# LỜI NÓI ĐẦU

Với sự phát triển nhảy vọt của công nghệ thông tin hiện nay, Internet ngày càng giữ vai trò quan trọng trong các lĩnh vực khoa học kĩ thuật và đời sống. Dĩ nhiên các bạn đã được nghe nói nhiều về Internet, nói một cách đơn giản, Internet là một tập hợp máy tính nối kết với nhau, là một mạng máy tính toàn cầu mà bất kì ai cũng có thể kết nối bằng máy PC của họ. Với mạng Internet, tin học thật sự tạo nên một cuộc cách mạng trao đổi thông tin trong mọi lĩnh vực văn hóa, xã hội, chính trị, kinh tế...

Tận dụng những tính năng mạnh mẽ đó của Internet, những công nghệ mới về thông tin và truyền thông, người ta đã xây dựng nên những hệ thống đặc biệt để mọi người có thể ứng dụng đưa vào cuộc sống thực tiễn, và điều đó thực sự sẽ làm thay đổi cuộc sống của xã hội chúng ta theo chiều hướng tiện lợi hơn và văn minh hơn. Chỉ với một máy tính cá nhân có kết nối mạng, bạn có thể truy cập các trang WEB của các công ty, dịch vụ; thoải mái chọn lựa các mặt hàng mình thích, các thông tin mình cần ngay tại chỗ. Các công ty, các doanh nghiệp có thể quảng cảo các sản phẩm, các dịch vụ của mình trên các WEBSITE. Điều ngày sẽ giúp tăng doanh thu, tăng khách hàng và đặc biệt là dễ dàng tìm kiếm các đối tác kinh doanh. Cho nên cần phải có những công nghệ mới để quản lý, quản trị WEBSITE.

Với lí do đó, được sự hướng dẫn và giúp đỡ của thầy giáo *Ths. Mai Văn Hoàn*, em đã chọn đề tài: ” ***Xây dựng công cụ chăm sóc khách hàng thời gian thực cho các trang thương mại điện tử***.” làm đề tài cho Thực tập tốt nghiệp của mình.

Em xin chân thành cảm ơn Thạc sỹ Mai Văn Hoàn– giáo viên đã hưỡng dẫn trong quá trình làm thực tập chuyên ngành.

Tuy nhiên, do thời gian hạn hẹp, mặc dù đã nỗ lực hết sức mình nhưng chắc rằng đồ án khó tránh khỏi thiếu sót. Em rất mong nhận được sự thông cảm, những lời góp ý và chỉ bảo tận tình của quý Thầy Cô và các bạn.

# CHƯƠNG I. TÌM HIỂU VỀ CÔNG NGHỆ NODE.JS

**1.1 Khái niệm về node.js**

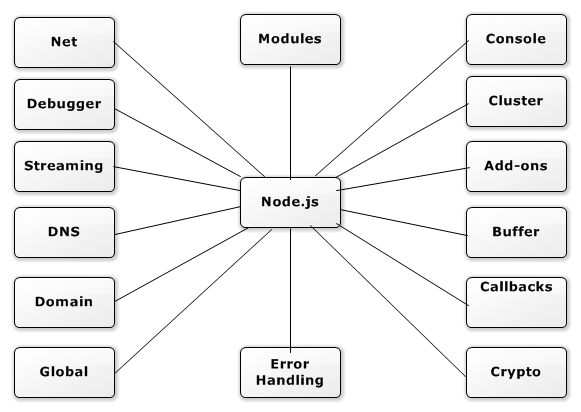
*Node.js* là một hệ thống phần mềm được thiết kế để viết các ứng dụng internet có khả năng mở rộng, đặc biệt là may chủ web.Chương trình được viết bằng JavaScript, sử dụng kỹ thật điều khển theo sự kiện, nhập/xuất khẩu không đồng bộ để tối tiểu tổng chi phí và tối đại khaả năng mở rộng. Node.js bao gồm có V8 JavaScript runtime của Google, libUV, và vài thư viện khác.

*Node.js* được tạo bởi Ryan Dahl từ năm 2009, và phát triển dưới sự bảo trợ của Joyent.

Trong một môi trường server điển hình LAMP (Linux-Apache-MySQL-PHP), bạn có một web server là Apache hoặc NGINX nằm dưới, cùng với PHP chạy trên nó. Mỗi một kết nối tới server sẽ sinh ra một thread mới, và điều này khiến ứng dụng nhanh chóng trở nên chậm chạp hoặc quá tải - cách duy nhất để hỗ trợ nhiều người dùng hơn là bằng cách bổ sung thêm nhiều máy chủ. Đơn giản là nó không có khả năng mở rộng tốt. Nhưng với Node.js thì điều này không phải là vấn đề. Không có một máy chủ Apache lắng nghe các kết nối tới và trả về mã trạng thái HTTP - bạn sẽ phải tự quản lý kiến trúc lõi của máy chủ đó. May mắn thay, có một số module giúp thực hiện điều này được dễ dàng hơn, nhưng công việc này vẫn gây cho bạn một chút khó khăn khi mới bắt đầu. Tuy nhiên, kết quả thu được là một ứng dụng web có tốc độ thực thi cao.

