

**TRƯỜNG ĐẠI HỌC GIAO THÔNG VẬN TẢI
PHÂN HIỆU TẠI TP. HỒ CHÍ MINH
BỘ MÔN CÔNG NGHỆ THÔNG TIN**



BÁO CÁO ĐỒ ÁN TỐT NGHIỆP

**ĐỀ TÀI: NGHIÊN CỨU CÁC PHƯƠNG PHÁP BẢO MẬT
THÔNG TIN VÀ ỨNG DỤNG XÂY DỰNG MẠNG XÃ HỘI
DUNG-SOCIAL**

Giảng viên hướng dẫn: Ths. TRẦN PHONG NHÃ

Sinh viên thực hiện: LÊ ĐOÀN HỮU DŨNG

Lớp : CÔNG NGHỆ PHẦN MỀM

Khoá : K56

Tp. Hồ Chí Minh, năm 2023

**TRƯỜNG ĐẠI HỌC GIAO THÔNG VẬN TẢI
PHÂN HIỆU TẠI TP. HỒ CHÍ MINH
BỘ MÔN CÔNG NGHỆ THÔNG TIN**



BÁO CÁO ĐỒ ÁN TỐT NGHIỆP

**ĐỀ TÀI: NGHIÊN CỨU CÁC PHƯƠNG PHÁP BẢO MẬT
THÔNG TIN VÀ ỨNG DỤNG XÂY DỰNG MẠNG XÃ HỘI
DUNG-SOCIAL**

Giảng viên hướng dẫn: Ths. TRẦN PHONG NHÃ

Sinh viên thực hiện: LÊ ĐOÀN HỮU DŨNG

Lớp : CÔNG NGHỆ PHẦN MỀM

Khoá : K56

Tp. Hồ Chí Minh, năm 2023

NHIỆM VỤ THIẾT KẾ TỐT NGHIỆP
BỘ MÔN: CÔNG NGHỆ THÔNG TIN
-----***-----

Mã sinh viên: **Họ tên SV:**

.....

Khóa: **Lớp:**

.....

1. Tên đề tài
2. Mục đích, yêu cầu
3. Nội dung và phạm vi đề tài
4. Công nghệ, công cụ và ngôn ngữ lập trình
5. Các kết quả chính dự kiến sẽ đạt được và ứng dụng
6. Giáo viên và cán bộ hướng dẫn

Họ tên:

Đơn vị công tác:

Điện thoại:

Email:

Ngày tháng 03 năm 2023
Trưởng BM Công nghệ Thông tin

Đã giao nhiệm vụ TKTN
Giáo viên hướng dẫn

ThS. Trần Phong Nhã

Đã nhận nhiệm vụ TKTN

Sinh viên:

Ký tên:

Điện thoại:

Email:

LỜI CẢM ƠN

Trước hết em xin bày tỏ sự kính trọng và lòng biết ơn sâu sắc nhất đến thầy giáo hướng dẫn: Ths. Trần Phong Nhã, thầy là người đã trực tiếp hướng dẫn, giúp đỡ cho em để em có thể hoàn thành đồ án này. Trong quá trình học tập và nghiên cứu, nếu em có những sai sót gì, kính mong thầy cô bỏ qua cho em!

Em xin kính chúc các thầy cô luôn luôn khỏe mạnh và ngày một thành công hơn trên con đường giảng dạy của mình.

Báo cáo này sẽ giới thiệu quá trình em phát triển trang mạng xã hội, từ việc nghiên cứu và thiết kế, đến quá trình phát triển và triển khai. Em cũng sẽ trình bày chi tiết về các tính năng chính của trang mạng xã hội, cũng như những thách thức mà em đã đối mặt và cách em đã giải quyết chúng.

Em hy vọng rằng trang mạng xã hội này sẽ mang lại nhiều giá trị cho người dùng, từ việc kết nối với bạn bè cũ, tìm kiếm người mới và thể hiện bản thân thông qua nội dung đa dạng. Em tin rằng trang mạng xã hội này có tiềm năng phát triển và trở thành một phần quan trọng của cuộc sống trực tuyến của cộng đồng người dùng.

Mong rằng báo cáo này sẽ giúp Thầy Cô có cái nhìn tổng quan về quá trình xây dựng một trang mạng xã hội và khám phá những tiềm năng và thách thức trong lĩnh vực này. Em rất mong được chia sẻ và trao đổi ý kiến để nâng cao và phát triển thêm nền tảng này trong tương lai.

NHẬN XÉT CỦA GIÁO VIÊN HƯỚNG DẪN

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

Tp. Hồ Chí Minh, ngày tháng năm
Giáo viên hướng dẫn

Trần Phong Nhã

MỤC LỤC

MỞ ĐẦU	11
1. Tổng quan về trang mạng xã hội.....	11
2. Mục tiêu và phạm vi nghiên cứu	11
3. Cấu trúc báo cáo tốt nghiệp	12
Chương I. CƠ SỞ LÝ THUYẾT	14
1. Nền tảng lập trình web và ngôn ngữ lập trình Typescript/Javascript.	14
1.1 Tổng quan về nền tảng lập trình web.	14
1.2 Giới thiệu về ngôn ngữ lập trình Typescript/Javascript.....	15
2. Framework Angular và vai trò của nó trong xây dựng trang mạng xã hội...	15
2.1 Lịch sử hình thành và phát triển của Framework Angular.	15
2.2 Tổng quan về Framework Angular.	16
3. Vai trò của Framework Angular trong xây dựng trang web.	17
4. Framework NestJS và ứng dụng trong hệ thống back end.	18
4.1 Lịch sử hình thành và phát triển của Framework NestJS.	18
4.2 Tổng quan về Framework NestJs.	19
4.3 Ứng dụng trong hệ thống back end.	20
5. Cơ sở dữ liệu PostgreSQL và lựa chọn cơ sở dữ liệu cho mạng xã hội.	21
5.1 Lịch sử hình thành và phát triển của cơ sở dữ liệu PostgreSQL.....	21
5.2 Tổng quan về cơ sở dữ liệu PostgreSQL.	22
CHƯƠNG 2. TRIỂN KHAI ĐỀ TÀI.....	24
1. Yêu cầu người dùng tiềm năng.....	24
2. Thiết kế giao diện và trải nghiệm người dùng	24
3. Kiến trúc hệ thống và cấu trúc dữ liệu	25
3.1 Kiến trúc hệ thống	25
3.2 Cấu trúc dữ liệu	27

4. Tính năng và chức năng chính	29
4.1 Đăng ký và đăng nhập.....	29
4.2 Tạo và quản lý hồ sơ cá nhân.	33
4.3 Kết nối và tương tác với bạn bè.....	36
4.4 Chia sẻ nội dung và bài đăng.....	37
4.5 Nhắn tin và giao tiếp.....	42
4.6 Kỹ thuật áp dụng	44
5. Thách thức và giải pháp.....	53
5.1 Bảo mật và quản lý dữ liệu cá nhân.....	53
5.2 Nhắn tin thời gian thực giữa người dùng với nhau	54
KẾT QUẢ.....	56
TÀI LIỆU THAM KHẢO.....	57

BẢNG BIỂU, SƠ ĐỒ, HÌNH VẼ

Hình 1.1: Mô hình cấu trúc của Framework Angular.....	17
Hình 1.2: Mô hình kiến trúc của Framework NestJs.....	20
Hình 2.1: Sơ đồ mô tả kiến trúc hệ thống.....	26
Hình 2.2: Sơ đồ quan hệ cấu trúc dữ liệu	28
Hình 2.3: Màn hình đăng ký tài khoản	30
Hình 2.4: Trường hợp các trường dữ liệu không hợp lệ.....	30
Hình 2.5: Rê chuột vào dấu “X” để xem nguyên nhân.....	31
Hình 2.6: Mật khẩu phải có độ phức tạp nhất định nhằm đảm bảo tính bảo mật cao	31
Hình 2.7: Trường hợp tất cả các trường dữ liệu đã hợp lệ.	32
Hình 2.8: Màn hình đăng nhập	32
Hình 2.9: Trường hợp các trường dữ liệu không hợp lệ.....	33
Hình 2.10: Màn hình đăng nhập có recaptcha.....	33
Hình 2.11: Màn hình quản lý hồ sơ cá nhân.....	34
Hình 2.12: Màn hình quản lý hồ sơ cá nhân sau khi đã cập nhật	34
Hình 2.13: Hộp thoại cập nhật hình đại diện.....	35
Hình 2.14: Hình ảnh đại diện sau khi thực hiện cập nhật.....	35
Hình 2.15: Màn hình thay đổi mật khẩu hiện tại.....	36
Hình 2.16: Trang kết nối bạn bè.....	36
Hình 2.17: Chọn theo dõi để theo dõi người bạn đó hoặc tái nhấn để huỷ theo dõi.	37
Hình 2.18: Danh sách bài đăng của những người bạn đã theo dõi.	37
Hình 2.19: Nhấn vào trường “Bạn đang nghĩ gì thế” để mở hộp thoại đăng bài	38
Hình 2.20: Hộp thoại tạo bài viết mới.	38
Hình 2.21: Người dùng soạn thảo bài viết và chèn hình ảnh.	39
Hình 2.22: Danh sách các bài đăng.	40
Hình 2.23: Thông báo về bài viết mới từ những người bạn	40
Hình 2.24: Người dùng tương tác trên bài viết.....	41
Hình 2.25: Chi tiết bài đăng, xóa hay hiệu chỉnh bài đăng	41
Hình 2.26: Hộp thoại hiệu chỉnh bài viết.....	42
Hình 2.27: Giao diện trang nhấn tin	43
Hình 2.28: Ví dụ về đoạn hội thoại giữa hai người dùng.	44

Ghi chú:

- Xếp sau trang Mục lục
- Chữ số thứ nhất chỉ tên chương
- Chữ số thứ hai chỉ thứ tự bảng biểu, sơ đồ, hình,...trong mỗi chương
- Ở cuối mỗi bảng biểu, sơ đồ, hình,...trong mỗi chương phải có ghi chú, giải thích, nêu rõ nguồn trích hoặc sao chụp,...

MỞ ĐẦU

1. Tổng quan về trang mạng xã hội

Mạng xã hội là một hệ thống trực tuyến hoặc nền tảng kỹ thuật số mà người dùng có thể tạo hồ sơ cá nhân, chia sẻ thông tin, tương tác và kết nối với người khác thông qua một loạt các chức năng và công cụ. Nó tạo ra một không gian kỹ thuật số cho cá nhân, cộng đồng, tổ chức và doanh nghiệp để giao tiếp, chia sẻ nội dung, xây dựng mối quan hệ và tham gia vào các hoạt động mạng xã hội.

Mạng xã hội cho phép người dùng tạo hồ sơ cá nhân, trong đó họ có thể cung cấp thông tin về bản thân, sở thích, sự nghiệp và hình ảnh cá nhân. Người dùng có thể chia sẻ nội dung, bao gồm bài đăng văn bản, hình ảnh, video và liên kết, và các thành viên khác có thể tương tác với nội dung này bằng cách bình luận, like, chia sẻ hoặc reblog.

Mạng xã hội cung cấp các chức năng tương tác để người dùng có thể tìm kiếm, kết bạn, theo dõi hoặc kết nối với những người khác có sở thích tương tự. Nó cung cấp một cơ chế để người dùng gửi tin nhắn, tham gia vào các nhóm hoặc cộng đồng, và tham gia vào các sự kiện hoặc nhóm quan tâm chung.

Mạng xã hội cũng đóng vai trò là nền tảng cho việc trao đổi thông tin, tạo ra mối quan hệ cá nhân hoặc chuyên nghiệp, xây dựng thương hiệu cá nhân hoặc doanh nghiệp, và tham gia vào các hoạt động xã hội, văn hóa hoặc chính trị. Nó có thể mang lại lợi ích kết nối, tạo mạng lưới kinh doanh, thúc đẩy sự phát triển cá nhân và mở ra những cơ hội mới.

Tổng thể, mạng xã hội tạo ra một môi trường kỹ thuật số cho con người tương tác, giao tiếp và chia sẻ thông tin trong một cộng đồng trực tuyến, mở ra những khả năng kết nối và tương tác mà trước đây chỉ có thể thực hiện trong thế giới thực.

2. Mục tiêu và phạm vi nghiên cứu

Mục tiêu chung mà em hướng tới là tạo ra một nền tảng trực tuyến thu hút người dùng và tạo ra môi trường kết nối, tương tác giữa các thành viên. Dưới đây là một số mục tiêu cụ thể mà em đã đặt ra khi làm đề tài.

Xây dựng giao diện người dùng hấp dẫn: Một trong những mục tiêu chính là tạo ra một giao diện người dùng thân thiện, tương tác và hấp dẫn. Điều này đòi hỏi sự chú trọng đến thiết kế UI/UX, bố cục trang, màu sắc, biểu đồ và các yếu tố khác để tạo ra trải nghiệm người dùng tốt nhất.

Xây dựng hệ thống đăng ký và đăng nhập: Đảm bảo rằng người dùng có thể dễ dàng đăng ký và đăng nhập vào trang mạng xã hội. Hệ thống này cần cung cấp các phương thức xác thực an toàn và bảo mật, như đăng ký bằng email, xác minh số điện thoại hoặc sử dụng tài khoản mạng xã hội khác.

Xây dựng chức năng tương tác và kết nối: Tạo ra các chức năng cho phép người dùng tương tác với nhau, như kết bạn, gửi tin nhắn, thả tim hoặc bình luận. Đồng thời, cung cấp các công cụ tìm kiếm và gợi ý người dùng khác nhằm mở rộng mạng lưới kết nối.

Quản lý nội dung và bảo vệ quyền riêng tư: Xây dựng hệ thống quản lý nội dung để người dùng có thể chia sẻ thông tin, hình ảnh và video một cách an toàn. Đảm bảo bảo vệ quyền riêng tư của người dùng và cung cấp cơ chế kiểm soát quyền riêng tư, như cài đặt riêng tư cho từng bài đăng hoặc phân quyền truy cập.

Tối ưu hóa hiệu suất và mở rộng: Đảm bảo rằng trang mạng xã hội có khả năng xử lý lượng dữ liệu lớn và có thể mở rộng để đáp ứng số lượng người dùng tăng lên. Tối ưu hóa hiệu suất và tăng cường bảo mật để đảm bảo trang web hoạt động ổn định và an toàn.

Phạm vi nghiên cứu của đề tài em sẽ tập trung vào xây dựng một trang mạng xã hội cơ bản, tập trung vào các chức năng cốt lõi như giao diện người dùng, đăng ký và đăng nhập, tương tác và kết nối, quản lý nội dung và bảo vệ quyền riêng tư, cũng như hiệu suất và mở rộng. Trong phạm vi hạn chế của đồ án tốt nghiệp, em sẽ tập trung vào những yếu tố quan trọng này để tạo ra một trang mạng xã hội đơn giản và hoạt động hiệu quả.

3. Cấu trúc báo cáo tốt nghiệp

3.1. Chương 1: Trình bày cơ sở lý thuyết của đồ án

3.2. Chương 2: Ứng dụng cơ sở lý thuyết và triển khai đề tài

3.3. Chương 3: Trình bày kết quả đề tài

Chương I. CƠ SỞ LÝ THUYẾT

1. Nền tảng lập trình web và ngôn ngữ lập trình Typescript/Javascript.

1.1 Tổng quan về nền tảng lập trình web.

Nền tảng lập trình web là một hệ thống kỹ thuật và công nghệ được sử dụng để xây dựng và phát triển các ứng dụng web. Nó cung cấp các công cụ, ngôn ngữ lập trình, framework và thư viện để tạo ra các trang web, ứng dụng web và dịch vụ trên Internet. Dưới đây là một tổng quan về các thành phần chính trong nền tảng lập trình web:

Ngôn ngữ lập trình: Các ngôn ngữ lập trình phổ biến trong lập trình web bao gồm HTML (HyperText Markup Language), CSS (Cascading Style Sheets) và JavaScript. HTML được sử dụng để định dạng và cấu trúc nội dung trên trang web, CSS dùng để xác định giao diện và kiểu dáng của trang, còn JavaScript là ngôn ngữ lập trình được sử dụng để xử lý sự kiện và tương tác trên trang web.

Framework và thư viện: Có nhiều framework và thư viện được sử dụng trong lập trình web để giúp xây dựng ứng dụng nhanh chóng và hiệu quả. Các framework phổ biến như Ruby on Rails, Django, Laravel, Express.js cung cấp cấu trúc và quy tắc phát triển để tạo ra ứng dụng web. Thư viện như React, Angular và Vue.js cung cấp các công cụ và thành phần để xây dựng giao diện người dùng tương tác.

Cơ sở dữ liệu: Lập trình web thường liên quan đến việc lưu trữ và truy xuất dữ liệu từ cơ sở dữ liệu. Các hệ quản trị cơ sở dữ liệu phổ biến như MySQL, PostgreSQL và MongoDB được sử dụng để quản lý dữ liệu trong ứng dụng web.

Giao thức và tiêu chuẩn: Các giao thức và tiêu chuẩn web quan trọng bao gồm HTTP (Hypertext Transfer Protocol) để truyền tải dữ liệu qua mạng, HTML và CSS để định dạng và hiển thị trang web, và JavaScript để tương tác và xử lý trên trình duyệt.

Mô hình phát triển: Có nhiều mô hình phát triển phổ biến trong lập trình web như mô hình Client-Server, mô hình 3-lớp và mô hình MVC (Model-View-Controller). Mô hình phát triển giúp tổ chức và quản lý codebase, tách biệt logic xử lý, hiển thị và dữ liệu.

Công cụ phát triển: Có nhiều công cụ hỗ trợ lập trình web như trình duyệt web và công cụ kiểm tra, trình biên dịch và gỡ lỗi, trình quản lý phiên bản và trình tự động hóa việc triển khai ứng dụng.

