thể tự đặt tên cho khóa để có thể sử dụng sau này; hoặc nếu ta không chỉ định tên cho khóa thì hệ thống sẽ phát sinh số ngẫu nhiên (id) cho entity của chúng ta. Vì thế, 1 entity hoặc là có tên khóa, hoặc là có số id của khóa. Nên chú ý rằng việc gán tên cho khóa chỉ xảy ra khi entity được khởi tạo và giá trị đó không thể thay đổi được.

Mỗi entity có 2 loại thuộc tính: thuộc tính tĩnh (fixed property) và thuộc tính động (dynamic property). Với thuộc tính tĩnh, ta có thể hình dung nó như là các cột trong CSDL quan hệ, có giá trị và cùng kiểu dữ liệu. Tuy nhiên đối với thuộc tính động, số lượng thuộc tính này thay đổi tùy vào mục đích của chúng ta. Vì thế, mặc dù cùng 1 entity nhưng 2 thể hiện có thể có số lượng thuộc tính động khác nhau, và có thể các thuộc tính động có tên giống nhau những kiểu dữ liệu khác nhau. Mỗi thuộc tính đều có kiểu dữ liệu riêng của nó, các kiểu dữ liệu trên datastore đã được đánh chỉ mục một cách tự động. Tuy nhiên chúng ta phải khai báo chỉ mục cho những thuộc tính cần sử dụng trong câu query vì đấy là yêu cầu bắt buộc của datastore.

Trong datastore, để có thể lấy được các entity chứa dữ liệu, ta có 2 cách. Thứ nhất là biết khóa của nó và lấy dựa vào khóa. Thứ 2, phổ biến hơn, là sử dụng câu truy vấn (query). Trong datastore, các thuộc tính được sử dụng trong câu query bắt buộc phải được khai báo chỉ mục. Nếu không câu query sẽ không hoạt động. Việc khai báo chỉ mục sẽ giúp câu query đạt được tốc độ tốt nhất.

Khi chúng ta sử dụng nhiều câu query cho cùng một mục đích và muốn đảm bảo rằng, hoặc tất cả câu query đều thành công hoặc nếu thất bại sẽ không có câu query nào được phép chạy. Khi đó ta cần transaction. Transaction để đảm bảo những việc như thế. Tuy nhiên, các entity được sử dụng trong transaction phải thuộc cùng một nhóm. Việc cấu hình để các entity thuộc cùng một nhóm được thực hiện lúc khởi tạo entity và sẽ không được thay đổi sau này.

Khi khởi tạo entity, nếu entity đó không là con của bất kì entity nào khác thì nó là entity gốc (root entity). Lúc đó entity gốc sẽ nằm trong nhóm của chính mình. Bất kì entity nào muốn nằm trong nhóm này thì hoặc sẽ phải khai báo entity này là cha của mình, hoặc khai báo bất kì entity nào khác nằm trong nhóm này là cha của mình.

Đơn giản bằng một ví dụ sau: khóa của đối tượng cuộc thi đấu ba môn phối hợp của tôi được biểu diễn như là một đối tượng Long Java thông thường và tôi sẽ để cho Google chọn giá trị thực tế bằng cách xác định một chiến lược giá trị.

|  |
| --- |
| import java.util.Date;  import javax.jdo.annotations.IdGeneratorStrategy;  import javax.jdo.annotations.PersistenceCapable;  import javax.jdo.annotations.Persistent;  import javax.jdo.annotations.PrimaryKey;  import javax.jdo.annotations.IdentityType;  import org.apache.commons.lang.builder.EqualsBuilder;  import org.apache.commons.lang.builder.HashCodeBuilder;  import org.apache.commons.lang.builder.ReflectionToStringBuilder;  @PersistenceCapable(identityType = IdentityType.APPLICATION)  public class Triathlon {  @PrimaryKey  @Persistent(valueStrategy = IdGeneratorStrategy.IDENTITY)  private Long id;  @Persistent  private Date date;  @Persistent  private String name;  @Persistent  private String description;  public Triathlon(Date date, String name, String description) {  super();  this.date = date;  this.name = name;  this.description = description;  }  //...setters and getters left out  public String toString() {  return ReflectionToStringBuilder.toString(this);  }  public int hashCode() {  return HashCodeBuilder.reflectionHashCode(this);  }  public boolean equals(Object obj) {  return EqualsBuilder.reflectionEquals(this, obj);  }  } |

## BLOBSTORE

Google App Engine bao gồm 1 dịch vụ gọi là Blobstore, cho phép ứng dụng có thể xử lý những đối tượng mang dữ liệu lên đến 2 gigabyte. Những đối tượng này gọi là giá trị của Blobstore hay là blob. Blob được tạo ra trong quá trình upload dữ liệu từ web form. Ứng dụng không tạo blob 1 cách trực tiếp mà tạo gián tiếp thông qua hành động submit web form hay những request theo dạng POST. Với Blobstore API, blob có thể dùng để phục vụ người dùng, hoặc để ứng dụng truy cập dưới dạng luồng (stream).

Blob được định nghĩa trong Blobstore không có liên hệ gì với kiểu blob được định nghĩa trong Datastore.Blob rất hữu ích khi dùng để xử lý những file lớn, như file video hay hình ảnh, và còn cho phép người dùng upload lượng lớn dữ liệu.

Blob không thể thay đổi sau khi được tạo, mặc dù ta có thể xóa nó. Mỗi blob có 1 record chứa dữ liệu được lưu trong datastore gọi là blob info, record này chứa thông tin về blob như thời gian nó được tạo ra và kiểu của nội dung. Bạn có thể dùng blob key để lấy những record blob info khi truy vấn.

Ứng dụng có thể đọc 1 phần dữ liệu của blob bất cứ lúc nào bằng cách gọi API. Kích cỡ tối đa của phần dữ liệu đó gần 1 megabyte. Con số này được lưu trong hằng google.appengine.ext.blobstore.MAX\_BLOB\_FETCH\_SIZE của python. Ứng dụng không thể tự động tạo hay thay đổi giá trị của Blobstore ngoại trừ việc upload dữ liệu từ phía người dùng.Sau đây là một ví dụ hiện thực nó:

Lớp Upload.java để xử lý blobkey vừa được tạo ra từ các file được upload thành công

|  |
| --- |
| **import** com.google.appengine.api.blobstore.BlobKey;  **import** com.google.appengine.api.blobstore.BlobstoreService;  **import** com.google.appengine.api.blobstore.BlobstoreServiceFactory;  @SuppressWarnings("serial")  **publicclass** Upload **extends** HttpServlet {  **private** BlobstoreService blobstoreService =  BlobstoreServiceFactory.*getBlobstoreService*();  **publicvoid** doPost(HttpServletRequest req, HttpServletResponse res)  **throws** ServletException, IOException {    Map<String, BlobKey> blobs =  blobstoreService.getUploadedBlobs(req);  BlobKey blobKey = blobs.get("fileupload");  **if** (blobKey == **null**){  res.sendRedirect("/");  } **else** {  res.sendRedirect("/serve?blob-key=" +  blobKey.getKeyString());  }  }  } |

Lớp Serve.java để xử lý việc download các blobkey

|  |
| --- |
| **import** com.google.appengine.api.blobstore.BlobKey;  **import** com.google.appengine.api.blobstore.BlobstoreService;  **import** com.google.appengine.api.blobstore.BlobstoreServiceFactory;  @SuppressWarnings("serial")  **publicclass** Serve **extends** HttpServlet {  **private** BlobstoreService blobstoreService =  BlobstoreServiceFactory.*getBlobstoreService*();  **publicvoid** doGet(HttpServletRequest req, HttpServletResponse res)  **throws** IOException {  BlobKey blobKey = **new** BlobKey(req.getParameter("blob-key"));  blobstoreService.serve(blobKey, res);  }  } |

## CAPABILITIES

Với Capabilities API, chúng ta có thể xác định được khi nào các service của App Engine có thể hoặc không thể phục vụ chúng ta.

|  |
| --- |
| **import** com.google.appengine.api.capabilities.\*;  CapabilitiesService service =  CapabilitiesServiceFactory.*getCapabilitiesService*();  CapabilityStatus status = service.getStatus(Capability.*IMAGES*).getStatus();  **if** (status == CapabilityStatus.DISABLED) {  // Images API is not available.  } |

Đây là những hằng số hỗ trợ:

| Capability | Tham số cho Capability |
| --- | --- |
| Availability of the blobstore | [*BLOBSTORE*](http://code.google.com/appengine/docs/java/javadoc/com/google/appengine/api/capabilities/Capability.html#BLOBSTORE) |
| Datastore reads | [*DATASTORE*](http://code.google.com/appengine/docs/java/javadoc/com/google/appengine/api/capabilities/Capability.html#DATASTORE) |
| Datastore writes | [*DATASTORE\_WRITES*](http://code.google.com/appengine/docs/java/javadoc/com/google/appengine/api/capabilities/Capability.html#DATASTORE_WRITE) |
| Availability of the Images service | [*IMAGES*](http://code.google.com/appengine/docs/java/javadoc/com/google/appengine/api/capabilities/Capability.html#IMAGES) |
| Availability of the Mail service | [*MAIL*](http://code.google.com/appengine/docs/java/javadoc/com/google/appengine/api/capabilities/Capability.html#MAIL) |
| Availability of the Memcache service | [*MEMCACHE*](http://code.google.com/appengine/docs/java/javadoc/com/google/appengine/api/capabilities/Capability.html#MEMCACHE) |
| Availability of the Task Queue service | [*TASKQUEUE*](http://code.google.com/appengine/docs/java/javadoc/com/google/appengine/api/capabilities/Capability.html#TASKQUEUE) |
| Availability of the URL Fetch service | [*URL\_FETCH*](http://code.google.com/appengine/docs/java/javadoc/com/google/appengine/api/capabilities/Capability.html#URL_FETCH) |
| Availability of the XMPP service | [*XMPP*](http://code.google.com/appengine/docs/java/javadoc/com/google/appengine/api/capabilities/Capability.html#XMPP) |

Hằng số hỗ trợ Capability cho Java

## MAIL

Ứng dụng App Engine có thể gửi mail thay mặt người quản trị hoặc những người có tài khoản Google và có thể nhận mail từ nhiều địa chỉ khác nhau. Ứng dụng gửi mail nhờ vào dịch vụ Mail của Google. Khi mail được gửi đến ứng dụng, App Engine sẽ nhận nó và tạo ra 1 request HTTP gửi đến ứng dụng của chúng ta thông qua phương thức POST.

Gửi mail là bất đồng bộ. Các API hỗ trợ gửi mail chỉ đơn thuần là đưa những tin nhắn đến dịch vụ Mail và hết. Dịch vụ Mail sẽ xếp các tin nhắn đó vào 1 hàng đợi và tất nhiên, cố gắng gửi nó đi nếu như mail server ở bên nhận đang gặp sự cố. Nếu có vấn đề trục trặc xảy ra, những tin nhắn chứa thông tin về lỗi sẽ được gửi lại cho người gửi.

Địa chỉ của người gửi (trường “From” trong mail) bắt buộc phải thuộc vào 1 trong những loại sau:

* Email của người quản trị ứng dụng. Có thể thêm người quản trị bằng cách vào màn hình quản trị để mời.
* Email của cá nhân nào đó hiện tại đang đăng nhập vào Google.
* Email thỏa mãn công thức sau: [string@APP-ID.appspotmail.com](mailto:string@APP-ID.appspotmail.com)

Email được gửi đến ứng dụng của chúng ta bằng những request theo giao thức HTTP. Để xử lý những mail này, chúng ta phải khai báo đoạn script để xử lý trong file cấu hình của ứng dụng và sau đó là hiện thực phần script đó.

Mail service Java API hiện thực interface JavaMail (javax.mail) để thực hiện gửi và nhận mail.Để gửi một thông điệp qua email, ứng dụng cần tạo đối tượng MimeMessage, thiết lập các thành phần cần thiết cho email, và gửi nó bằng phương thức static send() của lớp Transport. Thông điệp được tạo bằng đối tượng Session của JavaMail.

**import** java.util.Properties;

**import** javax.mail.Message;

**import** javax.mail.MessagingException;

**import** javax.mail.Session;

**import** javax.mail.Transport;

**import** javax.mail.internet.AddressException;

**import** javax.mail.internet.InternetAddress;

**import** javax.mail.internet.MimeMessage;

// ...

Properties props = **new** Properties();

Session session = Session.getDefaultInstance(props, **null**);

String msgBody = "...";

**try** {

Message msg = **new** MimeMessage(session);

msg.setFrom(**new** InternetAddress("admin@example.com",

"Example.com Admin"));

msg.addRecipient(Message.RecipientType.*TO*,

**new** InternetAddress("user@example.com", "Mr. User"));

msg.setSubject("Your Example.com account has been activated");

msg.setText(msgBody);

Transport.*send*(msg);

} **catch** (AddressException e) {

// ...

} **catch** (MessagingException e) {

// ...

}

MimeMessage cung cấp một số API cho việc gửi mail như sau:

* setFrom(Address address): thiết lập địa chỉ người gửi.
* addRecipient(Message.RecipientType type, Address address): thêm địa chỉ người nhận. RecipientType bao gồm Message.RecipientType.TO, Message.RecipientType.CC và Message.RecipientType.BCC.
* setSubject(String subject): thiết lập tiêu đề cho email.
* setText(String content): thiết lập nội dung cho email (plaintext).

Từ AppEngine SDK 1.2.6., GAE đã hỗ trợ cơ chế nhận mail trong ứng dụng. Chúng ta có thể nhận mail, phân tích mail và áp dụng các nghiệp vụ phù hợp tương ứng.

Trước tiên, ta cần bật dịch vụ nhận mail của GAE, thêm đọan cấu hình sau vào file appengine-web.xml :

<appengine-web-app>

...

<inbound-services>

<service>mail</service>

</inbound-services>

...

</appengine-web-app>

Mail gửi đến dưới dạng một HTTP request theo phương thức POST, gửi đến URL /\_ah/mail/<address>, với <address> là email của người nhận theo cấu trúc [string@appid.appspotmail.com](mailto:string@appid.appspotmail.com) với appid của ứng dụng. Do đó ta cần tạo một servlet map với URL /\_ah/mail/\* và hiện thực phương thức doPost để xử lý việc nhận mail.

<servlet>

<servlet-name>ReceiveEmailServlet</servlet-name>

<servlet-class>ReceiveEmailServlet</servlet-class>

</servlet>

<servlet-mapping>

<servlet-name>ReceiveEmailServlet</servlet-name>

<url-pattern>/\_ah/mail/admin\*</url-pattern>

</servlet-mapping>

<security-constraint>

<web-resource-collection>

<web-resource-name>Secure Pages</web-resource-name>

<url-pattern>/\_ah/mail/\*</url-pattern>

</web-resource-collection>

<auth-constraint>

<role-name>admin</role-name>

</auth-constraint>

</security-constraint>

Trong ví dụ trên, /\_ah/mail/\* ứng với tất cả các email của ứng dụng. Ta có thể tùy chỉnh dùng regular expression để điều hướng xử lý các mail cho các servlet thích hợp.

