

ĐẠI HỌC PHENIKAA
TRƯỜNG CÔNG NGHỆ THÔNG TIN PHENIKAA



ĐỒ ÁN LIÊN NGÀNH

**PU Connection - Nền tảng kết nối sinh viên
Phenikaa University**

Giảng viên hướng dẫn: TS.Vũ Quang Dũng
Sinh viên thực hiện: Quách Gia Bảo - 22010039
Hoàng Văn Bắc – 22010356
Lương Văn Thắng – 22010192
Khóa: K16 – 2022 – 2026
Lớp tín chỉ: Đồ án liên ngành-1-3-24(N05)
Chương trình đào tạo: Công nghệ thông tin

Hà Nội, tháng 11 năm 2025

Lời cảm ơn

Trong suốt quá trình thực hiện đồ án liên ngành, nhóm chúng em không chỉ tích lũy được kiến thức chuyên môn quý giá, mà còn thấm thía được ý nghĩa của sự kiên trì, tinh thần sáng tạo và trách nhiệm trong học tập. Tất cả những điều đó có được là nhờ sự hướng dẫn tận tâm và đầy nhiệt huyết của thầy **Vũ Quang Dũng**.

Thầy không chỉ là người đồng hành về mặt kỹ thuật, mà còn là nguồn động lực mạnh mẽ giúp chúng em tự tin hơn vào khả năng của bản thân. Mỗi buổi trao đổi, từng góp ý chuyên sâu, hay những lần thầy định hướng giải pháp đều là bước đệm quan trọng giúp chúng em vượt qua khó khăn, nhìn nhận vấn đề một cách logic và khoa học hơn.

Sự nghiêm túc trong chuyên môn, tinh thần trách nhiệm, cùng cách thầy khơi gợi tư duy phản biện đã tạo nên một môi trường học tập đầy cảm hứng. Nhờ sự dấn dắt tận tình của thầy, chúng em đã có thể kết hợp giữa lý thuyết và thực tiễn một cách vững chắc, xây dựng được một hệ thống mang tính ứng dụng cao. Qua đó, nhóm cũng trưởng thành hơn trong kỹ năng làm việc nhóm, xử lý vấn đề và quản lý tiến độ.

Chúng em xin gửi đến thầy lòng biết ơn sâu sắc nhất – không chỉ vì sự hỗ trợ về mặt học thuật, mà còn vì những giá trị về tinh thần nghề nghiệp và sự tận tụy mà thầy luôn thể hiện trong suốt hành trình thực hiện đồ án.

Với tất cả sự trân trọng, chúng em chân thành cảm ơn thầy.

Nhóm sinh viên thực hiện

Lời cam đoan

Chúng em xin cam đoan rằng báo cáo đồ án liên ngành này là kết quả của quá trình học tập, nghiên cứu nghiêm túc và nỗ lực không ngừng của cả nhóm. Các nội dung trình bày trong báo cáo — từ phần mô tả, phân tích, thiết kế hệ thống, đến phần triển khai kỹ thuật — đều do chính nhóm thực hiện dựa trên kiến thức đã học và những tài liệu tham khảo có kiểm chứng.

Tất cả dữ liệu, hình ảnh, đoạn mã hoặc ý tưởng từ các nguồn bên ngoài đều đã được chúng em trích dẫn đầy đủ, rõ ràng và phù hợp với chuẩn mực học thuật. Chúng em hiểu rõ vai trò và trách nhiệm của người học trong việc bảo đảm tính trung thực, bản quyền và giá trị khoa học trong từng phần trình bày.

Đồ án không chỉ phản ánh kết quả chuyên môn mà còn là minh chứng cho tinh thần làm việc nhóm, sự chủ động trong tư duy và quá trình rèn luyện tư cách người học. Chúng em hoàn toàn chịu trách nhiệm trước Nhà trường và giảng viên hướng dẫn về tính chính xác và nội dung của toàn bộ báo cáo.

Hà Nội, ngày . . . tháng . . . năm 2025

Phân công nhiệm vụ đồ án

Danh sách các công việc / nhiệm vụ

TT	Công việc	Mô tả tóm tắt công việc (đúng theo dự án Phenikaa Connect)
1	Phân tích yêu cầu	Thu thập và phân tích yêu cầu từ bối cảnh sinh viên Phenikaa; xác định chức năng chính (bài đăng, sự kiện, nhóm học tập, câu lạc bộ, thông báo); xây dựng Use-case, mô hình dữ liệu và sơ đồ hệ thống tổng quan.
2	Thiết kế kiến trúc & lựa chọn công nghệ	Xác định kiến trúc ứng dụng Flutter + Supabase; chọn công nghệ gồm: Flutter (frontend), Supabase (backend + database PostgreSQL + Auth), Storage, Realtime, Edge Functions; thiết kế database schema và RLS policies.
3	Thiết lập Backend trên Supabase	Tạo project Supabase; cấu hình Authentication; chạy các script SQL (schema.sql, functions.sql, mock_data.sql); xây dựng trigger tự động, stored procedures, indexes, và cài đặt chính sách bảo mật.
4	Phát triển API Services trong Flutter	Tạo lớp SupabaseService; xây dựng các phương thức CRUD cho posts, comments, questions, study groups, clubs, events; xử lý đăng nhập/đăng ký; quản lý session; đồng bộ real-time.
5	Phát triển giao diện Flutter (Frontend)	Thiết kế giao diện người dùng (UI) theo style của Phenikaa; xây dựng các màn hình chính: đăng nhập, trang chủ, bài đăng, câu hỏi, nhóm học tập, sự kiện, câu lạc bộ, thông báo; đảm bảo responsive trên mobile & web.
6	Quản lý trạng thái & logic ứng dụng	Tích hợp AppProvider; xây dựng luồng xử lý state, loading, error handling; đồng bộ dữ liệu Supabase Realtime; tối ưu hiệu năng tải dữ liệu (pagination, caching nhẹ).
7	Tính năng nâng cao của Phenikaa Connect	<ul style="list-style-type: none"> - Hệ thống thông báo sự kiện. - Tự động cập nhật số lượng like, tham gia (via trigger). - Bộ lọc và phân loại nội dung (theo câu lạc bộ, môn học, sự kiện).
8	Kiểm thử hệ thống	Viết test cases chức năng; kiểm thử API (authentication, posts, events); kiểm thử giao diện; test realtime; kiểm thử người dùng (UAT); sửa lỗi và tối ưu module quan trọng.
9	Tài liệu và báo cáo	Soạn tài liệu kỹ thuật Supabase; tài liệu hướng dẫn sử dụng ứng dụng; viết báo cáo đồ án chi tiết; ghi chú quá trình implement; chuẩn bị slide thuyết trình và video demo.
10	Quản lý dự án & triển khai	Quản lý timeline 12 tuần theo Agile; phân công công việc rõ ràng cho các thành viên; triển khai Supabase lên môi trường production; build Flutter Web + Android APK; kiểm tra sau triển khai.

Phân công nhiệm vụ cho từng thành viên

TT	MSSV	Họ và tên	Nhiệm vụ và nội dung đã thực hiện (đúng với dự án Phenikaa Connect)	Đánh giá
1	22010039	Quách Gia Bảo (Leader)	<ul style="list-style-type: none"> • Phân tích yêu cầu hệ thống, xác định đầy đủ các module: bài đăng, sự kiện, nhóm học tập, câu lạc bộ, chat, thông báo, người dùng. • Thiết kế kiến trúc tổng thể ứng dụng Flutter + Supabase (database, authentication, storage, realtime, edge functions). • Thiết kế database schema PostgreSQL gồm đầy đủ các bảng: users, posts, comments, events, study groups, clubs, messages, announcements,... • Thiết lập Supabase project: cấu hình Authentication, RLS, Policies, Functions, Triggers, Indexes. • Phát triển các APIs thông qua Supabase service: posts, questions, events, study groups, messages, users. • Tích hợp AppProvider, xây dựng state management, xử lý realtime, loading, error handling. • Thiết kế logic điều hướng, kiến trúc thư mục, build hệ thống router trong Flutter. • Triển khai ứng dụng: build APK, build web, cấu hình môi trường production của Supabase. • Quản lý tiến độ dự án, phân công công việc, họp nhóm, ra quyết định kỹ thuật. 	100%

2	22010356	Hoàng Văn Bắc (Thành viên)	<ul style="list-style-type: none"> • Hỗ trợ phân tích yêu cầu và xây dựng Use-case, Activity Diagram, Sequence Diagram. • Thiết kế giao diện người dùng bằng Flutter: trang chủ, trang bài đăng, câu hỏi, câu lạc bộ, nhóm học tập, thông báo,... • Xây dựng UI Components: PostCard, EventCard, InputField, CommentBox, NavigationBar,... • Tham gia phát triển các tính năng chính: CRUD bài đăng, like/unlike, bình luận, xem câu hỏi, xem nhóm học tập. • Kiểm thử giao diện, sửa lỗi UI/UX, đảm bảo ứng dụng responsive trên nhiều thiết bị (mobile, tablet, web). • Viết tài liệu hướng dẫn sử dụng giao diện người dùng, chụp ảnh minh họa cho báo cáo. 	100%
3	22010192	Lương Văn Thắng (Thành viên)	<ul style="list-style-type: none"> • Viết và chạy các file SQL: schema.sql, functions.sql, mock_data.sql trong Supabase. • Phát triển Supabase Functions: auto-update counters, triggers like/unlike, triggers join event/study group. • Xây dựng các module backend trong Flutter thông qua SupabaseService: events, study groups, questions, announcements. • Xử lý Authentication: đăng ký, đăng nhập, đăng xuất, quản lý session. • Thực hiện kiểm thử hệ thống: test API, test realtime, debug lỗi kết nối Supabase, test RLS Policies. • Biên soạn một phần tài liệu kỹ thuật: cấu trúc database, mô tả policies, hướng dẫn setup Supabase. • Chuẩn bị video demo và hỗ trợ hoàn thiện báo cáo cuối kỳ. 	100%

Mục Lục

Lời cảm ơn	1
Lời cam đoan	2
Phân công nhiệm vụ đồ án	3
Danh mục hình ảnh.....	4
Danh mục từ viết tắt.....	5
1 Giới thiệu	6
1.1 Đặt vấn đề.....	6
1.2 Các giải pháp đã có	6
1.3 Giải pháp đề xuất.....	7
2 Thiết kế	9
2.1 Các yêu cầu chức năng.....	9
2.2 Các yêu cầu phi chức năng.....	16
2.3 Các ràng buộc (Constraints)	21
2.3.1 Các ràng buộc về triển khai	21
2.3.2 Các ràng buộc kinh tế.....	23
2.3.3 Các ràng buộc về đạo đức.....	23
2.4 Mô hình hệ thống / Thiết kế giải pháp.....	24
2.4.1 Các kịch bản của hệ thống (Use-cases)	24
2.4.2 Mô hình Use-case	24
2.4.3 Đặc tả Use-case	38
2.4.4 Các biểu đồ tuần tự (Sequence Diagrams).....	41
2.4.5 Sơ đồ Database (DB Diagram)	60
2.4.6 Các màn hình giao diện người dùng.....	66
3 Cài đặt, kiểm thử và triển khai.....	90
3.1 Môi trường phát triển.....	90
3.2 Cấu trúc dự án	94
3.3 Thuật toán sử dụng.....	98
3.3.1 Cơ chế Tăng/Giảm Counter Tự động (Real-time Counters)	98
3.3.2 Thuật toán Xác thực và Ủy quyền (Authentication & Authorization)	99
3.3.3 Quy trình CRUD cho Tính năng Cộng đồng (Post/Question/Event)	100
3.4 Kiểm thử hệ thống.....	101

3.5 Triển khai hệ thống.....	115
4 Một số thành phần khác của đồ án	120
4.1. Kế hoạch dự án	120
4.2. Đản bảo làm việc nhóm	126
4.3. Đạo đức và làm việc chuyên nghiệp.....	128
4.4. Tác động xã hội	129
4.5. Kế hoạch học tập và phát triển kĩ năng	129
5 Tài liệu tham khảo	131

Danh mục hình ảnh

Hình 1 Usecase tổng quan.....	31
Hình 2 Usecase phân rã quản lý tài khoản	33
Hình 3 Usecase phân rã quản trị viên	34
Hình 4 Usecase phân rã tương tác cộng đồng.....	35
Hình 5 Usecase phân rã tham gia đời sống trường	36
Hình 6 Usecase phân rã Quản lý thông tin chung.....	37
Hình 7 Usecase phân rã Câu lạc bộ và sự kiện	37
Hình 8 Usecase phân rã Quản lý học tập	38
Hình 9 Biểu đồ tuần tự đăng ký tài khoản	41
Hình 10 Biểu đồ tuần tự đăng nhập	42
Hình 11 Sơ đồ tuần tự đăng bài viết	43
Hình 12 Biểu đồ tuần tự thích bài viết	44
Hình 13 Biểu đồ tuần tự tham gia sự kiện	45
Hình 14 Biểu đồ tuần tự tham gia câu lạc bộ	46
Hình 15 Biểu đồ tuần tự xem thông báo từ trường	47
Hình 16 Biểu đồ tuần tự xem thông báo từ trường	48
Hình 17 Biểu đồ tuần tự Trả lời câu hỏi	49
Hình 18 Biểu đồ tuần tự Tạo câu hỏi	50
Hình 19 Biểu đồ tuần tự Đăng ký tham gia Hoạt động CLB	51
Hình 20 Biểu đồ tuần tự Đăng bài trong CLB	52
Hình 21 Biểu đồ tuần tự Tạo nhóm học tập	53
Hình 22 Biểu đồ tuần tự Tham gia nhóm học tập	54
Hình 23 Biểu đồ tuần tự Xem lịch học cá nhân	55
Hình 24 Biểu đồ tuần tự Quản lý Người dùng	56
Hình 25 Biểu đồ tuần tự Quản lý Khóa học	57
Hình 26 Biểu đồ tuần tự Duyệt/Tù chối Bài viết Cộng đồng	58
Hình 27 Biểu đồ tuần tự Hoạt động CLB	59
Hình 28 Sơ đồ Database (DB Diagram)	65
Hình 29 Màn hình đăng nhập.....	66
Hình 30 Màn hình đăng ký	67
Hình 31 Màn hình trang chủ	69
Hình 32 Màn hình học tập	72
Hình 33 Màn hình cộng đồng-Tất cả	73
Hình 34 Màn hình đời sống	75
Hình 35 Màn hình cá nhân.....	77
Hình 36 Màn hình tổng quan của Admin.....	79
Hình 37 Màn hình Quản lý thông báo của Admin.....	81
Hình 38 Màn hình quản lý sự kiện của Admin	81
Hình 39 Màn hình Quản lý người dùng của Admin	83
Hình 40 Màn hình Quản lý CLB của Admin	84
Hình 41 Màn hình quản lý Tổng quan của Chủ nhiệm CLB	85
Hình 42 Màn hình quản lý Hoạt động của Chủ nhiệm CLB	86
Hình 43 Màn hình quản lý Sự kiện của Chủ nhiệm CLB	87
Hình 44 Màn hình quản lý Bài viết của Chủ nhiệm CLB	88
Hình 45 Màn hình quản lý Bài viết của Chủ nhiệm CLB	89

Danh mục từ viết tắt

Từ viết tắt	Nghĩa tiếng Anh	Nghĩa tiếng Việt
API	Application Programming Interface	Giao diện lập trình ứng dụng
CRUD	Create – Read – Update – Delete	Tạo – Đọc – Cập nhật – Xóa
DB	Database	Cơ sở dữ liệu
SQL	Structured Query Language	Ngôn ngữ truy vấn có cấu trúc
JSON	JavaScript Object Notation	Định dạng trao đổi dữ liệu
JWT	JSON Web Token	Token xác thực người dùng
RLS	Row Level Security	Bảo mật cấp hàng trong cơ sở dữ liệu
ID	Identifier	Mã định danh
UI	User Interface	Giao diện người dùng
UX	User Experience	Trải nghiệm người dùng
OTP	One-Time Password	Mã xác thực một lần
SDK	Software Development Kit	Bộ công cụ phát triển
HTTP	HyperText Transfer Protocol	Giao thức truyền tải dữ liệu
HTTPS	HyperText Transfer Protocol Secure	Giao thức truyền tải dữ liệu bảo mật
RT	Realtime	Thời gian thực
PG	PostgreSQL	Hệ quản trị cơ sở dữ liệu PostgreSQL
BaaS	Backend as a Service	Backend dưới dạng dịch vụ
FCM	Firebase Cloud Messaging	Dịch vụ gửi thông báo đẩy
Supabase	—	Nền tảng backend sử dụng trong dự án

1 Giới thiệu

1.1 Đặt vấn đề

Trong bối cảnh chuyển đổi số diễn ra mạnh mẽ trong lĩnh vực giáo dục đại học, việc xây dựng một nền tảng số thống nhất nhằm hỗ trợ sinh viên trong quá trình học tập, kết nối và tương tác xã hội trở nên vô cùng cần thiết. Hiện nay, sinh viên Trường Đại học Phenikaa phải sử dụng đồng thời nhiều kênh thông tin khác nhau như hệ thống quản lý học tập (LMS), website đào tạo, nhóm Facebook, Zalo hoặc các ứng dụng riêng lẻ của trường để theo dõi thông báo, lịch học, điểm số hay tham gia các hoạt động cộng đồng. Việc sử dụng quá nhiều nền tảng khiến thông tin bị phân tán, thiếu sự đồng bộ và gây khó khăn trong việc quản lý dữ liệu cũng như nâng cao trải nghiệm của người học.

Bên cạnh đó, các kênh thông tin hiện có chưa đáp ứng được đầy đủ nhu cầu của sinh viên trong việc kết nối, chia sẻ tài nguyên học tập, hình thành nhóm học tập hoặc tổ chức sự kiện trong phạm vi trường. Sinh viên thường gặp khó khăn khi tìm kiếm thông tin học phần, cập nhật lịch học, liên hệ giảng viên hoặc tham gia vào các hoạt động câu lạc bộ. Ngoài ra, các vấn đề thực tế trong đời sống sinh viên như đi chung xe, đồ thất lạc, thông báo khẩn,... chưa có hệ thống nào quản lý và hỗ trợ hiệu quả.

Vì vậy, việc xây dựng một ứng dụng di động tổng hợp, cho phép sinh viên tiếp cận toàn bộ thông tin học tập, cộng đồng và tiện ích trong cùng một giao diện, là nhu cầu cấp thiết. Ứng dụng cần không chỉ hỗ trợ các chức năng học thuật cơ bản như theo dõi tiến độ học tập, đăng ký học phần hay xem kết quả học tập, mà còn tạo môi trường kết nối sinh viên với nhau thông qua các tính năng như bảng tin, chat, nhóm học, câu lạc bộ và sự kiện. Đồng thời, hệ thống cần đảm bảo tính bảo mật, khả năng mở rộng, và cung cấp trải nghiệm người dùng hiện đại, thân thiện.

Từ những lý do trên, nhóm đề xuất phát triển ứng dụng Phenikaa Connect — một nền tảng số toàn diện dành cho sinh viên Đại học Phenikaa. Ứng dụng hướng tới việc tích hợp các chức năng quản lý học tập, tương tác xã hội và tiện ích sinh viên trong một hệ sinh thái thống nhất, đồng thời áp dụng các công nghệ hiện đại như Flutter cho frontend và Supabase cho backend. Giải pháp này không chỉ giúp tối ưu hóa việc quản lý thông tin và hoạt động sinh viên, mà còn góp phần thúc đẩy quá trình chuyển đổi số trong môi trường giáo dục của nhà trường.

1.2 Các giải pháp đã có

Trước khi phát triển ứng dụng Phenikaa Connect, nhiều trường đại học trong và ngoài nước đã triển khai các hệ thống hoặc ứng dụng riêng lẻ nhằm hỗ trợ sinh viên trong việc quản lý học tập và kết nối thông tin. Tuy nhiên, phần lớn các giải pháp này chỉ đáp ứng một số khía cạnh cụ thể của nhu cầu sinh viên, chưa hình thành được một nền tảng số thống nhất, đa chức năng và tích hợp toàn diện.

Hiện nay, sinh viên Đại học Phenikaa phải sử dụng nhiều kênh khác nhau để phục vụ các mục đích riêng biệt. Ví dụ, việc đăng ký học phần và xem điểm số được thực hiện trên hệ thống quản lý đào tạo (LMS), trong khi việc trao đổi học tập hoặc tổ chức sự kiện thường diễn ra trên các nền tảng mạng xã hội như Facebook hoặc Zalo. Ngoài ra, các thông báo quan trọng từ nhà trường thường được gửi qua email hoặc trang web chính thức, dẫn đến tình trạng thông tin bị phân tán và sinh viên phải truy cập nhiều hệ thống khác nhau để nắm bắt được toàn bộ nội dung cần thiết. Điều này không chỉ gây bất tiện trong quá trình sử dụng, mà còn làm giảm hiệu quả truyền tải thông tin giữa nhà trường và sinh viên.

Một số ứng dụng quản lý sinh viên của các trường khác tại Việt Nam đã được triển khai, chẳng hạn như MyBK (Đại học Bách Khoa TP.HCM) hay UEL Student (Đại học Kinh tế – Luật). Các ứng dụng này hỗ trợ sinh viên xem thời khóa biểu, điểm học phần và lịch thi, tuy nhiên vẫn mang tính chất quản lý hành chính đơn thuần, chưa tập trung vào yếu tố cộng đồng hay kết nối sinh viên. Các hệ thống này cũng thường thiếu các tính năng mở rộng như diễn đàn hỏi đáp, nhóm học tập, đi chung xe, hay thông báo sự kiện theo thời gian thực. Ngoài ra, hầu hết các ứng dụng đều phụ thuộc vào cơ sở dữ liệu nội bộ của trường, khó mở rộng hoặc tích hợp với các dịch vụ bên ngoài.

Bên cạnh đó, các mạng xã hội phổ biến như Facebook hoặc Telegram tuy đáp ứng tốt nhu cầu trao đổi thông tin nhanh và tương tác cộng đồng, nhưng lại không đảm bảo tính chính thống, bảo mật và kiểm soát nội dung. Sinh viên dễ bị phân tán bởi các nguồn thông tin không chính xác hoặc không liên quan đến học tập. Việc sử dụng mạng xã hội công cộng cho mục đích học đường cũng tiềm ẩn nhiều rủi ro về quyền riêng tư, dữ liệu cá nhân và tính minh bạch trong quản lý.

Từ thực tế trên có thể thấy, mặc dù đã có nhiều hệ thống và công cụ hỗ trợ sinh viên, nhưng hầu hết đều thiếu tính liên kết, chưa tích hợp đồng bộ giữa học tập, cộng đồng và đời sống sinh viên. Chưa có một ứng dụng nào cung cấp giải pháp toàn diện giúp sinh viên quản lý thông tin học tập, tham gia cộng đồng, tổ chức sự kiện, đồng thời sử dụng các tiện ích hàng ngày trong một nền tảng thống nhất.

Chính vì vậy, cần có một giải pháp công nghệ mới mang tính tổng thể, vừa đảm bảo các chức năng học thuật, vừa hỗ trợ kết nối xã hội, vừa đáp ứng các nhu cầu đời sống của sinh viên trong môi trường đại học hiện đại. Đây cũng chính là tiền đề cho sự ra đời của ứng dụng Phenikaa Connect, được thiết kế để khắc phục những hạn chế của các giải pháp hiện có và hướng đến việc xây dựng một hệ sinh thái số toàn diện dành riêng cho sinh viên Đại học Phenikaa.

1.3 Giải pháp đề xuất

Trước thực trạng các hệ thống hiện có còn rời rạc và thiếu tính kết nối, nhóm đề xuất phát triển ứng dụng Phenikaa Connect – một giải pháp công nghệ toàn diện nhằm xây dựng hệ sinh thái số dành riêng cho sinh viên Đại học Phenikaa. Ứng dụng được thiết kế để tích hợp đồng thời ba nhóm chức năng trọng tâm: quản lý học tập, kết nối cộng đồng sinh viên, và hỗ trợ các tiện

ích đời sống học đường trong cùng một nền tảng thống nhất, hiện đại và thân thiện với người dùng.

Về mặt công nghệ, ứng dụng được xây dựng bằng Flutter, một framework đa nền tảng cho phép phát triển đồng thời trên cả Android và iOS với hiệu năng cao và giao diện thống nhất. Flutter mang lại lợi thế lớn trong việc tái sử dụng mã nguồn, tăng tốc độ phát triển và đảm bảo trải nghiệm người dùng nhất quán. Ứng dụng sử dụng ngôn ngữ lập trình Dart cùng với kiến trúc MVVM (Model-View-ViewModel) kết hợp Provider pattern để quản lý trạng thái (state management). Cách tiếp cận này giúp mã nguồn rõ ràng, dễ bảo trì, đồng thời đảm bảo dữ liệu luôn đồng bộ giữa giao diện và logic xử lý.

Ở tầng backend, hệ thống được triển khai trên nền tảng Supabase, cung cấp các dịch vụ cơ bản như cơ sở dữ liệu PostgreSQL, xác thực người dùng (Authentication), đồng bộ dữ liệu thời gian thực (Realtime Subscriptions), lưu trữ tệp (Storage) và Edge Functions để xử lý logic nghiệp vụ phía máy chủ. Supabase cho phép ứng dụng hoạt động ổn định, bảo mật và có khả năng mở rộng, đồng thời giảm tải việc tự xây dựng backend phức tạp. Việc áp dụng Row Level Security (RLS) đảm bảo rằng mỗi sinh viên chỉ có thể truy cập dữ liệu của chính mình, đáp ứng các tiêu chuẩn bảo mật dữ liệu người dùng.

Về mặt chức năng, Phenikaa Connect được thiết kế với nhiều mô-đun đáp ứng nhu cầu học tập và đời sống của sinh viên. Ứng dụng cung cấp hồ sơ cá nhân chi tiết, cho phép sinh viên theo dõi tiến độ học tập, điểm số, lịch học và các môn học đã đăng ký. Bên cạnh đó, hệ thống có bảng tin (News Feed) nơi sinh viên có thể đăng bài, chia sẻ kinh nghiệm học tập, tham gia bình luận và tương tác với bạn bè. Các nhóm học tập (Study Groups) giúp sinh viên dễ dàng tìm bạn cùng môn, tạo nhóm học và trao đổi tài liệu. Ngoài ra, ứng dụng còn tích hợp các tính năng tiện ích như đi chung xe (Carpool), đồ thất lạc (Lost & Found) và quản lý sự kiện – câu lạc bộ (Events & Clubs), mang lại trải nghiệm toàn diện hơn cho sinh viên trong đời sống học đường.

Một trong những điểm nổi bật của giải pháp là khả năng tương tác thời gian thực giữa các thành phần. Khi người dùng thực hiện hành động như đăng bài, thích bài viết, hoặc tham gia sự kiện, hệ thống sẽ tự động cập nhật dữ liệu trên toàn bộ thiết bị đang sử dụng ứng dụng. Điều này giúp sinh viên có trải nghiệm liên tục, không cần làm mới thủ công. Bên cạnh đó, hệ thống còn hỗ trợ các tính năng bảo mật nâng cao, bao gồm đăng nhập qua tài khoản trường học, xác thực nhiều lớp (multi-factor authentication), và lưu trữ dữ liệu người dùng theo tiêu chuẩn an toàn của Supabase.

Giải pháp Phenikaa Connect không chỉ tập trung vào tính năng mà còn chú trọng đến trải nghiệm người dùng (UX/UI). Giao diện ứng dụng được thiết kế hiện đại, dễ sử dụng, phù hợp với sinh viên đại học, sử dụng màu sắc thương hiệu của Trường Đại học Phenikaa. Các thành phần được bố trí logic, giúp người dùng dễ dàng truy cập đến các chức năng quan trọng như bảng tin, học tập, sự kiện, và hồ sơ cá nhân chỉ với vài thao tác.

Tổng thể, Phenikaa Connect hướng đến việc trở thành nền tảng số thống nhất cho toàn bộ sinh viên Đại học Phenikaa, giúp đơn giản hóa việc quản lý thông tin, tăng cường khả năng tương tác và tạo môi trường học tập – kết nối tích cực. Đây là bước tiến quan trọng trong hành trình chuyên đổi số của nhà trường, đồng thời mở ra cơ hội để áp dụng các công nghệ hiện đại vào việc hỗ trợ học tập và gắn kết cộng đồng sinh viên trong kỷ nguyên số.

2 Thiết kế

2.1 Các yêu cầu chức năng

Hệ thống Phenikaa Connect được thiết kế để phục vụ hai nhóm người dùng chính: Sinh viên (User) và Quản trị viên (Admin). Các yêu cầu chức năng được định nghĩa dựa trên cấu trúc cơ sở dữ liệu và những tính năng đã được triển khai trong hệ thống.

Các yêu cầu cho Sinh viên

- R1. Quản lý tài khoản và xác thực
 - R1.1 Hệ thống cho phép sinh viên đăng nhập qua email Phenikaa.
 - R1.2 Hệ thống tự động tạo hồ sơ người dùng khi đăng nhập lần đầu với các thông tin: email, student_id, name, major, year.
 - R1.3 Hệ thống cho phép sinh viên cập nhật thông tin cá nhân (họ tên, MSSV, khoa/ngành, năm học, avatar, số điện thoại).
 - R1.4 Hệ thống cho phép sinh viên thay đổi mật khẩu, đăng xuất hoặc yêu cầu xóa tài khoản.
 - R1.5 Hệ thống đảm bảo bảo mật thông tin thông qua token JWT (JSON Web Token).
 - R1.6 Hệ thống lưu trữ metadata bổ sung của người dùng dưới dạng JSON.
- R2. Quản lý bài viết và tương tác xã hội
 - R2.1 Hệ thống cho phép sinh viên tạo bài viết mới với nội dung văn bản và hình ảnh.
 - R2.2 Hệ thống hiển thị danh sách bài viết theo thứ tự thời gian (News Feed).
 - R2.3 Hệ thống cho phép sinh viên chỉnh sửa hoặc xóa bài viết cá nhân.
 - R2.4 Hệ thống cho phép sinh viên thích (like) bài viết và hiển thị số lượng lượt thích.
 - R2.5 Hệ thống cho phép sinh viên bình luận (comment) vào bài viết.
 - R2.6 Hệ thống hỗ trợ bình luận lồng nhau (nested comments) với parent_id.
 - R2.7 Hệ thống tự động cập nhật số lượng lượt thích và bình luận của mỗi bài viết.
 - R2.8 Hệ thống cho phép sinh viên chỉnh sửa hoặc xóa bình luận của mình.
- R3. Quản lý thông báo
 - R3.1 Hệ thống hiển thị danh sách thông báo từ nhà trường hoặc quản trị viên.
 - R3.2 Hệ thống phân loại thông báo theo mức độ ưu tiên (high, normal, low).
 - R3.3 Hệ thống hiển thị thông báo theo đối tượng mục tiêu (target_audience).
 - R3.4 Hệ thống phân loại thông báo theo danh mục (category).
 - R3.5 Hệ thống cho phép sinh viên xem chi tiết nội dung thông báo.

- R4. Quản lý thời khóa biểu
 - R4.1 Hệ thống cho phép sinh viên thêm lịch học cá nhân với các thông tin: ngày trong tuần, giờ bắt đầu, giờ kết thúc, môn học, phòng học, giảng viên.
 - R4.2 Hệ thống hiển thị thời khóa biểu theo tuần với mã màu phân biệt từng môn học.
 - R4.3 Hệ thống cho phép sinh viên chỉnh sửa hoặc xóa lịch học.
 - R4.4 Hệ thống tự động sắp xếp lịch học theo thứ tự thời gian trong ngày.
- R5. Quản lý môn học và học tập
 - R5.1 Hệ thống cho phép sinh viên xem danh sách các môn học đang tham gia.
 - R5.2 Hệ thống hiển thị chi tiết từng môn học: tên môn, mã môn (code), giảng viên, tiến độ học tập (progress).
 - R5.3 Hệ thống hiển thị số lượng câu hỏi Q&A và số thành viên trong mỗi môn học.
 - R5.4 Hệ thống cho phép sinh viên theo dõi tiến độ học tập cá nhân (theo phần trăm hoàn thành).
 - R5.5 Hệ thống phân biệt môn học theo mã màu để dễ nhận diện.
- R6. Hệ thống hỏi đáp học thuật (Q&A)
 - R6.1 Hệ thống cho phép sinh viên đặt câu hỏi học thuật với tiêu đề, nội dung và môn học tương ứng.
 - R6.2 Hệ thống hiển thị danh sách câu hỏi theo môn học.
 - R6.3 Hệ thống cho phép sinh viên trả lời câu hỏi của người khác.
 - R6.4 Hệ thống hỗ trợ trả lời lồng nhau (nested replies) với parent_id.
 - R6.5 Hệ thống cho phép đánh dấu một câu trả lời là giải pháp (is_solution).
 - R6.6 Hệ thống tự động đánh dấu câu hỏi là "đã giải quyết" (solved) khi có câu trả lời được chọn là giải pháp.
 - R6.7 Hệ thống hiển thị số lượng câu trả lời cho mỗi câu hỏi.
 - R6.8 Hệ thống cho phép sinh viên chỉnh sửa hoặc xóa câu hỏi/câu trả lời của mình.
- R7. Quản lý nhóm học tập
 - R7.1 Hệ thống cho phép sinh viên tạo nhóm học tập cho từng môn học với thông tin: tên nhóm, mô tả, thời gian gấp, địa điểm, số lượng thành viên tối đa.
 - R7.2 Hệ thống hiển thị danh sách các nhóm học tập theo môn học.
 - R7.3 Hệ thống cho phép sinh viên tham gia nhóm học tập.
 - R7.4 Hệ thống cho phép sinh viên rời khỏi nhóm học tập.
 - R7.5 Hệ thống tự động cập nhật số lượng thành viên trong nhóm.
 - R7.6 Hệ thống giới hạn số lượng thành viên theo max_members đã đặt.

- R7.7 Hệ thống hiển thị thông tin người tạo nhóm (creator).
- R8. Quản lý câu lạc bộ (Clubs)
 - R8.1 Hệ thống hiển thị danh sách các câu lạc bộ đang hoạt động.
 - R8.2 Hệ thống cho phép sinh viên xem chi tiết câu lạc bộ: tên, mô tả, danh mục, hình ảnh, số lượng thành viên.
 - R8.3 Hệ thống cho phép sinh viên đăng ký tham gia câu lạc bộ.
 - R8.4 Hệ thống hiển thị trạng thái đăng ký thành viên (pending/approved/rejected).
 - R8.5 Hệ thống cho phép sinh viên rời khỏi câu lạc bộ.
 - R8.6 Hệ thống tự động cập nhật số lượng thành viên của câu lạc bộ.
 - R8.7 Hệ thống hiển thị thông tin trưởng/chủ nhiệm câu lạc bộ (leader_id).
 - R8.8 Hệ thống cho phép sinh viên xem trạng thái phê duyệt của câu lạc bộ (pending/approved/rejected).
- R9. Quản lý bài viết trong câu lạc bộ
 - R9.1 Hệ thống cho phép thành viên câu lạc bộ đăng bài viết trong câu lạc bộ.
 - R9.2 Hệ thống hỗ trợ đính kèm tệp tin (attachments) dưới dạng JSON array.
 - R9.3 Hệ thống cho phép ghim bài viết quan trọng (pinned).
 - R9.4 Hệ thống cho phép thành viên bình luận vào bài viết câu lạc bộ.
 - R9.5 Hệ thống hỗ trợ bình luận lồng nhau trong bài viết câu lạc bộ.
 - R9.6 Hệ thống cho phép thành viên chỉnh sửa hoặc xóa bài viết/bình luận của mình trong câu lạc bộ.
- R10. Quản lý hoạt động câu lạc bộ
 - R10.1 Hệ thống cho phép thành viên câu lạc bộ tạo hoạt động mới với thông tin: tiêu đề, mô tả, ngày, giờ, địa điểm.
 - R10.2 Hệ thống hiển thị trạng thái phê duyệt của hoạt động (pending/approved/rejected).
 - R10.3 Hệ thống cho phép sinh viên đăng ký tham gia hoạt động câu lạc bộ.
 - R10.4 Hệ thống hiển thị danh sách người tham gia hoạt động.
 - R10.5 Hệ thống cho phép sinh viên hủy đăng ký tham gia hoạt động.
- R11. Quản lý sự kiện
 - R11.1 Hệ thống hiển thị danh sách các sự kiện được tổ chức trong trường.
 - R11.2 Hệ thống phân loại sự kiện theo danh mục (category): học thuật, thể thao, văn hóa.
 - R11.3 Hệ thống hiển thị chi tiết sự kiện: tiêu đề, mô tả, ngày, giờ, địa điểm, hình ảnh, số lượng người tham gia tối đa.

- R11.4 Hệ thống cho phép sinh viên đăng ký tham gia sự kiện.
- R11.5 Hệ thống tự động cập nhật số lượng người tham gia.
- R11.6 Hệ thống giới hạn số lượng người tham gia theo max_attendees.
- R11.7 Hệ thống cho phép sinh viên hủy đăng ký tham gia sự kiện.
- R11.8 Hệ thống hiển thị thông tin người tổ chức sự kiện (organizer).
- R11.9 Hệ thống hiển thị câu lạc bộ tổ chức (nếu có).
- R11.10 Hệ thống hiển thị trạng thái phê duyệt của sự kiện (pending/approved/rejected).
- R11.11 Hệ thống lưu trữ metadata bổ sung của sự kiện dưới dạng JSON.
- R12. Quản lý địa điểm
 - R12.1 Hệ thống hiển thị danh sách các địa điểm trong khuôn viên trường.
 - R12.2 Hệ thống phân loại địa điểm theo loại (type): phòng học, thư viện, căn tin, khu vực sinh hoạt.
 - R12.3 Hệ thống hiển thị thông tin chi tiết: tên địa điểm, tòa nhà, tầng, mô tả.
 - R12.4 Hệ thống hỗ trợ lưu tọa độ địa lý (coordinates) dưới dạng point.
 - R12.5 Hệ thống đánh dấu các địa điểm phổ biến (popular).
 - R12.6 Hệ thống cho phép sinh viên tìm kiếm địa điểm theo tên hoặc loại.

Các yêu cầu cho Quản trị viên

- R13. Quản lý tài khoản người dùng
 - R13.1 Quản trị viên có thể thực hiện tất cả các chức năng như sinh viên.
 - R13.2 Hệ thống cho phép quản trị viên xem danh sách tất cả người dùng.
 - R13.3 Hệ thống cho phép tìm kiếm người dùng theo email, student_id, tên, ngành học.
 - R13.4 Hệ thống cho phép quản trị viên khóa/mở khóa tài khoản người dùng (is_locked).
 - R13.5 Hệ thống cho phép quản trị viên thay đổi trạng thái tài khoản (account_status).
 - R13.6 Hệ thống cho phép quản trị viên chỉnh sửa thông tin người dùng.
 - R13.7 Hệ thống cho phép quản trị viên thay đổi vai trò người dùng (user/admin).
 - R13.8 Hệ thống cho phép quản trị viên xóa tài khoản vi phạm.
- R14. Quản lý bài viết và bình luận
 - R14.1 Hệ thống cho phép quản trị viên xem tất cả bài viết trong hệ thống.

- R14.2 Hệ thống cho phép quản trị viên xóa bài viết vi phạm nội quy.
- R14.3 Hệ thống cho phép quản trị viên xóa bình luận không phù hợp.
- R14.4 Hệ thống hiển thị thống kê số lượng bài viết, lượt thích và bình luận.
- R14.5 Hệ thống cho phép quản trị viên tìm kiếm bài viết theo người đăng hoặc nội dung.
- R15. Quản lý thông báo
 - R15.1 Hệ thống cho phép quản trị viên tạo thông báo mới.
 - R15.2 Hệ thống cho phép thiết lập mức độ ưu tiên cho thông báo (high, normal, low).
 - R15.3 Hệ thống cho phép thiết lập đối tượng nhận thông báo (target_audience).
 - R15.4 Hệ thống cho phép phân loại thông báo theo danh mục.
 - R15.5 Hệ thống cho phép quản trị viên chỉnh sửa hoặc xóa thông báo.
 - R15.6 Hệ thống ghi lại thông tin người tạo thông báo (created_by).
- R16. Quản lý câu lạc bộ
 - R16.1 Hệ thống cho phép quản trị viên tạo câu lạc bộ mới.
 - R16.2 Hệ thống cho phép quản trị viên phê duyệt hoặc từ chối câu lạc bộ mới (status: pending/approved/rejected).
 - R16.3 Hệ thống cho phép quản trị viên chỉnh sửa thông tin câu lạc bộ.
 - R16.4 Hệ thống cho phép quản trị viên vô hiệu hóa câu lạc bộ (active: false).
 - R16.5 Hệ thống cho phép quản trị viên xóa câu lạc bộ.
 - R16.6 Hệ thống cho phép quản trị viên quản lý danh sách thành viên câu lạc bộ.
 - R16.7 Hệ thống cho phép quản trị viên phê duyệt hoặc từ chối đơn xin tham gia câu lạc bộ.
- R17. Quản lý bài viết và hoạt động câu lạc bộ
 - R17.1 Hệ thống cho phép quản trị viên xem tất cả bài viết trong câu lạc bộ.
 - R17.2 Hệ thống cho phép quản trị viên xóa bài viết hoặc bình luận vi phạm trong câu lạc bộ.

- R17.3 Hệ thống cho phép quản trị viên phê duyệt hoặc từ chối hoạt động câu lạc bộ.
- R17.4 Hệ thống cho phép quản trị viên chỉnh sửa hoặc xóa hoạt động câu lạc bộ.
- R18. Quản lý sự kiện
 - R18.1 Hệ thống cho phép quản trị viên tạo sự kiện mới.
 - R18.2 Hệ thống cho phép quản trị viên phê duyệt hoặc từ chối sự kiện (status: pending/approved/rejected).
 - R18.3 Hệ thống cho phép quản trị viên chỉnh sửa thông tin sự kiện.
 - R18.4 Hệ thống cho phép quản trị viên xóa sự kiện.
 - R18.5 Hệ thống cho phép quản trị viên quản lý danh sách người tham gia sự kiện.
 - R18.6 Hệ thống hiển thị thống kê số lượng sự kiện theo danh mục và thời gian.
- R19. Quản lý môn học
 - R19.1 Hệ thống cho phép quản trị viên tạo môn học mới với thông tin: tên môn, mã môn, giảng viên.
 - R19.2 Hệ thống cho phép quản trị viên chỉnh sửa thông tin môn học.
 - R19.3 Hệ thống cho phép quản trị viên xóa môn học.
 - R19.4 Hệ thống hiển thị thống kê số lượng câu hỏi và thành viên của từng môn học.
- R20. Quản lý câu hỏi học thuật (Q&A)
 - R20.1 Hệ thống cho phép quản trị viên xem tất cả câu hỏi trong hệ thống.
 - R20.2 Hệ thống cho phép quản trị viên xóa câu hỏi hoặc câu trả lời vi phạm.
 - R20.3 Hệ thống cho phép quản trị viên đánh dấu câu trả lời là giải pháp.
 - R20.4 Hệ thống hiển thị thống kê số lượng câu hỏi theo môn học và trạng thái (đã giải quyết/chưa giải quyết).
- R21. Quản lý nhóm học tập
 - R21.1 Hệ thống cho phép quản trị viên xem tất cả nhóm học tập.
 - R21.2 Hệ thống cho phép quản trị viên xóa nhóm học tập vi phạm.

- R21.3 Hệ thống hiển thị thống kê số lượng nhóm học tập theo môn học.
- R22. Quản lý địa điểm
 - R22.1 Hệ thống cho phép quản trị viên thêm địa điểm mới.
 - R22.2 Hệ thống cho phép quản trị viên chỉnh sửa thông tin địa điểm.
 - R22.3 Hệ thống cho phép quản trị viên xóa địa điểm.
 - R22.4 Hệ thống cho phép quản trị viên đánh dấu địa điểm phổ biến.

Các yêu cầu chung cho hệ thống

- R23. Phân quyền và bảo mật
 - R23.1 Hệ thống phân quyền rõ ràng giữa Sinh viên (User) và Quản trị viên (Admin).
 - R23.2 Hệ thống yêu cầu xác thực JWT trước khi truy cập dữ liệu cá nhân.
 - R23.3 Hệ thống mã hóa dữ liệu truyền qua mạng bằng HTTPS.
 - R23.4 Hệ thống tự động hết hạn phiên làm việc sau một khoảng thời gian không hoạt động.
 - R23.5 Hệ thống tuân thủ chính sách RLS (Row Level Security) của Supabase.
 - R23.6 Hệ thống kiểm tra quyền truy cập trước khi cho phép thao tác CRUD trên dữ liệu.
- R24. Đồng bộ và cập nhật dữ liệu
 - R24.1 Hệ thống tự động cập nhật số lượng thành viên khi có người tham gia/rời khỏi câu lạc bộ.
 - R24.2 Hệ thống tự động cập nhật số lượng người tham gia sự kiện.
 - R24.3 Hệ thống tự động cập nhật số lượng lượt thích và bình luận của bài viết.
 - R24.4 Hệ thống tự động cập nhật số lượng câu trả lời cho mỗi câu hỏi.
 - R24.5 Hệ thống tự động cập nhật số lượng thành viên trong nhóm học tập.
 - R24.6 Hệ thống ghi lại thời gian tạo và cập nhật cho tất cả các bản ghi (created_at, updated_at).
- R25. Giao diện và trải nghiệm người dùng
 - R25.1 Giao diện được thiết kế thân thiện, tuân theo phong cách thương hiệu Phenikaa.
 - R25.2 Hệ thống hỗ trợ đa ngôn ngữ (Tiếng Việt, Tiếng Anh).
 - R25.3 Giao diện tương thích tốt với điện thoại và máy tính bảng (responsive design).
 - R25.4 Hệ thống cung cấp hướng dẫn sử dụng cơ bản cho người mới.
 - R25.5 Hệ thống hiển thị thông báo lỗi rõ ràng khi có sự cố.
 - R25.6 Hệ thống cung cấp feedback trực quan khi người dùng thực hiện thao tác (loading, success, error states).

