TRƯỜNG ĐẠI HỌC BÀ RỊA - VŨNG TÀU
KHOA KỸ THUẬT - CÔNG NGHỆ

BARIA VUNGTAU
UNIVERSITY
CAP SAINT JACQUES

BÁO CÁO ĐỒ ÁN TỐT NGHIỆP

ĐỀ TÀI:

THIẾT KẾ VÀ PHÁT TRIỂN WEBSITE MẠNG XÃ HỘI

Trình độ đào tạo	: Đại học
Ngành	: Công nghệ thông tin
Chuyên ngành	: Công nghệ thông tin
Khóa học	: 2019-2023
Lớp	: DH19CT
Sinh viên thực hiện	: Nguyễn Minh Luân
Mã số sinh viên	: 19034008
GVHD	: TS. Phan Ngọc Hoàng

VŨNG TÀU, NĂM 2023

LỜI CẢM ƠN

Để hoàn thành đề tài báo cáo đồ án này trước hết, em xin gửi đến quý thầy, cô giáo trong khoa Kỹ thuật – Công Nghệ trường Đại học Bà Rịa – Vũng Tàu lời cảm ơn chân thành.

Đặc biệt, em xin gửi đến thầy Phan Ngọc Hoàng, người đã tận tình hướng dẫn, giúp đỡ em hoàn thành đề tài đồ án tốt nghiệp lời cảm ơn sâu sắc nhất.

Em xin chân thành cảm ơn Ban Lãnh Đạo, các phòng ban của công ty TNHH Niềm Tin, đã tạo điều kiện thuận lợi cho em được tìm hiểu thực tiễn trong suốt quá trình thực tập tại công ty.

Đồng thời xin chân thành cảm ơn Ban lãnh đạo trường Đại học Bà Rịa – Vũng Tàu đã cho môi trường học tập lành mạnh và năng động. Kính chúc Trường Đại học Bà Rịa – Vũng Tàu tiếp túc gặt hái nhiều thành công trên con đường giáo dục thế hệ trẻ.

Vì kiến thức bản thân còn hạn chế và trình độ lý luận cũng như kinh nghiệm thực tiễn còn hạn chế nên bài báo cáo không thể tránh khỏi những thiếu sót, em rất mong nhận được ý kiến đóng góp thầy, cô để em học thêm được nhiều kinh nghiệm và sẽ hoàn thiện bản thân đáp ứng với nhu cầu công việc thực tế trong tương lai.

Em xin chân thành cảm ơn!

LỜI CAM ĐOAN

Tôi xin cam đoan kết quả đạt được trong đồ án là sản phẩm của riêng cá nhân, không sao chép lại của người khác. Trong toàn bộ nội dung của đồ án, những điều được trình bày hoặc là của cá nhân hoặc là được tổng hợp từ nhiều nguồn tài liệu. Tất cả các tài liệu tham khảo đều có xuất xứ rõ ràng và được trích dẫn hợp pháp.

Tôi xin hoàn toàn chịu trách nhiệm và chịu mọi hình thức kỷ luật theo quy định cho lời cam đoan của mình.

Vũng Tàu, ngày 01 tháng 05 năm 2023

Sinh viên thực hiện

Nguyễn Minh Luân

NHẬN XÉT CỦA GIẢNG VIÊN HƯỚNG DẪN

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

Vũng Tàu, ngày … tháng … năm 20…

Người hướng dẫn

TS. Phan Ngọc Hoàng

NHẬN XÉT CỦA GIẢNG VIÊN PHẢN BIỆN

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

MỤC LỤC

DANH MỤC BẢNG

DANH MỤC SƠ ĐỒ

DANH MỤC HÌNH

LỜI NÓI ĐẦU

Chúng ta đang sống trong thời đại còng nghệ 4.0, khoa học và công nghệ ngày càng phát triển như vũ bão, cụ thể là sự phát triển của công nghệ thông tin. Internet kết nối mọi người ờ khắp nơi trên thế giới, vượt qua mọi khoảng cách về địa lý, không gian và thời gian, tạo điều kiện để mọi người giao lưu, gặp gỡ. Chỉ cần có một chiếc điện thoại hay máy tính kết nối Internet mọi người đều có thể tìm kiếm thông tin, giải trí, trao đổi thông tin, trò chuyện, ... một cách dễ dàng và nhanh chóng. Hiện nay, từ người lớn cho đến trẻ nhỏ không ai là không biết đến Internet, hầu hết mọi người ai cũng đang sử dụng Internet. Cùng với sự phát triển của Internet, các mạng xã hội cũng đã trở nên rất quen thuộc với tất cả mọi người và gần như nó là một phần không thể thiếu đối với tất cả mọi người trong xã hội hiện đại ngày nay, đặc biệt là giới trẻ.

Mạng xã hội được hiểu là một trang web hay một nền tảng trực tuyến giúp kết nối dễ dàng tất cà mọi người từ bất cứ nơi đâu, với nhiều dạng hình thức và tính năng khác nhau. Mạng xã hội hay còn được gọi là mạng xã hội ảo (Social Network). Mạng xã hội trở thành công cụ kết nối phổ biến được rất nhiều người sử dụng, thế nhưng các trang mạng xã hội lớn hiện nay đa số là dành cho tất cả người dùng trên toàn thế giới với nhiều loại hình văn hóa khác nhau, nguy cơ làm cho giới trẻ có suy nghĩ lệch lạc do tiếp cận với các văn hóa tiêu cực và làm mất đi bản sắc văn hóa của người Việt Nam.

Nhận thấy được tiềm năng phát triển của mạng xã hội ở Việt Nam, đồng thời với mong muốn “làm sạch” nguồn thông tin trên mạng xã hội, tránh du nhập các văn hóa tiêu cực vào nước ta, đó là lí do đề tài “Xây dựng website Mạng xã hội” được ra đời. Hướng tới những người dùng Việt vừa muốn có những trải nghiệm ảo trên mạng xã hội vừa tiếp thu những thông tin “sạch”, chính thống, có nguồn gốc. Đặc biệt, là nhấn mạnh vào tiêu chí “Người Việt dùng hàng Việt”. Hi vọng đem đến cho người dùng những trải nghiệm tốt nhất trên mạng xã hội.

CHƯƠNG 1 TỔNG QUAN

1.1 Lý do chọn đề tài

Mạng xã hội với cách gọi đầy đủ là "dịch vụ mạng xã hội" hay "trang mạng xã hội", là nền tảng trực tuyến nơi mọi người dùng để xây dựng các mối quan hệ với người khác có chung tính cách, nghề nghiệp, công việc, trình độ, … hay có mối quan hệ ngoài đời thực.

Mạng xã hội có nhiều dạng thức và tính năng khác nhau, có thể được trang bị thêm nhiều công cụ mới, và có thể vận hành trên tất cả các nền tảng như máy tính để bàn, máy tính xách tay, máy tính bảng hay điện thoại thông minh.

Mạng xã hội cho phép người dùng chia sẻ câu chuyện, bài viết, ý tưởng cá nhân, đăng ảnh, video, đồng thời thông báo về hoạt động, sự kiện trên mạng hoặc trong thế giới thực. Nếu như trong mô hình mạng xã hội truyền thống, ví dụ như sự kiện hội chợ, đã tồn tại từ lâu trong lịch sử thì mạng xã hội trên web giúp người dùng kết nối với những người sống ở nhiều vùng đất khác nhau, ở thành phố khác hoặc trên toàn thế giới.

Trong thời đại công nghệ 4.0, mạng xã hội đã và đang phát triển mạnh mẽ hơn nhờ vào sự hổ trợ của các công cụ trí tuệ nhân tạo. Song, trên đà phát triển mạnh mẽ của các mạng xã hội phổ biến, mang tính toàn cầu thì trên thị trường lại không xuất hiện các mạng xã hội dành riêng cho người Việt. Đề tài được phát triển với mong muốn tạo ra một mạng xã hội dành cho người Việt, to lớn hơn là để bảo mật tất cả các thông tin trong nước.

1.2 Mục tiêu và chức năng của hệ thống

1.2.1 Mục tiêu

Tạo ra một hệ thống cho phép người dùng có thể kết nối, giao lưu, chia sẻ những thông tin hữu ích trên nền tảng Internet.

Ngoài ra, hệ thống còn có mục tiêu là tạo nên một cộng đồng có giá trị, nâng cao vai trò của mỗi người dùng trong việc xây dựng các mối quan hệ.

1.2.2 Chức năng

- ✓ Đăng ký, đăng nhập

- ✓ Quản lý thông tin người dùng
- ✓ Quản lý bài đăng
- ✓ Quản lý nhóm
- ✓ Hiển thị bài đăng dưới dạng Timeline
- ✓ Hiển thị chi tiết bài đăng
- ✓ Tương tác với bài đăng
- ✓ Theo dõi qua lại giữa các người dùng
- ✓ Nhắn tin giữa các người dùng
- ✓ Thông báo

1.3 Mô tả bài toán

- ➢ Mục đích của website mạng xã hội là giao diện để người dùng có thể tương tác với các bài đăng, các nhóm, các bình luận, … Chi tiết các chức năng chính của hệ thống như sau:
  - ✓ Đăng ký, đăng nhập:
    - o Đăng ký: Người dùng đăng ký tài khoản với tên đăng nhập là địa chỉ email.
    - o Đăng nhập: Người dùng đăng nhập vào hệ thống nếu đã có tài khoản.
  - ✓ Hiển thị các bài đăng dưới dạng Timeline:
    - o Mỗi bài đăng trên hệ thống được thiết kế theo dạng thẻ, hiển thị các thông tin như: Người đăng, thời gian đăng tải, trạng thái, hình ảnh, lượt thích, ...
  - ✓ Hiển thị chi tiết bài đăng:
    - o Được điều hướng từ mỗi bài viết tương ứng hiển thị ở Timeline.
  - ✓ Tương tác với bài đăng:
    - o Thích: Người dùng có thể để lại một lượt thích cho mỗi bài viết hiển thị ở Timeline.
    - o Bình luận: Người dùng có thể để lại bình luận cho mỗi bài viết hiển thị ở Timeline.
    - o Xóa bài: người dùng có thể xóa các bài đăng của mình.
  - ✓ Tương tác với nhóm:

- Người dùng có thể tham gia vào bất kì nhóm nào, đăng các bài viết vào nhóm mình tham gia.
- Admin nhóm có thể duyệt thành viên, duyệt bài đăng mới, cập nhật thông tin cho nhóm, mời người dùng khác ra khỏi nhóm của mình.

✓ Theo dõi qua lại giữa các người dùng:

- Theo dõi các người dùng khác để có thể thấy được các bài viết của họ ở Timeline.
- Những người dùng theo dõi nhau mới có thể nhắn tin cho nhau.

✓ Nhắn tin giữa các người dùng:

- Người dùng có thể giao tiếp với nhau qua chức năng nhắn tin.

✓ Thông báo:

- Người dùng sẽ nhận được thông báo từ hệ thống khi một ai khác tương tác với mình (Tương tác với bài đăng hoặc nhắn tin).

CHƯƠNG 2 CƠ SỞ LÝ THUYẾT

2.1 Giới thiệu về ASP.NET

2.1.1 Tổng quát

Hình 2.1 Biểu tượng của ASP.NET

ASP.NET là một mã nguồn mở dành cho web được tạo bởi Microsoft. Hiện mã nguồn này chạy trên nền tảng Windows và được bắt đầu vào đầu những năm 2000.

ASP.NET cho phép các nhà phát triển tạo các ứng dụng web, dịch vụ web và các trang web động.

Phiên bản ASP.NET đầu tiên được triển khai là 1.0 được ra mắt vào tháng 1 năm 2002 và hiện nay, phiên bản ASP.NET mới nhất là 4.6. ASP.NET được phát triển để tương thích với giao thức HTTP. Đó là giao thức chuẩn được sử dụng trên tất cả các ứng dụng web.

ASP.NET được biên dịch dưới dạng Common Language Runtime (CLR), có khả năng hỗ trợ các lập trình viên viết mã ASP.NET với bất kỳ ngôn ngữ nào được hỗ trợ bởi .NET language, ví dụ: C#, VB.Net và J#, …

2.1.2 Lịch sử phát triển của ASP.NET

Sau khi phiên bản Internet Information Service 4.0 được phát hành vào năm 1997, Microsoft tạo mô hình nghiên cứu ứng dứng web để giải quyết những khó khăn của ASP.NET, mục đích chính là tách biệt phần thể hiện và phần nội dung cũng như cách viết rõ ràng hơn.

Người được giao nhiệm vụ định hình mô hình ASP lúc đó chính là Mark Anders, quản lý của nhóm IIS và Scott Guthrie.

Những thiết kế ban đầu về ASP.NET được Anders và Guthrie thực hiện chỉ vỏn vẹn trong khoảng 2 tháng. Những mã prototype đầu tiên được Guthrie viết trong khoảng thời gian nghỉ lễ Giáng sinh năm 1997.

Hình 2.2 Lịch sử phát triển của ASP.NET

2.1.3 Một số đặc điểm của ASP.NET

- **Trạng thái code rời/Code Behind Mode**

ASP.NET tách rời những mã code và những bản thiết kế để đảm bảo khả năng duy trì tính ổn định của ứng dụng. Các file trong ASP.NET được tách rời thành nhiều tệp, giúp người phát triển có khả năng giữ cho ASPX.NET dễ dàng, đơn giản.

- **Khả năng quản lý các trạng thái/State Management**

ASP.NET có các công cụ để quản trị, quản lý trạng thái, trong khi HTTP được biết tới là một giao thức "không trạng thái". Xem xét qua ví dụ về ứng dụng giỏ hàng: khi một người dùng chọn lựa xong những thứ họ muốn mua, tiếp theo họ muố đưa ra quyết định mua hàng trên trang web, lúc này người dùng sẽ nhấn nút gửi.

Ứng dụng lúc này cần ghi nhớ các mục mà người dùng đã mua. Đó chính là hành động nhớ trạng thái của một ứng dụng tại thời điểm mua hàng. Vì HTTP được coi là giao thức không trạng thái nên khi người dùng truy cập các trang web bán hàng, HTTP sẽ không được phép lưu trữ thông tin trên các giỏ hàng.

- **Bộ nhớ Cache**

Ngôn ngữ ASP.NET có nhiệm vụ lưu trữ bộ nhớ Cache giúp cải thiện hiệu năng làm việc của những ứng dụng. Công việc này giúp hỗ trợ lưu thông tin, hỗ trợ truy xuất, phản hồi trong một khoảng thời gian ngắn, từ đó tăng hiệu suất làm việc của ứng dụng.

2.1.4 Cấu trúc và những thành phần của ASP.NET

Dưới đây là những phân tích về cấu trúc thành phần của ASP.NET là gì, nó bao gồm những yếu tố: Ngôn ngữ, thư viện và thời gian chạy CLR:

- **Language/Ngôn ngữ**

Là tập con của .NET Framework, tại đây có rất nhiều ngôn ngữ lập trình khác nhau phải kể đến như C#, VB.net, PHP, JavaScript, … Trong đó C# và VB.net được sử dụng phổ biến nhất trong ứng dụng phát triển ứng dụng web.

- **Library/Thư viện**

ASP. NET Framework gồm có một bộ các lớp library chuẩn. Web library là thư viện được dùng phổ biến nhất cho các ứng dụng web. Web library bao gồm tất cả các thành phần cần thiết sử dụng trong phát triển các ứng dụng web-based.

- **Common Language Runtime/Thời gian chạy CLR**

CLR là một trong các cơ sở hạ tầng của phần đông các kiểu ngôn ngữ lập trình phổ thông. Trong đó, CLR thực hiện các tác vụ chính để xử lý các trường hợp đặc biệt và thu gom rác.

2.1.5 Tại sao nên chọn ASP.NET

- **Không cần máy tính cấu hình cao**

Có khá nhiều lý do chính đáng để phát triển một trang web hoặc một ứng dụng trên điện thoại. Đó là, tốc độ cao, chi phí thấp và hỗ trợ ngôn ngữ đa dạng. Đây là một trong những lợi ích đáng kể nhất của mã nguồn này.

Ngôn ngữ ASP.NET được xây dựng trong môi trường máy chủ Windows quen thuộc. Cũng vậy, ASP.NET yêu cầu cấu hình máy không cần quá mạnh giống như các phần mềm khác. Các mã nguồn mở này phải cài đặt cấu hình riêng. Nhất là, đủ mạnh để đáp ứng việc xây dựng web. Sự phổ biến của ASP.NET giúp cho việc truy xuất và tìm kiếm của các coder dễ dàng hơn.

Vì thế, các trang web và app được viết bằng mã nguồn ASP.NET có thể nhanh hơn. Và, mã nguồn này còn hiệu quả hơn về SEO so với trang web được viết bằng PHP, và các mã nguồn khác.

- **Vì sao không cần cấu hình mạnh?**

Bởi vì, các ứng dụng ASP.NET được chuyển đổi thành các mã nguồn thực thi. Quá trình chuyển mã này mất một khoảng thời gian nhỏ, nhưng chỉ diễn ra một lần là xong.

Sau khi biên dịch, code này có thể được thực thi lặp đi lặp lại trên nền tảng .Net một cách nhanh chóng. Cũng vậy, mã không được thực thi trực tiếp bởi phần mềm mà phải được đọc và biên dịch mỗi lần trước khi được thực thi. Code được biên dịch thường nhanh hơn và dễ tuỳ biến hơn. Ví dụ như các ngôn ngữ lập trình PHP, JavaScript và Ruby. Quá trình biên dịch cũng cung cấp việc xác nhận tất cả các code trên trang đều được nhất quán.

Ví dụ, nếu một phương thức có tên GetUser được đổi tên thành GetEFastee như một phần của code đã được cập nhật, thì bất kỳ tham chiếu nào đến GetUser trong suốt phần còn lại của ứng dụng sẽ xuất hiện lỗi trong quá trình biên dịch, điều này giúp nhà phát triển dễ dàng xác định để sữa lỗi. Các ngôn ngữ được biên dịch sẽ không xác định được lỗi này cho đến khi code thực sự được chạy và kiểm tra.

Trong một ứng dụng lớn, rất tốn thời gian để kiểm tra thủ công mọi tình huống hoặc viết và duy trì mã bổ sung để kiểm tra mọi kịch bản, bất cứ khi nào mã được thay đổi.

- **Dùng miễn phí, chỉ trả phí thấp khi là công ty lớn**
  - ✓ **Phần mềm nào viết ASP.NET miễn phí**

Điều tuyệt vời là, chi phí mua và sử dụng phần mềm để viết ASP.net là một yếu tố quan trọng khi phát triển một trang web. Trước đây, các nhà phát triển phải mua phần mềm đắt tiền (được gọi là Môi trường phát triển tích hợp, hoặc IDE) để xây dựng ứng dụng. Ngày nay, hầu hết mã có thể được viết bằng các công cụ miễn phí. Mã ASP.NET có thể được viết bằng một trình soạn thảo văn bản đơn giản.

Cũng thế, Nhiều trình soạn thảo văn bản lập trình miễn phí khác nhau và phổ biến nhất là ứng dụng Visual Studio của Microsoft, có sẵn trong một phiên bản Cộng đồng miễn phí dành cho cá nhân và doanh nghiệp nhỏ.

  - ✓ **Khi nào dùng Visual Studio cần trả phí**

Các doanh nghiệp lớn với nhiều lập trình viên sử dụng phải mua giấy phép của Visual Studio, nhưng chi phí này khá cạnh tranh với các IDE chuyên nghiệp trên các nền tảng

khác. Microsoft cũng cung cấp Visual Studio Code, một IDE và trình soạn thảo nhẹ hơn, miễn phí cho tất cả mọi người, kể cả các công ty.

Hình 2.3 Tại sao nên chọn ASP.NET

Ngoài IDE, các trang web yêu cầu phần mềm máy chủ ứng dụng web và máy chủ lưu trữ. Hầu hết các ứng dụng ASP.NET sử dụng Microsoft IIS (Máy chủ thông tin Internet). IIS có sẵn cho tất cả các phiên bản của Microsoft Windows mà không phải trả thêm phí.

Song, Máy chủ lưu trữ Windows thường đắt tiền hơn máy chủ Linux tương đương, thường được sử dụng để chạy các ứng dụng PHP, JavaScript và Ruby.

Tuy nhiên, với việc phát hành ASP.NET Core vào năm 2016, các ứng dụng ASP.NET không còn phụ thuộc vào IIS nữa và hiện có thể được lưu trữ trên Linux và MacOS cũng như Windows.

- **Thích hợp các cơ sở dữ liệu**

Các Loại cơ sở dữ liệu nào để sử dụng là một quyết định quan trọng khi phát triển một ứng dụng web.

Các ứng dụng ASP.NET có thể sử dụng tất cả các cơ sở dữ liệu phổ biến, bao gồm Microsoft SQL Server, MySQL, MariaDB, Postgres, MongoDB và CouchDB.

Mặc dù, Microsoft SQL Server có thể yêu cầu giấy phép phần mềm đắt tiền để triển khai phức tạp, nhưng phiên bản Express miễn phí phù hợp với đại đa số các trang web. Tất cả các cơ sở dữ liệu khác được đề cập ở đây đều là mã nguồn mở và miễn phí.

- **Viết giống C#**

ASP.NET được viết bằng các ngôn ngữ lập trình hướng đối tượng như C# hoặc VB.net. Lập trình hướng đối tượng cung cấp một cách viết chung để tổ chức và sử dụng lại code.

Trong khi VB.net là sản phẩm tiếp theo từ sản phẩm Visual Basic kế thừa của Microsoft và hầu như không được các nhà phát triển ưa chuộng, C# là ngôn ngữ lập trình hạng nhất và luôn được xếp hạng trong số các ngôn ngữ lập trình được yêu cầu và sử dụng nhiều nhất trên thế giới.

- **Được cập nhật thường xuyên**

Cuối cùng, mặc dù ASP.NET là mã nguồn mở và miễn phí sử dụng. ASP.NET được phát triển, và hỗ trợ tích cực bởi công ty phần mềm lớn nhất thế giới, đó là Microsoft.

Khá hay là, Microsoft đã đầu tư rất nhiều vào các nền tảng phát triển, cộng đồng những người phát triển. Đồng thời, hỗ trợ các công ty phần mềm sử dụng để ứng dụng này.

2.1.6 Ứng dụng ASP.NET

- Phát triển ứng dụng website
- Phát triển dịch vụ website
- Phát triển các trang web động

2.2 Giới thiệu Telerik

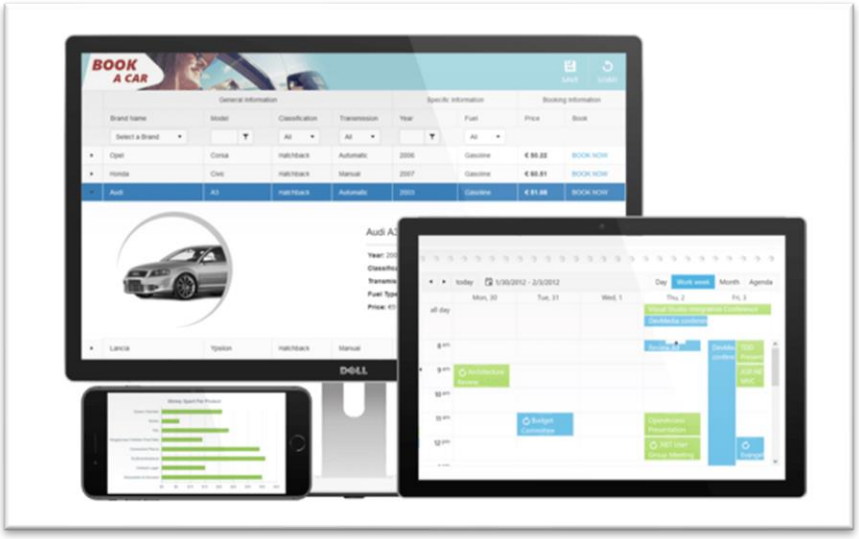
2.2.1 Telerik là gì?

Hình 2.4 Biểu tượng của Telerik

Telerik khởi đầu là một công ty cung cấp các công cụ cho cộng đồng phát triển .NET và trong những năm này, họ đã xây dựng các framework xung quanh phát triển di động cũng như phát triển JavaScript.

Khi JavaScript đang trở nên nổi bật hơn với NodeJS, Telerik by Progress dường như là một lựa chọn tốt cho các thư viện phát triển của các nhà phát triển.

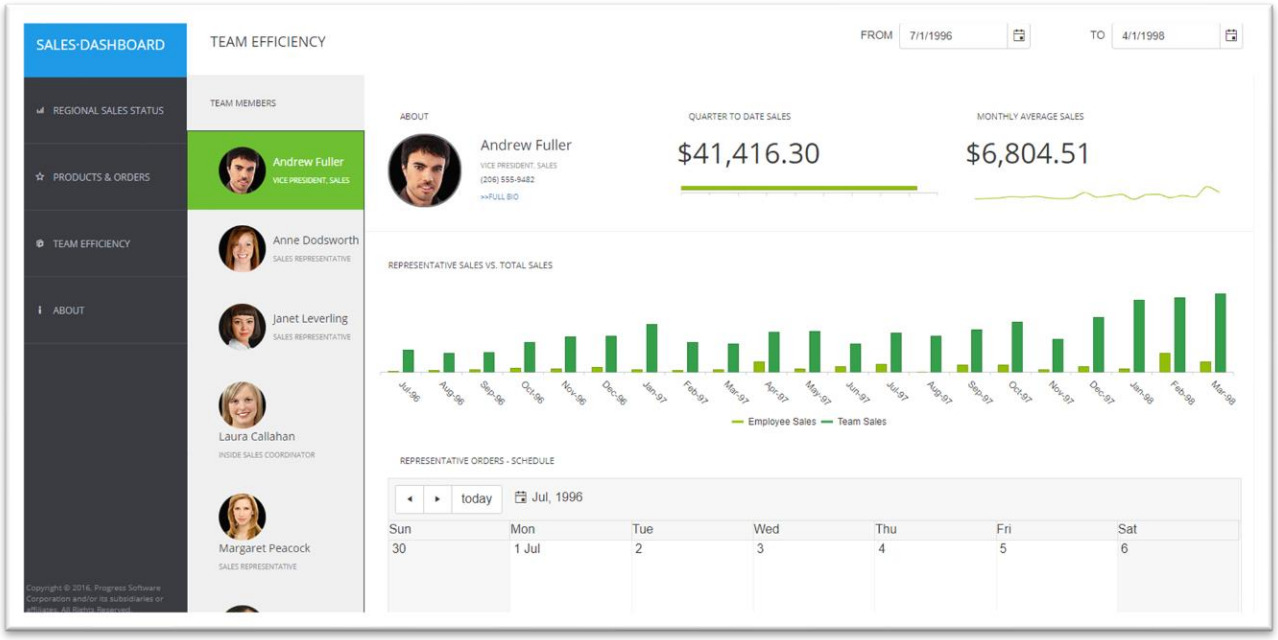
2.2.2 Tại sao cần sử dụng Telerik?



Hình 2.5 Công cụ Telerik

Với .NET, các lập trình viên có thể cung cấp các giải pháp nhanh hơn nhiều vì các biểu mẫu và bộ điều khiển web đã được xác định rất rõ ràng. Khi mọi thứ phát triển theo thời gian, nhiều tính năng cấp kiểm soát hơn bắt đầu được khách hàng yêu cầu cũng như tập trung nhiều vào giao diện và UX (trải nghiệm người dùng). Ở giai đoạn này, các bộ điều khiển thông thường chưa thể đáp ứng được. Vì vậy, Telerik được phát triển để giải quyết các vấn đề kiểm soát dữ liệu và nâng cao trải nghiệm người dùng.

2.2.3 Giới thiệu Telerik UI cho ASP.NET AJAX



Hình 2.6 Ứng dụng Telerik UI cho ASP.NET

Bộ giao diện người dùng Telerik cho ASP.NET AJAX bao gồm hơn 120 thành phần ASP.NET Web Forms được tối ưu hóa hiệu suất và linh hoạt giúp nhà phát triển xây dựng các dự án kinh doanh chuyên nghiệp, chất lượng cao. Bộ này trước đây được gọi là "RadControls cho ASP.NET AJAX", trong đó "RAD" là viết tắt của Rapid Application Development.

Từ lưới dữ liệu AJAX hàng đầu đến Trình soạn thảo HTML đầy đủ tính năng được Microsoft sử dụng trên các trang web như MSDN, Telerik UI cho ASP.NET AJAX cung

cấp tất cả các khối xây dựng để tối đa hóa năng suất và xây dựng các ứng dụng phong phú hơn, bóng bẩy hơn và hoạt động tốt hơn một cách dễ dàng.

Giao diện người dùng Telerik cho ASP.NET AJAX hoạt động trên .NET framework phiên bản 3.5, 4.0, 4.5, 4.6, 4.7 và 4.8. Các bộ điều khiển giống nhau hoạt động cho cả máy 32-bit và 64-bit.

2.3 Giới thiệu SignalR

2.3.1 SignalR là gì?

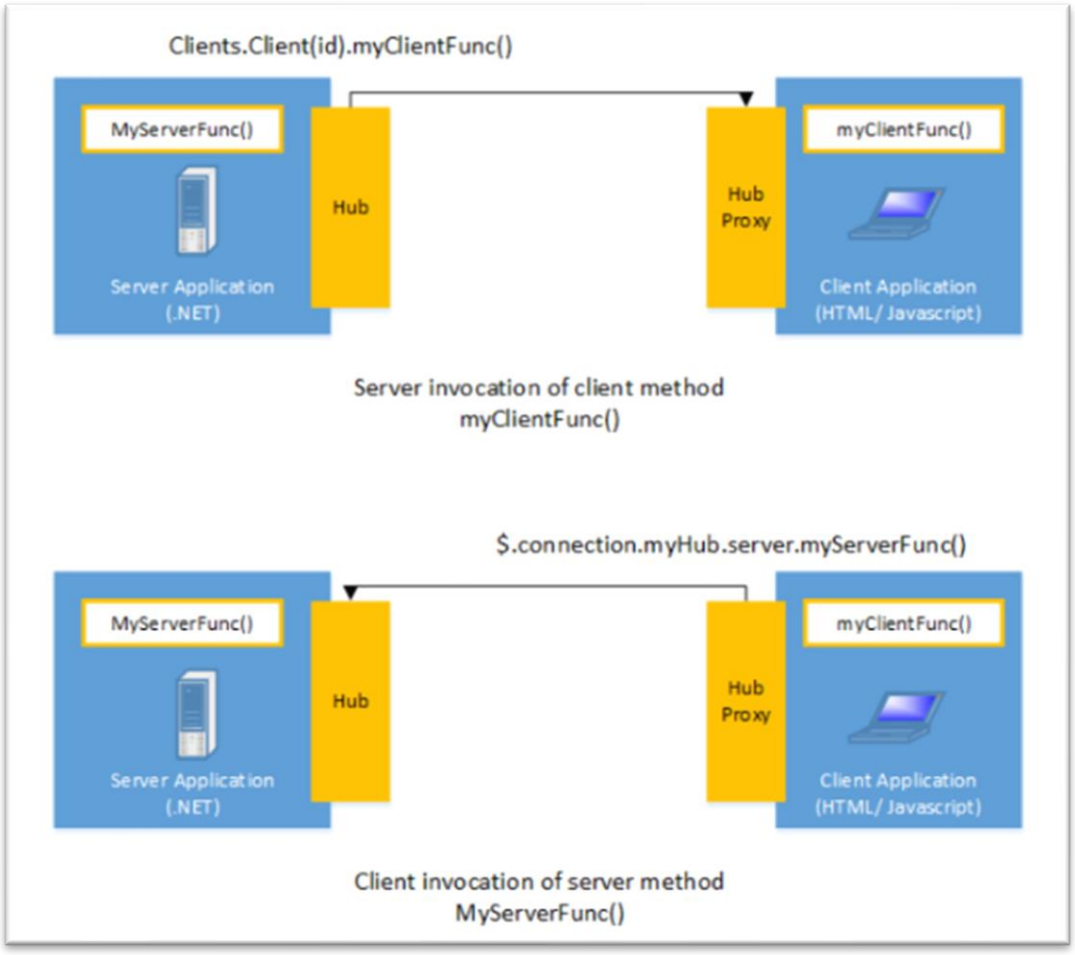
SignalR là một thư viện cho các lập trình viên Asp.Net đơn giản hóa quá trình thêm chức năng web real-time trong phát triển ứng dụng.

Real-time web functionality là gì? Đó là khả năng server đẩy những nội dung tới client đã được kết nối một cách tức thì. Nó khác với giao thức HTTP thông thường: server đợi những yêu cầu từ client và trả về nội dung tương ứng.