Để phân tích nội dung mail nhận được, chúng ta dùng đối tượng MimeMessage, truyền vào java.io.InputStream và JavaMail session.

**import** java.io.IOException;

**import** java.util.Properties;

**import** javax.mail.Session;

**import** javax.mail.internet.MimeMessage;

**import** javax.servlet.http.\*;

**publicclass** MailHandlerServlet **extends** HttpServlet {

**publicvoid** doPost(HttpServletRequest req, HttpServletResponse resp)

**throws** IOException {

Properties props = **new** Properties();

Session session = Session.getDefaultInstance(props,**null**);

MimeMessage message = **new** MimeMessage(session, req.getInputStream();

//...

Sau đó, ta có thể sử dụng các API của đối tượng MimeMessageđể lấy ra thông tin:

* Address[] getFrom(): trả về địa chỉ mail người gửi.
* String getContentType(): trả về content type.
* Object getContent(): trả về đối tượng implerments interface Multipart. Bạn có thể gọi phương thức getCount() để xác định có bao nhiêu phần trong nội dung email, và getBodyPart(int index) để lấy ra các phần riêng biệt.

## MEMCACHE

Một ứng dụng web có khả năng mở rộng cao thường sử dụng 1 bộ nhớ để cache lại dữ liệu trước khi phân tán hoặc để tránh liên tục truy cập CSDL. App Engine hỗ trợ dịch vụ Memcache cho mục đích này.

Một trong những cách để sử dụng Memcache là để tăng tốc độ truy vấn CSDL. Nếu nhiều request thực hiện đồng thời 1 câu truy vấn thì ta có thể cache lại kết quả của câu truy vấn đó. Những request sau này chỉ cần xem trong cache, nếu như không có dữ liệu hoặc dữ liệu bị hết hạn thì mới truy vấn xuống CSDL. Dữ liệu của session, thông tin user hay kết quả những câu query nên được cache lại.

Memcache dùng để lưu dữ liệu tạm thời rất tốt. Tuy nhiên, nên nhớ là dữ liệu trong Memcache chưa được lưu trữ xuống CSDL, vì thế nó có thể mất bất cứ lúc nào. Dữ liệu trong Memcache có thể hết hạn bất cứ lúc nào và có khi còn trước cả thời gian được chỉ định. Vì thế những thông tin trong Memcache, tốt nhất cũng nên lưu xuống CSDL.

Mặc định thì dữ liệu trong Memcache sẽ tồn tại rất lâu nếu có thể. Dữ liệu chỉ bị xóa khỏi cache khi lượng memory cho cache còn quá ít. Và dữ liệu nào được thêm vào cache đầu tiên nhất sẽ bị xóa trước tiên. Chương trình có thể cung cấp thời gian hết hạn cho dữ liệu trong cache bằng cách: điền 1 con số tính bằng giây so với thời điểm hiện tại, hoặc là một thời điểm nào đó trong tương lai.

Nếu chẳng may gặp sự cố thì dữ liệu trong cache sẽ biến mất. Vì thế không nên tin tưởng tuyệt đối vào dữ liệu được lưu trong cache.

Giá trị được lưu trong cache có dung lượng tối đa là 1megabyte. Không có độ dài giới hạn cho key, tuy nhiên nếu key có độ dài lớn hơn 250 byte thì sẽ được lưu bản hash 250 byte. Có thể làm việc cùng lúc với nhiều dữ liệu trong cache, tuy nhiên tổng dung lượng không được vượt quá 32 megabyte.

##### **API CỦA APP ENGINE**

Để làm việc với Memcache trong API của App Engine, chúng ta cần import thư viện

**import** com.google.appengine.api.memcache.\*;

Để tương tác với Memcache chúng ta cần tạo ra đối tượng MemcacheService như sau:

MemcacheService memcache =

MemcacheServiceFactory.getMemcacheService();

Và tất cả những gì chúng ta cần để làm việc được với Memcache là tìm hiểu về đối tượng MemcacheService này.

MemcacheService lưu dữ liệu với hai tham số key và value, thông qua phương thức put(key, value):

* Lưu trữ 1 đối tượng:

memcache.**put**("cachekey", cachedata);

* Lưu trữ nhiều đối tượng:

Map<Object, Object> listcacheobj = **new** HashMap<Object,Object>();

listcacheobj.put("cachekey1", "cachedata1");

listcacheobj.put("cachekey2", "cachedata2");

memcache.putAll(listcacheobj);

* MemcacheService lấy các dữ liệu đã lưu dựa vào key mà chúng ta đã đặt ban đầu thông qua phương thức get(key);
  + Lấy một đối tượng:

memcache.**get**("cachekey");

* + Lấy nhiều đối tượng:

List<Object> listcacheobj = Arrays<Object>.*asList*("cachekey1","cachekey2");

Map<Object, Object> articleSummaries = memcache.getAll(listcacheobj);

* + MemcacheService có thể xóa một hoặc nhiều đối tượng dựa vào key của nó, bằng phương thức delete(), và deleteAll(), tuy nhiên đối với phương thức deleteAll() thì tổng kích thước cho mỗi lần thực hiện phải nhỏ hơn 1MB.

Chúng ta có thể lưu trữ các đối tượng memcache với không gian tên khác nhau để tránh tình trạng xung đột khóa. Ví dụ chúng ta có hai lĩnh vực cần lưu trữ là Tin tức và Giải trí, chúng ta có thể lưu với hai không gian tên khác nhau và khi đó các key ở không gian tên TinTuc hoàn toàn có thể trùng với key ở không gian tên GiaiTri mà không hề xảy ra xung đột. Cách thiết lập không gian tên như sau: memcache.setNameSpace("TinTuc");

Ngoài ra chúng ta còn có thể thiết lập thời gian lưu trữ của memcache, cũng như lấy các số liệu thống kê trên memcache…

## URL FETCH

Ứng dụng App Engine có thể tương tác với những ứng dụng khác hay truy cập những nguồn tài nguyên khác trên web bằng địa chỉ URL. Ứng dụng có thể sử dụng URL Fetch Service để có thể tương tác với các ứng dụng khác thông qua giao thức HTTP và HTTPS.

Ứng dụng có thể lấy dữ liệu từ địa chỉ URL dạng HTTP (http://...) hay HTTPS (https://...)

Địa chỉ URL để lấy dữ liệu có thể sử dụng các cổng: 80 – 90, 440 – 450, 1024 – 65535. Nếu trong địa chỉ URL không chỉ định số cổng thì dạng URL http://... sẽ sử dụng cổng 80, còn dạng https://... sẽ dùng cổng 443.

URL Fetch Service hỗ trợ mọi cách thức lấy dữ liệu trong HTTP như: GET, POST, PUT, DELETE, HEAD; có thể bao gồm header và payload (phần thân của HTTP request).

Không nên để ứng dụng của mình lấy dữ liệu từ URL đang làm việc vì sẽ gây ra tình trạng lặp vô tận.

Có thể đặt thời gian lấy dữ liệu: mặc định là 5 giây và tối đa là 10 giây.

URL Fetch Service hỗ trợ cả 2 dạng request là đồng bộ và bất đồng bộ. Và ứng dụng của chúng ta có thể có tới 10 request bất đồng bộ tại cùng 1 thời điểm.

Nếu chúng ta lấy dữ liệu theo dạng HTTPS, mọi dữ liệu trên request và response đều được mã hóa.

Ứng dụng có thể thiết lập HTTP header cho request. Khi gửi 1 request dạng HTTP POST, nếu không chỉ định phần Content-Type ở header, thì header sẽ được thiết lập giá trị . Đây là Content-Type được dùng bởi web form.

Vì lý do bảo mật, các trường sau đây trong header không được thay đổi:

* Content-Length
* Host
* Vary
* Via
* X-Forward-For

URL Fetch Service sẽ trả về dữ liệu từ phía response, bao gồm code, header và body. Mặc định, nếu dữ liệu trả về là chứa mã code tham khảo đến địa chỉ khác, service sẽ kết nối đến địa chỉ đó để lấy dữ liệu cho ứng dụng. Tối đa là qua 5 tầng kết nối, nếu quá số lượng trên thì service sẽ báo lỗi.

Nếu bạn có sử dụng tường lửa, ứng dụng của bạn có thể dử dụng Google Secure Data Connector (SDC). Thông tin chi tiết về SDC, xin tham khảo: <http://code.google.com/securedataconnector/>

URL Fetch Service cho request có dung lượng tối đa là 1 megabyte và response tối đa là 32 megabyte.

## 1.2 Kiến trúc Android

Việc hiểu được các thành phần (component) hay kiến trúc của một ứng dụng Android là rất cần thiết cho việc lập trình. Các thành phần này được chia làm 6 loại bao gồm:

1. **Activity:** hiểu một cách đơn giản thì Activity là nền của 1 ứng dụng giống như 1 Midlet ở lập trình J2ME. Khi khởi động 1 ứng dụng Android nào đó thì bao giờ cũng có 1 main Activity được gọi, hiển thị màn hình giao diện của ứng dụng cho phép người dùng tương tác.
2. **Service:** thành phần chạy ẩn trong Android. Service sử dụng để update dữ liệu, đưa ra các cảnh báo (Notification) và không bao giờ hiển thị cho người dùng thấy.
3. **Content Provider:** kho dữ liệu chia sẻ. Content Provider được sử dụng để quản lý và chia sẻ dữ liệu giữa các ứng dụng.
4. **Intent:** nền tảng để truyền tải các thông báo. Intent được sử dụng để gửi các thông báo đi nhằm khởi tạo 1 Activity hay Service để thực hiện công việc bạn mong muốn. VD: khi mở 1 trang web, bạn gửi 1 intent đi để tạo 1 activity mới hiển thị trang web đó.
5. **Broadcast Receiver:** thành phần thu nhận các Intent bên ngoài gửi tới. VD: bạn viết 1 chương trình thay thế cho phần gọi điện mặc định của Android, khi đó bạn cần 1 BR để nhận biết các Intent là các cuộc gọi tới.
6. **Notification:** đưa ra các cảnh báo mà không làm cho các Activity phải ngừng hoạt động.

**1.3 Vòng đời sống của một ứng dụng Android:**

1. **Vòng đời sống của process:**

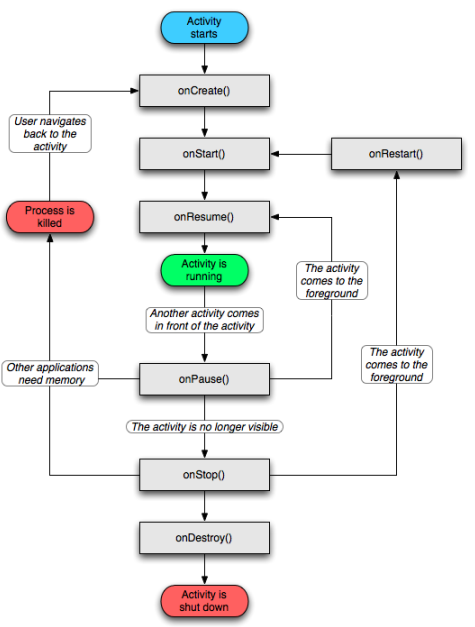
Android có cơ chế quản lý các process theo chế độ ưu tiên. Các process có priority thấp sẽ bị Android giải phóng mà không hề cảnh báo nhằm đảm bảo tài nguyên.

* Foreground process: là process của ứng dụng hiện thời đang được người dùng tương tác.
* Visible process: là process của ứng dụng mà activity đang hiển thị đối với người dùng (onPaused() của activity được gọi).
* Service process: là Service đang running.
* Background process: là process của ứng dụng mà các activity của nó ko hiển thị với người dùng (onStoped() của activity được gọi).
* Empty process: process không có bất cứ 1 thành phần nào active.

Theo chế độ ưu tiên thì khi cần tài nguyên, Android sẽ tự động kill process, trước tiên là các empty process.

1. **Vòng đời sống của một Activity:**

Biểu đồ miêu tả Activity state



Vòng đời của Activity:

- Entire lifetime: Từ phương thức onCreate( ) cho tới onDestroy( )

- Visible liftetime: Từ phương thức onStart( ) cho tới onStop( )

- Foreground lifetime: Từ phương thức onResume( ) cho tới onPause( )

* 1. **Các thành phần trong Android**

**1.4.1 Actitvity:**

**Actitvity** là thành phần quan trọng nhất và đóng vai trò chính trong xây dựng ứng dụng Android. Hệ điều hành Android quản lý Activity theo dạng stack: khi một Activity mới được khởi tạo, nó sẽ được xếp lên đầu của stack và trở thành running activity, các Activity trước đó sẽ bị tạm dừng và chỉ hoạt động trở lại khi Activity mới được giải phóng.

**Activity bao gồm 4 state:**

- active (running): Activity đang hiển thị trên màn hình (foreground).

- paused: Activity vẫn hiển thị (visible) nhưng không thể tương tác (lost focus).

VD: một activity mới xuất hiện hiển thị giao diện đè lên trên activity cũ, nhưng giao diện này nhỏ hơn giao diện của activity cũ, do đó ta vẫn thấy được 1 phần giao diện của activity cũ nhưng lại không thể tương tác với nó.

- stop: Activity bị thay thế hoàn toàn bởi Activity mới sẽ tiến đến trạng thái stop

- killed: Khi hệ thống bị thiếu bộ nhớ, nó sẽ giải phóng các tiến trình theo nguyên tắc ưu tiên. Các Activity ở trạng thái stop hoặc paused cũng có thể bị giải phóng và khi nó được hiển thị lại thì các Activity này phải khởi động lại hoàn toàn và phục hồi lại trạng thái trước đó.

**1.4.2 Intent:**

Theo định nghĩa của Google, Intent là một miêu tả về một hoạt động cần được thực hiện. Còn nói một cách đơn giản và dễ hiểu hơn, Intent là một cơ cấu cho phép truyền thông điệp giữa các thành phần của 1 ứng dụng và giữa các ứng dụng với nhau.

**a. Các thuộc tính của Intent:**

- action: là hành động được thực hiện, vd: ACTION\_VIEW, ACTION\_MAIN

- data: là dữ liệu sẽ được xử lý trong action, thường được diễn tả là một Uri (Uniform Resource Identifier).