- R26. Hiệu năng và khả năng mở rộng
 - R26.1 Hệ thống sử dụng UUID cho khóa chính để đảm bảo tính duy nhất và bảo mật.
 - R26.2 Hệ thống sử dụng JSONB để lưu trữ metadata linh hoạt.
 - R26.3 Hệ thống tối ưu hóa truy vấn database với các index phù hợp.
 - R26.4 Hệ thống hỗ trợ phân trang (pagination) cho các danh sách dài.
 - R26.5 Hệ thống cache dữ liệu thường xuyên được truy cập để cải thiện hiệu năng.
- R27. Tuân thủ và quản lý dữ liệu
 - R27.1 Hệ thống đảm bảo tính toàn vẹn dữ liệu với ràng buộc khóa ngoại (foreign keys).
 - R27.2 Hệ thống đảm bảo tính duy nhất cho email và student_id.
 - R27.3 Hệ thống sử dụng các enum types cho các trường có giá trị cố định (user_role, approval_status, membership_status).
 - R27.4 Hệ thống hỗ trợ xóa cascade để đảm bảo tính nhất quán khi xóa dữ liệu liên quan.
 - R27.5 Hệ thống lưu trữ lịch sử thay đổi quan trọng thông qua các trường timestamp.

2.2 Các yêu cầu phi chức năng

Yêu cầu về hiệu suất (Performance Requirements)

- Thời gian phản hồi:
 - Ứng dụng di động Phenikaa Connect phải tải trang chủ hoặc dashboard trong vòng 2 giây trên thiết bị có kết nối trung bình.
 - Các thao tác cơ bản như đăng bài (posts), thích (post_likes), bình luận (comments), hoặc cập nhật hồ sơ người dùng (users) phải hoàn thành trong vòng 3 giây.
 - Tính năng tải danh sách bài viết (posts), sự kiện (events), câu hỏi (questions), hoặc thông báo (announcements) phải phản hồi trong vòng 2 giây.
 - Các thao tác tìm kiếm hoặc truy vấn dữ liệu (posts, users, events, clubs, questions) từ Supabase phải trả về kết quả trong vòng 1,5 giây.
 - Tải thời khóa biểu cá nhân (class_schedules) và danh sách khóa học (courses) phải hoàn thành trong vòng 1 giây.
- Throughput:
 - Hệ thống phải xử lý được ít nhất 100 người dùng đồng thời trên cùng một phiên bản backend.
 - Supabase phải hỗ trợ tối thiểu 200 request/giây mà không làm gián đoạn dịch vụ.
 - Tác vụ upload ảnh hoặc file (avatar_url, image_url trong posts/events/clubs) dưới 5MB phải hoàn thành trong vòng 5 giây.

- Các thao tác đọc/ghi dữ liệu lớn (danh sách bài viết, sự kiện, câu hỏi, thành viên câu lạc bộ) phải hoàn tất trong vòng 3 giây.
- Các thao tác liên quan đến club_posts, club_post_comments và club_activities phải xử lý trong vòng 2 giây.
- **Khả năng mở rộng (Scalability):**
 - Hệ thống phải có khả năng mở rộng để phục vụ tối thiểu 5000 người dùng hoạt động cùng lúc.
 - Cơ sở dữ liệu Supabase (PostgreSQL) phải lưu trữ được dữ liệu của ít nhất 50.000 người dùng (users) và 1 triệu bản ghi (posts, comments, questions, events, club_posts).
 - Hệ thống phải hỗ trợ lưu trữ tối thiểu 10.000 câu lạc bộ (clubs) với 100.000 thành viên (club_members).
 - Hệ thống phải hỗ trợ lưu trữ tối thiểu 50.000 câu hỏi (questions) với 200.000 câu trả lời (question_replies).
 - Hệ thống phải cho phép mở rộng thêm mô-đun mới (E-learning, Attendance) mà không làm ảnh hưởng đến hiệu năng hiện tại.

Yêu cầu về tính khả dụng (Availability Requirements)

- **Uptime:**
 - Dịch vụ Supabase và hệ thống backend phải đạt tỷ lệ uptime tối thiểu 99% trong giờ hành chính (8h – 17h).
 - Trong thời gian ngoài giờ hành chính, hệ thống phải duy trì tỷ lệ uptime tối thiểu 97%.
 - Thời gian bảo trì định kỳ không được vượt quá 4 giờ mỗi tháng và phải thông báo trước cho người dùng thông qua bảng announcements.
- **Khôi phục:**
 - Hệ thống phải có khả năng khôi phục hoạt động trong vòng 1 giờ sau khi gặp sự cố nghiêm trọng.
 - Toàn bộ dữ liệu (users, posts, comments, questions, events, clubs, study_groups, announcements) phải được sao lưu tự động hàng ngày.
 - Dữ liệu backup phải có khả năng khôi phục trong vòng 30 phút.
 - Supabase Storage phải có cơ chế tự động lưu trữ dự phòng (redundancy) cho các file media (avatar_url, image_url, attachments trong club_posts).

- Dữ liệu metadata và attachments dạng JSONB phải được backup đầy đủ.

Yêu cầu về bảo mật (Security Requirements)

- Xác thực và phân quyền:
 - Ứng dụng phải sử dụng Supabase Authentication kết hợp Google OAuth 2.0 cho đăng nhập an toàn.
 - Hệ thống phải phân quyền rõ ràng dựa trên cột role trong bảng users (user_role: 'user', 'admin').
 - Phiên đăng nhập tự động hết hạn sau 24 giờ không hoạt động hoặc khi người dùng đăng xuất.
 - Hệ thống phải kiểm soát trạng thái tài khoản thông qua account_status và is_locked trong bảng users.
 - Hệ thống phải ghi log đầy đủ các thao tác quan trọng: tạo/sửa/xóa posts, club_posts, events, questions, thay đổi membership_status, approval_status.
- Bảo vệ dữ liệu:
 - Tất cả dữ liệu nhạy cảm (email, phone, student_id trong bảng users) phải được mã hóa AES-256 khi lưu trữ.
 - Kết nối giữa client và server phải sử dụng HTTPS/TLS 1.3 để bảo vệ dữ liệu truyền tải.
 - Hệ thống phải có biện pháp chống SQL Injection, Cross-Site Scripting (XSS) và Cross-Site Request Forgery (CSRF).
 - Dữ liệu cá nhân của sinh viên (email, phone, student_id, major, year) phải được bảo vệ theo tiêu chuẩn tương tự GDPR.
 - Dữ liệu metadata dạng JSONB trong các bảng (users, clubs, events) phải được validate và sanitize trước khi lưu trữ.
- Kiểm soát truy cập:
 - Chỉ admin (role = 'admin') mới có quyền duyệt/tù chối các yêu cầu có approval_status (clubs, events, club_activities).
 - Chỉ leader_id của clubs hoặc creator_id của club_activities có quyền chỉnh sửa thông tin câu lạc bộ/hoạt động.
 - Sinh viên chỉ có thể xem, chỉnh sửa hoặc xóa dữ liệu cá nhân của chính mình (posts, comments, questions, question_replies).
 - Hệ thống phải kiểm soát membership_status ('pending', 'approved', 'rejected') khi người dùng tham gia club_members.

- Hệ thống phải giới hạn số lần đăng nhập sai liên tiếp để ngăn chặn brute-force attack.
- Tất cả các file upload (avatar_url, image_url, attachments) phải được kiểm tra định dạng, kích thước và nội dung trước khi lưu trữ.

Yêu cầu về tính dễ sử dụng (Usability Requirements)

- Giao diện người dùng:
 - Giao diện ứng dụng phải trực quan, thân thiện và nhất quán theo phong cách thương hiệu Phenikaa.
 - Toàn bộ các trang (Home, Study, Social, Events, Profile, Clubs, Q&A) phải có bố cục rõ ràng, dễ thao tác.
 - Người dùng mới có thể làm quen với các chức năng cơ bản (xem posts, tham gia events, đặt câu hỏi, tham gia clubs) trong vòng 10 phút.
 - Ứng dụng phải có hướng dẫn sử dụng (Onboarding) cho người dùng lần đầu đăng nhập.
 - Giao diện phải responsive, tương thích trên nhiều kích thước màn hình điện thoại và máy tính bảng.
 - Màu sắc trong class_schedules và courses phải được hiển thị nhất quán và dễ phân biệt.
- Khả năng tiếp cận:
 - Ứng dụng phải hỗ trợ hiển thị tốt trên các trình duyệt WebView tích hợp (Chrome, Safari).
 - Các nút và biểu tượng phải có kích thước tối thiểu 40px để dễ thao tác.
 - Tất cả các form nhập liệu (tạo posts, questions, events, clubs) phải có validation và thông báo lỗi rõ ràng khi nhập sai.
 - Hệ thống phải hiển thị thông báo xác nhận cho các thao tác quan trọng (xóa posts/comments, rời club_members, hủy tham gia event_attendees).
 - Trạng thái approval_status ('pending', 'approved', 'rejected') phải được hiển thị rõ ràng cho người dùng.

Yêu cầu về tính tin cậy (Reliability Requirements)

- Tính chính xác:
 - Hệ thống phải đảm bảo tính nhất quán dữ liệu giữa frontend và backend trong mọi thao tác.
 - Các phép tính thống kê như likes_count, comments_count trong posts; members_count trong clubs; attendees_count trong events; replies_count trong questions phải có độ chính xác 100%.

- Hệ thống phải tự động cập nhật các counter khi có thao tác thêm/xóa (post_likes, comments, club_members, event_attendees, question_replies).
- Hệ thống phải đảm bảo dữ liệu không bị mất hoặc trùng lặp khi xảy ra sự cố kết nối.
- Các thao tác thêm/sửa/xóa phải được xác nhận từ backend trước khi hiển thị thay đổi trên giao diện.
- Trạng thái solved trong questions phải được cập nhật chính xác khi có is_solution = true trong question_replies.
- Xử lý lỗi:
 - Hệ thống phải xử lý an toàn tất cả các lỗi không mong muốn (graceful error handling).
 - Mọi lỗi xảy ra phải được hiển thị dưới dạng thông báo thân thiện cho người dùng.
 - Hệ thống có cơ chế rollback tự động nếu xảy ra lỗi trong quá trình ghi dữ liệu (posts, events, clubs, questions).
 - Toàn bộ log lỗi được lưu lại trong Supabase Logs hoặc file hệ thống để phục vụ debug.
 - Hệ thống phải xử lý các ràng buộc khóa ngoại (foreign key) một cách an toàn khi xóa dữ liệu liên quan.

Yêu cầu về khả năng tương thích (Compatibility Requirements)

- Tương thích nền tảng:
 - Ứng dụng phải chạy ổn định trên Android (API ≥ 21) và iOS (≥ 13.0).
 - Flutter SDK tối thiểu: 3.9.2; Dart SDK: $>= 2.19.0$.
 - Backend phải tương thích với Supabase SDK và hỗ trợ PostgreSQL ≥ 14.0 (hỗ trợ đầy đủ JSONB, UUID, Point).
 - Ứng dụng phải tương thích với trình giả lập và thiết bị thật trên môi trường phát triển Flutter.
 - Database phải hỗ trợ các kiểu dữ liệu đặc biệt: uuid, jsonb, point (cho coordinates trong locations), time, date.
- Tương thích dữ liệu:
 - API giao tiếp giữa client và server sử dụng định dạng JSON chuẩn (UTF-8).
 - Hệ thống hỗ trợ lưu trữ và hiển thị hình ảnh ở định dạng .png, .jpg, .jpeg cho avatar_url, image_url.

- Dữ liệu JSONB (metadata, attachments) phải tuân theo chuẩn JSON và có schema validation.
- File dữ liệu backup sử dụng định dạng .sql hoặc .csv để dễ dàng di chuyển giữa môi trường.
- Dữ liệu tiếng Việt phải được xử lý và hiển thị chính xác với encoding UTF-8 (tên người dùng, nội dung posts, questions, announcements).
- Hệ thống phải hỗ trợ các enum types: user_role, approval_status, membership_status.
- Dữ liệu coordinates trong locations phải tương thích với định dạng Point của PostgreSQL.

2.3 Các ràng buộc (Constraints)

2.3.1 Các ràng buộc về triển khai

- Ràng buộc về công nghệ:

Hệ thống Phenikaa Connect phải được phát triển bằng Flutter sử dụng ngôn ngữ lập trình Dart, nhằm hỗ trợ triển khai đa nền tảng (Android, iOS, và Web). Backend được xây dựng dựa trên nền tảng Supabase – bao gồm PostgreSQL, Authentication, Storage, và Edge Functions. Ứng dụng phải tương thích với Flutter SDK phiên bản ≥ 3.10 và Dart SDK ≥ 2.19 . Tất cả thư viện (dependencies) được sử dụng phải là mã nguồn mở (open-source) với license cho phép sử dụng trong dự án học thuật và thương mại.

- Ràng buộc về môi trường:

Ứng dụng phải hoạt động ổn định trên các thiết bị di động có cấu hình tối thiểu 2GB RAM, hệ điều hành Android 8.0 trở lên hoặc iOS 13.0 trở lên. Phiên bản web của hệ thống phải tương thích với các trình duyệt phổ biến như Chrome, Safari, Firefox, và Edge. Giao diện người dùng phải được thiết kế responsive, bảo đảm hiển thị đúng kích thước và bố cục trên các thiết bị khác nhau.

- Ràng buộc về thời gian:

Hệ thống phải được phát triển và bàn giao trong thời gian quy định của đồ án tốt nghiệp. Các mô-đun cốt lõi, bao gồm đăng nhập, bảng tin, nhóm học tập, sự kiện, và hồ sơ sinh viên, phải được hoàn thiện trước giai đoạn thử nghiệm. Quá trình triển khai ứng dụng không được vượt quá 1 ngày cài đặt, và phải kèm theo tài liệu hướng dẫn chi tiết để giảng viên hoặc người dùng có thể dễ dàng chạy thử trên thiết bị.

- Ràng buộc về bảo mật:

Hệ thống phải sử dụng OAuth nội bộ của trường làm phương thức xác thực chính. Toàn bộ kết nối giữa client và server phải thông qua HTTPS. Dữ liệu người dùng được lưu trữ trên Supabase phải được mã hóa và bảo vệ bởi chính sách RLS (Row Level Security). Hệ thống cần có cơ chế phân quyền rõ ràng giữa các loại người dùng (Sinh viên, Quản trị viên) và ghi log tất cả các thao tác quan trọng trong cơ sở dữ liệu.

2.3.2 Các ràng buộc kinh tế

- Ràng buộc về ngân sách:

Ứng dụng Phenikaa Connect được phát triển trong khuôn khổ án nên ngân sách hạn chế. Toàn bộ công nghệ được sử dụng phải là miễn phí hoặc open-source, bao gồm Flutter SDK, Supabase, và các thư viện UI. Không được sử dụng các dịch vụ đám mây trả phí trong quá trình phát triển và thử nghiệm. Các công cụ lập trình, mô phỏng và kiểm thử như VS Code, Android Studio và Postman đều phải ở phiên bản miễn phí.

- Ràng buộc về chi phí vận hành:

Hệ thống cần được thiết kế để tối ưu chi phí duy trì trong dài hạn. Việc lưu trữ dữ liệu và file phải được tận dụng từ gói miễn phí của Supabase trong giai đoạn phát triển và demo. Ứng dụng phải có khả năng triển khai trên hosting giá rẻ hoặc server cục bộ (localhost) mà không yêu cầu cấu hình phần cứng cao. Quá trình bảo trì hệ thống phải dễ thực hiện mà không cần đội ngũ kỹ thuật chuyên sâu.

- Ràng buộc về hiệu quả chi phí:

Kiến trúc của hệ thống phải đảm bảo hiệu quả tối đa với chi phí thấp nhất, đồng thời có khả năng mở rộng hoặc tái sử dụng cho các cơ sở giáo dục khác mà không cần thiết kế lại toàn bộ. Việc nâng cấp tính năng (ví dụ thêm module học phần, lịch thi, thư viện điện tử) không được ảnh hưởng đến kiến trúc hiện tại.

- Tái sử dụng cho các trường đại học khác mà không cần đầu tư phát triển lại từ đầu.

2.3.3 Các ràng buộc về đạo đức

- Bảo vệ quyền riêng tư:

Hệ thống phải bảo vệ tuyệt đối thông tin cá nhân của sinh viên như họ tên, email, ảnh đại diện và nội dung trao đổi. Dữ liệu cá nhân chỉ được thu thập khi có sự đồng ý từ người dùng và được sử dụng đúng mục đích đã nêu trong chính sách bảo mật (Privacy Policy). Người dùng phải có quyền yêu cầu xem, chỉnh sửa hoặc xóa thông tin cá nhân của mình.

- Công bằng và minh bạch:

Hệ thống phải đảm bảo mọi sinh viên đều có quyền truy cập và sử dụng tính năng như nhau, không thiên vị hoặc phân biệt đối xử. Các thuật toán đề xuất (ví dụ: gợi ý nhóm học tập, sự kiện phù hợp) phải hoạt động minh bạch và có thể giải thích được. Tất cả các hành vi xử lý dữ liệu người dùng đều phải công khai và có sự đồng thuận.

- Trách nhiệm xã hội:

Ứng dụng phải hướng đến việc tăng cường kết nối, hỗ trợ học tập, chia sẻ kiến thức và phát triển cá nhân cho sinh viên. Các tính năng thông báo, bình luận và phản hồi phải được kiểm duyệt hoặc giới hạn nội dung không phù hợp, tránh gây ảnh hưởng tiêu cực đến cộng đồng sinh viên. Giao diện và tính năng phải tuân thủ tiêu chuẩn web accessibility, đảm bảo mọi người đều có thể sử dụng.

- Đạo đức nghề nghiệp:

Trong quá trình phát triển, nhóm phải tuân thủ nguyên tắc đạo đức nghề nghiệp trong lập trình phần mềm, bao gồm tính trung thực, minh bạch và trách nhiệm. Mã nguồn phải rõ ràng, có chủ thích (comment) và tuân theo quy ước coding chuẩn của Flutter/Dart. Hệ thống phải được kiểm thử kỹ lưỡng để tránh lỗi ảnh hưởng đến tính đúng đắn hoặc bảo mật của dữ liệu.

- Tính bền vững:

Hệ thống phải được thiết kế với tầm nhìn dài hạn, có khả năng mở rộng, bảo trì và nâng cấp trong tương lai. Kiến trúc phải module hóa rõ ràng, tài liệu kỹ thuật cần đầy đủ để lập trình viên khác có thể tiếp tục phát triển mà không gặp khó khăn. Mọi tài nguyên (hình ảnh, dữ liệu, tài liệu hướng dẫn) phải được quản lý thống nhất để đảm bảo tính liên tục và kế thừa của dự án.

2.4 Mô hình hệ thống / Thiết kế giải pháp

2.4.1 Các kịch bản của hệ thống (Use-cases)

Actors (Tác nhân)

- Sinh viên (Student): Người dùng đã đăng nhập (users với role = 'user', account_status = 'active', is_locked = false), có thể sử dụng đầy đủ các tính năng như đăng bài (posts), bình luận (comments), tham gia câu lạc bộ (club_members), tạo câu hỏi học tập (questions), tham gia nhóm học tập (study_group_members), đăng ký sự kiện (event_attendees), quản lý thời khóa biểu (class_schedules), và quản lý hồ sơ cá nhân.
- Quản trị viên (Administrator): Người dùng có quyền quản lý toàn hệ thống (users với role = 'admin'), có thể thực hiện tất cả chức năng của sinh viên, đồng thời quản lý người dùng, bài viết, sự kiện, câu lạc bộ, thông báo toàn trường (announcements) và duyệt các yêu cầu có approval_status và membership_status.

Nhóm Use Cases cho Sinh viên

- UC01: Đăng nhập / Đăng ký tài khoản
 - Mục đích: Cho phép người dùng truy cập vào hệ thống để sử dụng đầy đủ chức năng.
 - Mô tả: Người dùng chọn đăng nhập bằng tài khoản Phenikaa hoặc Google OAuth 2.0; hệ thống xác thực và tạo bản ghi mới trong bảng users nếu chưa tồn tại (bao gồm email, student_id, name, major, year). Mặc định role = 'user', account_status = 'active', is_locked = false.
 - Dữ liệu liên quan: users (INSERT/SELECT).

- UC02: Quản lý hồ sơ cá nhân
 - Mục đích: Quản lý thông tin cá nhân trong hồ sơ sinh viên.
 - Mô tả: Sinh viên có thể xem và cập nhật thông tin trong bảng users bao gồm: avatar_url, phone, major, year, metadata (thông tin bổ sung dạng JSON). Hệ thống tự động cập nhật updated_at khi có thay đổi.
 - Dữ liệu liên quan: users (SELECT, UPDATE).
- UC03: Quản lý thời khóa biểu
 - Mục đích: Giúp sinh viên quản lý lịch học cá nhân.
 - Mô tả: Sinh viên có thể thêm, xem, sửa, xóa lịch học trong bảng class_schedules (bao gồm day_of_week, start_time, end_time, subject, room, instructor, color). Hệ thống hiển thị thời khóa biểu theo tuần với màu sắc phân biệt cho từng môn học.
 - Dữ liệu liên quan: class_schedules (INSERT, SELECT, UPDATE, DELETE), users (SELECT).
- UC04: Đăng bài viết và tương tác cộng đồng
 - Mục đích: Tạo và chia sẻ nội dung trong cộng đồng Phenikaa.
 - Mô tả: Sinh viên đăng bài viết mới trong bảng posts (content, image_url), thích bài viết (tạo bản ghi trong post_likes và tăng likes_count), bình luận (tạo bản ghi trong comments, hỗ trợ parent_id để trả lời bình luận, và tăng comments_count). Sinh viên có thể xem, sửa, xóa bài viết của chính mình.
 - Dữ liệu liên quan: posts (INSERT, SELECT, UPDATE, DELETE), post_likes (INSERT, DELETE), comments (INSERT, SELECT, UPDATE, DELETE), users (SELECT).
- UC05: Quản lý khóa học cá nhân
 - Mục đích: Theo dõi tiến độ học tập và quản lý các khóa học đang theo học.
 - Mô tả: Sinh viên có thể thêm, xem, cập nhật thông tin khóa học trong bảng courses (name, code, instructor, progress, color). Hệ thống tự động đếm số câu hỏi (questions) và số thành viên (members) liên quan đến khóa học.
 - Dữ liệu liên quan: courses (INSERT, SELECT, UPDATE, DELETE), users (SELECT).
- UC06: Tạo và trả lời câu hỏi học tập (Q&A)
 - Mục đích: Hỗ trợ sinh viên giải đáp thắc mắc về môn học.
 - Mô tả: Sinh viên tạo câu hỏi mới trong bảng questions (course, title, content), trả lời câu hỏi trong bảng question_replies (content, parent_id để trả lời câu trả lời khác). Người trả lời có thể đánh dấu is_solution = true, hệ thống tự động cập nhật solved = true trong questions và tăng replies_count.

- Dữ liệu liên quan: questions (INSERT, SELECT, UPDATE, DELETE), question_replies (INSERT, SELECT, UPDATE, DELETE), users (SELECT), courses (SELECT).

- UC07: Tham gia và quản lý nhóm học tập (Study Groups)
 - Mục đích: Kết nối sinh viên có cùng môn học hoặc sở thích.
 - Mô tả: Sinh viên tìm kiếm nhóm học tập trong bảng study_groups (course, name, description, meet_time, location, max_members), gửi yêu cầu tham gia (tạo bản ghi trong study_group_members), hoặc tạo nhóm mới (tự động trở thành creator_id). Hệ thống tự động tăng members_count khi có thành viên mới và kiểm tra max_members.
 - Dữ liệu liên quan: study_groups (INSERT, SELECT, UPDATE, DELETE), study_group_members (INSERT, SELECT, DELETE), users (SELECT).

- UC08: Quản lý câu lạc bộ (Clubs)
 - Mục đích: Tham gia và quản lý các câu lạc bộ sinh viên.
 - Mô tả: Sinh viên xem danh sách CLB trong bảng clubs (với active = true, status = 'approved'), gửi yêu cầu tham gia (tạo bản ghi trong club_members với status = 'pending'), hoặc tạo CLB mới (tự động trở thành leader_id, status = 'pending' chờ admin duyệt). Sinh viên có thể đăng bài trong CLB (club_posts), bình luận (club_post_comments hỗ trợ parent_id), và tạo hoạt động CLB (club_activities với status = 'pending'). Hệ thống tự động tăng members_count khi có thành viên mới.
 - Dữ liệu liên quan: clubs (INSERT, SELECT, UPDATE), club_members (INSERT, SELECT, DELETE), club_posts (INSERT, SELECT, UPDATE, DELETE), club_post_comments (INSERT, SELECT, UPDATE, DELETE), club_activities (INSERT, SELECT, UPDATE), club_activity_participants (INSERT, SELECT, DELETE), users (SELECT).

- UC09: Quản lý sự kiện (Events)
 - Mục đích: Quản lý, theo dõi và tham gia các sự kiện của trường.
 - Mô tả: Sinh viên xem danh sách sự kiện trong bảng events (với status = 'approved'), đăng ký tham gia (tạo bản ghi trong event_attendees và tăng attendees_count), hủy đăng ký (xóa bản ghi và giảm attendees_count), hoặc tạo sự kiện mới (tự động trở thành organizer_id, status = 'pending' chờ admin duyệt). Sự kiện có thể liên kết với club_id nếu do CLB tổ chức. Hệ thống kiểm tra max_attendees trước khi cho phép đăng ký.
 - Dữ liệu liên quan: events (INSERT, SELECT, UPDATE, DELETE), event_attendees (INSERT, SELECT, DELETE), clubs (SELECT), users (SELECT), locations (SELECT).

- UC10: Xem và quản lý thông báo (Announcements)
 - Mục đích: Theo dõi các thông báo từ nhà trường.
 - Mô tả: Sinh viên nhận và xem thông báo từ bảng announcements (title, content, priority, category, target_audience). Thông báo có thể được lọc theo priority ('normal', 'high', 'urgent') và target_audience (theo major, year hoặc toàn trường).
 - Dữ liệu liên quan: announcements (SELECT), users (SELECT).
- UC11: Tìm kiếm địa điểm trong trường
 - Mục đích: Giúp sinh viên tìm kiếm và định vị các địa điểm trong campus.
 - Mô tả: Sinh viên tìm kiếm địa điểm trong bảng locations (name, type, building, floor, description). Hệ thống hiển thị tọa độ (coordinates dạng point) và đánh dấu các địa điểm phổ biến (popular = true).
 - Dữ liệu liên quan: locations (SELECT).

Nhóm Use Cases cho Quản trị viên

- UC12: Quản lý người dùng
 - Mục đích: Quản lý tài khoản và quyền truy cập của người dùng.
 - Mô tả: Admin có thể xem danh sách người dùng trong bảng users, chỉnh sửa thông tin (name, email, major, year, phone, metadata), phân quyền (thay đổi role giữa 'user' và 'admin'), thay đổi trạng thái tài khoản (account_status), khóa tài khoản (is_locked = true) hoặc xóa tài khoản (DELETE với cascade các bản ghi liên quan).
 - Dữ liệu liên quan: users (SELECT, UPDATE, DELETE), posts (SELECT), comments (SELECT), club_members (SELECT), event_attendees (SELECT).
- UC13: Quản lý bài viết và bình luận
 - Mục đích: Duy trì môi trường cộng đồng lành mạnh.
 - Mô tả: Admin có thể xem, chỉnh sửa, hoặc xóa bài viết trong bảng posts và bình luận trong bảng comments vi phạm quy định. Khi xóa bài viết, hệ thống tự động xóa các bản ghi liên quan trong post_likes và comments (cascade delete).
 - Dữ liệu liên quan: posts (SELECT, UPDATE, DELETE), comments (SELECT, UPDATE, DELETE), post_likes (DELETE), users (SELECT).
- UC14: Quản lý câu lạc bộ và hoạt động
 - Mục đích: Kiểm duyệt và quản lý các câu lạc bộ do sinh viên tạo.
 - Mô tả: Admin duyệt yêu cầu tạo CLB (cập nhật status trong clubs từ 'pending' thành 'approved' hoặc 'rejected'), duyệt yêu cầu tham gia CLB (cập nhật status trong club_members), duyệt hoạt động CLB (cập nhật status trong club_activities), chỉnh sửa

thông tin CLB, hoặc vô hiệu hóa CLB (active = false). Admin có thể xem tất cả club_posts và club_post_comments để kiểm duyệt nội dung.

- Dữ liệu liên quan: clubs (SELECT, UPDATE, DELETE), club_members (SELECT, UPDATE, DELETE), club_activities (SELECT, UPDATE, DELETE), club_posts (SELECT, UPDATE, DELETE), club_post_comments (SELECT, UPDATE, DELETE), users (SELECT).
- UC15: Quản lý sự kiện
 - Mục đích: Kiểm duyệt và quản lý các sự kiện do sinh viên tạo.
 - Mô tả: Admin duyệt yêu cầu tạo sự kiện (cập nhật status trong events từ 'pending' thành 'approved' hoặc 'rejected'), chỉnh sửa thông tin sự kiện (title, description, event_date, event_time, location, category, max_attendees, metadata), xem danh sách người tham gia (event_attendees), hoặc xóa sự kiện (DELETE với cascade event_attendees).
 - Dữ liệu liên quan: events (SELECT, UPDATE, DELETE), event_attendees (SELECT, DELETE), clubs (SELECT), users (SELECT).
- UC16: Quản lý thông báo hệ thống
 - Mục đích: Gửi thông báo và cập nhật thông tin đến toàn bộ sinh viên.
 - Mô tả: Admin tạo thông báo mới trong bảng announcements (title, content, priority, category, target_audience), chỉnh sửa hoặc xóa thông báo. Trường created_by lưu user_id của admin tạo thông báo. Hệ thống tự động cập nhật updated_at khi có thay đổi.
 - Dữ liệu liên quan: announcements (INSERT, SELECT, UPDATE, DELETE), users (SELECT).
- UC17: Quản lý câu hỏi và nhóm học tập
 - Mục đích: Kiểm duyệt nội dung học tập và quản lý nhóm học tập.
 - Mô tả: Admin có thể xem, chỉnh sửa hoặc xóa câu hỏi trong bảng questions và câu trả lời trong question_replies vi phạm quy định. Admin cũng có thể quản lý nhóm học tập trong study_groups, xem danh sách thành viên (study_group_members), và xóa nhóm nếu cần thiết.
 - Dữ liệu liên quan: questions (SELECT, UPDATE, DELETE), question_replies (SELECT, UPDATE, DELETE), study_groups (SELECT, UPDATE, DELETE), study_group_members (SELECT, DELETE), users (SELECT).
- UC18: Quản lý địa điểm
 - Mục đích: Quản lý danh sách địa điểm trong campus.

- Mô tả: Admin có thể thêm, chỉnh sửa, xóa địa điểm trong bảng locations (name, type, building, floor, description, coordinates, popular). Admin có thể đánh dấu các địa điểm phổ biến (popular = true) để hiển thị ưu tiên.
- Dữ liệu liên quan: locations (INSERT, SELECT, UPDATE, DELETE).
 - UC19: Giám sát và thống kê hoạt động hệ thống
- Mục đích: Theo dõi hoạt động của người dùng và hiệu suất hệ thống.
- Mô tả: Admin xem báo cáo tổng quan, thống kê số lượng người dùng (users), bài đăng (posts), sự kiện (events), câu lạc bộ (clubs), câu hỏi (questions), nhóm học tập (study_groups) theo thời gian. Hệ thống tổng hợp dữ liệu từ các trường likes_count, comments_count, members_count, attendees_count, replies_count để hiển thị thống kê.
- Dữ liệu liên quan: users (SELECT), posts (SELECT), events (SELECT), clubs (SELECT), questions (SELECT), study_groups (SELECT), comments (SELECT), post_likes (SELECT).

Mối quan hệ giữa các Use Cases

- Quan hệ bao gồm (Include):
 - UC02 (Quản lý hồ sơ) là điều kiện tiên quyết cho UC04 (Đăng bài viết) - người dùng cần có hồ sơ đầy đủ để đăng bài.
 - UC01 (Đăng nhập) là điều kiện tiên quyết cho tất cả UC02-UC11 - phải đăng nhập trước khi sử dụng chức năng.
 - UC11 (Tìm kiếm địa điểm) được sử dụng trong UC07 (Nhóm học tập) và UC09 (Sự kiện) - chọn location từ bảng locations.
 - UC05 (Khóa học) liên kết với UC06 (Q&A) - câu hỏi được phân loại theo course.
- Quan hệ mở rộng (Extend):
 - UC08 (Quản lý CLB) mở rộng UC09 (Sự kiện) - CLB có thể tạo sự kiện thông qua club_id trong events.
 - UC12-UC19 (Chức năng Admin) mở rộng UC01-UC11 - Admin có thể thực hiện các hành động của sinh viên kèm quyền quản lý cao hơn.
 - UC08 (Quản lý CLB) mở rộng để bao gồm club_posts, club_activities - các hoạt động nội bộ của CLB.
- Quan hệ kế thừa (Generalization):
 - Quản trị viên (role = 'admin') kế thừa toàn bộ quyền của Sinh viên (role = 'user').

- club_post_comments và comments đều kế thừa cấu trúc tương tự (hỗ trợ parent_id cho nested comments).
- question_replies tương tự comments nhưng có thêm is_solution để đánh dấu câu trả lời đúng.

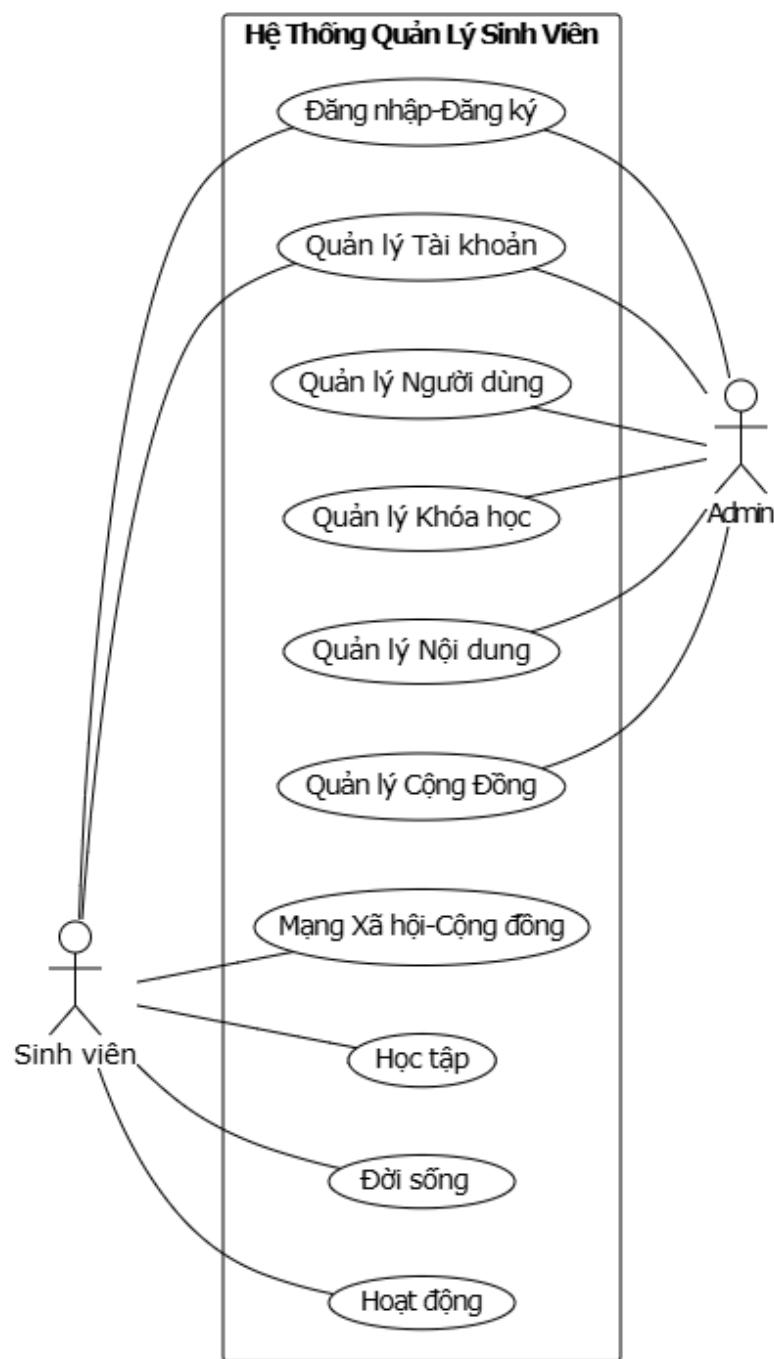
Ưu tiên thực hiện Use Cases

- Mức độ cao (Core Functions):
 - UC01: Đăng nhập hệ thống.
 - UC02: Quản lý hồ sơ cá nhân.
 - UC04: Đăng bài và tương tác cộng đồng (posts, post_likes, comments).
 - UC06: Q&A học tập (questions, question_replies).
 - UC07: Nhóm học tập (study_groups, study_group_members).
 - UC09: Quản lý sự kiện (events, event_attendees).
 - UC12-UC19: Các chức năng quản lý của Admin.
- Mức độ trung bình (Supporting Functions):
 - UC03: Quản lý thời khóa biểu (class_schedules).
 - UC05: Quản lý khóa học cá nhân (courses).
 - UC08: Câu lạc bộ (clubs, club_members, club_posts, club_activities).
 - UC10: Thông báo (announcements).
- Mức độ thấp (Enhanced Functions):
 - UC11: Tìm kiếm địa điểm (locations).
 - UC18: Quản lý địa điểm (dành cho Admin).

2.4.2 Mô hình Use-case

Use-case tổng quan

Hệ thống Phenikaa Connect được thiết kế để phục vụ nhu cầu của sinh viên trong việc quản lý học tập, kết nối cộng đồng và sử dụng các tiện ích trường học. Ứng dụng cung cấp một hệ sinh thái số toàn diện với các chức năng chính bao gồm quản lý học tập, tương tác xã hội, tham gia sự kiện và quản lý thông tin cá nhân.



Hình 1 Usecase tổng quan

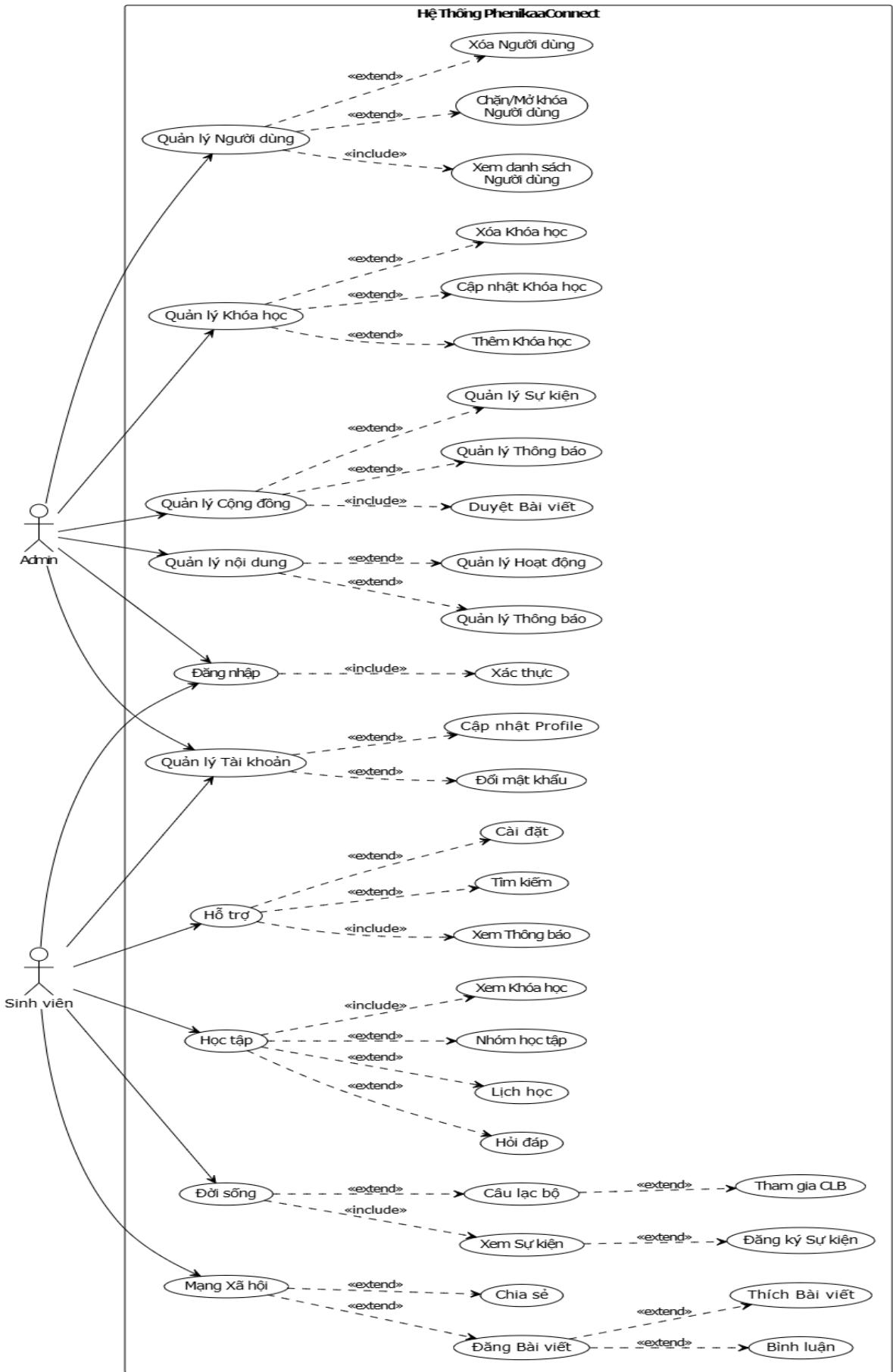
Phân rã Quản lý tài khoản

Quản lý Tài khoản là nền tảng cơ bản và quan trọng nhất của hệ thống PhenikaaConnect. Module này có trách nhiệm quản lý toàn bộ quy trình xác thực, cấp quyền truy cập, và bảo mật dữ liệu cá nhân cho cả Admin và Sinh viên.

Dựa trên sơ đồ Use Case, Quản lý Tài khoản có thể được phân rã thành các chức năng chính, tích hợp cả các Use Case liên quan:

- Xác thực & Đăng nhập: Use Case Đăng nhập là bắt buộc và bao gồm hành động Xác thực. Chức năng này kiểm soát quy trình người dùng nhận diện bản thân với hệ thống và cấp quyền truy cập.
- Quản lý Hồ sơ Cá nhân: Chức năng chính là Quản lý Tài khoản, được mở rộng bởi Cập nhật Profile. Chức năng này cho phép người dùng xem và thay đổi thông tin cá nhân.
- Quản lý Bảo mật Tài khoản: Chức năng Quản lý Tài khoản cũng được mở rộng bởi Đổi mật khẩu. Ngoài ra, Cài đặt (mở rộng từ Hỗ trợ) cũng có thể bao gồm các tùy chọn bảo mật cá nhân (ví dụ: xác thực hai yếu tố, quản lý thiết bị).
- Quản lý Quyền & Phân quyền: Mặc dù không được thể hiện trực tiếp dưới dạng Quản lý Tài khoản nhưng được ngầm định thông qua vai trò của các Actor: Admin có quyền quản lý toàn bộ (Quản lý Người dùng, Khóa học), còn Sinh viên chỉ có quyền truy cập các chức năng người dùng (Học tập, Đời sống).

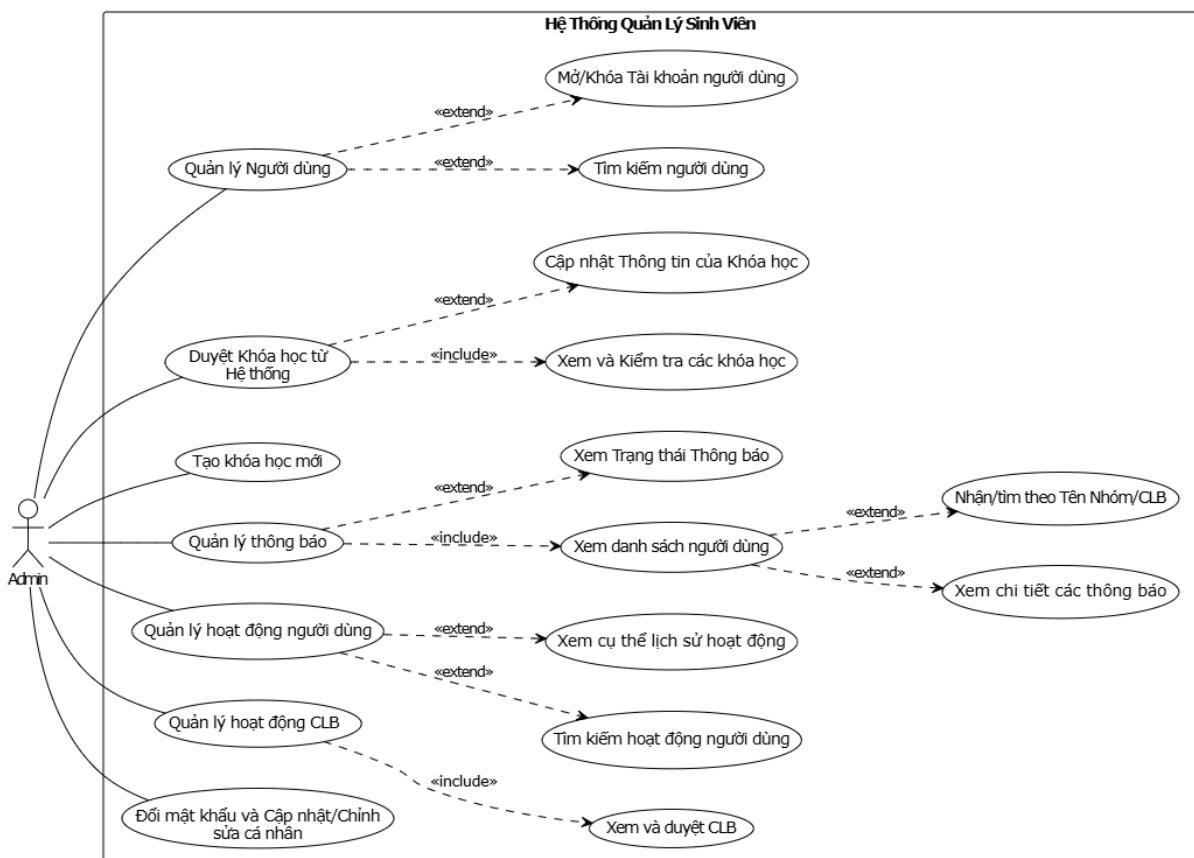
Mỗi chức năng con được thiết kế để đảm bảo người dùng (Sinh viên và Admin) có thể truy cập hệ thống một cách an toàn, quản lý thông tin cá nhân hiệu quả, bảo vệ tài khoản khỏi truy cập trái phép, và tuân thủ các quy định về quyền hạn trong hệ thống. Các use case trong module này đóng vai trò nền tảng cho tất cả các module khác, vì tất cả các hoạt động khác (như Quản lý Khóa học, Học tập, Mạng xã hội,...) đều phải thông qua Xác thực tài khoản trước khi được phép thực hiện.



