

**APP CHAT TT– NHÓM09**

mÔN CÔNG NGHỆ MỚI TRONG PHÁT TRIỂN ỨNG DỤNG CNTT

Bộ môn Kỹ thuật phần mềm – Khoa Công nghệ thông tin – ĐH Công nghiệp TP. HCM

**NỘI DUNG**

# **GIỚI THIỆU CHUNG**

## Tính cấp thiết của đề tài

Ngày nay, ứng dụng công nghệ và việc tin học hóa được xem là một trong những yếu tố mang tính quyết định trong hoạt động của các chính phủ , tổ chức, cũng như các công ty, nó đóng vai trò hết sức quan trọng , có thể tạo ra những bước đột phá mạnh mẽ. Với sự phát triển ngày càng nhanh chóng này của xã hội thì việc phát triển hệ thống mạng lưới thông tin là một điều kiện tiên quyết. Và khi hệ thống mạng lưới thông tin được phủ sóng rộng rãi, chúng ta chắc chắn sẽ thực hiện được nhiều công việc, chuyển thông tin, dữ liệu với tốc độ nhanh hơn và chi phí thấp hơn nhiều so với cách thức truyền thống. Chính điều này, đã thúc đẩy sự khai sinh và phát triển ứng dụng Chat -ứng dụng truyền thông giúp cho việc truyền thông tin nay trở nên dễ dàng hơn và tiết kiệm hơn, nhanh chóng hơn, làm biến đổi đáng kể bộ mặt văn hóa, nâng cao chất lượng cuộc sống con người.

## Đối tượng và phạm vi nghiên cứu

Nghiên cứu phát triển ứng dụng chat trên hai nền tảng phổ biến nhất hiện nay là androi và web browser

## Phương pháp nghiên cứu

Sử dụng kiến thức đã học và các tài liệu liên quan trong và ngoài nước liên quan đến đề tài nghiên cứu đặc biệt là các tài liệu về phát triển ứng dụng , thiết kế hệ thống trong hệ thống thông tin di động và web.

Nghiên cứu và tìm hiểu tập trung bám sát đề cương dưới sự hướng dẫn của giảng viên

## Mục tiêu nghiên cứu đề tài

Nghiên cứu và phát triển thành công để ứng dụng có thể truyền thông tin trực tiếp theo thời gian thực và chạy ổn định trên hai nền tảng android và web

# **CƠ SỞ LÝ THUYẾT**

## Giới thiệu

## Dưới sự phát triển càng nhanh của xã hội thì theo đó là sự lớn mạnh một cách nhanh chóng của công nghệ thông tin. Vì lẽ đó nên chúng ta ngày càng được chứng kiến nhiều sự xuất hiện của rất nhiều công nghệ mới, nền tảng phát triển mới như hiện nay. Vậy thì, việc lựa chọn ngôn ngữ, lựa chọn nền tảng để bắt đầu một dự án là một việc hết sức phức tạp và phụ thuộc vào rất nhiều yếu tố cũng như mục đích của dự án. Và ở đây chúng ta sẽ nói về lý do để lựa chọn và đặc điểm của các công nghệ được sử dụng cho việc phát triển hệ thống chat.

## Giới thiệu chung về hệ thống chat

## Để xây dựng một hệ thống Chat lấy dữ liệu thời gian thực thì ứng dụng phải gồm 2 phần chính là phần giao diện người dùng và phần máy chủ để xử lý và lưu trữ dữ liệu.

## Trước tiên là phần giao diện người dùng, Ở đây mục tiêu của chúng ta là xây dựng và phát triển ứng dụng chat trên cả hai nền tảng phổ biến nhất hiện nay là android và web browser, vậy thì thông thường chúng ta sẽ cần viết riêng ứng dụng bằng ngôn ngữ java để có chạy trên nền tảng android và web.

## Và những ngôn ngữ này được gọi là ngôn ngữ native, và có một sự thật là ngôn ngữ native thì đêm lại cho chúng ta rất nhiều ưu điểm như là :

## Về mặt hiệu năng thì khi sử dụng ngôn ngữ để lập trình thì ứng dụng sẽ chắc chắn chạy nhanh hơn trong hầu hết trường hợp. Tuy nhiên sự chênh lệch ở các ứng dụng có thể là nhỏ, chấp nhận được và khó nhận biết bởi người dùng. Và có một sự thật là chỉ ngôn ngữ native mới có khả năng truy cập các phần cứng (như camera, GPS, thiết bị thu âm... ) và có thể truy cập đến các quyền như : sao chép, tạo, ghi, đọc tập tin (trên bộ nhớ của điện thoại), các thông tin danh bạ, thông tin cá nhân, thực hiện cuộc gọi, nhắn tin... Nói cách khác, khi sử dụng ngôn ngữ native thì ứng dụng có thể khai thác được sức mạnh phần cứng nhiều nhất của nền tảng.

Tuy nhiên thì với những dự án vừa và nhỏ thì công cuộc xây dựng ứng dụng kiểu này sẽ thực sự tốn kxm chi phí và thời gian cũng như công sức phát triển và bảo trì ứng dụng. Và với một dự án nhỏ về hệ thống chat như thế này, thì ngôn ngữ native chắc chắn không phải là ngôn ngữ tối ưu nhất để phát triển ứng dụng. Và sự lựa chọn của tôi trong dự án này chắc chắn phải là một nền tảng giúp tôi chỉ việc code một lần để sở hữu ứng dụng có thể vận hành trên cả hai nền tảng.

Sự thật là hiện nay có một vài framework khá tốt như react native, flux Và lựa chọn của nhóm trong ứng dụng này sẽ là React native,Chúng ta sẽ bàn về lý do chúng ta sử dụng react native ở phần sau. React native thì được viết trên ngôn ngữ javascript và đẩy cũng chính là điểm mạnh của framework này.

Vậy thì với react native và javascript chúng ta có thể dễ dàng xây dựng giao diện người dùng.

Còn lại bên phía máy chủ, có hai phần chính để xây dựng một hệ thống máy chủ hoàn chỉnh đó chính là api và cơ sở dữ liệu, và cũng tồn tại khá nhiều loại api như là graphql server, hay restful api , rest api .... Và để mỗi loại api thì có các ngôn ngữ, cách thực lập trình khác nhau. Và chọn loại api nào sẽ phụ thuộc vào mục đích sử dụng của chúng ta. Api sẽ có nhiệm vụ kết nối giữa phía người dùng và hệ thống cơ sở dữ liệu. Và có một vài cơ sở dữ liệu có thể kể đến như là hệ thống cơ sở quản lý dữ liệu quan hệ (Oracle, M, MS server, PostgreSQL), hệ cơ sở dữ liệu hướng tài liệu (MongoDB), Cơ sở dữ liệu dạng khoá ...

Ngoài ra thì chúng ta cũng cần phải có thư viện cung cấp các phương thức để phía người dùng có thể giao tiếp với api để lấy dữ liệu từ cơ sở dữ liệu. Cũng phải nói rằng mỗi loại api khác nhau thì có các cách để gửi http request khác nhau và các thư viện cung cấp các phương thức gửi http request khác nhau như là axios, apollo client ....