**1.2. Các đặc tính của Node.js**

* **Không đồng bộ**: Tất cả các API của NodeJS đều không đồng bộ (none-blocking), nó chủ yếu dựa trên nền của NodeJS Server và chờ đợi Server trả dữ liệu về. Việc di chuyển máy chủ đến các API tiếp theo sau khi gọi và cơ chế thông báo các sự kiện của Node.js giúp máy chủ để có được một phản ứng từ các cuộc gọi API trước (Realtime).
* **Chạy rất nhanh**: NodeJ được xây dựng dựa vào nền tảng V8 Javascript Engine nên việc thực thi chương trình rất nhanh.
* **Đơn luồng nhưng khả năng mở rộng cao**: Node.js sử dụng một mô hình luồng duy nhất với sự kiện lặp. cơ chế tổ chức sự kiện giúp các máy chủ để đáp ứng một cách không ngăn chặn và làm cho máy chủ cao khả năng mở rộng như trái ngược với các máy chủ truyền thống mà tạo đề hạn chế để xử lý yêu cầu. Node.js sử dụng một chương trình đơn luồng và các chương trình tương tự có thể cung cấp dịch vụ cho một số lượng lớn hơn nhiều so với yêu cầu máy chủ truyền thống như Apache HTTP Server.
* **Không đệm**: NodeJS không đệm bất kì một dữ liệu nào và các ứng dụng này chủ yếu là đầu ra dữ liệu.
* **Có giấy phép**: NodeJS đã được cấp giấy phép bởi [*MIT License*](https://raw.githubusercontent.com/joyent/node/v0.12.0/LICENSE).
* **Các thành phần quan trọng trong Node.js:**



**1.3 JavaScript**

***JavaScript*** là một ngôn ngữ dựa trên sự kiện, vì vậy bất cứ thứ gì xảy ra trên server đều tạo ra một sự kiện non-blocking. Mỗi kết nối mới sinh ra một sự kiện; dữ liệu nhận được từ một upload form sinh ra một sự kiện data-received; việc truy vấn dữ liệu từ database cũng sinh ra một sự kiện. Trong thực tế, điều này có nghĩa là một trang web Node.js sẽ chẳng bao giờ bị khóa (lock up) và có thể hỗ trợ cho hàng chục nghìn user truy cập cùng lúc. Node.js đóng vai trò của server - Apache - và thông dịch mã ứng dụng chạy trên nó. Giống như Apache, có rất nhiều module (thư viện) có thể được cài đặt để bổ sung thêm các đặc trưng và chức năng - như lưu trữ dữ liệu, hỗ trợ file Zip, đăng nhập bằng Facebook, hoặc các cổng thanh toán. Dĩ nhiên, nó không có nhiều thư viện như PHP, nhưng Node.js vẫn đang ở trong giai đoạn ban đầu và có một cộng đồng rất mạnh mẽ ở đằng sau nó.

**1.4 Tại sao lại sử dụng Node.js**

Đầu tiên là ưu điểm về tốc độ thực thi và khả năng mở rộng. Node.js có tốc độ rất nhanh. Đó là một yêu cầu khá quan trọng khi bạn là một startup đang cố gắng tạo ra một sản phẩm lớn và muốn đảm bảo có thể mở rộng nhanh chóng, đáp ứng được một lượng lớn người dùng khi trang web của bạn phát triển lên.

Node.js có thể xử lý hàng ngàn kết nối đồng thời trong khi PHP sẽ chỉ có nước sụp đổ. Bên cạnh các lợi ích về tốc độ thực thi và khả năng mở rộng, có thể bạn cũng đã biết một chút về JavaScript, vì vậy tại sao lại phải phiền toái để học thêm về một ngôn ngữ lập trình hoàn toàn mới như PHP? Và sau đó bạn sẽ có một sự phấn khích khi học về một cái gì đó mới mẻ và gần như chưa được khám phá. Bạn còn nhớ cái cảm giác khi mà một cái gì đó mới xuất hiện và sau đó trở thành phổ biến khắp mọi nơi mà bạn hối tiếc đã không học về nó sớm hơn, và mãi mãi chỉ là người đến sau? Đừng phạm phải sai lầm như vậy lần này nữa. Node.js đang ngày càng trở nên lớn mạnh hơn.

**1.5 Ưu điểm của Node.js**

*a, JSON APIs*  
Bởi lẽ REST/JSON APIs gọn nhẹ là điều khiến NodeJS tỏa sáng. Với cơ chế event-driven, non-blocking I/O(Input/Output) và mô hình kết hợp với Javascript là sự lựa chọn tuyệt vời cho các dịch vụ Webs làm bằng JSON.  
*b, Ứng dụng trên 1 trang*  
Nếu bạn định viết 1 ứng dụng thể hiện trên 1 trang (Gmail?) NodeJS rất phù hợp để làm. Với khả năng xử lý nhiều Request/s đồng thời thời gian phản hồi nhanh. Các ứng dụng bạn định viết không muốn nó tải lại trang, gồm rất nhiều request từ người dùng cần sự hoạt động nhanh để thể hiện sự chuyên nghiệp thì NodeJS sẽ là sự lựa chọn của bạn.  
*c, Shelling tools unix*  
NodeJS sẽ tận dụng tối đa Unix để hoạt động. Tức là NodeJS có thể xử lý hàng nghìn Process và trả ra 1 luồng khiến cho hiệu xuất hoạt động đạt mức tối đa nhất và tuyệt vời nhất.  
*d, Streamming Data (Luồng dữ liệu)*  
Các web thông thường gửi HTTP request và nhận phản hồi lại (Luồng dữ liệu). Giả xử sẽ cần xử lý 1 luồng giữ liệu cực lớn, NodeJS sẽ xây dựng các Proxy phân vùng các luồng dữ liệu để đảm bảo tối đa hoạt động cho các luồng dữ liệu khácl  
*e, Ứng dụng Web thực*  
Giả sử bạn xây dựng 1 ứng dụng chat, feed ... Facebook, Twitter là điển hình cho Web thực. NodeJS làm khá tốt điều đó!