VD:

* ACTION\_VIEW content://contacts/people/1 - Hiển thị thông tin về người với mã danh 1
* ACTION\_DIAL content://contacts/people/1 - Hiển thị màn hình gọi đến người với mã danh 1
* ACTION\_DIAL tel:123 - Hiển thị màn hình gọi với số gọi là 123

**Ngoài ra còn có 1 số thuộc tính mà ta có thể bổ sung vào Intent:**

- category: bổ sung thêm thông tin cho action của Intent. CATEGORY\_LAUNCHER thông báo sẽ thêm vào Launcher như là một ứng dụng top-level

- type: chỉ rõ kiểu của data

- component: chỉ rõ thành phần sẽ nhận và xử lý intent. Khi thuộc tính này được xác định thì các thuộc tính khác sẽ trở thành thuộc tính phụ.

- extras: mang theo đối tượng Bundle chứa các giá trị bổ sung.

VD:

* ACTION\_MAIN và CATEGORY\_HOME: trở về màn hình Home của Android (khi bấm nút Home của di động)

1. **Phân loại Intent:**

Intent được chia làm 2 loại:

- Explicit Intents: intent đã được xác định thuộc tính component, nghĩa là đã chỉ rõ thành phần sẽ nhận và xử lý intent. Thông thường intent dạng này sẽ không bổ sung thêm các thuộc tính khác như action, data. Explicit Intent thương được sử dụng để khởi chạy các activity trong cùng 1 ứng dụng.

- Implicit Intents: Intent không chỉ rõ component xử lý, thay vào đó nó bổ sung thông tin trong các thuộc tính. Khi intent được gửi đi, hệ thống sẽ dựa vào những thông tin này để quyết định component nào thích hợp nhất để xử lý nó.

**c. Intent Filter:**

Activity, Service và BroadCast Receiver sử dụng Intent Filter để thông báo cho hệ thống biết các dạng Implicit Intent mà nó có thể xử lý. Nói cách khác, Intent Filter là bộ lọc Intent, chỉ cho những Intent được phép đi qua nó.

Intent Filter mô tả khả năng của component định nghĩa nó. Khi hệ thống bắt được 1 Implicit Intent (chỉ chứa 1 số thông tin chung chung về action, data và category...), nó sẽ sử dụng những thông tin trong Intent này, kiểm tra đối chiếu với Intent Filter của các component các ứng dụng, sau đó quyết định khởi chạy ứng dụng nào thích hợp nhất để xử lý Intent bắt được. Nếu có 2 hay nhiều hơn ứng dụng thích hợp, người dùng sẽ được lựa chọn ứng dụng mình muốn.

VD:

* ACTION\_DIAL tel:123 thông thường sẽ được hệ thống giao cho activity Phone Dialer mặc định của Android xử lý.

1. **Một số action thường sử dụng trong Intent:**

* ACTION\_ANSWER - mở Activity để xử lý cuộc gọi tới, thường là Phone Dialer của Android
* ACTION\_CALL - mở 1 Phone Dialer (mặc định là PD của Android) và ngay lập tức thực hiện cuộc gọi dựa vào thông tin trong data URI
* ACTION\_DELETE - mở Activity cho phép xóa dữ liệu mà địa chỉ của nó chứa trong data URI
* ACTION\_DIAL - mở 1 Phone Dialer (mặc định là PD của Android) và điền thông tin lấy từ địa chỉ chứa trong data URI
* ACTION\_EDIT - mở 1 Activity cho phép chỉnh sửa dữ liệu mà địa chỉ lấy từ data URI
* ACTION\_SEND - mở 1 Activity cho phép gửi dữ liệu lấy từ data URI, kiểu của dữ liệu xác định trong thuộc tính type
* ACTION\_SENDTO - mở 1 Activity cho phép gửi thông điệp tới địa chỉ lấy từ data URI
* ACTION\_VIEW - action thông dụng nhất, khởi chạy activity thích hợp để hiển thị dữ liệu trong data URI
* ACTION\_MAIN - sử dụng để khởi chạy 1 Activity

**1.4.3 Service:**

**a. Khái quát về service:**

Service là 1 trong 4 thành phần chính trong 1 ứng dụng Android ( Activity, Service, BroadcastReceiver, ContentProvider) thành phần này chạy trong hậu trường và làm những công việc không cần tới giao diện như chơi nhạc, download, xử lí tính toán…

Một Service có thể được sử dụng theo 2 cách:

- Nó có thể được bắt đầu và được cho phép hoạt động cho đến khi một người nào đó dừng nó lại hoặc nó tự ngắt. Ở chế độ này, nó được bắt đầu bằng cách gọi Context.startService() và dừng bằng lệnh Context.stopService(). Nó có thể tự ngắt bằng lệnh Service.stopSelf() hoặc Service.stopSelfResult(). Chỉ cần một lệnh stopService() để ngừng Service lại cho dù lệnh startService() được gọi ra bao nhiêu lần

- Service có thể được vận hành theo như đã được lập trình việc sử dụng một Interface mà nó định nghĩa. Các người dùng thiết lập một đường truyền tới đối tượng Service và sử dụng đường kết nói đó để thâm nhập vào Service. Kết nối này được thiết lập bằng cách gọi lệnh Context.bindService() và được đóng lại bằng cách gọi lệnh Context.unbindService(). Nhiều người dùng có thể kết nối tới cùng một thiết bị. Nếu Service vẫn chưa được khởi chạy, lệnh bindService() có thể tùy ý khởi chạy nó. Hai chế độ này thì không tách biệt toàn bộ. Bạn có thể kết nối với một Service mà nó đã được bắt đầu với lệnh startService().

**b. Vòng đời của Service**

Giống như một Activity, một Service cũng có các phương thức chu kỳ thời gian mà bạn có thể cài đặt để kiểm soát những sự thay đổi trong trạng thái của nó.

Những phương thức của Service thì ít hơn là của Activity – chỉ có 3 và chúng thì được sử dụng rộng rãi, không được bảo vệ.

void onCreate()

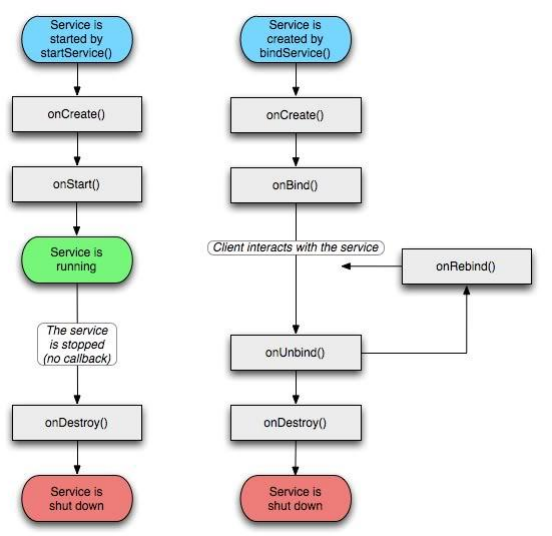
void onStart(Intent intent)

void onDestroy()

Bằng việc thực hiện những phương thức này, bạn có thể giám sát 2 vòng lặp của chu kỳ thời gian của mỗi Service Entire lifetime của một Service diễn ra giữa thời gian onCreate() được gọi ra và thời gian mà onDestroy() trả lại. Giống như một Activity, một Service lại tiết hành cài đặt ban đầu ở onCreate(), và giải phóng tát cả các tài nguyên còn lại ở onDestroy() Ví dụ, một Service phát lại nhạc có thể tạo ra một luồng và bắt đầu chơi nhạc onCreate(),và sau đó luồng chơi nhạc sẽ dừng lại ở onCreate(), Active lifetime của một Service bắt đầu bằng một lệnh tới onStart(). Đâylà phương thức được chuyển giao đối tượng Intent mà đã được thông qua để tới startService() Service âm nhạc sẽ mở đối tượng Intent để quyết định xem sẽ chơi loại nhạc nào và bắt đầu phát nhạc. Không có callback tương đương nào cho thời điểm Service ngừng lại – không có phương thức onStop() .

Các phương thức onCreate() và onDestroy() được gọi cho tất cả các Service dù chúng có được bắt đầu bằng Context.startService() hoặc Context.bindService() hay không. Tuy nhiên thì, onStart() chỉ được gọi ra đối với các Service bắt đầu bằng startService(). Nếu một Service cho phép những Service khác kết nối với nó thì sẽ có thêm các phương thức callback dành cho Service đó để thực hiên IBinder onBind(Intent intent) boolean onUnbind(Intent intent) void onRebind(Intent intent)

Hàm callback onBind() thông qua đối tượng Intent đã đựoc truyền đến bindService và onUnbind() được chuyển giao đối tượng mà đã được chuyển đến. Nếu Service đang được chỉ định (binding), onBind() quay trở lại kênh thông tin mà người dùng sử dụng để tương tác với Service. Phương thức onUnbind() có thể yêu cầu onRebind() được gọi nếu một người dùng kết nối với Service .

Biểu đồ dưới đây minh họa cho các phương thức callback giành cho một Service.

Mặc dù, nó phân tách các Service được tạo ra thông qua startService với các Service mà được tạo ra bằng bindService(). Hãy nhớ rằng bất kì Service nào, cho dù nó được khởi tạo như thế nào thì nó vẫn có thể cho phép các người dùng kết nối tới nó một cách hiệu quả nhất, cho nên bất kì Service nào cũng có thể được chỉ định thông qua các các phương thức onBind()và onUnbind()

**1.4.4 BroadcastReceiver**

BroadcastReceiver (có thể gọi là Receiver là một trong bốn loại thành phần trong ứng dụng Android.Chức năng dùng để nhận các sự kiện mà các ứng dụng hoặc hệ thống phát đi.

Có 2 cách phát-nhận đó là:

* Không có thứ tự: receiver nào đủ điều kiện thì nhận hết, không phân biệt và cũng tách rời nhau.
* Có thứ tự: receiver nào đăng ký ưu tiên hơn thì nhận trước, và có thể truyền thêm thông tin xử lý cho các receiver sau.

Thực ra lifecycle của BroadcastReceiver chỉ có duy nhất một phương thức onReceive().

* Khi có sự kiện mà BroadcastReceiver đã đăng ký nhận được phát đi, thì phương thức onReceive() của BroadcastReceiver đó sẽ được gọi.Sau khi thực thi xong phương thức này, lifercycle của Receiver kết thúc.Ngay khi onReceive() kết thúc, hệ thống coi như receiver đã không còn hoạt động và có thể kill process chứa receiver này bất cứ lúc nào.
* Vì Receiver không kế thừa từ Context nên cần truyền context mà receiver này đang chạy vào. Thứ nhất, để có thể xử lý các phương thức yêu cầu truyền thêm Context, thứ 2, để sử dụng các phương thức của lớp Context. (còn nữa hay không thì các bạn giúp mình luôn nhé)
* Intent được truyền vào sẽ có đầy đủ thông tin như sự kiện nào mà receiver này đăng ký đã xảy ra dẫn đến onReceive() được gọi. Có gửi kèm thông tin gì hoặc dữ liệu gì hay không. Xem các api: Intent.getAction() Intent.get…Extra(String dataName)

**1.4.5 Content Provider**

Content Provider là 1 trong 4 thành phần cơ bản của 1 ứng dụng Android thường có bao gồm:

1. Activity

2. Service

3. Broadcast Receiver

4. Content Provider

Một Content Provider cung cấp một tập chi tiết dữ liệu ứng dụng đến các ứng dụng khác. Thường được sử dụng khi chúng ta muốn tạo cơ sở dữ liệu dưới dạng public (các ứng dụng khác có thể truy xuất ). Dữ liệu thường được lưu trữ ở file hệ thống, hoặc trong một SQLite database. Đơn giản để các bạn có thể hình dung như :Danh bạ, Call log, cấu hình cài đặt...trên điện thoại là dữ liệu dưới dạng Content Provider.

Content Provider hiện thực một tập phương thức chuẩn mà các ứng dụng khác có thể truy xuất và lưu trữ dữ liệu của loại nó điều khiển.Tuy nhiên, những ứng dụng không thể gọi các phương thức trực tiếp. Hơn thế

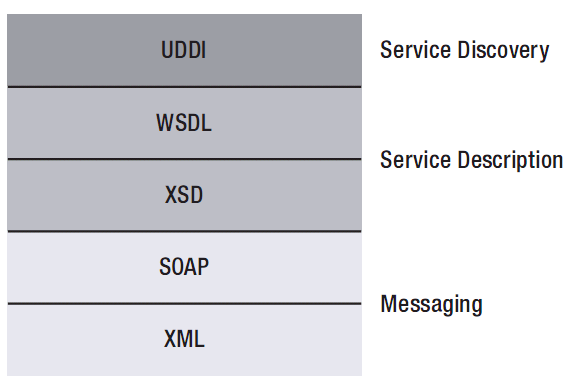
chúng dùng lớp Content Resolver và gọi những phương thức đó. Một Content Resolver có thể giao tiếp đến nhiều content provider; nó cộng tác với các provider để quản lý bất kỳ giao tiếp bên trong liên quan.

**Tổng quan về Web Service**

1. **Web Service** **là gì?**

Một Web service được định nghĩa là một tập các phương thức có thể được định vị thông qua địa chỉ URL, các phương thức này được công bố trên hệ thống mạng và được dùng như những khối cơ bản để xây dựng phân tán.Nó là tập hợp các phương thức có thể được các ứng dụng triệu gọi từ xa (RPC – Remote Procedure Call) để hình thành nên một hệ thống ứng dụng phân tán.

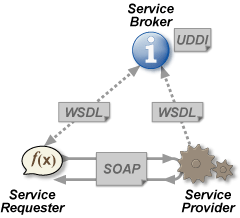
1. **Cấu trúc Web service**



Hình 2: Web service protocol stack

Tương tự với SOA, có 3 actor chính tham gia vào Web service.

* Service Provider: Dùng Web Services Description Language (WSDL) để mô tả dịch vụ mà mình có thể cung cấp cho Service Broker (tương tự với Service Registry trong SOA).
* Service Broker: Lưu trữ thông tin về các service được cung cấp bởi các Service Provider. Cung cấp chức năng tìm kiếm hỗ trợ Service Requester (Service Consumer trong SOA) trong việc xác định Service Provider phù hợp. Thành phần chính của Service Broker là Universal Discovery, Description, and Integration (UDDI) repositories.
* Service Requester: Dùng WSDL để đặc tả nhu cầu sử dụng (loại service, thời gian sử dụng, resource cần thiết, mức giá ...) và gởi cho Service Broker. Bằng việc sử dụng UDDI và chức năng tìm kiếm của Service Broker, Service Requester có thể tìm thấy Service Provider thích hợp. Ngay sau đó, giữa Service Requester và Service Provider thiết lập kênh giao tiếp sử dụng SOAP để thương lượng giá cả và các yếu tố khác trong việc sử dụng service.



