

**TRƯỜNG ĐẠI HỌC CÔNG NGHIỆP HÀ NỘI KHOA CÔNG NGHỆ THÔNG TIN**

**---**🙤🕮🙦**---**

ĐỒ ÁN TỐT NGHIỆP ĐẠI HỌC

NGÀNH KỸ THUẬT PHẦN MỀM

**Xây dựng website quản lý gia phả cho dòng họ Ngô**

|  |  |
| --- | --- |
| **GVHD** | TS. Nguyễn Thị Mỹ Bình |
| **Sinh viên** | Ngô Kim Đông |
| **Mã sinh viên** | 2020604782 |
| **Lớp** | KTPM03 – K15 |

**Hà Nội, Năm 2024**

# LỜI CẢM ƠN

*Lời đầu tiên, em xin gửi lời cảm ơn chân thành nhất tới cô Nguyễn Thị Mỹ Bình. Trong suốt quá trình học tập và thực hiện đề tài này, chúng em đã được nhận sự chú ý, hướng dẫn tận tâm và tâm huyết từ cô. Những điều mà chúng em nhận được không chỉ giới hạn ở kiến thức chuyên ngành, mà còn vượt ra ngoài điều đó - đó là những lời khuyên, chia sẻ thực tế từ thầy. Phương pháp giảng dạy của thầy đã giúp chúng em khám phá và phát triển tiềm năng của bản thân. Với sự quan tâm, dạy dỗ, chỉ bảo tận tình chu đáo của cô , đến nay em đã có thể hoàn thành bài báo cáo đồ án tốt nghiệp, đề tài: “Xây dựng Website quản lý gia phả cho dòng họ Ngô”.*

*Em cũng xin gửi lời cảm ơn đến Ban giám hiệu nhà trường và các thầy cô tại khoa Công nghệ Thông tin đã tạo điều kiện thuận lợi về môi trường học tập, cơ sở vật chất và các nguồn tài liệu hữu ích, giúp em có thể hoàn thành tốt chương trình học và đề tài tốt nghiệp này.*

*Bên cạnh đó, em muốn bày tỏ lòng biết ơn tới gia đình và bạn bè – những người luôn ủng hộ, động viên em cả về tinh thần lẫn vật chất trong suốt thời gian học tập và thực hiện đồ án. Sự khích lệ từ họ là chỗ dựa vững chắc để em tiếp tục cố gắng và hoàn thành mục tiêu của mình.*

*Cuối cùng, em xin gửi lời cảm ơn tới các bạn cùng lớp, những người đã không ngại chia sẻ kiến thức, góp ý chân thành, và cùng em vượt qua những thử thách trong quá trình thực hiện đồ án.*

*Dù đã cố gắng hoàn thành dự án một cách tốt nhất, nhưng chắc chắn sẽ không tránh khỏi những thiếu sót. Em mong nhận được những góp ý quý báu từ thầy cô và các bạn để cải thiện và phát triển dự án trong tương lai.*

Em xin chân thành cảm ơn!

Ngô Kim Đông

**MỤC LỤC**

[LỜI CẢM ƠN 1](#_Toc185550456)

[DANH MỤC HÌNH ẢNH 4](#_Toc185550457)

[DANH MỤC BẢNG BIỂU 6](#_Toc185550458)

[CHƯƠNG 1. GIỚI THIỆU VỀ ĐỀ TÀI 7](#_Toc185550459)

[1.1 Lý do chọn đề tài 7](#_Toc185550460)

[1.2 Mục tiêu đề tài 7](#_Toc185550461)

[1.3 Phạm vi nghiên cứu 8](#_Toc185550462)

[1.4 Công nghệ sử dụng 9](#_Toc185550463)

[1.4.1 VUEJS 9](#_Toc185550464)

[1.4.2 CSS - Bootstrap 10](#_Toc185550465)

[1.4.3 C# 11](#_Toc185550466)

[1.4.4 MongoDB 12](#_Toc185550467)

[1.5 Kết quả dự kiến 14](#_Toc185550468)

[1.6 Kết luận 14](#_Toc185550469)

[CHƯƠNG 2. KHẢO SÁT, PHÂN TÍCH HỆ THỐNG 15](#_Toc185550470)

[2.1 Khảo sát hệ thống 15](#_Toc185550471)

[2.2 Yêu cầu chức năng 15](#_Toc185550472)

[2.3 Yêu cầu phi chức năng 15](#_Toc185550473)

[2.4 Phân tích hệ thống 15](#_Toc185550474)

[2.4.1 Biểu đồ use case 16](#_Toc185550475)

[2.4.2 Đặc tả use case 16](#_Toc185550476)

[CHƯƠNG 3. THIẾT KẾ VÀ XÂY DỰNG GIAO DIỆN WEBSITE 30](#_Toc185550477)

[3.1 Thiết kế các use case 30](#_Toc185550478)

[3.1.1 Use case đăng nh*ập* 30](#_Toc185550479)

[3.1.2 Use case góp ý 33](#_Toc185550480)

[3.1.3 Use case thêm tiểu sử 36](#_Toc185550481)

[3.1.4 Use case xem chi tiết mối quan hệ 39](#_Toc185550482)

[3.1.5 Use case cấp tài khoản 42](#_Toc185550483)

[3.1.6 Use case quản lý tài khoản 45](#_Toc185550484)

[3.1.7 Use case quản lý sự kiện 49](#_Toc185550485)

[3.1.8 Use case quản lý loại tài khoản 53](#_Toc185550486)

[3.1.9 Use case quản lý mối quan hệ 58](#_Toc185550487)

[3.1.10 Use case tìm kiếm người trong gia phả 62](#_Toc185550488)

[3.2 Biểu đồ lớp tổng quát 65](#_Toc185550489)

[3.3 Xây dựng giao diện 66](#_Toc185550490)

[3.3.2 Giao diện cây gia phả 67](#_Toc185550491)

[3.3.3 Giao diện xem chi tiết 67](#_Toc185550492)

[3.3.4 Giao diện các trang quản lý 68](#_Toc185550493)

[CHƯƠNG 4. CÀI ĐẶT VÀ KIỂM THỬ 68](#_Toc185550494)

[4.1 Cài đặt hệ thống 68](#_Toc185550495)

[4.1.1 Cài đặt phía giao diện (Front-end) 68](#_Toc185550496)

[4.1.2 Cài đặt phía Back-end 69](#_Toc185550497)

[4.2 Kiểm thử hệ thống 71](#_Toc185550498)

[KẾT LUẬN 73](#_Toc185550500)

[TÀI LIỆU HƯỚNG DẪN 74](#_Toc185550501)

# DANH MỤC HÌNH ẢNH

[Hình 1: Biểu đồ use case tổng quát 16](#_Toc185550503)

[Hình 2: Biểu đồ phân ra use case đăng nhập 17](#_Toc185550504)

[Hình 3: Biểu đồ phân ra use case góp ý 18](#_Toc185550505)

[Hình 4: Biểu đồ phân rã use case thêm tiểu sử 20](#_Toc185550506)

[Hình 5: Biểu đồ phân rã use case xem chi tiết mối quan hệ 21](#_Toc185550507)

[Hình 6: Biểu đồ phân rã use case cấp tài khoản 22](#_Toc185550508)

[Hình 7: Biểu đồ phân rã use case quản lý tài khoản 23](#_Toc185550509)

[Hình 8: Biểu đồ phân rã use case quản lý sự kiện 25](#_Toc185550510)

[Hình 9: Biểu đồ phân rã use case quản lý tài khoản 27](#_Toc185550511)

[Hình 10: Biểu đồ phân rã use case quản lý mối quan hệ 28](#_Toc185550512)

[Hình 11: Biểu đồ phân rã use case tìm kiếm 29](#_Toc185550513)

[Hình 12: Biểu đồ hoạt động use case đăng nhập 30](#_Toc185550514)

[Hình 13: Biểu đồ trạng thái use case đăng nhập 31](#_Toc185550515)

[Hình 14: Biểu đồ tuần tự use case đăng nhập 31](#_Toc185550516)

[Hình 15: Biểu đồ lớp use case đăng nhập 32](#_Toc185550517)

[Hình 16: Biểu đồ hoạt động use case góp ý 33](#_Toc185550518)

[Hình 17: Biểu đồ trạng thái use case góp ý 34](#_Toc185550519)

[Hình 18: Biều đồ tuần tự use case góp ý 34](#_Toc185550520)

[Hình 19: Biểu đồ lớp use case góp ý 35](#_Toc185550521)

[Hình 20: Biểu đồ hoạt động use case thêm tiểu sử 36](#_Toc185550522)

[Hình 21: Biểu đồ trạng thái use case thêm tiểu sử 37](#_Toc185550523)

[Hình 22: Biểu đồ tuần tự use case thêm tiểu sử 37](#_Toc185550524)

[Hình 23: Biểu đồ lớp use case thêm tiểu sử 38](#_Toc185550525)

[Hình 24: Biểu đồ hoạt động use case xem chi tiết mối quan hệ 39](#_Toc185550526)

[Hình 25: Biểu đồ trạng thái use case xem chi tiết mối quan hệ 39](#_Toc185550527)

[Hình 26: Biểu đồ tuần tự use case xem chi tiết mối quan hệ 40](#_Toc185550528)

[Hình 27: Biểu đồ lớp use case xem chi tiết mối quan hệ 41](#_Toc185550529)

[Hình 28: Biểu đồ hoạt động use case cấp tài khoản 42](#_Toc185550530)

[Hình 29: Biểu đồ trạng thái use case cấp tài khoản 43](#_Toc185550531)

[Hình 30: Biểu đồ tuần tự use case cấp tài khoản 43](#_Toc185550532)

[Hình 31: Biểu đồ lớp use case cấp tài khoản 44](#_Toc185550533)

[Hình 32: Biểu đồ hoạt động quản lý tài khoản 45](#_Toc185550534)

[Hình 33: Biểu đồ trạng thái use case quản lý tài khoản 46](#_Toc185550535)

[Hình 34: Biểu đồ tuân tự use case quản lý tài khoản 47](#_Toc185550536)

[Hình 35: Biểu đồ lớp use case quản lý tài khoản 48](#_Toc185550537)

[Hình 36: Biểu đồ hoạt động use case quản lý sự kiện 49](#_Toc185550538)

[Hình 37: Biểu đồ trạng thái use case quản lý sự kiện 50](#_Toc185550539)

[Hình 38: Biểu đồ tuần tự use case quản lý sự kiện 51](#_Toc185550540)

[Hình 39: Biểu đồ lớp use case quản lý sự kiện 52](#_Toc185550541)

[Hình 40: Biểu đồ hoạt động use case quản lý loại tài khoản 53](#_Toc185550542)

[Hình 41: Biểu đồ trạng thái hoạt động use case quản lý loại tài khoản 54](#_Toc185550543)

[Hình 42: Biểu đồ tuần tự use case quản lý loại tài khoản 56](#_Toc185550544)

[Hình 43: Biểu đồ lớp use case quản lý loại tài khoản 57](#_Toc185550545)

[Hình 44: Biểu đồ hoạt động use case quản lý mối quan hệ 58](#_Toc185550546)

[Hình 45: Biểu đồ trạng thái use case quản lý mối quan hệ 59](#_Toc185550547)

[Hình 46: Biểu đồ tuần tự use case quản lý mối quan hệ 60](#_Toc185550548)

[Hình 47: Biểu đồ lớp use case quản lý mối quan hệ 61](#_Toc185550549)

[Hình 48: Biểu đồ hoạt động use case tìm kiếm người trong gia phả 62](#_Toc185550550)

[Hình 49: Biểu đồ trạng thái use case tìm kiếm người trong gia phả 62](#_Toc185550551)

[Hình 50: Biểu đồ tuần tự use case tìm kiếm người trong gia phả 63](#_Toc185550552)

[Hình 51: Biểu đồ tuần tự use case tìm kiếm người trong gia phả 64](#_Toc185550553)

[Hình 52: Biểu đồ lớp tổng quát 65](#_Toc185550554)

[Hình 53: Giao diện trang chủ 66](#_Toc185550555)

[Hình 54: Giao diện trang gia phả 67](#_Toc185550556)

[Hình 55: Giao diện xem chi tiết 67](#_Toc185550557)

[Hình 56: Giao diện các trang quản lý 68](#_Toc185550558)

[Hình 57: Quy trình cài đặt 69](#_Toc185550559)

[Hình 58: Quy trình cài đặt chuỗi kết nối 70](#_Toc185550560)

[Hình 59: Chọn Starup Project 70](#_Toc185550561)

# DANH MỤC BẢNG BIỂU

[Bảng 1: Đặc tả use case đăng nhập 15](#_Toc185540966)

[Bảng 2: Đặc tả use case góp ý 16](#_Toc185540967)

[Bảng 3: Đặc tả use case thêm tiểu sử 17](#_Toc185540968)

[Bảng 4: Đặc tả use case xem chi tiết mối quan hệ 18](#_Toc185540969)

[Bảng 5: Đặc tả use case cấp tài khoản 20](#_Toc185540970)

[Bảng 6: Đặc tả use case quản lý tài khoản 21](#_Toc185540971)

[Bảng 7: Đặc tả use case quản lý sự kiện 23](#_Toc185540972)

[Bảng 8: Đặc tả use case quản lý loại tài khoản 24](#_Toc185540973)

[Bảng 9: Đặc tả use case quản lý mối quan hệ 26](#_Toc185540974)

[Bảng 10: Đặc tả use case tìm kiếm người trong gia phả 27](#_Toc185540975)

[Bảng 11: Kiểm thử hệ thống 67](#_Toc185540976)

# GIỚI THIỆU VỀ ĐỀ TÀI

## Lý do chọn đề tài

Lý do chọn đề tài “Xây dựng website quản lý gia phả họ Ngô” xuất phát từ mong muốn bảo tồn và lưu trữ thông tin quan trọng về dòng họ một cách bền vững. Gia phả là tài sản quý giá của mỗi gia đình, ghi lại những câu chuyện và truyền thống qua nhiều thế hệ. Thay vì lưu trữ bằng các phương pháp truyền thống dễ bị hư hỏng theo thời gian, việc xây dựng website sẽ giúp thông tin được bảo quản lâu dài, giảm thiểu rủi ro mất mát, hư hỏng do yếu tố tự nhiên.

Một website gia phả giúp các thành viên trong gia đình, dù ở bất kỳ đâu, có thể truy cập và tìm hiểu về cội nguồn, ông bà, tổ tiên. Điều này không chỉ giúp kết nối các thế hệ mà còn tạo ra sự gắn kết sâu sắc giữa các thành viên trong dòng họ. Việc truy cập dễ dàng và chia sẻ thông tin trên nền tảng kỹ thuật số cũng khuyến khích mọi người duy trì sự liên lạc, đoàn kết hơn trong gia đình, đặc biệt trong bối cảnh cuộc sống ngày càng bận rộn.

Bên cạnh đó, website gia phả cho phép thông tin được quản lý và cập nhật một cách linh hoạt. Khi có sự thay đổi như thành viên mới, sửa đổi thông tin hay cập nhật các sự kiện, người quản lý có thể nhanh chóng điều chỉnh. Điều này không chỉ giúp thông tin luôn chính xác và kịp thời mà còn đảm bảo tính liên tục trong việc ghi chép dòng họ, đáp ứng nhu cầu theo dõi và bảo tồn truyền thống một cách toàn diện.

Việc xây dựng website quản lý gia phả cũng là một bước tiến trong ứng dụng công nghệ vào quản lý thông tin gia đình. Đây là cách làm hiện đại, sáng tạo, phù hợp với xu hướng số hóa hiện nay. Đặc biệt, thế hệ trẻ dễ dàng tiếp cận với công nghệ và sẽ có cảm giác gần gũi, hào hứng khi tìm hiểu gia phả qua nền tảng trực tuyến, thay vì qua các quyển gia phả truyền thống ít trực quan.

Dự án không chỉ là một bài tập ứng dụng kiến thức mà còn là một sản phẩm thực tiễn, có tiềm năng trong thị trường hiện nay. Với những lý do trên, tôi tin rằng việc xây dựng hệ thống website quản lý gia phả họ Ngô không chỉ đáp ứng nhu cầu cấp thiết của người trong dòng họ còn mang lại giá trị lớn cho bản thân trong quá trình học tập và phát triển kỹ năng công nghệ thông tin.

## Mục tiêu đề tài

Mục tiêu chính của dự án là phát triển một hệ thống website gia phả cho gia đình họ Ngô, với khả năng hỗ trợ tổng quát thành viên trong gia phả.

* **Lưu trữ và bảo tồn thông tin gia phả**: Xây dựng một hệ thống trực tuyến giúp lưu trữ thông tin về dòng họ một cách có hệ thống và bền vững. Thông qua website, các thông tin về tổ tiên, các thế hệ, và các mối quan hệ trong gia đình sẽ được ghi chép và bảo tồn một cách an toàn, không lo hư hại như với các tài liệu giấy.
* **Tạo sự kết nối và chia sẻ trong dòng họ**: Website gia phả sẽ giúp các thành viên dễ dàng truy cập và chia sẻ thông tin, bất kể họ đang ở đâu. Qua đó, tạo ra một nền tảng kết nối, giúp tăng cường sự gắn kết và tình cảm gia đình, đặc biệt đối với những thành viên ở xa.
* **Đảm bảo khả năng quản lý và cập nhật thông tin linh hoạt**: Thiết lập hệ thống giúp người quản trị dễ dàng cập nhật thông tin khi có sự thay đổi như thành viên mới, sự kiện gia đình, hoặc các chi tiết khác. Mục tiêu này đảm bảo gia phả luôn chính xác và kịp thời, phù hợp với thực tế.
* **Ứng dụng công nghệ hiện đại vào việc quản lý gia phả**: Mục tiêu của đề tài là phát triển một hệ thống ứng dụng công nghệ, thân thiện với người dùng và đáp ứng nhu cầu tìm hiểu cội nguồn của các thế hệ trẻ. Qua đó, giúp thế hệ sau có thể dễ dàng tiếp cận và hiểu hơn về dòng họ của mình thông qua một giao diện trực tuyến hiện đại, trực quan.

Dự án còn giúp nâng cao kiến thức và kỹ năng lập trình, phát triển ứng dụng thực tế cho các thành viên trong nhóm, tạo tiền đề cho các dự án quy mô lớn hơn trong tương lai.

## Phạm vi nghiên cứu

Phạm vi nghiên cứu của đề tài xây dựng website quản lý gia phả được giới hạn ở các khía cạnh chính sau đây, nhằm đảm bảo tính khả thi và phù hợp với mục tiêu dự án:

* Phạm vi về chức năng: Nghiên cứu tập trung vào các chức năng cơ bản và quan trọng nhất của một hệ thống quản lý. Các chức năng dành cho người dùng bao gồm đăng nhập, tìm kiếm, thêm, sửa, xóa thành viên trong gia phả. Đối với, phạm vi nghiên cứu bao gồm các chức năng quản lý danh mục tài khoản.
* Phạm vi về người dùng: Đối tượng người dùng trong phạm vi nghiên cứu là những người trong dòng họ Ngô
* Phạm vi kỹ thuật: Về công nghệ, phạm vi nghiên cứu giới hạn trong các công cụ và ngôn ngữ lập trình như mongodb, C#(.NET) và JavaScript (Vuejs) để phát triển giao diện và chức năng của hệ thống. Hệ thống cũng sử dụng mongodb làm cơ sở dữ liệu chính để lưu trữ thông tin người dùng, và các dữ liệu liên quan.
* Phạm vi về triển khai và thử nghiệm: Dự án giới hạn trong việc xây dựng hệ thống trên môi trường máy chủ cục bộ và môi trường thử nghiệm, nhằm đảm bảo các tính năng hoạt động ổn định và đáp ứng nhu cầu cơ bản. Phạm vi thử nghiệm sẽ bao gồm các thao tác chính như đăng nhập, tìm kiếm, thêm, sửa, xóa thành viên và cập nhật dữ liệu. Hệ thống chưa bao gồm các tính năng nâng cao như tối ưu hóa cho các nền tảng di động.

## Công nghệ sử dụng

### VUEJS

**Vue.js** là một framework JavaScript mã nguồn mở, được sử dụng để xây dựng giao diện người dùng và các ứng dụng web một cách nhanh chóng và hiệu quả. Vue.js nổi bật nhờ tính linh hoạt và khả năng tùy biến cao, giúp các nhà phát triển dễ dàng tạo ra các ứng dụng web có tính tương tác cao.

**Ưu điểm của Vue.js**

* Giao diện tương tác mượt mà: Vue.js hỗ trợ reactivity (tính phản ứng) giúp giao diện cập nhật tự động khi dữ liệu thay đổi, phù hợp cho các ứng dụng cần tính tương tác cao như quản lý gia phả, bảng dữ liệu động, hoặc hiển thị thông tin chi tiết của từng thành viên trong hệ thống.
* Cấu trúc dự án rõ ràng và dễ bảo trì: Vue.js sử dụng các component (thành phần) độc lập, giúp tổ chức code rõ ràng, dễ bảo trì và tái sử dụng.
* Hệ sinh thái phong phú: Vue.js có hệ sinh thái phong phú bao gồm Vuex để quản lý trạng thái, Vue Router để điều hướng, và các thư viện khác hỗ trợ phát triển ứng dụng. Điều này giúp tích hợp các tính năng phức tạp như hệ thống đăng nhập, phân quyền, và hiển thị thông tin gia phả.
* Tài liệu và cộng đồng lớn: Vue.js có tài liệu chi tiết và cộng đồng mạnh mẽ, giúp người phát triển dễ dàng tìm kiếm và áp dụng các giải pháp cho các vấn đề gặp phải.

Nhược điểm của Vue.js

* Không phổ biến bằng một số framework khác: Vue.js có thể ít phổ biến hơn so với các framework như React hoặc Angular, điều này có thể hạn chế khả năng tìm kiếm nhà phát triển có kinh nghiệm trong Vue.js ở một số thị trường.
* Hạn chế khi phát triển ứng dụng lớn: Mặc dù Vue.js phù hợp cho các ứng dụng vừa và nhỏ, nhưng khi xây dựng các ứng dụng lớn và phức tạp, có thể cần tích hợp thêm các công cụ khác để tối ưu hóa hiệu năng và quản lý dự án, đòi hỏi sự hiểu biết sâu hơn về hệ sinh thái Vue.js.
* Quản lý trạng thái phức tạp: Khi ứng dụng có nhiều trạng thái phức tạp, Vuex có thể yêu cầu thiết lập nhiều cấu trúc phức tạp, đòi hỏi thêm thời gian và công sức để tổ chức và quản lý hiệu quả.
* Khả năng bảo mật hạn chế: HTML không hỗ trợ các tính năng bảo mật như mã hóa dữ liệu, bảo vệ tài khoản người dùng, hoặc ngăn chặn các cuộc tấn

### C#

**C#** là một ngôn ngữ lập trình hiện đại, hướng đối tượng, được phát triển bởi Microsoft, thường sử dụng cùng với .NET Framework hoặc .NET Core (phiên bản mã nguồn mở). **.NET** là nền tảng phát triển đa năng của Microsoft, hỗ trợ nhiều ngôn ngữ lập trình và cung cấp một môi trường nhất quán để xây dựng, triển khai và quản lý các ứng dụng trên cả máy chủ và máy khách. C# và .NET thường được sử dụng để phát triển các ứng dụng web, ứng dụng di động, và phần mềm doanh nghiệp với hiệu năng cao và khả năng mở rộng.

**Ưu điểm của C# và .NET**

* Tính đa nền tảng: Với sự ra đời của .NET Core và hiện nay là .NET 5/6/7, các ứng dụng C# có thể chạy trên nhiều nền tảng như Windows, Linux và macOS, giúp giảm chi phí và mở rộng phạm vi sử dụng.
* Hiệu năng cao: .NET được tối ưu hóa để cung cấp hiệu năng cao, đặc biệt trong các ứng dụng web và dịch vụ API. .NET sử dụng công nghệ Just-In-Time (JIT) compilation và hỗ trợ tối ưu hóa bộ nhớ, giúp xử lý tốt các ứng dụng lớn và phức tạp.
* Hỗ trợ mạnh mẽ cho ứng dụng web: Với .NET, các công cụ như ASP.NET Core, MVC, và Web API giúp xây dựng các ứng dụng web hiện đại, nhanh chóng và bảo mật. .NET cũng tích hợp tốt với các công nghệ JavaScript như React và Angular để phát triển các ứng dụng giao diện người dùng phong phú.
* Thư viện phong phú và cộng đồng mạnh mẽ: .NET đi kèm với một thư viện phong phú, cung cấp nhiều công cụ và tiện ích giúp giảm thiểu thời gian phát triển. Ngoài ra, cộng đồng lập trình viên sử dụng .NET rất lớn và sôi động, hỗ trợ giải pháp cho các vấn đề phát sinh.
* Hỗ trợ phát triển ứng dụng doanh nghiệp: C# và .NET rất phổ biến trong các tổ chức lớn nhờ tính bảo mật và tính ổn định cao, cùng với sự hỗ trợ lâu dài từ Microsoft. Các ứng dụng như quản lý quan hệ khách hàng (CRM), quản lý nguồn nhân lực (HRM), và hệ thống tài chính thường được phát triển bằng C# và .NET.
* Khả năng mở rộng và bảo trì tốt: C# là ngôn ngữ hướng đối tượng, dễ hiểu, dễ bảo trì và mở rộng. Điều này rất quan trọng cho các dự án doanh nghiệp dài hạn, khi yêu cầu bảo trì và nâng cấp thường xuyên..

**Nhược điểm của C# và .NET**

* Phụ thuộc vào Microsoft: Dù .NET Core đã trở thành mã nguồn mở, nhưng hệ sinh thái .NET vẫn phụ thuộc khá lớn vào Microsoft, đặc biệt khi có các thay đổi về chiến lược và công nghệ mới.
* Kích thước runtime lớn: Đối với các ứng dụng nhỏ, runtime của .NET có thể tạo ra các ứng dụng có kích thước lớn hơn so với các ngôn ngữ khác, làm cho việc triển khai có thể trở nên phức tạp và tốn kém tài nguyên.
* Độ phức tạp cao đối với người mới: .NET có rất nhiều công nghệ và thư viện đi kèm, như ASP.NET, WPF, WinForms, và các công cụ quản lý như Entity Framework. Đối với người mới, việc làm quen với hệ sinh thái lớn và phức tạp này có thể mất nhiều thời gian.
* Chi phí cao cho các công cụ phát triển: Dù .NET Core và các công cụ phát triển liên quan đã được cung cấp miễn phí, nhưng việc sử dụng Visual Studio (phiên bản đầy đủ) thường yêu cầu chi phí giấy phép khá cao trong môi trường doanh nghiệp..

### MongoDB

**MongoDB** là một hệ quản trị cơ sở dữ liệu NoSQL, mã nguồn mở, được thiết kế để lưu trữ dữ liệu theo mô hình tài liệu (document-oriented) và tối ưu cho các ứng dụng đòi hỏi khả năng mở rộng cao, dữ liệu phong phú và đa dạng. MongoDB lưu trữ dữ liệu dưới dạng các tài liệu JSON có cấu trúc linh hoạt, điều này giúp việc thao tác và xử lý dữ liệu linh hoạt hơn so với các cơ sở dữ liệu quan hệ truyền thống.