SignalR có thể sử dụng trong bất kì chức năng web real-time nào. Trong đó ứng dụng chat trên web là một ví dụ điển hình. Ngoài ra, các ứng dụng cho dashboards, monitoring, collaborative là những gợi ý cho việc sử dụng SignalR.

SignalR cung cấp một API đơn giản cho việc tạo server-to-client remote procedure call (RPC) để gọi những hàm javascript trong trình duyệt (và những nền tảng khác) từ code .Net của server-side. SignalR cũng bao gồm API cho việc quản lý kết nối (connect và disconnect events) và những kết nối nhóm.

SignalR xử lý quản lý kết nối một cách tự động, và cho phép người dùng truyền đi thông điệp tới tất cả các client đã được kết nối một cách đồng loạt, giống như một chat room. Người dùng cũng có thể gửi những thông điệp tới những client được xác định. Kết nối giữa client và server là liên tục, không giống như kết nối HTTP cổ điển, cái mà sẽ thành lập lại kết nối cho mỗi lần giao tiếp.



Hình 2.7 Sơ đồ cách thức hoạt động của SignalR

SignalR hỗ trợ chức năng "server push", trong server code có thể gọi tới client code trong trình duyệt bởi "Remote Procedure Calls" (RPC), hơn là sử dụng Service Bus, SQL Server hay Redis.

SignalR là mã nguồn mở, có thể truy cập thông qua GitHub.

2.3.2 SignalR và WebSocket

SignalR sử dụng phương thức truyền tải WebSocket mới, và trở lại với phương thức truyền tải cũ hơn nơi cần thiết. Trong khi nhà phát triển có thể đảm bảo viết ứng dụng của mình sử dụng WebSocket một cách trực tiếp, sử dụng SignalR nghĩa là rất nhiều chức năng mở rộng mà nhà phát triển sẽ cần để triển khai đã được làm sẵn. Hầu hết các phần quan trọng, điều này muốn nói rằng nhà phát triển có thể code ứng dụng với những ưu điểm của

WebSocket mà không phải lo lắng về việc phân chia code cho những client cũ hơn. SignalR cũng sẽ giúp nhà phát triển về những cập nhật của WebSocket, từ đó SignalR sẽ tiếp tục được cập nhật để hỗ trợ những thay đổi trong truyền tải tầng bên dưới, cung cấp cho ứng dụng của nhà phát triển một giao diện thống nhất xuyên suốt các phiên bản của WebSocket.

Trong khi nhà phát triển đã có thể chắc chắn tạo một giải pháp sử dụng WebSocket, SignalR cung cấp tất cả đủ để lập trình theo cách nhà phát triển muốn, giống như việc trở lại các phương thức truyền tải khác và xem xét lại ứng dụng của nhà phát triển cho những cập nhật tới WebSocket.

2.3.3 Transport và fallbacks

SignalR là một tầng trừu tượng trên một số truyền tải mà yêu cầu để làm công việc thời gian thực giữa client và server. Một kết nối SignalR bắt đầu như một HTTP, và tiếp theo được đẩy lên một kết nối WebSocket nếu nó là có sẵn. WebSocket là một ý tưởng truyền tải cho SignalR, vì nó làm cho việc sử dụng bộ nhớ server hiệu quả nhất, có độ trễ thấp nhất, và có những tính năng cơ bản nhất (như giao tiếp hai chiều đầy đủ giữa client và server), nhưng nó cũng có những yêu cầu nghiêm ngặt nhất: WebSocket yêu cầu server sử dụng Windows Server 2012 hoặc Windows 8, và .Net 4.5. Nếu những yêu cầu này không được đáp ứng, SignalR sẽ cố gắng để sử dụng những truyền tải khác để làm những kết nối của nó.

2.3.4 Transport selection process

Danh sách bên dưới chỉ ra những bước mà SignalR sử dụng để quyết định truyền tải nào được sử dụng:

- Nếu trình duyệt là IE 8 hoặc cũ hơn, Long Polling được sử dụng.
- Nếu JSONP được cấu hình (tham số jsonp được gán là true khi kết nối được bắt đầu), Long Polling được sử dụng.
- Nếu một kết nối cross-domain được sử dụng (nếu SignalR endpoint không giống domain của hosting page) thì WebSocket sẽ được sử dụng nếu các tiêu chí bên dưới được đáp ứng:

- ✓ Client hỗ trợ CORS (Cross-Origin Resource Sharing).
- ✓ Client hỗ trợ WebSocket.
- ✓ Server hỗ trợ WebSocket. Nếu có bất kì tiêu chí nào ở trên không được đáp ứng, Long Polling sẽ được sử dụng.

- ➢ Nếu JSONP không được cấu hình và kết nối không cross-domain. WebSocket sẽ được sử dụng nếu cả client và server đều hỗ trợ nó.
- ➢ Nếu một trong hai đối tượng server và client không hỗ trợ WebSocket, Server Sent Events sẽ được sử dụng nếu nó có sẵn.
- ➢ Nếu Server Sent Events không có sẵn, Forever Frame được sử dụng.
- ➢ Nếu Forever Frame thất bại, Long Polling được sử dụng.

2.4 Giới thiệu Mircrosoft SQL Server

2.4.1 SQL sever là gì?

SQL server hay còn được gọi là Microsoft SQL Server, nó từ viết tắt của MS SQL Server. Đây chính là một loại phần mềm đã được phát triển bởi Microsoft và nó được sử dụng để có thể dễ dàng lưu trữ cho những dữ liệu dựa theo tiêu chuẩn RDBMS.

Từ đó, người ta sẽ lưu trữ dữ liệu dựa vào tiêu chuẩn RDBMS và nó cũng là một trong những hệ quản trị cơ sở dữ liệu dạng quan hệ đối tượng.

Hình 2.8 SQL Server là gì?

SQL Server có khả năng cung cấp đầy đủ các công cụ cho việc quản lý từ giao diện GUI đến sử dụng ngôn ngữ cho việc truy vấn SQL. Điểm mạnh của SQL điểm mạnh của nó là có nhiều nền tảng được kết hợp cùng như: ASP.NET, C# để xây dựng Winform cũng chính nó có khả năng hoạt động độc lập. Tuy nhiên, SQL Server thường đi kèm với việc thực hiện riêng các ngôn ngữ SQL, T-SQL, ...

Cụ thể như sau:

- T-SQL là một trong những loại ngôn ngữ thuộc quyền sở hữu của Microsoft và được gọi với cái tên Transact-SQL. Nó thường cung cấp thêm rất nhiều cho các khả năng khai báo biến, thủ tục lưu trữ và xử lý ngoại lệ, ...
- SQL Server Management Studio là một loại công cụ giao diện chính cho máy chủ cơ sở của chính dữ liệu SQL, thông thường thì nó hỗ trợ cho cả môi trường 64-bit và 32-bit.

2.4.2 Mục đích khi sử dụng SQL Server

SQL Server thông thường được sử dụng cho mục đích lưu trữ dữ liệu. Ngoài ra, nó còn mang lại những tính năng làm việc giúp người dùng làm việc hiệu quả hơn như sau:

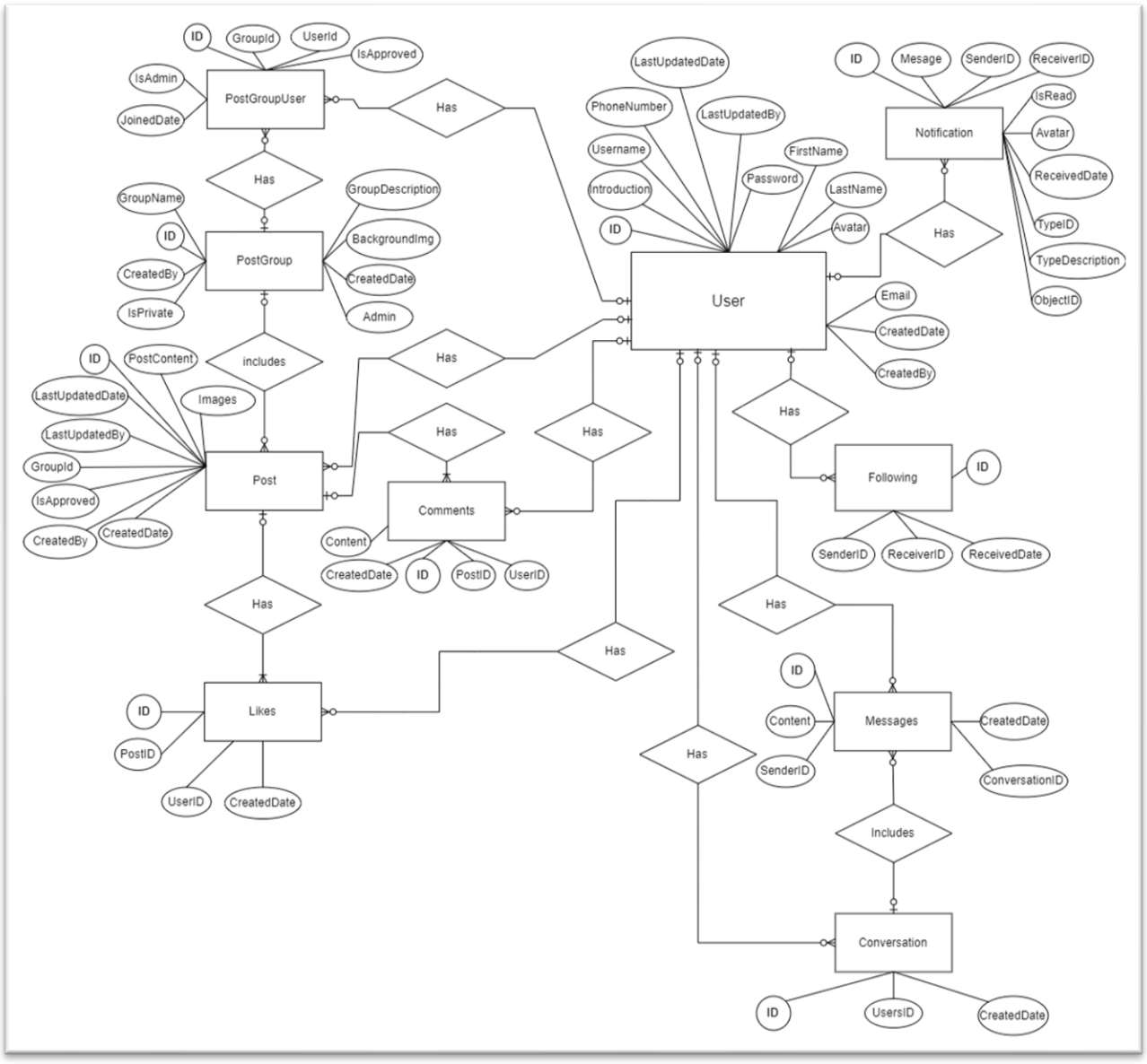
- Giúp người sử dụng có thể duy trì việc lưu trữ bền vững.
- Cho phép tạo ra nhiều cơ sở dữ liệu hơn.
- Có khả năng phân tích dữ liệu bằng SSAS
- Nó có khả năng bảo mật cao
- Việc tạo ra được các báo cáo bằng SSRS — SQL Server Reporting Services sẽ được dễ dàng hơn.
- Các quá trình sẽ được thực hiện bằng SSIS — SQL Server Integration Services.

2.5 Các thư viện khác

- MDBootstrap UI Kits
- Jquery
- Fontawesome

CHƯƠNG 3 PHÂN TÍCH HỆ THỐNG

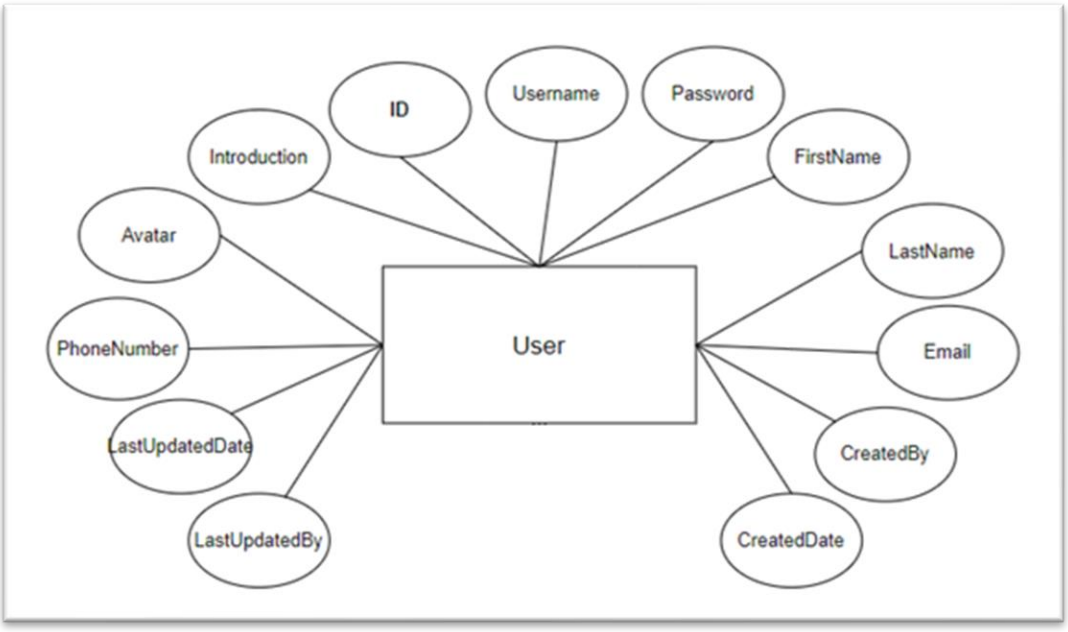
3.1 Mô tả hệ thống



Hình 3.1 Mô hình ER toàn hệ thống

➢ **Quản lý người dùng**

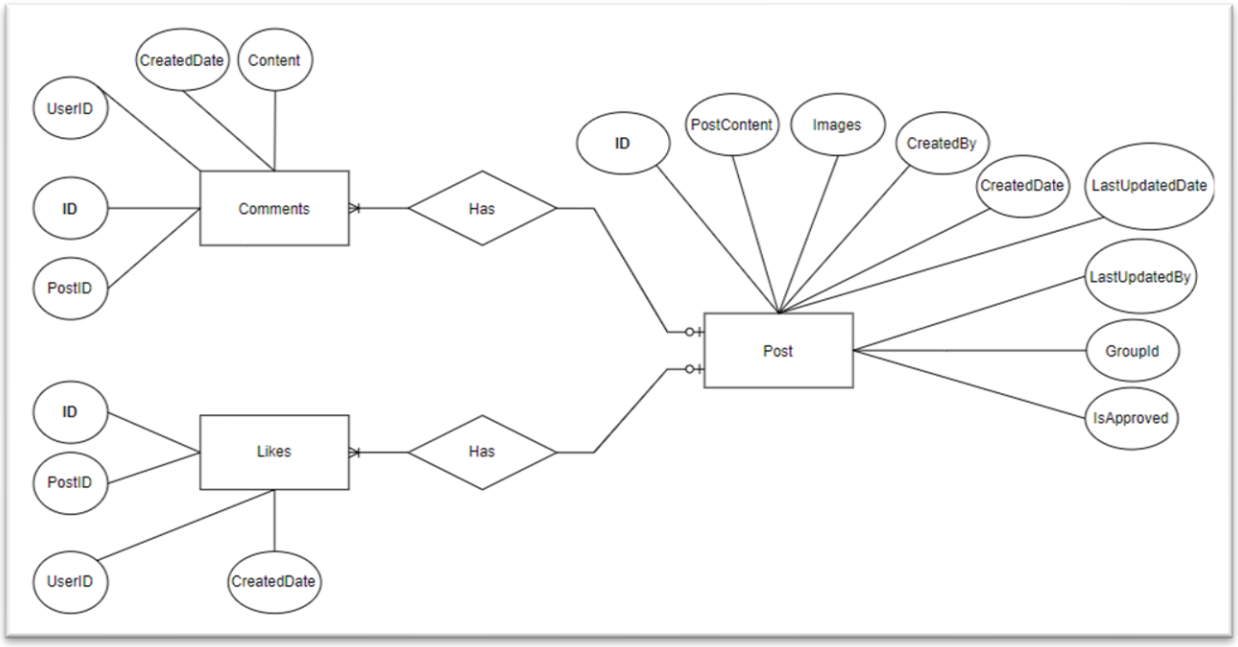
- Quản lý người dùng bao gồm tên người dùng, mật khẩu, họ, tên, email, mã người tạo, ngày tạo, người cập nhật lần cuối, ngày cập nhật lần cuối, số điện thoại, hình đại diện, giới thiệu bản thân.
- Một người dùng được quản lý theo quyền truy cập. Một người dùng chỉ được có một quyền. Sẽ có nhiều quyền truy cập trong đó chứa nhiều người dùng cùng quyền.



Hình 3.2 Mô hình ER quản lý người dùng

➢ **Quản lý bài đăng**

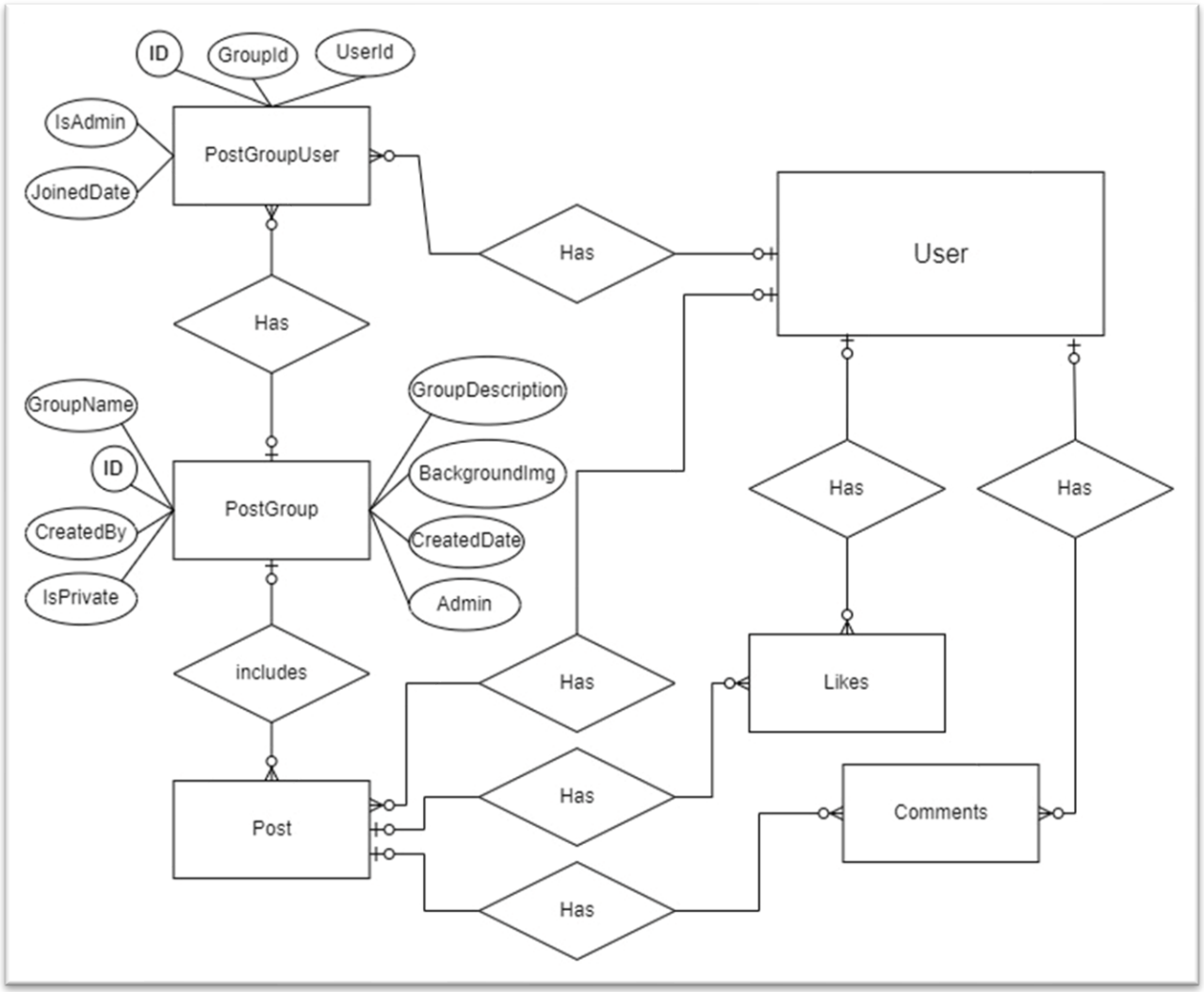
- Quản lý bài đăng bao gồm nội dung, hình ảnh, người tạo, ngày tạo, người cập nhật lần cuối, ngày cập nhật lần cuối, nhóm (nếu có), trạng thái (nếu đăng bài vào nhóm riêng tư).
- Một bài đăng sẽ có nhiều lượt thích.
- Một bài đăng sẽ có nhiều lượt bình luận.



Hình 3.3 Mô hình ER quản lý bài đăng

- **Quản lý nhóm**

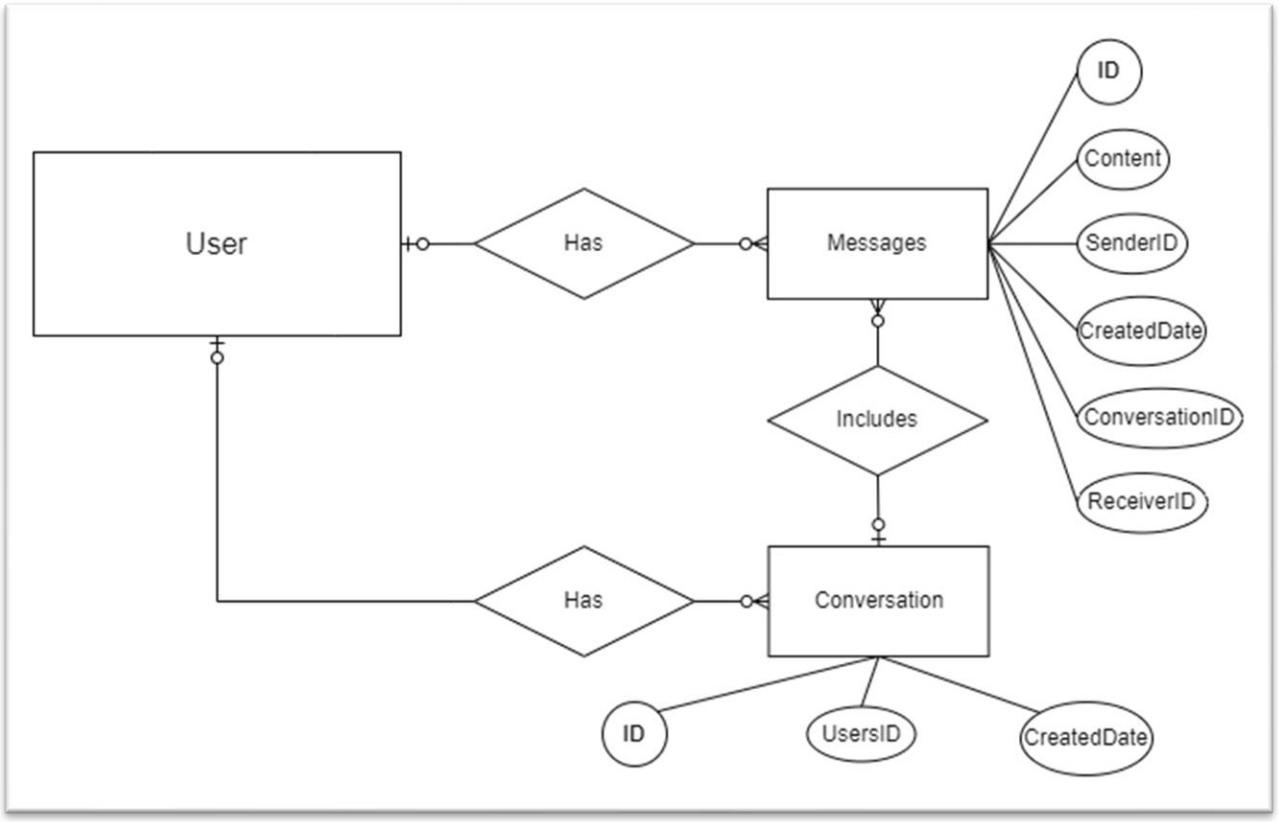
- Quản lý nhóm bao gồm các thông tin tên nhóm, mô tả nhóm, ảnh nền, ngày tạo, quản trị viên, người tạo nhóm, trạng thái nhóm (nhóm công khai hoặc nhóm riêng tư).
- Một nhóm sẽ gồm nhiều người dùng.
- Một nhóm sẽ gồm nhiều bài viết.



Hình 3.4 Mô hình ER quản lý nhóm

- **Quản lý tin nhắn**

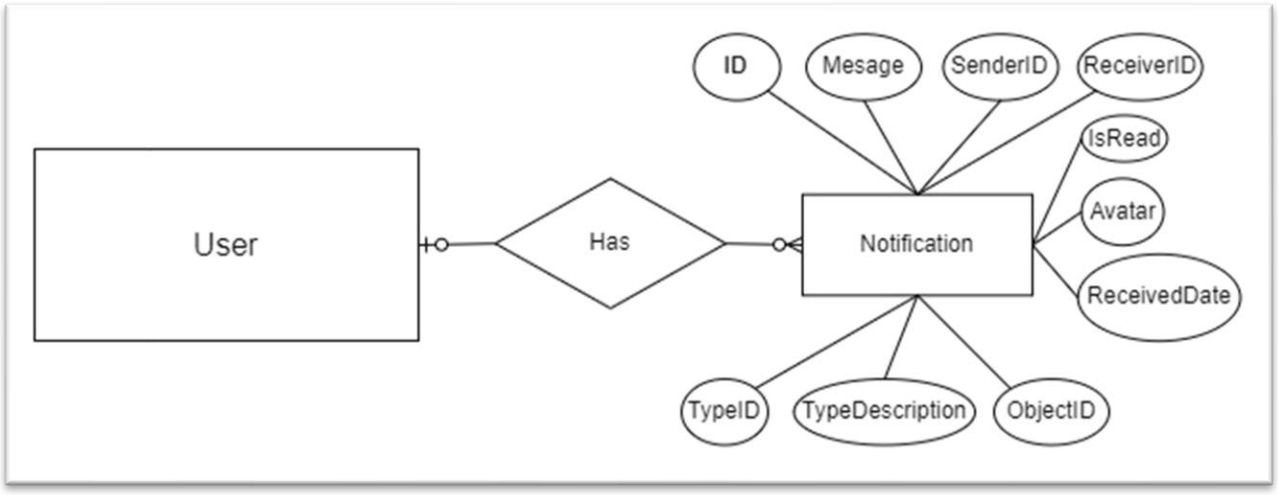
- Quản lý tin nhắn gồm các thông tin mã tin nhắn, nội dung tin nhắn, người gửi, ngày gửi, người nhận, mã đoạn hội thoại.
- Đoạn hội thoại gồm các thông tin mã hội thoại, các người dùng tham gia, ngày tạo.
- Một người dùng sẽ có nhiều tin nhắn, tham gia vào nhiều đoạn hội thoại.
- Một đoạn hội thoại sẽ có nhiều tin nhắn.



Hình 3.5 Mô hình ER quản lý tin nhắn

- **Quản lý thông báo**

- Quản lý thông báo gồm các thông tin mã thông báo, nội dung thông báo, người gửi, người nhận, trạng thái đọc, ảnh đại diện người gửi, ngày nhận, mã loại thông báo, mô tả loại thông báo, mã đối tượng thông báo.
- Một người dùng có nhiều thông báo.



Hình 3.6 Mô hình ER quản lý thông báo

- **Các chức năng của Website mạng xã hội**

- Xem tất cả các bài đăng trên hệ thống.
- Cập nhật thông tin.
- Thao tác với bài đăng.
- Thao tác với nhóm.
- Tìm kiếm nhóm, người dùng có trên hệ thống.
- Theo dõi người dùng khác.
- Nhắn tin với người dùng khác.
- Nhận thông báo.

3.2 Xây dựng các chức năng của hệ thống

3.2.1 Mô tả các chức năng chi tiết

- Xem tất cả các bài đăng trên hệ thống.

Bảng 3-1 Bảng mô tả chứa năng xem tất cả bài đăng.

STT	Chức năng	Diễn giải
1	Truy xuất dữ liệu	Cho phép truy xuất tất cả bài đăng có trên hệ thống và hiển thị dưới dạng Timeline

- Cập nhật thông tin.

Bảng 3-2 Bảng mô tả chứa năng cập nhật thông tin.

STT	Chức năng	Diễn giải
1	Cập nhật thông tin tài khoản	Cho phép người dùng cập nhật thông tin tại khoản của mình

- Thao tác với bài đăng.

Bảng 3-3 Bảng mô tả chức năng thao tác với bài đăng.

STT	Chức năng	Diễn giải
1	Thích	Cho phép người dùng để lại một lượt thích cho bài đăng bất kì
2	Bình luận	Cho phép người dùng để lại bình luận cho bài đăng bất kì
3	Xóa	Cho phép người dùng xóa bài đăng của mình
4	Cập nhật thông tin	Cho phép người dùng cập nhật thông tin bài đăng của minh
5	Đăng bài mới	Cho phép người dùng đăng tải bài viết mới

- Tìm kiếm tất cả các nhóm, người dùng có trên hệ thống.

Bảng 3-4 Bảng mô tả chức năng tìm kiếm nhóm, người dùng có trên hệ thống.

STT	Chức năng	Diễn giải
1	Tìm kiếm	Cho phép người dùng tìm kiếm tất cả các nhóm, người dùng có trên hệ thống bằng chuỗi kí tự

- Thao tác với nhóm bài đăng.

Bảng 3-5 Bảng mô tả chức năng thao tác với nhóm.

STT	Chức năng	Diễn giải
1	Tham gia	Cho phép người dùng tham gia vào nhóm bất kì
2	Thoát	Cho phép người dùng thoát khỏi nhóm đang tham gia
3	Đăng bài mới	Cho phép người dùng đăng tải bài viết mới vào nhóm
4	Tạo mới	Cho phép người dùng tạo nhóm mới

5	Mời khỏi nhóm	Cho phép Admin nhóm mời người dùng bất kì ra khỏi nhóm
6	Duyệt thành viên	Cho phép Admin nhóm duyệt thành viên vào nhóm
7	Duyệt bài đăng	Cho phép Admin nhóm duyệt bài đăng vào nhóm

- Theo dõi người dùng khác.

Bảng 3-6 Bảng mô tả chức năng theo dõi người dùng khác.

STT	Chức năng	Diễn giải
1	Theo dõi	Cho phép người dùng theo dõi người dùng khác
2	Được theo dõi	Cho phép người dùng được người dùng khác theo dõi

- Nhắn tin với người dùng khác.

Bảng 3-7 Bảng mô tả chức năng nhắn tin với người dùng khác.

STT	Chức năng	Diễn giải
1	Gửi tin nhắn	Cho phép người dùng gửi tin nhắn đến người dùng khác
2	Nhận tin nhắn	Cho phép người dùng nhận tin nhắn từ người khác

- Nhận thông báo.

