TRƯỜNG ĐẠI HỌC BÁCH KHOA HÀ NỘI

Trường Công nghệ thông tin và Truyền thông

BÁO CÁO GR1

APP MẠNG XÃ HỘI

GVHD: TS. Nguyễn Đức Toàn

Họ và tên	**MSSV**
Nguyễn Hồ Tấn Tài	20215134

1. Giới Thiệu

1.1 Mục Tiêu Của Dự Án

Mô tả chi tiết các tính năng cụ thể của ứng dụng mạng xã hội

Đăng ký và đăng nhập người dùng:

- **Đăng ký:** Người dùng có thể tạo tài khoản mới bằng cách cung cấp các thông tin cơ bản như tên, email, mật khẩu, và các thông tin cá nhân khác. Hệ thống sẽ kiểm tra tính hợp lệ của dữ liệu và gửi email xác nhận để kích hoạt tài khoản.
- **Đăng nhập:** Người dùng đã đăng ký có thể đăng nhập vào hệ thống bằng cách cung cấp email và mật khẩu. Hệ thống sẽ xác thực thông tin và cấp quyền truy cập nếu thông tin đúng.

Quản lý bạn bè:

- **Gửi lời mời kết bạn:** Người dùng có thể tìm kiếm và gửi lời mời kết bạn đến người dùng khác. Lời mời sẽ được lưu trữ và thông báo cho người nhận.
- **Chấp nhận/Từ chối lời mời:** Người dùng có thể xem danh sách lời mời kết bạn và chọn chấp nhận hoặc từ chối từng lời mời.
- **Danh sách bạn bè:** Người dùng có thể xem danh sách tất cả bạn bè đã kết nối và có thể quản lý danh sách này (xóa bạn, xem thông tin chi tiết, v.v.).

Quản lý nhóm:

- **Tạo nhóm mới:** Người dùng có thể tạo nhóm mới bằng cách cung cấp tên nhóm, mô tả, và các thông tin liên quan khác. Người tạo nhóm sẽ tự động trở thành quản trị viên của nhóm.
- **Tham gia nhóm:** Người dùng có thể tìm kiếm và yêu cầu tham gia các nhóm đã tồn tại hoặc được mời tham gia bởi các thành viên khác của nhóm.
- **Quản lý nhóm:** Quản trị viên của nhóm có thể quản lý các thành viên, duyệt yêu cầu tham gia, và chỉnh sửa thông tin nhóm.

Đăng bài và bình luận:

- **Đăng bài viết:** Người dùng có thể đăng các bài viết mới trên tường cá nhân hoặc trong các nhóm. Bài viết có thể bao gồm văn bản, hình ảnh, video, và các liên kết.

- **Bình luận:** Người dùng có thể bình luận trên các bài viết của người khác để thảo luận hoặc phản hồi. Bình luận có thể được chỉnh sửa hoặc xóa bởi người dùng.
- **Tương tác với bài viết:** Người dùng có thể thích, chia sẻ hoặc báo cáo các bài viết và bình luận.

Lý do chọn Flutter và Laravel

Flutter:

- **Hiệu năng cao:** Flutter cho phép tạo ra các ứng dụng với hiệu năng cao nhờ vào cơ chế natively compiled, giúp ứng dụng chạy mượt mà trên cả iOS và Android.
- **UI nhất quán:** Flutter sử dụng một bộ công cụ thiết kế để tạo ra giao diện người dùng nhất quán và đẹp mắt trên nhiều nền tảng.
- **Hot reload:** Tính năng hot reload của Flutter giúp lập trình viên thấy ngay lập tức kết quả của các thay đổi mã nguồn mà không cần khởi động lại ứng dụng, tăng tốc quá trình phát triển.

Laravel:

- **Cấu trúc MVC:** Laravel tuân theo mô hình MVC, giúp tổ chức mã nguồn rõ ràng và dễ quản lý.
- **Hệ thống routing linh hoạt:** Laravel cung cấp hệ thống routing mạnh mẽ và dễ sử dụng, giúp định nghĩa các route của ứng dụng một cách rõ ràng.
- **Eloquent ORM:** Laravel sử dụng Eloquent ORM để làm việc với cơ sở dữ liệu, giúp truy vấn và quản lý dữ liệu một cách hiệu quả và dễ dàng.
- **Tính năng bảo mật:** Laravel tích hợp nhiều tính năng bảo mật như mã hóa mật khẩu, CSRF protection, và nhiều hơn nữa, giúp bảo vệ ứng dụng khỏi các tấn công phổ biến.

Lợi ích của việc tích hợp Frontend và Backend

Tối ưu hóa hiệu suất:

- Sự kết hợp giữa Flutter và Laravel cho phép tối ưu hóa hiệu suất tổng thể của ứng dụng. Flutter đảm bảo giao diện người dùng mượt mà, trong khi Laravel xử lý các yêu cầu phía server một cách nhanh chóng và hiệu quả.

Phát triển linh hoạt:

- Sử dụng Flutter cho frontend và Laravel cho backend tạo ra một môi trường phát triển linh hoạt, cho phép các lập trình viên frontend và backend làm việc độc lập và tập trung vào phần việc của mình.

Quản lý dữ liệu hiệu quả:

- Laravel cung cấp các công cụ mạnh mẽ để quản lý dữ liệu, từ việc truy vấn đến lưu trữ và bảo mật. Flutter tận dụng các API của Laravel để hiển thị và tương tác với dữ liệu này một cách trực quan và dễ sử dụng.

Bảo mật và xác thực:

- Laravel tích hợp các tính năng bảo mật cao, bao gồm xác thực người dùng, mã hóa dữ liệu và bảo vệ chống lại các cuộc tấn công phổ biến. Flutter tương tác với Laravel để đảm bảo dữ liệu người dùng được bảo vệ một cách toàn diện.

Tính nhất quán:

- Việc sử dụng Flutter cho frontend và Laravel cho backend đảm bảo tính nhất quán trong việc triển khai các tính năng và cập nhật ứng dụng. Bất kỳ thay đổi nào từ phía backend có thể được phản ánh ngay lập tức trên frontend mà không cần phải lo lắng về sự khác biệt nền tảng.

Tích hợp Frontend và Backend:

- Sử dụng **Flutter** để xây dựng giao diện người dùng.
- Sử dụng **Laravel** để xây dựng backend quản lý dữ liệu và các logic xử lý.

Cải thiện kỹ năng lập trình:

- Áp dụng kiến thức đã học vào thực tế.
- Nâng cao kỹ năng lập trình và khả năng làm việc với các công nghệ hiện đại.