**Ưu điểm của MongoDB**

* Linh hoạt về cấu trúc dữ liệu: MongoDB không đòi hỏi một schema cố định như các cơ sở dữ liệu SQL. Điều này giúp dễ dàng thay đổi cấu trúc của tài liệu khi cần, và phù hợp với dữ liệu không có cấu trúc hoặc có cấu trúc phức tạp, cho phép lưu trữ các đối tượng và mảng lồng nhau.
* Hiệu năng cao và mở rộng theo chiều ngang (horizontal scaling): MongoDB hỗ trợ sharding – kỹ thuật chia nhỏ dữ liệu ra nhiều máy chủ để xử lý song song, giúp tăng khả năng xử lý và lưu trữ dữ liệu lớn mà không ảnh hưởng đến hiệu năng.
* Truy vấn và thao tác nhanh: MongoDB cung cấp các phương thức truy vấn đa dạng và hỗ trợ các chỉ mục (indexing) mạnh mẽ, giúp thực hiện các thao tác đọc/ghi nhanh chóng, ngay cả đối với lượng dữ liệu lớn.
* Phù hợp với dữ liệu lớn và phân tán: MongoDB rất thích hợp cho các hệ thống có dữ liệu lớn và yêu cầu khả năng phân phối dữ liệu trên nhiều máy chủ, như các ứng dụng phân tích dữ liệu, ứng dụng thời gian thực hoặc các ứng dụng phân tán đa vùng.
* Khả năng tích hợp linh hoạt: MongoDB có thể tích hợp với nhiều ngôn ngữ lập trình như Node.js, Python, Java, C#... nhờ có các thư viện và API hỗ trợ đa dạng. Điều này làm cho MongoDB trở thành lựa chọn phổ biến cho các dự án với yêu cầu kết hợp nhiều công nghệ.
* Hỗ trợ tốt cho các loại dữ liệu phi cấu trúc: Do MongoDB lưu trữ dữ liệu dưới dạng JSON hoặc BSON, nó rất phù hợp để lưu các loại dữ liệu phi cấu trúc hoặc bán cấu trúc như thông tin người dùng, nội dung tin nhắn, hoặc dữ liệu từ các thiết bị IoT.

**Nhược điểm của MongoDB**

* Thiếu các tính năng ACID cho giao dịch phức tạp: Dù đã hỗ trợ giao dịch đa tài liệu từ MongoDB 4.0 trở đi, MongoDB vẫn chưa cung cấp khả năng đảm bảo các thuộc tính ACID một cách mạnh mẽ như các cơ sở dữ liệu quan hệ. Điều này có thể gây khó khăn cho các ứng dụng cần xử lý các giao dịch tài chính hoặc dữ liệu nghiêm ngặt về tính nhất quán.
* Không phù hợp cho các ứng dụng có yêu cầu phức tạp về quan hệ: MongoDB lưu trữ dữ liệu dưới dạng tài liệu độc lập, vì vậy các mối quan hệ phức tạp giữa các bảng như trong SQL có thể khó quản lý và tối ưu trong MongoDB. Thông thường, các ứng dụng quan hệ phức tạp sẽ yêu cầu cơ sở dữ liệu quan hệ truyền thống hơn.
* Dung lượng bộ nhớ cao hơn: MongoDB lưu trữ dữ liệu dưới dạng BSON (Binary JSON) và lưu thêm các metadata cho mỗi tài liệu, nên sẽ chiếm dung lượng lưu trữ lớn hơn so với các cơ sở dữ liệu quan hệ truyền thống.
* Đòi hỏi quản lý sharding phức tạp: Đối với các hệ thống lớn, khi sử dụng sharding, việc cấu hình và bảo trì hệ thống có thể trở nên phức tạp, đòi hỏi sự quản lý chi tiết và kinh nghiệm từ người vận hành..

## Kết quả dự kiến

Website gia phả hoạt động ổn định và thân thiện với người dùng: Hoàn thành một website với giao diện trực quan, thân thiện, dễ dàng sử dụng cho mọi đối tượng, bao gồm cả những người không quen thuộc với công nghệ. Website có cấu trúc rõ ràng, giúp người dùng dễ dàng truy cập, tìm kiếm và thao tác với thông tin gia phả.

Hệ thống lưu trữ và quản lý thông tin gia phả đầy đủ và chi tiết: Đảm bảo website lưu trữ toàn bộ thông tin chi tiết về từng thành viên trong dòng họ như tên, năm sinh, mối quan hệ gia đình, hình ảnh và các sự kiện quan trọng. Thông tin được tổ chức theo cây phả hệ, hiển thị rõ ràng các mối quan hệ giữa các thế hệ.

Chức năng cập nhật và chỉnh sửa thông tin dễ dàng: Website cung cấp chức năng cho phép người quản trị dễ dàng cập nhật thông tin, thêm mới hoặc chỉnh sửa thông tin của từng thành viên. Điều này giúp đảm bảo gia phả luôn được cập nhật, phù hợp với những thay đổi trong dòng họ.

Hỗ trợ quản lý và thông báo sự kiện dòng họ: Website bao gồm tính năng thông báo và lên lịch cho các sự kiện chung của gia đình như họp mặt, giỗ tổ, lễ kỷ niệm. Chức năng này giúp các thành viên dễ dàng theo dõi và tham gia các sự kiện, từ đó tăng cường sự gắn kết trong gia đình.

Nâng cao hiểu biết và gắn kết trong dòng họ: Qua website, các thành viên có thể tìm hiểu sâu sắc hơn về cội nguồn và các giá trị văn hóa, lịch sử của dòng họ. Đây là một kết quả quan trọng, góp phần xây dựng niềm tự hào và tăng cường sự đoàn kết giữa các thế hệ.

## Kết luận

Trong chương I, dự án đã trình bày tổng quan về ý tưởng xây dựng website quản lý gia phả họ Ngô, với mục tiêu hỗ trợ người dùng nắm được thông tin dòng họ một cách trực quan. Lý do lựa chọn đề tài xuất phát từ nhu cầu muốn tìm hiểu nguồn gốc của gia đình. Dự án cũng đã nêu rõ các mục tiêu nghiên cứu, nhằm đáp ứng yêu cầu của cả người dùng và quản trị viên, đồng thời giới hạn phạm vi trong các tính năng và kỹ thuật cốt lõi để đảm bảo tính khả thi.

Với những kết quả dự kiến đạt được, dự án hướng tới việc cung cấp một hệ thống hoàn thiện, hiện đại và dễ sử dụng. Chương I đã đặt nền tảng cơ bản, làm tiền đề cho các nội dung chi tiết trong các chương tiếp theo.

# KHẢO SÁT, PHÂN TÍCH HỆ THỐNG

## Khảo sát hệ thống

Để truy cập và sử dụng các chức năng hệ thống, người dùng cần phải đăng nhập khi có tài khoản (tài khoản được cấp do admin khi đã có thông tin trong gia phả). Để tránh dữ liệu bị thay đổi mất kiểm soát, tài khoản được cấp chỉ có thể xem chi tiết mối quan hệ, chứ không được có hành động như thêm, sửa, xóa, những hành động này do tài khoản admin thực hiện. Trong năm,dòng họ có thể tổ chức sự kiện nào đó. Ngoài ra, cần có mục góp ý để các thành viên trong họ có thể đóng góp trực tiếp cho người phát triển.

## Yêu cầu chức năng

Từ đoạn mô tả về hệ thống, ta có những chức này như sau:

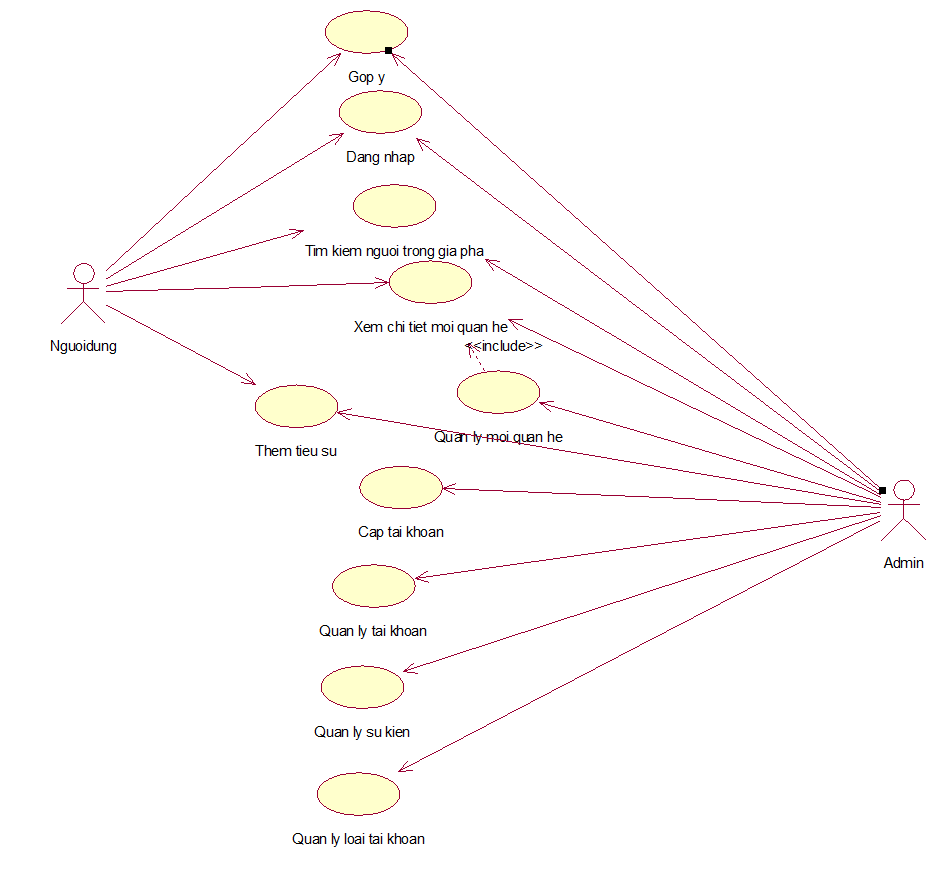
* Đăng nhập
* Góp ý
* Xem chi tiết mối quan hệ
* Tìm kiếm người trong gia phả
* Thêm tiểu sử
* Quản lý mối quan hệ
* Cấp tài khoản
* Quản lý tài khoản
* Quản lý sự kiện
* Quản lý loại tài khoản

## Yêu cầu phi chức năng

* Giao diện thân thiện, dễ hiểu, và trực quan với người dùng. Cung cấp hướng dẫn rõ ràng khi xảy ra lỗi hoặc thực hiện các thao tác phức tạp.
* Mọi thành viên của dòng họ khi được ghi thông tin vào gia phả đều phải có “Họ và tên”, “Ngày tháng năm sinh”, “Năm mất” (ghi theo Âm lịch - nếu đã tạ thế), “Nơi an nghỉ/Nơi an táng”
* Nếu con trai đã lấy vợ thì ghi họ tên, ngày tháng năm sinh của vợ, quê quán ở đâu và sinh được mấy người con. Tương tự thế, các con của người đó cũng sẽ cần ghi lại đầy đủ thông tin như bố, mẹ.
* Nếu là con gái của dòng họ thì cũng ghi họ tên, ngày tháng năm sinh, năm mất (nếu đã tạ thế), gả đi lấy chồng là ai, quê quán ở đâu, con cái là những ai, tên là gì là đủ, không ghi chú các đời sau đó nữa.

## Phân tích hệ thống

### Biểu đồ use case



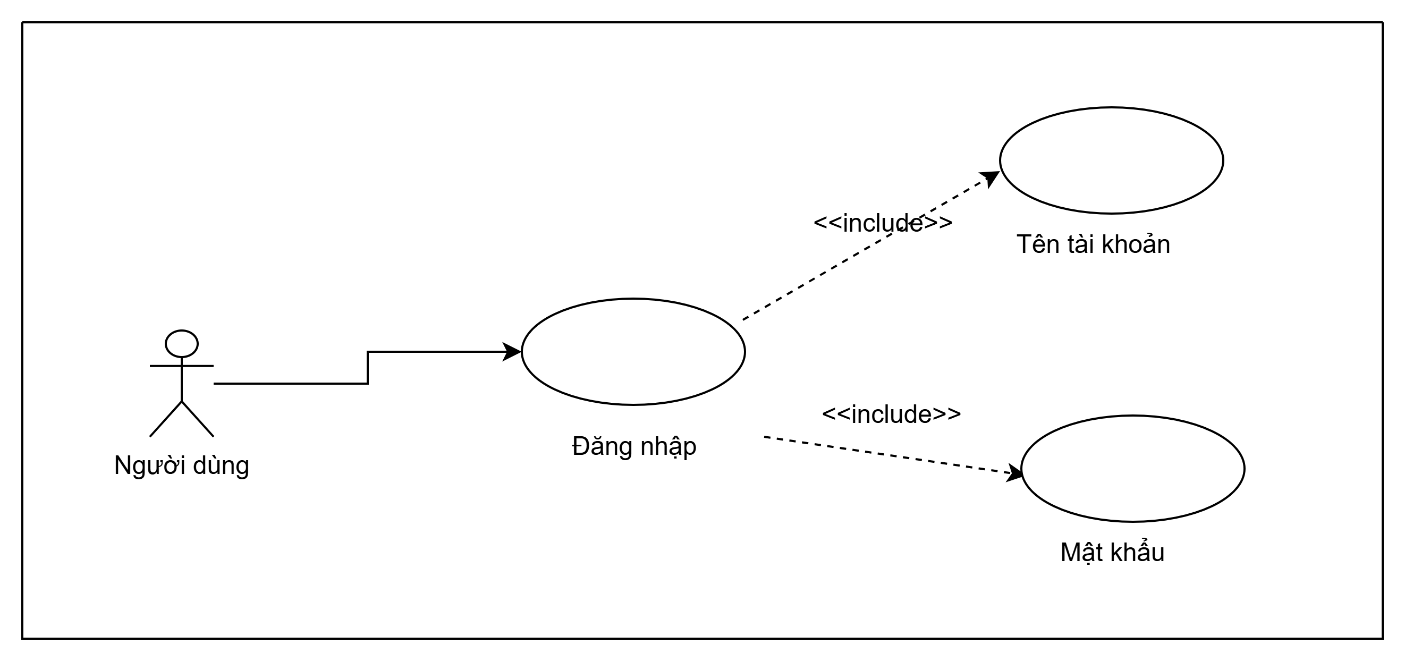
Hình : Biểu đồ use case tổng quát

### Đặc tả use case

#### Đặc tả use case đăng nhập

|  |  |
| --- | --- |
| Tên use case | Đăng nhập |
| Mô tả | Use case cho phép người dùng đăng nhập hệ thống |
| Luồng sự kiện | Luồng cơ bản:   1. Người dùng truy cập vào trang chủ hoặc di chuyển đến vùng có form đăng nhập hoặc bấm vào nút đăng nhập trên thanh menu 2. Người dùng điền đầy đủ thông tin đăng nhập: tài khoản và mật khẩu và chọn nút “đăng nhập” 3. Hệ thống kiểm tra dữ liệu nhập vào 4. Thông tin hợp lệ, hệ thống chuyển hướng đến trang gia phả. Use case kết thúc.   Luồng rẽ nhánh:   1. Tại bước 3 luồng cơ bản: Hệ thống kiêm tra dữ liệu không hợp lệ sẽ thông báo lỗi. Use case kết thúc. 2. Tại bất kỳ bước nào trong luồng cơ bản, nếu không kết nối được với cơ sở dữ liệu thì hệ thống sẽ hiển thị một thông báo lỗi và use case kết thúc. |
| Tiền điều kiện |  |
| Hậu điều kiện |  |

Bảng : Đặc tả use case đăng nhập

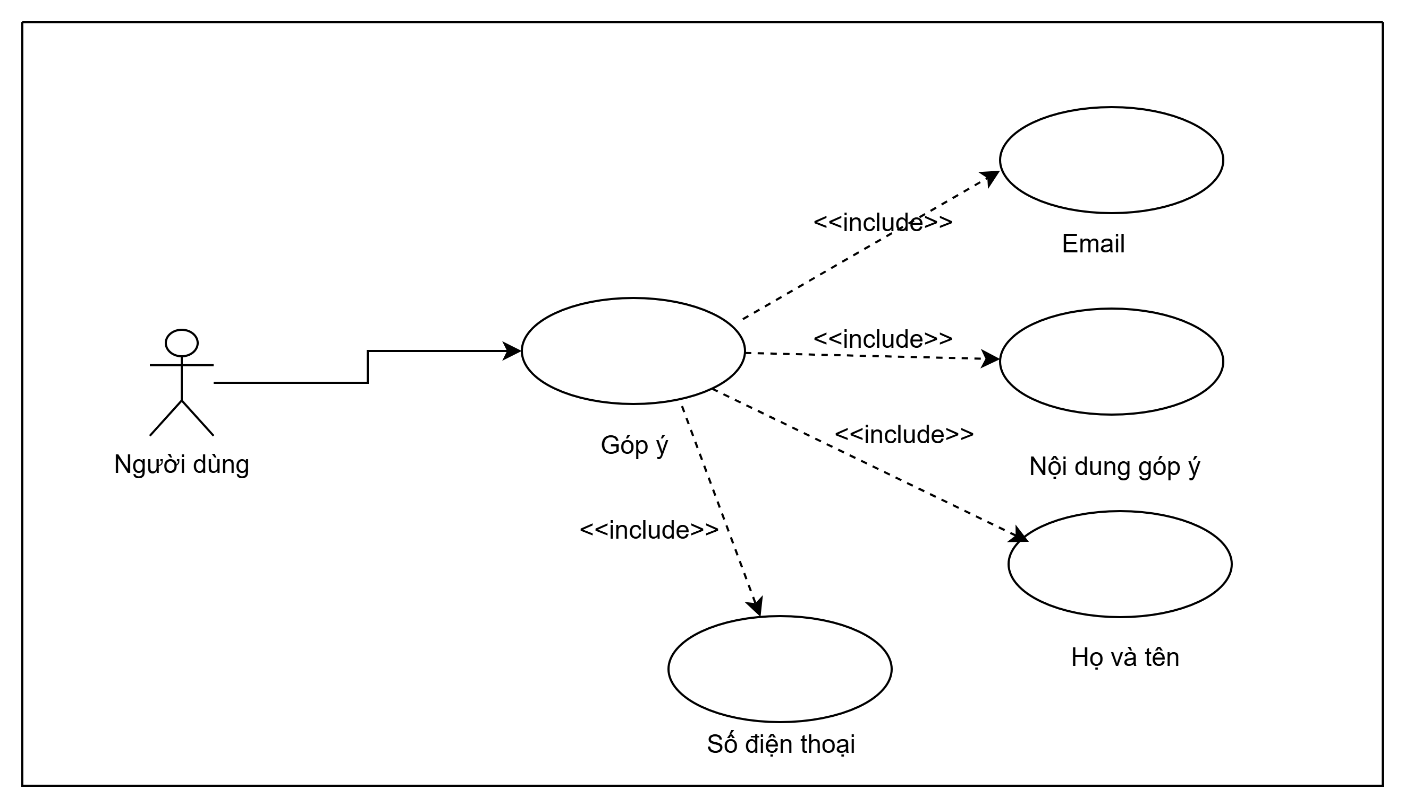


Hình : Biểu đồ phân ra use case đăng nhập

#### Đặc tả use case góp ý

|  |  |
| --- | --- |
| Tên use case | Góp ý |
| Mô tả | Use case cho phép người dùng có thể đóng góp ý kiến để trang web cải thiện hơn |
| Luồng sự kiện | Luồng cơ bản:   1. Người dùng truy cập vào trang chủ và kích chuột vào menu “Góp ý” hoặc di chuyển xuống vùng góp ý 2. Người dùng điền đầy đủ thông tin góp ý: email, số điện thoại, họ và tên, nội dung góp ý và chọn nút “Góp ý” 3. Hệ thống kiểm tra dữ liệu nhập vào 4. Hệ thống ghi nhận thông tin và thông báo cảm ơn 5. Use case kết thúc.   Luồng rẽ nhánh:   1. Tại bước 4 ở luồng cơ bản: Nếu người dùng nhập sai định dạng các trường. Hệ thống thông báo lỗi định dạng ở dưới ô nhập dữ liệu. 2. Tại bất kỳ bước nào trong luồng cơ bản, nếu không kết nối được với cơ sở dữ liệu thì hệ thống sẽ hiển thị một thông báo lỗi và use case kết thúc. |
| Tiền điều kiện | * Người dùng đã truy cập vào trang chủ. |
| Hậu điều kiện |  |

Bảng : Đặc tả use case góp ý

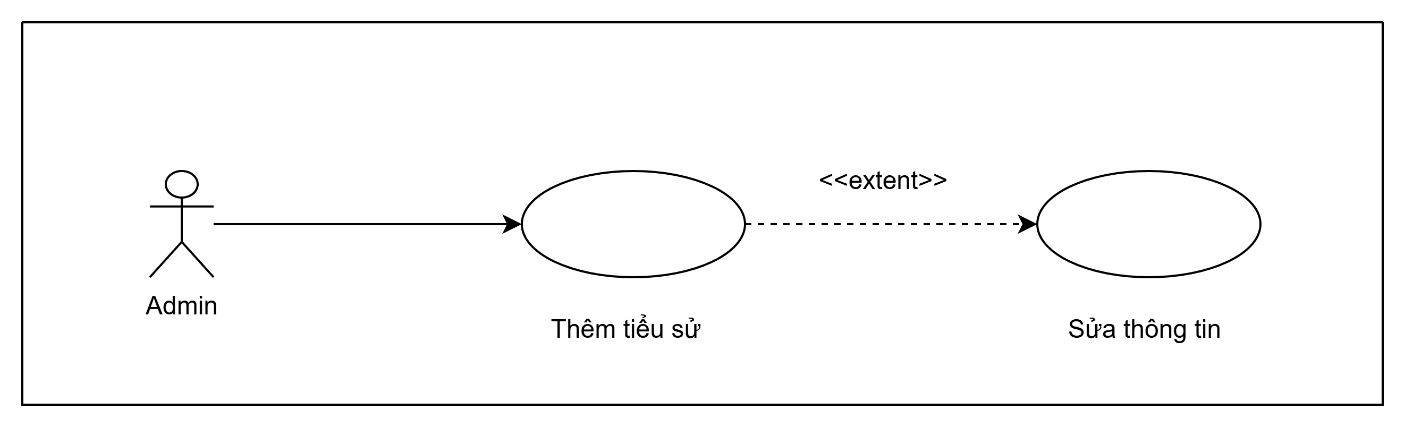


Hình : Biểu đồ phân ra use case góp ý

#### Đặc tả use case thêm tiểu sử

|  |  |
| --- | --- |
| Tên use case | Thêm tiểu sử |
| Mô tả | Use case cho phép người dùng bổ sung hoặc cập nhật tiểu sử của thành viên trong gia phả |
| Luồng sự kiện | Luồng cơ bản:   1. Người dùng truy cập trang gia phả. 2. Người dùng kích vào biểu tượng thêm tiểu sử. 3. Người dùng có thể thêm (nếu chưa có thông tin), sửa tiểu sử của 1 thành viên trong gia phả. 4. Người dùng có thể sửa thông tin cá nhân của đối tượng đang thêm tiểu sử và bấm nút “Lưu”. 5. Hệ thống kiểm tra thông tin người dùng nhập và xử lý. 6. Hệ thống hiển thị thông báo thay đổi thành công. 7. Use case kết thúc   Luồng rẽ nhánh:   1. Tại bước 5 ở luồng cơ bản: Nếu thông tin không hợp lệ, hệ thống thông báo lỗi và yêu cầu nhập lại hoặc kích nút “Thoát” để kết thúc 2. Tại bất kỳ bước nào trong luồng cơ bản, nếu không kết nối được với cơ sở dữ liệu thì hệ thống sẽ hiển thị một thông báo lỗi và use case kết thúc. |
| Các yêu cầu đặc biệt | * Người dùng chỉ được thêm tiểu sử của chính mình. * Admin có thể thêm được cho tất cả thành viên trong gia phả |
| Tiền điều kiện | * Người dùng đã đăng nhập thành công. * Hệ thống có cơ sở dữ liệu chứa thông tin cá nhân của người dùng. |
| Hậu điều kiện |  |

Bảng : Đặc tả use case thêm tiểu sử

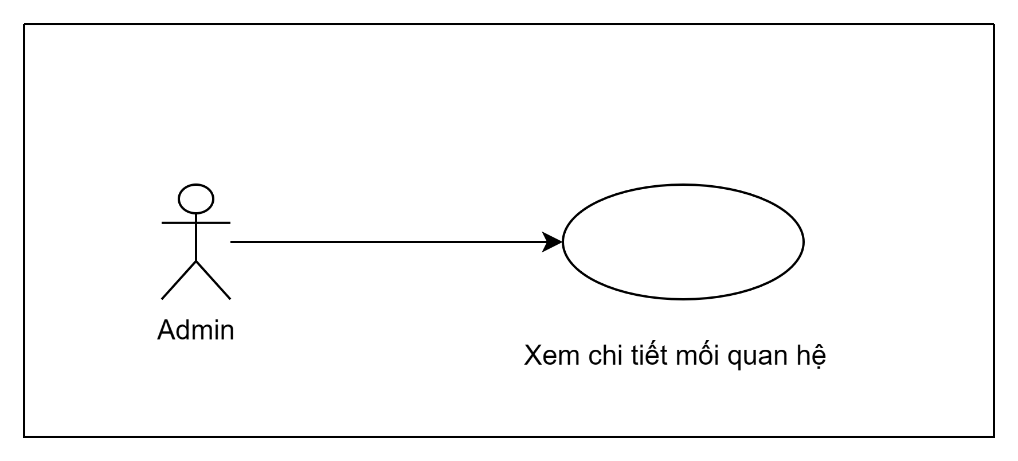


Hình : Biểu đồ phân rã use case thêm tiểu sử

#### Đặc tả use case xem chi tiết mối quan hệ

|  |  |
| --- | --- |
| Tên use case | Xem chi tiết mối quan hệ |
| Mô tả | Use case cho phép người dùng xem các mối quan hệ xung quanh thành viên đã chọn |
| Luồng sự kiện | Luồng cơ bản:   1. Người dùng truy cập vào trang gia phả 2. Người dùng kích chuột vào tên của người bất kỳ có trong cây gia phả 3. Hệ thống kiểm tra thông tin mối quan hệ xung quanh người đó như bố mẹ, anh chị em, vợ con và hiển thị lên màn hình   Luồng rẽ nhánh:   1. Tại bất kỳ bước nào trong luồng cơ bản, nếu không kết nối được với cơ sở dữ liệu thì hệ thống sẽ hiển thị một thông báo lỗi và use case kết thúc. |
| Tiền điều kiện | * Người dùng đã đăng nhập thành công. * Hệ thống có cơ sở dữ liệu chứa thông tin cá nhân của người dùng. |
| Hậu điều kiện |  |

Bảng : Đặc tả use case xem chi tiết mối quan hệ

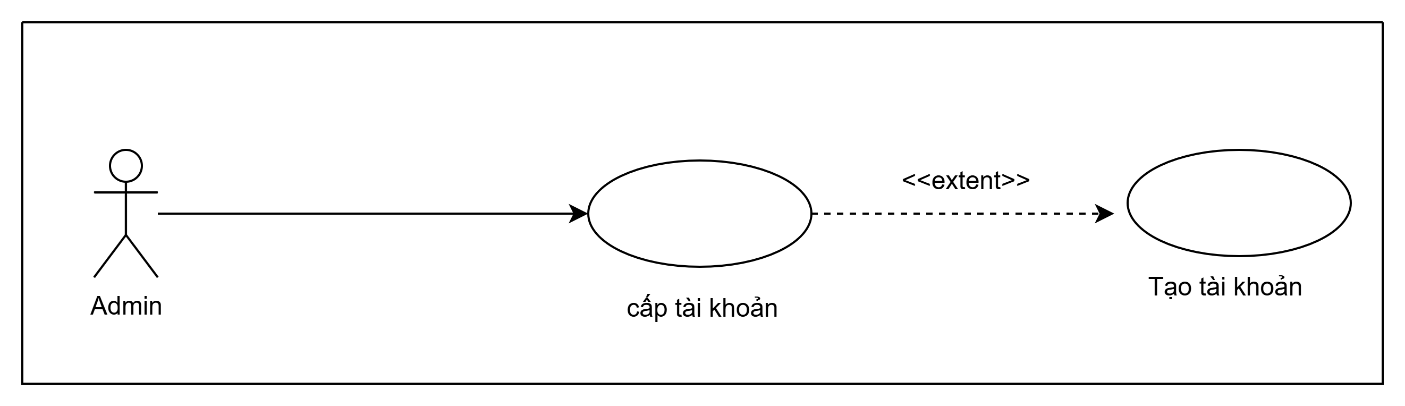


Hình : Biểu đồ phân rã use case xem chi tiết mối quan hệ

#### Đặc tả use case cấp tài khoản

|  |  |
| --- | --- |
| Tên use case | Cấp tài khoản |
| Mô tả | Use case cho phép cấp tài khoản cho một thành viên trong gia phả |
| Luồng sự kiện | Luồng cơ bản:   1. Người dùng kích vào biểu tượng cấp tài khoản 2. Hệ thống tự động sinh ra “Tên đăng nhập” và “mật khẩu” cho đối tượng cấp (là tên kết hợp với ngày sinh) và hiển thị form gồm 2 trường đã lêu 3. Người dùng có thể sửa tên đăng nhập hoặc mật khẩu nếu cần thiết 4. Nhấn “Lưu” để cấp tài khoản   Luồng rẽ nhánh:   1. Tại bất kỳ bước nào trong luồng cơ bản, nếu không kết nối được với cơ sở dữ liệu thì hệ thống sẽ hiển thị một thông báo lỗi và use case kết thúc. 2. Tại bước 4, nếu người dùng bấm nút “Thoát” thì use case kết thúc |
| Tiền điều kiện | * Người dùng đã thực hiện đăng nhập. * Tài khoản để cấp phải là tài khoản admin |
| Hậu điều kiện |  |