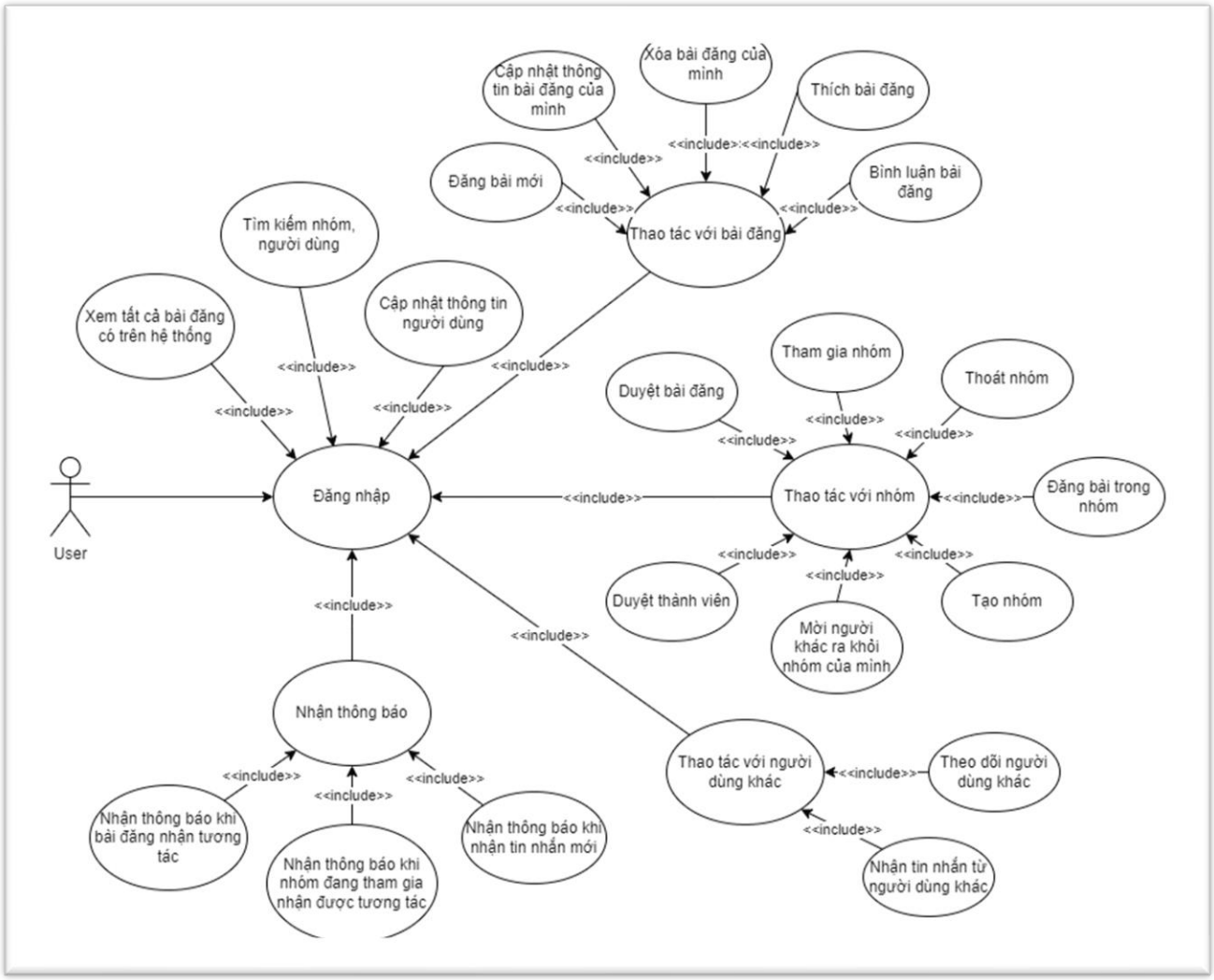
Bảng 3-8 Bảng mô tả chức năng nhận thông báo.

STT	Chức năng	Diễn giải
1	Nhận thông báo khi bài đăng nhận được tương tác	Cho phép người dùng nhận thông báo khi bài đăng của người dùng đó được thích hoặc bình luận
2	Nhận thông báo khi nhận tin nhắn mới	Cho phép người dùng nhận thông báo khi có tin nhắn mới
3	Nhận thông báo khi nhóm đang tham gia nhận được tương tác	Cho phép người dùng nhận thông báo khi nhóm đang tham gia có bài viết mới hoặc bị Admin nhóm mời ra khỏi nhóm

3.2.2 Xác định use case

User truy cập vào website cho người dùng để xem các bài đăng, thao tác với bài đăng, thao tác với nhóm, cập nhật thông tin người dùng, theo dõi người dùng khác, nhắn tin với người dùng khác, nhận thông báo, tìm kiếm nhóm và người dùng có trên hệ thống.

Ngoài ra, khi người dùng là Admin của một nhóm, người dùng còn có thêm các chức năng quản lý các đối tuợng trong nhóm đó.

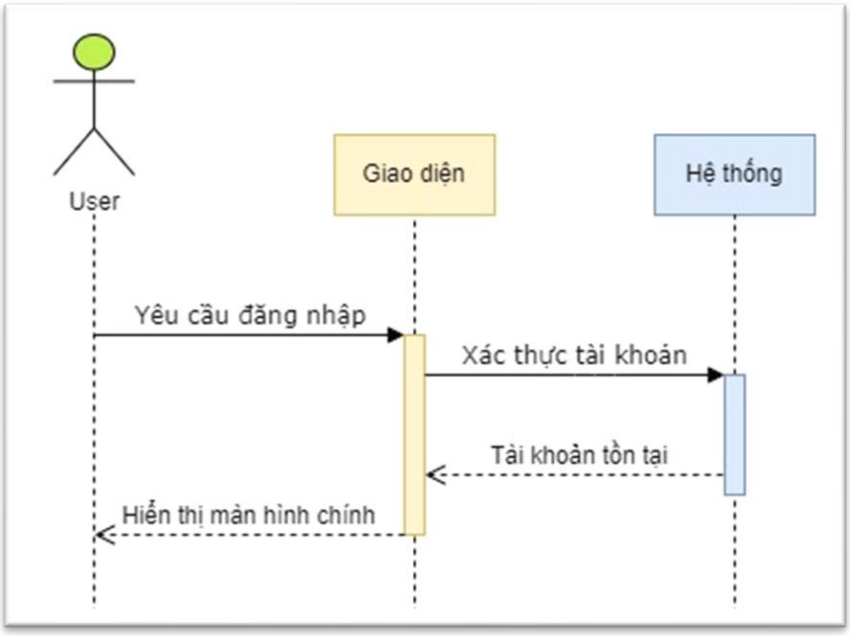


Sơ đồ 3-1 Sơ đồ Use case.

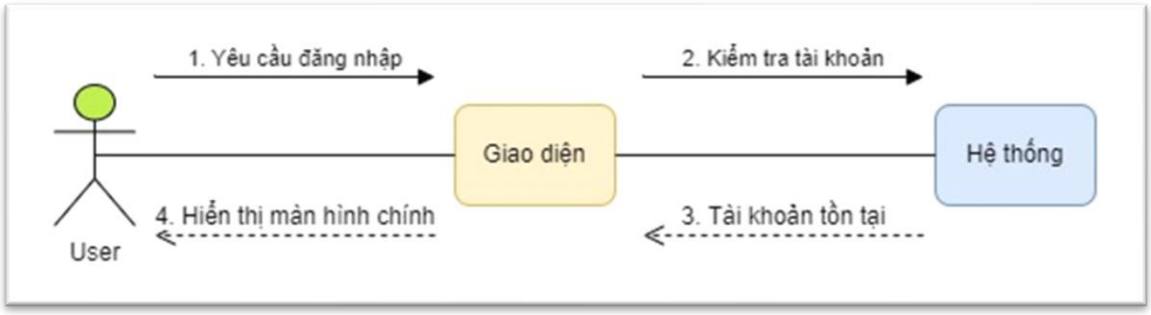
3.2.3 Đặc tả các use case

3.2.3.1 Mô tả quá trình đăng nhập

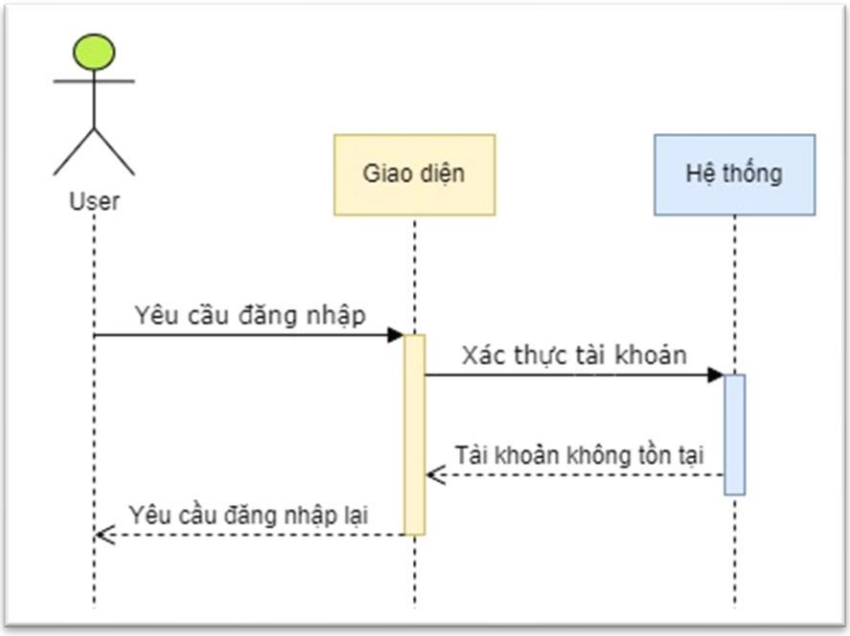
Đặc tả: Use case mô tả quá trình đăng nhập vào hệ thống. Người dùng đăng nhập email/password. Nếu tài khoản được xác minh hiển thị giao diện trang màn hình chính. Nếu tài khoản không chính xác yêu cầu đăng nhập lại.



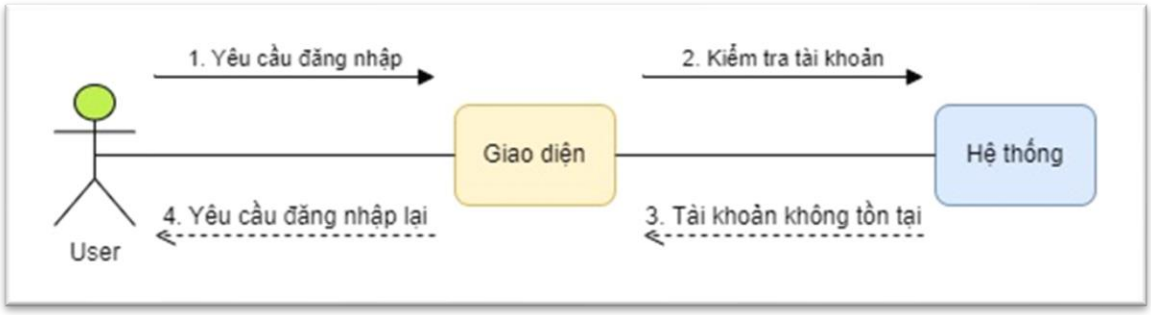
Sơ đồ 3-2 Sơ đồ tuần tự đăng nhập thành công



Sơ đồ 3-3 Sơ đồ hợp tác đăng nhập thành công



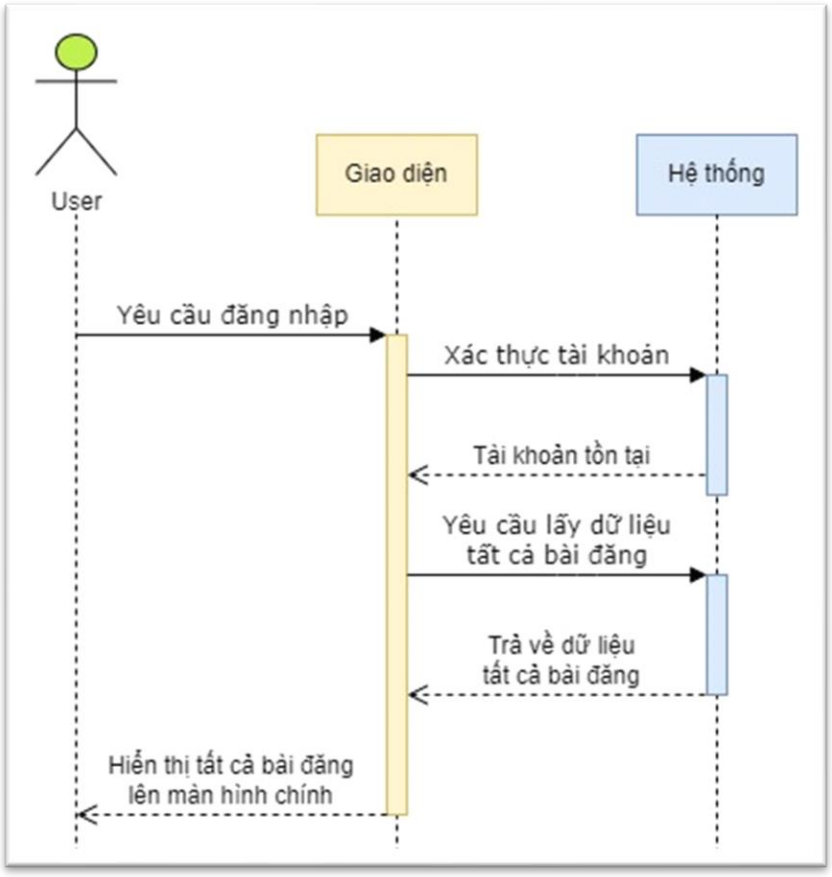
Sơ đồ 3-4 Sơ đồ tuần tự đăng nhập thất bại



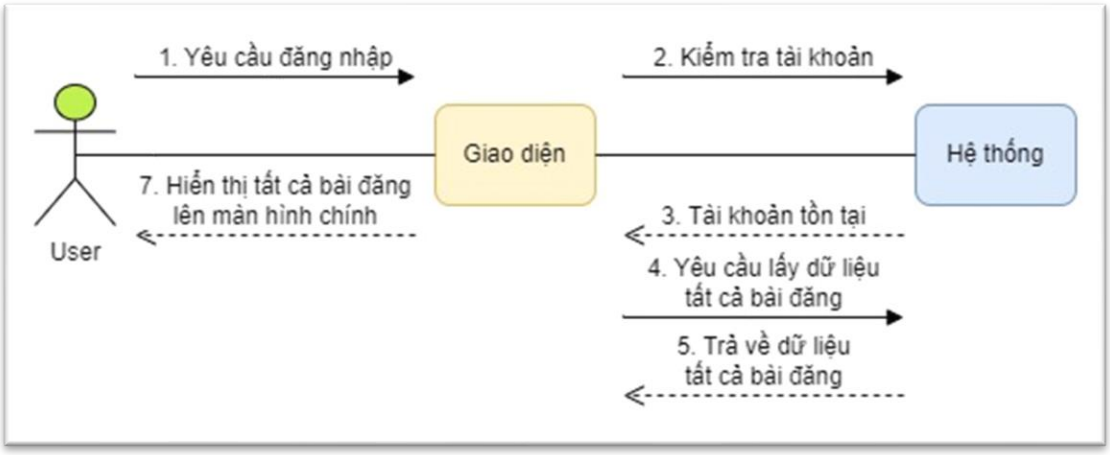
Sơ đồ 3-5 Sơ đồ hợp tác đăng nhập thất bại

3.2.3.2 Mô tả chức năng xem tất cả bài đăng trên hệ thống

Đặc tả: Khi người dùng đăng nhập và tài khoản đã được xác minh hiển thị trang màn hình chính. Tại trang chính của website người dùng, tất cả các bài đăng có trên hệ thống sẽ được hiển thị dưới theo dạng Timeline và được sắp xếp theo thứ tự thời gian từ mới đến cũ.



Sơ đồ 3-6 Sơ đồ tuần tự chức năng xem tất cả bài đăng trên hệ thống

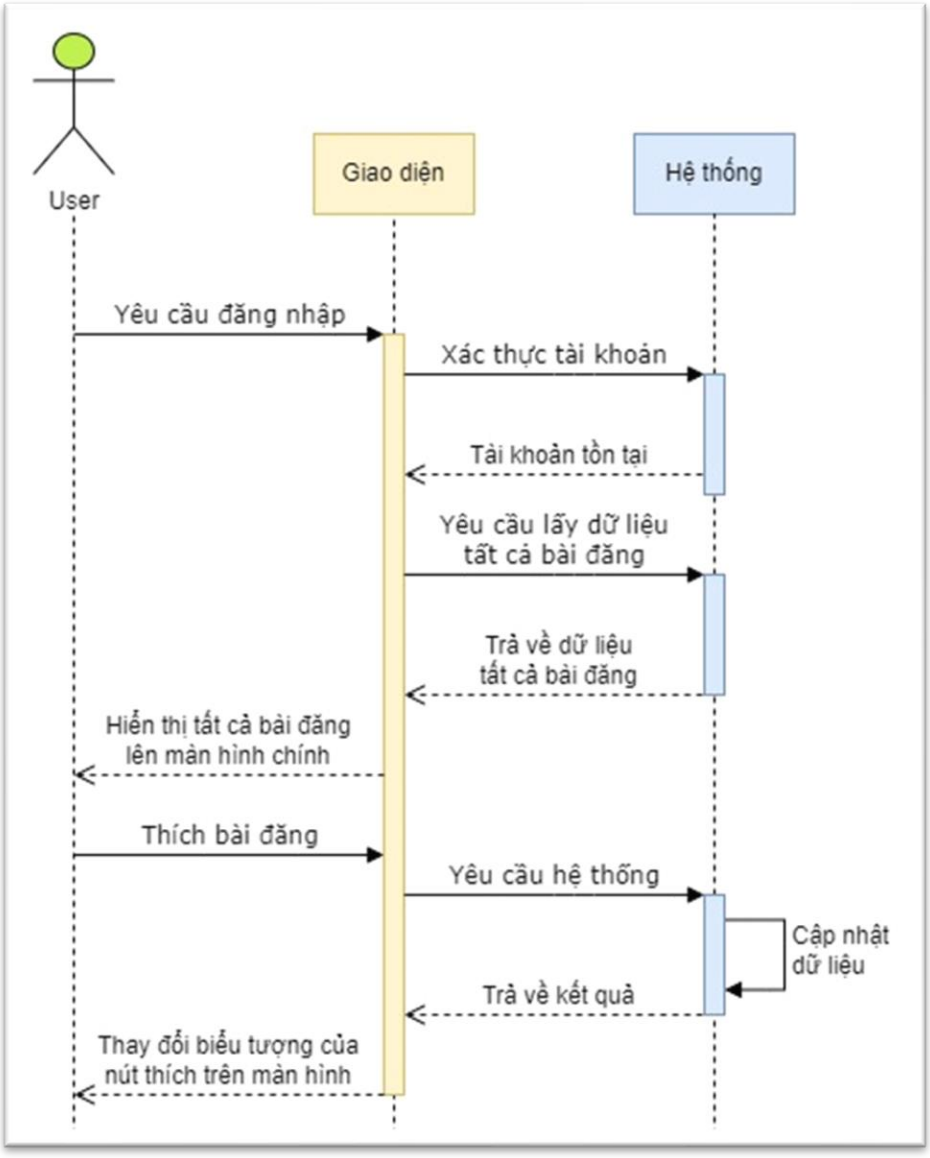


Sơ đồ 3-7 Sơ đồ hợp tác chức năng xem tất cả bài đăng trên hệ thống

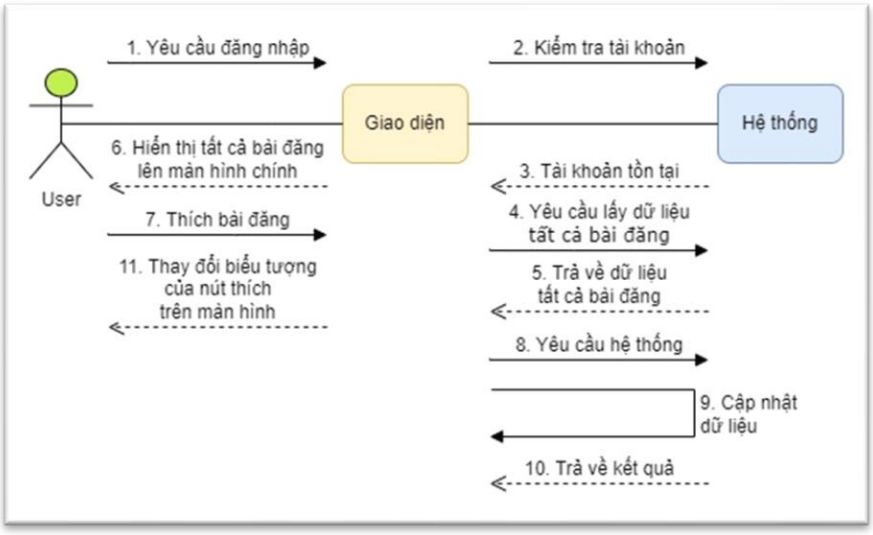
3.2.3.3 Mô tả chức năng thao tác với bài đăng

Đặc tả: Khi người dùng đăng nhập và tài khoản đã được xác minh hiển thị trang màn hình chính. Tất cả các bài đăng trên hệ thống sẽ được hiển thị ở trang chính và cho phép người dùng thích, để lại bình luận cho mỗi bài đăng.

Chức năng thích bài đăng của người dùng:

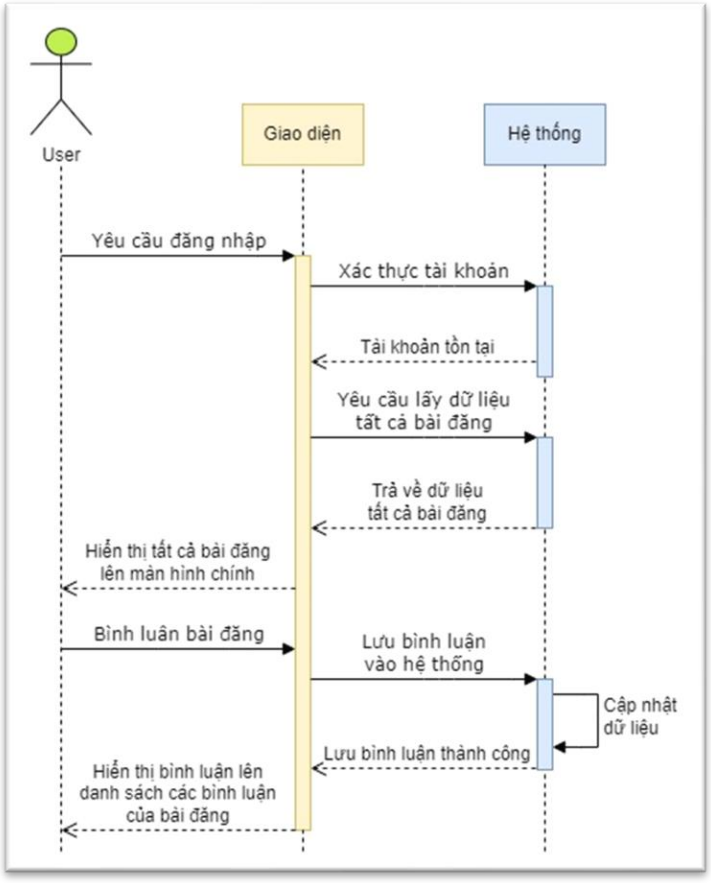


Sơ đồ 3-8 Sơ đồ tuần tự chức năng thích bài đăng

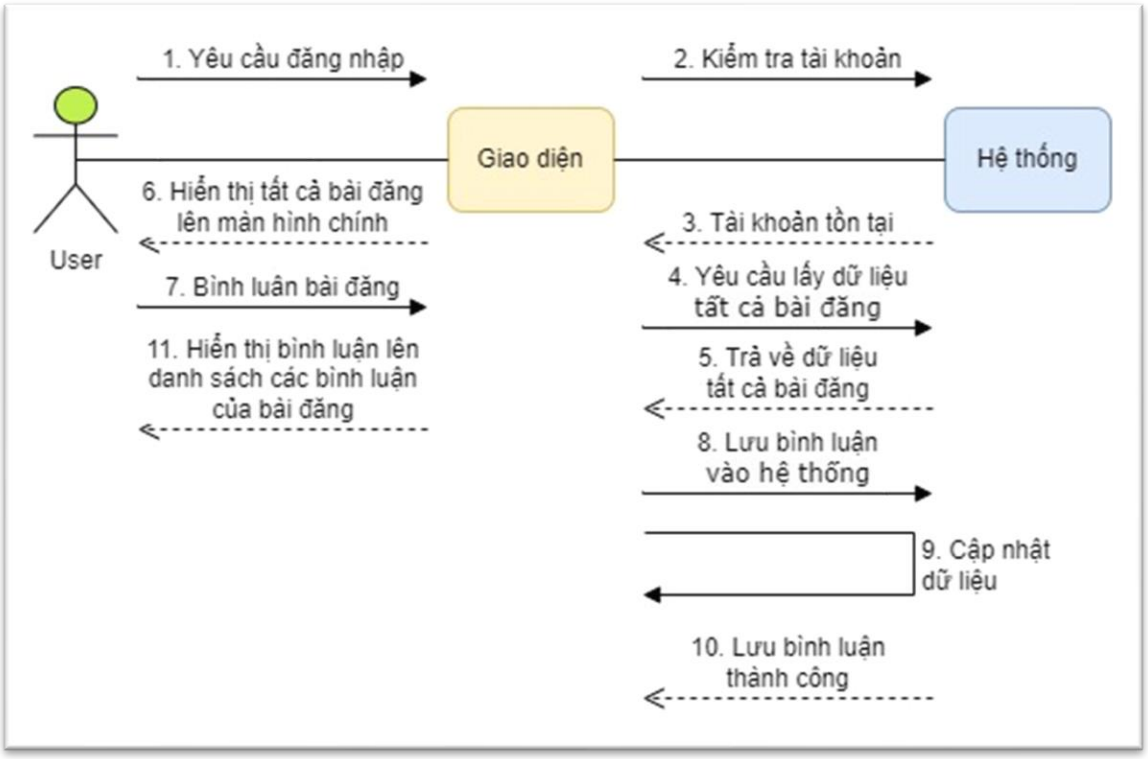


Sơ đồ 3-9 Sơ đồ hợp tác chức năng thích bài đăng

Chức năng bình luận bài đăng của người dùng:



Sơ đồ 3-10 Sơ đồ tuần tự chức năng bình luận bài đăng

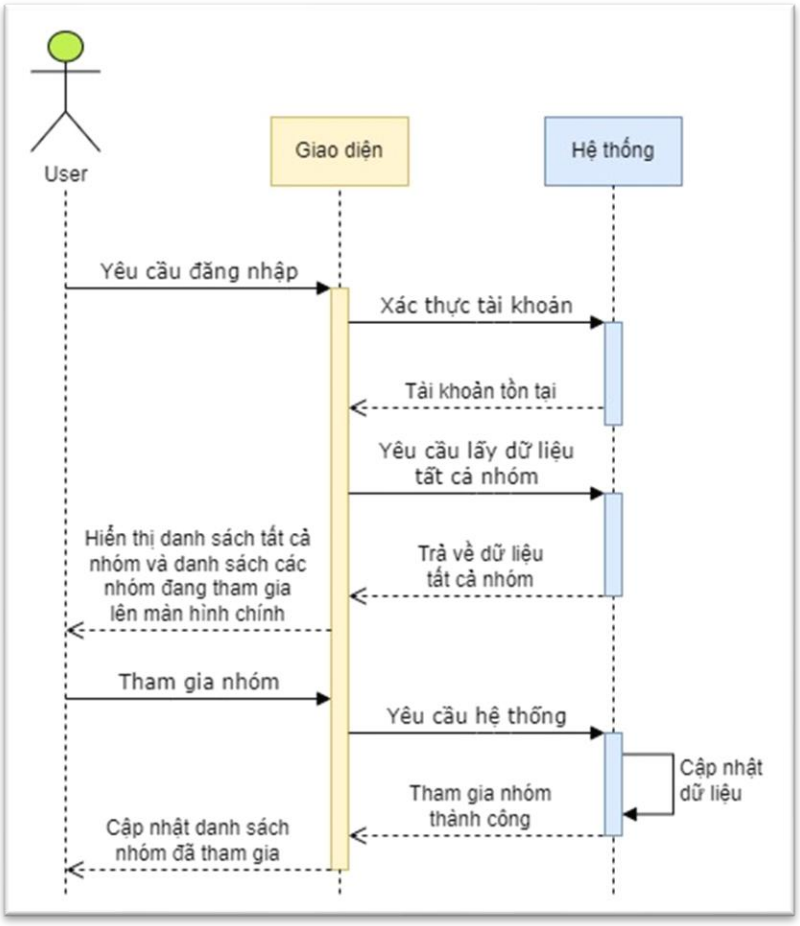


Sơ đồ 3-11 Sơ đồ hợp tác chức năng bình luận bài đăng

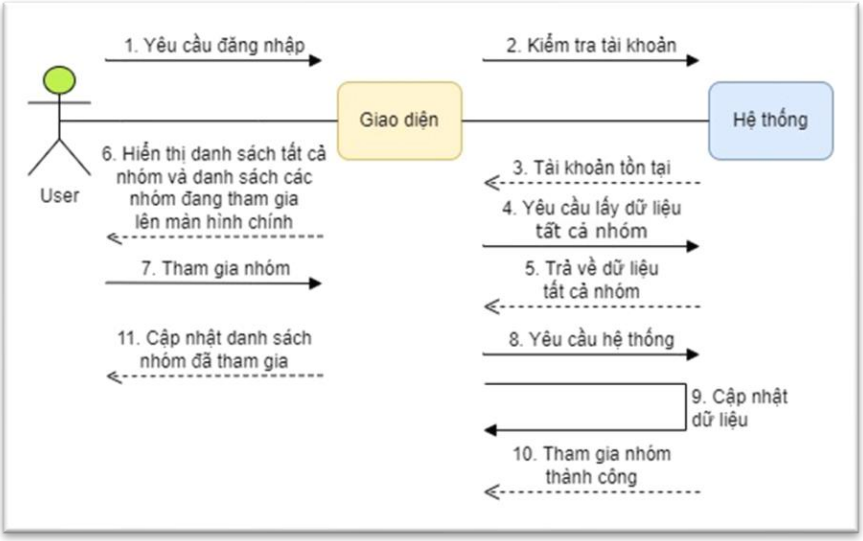
3.2.3.4 Mô tả chức năng thao tác với nhóm

Đặc tả: Khi người dùng đăng nhập và tài khoản đã được xác minh hiển thị trang màn hình chính. Tất cả các nhóm trên hệ thống sẽ được hiển thị ở trang chính và cho phép người dùng tham gia vào nhóm. Đối với các nhóm đang tham gia, hệ thống sẽ cho phép người dùng đăng bài vào nhóm, rời khỏi nhóm. Admin nhóm sẽ có chức năng cập nhật thông tin nhóm, xóa thành viên khỏi nhóm.