Tuy nhiên thì ở đây, Để tạo ra một ứng dụng chat có sử dụng dữ liệu thời gian thực, thì đòi hỏi phía máy chủ và phía máy khách phải liên tục gửi dữ liệu sang cho nhau với độ trễ thấp nhất có thể ( ms ). Vì vậy nên sự lựa chọn của tôi trong dự án này là cơ sở dữ liệu phi quan hệ ( MongoDB) Cùng với đó tôi sẽ sử dụng các phương thức cung cấp để phía máy khách có thể truy cập vào và lấy dữ liệu trực tiếp từ hệ thống cơ sở dữ liệu.

## GIỚI THIỆU CÁC CÔNG NGHỆ SỬ DỤNG CHO HỆ THỐNG CHAT

## Trước tiên là về phía giao diện người dùng, Tôi sẽ phân tích sâu hơn về framework react native, lý do sử dụng nó và ngôn ngữ được sử dụng đó chính là javascript.

## **2.3.1 Ngôn ngữ javascript**

## Ngôn ngữ lập trình javascript được giới thiệu đầu tiên vào năm 1995. Mục đích là để đưa những chương trình vào trang web ở trình duyệt Netscape Navigator - một trình duyệt web phổ biến những năm 1990.

## JavaScript được phát triển bởi Brendan Eich tại Hãng truyền thông Netscape với cái tên đầu tiên là Mocha, rồi sau đó đổi tên thành LiveScript, và cuối cùng thành xavaScript. Có lẽ việc đổi tên như vậy là để giúp javaScript được chú ý nhiều hơn. Bởi tại thời điểm này, java đang được coi là một hiện tượng và trở nên phổ biến.

Trên thực tế, javaScript không được phát triển dựa trên java. Và chúng là hai ngôn ngữ hoàn toàn khác biệt. Các ta sẽ thấy rõ hơn điều này ở phần sau .

Đặc điểm chung của javascript

Là ngôn ngữ lập trình bậc cao (high-level) giống như: C/C++, xava, Python, Ruby,... Nó rất gần với ngôn ngữ tự nhiên của con người. Trong khi ngôn ngữ lập trình bậc thấp (low-level) như: Assembly... sẽ gần với máy tính hơn.

Là ngôn ngữ lập trình động (dynamic programming language): như Python, Ruby, Perl,... Chúng được tối ưu hoá nhằm nâng cao hiệu suất cho lập trình viên. Trong khi ngôn ngữ lập trình tĩnh (static programming language): như C/C++,... lại được tối ưu hoá để nâng cao hiệu suất cho phần cứng máy tính.

Là ngôn ngữ lập trình kịch bản (scripting language): nghĩa là không cần biên dịch (compile) hay liên kết (linked) giống như ngôn ngữ lập trình biên dịch (C/C++, xava,...) mà nó sẽ được dịch tại thời điểm chạy.

Là ngôn ngữ dựa trên đối tượng (object-based): tức nó gần giống như ngôn ngữ lập trình hướng đối tượng, ngoại trừ xavaScript không hỗ trợ tính kế thừa và đa hình.

Là ngôn ngữ dựa trên nguyên mẫu (prototype-based): là một kiểu của lập trình hướng đối tượng, trong đó các hành vi của đối tượng được sử dụng lại.

javaScript có rất nhiều ứng dụng trên nhiều môi trường và nền tảng khác khau:

javaScript cùng với HTML, CSS trở thành ngôn ngữ không thể thiếu đối với website

Ngoài ra, có rất nhiều framework javascript khác nhau phía front-end: Angular.js, Angular2, ReactxS, Vue.js, Ember.js, Meteor.js, backbone...

javaScript có thể sử dụng phía server với framework : Node.js

Một số database sử dụng javaScript như là kịch bản và là ngôn ngữ query: MongoDB, CouchDB

javaScript có thể dùng để xây dựng ứng dụng Desktop với framework: Electron. Những ứng dụng nổi tiếng có thể kể đến là: Atom, Visual Studio Code, GitKraken, Wordpress.com,…

**2.3.2 Công nghệ React Native**

React Native là công nghệ được tạo bởi Facebook, cho phxp các lập trình viên sử dụng xavaScript để làm mobile apps trên cả Android và iOS với cảm nhận và giao diện native. Một ví dụ để làm rx cách vận hành của React Native là các code bọc ngoài của code native. Có rất nhiều thành tổ được tạo nên khi đang xbọc” chức năng native của iOS hoặc Android.

Một lý do lập trình mobile app rất khó và tốn thời gian là vì thực tế ta cần tìm hiểu 2 hệ sinh thái hoàn toàn khác biệt. Nếu muốn lập trình app iOS, chúng ta cần phải học Swift hoặc Objective-C và Cocoa Pods. Nếu muốn lập trình app Android, Ta cần học java và Android SDK.

Tuy nhiên, điều tôi có thể nói là chúng khác nhau và việc học từng ngôn ngữ đó sẽ tốn khá nhiều thời gian. Điều tương tự cũng xảy ra với các frameworks: Cocoa Touch và Android SDK.

Tất nhiên, mỗi frameworks luôn có 1 gói các công cụ như công cụ testing, các libs, packages... và việc các dev phải cập nhật các tính năng mới nhất của mỗi hệ sinh thái là điều không thể bàn cãi.

Mặc khác, nếu chúng ta chọn lập trình trên React Native, phần lớn thời gian chúng ta sẽ chỉ cần học 1 bộ công cụ. Có rất nhiều thứ để làm quen như: javaScript, Node, React Native... nhưng chỉ có 1 công cụ duy nhất để học.

Chúng ta sẽ build được ứng dụng Native, và chúng ta cũng có thể build ứng dụng đó một cách đa nền tảng (multi-platform) chứ không phải là một xmobile web app”, không phải là xHTML5 app”, và cũng không phải là một xhybrid app” hay cũng không chỉ build trên iOS hay Android mà chúng ta build và chạy được cả hai hệ sinh thái .

Một điểm hay ho nữa mà tôi có đề cập là giảm chi phí tái biên dịch của Native bằng cách sử dụng Hot-Loading tức là ta không cần phải build lại ứng dụng từ đầu nên việc chỉnh sửa diễn ra rất nhanh chóng. Giúp cho lập trình viên có thể thấy được những chỉnh sửa của họ một cách nhanh chóng trực quan, không còn phải bỏ quá nhiều thời gian trong việc build và chạy ứng dụng nữa.

Khả năng sử dụng lại đóng vai trò quan trọng trong lập trình phần mềm, nên mỗi khi ta có thể sử dụng lại code thì React Native là công cụ tốt.

**2.3.3 Công nghệ NodeJS**

NodeJS là một môi trường chạỵ javascript đa nền tảng và có mã nguồn mở, được sử dụng để chạy các ứng dụng web bên ngoài trình duyệt của client. Nền tảng này được phát triển bởi Ryan Dahl vào năm 2009, được xem là một giải pháp hoàn hảo cho các ứng dụng sử dụng nhiều dữ liệu nhờ vào mô hình hướng sự kiện không đồng bộ.

Node.js là một Framework mạnh mẽ dựa trên nền tảng Google Chrome Javascript V8 Engine. Node.js được dùng để phát triển các ứng dụng hướng nhiều đến I/O trên nền tảng web như Video Streaming Site, các ứng dụng Single-page App hay các ứng dụng web khác.

NodeJs hoàn toàn mã nguồn mở, miễn phí và được sử dụng rất rộng rãi phổ biến hiện nay.