**1.6 Nhược điểm của Node.js**

*a,Ứng dụng nặng tốn tài nguyên*  
Nếu bạn cần xử lý các ứng dụng tốn tài nguyên CPU như encoding video, convert file, decoding encryption... hoặc các ứng dụng tương tự như vậy thì không nên dùng NodeJS (Lý do: NodeJS được viết bằng C++ & Javascript, nên phải thông qua thêm 1 trình biên dịch của NodeJS sẽ lâu hơn 1 chút ). Trường hợp này bạn hãy viết 1 Addon C++ để tích hợp với NodeJS để tăng hiệu suất tối đa !(Việc tích hợp rất thân thiện và nhanh chóng)!  *b, NodeJS và ngôn ngữ khác*  
NodeJS, PHP, Ruby, Python .NET ...thì việc cuối cùng là phát triển các App Web. NodeJS mới sơ khai như các ngôn ngữ lập trình khác. Vậy nên bạn đừng hi vọng NodeJS sẽ hơn PHP,Ruby,Python... ở thời điểm này. Nhưng với NodeJS bạn có thể có 1 ứng dụng như mong đợi, điều đó là chắc chắn (perfect)!   
Với những gì các ngôn ngữ tiền bối đang có(cộng đồng lâu năm, framework, cms, opensource...) Nếu bạn/doanh nghiệp chưa biết về NodeJS thì việc cần xây dựng dự án quan trọng, kinh doanh phát triển trên NodeJS sẽ không phải lựa chọn bây giờ.

**1.7 Các modul quan trọng trong Node.js và tạo một chương trình đơn giản**

- Trong Nodejs hỗ trợ rất nhiều module với nhưng chức năng khác nhau,mỗi file đóng vai trò như 1 module và để gọi module nào đó ra chỉ cần sử dụng hàm require,ví dụ:

|  |  |
| --- | --- |
| 1 | var express = require('express'); |

Lệnh trên nạp module express vào biến express.

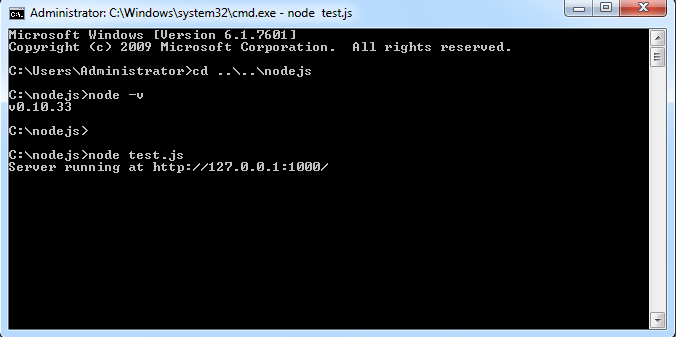
* Một số module hay sử dụng trong nodejs:
* **express**: đây được là là web framework của nodejs,cái này chúng ta sẽ tìm hiểu kỹ ở phần sau
* **mongoose** : quản lý mongdb.
* **jade** : Một dạng template mới dùng trong nodejs view,nhằm rút ngắn gọn view.
* **connect-mongodb** : Giúp kết nối với mongodb.
* **nodemailer** : Module gửi mail.
* **socket.io** : Sử dụng trong những ứng dụng cần realtime như chat.
* **request** : Quản lý yêu cầu và dữ liệu được truyền giữa client và server nodejs.
* Cài đặt Node.js: Phần mềm node.js rất dễ cài đặt. Down phần mềm node.js và cài đặt.
* Chạy chương trình đầu tiên:
* Trước tiên chúng ta sẽ tạo file test.js với nội dung

|  |  |
| --- | --- |
| 1  2  3  4  5  6 | var http = require('http');  http.createServer(function (req, res) {  res.writeHead(200, {'Content-Type': 'text/plain'});  res.end('Hello Node.js\n');  }).listen(1000, "127.0.0.1");  console.log('Server running at [**http://127.0.0.1:1000/**](http://127.0.0.1:1000/)'); |

* Chúng ta chưa vội tìm hiểu tới nội dung của file này,mà sẽ tìm hiểu dần ở các phần sau nhé.Bước tiếp theo chúng ta lại mở cửa sổ cmd lên và gõ dòng lệnh để thực thi file test.js này:

|  |  |
| --- | --- |
| 1 | node test.js |

Và sẽ được kết quả như sau:



* Kế tiếp các bạn có thể test bằng cách mở trình duyệt lên và gõ **http://127.0.0.1:1000/** sẽ thấy kết quả hiển thị ra màn hình dòng chữ Hello Node.js

# CHƯƠNG II. PHÂN TÍCH VÀ THIẾT KẾ HỆ THỐNG

Trong chương này chúng ta sẽ phân tích các yêu cầu của đề tài như việc tóm tắt hoạt động của hệ thống mà dự án sẽ được ứng dụng, phạm vi ứng dụng của đề tài và đối tượng sử dụng, mục đích của dự án. Xác định yêu cầu của khách hàng: Xuất phát từ hệ thống hiện hành của cửa hàng cùng với yêu cầu của khách hàng để vạch ra được yêu cầu hệ thống cần xây dựng như việc thiết kế giao diện, yêu cầu về chức năng và yêu cầu về bảo mật. Trong chương này chúng ta đi xây dựng một số biểu đồ Use Case, biểu đồ hoạt động của các chức năng trong hệ thống và thông tin cơ sở dữ liệu của chương trình.

1. **Phân tích yêu cầu của đề tài**

* *Tóm tắt hoạt động của hệ thống* ***mà dự án sẽ được ứng dụng:***
* Là một công cụ hỗ trợ cho web bán hàng**thế chủ động trong việc tiếp cận khách hàng** bằng khả năng chủ động mời chat. Theo dõi , thống kê được các hoạt động của khách hàng tạo ra khả năng cung cấp dịch vụ chăm sóc tốt nhất.
* Cung cấp cho các trang web bán hàng khả năng tạo ra những nhân viên thật, hỗ trợ ngay lập tức trên không gian ảo.
* Chat với khách hàng**ngay lập tức.**
* **Không yêu cầu** khách hàng phải cài đặt bất kì phần mềm hay đăng ký phức tạp nào để bắt đầu chat.
* ***Phạm vi dự án được ứng dụng:***
* Dành cho mọi đối tượng có nhu cầu mua sắm và tham khảo mặt hàng trên mạng.
* Do nghiệp vụ của cửa hàng kết hợp với công nghệ mới và được xử lý trên hệ thống máy tính nên công việc liên lạc nơi khách hàng . Rút ngắn được thời gian làm việc, cũng như đưa thông tin cần thiết nhanh chóng đến cho khách hàng.
* ***Đối tượng sử dụng:***
* Có 3 đối tượng sử dụng cơ bản: Administrator, customer, supporter.
* Administrator: là người quản trị toàn bộ hệ thống, có quyền và chức năng thêm, sửa, xóa supporter, xóa lịch sử chat.
* Customer: là khách hàng khi truy cập trang web, điền thông tin cá nhân để có thể chat trực tiếp với chuyên viên tư vấn của trang web.
* Supporter: là thành viên của website có có nhiệm vụ trả lời khách hàng khi có tin nhắn gửi đến, tư vấn hỗ trợ khách hàng.
* ***Mục đích của dự án:***