Chức năng tham gia nhóm:

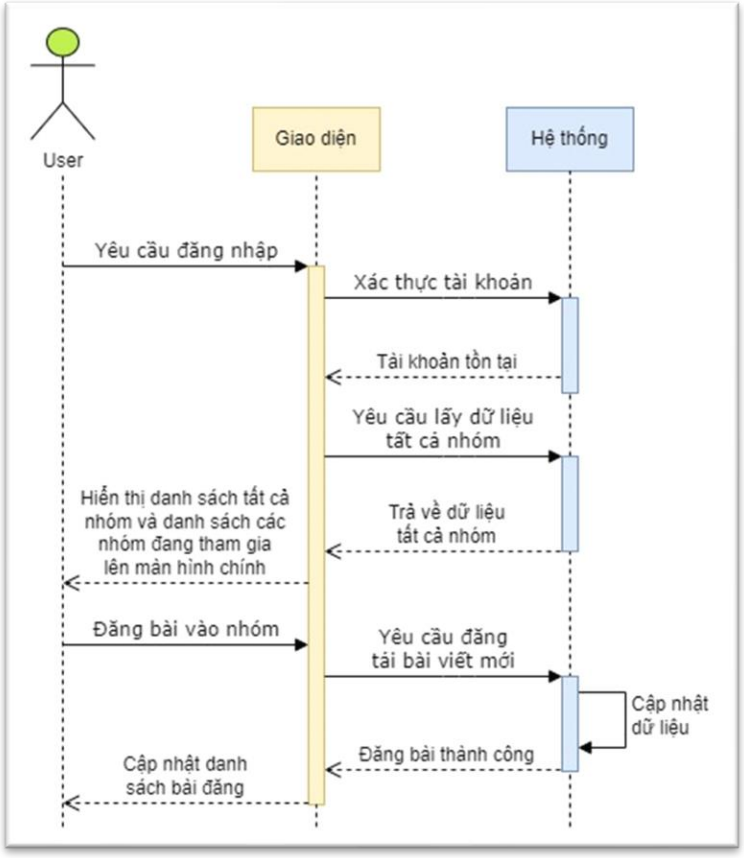


Sơ đồ 3-12 Sơ đồ tuần tự chức năng tham gia nhóm

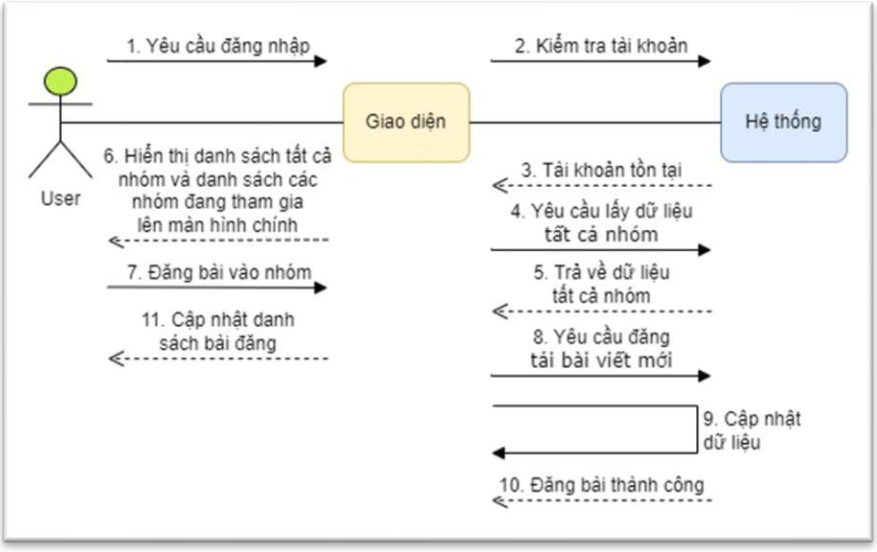


Sơ đồ 3-13 Sơ đồ hợp tác chức năng tham gia nhóm

Chức năng đăng bài vào nhóm:

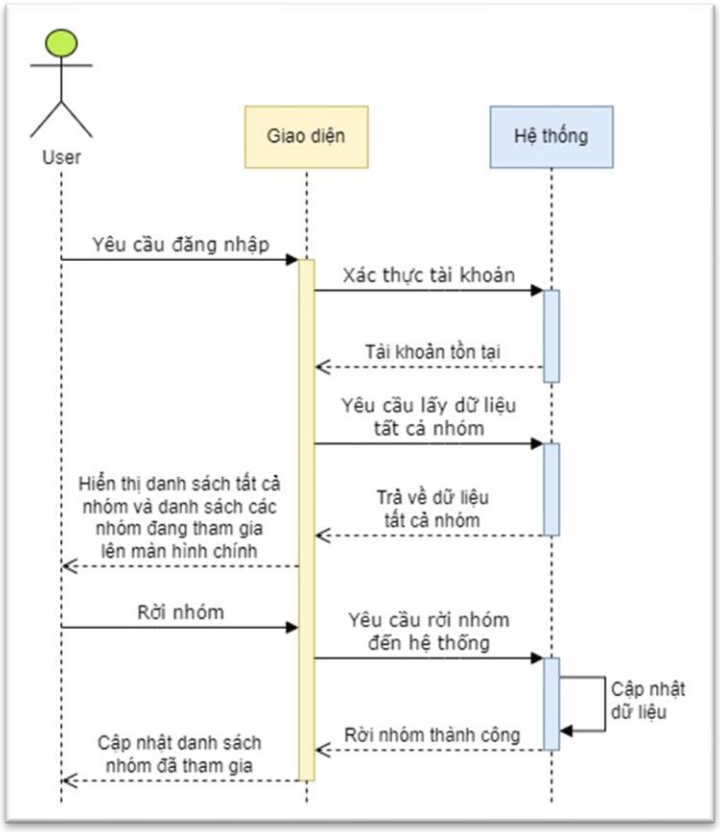


Sơ đồ 3-14 Sơ đồ tuần tự chức năng đăng bài vào nhóm

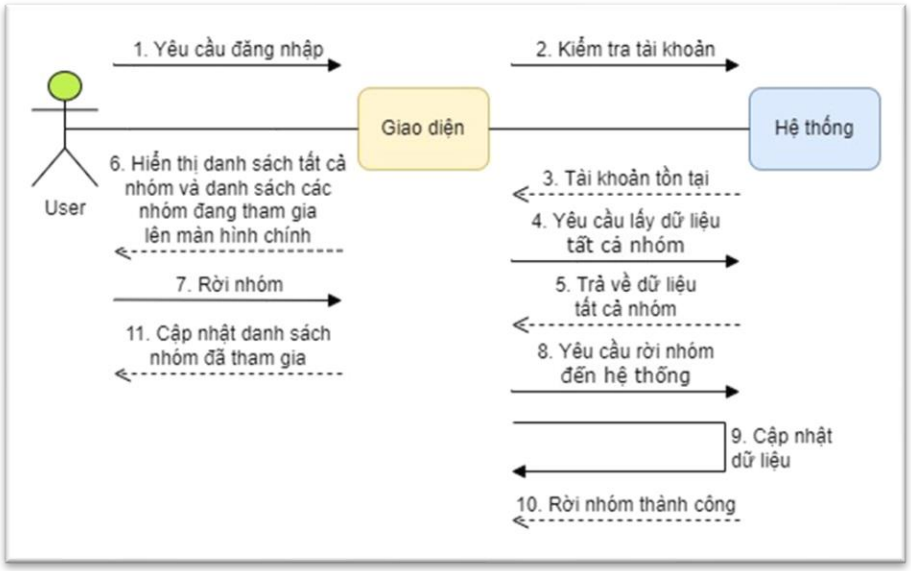


Sơ đồ 3-15 Sơ đồ hợp tác chức năng đăng bài vào nhóm

Chức năng rời nhóm:

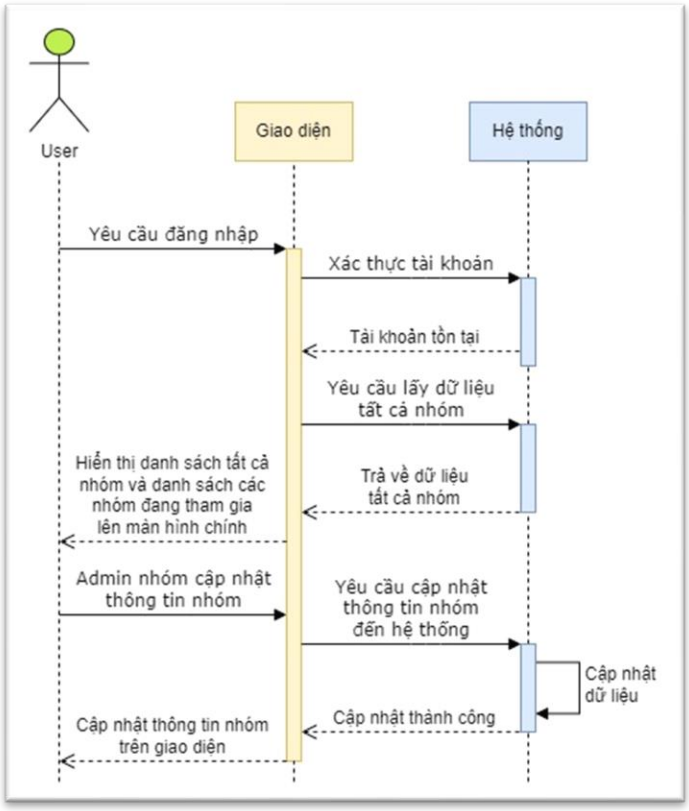


Sơ đồ 3-16 Sơ đồ tuần tự chức năng rời nhóm

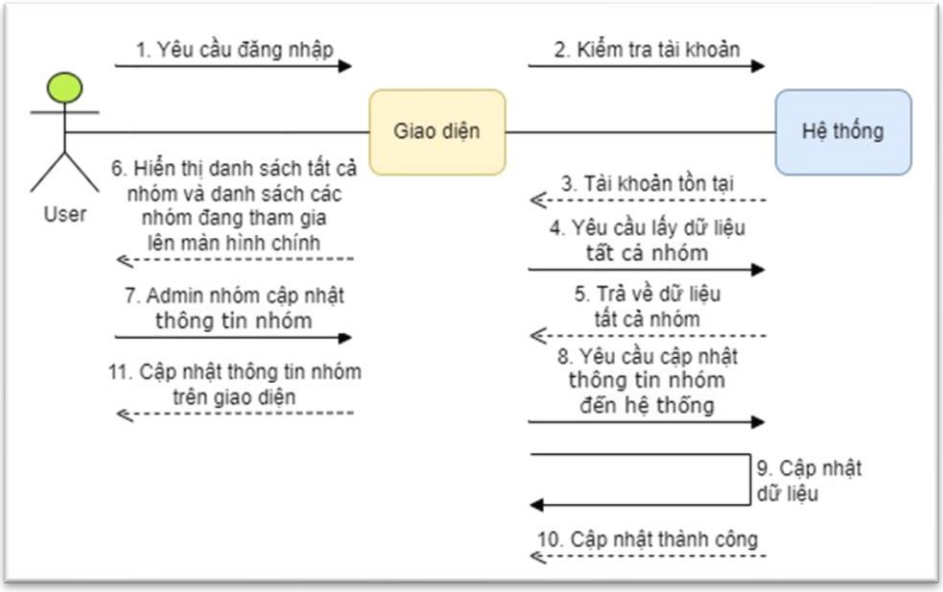


Sơ đồ 3-17 Sơ đồ hợp tác chức năng rời nhóm

Chức năng cập nhật thông tin nhóm cho Admin nhóm:

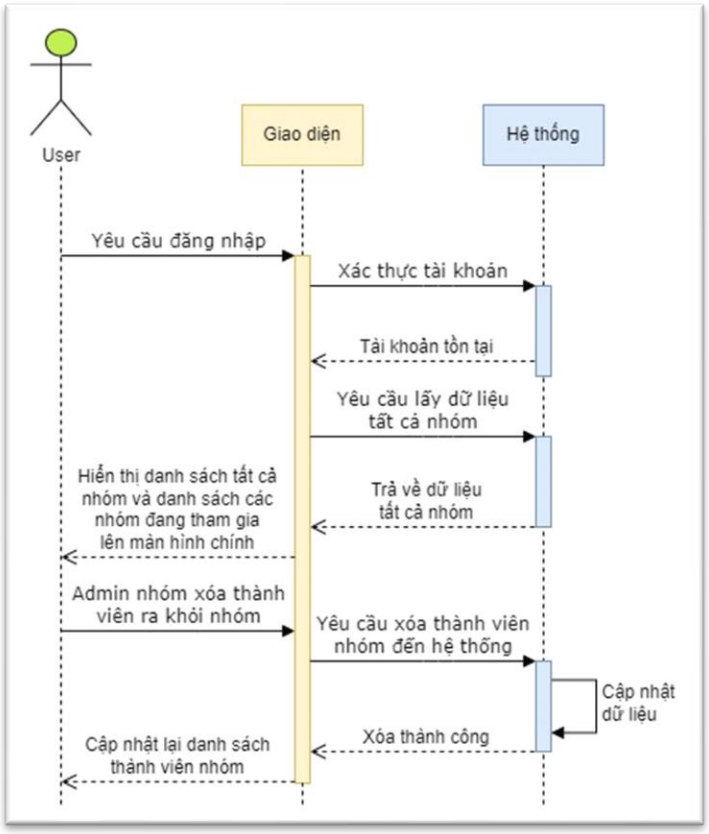


Sơ đồ 3-18 Sơ đồ tuần tự chức năng cập nhật thông tin nhóm cho Admin nhóm

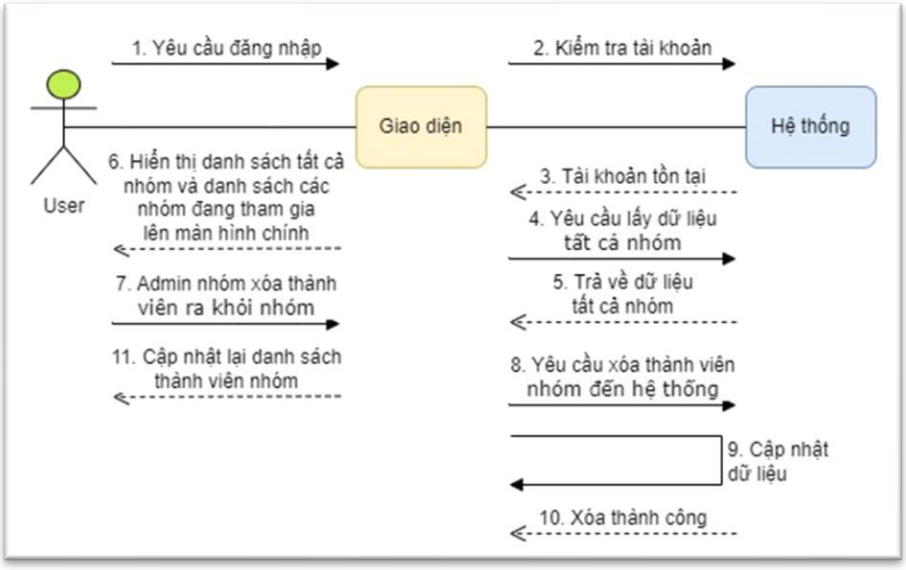


Sơ đồ 3-19 Sơ đồ hợp tác chức năng cập nhật thông tin nhóm cho Admin nhóm

Chức năng xóa thành viên của Admin nhóm:



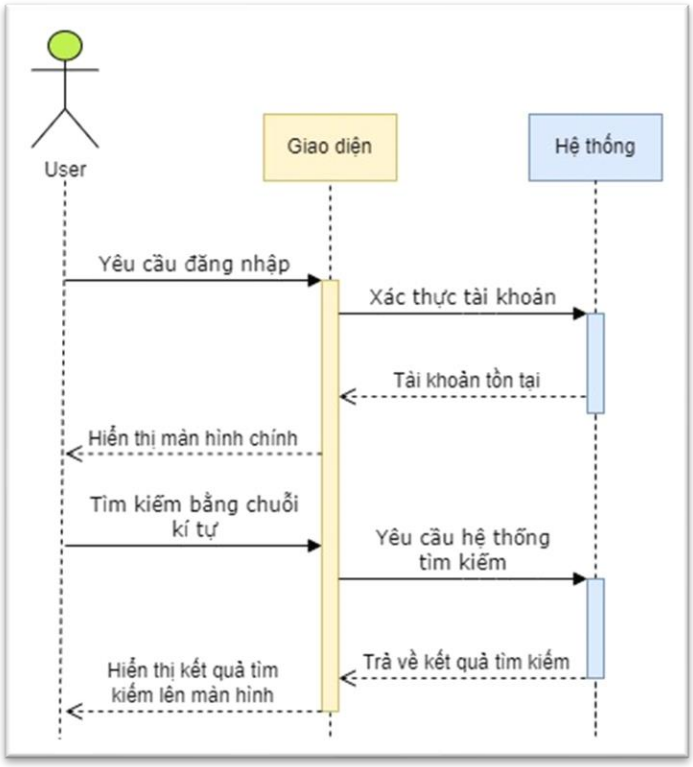
Sơ đồ 3-20 Sơ đồ tuần tự chức năng xóa thành viên của Admin nhóm



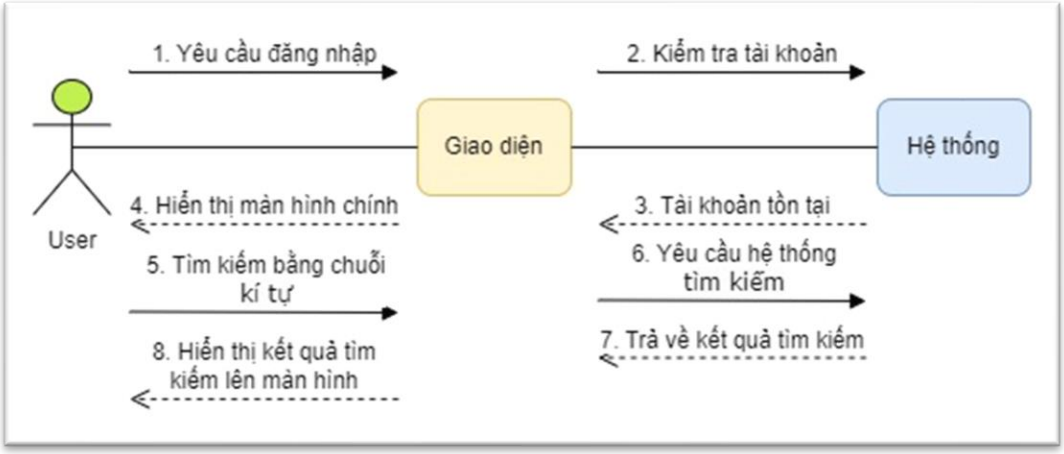
Sơ đồ 3-21 Sơ đồ hợp tác chức năng xóa thành viên của Admin nhóm

3.2.3.5 Mô tả chức năng tìm kiếm nhóm, người dùng có trên hệ thống

Đặc tả: Khi người dùng đăng nhập và tài khoản đã được xác minh hiển thị trang màn hình chính. Chức năng tìm kiếm sẽ cho phép người dùng tìm kiếm được nhóm, người dùng khác trên hệ thống.



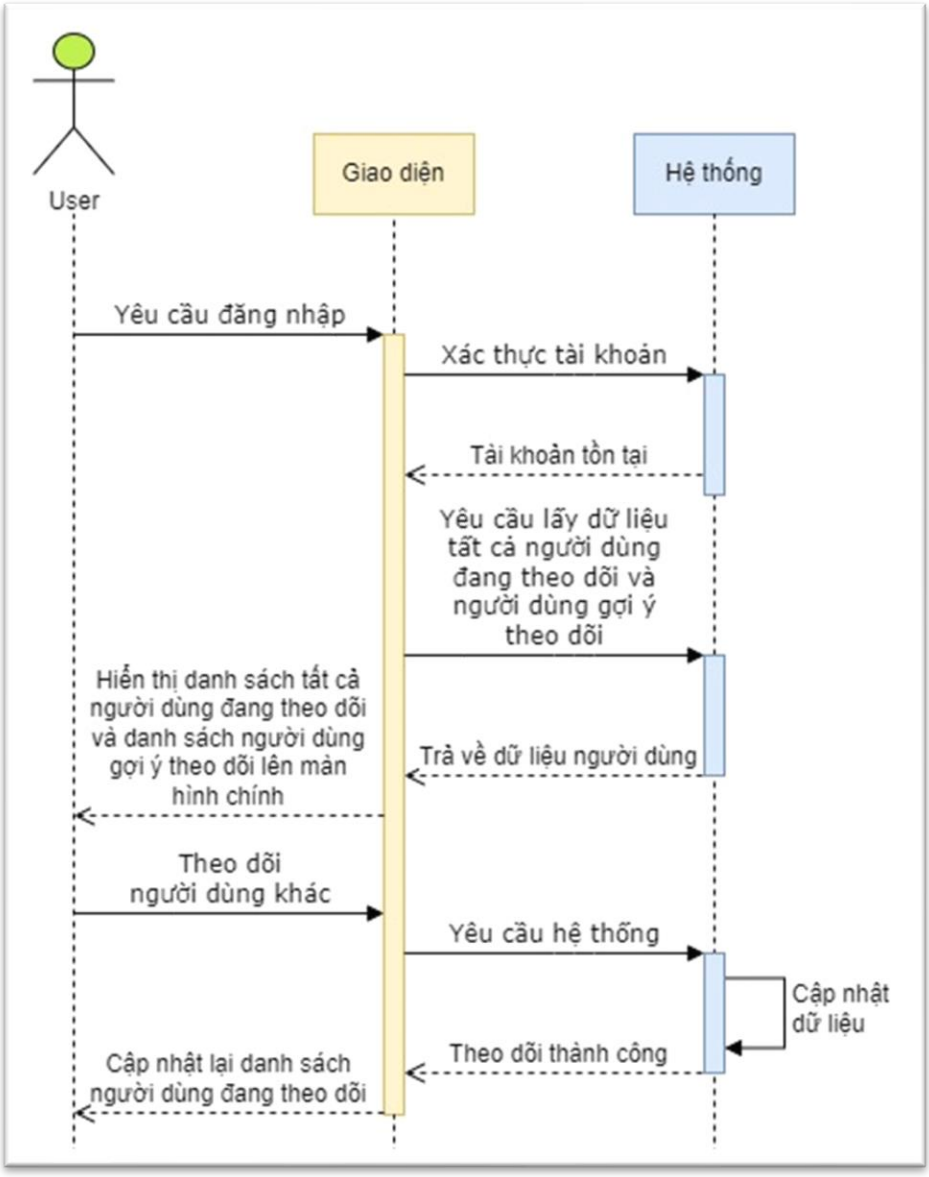
Sơ đồ 3-22 Sơ đồ tuần tự chức năng tìm kiếm của người dùng



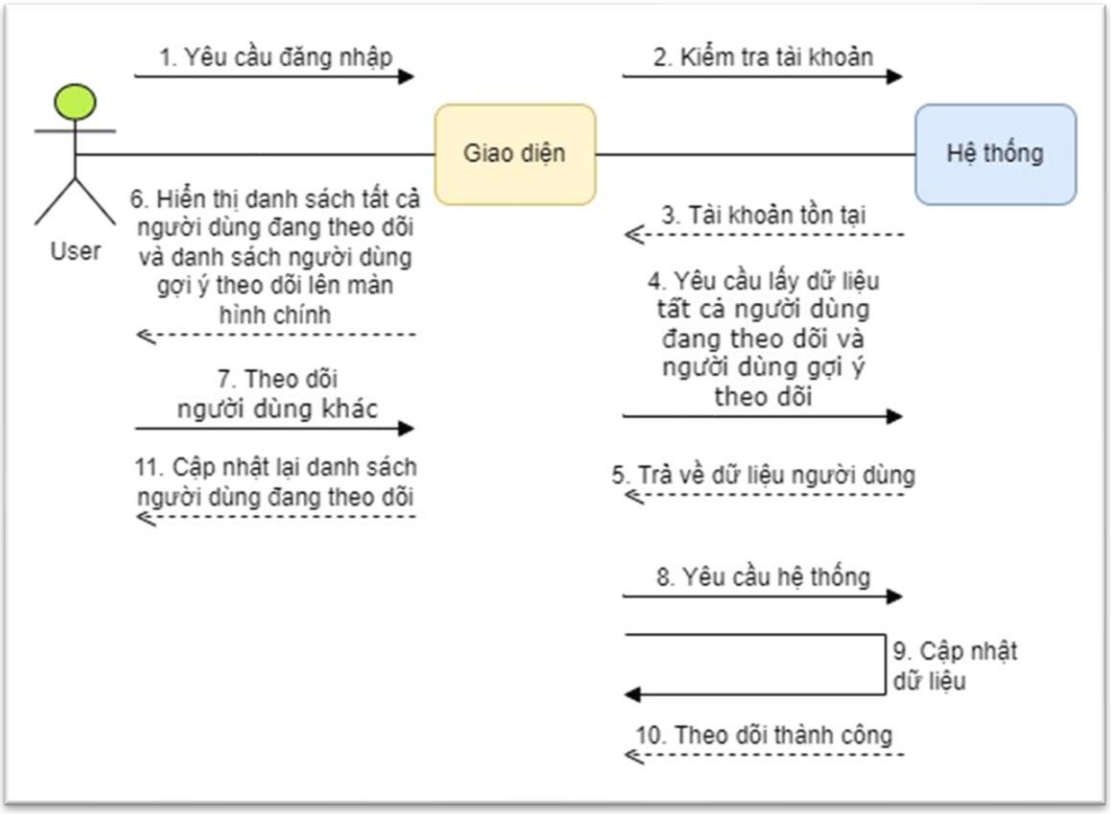
Sơ đồ 3-23 Sơ đồ hợp tác chức năng tìm kiếm của người dùng

3.2.3.6 Mô tả chức năng theo dõi người dùng khác

Đặc tả: Khi người dùng đăng nhập và tài khoản đã được xác minh hiển thị trang màn hình chính. Tại màn hình chính, danh sách đang theo dõi và danh sách người dùng được gợi ý theo dõi sẽ được hiển thị. Hệ thống sẽ cho phép người dùng theo dõi lẫn nhau để tăng tính tương tác và có thể liên lạc với nhau thông qua tin nhắn trực tiếp trên hệ thống.



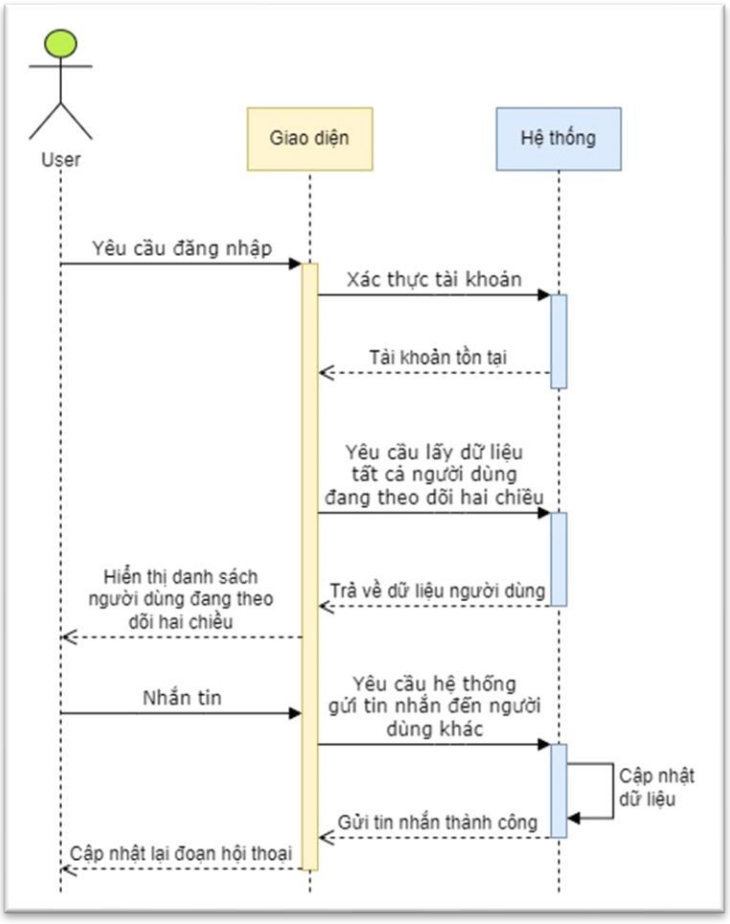
Sơ đồ 3-24 Sơ đồ tuần tự chức năng theo dõi người dùng khác



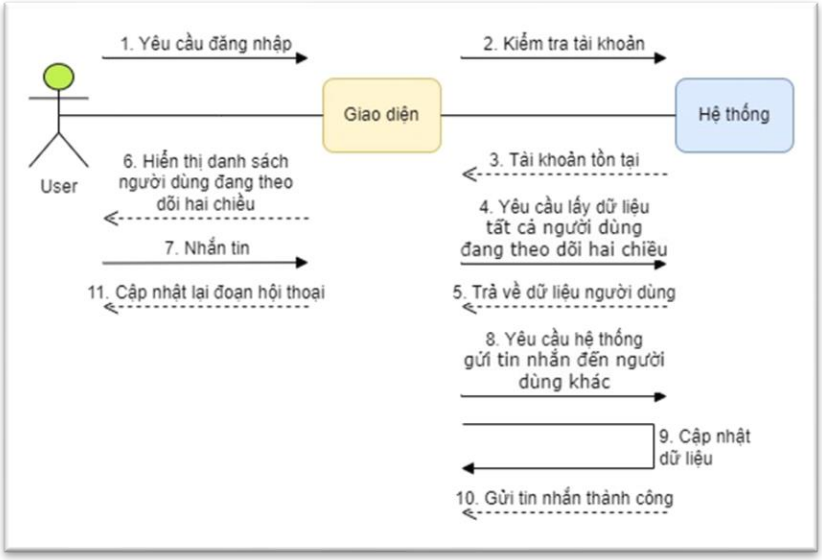
Sơ đồ 3-25 Sơ đồ hợp tác chức năng theo dõi người dùng khác

3.2.3.7 Mô tả chức năng nhắn tin với người dùng khác

Đặc tả: Khi người dùng đăng nhập và tài khoản đã được xác minh hiển thị trang màn hình chính. Hệ thống sẽ cho phép người dùng nhắn tin cho người dùng khác và đồng thời hệ thống cũng cho phép nhận tin nhắn từ người dùng khác.



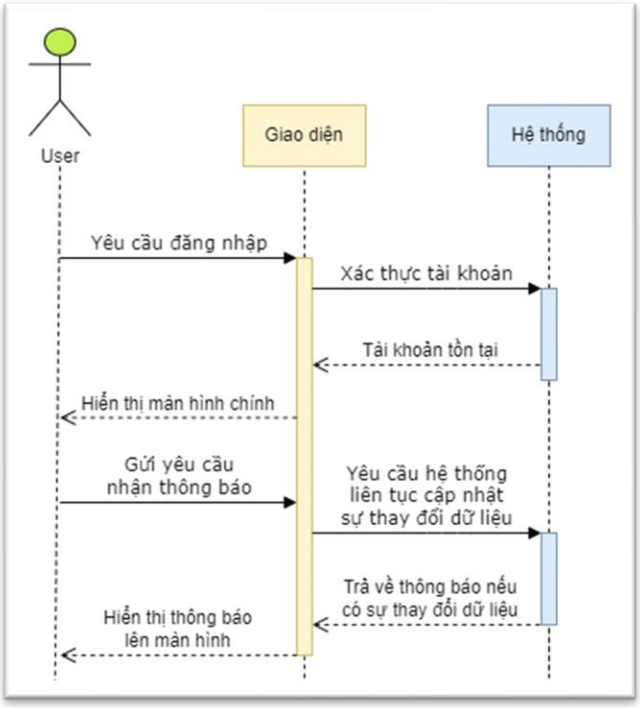
Sơ đồ 3-26 Sơ đồ tuần tự chức năng nhắn tin với người dùng khác



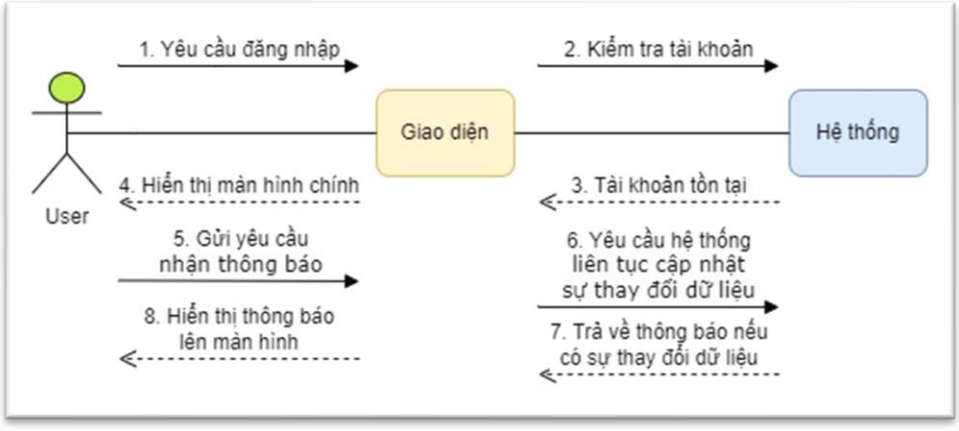
Sơ đồ 3-27 Sơ đồ hợp tác chức năng nhắn tin với người dùng khác

3.2.3.8 Mô tả chức năng nhận thông báo

Đặc tả: Khi người dùng đăng nhập và tài khoản đã được xác minh hiển thị trang màn hình chính. Hệ thống sẽ cho phép người dùng nhận thông báo thời gian thực mỗi khi bài viết của người dùng đó nhận được tương tác, nhóm người dùng đó đang tham gia nhận được tương tác và khi nhận được tin nhắn từ người dùng khác.