Đây là cách PHP hoặc ASP xử lý một yêu cầu tệp:

1. Gửi tác vụ đến hệ thống tệp của máy tính.

2. Chờ trong khi hệ thống tệp mở và đọc tệp.

3. Trả về nội dung cho khách hàng.

4. Sẵn sàng để xử lý các yêu cầu tiếp theo.

Đây là cách Node.js xử lý yêu cầu tập:

1. Gửi tác vụ đến hệ thống tệp của máy tính.

2. Sẵn sàng để xử lý các yêu cầu tiếp theo.

3. Khi hệ thống tệp đã mở và đọc tệp, máy chủ sẽ trả lại nội dung cho máy khách.

Node.js bỏ phần chờ đợi và tiếp tục bước tiếp theo, nên tốc độ nhanh hơn và hiệu quả về bộ nhớ.

**2.3.4 Công nghệ Socket.IO**

Socket.IO được xem là 1 module trong Node.js được nhà sáng chế tạo ra và phát triển từ năm 2010. Mục đích lớn nhất của Socket io là để tạo môi trường giao tiếp thuận lợi trên Internet giúp trả về các giá trị thực ngay tại thời điểm giao tiếp giữa các bên với nhau (thường là giữa server và client. Việc giao tiếp 2 chiều giữa máy khách và máy chủ được thực hiện bởi socket io khi và chỉ khi máy khách có module này trong trình duyệt và máy chủ cũng đã tích hợp sẵn gói socket io. Các ứng dụng sử dụng socket io thường đòi hỏi tốc độ phản hồi ngay lập tức. Một số ví dụ điển hình như xổ số, trực tiếp bóng đá, chat…

Socket.IO không phải là mô hình Socket duy nhất hiện nay, và cũng không phải là mô hình web socket duy nhất hiện nay. Socket là cách bạn tổ chức mô hình client-server để một trong 2 bên luôn trong tình trạng sẵn sàng trả lời bên kia và ngược lại. Để đảm bảo việc này, kết nối giữa Client và Server phải ở trạng thái “keep-alive” và phải luôn xảy ra quá trình đồng bộ giữa Client-Server. Socket sẽ mang lại khả năng trả lời tức thì từ một trong 2 bên khi bên kia đưa ra một sự kiện, thay vì phải thực thi lại một loạt các thủ tục kết nối phức tạp như trước, và ứng dụng của bạn sẽ trở thành ứng dụng thời gian thực ví dụ: Yahoo Messenger, Skype v.v… đều là các ứng dụng được xây dựng theo mô hình Socket.

**2.3.5 Công nghệ React .JS**

React.JS là một thư viện Javascript dùng để xây dựng giao diện người dùng, React hỗ trợ việc xây dựng những thành phần (components) UI có tính tương tác cao, có trạng thái và có thể sử dụng lại được.

Khi sử dụng react JS người dùng có thể khai thác được nhiều tính năng khác nhau, trong đó phải kể đến:

Viết ứng dụng trực tiếp trên JavaScript

Phá vỡ những cấu trúc UI phức tạp, biến chúng trở thành các component độc lập.

Chuyển các dữ liệu đã được tùy biến đến một UI component cụ thể

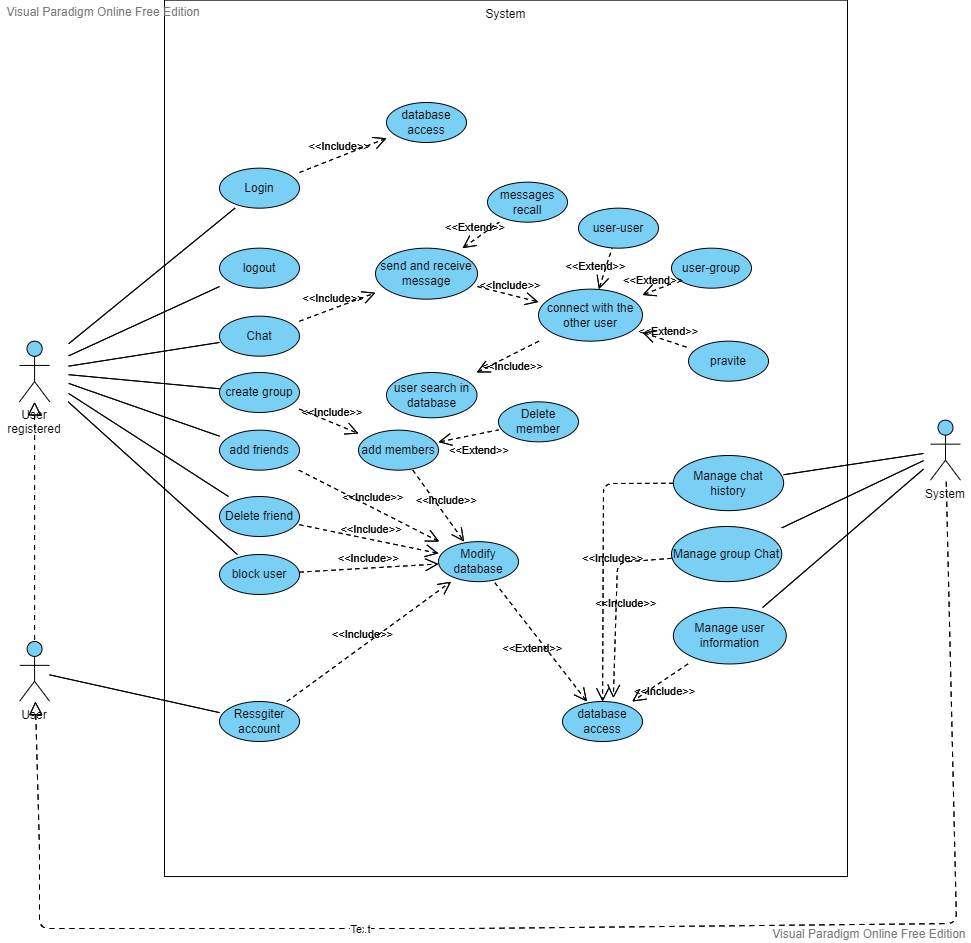
Thay đổi trạng thái cho nhiều component (child) trên ứng dụng nhưng không làm ảnh hưởng tới các component gốc (parent) đang ở trạng thái Stateful.

Biết được chính xác khi nào cần render lại hoặc khi nào cần bỏ đi các DOM chính.