Bảng : Đặc tả use case cấp tài khoản

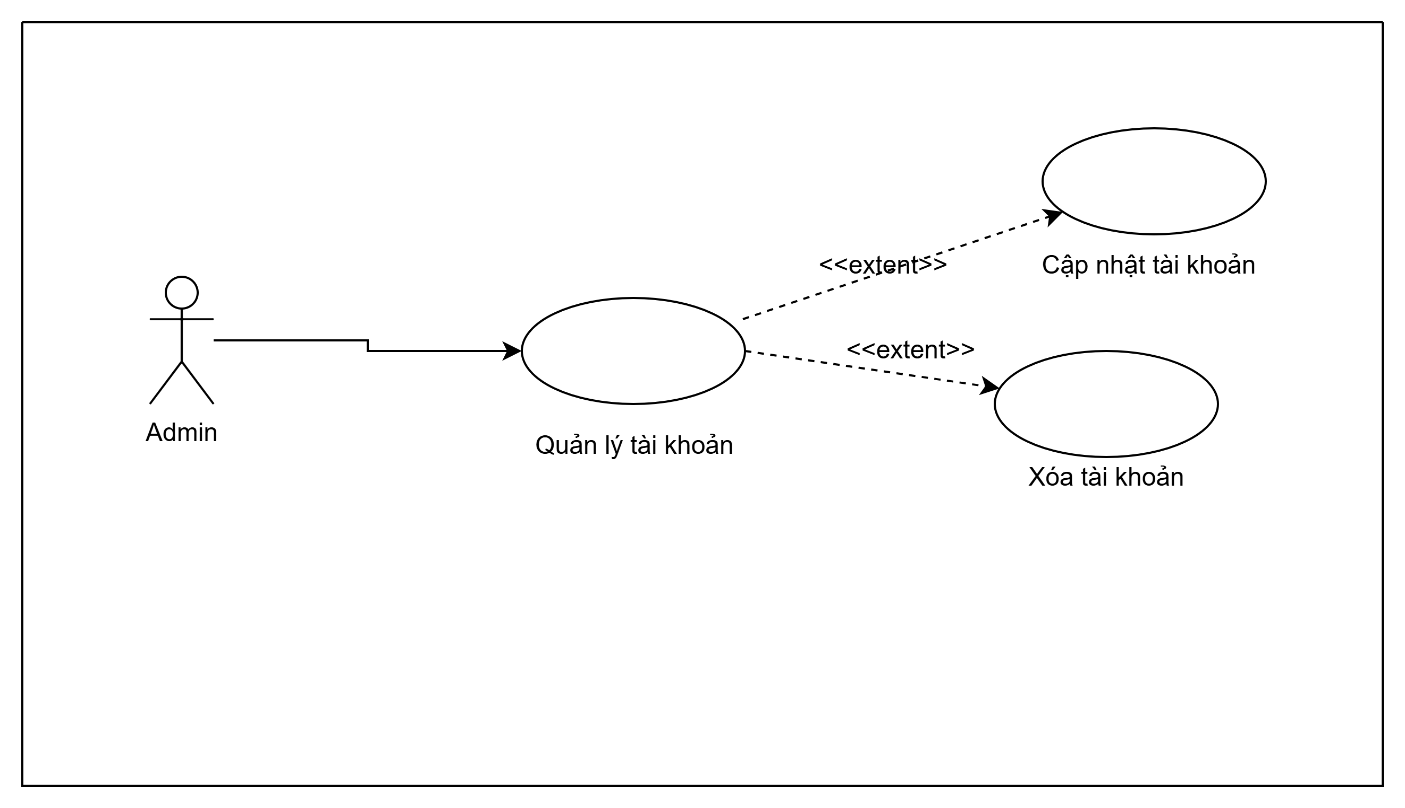


Hình : Biểu đồ phân rã use case cấp tài khoản

#### Đặc tả use case quản lý tài khoản

|  |  |
| --- | --- |
| Tên use case | Quản lý tài khoản |
| Mô tả | Use case cho phép người dùng sửa, xóa tài khoản trong hệ thống |
| Luồng sự kiện | Luồng cơ bản:   1. Người dùng truy cập trang quản lý tài khoản 2. Người dùng kích chọn vào biểu tượng sửa trong danh sách 3. Hệ thống kiểm tra thông tin và điền thông tin sẵn có và hiển thị form sửa. 4. Người dùng nhập thông tin muốn sửa và kích nút “Lưu” 5. Hệ thống kiểm tra thông tin và cập nhật lại thông tin vào cơ sở dữ liệu 6. Người dùng kích chọn vào biểu tượng xóa trong danh sách 7. Hệ thống hiển thị thông báo xác nhận xóa. 8. Nếu người dùng chọn “OK” thì hệ thống sẽ xóa tài khoản và thông báo xóa thành công   Luồng rẽ nhánh:   1. Tại bất kỳ bước nào trong luồng cơ bản, nếu không kết nối được với cơ sở dữ liệu thì hệ thống sẽ hiển thị một thông báo lỗi và use case kết thúc. 2. Tại bước thứ 7 trong luồng cơ bản, nếu người dùng bấm nút “Cancel” thì thông tin sẽ không được xóa và use case kết thúc 3. Tại bước thứ 4 trong luồng cơ bản, nếu thông tin không hợp lệ, hệ thống thông báo lỗi không cập nhật lại cơ sở dữ liệu |
| Tiền điều kiện | * Người dùng đã thực hiện đăng nhập. * Hệ thống có cơ sở dữ liệu chứa thông tin cá nhân của người dùng. |
| Hậu điều kiện |  |

Bảng : Đặc tả use case quản lý tài khoản

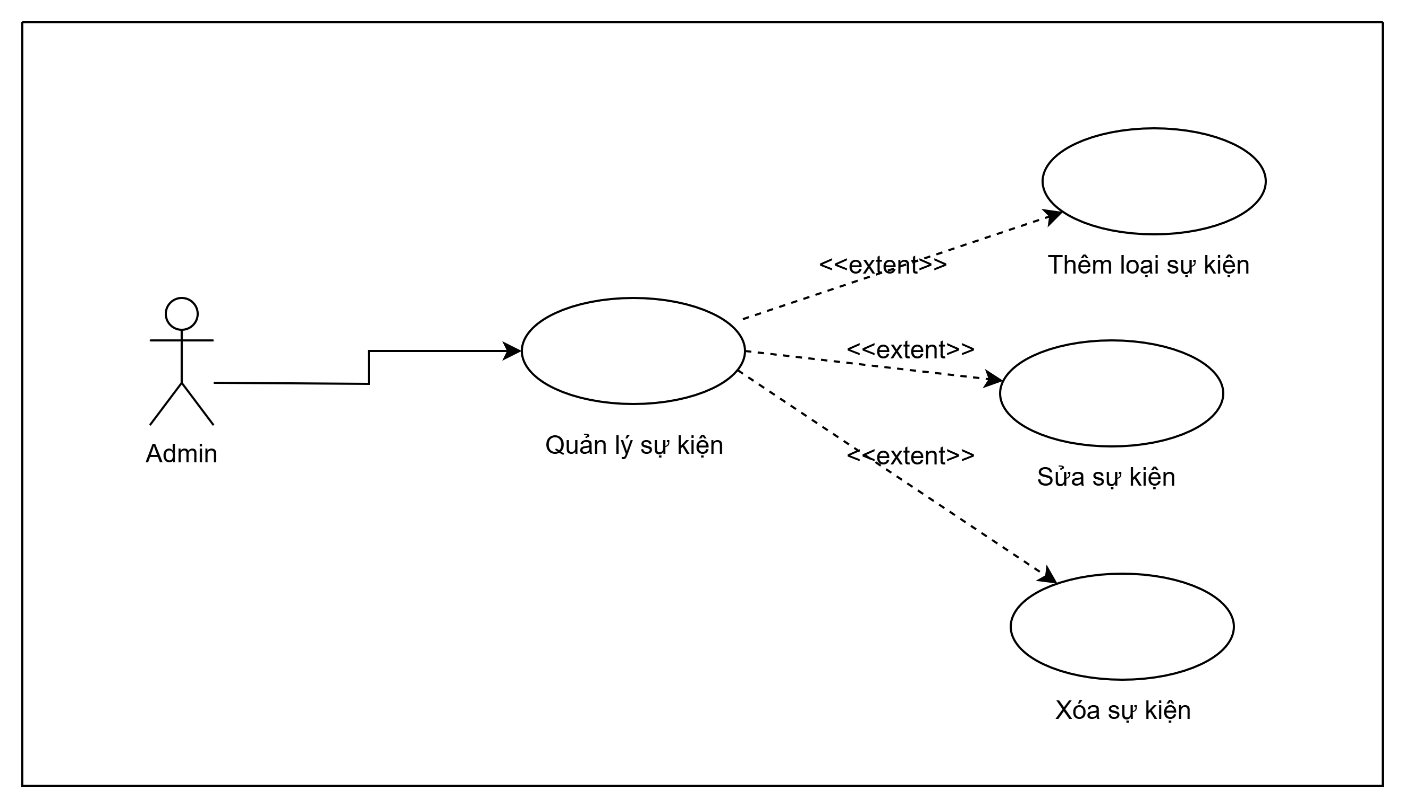


Hình : Biểu đồ phân rã use case quản lý tài khoản

#### Đặc tả use case quản lý sự kiện

|  |  |
| --- | --- |
| Tên use case | Quản lý sự kiện |
| Mô tả | Use case này cho phép người dùng thêm, sửa, xóa sự kiện trong họ |
| Luồng sự kiện | Luồng cơ bản:   1. Người dùnng truy cập vào quản lý sự kiện. 2. Admin chọn biểu tượng thêm sự kiện. 3. Hệ thống hiện form nhập thêm mới sự kiện. 4. Người dùng điền thông tin mới và nhấn “Lưu” 5. Hệ thống kiểm tra thông tin người dùng nhập và xử lý. 6. Hệ thống hiển thị thông báo lưu thành công. 7. Admin chọn biểu tượng sửa sự kiện trong danh sách sự kiện. 8. Hệ thống hiện form và điền sẵn thông tin đã có. 9. Người dùng cập nhật thông tin và nhấn “Lưu” 10. Hệ thống kiểm tra thông tin người dùng nhập và xử lý. 11. Hệ thống hiển thị thông báo lưu thành công. 12. Admin chọn biểu tượng xóa sự kiện trong danh sách. 13. Hệ thống hiện thị thông báo xác nhận xóa. 14. Người dùng bấm nút “OK” 15. Hệ thống kiểm tra thông tin và xử lý. 16. Hệ thống hiển thị thông báo xóa thành công.   Luồng rẽ nhánh:   1. Tại bất kỳ bước nào trong luồng cơ bản, nếu không kết nối được với cơ sở dữ liệu thì hệ thống sẽ hiển thị một thông báo lỗi và use case kết thúc. 2. Tại bước thứ 14 trong luồng cơ bản, nếu người dùng bấm nút “Cancel” thì thông tin sẽ không được xóa và use case kết thúc |
| Tiền điều kiện | * Tài khoản đăng nhập là tài khoản admin |
| Hậu điều kiện |  |

Bảng : Đặc tả use case quản lý sự kiện

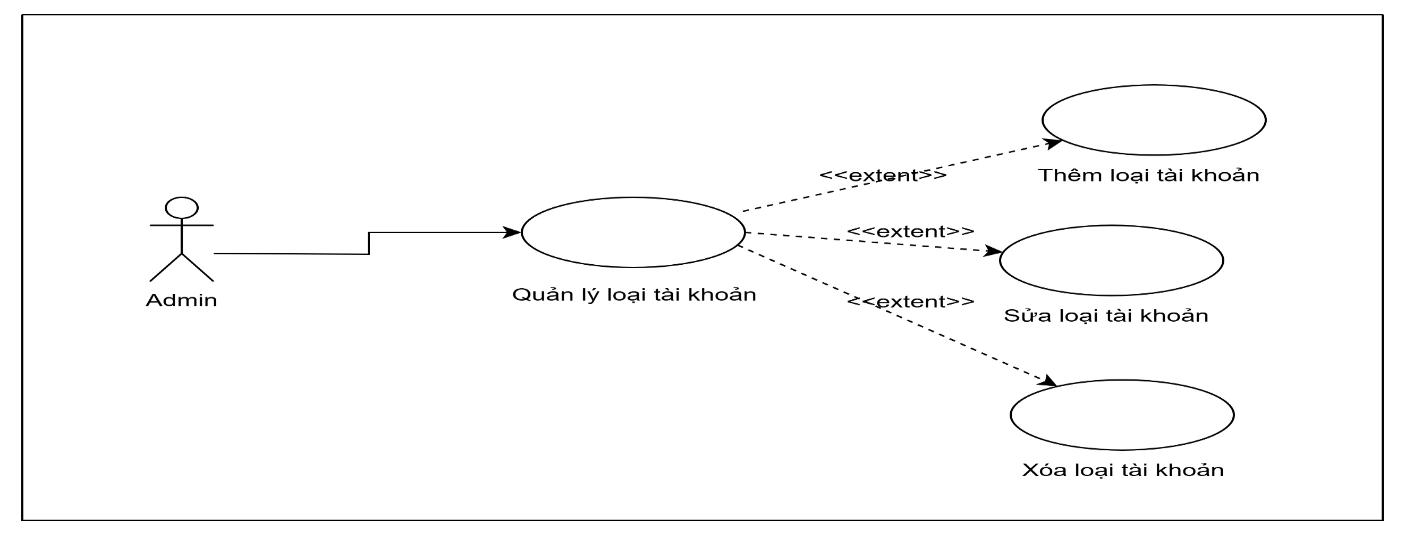


Hình : Biểu đồ phân rã use case quản lý sự kiện

#### Đặc tả use case quản lý loại tài khoản

|  |  |
| --- | --- |
| Tên use case | Quản lý loại tài khoản |
| Mô tả | Use case này cho phép người dùng thêm, sửa, xóa loại tài khoản trong hệ thống |
| Luồng sự kiện | Luồng cơ bản:   1. Người dùnng truy cập vào trang loại tài khoản. 2. Admin chọn biểu tượng thêm loại tài khoản. 3. Hệ thống hiện form nhập thêm mới loại tài khoản. 4. Người dùng điền thông tin mới và nhấn “Lưu” 5. Hệ thống kiểm tra thông tin người dùng nhập và xử lý. 6. Hệ thống hiển thị thông báo lưu thành công. 7. Admin chọn biểu tượng sửa loại tài khoản trong danh sách. 8. Hệ thống hiện form và điền sẵn thông tin đã có. 9. Người dùng cập nhật thông tin và nhấn “Lưu” 10. Hệ thống kiểm tra thông tin người dùng nhập và xử lý. 11. Hệ thống hiển thị thông báo lưu thành công. 12. Admin chọn biểu tượng xóa trong danh sách. 13. Hệ thống hiện thị thông báo xác nhận xóa. 14. Người dùng bấm nút “OK” 15. Hệ thống kiểm tra thông tin và xử lý. 16. Hệ thống hiển thị thông báo xóa thành công.   Luồng rẽ nhánh:   1. Tại bất kỳ bước nào trong luồng cơ bản, nếu không kết nối được với cơ sở dữ liệu thì hệ thống sẽ hiển thị một thông báo lỗi và use case kết thúc. 2. Tại bước thứ 14 trong luồng cơ bản, nếu người dùng bấm nút “Cancel” thì thông tin sẽ không được xóa và use case kết thúc |
| Tiền điều kiện | * Tài khoản đăng nhập là tài khoản admin |
| Hậu điều kiện |  |

Bảng : Đặc tả use case quản lý loại tài khoản

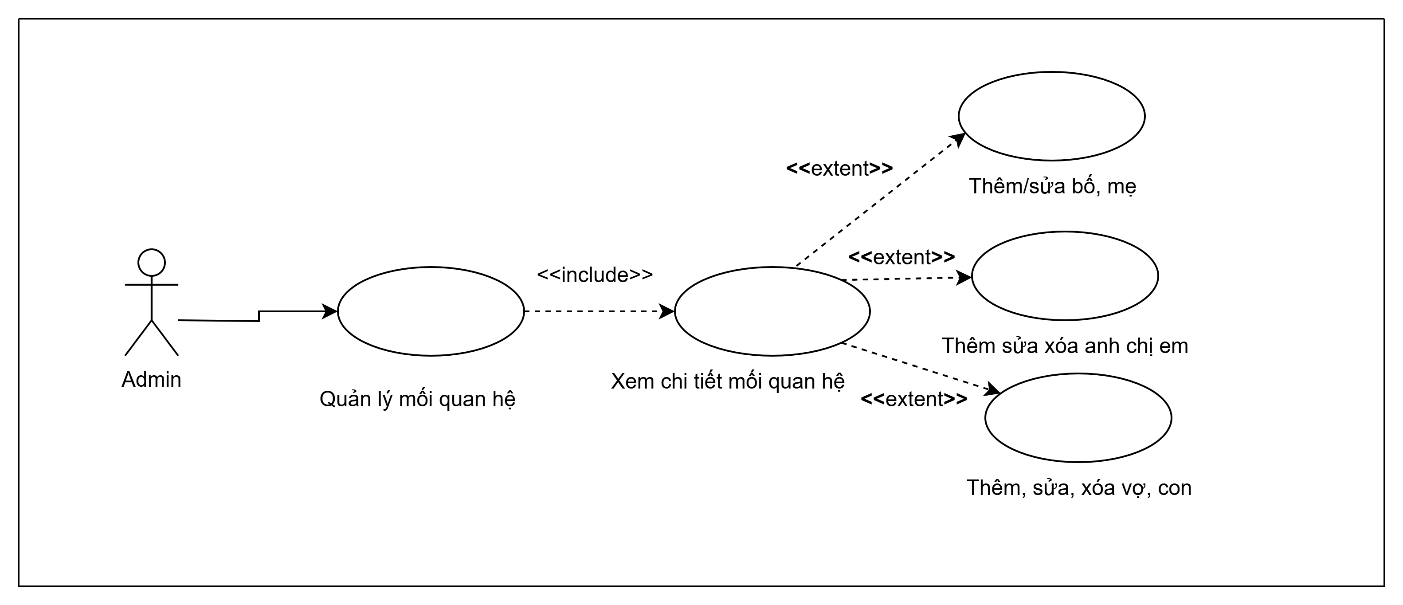


Hình : Biểu đồ phân rã use case quản lý tài khoản

#### Đặc tả use case quản lý mối quan hệ

|  |  |
| --- | --- |
| Tên use case | Quản lý mối quan hệ |
| Mô tả | Use case cho phép người dùng thêm, sửa, xóa các mối quan hệ xung quanh như bố me, anh chị em, vợ con |
| Luồng sự kiện | Luồng cơ bản:   1. Người dùng truy cập vào trang gia pha của trang web. 2. Người dùng có thể lựa chọn thêm hoặc sửa thông tin của thành viên trong gia đình (bố mẹ, anh chị em, hoặc vợ con). 3. Hệ thống hiển form để người dùng nhập hoặc sửa thông tin 4. Người dùng điền thông tin cần thiết như tên, tuổi, giới tính, năm sinh,… sau đó chọn “lưu” để xác nhận . 5. Hệ thống kiểm tra thông tin Admin nhập và xử lý. 6. Hệ thống hiển thị thông báo thay đổi thành công 7. Người dùng nhấn biểu tượng xóa trong danh sách vợ con hoặc danh sách anh chị em. 8. Hệ thống hiển thị thông báo xác nhận xóa. 9. Người dùng bấm nút “OK” 10. Hệ thống kiểm tra thông tin và thông báo cho người dùng   Luồng rẽ nhánh:   1. Tại bất kỳ bước nào trong luồng cơ bản, nếu không kết nối được với cơ sở dữ liệu thì hệ thống sẽ hiển thị một thông báo lỗi và use case kết thúc. 2. Tại bước 10 trong luồng cơ bản, hệ thống kiểm tra thông tin và phát hiện thông tin không hợp lệ sẽ thông báo cho người dùng |
| Tiền điều kiện | * Người dùng đã thực hiện đăng nhập. * Hệ thống có cơ sở dữ liệu chứa thông tin cá nhân của người dùng. * Người dùng đang thực hiện use case xem chi tiết mối quan hệ * Tài khoản đăng nhập là admin |
| Hậu điều kiện |  |

Bảng : Đặc tả use case quản lý mối quan hệ



Hình : Biểu đồ phân rã use case quản lý mối quan hệ

#### Đặc tả use case tìm kiếm người trong gia phả

|  |  |
| --- | --- |
| Tên use case | Tìm kiếm người trong gia phả |
| Mô tả | Use case cho phép người dùng tìm kiếm thông tin muốn tìm |
| Luồng sự kiện | Luồng cơ bản:   1. Người dùng truy cập vào trang gia pha của trang web. 2. Hệ thống hiển thị ô tìm kiếm ở góc phải màn hình 3. Người dùng chọn trong danh sách đổ ra từ ô tìm kiếm (là 1 danh sách lựa chọn người trong gia phả) 4. Hệ thống tìm kiếm thông tin và di chuyển màn hình đến vị trí người đó trong cây gia phả và làm nổi bật ô thể hiện người đó trong gia phả |
| Tiền điều kiện | * Người dùng đã thực hiện đăng nhập. |
| Hậu điều kiện |  |

Bảng : Đặc tả use case tìm kiếm người trong gia phả



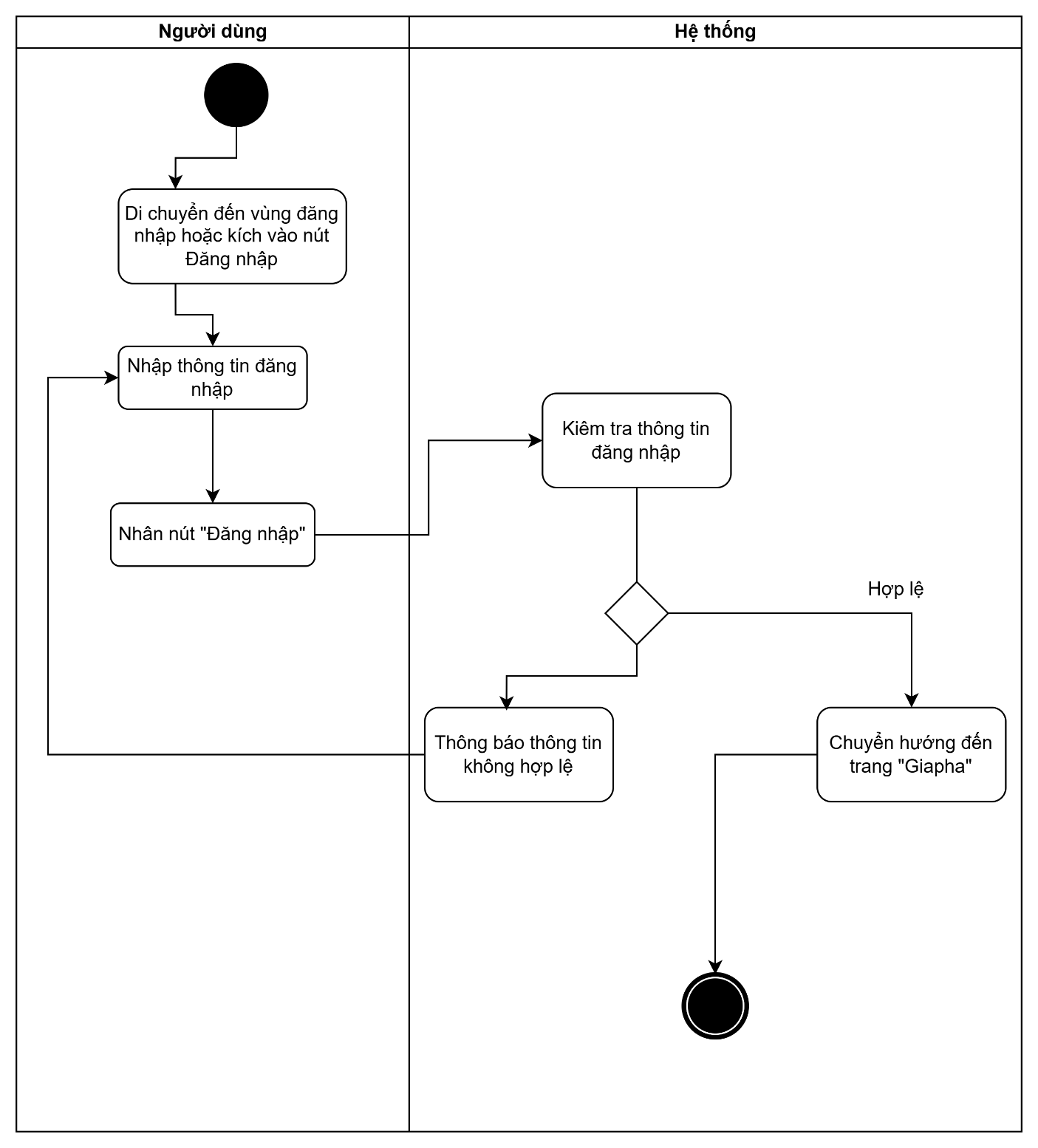
Hình : Biểu đồ phân rã use case tìm kiếm