Hình 2 Usecase phân rã quản lý tài khoản

Phân rã Quản trị viên

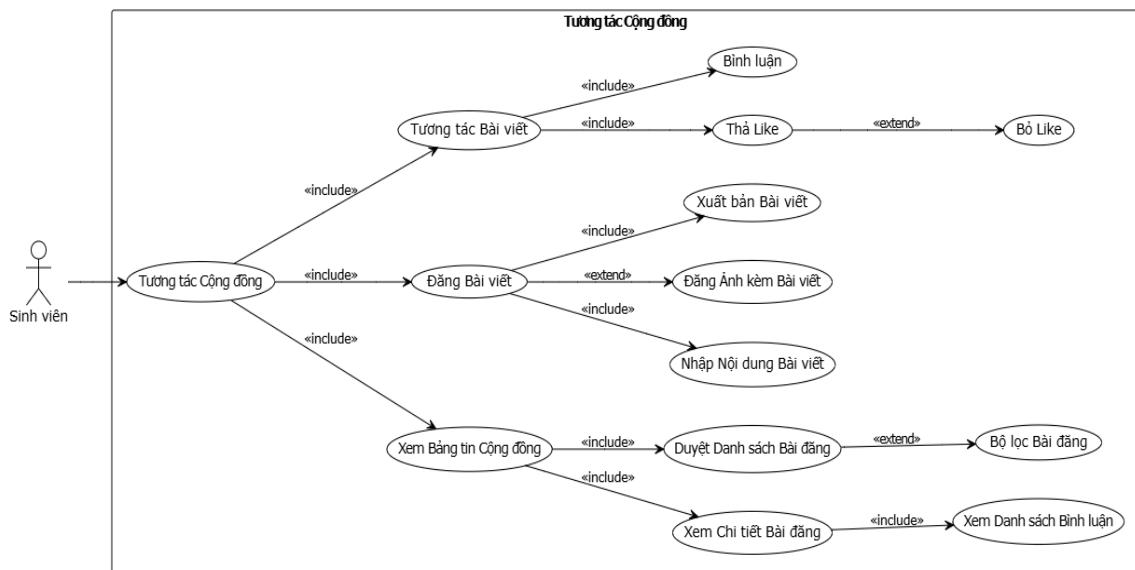
Tác nhân Quản trị viên (Admin) là trung tâm của Hệ thống Quản lý Sinh viên, đảm nhận vai trò cốt lõi trong việc duy trì ổn định và chất lượng hệ thống thông qua nhiều chức năng quan trọng. Về Quản lý Người dùng, Admin có thể Mở/Khóa Tài khoản và Tìm kiếm người dùng để đảm bảo an ninh. Admin chịu trách nhiệm kiểm duyệt nội dung bằng cách Duyệt Khóa học từ Hệ thống, bao gồm Xem và Kiểm tra các khóa học (sau đó có thể Cập nhật Thông tin của khóa học), đồng thời Admin cũng có quyền Tạo khóa học mới. Đối với việc truyền thông, Admin Quản lý thông báo bằng cách Xem trạng thái Thông báo và Xem danh sách người dùng để phân loại nhận thông báo (bao gồm cả việc Nhắn/Tạo Tên Nhóm/CLB và Xem chi tiết các thông báo). Chức năng giám sát được thực hiện qua Quản lý hoạt động người dùng (cho phép Xem cụ thể lịch sử hoạt động và Tìm kiếm hoạt động người dùng) và Quản lý hoạt động CLB. Cuối cùng, Admin cũng quản lý tài khoản của mình thông qua chức năng Đổi mật khẩu và Cập nhật/Chỉnh sửa cá nhân, trong đó bao gồm cả chức năng Xem và Duyệt CLB, cho thấy sự can thiệp trực tiếp vào quy trình hoạt động của các tổ chức này.



Hình 3 Usecase phân rã quản trị viên

Phân rã Tương tác cộng đồng

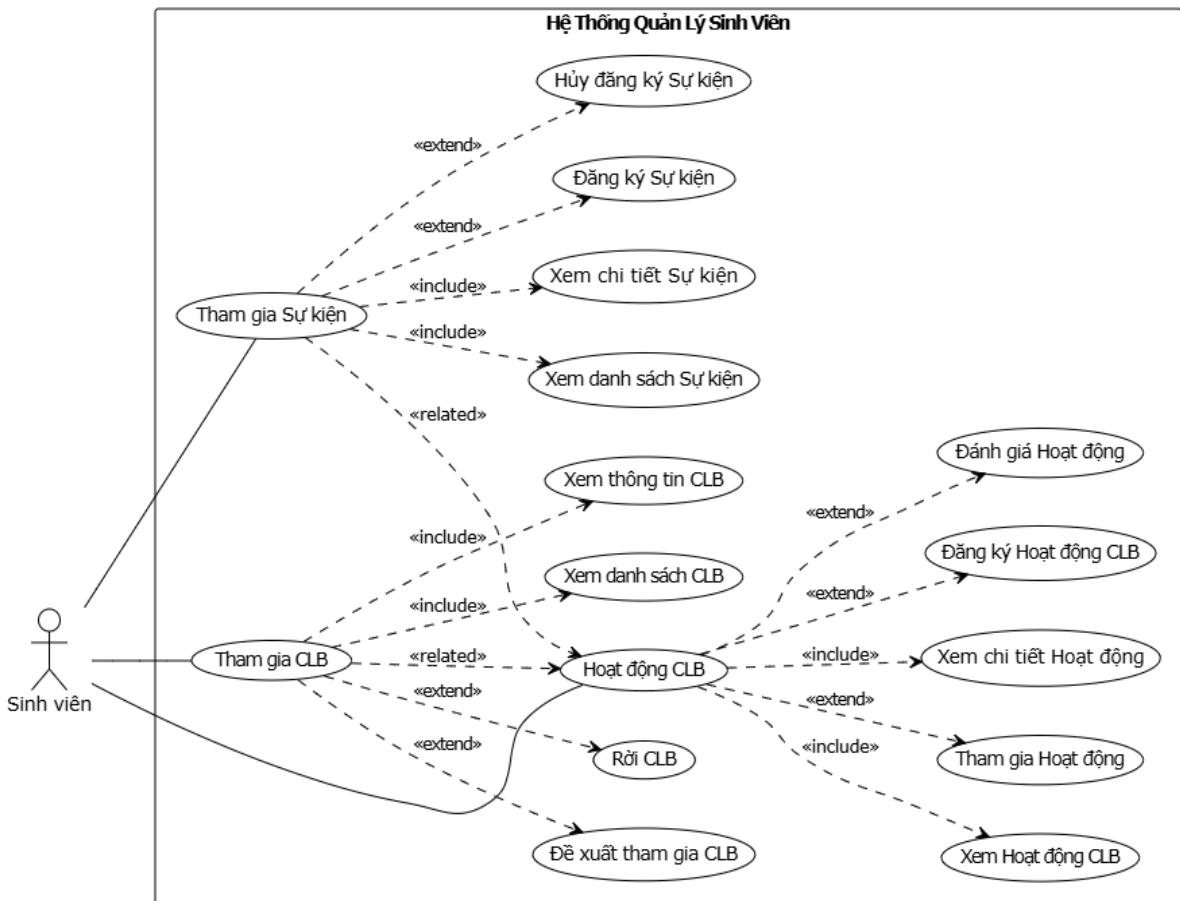
Tương tác cộng đồng tạo ra một không gian số nơi sinh viên có thể chia sẻ suy nghĩ, kết nối với bạn bè và tham gia các cuộc thảo luận. Chức năng này bao gồm việc đăng bài, tương tác với nội dung của người khác thông qua like và comment, đồng thời duyệt các bài đăng từ cộng đồng. Hệ thống hỗ trợ đăng ảnh kèm theo nội dung để tăng tính trực quan và hấp dẫn.



Hình 4 Usecase phân rã tương tác cộng đồng

Phân rã Tham gia đời sống trường

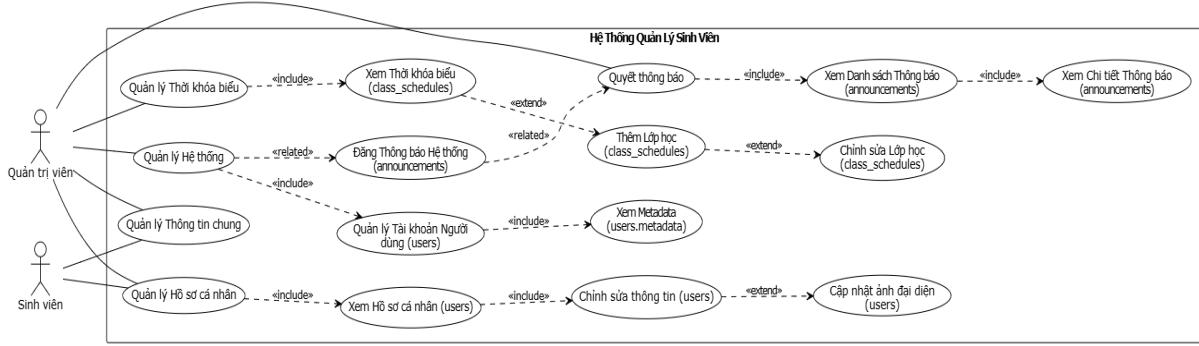
Chức năng Tham gia đời sống trường tập trung vào việc Sinh viên tương tác với các sự kiện và câu lạc bộ. Khi Tham gia Sự kiện, sinh viên bắt buộc phải Xem danh sách Sự kiện và Xem chi tiết Sự kiện trước khi thực hiện các hành động mở rộng như Đăng ký Sự kiện hoặc Hủy đăng ký Sự kiện. Chức năng này có liên hệ với việc Xem thông tin CLB (Câu lạc bộ) – đơn vị thường tổ chức sự kiện. Đối với Tham gia CLB, sau khi Xem danh sách CLB, sinh viên có thể Đề xuất tham gia CLB hoặc Rời CLB (tính năng mở rộng). Việc tham gia CLB đồng thời bao gồm việc Xem Hoạt động CLB và tham gia vào Hoạt động CLB cụ thể: tại đây, sinh viên bắt buộc Xem chi tiết Hoạt động và Tham gia Hoạt động. Các tùy chọn mở rộng trong hoạt động CLB bao gồm Đăng ký Hoạt động CLB và Đánh giá Hoạt động sau khi tham gia.



Hình 5 Usecase phân rã tham gia đòn sóng trường

Phân rã Quản lý thông tin chung

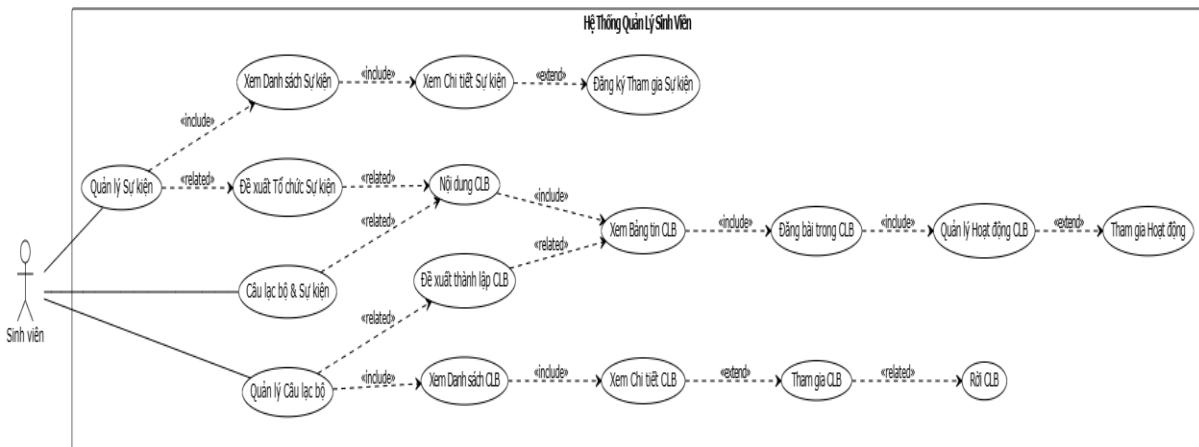
Phạm vi Quản lý Thông tin Chung là nền tảng cốt lõi, quản lý các dữ liệu cơ bản như hồ sơ người dùng, thông báo và thời khóa biểu. Sinh viên là tác nhân chính trong việc tự quản lý thông tin cá nhân và học tập: họ thực hiện Quản lý Hồ sơ cá nhân (bao gồm Xem và Chính sửa thông tin, có thể Cập nhật ảnh đại diện), Quản lý Thời khóa biểu cá nhân (để Xem Thời khóa biểu), và có quyền Duyệt Thông báo (Xem Danh sách và Xem chi tiết). Song song đó, Quản trị viên đóng vai trò giám sát và quản trị hệ thống ở cấp độ cao hơn thông qua Quản lý Hệ thống, bao gồm Đăng Thông báo hệ thống và Quản lý Tài khoản Người dùng (có chức năng Xem Metadata). Admin cũng có quyền Quản lý Thời khóa biểu ở cấp độ toàn hệ thống, cho phép Thêm Lớp học và Chính sửa Lớp học, đồng thời Admin cũng tham gia vào chức năng Duyệt Thông báo chung. Như vậy, sinh viên tập trung vào quản lý cá nhân, trong khi Quản trị viên quản lý và can thiệp vào dữ liệu nền tảng trên toàn hệ thống.



Hình 6 Usecase phân rã Quản lý thông tin chung

Phân rã Câu lạc bộ & Sự kiện

Mô tả các chức năng cho phép Sinh viên tương tác, tham gia và tổ chức các hoạt động ngoại khóa trên nền tảng. Chức năng được phân chia thành ba nhóm: Quản lý Câu lạc bộ, tập trung vào việc Đề xuất thành lập CLB, Xem danh sách và thực hiện các hành động Tham gia/Rời CLB; Nội dung CLB, cho phép sinh viên Đăng bài và Xem Bảng tin CLB, cùng với khả năng Quản lý và Tham gia Hoạt động CLB; và cuối cùng là Quản lý Sự kiện, bao gồm việc Đề xuất Tổ chức Sự kiện, Duyệt Danh sách Sự kiện và Đăng ký tham gia Sự kiện. Các chức năng này có mối liên hệ chặt chẽ, ví dụ như việc xem chi tiết CLB là điều kiện tiên quyết để xem nội dung hoặc tham gia, và việc đề xuất CLB thường liên quan đến việc tổ chức sự kiện.

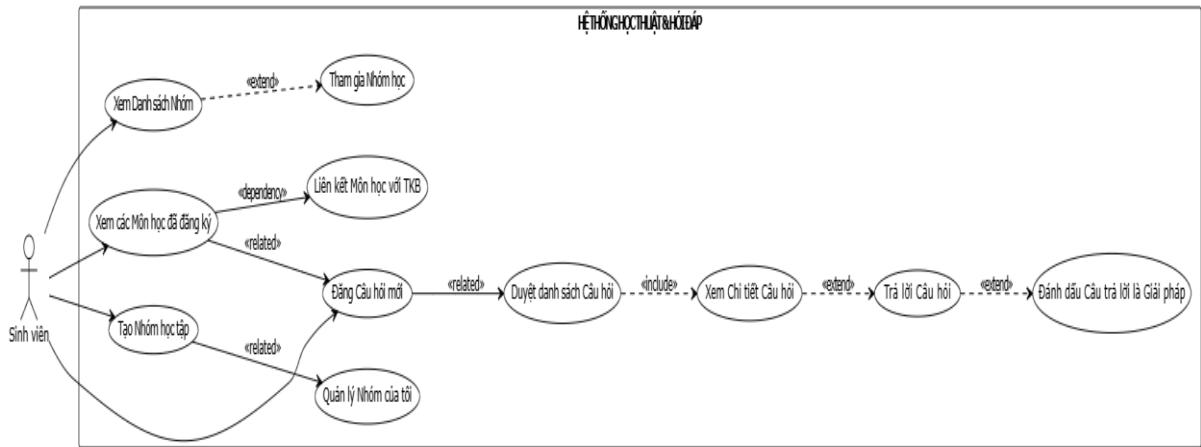


Hình 7 Usecase phân rã Câu lạc bộ và sự kiện

Phân rã Quản lý học tập

Phân rã Học thuật & Hỏi đáp là nhóm chức năng quan trọng hỗ trợ sinh viên trong hành trình học tập. Nhóm này cho phép Sinh viên sử dụng nền tảng như một công cụ học tập hợp tác, bao gồm ba mảng chính: Quản lý Hỏi đáp (Q&A), nơi sinh viên có thể Đăng Câu hỏi mới về các môn học, Trả lời và Đánh dấu giải pháp cho các thắc mắc của bạn bè để xây dựng kho kiến thức chung; Quản lý Nhóm học,

nơi họ có thể Tạo hoặc Tham gia Nhóm học tập để tổ chức các buổi học và chia sẻ tài nguyên; và Quản lý Môn học, giúp Xem các Môn học đã đăng ký và Liên kết chúng với thời khóa biểu, đảm bảo các hoạt động Q&A luôn gắn liền với chương trình học tập hiện tại.



Hình 8 Usecase phân rã Quản lý học tập

2.4.3 Đặc tả Use-case

UC1.1: Đăng ký tài khoản

Sinh viên muốn tạo một tài khoản mới trong hệ thống để có thể sử dụng các chức năng của ứng dụng. Hệ thống yêu cầu người dùng cung cấp thông tin cơ bản bao gồm email, mật khẩu, họ tên, mã số sinh viên, ngành học, năm học và số điện thoại. Sau khi người dùng điền đầy đủ thông tin và nhấn nút đăng ký, hệ thống sẽ kiểm tra tính hợp lệ của email, tính duy nhất của email và mã số sinh viên, sau đó tạo tài khoản mới trong cơ sở dữ liệu. Nếu quá trình diễn ra thành công, người dùng sẽ được chuyển đến màn hình chính của ứng dụng. Trong trường hợp có lỗi, hệ thống sẽ hiển thị thông báo lỗi cụ thể để người dùng biết và có thể khắc phục.

UC1.2: Đăng nhập

Sinh viên đã có tài khoản muốn đăng nhập vào hệ thống để truy cập các chức năng của ứng dụng. Người dùng nhập email và mật khẩu vào form đăng nhập, sau đó nhấn nút đăng nhập. Hệ thống sẽ xác thực thông tin đăng nhập bằng cách kiểm tra trong cơ sở dữ liệu xác thực. Nếu thông tin chính xác, hệ thống sẽ tạo phiên đăng nhập và lưu trữ thông tin người dùng hiện tại, sau đó chuyển hướng đến màn hình chính. Nếu thông tin không chính xác, hệ thống hiển thị thông báo lỗi và yêu cầu người dùng nhập lại.

UC1.4: Quên mật khẩu

Sinh viên quên mật khẩu đăng nhập có thể yêu cầu đặt lại mật khẩu thông qua email. Người dùng nhấp vào liên kết "Quên mật khẩu" trên màn hình đăng nhập, sau đó nhập email đã đăng ký vào hộp thoại. Hệ thống sẽ gửi email chứa liên kết đặt lại mật khẩu đến địa chỉ email đó.

Người dùng mở email và nhấp vào liên kết để được chuyển đến trang đặt lại mật khẩu, nơi họ có thể nhập mật khẩu mới.

UC2.1: Xem lịch học

Sinh viên muốn xem lịch học của mình trong tuần để biết được các môn học sẽ diễn ra ở đâu và vào thời gian nào. Khi người dùng truy cập vào tab lịch học, hệ thống sẽ lấy dữ liệu lịch học từ cơ sở dữ liệu dựa trên thông tin người dùng hiện tại. Lịch học được hiển thị theo từng ngày trong tuần, với khả năng chuyển đổi giữa các ngày thông qua các nút chọn ngày. Mỗi tiết học hiển thị thông tin về tên môn học, giảng viên, phòng học và thời gian.

UC2.3: Tham gia Q&A

Sinh viên có thể tham gia diễn đàn hỏi đáp để đặt câu hỏi về các môn học hoặc trả lời câu hỏi của người khác. Khi truy cập vào tab Q&A, hệ thống hiển thị danh sách các câu hỏi được sắp xếp theo thời gian, mỗi câu hỏi hiển thị môn học liên quan, tiêu đề, tác giả và số lượng câu trả lời. Người dùng có thể sử dụng chức năng tìm kiếm để lọc câu hỏi theo từ khóa hoặc môn học. Khi nhấn vào một câu hỏi, người dùng có thể xem chi tiết nội dung câu hỏi và các câu trả lời, đồng thời có thể thêm câu trả lời mới hoặc đặt câu hỏi mới thông qua nút "Đặt câu hỏi".

UC3.1: Đăng bài viết

Sinh viên muốn chia sẻ suy nghĩ, hình ảnh hoặc cập nhật trạng thái với cộng đồng thông qua việc đăng bài viết. Người dùng nhập nội dung bài viết vào ô nhập liệu trên màn hình cộng đồng, có thể chọn thêm ảnh từ thiết bị của mình. Sau khi hoàn tất, người dùng nhấn nút "Đăng bài". Hệ thống sẽ kiểm tra xác thực người dùng, nếu có ảnh thì tải lên storage, sau đó lưu bài viết vào cơ sở dữ liệu. Bài viết mới sẽ xuất hiện ở đầu danh sách bảng tin và hiển thị cho tất cả người dùng khác.

UC3.2: Xem bảng tin

Sinh viên muốn xem các bài viết từ cộng đồng để cập nhật thông tin và tương tác với bạn bè. Khi người dùng mở màn hình cộng đồng, hệ thống sẽ tải danh sách các bài viết từ cơ sở dữ liệu, được sắp xếp theo thời gian đăng mới nhất. Mỗi bài viết hiển thị thông tin tác giả, nội dung, hình ảnh nếu có, số lượng lượt thích và bình luận. Người dùng có thể cuộn để xem thêm bài viết hoặc nhấn vào một bài viết để xem chi tiết và các bình luận.

UC3.3: Thích bài viết

Sinh viên muốn thể hiện sự đồng cảm hoặc đánh giá tích cực về một bài viết thông qua việc thích bài viết. Khi người dùng nhấn vào biểu tượng tim trên một bài viết, hệ thống sẽ kiểm tra xem người dùng đã thích bài viết này chưa. Nếu chưa thích, hệ thống sẽ thêm một bản ghi vào

bảng post_likes và tăng số lượng lượt thích. Nếu đã thích, hệ thống sẽ xóa bản ghi và giảm số lượng lượt thích. Giao diện sẽ cập nhật ngay lập tức để phản ánh trạng thái mới.

UC4.2: Tham gia sự kiện

Sinh viên muốn tham gia các sự kiện được tổ chức trong trường để mở rộng mạng lưới và học hỏi. Khi người dùng mở tab sự kiện, hệ thống hiển thị danh sách các sự kiện sắp tới, có thể lọc theo danh mục như học thuật, văn hóa, thể thao hoặc nghề nghiệp. Mỗi sự kiện hiển thị tiêu đề, thời gian, địa điểm, người tổ chức và số lượng người tham gia. Người dùng có thể nhấn vào sự kiện để xem chi tiết, sau đó nhấn nút đăng ký tham gia. Hệ thống sẽ thêm người dùng vào danh sách người tham gia và cập nhật số lượng.

UC4.3: Tham gia câu lạc bộ

Sinh viên muốn tham gia các câu lạc bộ trong trường để theo đuổi sở thích và phát triển kỹ năng. Khi người dùng mở tab câu lạc bộ, hệ thống hiển thị danh sách các câu lạc bộ đang hoạt động, mỗi câu lạc bộ hiển thị tên, mô tả, danh mục, số lượng thành viên và trạng thái hoạt động. Người dùng có thể xem chi tiết câu lạc bộ và nhấn nút "Tham gia" để trở thành thành viên. Hệ thống kiểm tra xem người dùng đã là thành viên chưa, nếu chưa thì thêm vào bảng club_members và cập nhật số lượng thành viên.

UC5.1: Xem và chỉnh sửa hồ sơ

Sinh viên muốn xem và cập nhật thông tin cá nhân của mình để đảm bảo thông tin luôn chính xác và đầy đủ. Khi người dùng truy cập màn hình cá nhân, hệ thống hiển thị thông tin hồ sơ hiện tại bao gồm tên, mã số sinh viên, ngành học, năm học, email và số điện thoại. Người dùng có thể nhấn nút "Chỉnh sửa hồ sơ" để mở màn hình chỉnh sửa, nơi có thể cập nhật các thông tin này. Sau khi lưu, hệ thống sẽ cập nhật thông tin trong cơ sở dữ liệu và cập nhật giao diện hiển thị.

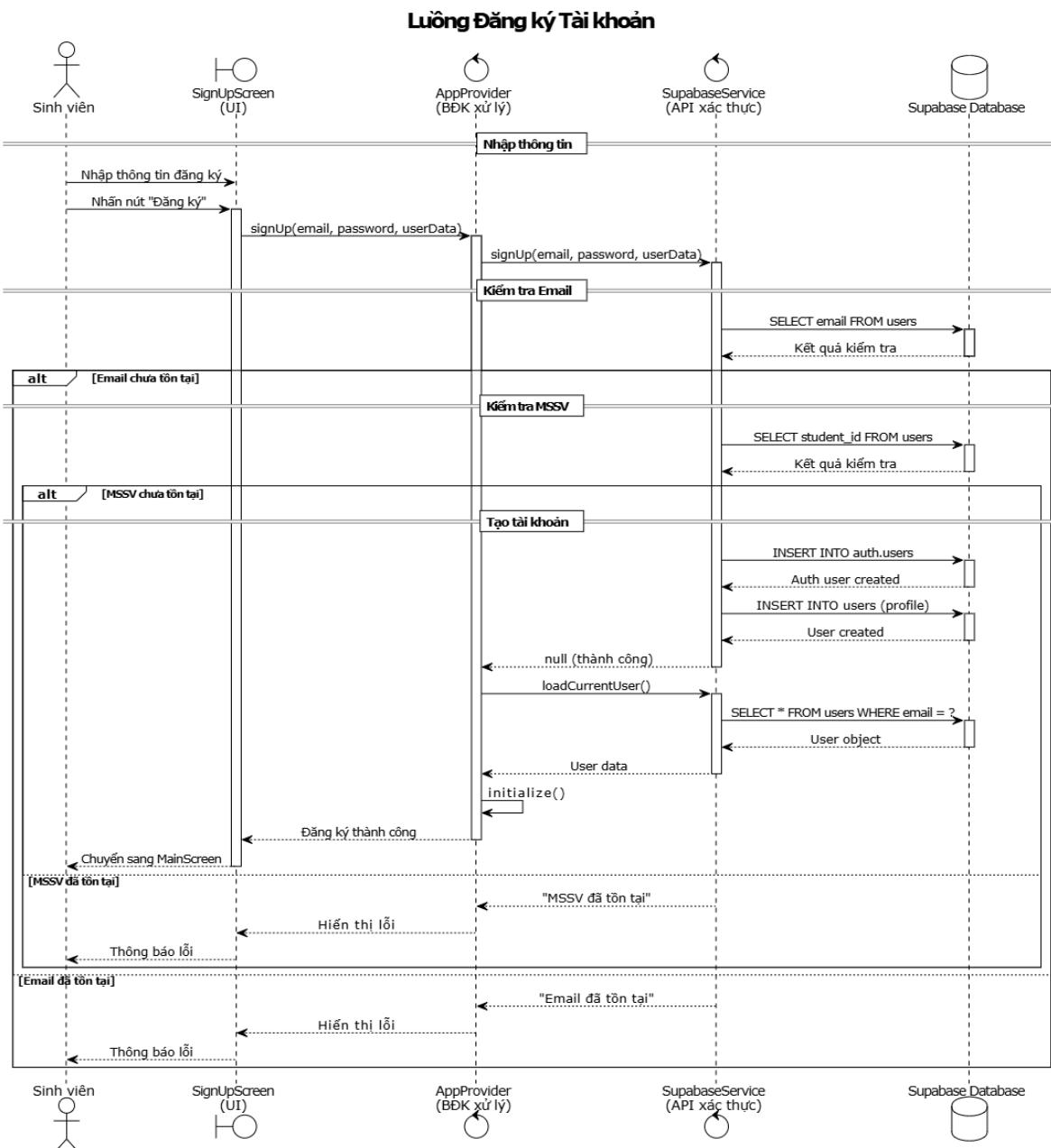
UC5.3: Xem thông báo từ trường

Sinh viên muốn xem các thông báo quan trọng từ nhà trường để không bỏ lỡ thông tin cần thiết. Khi người dùng truy cập tab thông báo, hệ thống sẽ tải danh sách các thông báo từ cơ sở dữ liệu, được sắp xếp theo thời gian mới nhất. Mỗi thông báo hiển thị tiêu đề, mức độ ưu tiên và thời gian đăng. Người dùng có thể nhấn vào thông báo để xem chi tiết nội dung. Hệ thống sẽ đánh dấu thông báo là đã đọc khi người dùng xem chi tiết, và lưu trạng thái này để không hiển thị lại trong số thông báo chưa đọc.

2.4.4 Các biểu đồ tuần tự (Sequence Diagrams)

Sequence Diagram: Đăng ký tài khoản

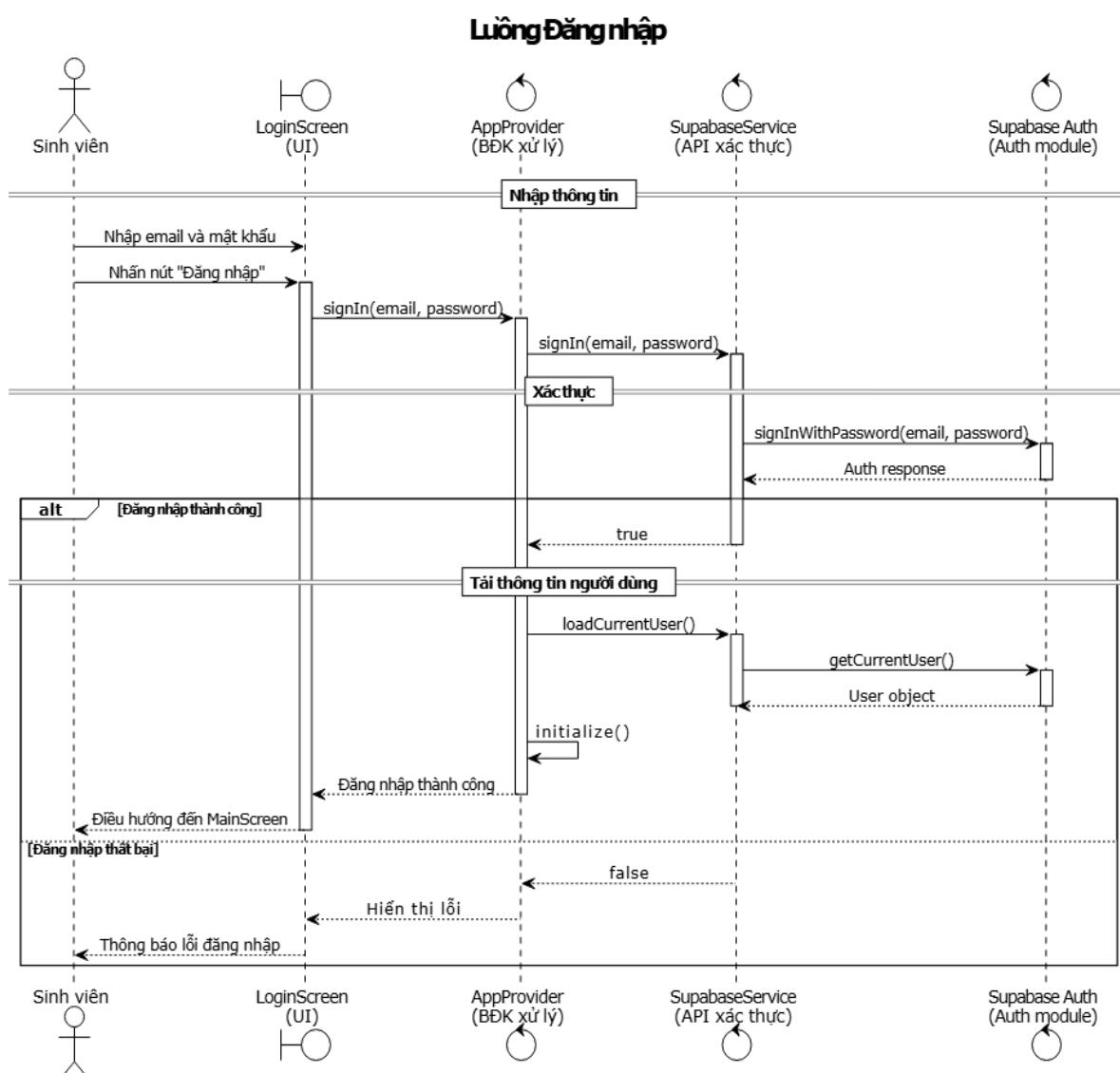
Quy trình đăng ký tài khoản bắt đầu khi người dùng điền thông tin vào form đăng ký và nhấn nút đăng ký. Ứng dụng sẽ gửi yêu cầu đến AppProvider, lớp này sẽ gọi SupabaseService để kiểm tra email và mã số sinh viên có trùng lặp không. Nếu không trùng, SupabaseService sẽ tạo tài khoản trong hệ thống xác thực và thêm thông tin người dùng vào bảng users. Sau khi hoàn tất, thông tin người dùng mới được trả về và AppProvider sẽ tải dữ liệu người dùng hiện tại, sau đó điều hướng đến màn hình chính.



Hình 9 Biểu đồ tuần tự đăng ký tài khoản

Sequence Diagram: Đăng nhập

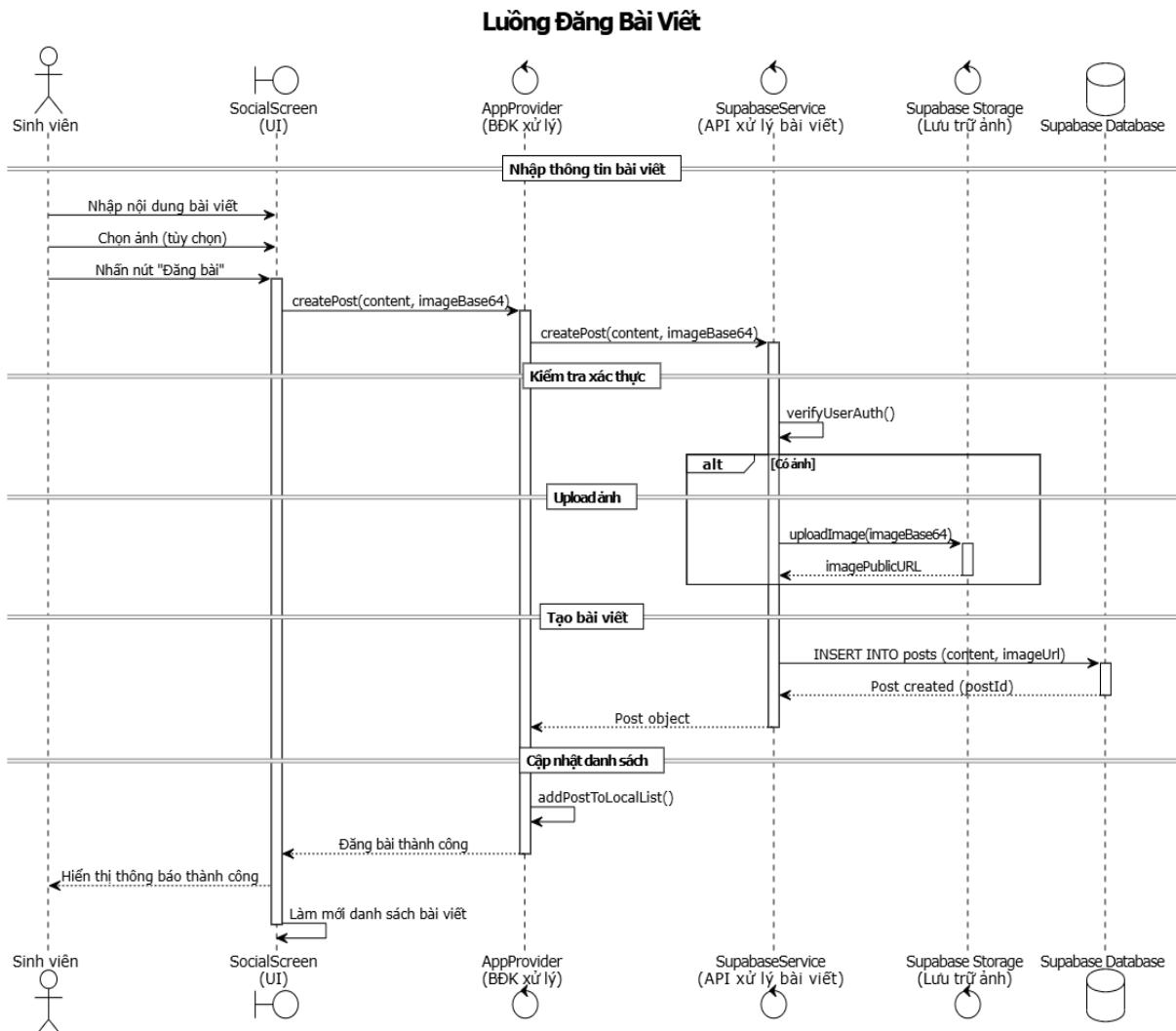
Quy trình đăng nhập diễn ra khi người dùng nhập email và mật khẩu vào form đăng nhập và nhấn nút đăng nhập. Ứng dụng gửi thông tin đăng nhập đến AppProvider, lớp này sẽ gọi SupabaseService để xác thực. SupabaseService sẽ kiểm tra thông tin đăng nhập với hệ thống xác thực của Supabase. Nếu thông tin chính xác, hệ thống sẽ tạo phiên đăng nhập và trả về thông tin người dùng. AppProvider sẽ tải thông tin người dùng hiện tại và khởi tạo dữ liệu, sau đó điều hướng đến màn hình chính.



Hình 10 Biểu đồ tuần tự đăng nhập

Sequence Diagram: Đăng bài viết

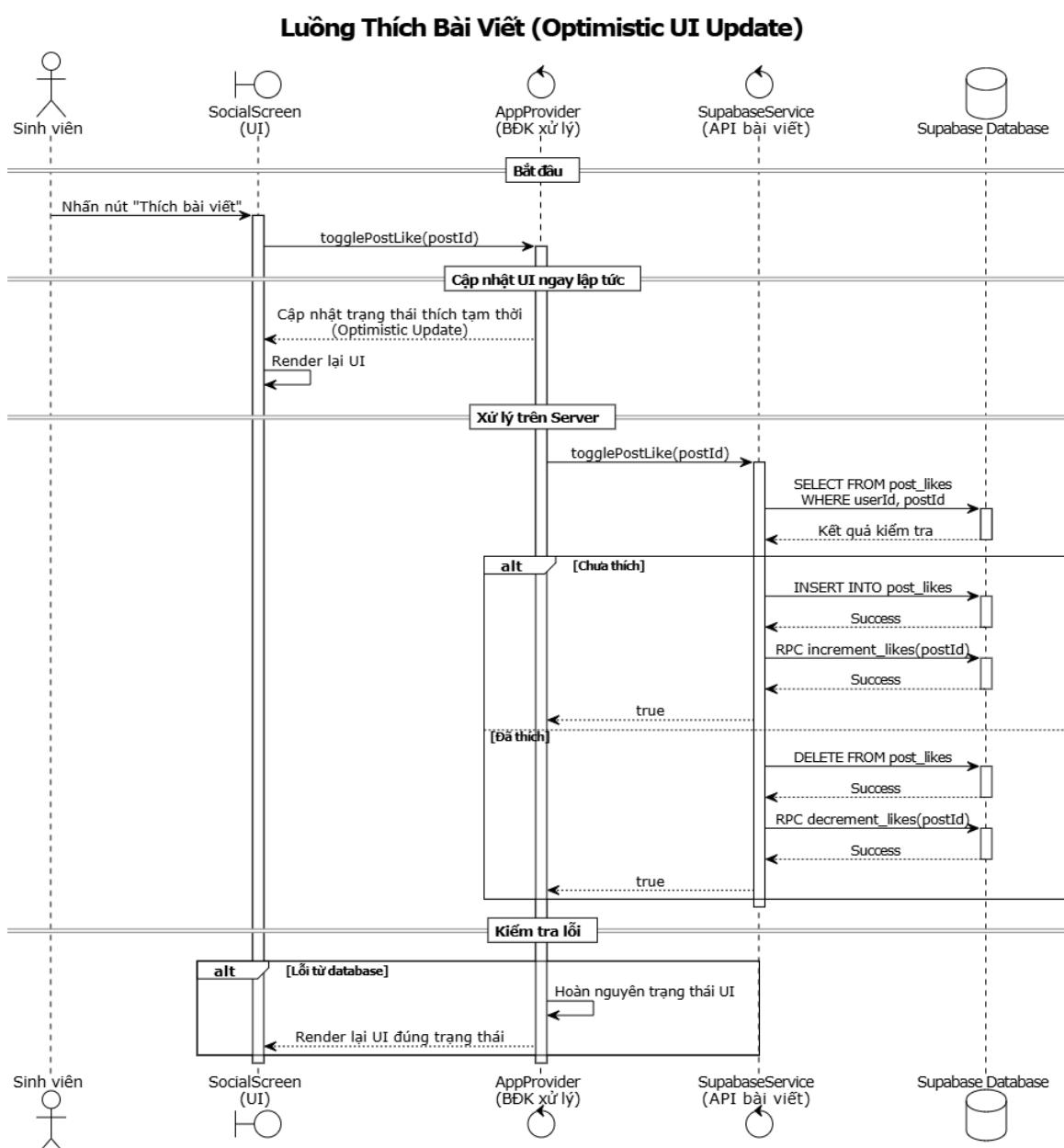
Khi người dùng muốn đăng một bài viết mới, họ sẽ nhập nội dung vào ô nhập liệu trên màn hình cộng đồng, có thể chọn thêm ảnh. Sau khi nhấn nút đăng bài, SocialScreen sẽ gọi AppProvider để tạo bài viết. AppProvider gọi SupabaseService, lớp này sẽ kiểm tra xác thực người dùng. Nếu có ảnh, SupabaseService sẽ tải ảnh lên Supabase Storage và lấy URL công khai. Sau đó, SupabaseService sẽ thêm bài viết vào bảng posts trong database. Bài viết mới được trả về và AppProvider sẽ thêm vào danh sách bài viết, sau đó cập nhật giao diện.



Hình 11 Sơ đồ luồng tự đăng bài viết

Sequence Diagram: Thích bài viết

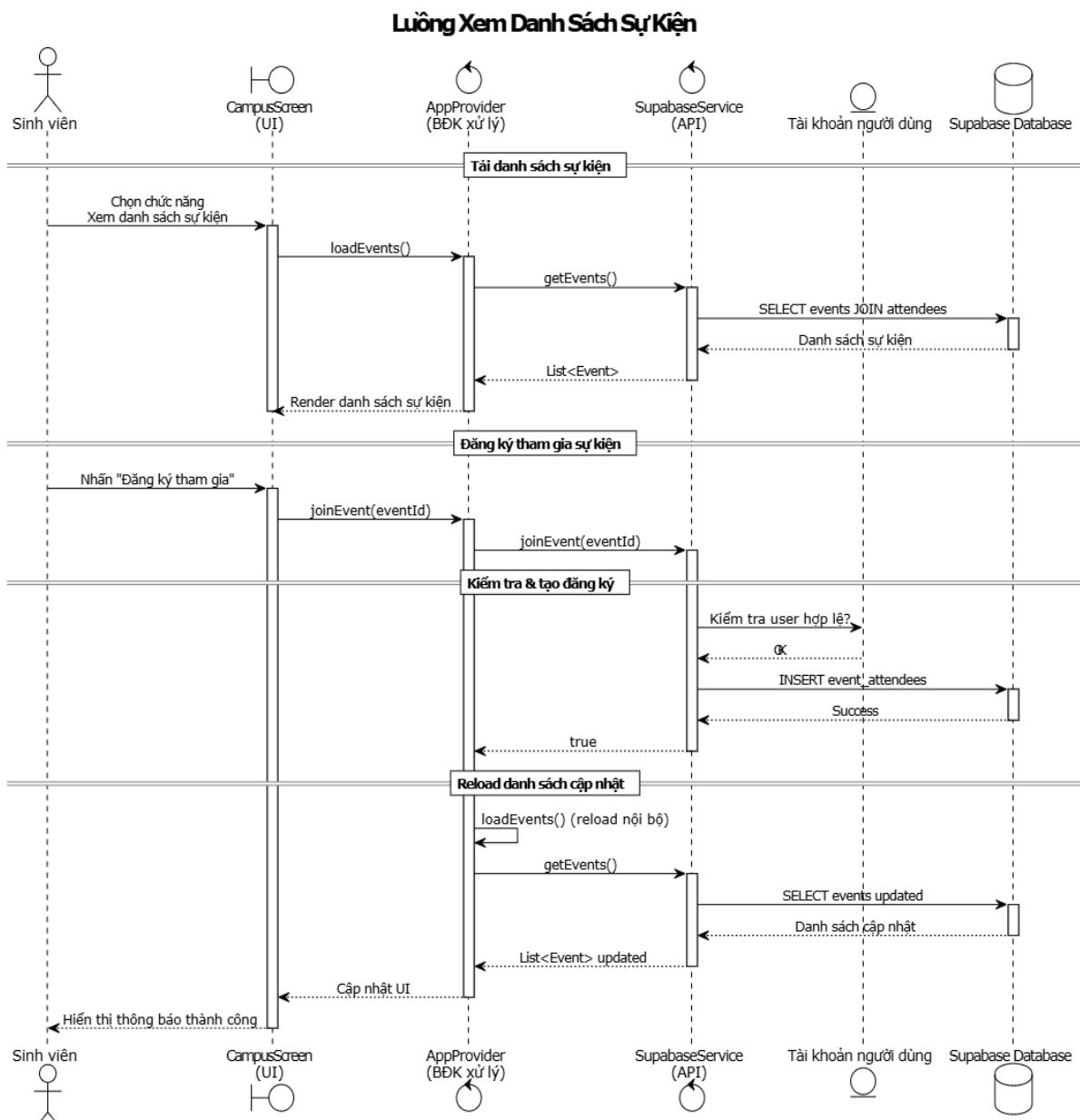
Quy trình thích bài viết được tối ưu hóa để cung cấp phản hồi ngay lập tức cho người dùng. Khi người dùng nhấn vào biểu tượng tim trên một bài viết, SocialScreen sẽ gọi AppProvider để thực hiện thao tác thích. AppProvider sẽ cập nhật trạng thái ngay lập tức trong bộ nhớ đệm giao diện phản hồi tức thì, sau đó gọi SupabaseService để đồng bộ với database. SupabaseService sẽ kiểm tra xem người dùng đã thích bài viết chưa bằng cách tìm trong bảng post_likes. Nếu chưa thích, hệ thống sẽ thêm bản ghi mới và tăng số lượng lượt thích. Nếu đã thích, hệ thống sẽ xóa bản ghi và giảm số lượng lượt thích.