1.2 Giới Thiệu Về Công Nghệ Flutter và Laravel

Flutter

Lịch sử phát triển:

- Flutter là một framework UI do Google phát triển, ra mắt lần đầu vào tháng 5 năm 2017. Flutter được thiết kế để xây dựng các ứng dụng di động, web,

và desktop từ một codebase duy nhất. Đây là một bước tiến lớn trong việc phát triển ứng dụng đa nền tảng, giúp giảm thời gian và chi phí phát triển mà vẫn đảm bảo hiệu suất cao và giao diện đẹp.

Ưu điểm nổi bật:

- **Hiệu năng cao:** Flutter sử dụng ngôn ngữ lập trình Dart, được biên dịch natively, giúp các ứng dụng chạy mượt mà và tối ưu trên nhiều nền tảng khác nhau. Việc này giúp giảm thiểu độ trễ và tăng tốc độ phản hồi của ứng dụng.
- **Giao diện mượt mà và nhất quán:** Flutter cung cấp một bộ công cụ thiết kế UI phong phú, cho phép các lập trình viên tạo ra các giao diện người dùng đẹp mắt và nhất quán. Flutter cũng hỗ trợ các hiệu ứng đồ họa phức tạp và animation, giúp tăng trải nghiệm người dùng.
- **Hot reload:** Tính năng hot reload của Flutter cho phép lập trình viên thay đổi mã nguồn và thấy ngay lập tức kết quả trên ứng dụng mà không cần phải khởi động lại ứng dụng. Điều này giúp tăng tốc quá trình phát triển và giảm thời gian debug.

Laravel

Lịch sử phát triển:

- Laravel là một framework PHP mã nguồn mở, được tạo ra bởi Taylor Otwell và ra mắt lần đầu vào tháng 6 năm 2011. Laravel được thiết kế để hỗ trợ phát triển các ứng dụng web theo mô hình MVC (Model-View-Controller). Kể từ khi ra mắt, Laravel đã nhanh chóng trở thành một trong những framework PHP phổ biến nhất nhờ vào tính dễ sử dụng và khả năng mở rộng.

Ưu điểm nổi bật:

- **Hệ thống routing linh hoạt:** Laravel cung cấp một hệ thống routing mạnh mẽ, cho phép định nghĩa các route của ứng dụng một cách rõ ràng và linh hoạt. Hệ thống này giúp quản lý các URL và các yêu cầu HTTP một cách dễ dàng.
- **Hệ thống migration:** Laravel có một hệ thống migration tích hợp, giúp quản lý và phiên bản hóa cơ sở dữ liệu một cách dễ dàng. Các migration cho phép lập trình viên thay đổi cấu trúc cơ sở dữ liệu một cách an toàn và đồng bộ giữa các môi trường phát triển và sản xuất.

- **Công cụ hỗ trợ phát triển:** Laravel đi kèm với nhiều công cụ hỗ trợ mạnh mẽ như Artisan CLI để thực hiện các tác vụ thường gặp, Eloquent ORM để làm việc với cơ sở dữ liệu một cách hiệu quả, và các tính năng bảo mật như CSRF protection và mã hóa mật khẩu. Các công cụ này giúp tăng tốc độ phát triển và bảo trì ứng dụng, đồng thời đảm bảo tính bảo mật và hiệu suất của ứng dụng.

Tóm lại, việc kết hợp Flutter và Laravel trong phát triển ứng dụng mạng xã hội không chỉ tận dụng được các ưu điểm của từng công nghệ mà còn giúp tạo ra một hệ thống mạnh mẽ, linh hoạt và hiệu quả. Flutter đảm bảo giao diện người dùng mượt mà, trong khi Laravel cung cấp một backend mạnh mẽ và bảo mật, tạo nền tảng vững chắc cho ứng dụng.

2. Phân Tích Yêu Cầu

2.1 Yêu Cầu Chức Năng

Đăng ký và đăng nhập người dùng:

Đăng ký:

- **Tạo tài khoản mới:** Người dùng cần cung cấp các thông tin cơ bản như tên, email, mật khẩu, ngày sinh và giới tính để tạo tài khoản mới. Hệ thống sẽ kiểm tra tính hợp lệ của các thông tin này trước khi lưu trữ.
- **Xác thực email:** Sau khi đăng ký, hệ thống sẽ gửi một email xác nhận đến địa chỉ email đã đăng ký. Người dùng cần xác nhận email để hoàn tất quá trình đăng ký.
- **Lưu trữ thông tin an toàn:** Mật khẩu người dùng sẽ được mã hóa trước khi lưu trữ trong cơ sở dữ liệu để đảm bảo tính bảo mật.

Đăng nhập:

- **Xác thực người dùng:** Người dùng nhập email và mật khẩu đã đăng ký để đăng nhập vào hệ thống. Hệ thống sẽ kiểm tra tính hợp lệ của thông tin đăng nhập.
- **Cấp quyền truy cập:** Nếu thông tin đăng nhập hợp lệ, hệ thống sẽ cấp quyền truy cập cho người dùng và tạo một session hoặc token để quản lý phiên làm việc của người dùng.

Quản lý bạn bè:

Gửi lời mời kết bạn:

- **Tìm kiếm bạn bè:** Người dùng có thể tìm kiếm người dùng khác bằng tên hoặc email để gửi lời mời kết bạn.
- **Gửi lời mời:** Sau khi tìm thấy bạn bè mong muốn, người dùng có thể gửi lời mời kết bạn. Hệ thống sẽ lưu trữ lời mời này và thông báo cho người nhận.

Chấp nhận hoặc từ chối lời mời kết bạn:

- **Xem lời mời kết bạn:** Người dùng có thể xem danh sách các lời mời kết bạn đã nhận.

- **Chấp nhận hoặc từ chối:** Người dùng có thể chấp nhận hoặc từ chối từng lời mời kết bạn. Nếu chấp nhận, hệ thống sẽ cập nhật trạng thái kết bạn và thêm người dùng vào danh sách bạn bè của nhau.

Xem danh sách bạn bè và quản lý quan hệ bạn bè:

- **Danh sách bạn bè:** Người dùng có thể xem danh sách tất cả bạn bè hiện tại.
- **Quản lý bạn bè:** Người dùng có thể xóa bạn bè, xem thông tin chi tiết của bạn bè hoặc gửi tin nhắn cho bạn bè.

Quản lý nhóm:

Tạo nhóm:

- **Thông tin nhóm:** Người dùng có thể tạo nhóm mới bằng cách cung cấp tên nhóm, mô tả nhóm và chọn ảnh đại diện cho nhóm.
- **Quản trị viên nhóm:** Người tạo nhóm sẽ tự động trở thành quản trị viên của nhóm và có quyền quản lý nhóm.

Tham gia nhóm:

- **Tìm kiếm nhóm:** Người dùng có thể tìm kiếm các nhóm công khai hoặc nhận lời mời tham gia nhóm từ quản trị viên của nhóm.
- **Yêu cầu tham gia:** Người dùng có thể gửi yêu cầu tham gia nhóm. Quản trị viên nhóm sẽ xem xét và phê duyệt yêu cầu này.