# THIẾT KẾ VÀ XÂY DỰNG GIAO DIỆN WEBSITE

## Thiết kế các use case

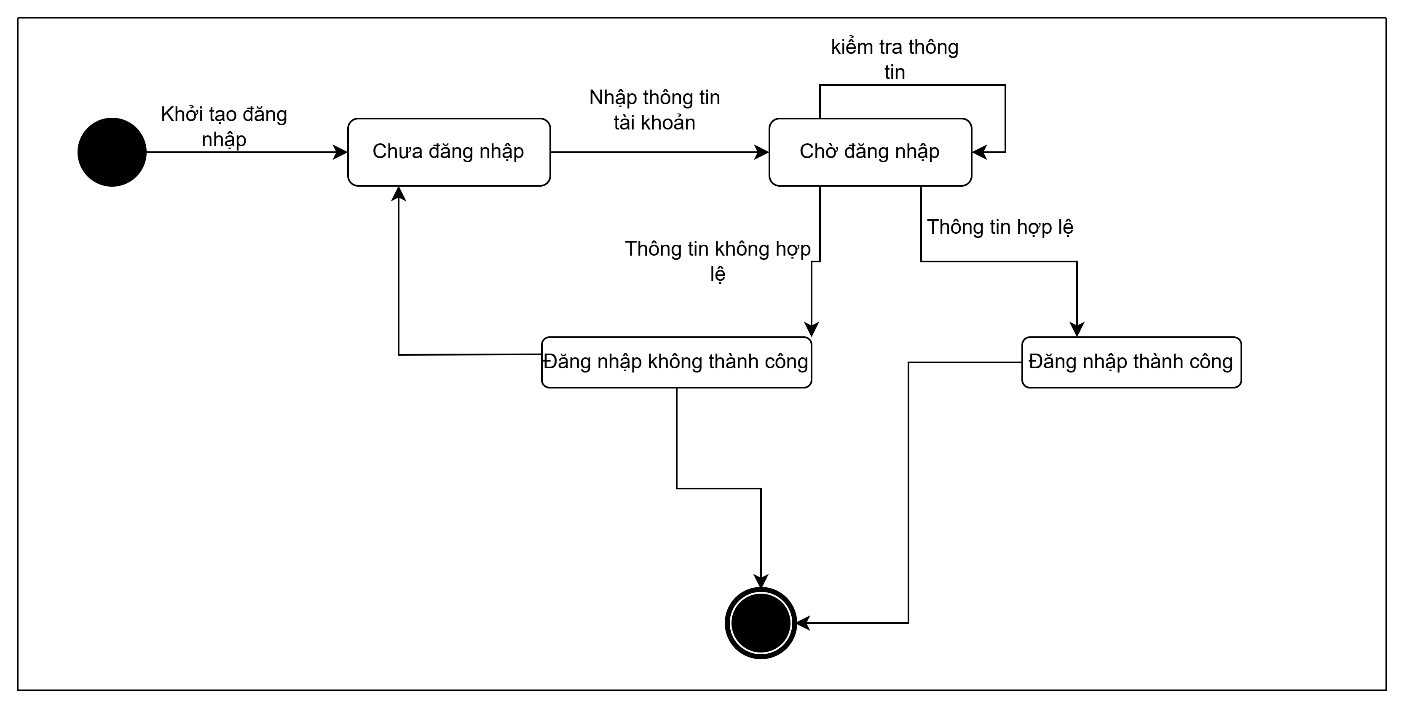
### Use case đăng nhập

#### Biểu đồ hoạt động



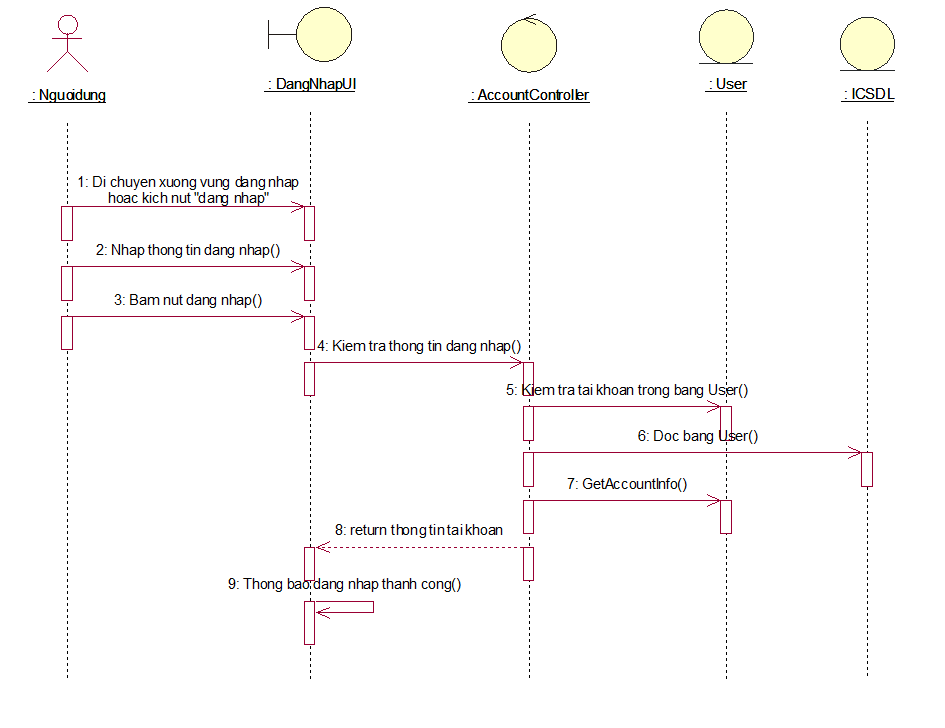
Hình : Biểu đồ hoạt động use case đăng nhập

#### Biểu đồ trạng thái



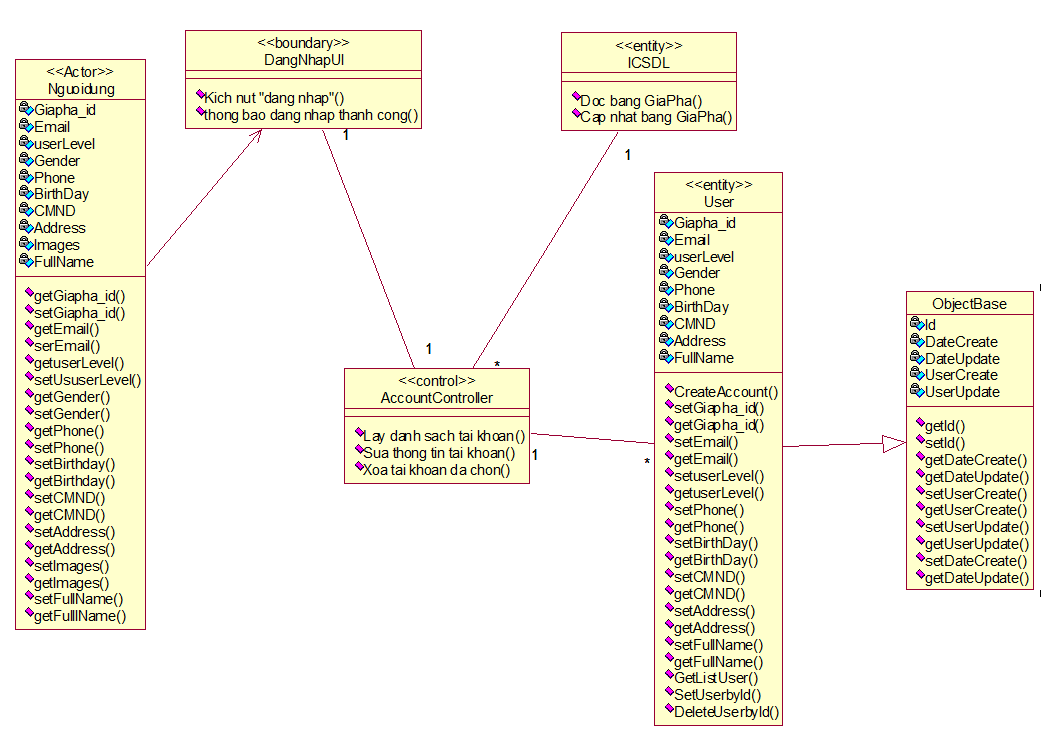
Hình : Biểu đồ trạng thái use case đăng nhập

#### Biểu đồ tuần tự



Hình 14: Biểu đồ tuần tự use case đăng nhập

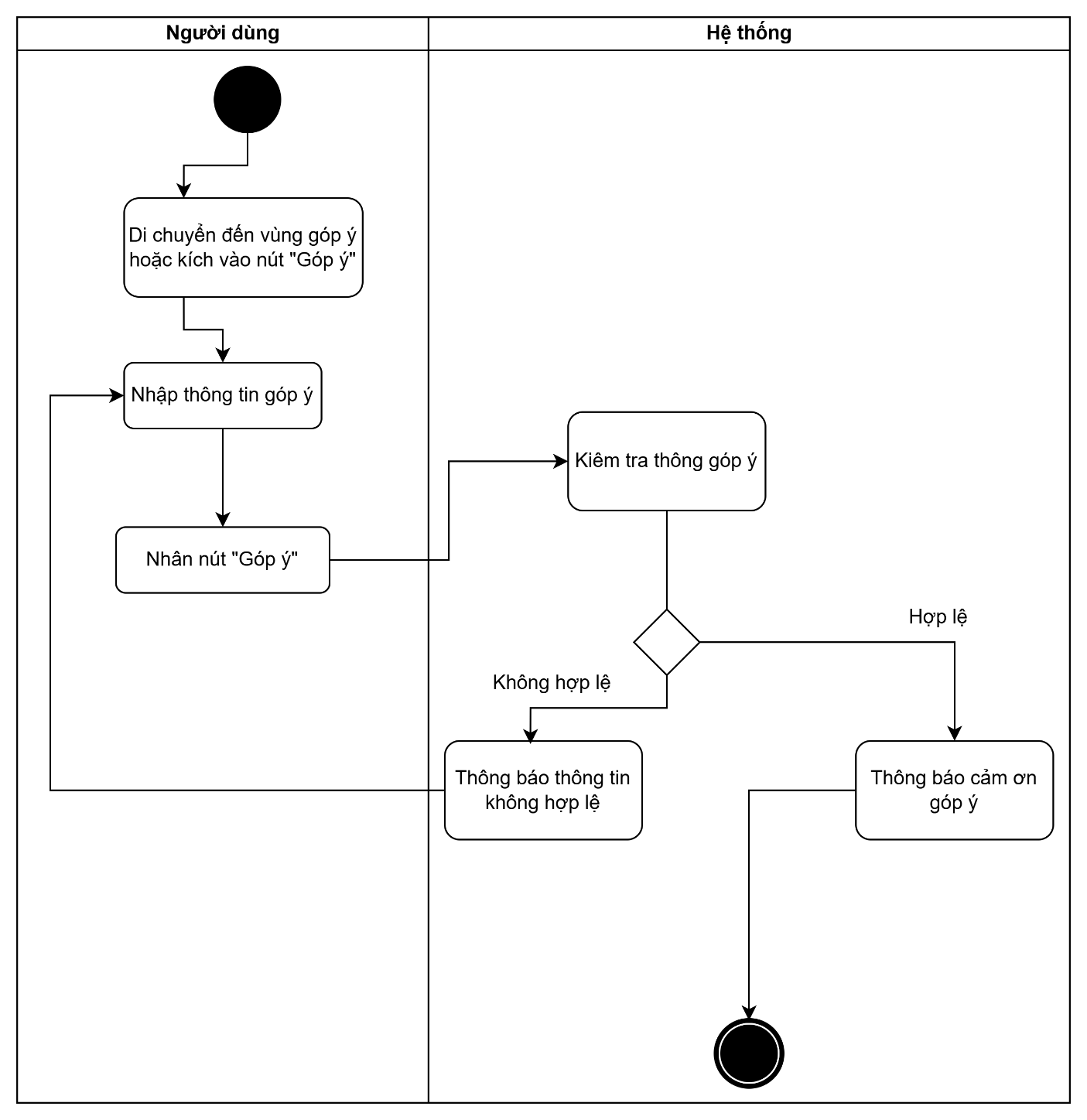
#### Biểu đồ lớp



Hình : Biểu đồ lớp use case đăng nhập

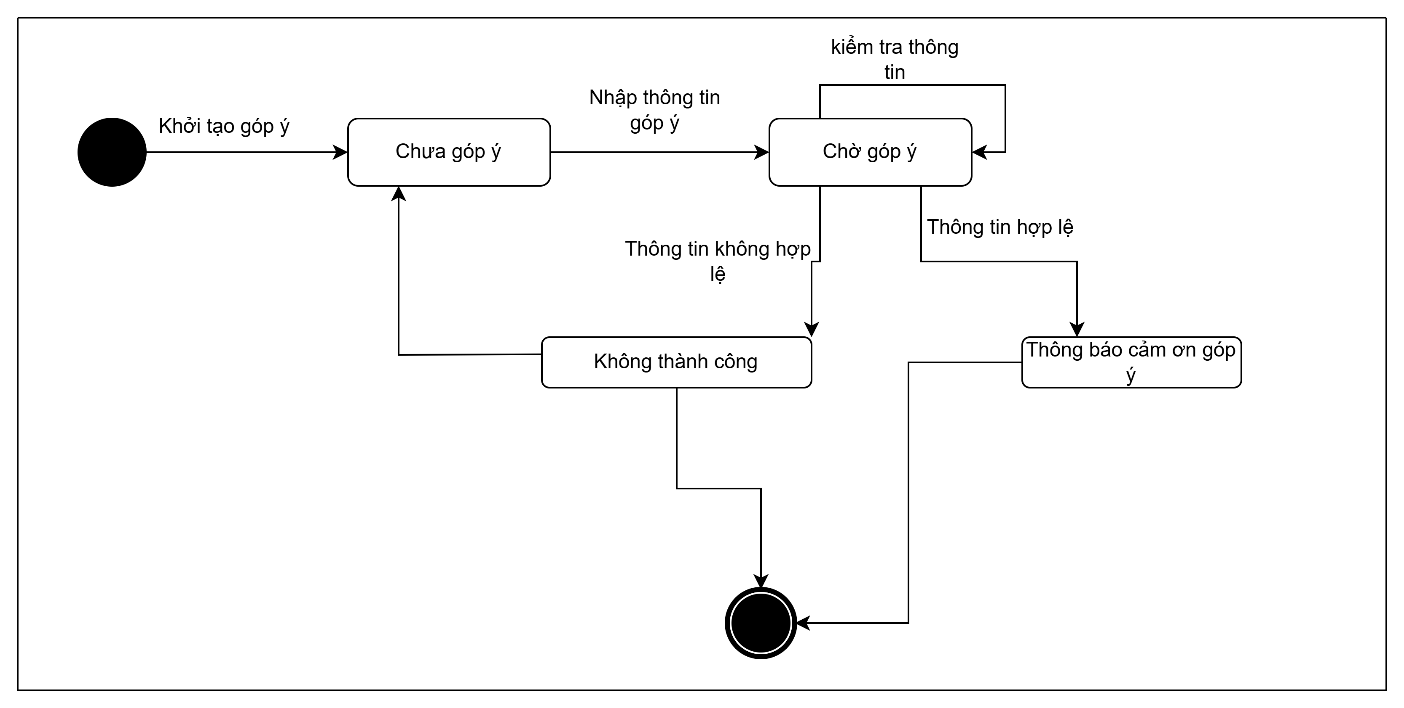
### Use case góp ý

#### Biểu đồ hoạt động



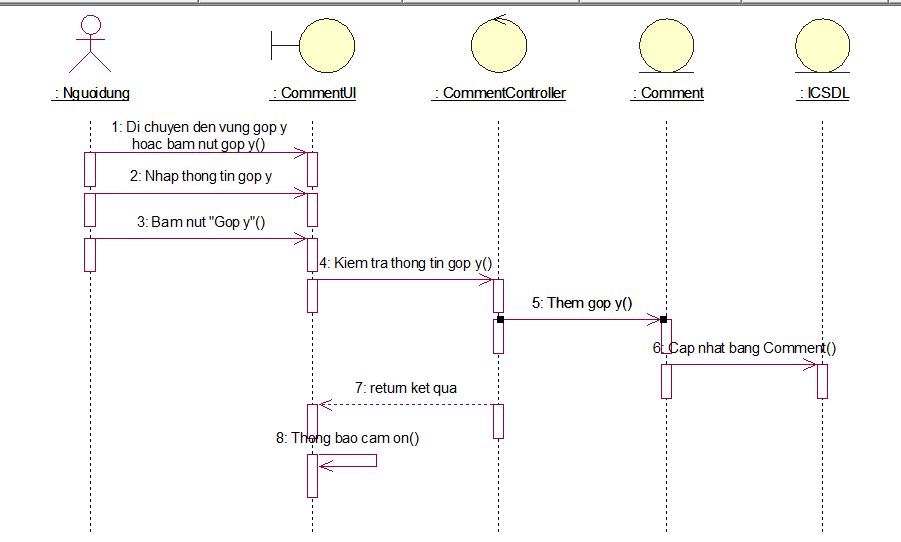
Hình 16: Biểu đồ hoạt động use case góp ý

#### Biểu đồ trạng thái

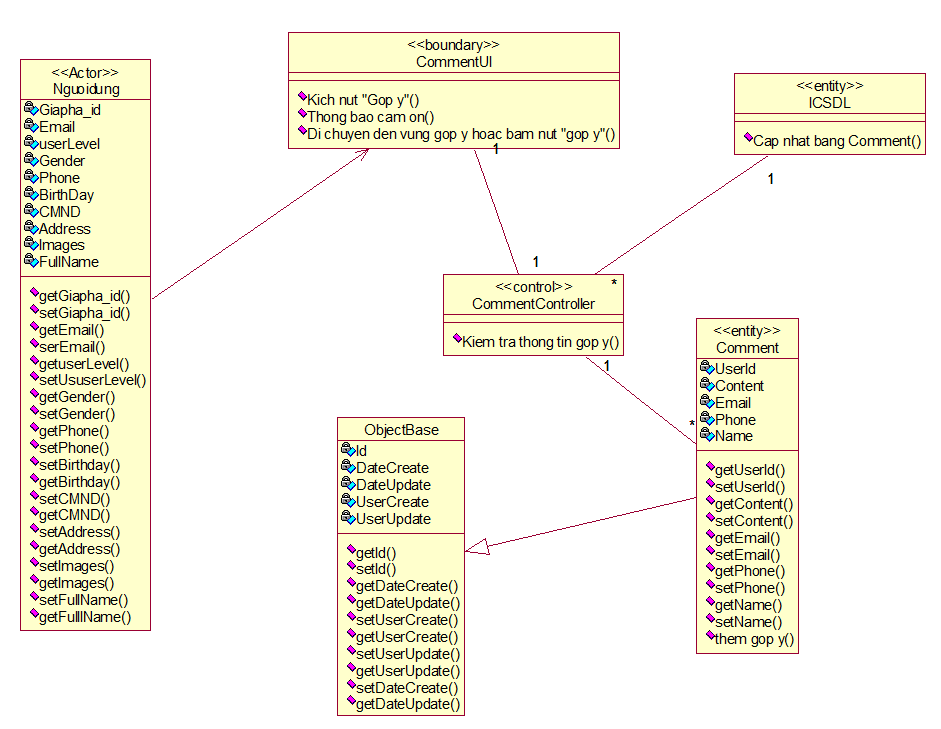


Hình 17: Biểu đồ trạng thái use case góp ý

#### Biểu đồ tuần tự



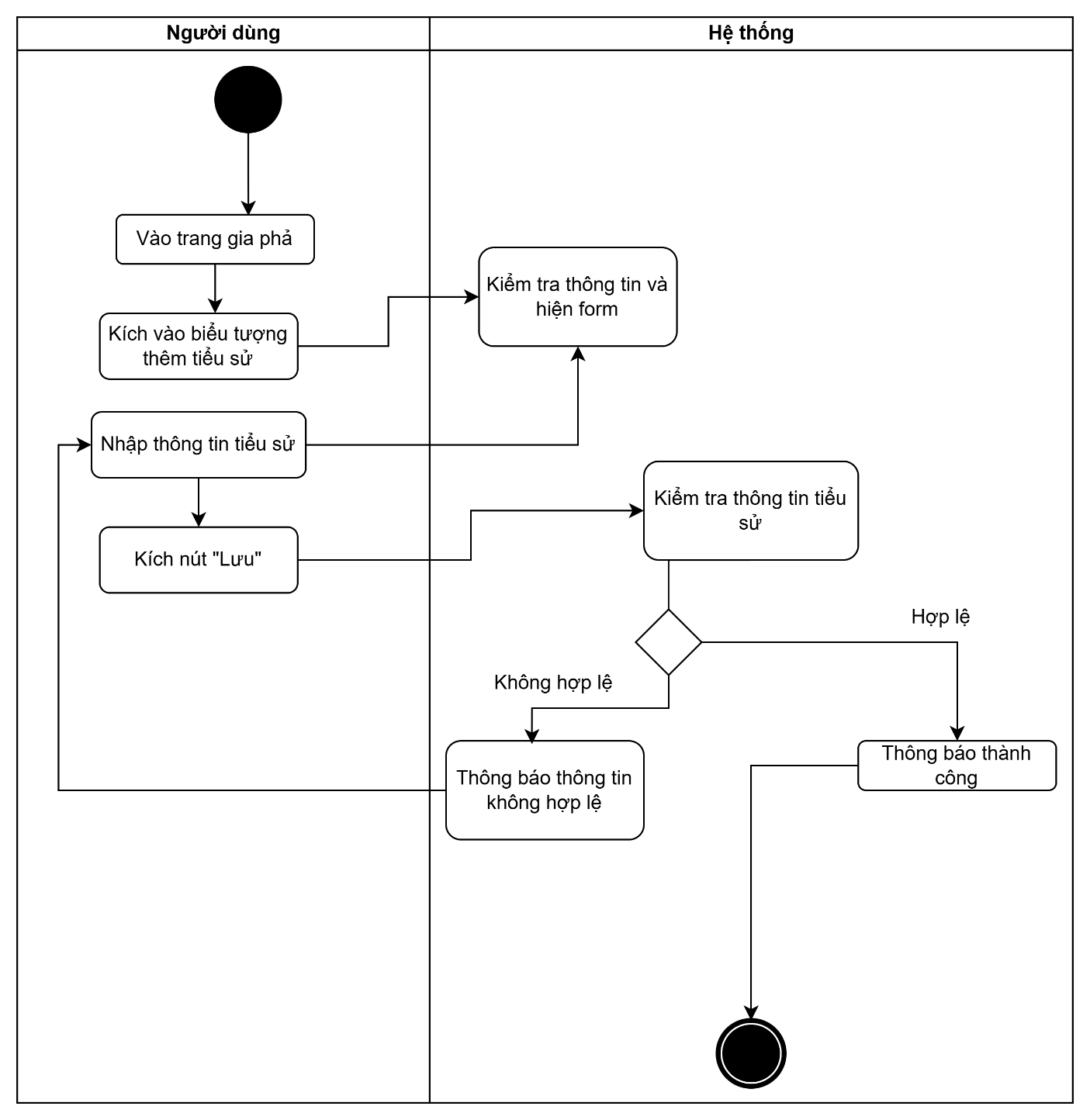
Hình : Biều đồ tuần tự use case góp ý



Hình 19: Biểu đồ lớp use case góp ý

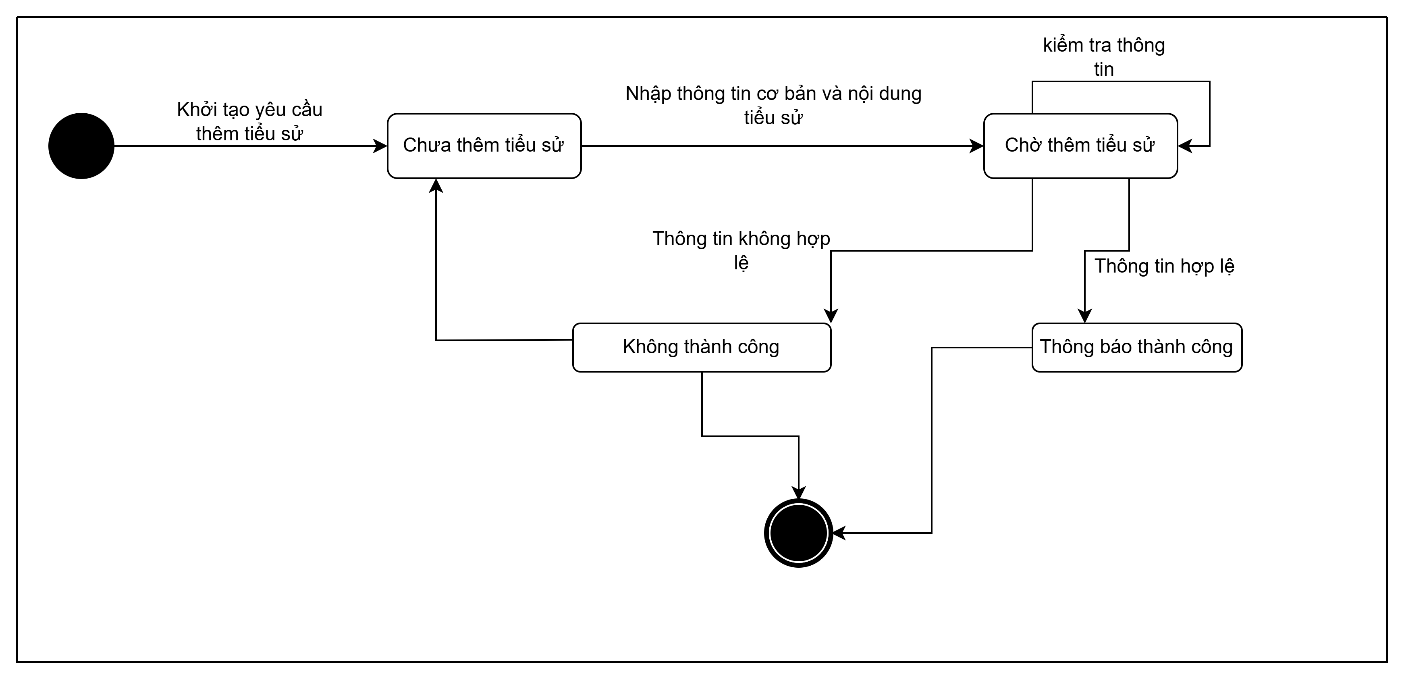
### Use case thêm tiểu sử

#### Biểu đồ hoạt động



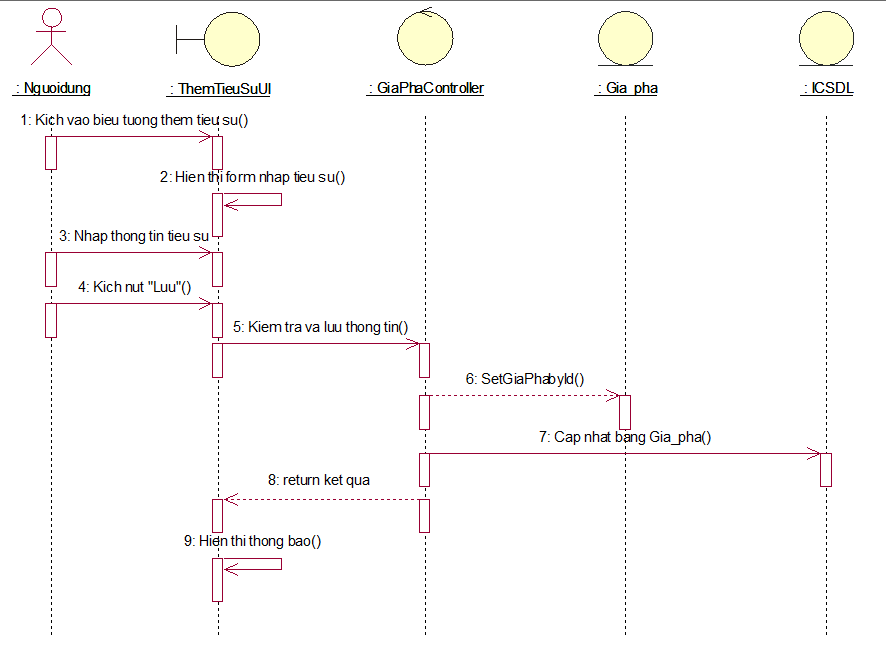
Hình : Biểu đồ hoạt động use case thêm tiểu sử