Hình 12 Biểu đồ tuần tự thích bài viết

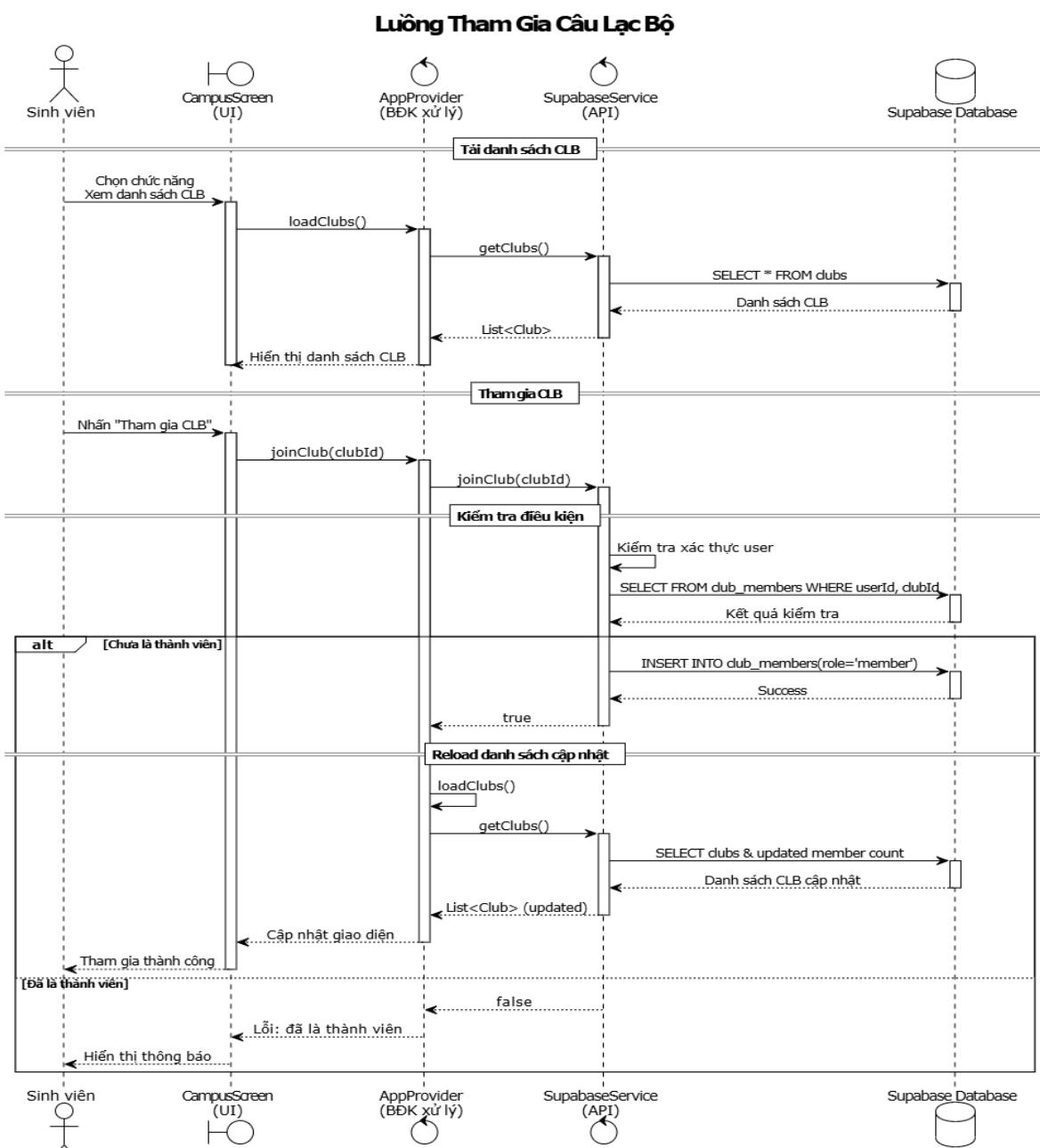
Sequence Diagram: Tham gia sự kiện

Khi người dùng muốn tham gia một sự kiện, họ sẽ xem danh sách sự kiện và nhấp vào nút đăng ký tham gia trên một sự kiện cụ thể. CampusScreen sẽ gọi AppProvider, lớp này sẽ gọi SupabaseService để đăng ký người dùng tham gia sự kiện. SupabaseService sẽ kiểm tra xác thực người dùng và thêm bản ghi vào bảng event_attendees để ghi nhận việc tham gia. Sau đó, AppProvider sẽ tải lại danh sách sự kiện để cập nhật số lượng người tham gia, và giao diện sẽ được cập nhật để phản ánh thay đổi này.



Sequence Diagram: Tham gia câu lạc bộ

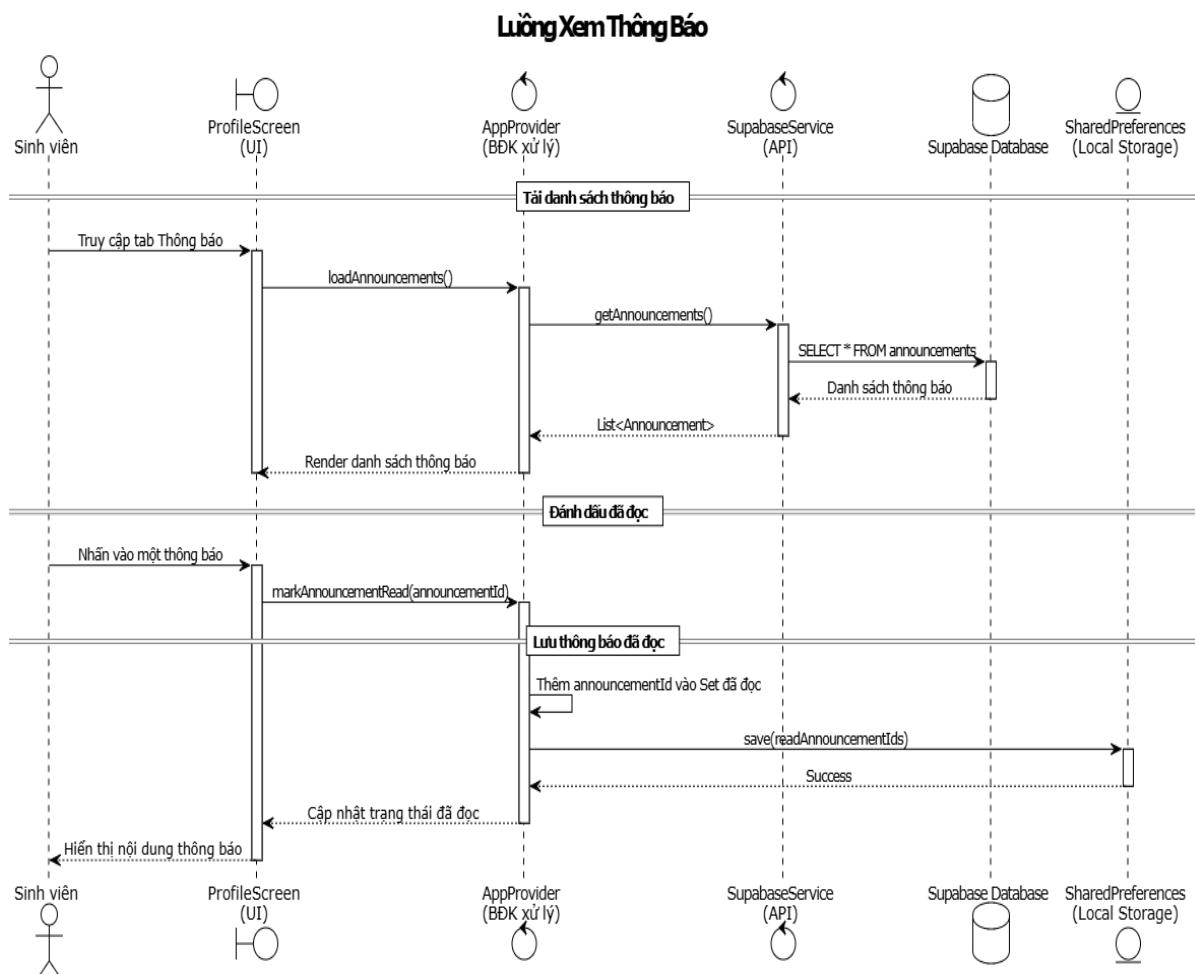
Quy trình tham gia câu lạc bộ bắt đầu khi người dùng xem danh sách các câu lạc bộ và quyết định tham gia một câu lạc bộ cụ thể. Khi người dùng nhấn nút "Tham gia", CampusScreen sẽ gọi AppProvider, lớp này sẽ gọi SupabaseService để xử lý yêu cầu. SupabaseService sẽ kiểm tra xem người dùng đã là thành viên của câu lạc bộ chưa bằng cách tìm kiếm trong bảng club_members. Nếu chưa phải thành viên, hệ thống sẽ thêm bản ghi mới với vai trò là thành viên thông thường. Sau đó, AppProvider sẽ tải lại danh sách câu lạc bộ để cập nhật số lượng thành viên, và giao diện sẽ được cập nhật tương ứng.



Hình 14 Biểu đồ tuần tự tham gia câu lạc bộ

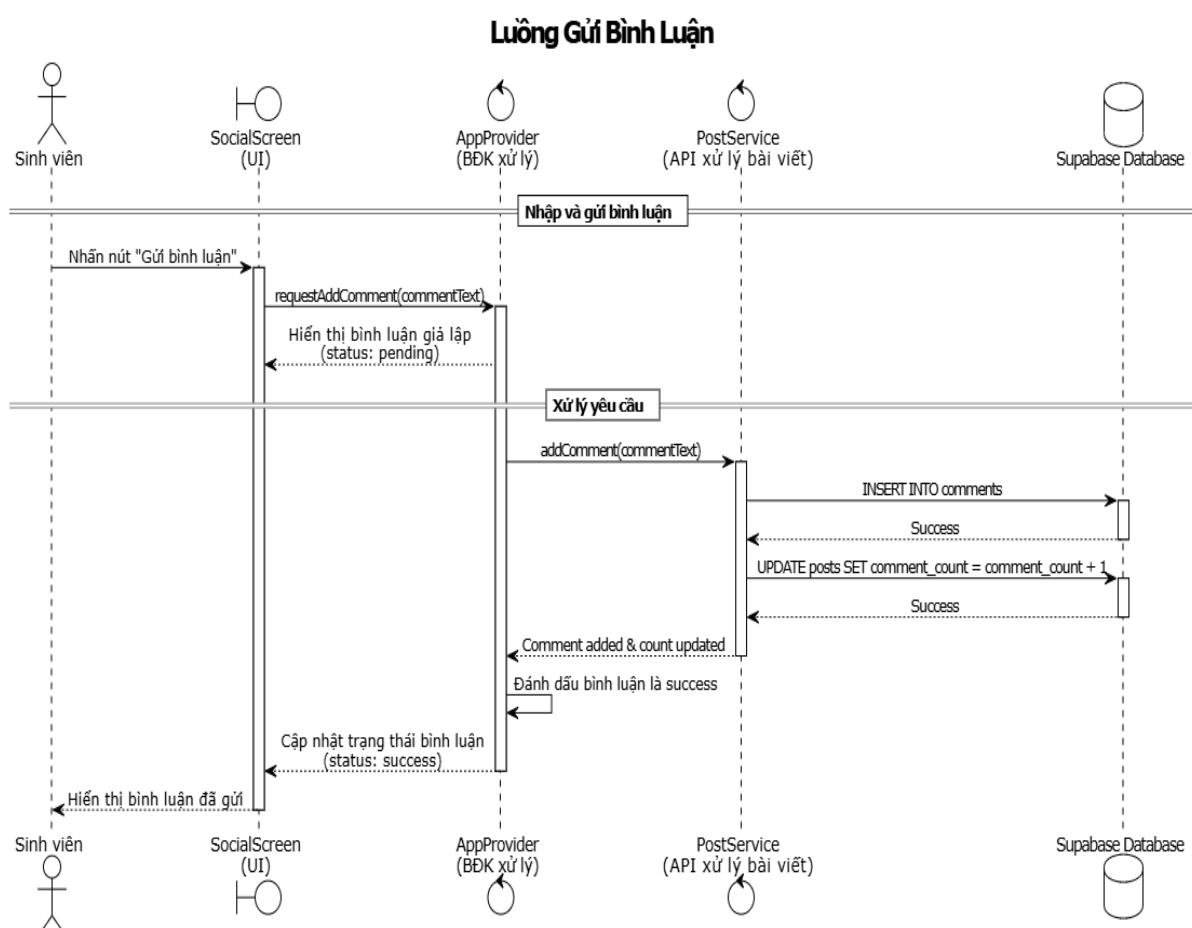
Sequence Diagram: Xem thông báo từ trường

Khi người dùng muốn xem các thông báo từ trường, họ sẽ truy cập vào tab thông báo trong màn hình cá nhân hoặc từ trang chủ. ProfileScreen sẽ yêu cầu AppProvider tải danh sách thông báo. AppProvider sẽ gọi SupabaseService để lấy danh sách thông báo từ database, được sắp xếp theo thời gian mới nhất. Sau khi nhận được danh sách, hệ thống sẽ hiển thị các thông báo trên giao diện. Khi người dùng nhấn vào một thông báo để xem chi tiết, hệ thống sẽ đánh dấu thông báo là đã đọc bằng cách lưu ID thông báo vào SharedPreferences. Việc này giúp hệ thống theo dõi những thông báo nào người dùng đã xem và không hiển thị lại trong số thông báo chưa đọc.



Sequence Diagram: Đăng Bình luận

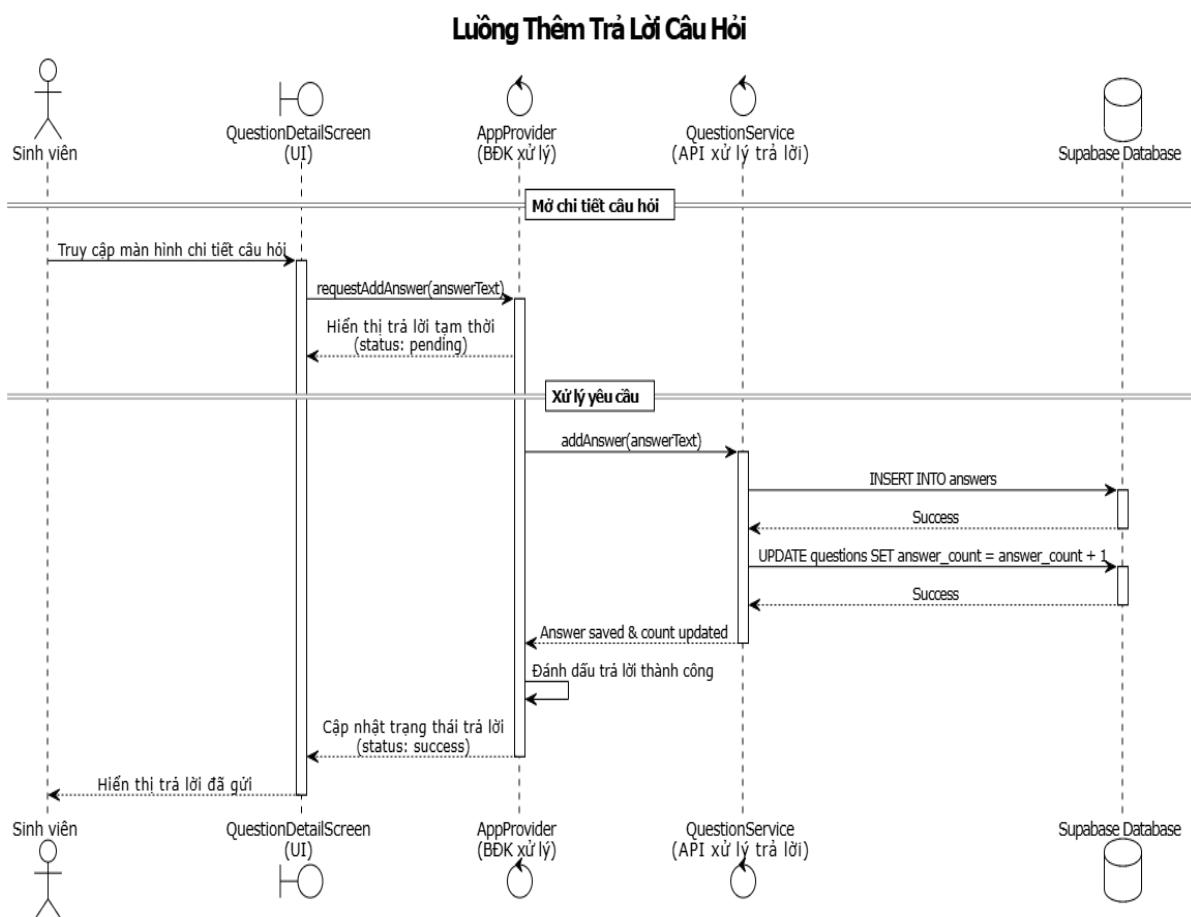
Khi người dùng muốn thêm một bình luận, họ Nhấn nút Gửi Bình luận trên SocialScreen. Màn hình này sẽ gửi yêu cầu Thêm Bình luận đến AppProvider đồng thời Hiển thị Bình luận ngay lập tức (đang chờ) cho người dùng. AppProvider tiếp nhận và gọi hàm Xử lý Thêm Bình luận trong PostService. PostService có trách nhiệm tương tác với SupabaseDB để Lưu bản ghi Bình luận mới và Cập nhật Số lượng bình luận của bài đăng. Sau khi nhận được Xác nhận Thành công từ dịch vụ, AppProvider thực hiện thao tác Hoàn thành Update và gửi thông báo Cập nhật trạng thái thành công về SocialScreen, từ đó hiển thị trạng thái đã được lưu trữ chính thức cho người dùng.



Hình 16 Biểu đồ tuần tự xem thông báo từ trường

Sequence Diagram: Trả lời Câu hỏi

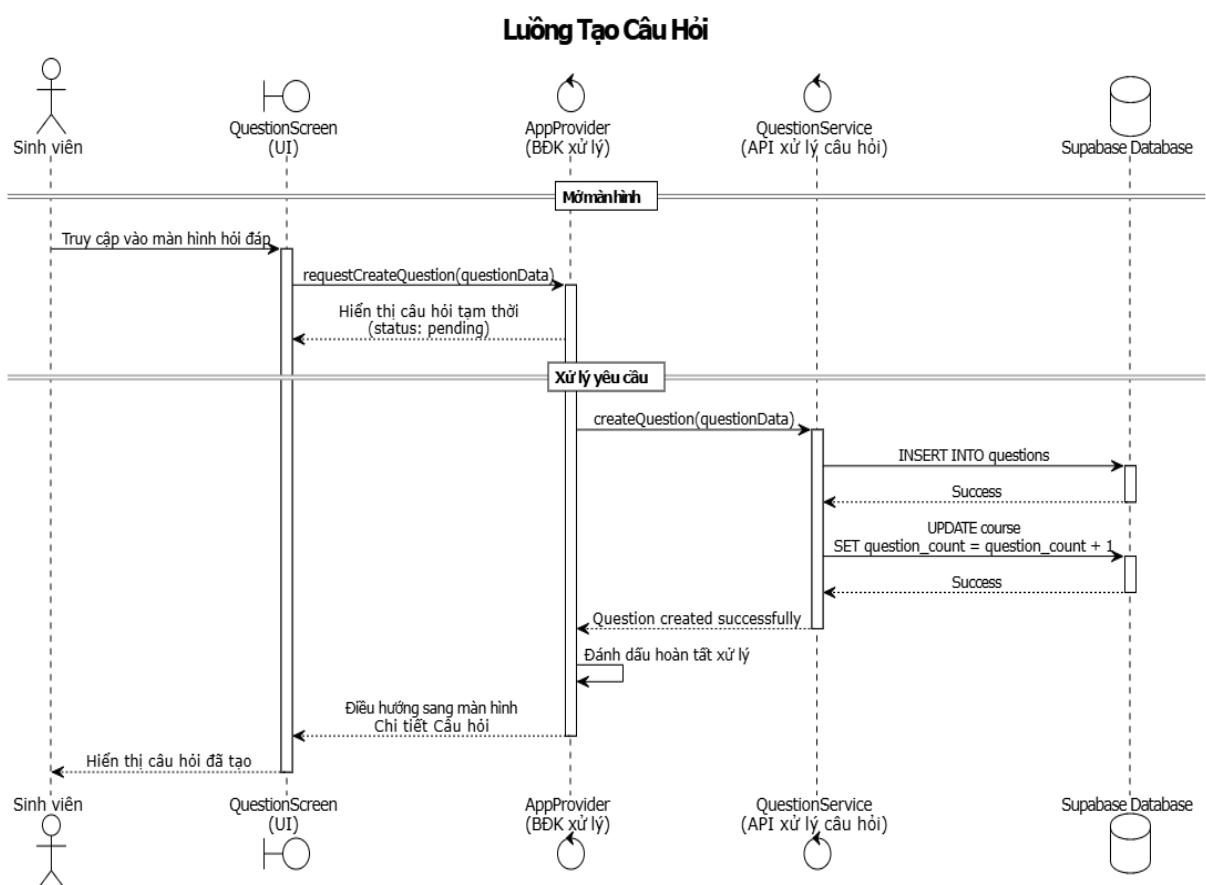
Khi người dùng (Sinh viên) Truy cập vào màn hình câu hỏi (QuestionDetailScreen) và gửi câu trả lời, QuestionDetailScreen sẽ gửi yêu cầu Thêm Trả lời đến AppProvider và đồng thời Hiển thị Trả lời ngay lập tức (đang chờ) trên giao diện. AppProvider tiếp nhận yêu cầu và gọi Yêu cầu Xử lý Thêm Trả lời trong QuestionService. QuestionService thực hiện lưu trữ vào SupabaseDB bằng cách Lưu Trả lời mới và Cập nhật số lượng Trả lời Câu hỏi liên quan. Sau khi hoàn tất các thao tác database, QuestionService gửi Xác nhận Thành công về cho AppProvider. AppProvider tiến hành Hoàn thành Update nội bộ và gửi thông báo Cập nhật trạng thái thành công trở lại QuestionDetailScreen, từ đó hiển thị thông tin đã được lưu trữ chính thức cho người dùng (Sinh viên).



Hình 17 Biểu đồ tuần tự Trả lời câu hỏi

Sequence Diagram: Tạo câu hỏi

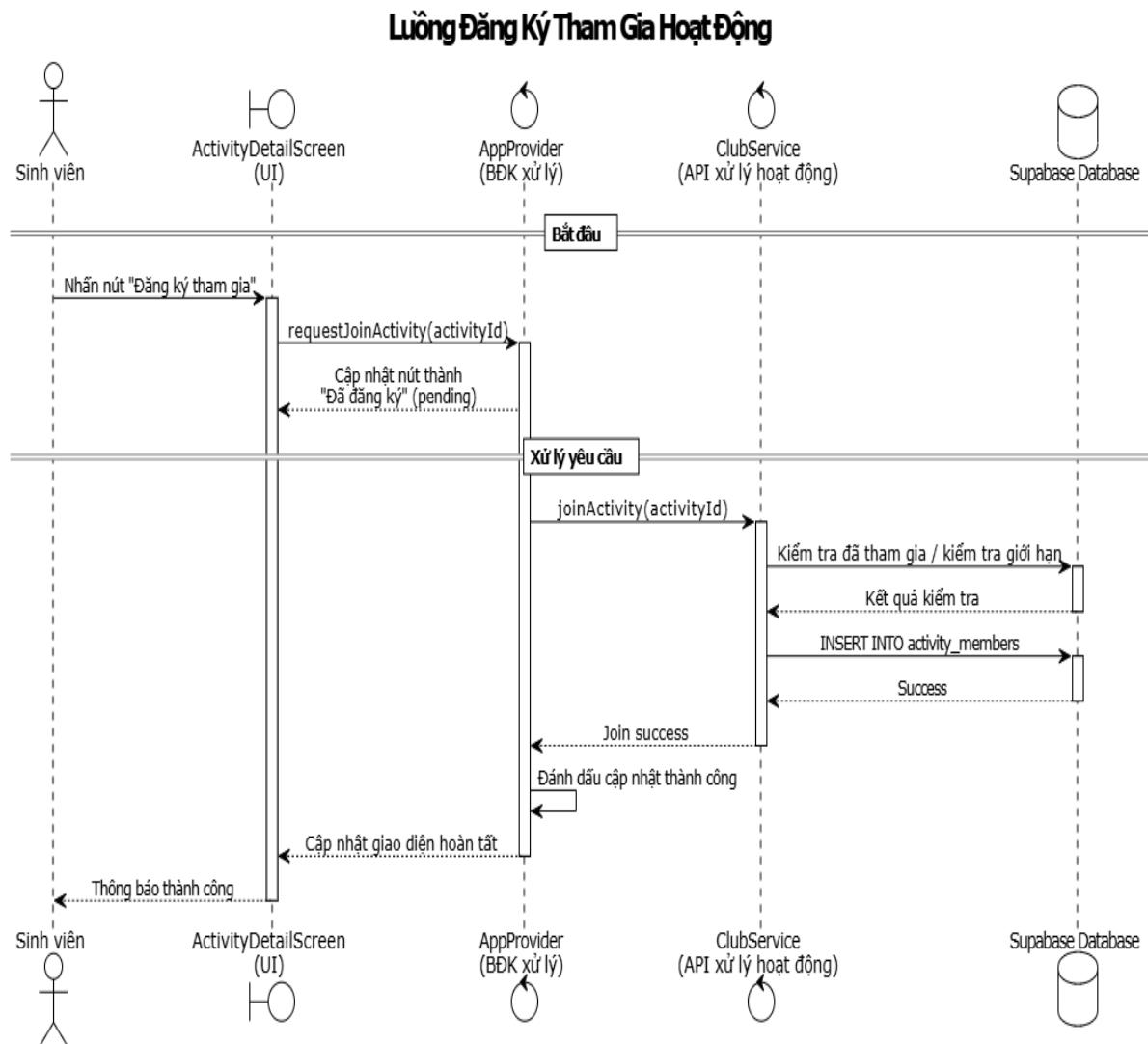
Khi người dùng (Sinh viên) muốn đặt một câu hỏi mới, họ sẽ Truy cập vào màn hình hỏi đáp (QuestionScreen). QuestionScreen sẽ Gửi yêu cầu Tạo câu hỏi đến AppProvider và đồng thời Cập nhật giao diện ngay lập tức (để hiển thị trạng thái đang xử lý hoặc chuyển đổi tạm thời). AppProvider tiếp nhận yêu cầu và gọi Yêu cầu Xử lý Tạo Câu hỏi trong QuestionService. QuestionService thực hiện lưu trữ vào SupabaseDB bằng cách Lưu Câu hỏi mới và Cập nhật tổng số câu hỏi trong khóa học hoặc chủ đề liên quan. Sau khi hoàn tất các thao tác database, QuestionService gửi Trả về Thành công về cho AppProvider. AppProvider tiến hành Hoàn thành Update nội bộ và sau đó thực hiện Điều hướng đến trang chi tiết Câu hỏi trên QuestionScreen, cuối cùng Hiển thị trạng thái đã được lưu trữ chính thức cho người dùng.



Hình 18 Biểu đồ tuần tự Tạo câu hỏi

Sequence Diagram: Đăng ký tham gia Hoạt động CLB

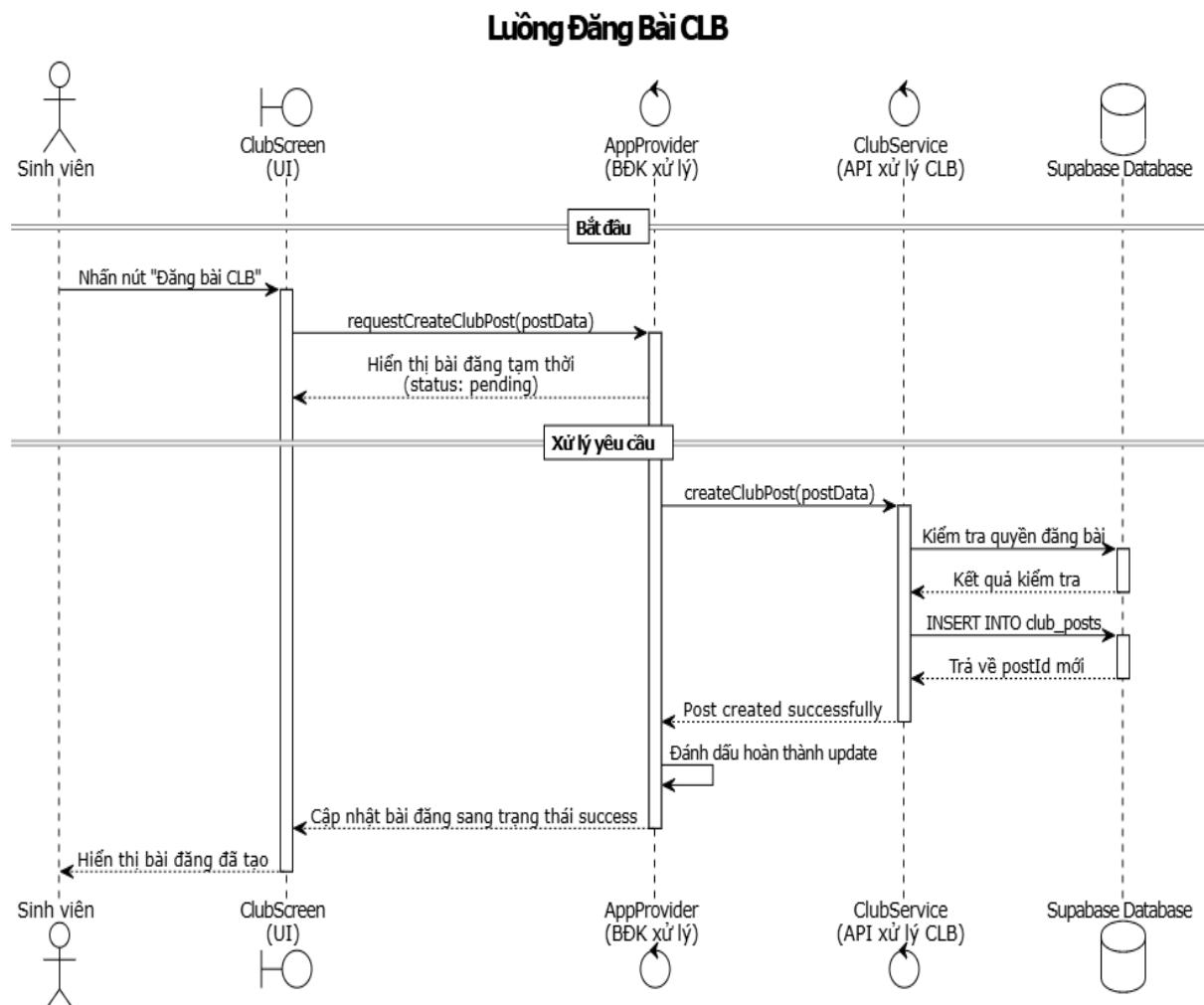
Khi người dùng (Sinh viên) muốn đăng ký tham gia một hoạt động câu lạc bộ, họ sẽ Nhấn nút Đăng ký tham gia trên ActivityDetailScreen. Màn hình này sẽ Gửi yêu cầu Tham gia Hoạt động đến AppProvider và đồng thời cập nhật giao diện bằng cách làm cho Nút chuyển sang "Đã đăng ký" ngay lập tức. AppProvider tiếp nhận yêu cầu và gọi Yêu cầu Xử lý Tham gia Hoạt động trong ClubService. ClubService có trách nhiệm tương tác với SupabaseDB, đầu tiên Kiểm tra đã tham gia/giới hạn để đảm bảo tính hợp lệ. Sau khi nhận được Kết quả kiểm tra, nếu hợp lệ, ClubService sẽ Lưu bản ghi tham gia vào database. Sau đó, ClubService gửi Xác nhận Thành công về cho AppProvider. AppProvider tiến hành Hoàn thành Update nội bộ và gửi thông báo Cập nhật Giao diện trở lại ActivityDetailScreen, cuối cùng Hiển thị trạng thái đăng ký thành công chính thức cho người dùng (Sinh viên).



Hình 19 Biểu đồ tuần tự Đăng ký tham gia Hoạt động CLB

Sequence Diagram: Đăng bài trong Câu lạc bộ

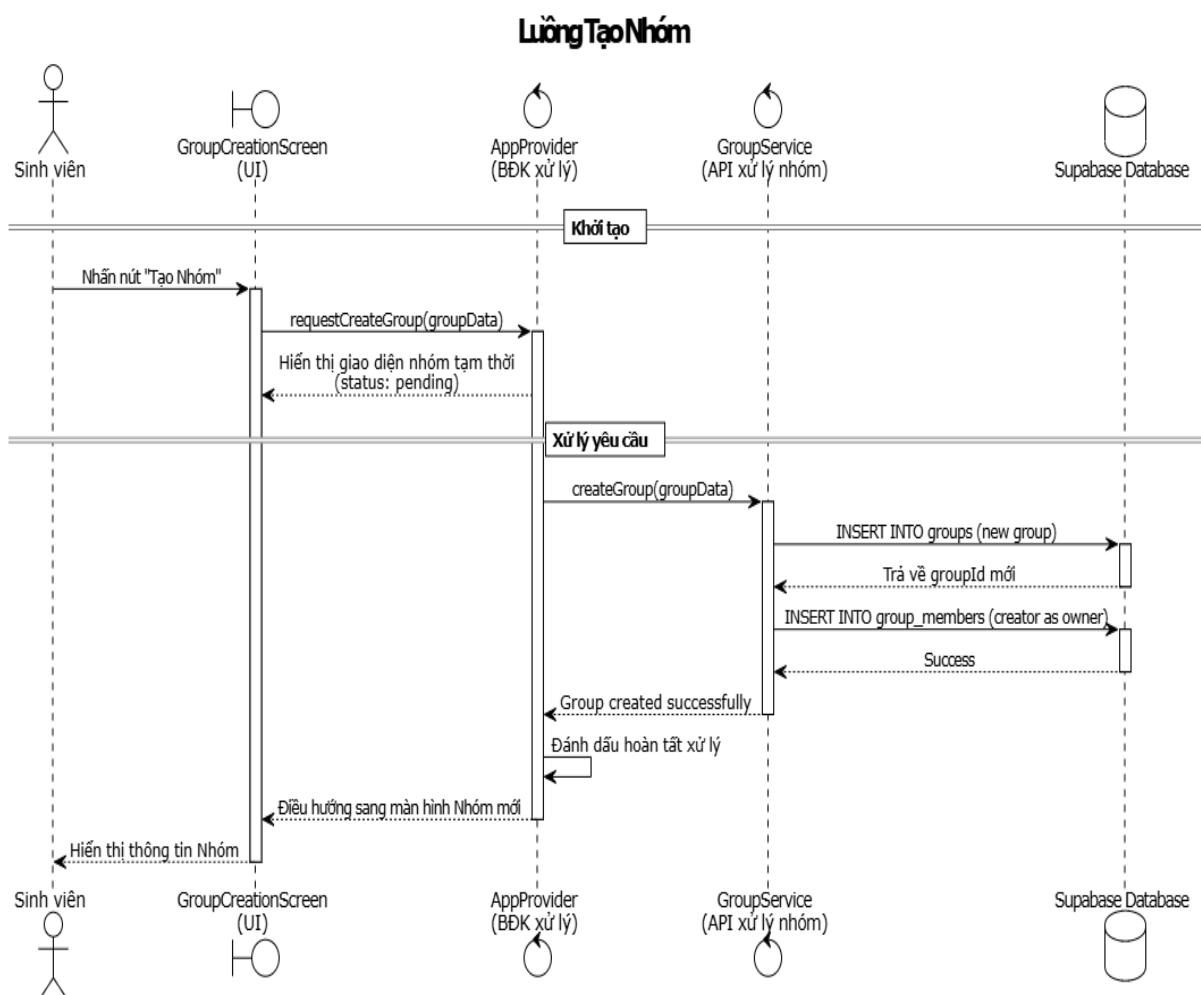
Khi người dùng (Sinh viên) muốn đăng một bài viết mới trong câu lạc bộ, họ sẽ Nhấn nút Đăng bài CLB trên ClubScreen. Màn hình này sẽ Gửi yêu cầu Tạo Bài đăng đến AppProvider và đồng thời Hiển thị Bài đăng ngay lập tức (ở trạng thái đang chờ xử lý) trên giao diện. AppProvider tiếp nhận yêu cầu và gọi Yêu cầu Xử lý Tạo Bài đăng trong ClubService. ClubService có trách nhiệm tương tác với SupabaseDB và trước hết sẽ Kiểm tra quyền đăng bài của người dùng. Sau khi nhận được Kết quả kiểm tra, nếu người dùng có quyền, ClubService sẽ Lưu Bài đăng CLB vào database và Trả về ID bài đăng mới. Sau đó, ClubService gửi Trả về Thành công về cho AppProvider. AppProvider tiến hành Hoàn thành Update nội bộ và gửi thông báo Cập nhật trạng thái thành công trở lại ClubScreen, cuối cùng Hiển thị bài đăng đã được lưu trữ chính thức cho người dùng (Sinh viên).



Hình 20 Biểu đồ tuần tự Đăng bài trong CLB

Sequence Diagram: Tạo Nhóm học tập

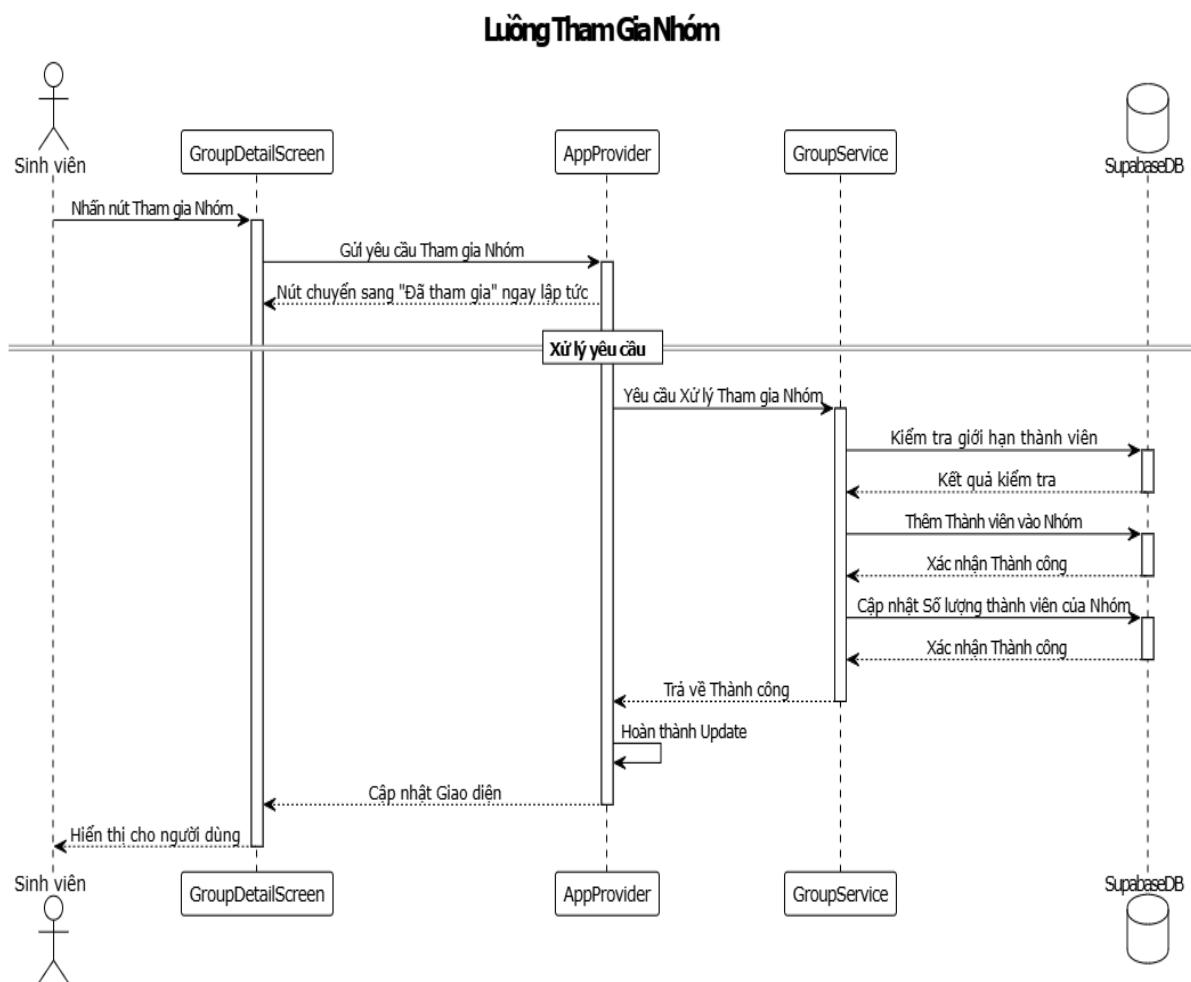
Khi người dùng (Sinh viên) muốn tạo một nhóm học tập mới, họ sẽ Nhấn nút Tạo Nhóm trên GroupCreationScreen. Màn hình này sẽ Gửi yêu cầu Tạo Nhóm đến AppProvider và đồng thời Cập nhật Giao diện ngay lập tức để phản hồi hành động của người dùng. AppProvider tiếp nhận yêu cầu và gọi Yêu cầu Xử lý Tạo Nhóm trong GroupService. GroupService có trách nhiệm tương tác với SupabaseDB để Lưu thông tin Nhóm mới và sau đó Trả về ID Nhóm vừa tạo. Tiếp theo, dịch vụ sẽ Thêm người tạo vào danh sách Thành viên trong database. Sau khi hoàn tất các thao tác lưu trữ, GroupService gửi Xác nhận Thành công về cho AppProvider. AppProvider tiến hành Hoàn thành Update nội bộ và sau đó thực hiện Điều hướng đến Nhóm mới trên GroupCreationScreen, cuối cùng Hiển thị thông tin nhóm đã được tạo chính thức cho người dùng (Sinh viên).



Hình 21 Biểu đồ tuần tự Tạo nhóm học tập

Sequence Diagram: Tham gia Nhóm học tập

Khi người dùng (Sinh viên) muốn tham gia một nhóm học tập đã có sẵn, họ sẽ Nhấn nút Tham gia Nhóm trên GroupDetailScreen. Màn hình này sẽ Gửi yêu cầu Tham gia Nhóm đến AppProvider và đồng thời cập nhật giao diện bằng cách làm cho Nút chuyển sang "Đã tham gia" ngay lập tức. AppProvider tiếp nhận yêu cầu và gọi Yêu cầu Xử lý Tham gia Nhóm trong GroupService. GroupService có trách nhiệm tương tác với SupabaseDB, đầu tiên Kiểm tra giới hạn thành viên để đảm bảo nhóm còn chỗ. Sau khi nhận được Kết quả kiểm tra, nếu hợp lệ, GroupService sẽ Thêm Thành viên vào Nhóm và Cập nhật Số lượng thành viên của Nhóm trong database. Sau đó, GroupService gửi Xác nhận Thành công về cho AppProvider. AppProvider tiến hành Hoàn thành Update nội bộ và gửi thông báo Cập nhật Giao diện trở lại GroupDetailScreen, cuối cùng Hiển thị trạng thái tham gia thành công chính thức cho người dùng (Sinh viên).

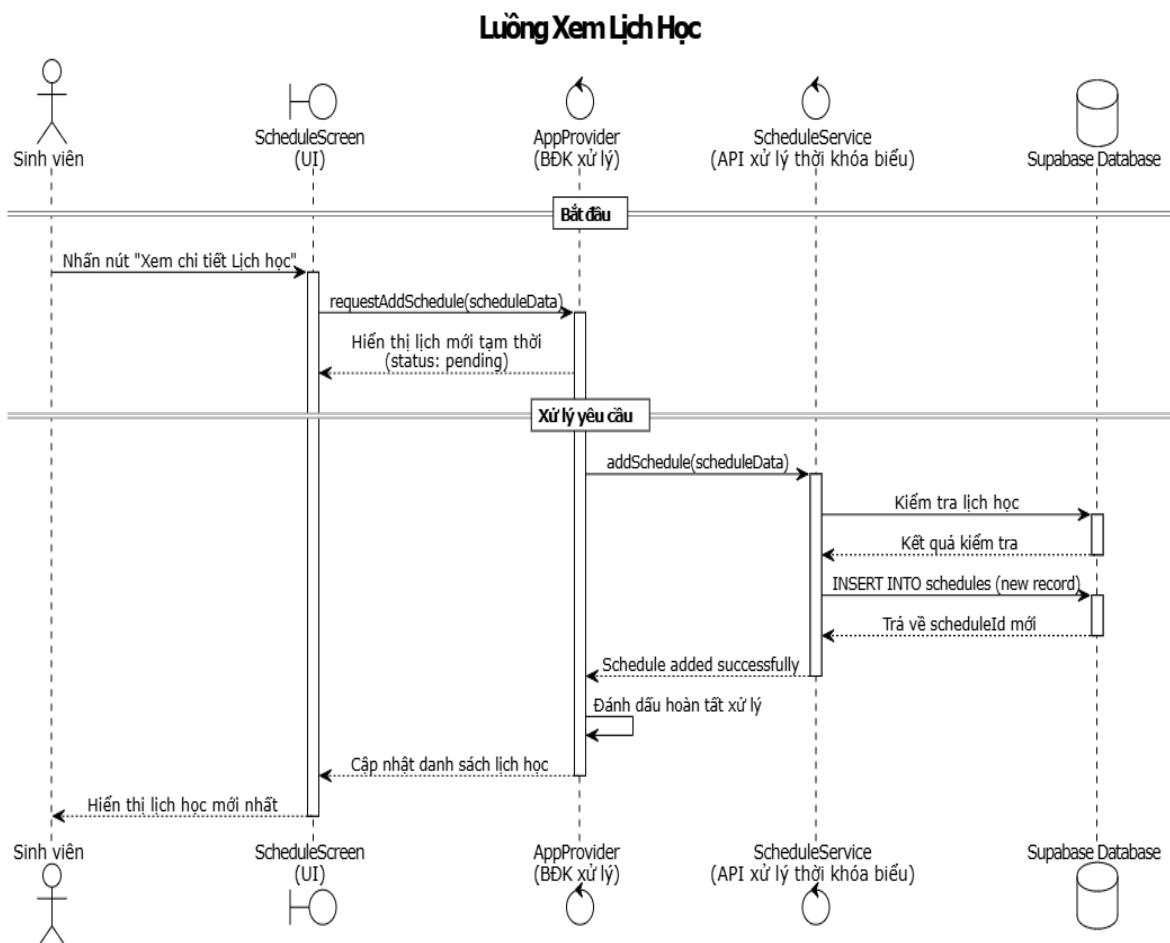


Hình 22 Biểu đồ tuần tự Tham gia nhóm học tập

Sequence Diagram: Xem Lịch học Cá nhân

Khi người dùng (Sinh viên) muốn Xem Lịch Học, họ sẽ Nhấn nút Xem chi tiết Lịch học trên ScheduleScreen. Màn hình này sẽ gửi yêu cầu requestAddSchedule(scheduleData) (thực chất là yêu cầu cập nhật danh sách) đến AppProvider và đồng thời Hiển thị lịch học mới tạm thời (status: pending) (để hiển thị trạng thái đang tải dữ liệu) để phản hồi hành động của người dùng. AppProvider tiếp nhận yêu cầu và gọi Yêu cầu Xử lý Cập nhật Lịch học cá nhân mới (addSchedule(scheduleData)) trong ScheduleService.