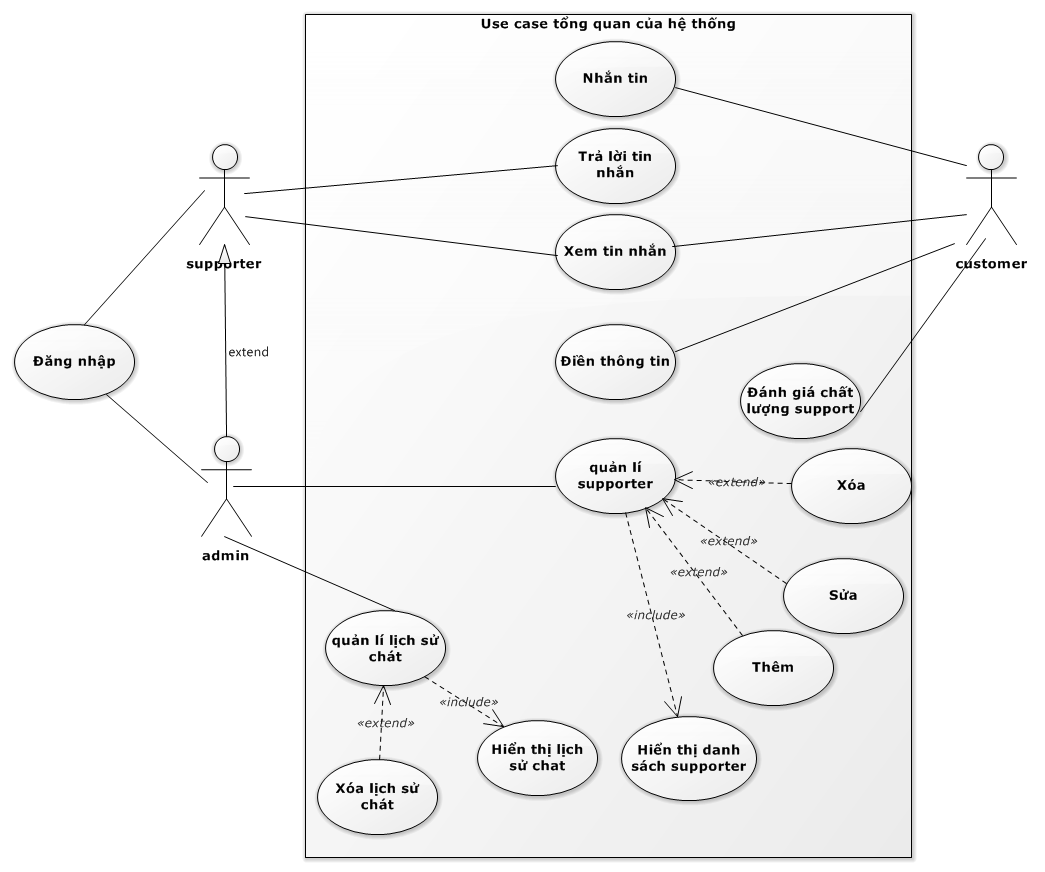
- Đáp ứng nhu cầu mua bán ngày càng phát triển của xã hội, xây dựng nên một môi trường làm việc hiệu quả.

- Thúc đẩy phát triển buôn bán trực tuyến.

- Rút ngắn khoảng cách giữa người mua và người bán, tạo ra một Website trực tuyến có thể đưa nhanh thông tin cũng như việc trao đổi mua bán các sản phẩm qua mạng.

* ***Xác định yêu cầu của giao diện:***
* Phải có một giao diện thân thiện dễ sử dụng
* Màu sắc hài hoà, font chữ thống nhất, tiện lợi khi sử dụng.

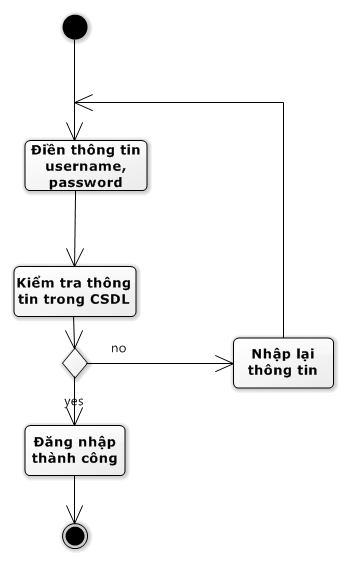
**2.2 Các tác nhân và USE CASE của hệ thống**

****

*Hình 1. Use case tổng quan của hệ thống*

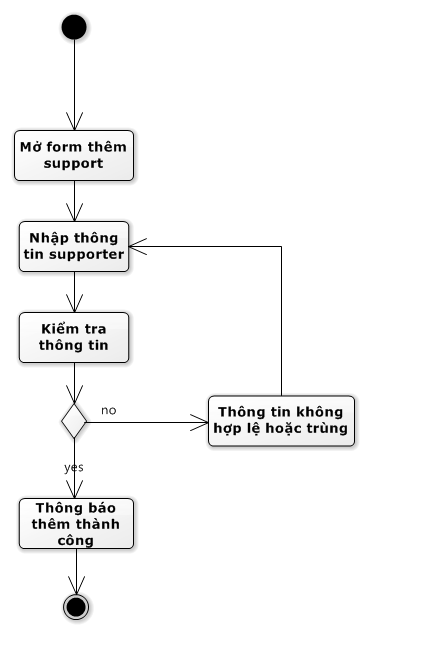
**2.3 Biểu đồ hoạt động của hệ thống**

* **Đăng nhập**

****

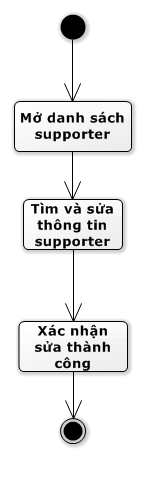
*Hình 2 : Biểu đồ hoạt động đăng nhập hệ thống*