#### Biểu đồ trạng thái



Hình 21: Biểu đồ trạng thái use case thêm tiểu sử

#### Biểu đồ tuần tự



Hình 22: Biểu đồ tuần tự use case thêm tiểu sử

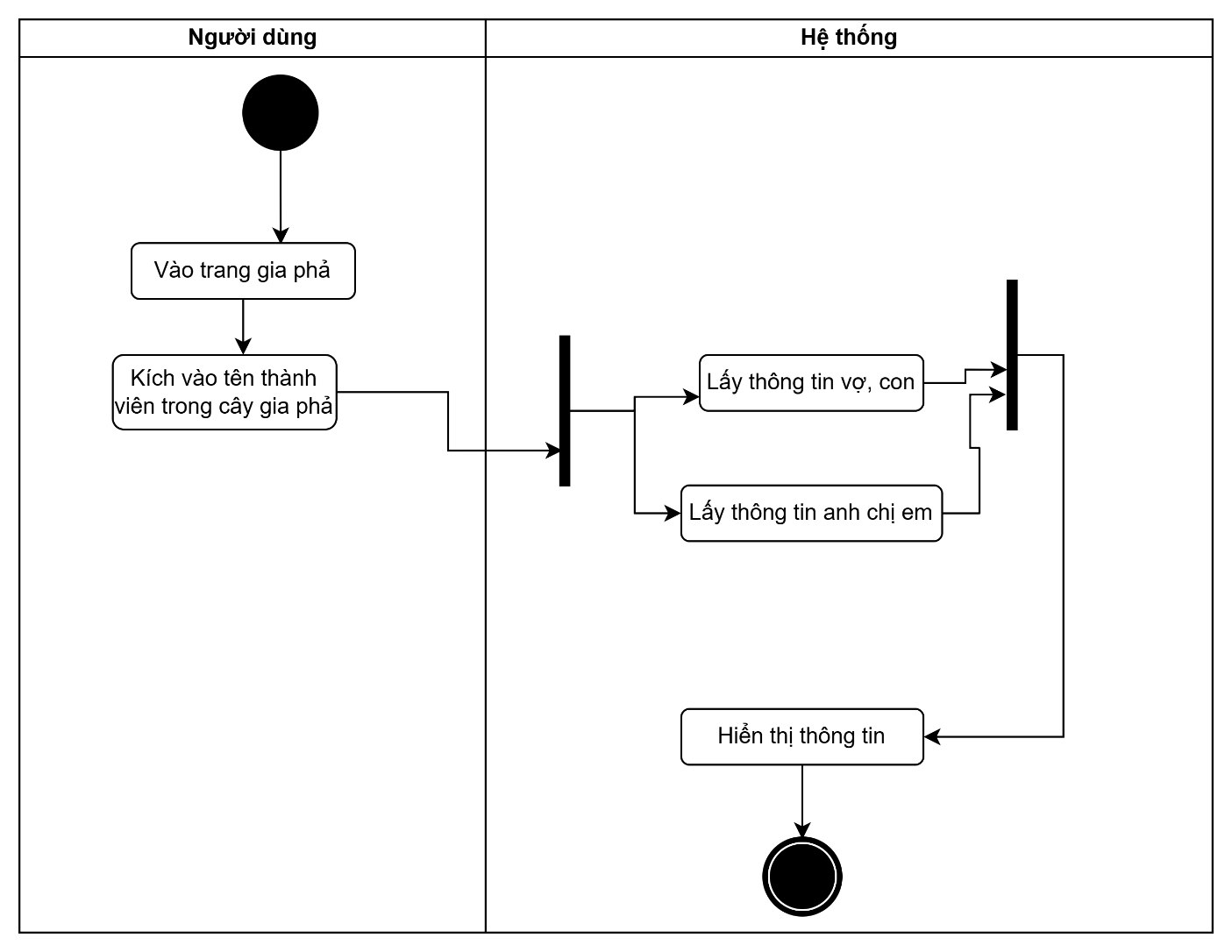
#### Biểu đồ lớp



Hình 23: Biểu đồ lớp use case thêm tiểu sử

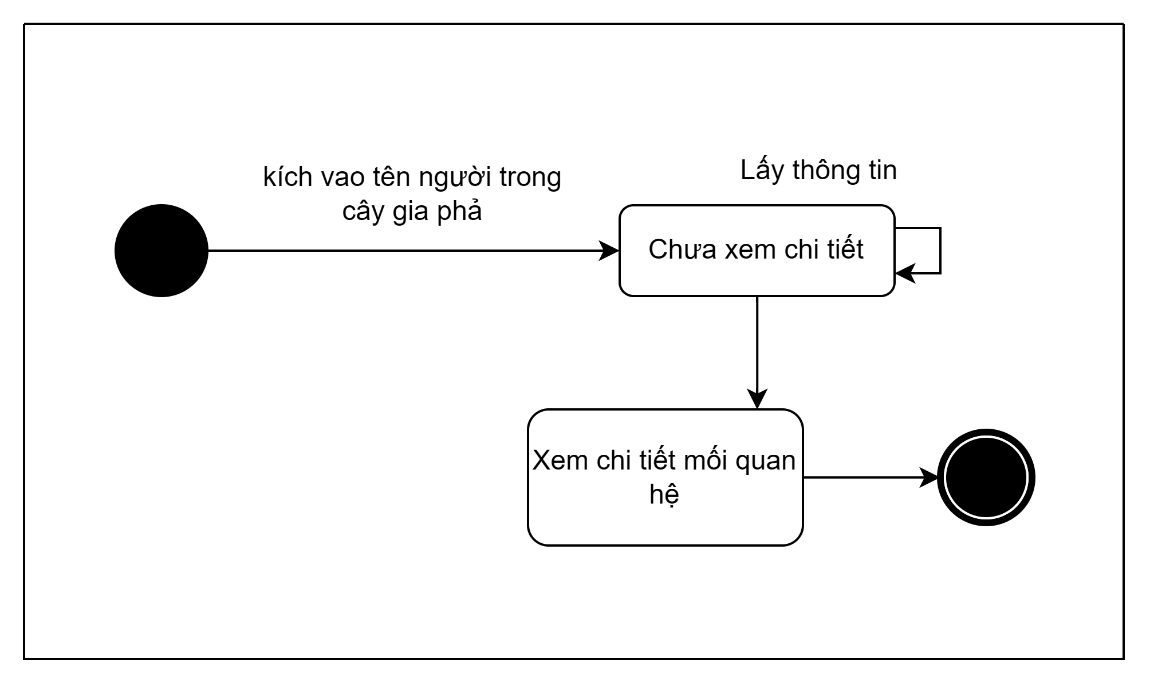
### Use case xem chi tiết mối quan hệ

#### Biểu đồ hoạt động



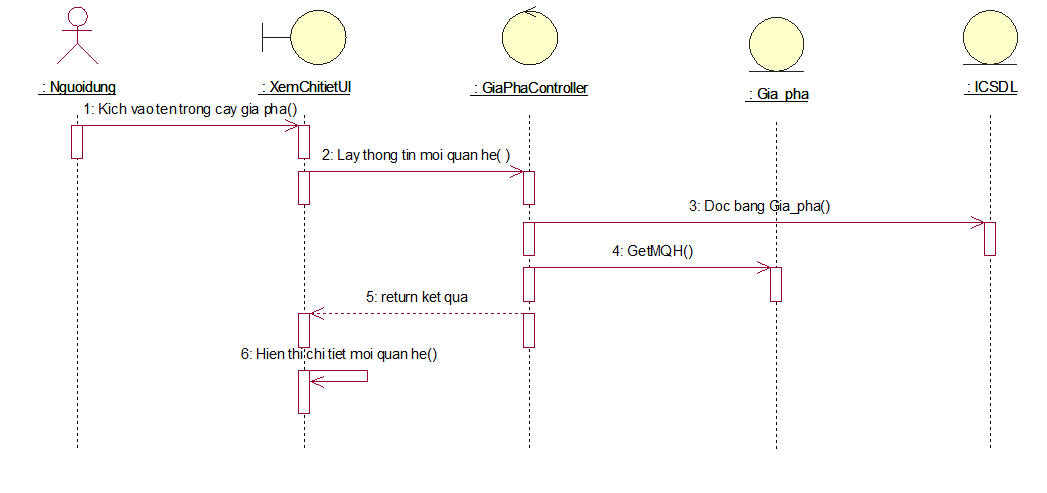
Hình : Biểu đồ hoạt động use case xem chi tiết mối quan hệ

#### Biểu đồ trạng thái



Hình : Biểu đồ trạng thái use case xem chi tiết mối quan hệ

#### Biểu đồ tuần tự



Hình : Biểu đồ tuần tự use case xem chi tiết mối quan hệ

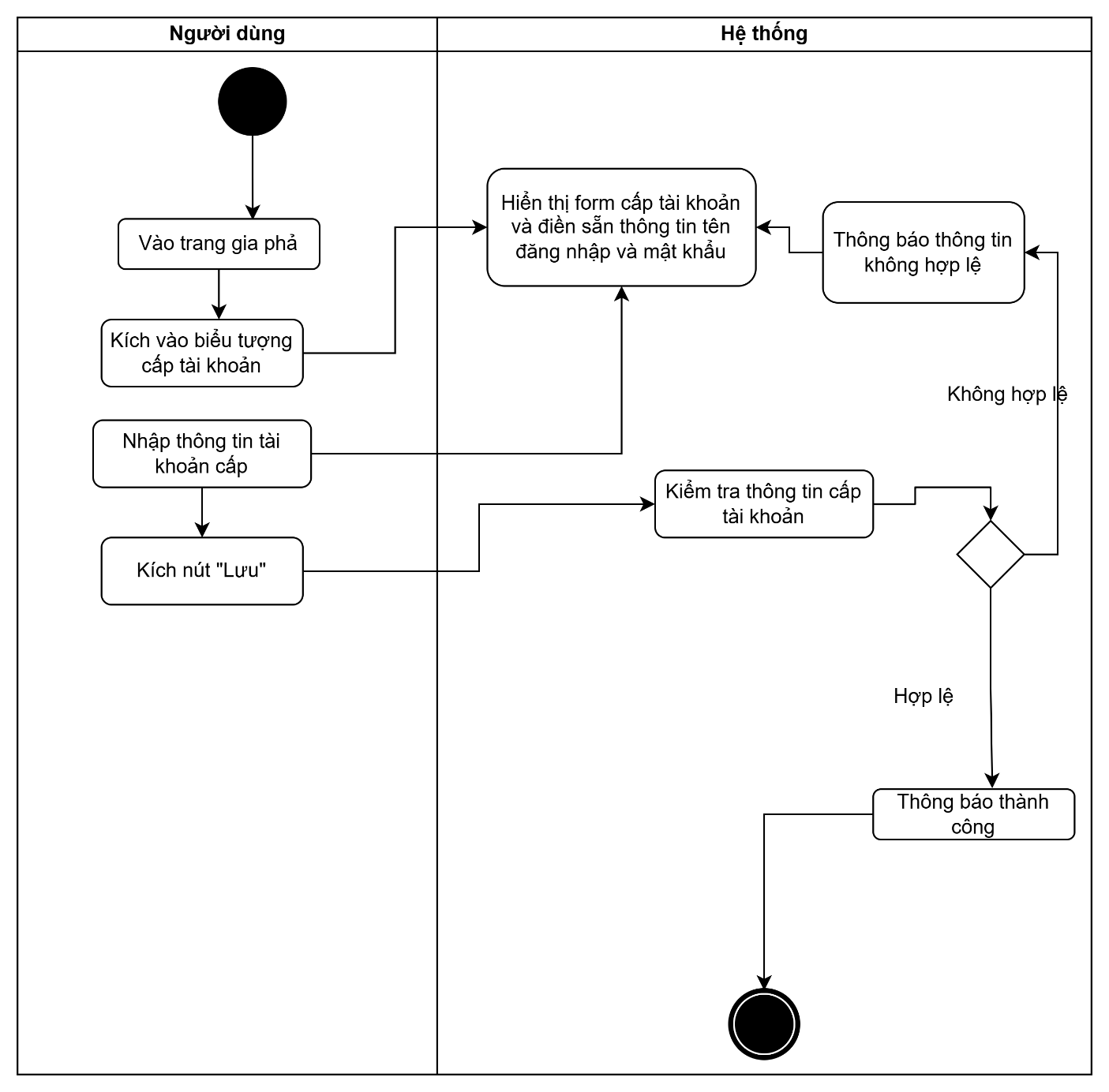
#### Biểu đồ lớp



Hình : Biểu đồ lớp use case xem chi tiết mối quan hệ

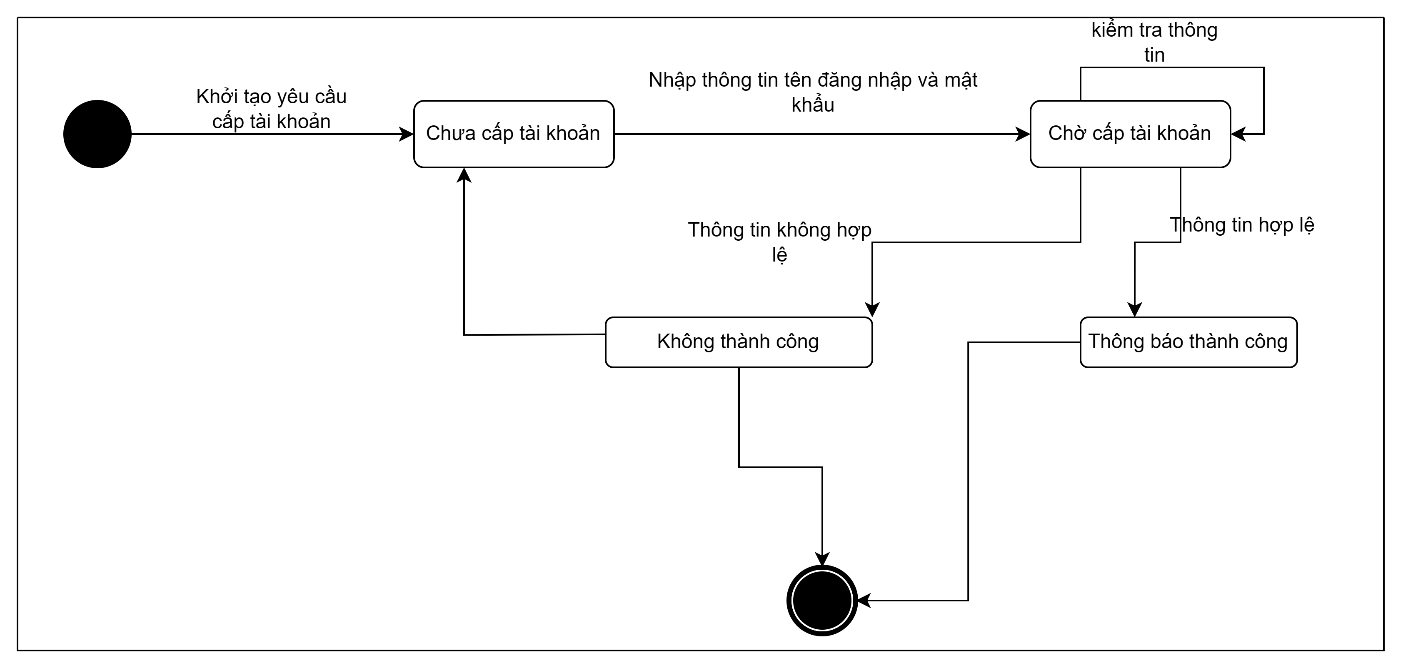
### Use case cấp tài khoản

#### Biểu đồ hoạt động



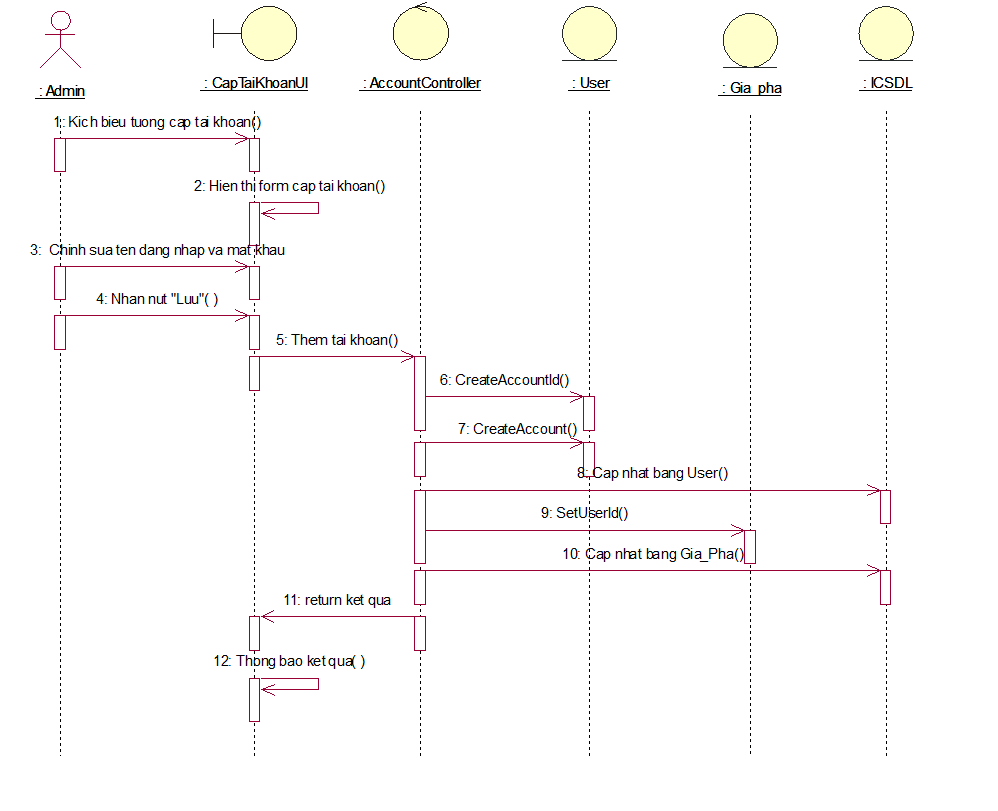
Hình 28: Biểu đồ hoạt động use case cấp tài khoản

#### Biểu đồ trạng thái



Hình 29: Biểu đồ trạng thái use case cấp tài khoản

#### Biểu đồ tuần tự



Hình 30: Biểu đồ tuần tự use case cấp tài khoản

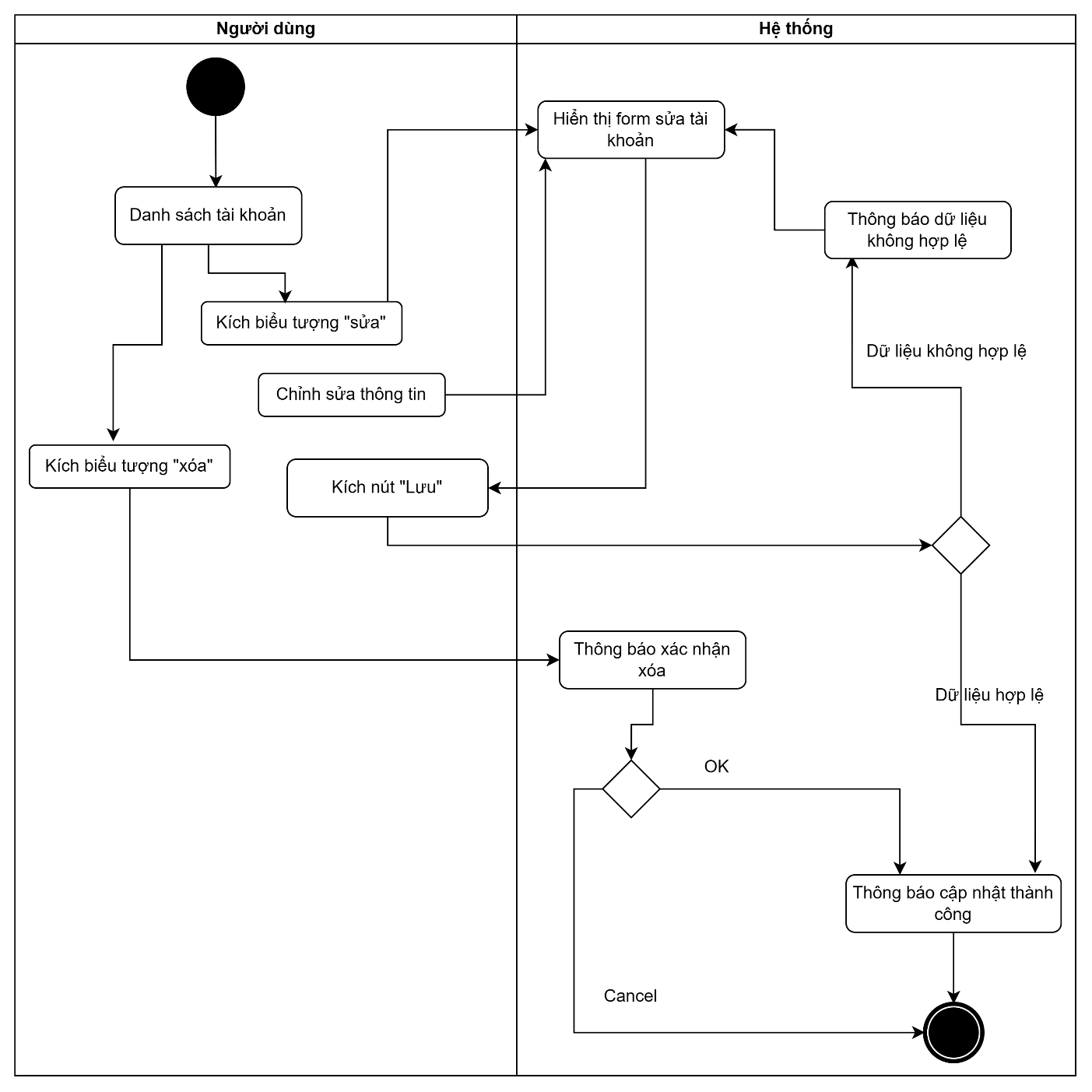
#### Biểu đồ lớp



Hình 31: Biểu đồ lớp use case cấp tài khoản

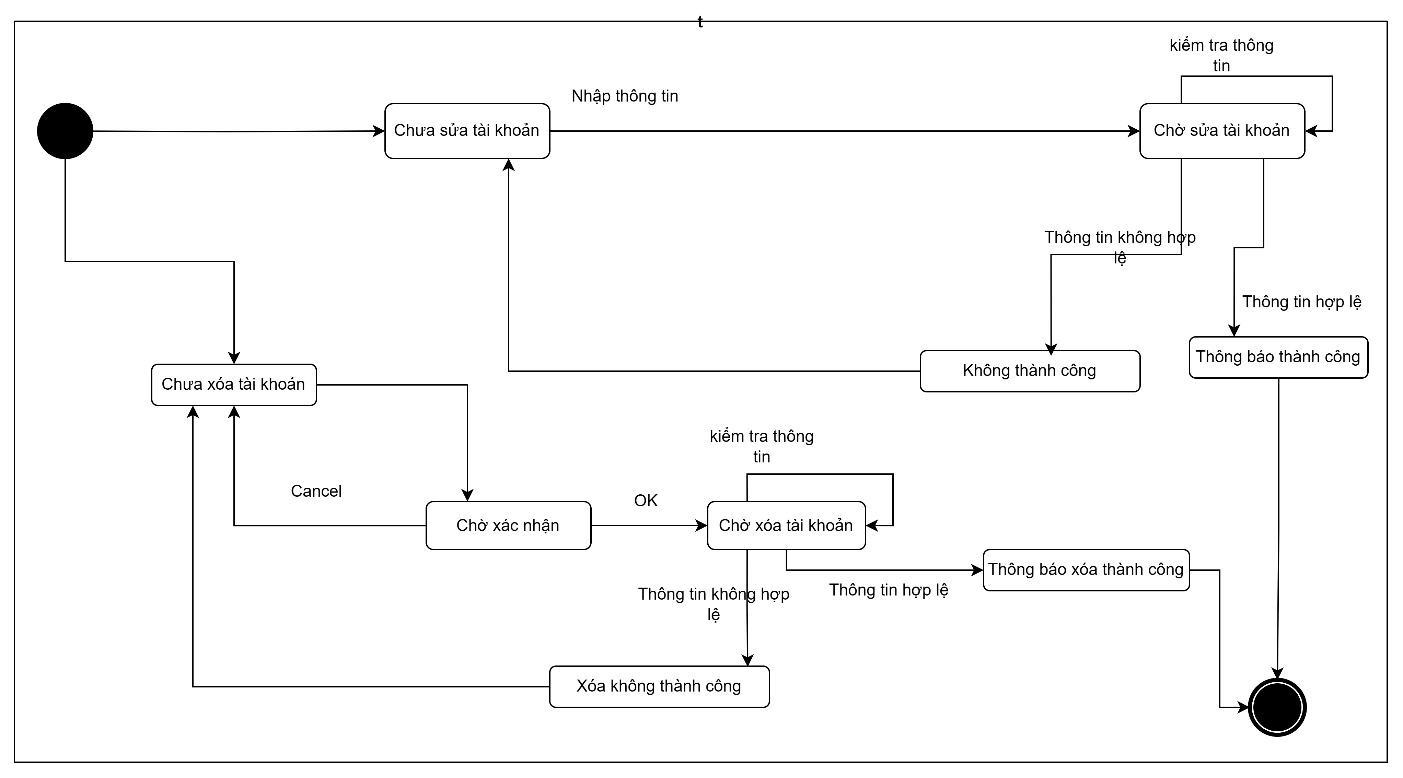
### Use case quản lý tài khoản

#### Biểu đồ hoạt động



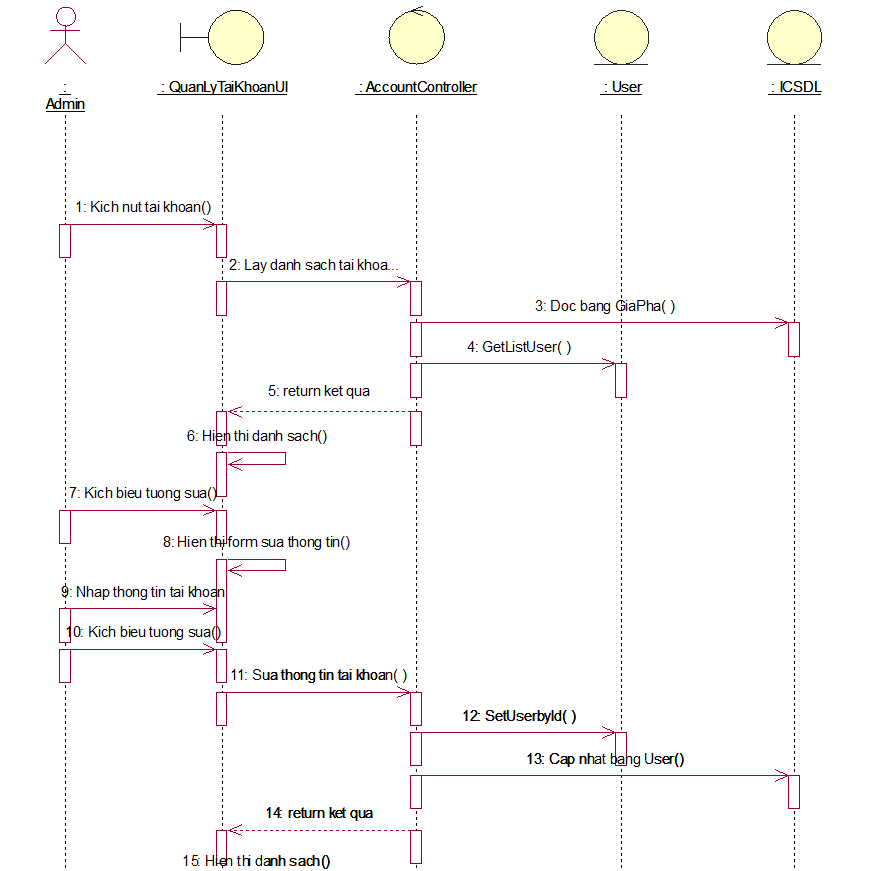
Hình : Biểu đồ hoạt động quản lý tài khoản