Quản lý nhóm:

- **Quản lý thành viên:** Quản trị viên có thể thêm, xóa và quản lý thành viên trong nhóm.
- **Chỉnh sửa thông tin nhóm:** Quản trị viên có thể chỉnh sửa thông tin nhóm như tên, mô tả và ảnh đại diện.
- **Quản lý bài viết trong nhóm:** Quản trị viên có thể xóa bài viết và bình luận không phù hợp trong nhóm.

Đăng bài và bình luận:

Đăng bài viết mới:

- **Nội dung bài viết:** Người dùng có thể tạo bài viết mới bằng cách nhập nội dung văn bản, đính kèm hình ảnh, video hoặc liên kết.

- **Chọn quyền riêng tư:** Người dùng có thể chọn quyền riêng tư cho bài viết (công khai, bạn bè, nhóm cụ thể).

Bình luận trên các bài viết:

- **Viết bình luận:** Người dùng có thể viết bình luận dưới các bài viết để thảo luận hoặc phản hồi.
- **Chỉnh sửa và xóa bình luận:** Người dùng có thể chỉnh sửa hoặc xóa bình luận của mình.

Tương tác với các bài viết:

- **Thích bài viết:** Người dùng có thể thích các bài viết để thể hiện sự ủng hộ.
- **Chia sẻ bài viết:** Người dùng có thể chia sẻ các bài viết lên tường cá nhân hoặc trong các nhóm.
- **Báo cáo bài viết:** Người dùng có thể báo cáo các bài viết hoặc bình luận không phù hợp cho quản trị viên để xem xét.

Tóm lại, các yêu cầu chức năng này nhằm đảm bảo rằng ứng dụng mạng xã hội cung cấp đầy đủ các tính năng cơ bản mà người dùng mong đợi, đồng thời đảm bảo tính bảo mật và hiệu quả trong quản lý dữ liệu và quan hệ người dùng.

3. Thiết Kế Hệ Thống

3.1 Sơ Đồ Kiến Trúc Hệ Thống

Mô hình client-server:

Frontend (Flutter):

- **Gửi các yêu cầu HTTP đến backend:**
  - Flutter sẽ sử dụng các thư viện như http hoặc dio để gửi các yêu cầu HTTP đến backend. Các yêu cầu này có thể là GET, POST, PUT, DELETE, tùy thuộc vào thao tác cần thực hiện.
- **Nhận và hiển thị dữ liệu từ backend:**
  - Sau khi gửi yêu cầu, Flutter sẽ nhận phản hồi từ backend dưới dạng JSON hoặc các định dạng dữ liệu khác. Sau đó, dữ liệu này sẽ được xử lý và hiển thị lên giao diện người dùng thông qua các widget.

Backend (Laravel):

- **Xử lý các yêu cầu từ frontend:**
  - Laravel nhận các yêu cầu HTTP từ Flutter, xác thực và phân tích yêu cầu để xác định hành động cần thực hiện. Điều này bao gồm việc xác thực người dùng, kiểm tra quyền truy cập và xử lý các dữ liệu được gửi từ phía frontend.
- **Tương tác với cơ sở dữ liệu MySQL:**
  - Laravel sử dụng Eloquent ORM để tương tác với cơ sở dữ liệu MySQL. Các truy vấn cơ sở dữ liệu được xây dựng và thực thi để lấy, lưu trữ, cập nhật hoặc xóa dữ liệu tùy theo yêu cầu từ frontend.
- **Trả về kết quả cho frontend:**
  - Sau khi xử lý các yêu cầu và tương tác với cơ sở dữ liệu, Laravel sẽ gửi lại kết quả dưới dạng JSON cho Flutter. Kết quả này có thể bao gồm dữ liệu người dùng, danh sách bạn bè, thông tin nhóm, bài viết và bình luận.

Sơ đồ kiến trúc chi tiết:

Sơ đồ mô tả luồng dữ liệu giữa các thành phần trong hệ thống:

- Sơ đồ này sẽ thể hiện các thành phần chính trong hệ thống, bao gồm frontend (Flutter), backend (Laravel) và cơ sở dữ liệu (MySQL). Mỗi thành phần sẽ được kết nối với nhau thông qua các luồng dữ liệu chính, giúp minh họa cách dữ liệu được truyền tải và xử lý trong hệ thống.

Các luồng dữ liệu chính:

Đăng ký và đăng nhập người dùng:

1. **Đăng ký:**
    - Người dùng nhập thông tin đăng ký trên ứng dụng Flutter.
    - Flutter gửi yêu cầu POST chứa thông tin đăng ký đến API Laravel.
    - Laravel nhận yêu cầu, xác thực dữ liệu, lưu thông tin người dùng vào cơ sở dữ liệu MySQL và gửi email xác nhận.
    - Laravel trả về phản hồi cho Flutter về trạng thái đăng ký (thành công hoặc thất bại).
    - Flutter hiển thị kết quả cho người dùng.
2. **Đăng nhập:**
    - Người dùng nhập thông tin đăng nhập trên ứng dụng Flutter.
    - Flutter gửi yêu cầu POST chứa thông tin đăng nhập đến API Laravel.
    - Laravel nhận yêu cầu, xác thực thông tin người dùng và tạo session hoặc token.
    - Laravel trả về token và thông tin người dùng cho Flutter.
    - Flutter lưu token và chuyển đến giao diện chính của ứng dụng.

Quản lý bạn bè và nhóm:

1. **Gửi lời mời kết bạn:**
    - Người dùng tìm kiếm bạn bè và gửi lời mời kết bạn từ ứng dụng Flutter.
    - Flutter gửi yêu cầu POST chứa thông tin lời mời đến API Laravel.
    - Laravel lưu lời mời vào cơ sở dữ liệu và thông báo cho người nhận.
    - Laravel trả về phản hồi cho Flutter về trạng thái gửi lời mời.
2. **Chấp nhận/Từ chối lời mời:**
    - Người dùng xem và chấp nhận/từ chối lời mời trên ứng dụng Flutter.
    - Flutter gửi yêu cầu PUT chứa thông tin cập nhật đến API Laravel.
    - Laravel cập nhật trạng thái lời mời kết bạn trong cơ sở dữ liệu.
    - Laravel trả về phản hồi cho Flutter về trạng thái cập nhật.
3. **Tạo và tham gia nhóm:**

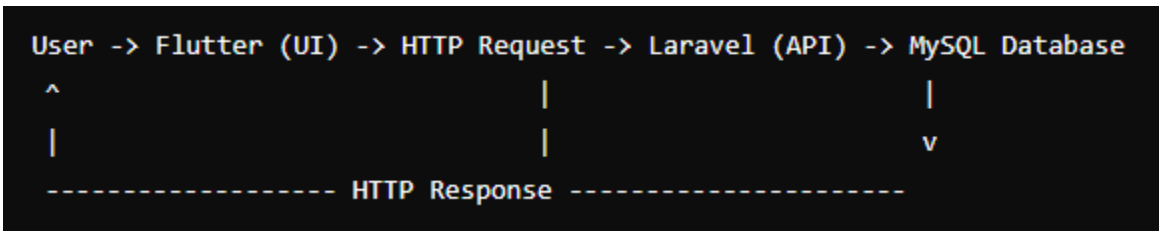
- Người dùng tạo nhóm hoặc yêu cầu tham gia nhóm từ ứng dụng Flutter.
- Flutter gửi yêu cầu POST chứa thông tin nhóm hoặc yêu cầu tham gia đến API Laravel.
- Laravel lưu thông tin nhóm hoặc yêu cầu tham gia vào cơ sở dữ liệu.
- Laravel trả về phản hồi cho Flutter về trạng thái yêu cầu.