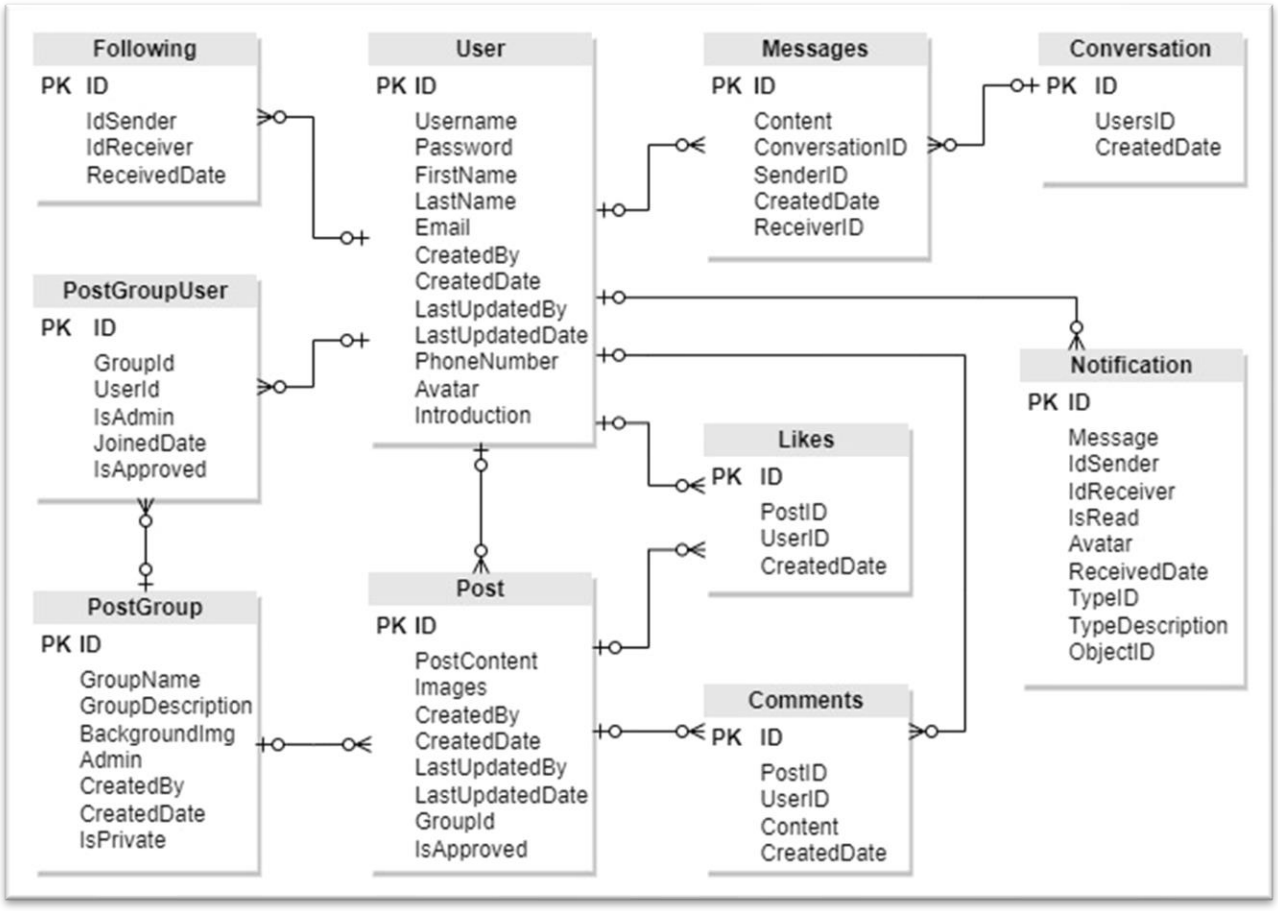
Sơ đồ 3-28 Sơ đồ tuần tự chức năng nhận thông báo



Sơ đồ 3-29 Sơ đồ hợp tác chức năng nhận thông báo

3.3 Thiết kế hệ thống

3.3.1 Sơ đồ cơ sở dữ liệu



Sơ đồ 3-30 Sơ đồ cơ sở dữ liệu

3.3.2 Chi tiết cơ sở dữ liệu

- Quản lý người dùng

Bảng 3-9 Cơ sở dữ liệu bảng người dùng

STT	Tên bảng	Tên cột	Kiểu	Mục đích
1	User	**ID**	Guid	Mã người dùng
2		Username	String (50)	Tên tài khoản
3		Password	String (50)	Mật khẩu người dùng
4		FirstName	String (100)	Họ người dùng
5		LastName	String (100)	Tên người dùng
6		Email	String (100)	Email người dùng
7		CreatedBy	Guid	Mã người tạo tài khoản
8		CreatedDate	Datetime	Ngày tạo tài khoản
9		LastUpdatedBy	Guid	Mã người cập nhật lần cuối
10		LastUpdatedDate	Datetime	Ngày cập nhật lần cuối
11		PhoneNumber	Char(20)	Số điện thoại người dùng
12		Avatar	String (1000)	Hình đại diện
13		Introduction	String(MAX)	Giới thiệu

- Quản lý theo dõi

Bảng 3-10 Cơ sở dữ liệu bảng theo dõi

STT	Tên bảng	Tên cột	Kiểu	Mục đích
1	Following	**ID**	Guid	Mã theo dõi
2		IdSender	Guid	Mã người theo dõi
3		IdReceiver	Guid	Mã người nhận theo dõi
4		ReceivedDate	Datetime	Ngày theo dõi

- Quản lý thông báo

Bảng 3-11 Cơ sở dữ liệu bảng thông báo

STT	Tên bảng	Tên cột	Kiểu	Mục đích
1	Notification	**ID**	Guid	Mã thông báo
2		Message	String(Max)	Nội dung thông báo
3		IdSender	Guid	Người gửi
4		IdReceiver	Guid	Người nhận
5		IsRead	Boolean	Trạng thái đọc
6		Avatar	String(Max)	Ảnh đại diện người gửi
7		ReceivedDate	Datetime	Ngày gửi
8		TypeID	Int	Mã loại thông báo
9		TypeDescription	String(50)	Mô tả loại thông báo
10		ObjectID	Guid	Mã đối tượng thông báo

- Quản lý đoạn hội thoại

Bảng 3-12 Cơ sở dữ liệu bảng hội thoại

STT	Tên bảng	Tên cột	Kiểu	Mục đích
1	Conversation	**ID**	Guid	Mã đoạn hội thoại
2		UsersID	String(Max)	Mã các người dùng tham gia
3		CreatedDate	Datetime	Ngày tạo đoạn hội thoại

- Quản lý tin nhắn

Bảng 3-13 Cơ sở dữ liệu bảng tin nhắn

STT	Tên bảng	Tên cột	Kiểu	Mục đích
1	Messages	**ID**	Guid	Mã tin nhắn
2		Content	String(MAX)	Nội dung tin nhắn
3		ConversationID	Guid	Mã đoạn hội thoại
4		SenderID	Guid	Người gửi
5		ReceiverID	Guid	Người nhận
6		CreatedDate	Datetime	Ngày gửi

- Quản lý bài đăng

Bảng 3-14 Cơ sở dữ liệu bảng bài đăng

STT	Tên bảng	Tên cột	Kiểu	Mục đích
1	Post	**ID**	Guid	Mã bài đăng
2		PostContent	String(MAX)	Nội dung bài đăng
3		Images	String(MAX)	Hình ảnh trong bài đăng
4		CreatedBy	Guid	Người tạo bài đăng
5		CreatedDate	Datetime	Ngày tạo bài đăng
6		LastUpdatedBy	Guid	Người chỉnh sửa lần cuối
7		LastUpdatedDate	Datetime	Ngày chỉnh sửa lần cuối
8		GroupId	Guid	Mã nhóm (Nếu có)
9		IsApproved	Boolean	Trạng thái bài đăng

- Quản lý lượt thích

Bảng 3-15 Cơ sở dữ liệu bảng lượt thích

STT	Tên bảng	Tên cột	Kiểu	Mục đích
1	Likes	**ID**	Guid	Mã lượt thích
2		PostID	Guid	Mã bài đăng
3		UserID	Guid	Mã người thích
4		CreatedDate	Datetime	Ngày thích

- Quản lý lượt bình luận

Bảng 3-16 Cơ sở dữ liệu bảng bình luận

STT	Tên bảng	Tên cột	Kiểu	Mục đích
1	Comments	**ID**	Guid	Mã lượt bình luận
2		PostID	Guid	Mã bài đăng
3		UserID	Guid	Mã người bình luận
4		CreatedDate	Datetime	Ngày bình luận
5		Content	String(MAX)	Nội dung bình luận

- Quản lý nhóm bài đăng

Bảng 3-17 Cơ sở dữ liệu bảng nhóm bài đăng

STT	Tên bảng	Tên cột	Kiểu	Mục đích
1	PostGroup	**ID**	Guid	Mã nhóm
2		GroupName	String(100)	Tên nhóm
3		GroupDescription	String(MAX)	Mô tả nhóm
4		BackgroundImg	String(MAX)	Ảnh nền của nhóm
5		Admin	Guid	Quản trị viên của nhóm
6		CreatedBy	Guid	Mã người tạo nhóm
7		CreatedDate	Datetime	Ngày tạo nhóm
8		IsPrivate	Boolean	Trạng thái nhóm

- Quản lý thành viên nhóm

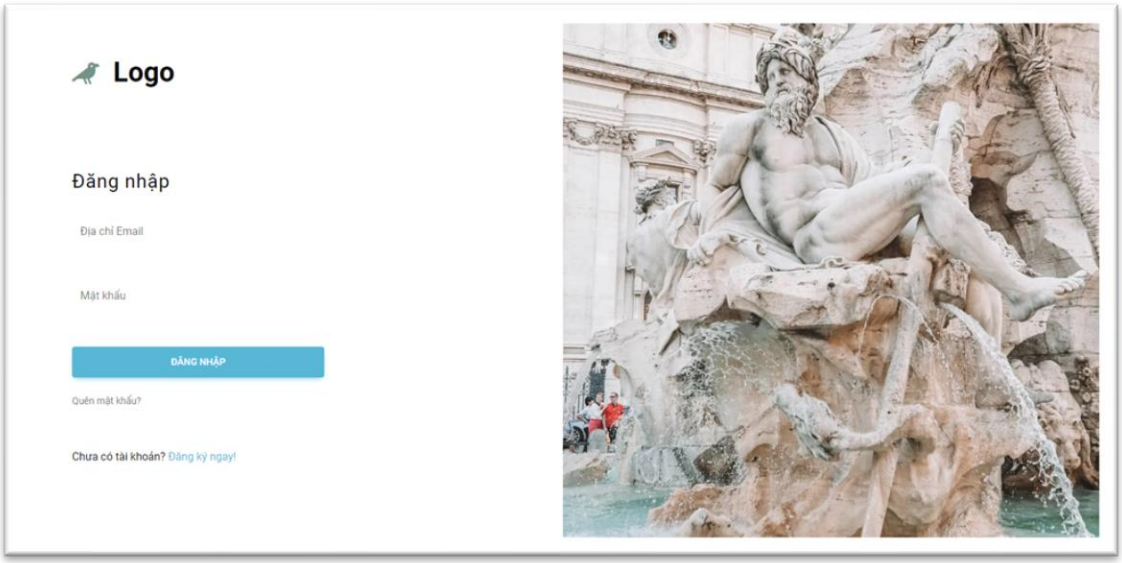
Bảng 3-18 Cơ sở dữ liệu bảng thành viên nhóm

STT	Tên bảng	Tên cột	Kiểu	Mục đích
1	PostGroupUser	**ID**	Guid	Mã
2		GroupId	Guid	Mã nhóm
3		UserId	Guid	Mã người dùng
4		IsAdmin	Boolean	Chức vụ của thành viên trong nhóm
5		JoinedDate	Datetime	Ngày tham gia
6		IsApproved	Boolean	Trạng thái của người dùng trong nhóm

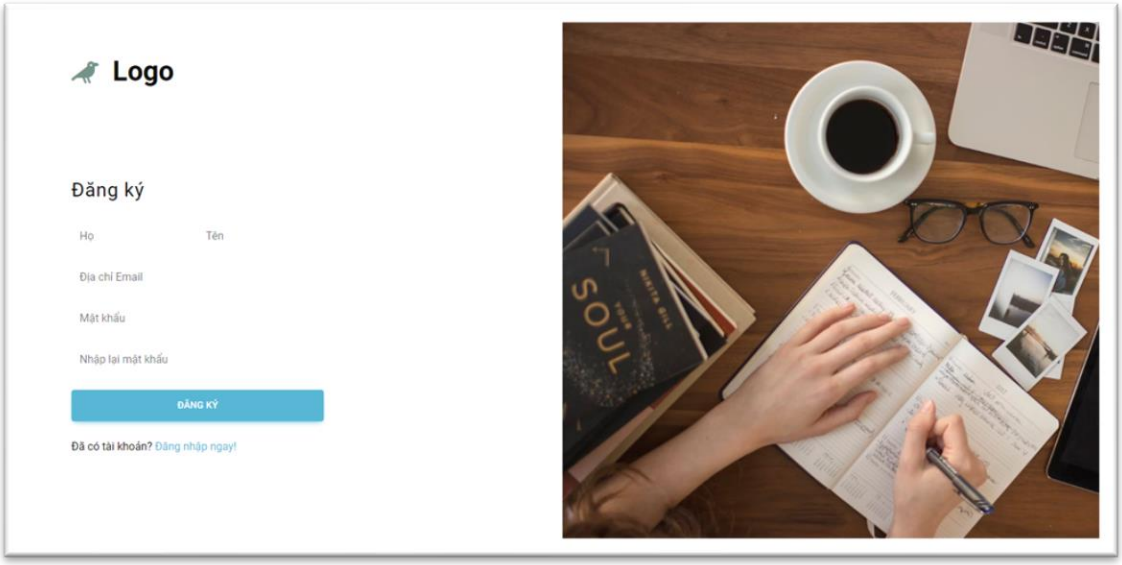
CHƯƠNG 4 XÂY DỰNG WEBSITE

4.1 Giao diện đăng nhập và đăng ký.

Khi người dùng truy cập trang web, hiển thị đầu tiên là trang đăng nhập. Nếu chưa có tài khoản, người dùng có thể điều hướng đến trang đăng ký để tạo tài khoản mới.



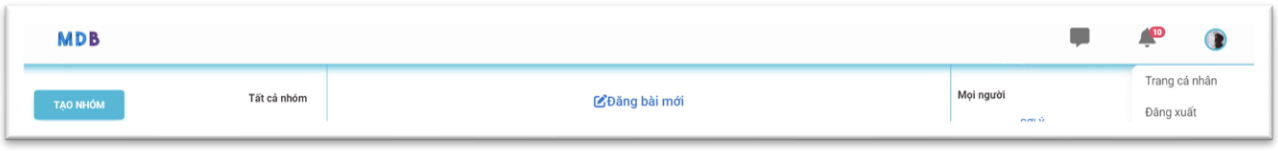
Hình 4.1 Trang đăng nhập



Hình 4.2 Trang đăng ký

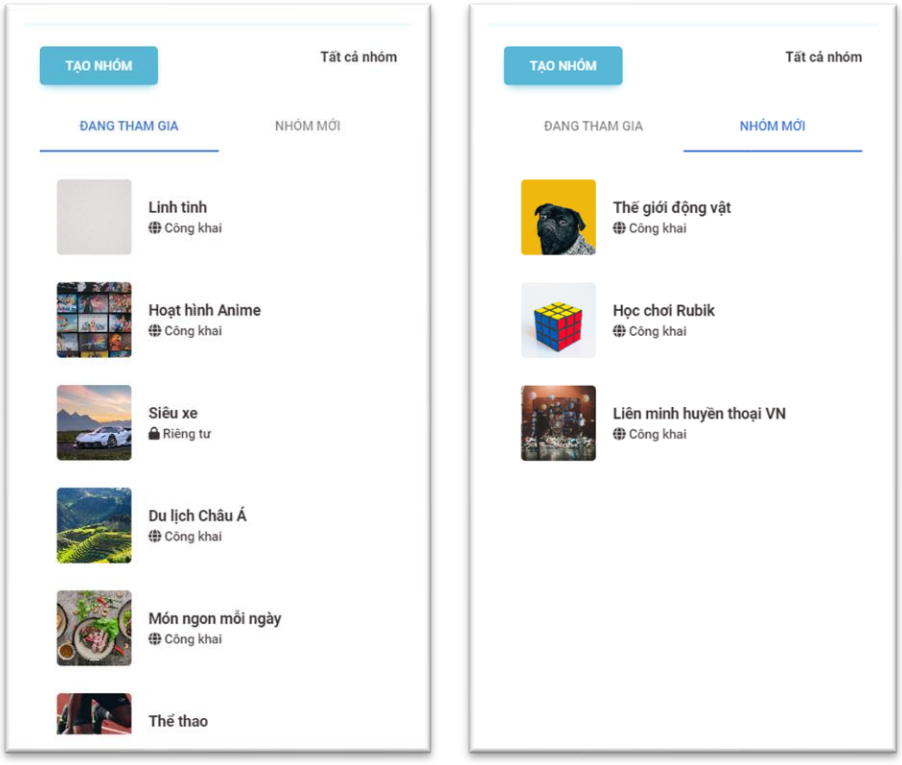
4.2 Giao diện và chức năng.

Khi khách hàng truy cập trang web, hiển thị đầu tiên là trang chủ. Giao diện trang chủ chia làm các khối riêng biệt. Thanh điều hướng sẽ xuất hiện ở tất cả các trang giao diện người dùng.



Hình 4.3 Thanh điều hướng

Thanh điều hướng hình 4.3 bao gồm phần header chứa logo của trang web, nút điều hướng đến trang tin nhắn, nút thông báo để mở cửa sổ các thông báo đã nhận, nút có hình đại diện của người dùng để mở ra 2 nút chức năng khác là điều hướng đến trang cá nhân, đăng xuất.

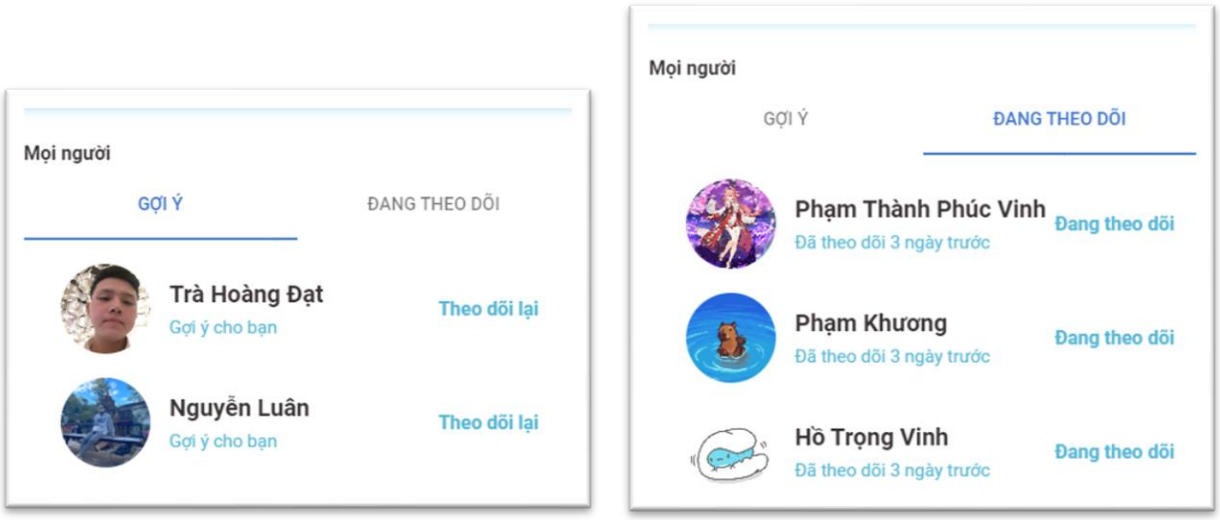


Hình 4.4 Phần nhóm

Phần nhóm hình 4.4 chia làm 2 thẻ, đầu tiên khi người dùng tải xong trang chủ, hệ thống sẽ hiển thị các nhóm mà người dùng đang tham gia (thẻ 1). Thẻ 2 sẽ hiển thị danh sách tất cả các nhóm trên hệ thống (giới hạn 6 nhóm có ngày tạo gần nhất).

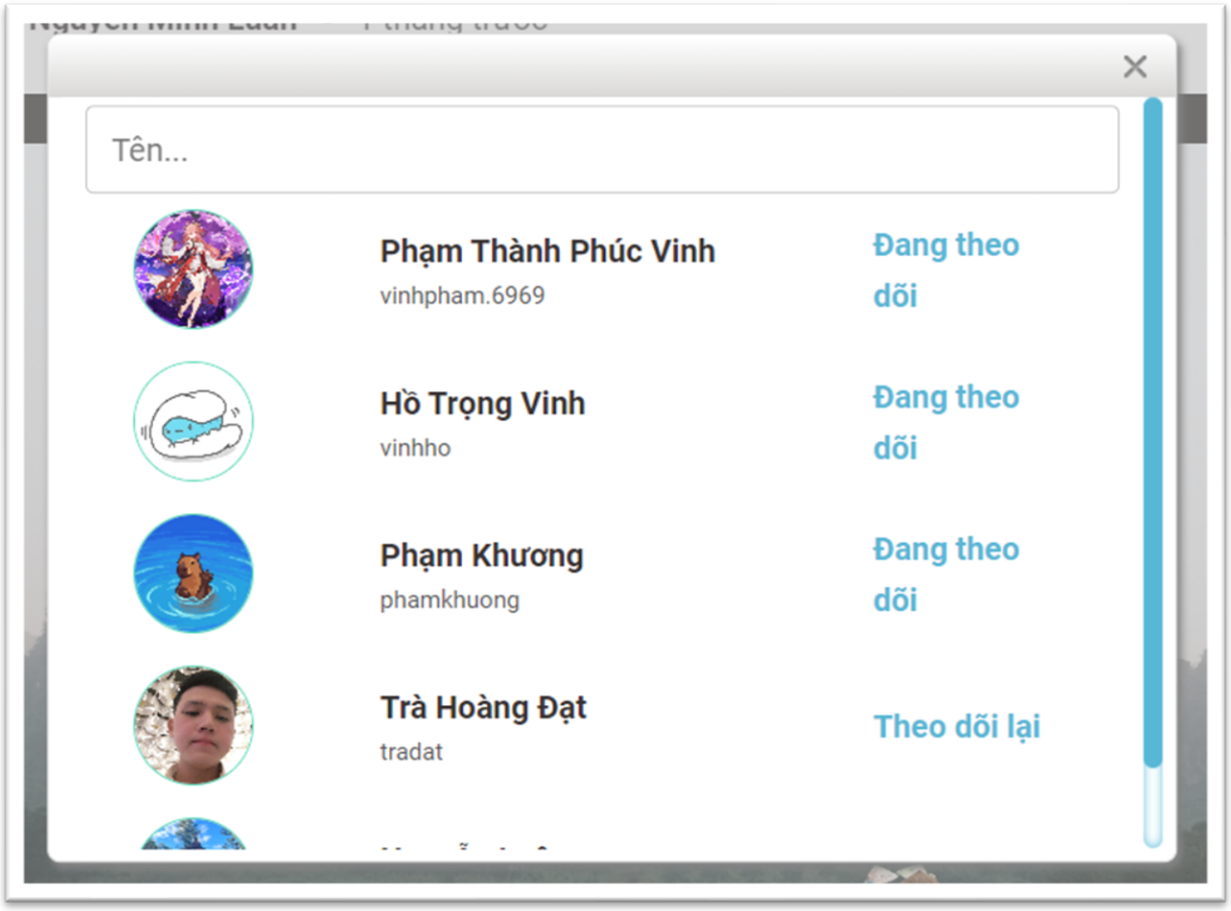
Hình 4.5 Phần Timeline của các bài đăng

Trên cùng của phần Timeline là nút tạo bài viết mới. Phần Timeline của các bài đăng hình 4.5 sẽ hiển thị 5 bài đăng mới nhất, để xem thêm thì người dùng nhấn vào nút xem thêm ở cuối trang.



Hình 4.6 Phần đề xuất theo dõi

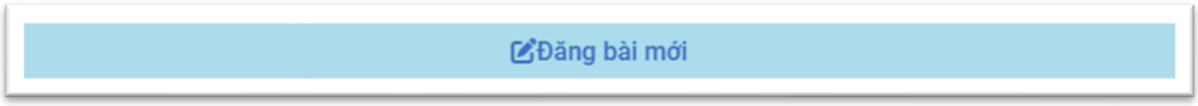
Khối tiếp theo hình 4.6 đầu tiên khi người dùng tải xong trang chủ, hệ thống sẽ hiển thị các người dùng được gợi ý (thẻ 1) (giới hạn 9 người dùng có ngày tạo gần nhất). Thẻ 2 sẽ hiển thị danh sách tất cả các người dùng đang theo dõi trên hệ thống (giới hạn 9 người dùng có ngày tạo gần nhất). Nút theo dõi sẽ hiển thị theo 3 trạng thái. Nếu được người dùng khác theo dõi nhưng chưa theo dõi lại thì trạng thái của nút sẽ là “Theo dõi lại”, nếu 2 người dùng khác nhau ở trạng thái theo dõi 2 chiều thì trạng thái của nút sẽ là “Đang theo dõi”, nếu chưa theo dõi giữa các người dùng thì trạng thái của nút sẽ là “Theo dõi”. Người dùng cũng có thể mở danh sách tất cả các người dùng khác trong hệ thống bằng cách nhấn vào nút “Mọi người”.



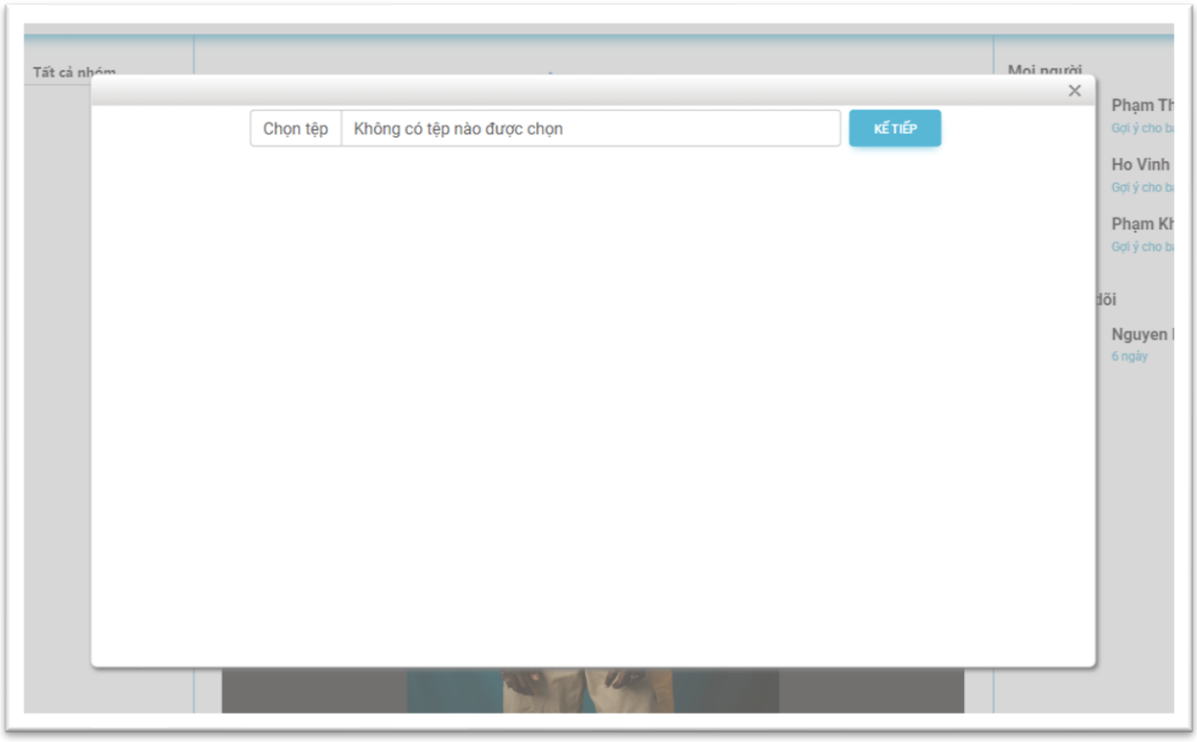
Hình 4.7 Giao diện xem tất cả người dùng khác trên hệ thống

Hình 4.7 là giao diện xem tất cả người dùng khác trên hệ thống, tại đây người dùng có thể thao tác tìm kiếm theo tên bằng cách nhập vào ô trống ở trên cùng danh sách. Ngoài ra người dùng cũng có thể thao tác theo dõi đến người dùng khác tương tự như ở trang chính đã mô tả.

Tiếp theo khi người dùng chọn đăng bài mới, hệ thống sẽ mở ra một cửa sổ để người dùng viết bài.

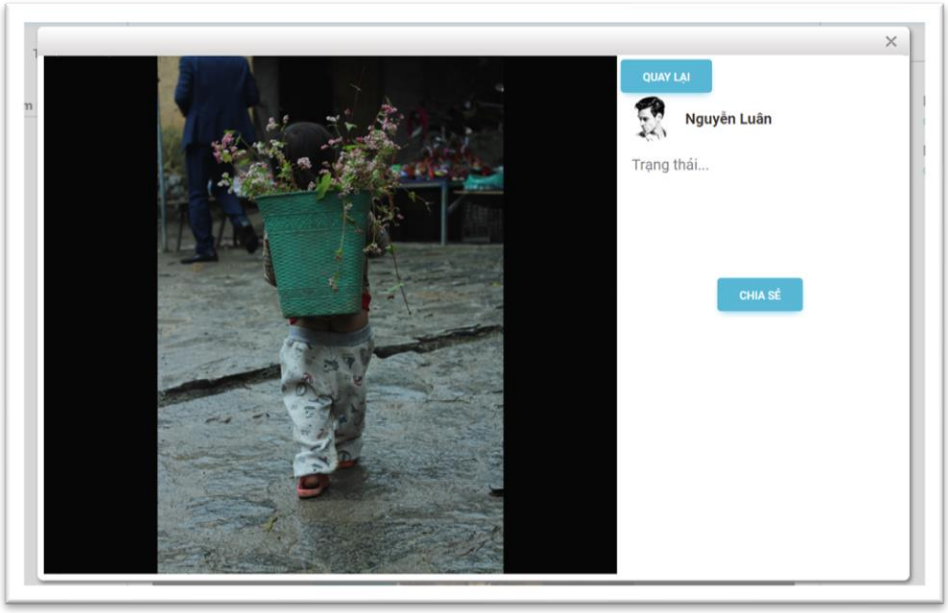


Hình 4.8 Nút đăng bài mới



Hình 4.9 Giao diện chọn hình ảnh cho bài viết - cửa sổ viết bài

Sau khi chọn hình ảnh để đăng tải, nút kết tiếp sẽ đưa người dùng đến bước tiếp theo.



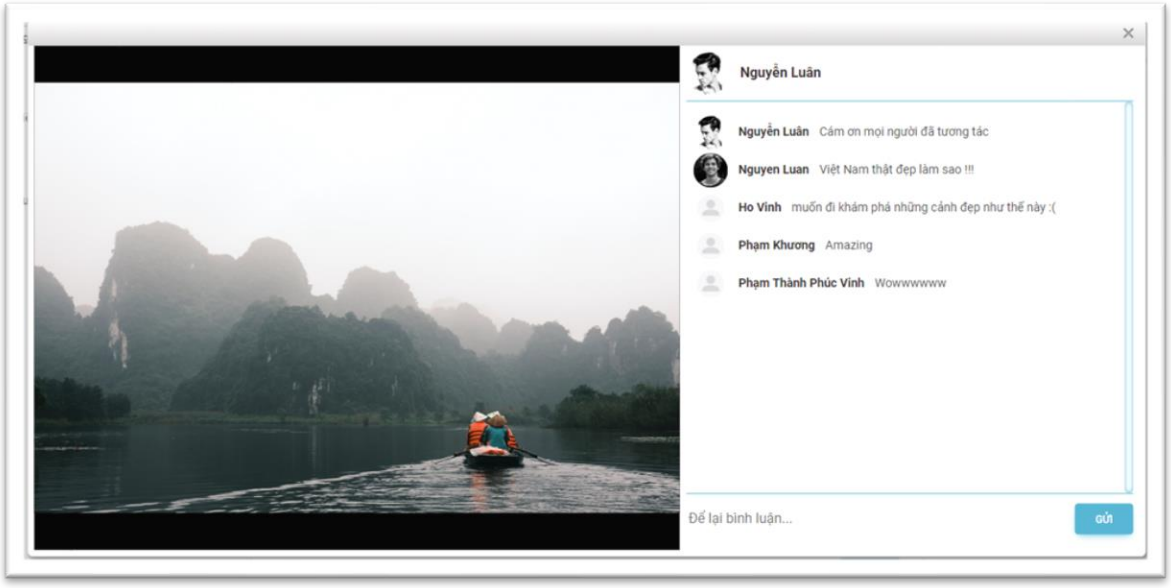
Hình 4.10 Giao diện viết trạng thái - cửa sổ viết bài

Nút quay lại sẽ đưa người dùng quay về bước chọn hình ảnh. Khi người dùng nhấn chia sẻ, hệ thống sẽ đăng tải nội dung bài viết của người vừa nhập.

Tiếp theo, người dùng có thể để lại lượt yêu thích và mở trang bình luận cho các bài đăng hiển thị trên trang chính.

Hình 4.11 Giao diện chức năng tương tác với bài đăng

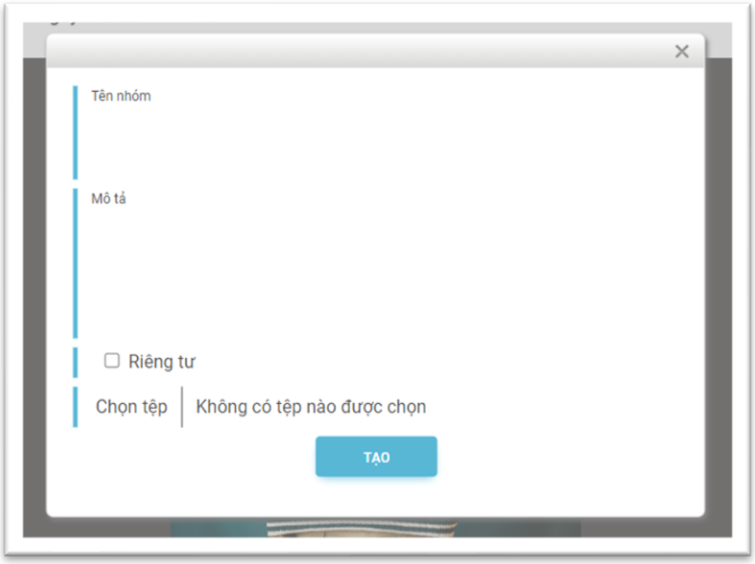
Tại đây, người dùng có thể biết được mình đã thích bài viết hay chưa bằng trạng thái tô màu của nút "trái tim", màu đỏ có nghĩa là người dùng đã thích bài biết này. Người dùng cũng sẽ biết được có bao nhiêu người thích bài viết này. Để xem các bình luận, người dùng chọn vào nút bình luận, hệ thống sẽ mở ra giao diện bình luận chi tiết.