Hình 3: Web service actors

## Simple Object Access Protocol – SOAP:

SOAP là một protocol giao tiếp dùng trong Web service được xây dựng dựa trên XML. SOAP được sử dụng để đặc tả và trao đổi thông tin về các cấu trúc dữ liệu cũng như các kiểu dữ liệu giữa các thành phần trong hệ thống.

Sử dụng SOAP, ứng dụng có thể yêu cầu thực thi method trên máy tính ở xa mà không cần quan tâm đến chi tiết về platform cũng như các phần mềm trên máy tính đó.

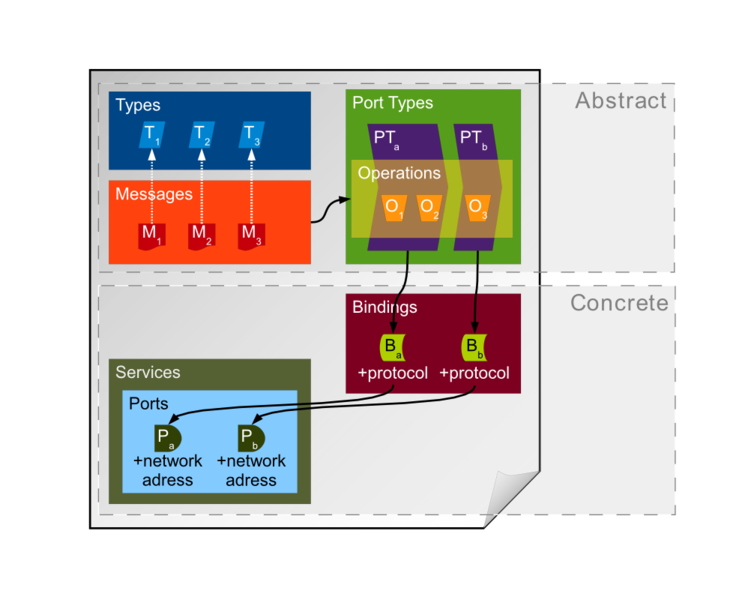
### Các đặc điểm của SOAP

* Khả năng mở rộng (Extensible): Cung cấp khả năng mở rộng phục vụ cho nhu cầu đặc thù của ứng dụng và nhà cung cấp. Các chức năng về bảo mật, tăng độ tin cậy có thể đưa vào phần mở rộng của SOAP. Các nhà cung cấp dịch vụ khác nhau, tùy vào đặc điểm hệ thống của mình có thể định nghĩa thêm các chức năng mở rộng nhằm tăng thêm lợi thế cạnh tranh cũng như cung cấp thêm tiện ích cho người sử dụng.
* Có thể hoạt động trên các network protocol đã được chuẩn hóa (HTTP, SMTP, FTP, TCP, ...)
* Độc lập với platform, ngôn ngữ lập trình hay programming model được sử dụng.

## Web Service Description Language – WSDL

Việc đặc tả chức năng, interface giao tiếp của một web service được thực hiện dựa vào Web Service Description Language (WSDL). Việc đặc tả này bao gồm protocol được sử dụng, cấu trúc và định dạng dữ liệu dùng để giao tiếp...

mô tả các thành phần cơ bản của một file WSDL dùng để đặc tả một web service.

* Services: Chứa các method có thể được sử dụng thông qua các web protocol.
* Ports: Địa chỉ dùng để kết nối đến web service. Thông thường, ports được mô tả bằng một HTTP URL. 

Hình 4: Overview of WSDL

* Port Types: định nghĩa một web service, các tác vụ mà service cung cấp và định dạng các thông điệp được sử dụng để khởi động các tác vụ này.
* Operations: Mỗi operation có thể được xem như một method hay một lời gọi hàm trong các ngôn ngữ lập trình cổ điển.
* Binding: chỉ định port type, các operation, SOAP binding stype (RPC/Document), SOAP protocol được dùng.
* Message: Mỗi message tương ứng với một operation và chứa các thông tin cần thiết để thực thi operation đó. Mỗi message có một name duy nhất và một hay nhiều logical part. Các logical part được phân biệt với nhau qua name và có thể lưu trữ các tham số cần cho operation.
* Element: Được định nghĩa trong Types. Mỗi element có một name duy nhất và kiểu dữ liệu. Element được dùng để đặc tả dữ liệu dùng trong message. Element có thể đặc tả các dữ liệu đơn giản (string, integer) hay phức tạp hơn như array, struct, ...
* XSD file: Các element thường được định nghĩa trong các XML Schema Definition (XSD) file. XSD file có thể ở trong cùng file WSDL hoặc ở file riêng biệt.

1. **Ưu điểm của Web service**

Web service có nhiều ưu điểm vượt trội so với các mô hình ứng dụng phân tán ra đời trước đó.Chính những ưu điểm này làm cho web service ngày càng được phổ biến và ứng dụng trong nhiều lĩnh vực khác nhau.

* **Khả năng vượt Firewall**: web service hoạt động trên nền giao thức HTTP và sữ dụng port 80 để giao tiếp. Đây là một lợi thế vượt trội so với các mô hình ứng dụng phân tán (RPC) trước đây. Hầu hết các Firewall hiện nay đều cho phép đi qua port “well-known” 80 nên web service có thể hoạt động bất kì nơi nào.
* **Hoạt động trên đa môi trường:** web service dựa trên công nghệ XML, mà được hỗ trợ ở bất kì hệ điều hành nào, ngay cả các thiết bị di động. Điều đó giúp web service hoạt động ở bất kì môi trường, hầu hết các ngôn ngữ lập trình.
* **Tính linh hoạt:** Web service chỉ đơn thuần dạng text không có những tag định nghĩa như HTML. Khi cần thay đổi các chức năng của web service đó đơn thuần là thêm hay bớt các hàm, chúng ta không phải xây dựng lại giao diện phức tạp như với trang web. Việc truyền các dữ liệu định dạng đi làm giảm khá nhiều băng thông trên kênh truyền.

**jQueryframework**

Với sự phát triển của Internet, người dùng ngày càng quan tâm hơn đến hình thức của một trang web. Trước đây một trang web chỉ cần có banner, nội dung và ít footer hời hợt là đã được cho là một trang web hoàn chỉnh. Nhưng bây giờ trang web đó phải có banner bắt mắt, nội dung hay và còn nhiều hiệu ứng lạ mắt khác nữa thì mới có thể thu hút được người đọc.

Chính vì thế những web designer bắt đầu chú ý đến các thư viện JavaScript mở như jQuery để tạo ra các hiệu ứng có thể tương tác trực tiếp với người đọc một cách nhanh chóng và dễ dàng hơn rất nhiều là sử dụng thuần JavaScript.

Nhưng nếu bạn là người mới làm quen với jQuery bạn sẽ thấy không biết phải bắt đầu từ đâu vì jQuery cũng giống như bất cứ thư viện nào khác cũng có rất nhiều functions. Cho dù bạn có đọc phần tài liệu hướng dẫn sử dụng của jQuery thì bạn vẫn thấy rất phức tạp và khó hiểu. Nhưng bạn yên tâm một điều là jQuery có cấu trúc rất mạch lạc và theo hệ thống. Cách viết code của jQuery được vay mượn từ các nguồn mà các web designer đa phần đã biết như HTML và CSS. Nếu từ trước đến nay bạn chỉ là Designer chứ không phải coder, bạn cũng có thể dễ dàng học jQuery vì kiến thức về CSS giúp bạn rất nhiều khi bắt đầu với jQuery.

**Những thứ Jquery có thể làm**

**Hướng tới các thành phần trong tài liệu HTML**. Nếu không sử dụng thư viện JavaScript này, bạn phải viết rất nhiều dòng code mới có thể đạt được mục tiêu là di chuyển trong cấu trúc cây (hay còn gọi là DOM = Document Object Model) của một tài liệu HTML và chọn ra các thành phần liên quan. Jquery cho phép bạn chọn bất cứ thành phần nào để thao tác một cách dễ dàng nhất.

**Thay đổi giao diện của một trang web.** CSS là công cụ rất mạnh để định dạng một trang web nhưng nó có một nhược điểm là không phải tất cả các trình duyệt đều hiển thị giống nhau. Cho nên jQuery ra đời để lấp chỗ trống này, vì vậy các bạn có thể sử dụng nó để giúp trang web có thể hiển thị tốt trên hầu hết các trình duyệt. Hơn nữa jQuery cũng có thể thay đổi class hoặc những định dạng CSS đã được áp dụng lên bất cứ thành phần nào của tài liệu HTML ngay cả khi trang web đó đã được trình duyệt load thành công.

**Tương tác với người dùng.** Với thư viện javaScript như jQuery, nó cho bạn nhiều cách để tương tác với người dùng ví dụ như khi người dùng nhấp chuột vào đường link thì sẽ có gì xảy ra. Nhưng cái hay của nó là không làm cho code HTML của bạn rối tung lên chính là nhờ các Event Handlers. Hơn nữa Event Handler API sẽ bảo đảm rằng trang web của bạn tương thích hầu hết với các trình duyệt, điều này đã và đang làm đau đầu rất nhiều các web designer.

**Tạo hiệu ứng động cho những thay đổi của tài liệu.** Để tương tác tốt với người dùng, các web designer phải cho người dùng thấy được hiệu ứng gì sẽ xảy ra khi họ làm một tác vụ nào đó. Jquery cho phép bạn sử dụng rất nhiều hiệu ứng động như mờ dần, chạy dọc chạy ngang v.v.. và nếu vẫn chưa đủ, nó còn cho phép bạn tự tạo ra các hiệu ứng của riêng mình.

**Lấy thông tin từ server mà không cần tải lại trang web.** Đây chính là công nghệ ngày càng trở nên phổ biến Asynchronous JavaScript And XML (AJAX), nó giúp người thiết kế web tạo ra những trang web tương tác cực tốt và nhiều tính năng. Thư viện jQuery loại bỏ sự phức tạp của trình duyệt trong quá trình này và cho phép người phát triển web có thể tập trung vào các tính năng đầu cuối.

**Sơ lược jquery framework**

**Tạo một trang web nhúng jquery framework**

|  |
| --- |
| <html>  <head>  <script type="text/javascript" src="path/to/jquery.js"></script>  <script type="text/javascript">  // Your code goes here  </script>  </head>  <body>  <a href="http://jquery.com/">jQuery</a>  </body>  </html> |

Sửa thuộc tính src trong thẻ script để trỏ tới jquery.js Ví dụ, nếu jquery.js ở cùng thư mục với tệp HTML, bạn có thể dùng:  
  
 <script type="text/javascript" src="jquery.js"></script>

**Chạy mã lúc Document Ready**   
Có thể thấy rằng hầu hết các lập trình viên Javascript kết thúc mã thực thi bằng cách thêm các dòng mã vào script, tương tự như sau:

|  |
| --- |
| window.onload = function(){ ... } |

Các mã bên trong đó sẽ được thực thi khi trang được tải xong. Không còn gì phải bàn, tuy nhiên, mã Javascript không chạy cho đến khi tất cả các hình ảnh trong trang được tải xong (kể cả banner quảng cáo). Lý do cho việc sử dụng window.onload là để chắc chắn rằng HTML 'document' phải được tải xuống hết, sau đó mới chạy mã kia.  
Để giải quyết, jQuery có câu lệnh đơn giản, được biết như ready event:

|  |
| --- |
| $(document).ready(function(){    // Mã của bạn  }); |

Mã này kiểm tra document và đợi cho đến khi nó sẵn sàng để sử dụng - theo ý bạn muốn. Hãy lấy đoạn mã mẫu ở trên và cho vào tài liệu HTML của bạn. Các ví dụ còn lại của jQuery sẽ được đặt bên trong phần callback của hàm này, chúng sẽ được thực thi khi tài liệu sẵn sàng.   
**Sự kiện Click**  
Đầu tiên, chúng ta sẽ thử thực hiện động tác click với clicked. Bên trong ready function (từ ví dụ ở trên), thêm mã như sau:

|  |
| --- |
| $("a").click(function(){    alert("Cảm ơn đã ghé thăm!");  }); |

Lưu tệp HTML và tải lại trang thử nghiệm trên trình duyệt (Ctrl+F5). Bấm vào link nào đó trên trang và trình duyệt sẽ hiện ra một thông báo pop-up.  
Đối với sự kiện click và hầu hết các sự kiện khác events, bạn có thể can thiệp vào hành động mặc định - ở đây, sau đây là link dẫn tới jquery.com - bằng cách trả về giá trị false từ điều khiển sự kiện:

|  |
| --- |
| $("a").click(function(){    alert("Cảm ơn đã ghé thăm!");    return false;  }); |

Nếu không dùng các đoạn mã trên, nếu bạn nhấn link tới jquery.com trong trang, thì trình duyệt sẽ duyệt tới. Nhưng ở đây, trình duyệt sẽ hiện thông báo và không chuyển đi đâu cả.   
  
**Thêm một Class (lớp với CSS)**  
Đây là tác vụ dùng để thêm vào (hoặc bỏ đi) các class từ phần tử nào đó, ví dụ:

|  |
| --- |
| $("a").addClass("test"); |

Nếu bạn đặt đoạn mã trên vào phần script trong header của trang HTML, thì mã tự động sinh ra sẽ là:

|  |
| --- |
| <style type="text/css">a.test { font-weight: bold; }  </style> |

và sau đó addClass được gọi - tất cả những phần từ A sẽ được làm đậm. Để bỏ class, bạn có thể sử dụng removeClass   
**Các hiệu ứng khác**  
Trong jQuery, có cung cấp các thành phần effects, để làm cho website của bạn thật sự nổi bật. Để chạy thử, hãy thay đoạn mã như sau:

|  |
| --- |
| $("a").click(function(){    $(this).hide("slow");    return false;  }); |

Bây giờ, nếu bạn nhấn vào liên kết có trong trang, bạn sẽ thấy nó biến mất một cách từ từ.  
  