* **Thêm supporter**

****

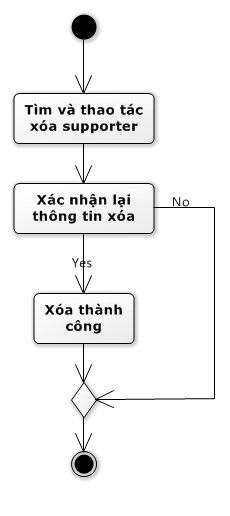
*Hình 3: Biểu đồ hoạt động thêm supporter*

* **Sửa supporter**

****

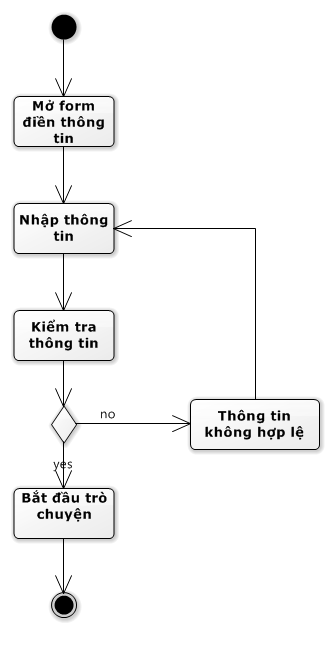
*Hình 4: Biểu đồ hoạt động sửa supporter*

* **Xóa supporter**

****

*Hình 5: Biểu đồ hoạt động xóa supporter*

* **Điền thông tin customer**

****

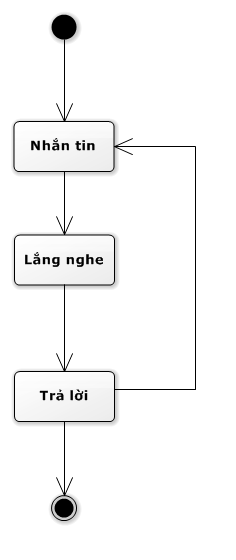
*Hình 6: Biểu đồ hoạt động điền thông tin customer*

* **Đánh giá**

****

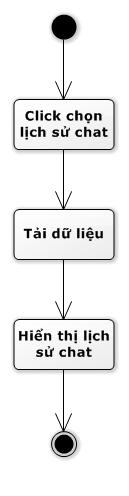
*Hình 7: Biểu đồ hoạt động đánh giá*

* **Nhắn tin**

****

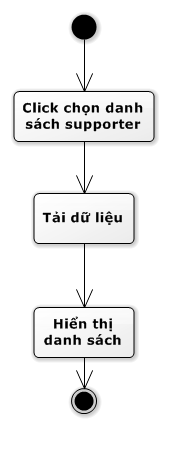
*Hình 8: Biểu đồ hoạt động nhắn tin*

* **Hiển thị lịch sử chat**

****

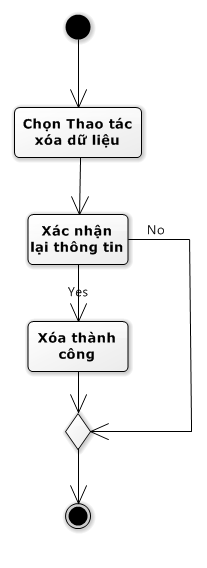
*Hình 9: Biểu đồ hoạt động lịch sử chat*