## 

# **PHÂN TÍCH VÀ THIẾT KẾ**

## Mô hình Use case

**

## Danh sách các tác nhân và mô tả

|  |  |  |
| --- | --- | --- |
| Tác nhân | Mô tả tác nhân | Ghi chú |
| User | Người dùng (User) khi chưa có tài khoản có thể thực hiện chức năng đăng ký tài khoản mới (register account) |  |
| User registered | Người dùng đã có tài khoản (User registered ) là người thực hiện chức năng: đăng nhập, đăng xuất, chat, tạo nhóm, thêm bạn, …. |  |
|  |  |  |

## Danh sách Use case và mô tả

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ID | Tên Use case | Mô tả ngắn gọn Use case | Chức năng | Ghi chú |
| UC001 | Đăng ký tài khoản | Use case mô tả các bước đăng ký một tài khoản | Đăng ký |  |
| UC002 | Đăng nhập tài khoản | Use case mô tả cách đăng nhập trên ứng dụng | Đăng nhập |  |
| UC003 | Đăng xuất | Use case mô tả các bước đăng xuất tài khoản trên ứng dụng | Đăng xuất |  |
| UC004 | Gửi tin nhắn | Use case mô tả các bước để gửi tin nhắn trên ứng dụng | Gửi tin nhắn |  |
| UC005 | Tạo nhóm chat | Use case mô tả các bước để tạo một nhóm chat trên ứng dụng | Tạo mới |  |
| UC006 | Thêm bạn bè | Use case mô tả các bước kết bạn trên ứng dụng | Cập nhật |  |
| UC007 | Hủy kết bạn | Use case mô ta các bước hủy kết bạn trên ứng dụng | Cập nhật |  |
| UC008 | Chặn người dùng | Use case mô tả các bước chặn một người dùng trên ứng dụng | Cập nhật |  |
| UC009 | Thêm thành viên vào nhóm | Use case mô tả các bước thêm một thành viên vào nhóm chat trên ứng dụng | Cập nhật |  |
| UC0010 | Xóa thành viên ra khỏi nhóm chat | Use case mô tả các bước xóa một thành viên ra khỏi nhóm chat trên ứng dụng | Cập nhật |  |
|  |  |  |  |  |

# **ĐẶC TẢ CÁC YÊU CẦU CHỨC NĂNG (FUNCTIONAL)**

Xác định yêu cầu chức năng:

* User tạo một tài khoản mới
* User registered đăng nhập vào hệ thống
* User registered đăng xuất ra khỏi hệ thống
* User registered thêm bạn bè mới
* User registered hủy kết bạn
* User registered gửi tin nhắn chat
* User registered tạo nhóm chat mới
* User registered chặn người dùng
* User registered thêm thành viên vào nhóm
* User registered xóa thành viên ra khỏi nhóm

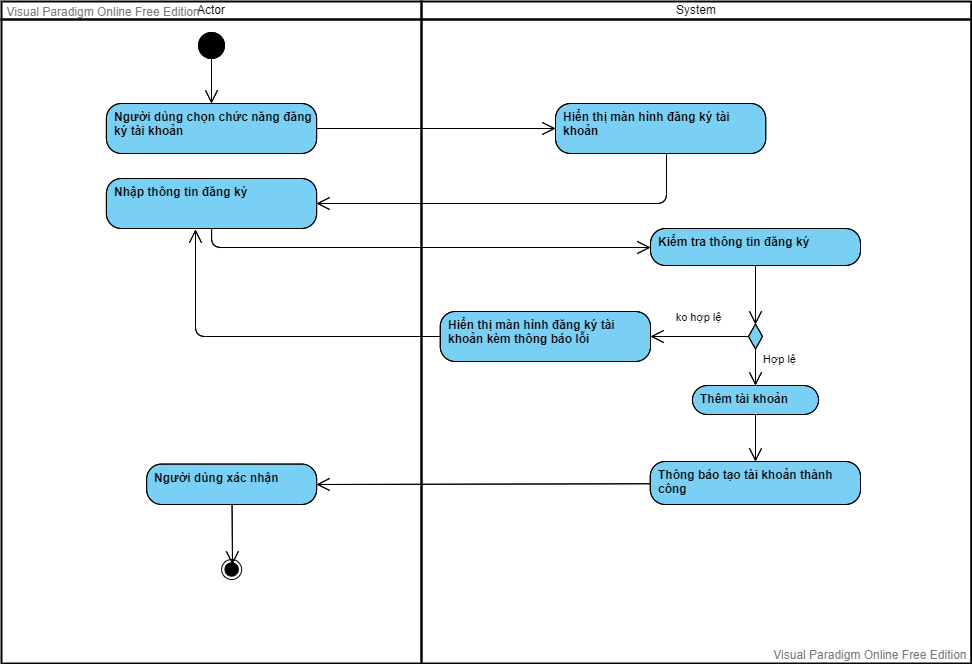
## UC001\_Đăng ký tài khoản

### Mô tả use case UC001

|  |  |  |  |
| --- | --- | --- | --- |
| **Use case: Tìm kiếm tình trạng sản phẩm** | | | |
| Mục đích: | Giúp người dùng đăng ký tài khoản để đăng nhập vào hệ thống | | |
| Mô tả: | Use case mô tả các bước đăng ký tài khoản | | |
| Tác nhân: | User | | |
| Điều kiện trước: | User phải có số điện thoại và email | | |
| Điều kiện sau: | Nếu tạo tài khoản thành công người dùng có thể đăng nhập vào hệ thống | | |
| Luồng sự kiện chính (Basic flows): | | | |
| Actor | | System | |
| 1. Người dùng chọn chức năng đăng ký tài khoản | |  | |
|  | | 1. Hệ thống hiển thị màn hình đăng ký tài khoản | |
| 1. Người dùng nhập thông tin đăng ký | |  | |
|  | | 1. Hệ thống kiểm tra thông tin đăng ký | |
|  | | 1. Hệ thống thêm thông tin tài khoản | |
|  | | 1. Hệ thống thông báo tạo tài khoản thành công | |
| 1. Người dùng xác nhận | |  | |
| Luồng sự kiện phụ (Alternative Flows): | | | |
|  | | | 4.1 Hệ thống hiển thị màn hình đăng ký kèm thông báo lỗi |
| 4.2 Người dùng nhập thông tin đăng ký | | |  |
|  | | |  |