**Khả năng liên kết thành chuỗi**  
jQuery cung cấp cho bạn một cách viết mã đơn giản và ngắn gọn. Nó tương tự như cách thức lập trình hướng đối tượng (object-oriented programming), nói đúng hơn đây là cách viết thẳng một hàng.  
Tóm lại: Mỗi phương thức trong jQuery trả về câu truy vấn đối tượng của chính nó, nó cho phép bạn 'xích' chúng lại với nhau, ví dụ:

|  |
| --- |
| $("a").addClass("test").show().html("foo"); |

Câu lệnh này chứa những phương thức cá thể (addClass, show, và html) trả về đối tượng jQuery, nó cho phép bạn có thể tiếp dụng sử dụng phương thức để đặt cho phần tử hiện tại.  
Bạn có thể dùng xa hơn nữa, bằng cách thêm hoặc bỏ phần tử từ vùng chọn, thay đổi các phần tử này và sau đó trả giá trị lại cho vùng chọn trước, ví dụ:

|  |
| --- |
| $("a")    .filter(".clickme")      .click(function(){        alert("You are now leaving the site.");      })    .end()    .filter(".hideme")      .click(function(){        $(this).hide();        return false;      })    .end(); |

**Hàm hồi quy, Hàm, và “this”**  
Hàm hồi quy là một hàm được truyền như là một thông số cho một hàm khác (hàm gọi/hàm bao ngoài) và sẽ được thực thi sau khi hàm gọi hoàn tất. Điều đáng lưu ý về hàm hồi quy là tất cả những hàm xuất hiện sau "hàm gọi" có thể thực thi trước hàm hồi quy.  
Một điều quan trọng khác cần lưu ý là làm thế nào dùng đúng hàm hồi quy. Đây là điều mà tôi thường hay quên cú pháp câu lệnh.  
Đối với một hàm hồi quy không thông số, bạn có thể gọi theo cách sau:

|  |
| --- |
| $.get('myhtmlpage.html', myCallBack); |

Lưu ý thông số thứ hai chỉ đơn giản là tên hàm (không truyền như một giá trị chuỗi ký tự và không có dấu ngoặc đơn). Các hàm trong Javascript được xem là những 'khách hàng hạng sang' và vì vậy có thể truyền như một biến số và sẽ được thực thi sau đó.

**PHÂN TÍCH VÀ ĐÁNH GIÁ MỘT SỐ ỨNG DỤNG THƯƠNG MẠI ĐIỆN TỬ**

Hiện nay, các website TMĐT tại Việt Nam đã phát triển khá đa dạng về hình thức và chất lượng ngày càng được nâng cao.Cộng thêm nhiều dịch vụ thanh toán trực tuyến ra đời đã giúp cho người tiêu dùng có thể thanh toán cho các sản phẩm, dịch vụ mà mình đã chọn thông qua hệ thống mạng Internet dễ dàng và tiện lợi.Sự thành công của những website TMĐT đã dần dần làm thay đổi hình thức tiêu dùng của người sử dụng và xu thế phát triễn kinh doanh mới cho các doanh nghiệp.

Áp dụng TMĐT đòi hỏi các doanh nghiệp phải tự thay đổi phương thức kinh doanh của mình, và thay đổi cơ sở hạ tầng để đáp ứng tốt cho các ứng dụng TMĐT.Nó là một thách thức và cũng là động lực cho các doanh nghiệp phát triển trong thời kỳ hội nhập toàn cầu.

Wesite thương mại điện tử Nhommua

**Object IDXu**

Là object thông tin về giá trị thẻ nạp tiền, số cào của thẻ tiền sẻ được phát hành với các mệnh giá khác nhau. Object IDXu gồm các thông tin sau:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| STT | Tên | Kiểu | Diễn giải | Chú ý |
| 1 | id | String | Mã số ID Xu | Khóa chính |
| 2 | money | Long | Mệnh giá Xu |  |
| 3 | flag | String | Trạng thái |  |

Bảng IdXu

**Object PhoneXu**

Là object thông tin về tài khoản nạp tiền bằng điện thoại. Object PhoneXu gồm các thông tin sau:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| STT | Tên | Kiểu | Diễn giải | Chú ý |
| 1 | Phone | String | Số điện thoại nạp tiền | Khóa chính |
| 2 | money | Long | Mệnh giá Xu |  |

Bảng PhoneXu

**Object XuHistory**

Là object thông tin về lịch sử nạp tiền của người dùng. Object XuHistory gồm các thông tin sau:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| STT | Tên | Kiểu | Diễn giải | Chú ý |
| 1 | id | Long | Mã số XuHistory | Khóa chính |
| 2 | IdXu | String | IdXu nạp tiền |  |
| 3 | Date | String | Thời gian nạp tiền |  |
| 4 | Username | String | Tài khoản người dùng được nạp |  |

Bảng XuHistory

**Object PhoneHistory**

Là object thông tin về lịch sử nạp tiền vào số điện thoại. Object PhoneHistory gồm các thông tin sau:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| STT | Tên | Kiểu | Diễn giải | Chú ý |
| 1 | id | Long | Mã số PhoneHistory | Khóa chính |
| 2 | Phone | String | Mệnh giá Xu |  |
| 3 | Date | String | Thời gian nạp tiền |  |

Bảng PhoneHistory

**Object Adress**

Là oibject thông tin về địa chỉ khách hàng khi đăng ký tại website. Object Address gồm các thông tin sau:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| STT | Tên | Kiểu | Diễn giải | Chú ý |
| 1 | buildingName | String | Mã số PhoneHistory | Khóa chính |
| 2 | districtName | String | Mệnh giá Xu |  |
| 3 | homeNumber | String | Thời gian nạp tiền |  |
| 4 | streetName | String |  |  |
| 5 | wardName | String |  |  |

Bảng Adress

**Object Customer**

Là object thông tin chi tiết về khách hàng khi đang ký tại website. Object custiomer gồm các thông tin sau:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| STT | Tên | Kiểu | Diễn giải | Chú ý |
| 1 | customerId | String | Mã số PhoneHistory | Khóa chính |
| 2 | customerPassword | String | Mệnh giá Xu |  |
| 3 | customerName | String | Thời gian nạp tiền |  |
| 4 | streetName | String |  |  |
| 5 | customerBirth | String |  |  |
| 6 | customerEmail | String |  |  |
| 7 | customerPhone | String |  |  |
| 8 | customerAddress | Address |  |  |
| 9 | customerType | String |  |  |
| 10 | customerMark | Integer |  |  |
| 11 | isPay | boolean |  |  |
| 12 | accountOnline | Long |  |  |
| 13 | payOnlineType | String |  |  |
| 14 | isAdEmail | boolean |  |  |
| 15 | isAdPhone | boolean |  |  |
| 16 | isActive | boolean |  |  |
| 17 | xuOnline | Long |  |  |

Bảng Customer

**Object Provider**

Là object thông tin về các cửa hàng bán sản phẩm trên website. Object Provider gồm các thông tin sau:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| STT | Tên | Kiểu | Diễn giải | Chú ý |
| 1 | providerId | String | Mã số PhoneHistory | Khóa chính |
| 2 | providerPassword | String | Mệnh giá Xu |  |
| 3 | providerName | String | Thời gian nạp tiền |  |
| 4 | providerGender | String |  |  |
| 5 | providerBirth | date |  |  |
| 6 | providerEmail | String |  |  |
| 7 | providerPhone | String |  |  |
| 8 | providerAddress | Address |  |  |
| 9 | providerAccountPayment | String |  |  |
| 10 | xuOnline | long |  |  |
| 11 | authentication | boolean |  |  |
| 12 | loginDate | date |  |  |

Bảng Provider

**Object DetailOrder**

Là object thông tin về chi tiết hóa đơn đặt hàng. Object DetailOrder gồm các thông tin sau:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| STT | Tên | Kiểu | Diễn giải | Chú ý |
| 1 | id | Long | Mã số PhoneHistory | Khóa chính |
| 2 | idProduct | Liong | Mệnh giá Xu |  |
| 3 | name | String | Thời gian nạp tiền |  |
| 4 | number | Integer |  |  |
| 5 | subPrice | long |  |  |
| 6 | orderId | long |  |  |

Bảng DetailOrder

**Object OrderBill**

Là object thông tin về hóa đơn đặt hàng trên website. Object OrderBill gồm các thông tin sau:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| STT | Tên | Kiểu | Diễn giải | Chú ý |
| 1 | id | Long | Mã số PhoneHistory | Khóa chính |
| 2 | idProduct | Long | Mệnh giá Xu |  |
| 3 | address | Address | Thời gian nạp tiền |  |
| 4 | email | String |  |  |
| 5 | phone | String |  |  |
| 6 | status | Integer |  |  |
| 7 | dateOrder | date |  |  |
| 8 | dateShip | date |  |  |
| 9 | sumPrice | long |  |  |
| 10 | idEmployee | String |  |  |

Bảng OrderBill

**ObjectFood**

Là object thông tin về chi tiết mặt hàng bán trên website. Object Food gồm các thông tin sau:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| STT | Tên | Kiểu | Diễn giải | Chú ý |
| 1 | foodId | Long | Mã số PhoneHistory | Khóa chính |
| 2 | foodName | String | Mệnh giá Xu |  |
| 3 | foodPriceLevelId | String | Thời gian nạp tiền |  |
| 4 | productAttributeId | String |  |  |
| 5 | foodStatusId | String |  |  |
| 6 | foodTypeId | String |  |  |
| 7 | startDate | date |  |  |
| 8 | expDate | Date |  |  |
| 9 | number | Integer |  |  |
| 10 | numberOrder | Integer |  |  |
| 11 | detail | String |  |  |
| 12 | cooking | String |  |  |
| 13 | calo | String |  |  |
| 14 | price | Long |  |  |
| 15 | pricePromotion | Long |  |  |
| 16 | providerID | String |  |  |
| 17 | url | String |  |  |
| 18 | reviewer | Integer |  |  |
| 19 | uploadDate | Integer |  |  |
| 20 | isDisplay | Integer |  |  |

Bảng Food

**ObjectFoodAttribute**

Là object phân loại hàng hóa bán trên website, phân loại theo món ăn hoặc hình thức cửa hàng. Object FoodAttribute bao gồm thông tin sau:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| STT | Tên | Kiểu | Diễn giải | Chú ý |
| 1 | productAttributeId | String | Mã số PhoneHistory | Khóa chính |
| 2 | productAttributeName | String | Mệnh giá Xu |  |

Bảng FoodAttribute

**Object FoodPriceLevel**

Là object thông tin về các giá tiền khi bán và các kiểu định khoảng giá. Object FoodPriceLevel gồm các thông sau:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| STT | Tên | Kiểu | Diễn giải | Chú ý |
| 1 | foodPriceLevelId | String | Mã số PhoneHistory | Khóa chính |
| 2 | startPrice | Long | Mệnh giá Xu |  |
| 3 | endPrice | long |  |  |

Bảng FoodPriceLevel

**Object FoodStatus**

Là object thông tin về trạng thái món hàng đang bán. Object FoodStatus gồm các thông sau:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| STT | Tên | Kiểu | Diễn giải | Chú ý |
| 1 | foodStatusId | String | Mã số PhoneHistory | Khóa chính |
| 2 | foodStatusName | String | Mệnh giá Xu |  |

Bảng FoodStatus

**Object FoodType**

Là object thông tin về các loại thực phẩm bán trên website. Object FoodType gồm các thông sau:

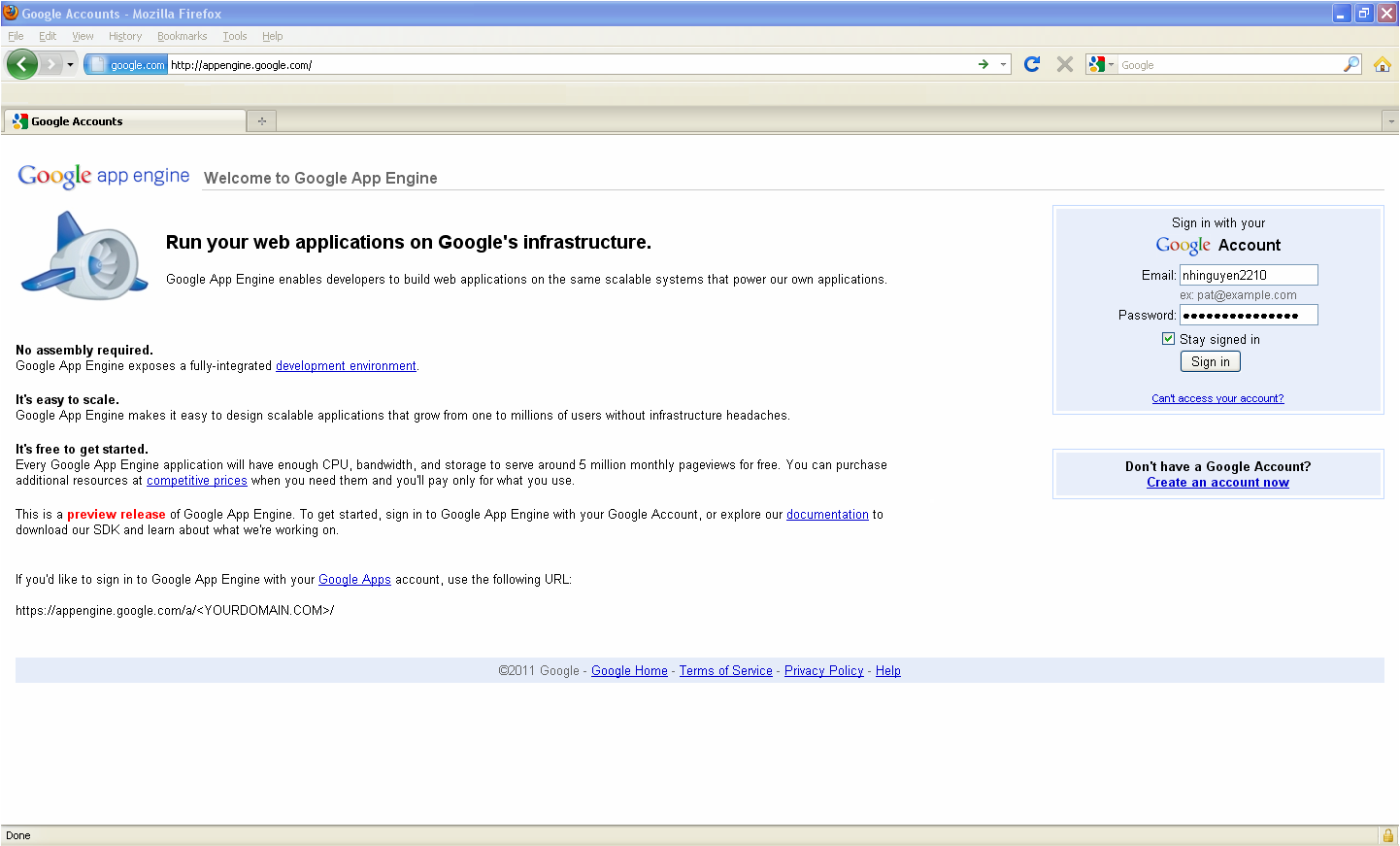
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| STT | Tên | Kiểu | Diễn giải | Chú ý |
| 1 | productTypeId | String | Mã số PhoneHistory | Khóa chính |
| 2 | productTypeName | String | Mệnh giá Xu |  |

Bảng FoodType

**Phụ lục A**

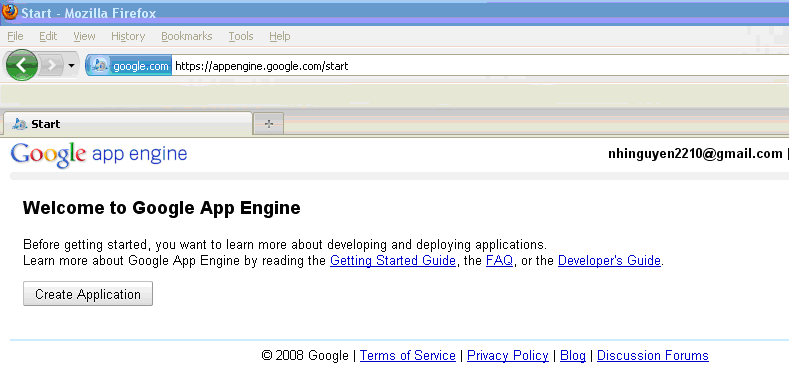
**ĐĂNG KÝ GOOGLE APP ENGINE**

Để có thể sử dụng các dịch vụ do GAE cung cấp, chúng ta cần đăng ký một tài khoản Google, hoặc chúng ta có thể sử dụng tài khoản Google ( hay Gmail, hoặc các ứng dụng khác…). Sau khi đã có tài khoản, vào trang [http://appengine.google.com](http://appengine.google.com/) để đăng nhập.