* **Hiển thị danh sách supporter**

****

*Hình 10: Biểu đồ hoạt động hiển thị danh sách supporter*

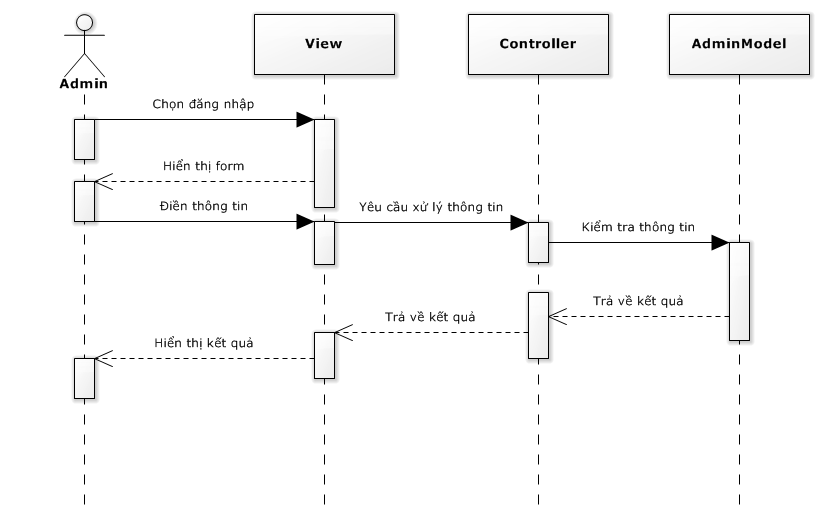
* **Xóa lịch sử chat**

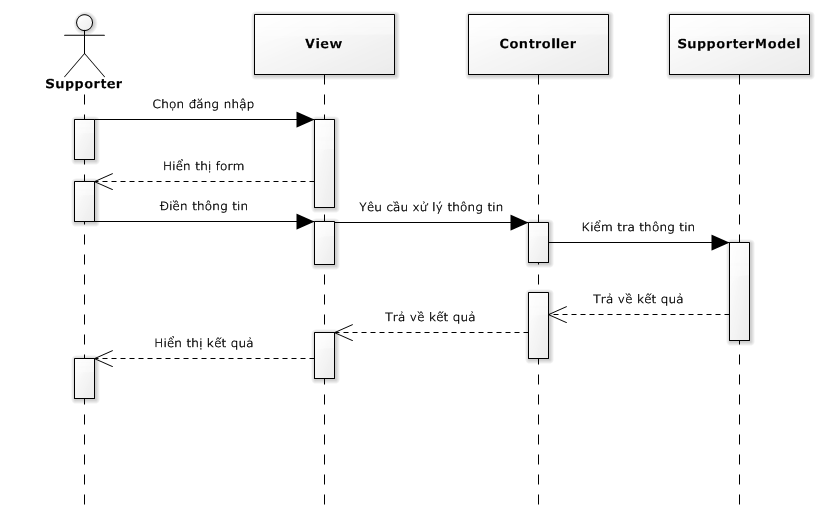
****

*Hình 11: Biểu đồ hoạt động xóa lịch sử chat*

**2.4 Biểu đồ trình tự của hệ thống**

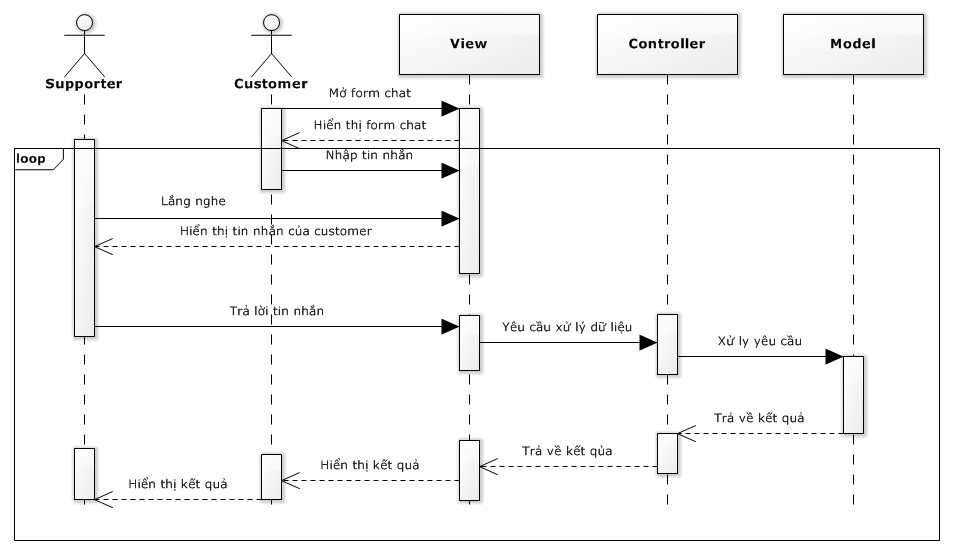
* **Đăng nhập**

**

**

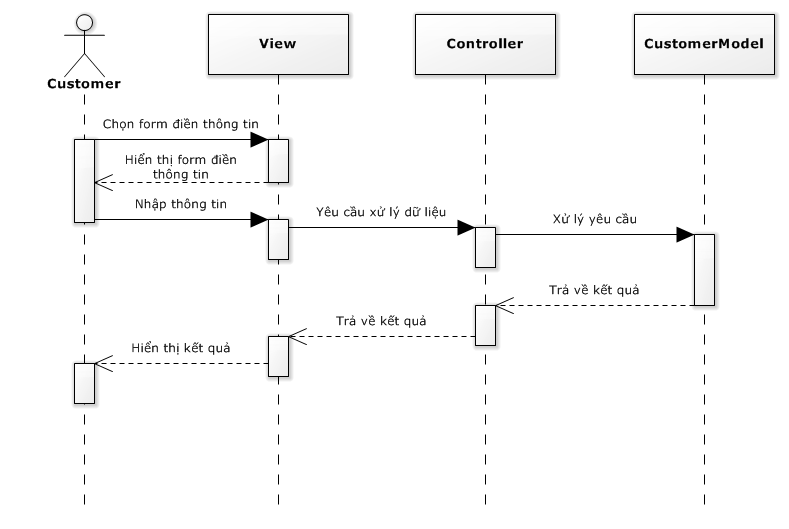
*Hình 12. Biểu đồ trình tự: Đăng nhập*

* **Nhắn tin**

**

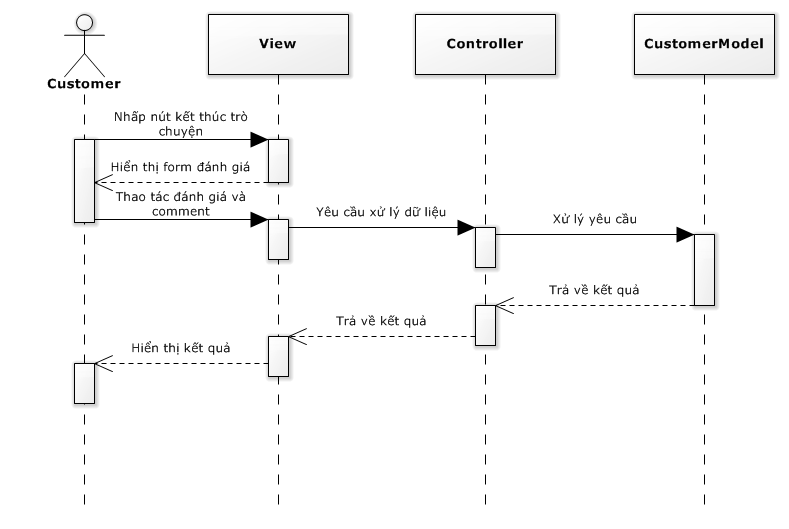