Hình 4.12 Giao diện bình luận chi tiết của bài đăng

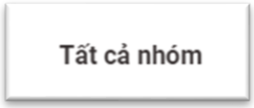
Tiếp theo người dùng chọn tạo nhóm mới, hệ thống sẽ hiển thị cửa sổ tạo nhóm.

Hình 4.13 Nút tạo nhóm mới

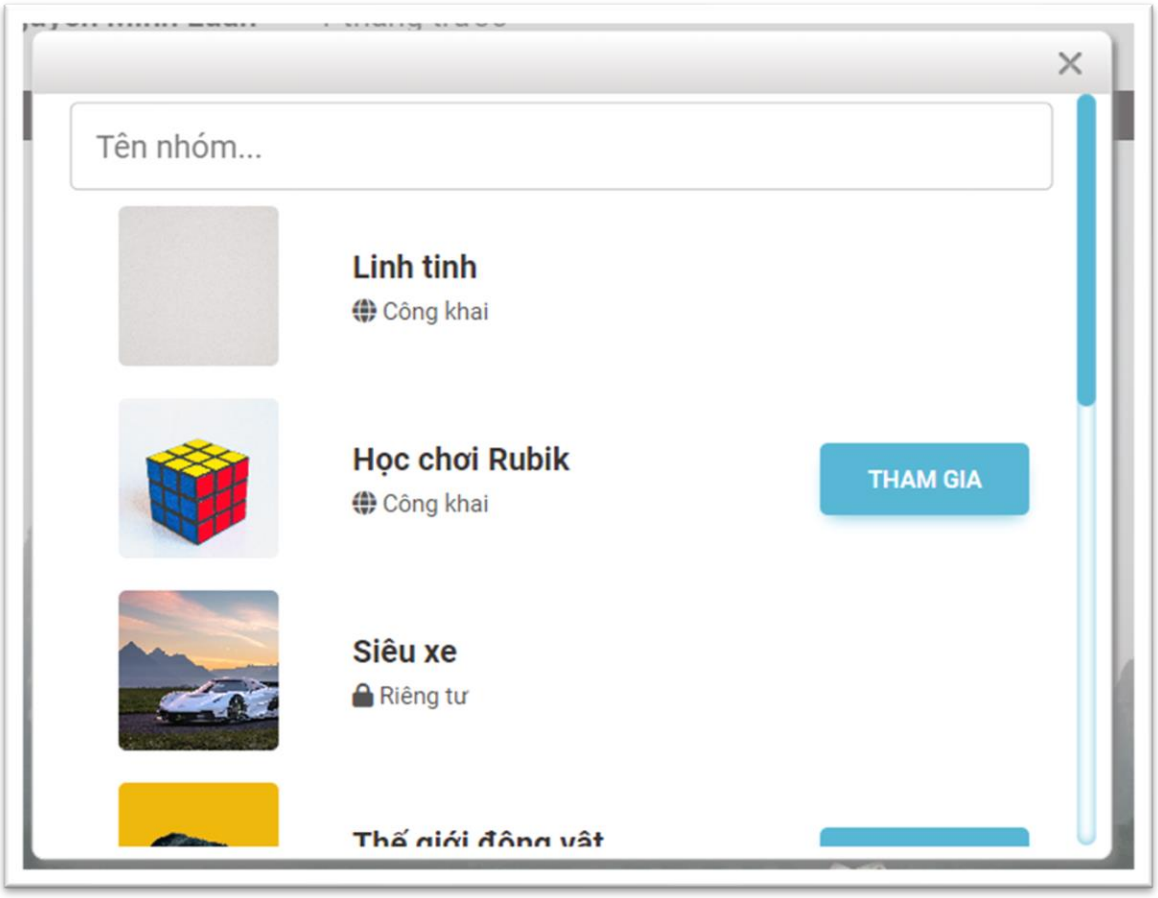


Hình 4.14 Giao diện tạo nhóm

Người dùng có thể xem, tìm kiếm các nhóm trên hệ thống.

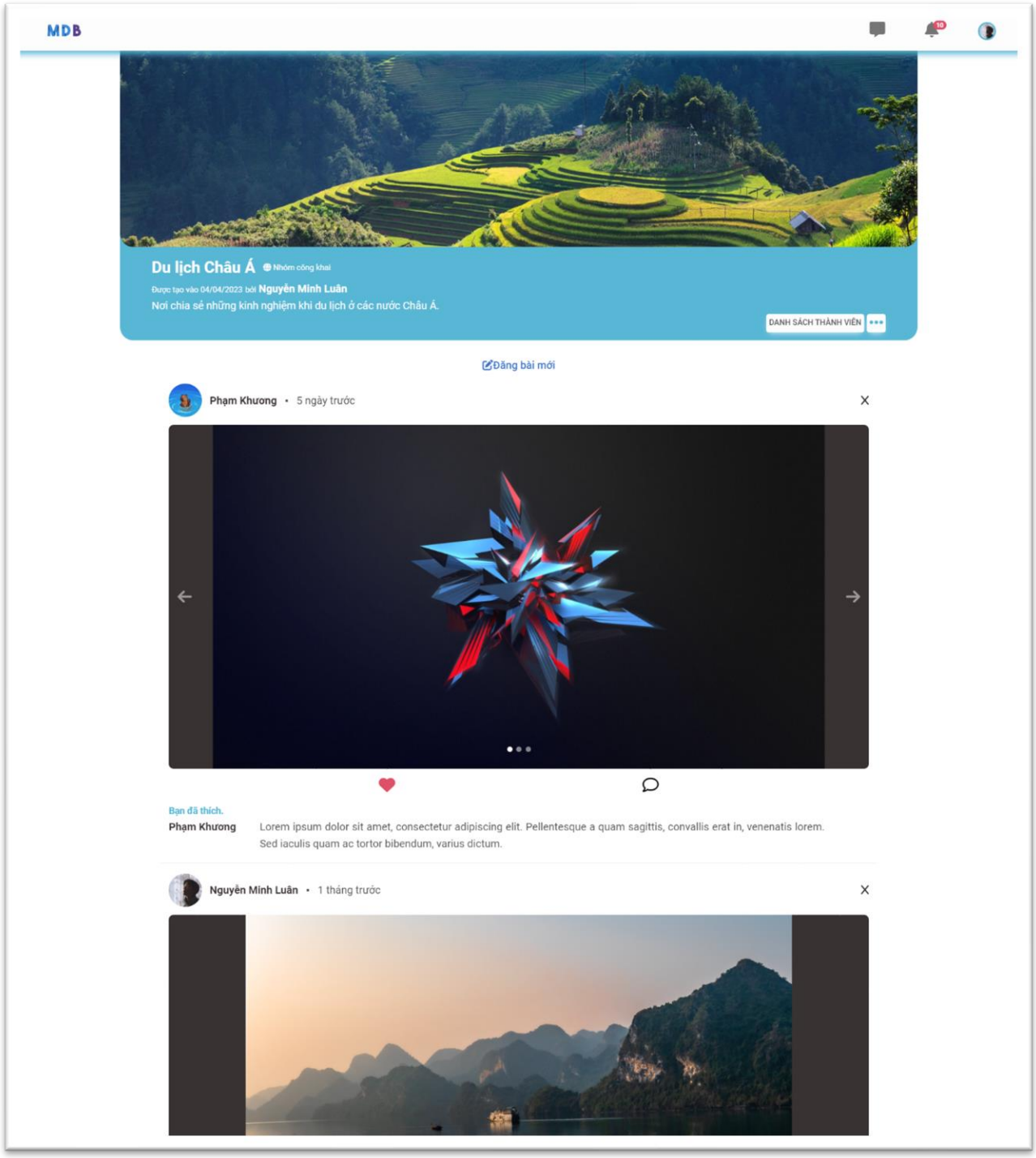


Hình 4.15 Nút xem tất cả nhóm trên hệ thống



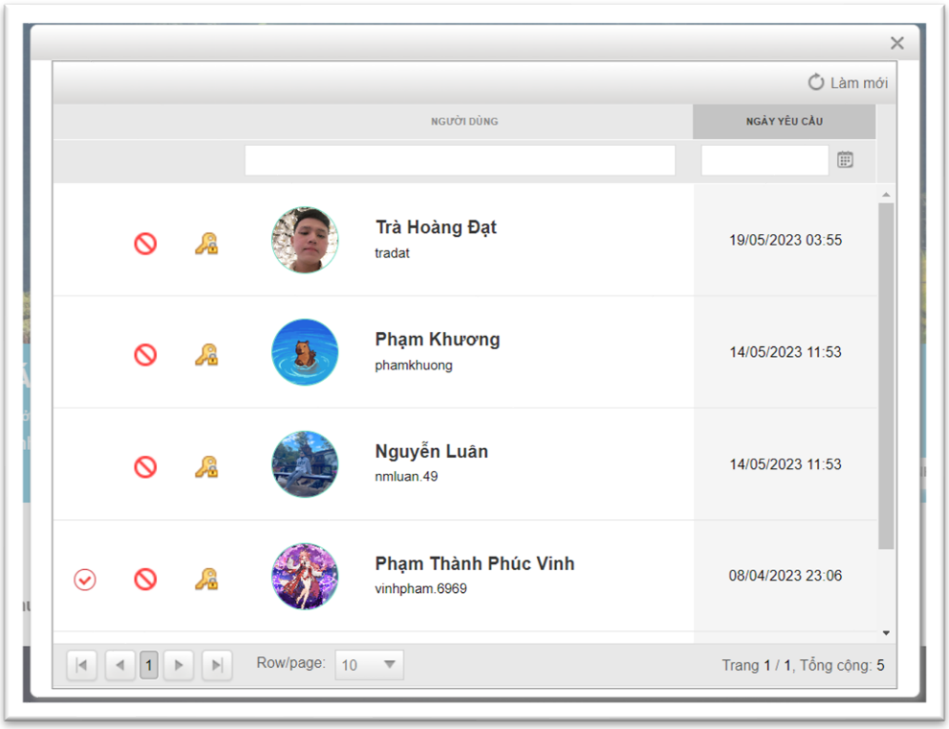
Hình 4.16 Giao diện xem tất cả nhóm trên hệ thống

Người dùng có thể tham gia vào các nhóm khác bằng cách nhấn vào các nút tham gia trong danh sách nhóm (giới hạn 6 nhóm mỗi trang, được sắp xếp theo thứ tự ngày tạo mới nhất), các nhóm người dùng đã tham gia hoặc đang chờ Admin nhóm xác nhận thì sẽ không xuất hiện nút tham gia. Người dùng cũng có thể tìm kiếm theo tên nhóm bằng cách nhập tên nhóm muốn tìm vào ô trống phía trên cùng của danh sách. Để người dùng có thể dễ dàng sử dụng, giao diện nhóm được thiết kế tương tự như phần Timeline ở trang chủ. Chức năng tương tác với các bài đăng, đăng bài cũng được xây dựng tương tự như ở trang chủ.

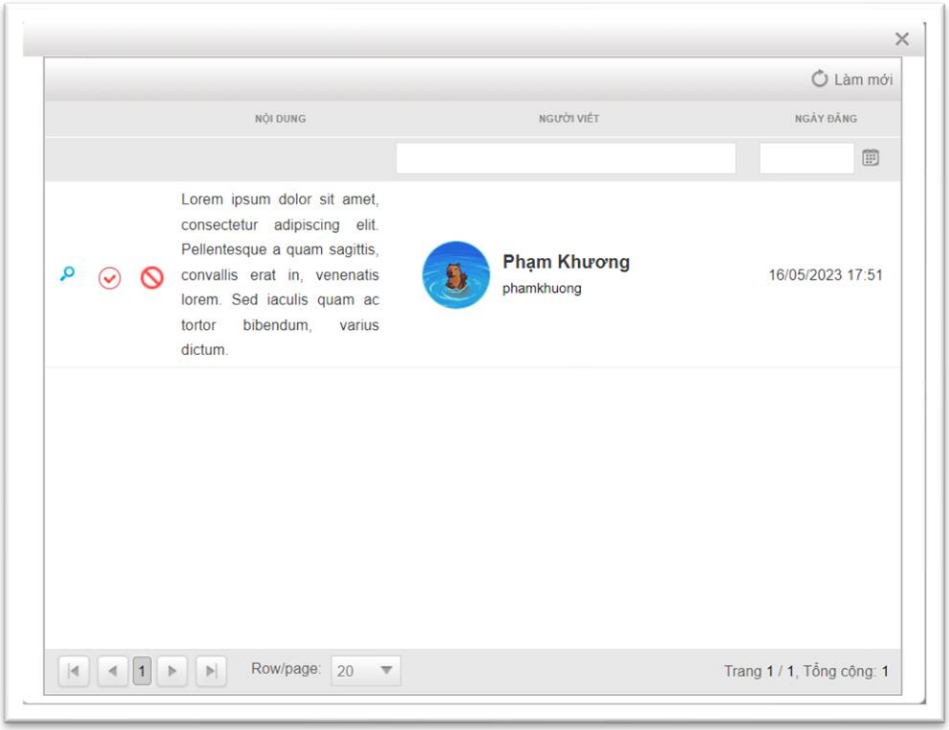


Hình 4.17 Giao diện nhóm bài đăng

Admin của nhóm sẽ có thêm chức năng cập nhật thông tin nhóm, duyệt bài đăng trong nhóm, duyệt thành viên (nếu trạng thái nhóm là "riêng tư").



Hình 4.18 Giao diện quản lý yêu cầu tham gia nhóm

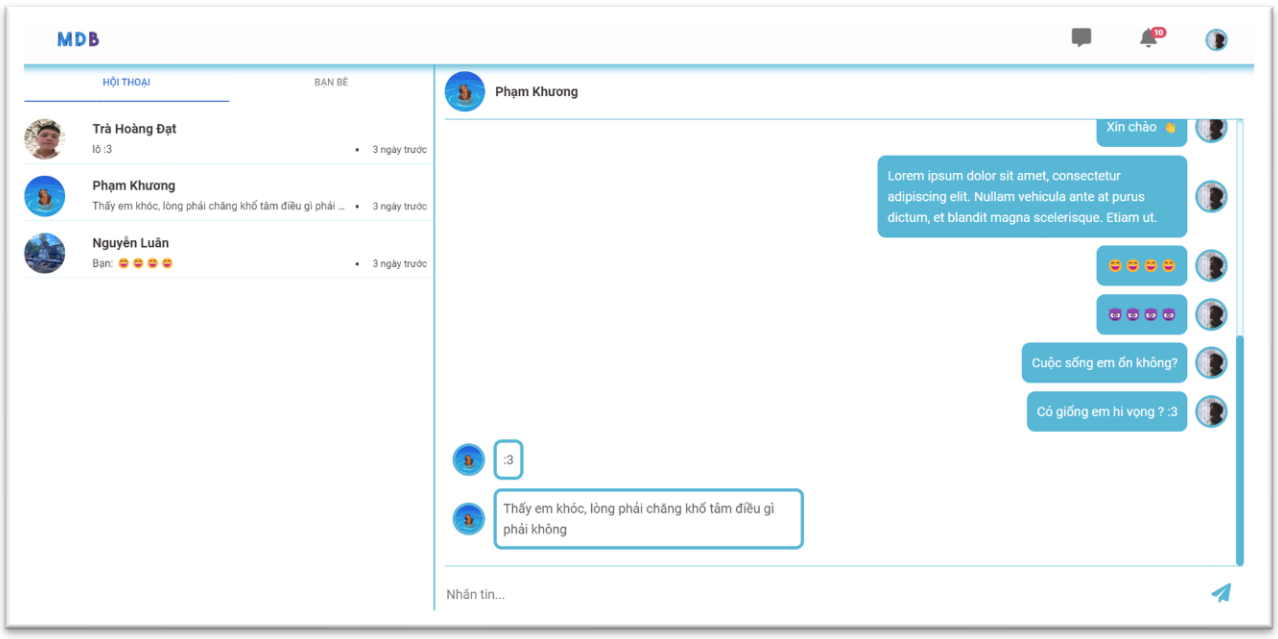


Hình 4.19 Giao diện quản lý duyệt bài đăng trong nhóm

Đối với người dùng là thành viên trong nhóm, hệ thống sẽ hiển thị trạng thái tham gia nhóm của người dùng đó theo 3 cấp độ.

Hình 4.20 Cấp độ trạng thái tham gia nhóm

Tiếp theo, khi giữa 2 người dùng khác nhau có theo dõi 2 chiều, 2 người dùng này có thể nhắn tin cho nhau thông qua giao diện nhắn tin.

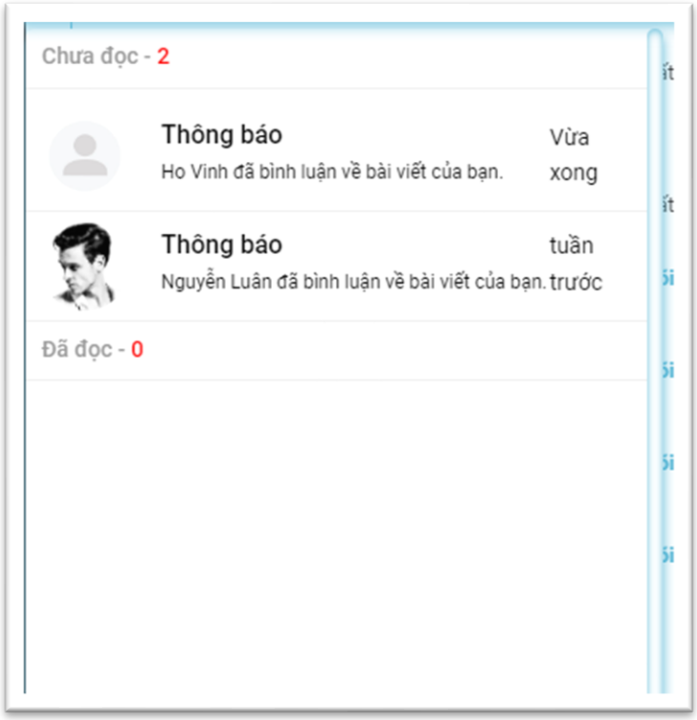


Hình 4.21 Giao diện chức năng nhắn tin

Trong hình 4.21 là giao diện chức năng nhắn tin được chia làm 2 phần, bên trái là danh sách các đoạn hội thoại của người dùng, bên phải là nội dung theo từng đoạn hội thoại tương ứng. Người dùng chọn vào đoạn hội thoại bất kì trong danh sách đoạn hội thoại, hệ thống sẽ tự tải dữ liệu của đoạn hội thoại.

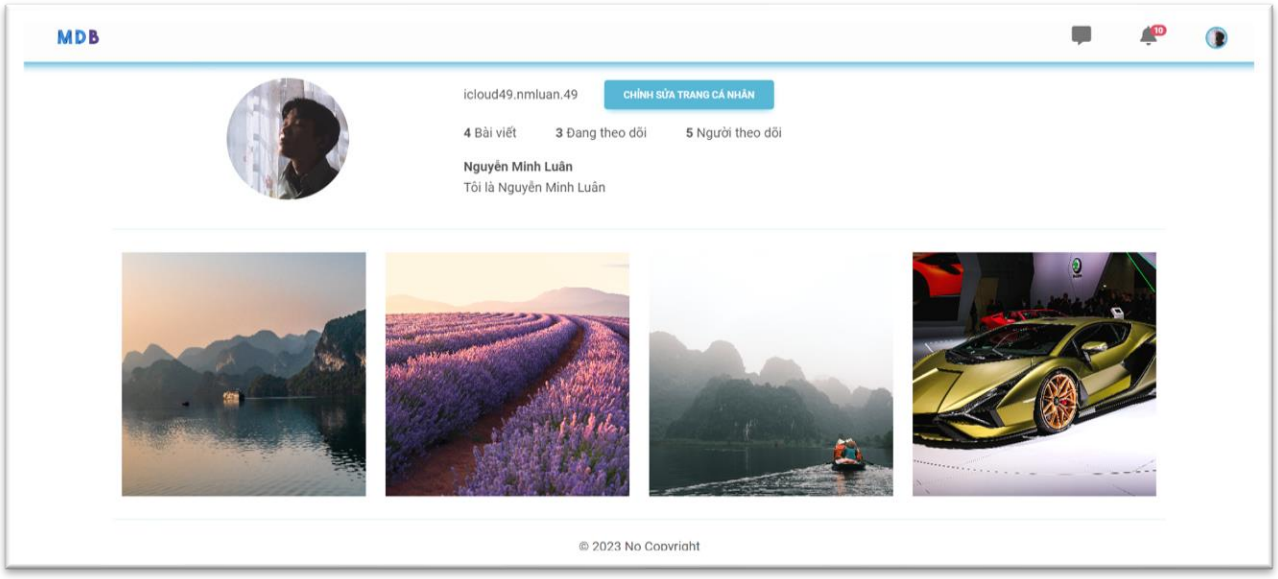
Tiếp theo, người dùng có thể nhận được thông báo thời gian thực khi bài đăng của mình nhận được tương tác, khi người dùng khác tương tác với mình hoặc khi nhận được tin nhắn mới.

Hình 4.22 Giao diện chức năng nhắn tin

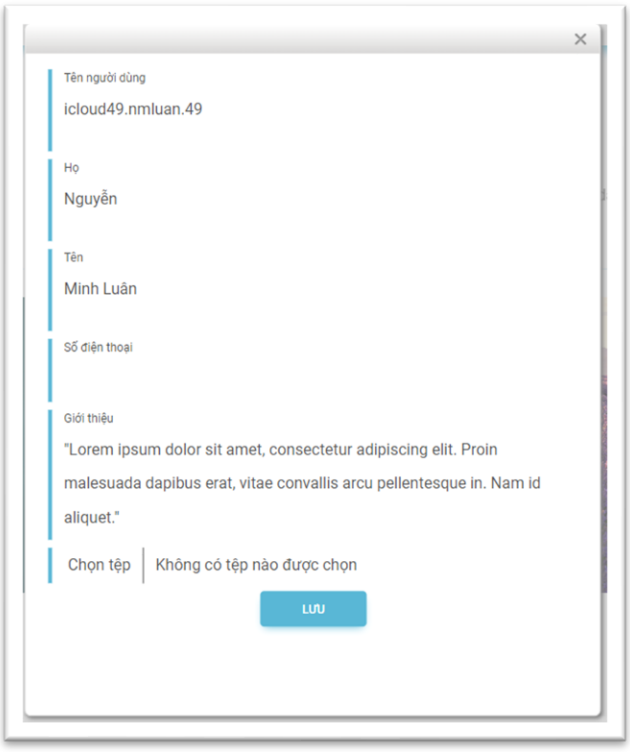


Hình 4.23 Giao diện thông báo

Phần cuối cùng là giao diện trang cá nhân của người dùng, tại đây người dùng có thể sử dụng chức năng cập nhật thông tin tài khoản, xem thống kê sơ bộ về quá trình hoạt động của tài khoản, xem lại tất cả các bài viết mình đã đăng dưới dạng ảnh thumbnail, người dùng cũng có thể xem lại chi tiết bài đăng bằng cách click vào các ảnh thumbnail đó.



Hình 4.24 Giao diện trang cá nhân



Hình 4.25 Giao diện cập nhật thông tin tài khoản

CHƯƠNG 5 KẾT LUẬN

5.1 Kết quả đạt được

- ✓ Vận dụng kiến thức về ngôn ngữ ASP.NET vào dự án thực tế
- ✓ Thực hành và nghiên cứu sâu .NET framework và ngôn ngữ C#
- ✓ Kỹ năng phân tích thiết kế, phân tích giao diện
- ✓ Xây dựng website đơn giản, dễ dùng, vẫn đảm bảo đầy đủ chức năng đề ra
- ✓ Xây dựng và chuẩn hóa dữ liệu đảm bảo tính đúng đắn
- ✓ Giao diện dễ nhìn, rõ ràng dễ sử dụng

5.2 Hướng phát triển

- ➢ Xây dựng hệ thống bảo mật, đảm bảo an toàn thông tin cho người sử dụng
- ➢ Nâng cấp giao diện thân thiện, dễ sử dụng cho người dùng
- ➢ Phát triển thêm trang quản lý cho Admin
- ➢ Mở rộng cơ sở dữ liệu, quản lý chi tiết hơn nhằm đáp ứng nhu cầu của người dùng
- ➢ Nghiên cứu công nghệ AI để ứng dụng vào quản lý, làm sạch các nguồn thông tin trên hệ thống

Do thời gian thực hiện còn hạn hẹp, kiến thức và kinh nghiệp còn nhiều hạn chế nên bài báo cáo này không tránh khỏi những sai sót. Rất mong nhận được những đóng góp ý kiến quý báu của thầy cô trường Đại học Bà Rịa – Vũng Tàu.

Một lần nữa em xin chân thành cảm ơn quý thầy cô trong khoa Kỹ thuật – Công nghệ và đặc biệt em xin gửi đến thầy Phan Ngọc Hoàng, người đã tận tình hướng dẫn, giúp đỡ em hoàn thành đề tài đồ án tốt nghiệp lời cảm ơn sâu sắc nhất.

TÀI LIỆU THAM KHẢO

[1] WebForms ListView Overview. 13/04/2023 Lấy từ URL:

https://docs.telerik.com/devtools/aspnet-ajax/controls/listview/overview

[2] Telerik Web Forms RadWindow Object. 13/04/2023 Lấy từ: URL:

https://docs.telerik.com/devtools/aspnet-ajax/controls/window/client-side-programming/radwindow-object

[3] ASP.NET là gì? Tìm hiểu về ngôn ngữ lập trình ASP.NET. 06/05/2023 Lấy từ: URL:

https://vietnix.vn/aspnet-la-gi/

[4] SignalR - Simple Chat Application in C#. 26/04/2023 Lấy từ URL:

https://www.c-sharpcorner.com/UploadFile/78d182/signalr-sample-chat-application-in-C-Sharp/

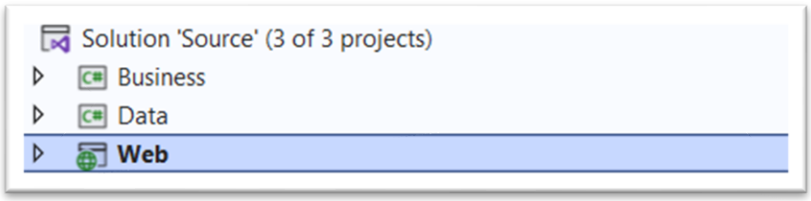
[5] SQL Server là gì? Hướng dẫn cài đặt SQL Server. 09/05/2023 Lấy từ: URL:

https://vietnix.vn/sql-server-la-gi/

[6] Các thảo luận trên cộng đồng Stack Over Flow

PHỤ LỤC

6.1 Sơ đồ cấu trúc



Hình phụ lục 1 Cấu trúc thư mục tổng

Bảng phụ lục 1 Bảng mô tả cấu trúc thư mục tổng

STT	Tên thư mục	Ý nghĩa
1	Data	Project chứa file data ADO.NET, các file Data Access để liên kết với các lớp dữ liệu từ SQL Server
2	Business	Project chứa các file dịch vụ web để tương tác với dữ liệu trong database
3	Web	Project chính chứa các file webform aspx và các file code xử lý

6.2 Code xử lý

6.2.1 File Site.Master

```
using Business.Services.Data;
using Business.Services.Security;
using Data.Entities;
using Microsoft.Ajax.Utilities;
using Microsoft.AspNet.SignalR;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.Security;
using System.Web.UI;
using System.Web.UI.WebControls;
using Web.Sessions;
using Web.SignalRHub;
```

```
namespace Web
{
   public partial class SiteMaster : MasterPage
   {
      private const string AntiXsrfTokenKey = "__AntiXsrfToken";
      private const string AntiXsrfUserNameKey = "__AntiXsrfUserName";
      private string _antiXsrfTokenValue;
      private readonly NotificationService notificationService = new NotificationService();
      private readonly UserService userService = new UserService();
      string message;

      protected void Page_Load(object sender, EventArgs e)
      {
         if (!IsPostBack)
         {
            if (UserSession.Current.IsAvailable)
            {
               this.userAvatar.Src = UserSession.Current.User.Avatar;
               showNotification(UserSession.Current.User);
            }
         }

      }

      public void showNotification(User user)
      {

         List<Notification> notifications = notificationService.GetAllByReceiverID(user.ID).OrderBy(t =>
t.ReceivedDate).ToList();
         int number_true = notifications.Count - notificationService.CountUnRead(user.ID);
         string no_true = "<h2> Đã đọc - <span id='valueCount'> " + number_true + " </span></h2>";
         string no_false = "<h2> Chưa đọc - <span id='valueCount'> " +
notificationService.CountUnRead(user.ID) + " </span></h2>";
         for (int i = notifications.Count - 1; i >= 0; i--)
         {
            Notification notification = notifications[i];
            if (notification.IsRead == true)
            {
               no_true += "<div class='notifications-item' onclick=\"customAlert('" + notification.ID + "','" +
notification.Message + "','Thông Báo!','" +
userService.GetByID(notification.IdSender.GetValueOrDefault()).FirstName + " " +
userService.GetByID(notification.IdSender.GetValueOrDefault()).LastName + "','" +
userService.GetByID(notification.IdReceiver.GetValueOrDefault()).FirstName + " " +
userService.GetByID(notification.IdReceiver.GetValueOrDefault()).LastName + "')\"> <img src ='" +
userService.GetByID(notification.IdSender.GetValueOrDefault()).Avatar + "' alt='img'/><div
class='text'><h4>Thông báo</h4><p>" + notification.Message + "</p></div><div class='time'><p>" +
TimeAgo(notification.ReceivedDate.GetValueOrDefault()) +  "</p></div></div>";
            }
            else
```

```
            {
                no_false += "<div class='notifications-item' onclick=\"customAlert('" + notification.ID + "','" +
notification.Message + "','Thông Báo!','" +
userService.GetByID(notification.IdSender.GetValueOrDefault()).FirstName + " " +
userService.GetByID(notification.IdSender.GetValueOrDefault()).LastName + "','" +
userService.GetByID(notification.IdReceiver.GetValueOrDefault()).FirstName + " " +
userService.GetByID(notification.IdReceiver.GetValueOrDefault()).LastName + "')\"> <img src ='" +
userService.GetByID(notification.IdSender.GetValueOrDefault()).Avatar + "' alt='img'/><div
class='text'><h4>Thông báo</h4><p>" + notification.Message + "</p></div><div class='time'><p>" +
TimeAgo(notification.ReceivedDate.GetValueOrDefault()) +  "</p></div></div>";
            }
        }
        message = no_false + no_true;
        if (box != null)
        {
            box.InnerHtml = no_false + no_true;
            number.InnerHtml = "" + notificationService.CountUnRead(UserSession.Current.User.ID) + "";
        }
    }

    public void getNotification(User user, Notification notification)
    {
        showNotification(user);
        string count = "" + notificationService.CountUnRead(user.ID) + "";
        SignalRHub.MySignalRHub.Message(count, message, user.Username, notification.ID.ToString(),
notification.Message, userService.GetByID(notification.IdSender.GetValueOrDefault()).Avatar,
userService.GetByID(notification.IdSender.GetValueOrDefault()).FirstName + " " +
userService.GetByID(notification.IdSender.GetValueOrDefault()).LastName,
userService.GetByID(notification.IdReceiver.GetValueOrDefault()).FirstName + " " +
userService.GetByID(notification.IdReceiver.GetValueOrDefault()).LastName);
    }

    protected void btnUserPage_Click(object sender, EventArgs e)
    {
        Response.Redirect("~/UserPage.aspx");
    }

    protected void Page_Init(object sender, EventArgs e)
    {
        if (!UserSession.Current.IsAvailable)
        {
            if (Request.RawUrl != "/Controls/Security/Login.aspx")
            {
                Session.Add("ReturnURL", Request.RawUrl);
            }

            Response.Redirect("~/Controls/Security/Login.aspx");
        }

        Page.PreLoad += master_Page_PreLoad;
```

```
    }

    protected void master_Page_PreLoad(object sender, EventArgs e)
    {
        if (!IsPostBack)
        {
            // Set Anti-XSRF token
            ViewState[AntiXsrfTokenKey] = Page.ViewStateUserKey;
            ViewState[AntiXsrfUserNameKey] = Context.User.Identity.Name ?? string.Empty;
        }
        else
        {
            // Validate the Anti-XSRF token
            if ((string)ViewState[AntiXsrfTokenKey] != _antiXsrfTokenValue
                || (string)ViewState[AntiXsrfUserNameKey] != (Context.User.Identity.Name ?? string.Empty))
            {
                throw new InvalidOperationException("Validation of Anti-XSRF token failed.");
            }
        }
    }

    protected void LogOut_Click(object sender, EventArgs e)
    {
        //UserLogOut(UserSession.Current.LogId);
        Session.Clear();
        Session.Abandon();
        FormsAuthentication.SignOut();

        UserSession.DestroySession();
        Session.Remove("ReturnURL");
        Response.Redirect("~/Controls/Security/Login.aspx");
    }

    public string TimeAgo(DateTime dateTime)
    {
        string result = string.Empty;
        var timeSpan = DateTime.Now.Subtract(dateTime);

        if (timeSpan <= TimeSpan.FromSeconds(60))
        {
            result = string.Format("Vừa xong", timeSpan.Seconds);
        }
        else if (timeSpan <= TimeSpan.FromMinutes(60))
        {
            result = timeSpan.Minutes > 1 ?
                String.Format("{0} phút", timeSpan.Minutes) :
                "1 phút";
        }
        else if (timeSpan <= TimeSpan.FromHours(24))
        {
```

```
            result = timeSpan.Hours > 1 ?
                String.Format("{0} giờ", timeSpan.Hours) :
                "1 giờ";
        }
        else if (timeSpan <= TimeSpan.FromDays(7))
        {
            result = timeSpan.Days > 1 && timeSpan.Days <= 7 ?
                String.Format("{0} ngày", timeSpan.Days) :
                "hôm qua";
        }
        else if (timeSpan <= TimeSpan.FromDays(30))
        {
            result = timeSpan.Days > 7 ?
                String.Format("{0} tuần", timeSpan.Days / 7) :
                "tuần trước";
        }
        else if (timeSpan <= TimeSpan.FromDays(365))
        {
            result = timeSpan.Days > 30 ?
                String.Format("{0} tháng", timeSpan.Days / 30) :
                "1 tháng";
        }
        else
        {
            result = timeSpan.Days > 365 ?
                String.Format("{0} năm", timeSpan.Days / 365) :
                "1 năm";
        }
        return result;
    }
  }
}
```