1.2 Giới thiệu về ngôn ngữ lập trình Typescript/Javascript.

JavaScript được tạo ra bởi Brendan Eich tại Netscape vào năm 1995. Ban đầu, nó được thiết kế để xử lý các sự kiện trên trình duyệt web Netscape Navigator. Nhưng sau đó nó nhanh chóng trở thành một ngôn ngữ lập trình phía client phổ biến, cho phép tạo ra các hiệu ứng động và tương tác trực tiếp trên trang web. Năm 1997, ECMAScript (tiêu chuẩn JavaScript) được ra đời để định nghĩa cú pháp và quy tắc phát triển của ngôn ngữ này. Các phiên bản ECMAScript tiếp theo (ES2, ES3, ES5) giúp nâng cao tính năng và khả năng của JavaScript. Cho đến hiện tại phiên bản mới nhất của ECMAScript đã là ES11.

TypeScript được phát triển bởi Microsoft và công bố lần đầu vào tháng 10 năm 2012. Anders Hejlsberg, người đã tham gia vào việc tạo ra ngôn ngữ C# và Turbo Pascal, đứng đầu dự án.

TypeScript là một superset của JavaScript, nghĩa là nó bao gồm tất cả các tính năng của JavaScript và mở rộng thêm một số tính năng mới. Nó được thiết kế với mục tiêu giúp phát triển ứng dụng JavaScript lớn hơn, dễ bảo trì và mở rộng hơn. Nó cung cấp kiểu dữ liệu tĩnh, lập trình hướng đối tượng và các tính năng khác để tăng tính đáng tin cậy và quản lý mã nguồn dễ dàng hơn. TypeScript sử dụng trình biên dịch để chuyển đổi mã nguồn TypeScript thành JavaScript trước khi chạy trên trình duyệt hoặc server-side.

Kể từ khi ra mắt, TypeScript đã được đánh giá cao và được sử dụng rộng rãi trong cộng đồng phát triển. Microsoft đã tiếp tục phát triển TypeScript, cung cấp cập nhật thường xuyên và hỗ trợ mạnh mẽ cho công cụ này.

2. Framework Angular và vai trò của nó trong xây dựng trang mạng xã hội.

2.1 Lịch sử hình thành và phát triển của Framework Angular.

Ban đầu AngularJS là một thư viện Javascript được ra mắt vào năm 2010. Nó được xây dựng nhằm hỗ trợ cho việc phát triển web động. AngularJS sử dụng

mô hình thiết kế MVC (Model-View-Controller) và cung cấp các tính năng như two-way data binding, dependency injection, và routing.

Kể từ phiên bản Angular 2 ra mắt vào năm 2016. Nó đã dần thay đổi để trở thành một framework với những sự thay đổi lớn so với AngularJS bao gồm việc sử dụng TypeScript làm ngôn ngữ chính, kiến trúc component, và mô hình Reactive Programming với RxJS. Đồng thời từ Angular 2 trở đi, Angular được đánh số phiên bản theo quy tắc semantic versioning, với các phiên bản nhỏ (minor) và sửa lỗi (patch) được cung cấp thường xuyên.

Với các phiên bản Angular 4 đến Angular 9 ra mắt trong giai đoạn năm 2017 đến năm 2019 mang đến nhiều cải tiến và tính năng mới cho Angular. Mỗi phiên bản tiếp theo cung cấp cải tiến hiệu suất, khả năng tương thích với các phiên bản trình duyệt mới, và cải thiện trải nghiệm phát triển.

Trong giai đoạn từ năm 2020 đến năm 2021. Angular cũng đã có thêm 3 phiên bản từ Angular 10 đến Angular 12. Các phiên bản này tiếp tục tập trung vào việc cải thiện hiệu suất, sửa lỗi và cung cấp các tính năng mới. Đáng chú ý là ở phiên bản Angular 10 đã đưa ra một cách tiếp cận được cải tiến về gói ngược (backward compatibility) và tăng cường tích hợp với các công cụ phát triển khác.

2.2 Tổng quan về Framework Angular.

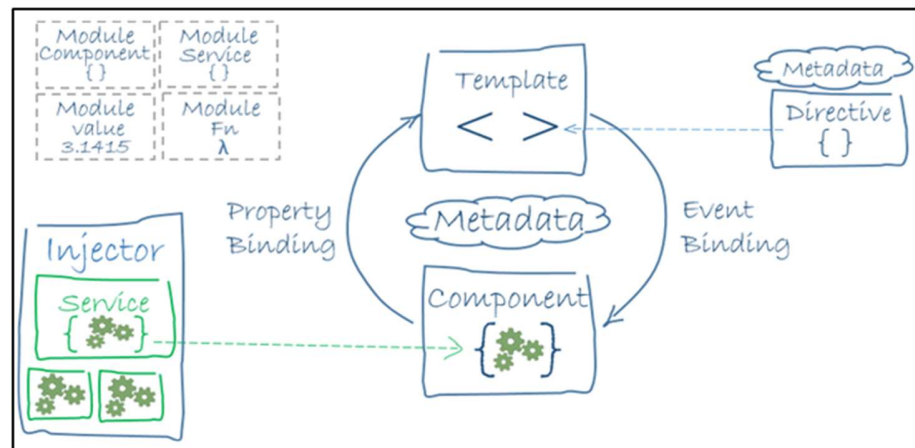
Sau đây em xin giới thiệu tổng quan về Framework Angular. Là Framework em đã sử dụng để xây dựng nên đề tài này.

Kiến trúc component: Angular sử dụng kiến trúc component để phân tách giao diện người dùng thành các thành phần độc lập và tái sử dụng được gọi là component. Mỗi component bao gồm HTML, CSS và TypeScript, tạo thành một phần của giao diện người dùng.

Dependency Injection: Angular có hệ thống Dependency Injection (DI) mạnh mẽ. Điều này giúp quản lý và cung cấp các thành phần phụ thuộc cho các thành phần khác trong ứng dụng một cách dễ dàng. DI cung cấp khả năng tạo và quản lý các instance của các service và module trong ứng dụng.

Routing: Angular cung cấp một hệ thống routing để quản lý việc điều hướng giữa các trang và thành phần trong ứng dụng. Điều này giúp xây dựng các ứng dụng đơn trang có khả năng điều hướng linh hoạt và thân thiện với người dùng.

Testing: Angular cung cấp các công cụ và thư viện mạnh mẽ để viết và chạy các bài kiểm tra (testing) cho ứng dụng. Bằng cách viết các bài kiểm tra tự động, người phát triển có thể đảm bảo rằng ứng dụng hoạt động đúng và không có lỗi.



Hình 1.1: Mô hình cấu trúc của Framework Angular

Với các tính năng mạnh mẽ, Angular là một framework phát triển ứng dụng web đáng tin cậy và phổ biến. Nó giúp xây dựng các ứng dụng web phức tạp và mạnh mẽ, đồng thời tạo điều kiện thuận lợi cho việc bảo trì và phát triển dự án.

3. Vai trò của Framework Angular trong xây dựng trang web.

Angular là một framework phát triển ứng dụng web mạnh mẽ và phổ biến. Nó được sử dụng để xây dựng các ứng dụng web đơn trang (Single Page Applications - SPA) và ứng dụng di động sử dụng HTML, CSS và JavaScript/TypeScript.

Angular định nghĩa một cấu trúc dựa trên thành phần (component-based) giúp tổ chức code dễ dàng và rõ ràng. Nó khuyến khích việc phân chia ứng dụng thành các thành phần nhỏ, có khả năng tái sử dụng và dễ bảo trì.

Cung cấp một hệ thống liên kết dữ liệu (data binding) mạnh mẽ, cho phép bạn đồng bộ hóa dữ liệu giữa các thành phần và giao diện người dùng. Điều này giúp làm giảm sự phức tạp của việc quản lý trạng thái ứng dụng và tăng hiệu suất phát triển.

Hệ thống routing linh hoạt, cho phép bạn xác định các tuyến đường (routes) và quản lý điều hướng trong ứng dụng web. Điều này giúp tạo ra các trang độc lập và quản lý điều hướng giữa chúng một cách dễ dàng.

Angular cung cấp một công cụ quản lý trạng thái ứng dụng gọi là Redux hoặc nâng cao hơn là NgRx. Điều này giúp quản lý trạng thái ứng dụng một cách hiệu quả, đồng bộ hóa dữ liệu và trạng thái giữa các thành phần khác nhau.

Cung cấp các công cụ và thư viện tích hợp sẵn giúp việc kiểm thử (testing) và tối ưu hóa ứng dụng trở nên thuận tiện hơn. Nó cung cấp các công cụ như Karma và Protractor để kiểm thử tự động và Angular CLI để tạo, xây dựng và triển khai ứng dụng một cách dễ dàng.

Cho phép xây dựng ứng dụng web đáp ứng (responsive) và ứng dụng di động sử dụng một mã nguồn duy nhất. Điều này giúp giảm công sức và thời gian phát triển khi muốn triển khai ứng dụng trên nhiều nền tảng khác nhau.

Với những vai trò trên, Angular là một framework rất phổ biến trong việc xây dựng ứng dụng web chuyên nghiệp và có hiệu suất cao. Nó giúp cải thiện quy trình phát triển, tăng khả năng bảo trì và mang lại trải nghiệm người dùng tốt hơn.

4. Framework NestJS và ứng dụng trong hệ thống back end.

4.1 Lịch sử hình thành và phát triển của Framework NestJS.

NestJS là một framework phát triển ứng dụng server-side phía Node.js được phát triển bởi Kamil Myśliwiec và cộng đồng của mình. Nó được giới thiệu lần đầu vào năm 2017. Về lý do tại sao ông lại phát triển NestJS. Ông nhận thấy rằng việc phát triển ứng dụng Nodejs trong một cấu trúc tốt và duy trì mã nguồn dễ dàng là một thách thức và quyết định xây dựng NestJS để giải quyết vấn đề này. NestJS lấy cảm hứng từ các framework phổ biến như Angular và Express.js. Nó kết hợp các khái niệm của Angular như Dependency Injection, Modules và Providers, cùng với cấu trúc cú pháp và cú pháp decorator trong TypeScript.

Lịch sử phát triển của NestJS đã trải qua các phiên bản và cập nhật liên tục để cung cấp tính năng mới và cải thiện hiệu suất. Các phiên bản chính của NestJS bao gồm:

NestJS 4.x: Ra mắt vào tháng 1 năm 2018, phiên bản này cung cấp nhiều cải tiến và bổ sung tính năng mới, như khả năng kiểm tra cấu hình, bộ lọc middleware và bộ ghi nhật ký.

NestJS 5.x: Ra mắt vào tháng 6 năm 2018, phiên bản này tập trung vào việc tăng cường hiệu suất và cải thiện khả năng kiểm tra (testing) bằng việc giới thiệu trình mock mới.

NestJS 6.x: Ra mắt vào tháng 10 năm 2018, phiên bản này tập trung vào việc tối ưu hóa và cải thiện hiệu suất của framework.

NestJS 7.x: Ra mắt vào tháng 11 năm 2019, phiên bản này giới thiệu một số cải tiến quan trọng, bao gồm hỗ trợ HTTP/2, WebSocket và gRPC.

NestJS 8.x: Ra mắt vào tháng 6 năm 2021, phiên bản này cung cấp các tính năng mới như Event Emitters, Caching, và Health Checks.

NestJS tiếp tục được phát triển và có một cộng đồng lớn và đam mê. Framework này đang trở thành một trong những lựa chọn phổ biến cho việc xây dựng các ứng dụng server-side phía Node.js, đặc biệt trong các dự án lớn và phức tạp.

4.2 Tổng quan về Framework NestJs.

NestJS là một framework phát triển ứng dụng server-side phía Node.js, được xây dựng dựa trên TypeScript và lấy cảm hứng từ Angular. Nó cung cấp một cấu trúc ứng dụng rõ ràng và mô-đun hóa, giúp tạo ra mã nguồn dễ bảo trì, mở rộng và kiểm thử. Nó được xem là một trong những framework phát triển ứng dụng Node.js phía server phổ biến và mạnh mẽ, và nó đang ngày càng thu hút sự quan tâm từ cộng đồng phát triển.

Dưới đây là một số điểm nổi bật và tổng quan về NestJS:

TypeScript: Tương tự Framework Angular, NestJs cũng lựa chọn Typescript để xây dựng và phát triển.

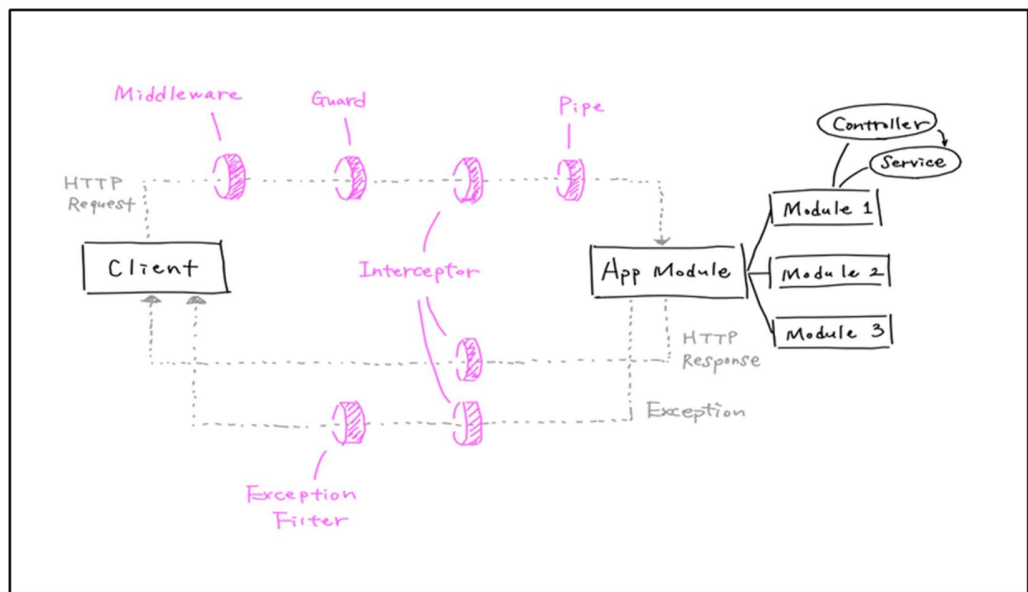
Kiến trúc module: NestJS sử dụng kiến trúc module để phân chia ứng dụng thành các phần nhỏ hơn, gọi là modules. Mỗi module có các thành phần như controllers, providers (services), và dependencies. Kiến trúc module giúp tổ chức mã nguồn một cách rõ ràng và dễ dàng tái sử dụng.

Dependency Injection (DI): NestJS sử dụng Dependency Injection để quản lý các phụ thuộc và cung cấp chúng cho các thành phần khác trong ứng dụng. DI giúp giảm sự phụ thuộc cứng rắ giữa các thành phần và tạo điều kiện thuận lợi cho việc kiểm thử và bảo trì.

Decorators và Metadata: NestJS sử dụng decorators để đánh dấu các thành phần như controllers, routes, middleware và providers. Decorators giúp xác định mục đích và chức năng của các thành phần này, đồng thời hỗ trợ sự tương tác với framework.

Middleware: NestJS hỗ trợ middleware để xử lý các yêu cầu trước khi chúng được chuyển đến các controller. Middleware giúp xử lý xác thực, ghi nhật ký, xử lý lỗi và các tác vụ trung gian khác.

Công cụ kiểm thử (Testing): NestJS cung cấp một loạt các công cụ và thư viện để kiểm thử ứng dụng một cách dễ dàng. Framework này hỗ trợ việc viết các bài kiểm tra tự động (automated testing) và cung cấp các phương pháp tiếp cận cho việc kiểm thử đơn vị (unit testing) và kiểm thử tích hợp (integration testing).



Hình 1.2: Mô hình kiến trúc của Framework NestJs

4.3 Ứng dụng trong hệ thống back end.

Vai trò của NestJS trong xây dựng hệ thống Backend (BE) là cung cấp một framework mạnh mẽ và hiệu quả cho việc phát triển ứng dụng server-side.

NestJS cung cấp một kiến trúc ứng dụng gọn gàng và có tổ chức, dựa trên nguyên tắc của kiến trúc ứng dụng hướng đối tượng (OOP) và kiến trúc ứng dụng lớp nền tảng (Layered Architecture). Điều này giúp dễ dàng tổ chức code, tái sử dụng các thành phần và bảo trì hệ thống BE.

Hỗ trợ Dependency Injection để quản lý các phụ thuộc (dependencies) giữa các thành phần của ứng dụng. Điều này giúp giảm sự rối rắm trong việc quản lý các phụ thuộc, tạo điều kiện cho việc kiểm thử và mở rộng ứng dụng.

NestJS cung cấp tích hợp sẵn các module cho việc thao tác với cơ sở dữ liệu, bao gồm các hệ quản trị cơ sở dữ liệu phổ biến như MySQL, PostgreSQL, MongoDB và nhiều loại cơ sở dữ liệu khác. Điều này giúp tăng tính linh hoạt và khả năng mở rộng của hệ thống BE.

Sử dụng Middleware và Interceptors để xử lý các yêu cầu HTTP và logic xử lý trung gian. Điều này cho phép thực hiện các thao tác chung trước và sau khi xử lý các yêu cầu, chẳng hạn như xác thực người dùng, ghi log, kiểm tra lỗi và nhiều tác vụ khác.

NestJS hỗ trợ sử dụng WebSocket để xây dựng các ứng dụng thời gian thực (real-time) như trò chuyện trực tuyến, thông báo sống và các ứng dụng theo thời gian thực khác. Điều này cho phép giao tiếp hai chiều giữa máy khách (client) và máy chủ (server) một cách liên tục và không đồng bộ.

Với công cụ và thư viện tích hợp sẵn cho việc kiểm thử và gỡ lỗi ứng dụng BE. Với việc hỗ trợ tốt cho việc viết các bài kiểm tra tự động (automated testing) và cung cấp các công cụ mạnh mẽ như Jest, Supertest và Swagger, NestJS giúp đảm bảo chất lượng và độ tin cậy của hệ thống.