*Hình 13. Biểu đồ trình tự: Nhắn tin*

* **Điền thông tin**

**

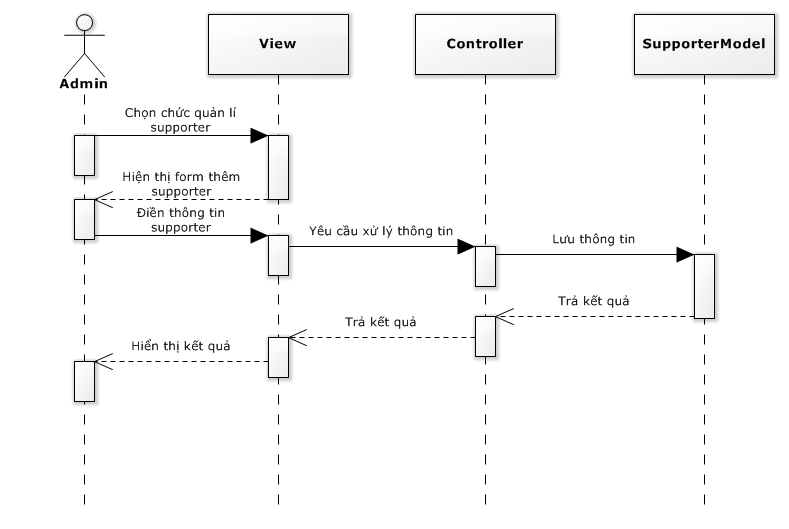
*Hình14. Biểu đồ trình tự: Điền thông tin*

* **Đánh giá**

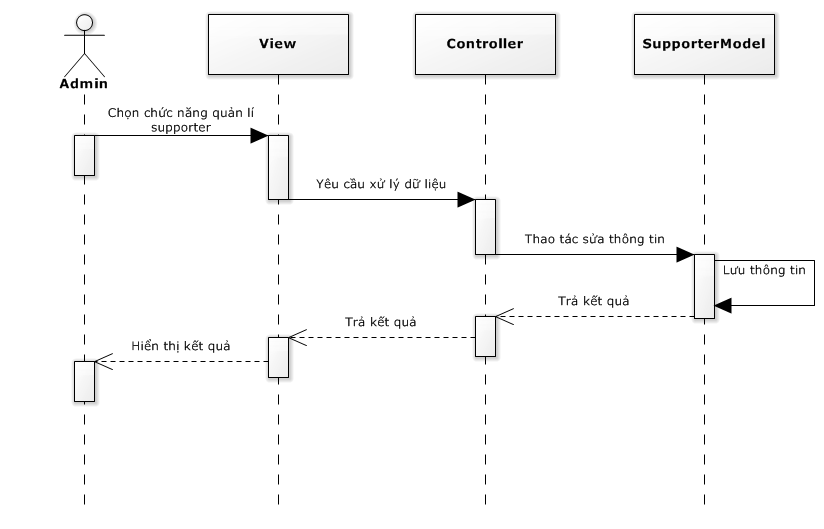
**

*Hình 15: Biểu đồ trình tự: Đánh giá*

* **Thêm supporetr**

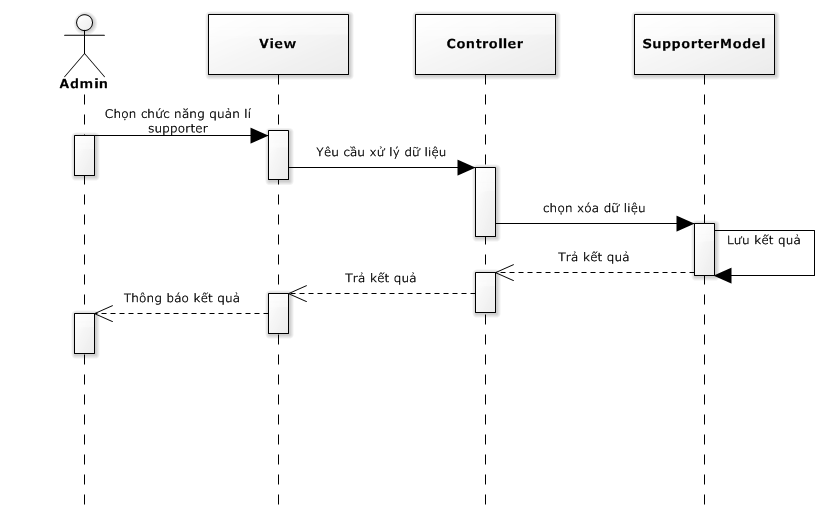
* Hình 16: Biểu đồ trình tự: Thêm supporter*

* **Sửa supporter**

**

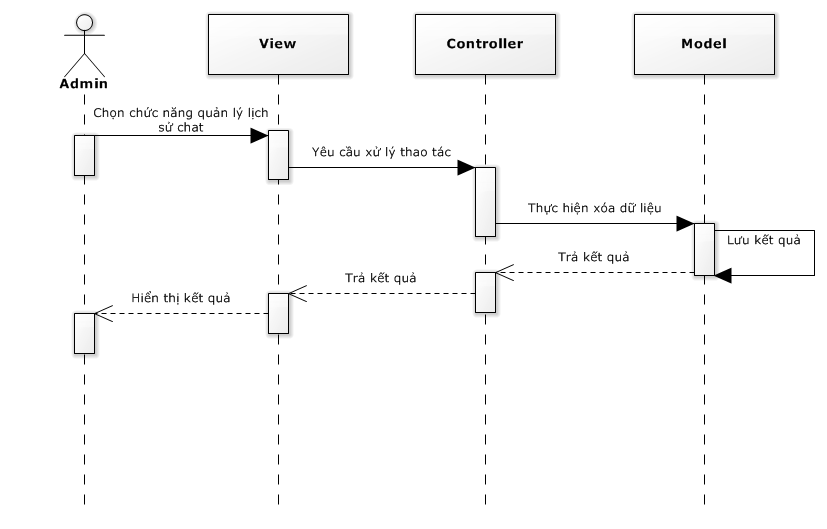
*Hình 17: Biểu đồ trình tự: Sửa supporter*