### Biểu đồ

1. Sơ đồ activity



1. Sơ đồ Senquence

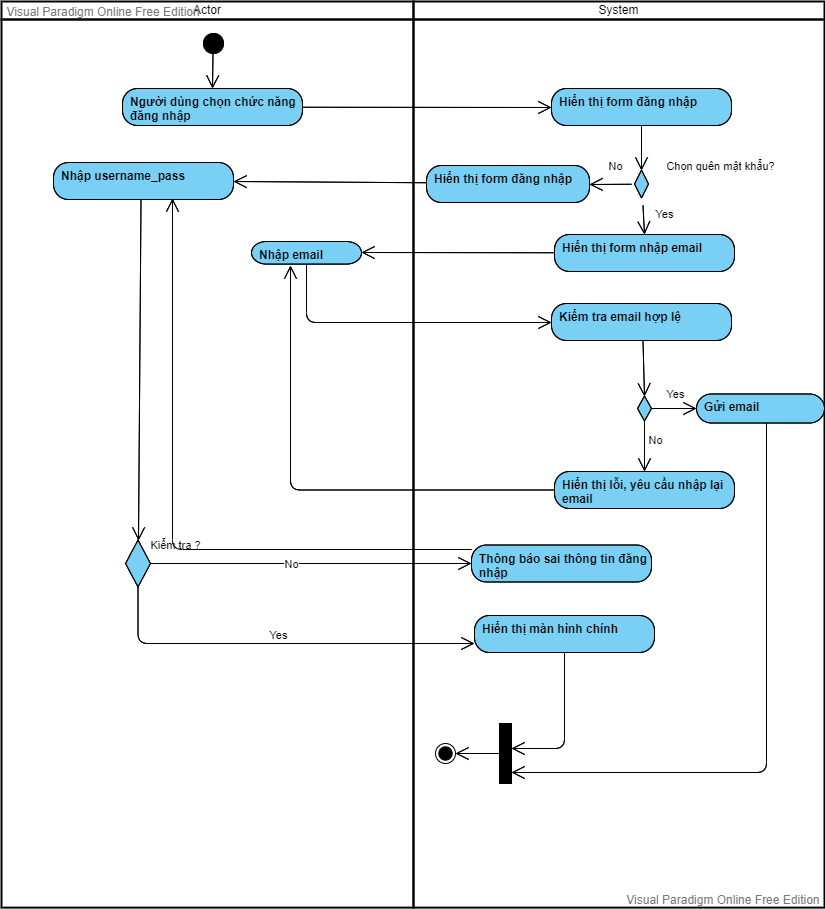
## UC002\_Đăng nhập tài khoản

### Mô tả use case UC002

|  |  |  |  |
| --- | --- | --- | --- |
| **Use case: Cập nhật tình trạng sản phẩm** | | | |
| Mục đích: | Giúp người dùng đăng nhập vào hệ thống | | |
| Mô tả: | Use case mô tả các bước đăng nhập vào hệ thống | | |
| Tác nhân: | User | | |
| Điều kiện trước: | Phải có tài khoản của hệ thống trước đó | | |
| Điều kiện sau: | Sau khi đăng nhập thành công thì hệ thống sẽ phân quyền sử dụng chức năng ứng dụng . | | |
| Luồng sự kiện chính (Basic flows): | | | |
| Actor | | System | |
| 1. Người dùng chọn chức năng đăng nhập | |  | |
|  | | 1. Hệ thống hiển thị form đăng nhập | |
| 1. Người dùng nhập tên đăng nhập và mật khẩu | |  | |
|  | | 1. Kiểm tra hợp lệ | |
|  | | 1. Hệ thống hiển thị màn hình chính | |
| Luồng sự kiện phụ (Alternative Flows): | | | |
| 3.1 Người dùng chọn quên mật khẩu | | | 3.2 Hệ thống hiển thị form nhập email |
| 3.4 Người dùng nhập email | | | 3.4 Kiểm tra hợp lệ |
|  | | | 3.5 Gửi email |
|  | | |  |
|  | | | 5.1 Hệ thống thông báo sai thông tin đăng nhập |
| 5.2 Quay lại bước 3 | | |  |
|  | | |  |
|  | | | 3.5.1 Hiển thị lỗi yêu cầu nhập lại email |
| 3.5.2 Quay lại bước 3.4 | | |  |
|  | | |  |

### Biểu đồ

1. Sơ đồ activity



1. Sơ đồ Sequence

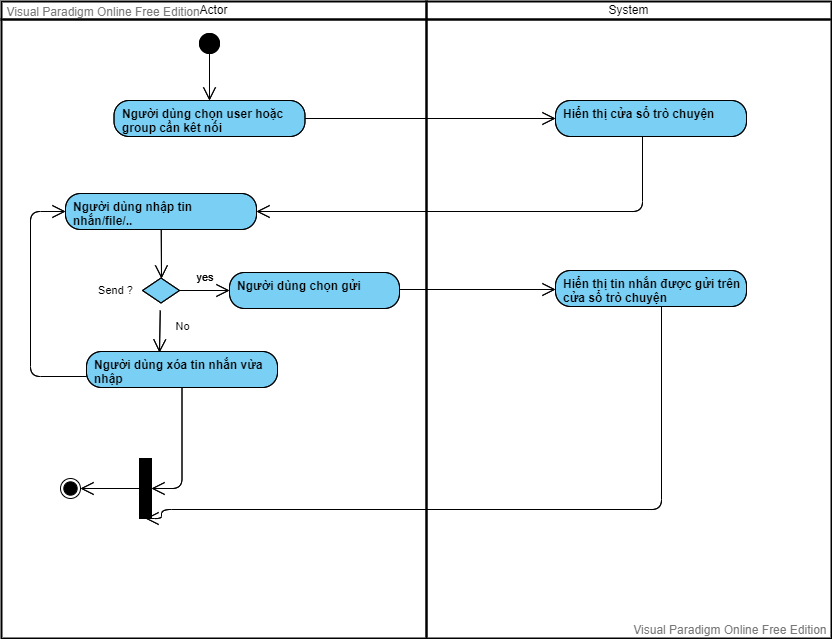
## UC003\_gửi tin nhắn Chat

### Mô tả use case UC003

|  |  |  |  |
| --- | --- | --- | --- |
| **Use case: Gửi tin nhắn chat** | | | |
| Mục đích: | Giúp user gửi tin nhắn văn bản, file,… tới nhóm chat hay user khác | | |
| Mô tả: | Use case mô tả các bước gửi tin nhắn chat | | |
| Tác nhân: | User | | |
| Điều kiện trước: | User phải đăng nhập vào hệ thống | | |
| Điều kiện sau: | Tin nhắn được gửi đi từ phía người dùng sẽ được nhìn thấy phía bên người nhận | | |
| Luồng sự kiện chính (Basic flows): | | | |
| Actor | | System | |
| 1. Người dùng chọn người dùng hoặc nhóm chat cần kết nối | |  | |
|  | | 1. Hệ thống hiển thị cửa sổ trò chuyện | |
| 1. Người dùng nhập tin nhắn/ file/… | |  | |
| 1. Người dùng chọn gửi | |  | |
|  | | 1. Hệ thống hiển thị tin nhắn được gửi trên phía cửa sổ trò chuyện | |
| Luồng sự kiện phụ (Alternative Flows): | | | |
| 4.1. Người dùng xóa tin nhắn vừa nhập | | |  |
|  | | | 4.2 Quay lại bước 2 |