5. Cơ sở dữ liệu PostgreSQL và lựa chọn cơ sở dữ liệu cho mạng xã hội.

5.1 Lịch sử hình thành và phát triển của cơ sở dữ liệu PostgreSQL.

PostgreSQL là một hệ quản trị cơ sở dữ liệu quan hệ mã nguồn mở (open source) được phát triển bởi một cộng đồng các nhà phát triển và được duy trì bởi PostgreSQL Global Development Group.

Năm 1986 tại đại học California, Berkeley bắt đầu dự án POSTGRES dưới sự lãnh đạo của Michael Stonebraker và các thành viên khác. Mục tiêu của dự án này là phát triển một hệ quản trị cơ sở dữ liệu quan hệ mới, có khả năng mở rộng và hỗ trợ tính toán phức tạp.

Năm 1989 PostgreSQL đã đạt được sự hoàn thiện và sẵn sàng cho việc sử dụng trong môi trường thực tế. Tuy nhiên, dự án PostgreSQL không được phát hành công khai và chỉ sử dụng trong các dự án nghiên cứu.

Năm 1994 Sau khi hoàn thành nghiên cứu và phát triển PostgreSQL, phiên bản đầu tiên của PostgreSQL đã được phát hành. PostgreSQL kế thừa nhiều tính năng của PostgreSQL và được phát hành dưới giấy phép mã nguồn mở.

Các phiên bản tiếp theo của PostgreSQL (2.0, 3.0, 4.0) đã được phát triển và cung cấp nhiều tính năng và cải tiến mới. PostgreSQL trở thành một hệ quản trị cơ sở dữ liệu quan hệ mạnh mẽ và linh hoạt, đồng thời hỗ trợ một loạt các tính năng tiên tiến như cấu trúc mở rộng, giao dịch, khóa và xử lý ngoại lệ.

Năm 2005 phiên bản PostgreSQL 8.0 được phát hành với nhiều cải tiến đáng kể bao gồm truy vấn chủ động (query optimization), hỗ trợ truy vấn địa lý và các tính năng mới khác.

Từ năm 2007 trở đi PostgreSQL tiếp tục phát triển và phát hành các phiên bản mới với nhiều cải tiến và tính năng mới, bao gồm khả năng replication, bảo mật tăng cường, hỗ trợ JSON và các tính năng NoSQL.

Đến hiện tại PostgreSQL là một trong những hệ quản trị cơ sở dữ liệu quan hệ phổ biến nhất trên thế giới và được sử dụng rộng rãi trong các ứng dụng doanh nghiệp và dự án phát triển phần mềm. Cộng đồng PostgreSQL phát triển và duy trì sự tiến bộ liên tục của hệ quản trị cơ sở dữ liệu này.

5.2 Tổng quan về cơ sở dữ liệu PostgreSQL.

PostgreSQL là một hệ quản trị cơ sở dữ liệu (Database Management System - DBMS) mã nguồn mở và được phát triển từ năm 1986. Đây là một hệ quản trị cơ sở dữ liệu quan hệ (Relational Database Management System - RDBMS) mạnh mẽ và đáng tin cậy, được rất nhiều tổ chức và doanh nghiệp lựa chọn sử dụng.

Đặc điểm chính của cơ sở dữ liệu PostgreSQL là hỗ trợ các tính năng cơ bản của cơ sở dữ liệu quan hệ như bảng, quan hệ, khóa ngoại, ràng buộc, giao dịch, và các nguyên tắc ACID (Atomicity, Consistency, Isolation, Durability). Hỗ trợ các tính năng mở rộng như truy vấn phức tạp, chức năng lưu trữ (stored procedures), ghi log, điều khiển truy cập, và phân quyền. Hỗ trợ các kiểu dữ liệu đa dạng bao gồm số, văn bản, ngày tháng, địa lý, hình ảnh, âm thanh và video. Hỗ trợ các công nghệ mở như JSON, XML, và đặc biệt hỗ trợ tốt cho việc làm việc với dữ liệu địa lý và dữ liệu ngôn ngữ tự nhiên. Hỗ trợ mô hình mở rộng (scalability) với khả năng xử lý tải cao và phân tán dữ liệu. Đồng thời PostgreSQL hỗ trợ đa nền tảng, có sẵn trên nhiều nền tảng hệ điều hành như Windows, Linux, macOS, và Unix.

Về độ tin cậy và bảo mật, PostgreSQL có kiến trúc bền vững và đáng tin cậy, với khả năng tự phục hồi và khả năng chống chịu lỗi.

Hỗ trợ các biện pháp bảo mật mạnh mẽ như kiểm soát truy cập, mã hóa dữ liệu, xác thực người dùng, và đăng nhập an toàn.

PostgreSQL có một cộng đồng lớn và nhiều người dùng trên toàn cầu, cung cấp sự hỗ trợ, tư vấn và chia sẻ kiến thức. Được phát triển và duy trì bởi một cộng đồng mở, PostgreSQL có sự phát triển liên tục và cập nhật thường xuyên.

Với những đặc điểm và ưu điểm trên, PostgreSQL được coi là một giải pháp mạnh mẽ và đáng tin cậy cho việc lưu trữ và quản lý cơ sở dữ liệu trong các ứng dụng doanh nghiệp và phần mềm quan trọng. Đó cũng là lý do em lựa chọn PostgreSQL làm cơ sở dữ liệu cho đề tài.

CHƯƠNG 2. TRIỂN KHAI ĐỀ TÀI

1. Yêu cầu người dùng tiềm năng.

Yêu cầu của người dùng tiềm năng cho một trang mạng xã hội có thể bao gồm các yêu cầu chung và yêu cầu cụ thể tùy thuộc vào mục đích và đặc điểm cụ thể của trang mạng xã hội. Dưới đây em xin nêu ra một số yêu cầu tiềm năng mà người dùng có thể mong đợi từ một trang mạng xã hội:

Tương tác xã hội: Người dùng muốn có khả năng tương tác và kết nối với những người khác trên mạng xã hội. Điều này có thể bao gồm kết bạn, theo dõi, gửi tin nhắn, bình luận, và chia sẻ nội dung với nhau.

Tạo và quản lý hồ sơ cá nhân: Người dùng muốn có khả năng tạo và quản lý hồ sơ cá nhân của mình trên trang mạng xã hội. Họ muốn có thể thêm thông tin cá nhân, hình ảnh, sở thích, và thông tin liên hệ để chia sẻ với người khác.

Bảo mật và quyền riêng tư: Người dùng quan tâm đến bảo mật và quyền riêng tư trên trang mạng xã hội. Họ muốn có khả năng kiểm soát ai có thể truy cập và xem thông tin cá nhân của mình, và có khả năng quản lý các cài đặt riêng tư như người có thể nhìn thấy nội dung, ai có thể liên hệ với họ, và ai có thể theo dõi hoạt động của họ.

Chia sẻ và khám phá nội dung: Người dùng mong muốn có khả năng chia sẻ nội dung như hình ảnh, video, bài viết, và liên kết với những người khác trên mạng xã hội. Họ cũng muốn có khả năng khám phá và tìm kiếm nội dung mới từ người dùng khác hoặc từ các trang và nhóm quan tâm.

Nhóm và cộng đồng: Người dùng có nhu cầu tham gia vào các nhóm và cộng đồng có quan tâm chung trên mạng xã hội. Họ muốn có khả năng tham gia thảo luận, chia sẻ thông tin, và kết nối với những người có sở thích tương tự.

Thông báo và thông tin cập nhật: Người dùng mong muốn nhận thông báo và thông tin cập nhật về hoạt động của họ trên trang mạng xã hội, bao gồm lời nhắn, nhắc nhở, bài viết mới từ người mà họ theo dõi, hoạt động trong nhóm, và sự kiện quan trọng.

Đa nền tảng và tiện ích di động: Người dùng mong muốn trang mạng xã hội có khả năng hoạt động trên nhiều nền tảng và thiết bị, bao gồm máy tính để bàn, điện thoại di động và máy tính bảng. Họ muốn có ứng dụng di động tiện ích để truy cập và sử dụng trang mạng xã hội mọi lúc, mọi nơi.

Trên đây chỉ là một số yêu cầu tiềm năng phổ biến từ người dùng. Tùy thuộc vào đặc điểm và mục tiêu cụ thể của trang mạng xã hội, yêu cầu có thể khác nhau.

2. Thiết kế giao diện và trải nghiệm người dùng

Thiết kế giao diện và trải nghiệm người dùng (User Experience - UX) là yếu tố quan trọng trong việc tạo ra một trang mạng xã hội hấp dẫn và thu hút người dùng. Dựa trên những yêu cầu của người dùng tiềm năng cũng như tham khảo từ các trang mạng xã hội phổ biến hiện nay em xin nêu ra một số ý kiến cá nhân về thiết kế giao diện người dùng cũng như từ đó hướng đi cho quá trình em xây dựng giao diện cho trang mạng xã hội của mình.

Thiết kế đơn giản và giao diện thân thiện: Tạo ra một giao diện đơn giản, dễ sử dụng và trực quan để người dùng có thể dễ dàng tìm hiểu và sử dụng trang mạng xã hội. Sử dụng màu sắc và hình ảnh hài hòa, hợp lý để tạo sự thu hút và tăng tính tương tác.

Tạo trải nghiệm cá nhân hóa: Cho phép người dùng tạo và quản lý hồ sơ cá nhân của mình với các thông tin, hình ảnh và sở thích riêng. Cung cấp khả năng tùy chỉnh các cài đặt cá nhân và lựa chọn riêng để người dùng có thể điều chỉnh trang mạng xã hội theo ý thích của họ.

Tạo khả năng tương tác và kết nối: Đưa ra các tính năng tương tác như kết bạn, theo dõi, bình luận, chia sẻ và đánh giá để người dùng có thể kết nối và tương tác với nhau. Tạo ra khả năng gửi tin nhắn riêng tư hoặc công khai giữa người dùng để tạo sự giao tiếp và trao đổi thông tin.

Đảm bảo tính linh hoạt và đa nền tảng: Thiết kế trang mạng xã hội sao cho nó có khả năng hoạt động trên nhiều thiết bị và nền tảng, bao gồm máy tính để bàn, điện thoại di động và máy tính bảng. Tối ưu hóa trang mạng xã hội để phù hợp với các màn hình khác nhau và đảm bảo trải nghiệm tốt trên mọi nền tảng.

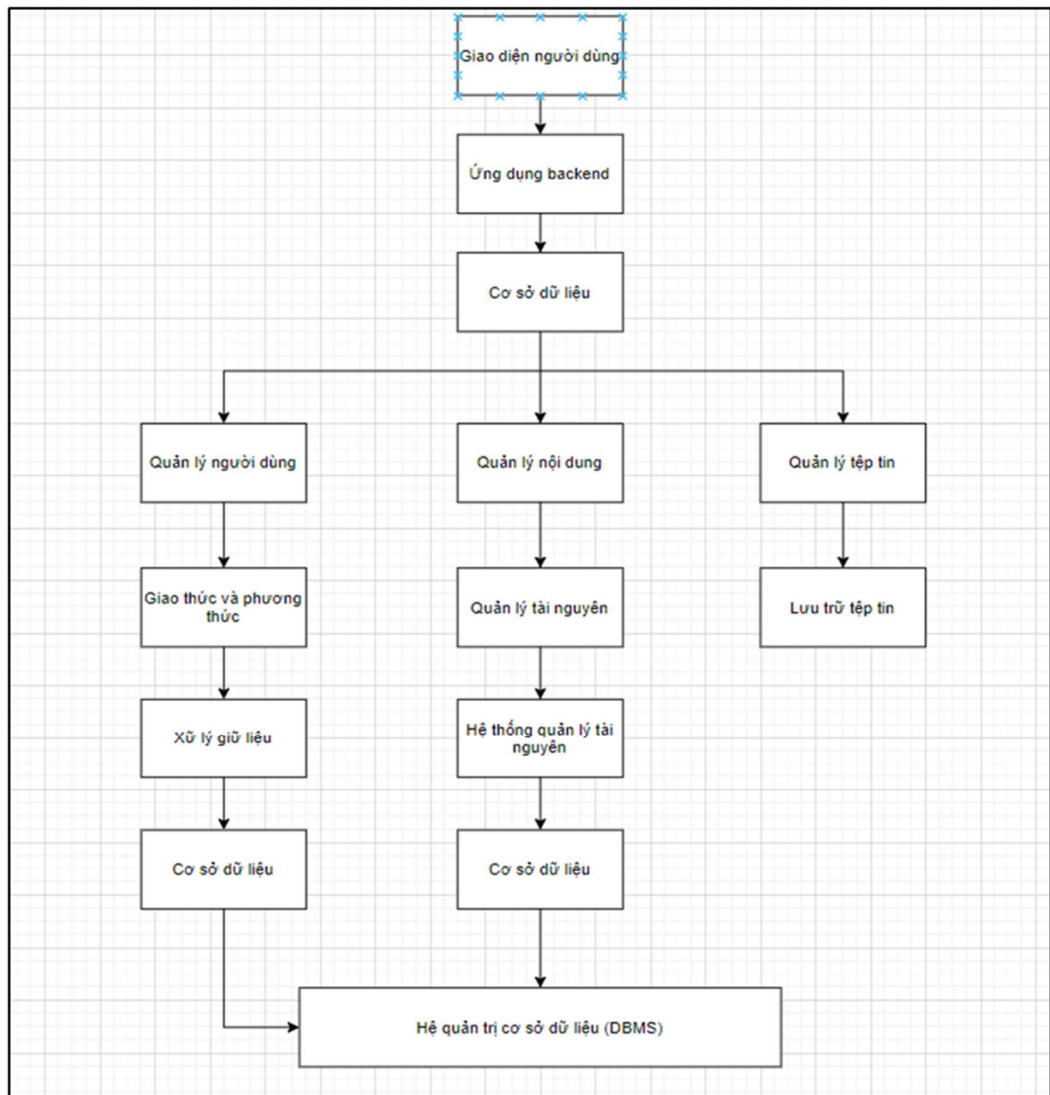
Quản lý bảo mật và quyền riêng tư: Đảm bảo rằng trang mạng xã hội có các biện pháp bảo mật và quyền riêng tư tốt để bảo vệ thông tin cá nhân và hoạt động của người dùng. Cho phép người dùng quản lý các cài đặt riêng tư và quyền riêng tư để kiểm soát ai có thể truy cập thông tin của họ.

Cung cấp tính năng và nội dung đa dạng: Cung cấp các tính năng và nội dung đa dạng để người dùng có thể khám phá, chia sẻ và tương tác. Cho phép người dùng tạo và chia sẻ nội dung như hình ảnh, video, bài viết và liên kết để tạo sự đa dạng và thú vị.

Chung quy về thiết kế giao diện và trải nghiệm người dùng hấp dẫn cho trang mạng xã hội đòi hỏi sự đơn giản, tính tương tác, cá nhân hóa, linh hoạt và bảo mật. Điều quan trọng là tạo ra một môi trường mà người dùng có thể tương tác, kết nối và chia sẻ thông tin một cách dễ dàng và thú vị.

3. Kiến trúc hệ thống và cấu trúc dữ liệu

3.1 Kiến trúc hệ thống

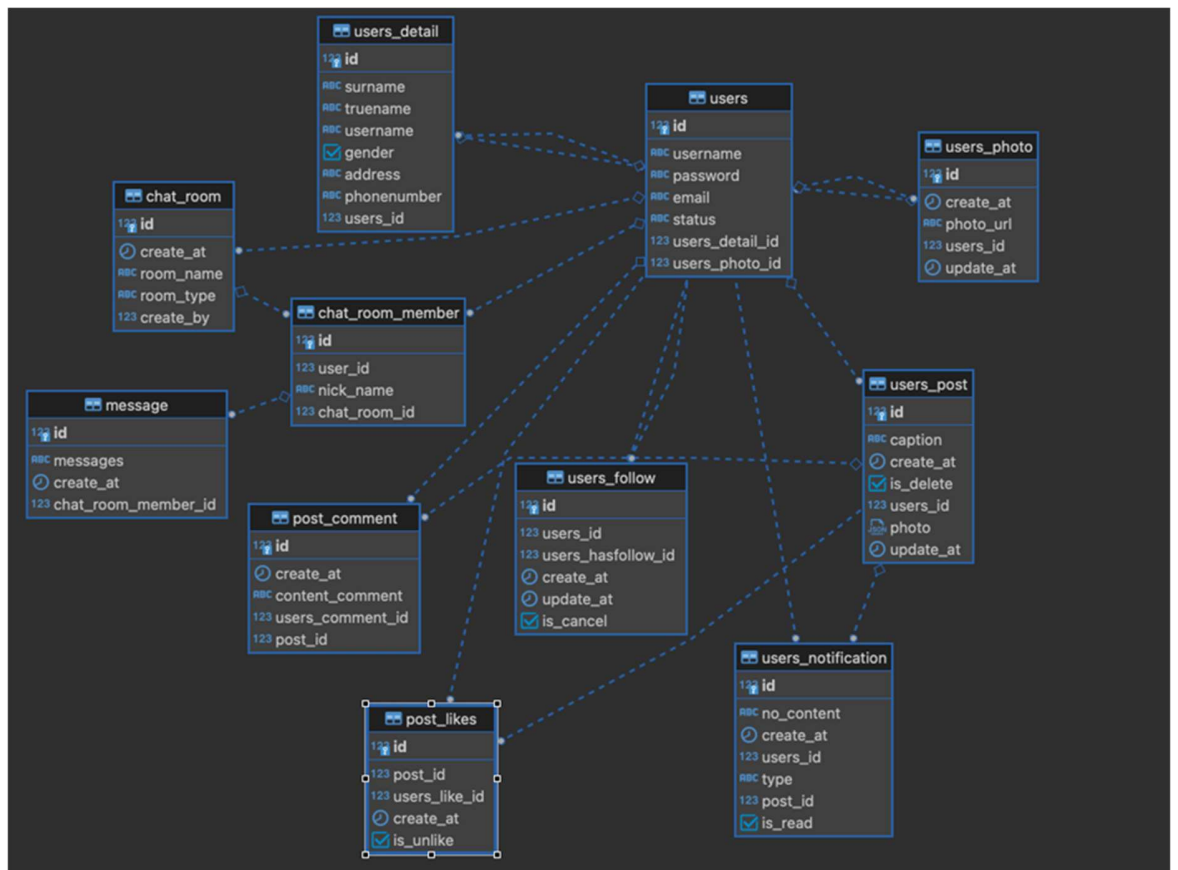


Hình 2.1: Sơ đồ mô tả kiến trúc hệ thống

- **Front-end (Giao diện người dùng):**
 - *Giao diện người dùng (UI):* Đây là phần mà người dùng tương tác trực tiếp với trang mạng xã hội. Nó bao gồm các trang, thành phần và giao diện người dùng để hiển thị nội dung, tương tác và trao đổi thông tin.
- **Back-end (Phía máy chủ):**
 - *Server:* Đây là phần xử lý logic và lưu trữ dữ liệu của trang mạng xã hội. Nó nhận và xử lý các yêu cầu từ giao diện người dùng và truy xuất, cập nhật và lưu trữ dữ liệu trong cơ sở dữ liệu.
 - *Cơ sở dữ liệu:* Đây là nơi lưu trữ dữ liệu của trang mạng xã hội. Cơ sở dữ liệu thường sử dụng hệ quản trị cơ sở dữ liệu