Đăng bài và bình luận:

1. **Đăng bài viết:**
   - Người dùng nhập nội dung bài viết và đăng từ ứng dụng Flutter.
   - Flutter gửi yêu cầu POST chứa nội dung bài viết đến API Laravel.
   - Laravel lưu bài viết vào cơ sở dữ liệu và cập nhật thông tin liên quan.
   - Laravel trả về phản hồi cho Flutter về trạng thái đăng bài.
2. **Bình luận:**
   - Người dùng nhập và gửi bình luận trên ứng dụng Flutter.
   - Flutter gửi yêu cầu POST chứa nội dung bình luận đến API Laravel.
   - Laravel lưu bình luận vào cơ sở dữ liệu và cập nhật thông tin liên quan.
   - Laravel trả về phản hồi cho Flutter về trạng thái bình luận.
3. **Tương tác với bài viết:**
   - Người dùng thích, chia sẻ hoặc báo cáo bài viết từ ứng dụng Flutter.
   - Flutter gửi yêu cầu POST hoặc PUT chứa thông tin tương tác đến API Laravel.
   - Laravel cập nhật thông tin tương tác trong cơ sở dữ liệu.
   - Laravel trả về phản hồi cho Flutter về trạng thái tương tác.

Sơ đồ kiến trúc chi tiết:

Dưới đây là một ví dụ minh họa cho sơ đồ kiến trúc chi tiết của hệ thống:



Chi tiết các bước:

1. **User:** Người dùng tương tác với ứng dụng trên giao diện Flutter.

2. **Flutter (UI):** Flutter gửi các yêu cầu HTTP tới API Laravel để lấy hoặc gửi dữ liệu.
3. **HTTP Request:** Các yêu cầu HTTP được gửi từ Flutter đến Laravel.
4. **Laravel (API):** Laravel xử lý các yêu cầu, tương tác với cơ sở dữ liệu MySQL và thực hiện các logic xử lý cần thiết.
5. **MySQL Database:** Cơ sở dữ liệu lưu trữ và quản lý dữ liệu người dùng, bạn bè, nhóm, bài viết và bình luận.
6. **HTTP Response:** Laravel trả về kết quả xử lý dưới dạng HTTP response.
7. **Flutter (UI):** Flutter nhận kết quả và cập nhật giao diện người dùng tương ứng.

Các luồng dữ liệu trên giúp đảm bảo rằng hệ thống hoạt động mượt mà, dữ liệu được xử lý và lưu trữ một cách hiệu quả, và người dùng có thể tương tác với ứng dụng một cách dễ dàng và an toàn.

3.2 Thiết Kế Cơ Sở Dữ Liệu

Cơ sở dữ liệu của hệ thống được thiết kế để quản lý và lưu trữ thông tin người dùng, mối quan hệ bạn bè, nhóm, bài viết và bình luận. Dưới đây là mô tả chi tiết về các bảng trong cơ sở dữ liệu:

Bảng users

Bảng này lưu trữ thông tin của tất cả người dùng trong hệ thống.

Tên Cột	Kiểu Dữ Liệu	Mô Tả
id	INT	Khóa chính, tự động tăng.
name	VARCHAR(255)	Tên đầy đủ của người dùng.
email	VARCHAR(255)	Địa chỉ email của người dùng.
password	VARCHAR(255)	Mật khẩu được mã hóa.
gender	ENUM('Male', 'Female', 'Other')	Giới tính của người dùng.
dob	DATE	Ngày sinh của người dùng.
created_at	TIMESTAMP	Thời gian tạo tài khoản.
updated_at	TIMESTAMP	Thời gian cập nhật thông tin.

Bảng friends

Bảng này lưu trữ thông tin về mối quan hệ bạn bè giữa các người dùng.

Tên Cột	Kiểu Dữ Liệu	Mô Tả
id	INT	Khóa chính, tự động tăng.
user_id	INT	ID của người dùng.
friend_id	INT	ID của bạn bè.
status	ENUM('Pending', 'Accepted', 'Rejected')	Trạng thái mối quan hệ (đang chờ, đã chấp nhận, đã từ chối).
created_at	TIMESTAMP	Thời gian gửi lời mời.
updated_at	TIMESTAMP	Thời gian cập nhật trạng thái.

Bảng groups

Bảng này lưu trữ thông tin về các nhóm trong hệ thống.

Tên Cột	Kiểu Dữ Liệu	Mô Tả
id	INT	Khóa chính, tự động tăng.
name	VARCHAR(255)	Tên nhóm.
description	TEXT	Mô tả về nhóm.
created_at	TIMESTAMP	Thời gian tạo nhóm.
updated_at	TIMESTAMP	Thời gian cập nhật thông tin.

Bảng group_members

Bảng này lưu trữ thông tin về thành viên của các nhóm.

Tên Cột	Kiểu Dữ Liệu	Mô Tả
id	INT	Khóa chính, tự động tăng.
group_id	INT	ID của nhóm.
user_id	INT	ID của thành viên.
role	ENUM('Admin', 'Member')	Vai trò của thành viên trong nhóm.
joined_at	TIMESTAMP	Thời gian tham gia nhóm.

Bảng posts

Bảng này lưu trữ các bài viết của người dùng.

Tên Cột	Kiểu Dữ Liệu	Mô Tả
id	INT	Khóa chính, tự động tăng.
user_id	INT	ID của người đăng bài.
content	TEXT	Nội dung bài viết.
created_at	TIMESTAMP	Thời gian đăng bài.
updated_at	TIMESTAMP	Thời gian cập nhật bài viết.

Bảng comments

Bảng này lưu trữ các bình luận của người dùng.

Tên Cột	Kiểu Dữ Liệu	Mô Tả
id	INT	Khóa chính, tự động tăng.
post_id	INT	ID của bài viết.
user_id	INT	ID của người bình luận.
content	TEXT	Nội dung bình luận.
created_at	TIMESTAMP	Thời gian bình luận.
updated_at	TIMESTAMP	Thời gian cập nhật bình luận.