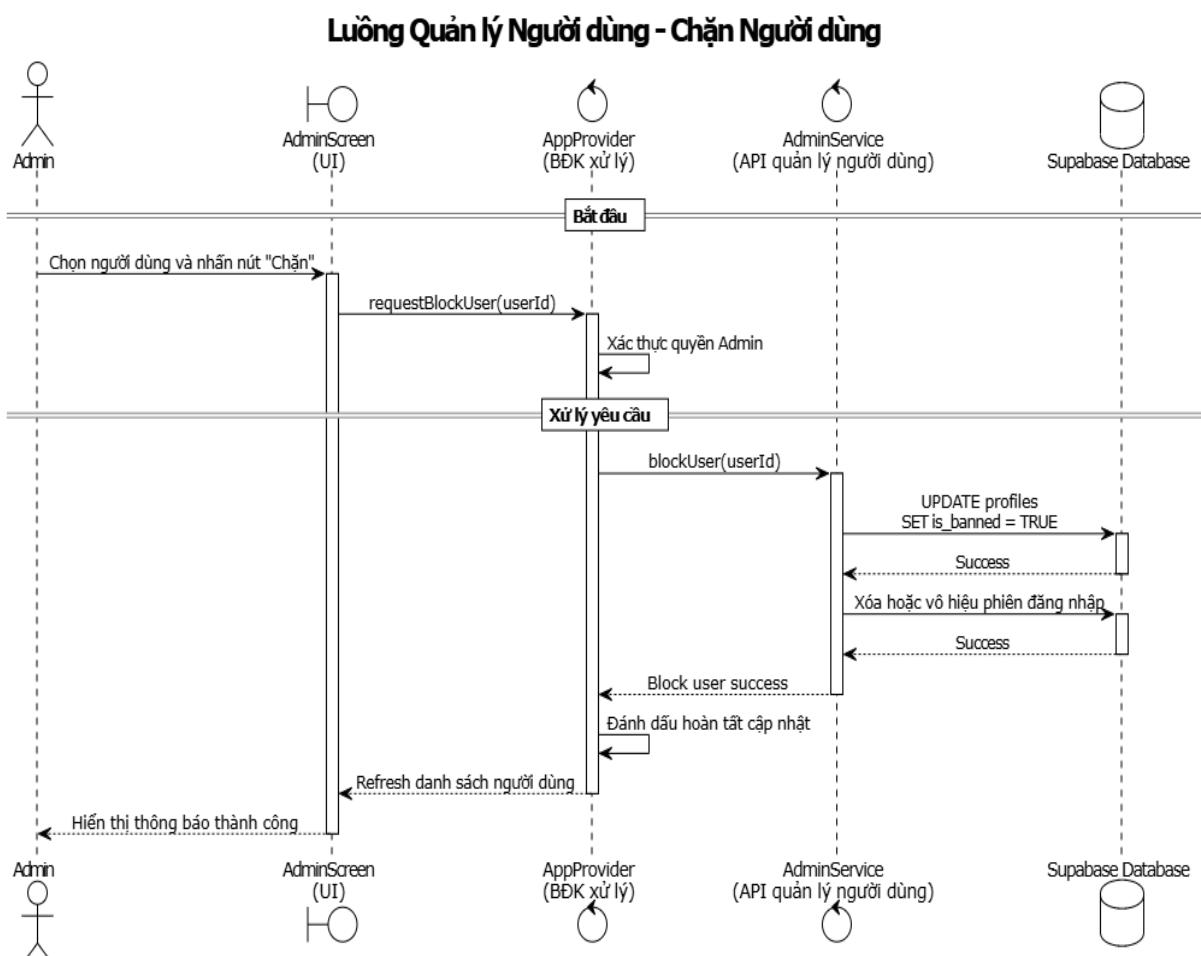
ScheduleService có trách nhiệm tương tác với Supabase Database, đầu tiên Kiểm tra lịch học (kiểm tra tính hợp lệ của dữ liệu). Sau khi nhận được Kết quả kiểm tra, ScheduleService sẽ thực hiện truy vấn SELECT lịch học mới nhất (thay vì INSERT) và Trả về ID mới của giao dịch. Sau đó, ScheduleService gửi Schedule added successfully (Thông báo lấy dữ liệu thành công) về cho AppProvider. AppProvider tiến hành Đánh dấu hoàn tất xử lý và gửi thông báo Cập nhật danh sách lịch trở lại ScheduleScreen, cuối cùng Hiển thị lịch học mới nhất (danh sách lịch học đã được truy vấn thành công) cho người dùng (Sinh viên).



Hình 23 Biểu đồ tuần tự Xem lịch học cá nhân

Sequence Diagram: Quản lý Người dùng

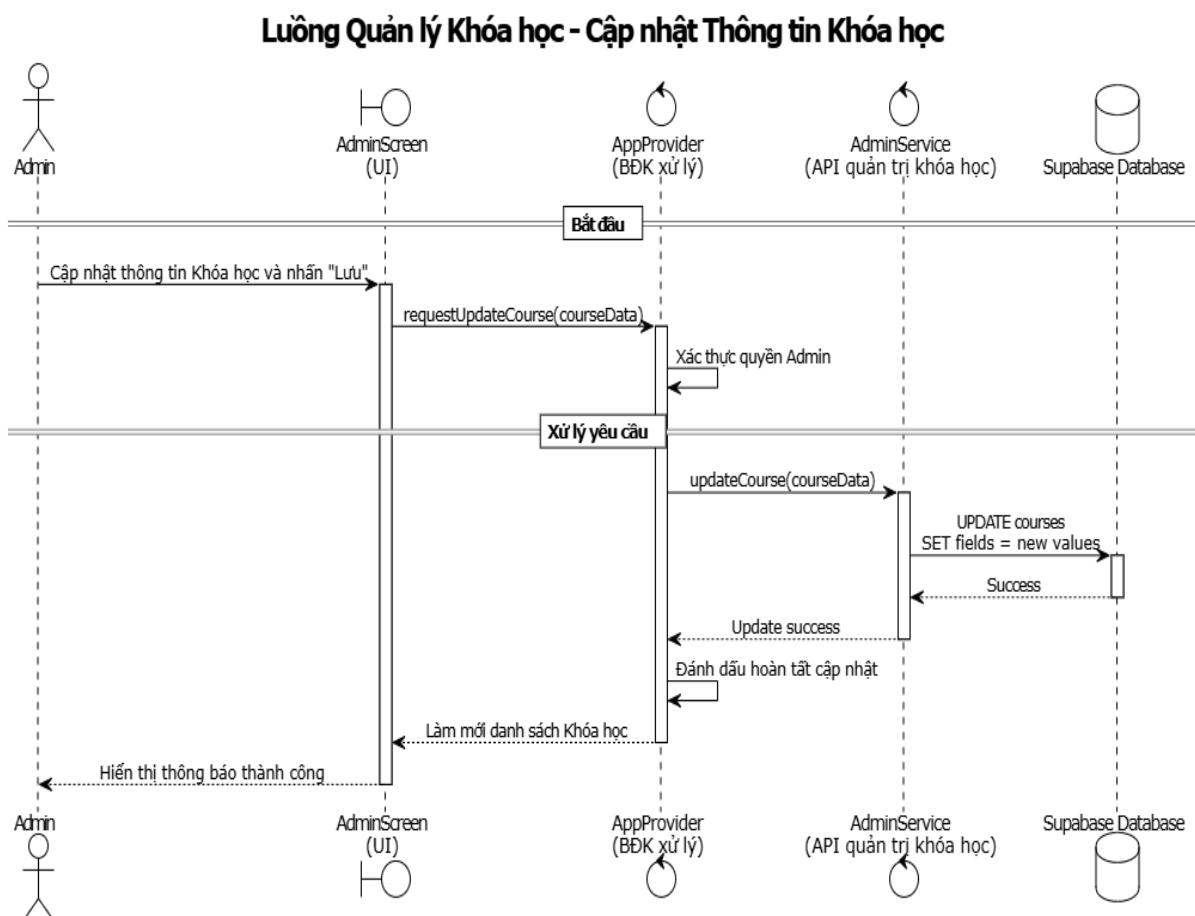
Khi người quản trị (Admin) muốn chặn một người dùng, họ sẽ Chọn người dùng và Nhấn nút Chặn trên AdminScreen. Màn hình này sẽ Gửi yêu cầu Cập nhật trạng thái người dùng đến AppProvider. AppProvider thực hiện Xác thực Admin và sau đó gọi Yêu cầu Xử lý Chặn người dùng trong AdminService. AdminService chịu trách nhiệm tương tác với SupabaseDB để UPDATE cột trạng thái người dùng (ví dụ: profiles.is_banned = TRUE) và Cập nhật thông tin phiên đăng nhập (tùy chọn) để đảm bảo người dùng bị chặn không thể tiếp tục truy cập. Sau khi hoàn tất các thao tác database, AdminService gửi Xác nhận Thành công về cho AppProvider. AppProvider tiến hành Hoàn thành Update nội bộ và gửi thông báo Cập nhật danh sách người dùng trở lại AdminScreen, cuối cùng Hiển thị thông báo thành công cho người quản trị (Admin).



Hình 24 Biểu đồ tuần tự Quản lý Người dùng

Sequence Diagram: Quản lý Khóa học

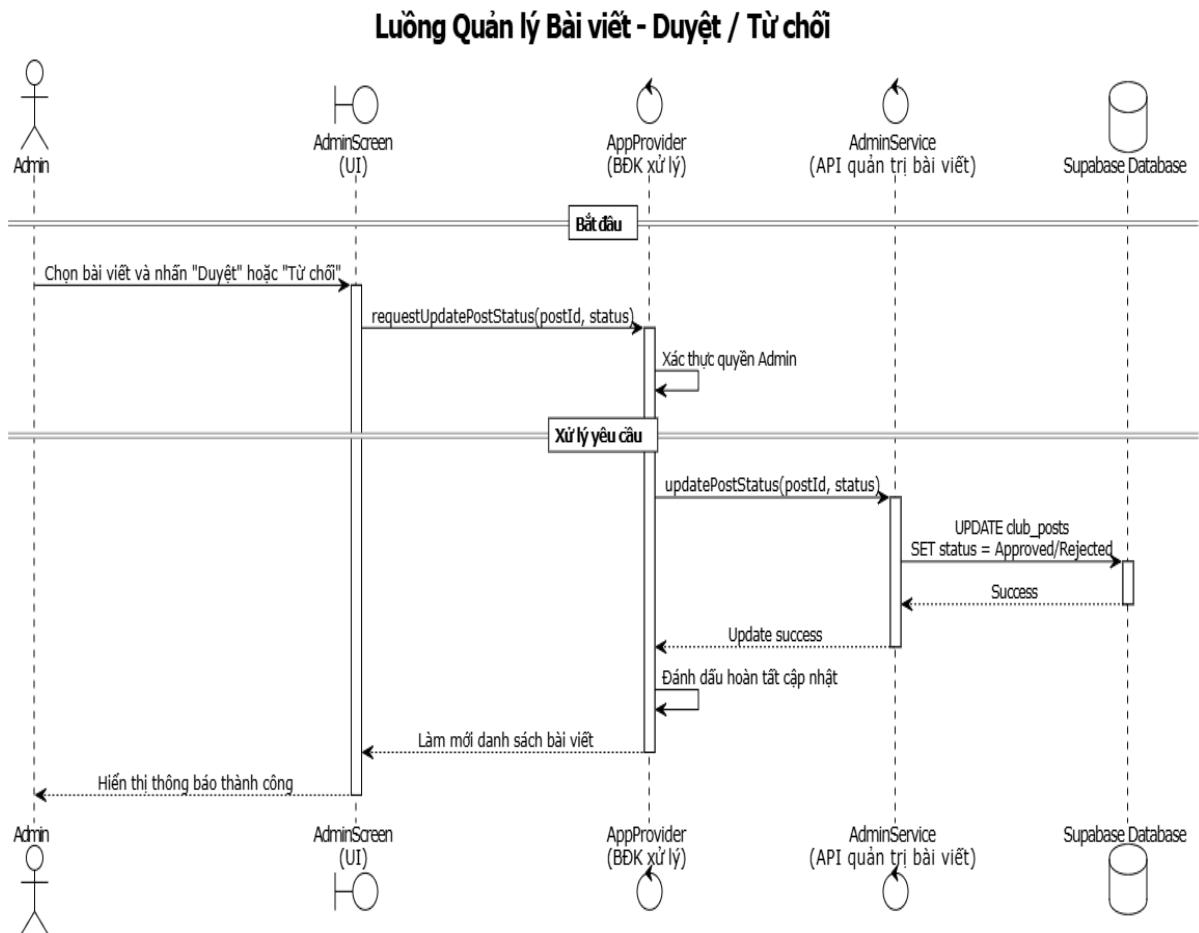
Khi người quản trị (Admin) muốn cập nhật thông tin một khóa học, họ sẽ Cập nhật thông tin Khóa học và Nhấn Lưu trên AdminScreen. Màn hình này sẽ Gửi yêu cầu Cập nhật Khóa học đến AppProvider. AppProvider thực hiện Xác thực Admin để đảm bảo quyền hạn, sau đó gọi Yêu cầu Xử lý Cập nhật Khóa học trong AdminService. AdminService chịu trách nhiệm tương tác với SupabaseDB để thực hiện UPDATE thông tin Khóa học (courses) trong database. Sau khi hoàn tất thao tác, AdminService gửi Xác nhận Thành công về cho AppProvider. AppProvider tiến hành Hoàn thành Update nội bộ, sau đó gửi thông báo Cập nhật danh sách Khóa học trở lại AdminScreen, cuối cùng Hiển thị thông báo thành công cho người quản trị (Admin).



Hình 25 Biểu đồ tuần tự Quản lý Khóa học

Sequence Diagram: Duyệt/Tù chối Bài viết Cộng đồng

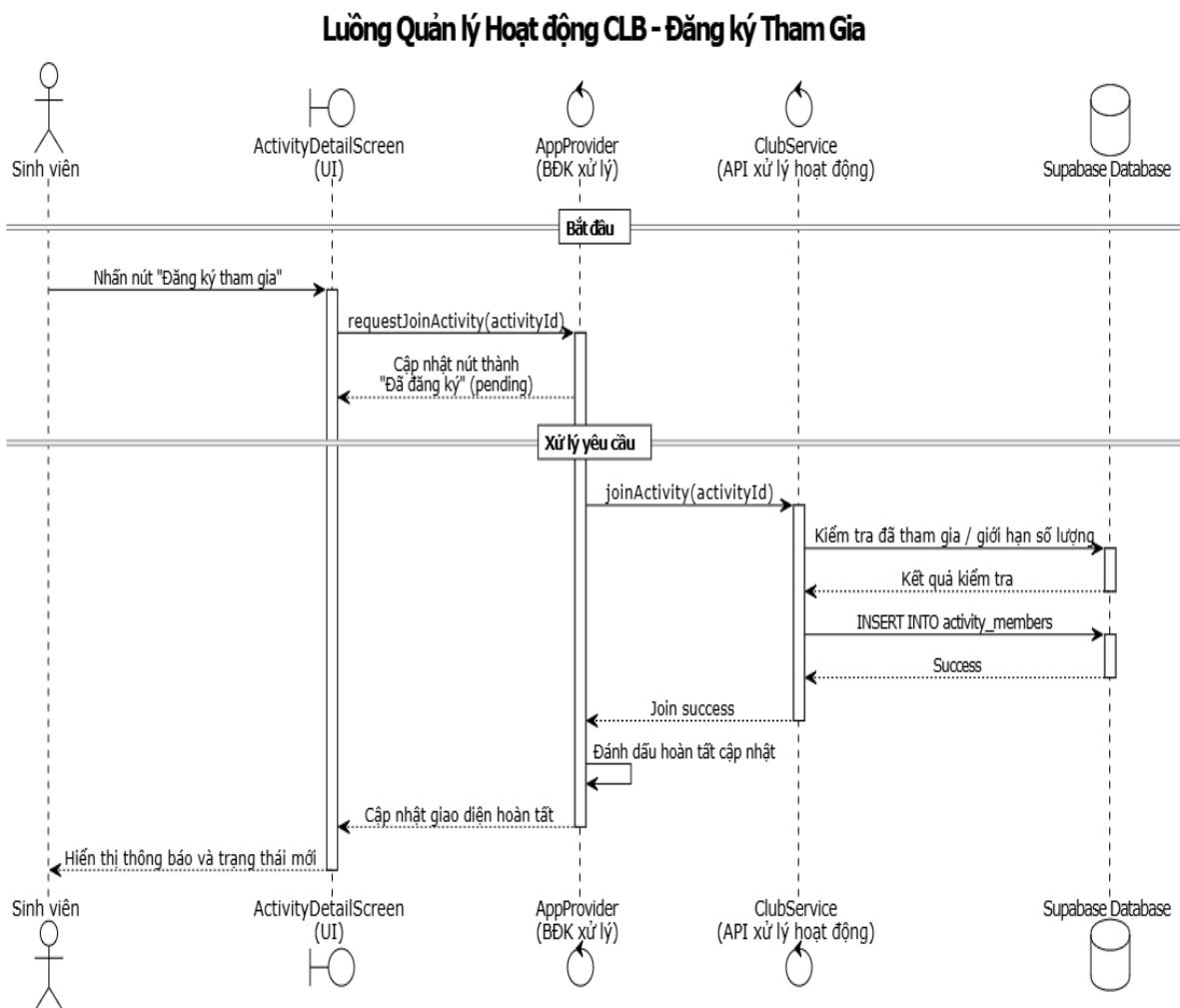
Khi người quản trị (Admin) muốn duyệt hoặc từ chối một bài viết cộng đồng, họ sẽ Chọn Bài viết và Nhấn nút Duyệt/Tù chối trên AdminScreen. Màn hình này sẽ Gửi yêu cầu Cập nhật trạng thái Bài viết đến AppProvider. AppProvider thực hiện Xác thực Admin để đảm bảo quyền hạn, sau đó gọi Yêu cầu Xử lý Cập nhật trạng thái trong AdminService. AdminService chịu trách nhiệm tương tác với SupabaseDB để thực hiện UPDATE trạng thái bài viết (sang trạng thái Duyệt Approved hoặc Từ chối Rejected) trong database. Sau khi hoàn tất thao tác, AdminService gửi Xác nhận Thành công về cho AppProvider. AppProvider tiến hành Hoàn thành Update nội bộ, sau đó gửi thông báo Cập nhật danh sách Bài viết trở lại AdminScreen, cuối cùng Hiển thị thông báo thành công cho người quản trị (Admin).



Hình 26 Biểu đồ luân tự Duyệt/Tù chối Bài viết Cộng đồng

Sequence Diagram: Quản lý Hoạt động Câu lạc bộ

Khi người quản trị (Admin) muốn tạo một hoạt động câu lạc bộ mới, họ sẽ Nhập thông tin Hoạt động và Nhấn Tạo trên AdminScreen. Màn hình này sẽ Gửi yêu cầu Tạo Hoạt động CLB mới đến AppProvider. AppProvider thực hiện Xác thực Admin để đảm bảo quyền hạn, sau đó gọi Yêu cầu Xử lý Tạo Hoạt động trong AdminService. AdminService chịu trách nhiệm tương tác với SupabaseDB để INSERT bản ghi hoạt động (club_activities) vào database và Trả về ID Hoạt động mới. Sau khi hoàn tất thao tác, AdminService gửi Trả về Thành công về cho AppProvider. AppProvider tiến hành Hoàn thành Update nội bộ, sau đó thực hiện Điều hướng đến trang chi tiết Hoạt động trên AdminScreen, cuối cùng Hiển thị thông báo thành công cho người quản trị (Admin).



Hình 27 Biểu đồ tuần tự Hoạt động CLB

2.4.5 Sơ đồ Database (DB Diagram)

Cơ sở dữ liệu trong sơ đồ được thiết kế nhằm hỗ trợ một nền tảng dành cho sinh viên, tích hợp nhiều chức năng trong cùng một hệ thống, bao gồm:

- Mạng xã hội nội bộ (bài đăng, bình luận, lượt thích)
- Hệ thống hỏi đáp học thuật
- Quản lý câu lạc bộ sinh viên
- Nhóm học tập
- Sự kiện sinh viên
- Thông báo nhà trường
- Lịch học – phòng học

CSDL được tổ chức theo mô hình quan hệ, đảm bảo:

- Tính toàn vẹn (integrity)
- Tính mở rộng (scalability)
- Dễ dàng tích hợp vào ứng dụng web/mobile
- Hỗ trợ truy vấn hiệu quả

Phân tích chi tiết từng bảng

1. Bảng users – Bảng trung tâm của hệ thống

Bảng users lưu thông tin toàn bộ người dùng của hệ thống, bao gồm sinh viên, giảng viên và quản trị viên.

Các thuộc tính quan trọng:

- email, password: thông tin đăng nhập
- name, phone, avatar_url: thông tin cá nhân
- interests: sở thích dùng để gợi ý nhóm/clb
- created_at, updated_at: thời gian tạo & chỉnh sửa

Bảng này liên kết trực tiếp đến hầu hết các bảng khác (posts, comments, clubs, events, study_groups...).

2. Bảng posts – Hệ thống bài đăng

Bảng này cho phép người dùng đăng bài viết trên nền tảng.

Các trường chính:

- user_id: người đăng bài (Foreign Key tới users)
- content: nội dung
- image_url: ảnh kèm theo

- likes_count, comments_count, shares_count: số liệu tương tác

3. Bảng comments – Bình luận

Dùng để lưu tất cả bình luận trên bài viết.

- post_id: bài viết thuộc về (FK tới posts)
- user_id: ai bình luận (FK tới users)
- parent_id: hỗ trợ bình luận lồng nhau (FK tự tham chiếu tới comments.id)

4. Bảng post_likes – Lượt thích

- Cho phép người dùng thích bài viết.
- Mối quan hệ: Nhiều-nhiều giữa users và posts (chỉ chứa user_id và post_id).

5. Bảng questions – Hệ thống hỏi đáp

Bảng này cho phép sinh viên đặt câu hỏi học thuật.

Trường chính:

- user_id: người hỏi (FK tới users)
- course: môn học
- replies_count: số lượng câu trả lời
- solved: trạng thái câu hỏi đã được giải quyết chưa (Boolean)

6. Bảng question_replies – Trả lời câu hỏi

- question_id: câu hỏi thuộc về (FK tới questions)
- user_id: người trả lời (FK tới users)
- is_solution: đánh dấu câu trả lời đúng nhất/tốt nhất (Boolean)

7. Bảng clubs – Quản lý câu lạc bộ

- name, description
- image_url
- members_count: số lượng thành viên
- creator_id: người tạo CLB (FK tới users)

8. Bảng club_members – Thành viên câu lạc bộ

- Thể hiện quan hệ Nhiều–Nhiều (N:N) giữa users và clubs. (chứa user_id và club_id)

9. Bảng study_groups – Nhóm học tập

Hỗ trợ sinh viên lập nhóm học theo môn.

Trường quan trọng:

- creator_id (FK tới users)
- course, description
- meet_time: thời gian gặp mặt/học nhóm
- max_members, members_count

10. Bảng study_group_members – Thành viên nhóm học

- Mỗi dòng là một thành viên của một nhóm học.
- Quan hệ N-N giữa users và study_groups. (chứa user_id và study_group_id)

11. Bảng events – Sự kiện sinh viên

- Lưu tất cả sự kiện: hội thảo, hoạt động ngoại khóa...
- organizer_id: người tổ chức (FK tới users)
- location: địa điểm (tên hoặc FK tới locations)
- event_time: thời gian diễn ra
- max_attendees, attendees_count

12. Bảng event_attendees – Người tham gia sự kiện

- Mỗi dòng lưu thông tin một người tham dự.
- Quan hệ Nhiều-nhiều (N:N) giữa users và events. (chứa user_id và event_id)

13. Bảng announcements – Thông báo

Cho phép admin/giảng viên đăng thông báo:

- title, content
- priority (ưu tiên)
- target_audience: đối tượng nhận thông báo
- created_by (FK tới users)

14. Bảng class_schedules – Lịch học

Bảng mô tả thời khóa biểu của từng người dùng:

- day_of_week
- start_time, end_time
- subject
- instructor
- room (hoặc FK tới locations)
- color (để hiển thị lịch trên giao diện)

- user_id (FK tới users) - cho biết lịch học của ai

15. Bảng locations – Phòng học / tòa nhà

Mô tả chi tiết vị trí trong trường:

- building, floor
- coordinate (để hiển thị trên bản đồ)
- type: phòng học, phòng lab, hội trường...

16. Bảng club_posts – Bài đăng Câu lạc bộ

- club_id: CLB tạo bài đăng (FK tới clubs)
- user_id: người đăng (FK tới users)
- content, attachments: nội dung và tệp đính kèm.
- Chức năng tương tự bảng posts nhưng trong phạm vi CLB.

17. Bảng club_post_comments – Bình luận Bài đăng CLB

- club_post_id: bài đăng CLB thuộc về (FK tới club_posts)
- user_id: ai bình luận (FK tới users)
- Dùng để lưu bình luận cho bài đăng trong CLB.

18. Bảng club_activities – Hoạt động Câu lạc bộ

- club_id: CLB tổ chức hoạt động (FK tới clubs)
- name, description
- activity_time, location
- Quản lý các sự kiện/hoạt động nhỏ hơn của riêng CLB.

19. Bảng club_activity_participants – Người tham gia Hoạt động CLB

- activity_id: hoạt động CLB tham gia (FK tới club_activities)
- user_id: ai tham gia (FK tới users)
- Quan hệ N:N giữa users và club_activities.

20. Bảng sessions – Phiên học / Tiết học

- user_id: người tạo phiên học (FK tới users)
- instructor: có thể là tên giảng viên hoặc người hướng dẫn
- members: có thể lưu trữ JSON ID của thành viên tham gia (nếu là học nhóm)

Bảng này có vẻ dùng để quản lý các phiên học/lớp học cá nhân hoặc nhóm nhỏ (tương tự class_schedules nhưng có thể linh hoạt hơn).

Quan hệ giữa các bảng

1. Quan hệ 1 – N (Một-Nhiều)

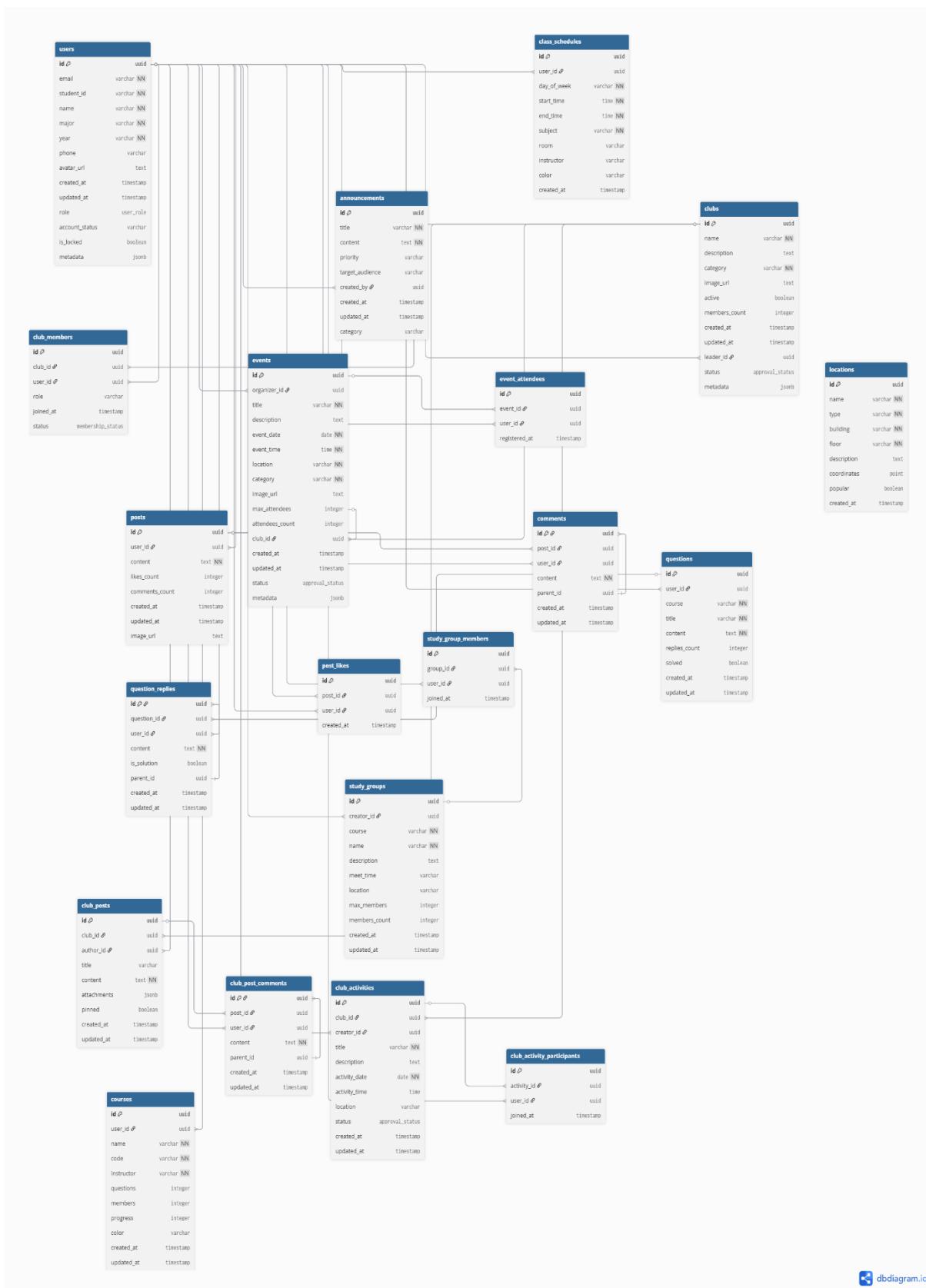
Quan hệ	Bảng 1 (Chính)	Bảng N (Phụ)	Ý nghĩa
users – posts	users	posts	1 người tạo nhiều bài đăng.
users – comments	users	comments	1 người tạo nhiều bình luận.
users – questions	users	questions	1 người đặt nhiều câu hỏi.
users – announcements	users	announcements	1 admin/giảng viên tạo nhiều thông báo.
users – class_schedules	users	class_schedules	1 người có nhiều mục trong lịch học.
users – study_groups	users	study_groups	1 người tạo nhiều nhóm học.
events – event_attendees	events	event_attendees	1 sự kiện có nhiều người tham gia.
clubs – club_members	clubs	club_members	1 CLB có nhiều thành viên.
study_groups – study_group_members	study_groups	study_group_members	1 nhóm học có nhiều thành viên.
clubs – club_posts	clubs	club_posts	1 CLB có nhiều bài đăng nội bộ.
club_posts – club_post_comments	club_posts	club_post_comments	1 bài đăng CLB có nhiều bình luận.
clubs – club_activities	clubs	club_activities	1 CLB có nhiều hoạt động.
club_activities – club_activity_participants	club_activities	club_activity_participants	1 hoạt động CLB có nhiều người tham gia.

2. Quan hệ N – N (Nhiều-Nhiều) qua bảng phụ

Quan hệ	Bảng Chính 1	Bảng Chính 2	Bảng Trung Gian
Thích Bài đăng	users	posts	post_likes
Thành viên CLB	users	clubs	club_members
Thành viên Nhóm học	users	study_groups	study_group_members
Tham gia Sự kiện	users	events	event_attendees
Tham gia Hoạt động CLB	users	club_activities	club_activity_participants

3. Quan hệ phân cấp (Hierarchy)

- comments có parent_id → tạo cấu trúc bình luận dạng cây/lồng nhau (nested comments).
- study_groups và study_group_members → quản lý người dùng theo nhóm.
- questions và question_replies → mô hình hỏi – đáp (1 câu hỏi có nhiều câu trả lời).
- club_posts và club_post_comments → mô hình bài đăng và bình luận trong phạm vi CLB.

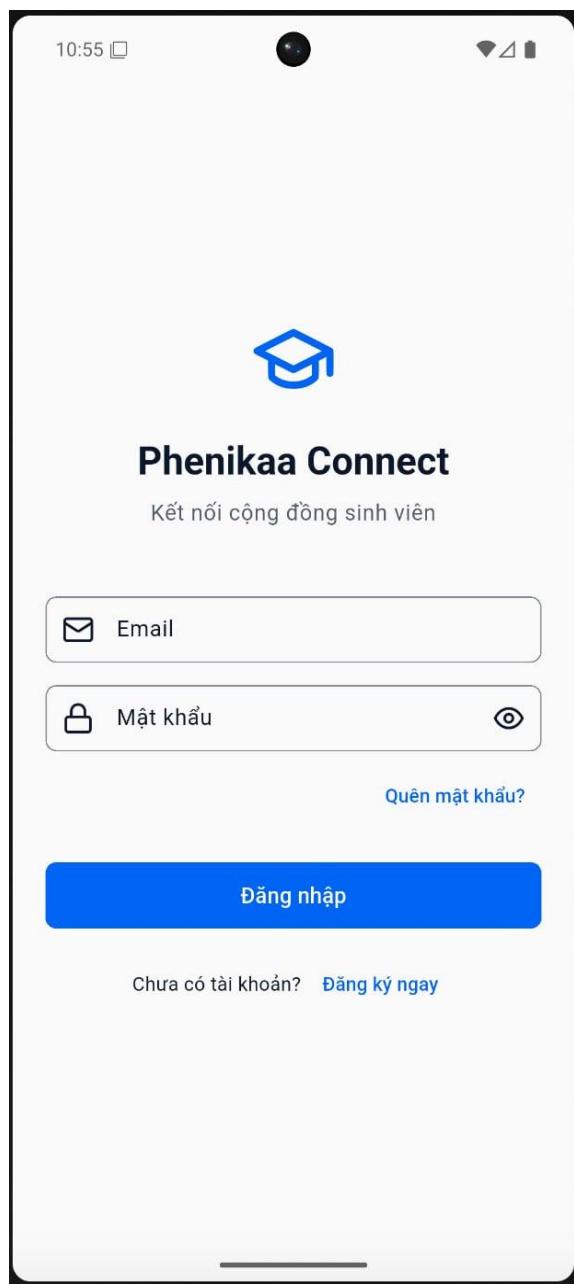


Hình 28 Sơ đồ Database (DB Diagram)

2.4.6 Các màn hình giao diện người dùng

Màn hình Đăng nhập

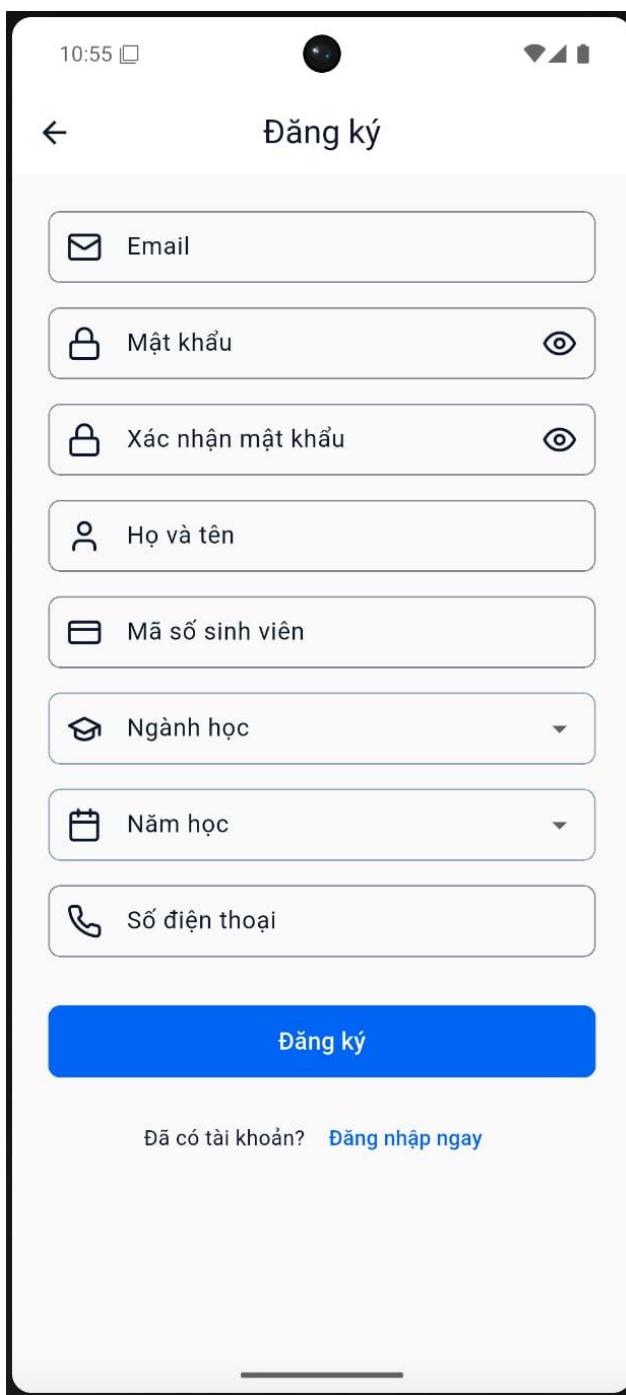
Màn hình đăng nhập là điểm đầu tiên mà người dùng gặp khi mở ứng dụng. Giao diện được thiết kế đơn giản và rõ ràng với logo Phenikaa Connect ở phía trên, tiếp theo là các trường nhập liệu cho email và mật khẩu. Người dùng có thể chọn hiển thị hoặc ẩn mật khẩu bằng biểu tượng mắt, và có liên kết "Quên mật khẩu" ở bên phải trường mật khẩu. Ở phía dưới có nút đăng nhập màu chủ đạo và liên kết chuyển đến màn hình đăng ký cho những người dùng chưa có tài khoản.



Hình 29 Màn hình đăng nhập

Màn hình Đăng ký

Màn hình đăng ký cung cấp form đầy đủ để người dùng tạo tài khoản mới. Form bao gồm các trường bắt buộc như email, mật khẩu, xác nhận mật khẩu, họ tên, mã số sinh viên, ngành học được chọn từ dropdown, năm học và số điện thoại. Giao diện được bố trí theo dạng cuộn dọc để đảm bảo tất cả các trường đều dễ dàng truy cập. Mỗi trường có biểu tượng tiền tố để dễ nhận biết và có validation để đảm bảo thông tin nhập vào đúng định dạng.



Hình 30 Màn hình đăng ký

Màn hình Trang chủ

Giao diện Trang chủ của ứng dụng Phenikaa Connect được thiết kế như một bảng điều khiển cá nhân hóa, cung cấp cho sinh viên cái nhìn tổng quan toàn diện về các thông tin học tập và hoạt động hàng ngày. Bộ cục giao diện được tổ chức theo từng khu vực rõ ràng, trực quan, giúp người dùng dễ dàng theo dõi và thao tác.

Ở phần đầu trang, hệ thống hiển thị lời giới thiệu về ứng dụng cùng thông tin nhận diện cá nhân của sinh viên. Khu vực này bao gồm tên người dùng, mã số sinh viên và ảnh đại diện, giúp tăng tính cá nhân hóa và hỗ trợ người dùng nhận biết tài khoản đang sử dụng.

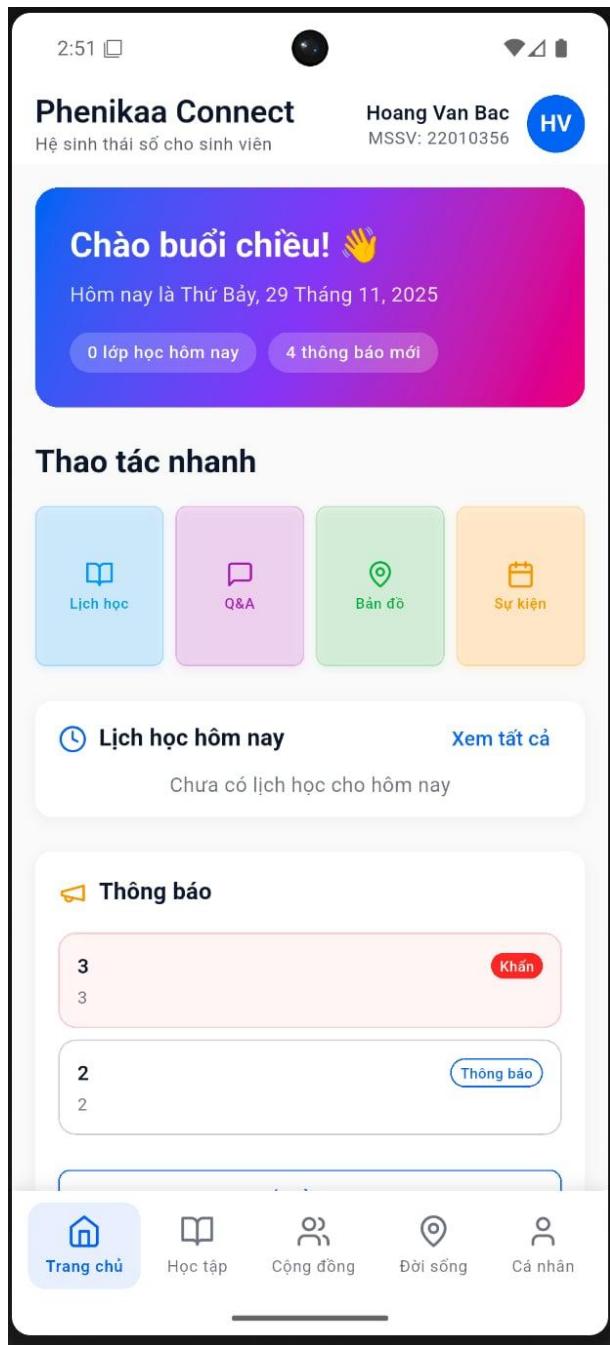
Ngay bên dưới là bảng tóm tắt trạng thái, nổi bật với màu sắc sinh động và lời chào theo thời gian thực như “Chào buổi chiều!”. Hệ thống cũng hiển thị ngày hiện tại và các thông kê quan trọng trong ngày, bao gồm số lớp học trong lịch hôm nay và số lượng thông báo mới. Những thông tin này giúp sinh viên nhanh chóng nắm bắt tình hình học tập và hoạt động cần lưu ý.

Tiếp theo là khu vực Thao tác nhanh (Quick Actions) với bốn nút chức năng lớn: Lịch học, Q&A, Bản đồ và Sự kiện. Đây là các tính năng được sinh viên sử dụng thường xuyên nhất, vì vậy giao diện được tối ưu để người dùng có thể truy cập chỉ với một chạm.

Phần nội dung chính của màn hình hiển thị Lịch học hôm nay và Thông báo. Mục Lịch học hôm nay cho biết chi tiết lịch trình trong ngày; trường hợp không có lịch học, hệ thống thông báo rõ ràng “Chưa có lịch học cho hôm nay”. Người dùng cũng có thể chọn “Xem tất cả” để chuyển đến giao diện lịch học đầy đủ. Phần Thông báo liệt kê các tin tức mới nhất, được phân loại bằng nhãn màu để giúp sinh viên nhận biết mức độ ưu tiên, chẳng hạn như nhãn Khẩn (màu đỏ) cho các thông báo quan trọng cần chú ý ngay lập tức và nhãn Thông báo (màu xanh dương) cho các tin chung.

Cuối cùng, thanh điều hướng chính được đặt cố định ở cuối màn hình, cho phép sinh viên di chuyển nhanh giữa các phân hệ của hệ sinh thái số. Thanh điều hướng gồm năm mục: Trang chủ, Học tập, Cộng đồng, Đời sống và Cá nhân, trong đó mục Trang chủ đang được kích hoạt.