- (Database Management System - DBMS) như PostgreSQL, MySQL hoặc MongoDB để quản lý và tương tác với dữ liệu.
- *Ứng dụng Back-end*: Đây là phần xử lý logic và chức năng của trang mạng xã hội. Nó điều phối các yêu cầu, xử lý xác thực, quản lý trạng thái người dùng, và cung cấp các API (Application Programming Interface) cho giao diện người dùng tương tác với dữ liệu.
 - *Cơ chế tương tác và giao tiếp*:
 - *API (Application Programming Interface)*: Đây là giao diện mà các ứng dụng khác hoặc bên thứ ba có thể sử dụng để truy xuất và tương tác với dữ liệu và chức năng của trang mạng xã hội. API thường được xây dựng bằng các giao thức như RESTful hoặc GraphQL.
 - *Giao thức và phương thức*: Trang mạng xã hội sử dụng các giao thức và phương thức để truyền dữ liệu và tương tác giữa giao diện người dùng, ứng dụng back-end và cơ sở dữ liệu. Các giao thức phổ biến bao gồm HTTP, WebSocket, và các phương thức như GET, POST, PUT, DELETE.
 - *Hệ thống quản lý và xử lý*:
 - *Quản lý người dùng*: Bao gồm quản lý đăng nhập, xác thực, phân quyền và quản lý thông tin cá nhân của người dùng.
 - *Quản lý nội dung*: Điều phối và quản lý nội dung được chia sẻ trên trang mạng xã hội, bao gồm bài viết, hình ảnh, video, bình luận và các hoạt động khác.
 - *Xử lý dữ liệu*: Điều phối và xử lý dữ liệu truy xuất và cập nhật từ cơ sở dữ liệu, bao gồm lưu trữ, truy vấn và tìm kiếm dữ liệu.
 - *Hệ thống lưu trữ và quản lý tài nguyên*:
 - *Lưu trữ tệp và hình ảnh*: Quản lý và lưu trữ các tệp và hình ảnh được tải lên bởi người dùng.
 - *Quản lý tài nguyên*: Điều phối và quản lý các tài nguyên như bộ nhớ, băng thông và khả năng mở rộng của hệ thống.

3.2 Cấu trúc dữ liệu



Hình 2.2: Sơ đồ quan hệ cấu trúc dữ liệu


- **users** [**id** (int), username (varchar), password (varchar), email (varchar), status (enum), *users_detail_id* (int), *users_photo_id* (int)]. Table users em sử dụng để lưu giữ thông tin người dùng. Mỗi người dùng sẽ có duy nhất một thông tin cá nhân và một hình đại diện. **Id** là khoá chính của table, *users_detail_id* và *users_photo_id* là khoá ngoại của table.
- **users_photo** [**id** (int), create_at (timestamp), photo_url (varchar), *users_id* (int), update_at (timestamp)]. Table users_photo em sử dụng để lưu giữ thông tin về hình đại diện của người dùng. **Id** là khóa chính của table, *users_id* là khoá ngoại của table.
- **users_detail** [**id** (int), surname (varchar), truenname (varchar), username (varchar), gender (boolean), address (varchar), phonenumber (varchar), *users_id* (int)]. Table users_detail là nơi mà em lưu trữ thông tin chi tiết của người dùng. **Id** là khoá chính, *users_id* là khoá ngoại.
- **users_post** [**id** (int), caption (varchar), create_at (timestamp), is_delete (boolean), *users_id* (int), photo (string array), update_at (timestamp)]. Table users_post là nơi mà em lưu trữ nội dung các bài đăng của người dùng. **Id** là

khoá chính, *users_id* là khoá ngoại.

- **post_likes** [**id** (int), *post_id* (int), *users_like_id* (int), *create_at* (timestamp), *is_unlike* (boolean)]. Table *post_likes* là nơi em lưu trữ lịch sử yêu thích bài post. Từ đó em có thể biết được là những ai đã và đang yêu thích một bài post nào đó. **Id** là khoá chính, *post_id* và *users_like_id* là những khoá ngoại.
- **post_comment** [**id** (int), *create_at* (timestamp), *content_comment* (varchar), *users_comment_id* (int), *post_id* (int)]. Table *post_comment* là nơi em lưu trữ lịch sử bình luận của một bài post nào đó. **Id** là khoá chính, *post_id* và *users_comment_id* là những khoá ngoại.
- **users_follow** [**id** (int), *users_id* (int), *users_hasfollow_id* (int), *create_at* (timestamp), *update_stamp* (timestamp), *is_cancel* (boolean)]. Table *users_follow* là nơi em lưu giữ dữ liệu về lịch sử kết bạn của người dùng. **Id** là khoá chính, *users_id* và *users_hasfollow_id* là những khoá ngoại.
- **chat_room** [**id** (int), *create_at* (timestamp), *room_name* (varchar), *room_type* (enum), *create_by* (int)]. Table *chat_room* là nơi mà em lưu lại thông tin các phòng chat. **Id** là khóa chính, *create_by* là khóa ngoại.
- **chat_room_member** [**id** (int), *user_id* (int), *nick_name* (varchar), *chat_room_id* (int)]. Table *chat_room_member* em sử dụng để lưu thông tin của người dùng tham gia các đoạn hội thoại. **Id** là khóa chính, *chat_room_id* và *user_id* là khóa ngoại.
- **message** [**id** (int), *messages* (varchar), *create_at* (timestamp), *chat_room_member_id* (int)]. Table *message* em sử dụng để lưu lại các đoạn hội thoại của người dùng. **Id** là khóa chính, *chat_room_member_id* là khóa ngoại.

4. Tính năng và chức năng chính

4.1 Đăng ký và đăng nhập.



TẠO TÀI KHOẢN

Tên đăng nhập

Email

Mật khẩu


Nhập lại mật khẩu

Submit

BẠN ĐÃ CÓ TÀI KHOẢN? [ĐĂNG NHẬP.](#)

Hình 2.3: Màn hình đăng ký tài khoản

Trường hợp các trường dữ liệu không hợp lệ một icon dấu “X” sẽ được hiển thị bên cạnh để thông báo cho người dùng.



TẠO TÀI KHOẢN

Tên đăng nhập X

Email X

Mật khẩu X

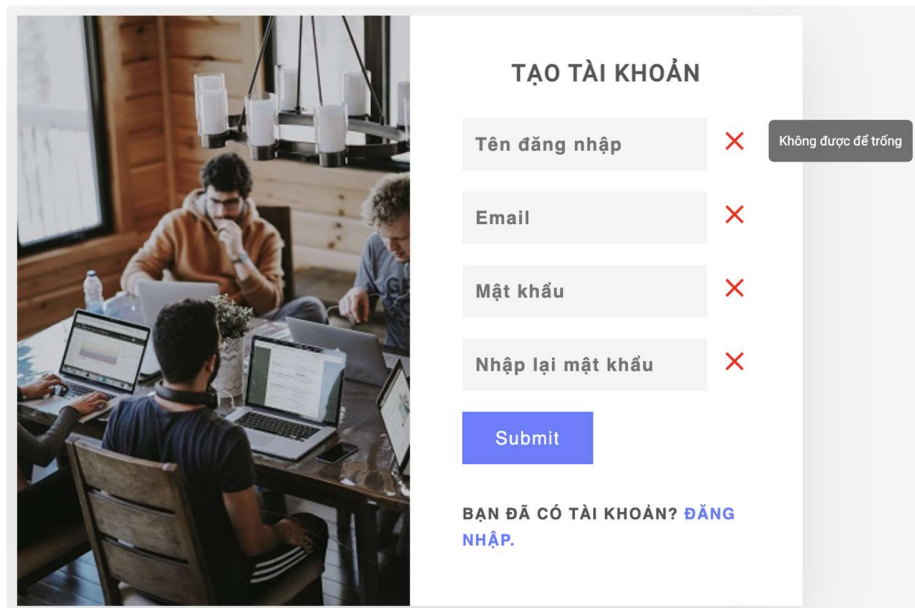
Nhập lại mật khẩu X

Submit

BẠN ĐÃ CÓ TÀI KHOẢN? [ĐĂNG NHẬP.](#)

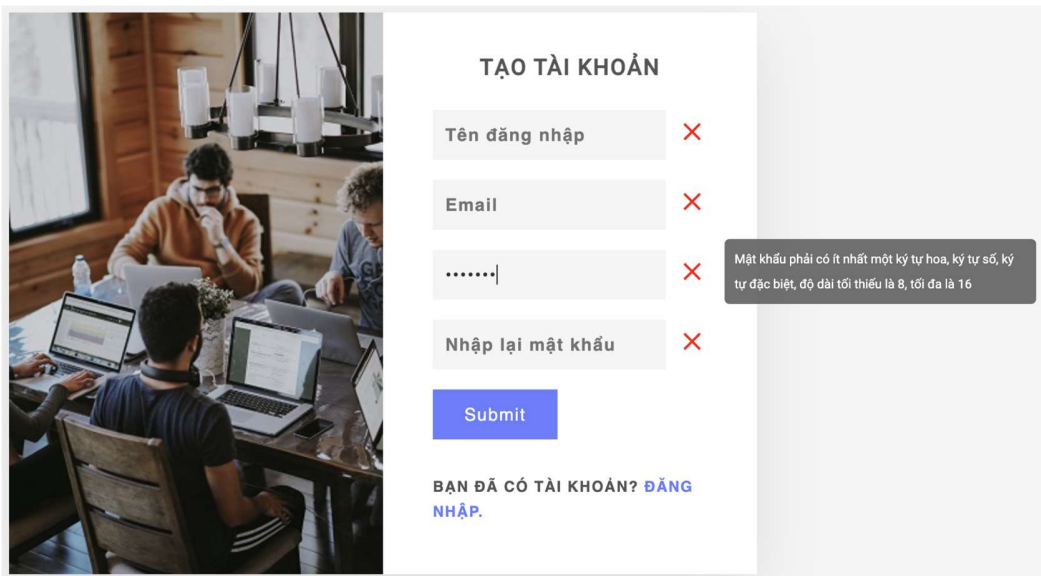
Hình 2.4: Trường hợp các trường dữ liệu không hợp lệ

Để biết được lý do các trường dữ liệu không hợp lệ, người dùng có thể rê chuột vào các dấu “X” để xem nguyên nhân. Điều này sẽ giúp giao diện tránh phải hiển thị quá nhiều thông tin một lần, gây rối mắt người dùng.




Hình 2.5: rê chuột vào dấu “X” để xem nguyên nhân

Đối với trường dữ liệu, Ngoài yêu cầu bắt buộc, mật khẩu phải thoả mãn có ít nhất một ký tự in hoa, ký tự số, ký tự đặc biệt và số lượng ký tự trong khoảng từ 8 đến 16.



Hình 2.6: Mật khẩu phải có độ phức tạp nhất định nhằm đảm bảo tính bảo mật cao

Tất cả sẽ chuyển về dấu tích xanh nếu tất cả các trường dữ liệu hợp lệ. Lúc này chúng ta chỉ cần nhận vào button submit. Nếu tạo thành công biểu mẫu đăng nhập sẽ được hiển thị.



TẠO TÀI KHOẢN

nguyenvana ✓

nguyenvana@gmail.cc ✓

..... ✓

.....| ✓

Submit

BẠN ĐÃ CÓ TÀI KHOẢN? [ĐĂNG NHẬP.](#)

Hình 2.7: Trường hợp tất cả các trường dữ liệu đã hợp lệ.

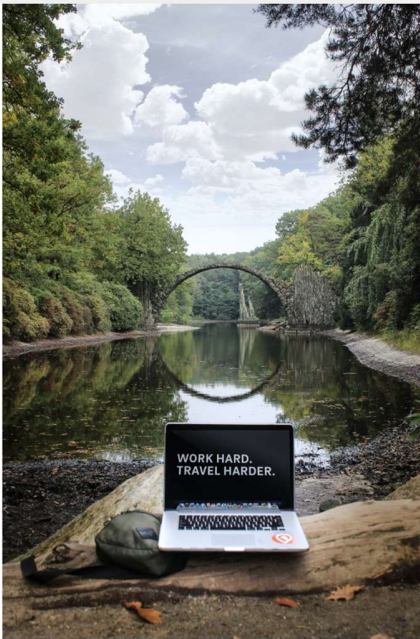
ĐĂNG NHẬP

Tên đăng nhập

Mật khẩu

Submit

BẠN CHƯA CÓ TÀI KHOẢN? [ĐĂNG KÝ.](#)



Hình 2.8: Màn hình đăng nhập

Tương tự với màn hình đăng ký, màn hình đăng nhập cũng được kiểm tra các trường dữ liệu.

ĐĂNG NHẬP

Tên đăng nhập

×

Mật khẩu

×

Submit

BẠN CHƯA CÓ TÀI KHOẢN?
[ĐĂNG KÝ.](#)


Hình 2.9: Trường hợp các trường dữ liệu không hợp lệ

ĐĂNG NHẬP

Tên đăng nhập

Mật khẩu

☐ I'm not a robot


reCAPTCHA
[Privacy](#) - [Terms](#)

Submit

BẠN CHƯA CÓ TÀI KHOẢN? [ĐĂNG KÝ.](#)

Hình 2.10: Màn hình đăng nhập có recaptcha

4.2 Tạo và quản lý hồ sơ cá nhân.

Đối với tài khoản vừa tạo. Trang quản lý hồ sơ cá nhân sẽ được để trống. Từ đây người dùng hoàn toàn có thể cập nhật lại thông tin các nhân của mình.

Chỉnh sửa trang cá nhân
Đổi mật khẩu
Đăng xuất

Họ tên

Họ và tên đệm

Tên

Tên người dùng

Tên người dùng

Giới tính

☒ Nam
☐ Nữ

Địa chỉ

Địa chỉ

Số điện thoại

Số điện thoại

Gửi

Hình 2.11: Màn hình quản lý hồ sơ cá nhân.

Chỉnh sửa trang cá nhân
Đổi mật khẩu
Đăng xuất

huudung

Họ tên

Lê Đoàn Hữu

Dũng

Tên người dùng

huudung

Giới tính

☒ Nam
☐ Nữ

Địa chỉ

Thành phố Thủ Đức

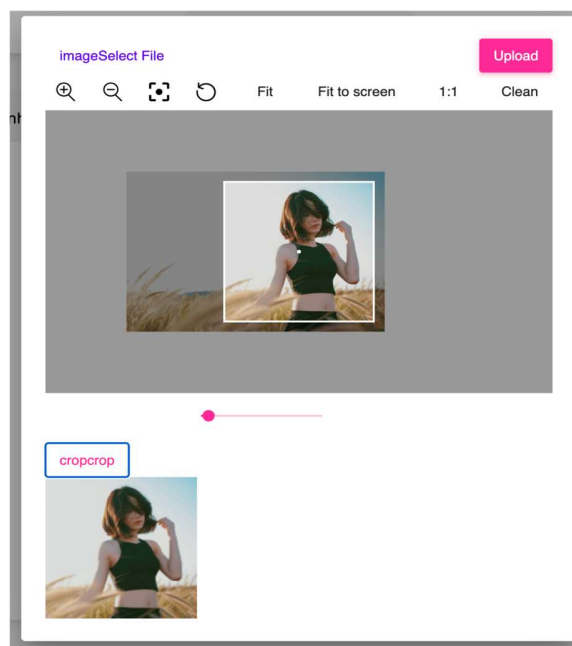
Số điện thoại

0123456789

Gửi

Hình 2.12: Màn hình quản lý hồ sơ cá nhân sau khi đã cập nhật

Đối với hình ảnh đại diện, người dùng có thể click vào hình tròn màu cam để mở hộp thoại cập nhật hình đại diện. Trong hộp thoại này sẽ có một số nút bấm như phóng to, thu nhỏ, xoay hình, tự điều chỉnh phù hợp khung... Từ đó người dùng có thể tự ý điều chỉnh sao cho phù hợp với sở thích bản thân. Sau khi đã chỉnh sửa xong, lúc này người dùng chỉ cần chọn nút upload để thay đổi hình đại diện.