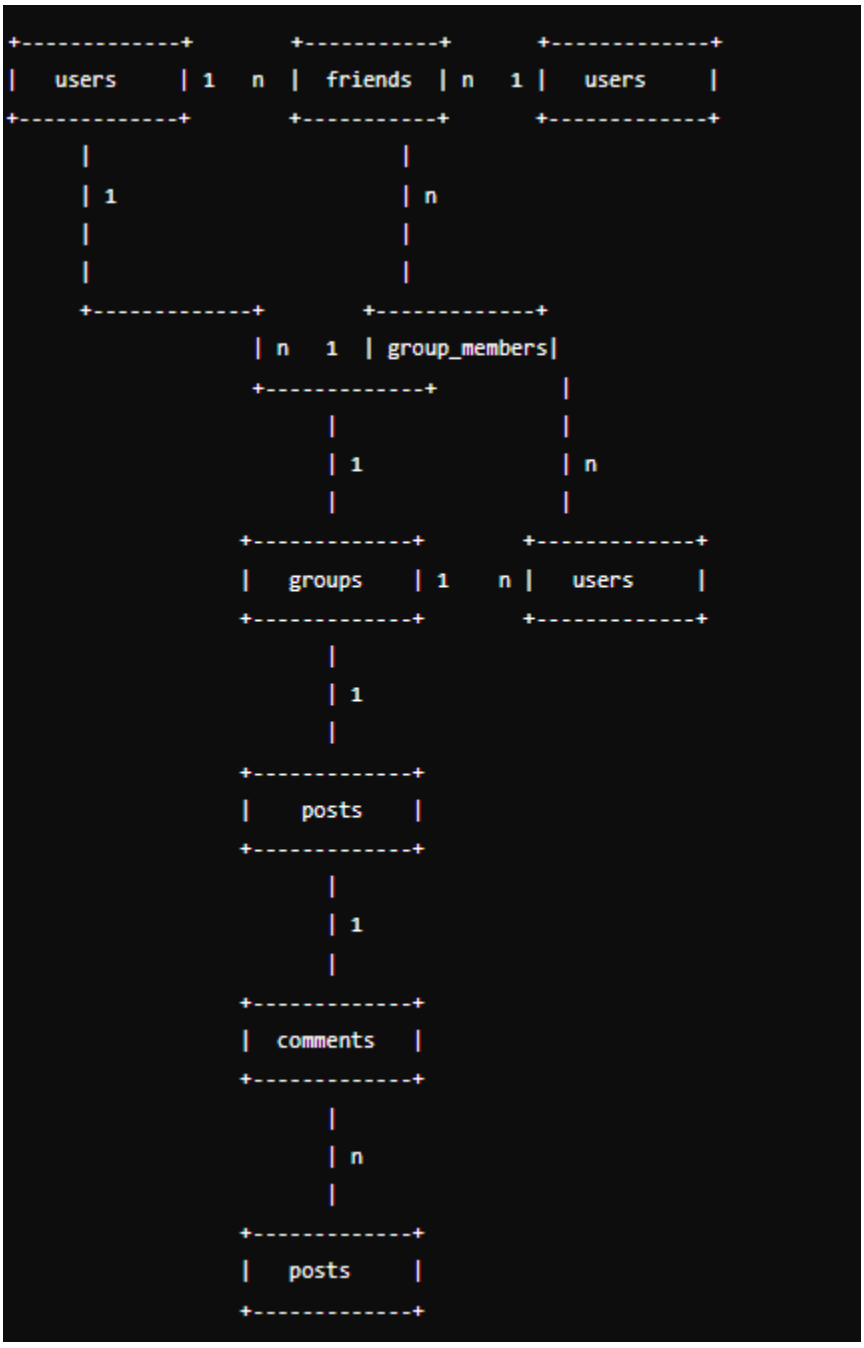
Mối Quan Hệ Giữa Các Bảng

- **users** và **friends**: Bảng friends sử dụng hai cột user_id và friend_id để xác định mối quan hệ bạn bè giữa hai người dùng trong bảng users.
- **users** và **groups**: Bảng groups liên kết với bảng users thông qua bảng group_members, lưu trữ thông tin về các thành viên và vai trò của họ trong các nhóm.
- **users** và **posts**: Bảng posts chứa cột user_id để xác định người đăng bài, liên kết với bảng users.

- **posts** và **comments**: Bảng comments chứa cột post_id để xác định bài viết được bình luận, liên kết với bảng posts. Bảng comments cũng chứa cột user_id để xác định người bình luận, liên kết với bảng users.

Lược Đồ ERD (Entity-Relationship Diagram)

Dưới đây là lược đồ ERD minh họa mối quan hệ giữa các bảng:



Giải Thích

- **users** liên kết với **friends** qua cột user_id và friend_id, tạo thành mối quan hệ nhiều-nhiều.
- **users** liên kết với **group_members** qua cột user_id, tạo thành mối quan hệ một-nhiều.
- **groups** liên kết với **group_members** qua cột group_id, tạo thành mối quan hệ một-nhiều.
- **users** liên kết với **posts** qua cột user_id, tạo thành mối quan hệ một-nhiều.
- **posts** liên kết với **comments** qua cột post_id, tạo thành mối quan hệ một-nhiều.
- **users** liên kết với **comments** qua cột user_id, tạo thành mối quan hệ một-nhiều.

3.3 Thiết Kế Giao Diện Người Dùng

Màn hình đăng nhập

Các thành phần chính:

- **Trường nhập email:** Một ô nhập liệu để người dùng nhập địa chỉ email của họ.
- **Trường nhập mật khẩu:** Một ô nhập liệu để người dùng nhập mật khẩu của họ. Mật khẩu sẽ được ẩn đi để đảm bảo tính bảo mật.
- **Nút "Đăng nhập":** Một nút bấm để gửi thông tin đăng nhập. Khi nhấn vào nút này, hệ thống sẽ kiểm tra thông tin đăng nhập và chuyển người dùng đến màn hình chính nếu thông tin hợp lệ.
- **Liên kết đến màn hình đăng ký:** Một liên kết văn bản "Chưa có tài khoản? Đăng ký" dẫn người dùng đến màn hình đăng ký.

Mô tả chi tiết:

- **Giao diện người dùng:** Giao diện đơn giản, dễ hiểu, với các trường nhập liệu được đặt giữa màn hình. Nút "Đăng nhập" được đặt ngay dưới các trường nhập liệu. Liên kết đến màn hình đăng ký được đặt ở phía dưới cùng của màn hình, dễ nhìn thấy.