6.2.2 File Default.aspx.cs

```
using Business.Services.Data;

using Business.Services.Security;

using Data.Entities;

using Microsoft.Exchange.WebServices.Data;

using System;

using System.Collections.Generic;

using System.Configuration;
```

```
using System.Linq;
using System.Net.Mail;
using System.Net;
using System.Text;
using System.Web.UI;
using System.Web.UI.HtmlControls;
using System.Web.UI.WebControls;
using Telerik.Web.UI;
using Web.Sessions;
using Telerik.Web.UI.PivotGrid.Core.Totals;


namespace Web
{
    public partial class _Default : Page
    {
        private readonly UserService userService;
        private readonly PostService postService;
        private readonly CommentService commentService;
        private readonly LikesService likesService;
        private readonly NotificationService notificationService;
        private readonly FollowingService followingService;
        private readonly PostGroupService postGroupService;
        private readonly PostGroupUserService postGroupUserService;
        private readonly ConversationService conversationService;
```

```
    private readonly MessageService messageService;
    public _Default()
    {
        this.userService = new UserService();
        this.postService = new PostService();
        this.commentService = new CommentService();
        this.likesService = new LikesService();
        this.notificationService = new NotificationService();
        this.followingService = new FollowingService();
        this.postGroupUserService = new PostGroupUserService();
        this.postGroupService = new PostGroupService();
        this.conversationService = new ConversationService();
        this.messageService = new MessageService();
    }
    protected void Page_Load(object sender, EventArgs e)
    {
        if (!this.IsPostBack)
        {
        }
    }

    protected void RadListView1_NeedDataSource(object sender,
Telerik.Web.UI.RadListViewNeedDataSourceEventArgs e)
    {
```

```
SiteMaster MasterPage = new SiteMaster();
var baiVietList = this.postService.GetAll().Where(t => string.IsNullOrEmpty(t.GroupId.ToString())).ToList();
var userList = this.userService.GetAll();
var postGroupList = this.postGroupService.GetAll();
int index = 0;
var query = (from a in baiVietList
             join b in userList on a.CreatedBy equals b.ID
             orderby a.CreatedDate descending
             select new {
                 postID = a.ID,
                 postContent = a.PostContent,
                 postImages = a.Images,
                 postCreatedBy = a.CreatedBy,
                 postTime = MasterPage.TimeAgo(a.CreatedDate.GetValueOrDefault()),
                 postLastUpdatedBy = a.LastUpdatedBy,
                 postLastUpdatedDate = a.LastUpdatedDate,
                 groupId = a.GroupId,
                 userID = b.ID,
                 userName = b.Username,
                 userFullName = b.FirstName + " " + b.LastName,
                 userAvatar = b.Avatar,
                 index = index++ }).ToList();

this.RadListView1.DataSource = query;
```

```
    }

    protected void RadListView1_ItemDataBound(object sender, Telerik.Web.UI.RadListViewItemEventArgs e)
    {
        if (e.Item is RadListViewDataItem)
        {
            var userList = this.userService.GetAll();
            var dataItem = e.Item as RadListViewDataItem;
            var objPost = this.postService.GetByID(new Guid(dataItem.GetDataKeyValue("postID").ToString()));
            var objLike = this.likesService.GetByPostAndUserID(objPost.ID, UserSession.Current.User.ID);

            var listImages = objPost.Images.Split(',');
            HtmlContainerControl images = (HtmlContainerControl)dataItem.FindControl("images");
            HiddenField index = (HiddenField)dataItem.FindControl("hfindexID");
            if (listImages.Count() > 1)
            {
                string carouselImages = "<div id='carouselExampleCrossfade" + index.Value + "' class='carousel
slide' data-mdb-interval='false'><div class='carousel-indicators'>";

                for (int i = 0; i < listImages.Count(); i++)
                {
                    if (i == 0)
                    {
```

```
                carouselImages += "<button type='button' data-mdb-target='#carouselExampleIndicators'
data-mdb-slide-to='" + i + "' class='active' aria-current='true' aria-label='Image " + (i + 1) + "'></button>";
            }
            else
            {
                carouselImages += "<button type='button' data-mdb-target='#carouselExampleIndicators'
data-mdb-slide-to='" + i + "' aria-current='true' aria-label='Image " + (i + 1) + "'></button>";
            }
        }

        carouselImages += "</div><div class='carousel-inner' style='border-radius: 0.5rem !important'>";
        for (int i = 0; i < listImages.Count(); i++)
        {
            if (i == 0)
            {
                carouselImages += "<div class='carousel-item active'><img src = '" + listImages[i].ToString()
+ "' style = 'width: -webkit-fill-available; height: 70vh; object-fit: contain; border-radius: inherit'/></div>";
            }
            else
            {
                carouselImages += "<div class='carousel-item'><img src = '" + listImages[i].ToString() + "'
style = 'width: -webkit-fill-available; height: 70vh; object-fit: contain; border-radius: inherit'/></div>";
            }
        }
```

```
                carouselImages += "</div><button class='carousel-control-prev' type='button' data-mdb-
target='#carouselExampleCrossfade" + index.Value + "' data-mdb-slide='prev'><i class='fa-solid fa-arrow-left'
style='font-size: 1.5rem'></i></button><button class='carousel-control-next' type='button' data-mdb-
target='#carouselExampleCrossfade" + index.Value + "' data-mdb-slide='next'><i class='fa-solid fa-arrow-
right' style='font-size: 1.5rem'></i></button></div>";


                images.InnerHtml = carouselImages;
            }
            else
            {
                string image = "<img src='" + listImages[0] + "' loading='lazy' style='width: -webkit-fill-available;
height: 70vh; object-fit: contain; border-radius: inherit' />";
                images.InnerHtml = image;
            }


            LinkButton btnLikes = (LinkButton)e.Item.FindControl("btnLikes");
            if (objLike != null)
            {
                btnLikes.CssClass = "fa-solid fa-heart link-danger";
            }
            else
            {
                btnLikes.CssClass = "fa-regular fa-heart";
            }
```

```
            HtmlContainerControl countLikeBox = (HtmlContainerControl)e.Item.FindControl("countLikeBox");
            LinkButton btnLatedLike = (LinkButton)e.Item.FindControl("btnLatedLike");
            Label lbAnd = (Label)e.Item.FindControl("lbAnd");
            LinkButton btnCountLikes = (LinkButton)e.Item.FindControl("btnCountLikes");
            int countLikes = likesService.CountLikeOfPost(objPost.ID);
            if (countLikes > 1)
            {
                var latedLikeUserID = this.likesService.GetLatedLike(objPost.ID);
                if (latedLikeUserID != null)
                {
                    if (latedLikeUserID == UserSession.Current.User.ID)
                    {
                        btnLatedLike.Text = "Bạn";
                    }
                    else
                    {
                        var latedLikeUser = this.userService.GetByID(latedLikeUserID);
                        btnLatedLike.Text = latedLikeUser.FirstName + " " + latedLikeUser.LastName;
                    }
                }
                btnCountLikes.Text = (likesService.CountLikeOfPost(objPost.ID) - 1).ToString() + " người khác đã
thích.";
            }
            else if (countLikes == 1)
```

```
        {
            var latedLikeUserID = this.likesService.GetLatedLike(objPost.ID);
            if (latedLikeUserID != null)
            {
                if (latedLikeUserID == UserSession.Current.User.ID)
                {
                    btnLatedLike.Text = "Bạn đã thích.";
                }
                else
                {
                    var latedLikeUser = this.userService.GetByID(latedLikeUserID);
                    btnLatedLike.Text = latedLikeUser.FirstName + " " + latedLikeUser.LastName + " đã thích.";
                }
            }
            lbAnd.Visible = false;
            btnCountLikes.Visible = false;
        }
        else
        {
            countLikeBox.Visible = false;
        }
    }
}
```

```
    protected void btnLikes_Click(object sender, EventArgs e)
    {
        int index = Convert.ToInt32(((LinkButton)sender).CommandArgument.ToString());
        HiddenField hiddenFieldPostID = (HiddenField)RadListView1.Items[index].FindControl("hfPostID");
        var objPost = this.postService.GetByID(new Guid(hiddenFieldPostID.Value));
        var likeObj = this.likesService.GetByPostAndUserID(new Guid(hiddenFieldPostID.Value),
UserSession.Current.User.ID);

        if (likeObj != null)
        {
            likesService.Delete(likeObj.ID);
            LinkButton btnLikes = (LinkButton)RadListView1.Items[index].FindControl("btnLikes");
            btnLikes.CssClass = "fa-regular fa-heart";
        }
        else
        {
            Like newLike = new Like()
            {
                ID = Guid.NewGuid(),
                PostID = new Guid(hiddenFieldPostID.Value),
                UserID = new Guid(UserSession.Current.User.ID.ToString()),
                CreatedDate = DateTime.Now,
            };
```

```
            likesService.Insert(newLike);
            LinkButton btnLikes = (LinkButton)RadListView1.Items[index].FindControl("btnLikes");
            btnLikes.CssClass = "fa-solid fa-heart";
            if (objPost.CreatedBy != UserSession.Current.User.ID)
            {
                string msg = UserSession.Current.User.FirstName + " " + UserSession.Current.User.LastName + " đã thích bài viết của bạn.";
                SendNotificationToApp(UserSession.Current.User.ID, objPost.CreatedBy.GetValueOrDefault(), msg);
            }
        }

        HtmlContainerControl countLikeBox = (HtmlContainerControl)RadListView1.Items[index].FindControl("countLikeBox");
        countLikeBox.Visible = true;
        LinkButton btnLatedLike = (LinkButton)RadListView1.Items[0].FindControl("btnLatedLike");
        Label lbAnd = (Label)RadListView1.Items[0].FindControl("lbAnd");
        lbAnd.Visible = true;
        LinkButton btnCountLikes = (LinkButton)RadListView1.Items[index].FindControl("btnCountLikes");
        btnCountLikes.Visible = true;
        int countLikes = likesService.CountLikeOfPost(objPost.ID);
        if (countLikes > 1)
        {
            var latedLikeUserID = this.likesService.GetLatedLike(objPost.ID);
            if (latedLikeUserID != null)
```

```
            {
                if (latedLikeUserID == UserSession.Current.User.ID)
                {
                    btnLatedLike.Text = "Bạn";
                }
                else
                {
                    var latedLikeUser = this.userService.GetByID(latedLikeUserID);
                    btnLatedLike.Text = latedLikeUser.FirstName + " " + latedLikeUser.LastName;
                }
            }
            btnCountLikes.Text = (likesService.CountLikeOfPost(objPost.ID) - 1).ToString() + " người khác đã
thích.";
        }
        else if (countLikes == 1)
        {
            var latedLikeUserID = this.likesService.GetLatedLike(objPost.ID);
            if (latedLikeUserID != null)
            {
                if (latedLikeUserID == UserSession.Current.User.ID)
                {
                    btnLatedLike.Text = "Bạn đã thích.";
                }
                else
```

```
            {
                var latedLikeUser = this.userService.GetByID(latedLikeUserID);
                btnLatedLike.Text = latedLikeUser.FirstName + " " + latedLikeUser.LastName + " đã thích.";
            }
        }
        lbAnd.Visible = false;
        btnCountLikes.Visible = false;
    }
    else
    {
        countLikeBox.Visible = false;
    }
    RadListView1.Rebind();
}

protected void ajaxDocument_AjaxRequest(object sender, AjaxRequestEventArgs e)
{

}

private void SendNotificationToApp(Guid Sender, Guid Reciever, string message)
{
    Notification notification = new Notification()
    {
```

```
            ID = Guid.NewGuid(),
            Message = message,
            IdSender = Sender,
            IdReceiver = Reciever,
            IsRead = false,
            Avatar = userService.GetByID(Sender).Avatar,
            ReceivedDate = DateTime.Now
        };
        notificationService.Insert(notification);
        SiteMaster MasterPage = new SiteMaster();
        MasterPage.getNotification(userService.GetByID(Reciever), notification);
    }

    protected void RadUserList_NeedDataSource(object sender, RadListViewNeedDataSourceEventArgs e)
    {
        var userList = this.userService.GetAll().Where(t => t.ID != UserSession.Current.User.ID).OrderBy(t =>
t.CreatedDate).ToList();
        var followingList = this.followingService.GetByUserID(UserSession.Current.User.ID);
        var _userList = userList.Where(t => !followingList.Select(f => f.IdSender).Contains(t.ID)).Take(9).ToList();
        RadUserList.DataSource = _userList;
    }

    protected void RadUserList_ItemDataBound(object sender, RadListViewItemEventArgs e)
    {
```

```
        if (e.Item is RadListViewDataItem)
        {
            var dataItem = e.Item as RadListViewDataItem;
            LinkButton btnFollow = (LinkButton)dataItem.FindControl("btnFollow");
            var senderID = dataItem.GetDataKeyValue("ID").ToString();
            var followingObj = this.followingService.CheckFollow(new Guid(senderID),
UserSession.Current.User.ID);
            var followingBackObj = this.followingService.CheckFollow(UserSession.Current.User.ID, new
Guid(senderID));
            if (followingObj != null && followingBackObj == null)
            {
                btnFollow.Text = "Theo dõi lại";
            }
            else if (followingObj == null && followingBackObj != null)
            {
                btnFollow.Text = "Đang theo dõi";
            }
            else if (followingObj != null && followingBackObj != null)
            {
                btnFollow.Text = "Đang theo dõi";
            }
            else if (followingObj == null && followingBackObj == null)
            {
                btnFollow.Text = "Theo dõi";
```

```
            }
        }
    }

    protected void RadFollowing_NeedDataSource(object sender, RadListViewNeedDataSourceEventArgs e)
    {
        SiteMaster siteMaster = new SiteMaster();
        var userList = this.userService.GetAll().Where(t => t.ID != UserSession.Current.User.ID).OrderBy(t =>
t.CreatedDate).ToList();
        var followingList = this.followingService.GetByUserID(UserSession.Current.User.ID);
        var query = (from a in userList
                     join b in followingList on a.ID equals b.IdSender
                     orderby b.ReceivedDate descending
                     select new
                     {
                         userID = a.ID,
                         userAvatar = a.Avatar,
                         userName = a.Username,
                         userFullName = a.FirstName + " " + a.LastName,
                         timeReceived = siteMaster.TimeAgo(b.ReceivedDate.GetValueOrDefault())
                     }).Take(9).ToList();

        RadFollowing.DataSource = query;
    }
```

```
    protected void RadFollowing_ItemDataBound(object sender, RadListViewItemEventArgs e)
    {
        if (e.Item is RadListViewDataItem)
        {
            var dataItem = e.Item as RadListViewDataItem;
            LinkButton btnFollow = (LinkButton)dataItem.FindControl("btnFollowBack");
            var senderID = dataItem.GetDataKeyValue("userID").ToString();
            var followingObj = this.followingService.CheckFollow(new Guid(senderID),
UserSession.Current.User.ID);
            var followingBackObj = this.followingService.CheckFollow(UserSession.Current.User.ID, new
Guid(senderID));
            if (followingObj != null && followingBackObj == null)
            {
                btnFollow.Text = "Theo dõi lại";
            }
            else if (followingObj == null && followingBackObj != null)
            {
                btnFollow.Text = "Đang theo dõi";
            }
            else if (followingObj != null && followingBackObj != null)
            {
                btnFollow.Text = "Đang theo dõi";
            }
```

```
            else if (followingObj == null && followingBackObj == null)
            {
                btnFollow.Text = "Theo dõi";
            }
        }
    }

    protected void btnFollow_Click(object sender, EventArgs e)
    {
        var button = (LinkButton)sender;
        var receveiID = button.CommandArgument.ToString();
        if (button.Text == "Đang theo dõi")
        {
            var followingObj = this.followingService.CheckFollow(UserSession.Current.User.ID, new
Guid(receveiID));
            var id = this.followingService.Delete(followingObj);
            if (id)
            {
                this.RadFollowing.Rebind();
                this.RadUserList.Rebind();
            }
        }
        else if (button.Text == "Theo dõi")
        {
```

```
            Following newFollowing = new Following()
            {
                ID = Guid.NewGuid(),
                IdSender = UserSession.Current.User.ID,
                IdReceiver = new Guid(receveiID),
                ReceivedDate = DateTime.Now
            };
            this.followingService.Insert(newFollowing);
            this.RadFollowing.Rebind();
            this.RadUserList.Rebind();
            string msg = UserSession.Current.User.FirstName + " " + UserSession.Current.User.LastName + " đã
bắt đầu theo dõi bạn.";
            SendNotificationToApp(UserSession.Current.User.ID, new Guid(receveiID), msg);
        }
        else if (button.Text == "Theo dõi lại")
        {
            Following newFollowing = new Following()
            {
                ID = Guid.NewGuid(),
                IdSender = UserSession.Current.User.ID,
                IdReceiver = new Guid(receveiID),
                ReceivedDate = DateTime.Now
            };
            this.followingService.Insert(newFollowing);
```

```
            string msg = UserSession.Current.User.FirstName + " " + UserSession.Current.User.LastName + " đã bắt đầu theo dõi bạn.";
            SendNotificationToApp(UserSession.Current.User.ID, new Guid(receveiID), msg);
            Data.Entities.Conversation newConversation = new Data.Entities.Conversation()
            {
                ID = Guid.NewGuid(),
                SenderID = UserSession.Current.User.ID,
                ReceiverID = new Guid(receveiID),
                CreatedDate = DateTime.Now
            };
            this.conversationService.Insert(newConversation);
            Message firstMessage = new Message()
            {
                ID = Guid.NewGuid(),
                Content = "Xin chào 👋",
                SenderID = UserSession.Current.User.ID,
                ReceiverID = new Guid(receveiID),
                ConversationID = newConversation.ID,
                CreatedDate = DateTime.Now
            };
            this.messageService.Insert(firstMessage);
            this.RadFollowing.Rebind();
            this.RadUserList.Rebind();
        }
```

```
        }


        protected void RadGroupList_NeedDataSource(object sender, RadListViewNeedDataSourceEventArgs e)
        {
            var postGroupUserList = this.postGroupUserService.GetAllByUserId(UserSession.Current.User.ID);
            var postGroupList = this.postGroupService.GetAll();
            var query = postGroupList.Where(t => postGroupUserList.Select(p =>
p.GroupId).Contains(t.ID)).Take(6).ToList();

            this.RadGroupList.DataSource = query;
        }


        protected void RadAllGroupList_NeedDataSource(object sender, RadListViewNeedDataSourceEventArgs e)
        {
            var allPostGroupList = this.postGroupService.GetAll().Take(6).ToList();

            this.RadAllGroupList.DataSource = allPostGroupList;
        }
    }
}
```

6.2.3 File ChatApp.aspx.cs

```
using Business.Services.Data;
using Business.Services.Security;
using Data.Entities;
using Microsoft.Ajax.Utilities;
using System;
using System.Collections.Generic;
```

```
using System.Linq;
using System.Reflection;
using System.Web;
using System.Web.UI;
using System.Web.UI.HtmlControls;
using System.Web.UI.WebControls;
using Telerik.Web.UI;
using Telerik.Web.UI.PivotGrid.Core.Totals;
using Web.Sessions;

namespace Web
{
    public partial class ChatApp : System.Web.UI.Page
    {
        private readonly ConversationService conversationService = new ConversationService();
        private readonly UserService userService = new UserService();
        private readonly MessageService messageService = new MessageService();
        protected void Page_Load(object sender, EventArgs e)
        {

        }

        protected void ConversationList_NeedDataSource(object sender,
Telerik.Web.UI.RadListViewNeedDataSourceEventArgs e)
        {
            SiteMaster MasterPage = new SiteMaster();
            var userList = this.userService.GetAll();
            var messageList = this.messageService.GetAll().Where(t => t.ReceiverID ==
UserSession.Current.User.ID).DistinctBy(t => t.SenderID).ToList();
            int index = 0;
            var query = (from a in messageList
                         join b in userList on a.SenderID equals b.ID
                         orderby a.CreatedDate descending
                         select new { messageID = a.ID, createdDate =
MasterPage.TimeAgo(a.CreatedDate.GetValueOrDefault()), messageContent = a.Content, conversationID =
a.ConversationID, userID = b.ID, userFullName = b.FirstName + " " + b.LastName, userAvatar = b.Avatar, index
= index++ }).ToList();

            ConversationList.DataSource = query;
        }

        protected void btnOpenChatConversation_Click(object sender, EventArgs e)
        {
            int index = Convert.ToInt32(((LinkButton)sender).CommandArgument.ToString());
            HiddenField hiddenFieldSenderID =
(HiddenField)ConversationList.Items[index].FindControl("hfSenderID");
            HiddenField hiddenFieldConversationID =
(HiddenField)ConversationList.Items[index].FindControl("hfConversationID");

            refreshMessageBox(hiddenFieldConversationID.Value, hiddenFieldSenderID.Value);
```

```
            this.noChat.Attributes.Add("style", "display: none !important");
            this.hadChat.Attributes.Add("style", "display: block !important; height: 100%");
        }

        protected void receiverAvatar_Click(object sender, EventArgs e)
        {
            Response.Redirect("UserPage.aspx?userID=" + hfUserID.Value);
        }

        protected void btnReceiverFullName_Click(object sender, EventArgs e)
        {
            Response.Redirect("UserPage.aspx?userID=" + hfUserID.Value);
        }

        protected void btnSubmitMessage_Click(object sender, EventArgs e)
        {
            if (!string.IsNullOrEmpty(this.txtMessage.Text))
            {
                var newMessage = new Message()
                {
                    ID = Guid.NewGuid(),
                    Content = this.txtMessage.Text,
                    ConversationID = new Guid(hfConver.Value),
                    SenderID = UserSession.Current.User.ID,
                    ReceiverID = new Guid(hfUserID.Value),
                    CreatedDate = DateTime.Now,
                };
                this.messageService.Insert(newMessage);

                refreshMessageBox(hfConver.Value, hfUserID.Value);
                this.txtMessage.Text = string.Empty;
            }
        }

        protected void refreshMessageBox(string converID, string senderID)
        {
            var messageListOfConversation = this.messageService.GetByConversationID(new
Guid(converID)).OrderBy(t => t.CreatedDate).ToList();

            this.hfUserID.Value = senderID;
            this.hfConver.Value = converID;
            this.receiverAvatar.ImageUrl = this.userService.GetByID(new Guid(senderID)).Avatar;
            this.btnReceiverFullName.Text = this.userService.GetByID(new Guid(senderID)).FirstName + " " +
this.userService.GetByID(new Guid(senderID)).LastName; ;

            string _message = "";
            foreach (var m in messageListOfConversation)
            {
                if (UserSession.Current.User.ID == m.SenderID)
```

```
            {
                _message += "<div class='d-flex justify-content-end' style='margin-bottom: 10px'><div
style='max-width: 40%; margin-right: 10px; border-radius: 10px; border: 0.25rem solid #54B4D3; display: flex;
align-items: center; padding: 0.5rem; background-color: #54B4D3'><p style='margin: 0; color: white'>" +
m.Content.ToString() + "</p></div><asp:LinkButton runat='server' OnClick='receiverAvatar_Click'
style='display: flex; align-items: center'><image src='" +
this.userService.GetByID(m.SenderID.GetValueOrDefault()).Avatar + "' class='rounded-circle' Width='40'
Height='40' style='border: 0.2rem solid #54B4D3' /></asp:LinkButton></div>";
            }
            else
            {
                _message += "<div class='d-flex justify-content-start' style='margin-bottom:
10px'><asp:LinkButton runat='server' OnClick='receiverAvatar_Click' style='display: flex; align-items:
center'><image src='" + this.userService.GetByID(m.SenderID.GetValueOrDefault()).Avatar + "' class='rounded-
circle' Width='40' Height='40' style='border: 0.2rem solid #54B4D3' /></asp:LinkButton><div style='max-
width: 40%; margin-left: 10px; border-radius: 10px; border: 0.25rem solid #54B4D3; display: flex; align-items:
center; padding: 0.5rem;'><p style='margin: 0'>" + m.Content.ToString() + "</p></div></div>";
            }
        }
        this.messageBox.InnerHtml = _message;
    }
  }
}
```

6.2.4 File AddNewPostForm.aspx.cs

```
using Business.Services.Data;

using Business.Services.Security;

using Data.Entities;

using System;

using System.Collections.Generic;

using System.IO;

using System.Linq;

using System.Web;

using System.Web.UI;

using System.Web.UI.HtmlControls;

using System.Web.UI.WebControls;
```

```
using Telerik.Web.UI;
using Web.Sessions;


namespace Web.Controls.Data
{
    public partial class AddNewPostForm : System.Web.UI.Page
    {
        private readonly PostService postService = new PostService();
        private readonly PostGroupService postGroupService = new PostGroupService();
        private readonly UserService userService = new UserService();
        private readonly NotificationService notificationService = new
NotificationService();
        private readonly PostGroupUserService postGroupUserService = new
PostGroupUserService();


        protected void Page_Load(object sender, EventArgs e)
        {
            if(!Page.IsPostBack)
            {
                this.masterUserAvatar.ImageUrl = UserSession.Current.User.Avatar;
                this.btnMasterUser.Text = UserSession.Current.User.FirstName + " " +
UserSession.Current.User.LastName;
```

```
        }
    }

    protected void btnNextStep_Click(object sender, EventArgs e)
    {
        if (Page.IsValid && imagesUpload1.HasFile)
        {
            var postedFiles = imagesUpload1.PostedFiles;
            Guid postID = Guid.NewGuid();
            string imgStringUrl = "";

            var images = this.images;
            if (postedFiles.Count() > 1)
            {
                Directory.CreateDirectory(Server.MapPath("~/DataSource/Images/" +
postID.ToString() + "/"));
                foreach (HttpPostedFile uploadedFile in postedFiles)
                {

uploadedFile.SaveAs(System.IO.Path.Combine(Server.MapPath("~/DataSource/Images/"
+ postID.ToString() + "/"), uploadedFile.FileName));
                }
```

```
            for (int i = 0; i < postedFiles.Count(); i++)
            {
                if (i < postedFiles.Count() - 1)
                {
                    imgStringUrl += "/DataSource/Images/" + postID.ToString() + "/" +
postedFiles[i].FileName + ",";
                }
                else
                {
                    imgStringUrl += "/DataSource/Images/" + postID.ToString() + "/" +
postedFiles[i].FileName;
                }
            }
            this.hfPostID.Value = postID.ToString();
            this.hfImgStringURL.Value = imgStringUrl;

            string carouselImages = "<div id='carouselExampleCrossfade'
class='carousel slide' data-mdb-interval='false'><div class='carousel-indicators'>";

            for (int j = 0; j < postedFiles.Count(); j++)
            {
```

```
            if (j == 0)
            {
                carouselImages += "<button type='button' data-mdb-target='#carouselExampleIndicators' data-mdb-slide-to='" + j + "' class='active' aria-current='true' aria-label='Image " + (j + 1) + "'></button>";
            }
            else
            {
                carouselImages += "<button type='button' data-mdb-target='#carouselExampleIndicators' data-mdb-slide-to='" + j + "' aria-current='true' aria-label='Image " + (j + 1) + "'></button>";
            }
        }

        carouselImages += "</div><div class='carousel-inner'>";
        for (int k = 0; k < postedFiles.Count(); k++)
        {
            if (k == 0)
            {
                carouselImages += "<div class='carousel-item active'><img src = '../../DataSource/Images/" + postID.ToString() + "/" + postedFiles[k].FileName + "' style = 'width: -webkit-fill-available; height: 100vh; object-fit: contain; border-radius: inherit'/></div>";
```

```
                }
                else
                {
                    carouselImages += "<div class='carousel-item'><img src =
'../../DataSource/Images/" + postID.ToString() + "/" + postedFiles[k].FileName + "' style
= 'width: -webkit-fill-available; height: 100vh; object-fit: contain; border-radius:
inherit'/></div>";
                }
            }

            carouselImages += "</div><button class='carousel-control-prev'
type='button' data-mdb-target='#carouselExampleCrossfade' data-mdb-slide='prev'><i
class='fa-solid fa-arrow-left' style='font-size: 1.5rem'></i></button><button
class='carousel-control-next' type='button' data-mdb-target='#carouselExampleCrossfade'
data-mdb-slide='next'><i class='fa-solid fa-arrow-right' style='font-size:
1.5rem'></i></button></div>";


            images.InnerHtml = carouselImages;
        }
        else
        {
```

```
            imgStringUrl += "/DataSource/Images/" + postID.ToString() + "/" +
postedFiles[0].FileName;

            this.hfPostID.Value = postID.ToString();

            this.hfImgStringURL.Value = imgStringUrl;

            Directory.CreateDirectory(Server.MapPath("~/DataSource/Images/" +
postID.ToString() + "/"));


imagesUpload1.SaveAs(System.IO.Path.Combine(Server.MapPath("~/DataSource/Image
s/" + postID.ToString() + "/"), imagesUpload1.FileName));

            images.InnerHtml = "<img src='../../DataSource/Images/" +
postID.ToString() + "/" + postedFiles[0].FileName + "' loading='lazy' style='width: -
webkit-fill-available; max-height: 100vh; object-fit: contain; border-radius: inherit'/>";

        }


            this.uploadBox.Attributes.Add("style", "display: none !important");

            this.postImages.Attributes.Add("class", "col-8");

            this.postImages.Attributes.Add("style", "display: flex !important");

            this.images.Attributes.Add("style", "display: flex !important; height: 100vh;
width: -webkit-fill-available");

            this.postInfo.Attributes.Add("class", "col-4");

            this.postInfo.Attributes.Add("style", "display: block !important; padding: 0
0.25rem; height: 100vh");

        }
```

```
}
```

```
protected void btnPreviousStep_Click(object sender, EventArgs e)
{
    this.uploadBox.Attributes.Add("class", "d-flex justify-content-center");
    this.uploadBox.Attributes.Add("style", "margin: 0.25rem 0; width: 100%");
    this.postImages.Attributes.Add("class", "col");
    this.postImages.Attributes.Add("style", "display: flex !important");
    this.images.Attributes.Add("style", "display: none !important");
    this.postInfo.Attributes.Add("style", "display: none !important");
    this.hfPostID.Value = string.Empty;
    this.hfImgStringURL.Value = string.Empty;
}

protected void btnPost_Click(object sender, EventArgs e)
{
    if (string.IsNullOrEmpty(this.Request.QueryString["groupID"]))
    {
        var newPost = new Post()
        {
            ID = new Guid(this.hfPostID.Value),
            Images = this.hfImgStringURL.Value,
            PostContent = this.txtStatus.Text,
            CreatedBy = new Guid(UserSession.Current.User.ID.ToString()),
```

```
            CreatedDate = DateTime.Now,
            IsApproved = true
        };
        this.postService.Insert(newPost);
    }
    else
    {
        var newPost = new Post()
        {
            ID = new Guid(this.hfPostID.Value),
            Images = this.hfImgStringURL.Value,
            PostContent = this.txtStatus.Text,
            CreatedBy = new Guid(UserSession.Current.User.ID.ToString()),
            CreatedDate = DateTime.Now,
            GroupId = new Guid(this.Request.QueryString["groupID"])
        };

        var groupId = this.Request.QueryString["groupID"];
        var groupObj = this.postGroupService.GetByID(new Guid(groupId));
        if (groupObj != null)
        {
            if (groupObj.IsPrivate.GetValueOrDefault())
            {
                newPost.IsApproved = false;
```

```
                this.postService.Insert(newPost);
                string msg = UserSession.Current.User.FirstName + " " + UserSession.Current.User.LastName + " vừa yêu cầu đăng bài viết vào nhóm @" + groupObj.GroupName + " của bạn.";
                SendNotificationToApp(UserSession.Current.User.ID, groupObj.Admin.GetValueOrDefault(), msg);
            }
            else
            {
                newPost.IsApproved = true;
                this.postService.Insert(newPost);
                string msg = UserSession.Current.User.FirstName + " " + UserSession.Current.User.LastName + " đã đăng bài viết vào nhóm @" + groupObj.GroupName + ".";
                var userInGroup = this.postGroupUserService.GetAllByGroupId(groupObj.ID);
                foreach (var p in userInGroup)
                {
                    if (p.UserId != UserSession.Current.User.ID)
                    {
                        SendNotificationToApp(UserSession.Current.User.ID, p.UserId.GetValueOrDefault(), msg);
                    }
                }
            }
        }
        ScriptManager.RegisterStartupScript(this, typeof(Page), "closeScript", "CloseAndRebind();", true);
    }
```

```
    private void SendNotificationToApp(Guid Sender, Guid Reciever, string message)
    {
        Notification notification = new Notification()
        {
            ID = Guid.NewGuid(),
            Message = message,
            IdSender = Sender,
            IdReceiver = Reciever,
            IsRead = false,
            Avatar = userService.GetByID(Sender).Avatar,
            ReceivedDate = DateTime.Now
        };
        notificationService.Insert(notification);
        SiteMaster MasterPage = new SiteMaster();
        MasterPage.getNotification(userService.GetByID(Reciever), notification);
    }
  }
}
```