Hình 2.13: Hộp thoại cập nhật hình đại diện

Chỉnh sửa trang cá nhân

Đổi mật khẩu

Đăng xuất



huudung

Họ tên

Lê Đoàn Hữu

Dũng

Tên người dùng

huudung

Giới tính

☒ Nam ☐ Nữ

Địa chỉ

Thành phố Thủ Đức

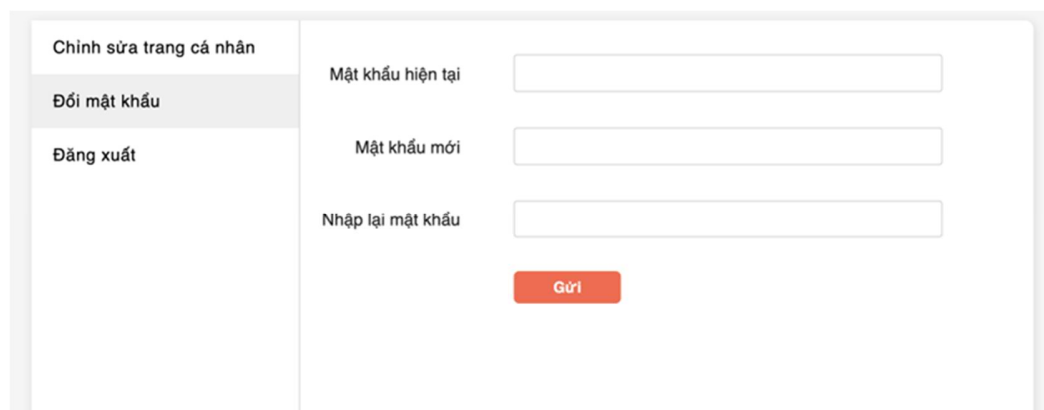
Số điện thoại

0123456789

Gửi

Hình 2.14: Hình ảnh đại diện sau khi thực hiện cập nhật

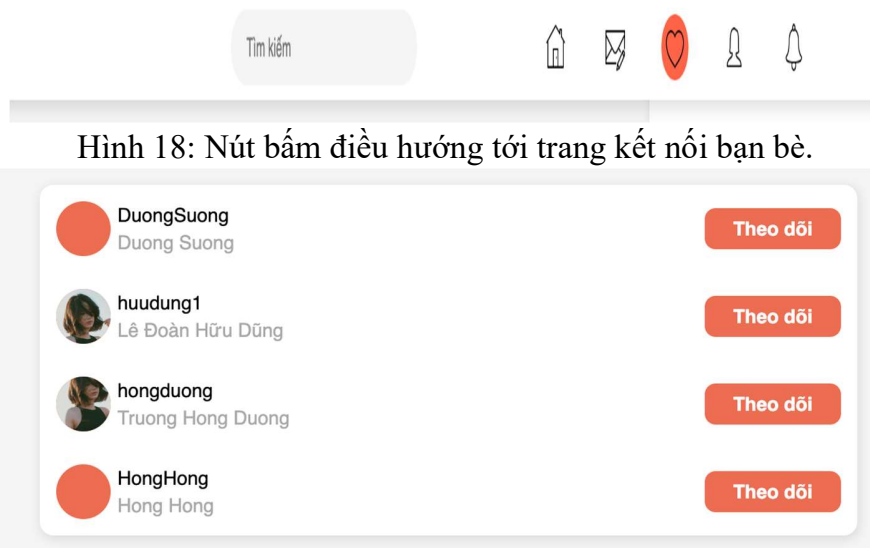
Cũng tại trang quản lý hồ sơ cá nhân, người dùng có thể thực hiện đổi mật khẩu hiện tại hoặc là đăng xuất khỏi tài khoản hiện tại.



Hình 2.15: Màn hình thay đổi mật khẩu hiện tại

4.3 Kết nối và tương tác với bạn bè

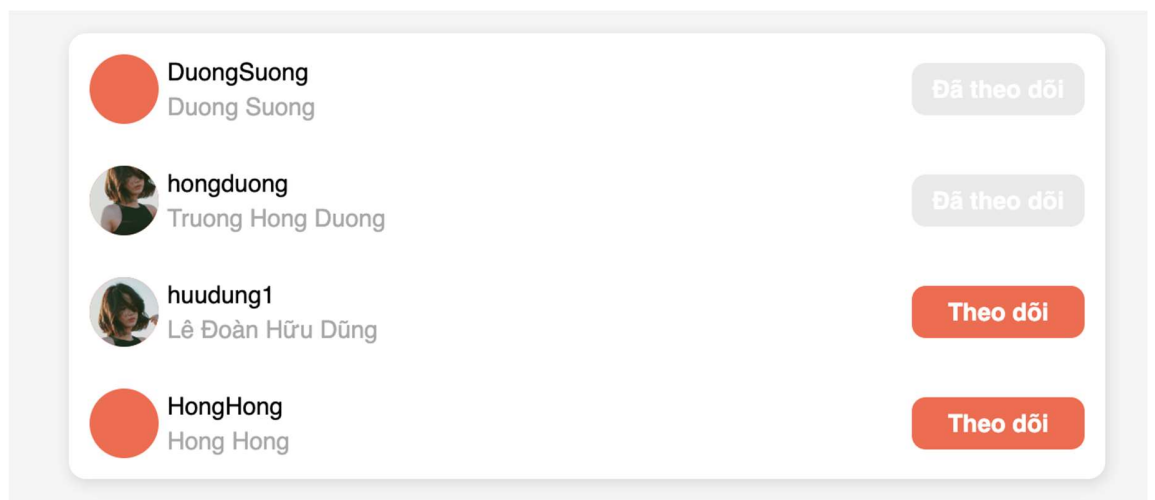
Tại góc phải màn hình. Nhấn chọn vào nút có hình trái tim. Người dùng sẽ được điều hướng tới trang kết nối bạn bè.



Hình 18: Nút bấm điều hướng tới trang kết nối bạn bè.

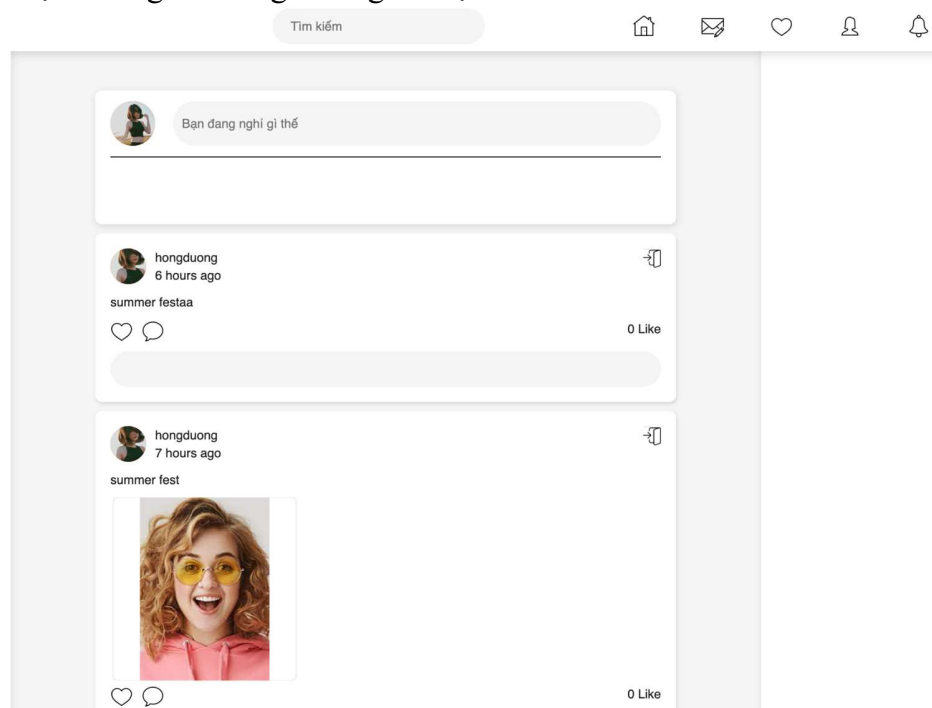
Hình 2.16: Trang kết nối bạn bè.

Nhấn chọn vào nút theo dõi để theo dõi người bạn đó. Từ đó bạn có thể tương tác cũng như xem các bài đăng cũng như chia sẻ của người dùng đó. Hoặc là nhấn vào nút đã theo dõi để hủy theo dõi.



Hình 2.17: Chọn theo dõi để theo dõi người bạn đó hoặc tái nhấn để huỷ theo dõi.

Sau khi đã lựa chọn theo dõi. Lúc này sau khi quay trở lại trang chính bạn sẽ thấy được những bài đăng của người bạn đó.



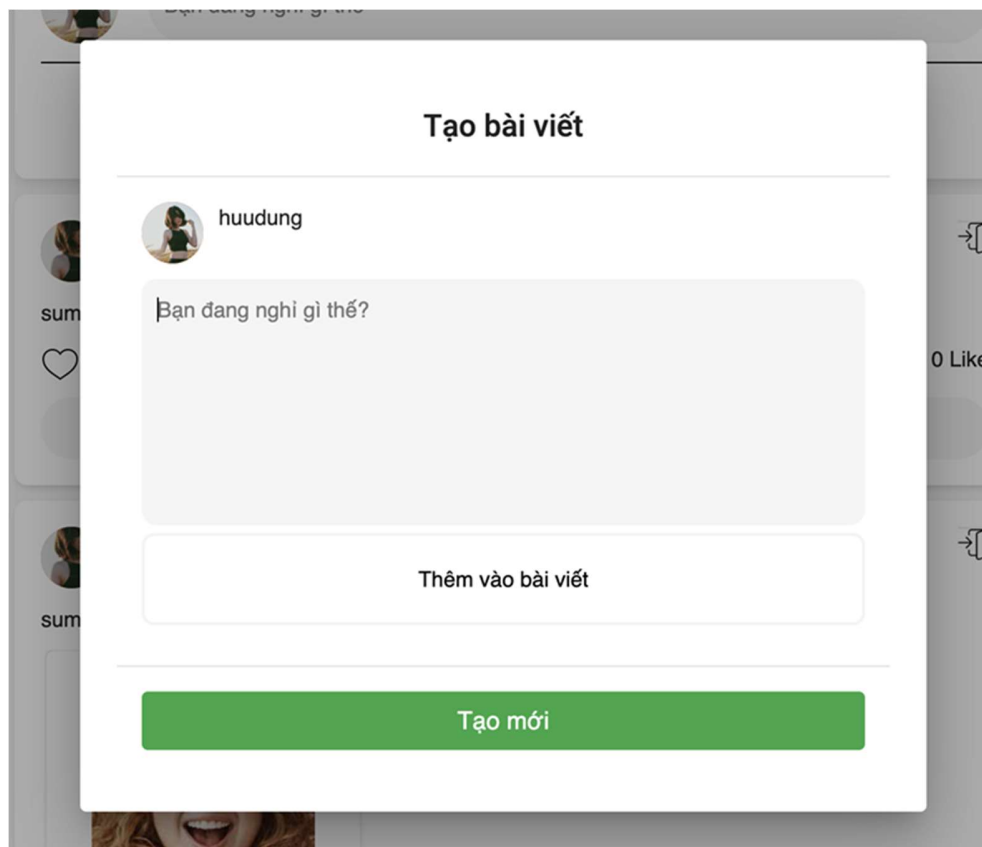
Hình 2.18: Danh sách bài đăng của những người bạn đã theo dõi.

4.4 Chia sẻ nội dung và bài đăng

Tại màn hình chính, người dùng có thể nhấn vào trường “Bạn đang nghĩ gì thế” để mở hộp thoại đăng bài.

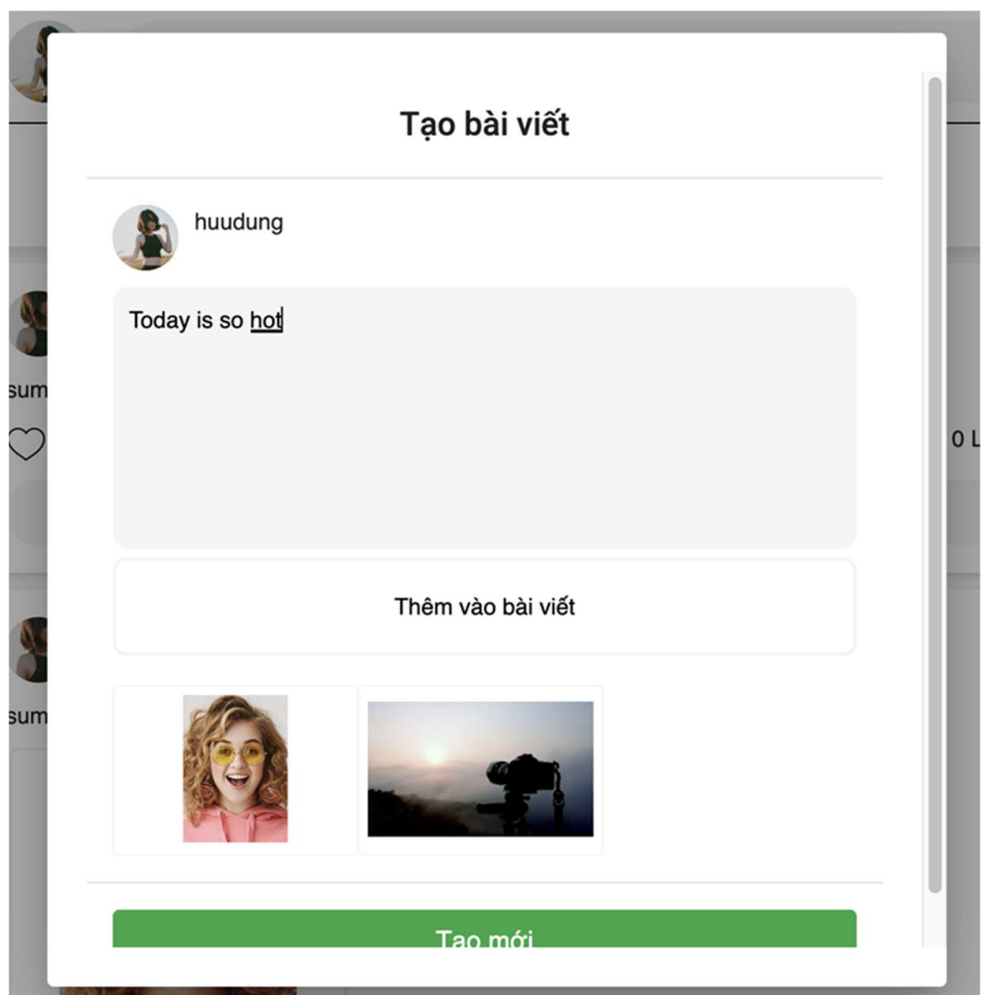


Hình 2.19: Nhấn vào trường “Bạn đang nghĩ gì thế” để mở hộp thoại đăng bài

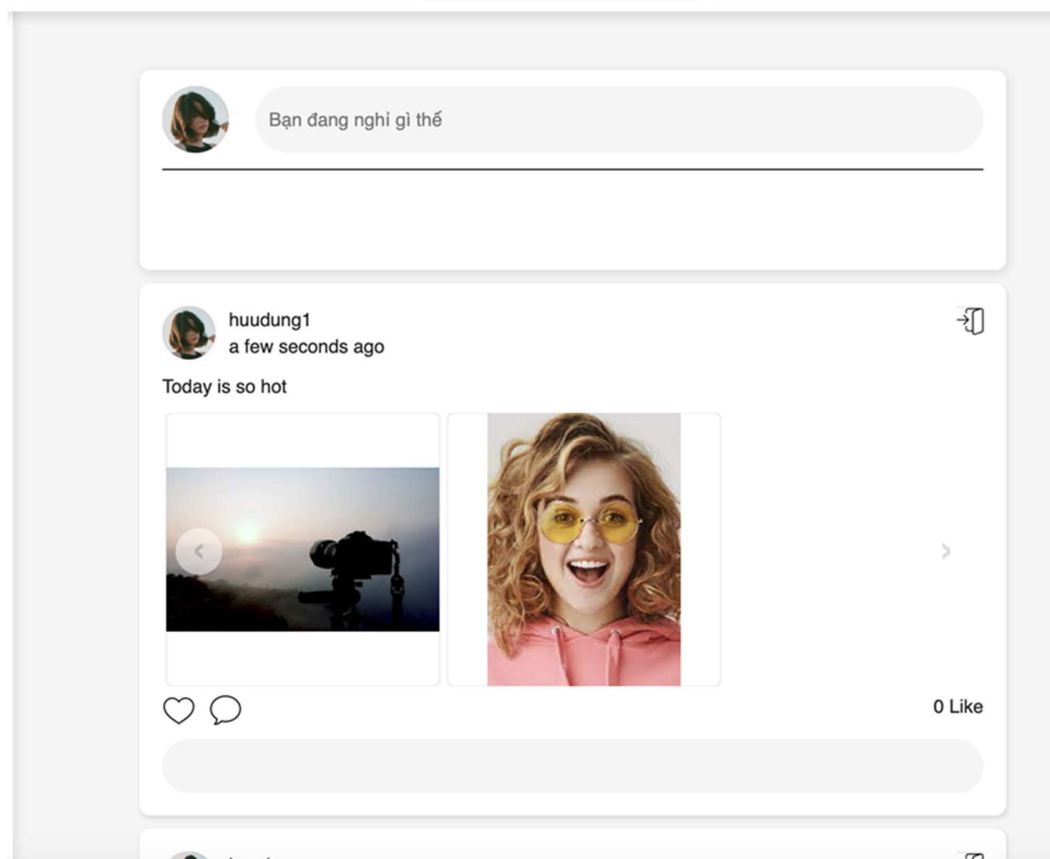


Hình 2.20: Hộp thoại tạo bài viết mới.

Người dùng có thể nhập những gì mà họ muốn vào trường nhập dữ liệu đồng thời nhấn chọn vào phần thêm vào bài viết để chèn hình ảnh. Sau khi hoàn tất người dùng chỉ cần nhấn chọn vào nút tạo mới bài viết. Bài viết tạo thành công sẽ ngay lập tức xuất hiện ở trang chính.