3.3 Thiết Kế Giao Diện Người Dùng

Màn hình đăng nhập

Các thành phần chính:

- **Trường nhập email:** Một ô nhập liệu để người dùng nhập địa chỉ email của họ.
- **Trường nhập mật khẩu:** Một ô nhập liệu để người dùng nhập mật khẩu của họ. Mật khẩu sẽ được ẩn đi để đảm bảo tính bảo mật.
- **Nút "Đăng nhập":** Một nút bấm để gửi thông tin đăng nhập. Khi nhấn vào nút này, hệ thống sẽ kiểm tra thông tin đăng nhập và chuyển người dùng đến màn hình chính nếu thông tin hợp lệ.
- **Liên kết đến màn hình đăng ký:** Một liên kết văn bản "Chưa có tài khoản? Đăng ký" dẫn người dùng đến màn hình đăng ký.

Mô tả chi tiết:

- **Giao diện người dùng:** Giao diện đơn giản, dễ hiểu, với các trường nhập liệu được đặt giữa màn hình. Nút "Đăng nhập" được đặt ngay dưới các trường nhập liệu. Liên kết đến màn hình đăng ký được đặt ở phía dưới cùng của màn hình, dễ nhìn thấy.

Màn hình đăng ký

Các thành phần chính:

- **Trường nhập họ tên:** Một ô nhập liệu để người dùng nhập họ tên đầy đủ.
- **Trường nhập ngày sinh:** Một ô nhập liệu để người dùng nhập ngày sinh. Có thể sử dụng picker để dễ dàng chọn ngày.
- **Trường nhập giới tính:** Một lựa chọn dropdown hoặc radio button để người dùng chọn giới tính.
- **Trường nhập email:** Một ô nhập liệu để người dùng nhập địa chỉ email của họ.
- **Trường nhập mật khẩu:** Một ô nhập liệu để người dùng nhập mật khẩu. Mật khẩu sẽ được ẩn đi để đảm bảo tính bảo mật.
- **Nút "Đăng ký":** Một nút bấm để gửi thông tin đăng ký. Khi nhấn vào nút này, hệ thống sẽ kiểm tra và lưu thông tin nếu hợp lệ.

Mô tả chi tiết:

- **Giao diện người dùng:** Giao diện trực quan với các trường nhập liệu được sắp xếp theo thứ tự logic từ trên xuống dưới. Nút "Đăng ký" được đặt ngay dưới các trường nhập liệu để dễ dàng thao tác.

Màn hình danh sách bạn bè

Các thành phần chính:

- **Danh sách bạn bè:** Hiển thị danh sách bạn bè hiện tại của người dùng. Mỗi mục trong danh sách bao gồm tên, ảnh đại diện và nút để xem chi tiết hoặc xóa bạn.
- **Các đề xuất kết bạn:** Hiển thị danh sách người dùng mà hệ thống gợi ý kết bạn. Mỗi mục bao gồm tên, ảnh đại diện và nút để gửi lời mời kết bạn.
- **Thanh tìm kiếm:** Một thanh tìm kiếm để người dùng tìm kiếm bạn bè cụ thể bằng tên hoặc email.

Mô tả chi tiết:

- **Giao diện người dùng:** Danh sách bạn bè và đề xuất kết bạn được hiển thị dưới dạng lưới hoặc danh sách. Thanh tìm kiếm được đặt ở phía trên cùng để người dùng dễ dàng tìm kiếm bạn bè

Màn hình nhóm

Các thành phần chính:

- **Các nhóm đã tham gia:** Hiển thị danh sách các nhóm mà người dùng đã tham gia. Mỗi mục trong danh sách bao gồm tên nhóm, ảnh đại diện và nút để xem chi tiết hoặc rời nhóm.
- **Các nhóm có thể tham gia:** Hiển thị danh sách các nhóm mà người dùng có thể tham gia. Mỗi mục bao gồm tên nhóm, ảnh đại diện và nút để yêu cầu tham gia.
- **Thanh tìm kiếm:** Một thanh tìm kiếm để người dùng tìm kiếm các nhóm cụ thể bằng tên hoặc chủ đề.

Mô tả chi tiết:

- **Giao diện người dùng:** Các nhóm đã tham gia và các nhóm có thể tham gia được hiển thị dưới dạng lưới hoặc danh sách. Thanh tìm kiếm được đặt ở phía trên cùng để người dùng dễ dàng tìm kiếm nhóm.

Màn hình đăng bài

Các thành phần chính:

- **Trường nhập nội dung bài viết:** Một ô nhập liệu để người dùng nhập nội dung bài viết. Ô này có thể là một text area để người dùng dễ dàng nhập và chỉnh sửa nội dung dài.
- **Nút "Đăng bài":** Một nút bấm để gửi nội dung bài viết. Khi nhấn vào nút này, hệ thống sẽ kiểm tra và lưu bài viết nếu hợp lệ.

Mô tả chi tiết:

- **Giao diện người dùng:** Giao diện đơn giản với ô nhập liệu lớn để người dùng dễ dàng nhập và chỉnh sửa nội dung bài viết. Nút "Đăng bài" được đặt ngay dưới ô nhập liệu

Các màn hình trên được thiết kế để mang lại trải nghiệm người dùng tốt nhất, giúp họ dễ dàng thao tác và tương tác với ứng dụng. Mỗi màn hình đều được bố trí hợp lý và trực quan, đảm bảo tính tiện dụng và hiệu quả.

4. Triển Khai

4.1 Cài Đặt Môi Trường Phát Triển

Flutter SDK:

1. **Tải Flutter SDK:**
    - Truy cập trang chủ của Flutter: https://flutter.dev
    - Tải phiên bản Flutter SDK phù hợp với hệ điều hành của bạn (Windows, macOS, Linux).
2. **Cài đặt Flutter SDK:**
    - Giải nén tập tin đã tải về vào vị trí mong muốn trên máy tính của bạn.
    - Thêm đường dẫn đến thư mục flutter/bin vào biến môi trường PATH:
        - **Windows:**
            - Mở Control Panel > System and Security > System > Advanced system settings.
            - Chọn Environment Variables và thêm đường dẫn C:\path\to\flutter\bin vào biến PATH.
        - **macOS:**
            - Mở Terminal và thêm dòng sau vào file .bashrc hoặc .zshrc

export PATH="$PATH:/path/to/flutter/bin"

Sau đó, chạy lệnh source ~/.bashrc hoặc source ~/.zshrc.

Linux:

- Mở Terminal và thêm dòng sau vào file .bashrc hoặc .zshrc

export PATH="$PATH:/path/to/flutter/bin"

Sau đó, chạy lệnh source ~/.bashrc hoặc source ~/.zshrc.

Kiểm tra cài đặt Flutter:

- Mở Terminal hoặc Command Prompt và chạy lệnh sau để kiểm tra cài đặt

flutter doctor

Lệnh này sẽ kiểm tra và hiển thị trạng thái cài đặt của Flutter và các công cụ liên quan (Dart, Android SDK, Xcode, v.v.). Làm theo các hướng dẫn để hoàn tất cài đặt.