Hình A1: Màn hình đăng nhập hệ thống GAE

Với lần đầu tiên đăng nhập vào hệ thống App Engine của Google, chúng ta sẽ được yêu cầu tạo ứng dụng đầu tiên.

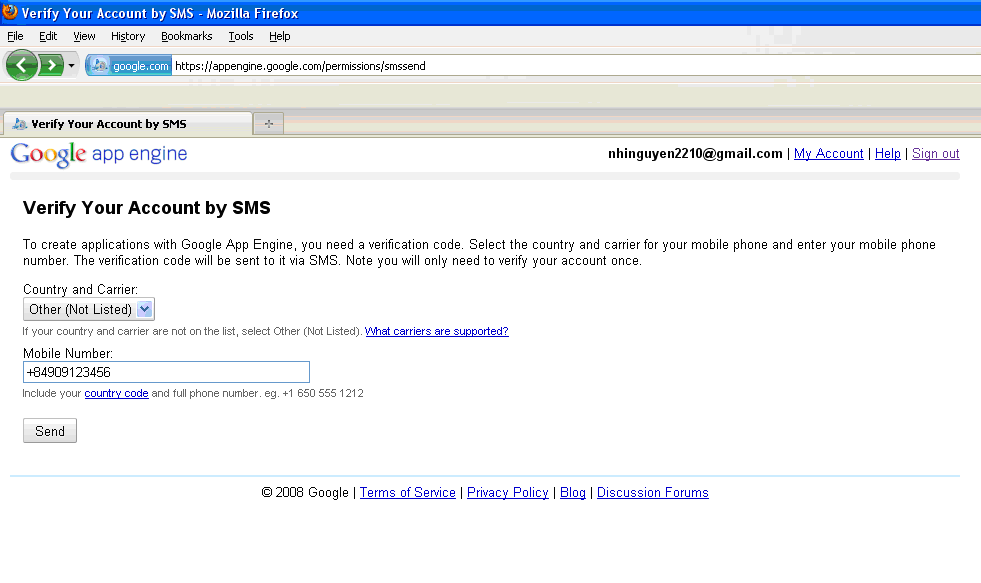


Hình A2: Màn hình chào mừng lần đầu tiên tạo ứng dụng GAE.

Nhấn vào nút Create Application. Chuyển sang trang xác thực tài khoản.

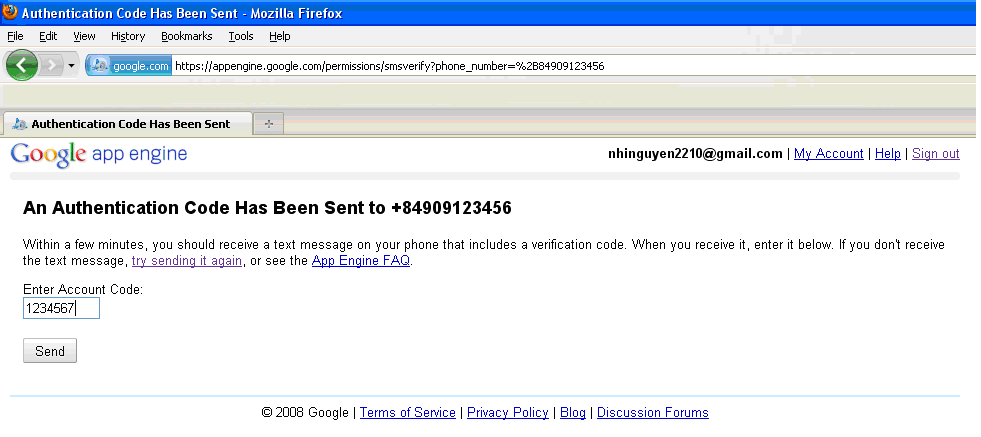
Chọn Country and Carrier là Other (Not Listed).

Nhập số điện thoại chúng ta muốn xác thực. Lưu ý: số điện thoại phải ghi rõ mã quốc gia. Việt Nam là +84.



Hình A3: Màn hình nhập số điện thoại để xác thực.

Nhấn nút Send. Google sẽ gửi mã kích hoạt đến số điện thoại mà chúng ta đăng ký.

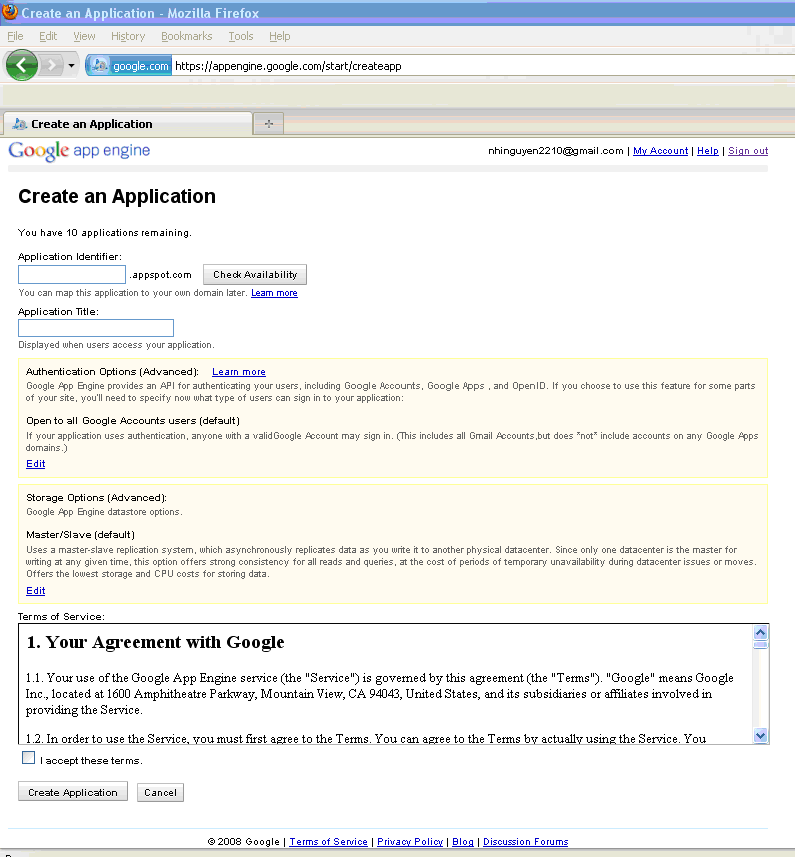


Hình A4: Màn hình xác thực tài khoản qua điện thoại.

Điền đoạn mã mà chúng ta nhận được vào khung text, nhấn Send, hoàn tất việc xác thực.

Lưu ý: bước xác thực này chỉ đòi hỏi trong quá trình tạo ứng dụng đầu tiên, các ứng dụng tiếp theo sẽ bỏ qua bước này.

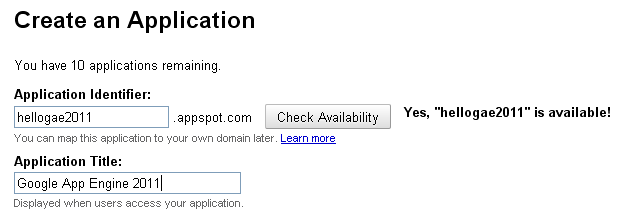
Chuyển đến trang khởi tạo ứng dụng.



Hình A5: Màn hình tạo ứng dụng mới

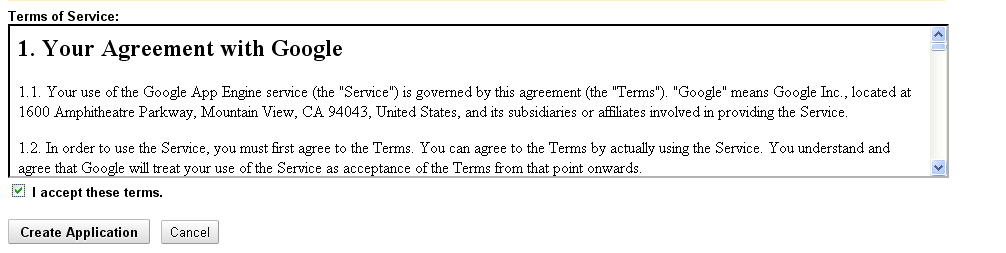
* Application Identifier: Như các chúng ta đã biết, GAE là một PaaS (Platform as a Service) – dịch vụ nền tảng, có nghĩa là chúng ta có thể phát triển và deploy ứng dụng của chúng ta lên hạ tầng của Google (trên các đám mây). Do đó các ứng dụng của chúng ta phải có tên để phân biệt với các ứng dụng khác trên đó. Mặc định, ứng dụng của chúng ta sẽ được truy cập bằng đường dẫn: APP\_ID.appspot.com. Với APP\_ID là bất kì tên nào mà chúng ta muốn gắn cho ứng dụng và tên này không được trùng với tên ứng dụng đã có khác.
* Application Title: tiêu đề trang web.

Điền Application Identifier, Application Title, nhấn nút Check Availability để kiểm tra tính hợp lệ của Application Identifier.



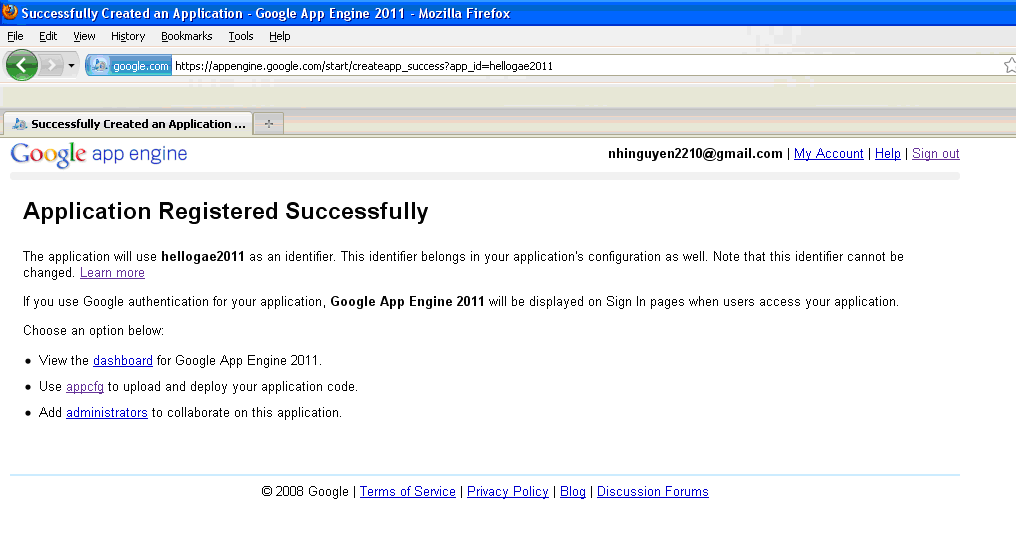
Hình A6: Nhập Application Identifier và Application Title

Check đồng ý các điều khoản của GAE



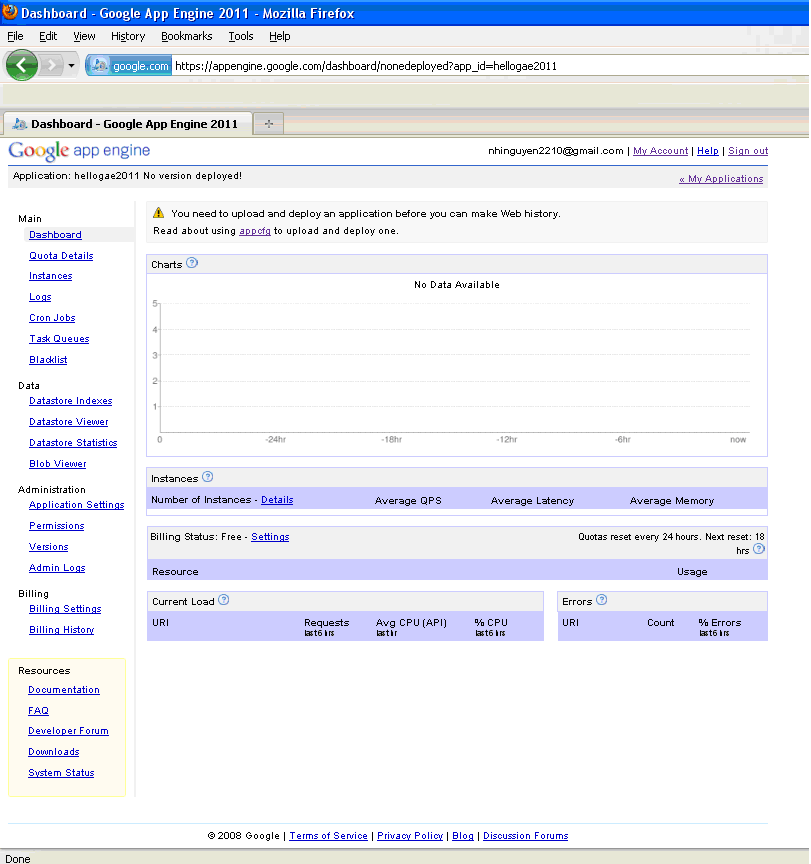
Hình A7: Đồng ý các điều khoản của GAE

Nhấn Create Application để tạo.



Hình A8: Màn hình tạo thành công ứng dụng.

Đây là màn hình quản lý ứng dụng:



Hình A9: Màn hình quản lý ứng dụng của GAE.

Ứng dụng của chúng ta đã được tạo thành công, lần sau đăng nhập vào tài khoản, các ứng dụng đã có sẽ được liệt kê như sau:



Hình A10: Màn hình liệt kê các ứng dụng trên GAE

GAE hiện chỉ cho phép chúng ta deploy 10 ứng dụng lên hệ thống, cho đến nay chưa có chức năng xóa ứng dụng, vì thế chúng ta nên cân nhắc trước khi deploy.