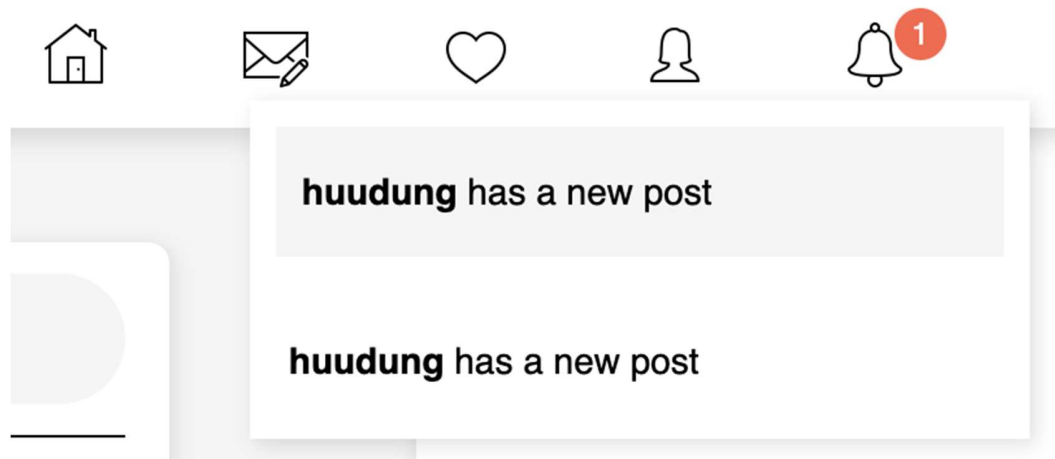


Hình 2.21: Người dùng soạn thảo bài viết và chèn hình ảnh.



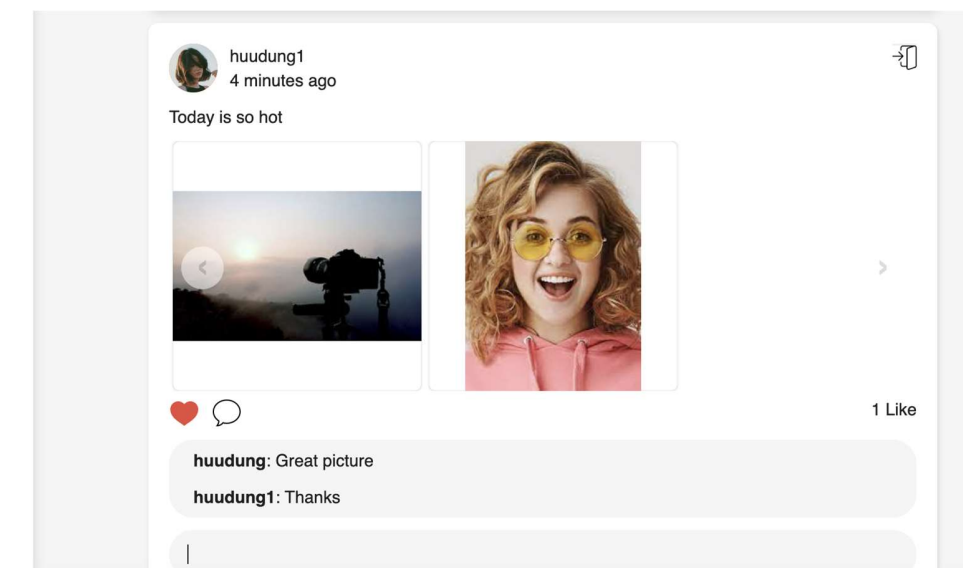
Hình 2.22: Danh sách các bài đăng.

Đồng thời thông báo về việc bạn vừa mới đăng bài cũng sẽ được gửi tới ngay lập tức cho những người dùng là bạn của bạn.



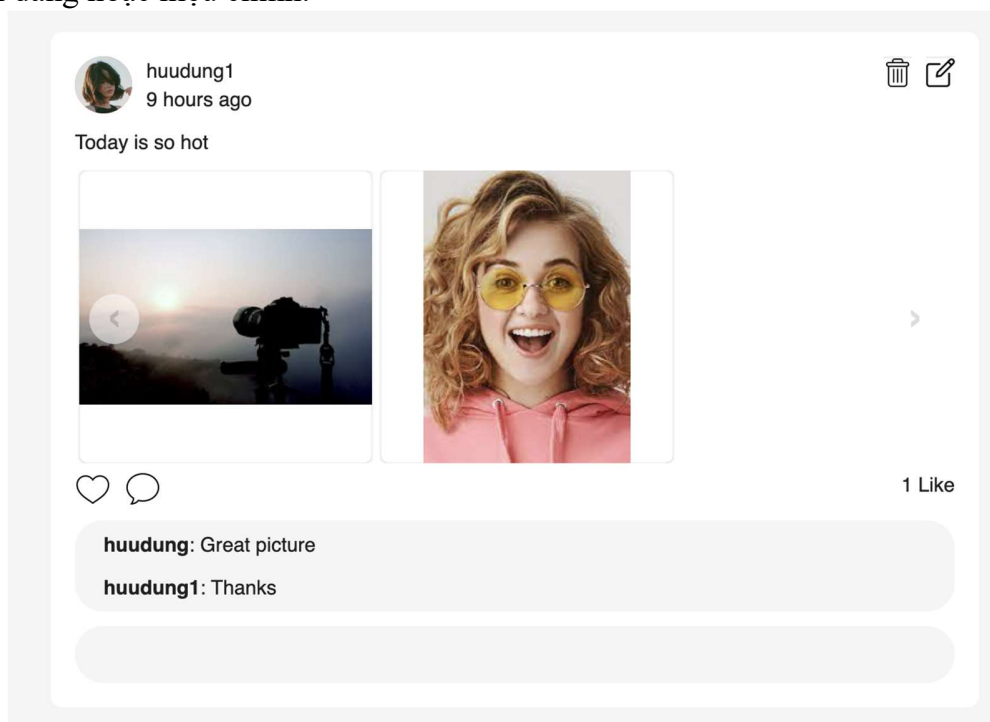
Hình 2.23: Thông báo về bài viết mới từ những người bạn

Người dùng còn có thể thả tim cho bài viết để thể hiện cảm xúc và thực hiện comment vào bài viết.

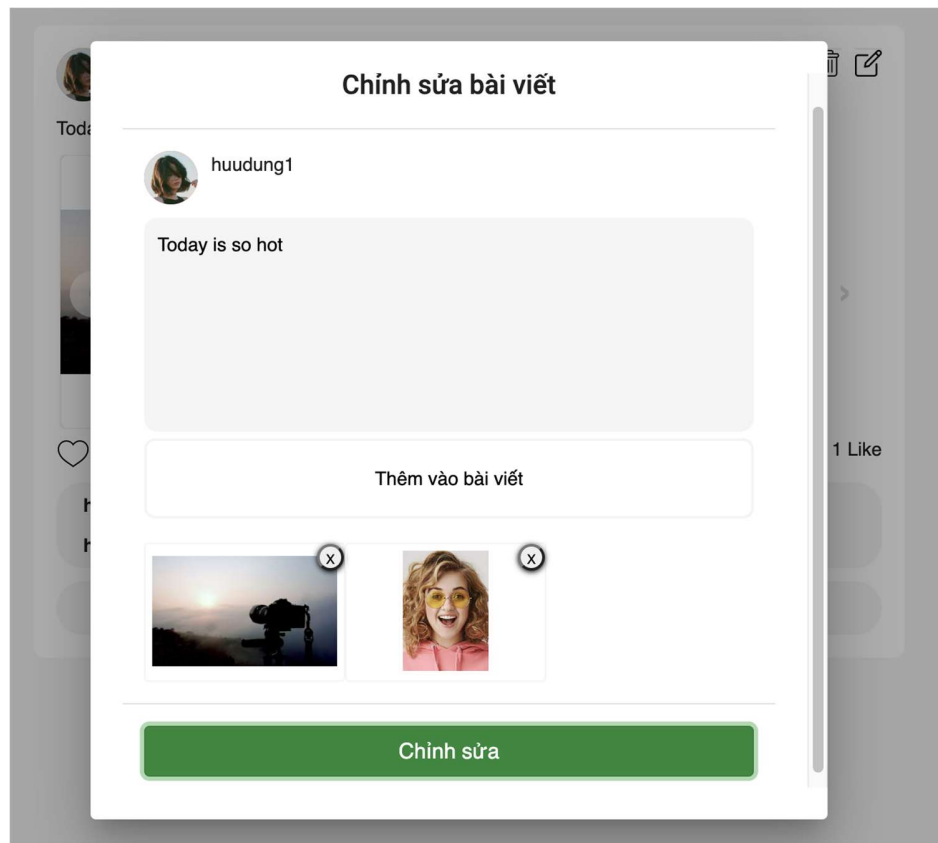


Hình 2.24: Người dùng tương tác trên bài viết

Ngoài ra người dùng có thể hiệu chỉnh bài viết bằng cách nhấn chuột vào icon nằm ở góc bên phải của bài đăng. Người dùng sau đó sẽ được chuyển đến một trang mới chỉ có duy nhất bài viết đó. Lúc này người dùng có thể lựa chọn xóa bài đăng hoặc hiệu chỉnh.



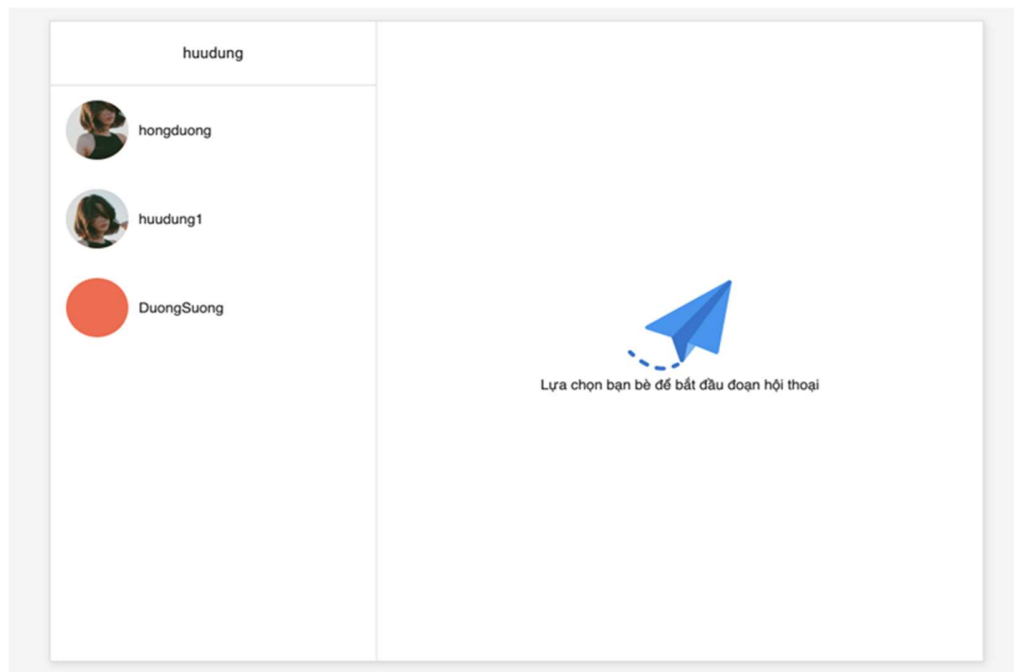
Hình 2.25: Chi tiết bài đăng, xóa hay hiệu chỉnh bài đăng



Hình 2.26: Hộp thoại hiệu chỉnh bài viết.

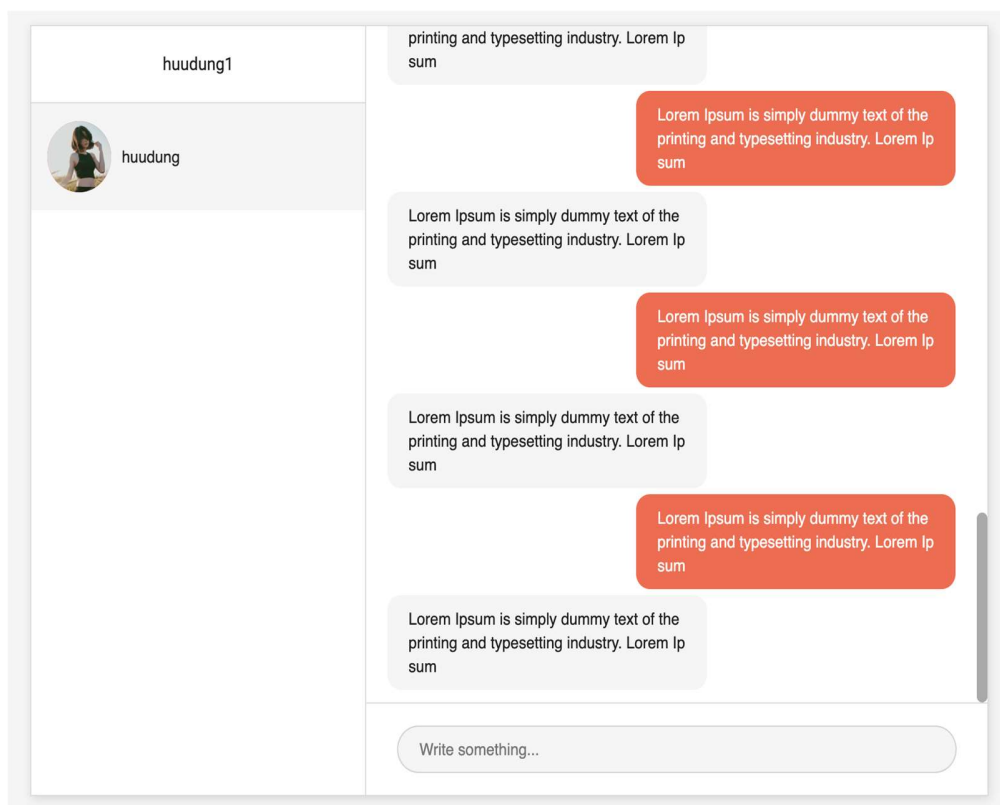
Với hộp thoại này người dùng có thể thay đổi nội dung bài viết hoặc loại bỏ những tấm hình đính kèm mà mình không mong muốn.

4.5 Nhắn tin và giao tiếp



Hình 2.27: Giao diện trang nhắn tin

Đối với phần nhắn tin và giao tiếp, trong phạm vi đề tài hiện tại em chỉ mới hỗ trợ việc nhắn tin giữa hai người. Trong tương lai tính năng này hoàn toàn có thể phát triển để có thể nhắn tin giữa nhiều người với nhau. Thách thức lớn nhất của tính năng nhắn tin là khả năng realtime. Em đã lựa chọn cách thức là sử dụng websocket. Mỗi lần người dùng nhắn tin, một socket event sẽ được bắn tới cho người dùng đích. Lúc này ứng dụng sẽ tự động bắt event đó đồng thời từ những đặc tả của đoạn tin nhắn để hiển thị phần tin nhắn mới ở phía người dùng đích.



Hình 2.28: Ví dụ về đoạn hội thoại giữa hai người dùng.

4.6 Kỹ thuật áp dụng

4.6.1 Mã hóa mật khẩu người dùng bằng thuật toán Bcrypt

Chỉ mới cách đây hai năm, cộng đồng lập trình đã từng xôn xao về việc một công ty công nghệ lớn của Việt Nam đang bị hacker xâm nhập chỉ với một phương pháp tấn công đơn giản là SQL Injection. Cụ thể khi người dùng nhập tên đăng nhập và mật khẩu. Ở dưới server sẽ thực hiện bước truy vấn cơ sở dữ liệu bằng một câu lệnh SQL như hình minh họa ở dưới.

```
SELECT count(*)
FROM user
WHERE username = 'username' AND userpassword = 'password'
```

Mọi chuyện vẫn sẽ diễn ra bình thường nếu như mật khẩu người dùng là một chuỗi ký tự chữ bình thường. Tuy nhiên với câu lệnh SQL như trên kẻ xấu hoàn toàn có thể dùng thủ thuật để kết quả SQL luôn trả về giá trị bằng cách thêm dấu nháy phẩy sửa đổi câu lệnh SQL.

```
john' or 1=1 ; --
```

Lúc này câu lệnh SQL sẽ bị thay đổi thành

```
SELECT count(*)  
FROM user  
WHERE username = 'username' AND userpassword = 'john' or 1 = 1; --'
```

Với câu lệnh SQL mới này thì kết quả câu lệnh sẽ luôn có giá trị trả về từ đó có thể dễ dàng đăng nhập vào hệ thống

- Giải pháp.

Vậy để xử lý trường hợp trên em sẽ thực hiện việc mã hóa mật khẩu người dùng sau đó sẽ lưu vào cơ sở dữ liệu thay vì lưu trực tiếp giá trị người dùng nhập vào từ bàn phím và lưu thẳng vào db. Với cách này nếu người dùng nhập dấu phẩy ở trường nhập liệu thì khi chạy vào câu lệnh SQL nó đã được biến đổi để loại bỏ dấu phẩy nháy từ đó có thể tránh bị tấn công hệ thống. Ngoài ra với việc mã hóa mật khẩu sẽ khiến cho những hacker khó thể lấy được mật khẩu của người dùng hay kể cả với những người làm việc trong dự án có quyền truy cập vào cơ sở dữ liệu cũng khó có thể biết được mật khẩu người dùng.

Thuật toán mã hóa mà em sử dụng là thuật toán Bcrypt.

bcrypt là một thuật toán hash mật khẩu được thiết kế để làm việc chậm hơn và tăng khả năng chống lại các cuộc tấn công brute-force. Nó được sử dụng phổ biến trong việc lưu trữ mật khẩu trong các hệ thống bảo mật. Cơ chế của thuật toán bcrypt bao gồm các bước sau:

Sử dụng một hằng số được gọi là "salt" (muối) để tạo một chuỗi ngẫu nhiên có độ dài cố định. Salt là một giá trị ngẫu nhiên, duy nhất được tạo ra cho mỗi mật khẩu. Salt này được lưu trữ cùng với mật khẩu đã mã hóa. Mật khẩu gốc cùng với salt được kết hợp và sau đó được chuyển đổi thành một chuỗi ký tự gốc (plaintext) có độ dài cố định.