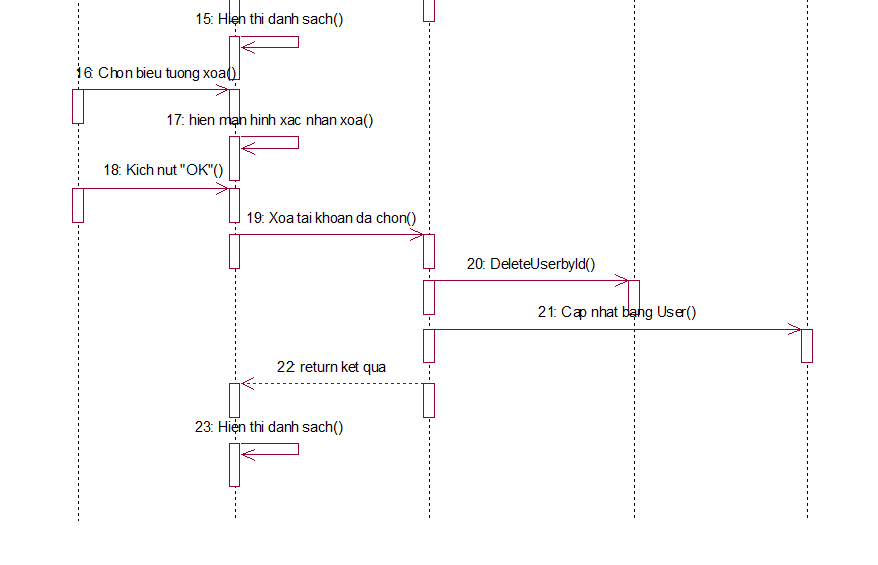
#### Biểu đồ trạng thái



Hình : Biểu đồ trạng thái use case quản lý tài khoản

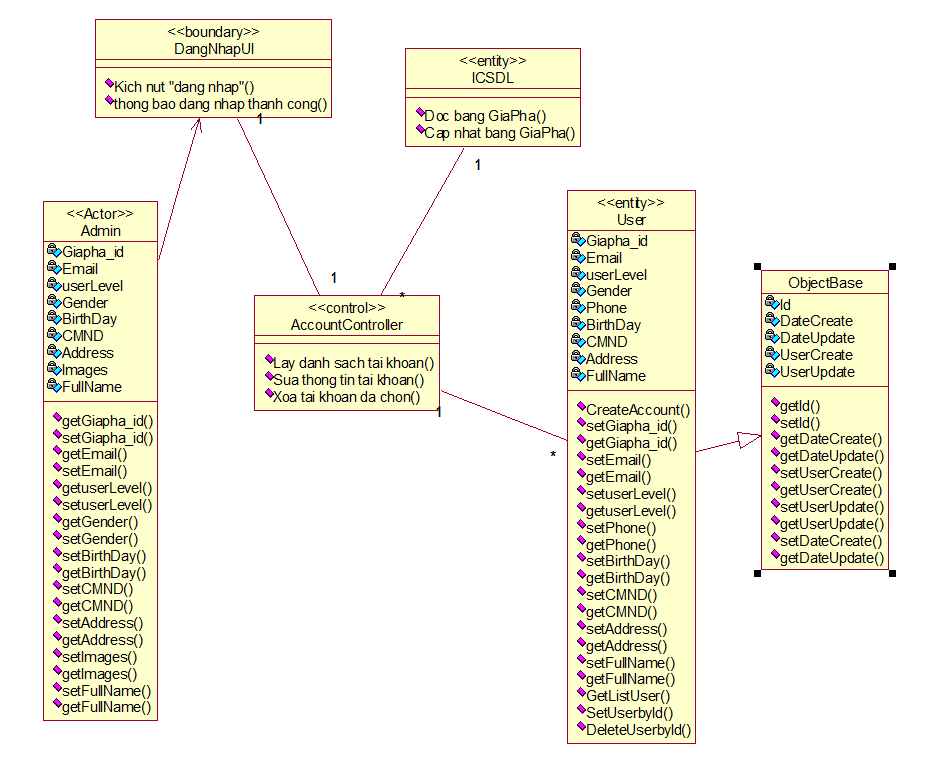
#### Biểu đồ tuần tự





Hình : Biểu đồ tuân tự use case quản lý tài khoản

#### Biểu đồ lớp

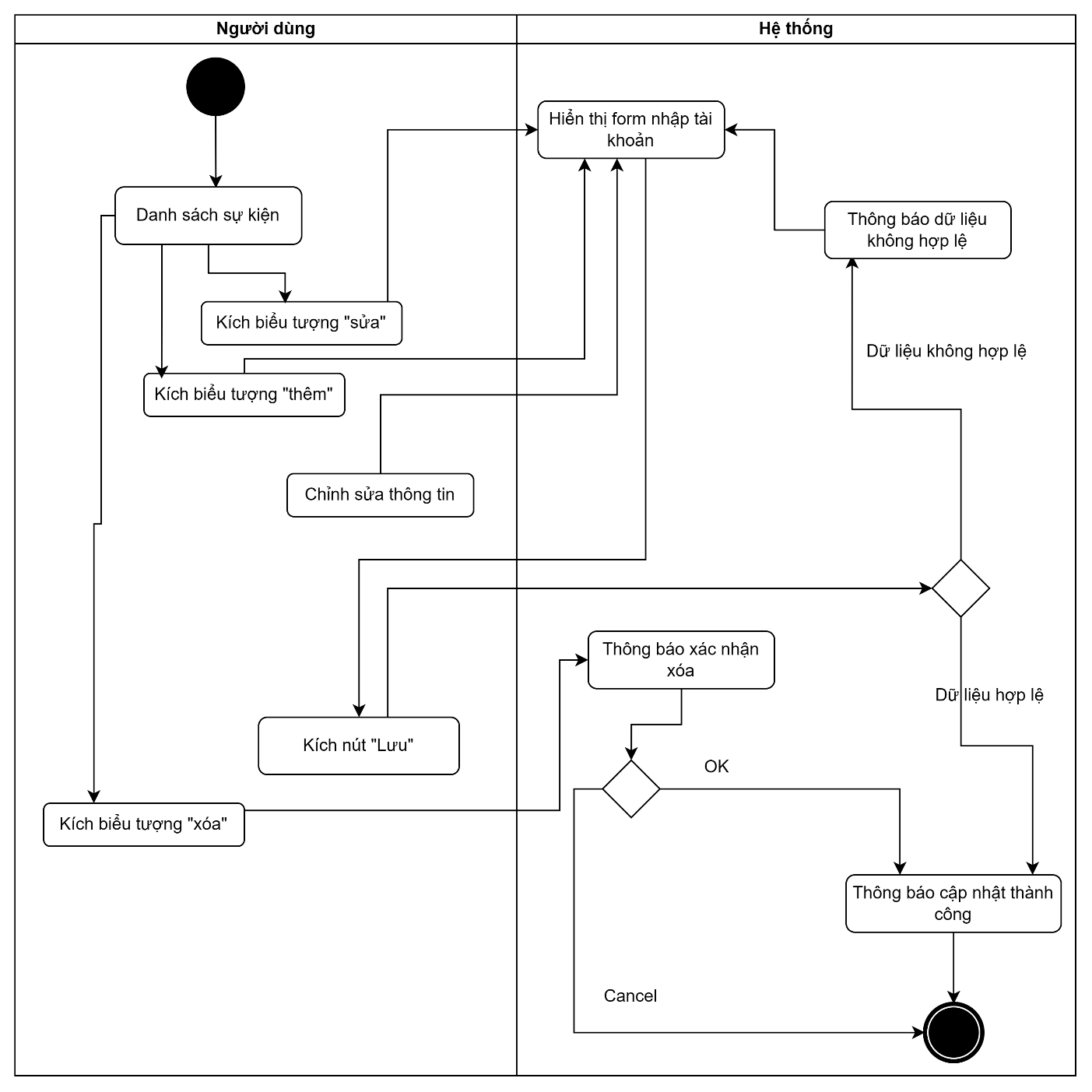


Hình 35: Biểu đồ lớp use case quản lý tài khoản

=

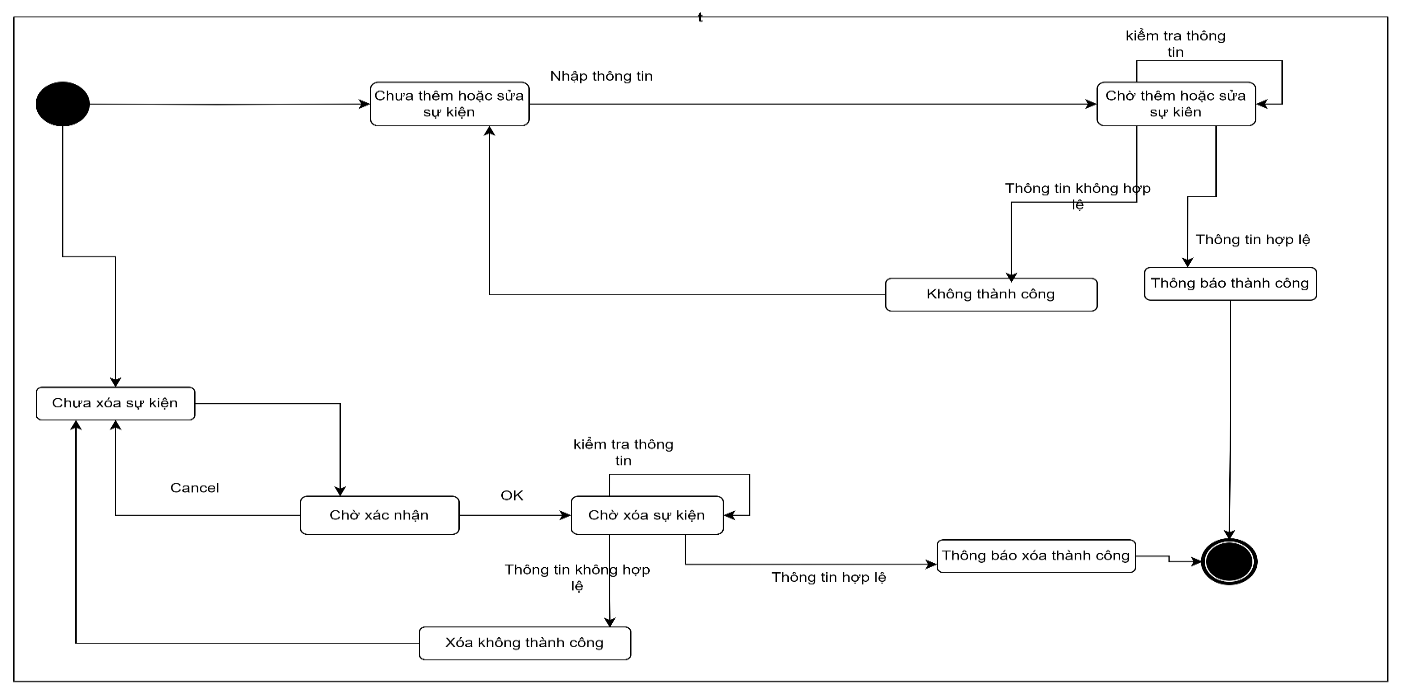
### Use case quản lý sự kiện

#### Biểu đồ hoạt động



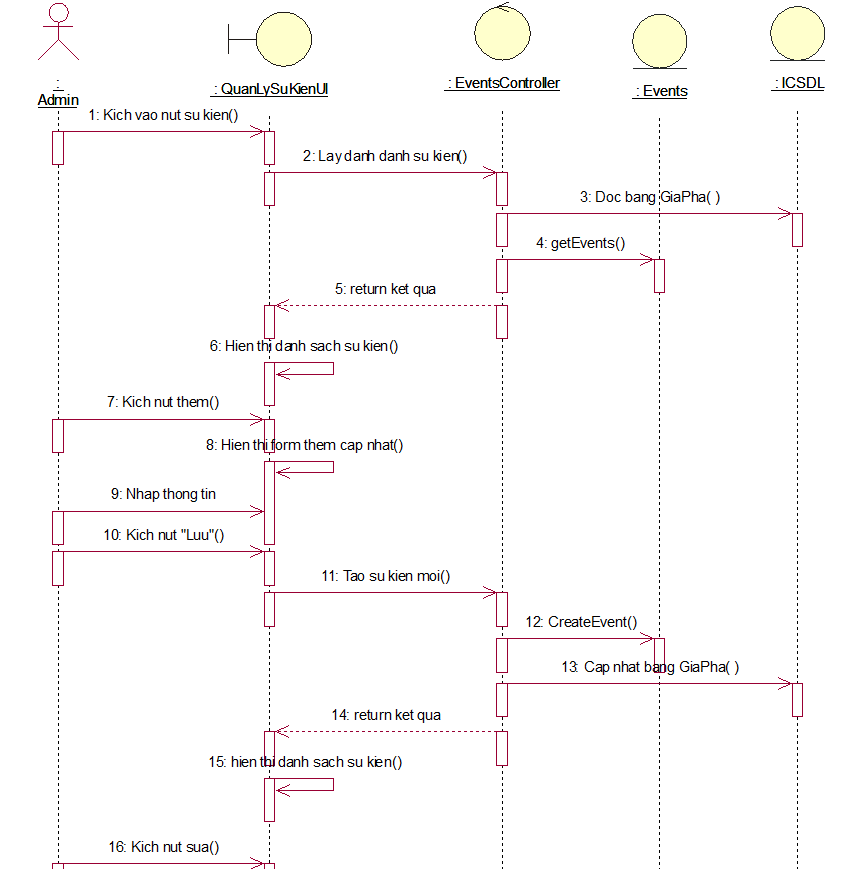
Hình : Biểu đồ hoạt động use case quản lý sự kiện

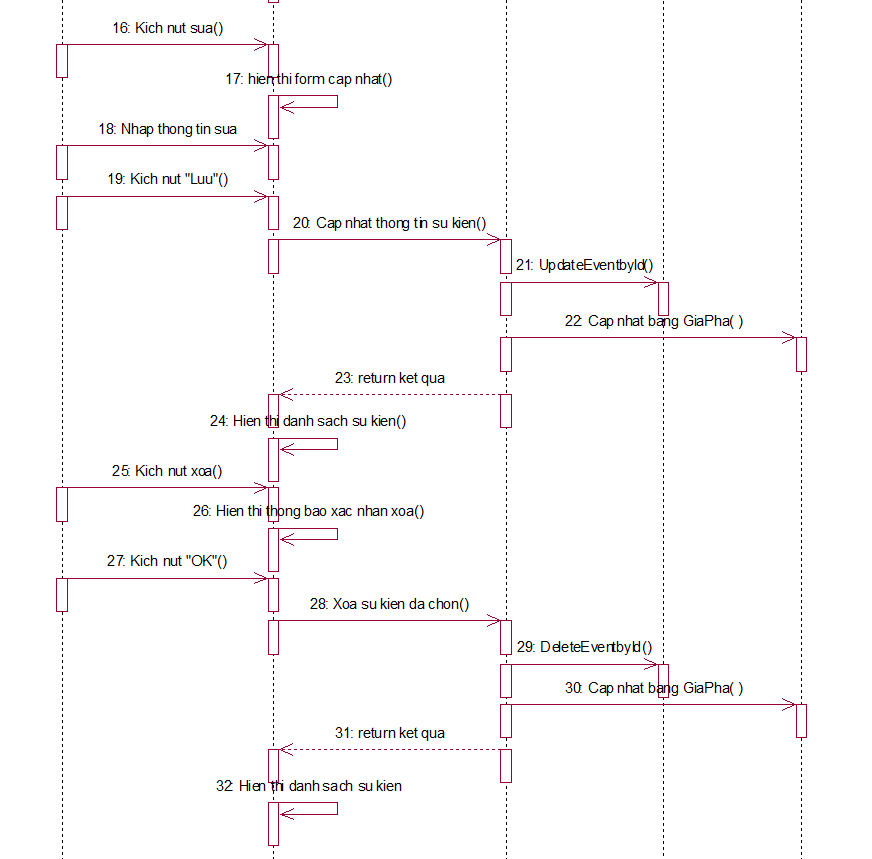
#### Biểu đồ trạng thái



Hình : Biểu đồ trạng thái use case quản lý sự kiện

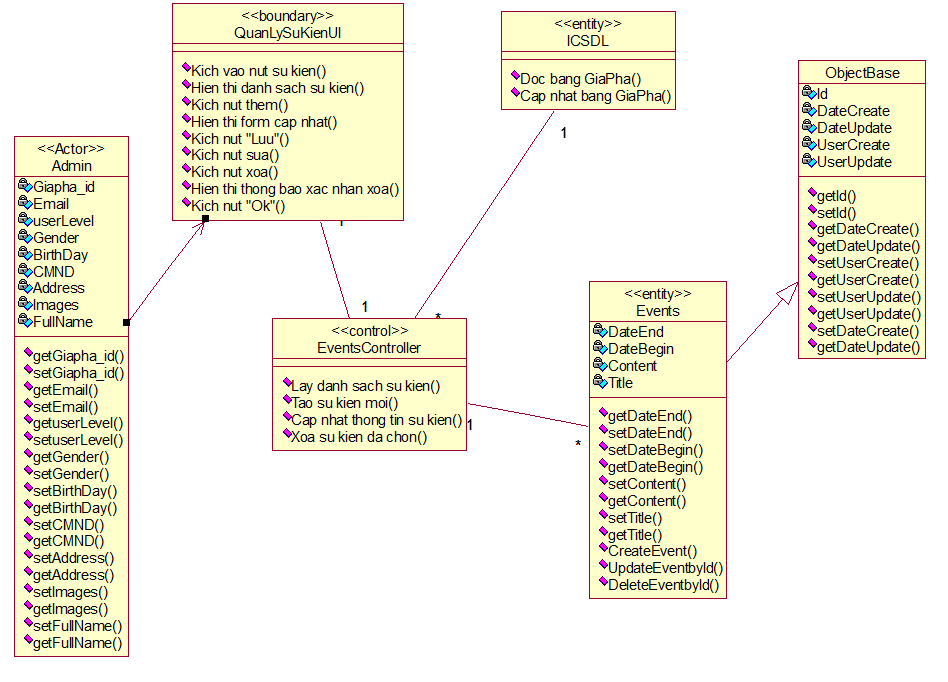
#### Biểu đồ tuần tự





Hình : Biểu đồ tuần tự use case quản lý sự kiện

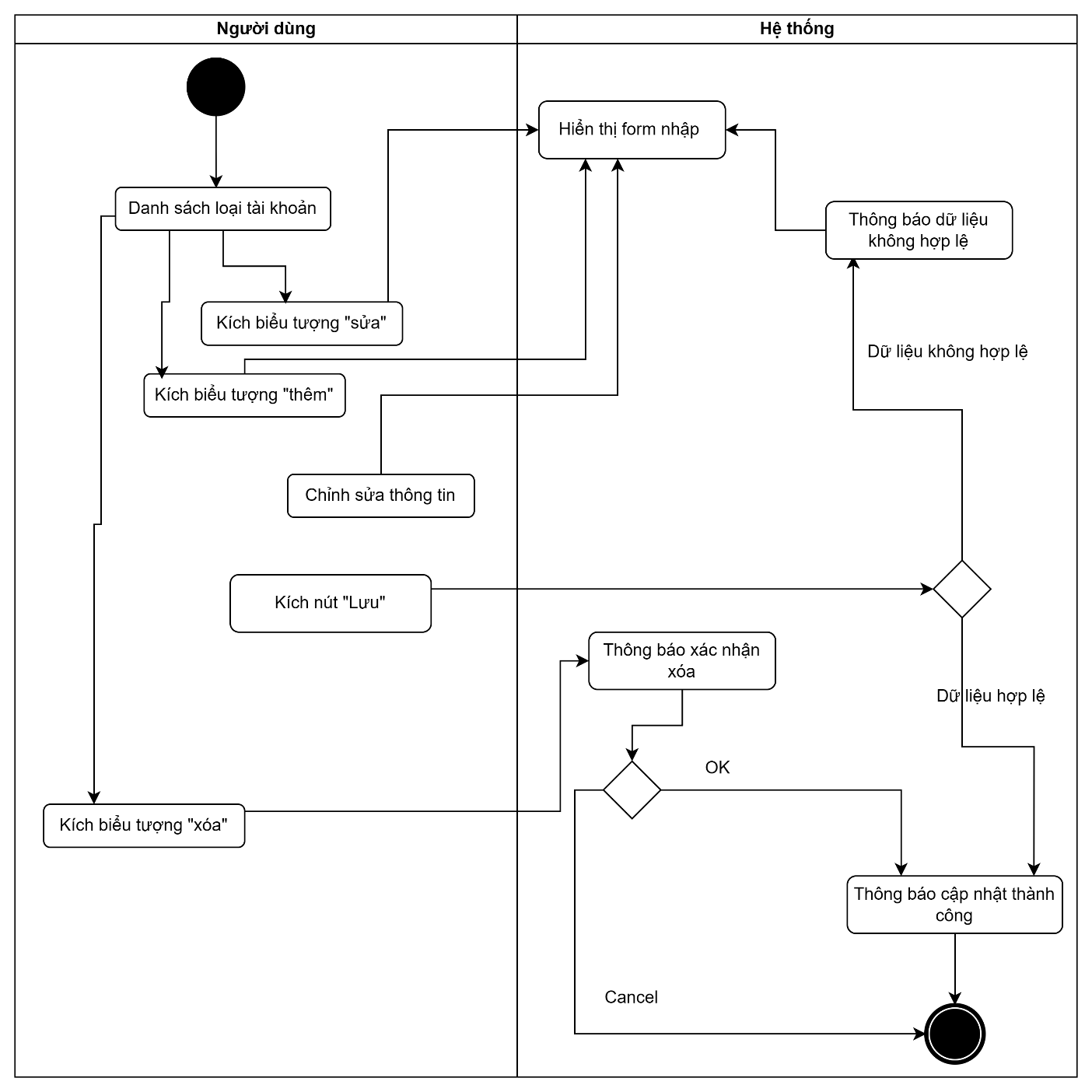
#### Biểu đồ lớp



Hình 39: Biểu đồ lớp use case quản lý sự kiện

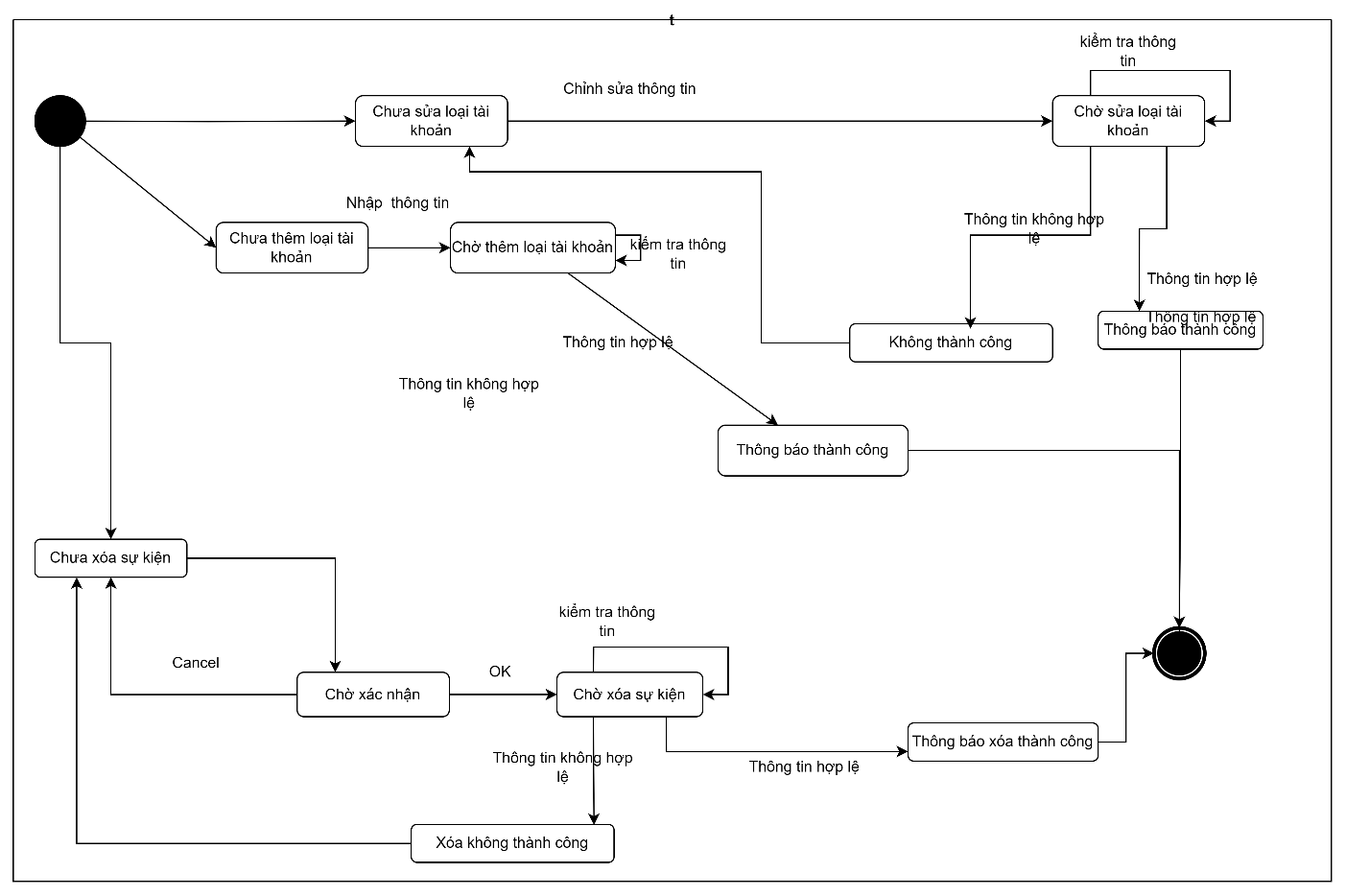
### Use case quản lý loại tài khoản

#### Biểu đồ hoạt động



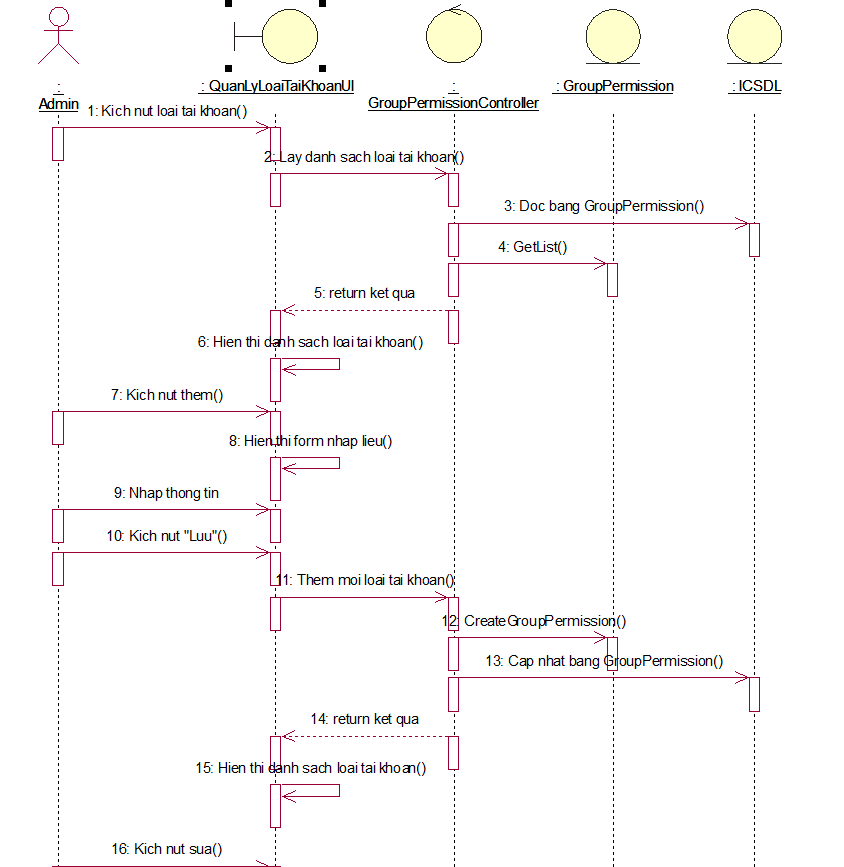
Hình : Biểu đồ hoạt động use case quản lý loại tài khoản