### Biểu đồ

1. Sơ đồ activity



1. Sơ đồ Sequence

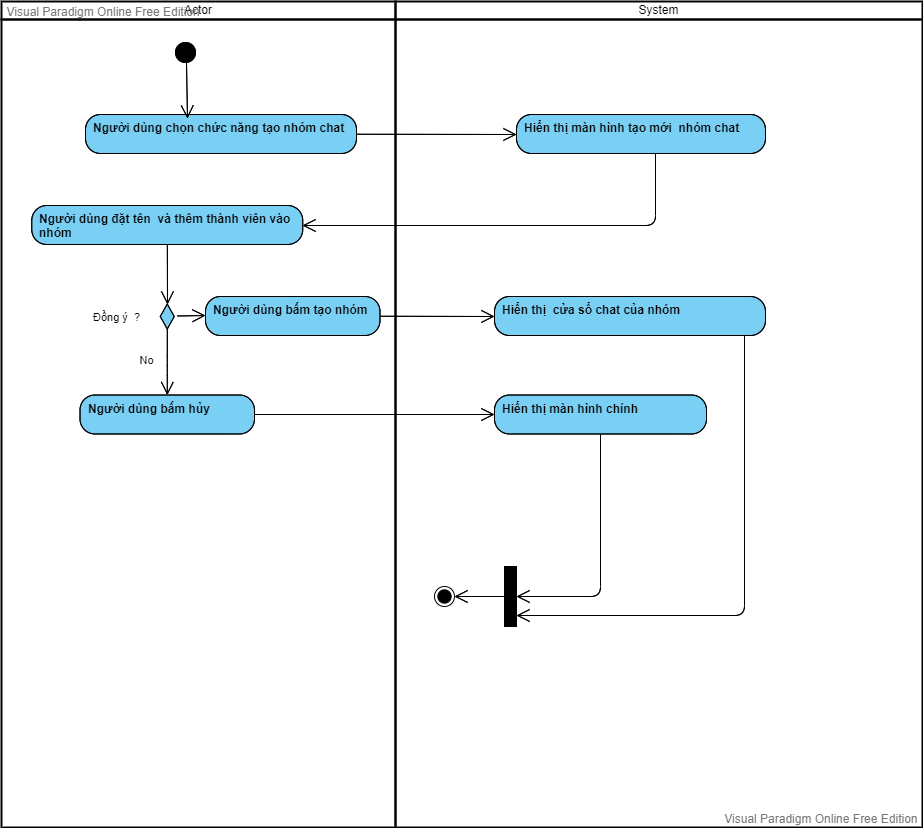
### 3.4 UC004\_Tạo nhóm chat

### 3.4.1 Mô tả Use case UC004

|  |  |  |  |
| --- | --- | --- | --- |
| **Use case: Tạo nhóm chat** | | |  |
| Mục đích: | Giúp cho người dùng tạo nhóm trò chuyện nhiều thành viên | | |
| Mô tả: | Use case mô tả các bước tạo một nhóm chat mới | | |
| Tác nhân: | User | | |
| Điều kiện tiên quyết: | User phải đăng nhập vào hệ thống | | |
| Điều kiện sau: | Sau khi tạo nhóm chat thành công các thành viên trong nhóm có thể gửi tin nhắn, file cho nhau. | | |
| Luồng sự kiện chính: | |  | |
| Actor | | | System |
| 1. Người dùng chọn chức năng tạo nhóm chat | | |  |
|  | | | 2. Hệ thống hiển thi màn hình tạo mới nhóm chat |
| 3. Người dùng đặt tên nhóm và thêm thành viên vào nhóm | | |  |
|  | | |  |
| 4. Người dùng chọn tạo nhóm | | |  |
|  | | | 5. Hệ thống hiển thị cửa sổ chat của nhóm |
| Luồng sự kiện thay thế: | | |  |
| 4.1. Người dùng chọn hủy | | |  |
|  | | | 7.2. Hệ thống hiển thị màn hình chính |

### 3.4.2 Sơ đồ

1. Sơ đồ activity



1. Sơ đồ sequence

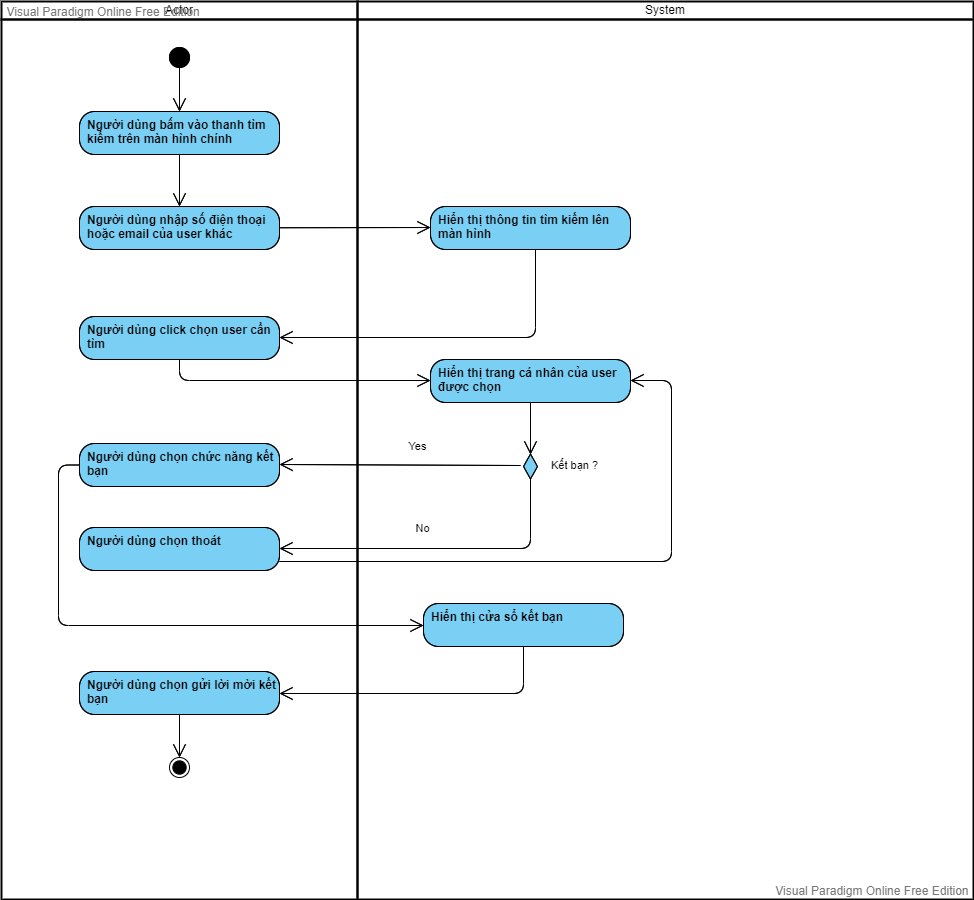
### 3.5 UC005\_Thêm bạn bè

### 3.5.1 Mô tả Use case UC005

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Use case: Thêm bạn bè** | | |  | |
| Mục đích: | Giúp người dùng thêm liên kết với người dùng khác | | | |
| Mô tả: | Use case mô tả các bước thêm bạn bè | | | |
| Tác nhân: | User | | | |
| Điều kiện tiên quyết: | Người dùng phải đăng nhập vào hệ thống | | | |
| Điều kiện sau: | Sau khi kết bạn thành công người dùng có thể gửi tin nhắn cho nhau | | | |
| Luồng sự kiện chính: | |  | | |
| Actor | | | | System |
| 1. Người dùng bấm vào thanh tìm kiếm trên màn hình chính | | | |  |
| 1. Người dùng nhập số điện thoại hoặc email của user khác | | | |  |
|  | | | | 1. Hệ thống hiển thị thông tin tìm kiếm trên màn hình |
| 1. Người dùng click chọn user cần tìm | | | |  |
|  | | | | 1. Hệ thống hiển thị trang cá nhân của user được chọn |
| 1. Người dùng chọn chức năng kết bạn | | | |  |
|  | | | | 1. Hệ thống hiển thị cửa sổ kết bạn |
| 1. Người dùng chọn gửi lời mời kết bạn | | | |  |
| Luồng sự kiện thay thế: | | | |  |
| 6.1 Người dùng chọn thoát | | | |  |
|  | | | | 6.2 Quay lại bước 5 |