Bcrypt sử dụng một hàm hash mạnh như Blowfish để thực hiện phép biến đổi trên chuỗi ký tự gốc. Thực hiện nhiều vòng biến đổi, mỗi vòng sẽ trộn kết quả với một phần của salt và kết quả của vòng trước đó.

Kết quả cuối cùng sau các vòng biến đổi được gọi là "bcrypt hash". Hash này là một chuỗi ký tự có độ dài cố định và không thể được chuyển ngược thành mật khẩu gốc ban đầu.

Bcrypt hash kết quả được lưu trữ trong cơ sở dữ liệu, cùng với salt được sử dụng cho mật khẩu đó.

Khi một người dùng cần xác minh mật khẩu, quá trình bcrypt sẽ được áp dụng lại. Mật khẩu nhập vào sẽ được kết hợp với salt đã lưu trữ, và sau đó được thực hiện các vòng biến đổi tương tự. Nếu kết quả hash của mật khẩu nhập vào khớp với bcrypt hash đã lưu trữ, thì mật khẩu được xác minh là chính xác.

4.6.2 Sử dụng JSON Web Token để giao tiếp an toàn giữa trang web và hệ thống backend

JSON Web Token (JWT) là 1 tiêu chuẩn mở (RFC 7519) định nghĩa cách thức truyền tin an toàn giữa bên bằng 1 đối tượng JSON. Thông tin này có thể được xác minh và đáng tin cậy vì nó được ký điện tử. JWT có thể được ký bằng cách dùng một secret (sử dụng thuật toán HMAC) hoặc cặp khóa public/private bằng thuật toán RSA hoặc ECDSA.

JSON Web Tokens bao gồm 3 phần được phân tách bằng dấu chấm (.) đó là:

- Header
- Payload
- Signature

JWT có cấu trúc như sau: xxxxx.yyyyyy.zzzzzz

Header: thường bao gồm hai phần: loại token, là JWT và thuật toán ký đang được sử dụng, chẳng hạn như HMAC SHA256 hoặc RSA. Ví dụ:

```
{
  "alg": "HS256",
  "typ": "JWT"
}
```

Payload: nó chứa các the claims. Claims thường chứa các thông tin entity (thông tin user) và các dữ liệu bổ sung. Có 3 loại claims: registered, public, và private claims. Ví dụ:

```
{
  "email": "user@gamil.com",
  "roleId": "12",
  "userId": "1"
}
```

Signature: Để tạo phần chữ ký, bạn phải lấy header được mã hóa, payload được mã hóa, một secret, thuật toán được chỉ định trong header và sign. Secret là một chuỗi ký tự, secret có thể có hoặc không. Ví dụ nếu sử dụng thuật toán HMAC SHA256 signature được tạo bằng cách:

```
HMACSHA256(
  base64UrlEncode(header) + "." +
  base64UrlEncode(payload),
  secret)
```

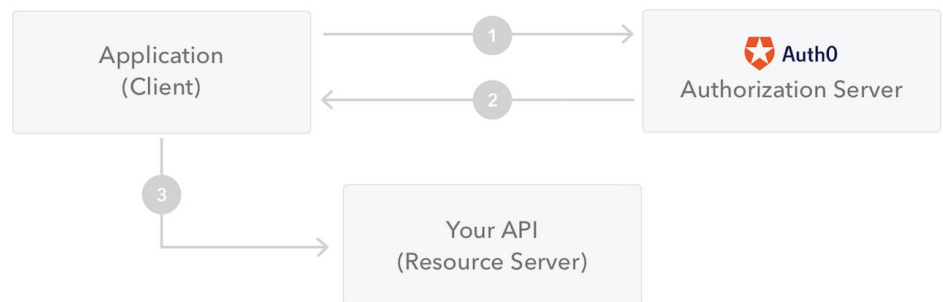
Lưu trữ JWT an toàn:

- Như đã giới thiệu ở trên một ứng dụng phổ biến nhất của JWT là xác thực API và để có thể làm được việc này cần lưu lại jwt. Vậy jwt cần lưu ở đâu cho an toàn. Nếu lưu trữ nó bên trong localStorage, nó có thể truy cập bằng bất kỳ script nào trong trang của bạn (một cuộc tấn công XSS có thể cho phép kẻ tấn công bên ngoài lấy được token).
- JWT cần được lưu trữ bên trong cookie httpOnly, một loại cookie đặc biệt chỉ được gửi trong các yêu cầu HTTP đến máy chủ và không bao giờ

có thể truy cập được (cả để đọc hoặc ghi) từ JavaScript đang chạy trong trình duyệt.

Cách JWT hoạt động:

- Trong xác thực API khi user login server sẽ trả về cho client một JWT. Ở các request sau để có thể truy cập vào tài nguyên người dùng đó thì cần gửi token này lên. Token này sẽ được gửi trong header với key là Authorization và value có dạng Bearer <token> token ở đây là JWT.
- Sơ đồ dưới đây cho thấy hoạt động xác thực API bằng JWT.



1: Ứng dụng hoặc máy client yêu cầu ủy quyền đến authorization server

2: Khi yêu cầu ủy quyền được xác nhận. authorization server sẽ trả về một access token cho ứng dụng (client)

3: ứng dụng (client) sử dụng access token để truy cập vào resource được bảo vệ

- Lưu ý rằng với các mã thông báo đã ký, tất cả thông tin có trong mã thông báo sẽ được hiển thị cho người dùng hoặc các bên khác, mặc dù họ không thể thay đổi nó. Điều này có nghĩa là bạn không nên đặt thông tin bí mật trong mã thông báo.

4.6.3 Google captcha và ứng dụng trong việc ngăn chặn tấn công brute force

Google reCAPTCHA là một dịch vụ cung cấp bởi Google để giúp xác định xem người dùng trên web là con người hay bot (chương trình máy tính tự động). reCAPTCHA thường được sử dụng để bảo vệ các trang web khỏi spam, tấn công tài khoản, và các hoạt động gian lận trực tuyến khác. reCAPTCHA yêu cầu người dùng thực hiện một số thao tác như điền vào một ô văn bản, chọn các hình ảnh tương tự nhau, hoặc thực hiện các thao tác kéo và thả để xác nhận rằng họ không phải là bot. Thông qua việc sử dụng các thuật toán tiên tiến, reCAPTCHA có thể xác định xem người dùng đang tương tác với trang web có phải là một con người thực sự hay không.

reCAPTCHA được sử dụng rộng rãi trên các trang web, đặc biệt là trong quá trình đăng ký, đăng nhập, hoặc khi thực hiện các hoạt động nhạy cảm


trên mạng. Nó giúp cải thiện bảo mật và tránh những hoạt động độc hại từ phía bot.

Cho đến thời điểm hiện tại google reCAPTCHA đã có 3 phiên bản.

- Cài đặt google ReCAPTCHA:

Truy cập vào trang web <https://www.google.com/recaptcha/admin/create>
Một trang web thiết lập sẽ xuất hiện. Tại đây em sẽ thiết lập một số thông tin bao gồm label, domain, và version google ReCAPTCHA mà em mong muốn.

Google reCAPTCHA



Now enterprise ready!

Enterprise adds advanced features like *MEAs*, spam/fraud protection & Google Cloud integration.

- ✓ Up to 1,000,000 assessments/month at no cost
- ✓ No Credit Card required

[Switch to create a classic key](#)

Label ⓘ

e.g. example.com

0 / 50

reCAPTCHA type ⓘ

☒ Score based (v3) Verify requests with a score

☐ Challenge (v2) Verify requests with a challenge

Domains ⓘ

+ Add a domain, e.g. example.com

✓ GOOGLE CLOUD PLATFORM

Sau khi đã điền đầy đủ thông tin.

Google reCAPTCHA

Adding reCAPTCHA to your site



We're still setting up reCAPTCHA Enterprise settings in Google Cloud, but you can get started using the key details below.

It should take around 1 minute to completely setup. Once completed, you'll have unlimited assessments and the ability to use advanced features like MFA and Account Defender.

Use this site key in the HTML code your site serves to users. [See client side integration](#)

COPY SITE KEY

6LfJUbkmAAAAAP4S0EWaPOxMfdSJy6PaPslD_v

Use this secret key for communication between your site and reCAPTCHA. [See server side integration](#)

COPY SECRET KEY

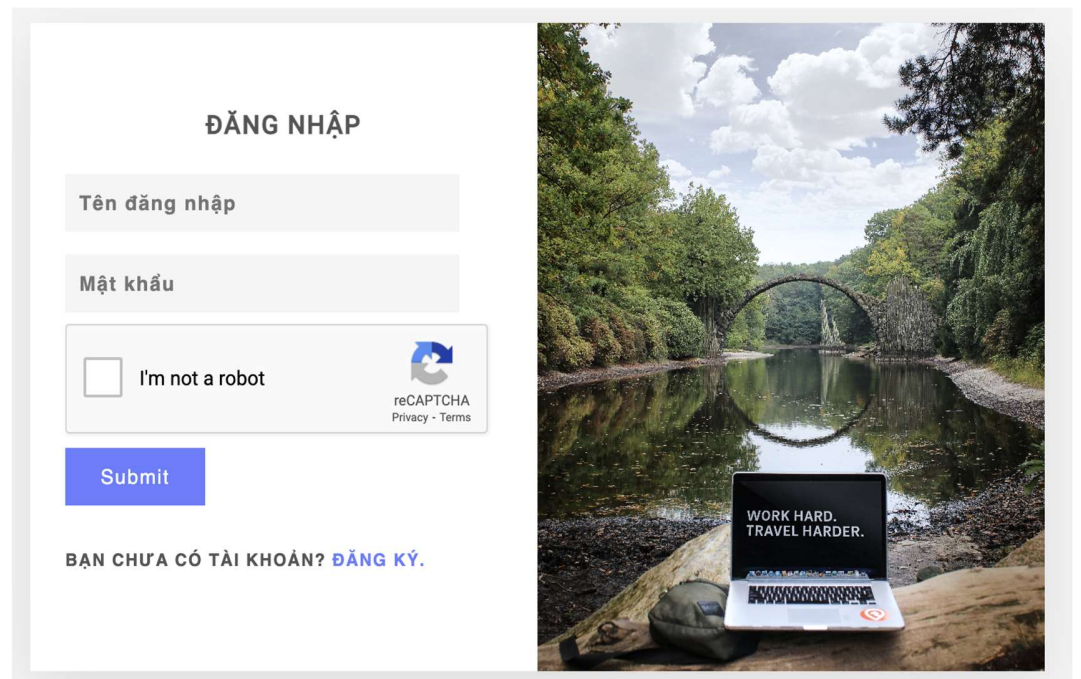
6LfJUbkmAAAAOkiBZBQ4yM5v7acFxmTEVKzCqhE

Từ đây google ReCAPTCHA sẽ cũng cấp cho em giá trị hay key. Em sẽ sử dụng nó để thiết lập google ReCAPTCHA cho ứng dụng.

Em sẽ cài đặt thư viện để thiết lập ReCAPTCHA cho ứng dụng của em. Vì ứng dụng được build từ framework nên em sẽ cài đặt thư viện <https://www.npmjs.com/package/ng-recaptcha>. Sau đó việc còn lại là import các dependency và nhúng component mà thư viện ng-recaptcha cung cấp.

```
<re-captcha
  #captchaRef="reCaptcha"
  siteKey="YOUR_SITE_KEY"
  size="invisible"
  (resolved)="$event && submit($event)"
></re-captcha>
```

Kết quả cuối cùng sẽ là:



4.6.4 Giao thức WebSocket.

Giới thiệu về giao thức WebSocket.

WebSocket là một giao thức giúp truyền dữ liệu hai chiều giữa server-client qua một kết nối TCP duy nhất. Hơn nữa, WebSocket là một giao thức được thiết kế để truyền dữ liệu bằng cách sử dụng cổng 80 và cổng 443 và nó là một phần của HTML5. Vì vậy, webSockets có thể hoạt động trên các cổng web tiêu chuẩn, nên không có rắc rối về việc mở cổng cho các ứng dụng, lo lắng về việc bị chặn bởi các tường lửa hay proxy server

Không giống với giao thức HTTP là cần client chủ động gửi yêu cầu cho server, client sẽ chờ đợi để nhận được dữ liệu từ máy chủ. Hay nói cách khác với giao thức WebSocket thì server có thể chủ động gửi thông tin đến client mà không cần phải có yêu cầu từ client.

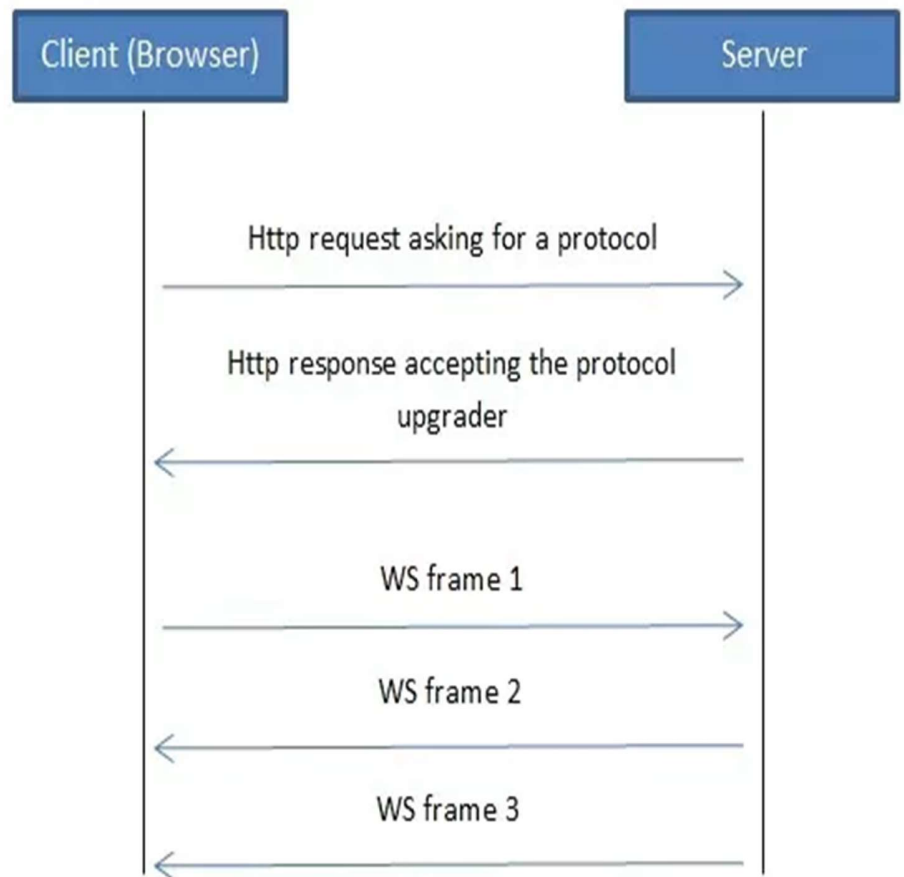
Tất cả dữ liệu giao tiếp giữa client-server sẽ được gửi trực tiếp qua một kết nối cố định làm cho thông tin được gửi đi nhanh chóng và liên tục khi cần thiết. WebSocket làm giảm độ trễ bởi vì một khi kết nối WebSocket được thành lập, server không cần phải chờ đợi cho một yêu cầu từ client.

Tương tự như vậy, client có thể gửi tin nhắn đến server bất cứ lúc nào. Yêu cầu duy nhất này giúp làm giảm đáng kể độ trễ, mà sẽ gửi một yêu cầu trong khoảng thời gian, cho dù thông điệp có sẵn.

Để có thể sử dụng được WebSocket thì không phải chỉ cần trình duyệt hỗ trợ mà còn phải có server WebSocket, server WebSocket có thể được tạo

ra bằng bất kỳ ngôn ngữ server-side nào, nhưng Node.js được sử dụng rộng rãi hơn cả vì nó viết bằng Javascript nên mang nhiều ưu điểm so với các ngôn ngữ server-side truyền thống khác.

- Hoạt động :



Giao thức có hai phần: Bắt tay và truyền dữ liệu Ban đầu client sẽ gửi yêu cầu khởi tạo kết nối websocket đến server, server kiểm tra và gửi trả kết quả chấp nhận kết nối, sau đó kết nối được tạo và quá trình gửi dữ liệu có thể được thực hiện, dữ liệu chính là các Ws frame

Đầu tiên client sẽ gửi một http request yêu cầu nâng cấp

GET /mychat HTTP/1.1

Host: server.example.com

Upgrade: websocket

Connection: Upgrade

Sec-WebSocket-Key: x3JJHMBDL1EzLkh9GBhXDw==

Sec-WebSocket-Protocol: chat

Sec-WebSocket-Version: 13

Origin: http://example.com

server trả về

HTTP/1.1 101 Switching Protocols

Upgrade: websocket

Connection: Upgrade

Sec-WebSocket-Accept: s3pPLMBiTxaQ9kYGzzhZRbK+xOo=

Sec-WebSocket-Protocol: chat

Để xác nhận việc kết nối, client sẽ gửi một giá trị Sec-WebSocket-Key được mã hóa bằng Based64 đến server.

Sau đó bên server sẽ thực hiện:

- Nối thêm chuỗi cố định là “258EAFa5-E914-47DA-95CA-C5AB0DC85B11” vào Sec-WebSocket-Key để được chuỗi mới là “x3JJHmbDL1EzLkh9GBhXDw==258EAFa5-E914-47DA-95CA-C5AB0DC85B11”.
- Thực hiện mã hóa SHA-1 chuỗi trên để được “1d29ab734b0c9585240069a6e4e3e91b61da1969”.
- Mã hóa kết quả vừa nhận được bằng Base64 để được “HSmrc0sMlYUkAGmm5OPpG2HaGWk=”
- Gửi response lại client kèm với giá trị Sec-WebSocket-Accept chính là chuỗi kết quả vừa tạo ra.