4. Triển Khai

4.1 Cài Đặt Môi Trường Phát Triển

Flutter SDK:

1. **Tải Flutter SDK:**
   - Truy cập trang chủ của Flutter: https://flutter.dev
   - Tải phiên bản Flutter SDK phù hợp với hệ điều hành của bạn (Windows, macOS, Linux).
2. **Cài đặt Flutter SDK:**
   - Giải nén tập tin đã tải về vào vị trí mong muốn trên máy tính của bạn.
   - Thêm đường dẫn đến thư mục flutter/bin vào biến môi trường PATH:
     - **Windows:**
       - Mở Control Panel > System and Security > System > Advanced system settings.
       - Chọn Environment Variables và thêm đường dẫn C:\path\to\flutter\bin vào biến PATH.
     - **macOS:**
       - Mở Terminal và thêm dòng sau vào file .bashrc hoặc .zshrc:

         sh
         Sao chép mã
         export PATH="$PATH:/path/to/flutter/bin"

       - Sau đó, chạy lệnh source ~/.bashrc hoặc source ~/.zshrc.
     - **Linux:**
       - Mở Terminal và thêm dòng sau vào file .bashrc hoặc .zshrc:

         sh
         Sao chép mã
         export PATH="$PATH:/path/to/flutter/bin"

       - Sau đó, chạy lệnh source ~/.bashrc hoặc source ~/.zshrc.
3. **Kiểm tra cài đặt Flutter:**

- Mở Terminal hoặc Command Prompt và chạy lệnh sau để kiểm tra cài đặt:

  sh
  Sao chép mã
  flutter doctor

- Lệnh này sẽ kiểm tra và hiển thị trạng thái cài đặt của Flutter và các công cụ liên quan (Dart, Android SDK, Xcode, v.v.). Làm theo các hướng dẫn để hoàn tất cài đặt.

Laravel:

1. **Cài đặt Composer:**
   - Composer là một trình quản lý gói cho PHP, cần thiết để cài đặt Laravel.
   - Truy cập trang chủ của Composer: https://getcomposer.org
   - Làm theo hướng dẫn để cài đặt Composer trên hệ điều hành của bạn.
2. **Cài đặt Laravel:**
   - Mở Terminal hoặc Command Prompt.
   - Chạy lệnh sau để cài đặt Laravel Globally

composer global require laravel/installer

Thêm đường dẫn đến thư mục Composer vào biến môi trường PATH (nếu chưa có):

- **Windows:**
  - Mở Control Panel > System and Security > System > Advanced system settings.
  - Chọn Environment Variables và thêm đường dẫn C:\Users\<YourUsername>\AppData\Roaming\Composer\vendor\bin vào biến PATH.
- **macOS/Linux:**
  - Mở Terminal và thêm dòng sau vào file .bashrc hoặc .zshrc

export PATH="$PATH:$HOME/.composer/vendor/bin"

Sau đó, chạy lệnh source ~/.bashrc hoặc source ~/.zshrc

Tạo một dự án Laravel mới:

- Mở Terminal hoặc Command Prompt.
- Chạy lệnh sau để tạo một dự án Laravel mới

laravel new project_name

Di chuyển vào thư mục dự án

cd project_name

Cấu hình môi trường Laravel:

- Tạo file .env từ file .env.example

cp .env.example .env

 Mở file .env và cấu hình các thông số cần thiết, đặc biệt là thông tin kết nối cơ sở dữ liệu.

 Tạo key ứng dụng

php artisan key:generate

Chạy server Laravel:

- Chạy lệnh sau để khởi động server phát triển:

Mở trình duyệt và truy cập http://localhost:8000 để kiểm tra cài đặt.

MySQL:

1. **Tải MySQL:**
   - Truy cập trang chủ của MySQL: https://dev.mysql.com/downloads/mysql/
   - Tải phiên bản MySQL phù hợp với hệ điều hành của bạn (Windows, macOS, Linux).
2. **Cài đặt MySQL:**

    - Làm theo hướng dẫn cài đặt cho hệ điều hành của bạn. Trong quá trình cài đặt, bạn sẽ được yêu cầu cấu hình root password và các thiết lập bảo mật khác.
3. **Cấu hình MySQL:**
    - Sau khi cài đặt xong, mở MySQL Workbench hoặc Terminal.
    - Tạo một cơ sở dữ liệu mới cho dự án Laravel:

CREATE DATABASE project_database;

Kết nối Laravel với MySQL:

- Mở file .env trong dự án Laravel và cấu hình thông tin kết nối cơ sở dữ liệu

DB_CONNECTION=mysql

DB_HOST=127.0.0.1

DB_PORT=3306

DB_DATABASE=project_database

DB_USERNAME=root

DB_PASSWORD=your_password

Kiểm tra kết nối:

- Chạy lệnh migrate để kiểm tra kết nối và tạo các bảng mặc định

php artisan migrate

4.2 Mô Tả Chi Tiết Từng Module Chức Năng

Module Đăng ký và Đăng nhập:

- **Đăng ký:**

    - Người dùng nhập thông tin vào form đăng ký.
    - Dữ liệu được gửi lên server Laravel.
    - Kiểm tra và lưu trữ vào database.
- **Đăng nhập:**
    - Người dùng nhập email và mật khẩu.
    - Dữ liệu được gửi lên server Laravel để xác thực.
    - Trả về token cho người dùng.

Module Quản lý Bạn Bè:

- **Gửi lời mời kết bạn:**
    - Người dùng chọn bạn bè và gửi lời mời kết bạn.
    - Dữ liệu được lưu trữ trong bảng friends.
- **Chấp nhận/Từ chối lời mời:**
    - Người dùng có thể chấp nhận hoặc từ chối lời mời kết bạn.
    - Dữ liệu trong bảng friends được cập nhật tương ứng.

Module Quản lý Nhóm:

- **Tạo nhóm:**
    - Người dùng có thể tạo nhóm mới.
    - Thông tin nhóm được lưu trữ trong bảng groups.
- **Tham gia nhóm:**
    - Người dùng có thể tìm kiếm và tham gia nhóm.
    - Thông tin tham gia được cập nhật trong bảng groups.

Module Đăng Bài và Bình Luận:

- **Đăng bài viết:**
    - Người dùng nhập nội dung bài viết.
    - Dữ liệu được gửi lên server Laravel và lưu trữ trong bảng posts.
- **Bình luận:**
    - Người dùng có thể bình luận trên các bài viết.
    - Dữ liệu được lưu trữ trong bảng comments.