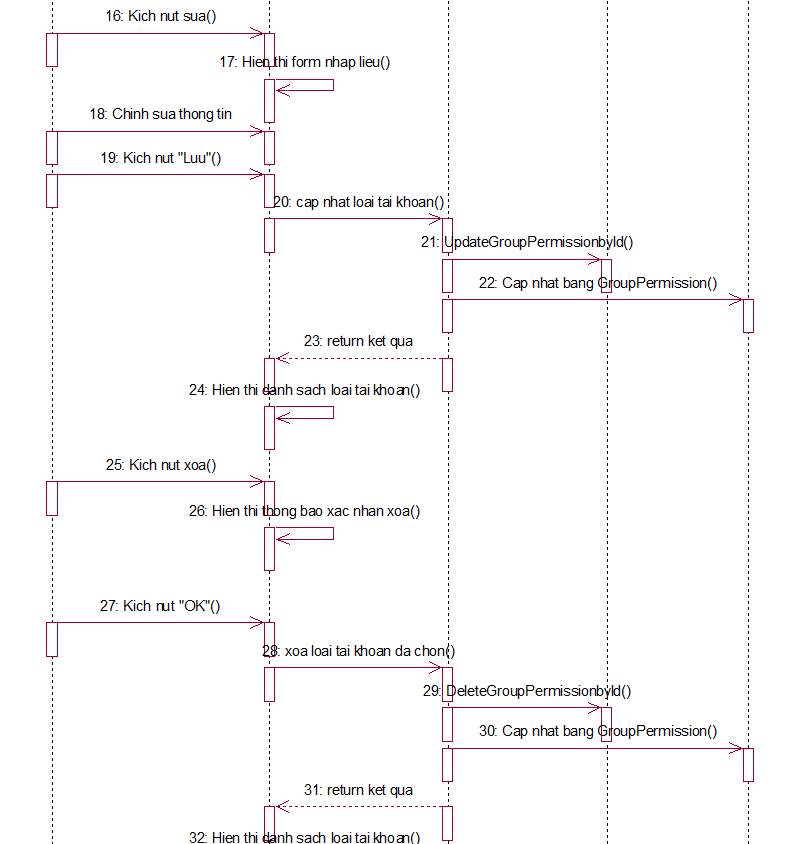
#### Biểu đồ trạng thái



Hình : Biểu đồ trạng thái hoạt động use case quản lý loại tài khoản

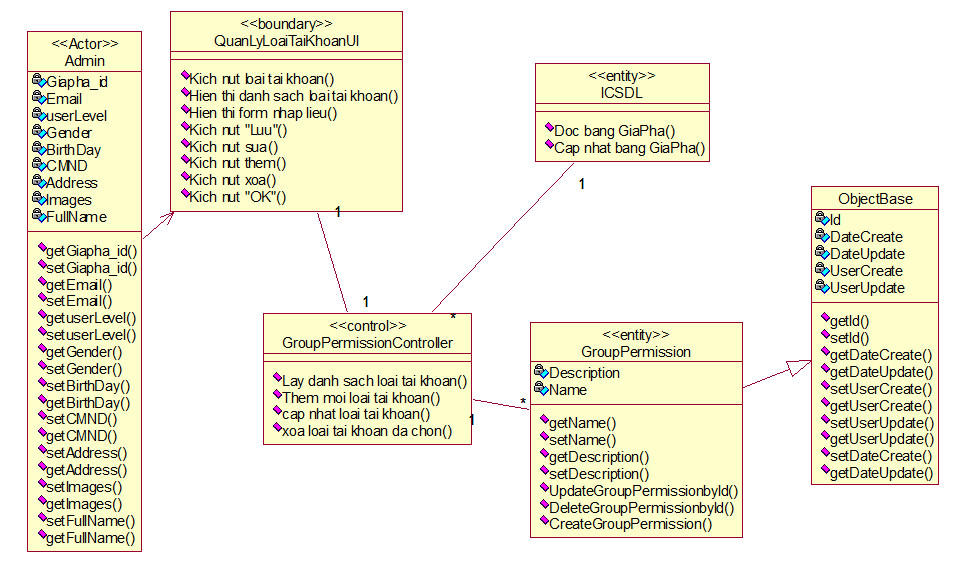
#### Biểu đồ tuần tự





Hình : Biểu đồ tuần tự use case quản lý loại tài khoản

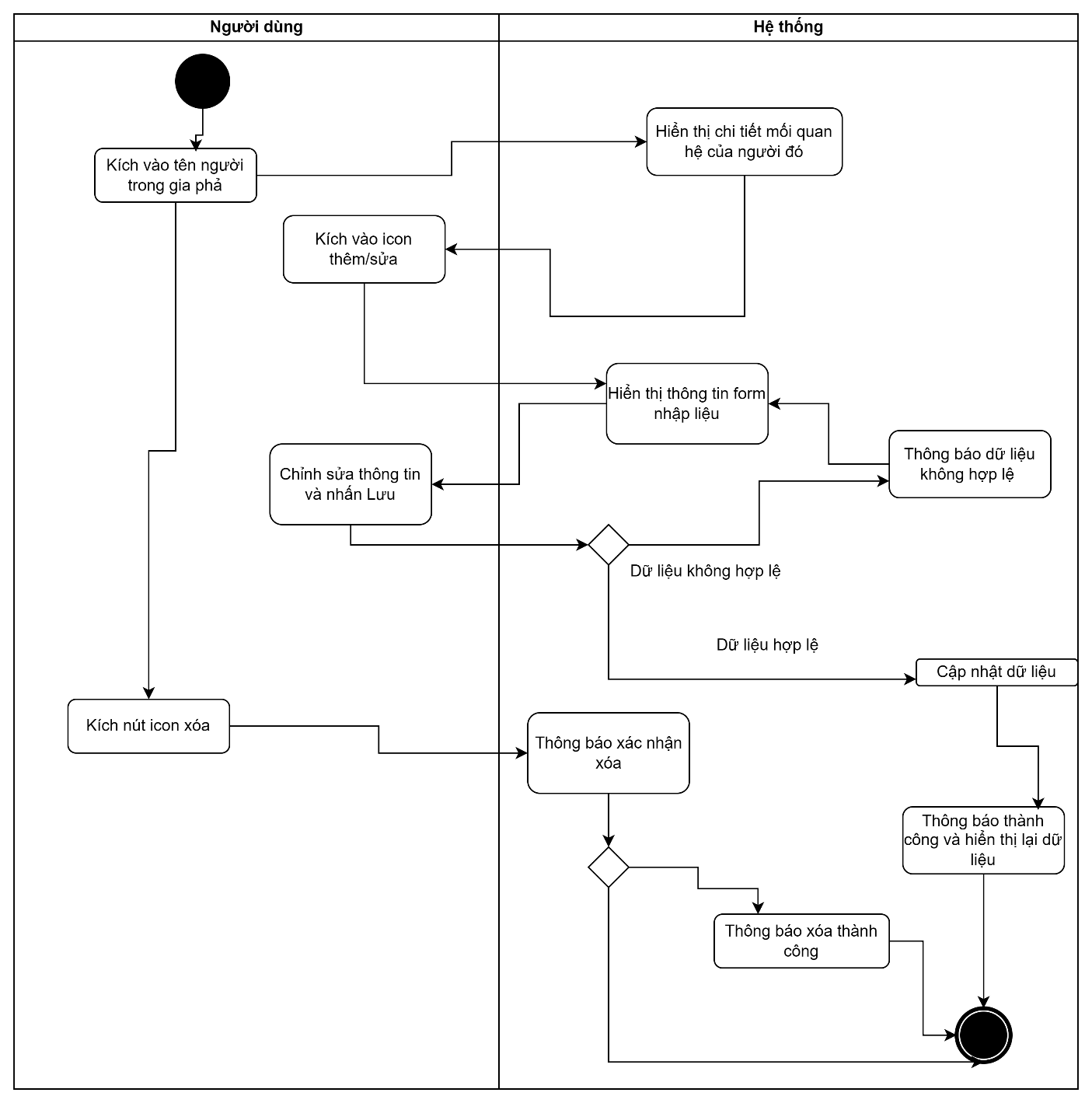
#### Biểu đồ lớp



Hình : Biểu đồ lớp use case quản lý loại tài khoản

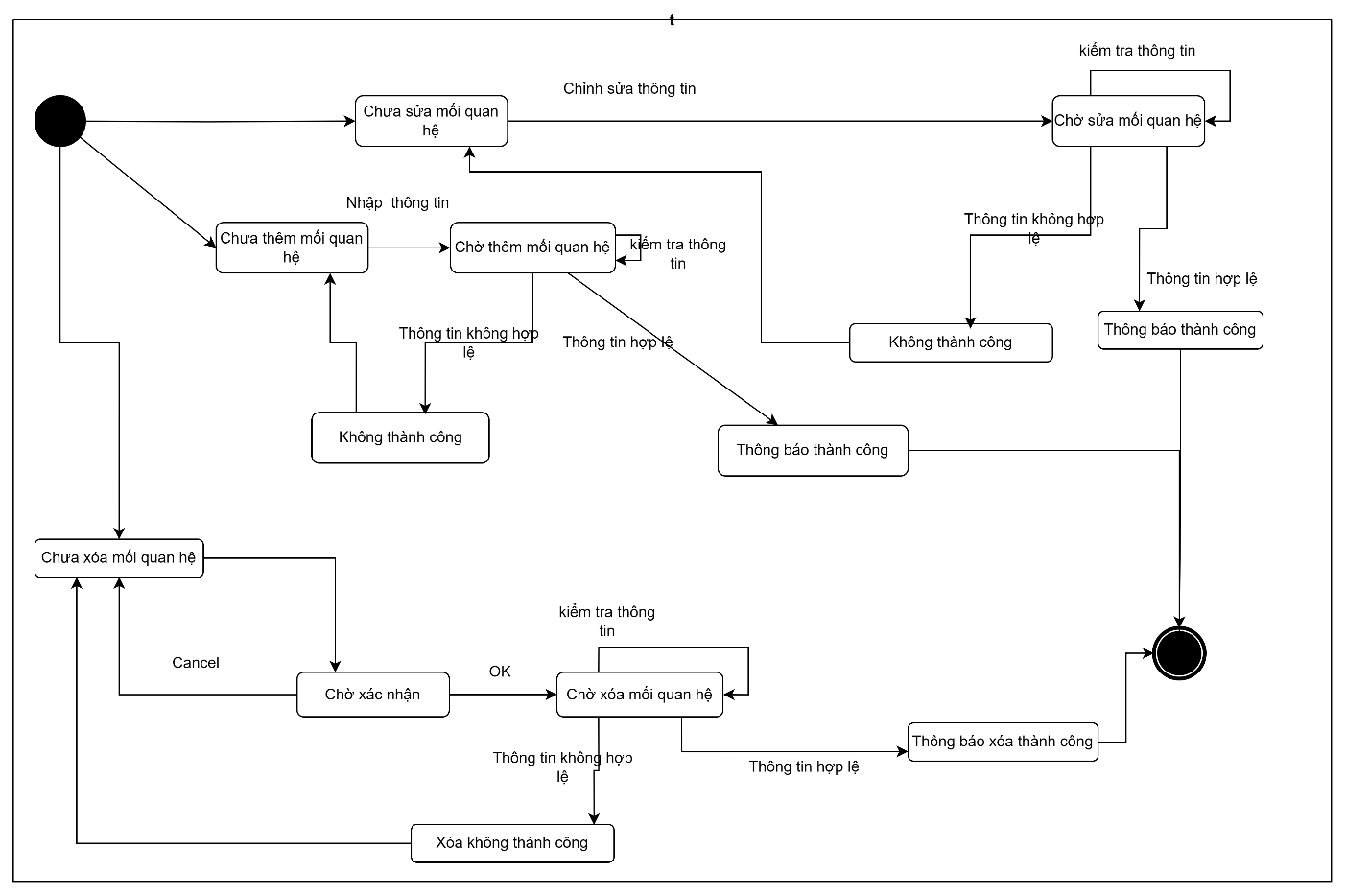
### Use case quản lý mối quan hệ

#### Biểu đồ hoạt động

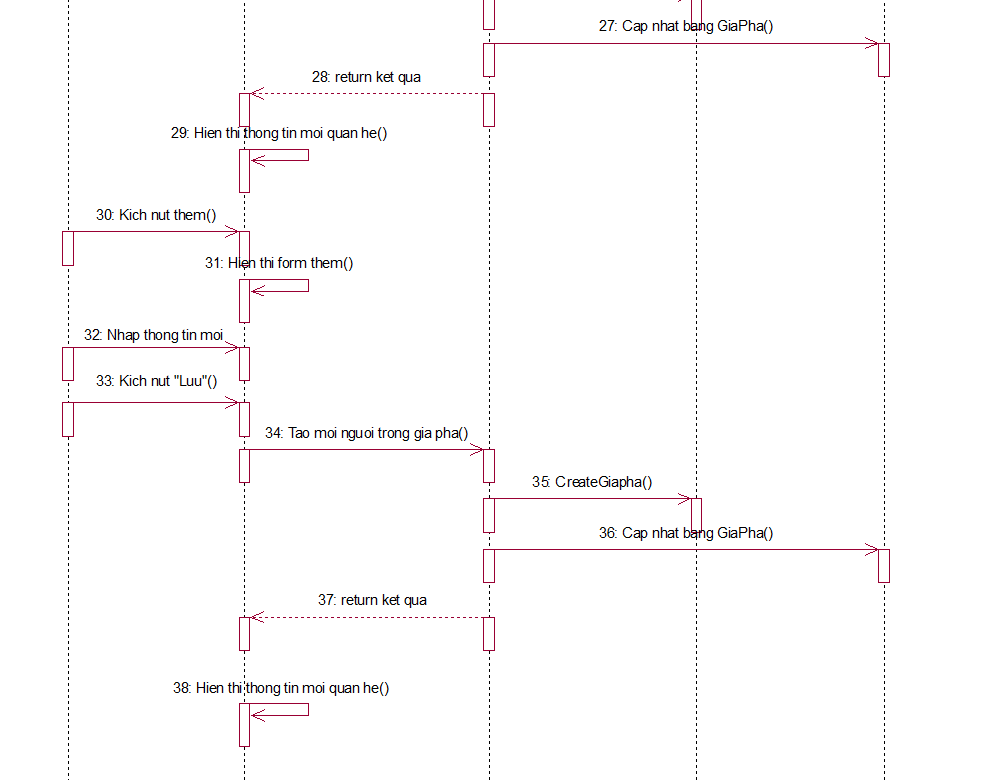
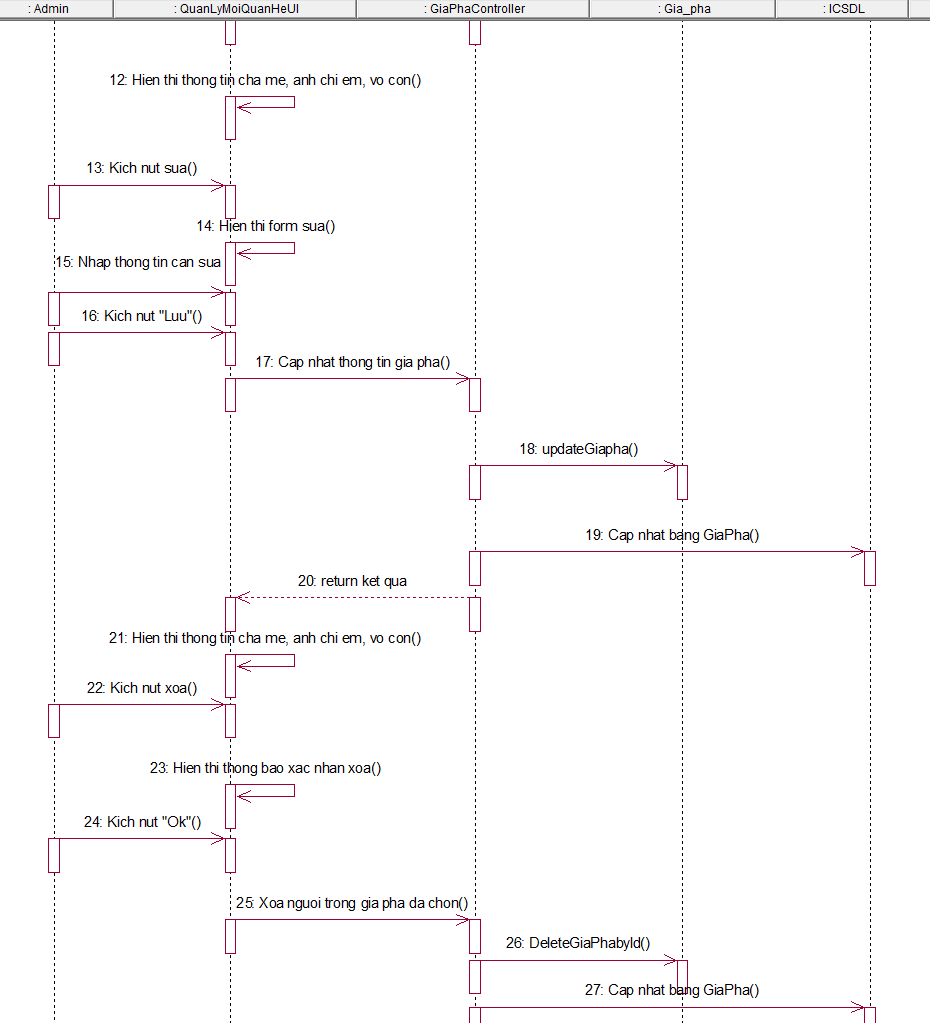
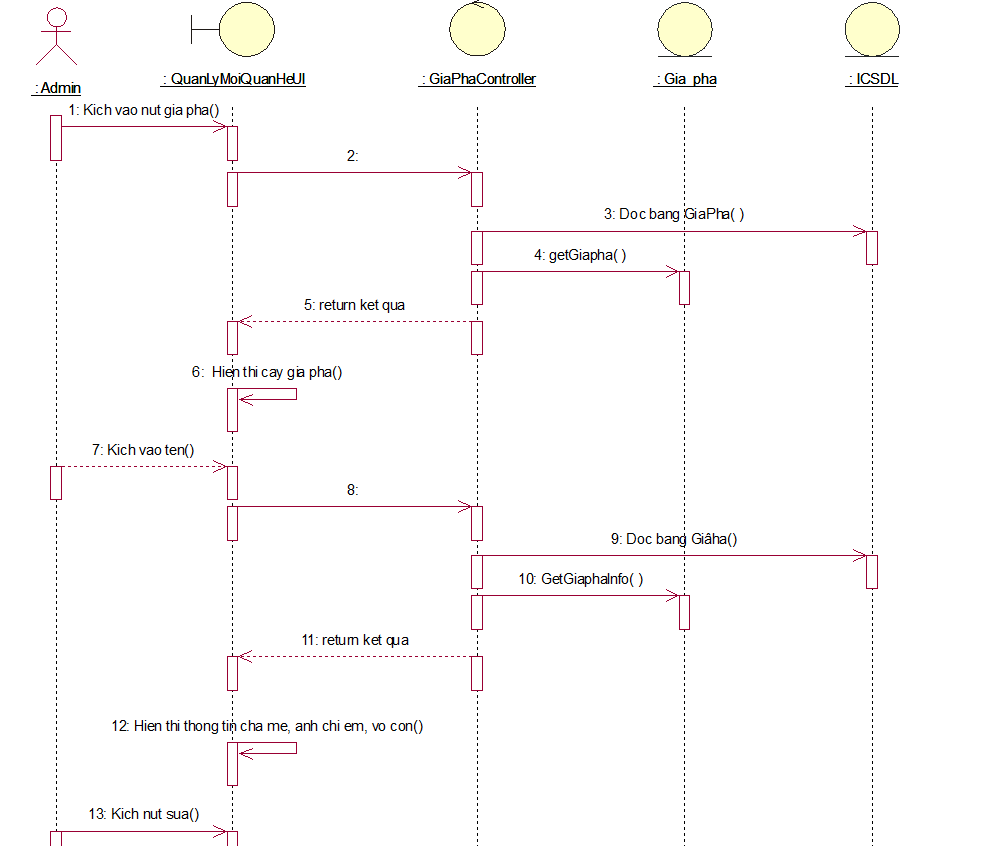


Hình : Biểu đồ hoạt động use case quản lý mối quan hệ

#### Biểu đồ trạng thái



Hình : Biểu đồ trạng thái use case quản lý mối quan hệ



Hình : Biểu đồ tuần tự use case quản lý mối quan hệ

#### Biểu đồ lớp



Hình 47: Biểu đồ lớp use case quản lý mối quan hệ

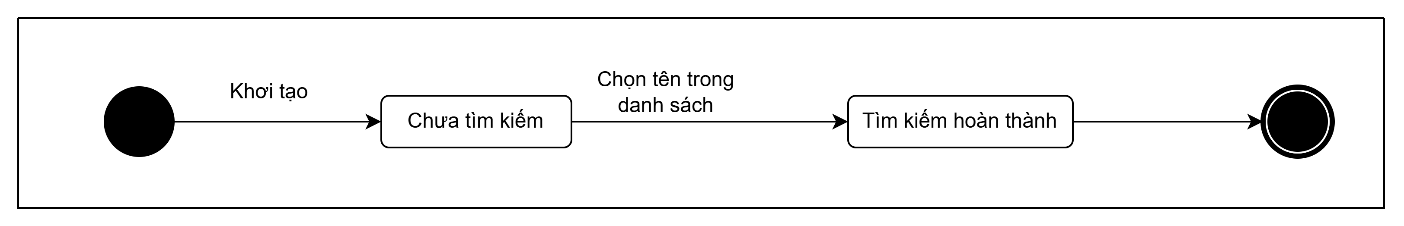
### Use case tìm kiếm người trong gia phả

#### Biểu đồ hoạt động



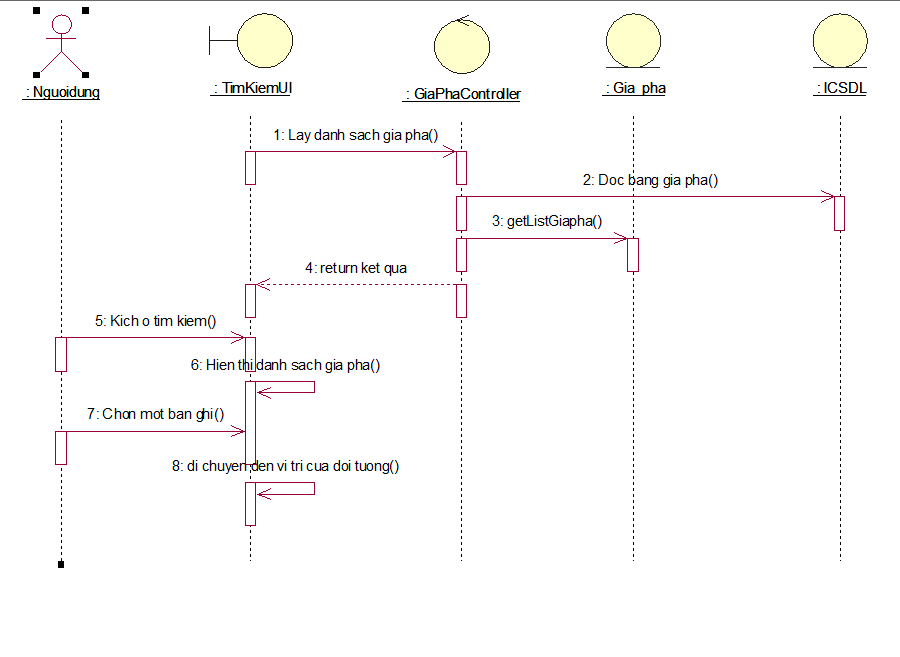
Hình 48: Biểu đồ hoạt động use case tìm kiếm người trong gia phả

#### Biểu đồ trạng thái



Hình 49: Biểu đồ trạng thái use case tìm kiếm người trong gia phả

#### Biểu đồ tuần tự



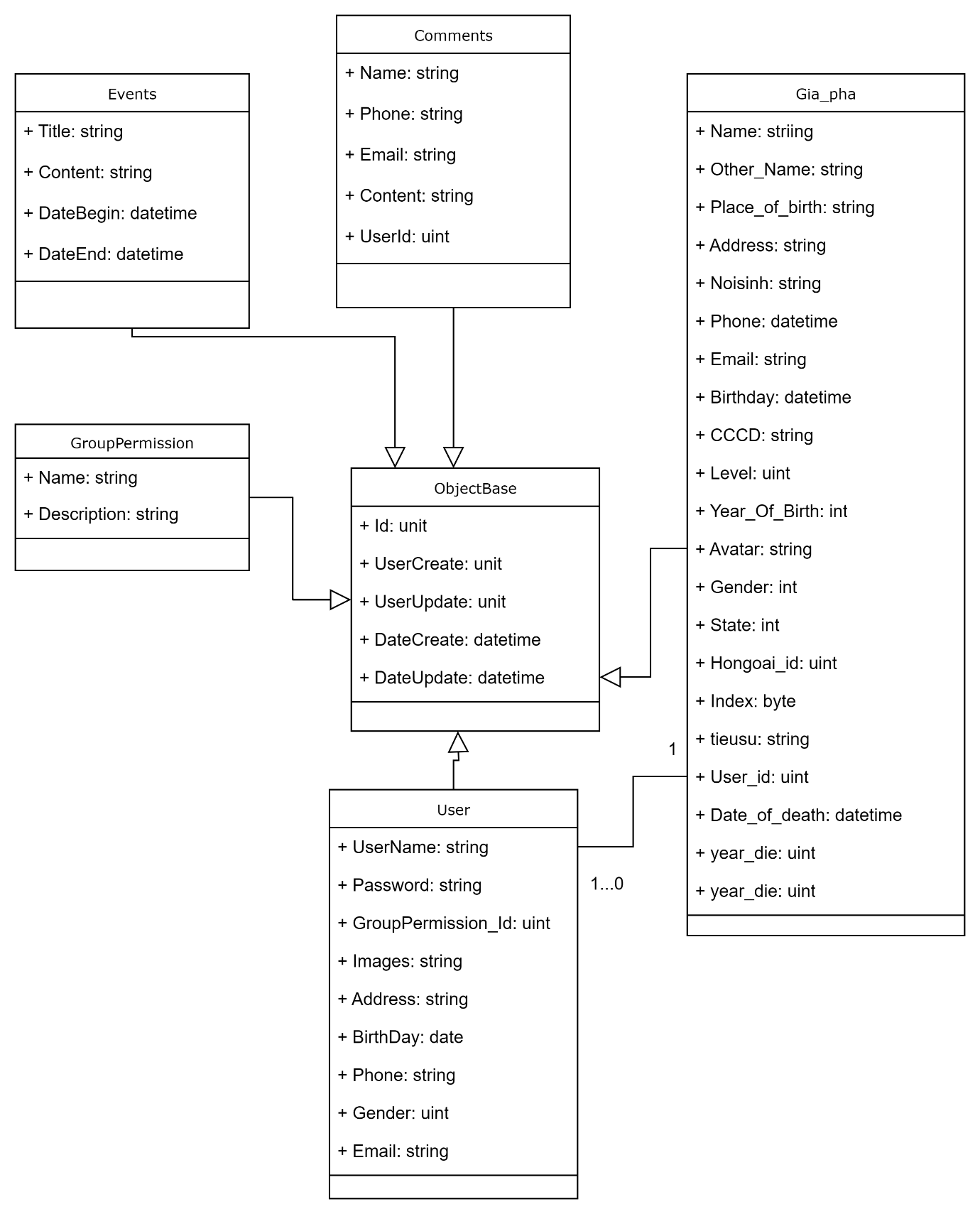
Hình 50: Biểu đồ tuần tự use case tìm kiếm người trong gia phả