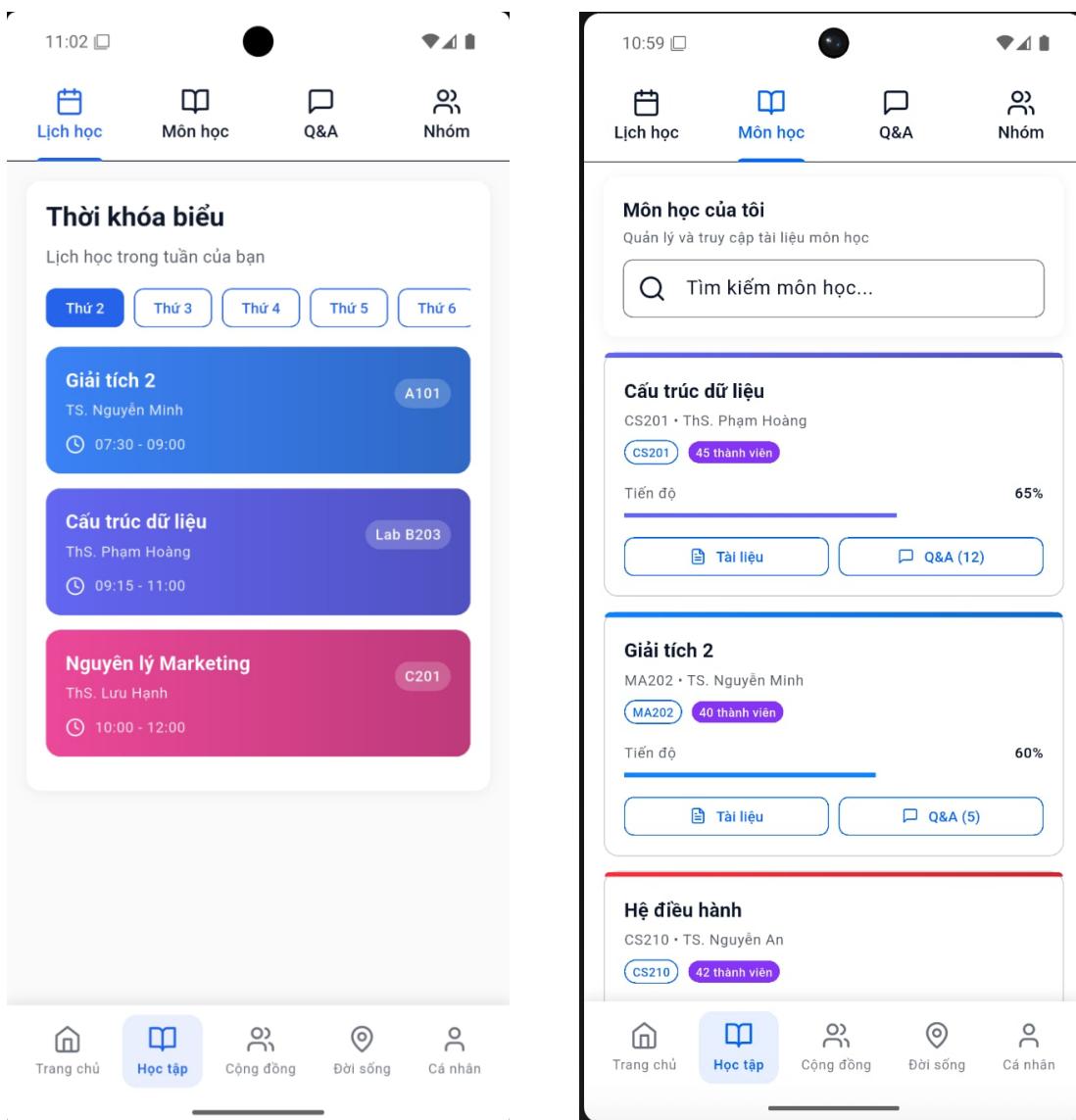
Tổng thể, giao diện Trang chủ được tối ưu về trải nghiệm người dùng, đóng vai trò như trung tâm điều phối thông tin quan trọng, đồng thời rút ngắn thời gian truy cập đến các tính năng cốt lõi mà sinh viên sử dụng hàng ngày.



Hình 31 Màn hình trang chủ

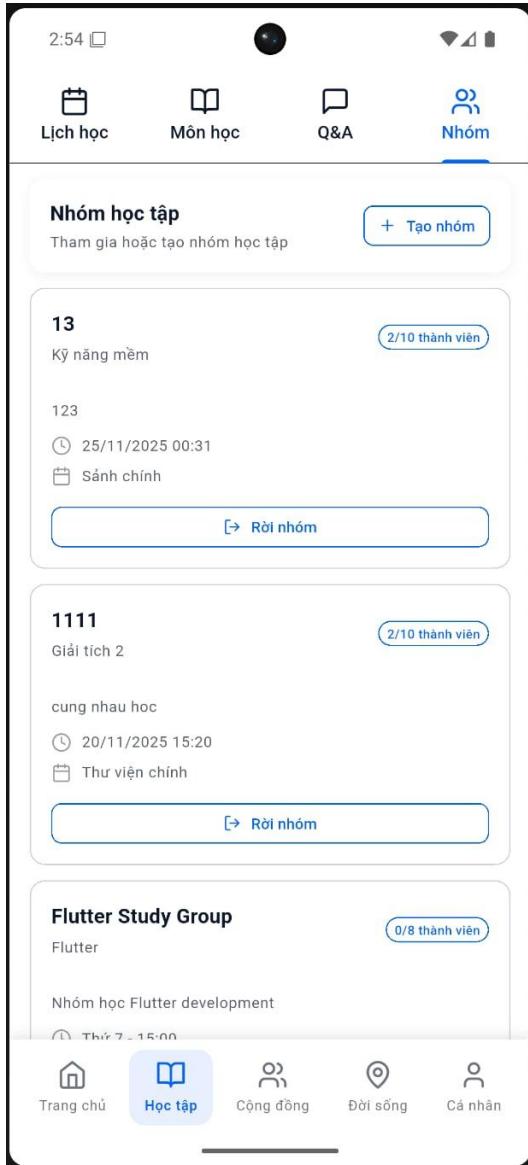
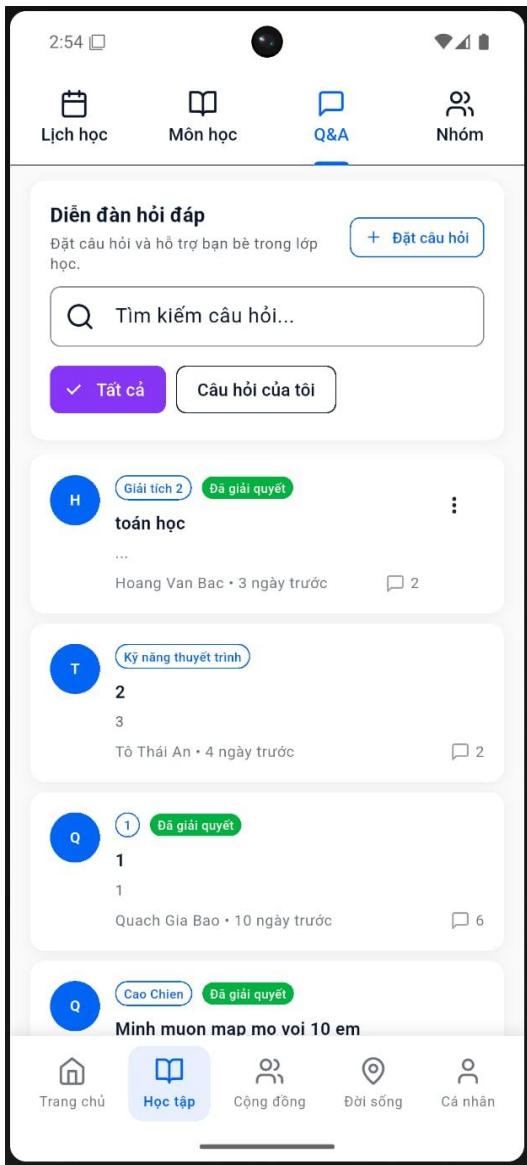
Màn hình Học tập

Màn hình học tập được tổ chức thành bốn tab chính để sinh viên dễ dàng truy cập các chức năng học tập. Tab đầu tiên là Lịch học, nơi hiển thị thời khóa biểu với khả năng chuyển đổi giữa các ngày trong tuần. Tab Môn học cho phép xem danh sách các môn học đang theo học với tiến độ hoàn thành và xem chi tiết được lịch trình của môn học. Tab Q&A là diễn đàn hỏi đáp nơi sinh viên có thể đặt câu hỏi và trả lời câu hỏi của người khác, với chức năng tìm kiếm để lọc theo từ khóa, và xem được danh sách các câu hỏi của bản thân cũng như xem các câu hỏi trên diễn đàn được hiển thị đã giải quyết hay chưa. Tab cuối cùng là Nhóm học tập, nơi sinh viên có thể tham gia hoặc tạo nhóm học tập mới.



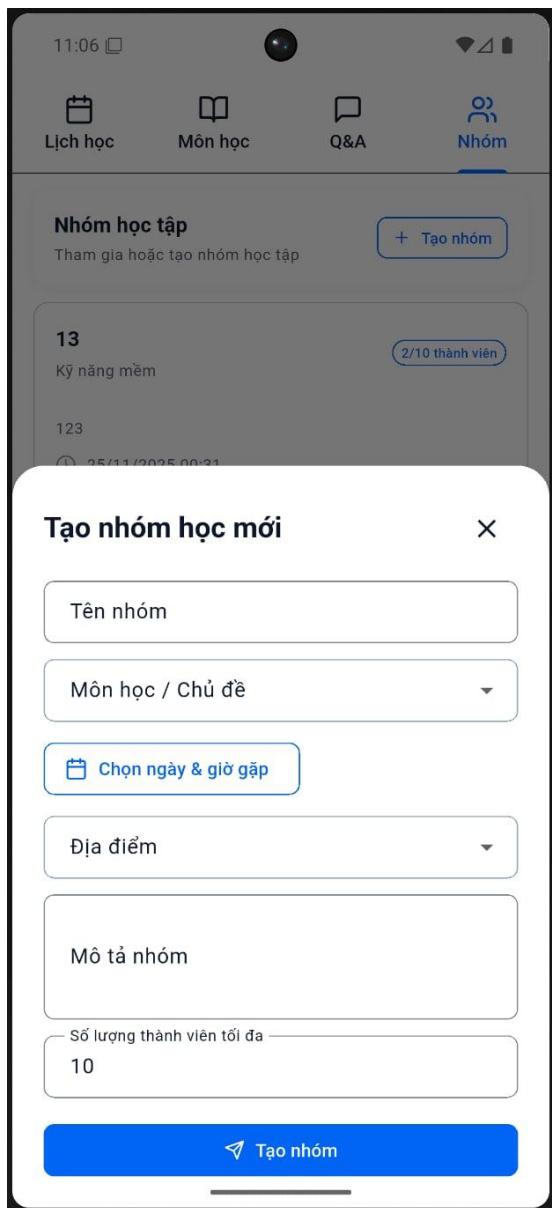
Hình 32.1 Màn hình học tập – Lịch học

Hình 32.2 Màn hình học tập – Môn học

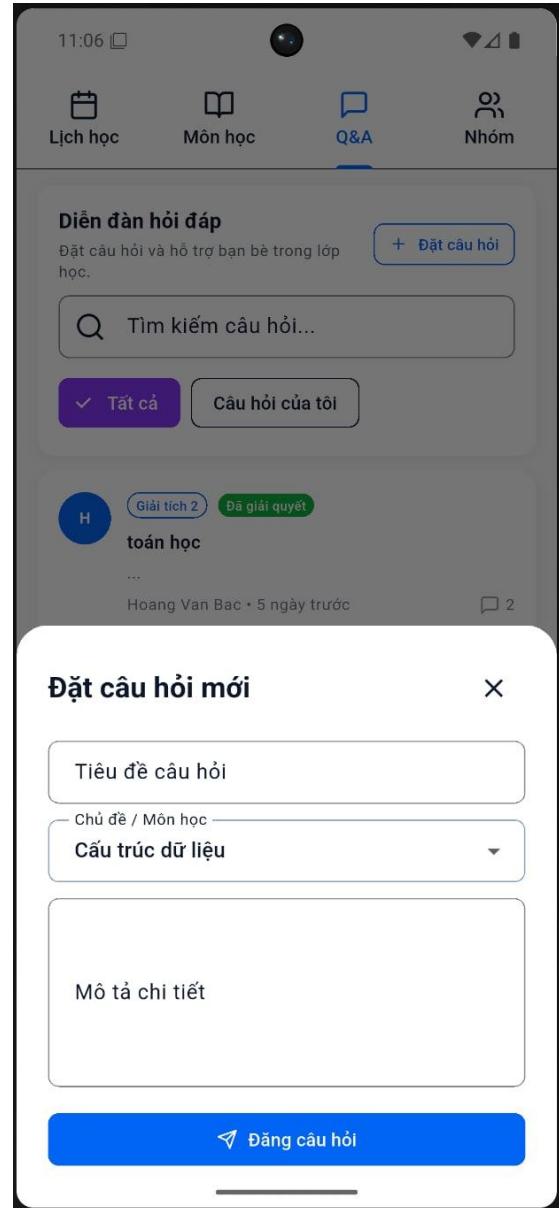


Hình 32.3 Màn hình học tập – Q&A

Hình 32.4 Màn hình học tập – Nhóm



Hình 32.5 Màn hình học tập – Tạo nhóm

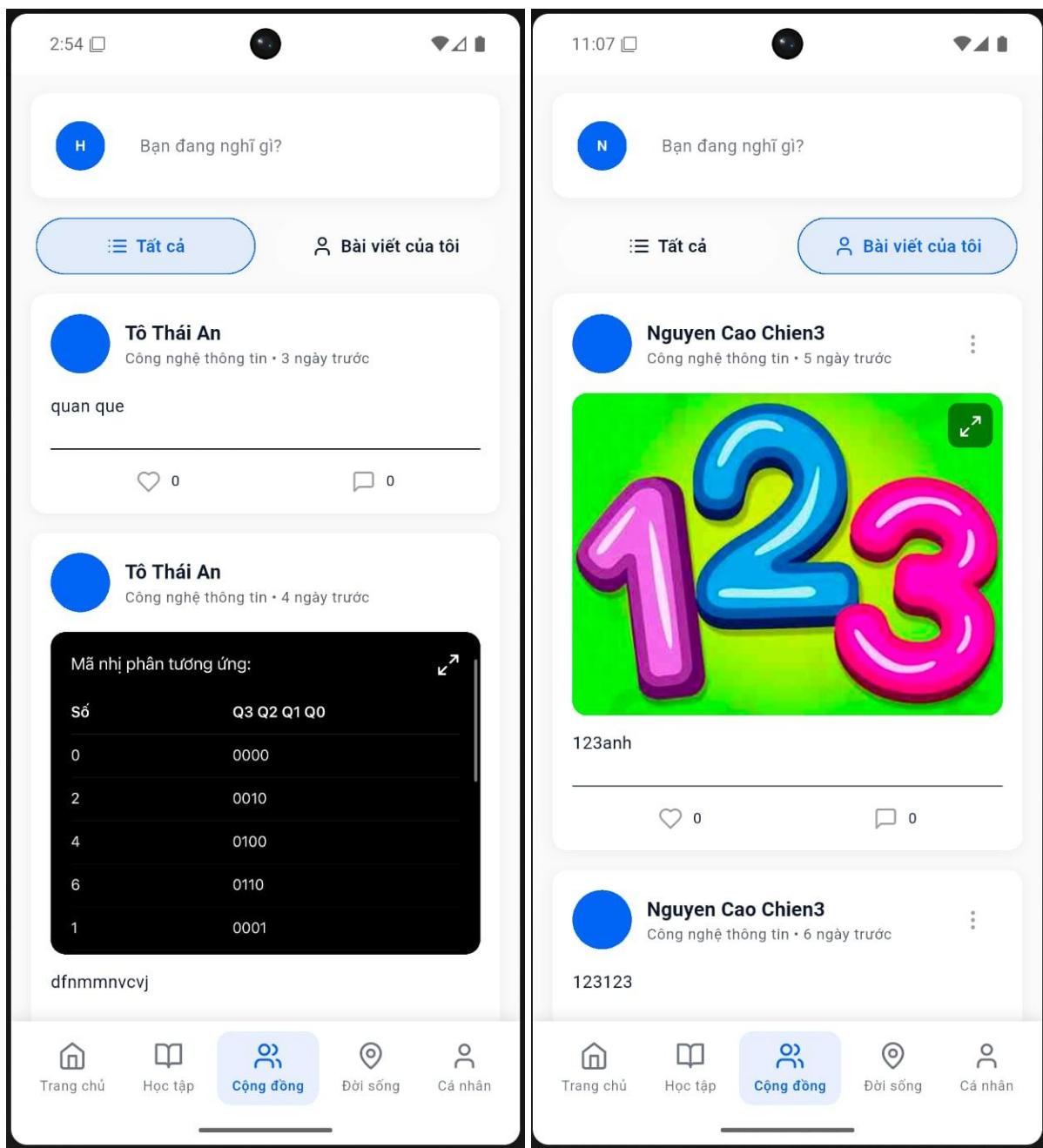


Hình 32.6 Màn hình học tập – Đặt câu hỏi

Hình 32 Màn hình học tập

Màn hình Cộng đồng

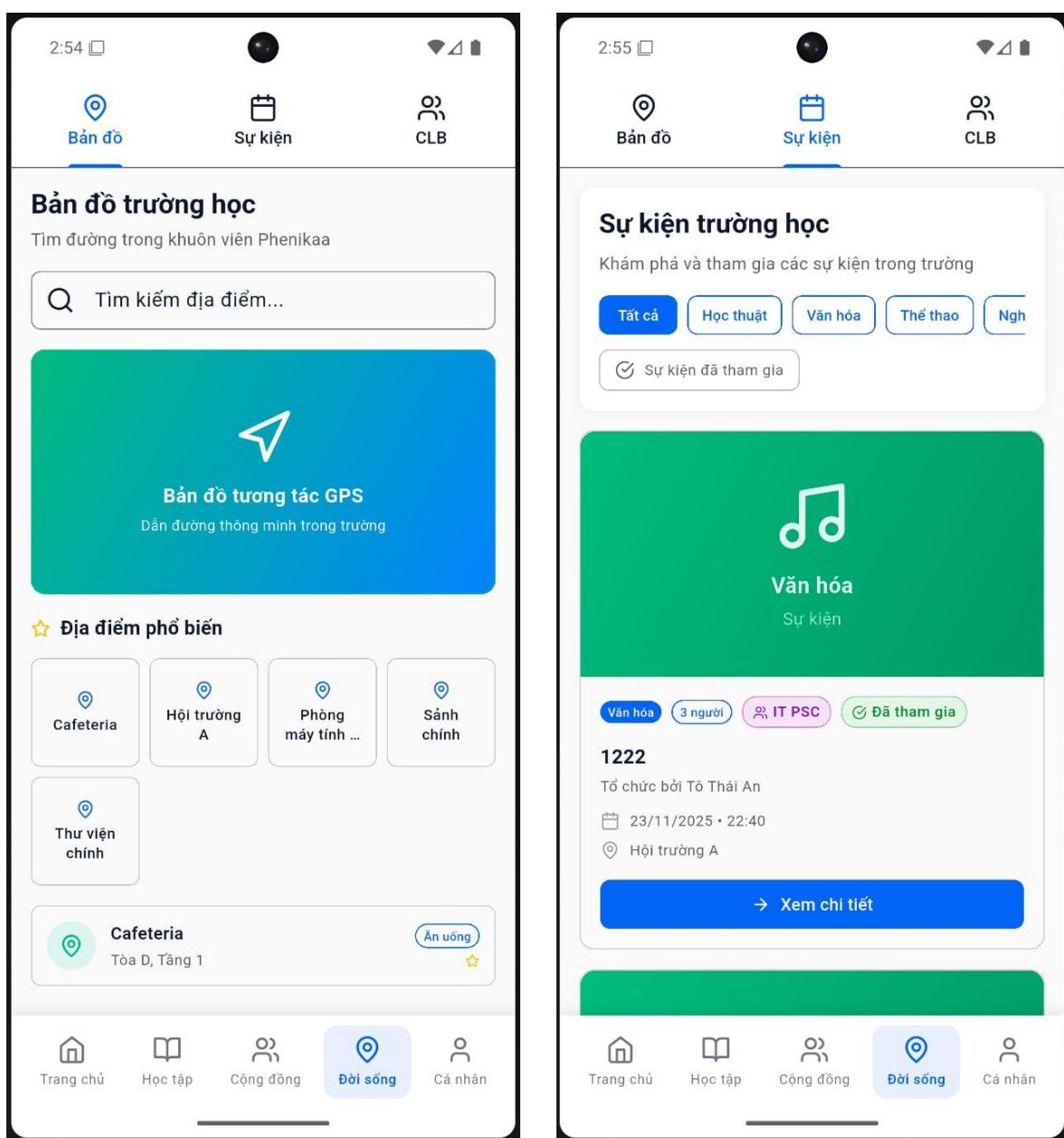
Màn hình cộng đồng là nơi sinh viên chia sẻ và tương tác với bạn bè trong trường. Phần đầu màn hình có card tạo bài viết mới với ô nhập liệu để nhập nội dung, các nút để thêm ảnh và cảm xúc, và nút đăng bài. Ngoài ra bên cạnh giao diện danh sách tất cả các bài viết là danh sách Bài viết của tôi là nơi hiển thị danh sách bài viết tôi đăng lên mạng xã hội cộng đồng. Bên dưới là danh sách các bài viết từ cộng đồng, mỗi bài viết hiển thị thông tin người đăng, nội dung, hình ảnh nếu có, và các nút tương tác để thích và bình luận. Giao diện được thiết kế theo phong cách mạng xã hội quen thuộc để người dùng dễ dàng sử dụng.

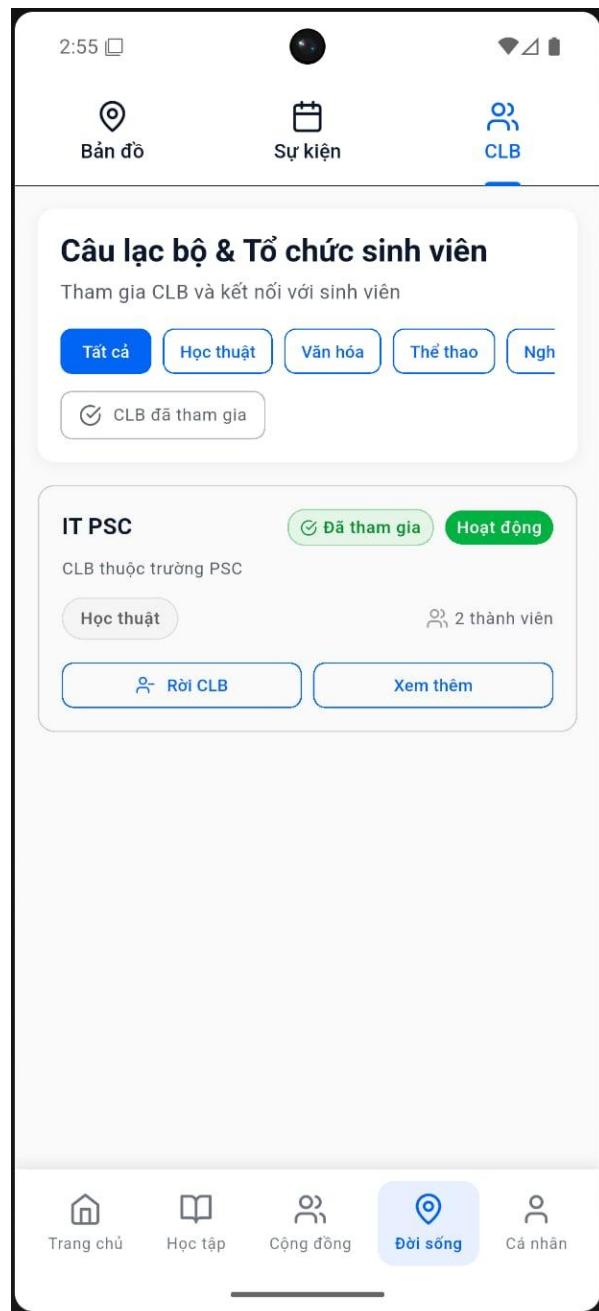


Hình 33 Màn hình cộng đồng-Tất cả

Màn hình Đời sống

Màn hình đời sống được chia thành ba tab chính để sinh viên khám phá các khía cạnh khác nhau của cuộc sống trong trường. Tab Bản đồ cung cấp giao diện tìm kiếm địa điểm và hiển thị danh sách các địa điểm phổ biến trong trường như thư viện, phòng học, căn tin. Tab Sự kiện hiển thị danh sách các sự kiện sắp tới với hình ảnh, thông tin chi tiết về thời gian, địa điểm và số lượng người tham gia, có thể lọc theo danh mục. Tab CLB cho phép sinh viên xem danh sách các câu lạc bộ đang hoạt động, xem thông tin chi tiết và tham gia các câu lạc bộ phù hợp với sở thích của mình.

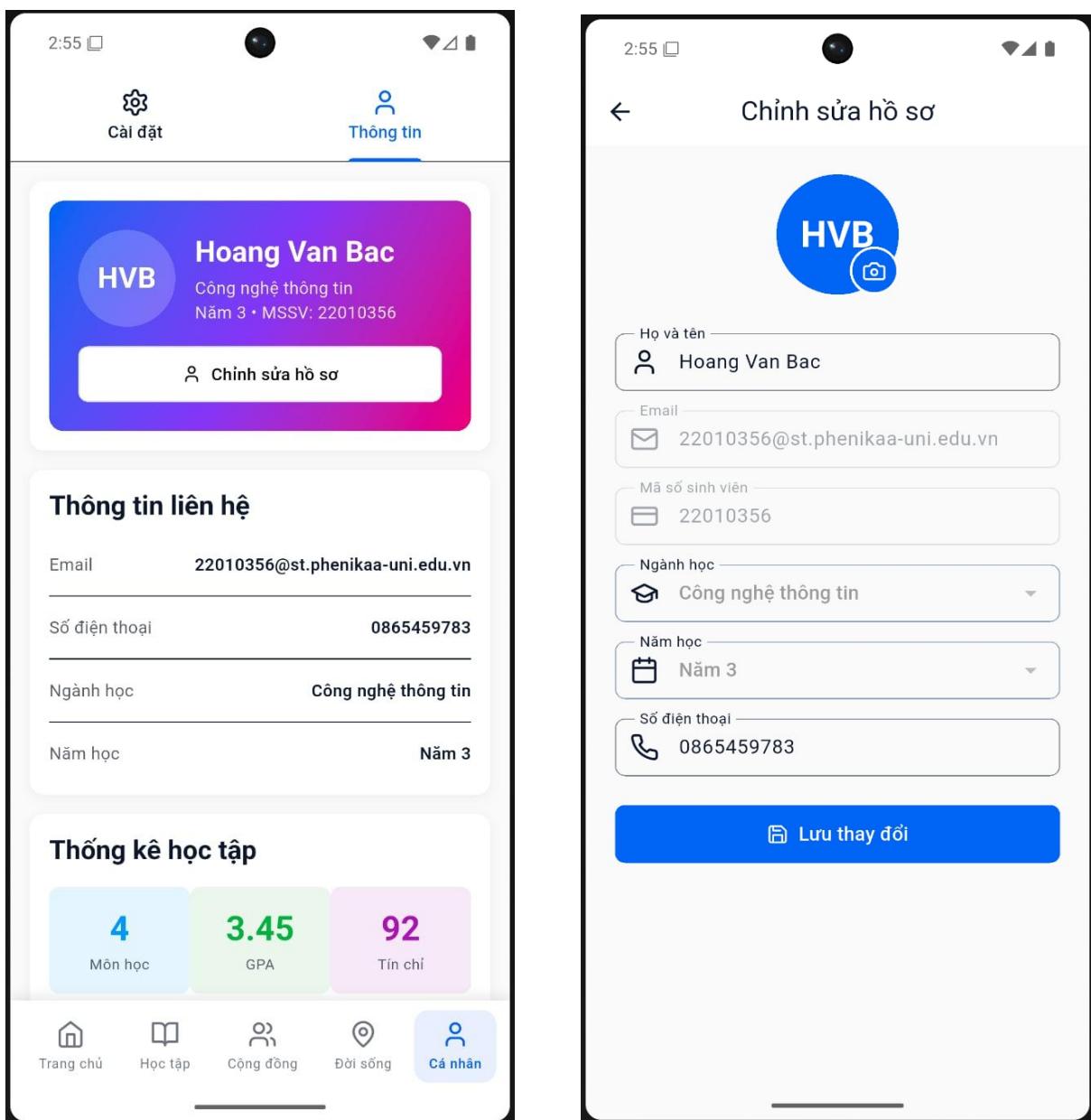




Hình 34 Màn hình đời sống

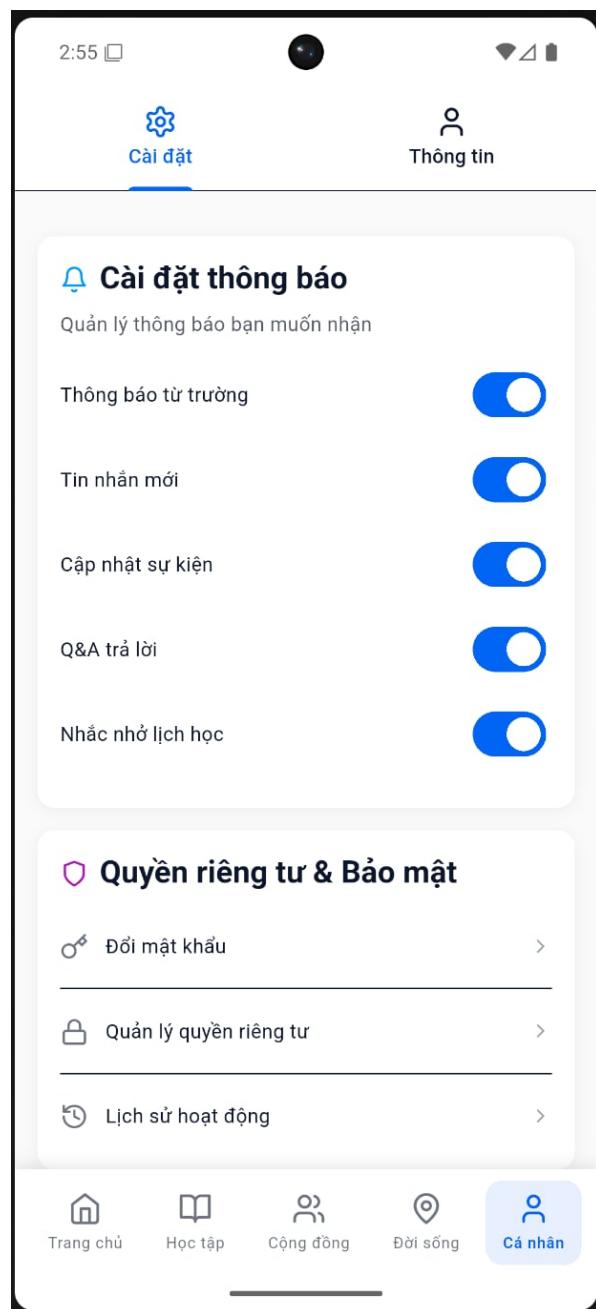
Màn hình Cá nhân

Màn hình cá nhân được tổ chức thành hai tab để quản lý thông tin và cài đặt. Tab Cài đặt cho phép người dùng tùy chỉnh các thiết lập thông báo, quản lý quyền riêng tư và bảo mật như đổi mật khẩu, xem lịch sử hoạt động, và truy cập các tùy chọn hỗ trợ. Tab Thông tin hiển thị hồ sơ cá nhân với avatar gradient đẹp mắt, thông tin liên hệ, thống kê học tập và các liên kết về ứng dụng ngoài ra ở Thông tin có thể chỉnh sửa phần hồ sơ cá nhân như thay đổi Họ và tên với số điện thoại còn các phần khác không thể sửa đổi.



Hình35.1 Màn hình cá nhân-Thông tin

Hình35.2 Màn hình cá nhân – Chính sửa hồ sơ



Hình35.3 Màn hình cá nhân – Cài đặt

Hình 35 Màn hình cá nhân

Màn hình Tổng quan của Admin

Màn hình Bảng điều hành Admin là giao diện trung tâm giúp người quản trị (Admin) theo dõi nhanh chóng tình trạng và các chỉ số quan trọng của toàn bộ hệ thống. Màn hình này được thiết kế với thanh điều hướng chính nằm phía trên, hiển thị tiêu đề chính và cung cấp các tab quản lý chuyên biệt: Tổng quan, Thông báo, Sự kiện, và Người dùng.

Nội dung chính của màn hình bắt đầu với khu vực Tổng quan hệ thống, nơi hiển thị các thẻ thống kê nhanh (statistic cards) về số lượng các đối tượng cơ bản:

- Sinh viên: Hiển thị tổng số sinh viên hiện có trong hệ thống.
- CLB: Hiển thị tổng số Câu lạc bộ đang hoạt động.
- Sự kiện: Hiển thị tổng số sự kiện đang có trong hệ thống.
- Thành viên CLB: Hiển thị tổng số thành viên đã tham gia vào các CLB.

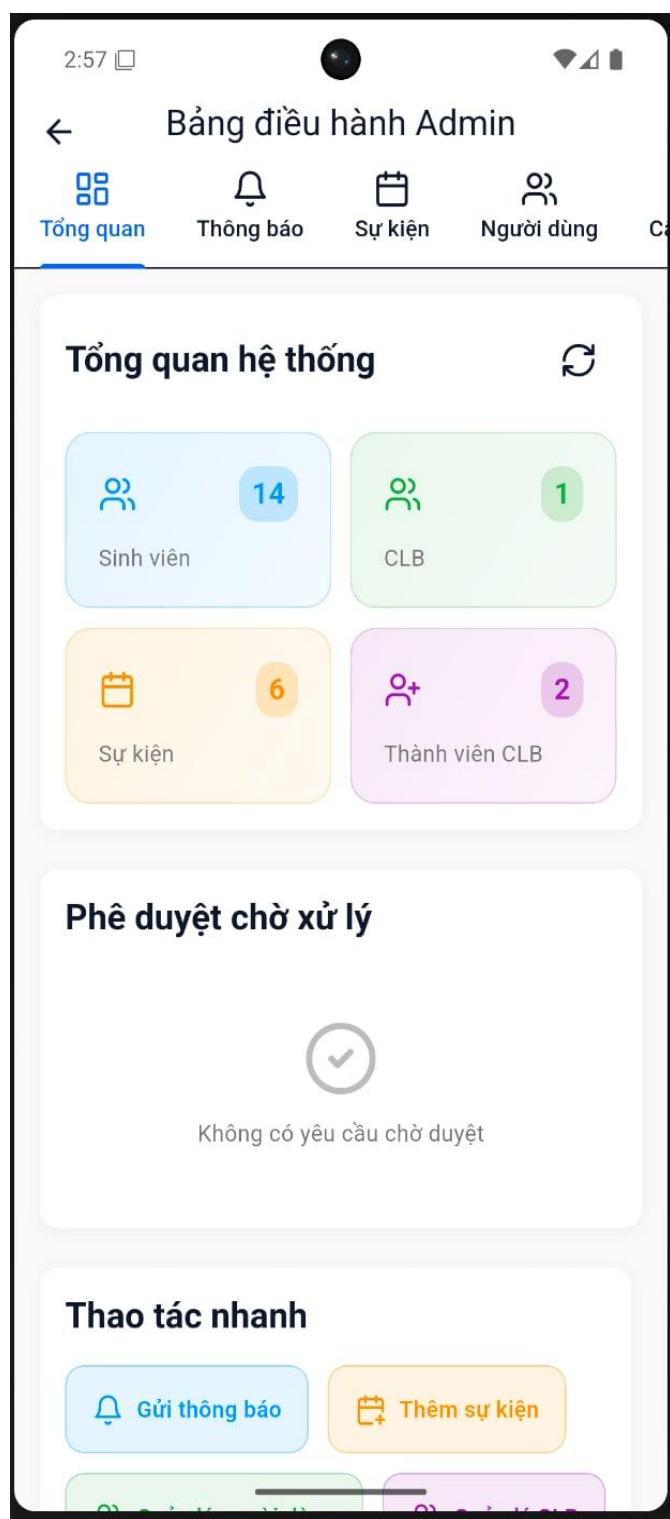
Các thẻ này sử dụng các biểu tượng và màu sắc khác nhau để cung cấp một cái nhìn tổng thể, cập nhật và trực quan về số liệu hệ thống. Đặc biệt, khu vực này còn có nút làm mới để Admin tải lại dữ liệu tức thời.

Tiếp theo, khu vực Phê duyệt chờ xử lý được đặt ở phía dưới, có chức năng làm nơi tập trung hiển thị các tác vụ, yêu cầu hoặc nội dung cần Admin can thiệp, kiểm tra và phê duyệt trước khi được công bố (ví dụ: đăng ký tham gia CLB, đăng bài cộng đồng). Trong hình ảnh cụ thể này, hệ thống đang thông báo "Không có yêu cầu chờ duyệt", cho thấy người quản trị đã xử lý hoàn tất mọi công việc kiểm duyệt pending. Đây là một cơ chế quan trọng để quản lý luồng công việc kiểm duyệt.

Cuối cùng là khu vực Thao tác nhanh được đặt ở phía cuối, có chức năng thực hiện nhanh các hoạt động quản lý thường xuyên, giúp Admin tiết kiệm thời gian điều hướng. Các thao tác nhanh bao gồm:

- Gửi thông báo: Cho phép Admin nhanh chóng soạn và gửi thông báo quan trọng đến người dùng.
- Thêm sự kiện: Cho phép Admin tạo mới một sự kiện trong hệ thống.
- Thêm thành viên câu lạc bộ: Cho phép Admin bổ sung trực tiếp một người dùng làm thành viên của một CLB cụ thể.
- Tạo tài khoản người dùng: Cho phép Admin khởi tạo nhanh một tài khoản người dùng mới (ví dụ: cho sinh viên hoặc giảng viên) vào hệ thống.

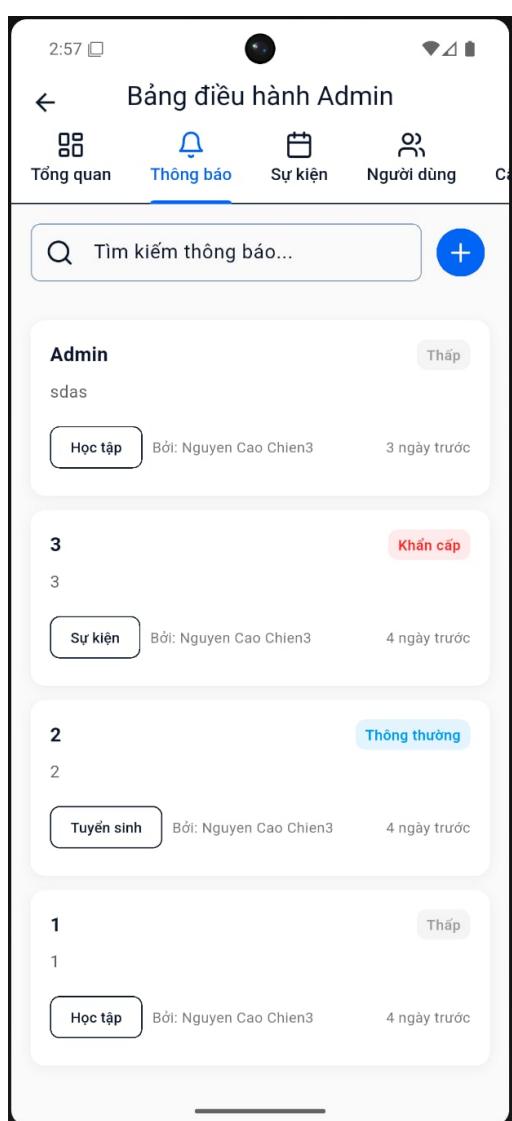
Nhìn chung, màn hình này được thiết kế nhằm giúp Admin dễ dàng nắm bắt tình hình hệ thống (qua số liệu thống kê), xử lý nhanh các hoạt động (qua Thao tác nhanh), đồng thời theo dõi và quản lý hiệu quả các luồng công việc quan trọng (qua Phê duyệt chờ xử lý), đảm bảo ứng dụng vận hành trơn tru và an toàn.



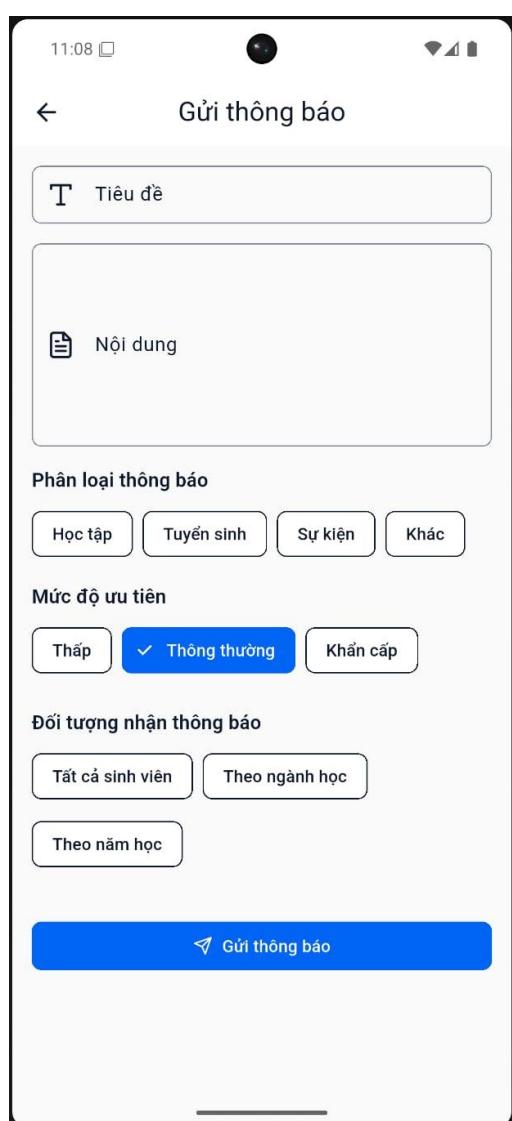
Hình 36 Màn hình tổng quan của Admin

Màn hình Quản lý thông báo – Admin

Màn hình quản lý thông báo của Admin được chia thành hai giao diện chính: thứ nhất là màn hình Danh sách Thông báo nằm trong tab Thông báo của Bảng điều hành Admin, nơi hiển thị nút "+ Gửi thông báo mới" cho phép tạo thông báo nhanh chóng. Khu vực chính của màn hình này liệt kê các thông báo đã gửi dưới dạng các thẻ chi tiết, mỗi thẻ hiển thị tiêu đề, người gửi, thời gian gửi, và quan trọng là cả Phân loại (Học tập, Sự kiện, Tuyển sinh,...) lẫn Mức độ ưu tiên (Thấp, Thông thường, Khẩn cấp), giúp người quản trị theo dõi và đánh giá tầm quan trọng của các thông báo đã phát hành một cách trực quan. Giao diện thứ hai là màn hình Gửi thông báo chuyên dụng, được thiết kế với các trường nhập liệu cho Tiêu đề và Nội dung; tiếp theo là các tùy chọn phân loại rõ ràng cho phép Admin chọn Phân loại thông báo (như Học tập, Tuyển sinh), xác định Mức độ ưu tiên (Thấp, Thông thường, Khẩn cấp), và đặc biệt là tùy chọn nhắm Đối tượng nhận thông báo cụ thể, bao gồm Tất cả sinh viên, Theo ngành học, hoặc Theo năm học, trước khi kết thúc bằng nút "Gửi thông báo" để hoàn tất quá trình.