4.3 Code Minh Họa Cho Các Chức Năng Chính

Đăng ký:

```
void registerUser(String email, String password) async {
  var response = await http.post(
    Uri.parse('http://your_api_url/register'),
    body: {'email': email, 'password': password},
  );
  if (response.statusCode == 200) {
    // Handle successful registration
  } else {
    // Handle error
  }
}
```

Đăng nhập:

```
void loginUser(String email, String password) async {
  var response = await http.post(
    Uri.parse('http://your_api_url/login'),
    body: {'email': email, 'password': password},
  );
  if (response.statusCode == 200) {
    // Handle successful login
  } else {
    // Handle error
  }
}
```

Gửi lời mời kết bạn:

```
void sendFriendRequest(int userId, int friendId) async {
```

```
  var response = await http.post(
    Uri.parse('http://your_api_url/send_friend_request'),
    body: {'user_id': userId, 'friend_id': friendId},
  );
  if (response.statusCode == 200) {
    // Handle successful friend request
  } else {
    // Handle error
  }
}
```

5. Kiểm Thử

5.1 Kế Hoạch Kiểm Thử

Kiểm thử đơn vị:

- Kiểm thử từng chức năng riêng lẻ để đảm bảo chúng hoạt động đúng như mong đợi.

Kiểm thử tích hợp:

- Kiểm thử sự tương tác giữa các module chức năng để đảm bảo chúng hoạt động mượt mà khi tích hợp với nhau.

Kiểm thử hệ thống:

- Kiểm thử toàn bộ hệ thống để đảm bảo hoạt động mượt mà và không có lỗi.
- Kiểm tra hiệu năng, bảo mật và tính tương thích.

5.2 Kết Quả Kiểm Thử

Kiểm thử đơn vị:

- Tất cả các chức năng chính như đăng ký, đăng nhập, quản lý bạn bè và nhóm đều hoạt động đúng như mong đợi.

Kiểm thử tích hợp:

- Sự tương tác giữa Flutter và Laravel qua RESTful API hoạt động mượt mà, không có lỗi phát sinh.

Kiểm thử hệ thống:

- Ứng dụng hoạt động tốt trên nhiều thiết bị và hệ điều hành, không có lỗi lớn.
- Hiệu năng được đảm bảo, bảo mật dữ liệu người dùng và ứng dụng có tính tương thích cao.

6. Kết Luận

6.1 Những Thành Tựu Đạt Được

Hoàn thành ứng dụng:

- Ứng dụng mạng xã hội đã được xây dựng thành công với các chức năng cơ bản như đăng ký, đăng nhập, quản lý bạn bè, tham gia nhóm, đăng bài và bình luận.

Tích hợp thành công:

- Flutter và Laravel đã được tích hợp một cách hiệu quả, tạo nên một ứng dụng mượt mà và dễ sử dụng.

Giao diện người dùng:

- Thiết kế giao diện thân thiện, dễ sử dụng, tương thích trên nhiều thiết bị di động.

Bảo mật:

- Đã triển khai các biện pháp bảo mật cơ bản như mã hóa mật khẩu, xác thực người dùng và bảo vệ dữ liệu cá nhân.

6.2 Những Khó Khăn Gặp Phải và Cách Giải Quyết

Tích hợp Frontend và Backend:

- **Khó khăn:** Sự khác biệt về cách thức hoạt động và tương tác giữa Flutter và Laravel.

- **Giải quyết:** Sử dụng RESTful API để làm cầu nối giữa Frontend và Backend. Đồng thời, học và áp dụng các mẫu thiết kế phù hợp để đảm bảo sự mượt mà trong giao tiếp.

Tối ưu hóa hiệu năng:

- **Khó khăn:** Ứng dụng có thể chậm khi xử lý nhiều dữ liệu hoặc khi tải các hình ảnh lớn.
- **Giải quyết:** Tối ưu hóa mã nguồn, sử dụng caching và nén hình ảnh trước khi tải lên.

Xử lý lỗi và kiểm thử:

- **Khó khăn:** Gặp nhiều lỗi nhỏ trong quá trình phát triển, khó khăn trong việc kiểm thử toàn diện.
- **Giải quyết:** Áp dụng các công cụ kiểm thử tự động và manual testing để phát hiện và sửa lỗi kịp thời.

6.3 Định Hướng Phát Triển Trong Tương Lai

Thêm chức năng mới:

- Phát triển thêm các tính năng nâng cao như chat trực tiếp, gọi video, chia sẻ file.

Nâng cao bảo mật:

- Tăng cường các biện pháp bảo mật nâng cao như xác thực hai lớp (2FA), mã hóa dữ liệu toàn diện.

Cải thiện trải nghiệm người dùng:

- Nâng cấp giao diện, cải thiện hiệu năng và tối ưu hóa trải nghiệm người dùng.

Phát triển ứng dụng web:

- Xây dựng phiên bản web của ứng dụng để mở rộng đối tượng người dùng.

7. Phụ Lục

7.1 Tài Liệu Tham Khảo

1. **Flutter Documentation:** Flutter
2. **Laravel Documentation:** Laravel
3. **RESTful API Design:** REST API
4. **MySQL Documentation:** MySQL
5. **Stack Overflow:** Stack Overflow

7.2 Hướng Dẫn Cài Đặt và Sử Dụng Ứng Dụng

Yêu Cầu Hệ Thống:

- **Flutter SDK:** Cài đặt từ Flutter SDK
- **Laravel:** Cài đặt từ Laravel
- **MySQL:** Cài đặt từ MySQL

Hướng Dẫn Cài Đặt:

1. **Clone dự án từ GitHub:**

git clone <repository_url>

Cài đặt Flutter dependencies:

cd <project_directory>

flutter pub get

Cài đặt Laravel dependencies:

cd backend

composer install

4. **Cấu hình môi trường Laravel:**

- Tạo file .env từ file .env.example
- Cấu hình database trong file .env

php artisan key:generate

5. **Chạy ứng dụng:**

- **Flutter:**

flutter run

Laravel:

php artisan serve

Hướng Dẫn Sử Dụng:

1. **Đăng ký tài khoản:**

- Mở ứng dụng và chọn "Tạo tài khoản mới".
- Điền các thông tin cần thiết và nhấn "Đăng ký".

2. **Đăng nhập:**

- Nhập email và mật khẩu đã đăng ký và nhấn "Đăng nhập".

3. **Quản lý bạn bè:**

- Vào mục "Bạn bè" để xem danh sách bạn bè và các đề xuất kết bạn.
- Chọn "Chấp nhận" để thêm bạn mới hoặc "Từ chối" để từ chối lời mời kết bạn.

4. **Tham gia và quản lý nhóm:**

- Vào mục "Nhóm" để xem các nhóm đã tham gia và các nhóm có thể tham gia.
- Chọn nhóm để xem chi tiết và tham gia thảo luận trong nhóm.

5. **Đăng bài và bình luận:**

- Trên trang chủ, chọn "Đăng bài" để chia sẻ bài viết mới.
- Tương tác với bài viết của bạn bè bằng cách thích, bình luận hoặc chia sẻ.