### 3.5.2 Sơ đồ

1. Sơ đồ activity



1. Sơ đồ Sequence

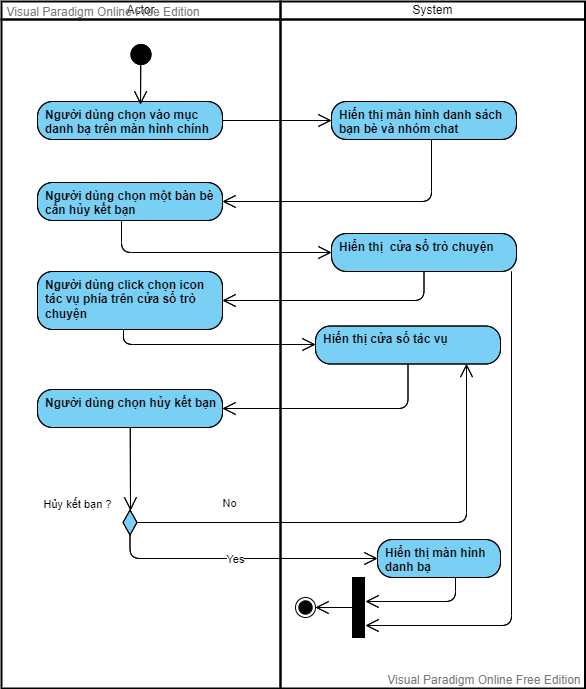
### 3.6 UC006\_Hủy kết bạn

### 3.6.1 Mô tả Use case UC006

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Use case: Hủy kết bạn** | | |  | |
| Mục đích: | Giúp cho người dùng có thể xóa một user khỏi danh sách bạn bè | | | |
| Mô tả: | Use case mô tả các bước hủy kết bạn | | | |
| Tác nhân: | User | | | |
| Điều kiện tiên quyết: | Người dùng phải đăng nhập vào hệ thống | | | |
| Điều kiện sau: | Sau khi người dùng xóa một người dùng khác ra khỏi danh sách bạn bè thì hệ thống sẽ cập nhật lại danh sách bạn bè | | | |
| Luồng sự kiện chính: | |  | | |
| Actor | | | | System |
| 1. Người dùng chọn vào mục danh bạ trên màn hình chính | | | |  |
|  | | | | 1. Hệ thống hiển thị danh sách bạn bè và nhóm chat |
| 1. Người dùng chọn một bạn bè cần hủy kết bạn | | | |  |
|  | | | | 1. Hệ thống hiển thị cửa sổ trò chuyện |
| 1. Người dùng click chọn icon tác vụ phía trên cửa sổ trò chuyện | | | |  |
|  | | | | 1. Hệ thống hiển thị cửa sổ tác vụ |
| 1. Người dùng chọn hủy kết bạn | | | |  |
|  | | | | 1. Hệ thống hiển thị màn hình danh bạ |
| Luồng sự kiện thay thế: | | | |  |
| 7.1 Người dùng chọn hủy | | | |  |
|  | | | | 7.2 Quay lại bước 6 |

### 3.6.2 Sơ đồ

1. Sơ đồ activity



1. Sơ đồ Sequence

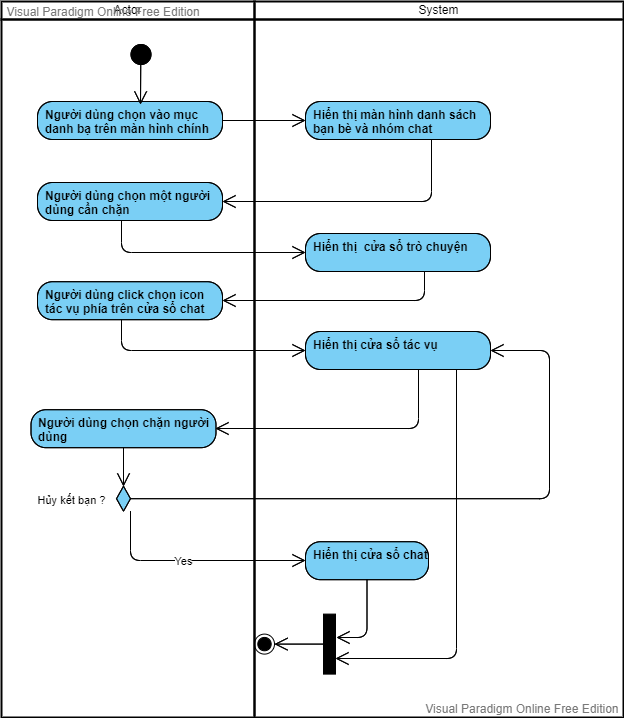
### 3.7 UC007\_Chặn người dùng

### 3.7.1 Mô tả Use case UC007

|  |  |  |
| --- | --- | --- |
| **Use case: Chặn người dùng** | |  |
| Mục đích: | Giúp cho người dùng chặn một người dùng | |
| Mô tả: | Use case mô tả các bước chặn một người dùng | |
| Tác nhân: | Người dùng | |
| Điều kiện tiên quyết: | Người dùng phải đăng nhập vào hệ thống | |
| Điều kiện sau: | Sau khi người dùng chặn một user khác thì user bị chặn không thể trả lời tin nhắn của user chặn được nữa | |
| Luồng sự kiện chính | |  |
| Actor | | System |
| 1. Người dùng chọn vào mục danh bạ trên màn hình chính | |  |
|  | | 2. Hệ thống hiển thị danh sách bạn bè và nhóm chat |
| 3. Người dùng chọn một user cần chặn | |  |
|  | | 4. Hệ thống hiển thị cửa sổ trò chuyện |
| 5. Người dùng click chọn icon tác vụ phía trên cửa sổ chat | |  |
|  | | 6. Hệ thống hiển thị cửa sổ tác vụ |
| 7. Người dùng chọn chặn người dùng | |  |
|  | | 8. Hệ thống hiển thị cửa sổ chat |
| Luồng sự kiện thay thế | |  |
| Actor | | System |
| 7.1. Xác nhận không chặn | |  |
|  | | 6.2. Quay lại bước 6 |

### 3.7.2 Sơ đồ

1. Sơ đồ activity



1. Sơ đồ Sequence

### 3.8 UC008\_Thêm thành viên vào nhóm

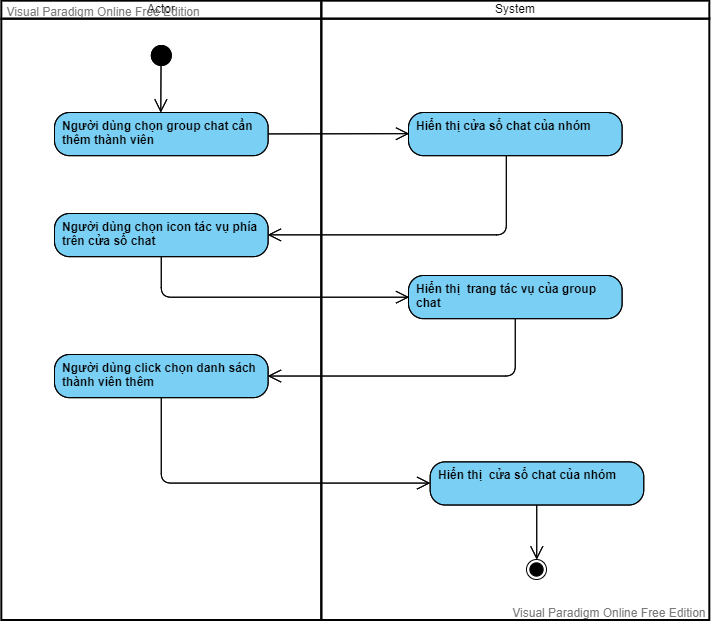
### 3.8.1 Mô tả Use case UC008

|  |  |
| --- | --- |
| **Tên Use case: Thêm thành viên vào nhóm** | |
| Miêu tả Use Case: | Giúp người dùng có thể thêm thành viên vào nhóm chat |
| Actor: | User |
| Điều kiện tiên quyết: | Người dùng phải đăng nhập vào hệ thống |
| Điều kiện hậu kỳ: | Sau khi thêm thành viên vào nhóm chat thành công thì số lượng thành viên sẽ được cập nhật |

|  |  |
| --- | --- |
| **Luồng sự kiện chính:** | |
| Actor | Hệ thống |
| 1/ Người dùng chọn nhóm chat cần thêm thành viên |  |
|  | 2/ Hệ thống hiển thị cửa sổ chat |
| 3/ Người dùng chọn icon tác vụ phía trên cửa sổ chat |  |
|  | 4/ Hệ thống hiện thị cửa sổ tác vụ |
| 5/ Người dùng click chọn danh sách thành viên thêm |  |
|  | 6/ Hệ thống hiển thị cửa sổ chat của nhóm |
| **Luồng sự kiện thay thế:** | |
|  |  |
| **Luồng sự kiện ngoại lệ:** | |