# PHỤ LỤC B

# HƯỚNG DẪN CÀI ĐẶT MÔI TRƯỜNG LẬP TRÌNH GAE CHO JAVA

**B1. Các thành phần cần thiết:**

* Java Development Kit 1.6.x.
* Eclipse 3.3/3.4/3.5.
* GAEJ Plug-in:

vào trang <http://code.google.com/eclipse/docs/getting_started.html> lấy link cài đặt plugin cho eclipse, lựa chọn đúng phiên bản eclipse của chúng ta.

* + Eclipse (Galileo): <http://dl.google.com/eclipse/plugin/3.5>
  + Eclipse (Ganymede): <http://dl.google.com/eclipse/plugin/3.4>
  + Eclipse (Europa): <http://dl.google.com/eclipse/plugin/3.3>

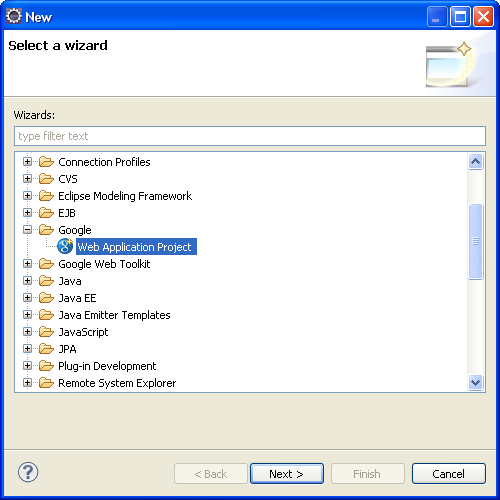
Sau khi đã cài đặt và khởi động lại Eclipse IDE, chúng ta sẽ thấy 4 icons của GAE trên thanh toolbar: .

**B2. Tạo project GAE**

Sau khi đăng ký Application Id và cài đặt GAEJ plug-in cho Eclipse. Chúng ta bắt tay vào viết ứng dụng “Hello World” theo các bước sau:

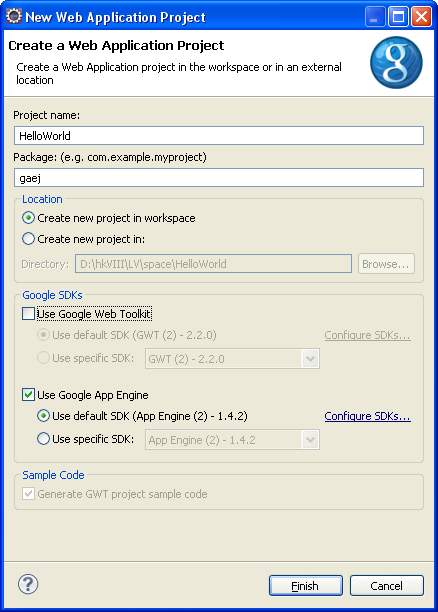
- Bước 1: Khởi động Eclipse.

- Bước 2: Vào menu File 🡪 New 🡪 Other hoặc nhấn Ctrl-N để tạo project. Chọn Google 🡪 Web Application project. Hoặc có thể nhấn vào icon Web Application Project trên thanh toolbar : . Nhấn Next.



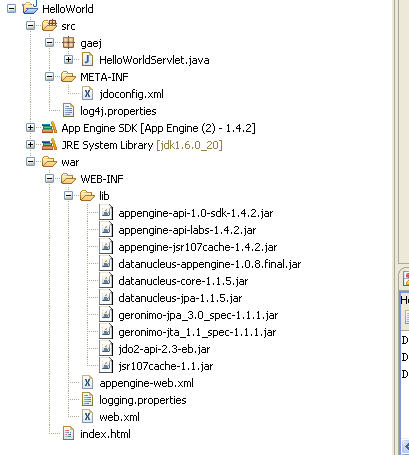
Hình B1: Cửa sổ tạo project GAE mới.

- Bước 3: điền Project name là tên project chúng ta tạo, tên package của project, và bỏ chọn Use Google Web Toolkit, nhấn Finish để tạo project.



Hình B2: Khung thiết lập thông số cho Project GAE mới.

**B3. Cấu trúc project GAE**



Hình B3: cấu trúc project GAEJ

* **src**: chứa các file mã nguồn. File HelloWorldServlet.java được khởi tạo mặc định cho chúng ta.
* **war**: các file trong thư mục này sẽ được deploy lên Servlet Container của App Engine. Trong thư mục war, có các thành phần sau:
* **WEB-INF\lib**: chứa các file JAR thư viện cần thiết.
* **index.html**: trang index.html được tạo mặc định, cũng là trang chủ khi chạy ứng dụng. Chúng ta có thể chỉnh sửa trang này.
* **logging.properties**: nếu đã quen với Java logging, chúng ta sẽ hiểu ý nghĩa của file này, là ghi log lại quá trình làm việc của ứng dụng.
* **web-xml**: đây là một file cấu hình ứng dụng web chuẩn. chúng ta có thể tìm thấy các entry định nghĩa Servlet ở đây.
* **appengine-web.xml**: đây là file cấu hình đặc trưng của GAE. Và phần tử chính quan trọng trong file chính là Application ID của ứng dụng mà ta đã đăng ký trên Google và phiên bản của nó.

**B4. Chạy ứng dụng tại máy local**

GAEJ chạy trên cổng 8888. Do đó hãy chắc rằng cổng này không được sử dụng bởi các tài nguyên khác.

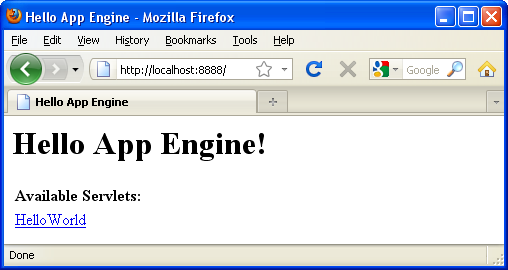
Để chạy ứng dụng, chúng ta thực hiện các bước sau:

Bước 1: Nhấp chuột phải vào ứng dụng, chọn Run As 🡪 Web application. Ứng dụng sẽ được biên dịch tại server local, và server sẽ được khởi động trên port 8888. Chúng ta có thể thấy thông báo như sau trong màn hình console:



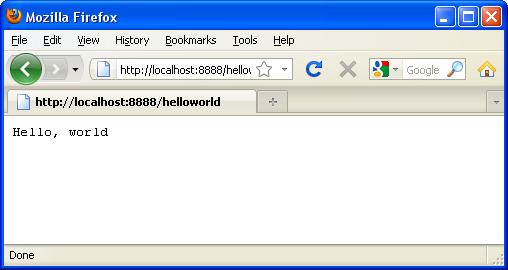
Hình B4: màn hình console khi chạy ứng dụng tại local

Bước 2: mở trình duyệt web, truy cập địa chỉ [http://localhost:8888](http://localhost:8888/). Hệ thống sẽ chuyển đến trang mặc định của ứng dụng, chính là index.html



Hình B5: Màn hình chạy ứng dụng tại local (1).

Nhấn vào link HelloWorld, chuyển đến trang servlet HelloWorldServlet.java:

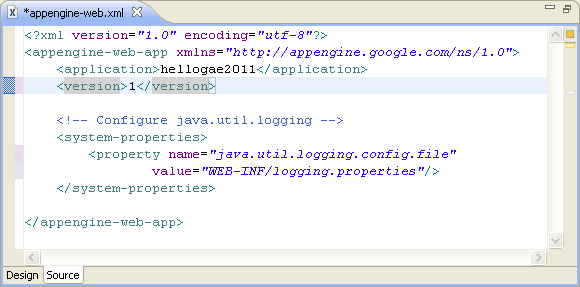


Hình B6: Màn hình chạy ứng dụng tại local (2).

**B5. Deploy ứng dụng lên đám mây của google**

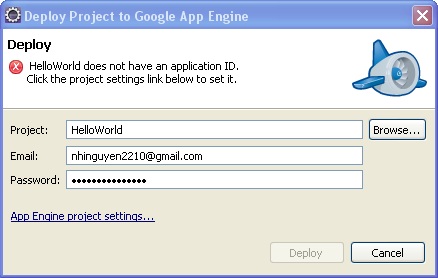
Để triển khai ứng dụng lên trên đám mây, thực hiện các bước sau:

Bước 1: Mở file cấu hình appengine-web.xml, nhập Application Identifier mà ta đã đăng ký trên GAE và version ứng dụng:



Hình B7: file cấu hình appengine-web.xml

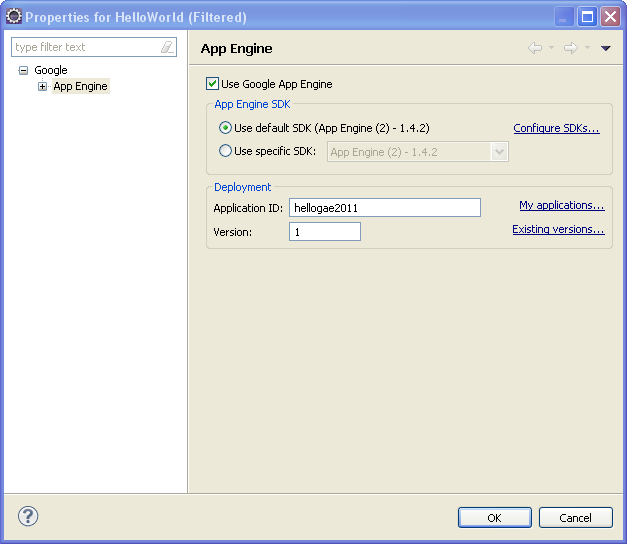
Bước 2: Nhấp phải vào project, chọn Google 🡪 Deploy to App Engine, hoặc click vào icon Deploy App Engine Project trên thanh toolbar : .



Hình B8: Màn hình deploy ứng dụng lên Google.

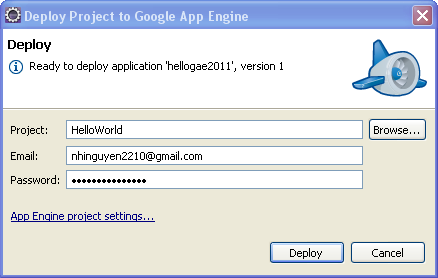
Bước 3: điền email và password của tài khoản Google.

Bước 4: Nếu Application Identifier và version của ứng dụng chưa được cấu hình trong file appengine-web.xml, hoặc bạn muốn thay đổi các thiết lập, trong cửa sổ deploy, chúng ta có thể cấu hình bằng cách nhấn vào liên kết App Engine project settings… Đây chính là Appication ID mà bạn đã đăng ký trên GAE. Version để mặc định hoặc chỉnh tùy ý.



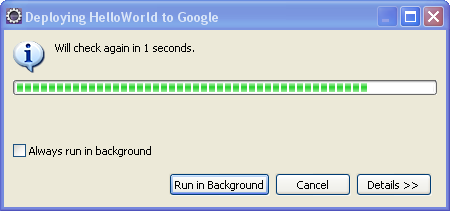
Hình B9: thiết lập Application ID cho ứng dụng

Bước 5: Nhấn Ok.



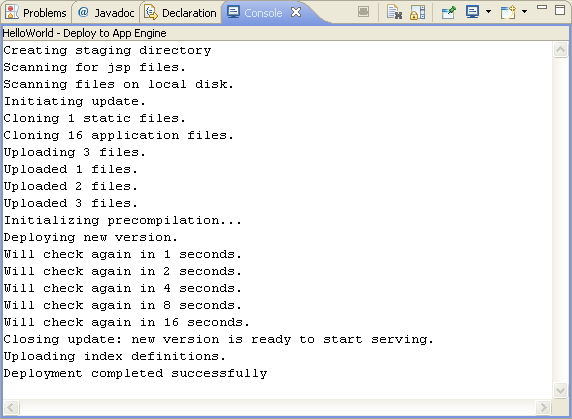
Hình B10: Màn hình deploy ứng dụng lên Google.

Bước 5: Nhấn Deploy, ứng dụng sẽ được deploy lên hệ thống GAE với Application Identifier mà bạn đã đăng ký trước đó.



Hình B11: Quá trình deploy ứng dụng lên Google

Màn hình console xuất hiện các dòng thông báo, và kết thúc là thông điệp báo việc triển khai ứng dụng lên hệ thống đã thành công.



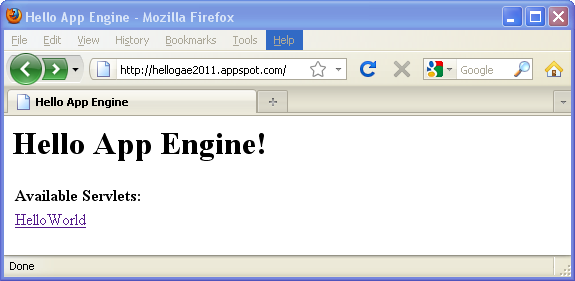
Hình B12: Màn hình console ghi log lại quá trình triển khai ứng dụng.

Ứng dụng của chúng ta đã được host lên GAE.

Chúng ta có thể deploy ứng dụng nhiều lần để update khi có thay đổi. Số lần thay đổi có giới hạn trong 1 ngày, nhưng cho đến thời điểm này, thì con số đó khá lớn cho chúng ta thoải mái chỉnh sửa.

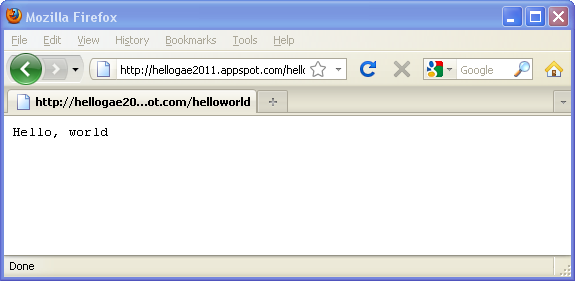
**B6. Chạy ứng dụng trực tuyến**

Ứng dụng đã được triển khai thành công, chúng ta có thể truy cập vào đường dẫn mặc định: [http://APP\_ID.appspot.com](http://APP_ID.appspot.com/) để vào ứng dụng. Trong bài hướng dẫn này, chúng ta đã đặt APP\_ID là hellogae2011, do đó, đường dẫn sẽ là [http://hellogae2011.appspot.com](http://hellogae2011.appspot.com/).



Hình B13: Màn hình chạy ứng dụng trên hệ thống GAE (1).

Click vào HelloWorld:



Hình B14: Màn hình chạy ứng dụng trên hệ thống GAE (2).