#### Biểu đồ lớp



Hình : Biểu đồ tuần tự use case tìm kiếm người trong gia phả

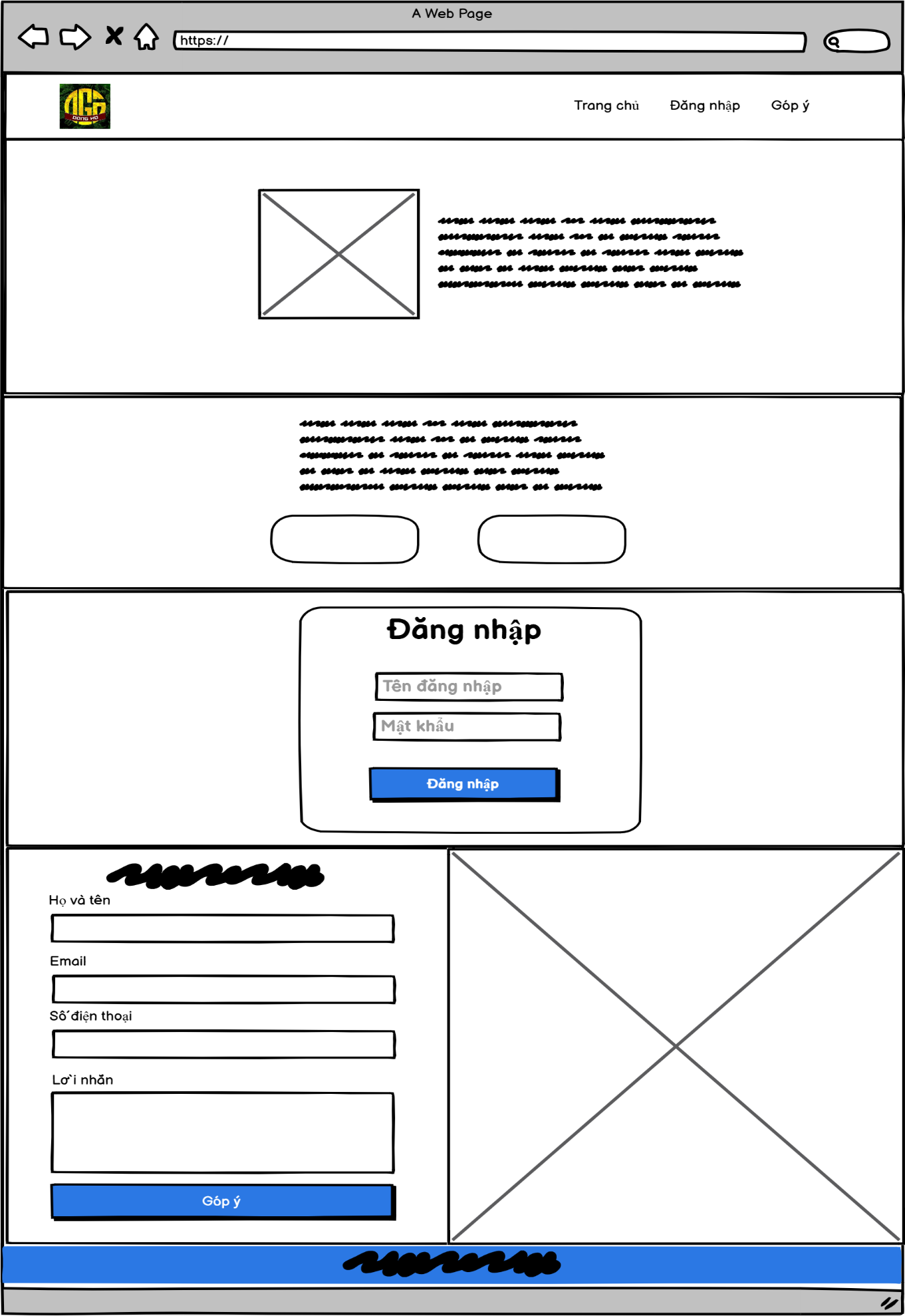
## Biểu đồ lớp tổng quát



Hình : Biểu đồ lớp tổng quát

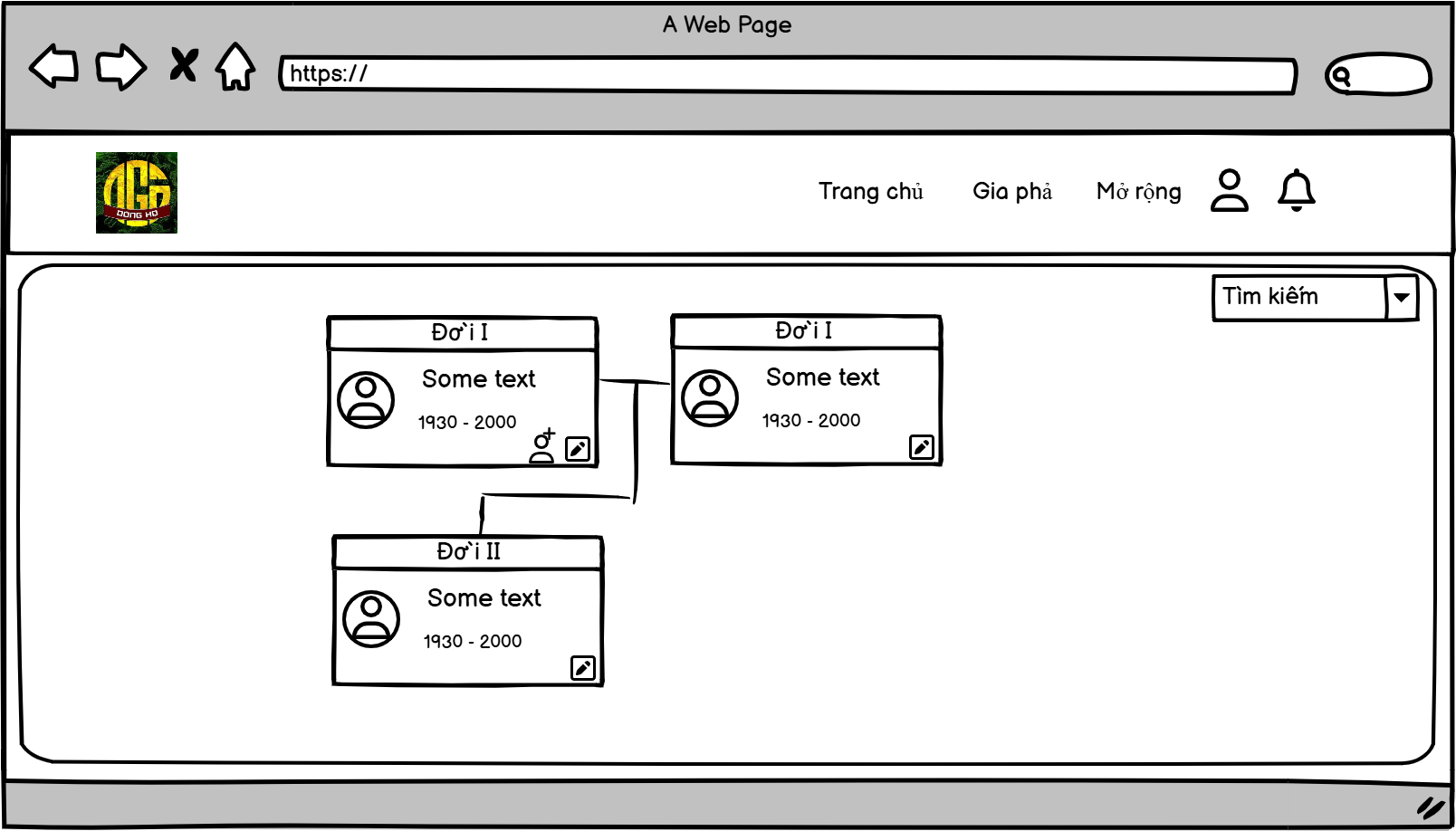
## Xây dựng giao diện

### Giao diện trang chủ



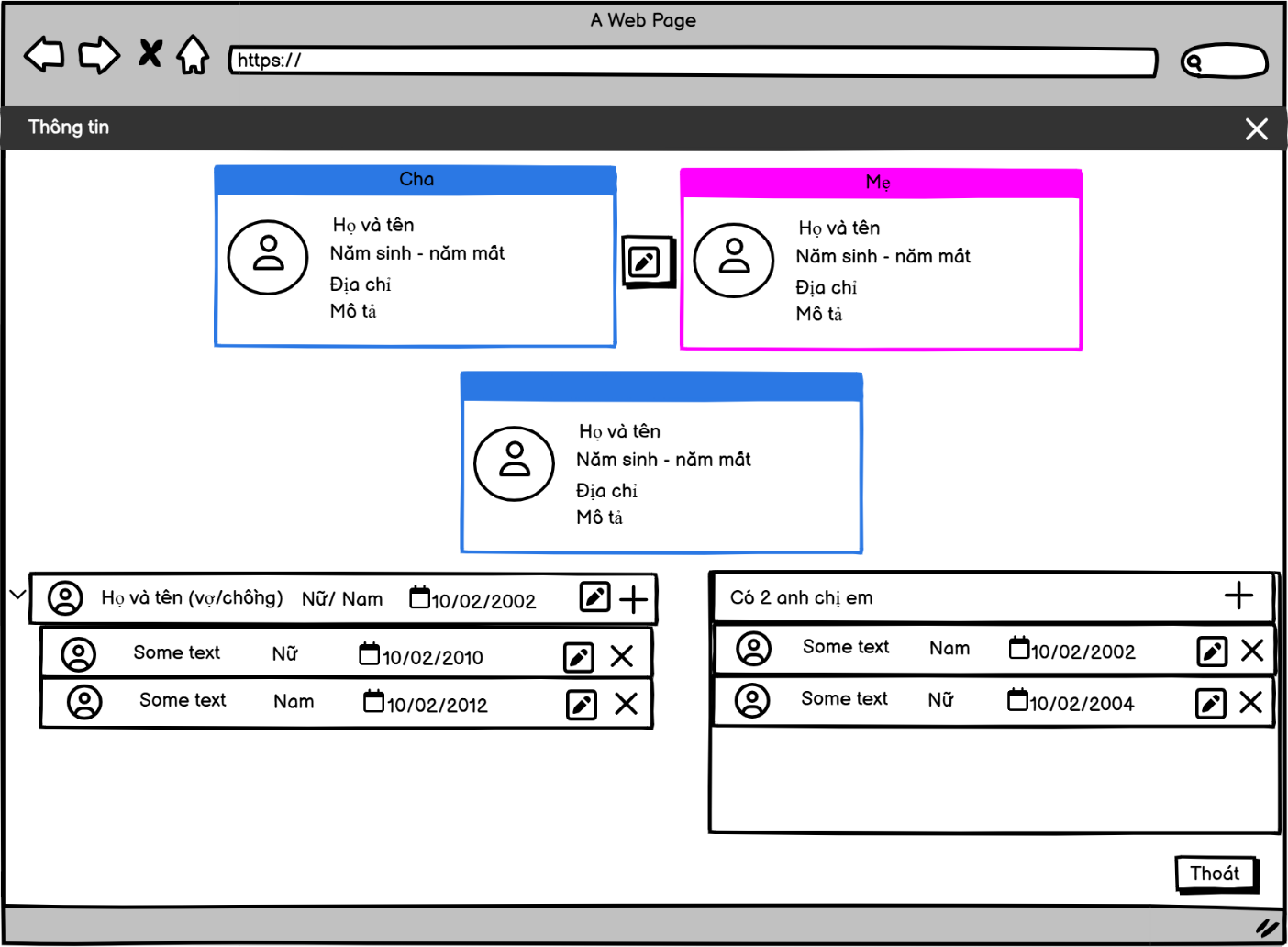
Hình : Giao diện trang chủ

### Giao diện cây gia phả



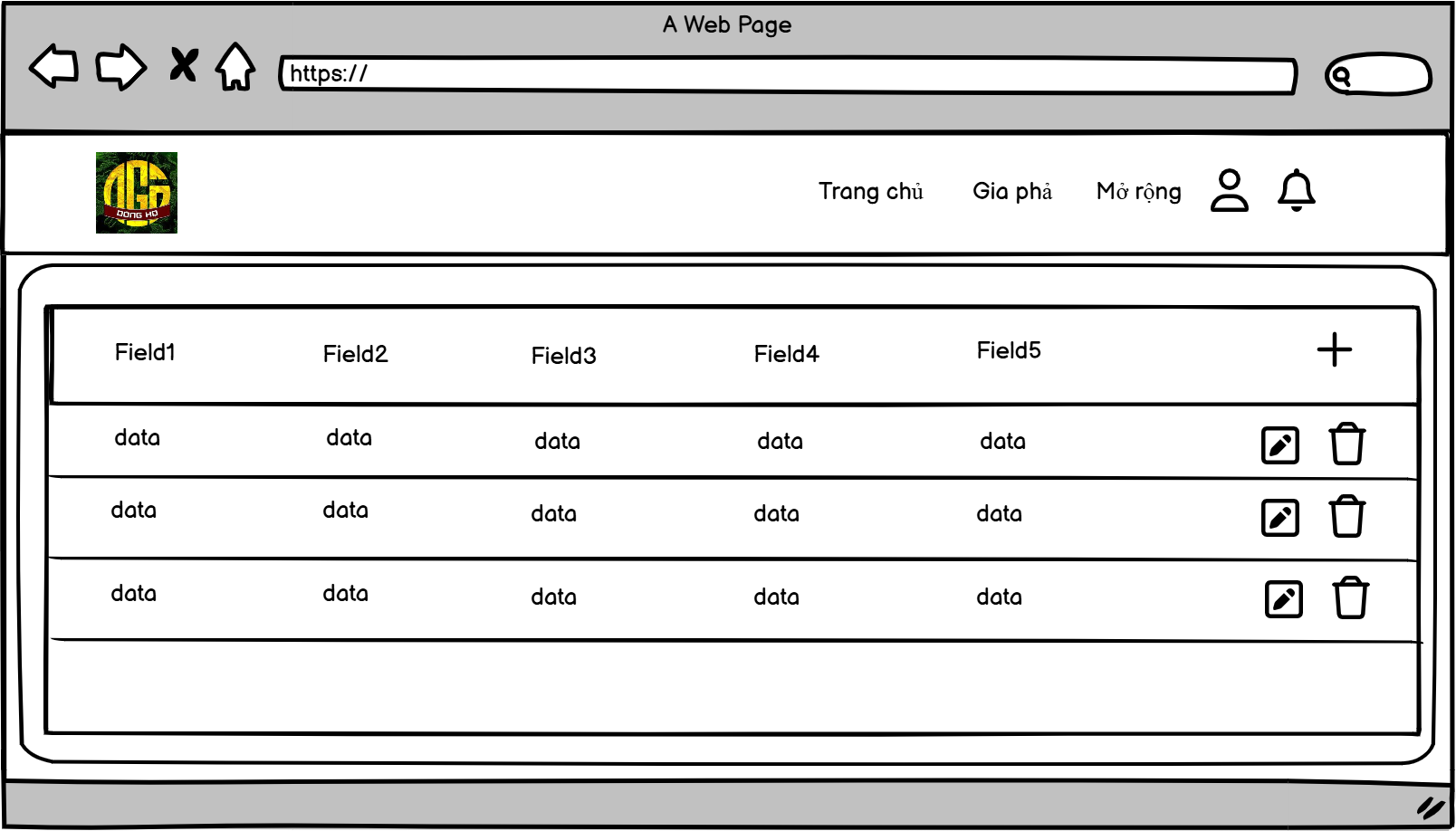
Hình : Giao diện trang gia phả

### Giao diện xem chi tiết



Hình : Giao diện xem chi tiết

### Giao diện các trang quản lý



Hình : Giao diện các trang quản lý

# CÀI ĐẶT VÀ KIỂM THỬ

## Cài đặt hệ thống

Chương trình được chia thành 2 phần Front-end và Back-end:

Front-end sử dụng ngôn ngũ vuejs và Nuxtjs framework

Back-end sẽ phụ trách phần API viết bằng C#, .Net frameword

### Cài đặt phía giao diện (Front-end)

#### Giới thiệu về Front-end

Nuxt.js là một framework mạnh mẽ được xây dựng dựa trên Vue.js, mang đến một cách tiếp cận hiện đại và hiệu quả để phát triển các ứng dụng web. Điểm đặc biệt của Nuxt.js là khả năng hỗ trợ **Server-Side Rendering (SSR)**, giúp các ứng dụng tải nhanh hơn, tối ưu hóa SEO và cải thiện trải nghiệm người dùng. Ngoài ra, Nuxt.js còn hỗ trợ **Static Site Generation (SSG)**, cho phép bạn tạo các trang tĩnh hiệu suất cao, rất phù hợp với các trang web cần tốc độ tải nhanh và tài nguyên nhẹ.

Một trong những tính năng nổi bật của Nuxt.js là kiến trúc module hóa, cho phép bạn dễ dàng tích hợp các công cụ và thư viện phổ biến như Axios, Tailwind CSS, PWA, và nhiều module khác mà không cần cấu hình thủ công. Điều này giúp giảm bớt thời gian phát triển và tập trung vào việc xây dựng chức năng chính của ứng dụng.

Ngoài ra, Nuxt.js còn cung cấp hệ thống routing tự động dựa trên cấu trúc thư mục. Chỉ cần tổ chức các file trong thư mục pages, Nuxt.js sẽ tự động tạo ra các routes tương ứng, giúp bạn tiết kiệm công sức và giữ cho mã nguồn gọn gàng, dễ quản lý. Hệ thống quản lý SEO mạnh mẽ của Nuxt.js cũng giúp bạn dễ dàng thêm và tùy chỉnh các thẻ meta, đảm bảo ứng dụng của bạn thân thiện với các công cụ tìm kiếm.

Cuối cùng, Nuxt.js hỗ trợ cả **JavaScript** và **TypeScript**, mang lại sự linh hoạt cho các nhà phát triển. Dù bạn đang xây dựng một ứng dụng nhỏ hay một hệ thống phức tạp, Nuxt.js cung cấp đầy đủ công cụ và tính năng để bạn tạo ra một ứng dụng mạnh mẽ, tối ưu và hiện đại.

#### Quy trình cài đặt

* **Bước 1**: Cài đặt Node.js

Trước tiên, cần cài đặt Node.js (phiên bản >= 14). Tải về từ Node.js Official Website.

* **Bước 2**: Mở terminal trong visual studio code và chạy lệnh:

Npm install hoặc yarn

* **Bước 3**: Sau khi cài đặt xong các dependency, chạy lệnh

Npm run dev hoặc yarn dev để chạy chương trình

* Truy cập trang web <https://nuxt.com/> để xem chi tiết

### Cài đặt phía Back-end

#### Giới thiệu về Back-end

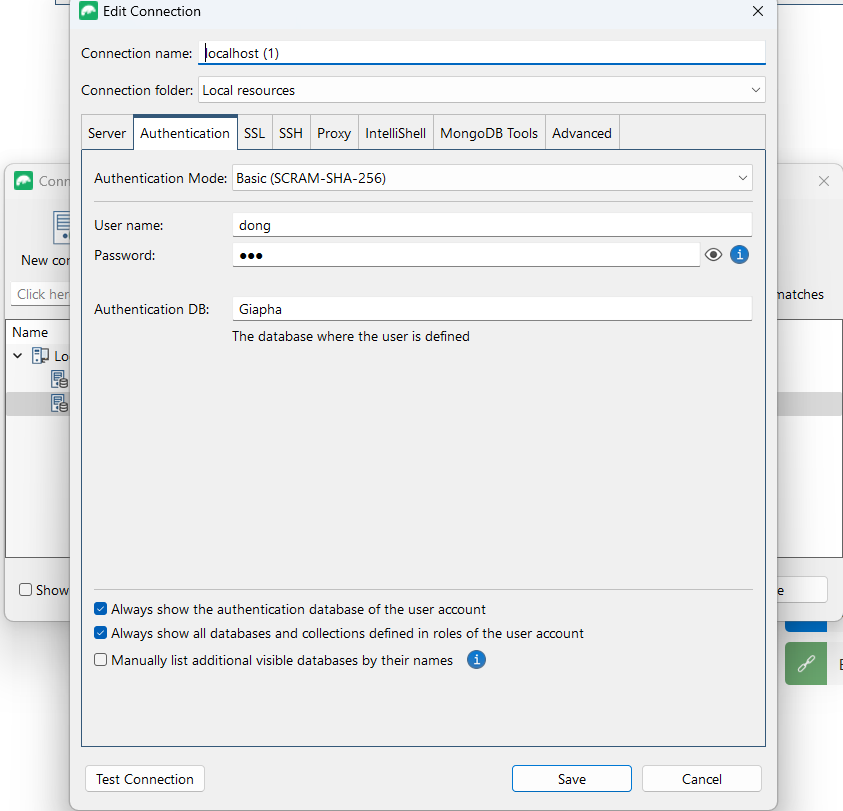
Trong dự án này, Back-end được xây dựng bằng ngôn ngữ C# trên nền tảng .NET Framework, một công cụ mạnh mẽ và đáng tin cậy để phát triển các ứng dụng web và dịch vụ. .NET Framework cung cấp một môi trường phát triển linh hoạt, hỗ trợ nhiều công cụ và thư viện sẵn có, giúp tối ưu hóa quy trình xây dựng và triển khai phần Back-end.

Với .NET Framework, việc quản lý dữ liệu trở nên hiệu quả nhờ vào tích hợp với các hệ thống cơ sở dữ liệu phổ biến. Đồng thời, tính năng xác thực người dùng và quản lý bảo mật được hỗ trợ mạnh mẽ, giúp đảm bảo an toàn cho các ứng dụng. Ngoài ra, .NET Framework còn hỗ trợ các tính năng xử lý nghiệp vụ phức tạp, từ việc tính toán dữ liệu lớn đến việc triển khai các quy trình kinh doanh tự động.

#### Quy trình cài đặt

**Bước 1:** Đảm bảo máy đã cài đặt mongodb và có phần mềm quản trị của mongodb như compass, studio3T,…

**Bước 2:** Tạo mới connection trong cơ sở dữ liệu (mật khẩu là 123)



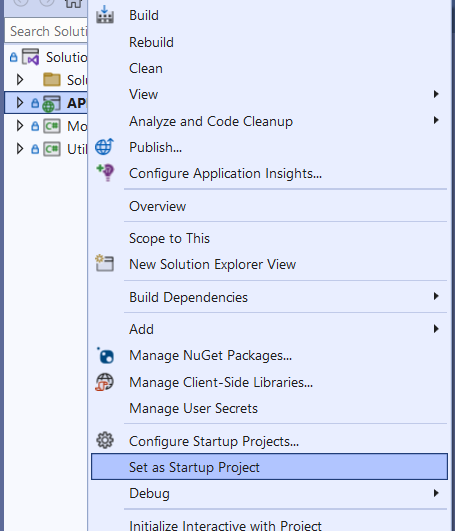
Hình : Quy trình cài đặt

**Bước 3:** Trong file C:\Users\gnodo\OneDrive\Desktop\đồ án\Quan-ly-gia-pha\Giapha\_API\API\Global.asax điền chuỗi kết nối với cơ sở dữ liệu



Hình : Quy trình cài đặt chuỗi kết nối

Bước 4: Kích chuột phải vào dự án API và kích chọn “Set as Starup Project”. Sau đó chạy chương trình



Hình : Chọn Starup Project

## Kiểm thử hệ thống

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **STT** | **Case** | **Đầu vào** | **Đầu ra mong muốn** | **Kết quả** |
| 1 | Đăng nhập | Nhập tài khoản và mật khảu | Đăng nhập thành công | Đạt |
| 2 |  | Nhập sai tài khoản hoặc mật khẩu | Thông báo lỗi “Tên đăng nhập hoặc mật khẩu không đúng” | Đạt |
| 3 | Góp ý | Nhập thông tin góp ý | Thông báo cảm ơn lời góp ý | Đạt |
| 4 |  | Nhập không đúng định dạng cái trường (email, số điện thoại,…) | Hiển thị lỗi ở dưới ô nhập thông tin | Đạt |
| 5 | Tìm kiếm thành viên trong gia phả | Chọn 1 người trong danh sách tìm kiếm | Giao diên di chyển đến vị trí người được tìm và làm nổi bật vị trí đó | Đạt |
|  |  | Bỏ chọn (Không tìm kiếm) | Xóa vị trí đã làm nổi bật của người đã được tìm từ trước | Đạt |
| 6 | Xem chi tiết mối quan hệ | Kích vào tên người trong gia phả | Hiển thị giao diện chi tiết mối quan hệ | Đạt |
| 7 | Thêm cha mẹ | Nhập thông tin của cha và mẹ (Tên, tuổi, căn cước, số điện thoại,…) | Thêm thông tin thành công và hiển thị thông tin cha mẹ | Đạt |
| 8 | Sửa thông tin cha mẹ | Sửa thông tin của cha và mẹ (Tên, tuổi, căn cước, số điện thoại,…) | Sửa thành công và hiển thị lại thông tin cha mẹ | Đạt |
| 9 | Thêm vợ/chồng | Nhập thông tin của vợ hoặc chồng (Tên, tuổi, căn cước, số điện thoại,…) | Thêm thành công và hiển thị thông tin vợ/chồng | Đạt |
| 10 |  | Dữ liệu nhập cần thiết bị thiếu | Thông báo điền hết thông tin cần thiết | Đạt |
| 11 | Sửa vợ/chồng | Thay đổi dữ liệu (Tên, tuổi, căn cước, số điện thoại,…) | Thông báo thành công và hiển thị lại thông tin vợ /chồng | Đạt |
| 12 | Thêm con | Nhập thông tin của vợ hoặc chồng (Tên, tuổi, căn cước, số điện thoại, giới tính…) | Thông báo thành công và hiển thị lại thông tin con | Đạt |
| 13 | Xóa con | Kích vào biểu tượng xóa | Thông tin người con đó đã xóa | Đạt |
| 14 | Thêm anh/chị/em | Nhập thông tin của anh/chị/em (Tên, tuổi, căn cước, số điện thoại,giới tính,…) | Thông bảo thêm thành công và hiển thị lại thông tin | Đạt |
| 15 | Xóa anh/chị/em | Kích vào biểu tượng xóa | Thông tin người con đó đã xóa | Đạt |
| 16 | Cấp tài khoản | Nhập tên đăng nhập và mật khảu | Thông báo thêm thành công | Đạt |
| 17 |  | tên tài khoản hoặc nhập khẩu bỏ trống | Thông báo yêu cầu nhập | Đạt |
| 18 | Thêm tiểu sử | Thông tin tiểu sử của người thêm | Thông báo thêm thành công | Đạt |

Bảng : Kiểm thử hệ thống

# KẾT LUẬN

Qua quá trình thực hiện đề tài **“Xây dựng website quản lý gia phả cho dòng họ Ngô”**, em đã hoàn thành việc xây dựng một website với các chức năng đơn giản, dễ sử dụng, hỗ trợ dòng họ trong việc quản lý thông tin gia phả. Đồng thời, em đã có cơ hội tìm hiểu, học hỏi thêm nhiều kinh nghiệm cũng như kiến thức, vận dụng những kỹ năng sẵn có để phát triển một ứng dụng trên nền tảng web

❖ Những kết quả đạt được trong quá trình thực hiện đồ án:

* Về công nghệ:

● Biết cách sử dụng các công cụ hỗ trợ lập trình, hệ quản trị cơ sở dữ liệu, và công cụ quản lý như Git, Visual Studio, Nuxt.js, và MongoDB Atlas.

● Hiểu rõ hơn về ngôn ngữ lập trình JavaScript,, framework Nuxt.js (Vue.js), C# .NET, và hệ quản trị cơ sở dữ liệu NoSQL MongoDB.

● Biết cách phân tích và thiết kế hệ thống UML, sử dụng các công cụ như draw.io và dbdiagram.io để trực quan hóa cấu trúc dữ liệu và luồng xử lý.

* Về chương trình:

● Xây dựng website gọn nhẹ, dễ dàng truy cập và sử dụng trên các thiết bị khác nhau

● Ứng dụng có tốc độ phản hồi nhanh, mang lại trải nghiệm mượt mà cho người dùng..

● Hỗ trợ người dùng quản lý và tra cứu thông tin gia phả một cách tiện lợi.

❖ Những hạn chế còn tồn đọng:

* Giao diện chưa được tối ưu hoàn toàn cho nhiều thiết bị với kích thước màn hình khác nhau.
* Chưa kiểm thử đầy đủ tất cả các trường hợp có thể xảy ra trong thực tế.
* Thiết kế cơ sở dữ liệu còn đơn giản, chưa đáp ứng được những trường hợp phức tạp hơn.

❖ Hướng phát triển

* Phát triển website tối ưu hơn với các kích thước màn hình khác nhau
* Bổ sung thông báo sự kiện sắp diễn ra sắp tới
* Cho phép gia phả ở bên người vợ (cưới người con trai trong gia phả)

# TÀI LIỆU HƯỚNG DẪN

1. Phùng Đức Hòa (Chủ biên); Hoàng Quang Huy; Hoàng Văn Hoành; Nguyễn Đức Lưu; Trịnh Bá Quý. "Giáo trình Nhập môn công nghệ phần mềm". NXB Thống kê, 2019.
2. Nguyễn Thị Thanh Huyền; Ngô Thị Bích Thúy; Phạm Kim Phượng. "Giáo trình Phân tích thiết kế hệ thống". NXB Giáo dục Việt Nam, 2011.