### 3.8.2 Sơ đồ

1. Sơ đồ activity



1. Sơ đồ Sequence

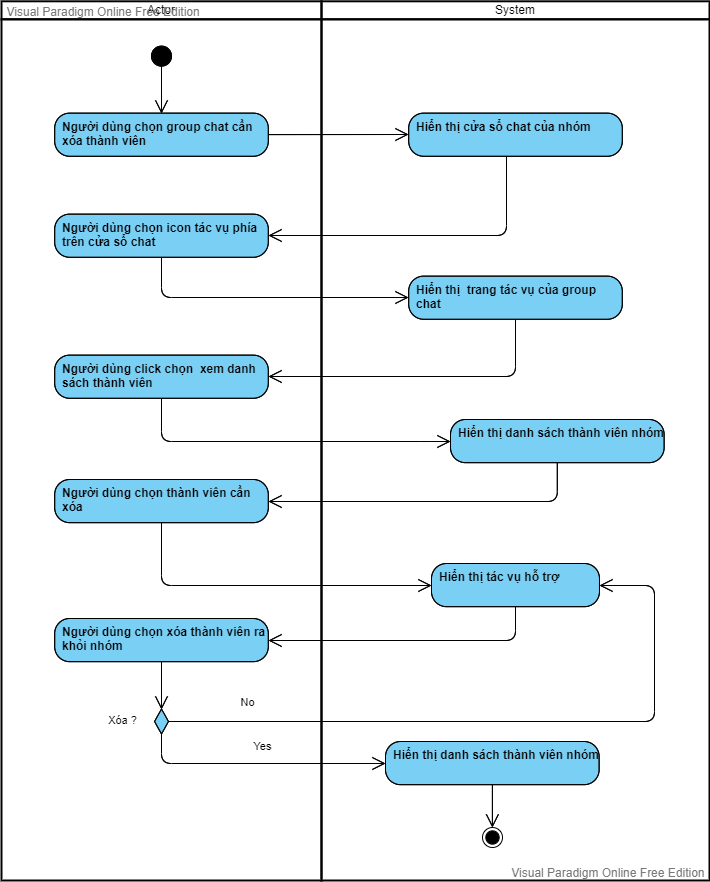
### 3.9 UC009\_Xóa thành viên ra khỏi nhóm

### 3.9.1 Mô tả Use case UC009

|  |  |
| --- | --- |
| **Tên Use Case: Xóa thành viên ra khỏi nhóm** | |
| Miêu tả Use Case: | Giúp người dùng có thể xóa một thành viên ra khỏi nhóm |
| Actor: | User |
| Điều kiện tiên quyết: | Người dùng phải đăng nhập vào hệ thống |
| Điều kiện hậu kỳ: | Sau khi xóathành viên ra khỏi nhóm chat thành công thì số lượng thành viên sẽ được cập nhật |

|  |  |
| --- | --- |
|  | |
| 1/ Người dùng chọn nhóm chat cần xóa thành viên |  |
|  | 2/ Hệ thống hiển thị cửa sổ chat |
| 3/ Người dùng chọn icon tác vụ phía trên cửa sổ chat |  |
|  | 4/ Hệ thống hiển thị cửa sổ tác vụ |
| 5/ Người dùng click chọn xem danh sách thành viên |  |
|  | 6/ Hệ thống hiển thị danh sách thành viên |
| 7/ Người dùng chọn thành viên cần xóa |  |
|  | 8/ Hệ thống hiển thị tác vụ hỗ trợ |
| 9/ Người dùng chọn xóa thành viên ra khỏi nhóm |  |
|  | 10/ Hệ thống hiển thị danh sách thành viên nhóm |
| **Luồng sự kiện thay thế:** | |
| 9.1/ Người dùng xác nhận hủy |  |
|  | 9.2/ Quay lại bước 8 |

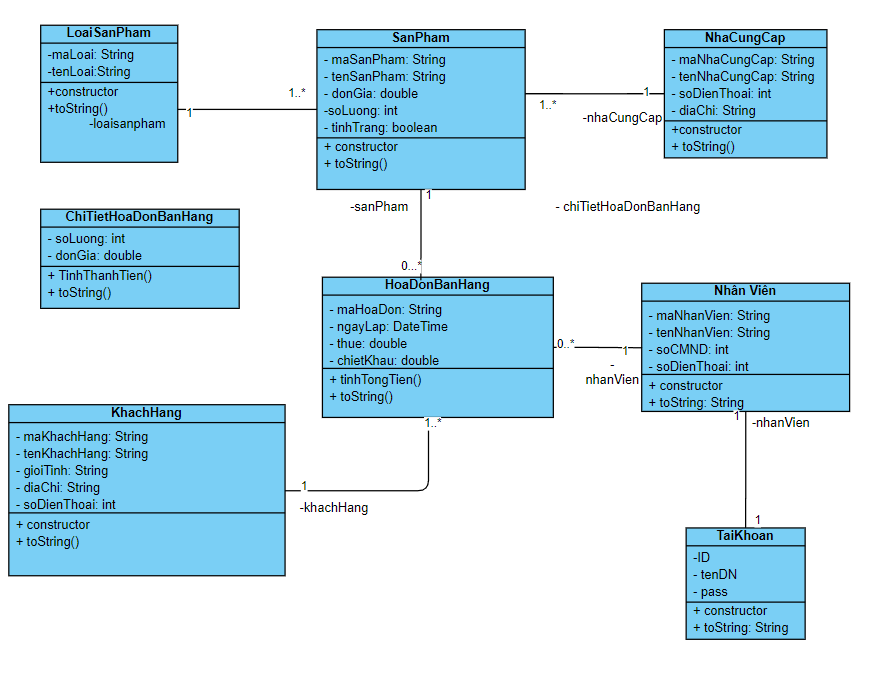
1. Sơ đồ activity



1. Sơ đồ Sequence

# **CÁC THÔNG TIN HỖ TRỢ KHÁC**

**4.1. Sơ đồ lớp (entity)**



1. **HIỆN THỰC**

## Cấu hình phần cứng, phần mềm

## Giao diện của hệ thống

1. **HIỆN THỰC**

**6.1 Kết quả đạt được**

**6.2 Hạn chế của đồ án**

**6.3 Hướng phát triển**

**TÀI LIỆU THAM KHẢO**