Hình 37.1 Màn hình Danh sách thông báo

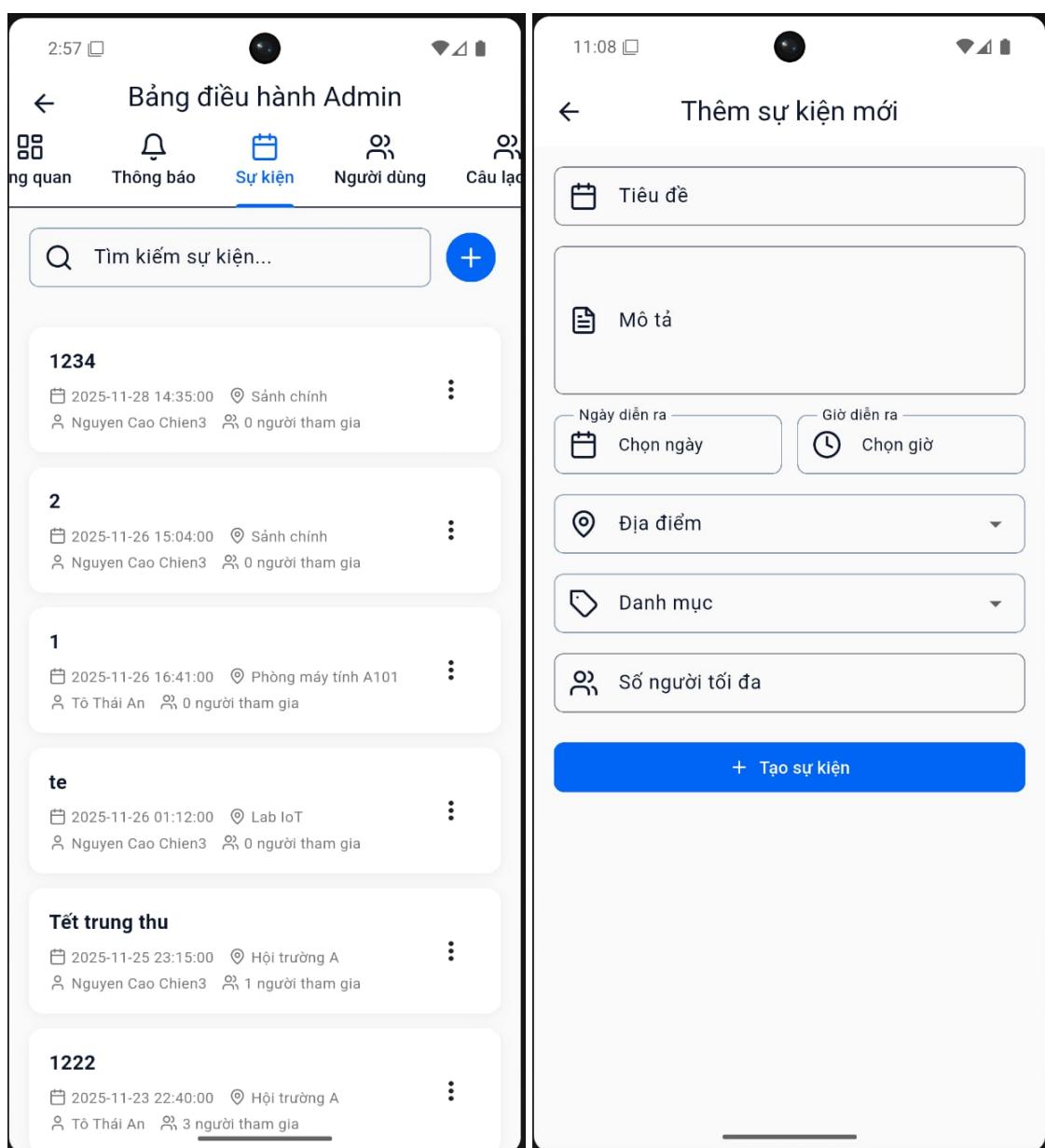


Hình 37.2 Màn hình Gửi thông báo

Hình 37 Màn hình Quản lý thông báo của Admin

Màn hình Quản lý sự kiện của Admin

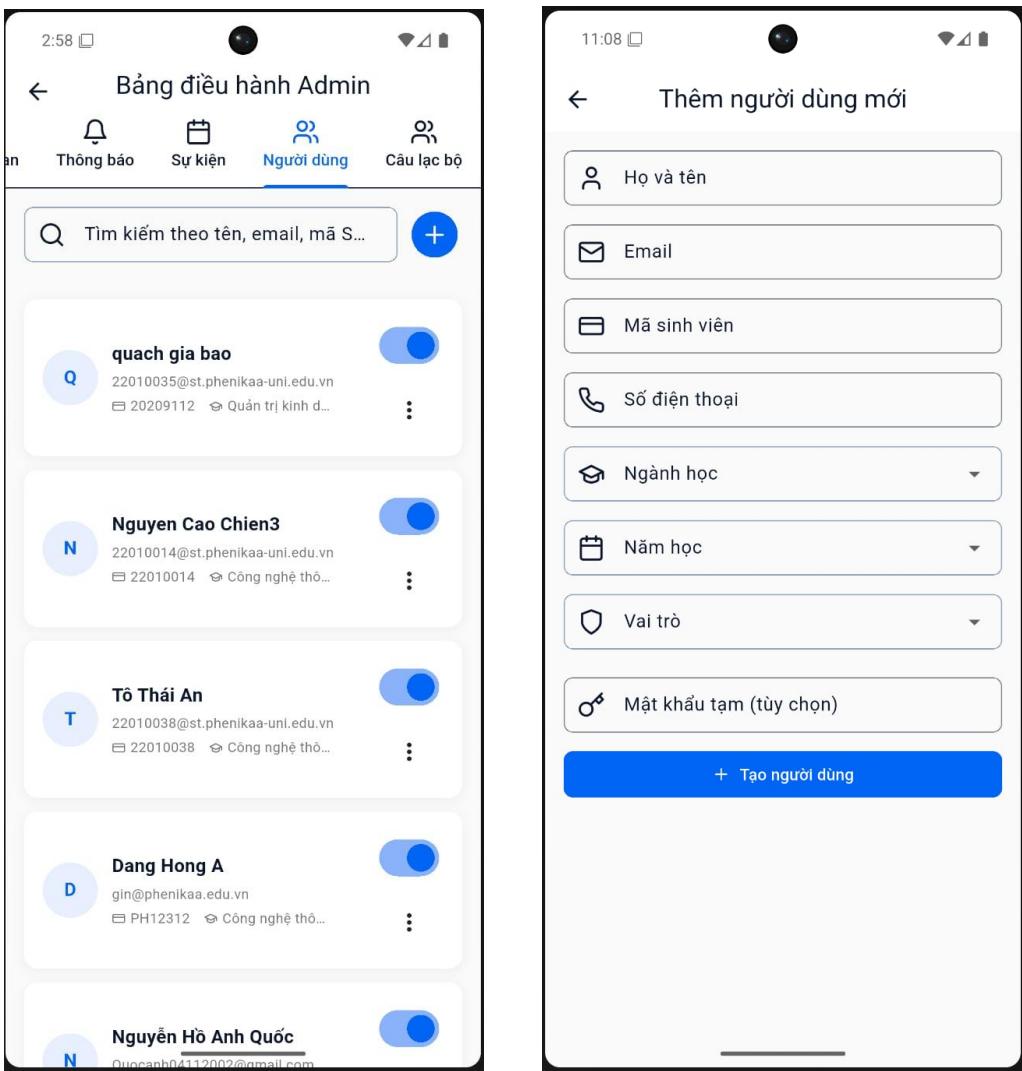
Màn hình Quản lý Sự kiện là một phần của Bảng điều hành Admin, với tab Sự kiện đang được chọn, có chức năng chính là cho phép người quản trị kiểm soát các sự kiện đã tạo trong hệ thống; màn hình bắt đầu với thanh tìm kiếm (Tìm kiếm sự kiện...) và nút "+" nổi bật để tạo sự kiện mới. Khu vực chính liệt kê các sự kiện dưới dạng các thẻ, mỗi thẻ hiển thị chi tiết tiêu đề, ngày giờ diễn ra, Địa điểm, người tạo và Số người tham gia (ví dụ: 0 người tham gia), kèm theo biểu tượng ba chấm (...) ở góc phải để thực hiện các thao tác quản lý bổ sung như chỉnh sửa hoặc xóa sự kiện. Màn hình này được thiết kế để Admin dễ dàng tìm kiếm, theo dõi thông tin cơ bản và quản lý vòng đời của tất cả các sự kiện một cách hiệu quả.



Hình 38 Màn hình quản lý sự kiện của Admin

Màn hình quản lý người dùng của Admin

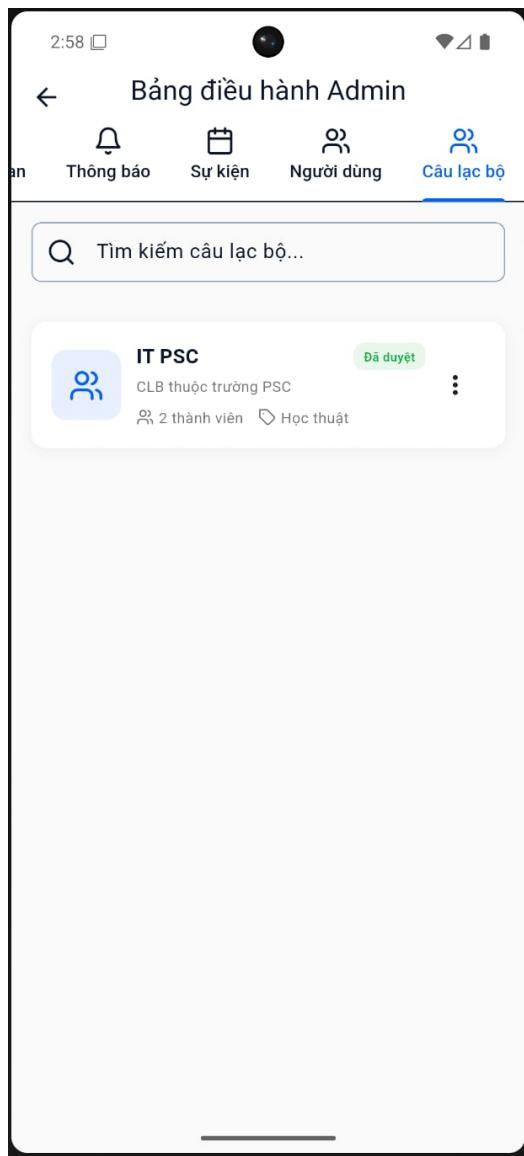
Màn hình Quản lý Người dùng là một phần quan trọng của Bảng điều hành Admin, tập trung vào việc cho phép Admin tra cứu, kiểm soát và quản lý các tài khoản sinh viên trong hệ thống; màn hình bắt đầu với thanh tìm kiếm linh hoạt (Tìm kiếm theo tên, email, mã SV...) và nút "+" để thực hiện các thao tác quản trị hàng loạt hoặc thêm người dùng mới. Khu vực chính liệt kê danh sách người dùng dưới dạng các thẻ chi tiết, mỗi thẻ hiển thị tên người dùng, email sinh viên, MSSV và thông tin ngành học cơ bản, đồng thời cung cấp hai công cụ quản trị trực quan: một Toggle Switch ở góc phải để Admin nhanh chóng thay đổi trạng thái kích hoạt/chặn tài khoản, và biểu tượng ba chấm (...) để truy cập các tùy chọn quản lý sâu hơn như chỉnh sửa hoặc reset mật khẩu, qua đó giúp Admin dễ dàng theo dõi và kiểm soát hoạt động của từng tài khoản người dùng một cách hiệu quả.



Hình 39 Màn hình Quản lý người dùng của Admin

Màn hình quản lý CLB của Admin

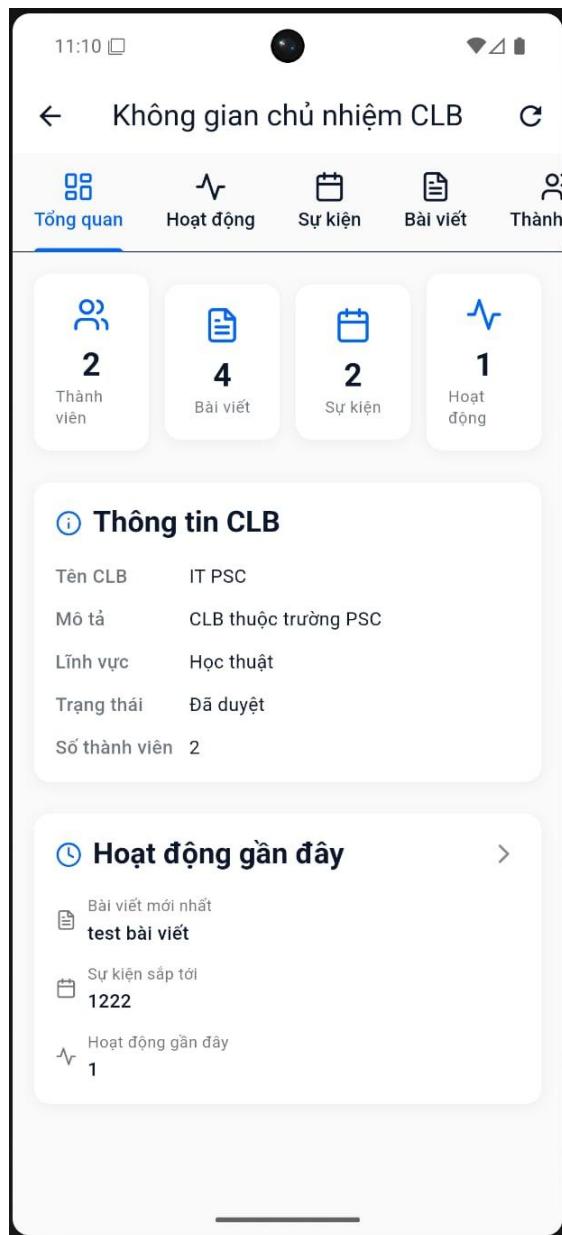
Màn hình Quản lý Câu lạc bộ là một phần của Bảng điều hành Admin, được thiết kế để Admin tra cứu và quản lý các CLB trong hệ thống, với tab Câu lạc bộ đang được chọn. Màn hình được bố trí với một thanh tìm kiếm linh hoạt (Tìm kiếm câu lạc bộ...) nằm ở phía trên, giúp Admin lọc nhanh danh sách. Khu vực chính liệt kê các câu lạc bộ dưới dạng các thẻ riêng biệt, mỗi thẻ hiển thị Tên CLB (ví dụ: IT PSC), mô tả ngắn gọn (CLB thuộc trường PSC), các chỉ số cơ bản như Số thành viên (2 thành viên), Lĩnh vực hoạt động (Học thuật), và trạng thái Đã duyệt được đánh dấu màu xanh lá cây nổi bật. Ngoài ra, mỗi thẻ CLB còn có biểu tượng ba chấm ngang (...) ở góc phải, cung cấp tùy chọn truy cập vào các hành động quản trị chi tiết hơn như chỉnh sửa thông tin CLB hoặc xem danh sách thành viên. Màn hình này cung cấp khả năng tìm kiếm và các công cụ quản lý trực quan để Admin kiểm soát thông tin và trạng thái của các CLB một cách hiệu quả.



Hình 40 Màn hình Quản lý CLB của Admin

Màn hình quản lý Tổng quan của Chủ nhiệm CLB

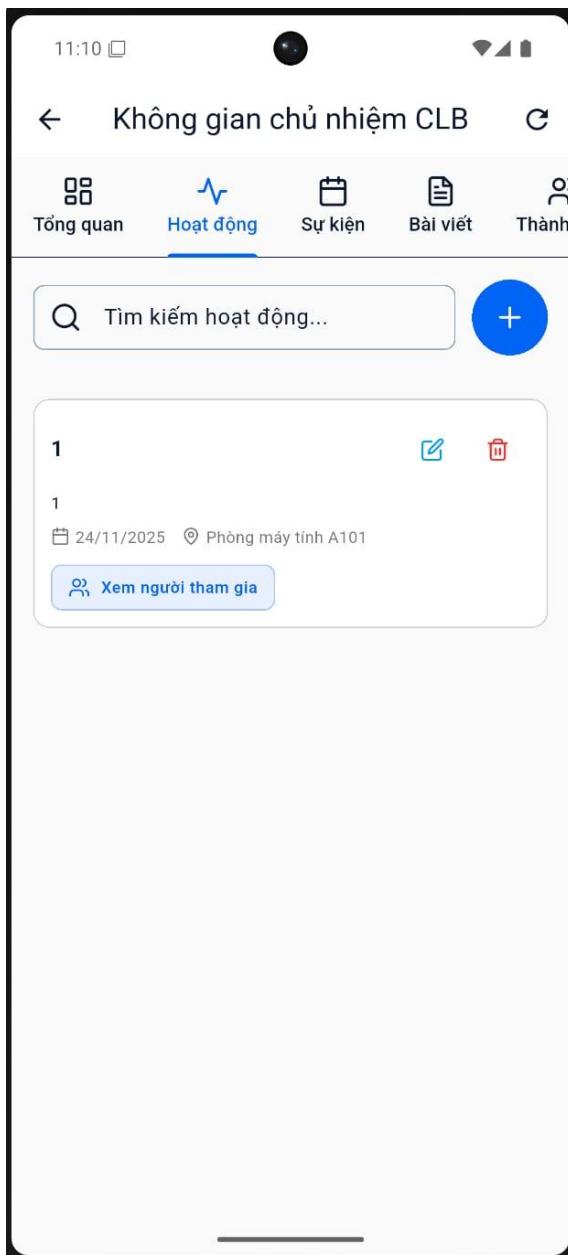
Màn hình này là giao diện Tổng quan (Dashboard) của ứng dụng quản lý Câu lạc bộ, được đặt trong "Không gian chủ nhiệm CLB". Giao diện này cung cấp một cái nhìn tổng thể và nhanh chóng về tình trạng hoạt động của CLB, bắt đầu bằng thanh thống kê nhanh hiển thị bốn chỉ số cốt lõi: Thành viên, Bài viết, Sự kiện, và Hoạt động. Tiếp theo là phần Thông tin CLB chi tiết, xác định CLB tên là IT PSC, hoạt động trong Lĩnh vực Học thuật, thuộc trường PSC, và đang ở trạng thái Đã duyệt. Cuối cùng, phần Hoạt động gòn đây tóm tắt các sự kiện mới nhất, bao gồm bài viết mới nhất và sự kiện sắp tới. Tổng quan cho thấy đây là một CLB chính thức, đang hoạt động trong lĩnh vực Học thuật và hiển thị số lượng thành viên.



Hình 41 Màn hình quản lý Tổng quan của Chủ nhiệm CLB

Màn hình quản lý Hoạt động của Chủ nhiệm CLB

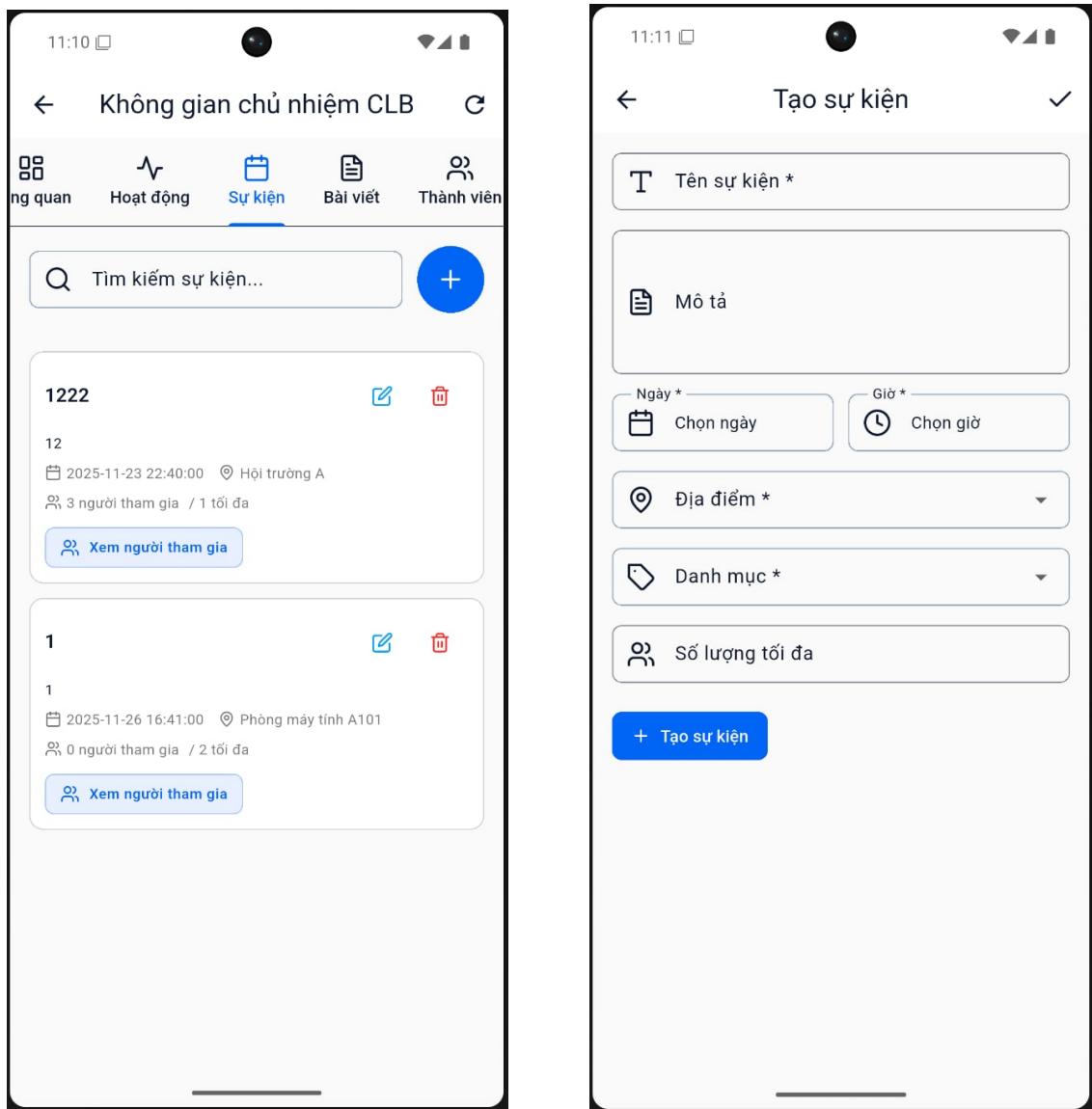
Màn hình này hiển thị giao diện "Không gian chủ nhiệm CLB" với tab "Hoạt động" đang được chọn, cho phép chủ nhiệm câu lạc bộ quản lý các hoạt động đã lên lịch. Ngay dưới thanh điều hướng là một thanh tìm kiếm với gợi ý "Tìm kiếm hoạt động..." và có thể thêm hoạt động mới. Phần danh sách hiển thị sẽ có tên hoạt động, thời gian và địa điểm diễn ra hoạt động. Hoạt động này có các tùy chọn quản lý đi kèm, bao gồm chỉnh sửa, xóa hoạt động và Xem người tham gia, cho phép chủ nhiệm kiểm tra danh sách thành viên đăng ký tham gia sự kiện này.



Hình 42 Màn hình quản lý Hoạt động của Chủ nhiệm CLB

Màn hình quản lý Sự kiện của Chủ nhiệm CLB

Màn hình này hiển thị giao diện quản lý Sự kiện trong "Không gian chủ nhiệm CLB", nơi chủ nhiệm có thể tạo mới, tìm kiếm và quản lý các sự kiện đã lên lịch. Danh sách liệt kê các sự kiện. Giao diện các sự kiện có các chức năng như tìm kiếm, thêm, sửa, xóa sự kiện. Ngoài ra có thể xem chi tiết số lượng người tham gia cũng như chi tiết của sự kiện. Phần tạo sự kiện ở trên là tiêu đề dưới nó là nơi để mô tả về sự kiện cũng như các phần chi tiết của sự kiện như thời gian, địa điểm, danh mục và số lượng người tham gia tối đa.

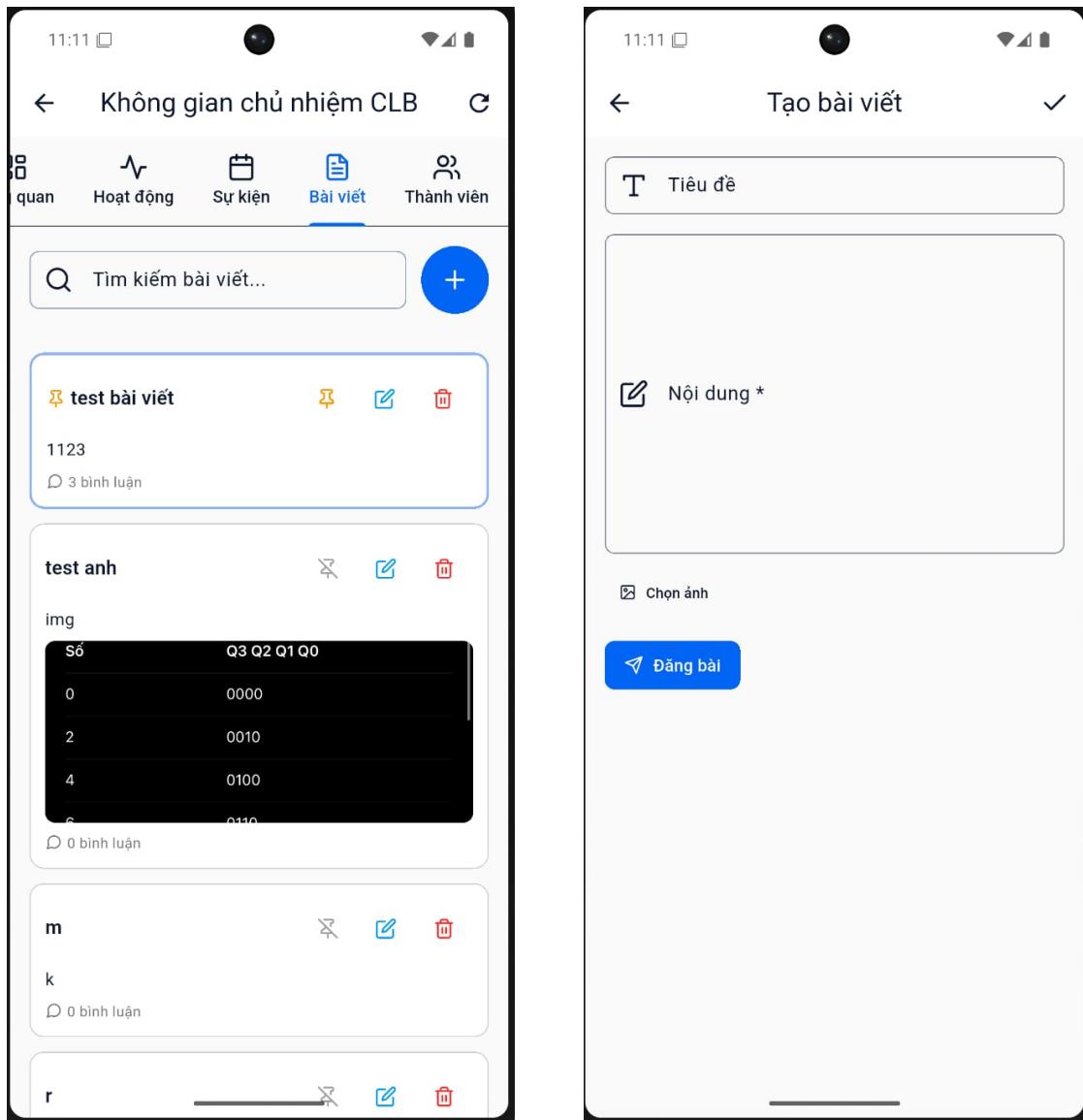


Hình 43 Màn hình quản lý Sự kiện của Chủ nhiệm CLB

Màn hình quản lý Bài viết của Chủ nhiệm CLB

Màn hình này hiển thị giao diện quản lý Bài viết trong "Không gian chủ nhiệm CLB", nơi chủ nhiệm có thể tạo mới, tìm kiếm và quản lý các bài viết đã được đăng. Danh sách liệt kê một số bài viết, bao gồm bài viết thứ nhất có một số bình luận, bài viết thứ hai có chèn hình ảnh cùng một bảng dữ liệu kỹ thuật, và các bài viết khác chỉ có tiêu đề ngắn gọn và hiện chưa có bình luận nào. Giao diện quản lý các bài viết có các chức năng như tìm kiếm bài viết, thêm bài viết mới bằng nút dấu cộng, và các tùy chọn sửa, xóa cho từng bài viết, đồng thời có thể gắn bài viết lên đầu (biểu tượng đinh ghim). Ngoài ra, có thể xem chi tiết số lượng bình luận và nội dung của bài viết. Phần tạo bài viết (màn hình thứ hai) ở trên là ô Tiêu đề, dưới nó là ô Nội

dung bắt buộc phải có, cùng với tùy chọn Chọn ảnh và nút Đăng bài để hoàn tất quá trình xuất bản.

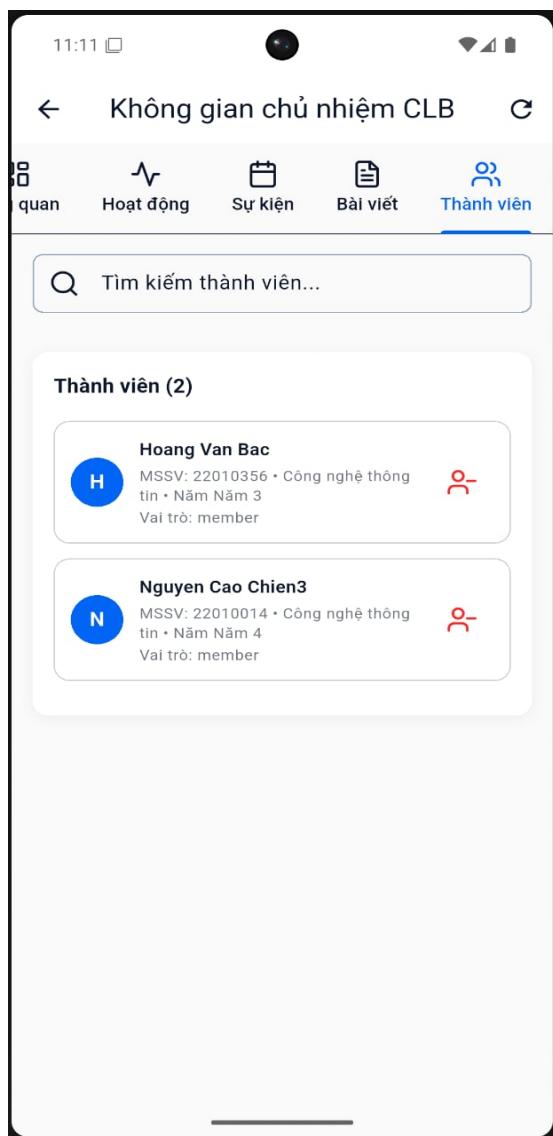


Hình 44 Màn hình quản lý Bài viết của Chủ nhiệm CLB

Màn hình quản lý Thành viên của Chủ nhiệm CLB

Màn hình này hiển thị giao diện quản lý Thành viên trong "Không gian chủ nhiệm CLB", nơi chủ nhiệm có thể tìm kiếm và quản lý các thành viên hiện tại. Danh sách liệt kê các thành viên trong CLB, hiện tại là hai thành viên. Giao diện quản lý thành viên có chức năng tìm kiếm thành viên bằng thanh tìm kiếm, và danh sách hiển thị chi tiết của từng thành viên, bao gồm tên thành viên, Mã số sinh viên (MSSV), ngành học năm học và Vai trò. Ngoài ra, mỗi thành viên có một biểu tượng quản lý hoặc xóa (biểu tượng người với dấu trừ màu đỏ) để chủ nhiệm có thể thực hiện các thao tác quản lý. Phần tạo thành viên mới (không hiển thị trong ảnh) sẽ là một biểu mẫu để thêm chi tiết của thành viên mới vào hệ thống, bao gồm tên, MSSV, vai trò

và các thông tin cá nhân liên quan.



Hình 45 Màn hình quản lý Bài viết của Chủ nhiệm CLB

3 Cài đặt, kiểm thử và triển khai

Phần này mô tả chi tiết quá trình cài đặt, kiểm thử và triển khai hệ thống Phenikaa Connect App – một nền tảng kết nối cộng đồng sinh viên Trường Đại học Phenikaa. Hệ thống sử dụng Supabase làm backend-as-a-service (BaaS) kết hợp với Flutter frontend để cung cấp trải nghiệm realtime, bảo mật và mở rộng dễ dàng. Toàn bộ quy trình phát triển được thực hiện theo hướng DevOps hiện đại, tập trung vào tự động hóa, chất lượng mã nguồn và khả năng mở rộng ổn định.

3.1 Môi trường phát triển

Technology Stack

Hệ thống Phenikaa Connect App được xây dựng với technology stack hiện đại, tận dụng hạ tầng Supabase cho backend, PostgreSQL cho cơ sở dữ liệu, và Flutter cho giao diện người dùng. Stack này đảm bảo hiệu suất cao, realtime, và dễ dàng mở rộng trong tương lai.

- *Backend Technologies:*

- Platform: Supabase (tích hợp PostgreSQL + Realtime + Auth + Storage)
- Database: PostgreSQL 15+
- Authentication: Supabase Auth với Google OAuth 2.0
- Functions: Edge Functions viết bằng Deno/TypeScript
- Real-time: Supabase Realtime API (subscription channels)
- Storage: Supabase Storage (quản lý file và hình ảnh)
- Security: Row Level Security (RLS) và policies cho từng bảng
- API Services: RESTful endpoints tự động sinh từ schema PostgreSQL
- ORM Layer: Supabase Dart Client (Flutter SDK)
- Trigger & Procedures: SQL-based functions quản lý counters và sự kiện

- *Frontend Technologies:*

- Framework: Flutter 3.x
- Programming Language: Dart 3.x
- UI Toolkit: Material Design 3
- State Management: Provider + ChangeNotifier
- Navigation: go_router
- Data Layer: Supabase Dart SDK

- Local Storage: SharedPreferences
- Realtime: Supabase subscription streams
- Charts & Visualization: fl_chart
- Notifications: Firebase Cloud Messaging (đang phát triển)
- Image Handling: image_picker, file_picker

- *Development Tools:*

- Code Editor: Visual Studio Code
- Version Control: Git + GitHub
- Database Tool: Supabase SQL Editor
- Testing Tools: Flutter Test, Postman (REST API validation)
- API Monitoring: Supabase Dashboard
- Package Manager: Flutter Pub
- Design & Prototyping: Figma
- Issue Tracking: GitHub Projects

Development Environment Setup

- *System Requirements:*

- Operating System: Windows 10 / macOS 12+ / Ubuntu 22.04+
- Processor: Intel i5 / Apple M1 trở lên
- Memory: 8GB RAM (tối thiểu 4GB)
- Storage: 10GB trống
- Network: Kết nối Internet ổn định để truy cập Supabase Cloud

- *Required Software:*

- Flutter SDK: v3.24.0+
- Dart SDK: v3.3+ (đi kèm Flutter)
- Supabase CLI: v1.190+ (tùy chọn cho Edge Functions test)
- Git: v2.45+
- VS Code Extensions: Flutter, Dart, Supabase

- *Environment Configuration:*

Tệp cấu hình chính được đặt tại:

lib/config/supabase_config.dart

```
class SupabaseConfig {  
  static const String supabaseUrl = 'YOUR_SUPABASE_URL';  
  static const String supabaseAnonKey = 'YOUR_SUPABASE_ANON_KEY';  
}  
  
import 'package:supabase_flutter/supabase_flutter.dart';  
  
class SupabaseConfig {  
  
  // Thay thế bằng URL và API key thực tế của Supabase project  
  
  static const String supabaseUrl = 'https://dvhuokxbokovcpwawrio.supabase.co';  
  
  static const String supabaseAnonKey =  
'eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9eyJpc3MiOiJzdXBhYmFzZSIsInJlZiI6ImR2aHVva3hib2tvdmNwd2F3cmhvIiwicm9sZSI6ImFub24iLCJpYXQiOjE3NjExNzk0MDQsImV4cCI6MjA3Njc1NTQwNH0.pagcZC4HYiB2HJynPhCV8r8pOuS8uNhHG6IoSs4sBTA';  
  
  static SupabaseClient get client => Supabase.instance.client;  
  
  static Future<void> initialize() async {  
  
    await Supabase.initialize(  
  
      url: supabaseUrl,  
  
      anonKey: supabaseAnonKey,  
  
    );  
  
  }  
}
```

*Environment Variables (.env):

```
SUPABASE_URL=https://your-project-id.supabase.co  
SUPABASE_ANON_KEY=your-anon-public-key  
SUPABASE_SERVICE_ROLE_KEY=your-service-role-key  
SUPABASE_PROJECT_ID=phenikaa-connect
```

Development Workflow

- *Supabase Setup:*

1. Truy cập <https://supabase.com>
2. Tạo project mới → chọn region và database password
3. Ghi lại Project URL và Anon Key
4. Chạy tuần tự 3 file SQL trong SQL Editor:
 - o supabase_schema.sql: Tạo bảng, indexes, triggers
 - o supabase_functions.sql: Khai báo functions & procedures
 - o supabase_mock_data.sql: Thêm dữ liệu mẫu (users, posts, events...)

- *Local Development Setup:*

```
# Clone project
```

```
gh repo clone bao0694/PhenikaaConnect
```

```
git clone <repository- https://github.com/bao0694/PhenikaaConnect.git>
```

```
cd phenikaa_connect_app
```

```
# Cài đặt dependencies
```

```
flutter pub get
```

```
# Cấu hình Supabase URL và API key
```

```
vi lib/config/supabase_config.dart
```

```
# Chạy ứng dụng
```

```
flutter run
```

- *Development Scripts:*

- Chạy ứng dụng debug:
- flutter run
- Build ứng dụng release:
- flutter build apk --release
- Chạy kiểm thử tự động:
- flutter test

Project Principles

- Cloud-first architecture: Ứng dụng được xây dựng hoàn toàn trên Supabase cloud.
- Realtime updates: Mọi thay đổi dữ liệu được đồng bộ tức thời thông qua subscription.
- Security by design: Mọi thao tác được bảo vệ bởi RLS policies.
- Separation of Concerns: Logic Flutter và Supabase SDK tách biệt rõ ràng.
- Scalability: Database và API sẵn sàng mở rộng khi người dùng tăng.
- Maintainability: Cấu trúc code rõ ràng, dễ debug và tái sử dụng.

3.2 Cấu trúc dự án

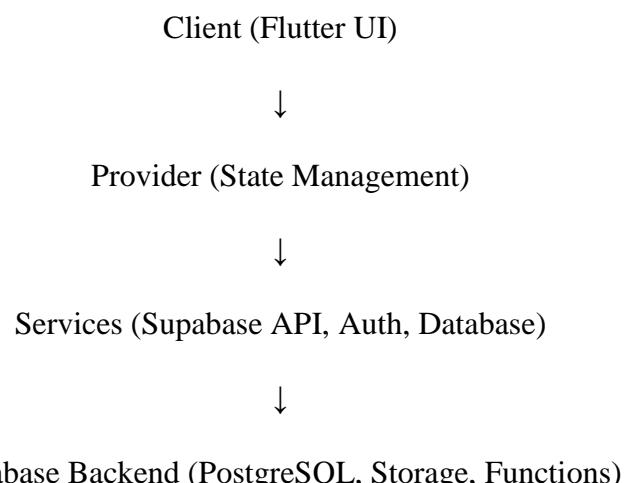
Kiến trúc tổng thể

Hệ thống Phenikaa Connect được phát triển theo kiến trúc MVVM (Model–View–ViewModel) kết hợp Provider Pattern để quản lý trạng thái. Ứng dụng được xây dựng bằng Flutter SDK, có khả năng triển khai đa nền tảng (Android, iOS, Web, Desktop) và kết nối với Supabase làm backend.

Mô hình tổng thể của hệ thống gồm ba tầng chính:

- Client Tier: Giao diện người dùng Flutter (UI/UX) hiển thị trên thiết bị di động hoặc trình duyệt web.
- Application Tier: Tầng logic nghiệp vụ (Business Logic Layer), bao gồm ViewModels, Providers và Services (API integration với Supabase).
- Data Tier: Cơ sở dữ liệu PostgreSQL được quản lý bởi Supabase, cùng với các dịch vụ Authentication, Storage và Realtime Subscriptions.

Kiến trúc hệ thống



Cấu trúc thư mục tổng quan

Thư mục chính của dự án được tổ chức theo tiêu chuẩn Flutter, đồng thời chia tách logic rõ ràng để thuận tiện cho phát triển, mở rộng và bảo trì lâu dài.

PhenikaaConnect/

```
├── android/
│   ├── app/
│   │   ├── src/
│   │   │   ├── debug/
│   │   │   ├── main/
│   │   │   └── profile/
│   │   └── build.gradle.kts
│   ├── gradle/
│   │   └── wrapper/
│   ├── .gitignore
│   ├── build.gradle.kts
│   ├── gradle.properties
│   └── settings.gradle.kts
└── ios/
    ├── Flutter/
    ├── Runner/
    ├── Runner.xcodeproj/
    ├── Runner.xcworkspace/
    ├── RunnerTests/
    ├── .gitignore
    └── Podfile
lib/
├── config/
│   └── supabase_config.dart
├── constants/
│   ├── app_constants.dart
│   └── app_theme.dart
├── models/
│   ├── course.dart
│   ├── event.dart
│   ├── post.dart
│   └── user.dart
├── providers/
│   └── app_provider.dart
└── screens/
    ├── academic_screen.dart
    ├── admin_announcement_detail_screen.dart
    ├── admin_announcement_form_screen.dart
    ├── admin_club_detail_screen.dart
    ├── admin_club_form_sheet.dart
    ├── admin_club_management_screen.dart
    └── admin_dashboard_screen.dart
```

```
    └── admin_event_detail_screen.dart
    └── admin_event_form_sheet.dart
    └── admin_event_management_screen.dart
    └── admin_user_detail_screen.dart
    └── admin_user_form_sheet.dart
    └── admin_user_management_screen.dart
    └── announcement_detail_screen.dart
    └── announcements_screen.dart
    └── auth_wrapper.dart
    └── campus_screen.dart
    └── club_activity_detail_screen.dart
    └── club_detail_screen.dart
    └── club_event_detail_screen.dart
    └── club_leader_activities_screen.dart
    └── club_leader_activity_form_sheet.dart
    └── club_leader_event_form_sheet.dart
    └── club_leader_events_screen.dart
    └── club_leader_members_screen.dart
    └── club_leader_post_form_sheet.dart
    └── club_leader_posts_screen.dart
    └── club_leader_workspace_screen.dart
    └── comments_screen.dart
    └── edit_profile_screen.dart
    └── event_detail_screen.dart
    └── home_screen.dart
    └── image_viewer_screen.dart
    └── login_screen.dart
    └── main_screen.dart
    └── post_detail_screen.dart
    └── profile_screen.dart
    └── question_detail_screen.dart
    └── signup_screen.dart
    └── social_screen.dart

    └── services/
        └── admin_service.dart
        └── club_leader_service.dart
        └── group_reminder_service.dart
        └── supabase_service.dart

    └── widgets/
        └── common_widgets.dart
        └── question_form_sheet.dart

    └── main.dart

└── linux/
    └── flutter/
    └── runner/
        └── .gitignore
```

```
└── CMakeLists.txt  
└── macos/  
    ├── Flutter/  
    ├── Runner/  
    ├── Runner.xcodeproj/  
    ├── Runner.xcworkspace/  
    ├── RunnerTests/  
    ├── Podfile  
    └── Podfile.lock  
    └── .gitignore  
  
└── test/  
    ├── user_role_test.dart  
    └── widget_test.dart  
  
└── web/  
    ├── icons/  
    │   └── favicon.png  
    ├── index.html  
    └── manifest.json  
  
└── windows/  
  
└── .gitignore  
└── .metadata  
└── pubspec.lock  
└── pubspec.yaml
```

Nguyên tắc phát triển dự án:

- Separation of Concerns: Phân tách rõ giữa logic, UI, và dữ liệu.
- Reusability: Các widget tái sử dụng được đặt trong widgets/.
- Configuration Management: Cấu hình tập trung tại constants/ và config/.
- Code Readability: Áp dụng conventions và linting (theo analysis_options.yaml).
- Scalability: Hỗ trợ mở rộng module (ví dụ: thêm “Dormitory Service”, “Library Info” mà không ảnh hưởng đến core system).

3.3 Thuật toán sử dụng

Hệ thống Phenikaa Connect App sử dụng các quy trình nghiệp vụ và cơ chế Real-time chính được triển khai ở tầng Backend (Supabase Database/Functions) và tích hợp qua lớp AppProvider để đảm bảo tính toàn vẹn và cập nhật dữ liệu.

3.3.1 Cơ chế Tăng/Giảm Counter Tự động (Real-time Counters)

Mục đích:

- Đảm bảo tính nhất quán và cập nhật tức thời các chỉ số tương tác (ví dụ: số lượng Likes cho bài đăng, số lượng Attendees cho sự kiện) mà không cần truy vấn phức tạp từ ứng dụng.

Phương pháp 1: Tính theo Database Trigger và Function

1. Bước 1: Nhận Yêu cầu & Thao tác Database

- Hệ thống nhận yêu cầu thao tác từ user thông qua AppProvider và SupabaseService.
- Thao tác Insert/Delete được thực hiện trên bảng liên quan (ví dụ: bảng post_likes hoặc event_attendees).

2. Bước 2: Kích hoạt Trigger

- Sau khi thao tác Insert/Delete thành công, một Database Trigger (được định nghĩa trong supabase_schema.sql) trên bảng post_likes sẽ được tự động kích hoạt.

3. Bước 3: Gọi Function Tính toán

- Trigger gọi một SQL Function (được định nghĩa trong supabase_functions.sql).
- Function thực hiện việc Tăng (+1) hoặc Giảm (-1) trực tiếp giá trị cột counter (ví dụ: likes_count) trong bảng chính (posts hoặc events).

4. Bước 4: Cập nhật Real-time

- Supabase Real-time Engine phát hiện thay đổi trên bảng chính.
- Ứng dụng Frontend (Flutter) đang đăng ký theo dõi (subscribe) sẽ nhận được cập nhật tức thời và tự động refresh UI.

3.3.2 Thuật toán Xác thực và Ủy quyền (Authentication & Authorization)

Mục đích:

- Xác minh danh tính người dùng (Authentication).
- Đảm bảo người dùng chỉ có thể thực hiện các thao tác được phép (Authorization).

Phương pháp 2: Kết hợp JWT và Row Level Security (RLS)

1. Bước 1: Đăng nhập & Cấp JWT

- User gọi appProvider.signIn(email, password).
- Supabase xác minh và cấp một JSON Web Token (JWT) cho user nếu thành công. Token này được lưu trữ trong ứng dụng.

2. Bước 2: Gửi Yêu cầu & Xác thực JWT

- Khi user thực hiện một thao tác (ví dụ: createPost()), ứng dụng gửi yêu cầu API đến Supabase, đính kèm JWT trong Header.
- Supabase Backend giải mã JWT để xác định danh tính (user_id, role) của user hiện tại.

3. Bước 3: Áp dụng RLS Policies

- Sau khi user được xác thực, Supabase áp dụng các RLS Policies (được định nghĩa trong supabase_schema.sql) cho bảng đang được truy cập (ví dụ: bảng posts hoặc users).

4. Bước 4: Kiểm tra Điều kiện RLS

- RLS Policy kiểm tra các điều kiện ủy quyền, ví dụ:
 - Policy SELECT: Cho phép xem thông tin user nếu (current_user_id() = id) hoặc (is_admin = true).
 - Policy INSERT: Cho phép tạo bài đăng nếu user đã được xác thực (authenticated = true).

5. Bước 5: Phản hồi

- Nếu Policy cho phép, thao tác Database được thực hiện.
- Nếu Policy từ chối, yêu cầu bị chặn ngay ở tầng Database với lỗi Permission Denied.

3.3.3 Quy trình CRUD cho Tính năng Cộng đồng (Post/Question/Event)

Mục đích:

- Chuẩn hóa các bước thao tác cơ bản (Tạo, Đọc, Cập nhật, Xóa) cho các đối tượng cốt lõi của ứng dụng.

Phương pháp 3: Tích hợp Lớp Dịch vụ và Xử lý Lỗi

1. Bước 1: Nhận Dữ liệu Đầu vào

- AppProvider nhận dữ liệu đầu vào đã được validate (ví dụ: nội dung không trống) từ UI.

2. Bước 2: Gọi Dịch vụ Backend

- AppProvider gọi phương thức tương ứng trong SupabaseService (ví dụ: supabaseService.createPost(data)).

3. Bước 3: Thực thi Query Database

- SupabaseService xây dựng và thực thi query tới PostgreSQL thông qua Supabase Client.

4. Bước 4: Kiểm tra Database và RLS

- Database kiểm tra ràng buộc Foreign Key, Data Types và áp dụng RLS Policies.

5. Bước 5: Xử lý Lỗi (Error Handling)

- SupabaseService nhận phản hồi và xử lý lỗi:
 - Nếu Lỗi Database/Network: Throw Exception.
 - Nếu Thành công: Parse dữ liệu và trả về.

6. Bước 6: Cập nhật State

- AppProvider nhận dữ liệu thành công, cập nhật Local State (ví dụ: thêm post mới vào danh sách posts).

7. Bước 7: Phản hồi

- UI hiển thị dữ liệu mới hoặc thông báo lỗi, hoàn thành vòng lặp nghiệp vụ.