* **Xóa supporter**

**

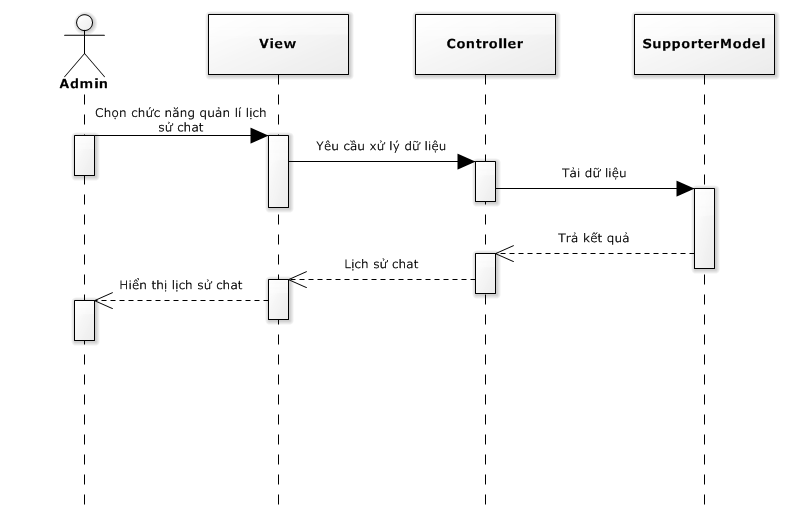
*Hình 18. Biểu đồ trình tự: Xóa supporter*

* **Xóa lịch sử chát**

**

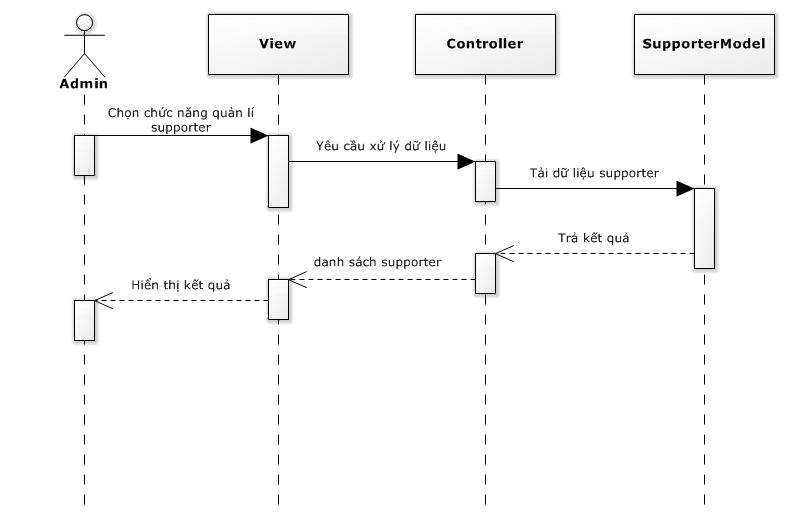
*Hình 19: Biều đồ trình tự: Xóa lịch sử chat*

* **Hiển thị lịch sử chat**

**

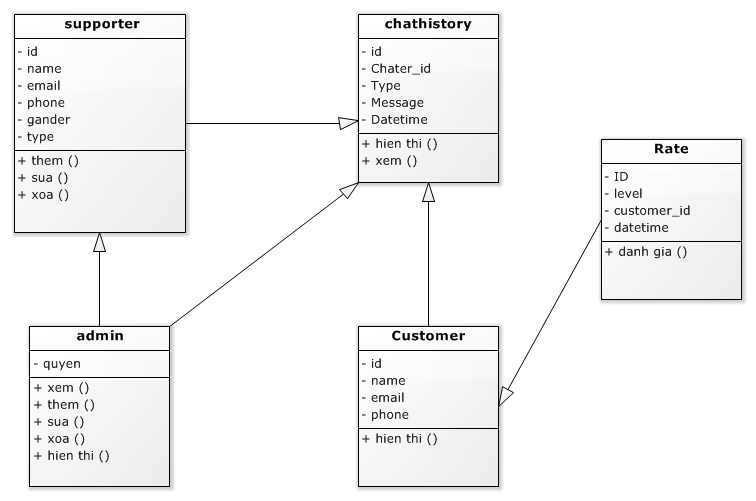
*Hình 20. Biểu đồ trình tự: Hiển thị lịch sử chat*

* **Hiển thị danh sách supporter**

**

*Hình 21. Biểu đồ trình tự: Hiển thị danh sách supporter*

**2.5 Biểu đồ lớp**

****

*Hình 22: Biều đồ lớp của hệ thống*

Bảng 1: Quản lí thông tin customer

|  |  |  |
| --- | --- | --- |
| **Tên thuộc tính** | **Kiểu dữ liệu** | **Mô tả** |
| ID | Int | Khóa chính |
| email | Varchar(255) |  |
| name | Varchar(100) |  |
| phone | Varchar(15) |  |

Bảng 2: Lịch sử chat

|  |  |  |
| --- | --- | --- |
| **Tên thuộc tính** | **Kiểu dữ liệu** | **Mô tả** |
| ID | Int | Khóa chính |
| Chater\_ID | Int |  |
| Type | Int |  |
| Message | Text |  |
| Datetime | Timestamp |  |

Bảng 3: Quản lí thông tin supporter

|  |  |  |
| --- | --- | --- |
| **Tên thuộc tính** | **Kiểu dữ liệu** | **Mô tả** |
| ID | Int | Khóa chính |
| name | Varchar(100) |  |
| phone | Varchar(15) |  |
| email | Varchar(255) |  |
| gender | Varchar(10) |  |
| type | Int |  |

Bảng 4: Bảng đánh giá rate

|  |  |  |
| --- | --- | --- |
| **Tên thuôc tính** | **Kiểu dữ liệu** | **Mô tả** |
| ID | Int | Khóa chính |
| level | Int |  |
| Customer\_id | Int |  |
| datetime | Timestamp |  |

# Chương 3. Xây dựng chương trình