Client sẽ kiểm tra status code (phải bằng 101) và Sec-WebSocket-Accept xem có đúng với kết quả mong đợi không và thực hiện kết nối.

Trên thực tế các trường dữ liệu trao đổi có thể khác nhau. Dưới đây là hình ảnh khi client dùng thư viện socket.io mới kết nối đến server socket

× Headers Frames Cookies Timing

▼ General

Request URL: ws://localhost:6969/socket.io/?EIO=3&transport=websocket&sid=6GLS3vfLV7LGIHbaAAAC

Request Method: GET

Status Code: 101 Switching Protocols

▼ Response Headers view source

Connection: Upgrade

Sec-WebSocket-Accept: wcSvn4w8pxg0CqKXuzzXR618x60=

Sec-WebSocket-Extensions: permessage-deflate; client_no_context_takeover

Sec-WebSocket-Version: 13

Upgrade: websocket

WebSocket-Server: uWebSockets

▼ Request Headers view source

Accept-Encoding: gzip, deflate, br

Accept-Language: vi,en;q=0.9,en-US;q=0.8

Cache-Control: no-cache

Connection: Upgrade

Cookie: currentToggle=notice; __ufpc=201710100915333567; __ywapbuk=0.7; _ga=GA1.1.949049508.1507706984; io=6GLS3vfLV7LGIHbaAAAC

Host: localhost:6969

Origin: file://

Pragma: no-cache

Sec-WebSocket-Extensions: permessage-deflate; client_max_window_bits

Sec-WebSocket-Key: o7j1GgyAaQK6wRiCGxrgZQ==

Sec-WebSocket-Version: 13

Upgrade: websocket

User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/63.0.3239.132 Safari/537.36

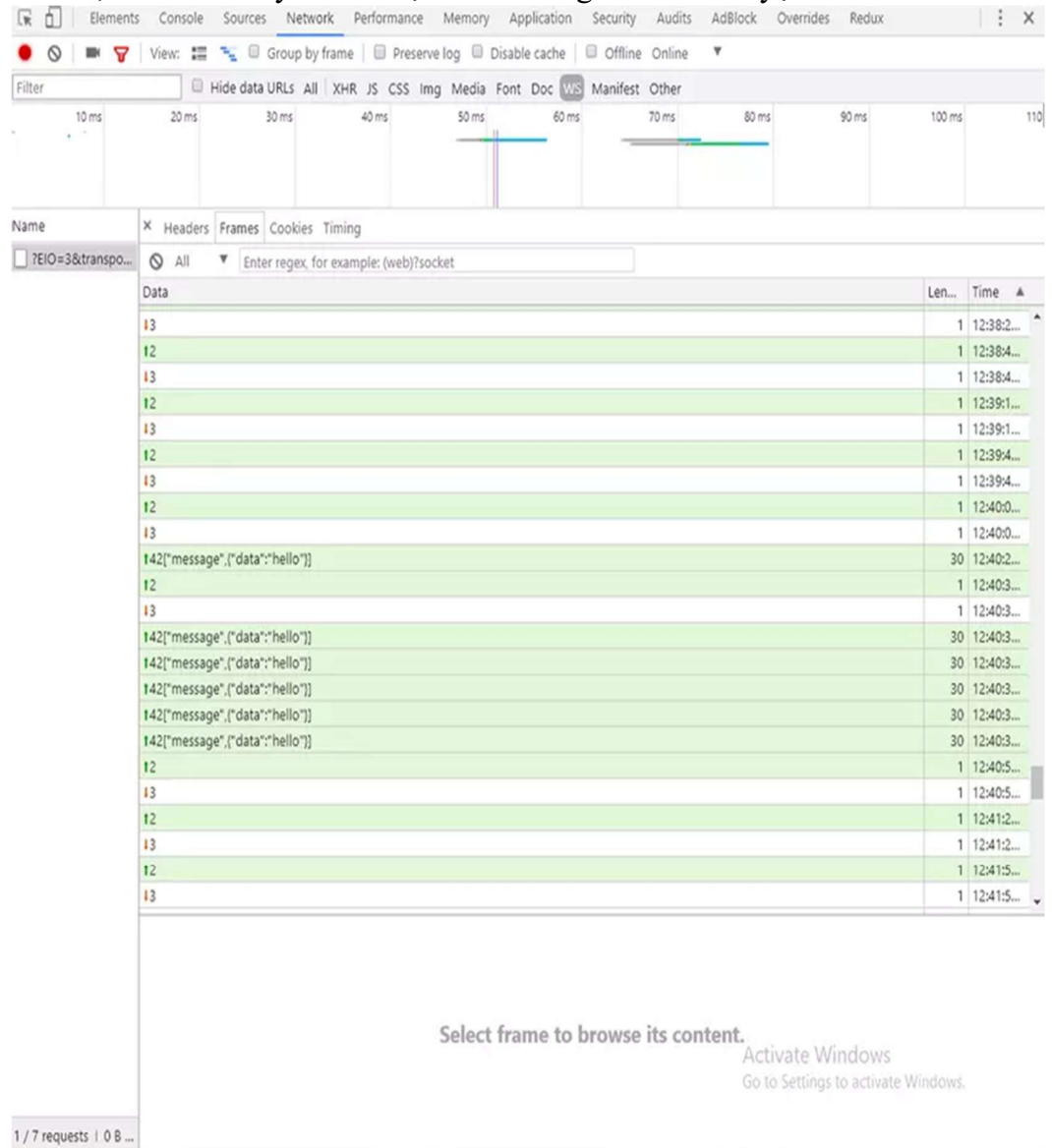
▼ Query String Parameters view source view URL encoded

EIO: 3

transport: websocket

sid: 6GLS3vfLV7LGIHbaAAAC

Đối với việc truyền dữ liệu. Dữ liệu sẽ được truyền thông qua một kết nối duy nhất được tạo ra sau quá trình bắt tay. Dữ liệu được truyền bằng các Frame, ta có thể thấy nó khi bật trình debug của trình duyệt lên



Với việc nhận và gửi tin nhắn ngay lập tức. Ta có thể ứng dụng websocket vào việc xây dựng ứng dụng chat thời gian thực hoặc tính năng hiển thị thông báo khi bạn bè có hoạt động mới trên mạng xã hội.

5. Thách thức và giải pháp

5.1 Bảo mật và quản lý dữ liệu cá nhân.

- Thách thức trong việc bảo mật và quản lý dữ liệu cá nhân.

Ở phần này em xin được nêu ra một số thách thức mà em đã gặp phải trong vấn đề bảo mật và quản lý dữ liệu cá nhân. Đây là những vấn đề mà em thấy thực sự quan trọng và cần có sự quan tâm đúng mức.

- Xác thực và đăng nhập: Việc người dùng đặt mật khẩu quá đơn giản có thể khiến cho kẻ gian có thể đoán được mật khẩu. Ngoài ra kẻ xấu còn có thể sử dụng

một số công cụ Brute force nhằm thử hết tất cả các chuỗi mật khẩu có thể có từ đó dò ra được mật khẩu của người dùng. Đồng thời mật khẩu của người dùng có thể bị lộ lọt nếu như lập trình viên hay các thành phần tham có quyền truy xuất dữ liệu người dùng. Đây cũng là một vấn đề rất quan trọng.

- Bảo vệ dữ liệu cá nhân: Kẻ xấu có thể sử dụng các công cụ hỗ trợ của trình duyệt để lấy được đường dẫn Api của hệ thống từ đó thực hiện các yêu cầu trả về thông tin người dùng tùy ý dựa vào các request parameter. Điều này là rất nguy hiểm.
- Bảo mật trong giao tiếp: Nếu không có các biện pháp ngăn chặn những kẻ xấu hoàn toàn có thể chủ động nhắn tin liên lạc với những người dùng khác như một cách để spam hoặc lan truyền mã độc đến người dùng khác.
- Quản lý quyền riêng tư: Nếu thông tin cá nhân của người dùng không được quản lý đúng cách. Nó hoàn toàn có thể lộ lọt ra ngoài và bị kẻ xấu lợi dụng để thực hiện các hành vi xấu.
- Giải pháp trong việc giải quyết các thách thức gặp phải.
Em sẽ đi qua tức vấn đề thách thức mà em gặp phải ở trên đồng thời nêu ra cách mà em đã giải quyết nó.
- Xác thực và đăng nhập: Để tránh việc người dùng đặt mật khẩu quá dễ đoán dẫn tới kẻ gian có thể khai thác. Em đã ngăn chặn bằng cách yêu cầu người dùng phải đặt mật khẩu có mức độ phức tạp cao từ đó kẻ xấu sẽ khó khăn hơn trong việc đoán mật khẩu người dùng. Một số yêu cầu như mật khẩu phải dài hơn 8 ký tự, có chữ cái in hoa, chữ số và cả ký tự đặc biệt. Đối với trường hợp ngăn chặn Brute force. Em đã sử dụng google recaptcha. Tài khoản sẽ phải xác thực là người dùng bằng cách click vào check box mới có thể đăng nhập được.
- Bảo vệ dữ liệu cá nhân: Với mỗi phiên đăng nhập em sẽ sinh ra cho mỗi phiên một token khác nhau và sẽ sử dụng như một chuỗi định danh mỗi lần gửi yêu cầu truy vấn dữ liệu lên server. Từ đó đảm bảo được tính bảo mật cho cả hệ thống.
- Bảo mật trong giao tiếp: Đối với tính năng nhắn tin. Hiện tại em sẽ chỉ cho phép nhắn tin với những người dùng đã theo dõi nhau. Nhằm đảm bảo sự cho phép giữa các bên tham gia vào đoạn giao tiếp. Tránh trường hợp làm phiền.
- Quản lý quyền riêng tư: Đây là một vấn đề đòi hỏi sự cẩn thận của người lập trình. Với mỗi yêu cầu nhận được từ phía server. Ta chỉ nên trả về những dữ liệu liên quan hay cần thiết đối với yêu cầu đó. Tránh việc trả về dữ liệu thừa hay là các thông tin nhạy cảm như số điện thoại, email, tên đăng nhập, mật khẩu.

5.2 Nhắn tin thời gian thực giữa người dùng với nhau

- Thách thức trong việc nhắn tin thời gian thực giữa người dùng với nhau:
 - Độ trễ mạng: Mạng internet không đảm bảo độ trễ nhất quán và có thể gây ra độ trễ trong việc gửi và nhận tin nhắn. Điều này có thể ảnh hưởng đến trải nghiệm người dùng trong việc nhận thông báo thời gian thực.
 - Đồng bộ hóa dữ liệu: Khi nhiều người dùng cùng tham gia vào cuộc trò chuyện thời gian thực, việc đồng bộ hóa dữ liệu giữa các thiết bị và người dùng trở thành một thách thức.

- Quản lý lưu trữ dữ liệu: Lưu trữ và truy xuất dữ liệu nhắn tin thời gian thực cần được quản lý một cách hiệu quả để đảm bảo tính sẵn sàng và khả năng mở rộng. Giải pháp cho vấn đề này là sử dụng cơ sở dữ liệu phân tán hoặc hệ thống lưu trữ được tối ưu hóa để lưu trữ và truy vấn dữ liệu nhanh chóng và hiệu quả. Giải pháp trong việc giải quyết những thách thức gặp phải khi xây dựng tính năng nhắn tin thời gian thực giữa người dùng với nhau.

- Giải pháp trong việc giải quyết những thách thức gặp phải khi xây dựng tính năng nhắn tin thời gian thực giữa người dùng với nhau:

Em sẽ đi qua tức vấn đề thách thức mà em gặp phải ở trên đồng thời nêu ra cách mà em đã giải quyết nó.

- Độ trễ mạng: Giải pháp cho vấn đề này là sử dụng các giao thức và công nghệ như WebSockets để thiết lập kết nối liên tục và truyền tin nhắn một cách nhanh chóng. Từ đó giảm độ trễ mạng xuống mức thấp nhất.

- Đồng bộ hóa dữ liệu: Giải pháp cho vấn đề này là làm sao nội dung tin nhắn hiển thị chính xác trên các thiết bị và người dùng đích khác nhau. Ở đây em sử dụng phương pháp lưu trữ từng đoạn hội thoại ở cơ sở dữ liệu. Đồng thời với mỗi tin nhắn đi được gửi thành công ở phía server cũng sẽ lưu lại đoạn tin nhắn đó. Về phía người gửi tin nhắn. Nội dung tin nhắn sẽ được lưu tại một biến trong đoạn code, nếu tin nhắn được xác nhận gửi thành công từ phía server, em sẽ cho xuất hiện đoạn tin nhắn lên khung chat. Từ đó đảm bảo dữ liệu xuất hiện với mỗi người dùng là chính xác và giống nhau trong khi người dùng không cần phải refresh page để thấy được sự thay đổi.

- Quản lý lưu trữ dữ liệu: Như đã nói ở giải pháp cho việc giải quyết đồng bộ hóa dữ liệu. Nội dung đoạn hội thoại sẽ phải được lưu lại ở cơ sở dữ liệu. Với mỗi người dùng dữ liệu sẽ trả sao cho phù hợp và hiển thị đúng với đoạn hội thoại người dùng đã thực hiện trước đó.

KẾT QUẢ

Thông qua đồ án này em đã nghiên cứu về phương pháp bảo mật thông tin và qua đó sử dụng ngôn ngữ lập trình Typescript/Javascript cùng với các framework như Angular, NestJS để xây dựng ứng dụng Dung-Social để ứng dụng những kiến thức em đã nghiên cứu vào thực tiễn.

Trong quá trình thực hiện đồ án em đã có thêm kinh nghiệm trong quá trình tìm hiểu các kỹ thuật để xây dựng ứng dụng: kỹ năng tra cứu, tham khảo kiến thức từ các cuốn sách, từ kinh nghiệm từ những người đi trước. Đồng thời trong quá trình không ngừng sửa chữa, cải tiến ứng dụng đã rèn luyện thêm cho ý chí của em một quyết tâm không được bỏ cuộc.

Các chức năng trong ứng dụng chưa được phong phú do nguồn tài nguyên và thời gian có hạn, trong thời gian tới em sẽ tiếp tục nghiên cứu và đầu tư để cải thiện và thêm nhiều chức năng, áp dụng thêm các kỹ thuật mới cho chương trình.

Đặc biệt, em xin bày tỏ sự kính trọng và lòng biết ơn sâu sắc nhất đến thầy giáo hướng dẫn: Ths. Trần Phong Nhã, thầy là người đã trực tiếp hướng dẫn, giúp đỡ cho em để em có thể hoàn thành đồ án này. Trong quá trình học tập và nghiên cứu, nếu em có những sai sót gì, kính mong thầy cô bỏ qua cho em!

Em xin kính chúc các thầy cô luôn luôn khỏe mạnh và ngày một thành công hơn trên con đường giảng dạy của mình.

Em xin trân trọng cảm ơn!

TÀI LIỆU THAM KHẢO

- [1]. Jennifer Robbins, *Learning Web Design 5th Edition*, O'Reilly Media, Inc., 2018
- [2]. Jon Duckett, *HTML and CSS: Design and Build Websites*, Wiley, 2011
- [3]. Mark Myers, *A Smarter Way to Learn JavaScript: The New Tech-Assisted Approach That Requires Half the Effort*, CreateSpace Independent Publishing Platform, 2014
- [4]. Philip Ackermann, *JavaScript: The Comprehensive Guide to Learning Professional JavaScript Programming*, Rheinwerk Computing, 2022
- [5]. Asim Hussain, *Angular: From Theory To Practice*, CodeCraft, 2017
- [6]. Nathan Murray, Felipe Coury, Ari Lerner , Carlos Taborda, *Ng-book: The Complete Guide to Angular 5th Edition*, CreateSpace Independent Publishing Platform, 2018
- [7]. Alan Beaulieu, *Learning SQL (3rd Edition): Generate, Manipulate, and Retrieve Data*, Ascent Audio, 2023
- [8]. Regina O. Obe, Leo S. Hsu, *PostgreSQL: Up and Running 3rd Edition*, O'Reilly Media Inc., 2017
- [9]. Hans-Jürgen Schönig, *Mastering PostgreSQL 12: Advanced techniques to build and administer scalable and reliable PostgreSQL database applications 3rd Edition*, Packt Publishing, 2019
- [10]. “HTML Tutorial”, [Online], Available: <https://www.w3schools.com/html>
- [11]. “HTML: HyperText Markup Language”, [Online], Available: <https://developer.mozilla.org/en-US/docs/Web/HTML>
- [12]. “CSS Documentation”, [Online], Available: <https://devdocs.io/css/>
- [13]. “CSS Tutorial”, [Online], Available: <https://www.tutorialspoint.com/css/index.htm>
- [14]. “W3School CSS Tutorial”, [Online], Available: <https://www.w3schools.com/css/>
- [15]. “HTTP Documentation”, [Online], Available: <https://developer.mozilla.org/en-US/docs/Web/HTTP>
- [16]. “JavaScript Documentation”, [Online], Available: <https://developer.mozilla.org/en-US/docs/Web/JavaScript>
- [17]. “JavaScript Tutorial”, [Online], Available: <https://www.javascripttutorial.net>
- [18]. “TypeScript Documentation”, [Online], Available: <https://www.typescriptlang.org>
- [19]. “TypeScript Tutorial”, [Online], Available: <https://www.typescripttutorial.net>
- [20]. “AngularJS MVC Architecture”, [Online], Available: <https://www.javatpoint.com/angularjs-mvc-architecture>
- [22]. “NestJS Documentation”, [Online], Available: <https://docs.nestjs.com/>