**Phụ lục B**

**CÁC VĂN BẢN PHÁP QUY VỀ TMĐT**

Sự ra đời của TMĐT đã và đang trở thành xu thế mới thay thế dần phương thức kinhdoanh trước đây với nhiều ưu thế nổi bật như: nhanh, rẻ, tiện dụng, hiệu quả và khôngbị giới hạn bởi không gian và thời gian. . .Thanh toán điện tử trong TMĐT được coi làphương thức tiện ích nhất hiện nay, là một mắt xích quan trọng để thúc đẩy TMĐT pháttriển. Thanh toán điện tử đã và đang phát triển hết sức mạnh mẽ trên thế giới.Trong đó, cơ sở pháp lý là yếu tố quantrọng quyết định sự phát triển của thanh toán điện tử. Chỉ khi có khung pháp lý chặt chẽ,đảm bảo phân định rõ quyền và nghĩa vụ của các bên có liên quan thì giao dịch điện tửmới có thể phát triển mạnh mẽ, rộng khắp. Tạo lập và hoàn thiện môi trường pháp lýlà yêu cầu cấp bách của giao dịch điện tử, thúc đẩy quá trình tăng trưởng kinh tế, tăngcường hiệu quả quản lý của các cơ quan nhà nước, hoạt động kinh doanh của doanhnghiệp và mở rộng dịch vụ đến cho cộng đồng.

Luật TMĐT ra đời điều chỉnh các quan hệ phát sinh từ giao dịch điện tử, bảo vệquyền và lợi ích hợp pháp của tổ chức, cá nhân tham gia quan hệ này.

* Thứ nhất, phải xác định giá trị pháp lý của văn bản điện tử. Một văn bản điện tử

đảm bảo các thành tố: khẳng định người ký, đảm bảo toàn vẹn của nội dung thông

tin phải được coi như có giá trị như văn bản trên giấy truyền thống.

* Thứ hai, cần quy định rõ trách nhiệm của các đối tượng tham gia giao dịch điện tử

(kể cả nhà cung cấp chứng thực số) trong việc cung cấp, nhận, xử lý thông tin, bảo

đảm an toàn hệ thống. . .

* Thứ ba, phải thay đổi quy trình của từng giao dịch cụ thể theo mức độ phổ biến, và

hành lang pháp lý của giao dịch điện tử trong xã hội. Trong một số giao dịch,theocách truyền thống, ngoài khai báo, bên có yêu cầu phải nộp bản photocopy và

xuất trình văn bản gốc cho người xử lý. Khi giao dịch điện tử chưa phát triển như

ở nước ta hiện nay thì việc khai thác các cơ sở dữ liệu điện tử có liên quan để đối

chiếu, xác minh ngay là chưa thực hiện được vì chưa có hoặc không được phổ biến

trên mạng (ví dụ như giấy phép xuất nhập khẩu, vận đơn. . . ). Do đó, phải có quy

định về giới hạn giao dịch và trách nhiệm lưu giữ bản gốc.

* Thứ tư, phải có chế tài xử lý nghiêm khắc các vi phạm. Trên thế giới, nhiều nước để

ban hành các đạo luật công nhận giá trị pháp lý của chữ ký điện tử, tạo hành lang

pháp lý cho giao dịch điện tử phát triển.

Cơ sở pháp lý điều chỉnh hoạt động TMĐT ở Việt Nam ra đời khá muộn so với nhiềunước trên thế giới. Cuối năm 2005, Việt Nam ban hành "Luật Giao dịch điện tử" vànăm 2006 mới ra đời Nghị định hướng dẫn thi hành luật này. Tới đầu năm 2007, Chínhphủ Việt Nam ban hành Nghị định số 27/2007/NĐ-CP ngày 23/02/2007 "Về giao dịchđiện tử trong hoạt động tài chính", số 26/2007/NĐ-CP ngày 15/02/2007 "Quy định chitiết thi hành Luật Giao dịch điện tử về chữ ký số và dịch vụ chứng thực chữ ký số", số35/2007/NĐ-CP ngày 08/03/2007 "Về giao dịch điện tử trong hoạt động ngân hàng".Năm 2007, bốn trong số năm nghị định hướng dẫn Luật Giao dịch điện tử đã được banhành, về cơ bản hoàn thành khung pháp lý cho việc triển khai ứng dụng giao dịch điệntử trong các lĩnh vực lớn của đời sống xã hội.

**B.1 Luật giao dịch điện tử**

Luật giao dịch điện tử số 51/2005/QH11, được Quốc hội khoá XI thông qua ngày29/11/2005 tại kỳ họp thứ 8, chính thức có hiệu lực từ ngày 01/03/2006. Luật gồm 8chương, với 54 điều bao gồm hầu hết các yếu tố, bên liên quan đến giao dịch điện tử như: Chữ ký điện tử, tổ chức cung cấp dịch vụ chứng thực chữ ký điện tử, giá trị pháp lý chữký điện tử, giá trị pháp lý của hợp đồng ký bằng chữ ký điện tử, trách nhiệm các bên liênquan đến bảo mật thông tin, giải quyết tranh chấp liên quan đến giao dịch điện tử cũngnhư quy định về giao dịch điện tử trong hoạt động của các cơ quan nhà nước; lĩnh vựcdân sự, kinh doanh, thương mại và các lĩnh vực khác do pháp luật quy định.

Luật Giao dịch điện tử ra đời đã tạo hành lang pháp lý cơ bản cho các giao dịch điện

tử.

* Chương I: Những quy định chung, gồm 9 điều, từ Điều 1 đến Điều 9.
* Chương II: Thông điệp dữ liệu, gồm 11 điều, từ Điều 10 đến Điều 20.
* Chương III: Chữ ký điện tử và chứng thực chữ ký điện tử, gồm 12 điều, từ Điều 21

đến Điều 32.

* Chương IV: Giao kết và thực hiện hợp đồng điện tử, gồm 6 điều, từ Điều 33 đến

Điều 38.

* Chương V: Giao dịch điện tử của cơ quan nhà nước, gồm 5 điều, từ Điều 39 đến

Điều 43.

* Chương VI: An ninh, an toàn, bảo vệ, bảo mật trong giao dịch điện tử, gồm 6 điềum

từ Điều 44 đến Điều 49.

* Chương VII: Giải quyết tranh chấp và xử lý vi phạm, gồm 3 điều, từ Điều 59 đến

Điều 52.

* Chương VIII: Điều khoản thi hành, gồm 2 điều, Điều 53 và Điều 54.

**B.2 Nghị định TMĐT**

Ngày 9 tháng 6 năm 2006, Thủ tướng Chính phủ đã ký ban hành Nghị định về TMĐTsố 57/2006/NĐ-CP. Đây là nghị định đầu tiên trong 5 nghị định hướng dẫn Luật Giaodịch điện tử và nghị định thứ sáu trong số 12 nghị định hướng dẫn Luật Thương mại (sửađổi) được ban hành.

Nghị định về TMĐT ra đời đánh dấu một bước tiến lớn trong việc tạo hành lang pháp lý để các doanh nghiệp yên tâm tiến hành giao dịch TMĐT, khuyến khích TMĐT pháttriển, bảo vệ quyền và lợi ích của các bên tham gia, đồng thời cũng là căn cứ pháp lý đểxét xử khi có tranh chấp liên quan đến hoạt động TMĐT.

Nghị định về TMĐT được xây dựng dựa trên một số quan điểm và mục tiêu: bám sátcác quy định tại Luật Thương mại, Bộ luật Dân sự và Luật Giao dịch điện tử; hỗ trợ tốiđa hoạt động kinh doanh hợp pháp của doanh nghiệp và cá nhân trong xã hội; bao quátcác loại hình TMĐT diễn ra trong thực tế, đồng thời có tính đến sự thay đổi, phát triểnnhanh chóng của những loại hình giao dịch mới. Nghị định cũng đảm bảo sự tương thíchvới luật pháp quốc tế, đồng thời phù hợp với điều kiện của Việt Nam.

Nghị định gồm 5 chương, 19 điều với những nội dung chính như sau:

* Chương I: Những quy định chung (từ Điều 1 đến Điều 6) nêu lên phạm vi điều

chỉnh, đối tượng áp dụng, giải thích một số thuật ngữ, xác định nội dung quản lý

nhà nước về TMĐT và cơ quan thực hiện nhiệm vụ này.

* Chương II: Giá trị pháp lý của chứng từ điện tử (từ Điều 7 đến Điều 10) khằng

định nguyên tắc cơ bản về thừa nhận giá trị pháp lý của chứng từ điện tử trong hoạt

động thương mại.

* Chương III: Chứng từ điện tử trong hoạt động thương mại (từ Điều 11 đến Điều 15) quy định chi tiết một số điều khoản về sử dụng chứng từ điện tử trong hoạtđộng thương mại như thời điểm, địa điểm nhận và gửi chứng từ điện tử, thông báovề đề nghị giao kết hợp đồng, sử dụng hệ thống thông tin tự động để giao kết hợpđồng, lỗi nhập thông tin trong chứng từ điện tử.
* Chương IV: Xử lý vi phạm (Điều 16, 17) và Chương V: Điều khoản thi hành (Điều18, 19) quy định các hành vi bị coi là vi phạm pháp luật về TMĐT, hình thức xửlý vi phạm, thời điểm hiệu lực của Nghị định và các cơ quan chịu trách nhiệm thihành.Để nghị định về TMĐT có thể đi vào cuộc sống, đồng thời tiếp tục hoàn thiện khungpháp lý cho hoạt động TMĐT tại Việt Nam, các cơ quan chức năng cần nghiên cứu xâydựng những văn bản hướng dẫn chi tiết hơn nữa về việc ứng dụng TMĐT trong các lĩnhvực hoạt động đặc thù như cung ứng hàng hóa dịch vụ kinh doanh có điều kiện, quảngcáo thương mại qua phương tiện điện tử, sử dụng chứng từ điện tử trong hoạt động thươngmại trực tuyến, chống thư rác, bảo vệ người tiêu dùng và các quy định liên quan khác.

**B.3 Nghị định hướng dẫn giao dịch TMĐT**

**B.3.1 Nghị định số 27/2007/NĐ-CP quy định chi tiết thi hành Luật Giao**

**dịch điện tử trong hoạt động tài chính**

Ngày 23/2/2007,Chính phủ ban hành Nghị định số 27/2007/NĐ-CP quy định chi tiếtthi hành Luật Giao dịch điện tử trong hoạt động tài chính. Nghị định này ra đời nhằmđảm bảo các điều kiện cần thiết để hình thành và phát triển một môi trường giao dịchđiện tử an toàn, hiệu quả; giúp Chính phủ quản lý được giao dịch điện tử trong hoạt độngnghiệp vụ tài chính, giảm thiểu hậu quả xấu phát sinh trong giao dịch điện tử như trốnthuế, gian lận khi lập hóa đơn chứng từ.

Nghị định bao gồm 5 Chương và 22 Điều:

* Chương I: Những qui định chung (Điều 1 – Điều 4).
* Chương II: Chứng từ điện tử (Điều 5 – Điều 10).
* Chương III: Giao dịch điện tử trong hoạt động tài chính (Điều 11 – Điều 16).
* Chương VI: Giải quyết tranh chấp, khiếu nại, tố cáo, thanh tra, kiểm tra và xử lý viphạm về giao dịch điện tử trong hoạt động tài chính (Điều 17 – Điều 20).
* Chương V: Điều khoản thi hành (Điều 21 – Điều 22).

**B.3.2 Nghị định số 26/2007/NĐ-CP quy định chi tiết thi hành Luật Giaodịch điện tử về Chữ ký số và Dịch vụ chứng thực chữ ký số**

Ngày 15/2/2007, Nghị định số 26/2007/NĐ-CP quy định chi tiết về Chữ ký số vàDịch vụ chứng thực chữ ký số được ban hành. Nghị định này quy định về chữ ký số vàcác nội dung cần thiết liên quan đến sử dụng chữ ký số, bao gồm chứng thư số và việcquản lý, cung cấp và sử dụng dịch vụ chứng thực chữ ký số. Đây là những quy định nềntảng để thiết lập một cơ chế đảm bảo an ninh an toàn cũng như độ tin cậy của các giaodịch điện tử, là điều kiện tiên quyết về mặt kỹ thuật để thúc đẩy ứng dụng TMĐT rộngrãi trong xã hội.

Nghị định gồm 11 Chương, 73 Điều với nội dung như sau:

* Chương I: Những qui định chung (Điều 1- Điều 7).
* Chương II: Chữ ký số và chứng thư số (Điều 8- Điều 12).
* Chương III: Cấp phép cung cấp dịch vụ chứng thực chữ ký số công cộng (Điều 13-

Điều 20.

* Chương IV: Hoạt động của tổ chức cung cấp dịch vụ chứng thực chữ ký số công

cộng (Điều 21- Điều 28).

* Chương V: Quyền và nghĩa vụ của các bên tham cung cấp và sử dụng dịch vụ

chứng thực chữ ký số công cộng (Điều 29- Điều 44).

* Chương VI: Tổ chức cung cấp dịch vụ chứng thực chữ ký số công cộng (Điều 45-

Điều 51).

* Chương VII: Công nhận chữ ký số, chứng thư số và hoạt động cung cấp dịch vụ

của tổ chức cung cấp dịch vụ chứng thực chữ ký số nước ngoài (Điều 52- Điều 55).

* Chương VIII: Tổ chức cung cấp dịch vụ chứng thực chữ ký số nước quốc gia(Điều

56- Điều 57).

* Chương IX: Tranh chấp, khiếu nại, tố cáo và bồi thường (Điều 58- Điều 60).
* Chương X: Thanh tra, kiểm tra và xử lý vi phạm (Điều 61- Điều 72).
* Chương XI: Điều khoản thi hành (Điều 73) .

**B.4 Nghị định về giao dịch điện tử trong lĩnh vực tài chính**

Ngày 8/3/2007, Nghị định số 35/2007/NĐ-CP về Giao dịch điện tử trong hoạt độngNH được ban hành tập trung hướng dẫn việc áp dụng Luật Giao dịch điện tử cho cáchoạt động NH cụ thể, bảo đảm những điều kiện cần thiết về môi trường pháp lý để củngcố, phát triển các giao dịch điện tử an toàn và hiệu quả đối với hệ thống NH.

Nghị định bao gồm 5 Chương và 30 Điều với nội dung như sau:

* Chương I: Những qui định chung (Điều 1 – Điều 3).
* Chương II: Giao dịch điện tử trong hoạt động ngân hàng (Điều 4 – Điều 7).
* Chương III: Chứng từ điện tử trong hoạt động ngân hàng (Điều 8 – Điều 24).
* Chương VI: Giải quyết tranh chấp, khiếu nại, tố cáo, thanh tra, kiểm tra và xử lý viphạm (Điều 25 – Điều 28).
* Chương V: Điều khoản thi hành (Điều 29 – Điều 30).