6.2.5 File PostDetailForm.aspx.cs

```
using Business.Services.Data;
using Business.Services.Security;
using Data.Entities;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
```

```
using System.Web.UI;
using System.Web.UI.HtmlControls;
using System.Web.UI.WebControls;
using Web.Sessions;

namespace Web.Controls.Data
{
   public partial class PostDetailForm : System.Web.UI.Page
   {
      protected readonly PostService postService;
      protected readonly CommentService commentService;
      protected readonly UserService userService;

      public PostDetailForm()
      {
         this.userService = new UserService();
         this.postService = new PostService();
         this.commentService = new CommentService();
      }
      protected void Page_Load(object sender, EventArgs e)
      {
         if (!this.IsPostBack)
         {
            var postObj = this.postService.GetByID(new Guid(this.Request.QueryString["postID"]));
            if (postObj != null)
            {
               var listImages = postObj.Images.Split(',');
               var images = this.images;
               if (listImages.Count() > 1)
               {
                  string carouselImages = "<div id='carouselExampleCrossfade' class='carousel slide' data-mdb-
interval='false'><div class='carousel-indicators'>";

                  for (int i = 0; i < listImages.Count(); i++)
                  {
                     if (i == 0)
                     {
                        carouselImages += "<button type='button' data-mdb-target='#carouselExampleIndicators'
data-mdb-slide-to='" + i + "' class='active' aria-current='true' aria-label='Image " + (i + 1) + "'></button>";
                     }
                     else
                     {
                        carouselImages += "<button type='button' data-mdb-target='#carouselExampleIndicators'
data-mdb-slide-to='" + i + "' aria-current='true' aria-label='Image " + (i + 1) + "'></button>";
                     }
                  }

                  carouselImages += "</div><div class='carousel-inner'>";
                  for (int i = 0; i < listImages.Count(); i++)
                  {
```

```
                    if (i == 0)
                    {
                        carouselImages += "<div class='carousel-item active'><img src = '" +
listImages[i].ToString() + "' style = 'width: -webkit-fill-available; height: 100vh; object-fit: contain; border-radius:
inherit'/></div>";
                    }
                    else
                    {
                        carouselImages += "<div class='carousel-item'><img src = '" + listImages[i].ToString() + "'
style = 'width: -webkit-fill-available; height: 100vh; object-fit: contain; border-radius: inherit'/></div>";
                    }
                }

                carouselImages += "</div><button class='carousel-control-prev' type='button' data-mdb-
target='#carouselExampleCrossfade' data-mdb-slide='prev'><i class='fa-solid fa-arrow-left' style='font-size:
1.5rem'></i></button><button class='carousel-control-next' type='button' data-mdb-
target='#carouselExampleCrossfade' data-mdb-slide='next'><i class='fa-solid fa-arrow-right' style='font-size:
1.5rem'></i></button></div>";

                images.InnerHtml = carouselImages;
            }
            else
            {
                string image = "<img src='" + listImages[0] + "' loading='lazy' Style='width: -webkit-fill-
available; max-height: 100%; object-fit: contain; border-radius: inherit' />";
                images.InnerHtml = image;
            }

            var userObj = this.userService.GetByID(postObj.CreatedBy.GetValueOrDefault());
            if (userObj != null)
            {
                this.masterUserAvatar.ImageUrl = userObj.Avatar;
                this.btnMasterUser.Text = userObj.FirstName + " " + userObj.LastName;
            }
        }
    }
}

protected void ListComment_NeedDataSource(object sender,
Telerik.Web.UI.RadListViewNeedDataSourceEventArgs e)
{
    var commentList = this.commentService.GetByPostID(new Guid(this.Request.QueryString["postID"]));
    var userList = this.userService.GetAll();
    var query = (from a in commentList
                 join b in userList on a.UserID equals b.ID
                 select new { commentID = a.ID, commentContent = a.Content, commentCreatedDate =
a.CreatedDate, userID = b.ID, userFullName = b.FirstName + " " + b.LastName, userName = b.Username,
userAvatar = b.Avatar }).ToList();
    this.ListComment.DataSource = query;
}
```

```
    protected void ListComment_ItemDataBound(object sender, Telerik.Web.UI.RadListViewItemEventArgs e)
    {

    }

    protected void btnMasterUser_Click(object sender, EventArgs e)
    {
        var postObj = this.postService.GetByID(new Guid(this.Request.QueryString["postID"]));
        Response.Redirect("../../UserPage.aspx?userID=" + postObj.CreatedBy);
    }

    protected void btnSubmitComment_Click(object sender, EventArgs e)
    {
        if (!string.IsNullOrEmpty(this.txtComment.Text))
        {
            var newComment = new Comment()
            {
                ID = Guid.NewGuid(),
                Content = this.txtComment.Text,
                PostID = new Guid(this.Request.QueryString["postID"]),
                UserID = UserSession.Current.User.ID,
                CreatedDate = DateTime.Now,
            };
            this.commentService.Insert(newComment);
            this.ListComment.Rebind();
            this.txtComment.Text = string.Empty;
        }
    }
  }
}
```

6.2.6 File UserPage.aspx.cs

```
using Business.Services.Data;
using Business.Services.Security;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.HtmlControls;
using System.Web.UI.WebControls;
using Telerik.Charting.Styles;
using Telerik.Web.UI;
using Web.Sessions;

namespace Web.Template.Data
{
  public partial class UserPage : System.Web.UI.Page
```

```
  {
      private readonly PostService postService = new PostService();
      private readonly UserService userService = new UserService();
      private readonly FollowingService followingService = new FollowingService();

      protected void Page_Load(object sender, EventArgs e)
      {
          if (!IsPostBack)
          {
              if (string.IsNullOrEmpty(this.Request.QueryString["userID"]))
              {
                  this.btnProfileEdit1.Visible = true;
                  Page.Title = (UserSession.Current.User.FirstName + " " +
UserSession.Current.User.LastName).ToString();
                  this.userAvatar.ImageUrl = UserSession.Current.User.Avatar.ToString();
                  this.lbUserName.Text = UserSession.Current.User.Username.ToString();
                  this.lbPostCount.Text = this.postService.CountByUser(UserSession.Current.User.ID).ToString();
                  this.lbFollowing.Text =
this.followingService.GetBySenderID(UserSession.Current.User.ID).Count().ToString();
                  this.lbFollower.Text =
this.followingService.GetByUserID(UserSession.Current.User.ID).Count().ToString();
                  this.lbFullName.Text = UserSession.Current.User.FirstName.ToString() + " " +
UserSession.Current.User.LastName.ToString();
                  if (string.IsNullOrEmpty(UserSession.Current.User.Introduction))
                  {
                      this.lbIntroduction.Text = "Người dùng chưa để lại giới thiệu !";
                  }
                  else
                  {
                      this.lbIntroduction.Text = UserSession.Current.User.Introduction.ToString();
                  }
              }
              else
              {
                  if (new Guid(this.Request.QueryString["userID"]) == UserSession.Current.User.ID)
                  {
                      this.btnProfileEdit1.Visible = true;
                      Page.Title = (UserSession.Current.User.FirstName + " " +
UserSession.Current.User.LastName).ToString();
                      this.userAvatar.ImageUrl = UserSession.Current.User.Avatar.ToString();
                      this.lbUserName.Text = UserSession.Current.User.Username.ToString();
                      this.lbPostCount.Text = this.postService.CountByUser(UserSession.Current.User.ID).ToString();
                      this.lbFollowing.Text =
this.followingService.GetBySenderID(UserSession.Current.User.ID).Count().ToString();
                      this.lbFollower.Text =
this.followingService.GetByUserID(UserSession.Current.User.ID).Count().ToString();
                      this.lbFullName.Text = UserSession.Current.User.FirstName.ToString() + " " +
UserSession.Current.User.LastName.ToString();
                      if (string.IsNullOrEmpty(UserSession.Current.User.Introduction))
                      {
```

```
                this.lbIntroduction.Text = "Người dùng chưa để lại giới thiệu !";
            }
            else
            {
                this.lbIntroduction.Text = UserSession.Current.User.Introduction.ToString();
            }
        }
        else
        {
            var userObj = this.userService.GetByID(new Guid(this.Request.QueryString["userID"]));
            this.btnProfileEdit1.Visible = false;
            Page.Title = (userObj.FirstName + " " + userObj.LastName).ToString();
            this.userAvatar.ImageUrl = userObj.Avatar.ToString();
            this.lbUserName.Text = userObj.Username.ToString();
            this.lbPostCount.Text = this.postService.CountByUser(userObj.ID).ToString();
            this.lbFollowing.Text = this.followingService.GetBySenderID(userObj.ID).Count().ToString();
            this.lbFollower.Text = this.followingService.GetByUserID(userObj.ID).Count().ToString();
            this.lbFullName.Text = userObj.FirstName.ToString() + " " + userObj.LastName.ToString();
            if (string.IsNullOrEmpty(userObj.Introduction))
            {
                this.lbIntroduction.Text = "Người dùng chưa để lại giới thiệu !";
            }
            else
            {
                this.lbIntroduction.Text = userObj.Introduction.ToString();
            }
        }
    }
}

protected void RadListView1_NeedDataSource(object sender,
Telerik.Web.UI.RadListViewNeedDataSourceEventArgs e)
{
    if (string.IsNullOrEmpty(this.Request.QueryString["userID"]))
    {
        this.RadListView1.DataSource = postService.GetAll().Where(t => t.CreatedBy ==
UserSession.Current.User.ID);
    }
    else
    {
        var userObj = this.userService.GetByID(new Guid(this.Request.QueryString["userID"]));
        this.RadListView1.DataSource = postService.GetAll().Where(t => t.CreatedBy == userObj.ID);
    }
}

protected void RadListView1_ItemDataBound(object sender, Telerik.Web.UI.RadListViewItemEventArgs e)
{
    if (e.Item is RadListViewDataItem)
    {
```

```
            var dataItem = e.Item as RadListViewDataItem;
            var objPost = this.postService.GetByID(new Guid(dataItem.GetDataKeyValue("ID").ToString()));
            var listImages = objPost.Images.Split(',');
            HtmlContainerControl images = (HtmlContainerControl)dataItem.FindControl("images");
            if (listImages.Count() > 1)
            {
                string image = "<i class='fa-solid fa-images' style='position: absolute; top: 10px; left: 20px; color: white'></i><image src = '" + listImages[0].ToString() + "' style = 'object-fit: cover; width: 300px; height: 300px'/>";

                images.InnerHtml = image;
            }
            else
            {
                string image = "<image src = '" + listImages[0].ToString()+"' style = 'object-fit: cover; width: 300px; height: 300px'/>";
                images.InnerHtml = image;
            }
        }
    }
  }
}
```

6.2.7 File GroupPage.aspx.cs

```
using Business.Services.Data;

using Business.Services.Security;

using Data.Entities;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Web;

using System.Web.UI;

using System.Web.UI.HtmlControls;

using System.Web.UI.WebControls;

using Telerik.Web.UI;

using Web.Sessions;
```

```
namespace Web
{
    public partial class GroupPage : System.Web.UI.Page
    {
        private readonly PostGroupService postGroupService = new PostGroupService();
        private readonly PostGroupUserService postGroupUserService = new PostGroupUserService();
        private readonly UserService userService = new UserService();
        private readonly PostService postService = new PostService();
        private readonly LikesService likesService = new LikesService();
        private readonly CommentService commentService = new CommentService();
        private readonly NotificationService notificationService = new NotificationService();
        protected void Page_Load(object sender, EventArgs e)
        {
            if (!this.IsPostBack)
            {
                this.hfGroupID.Value = this.Request.QueryString["groupID"];
                var groupObj = this.postGroupService.GetByID(new Guid(this.Request.QueryString["groupID"]));
                Page.Title = groupObj.GroupName;
                this.backGroupImg.ImageUrl = groupObj.BackGroundImg;
                this.groupName.Text = groupObj.GroupName;
                var checkAdminInGroup = this.postGroupUserService.CheckAdminInGroup(new
Guid(this.Request.QueryString["groupID"].ToString()), UserSession.Current.User.ID);
```

```
            var checkUserInGroup = this.postGroupUserService.CheckUserInGroup(new
Guid(this.Request.QueryString["groupID"].ToString()), UserSession.Current.User.ID);
            if (checkAdminInGroup)
            {
                this.navbarDropdownMenuLink1.Visible = true;
                this.btnInOutGroup.Visible = false;
            }
            else if (!checkAdminInGroup)
            {
                this.navbarDropdownMenuLink1.Visible = false;
                this.btnInOutGroup.Visible = true;
            }

            if (checkUserInGroup)
            {
                this.btnInOutGroup.Text = "Đã tham gia";
                this.btnNewPostInGroup.Visible = true;
                this.btnViewMore.Visible = true;
            }
            else if (!checkUserInGroup)
            {
                var postGroupUserObj =
this.postGroupUserService.GetByUserAndGroup(UserSession.Current.User.ID, new
Guid(this.Request.QueryString["groupID"].ToString()));
                if (postGroupUserObj != null)
```

```
            {
                this.btnInOutGroup.Text = "Chờ duyệt";
            }
            else
            {
                this.btnInOutGroup.Text = "Tham gia nhóm";
            }
            this.btnNewPostInGroup.Visible = false;
            this.btnViewMore.Visible = false;
        }

        if (groupObj.IsPrivate == true)
        {
            this.groupStatus.InnerHtml = "<p class='link-light m-0 align-self-center'><i class='fa-solid fa-
lock'></i> Nhóm riêng tư</p>";
            if (checkUserInGroup)
            {
                this.warning.Visible = false;
                this.groupContent.Visible = true;
            }
            else
            {
                this.warning.Visible = true;
                this.groupContent.Visible = false;
```

```
            }
        }
        else if (groupObj.IsPrivate != true)
        {
            this.warning.Visible = false;
            this.groupStatus.InnerHtml = "<p class='link-light m-0 align-self-center'><i class='fa-solid fa-globe'></i> Nhóm công khai</p>";
        }
        this.createdDate.Text = "Được tạo vào " + groupObj.CreatedDate.GetValueOrDefault().ToString("dd/MM/yyyy");
        this.btnAdmin.Text = this.userService.GetByID(groupObj.Admin.GetValueOrDefault()).FirstName + " " + this.userService.GetByID(groupObj.Admin.GetValueOrDefault()).LastName;
        this.groupDescription.Text = groupObj.GroupDescription;
      }
    }

    protected void RadListView1_NeedDataSource(object sender, Telerik.Web.UI.RadListViewNeedDataSourceEventArgs e)
    {
        SiteMaster MasterPage = new SiteMaster();
        var baiVietList = this.postService.GetAll().Where(t => t.GroupId == new Guid(this.Request.QueryString["groupID"]) && t.IsApproved == true).ToList();
        var userList = this.userService.GetAll();
        var postGroupList = this.postGroupService.GetAll();
        int index = 0;
```

```
    var query = (from a in baiVietList
                 join b in userList on a.CreatedBy equals b.ID
                 orderby a.CreatedDate descending
                 select new
                 {
                     postID = a.ID,
                     postContent = a.PostContent,
                     postImages = a.Images,
                     postCreatedBy = a.CreatedBy,
                     postTime = MasterPage.TimeAgo(a.CreatedDate.GetValueOrDefault()),
                     postLastUpdatedBy = a.LastUpdatedBy,
                     postLastUpdatedDate = a.LastUpdatedDate,
                     groupId = a.GroupId,
                     userID = b.ID,
                     userName = b.Username,
                     userFullName = b.FirstName + " " + b.LastName,
                     userAvatar = b.Avatar,
                     index = index++
                 }).ToList();

    this.RadListView1.DataSource = query;
}

protected void RadListView1_ItemDataBound(object sender, Telerik.Web.UI.RadListViewItemEventArgs e)
```

```
    {
        if (e.Item is RadListViewDataItem)
        {
            var userList = this.userService.GetAll();
            var dataItem = e.Item as RadListViewDataItem;
            var objPost = this.postService.GetByID(new Guid(dataItem.GetDataKeyValue("postID").ToString()));
            var objLike = this.likesService.GetByPostAndUserID(objPost.ID, UserSession.Current.User.ID);

            var listImages = objPost.Images.Split(',');
            HtmlContainerControl images = (HtmlContainerControl)dataItem.FindControl("images");
            HiddenField index = (HiddenField)dataItem.FindControl("hfindexID");
            if (listImages.Count() > 1)
            {
                string carouselImages = "<div id='carouselExampleCrossfade" + index.Value + "' class='carousel
slide' data-mdb-interval='false'><div class='carousel-indicators'>";

                for (int i = 0; i < listImages.Count(); i++)
                {
                    if (i == 0)
                    {
                        carouselImages += "<button type='button' data-mdb-target='#carouselExampleIndicators'
data-mdb-slide-to='" + i + "' class='active' aria-current='true' aria-label='Image " + (i + 1) + "'></button>";
                    }
                    else
```

```
                {
                    carouselImages += "<button type='button' data-mdb-target='#carouselExampleIndicators'
data-mdb-slide-to='" + i + "' aria-current='true' aria-label='Image " + (i + 1) + "'></button>";
                }
            }

            carouselImages += "</div><div class='carousel-inner' style='border-radius: 0.5rem !important'>";
            for (int i = 0; i < listImages.Count(); i++)
            {
                if (i == 0)
                {
                    carouselImages += "<div class='carousel-item active'><img src = '" + listImages[i].ToString()
+ "' style = 'width: -webkit-fill-available; height: 70vh; object-fit: contain; border-radius: inherit'/></div>";
                }
                else
                {
                    carouselImages += "<div class='carousel-item'><img src = '" + listImages[i].ToString() + "'
style = 'width: -webkit-fill-available; height: 70vh; object-fit: contain; border-radius: inherit'/></div>";
                }
            }

            carouselImages += "</div><button class='carousel-control-prev' type='button' data-mdb-
target='#carouselExampleCrossfade" + index.Value + "' data-mdb-slide='prev'><i class='fa-solid fa-arrow-left'
style='font-size: 1.5rem'></i></button><button class='carousel-control-next' type='button' data-mdb-
target='#carouselExampleCrossfade" + index.Value + "' data-mdb-slide='next'><i class='fa-solid fa-arrow-
right' style='font-size: 1.5rem'></i></button></div>";
```

```
                images.InnerHtml = carouselImages;
            }
            else
            {
                string image = "<img src='" + listImages[0] + "' loading='lazy' style='width: -webkit-fill-available;
height: 70vh; object-fit: contain; border-radius: inherit' />";
                images.InnerHtml = image;
            }

            LinkButton btnLikes = (LinkButton)e.Item.FindControl("btnLikes");
            if (objLike != null)
            {
                btnLikes.CssClass = "fa-solid fa-heart link-danger";
            }
            else
            {
                btnLikes.CssClass = "fa-regular fa-heart";
            }

            HtmlContainerControl countLikeBox = (HtmlContainerControl)e.Item.FindControl("countLikeBox");
            LinkButton btnLatedLike = (LinkButton)e.Item.FindControl("btnLatedLike");
            Label lbAnd = (Label)e.Item.FindControl("lbAnd");
            LinkButton btnCountLikes = (LinkButton)e.Item.FindControl("btnCountLikes");
```

```
            int countLikes = likesService.CountLikeOfPost(objPost.ID);
            if (countLikes > 1)
            {
                var latedLikeUserID = this.likesService.GetLatedLike(objPost.ID);
                if (latedLikeUserID != null)
                {
                    if (latedLikeUserID == UserSession.Current.User.ID)
                    {
                        btnLatedLike.Text = "Bạn";
                    }
                    else
                    {
                        var latedLikeUser = this.userService.GetByID(latedLikeUserID);
                        btnLatedLike.Text = latedLikeUser.FirstName + " " + latedLikeUser.LastName;
                    }
                }
                btnCountLikes.Text = (likesService.CountLikeOfPost(objPost.ID) - 1).ToString() + " người khác đã
thích.";
            }
            else if (countLikes == 1)
            {
                var latedLikeUserID = this.likesService.GetLatedLike(objPost.ID);
                if (latedLikeUserID != null)
                {
```

```
                if (latedLikeUserID == UserSession.Current.User.ID)
                {
                    btnLatedLike.Text = "Bạn đã thích.";
                }
                else
                {
                    var latedLikeUser = this.userService.GetByID(latedLikeUserID);
                    btnLatedLike.Text = latedLikeUser.FirstName + " " + latedLikeUser.LastName + " đã thích.";
                }
            }
            lbAnd.Visible = false;
            btnCountLikes.Visible = false;
        }
        else
        {
            countLikeBox.Visible = false;
        }
    }
}

protected void btnLikes_Click(object sender, EventArgs e)
{
    int index = Convert.ToInt32(((LinkButton)sender).CommandArgument.ToString());
    HiddenField hiddenFieldPostID = (HiddenField)RadListView1.Items[index].FindControl("hfPostID");
```

```
        var objPost = this.postService.GetByID(new Guid(hiddenFieldPostID.Value));
        var likeObj = this.likesService.GetByPostAndUserID(new Guid(hiddenFieldPostID.Value),
UserSession.Current.User.ID);


        if (likeObj != null)
        {
            likesService.Delete(likeObj.ID);
            LinkButton btnLikes = (LinkButton)RadListView1.Items[index].FindControl("btnLikes");
            btnLikes.CssClass = "fa-regular fa-heart";
        }
        else
        {
            Like newLike = new Like()
            {
                ID = Guid.NewGuid(),
                PostID = new Guid(hiddenFieldPostID.Value),
                UserID = new Guid(UserSession.Current.User.ID.ToString()),
                CreatedDate = DateTime.Now,
            };
            likesService.Insert(newLike);
            LinkButton btnLikes = (LinkButton)RadListView1.Items[index].FindControl("btnLikes");
            btnLikes.CssClass = "fa-solid fa-heart";
            if (objPost.CreatedBy != UserSession.Current.User.ID)
            {
```

```
                string msg = UserSession.Current.User.FirstName + " " + UserSession.Current.User.LastName + " đã thích bài viết của bạn.";
                SendNotificationToApp(UserSession.Current.User.ID, objPost.CreatedBy.GetValueOrDefault(), msg);
            }
        }

        HtmlContainerControl countLikeBox = (HtmlContainerControl)RadListView1.Items[index].FindControl("countLikeBox");
        countLikeBox.Visible = true;
        LinkButton btnLatedLike = (LinkButton)RadListView1.Items[0].FindControl("btnLatedLike");
        Label lbAnd = (Label)RadListView1.Items[0].FindControl("lbAnd");
        lbAnd.Visible = true;
        LinkButton btnCountLikes = (LinkButton)RadListView1.Items[index].FindControl("btnCountLikes");
        btnCountLikes.Visible = true;
        int countLikes = likesService.CountLikeOfPost(objPost.ID);
        if (countLikes > 1)
        {
            var latedLikeUserID = this.likesService.GetLatedLike(objPost.ID);
            if (latedLikeUserID != null)
            {
                if (latedLikeUserID == UserSession.Current.User.ID)
                {
                    btnLatedLike.Text = "Bạn";
                }
```

```
                else
                {
                    var latedLikeUser = this.userService.GetByID(latedLikeUserID);
                    btnLatedLike.Text = latedLikeUser.FirstName + " " + latedLikeUser.LastName;
                }
            }
            btnCountLikes.Text = (likesService.CountLikeOfPost(objPost.ID) - 1).ToString() + " người khác đã
thích.";
        }
        else if (countLikes == 1)
        {
            var latedLikeUserID = this.likesService.GetLatedLike(objPost.ID);
            if (latedLikeUserID != null)
            {
                if (latedLikeUserID == UserSession.Current.User.ID)
                {
                    btnLatedLike.Text = "Bạn đã thích.";
                }
                else
                {
                    var latedLikeUser = this.userService.GetByID(latedLikeUserID);
                    btnLatedLike.Text = latedLikeUser.FirstName + " " + latedLikeUser.LastName + " đã thích.";
                }
            }
```

```
            lbAnd.Visible = false;
            btnCountLikes.Visible = false;
        }
        else
        {
            countLikeBox.Visible = false;
        }
        RadListView1.Rebind();
    }

    protected void ajaxDocument_AjaxRequest(object sender, AjaxRequestEventArgs e)
    {

    }

    private void SendNotificationToApp(Guid Sender, Guid Reciever, string message)
    {
        Notification notification = new Notification()
        {
            ID = Guid.NewGuid(),
            Message = message,
            IdSender = Sender,
            IdReceiver = Reciever,
            IsRead = false,
```

```
            Avatar = userService.GetByID(Sender).Avatar,
            ReceivedDate = DateTime.Now
        };
        notificationService.Insert(notification);
        SiteMaster MasterPage = new SiteMaster();
        MasterPage.getNotification(userService.GetByID(Reciever), notification);
    }

    protected void btnInOutGroup_Click(object sender, EventArgs e)
    {
        if (this.btnInOutGroup.Text == "Đã tham gia" || this.btnInOutGroup.Text == "Chờ duyệt")
        {
            var postGroupUserObj = this.postGroupUserService.GetByUserAndGroup(UserSession.Current.User.ID,
new Guid(this.Request.QueryString["groupID"].ToString()));
            var id = this.postGroupUserService.Delete(postGroupUserObj);
            if (id)
            {
                Response.Redirect(Request.RawUrl);
            }
        }
        else if (this.btnInOutGroup.Text == "Tham gia nhóm")
        {
            var groupObj = this.postGroupService.GetByID(new
Guid(this.Request.QueryString["groupID"].ToString()));
```

```
            if (groupObj != null)
            {
                if (groupObj.IsPrivate == true)
                {
                    PostGroupUser newPostGroupUser = new PostGroupUser()
                    {
                        ID = Guid.NewGuid(),
                        GroupId = groupObj.ID,
                        UserId = UserSession.Current.User.ID,
                        IsAdmin = false,
                        IsApproved = false,
                        JoinedDate = DateTime.Now
                    };
                    this.postGroupUserService.Insert(newPostGroupUser);
                    string msg = UserSession.Current.User.FirstName + " " + UserSession.Current.User.LastName + "
đã yêu cầu tham gia vào nhóm @" + groupObj.GroupName + " của bạn.";
                    SendNotificationToApp(UserSession.Current.User.ID, groupObj.Admin.GetValueOrDefault(), msg);
                }
                else if (groupObj.IsPrivate != true)
                {
                    PostGroupUser newPostGroupUser = new PostGroupUser()
                    {
                        ID = Guid.NewGuid(),
                        GroupId = groupObj.ID,
```

```
                    UserId = UserSession.Current.User.ID,
                    IsAdmin = false,
                    IsApproved = true,
                    JoinedDate = DateTime.Now
                };
                this.postGroupUserService.Insert(newPostGroupUser);
            }
            Response.Redirect(Request.RawUrl);
        }
    }
  }
}
```

6.2.8 File Login.aspx.cs

```
using Business.Services.Security;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Security.Cryptography;
using System.Text;
using System.Web;
using System.Web.Security;
using System.Web.UI;
using System.Web.UI.WebControls;
using Web.Sessions;
using Web.Template.Data;

namespace Web.Template.Security
{
  public partial class Login : System.Web.UI.Page
  {
    private readonly UserService userService;

    public Login()
```

```
    {
        this.userService = new UserService();
    }

    protected void Page_Load(object sender, EventArgs e)
    {
        txtEmail.Focus();
        if (!this.IsPostBack)
        { }
    }

    protected void btnLogin_Click(object sender, EventArgs e)
    {
        var user = userService.GetUserByEmail(this.txtEmail.Text);
        if (user != null)
        {
            if (user.Password == ComputeMD5(this.txtPassword.Text))
            {
                UserSession.CreateSession(user);
                FormsAuthentication.RedirectFromLoginPage(user.Username, false);
                Session["UserName"] = user.Username;
                if (Session["ReturnURL"] != null)
                {
                    var returnUrl = Session["ReturnURL"].ToString();
                    Session.Remove("ReturnURL");
                    Response.Redirect("~" + returnUrl);
                }
                else
                {
                    Response.Redirect("~/Default.aspx");
                }
            }
            else
            {
                Response.Redirect("/Controls/Security/Login.aspx");
            }
        }
    }

    static string ComputeMD5(string s)
    {
        using (MD5 md5 = MD5.Create())
        {
            return BitConverter.ToString(md5.ComputeHash(Encoding.UTF8.GetBytes(s)))
                    .Replace("-", "");
        }
    }
  }
}
```

6.2.9 File Signup.aspx.cs

```
using Business.Services.Security;
using Data.Entities;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Security.Cryptography;
using System.Text;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;

namespace Web.Template.Security
{
   public partial class Signup : System.Web.UI.Page
   {
      protected readonly UserService userService;

      public Signup()
      {
         this.userService = new UserService();
      }

      protected void Page_Load(object sender, EventArgs e)
      {
      }

      protected void btn_DangKy_Click(object sender, EventArgs e)
      {
         if (Page.IsValid)
         {
            var userList = this.userService.GetAll();
            bool mailCheck = false;
            foreach (User u in userList)
            {
               if (u.Email.ToString() != this.txtEmail.Text.Trim())
               {
                  mailCheck = false;
               }
               else
               {
                  mailCheck = true;
                  break;
               }
            }

            if (mailCheck == false)
            {
               User user = new User()
```

```
                {
                    ID = Guid.NewGuid(),
                    FirstName = this.txtFirstName.Text,
                    LastName = this.txtLastName.Text,
                    Email = this.txtEmail.Text,
                    Password = ComputeMD5(this.txtPassword.Text),
                    CreatedDate = DateTime.Now,
                    Avatar = "/noneAvatar.jpg",
                    Username = this.txtEmail.Text.Split('@')[0]
                };
                userService.Insert(user);
            }
            else
            {
                mailCheck = false;
            }
        }
    }

    static string ComputeMD5(string s)
    {
        using (MD5 md5 = MD5.Create())
        {
            return BitConverter.ToString(md5.ComputeHash(Encoding.UTF8.GetBytes(s)))
                    .Replace("-", "");
        }
    }
  }
}
```