3.4 Kiểm thử hệ thống

TC001: Kiểm thử Đăng nhập và Quản lý Tài khoản

ID	Vai trò	Chức năng Kiểm thử	Bước Thực hiện	Kết quả Mong đợi	Bảng/Policy liên quan
TC001.1	User	Đăng ký thành công	1. Nhập email chưa tồn tại. 2. Click "Đăng ký".	Tài khoản được tạo thành công trong Supabase Auth, bản ghi user được tạo trong bảng users với role = 'user'.	users, Supabase Auth
TC001.2	User	Đăng nhập và Lấy Session	1. Đăng nhập thành công. 2. Gọi API getCurrentUser().	Trả về thông tin user hiện tại chính xác. JWT được sử dụng hợp lệ.	users, Supabase Auth
TC001.3	User	Cập nhật Profile (Hợp lệ)	Thay đổi name, bio (nếu có), phone, avatar_url. Click "Lưu".	Thông tin được cập nhật thành công trong bảng users và hiển thị đúng trên giao diện.	users
TC001.5	User	Cập nhật Profile - RLS Thất bại	User A cố gắng cập nhật email, student_id, hoặc role của User B (qua API backend).	Thao tác bị từ chối ở tầng database (Permission Denied/RLS Policy) do không có quyền sửa bản ghi của người khác.	users, RLS Policy
TC007.1	Admin	Khóa/Mở khóa Tài khoản	Admin cập nhật is_locked = true cho User A. User A cố gắng Đăng nhập.	Admin: Cập nhật thành công. User A: Đăng nhập bị từ chối hoặc bị đăng xuất (dựa trên logic Auth).	users

ID	Vai trò	Chức năng Kiểm thử	Bước Thực hiện	Kết quả Mong đợi	Bảng/Policy liên quan
TC009.2	Data Integrity	Đăng ký Student ID đã tồn tại	Có gắng Đăng ký với một student_id đã có trong DB.	Thao tác bị từ chối do Unique Constraint trên cột student_id của bảng users.	users

Kiểm thử xác thực người dùng là bước đầu tiên và quan trọng nhất trong quy trình kiểm thử, vì đây là nền tảng cho tất cả các chức năng khác của ứng dụng. Khi người dùng thực hiện đăng ký tài khoản mới, hệ thống sẽ kiểm tra tính duy nhất của email và mã số sinh viên trước khi tạo tài khoản trong Supabase Auth. Sau khi tài khoản được tạo thành công, hệ thống sẽ tự động tạo bản ghi tương ứng trong bảng users với đầy đủ thông tin cá nhân mà người dùng đã cung cấp. Quá trình này cần được kiểm tra để đảm bảo dữ liệu được đồng bộ chính xác giữa hệ thống xác thực và bảng người dùng.

Khi người dùng đăng nhập thành công, hệ thống sẽ tạo phiên đăng nhập và cấp phát JWT token. Việc gọi API getCurrentUser sau khi đăng nhập sẽ trả về thông tin người dùng hiện tại một cách chính xác, bao gồm tên, email, mã số sinh viên và các thông tin khác. JWT token được sử dụng để xác thực các yêu cầu tiếp theo và cần được kiểm tra để đảm bảo tính hợp lệ và tính bảo mật.

Chức năng cập nhật profile cho phép người dùng thay đổi thông tin cá nhân như tên hiển thị và mô tả bản thân. Khi người dùng thực hiện thay đổi và nhấn nút lưu, hệ thống sẽ cập nhật thông tin trong bảng users và phản ánh ngay lập tức trên giao diện người dùng. Quá trình này cần được kiểm tra để đảm bảo dữ liệu được cập nhật đúng và không có sự không nhất quán.

Chu trình đăng nhập và đăng xuất được kiểm thử để đảm bảo tính toàn vẹn của phiên làm việc. Khi người dùng đăng nhập thành công, họ có thể truy cập các chức năng yêu cầu xác thực. Khi đăng xuất, phiên sẽ được hủy bỏ hoàn toàn và người dùng không thể thực hiện các thao tác yêu cầu xác thực nữa cho đến khi đăng nhập lại.

TC002: Kiểm thử Bài đăng Mạng xã hội

ID	Vai trò	Chức năng Kiểm thử	Bước Thực hiện	Kết quả Mong đợi	Bảng/Policy liên quan
TC002.1	User	Tạo và Đọc Bài đăng	1. Tạo Post P1 (goicreatePost) 2. Kiểm tra P1 xuất hiện trong danh sách (getPosts).	Post được tạo thành công, có thể đọc được bởi tất cả user.	posts
TC002.2	User	Like/Unlike (Counter)	1. User A Like Post P1. 2. User A Unlike Post P1.	Lần 1: Bản ghi tạo trong post_likes, posts.likes_count tự động tăng (+1). Lần 2: Bản ghi bị xóa, posts.likes_count tự động giảm (-1) (Qua Trigger/Real-time).	post_likes, posts
TC002.6	User	Ngăn chặn Double Like	User A Like Post P1, sau đó Like P1 lần nữa.	Lần 2 bị từ chối/bỏ qua do Unique Constraint trên (post_id, user_id) của bảng post_likes.	post_likes
TC002.3	User	Bình luận và Cập nhật Counter	User A bình luận C1 Post P1.	Bản ghi tạo trong comments, posts.comments_count tự động tăng (+1) (Qua Trigger/Real-time).	comments, posts
TC002.4	User	Sửa Bài đăng - RLS Thất bại	User B cố gắng cập nhật bài đăng do User A tạo.	Thao tác bị từ chối ở tầng database (Permission Denied) do vi phạm RLS Policy (chỉ owner mới được Sửa/Xóa).	posts, RLS Policy
TC006.2	Data Integrity	FK Constraint (Likes)	Cố gắng tạo một bản ghi post_likes với post_id không tồn tại trong bảng posts.	Thao tác bị từ chối ở tầng database do lỗi Foreign Key Constraint.	post_likes, posts

Chức năng tạo và đọc bài đăng là một trong những tính năng cốt lõi của ứng dụng. Khi người dùng tạo một bài đăng mới thông qua hàm createPost, hệ thống sẽ lưu bài đăng vào bảng posts với đầy đủ thông tin bao gồm nội dung, người tạo và thời gian tạo. Sau khi bài đăng được tạo, việc gọi hàm getPosts sẽ trả về danh sách các bài đăng bao gồm bài đăng mới tạo. Bài đăng cần được kiểm tra để đảm bảo có thể đọc được bởi tất cả người dùng hoặc theo đúng chính sách RLS đã được thiết lập.

Chức năng thích và bỏ thích bài viết được thiết kế để cập nhật real-time, mang lại trải nghiệm mượt mà cho người dùng. Khi người dùng thích một bài viết lần đầu, hệ thống sẽ tạo bản ghi trong bảng post_likes và tự động tăng số lượng lượt thích trong bảng posts thông qua trigger hoặc stored procedure. Khi người dùng bỏ thích, bản ghi sẽ được xóa khỏi bảng post_likes và số lượng lượt thích sẽ tự động giảm. Việc cập nhật này cần diễn ra real-time để người dùng trên các thiết bị khác nhau có thể thấy ngay sự thay đổi mà không cần làm mới trang.

Chức năng bình luận cho phép người dùng tương tác sâu hơn với các bài đăng. Khi một người dùng bình luận trên một bài đăng, hệ thống sẽ tạo bản ghi trong bảng comments với thông tin về bài đăng, người bình luận và nội dung bình luận. Đồng thời, số lượng bình luận trong bảng posts sẽ tự động tăng lên một cách real-time, đảm bảo tính nhất quán của dữ liệu và trải nghiệm người dùng tốt.

Kiểm thử bảo mật RLS là một phần quan trọng để đảm bảo người dùng chỉ có thể thực hiện các thao tác mà họ có quyền. Khi một người dùng khác cố gắng cập nhật bài đăng không phải do mình tạo, hệ thống sẽ từ chối thao tác này ở tầng database thông qua các chính sách RLS. Việc từ chối này cần được xử lý một cách rõ ràng để người dùng hiểu lý do tại sao thao tác không thành công.

TC003: Kiểm thử Q&A và Lịch học

ID	Vai trò	Chức năng Kiểm thử	Bước Thực hiện	Kết quả Mong đợi	Bảng/Policy liên quan
TC001.6	User	Tạo Lịch học Cá nhân	User A thêm bản ghi mới vào class_schedules (môn học, ngày, giờ, phòng).	Bản ghi được tạo thành công với user_id chính xác của User A.	class_schedules
TC001.7	User	Đọc Lịch học - RLS Thất bại	User A cố gắng truy cập lịch học của User B (class_schedules).	Thao tác bị từ chối/Chỉ trả về lịch học của User A (RLS hoạt động chính xác trên user_id).	class_schedules, RLS Policy

ID	Vai trò	Chức năng Kiểm thử	Bước Thực hiện	Kết quả Mong đợi	Bảng/Policy liên quan
TC003.1	User	Tạo và Đọc Câu hỏi	1. Tạo Câu hỏi Q1 (createQuestion). 2. Lấy danh sách câu hỏi (getQuestions).	Câu hỏi được lưu và hiển thị thành công.	questions
TC003.4	User	Đặt Solved - RLS Thất bại	User B cố gắng set solved = true cho Question Q1 do User A tạo.	Thao tác bị từ chối (RLS: Chỉ người tạo câu hỏi mới được đánh dấu solved).	questions, RLS Policy
TC003.5	User	Tạo Reply - Is Solution	User B tạo Reply R1 cho Q1, set is_solution = true.	Bản ghi R1 được lưu thành công trong question_replies.	question_replies

Diễn đàn hỏi đáp cho phép sinh viên đặt câu hỏi và nhận câu trả lời từ cộng đồng. Khi người dùng tạo một câu hỏi mới thông qua hàm createQuestion, câu hỏi sẽ được lưu vào bảng questions với đầy đủ thông tin về môn học liên quan, tiêu đề, nội dung và người đặt câu hỏi. Hàm getQuestions sẽ trả về danh sách các câu hỏi được sắp xếp theo thời gian, giúp người dùng dễ dàng tìm thấy các câu hỏi mới nhất hoặc liên quan đến môn học mà họ quan tâm.

Khi một người dùng trả lời câu hỏi, hệ thống sẽ tạo bản ghi trong bảng question_replies với thông tin về câu hỏi, người trả lời và nội dung câu trả lời. Câu trả lời này sẽ được hiển thị cùng với câu hỏi để các sinh viên khác có thể tham khảo và học hỏi.

Chức năng nhóm học tập giúp sinh viên kết nối và học tập cùng nhau. Khi một người dùng tạo nhóm học tập mới thông qua hàm createStudyGroup, nhóm sẽ được lưu vào bảng study_groups với thông tin về tên nhóm, môn học liên quan, thời gian và địa điểm họp. Người tạo nhóm sẽ tự động được thêm vào bảng study_group_members với vai trò là người tạo nhóm. Hàm getStudyGroups sẽ trả về danh sách các nhóm học tập có sẵn, giúp sinh viên dễ dàng tìm và tham gia nhóm phù hợp.

TC004: Kiểm thử Sự kiện và CLB

ID	Vai trò	Chức năng Kiểm thử	Bước Thực hiện	Kết quả Mong đợi	Bảng/Policy liên quan
TC004.3	User	Đọc Danh sách CLB	Lấy danh sách Câu lạc bộ (clubs).	Trả về danh sách các CLB có status = 'approved' và active = true.	clubs
TC004.5	User	Tham gia CLB (Pending)	User A yêu cầu tham gia Club C1.	Bản ghi tạo trong club_members với user_id của User A và status = 'pending'.	club_members
TC007.3	Admin	Phê duyệt CLB	Admin cập nhật status của Club C1 từ 'pending' sang 'approved'.	Cập nhật thành công. CLB C1 hiển thị/được phép hoạt động. Trigger/Function: Có thể cập nhật members_count nếu tự động chấp nhận.	clubs
TC004.1	User	Tạo và Tham gia Sự kiện	1. Tạo Sự kiện E1. 2. User A Tham gia sự kiện (joinEvent).	Sự kiện được tạo. Bản ghi tạo trong event_attendees, events.attendees_count tự động tăng (+1) (Trigger).	events, event_attendees
TC004.4	User	Giới hạn Tham dự (Full)	Event E1 có max_attendees = 10. 10 User đã tham gia. User thứ 11 cố gắng tham gia.	Bị từ chối với thông báo lỗi (Constraint/Logic ở tầng ứng dụng/DB Trigger).	events, event_attendees
TC004.6	User	Tạo Post CLB - RLS Thất bại	User B (không phải thành viên CLB C1) cố gắng tạo club_posts cho C1.	Thao tác bị từ chối (RLS Policy: Chỉ thành viên của CLB mới được phép đăng bài).	club_posts, RLS Policy

Quản lý sự kiện là một tính năng quan trọng để sinh viên có thể tham gia các hoạt động trong trường. Khi một người dùng tạo sự kiện mới thông qua hàm createEvent, sự kiện sẽ được lưu vào bảng events với đầy đủ thông tin về tiêu đề, mô tả, thời gian, địa điểm và danh mục. Khi một người dùng khác tham gia sự kiện thông qua hàm joinEvent, hệ thống sẽ tạo bản ghi trong bảng event_attendees và tự động tăng số lượng người tham gia trong bảng events. Việc cập nhật này diễn ra real-time để người dùng có thể thấy ngay số lượng người đã đăng ký tham gia. Khi người dùng quyết định rời sự kiện thông qua hàm leaveEvent, hệ thống sẽ xóa bản ghi khỏi bảng event_attendees và tự động giảm số lượng người tham gia. Quá trình này cũng diễn ra real-time để đảm bảo thông tin luôn chính xác và cập nhật.

Chức năng câu lạc bộ cho phép sinh viên khám phá và tham gia các tổ chức sinh viên. Hàm getClubs sẽ trả về danh sách các câu lạc bộ có sẵn với thông tin về tên, mô tả, danh mục và số lượng thành viên. Sinh viên có thể xem chi tiết và tham gia các câu lạc bộ phù hợp với sở thích và năng lực của mình.

TC005: Kiểm thử Nhóm học tập và Thông báo

ID	Vai trò	Chức năng Kiểm thử	Bước Thực hiện	Kết quả Mong đợi	Bảng/Policy liên quan
TC003.7	User	Rời Nhóm học tập (Counter)	User A rời Nhóm G1 (xóa bản ghi khỏi study_group_members).	Bản ghi bị xóa. Cột members_count trong study_groups tự động giảm (-1).	study_group_members , study_groups
TC003.6	User	Đếm Thành viên Nhóm	Một User mới tham gia G1 (tạo bản ghi trong study_group_members).	Bản ghi tạo thành công. Cột member_count trong study_groups tự động tăng (+1).	study_group_members , study_groups

ID	Vai trò	Chức năng Kiểm thử	Bước Thực hiện	Kết quả Mong đợi	Bảng/Policy liên quan
TC005.3	User	Đọc Thông báo	Lấy danh sách các Thông báo (announcements).	Trả về danh sách các thông báo thỏa mãn điều kiện target_audience (nếu có) hoặc tất cả.	announcements
TC008.1	Admin	Tạo Thông báo Chính thức	Admin tạo Announcement A1 (title, content, priority).	Bản ghi được tạo thành công trong announcements với created_by là id của Admin.	announcements

Hệ thống cung cấp tính năng Nhóm Học Tập, cho phép sinh viên tạo và quản lý các nhóm theo môn học. Thông tin chi tiết của nhóm được lưu trữ trong bảng. Việc theo dõi thành viên được thực hiện thông qua bảng liên kết, nơi các bản ghi được tạo khi người dùng tham gia nhóm. Tính năng cốt lõi là cơ chế tự động theo dõi và cập nhật số lượng thành viên: khi một người dùng mới gia nhập ngược lại, khi một người dùng rời nhóm. Cơ chế này đảm bảo dữ liệu về quy mô nhóm luôn được duy trì chính xác và theo thời gian thực.

Hệ thống Thông Báo phục vụ mục đích gửi thông tin quan trọng từ nhà trường hoặc Ban quản trị (Admin) đến sinh viên. Các bản ghi thông báo được lưu trữ trong bảng thông báo. Về quyền hạn quản lý, chỉ Admin mới có quyền tạo Thông báo Chính thức bao gồm các thông tin như tiêu đề, nội dung và mức độ ưu tiên, với ID của Admin được ghi lại tại trường khởi tạo. Chức năng đọc thông báo trả về danh sách đã lưu, được sắp xếp theo thời gian tạo giảm dần để hiển thị thông báo mới nhất ở vị trí đầu. Danh sách này có thể được lọc theo điều kiện, đảm bảo thông báo chỉ hiển thị đến đúng đối tượng người nhận. Sinh viên có thể xem chi tiết từng thông báo, và hệ thống theo dõi trạng thái đã đọc để quản lý thông báo hiệu quả hơn.

TC006: Kiểm thử Bảo mật và Độ tin cậy (Security & Troubleshooting)

ID	Vai trò	Chức năng Kiểm thử	Bước Thực hiện	Kết quả Mong đợi	Bảng/Policy liên quan
TC006.1	User	Data Validation: Nội dung Rỗng	Có gắng tạo Post với content rỗng.	Thao tác bị từ chối ở tầng ứng dụng (client-side validation) hoặc tầng database (NOT NULL Constraint).	posts
TC006.4	User	Lỗi Authentication (Chưa đăng nhập)	User chưa đăng nhập có gắng gọi API createPost() hoặc joinEvent().	API bị từ chối với lỗi Authentication required (Status 401/403) do RLS Policy yêu cầu người dùng phải là authenticated.	Toàn bộ bảng yêu cầu Auth, RLS Policy
TC009.1	Admin	FK Constraint: Xóa User (Cascade)	User A có các bản ghi liên quan (Post, Like, Comment, Attendee). Admin xóa User A.	Nếu Policy là ON DELETE RESTRICT: Bị từ chối. Nếu Policy là ON DELETE CASCADE (thường dùng cho các bảng liên kết): Các bản ghi phụ thuộc (Post, Like, Comment) bị xóa tự động.	users, posts, post_likes, v.v.
TC009.3	User	FK Constraint: Xóa Post/Update Post	User A xóa Post P1. Kiểm tra các Comment và Like của P1.	Các bản ghi Comment và Like liên quan phải bị xóa tự động (do ON DELETE CASCADE trên FK).	posts, comments, post_likes

Kiểm thử validation dữ liệu đảm bảo hệ thống từ chối các dữ liệu không hợp lệ trước khi chúng được gửi đến database. Khi người dùng cố gắng tạo bài đăng với nội dung rỗng hoặc các trường bắt buộc không được điền, hệ thống sẽ hiển thị thông báo lỗi và từ chối thao tác. Việc kiểm tra này diễn ra ở tầng ứng dụng để giảm tải cho database và cải thiện trải nghiệm người dùng.

Kiểm thử ràng buộc khóa ngoại đảm bảo tính toàn vẹn của dữ liệu trong database. Khi hệ thống

có gắng tạo một bản ghi trong bảng post_likes với post_id không tồn tại trong bảng posts, database sẽ từ chối thao tác này và trả về lỗi Foreign Key Constraint. Điều này đảm bảo dữ liệu luôn nhất quán và không có các tham chiếu không hợp lệ.

Xử lý lỗi kết nối là một phần quan trọng để đảm bảo ứng dụng hoạt động ổn định ngay cả khi gặp sự cố mạng. Khi kết nối mạng bị ngắt và người dùng thử gọi các hàm như getPosts, ứng dụng cần hiển thị thông báo lỗi thân thiện thay vì bị crash. Điều này chứng tỏ hệ thống error handling đã hoạt động đúng và ứng dụng có khả năng phục hồi tốt.

Kiểm thử bảo mật xác thực đảm bảo các chức năng yêu cầu xác thực chỉ có thể được thực hiện bởi người dùng đã đăng nhập. Khi một người dùng chưa đăng nhập có gắng gọi các API như createPost hoặc joinEvent, hệ thống sẽ từ chối yêu cầu với thông báo lỗi yêu cầu xác thực. Việc này được thực hiện thông qua kiểm tra JWT token và các chính sách RLS ở tầng database.

Quy trình thực hiện kiểm thử

Quy trình kiểm thử được thực hiện theo các giai đoạn có hệ thống để đảm bảo tất cả các chức năng được kiểm tra kỹ lưỡng. Giai đoạn chuẩn bị bao gồm việc thiết lập môi trường phát triển và chuẩn bị dữ liệu kiểm thử. Các tài khoản người dùng ở các trạng thái khác nhau được tạo sẵn, bao gồm người dùng đã đăng nhập, người dùng chưa đăng nhập, người dùng sở hữu bài viết và người dùng không sở hữu bài viết. Dữ liệu mẫu bao gồm bài đăng, sự kiện, người dùng và câu lạc bộ được import vào database thông qua các SQL scripts để tạo môi trường kiểm thử giống với môi trường thực tế.

Việc chuẩn bị môi trường backend bao gồm việc xác nhận Supabase Project đang hoạt động và các thông tin như URL và Anon Key đã được cập nhật chính xác trong file cấu hình supabase_config.dart. Các chính sách RLS cho các bảng chính như posts, events và users cần được kiểm tra để đảm bảo chúng ở trạng thái ON và hoạt động đúng như mong đợi. Việc chuẩn bị các thiết bị và trình duyệt khác nhau giúp kiểm tra tính năng real-time đồng bộ trên nhiều nền tảng.

Giai đoạn thực thi kiểm thử được thực hiện theo thứ tự ưu tiên, bắt đầu với kiểm thử xác thực, tiếp theo là kiểm thử các thao tác CRUD, và cuối cùng là kiểm thử tính năng real-time và bảo mật. Mỗi chức năng được kiểm tra một cách cẩn thận để đảm bảo nó hoạt động đúng như đặc tả và không có lỗi tiềm ẩn.

Giai đoạn tài liệu hóa bao gồm việc ghi lại kết quả thực hiện cho từng trường hợp kiểm thử, bao gồm trạng thái Pass hoặc Fail và các chi tiết về lỗi nếu có. Các lỗi được báo cáo theo định dạng chuẩn với mã số bug, tiêu đề ngắn gọn, các bước để tái hiện lỗi, kết quả mong đợi và kết quả thực tế, mức độ ưu tiên và trạng thái xử lý.

Giai đoạn regression được thực hiện sau khi các lỗi đã được sửa bởi nhà phát triển. Trong giai đoạn này, các chức năng đã bị ảnh hưởng bởi việc sửa lỗi được kiểm tra lại để đảm bảo không có lỗi mới phát sinh và các chức năng khác vẫn hoạt động bình thường.

Quy trình kiểm thử được thiết lập nhằm đảm bảo tất cả các chức năng cốt lõi, đặc biệt là các tương tác Real-time và các chính sách bảo mật (RLS) của Supabase, hoạt động chính xác và ổn định.

Checklist trước khi Kiểm thử:

- Chuẩn bị data test: Chuẩn bị các tài khoản user ở các trạng thái khác nhau (đã đăng nhập, chưa đăng nhập, user sở hữu bài viết, user không sở hữu bài viết).
- Chuẩn bị Mock Data: Đảm bảo dữ liệu mẫu (Posts, Events, Users, Clubs) đã được import thành công qua các SQL scripts.
- Setup môi trường Backend: Xác nhận Supabase Project đang Active, URL và Anon Key đã được cập nhật chính xác trong code (supabase_config.dart).
- Kiểm tra RLS Policies: Xác nhận các chính sách RLS cho các bảng chính (posts, events, users) đang ở trạng thái ON và đã được kiểm tra sơ bộ.
- Setup browsers và devices test: Chuẩn bị các thiết bị/trình duyệt để kiểm tra tính năng Real-time đồng bộ.
- Chuẩn bị scenarios phức tạp: Thiết lập các kịch bản kiểm thử bảo mật (cố gắng cập nhật Post của người khác) và kịch bản Real-time đồng thời.

Quy trình thực hiện:

1. Preparation Phase: Setup môi trường phát triển (Dev/Staging) và chuẩn bị các Test Data cần thiết. Đảm bảo tất cả các dependencies đã được cài đặt và ứng dụng có thể build thành công.
2. Execution Phase: Thực hiện các Test Cases theo thứ tự ưu tiên: Authentication (TC001) → CRUD Operations (TC002, TC003, TC004) → Real-time & Security (TC005, TC006).
3. Documentation Phase: Ghi lại kết quả thực hiện cho từng Test Case (Pass/Fail) và ghi lại chi tiết các lỗi (Bug) phát hiện được theo format quy định.
4. Regression Phase: Sau khi các lỗi được Developer fix, thực hiện test lại các tính năng đã bị ảnh hưởng để đảm bảo không phát sinh lỗi mới.

Bug reporting format:

- ID: Mã số bug (Ví dụ: BUG_CONN_001)
- Title: Tiêu đề ngắn gọn (Ví dụ: Likes counter không tự động cập nhật Real-time)
- Steps to reproduce: Các bước tái hiện lỗi
- Expected result: Counter tự động tăng/giảm trên các thiết bị khác.
- Actual result: Counter chỉ tăng sau khi refresh thủ công.
- Priority: High/Medium/Low
- Status: Open/In Progress/Fixed/Closed

Test data chuẩn bị

Dữ liệu được chuẩn bị để bao quát các tình huống Authentication, tương tác cộng đồng, và kiểm tra quyền hạn.

- User accounts test:
 - Student 1: student1@st.phenikaa-uni.edu.vn (User thường, đã có Post/Event).
 - Student 2: student2@gmail.com (User thường, dùng để kiểm tra RLS và Real-time).
 - Invalid: invalid@yahoo.com (Dùng để kiểm tra Validation và Authentication thất bại).
- Dữ liệu Cộng đồng/Sự kiện test:
 - Post P1: Bài đăng có 5 Likes (dùng để test tăng/giảm Real-time).
 - Event E1: Sự kiện mở (dùng để test joinEvent/leaveEvent).
 - Chat Room C1: Phòng chat giữa Student 1 và Student 2.
- Dữ liệu Cấu hình:
 - Supabase URL/Key: Được cập nhật hợp lệ.

Quality assurance

Chất lượng mã nguồn được đảm bảo thông qua quy trình code review, giúp đảm bảo tính dễ đọc và tuân thủ các tiêu chuẩn và quy ước của Dart, Flutter và PostgreSQL. Mã nguồn cần được tài liệu hóa đầy đủ, đặc biệt là các lớp SupabaseService và các SQL Functions phức tạp, để giúp các nhà phát triển khác hiểu và bảo trì mã nguồn dễ dàng hơn.

Kiểm thử hiệu suất được thực hiện để đảm bảo ứng dụng có thể xử lý nhiều người dùng đồng thời mà không bị giảm hiệu suất. Việc mô phỏng 50 người dùng đồng thời thực hiện các thao tác như thích, bình luận và tham gia sự kiện giúp kiểm tra độ trễ của các bộ đếm real-time. Chỉ số hiệu suất quan trọng là phản hồi phải dưới 3 giây trong các điều kiện tải cao.

Tối ưu hóa truy vấn database được thực hiện bằng cách kiểm tra việc sử dụng các chỉ mục trên

các trường được truy vấn thường xuyên. Việc này giúp cải thiện hiệu suất của các truy vấn như getPosts, đảm bảo người dùng có thể tải danh sách bài đăng một cách nhanh chóng ngay cả khi có lượng dữ liệu lớn.

Giám sát sử dụng bộ nhớ giúp đảm bảo ứng dụng không tiêu tốn quá nhiều tài nguyên hệ thống. Khi xử lý lượng dữ liệu lớn như bảng tin có hơn 100 bài đăng, ứng dụng cần quản lý bộ nhớ một cách hiệu quả để tránh làm chậm thiết bị của người dùng.

Code quality:

- Code review process (hiện tại chưa có): Sẽ được triển khai để đảm bảo tính dễ đọc và tuân thủ Coding standards và conventions của Dart/Flutter và PostgreSQL/SQL.
- Documentation và comments: Yêu cầu đầy đủ cho các lớp SupabaseService và các SQL Functions phức tạp.

Performance testing:

- Load testing với nhiều concurrent users: Mô phỏng 50 user đồng thời thực hiện Like/Comment/Join Event để kiểm tra độ trễ của các Real-time Counters (KPI: Phản hồi dưới 3 giây).
- Database query optimization: Tối ưu hóa các query lấy Feed chính (getPosts) bằng cách kiểm tra việc sử dụng Indexes trên các trường được truy vấn thường xuyên.
- Memory usage monitoring: Theo dõi bộ nhớ sử dụng của ứng dụng Flutter khi xử lý lượng dữ liệu lớn (ví dụ: Feed có 100+ bài đăng).

Security testing:

- Authentication và authorization testing: Kiểm tra nghiêm ngặt tất cả các RLS Policies trên các bảng chính (TC002.4, TC006.4).
- SQL injection prevention: Thủ nghiệm input với các ký tự độc hại để đảm bảo Supabase/PostgreSQL ngăn chặn các lệnh SQL Injection.

Giám sát và ghi nhận

Hệ thống giám sát được triển khai để theo dõi sức khỏe của ứng dụng và backend Supabase một cách liên tục. Ghi nhận lỗi và theo dõi được thực hiện thông qua các công cụ như Sentry hoặc Firebase Crashlytics để theo dõi các lỗi ở phía Flutter và ghi lại các ngoại lệ từ SupabaseService. Việc này giúp phát hiện và khắc phục các lỗi một cách nhanh chóng.

Theo dõi hoạt động người dùng giúp hiểu cách người dùng sử dụng ứng dụng và tần suất sử dụng các API chính như createPost và joinEvent. Dữ liệu này có thể được sử dụng để tối ưu hóa trải nghiệm người dùng và ưu tiên phát triển các tính năng được sử dụng nhiều nhất.

Giám sát hiệu suất truy vấn giúp phát hiện các truy vấn chậm và các SQL Functions hoặc Triggers có thể cần được tối ưu hóa. Việc kiểm tra tính toàn vẹn dữ liệu định kỳ đảm bảo các bộ đếm tự động cập nhật không bị sai lệch và dữ liệu luôn nhất quán.

Hệ thống giám sát được triển khai để theo dõi sức khỏe ứng dụng và backend Supabase.

- Application monitoring:
 - Error logging và tracking: Sử dụng các công cụ như Sentry hoặc Firebase Crashlytics để theo dõi lỗi ở phía Flutter (Frontend) và ghi lại các ngoại lệ từ SupabaseService.
 - User activity monitoring: Theo dõi tần suất sử dụng các API chính (createPost, joinEvent).
- Database monitoring:
 - Query performance: Theo dõi hiệu suất truy vấn của các SQL Functions và Triggers để tìm kiếm các truy vấn chậm.
 - Data integrity checks: Định kỳ kiểm tra tính toàn vẹn của dữ liệu và đảm bảo Auto-update counters không bị sai lệch.

Kế hoạch cải thiện

Ngắn hạn (1-3 tháng):

- Triển khai kiểm thử unit: Viết unit tests cho các hàm nghiệp vụ trong lớp AppProvider và SupabaseService.
- Setup automated testing environment: Thiết lập môi trường chạy Integration Testing cơ bản cho các luồng Authentication và CRUD.

Trung hạn (3-6 tháng):

- Hoàn thiện integration testing: Xây dựng các bài kiểm thử tự động cho toàn bộ luồng nghiệp vụ.
- Implement CI/CD pipeline: Tích hợp kiểm thử tự động vào quy trình CI/CD (GitHub Actions/GitLab CI) để đảm bảo chất lượng liên tục.
- Setup monitoring và alerting: Thiết lập cảnh báo tự động khi Supabase gặp lỗi kết nối

hoặc hiệu suất truy vấn giảm.

Dài hạn (6-12 tháng):

- Full end-to-end testing automation: Tự động hóa hoàn toàn các kịch bản người dùng cuối (E2E Testing).
- Performance optimization: Tối ưu hóa các API hỗ trợ Advanced search và File upload.
- Security audit và penetration testing: Thực hiện kiểm tra an ninh toàn diện cho cả ứng dụng Flutter và cấu hình Backend Supabase.

3.5 Triển khai hệ thống

Giới thiệu

Sau khi hoàn thành phát triển và kiểm thử, hệ thống Phenikaa Connect App được triển khai lên môi trường Production. Khác với mô hình triển khai truyền thống, Phenikaa Connect App sử dụng mô hình BaaS (Backend as a Service) với Supabase là nền tảng cốt lõi, giúp đơn giản hóa quy trình deployment và đảm bảo tính ổn định, khả năng mở rộng tối đa.

Mục tiêu chính của giai đoạn này là đưa ứng dụng di động Flutter lên các App Store và kết nối nó với Supabase Backend Production đã được cấu hình bảo mật.

Kiến trúc triển khai

Kiến trúc triển khai của Phenikaa Connect App tuân theo mô hình Mobile ↔ Managed Cloud Backend, với luồng dữ liệu chính như sau:

Users → Mobile App (Flutter) → HTTPS → Supabase Backend (Managed PostgreSQL, Auth, Real-time)

Trong đó:

Users (Người dùng cuối)

- Vai trò: Là đối tượng truy cập và tương tác với các tính năng của ứng dụng (Community Feed, Events, Q&A).
- Chi tiết: Các request của người dùng (như đăng nhập, tạo bài đăng, Like) được khởi tạo từ thiết bị di động của họ.

Mobile App (Flutter)

- Vai trò: Là giao diện người dùng (Frontend) và chứa logic kinh doanh phía client.

- Chi tiết: Ứng dụng được xây dựng bằng Flutter, quản lý trạng thái thông qua lớp AppProvider và giao tiếp với Backend thông qua lớp SupabaseService. Ứng dụng chịu trách nhiệm gửi JWT (JSON Web Token) trong mỗi request API để xác thực.

HTTPS (Giao thức Bảo mật)

- Vai trò: Đảm bảo tính bảo mật và mã hóa dữ liệu khi truyền giữa ứng dụng di động và Backend.
- Chi tiết: Tất cả các giao tiếp (API Calls) đều sử dụng giao thức HTTPS qua cổng 443, đảm bảo dữ liệu (như mật khẩu, nội dung bài đăng) không bị nghe lén.

Supabase Backend (Managed PostgreSQL, Auth, Real-time)

- Vai trò: Nền tảng Backend-as-a-Service (BaaS) quản lý toàn bộ dịch vụ backend.
- Chi tiết: Đây là trái tim của hệ thống, cung cấp các dịch vụ sau:
 - Managed PostgreSQL: Nơi lưu trữ toàn bộ dữ liệu (bảng posts, users, events, v.v.). Chịu trách nhiệm thực thi SQL Functions và Triggers cho cơ chế Auto-update Counters.
 - Authentication: Xử lý luồng đăng nhập/đăng ký và tạo/xác thực JWT.
 - Real-time Engine: Cung cấp kênh WebSocket để phát các cập nhật dữ liệu tức thời (như khi có Like mới, số lượng sẽ tự động tăng trên giao diện người dùng).

Platform và Services

Kiến trúc triển khai của Phenikaa Connect App sử dụng mô hình BaaS (Backend as a Service) với các dịch vụ quản lý hàng đầu để tối ưu hóa việc phát triển ứng dụng di động.

- Supabase (Backend & Database Hosting):
 - Cung cấp toàn bộ Backend Services dưới dạng managed service.
 - Bao gồm PostgreSQL Database hiệu suất cao.
 - Hỗ trợ Authentication (Đăng ký/Đăng nhập) tích hợp sẵn.
 - Cung cấp Real-time Engine cho các tính năng cập nhật tức thời (ví dụ: Likes, Messages).
 - Đảm bảo High Availability và Automatic Backups cho database.
 - Thực thi Row Level Security (RLS) để kiểm soát quyền truy cập ở tầng Database.

- Flutter/Dart (Application Development):
 - Công nghệ phát triển ứng dụng di động đa nền tảng (iOS, Android).
 - Sử dụng Dart language và framework Flutter.
 - Chứa lớp AppProvider và SupabaseService để xử lý logic phía client và giao tiếp với Backend.
 - Cần cấu hình Environment Variables (Supabase URL/Key) riêng biệt cho từng môi trường (Dev/Prod).
- GitHub (Source Control):
 - Central repository (Kho lưu trữ trung tâm) cho toàn bộ mã nguồn Flutter.
 - Cung cấp Version control để quản lý các phiên bản code.
 - Sử dụng Webhook để kích hoạt quy trình CI/CD khi có thay đổi mã nguồn.
 - Hỗ trợ Collaboration trong quá trình phát triển nhóm.
- Google Play / Apple App Store (Distribution):
 - Nền tảng phân phối chính thức ứng dụng di động tới người dùng cuối.
 - Yêu cầu quy trình Ký số (Signing) và tuân thủ các quy tắc Review nghiêm ngặt trước khi ứng dụng được công bố.

Quy trình triển khai

Quy trình được chia làm hai phần chính: Chuẩn bị Backend Production và Triển khai Ứng dụng.

1. Chuẩn bị Backend Production (Supabase)

- Bước 1: Tạo Supabase Project Production
 1. Đăng ký và tạo một Project mới trên Supabase Console (tách biệt với môi trường Dev).
 2. Lưu lại Supabase URL và Anon Key của môi trường Production.
- Bước 2: Triển khai Database Schema và Logic
 1. Sử dụng Supabase SQL Editor hoặc Migrations CLI.
 2. Chạy file supabase_schema.sql để tạo toàn bộ bảng, Indexes và RLS Policies.
 3. Chạy file supabase_functions.sql để tạo các Functions và Triggers cho cơ chế Auto-update Counters (Real-time features).
- Bước 3: Cấu hình Bảo mật

1. Xác nhận toàn bộ Row Level Security (RLS) cho các bảng chính đã được kích hoạt (ON).
 2. Cấu hình chính sách bảo mật cho Storage Buckets (chuẩn bị cho tính năng File upload trong tương lai).
2. Cấu hình và Triển khai Ứng dụng (Flutter)
- Bước 1: Cấu hình Môi trường Production
 1. Cập nhật file lib/config/supabase_config.dart với Production URL và Production Anon Key.
 2. Cấu hình Build Flavor (nếu có) để phân biệt Dev/Staging/Production.
 - Bước 2: Xây dựng (Build) Ứng dụng
 1. Sử dụng quy trình CI/CD (ví dụ: GitHub Actions hoặc Codemagic).
 2. Thực hiện lệnh build cho Production: flutter build appbundle (Android) hoặc flutter build ipa (iOS).
 - Bước 3: Phân phối (Deployment Process)
 1. Tải file build (AAB/IPA) lên Google Play Console hoặc Apple App Store Connect.
 2. Hoàn thành các yêu cầu ký số (Signing), mô tả và gửi đi review.
 3. Sau khi review thành công, ứng dụng được phát hành chính thức cho người dùng.

Database Management và Security (Supabase)

Việc quản lý database được đơn giản hóa tối đa nhờ Supabase là dịch vụ Managed.

- Database Security:
 - Tất cả kết nối giữa ứng dụng và Supabase đều được mã hóa SSL/TLS (HTTPS).
 - Bảo mật cốt lõi dựa trên Row Level Security (RLS), đảm bảo logic ủy quyền được thực thi ở tầng database.
 - Supabase Auth cung cấp cơ chế bảo mật JWT tiêu chuẩn ngành.
- Quản lý và Bảo trì:
 - Supabase tự động thực hiện Daily automated backups và automatic failover (High availability).

- Developer sử dụng Supabase Console để quản lý dữ liệu, kiểm tra Logs và thực hiện các thay đổi Schema (migrations).

Monitoring và Maintenance

Application Monitoring (Flutter)

- Sử dụng các công cụ Error logging và tracking (như Firebase Crashlytics hoặc Sentry) để theo dõi hiệu suất và lỗi của ứng dụng di động sau khi deploy.
- Theo dõi Performance metrics (ví dụ: thời gian phản hồi của API) được ghi nhận trong SupabaseService.

Database Monitoring (Supabase Dashboard)

- Real-time application logs: Theo dõi trực tiếp các request API, các lỗi Auth và Database.
- Query performance: Giám sát các truy vấn chậm (slow queries) để đảm bảo các thao tác cộng đồng như getPosts() hoạt động hiệu quả.
- Backup status verification: Kiểm tra tình trạng các bản sao lưu tự động của database.

Kết luận

Quy trình triển khai Phenikaa Connect App dựa trên Supabase mang lại nhiều ưu điểm vượt trội:

- Simplicity & Speed: Giảm thiểu cấu hình Server/Database, cho phép triển khai nhanh chóng.
- Scalability & Reliability: Tận dụng khả năng mở rộng và độ ổn định cao của dịch vụ Managed Cloud.
- Security: Được bảo vệ bởi cơ chế bảo mật mạnh mẽ RLS và Auth tiêu chuẩn.
- Real-time Ready: Cơ chế Real-time đã sẵn sàng hoạt động ngay lập tức sau deployment.

Quy trình này đảm bảo hệ thống được triển khai một cách chuyên nghiệp, ổn định và có khả năng mở rộng để phục vụ cộng đồng sinh viên Phenikaa.

4 Một số thành phần khác của đồ án

4.1. Kế hoạch dự án

Mục tiêu và phạm vi dự án

Mục tiêu chính

- Phát triển ứng dụng Phenikaa Connect – nền tảng kết nối sinh viên Trường Đại học Phenikaa.
- Tích hợp hệ thống backend dựa trên Supabase bao gồm: Authentication, Database PostgreSQL, Storage, Realtime, Edge Functions.
- Xây dựng các module chính:
 - Mạng xã hội nội bộ (Posts – Likes – Comments)
 - Sự kiện (Events – Attendees)
 - Câu hỏi – Hỏi đáp (Q&A)
 - Nhóm học tập (Study Groups)
 - Câu lạc bộ (Clubs)
 - Chat real-time
 - Thông báo từ trường
- Đảm bảo hệ thống realtime ổn định, bảo mật cao, dễ mở rộng.
- Tối ưu trải nghiệm người dùng trên Flutter (Android/iOS).

Phạm vi dự án

Trong phạm vi

- Xây dựng ứng dụng mobile Flutter.
- Tích hợp Supabase làm backend.
- Thiết kế và xây dựng 18 bảng database: users, posts, comments, events, groups,...
- Viết service cho Supabase: Auth, Users, Posts, Events, Groups, Messages.
- Xây dựng UI/UX trên Flutter.
- Đảm bảo tính năng realtime và bảo mật (RLS).
- Triển khai bản thử nghiệm (APK).
- Viết báo cáo + tài liệu kỹ thuật.

Ngoài phạm vi

- Hệ thống điểm số – học phí – đăng ký tín chỉ.
- Máy chủ backend tự xây (không sử dụng Node.js/Express).
- Dashboard dành cho giảng viên/cán bộ.
- Ứng dụng quản trị.

Phân giai đoạn thực hiện (12 tuần)

Giai đoạn 1: Phân tích và thiết kế (Tuần 1–3)

Công việc thực hiện

- Thu thập yêu cầu từ các dịch vụ trong trường Phenikaa.
- Phân tích hành vi người dùng (students – clubs – event organizers).
- Thiết kế database schema gồm 18 bảng.
- Viết file SQL (supabase_schema.sql).
- Thiết kế cấu trúc Edge Functions cho các sự kiện realtime.
- Tạo mockup giao diện (Home, Posts, Event, Chat).
- Lựa chọn công nghệ:
 - Flutter 3.24
 - Supabase SDK 2.0
 - Provider/ChangeNotifier
- Viết tài liệu SRS và kiến trúc hệ thống.

Kết quả chính

- Sơ đồ Use-case toàn hệ thống.
- Sơ đồ Sequence cho bài đăng, sự kiện, chat.
- Database ERD hoàn chỉnh.
- Mockup 20+ màn hình giao diện.

Giai đoạn 2: Phát triển backend – Supabase (Tuần 4–6)

Công việc thực hiện

- Khởi tạo project Supabase.
- Cấu hình Authentication (email/password).
- Viết SQL scripts:

- Tạo bảng (users, posts, comments, likes, events...).
- Tạo indexes (tối ưu truy vấn).
- Viết triggers cập nhật counters.
- Tạo functions: joinEvent(), leaveEvent(), toggleLike(), sendMessage().
- Tạo mock data bảng supabase_mock_data.sql.
- Cấu hình Row Level Security (RLS) cho từng bảng.
- Tạo Storage buckets dành cho images.

Kết quả chính

- Backend Supabase hoàn chỉnh.
- Toàn bộ dữ liệu mẫu được import.
- RLS policies hoạt động ổn định.
- Realtime triggers chạy đúng.

Giai đoạn 3: Phát triển frontend – Flutter (Tuần 7–9)

Công việc thực hiện

- Xây dựng UI theo mockup:
 - Login/Register
 - Home feed
 - Event list + event detail
 - Q&A
 - Study Group
 - Chat
 - Club list
 - User profile
- Phát triển Provider (AppProvider) để quản lý trạng thái.
- Tích hợp với SupabaseService đã viết ở backend.
- Viết logic:
 - Tạo bài đăng, like bài đăng realtime
 - Tham gia sự kiện
 - Tạo câu hỏi, trả lời câu hỏi
 - Tạo nhóm học và thêm thành viên
 - Chat real-time bằng Supabase Channels

- Tối ưu UX – animations – responsive.

Kết quả chính

- 90% giao diện & logic hoàn thành.
- 100% API được frontend gọi thành công.
- Ứng dụng chạy mượt trên Android.

Giai đoạn 4: Kiểm thử và sửa lỗi (Tuần 10–11)

Công việc thực hiện

- Viết test cases cho:
 - Auth
 - Posts
 - Events
 - Q&A
 - Group
- Test realtime channels (likes/messages).
- Test RLS (chặn người lạ truy cập dữ liệu).
- Chạy integration test toàn hệ thống.
- Tối ưu hiệu suất Supabase queries.
- Sửa lỗi UI, lỗi logic.

Kết quả chính

- 50+ test cases được chạy đúng.
- Ứng dụng ổn định, không crash.
- Dữ liệu bảo mật tốt.

Giai đoạn 5: Triển khai & tài liệu (Tuần 12)

Công việc thực hiện

- Build APK release.
- Kiểm thử trên thiết bị thật.
- Hoàn thiện tài liệu kỹ thuật:
 - API Docs

- Database Docs
- Cấu hình Supabase
- Viết báo cáo đầy đủ chương 1–4.
- Chuẩn bị slide & kịch bản demo.

Kết quả chính

- Ứng dụng Phenikaa Connect bản release chạy stable.
- Tài liệu hoàn thiện đầy đủ.
- Sẵn sàng báo cáo trước hội đồng.

Quản lý tài nguyên

Nhân lực

- Quách Gia Bảo – Leader
 - Quản lý dự án – thiết kế kiến trúc – fullstack Flutter/Supabase.
- Hoàng Văn Bắc – Frontend Developer
 - Giao diện Flutter, responsive UI, xử lý logic client.
- Lương Văn Thắng – Backend Developer
 - Supabase Specialist: SQL tables, RLS, functions, triggers.

Phân công công việc chi tiết theo thời gian

Quách Gia Bảo (Leader – Fullstack – Kiến trúc hệ thống)

Tuần 1–3 (Thiết kế & phân tích)

- Thiết kế kiến trúc tổng thể ứng dụng.
- Xác định module chính & phân chia chức năng.
- Tạo mockup giao diện.
- Review database schema của Thắng.

Tuần 4–6 (Backend & cấu trúc hệ thống)

- Cấu hình Supabase project.
- Viết SupabaseService chính: auth, posts, events, groups.
- Setup AppProvider và state management.
- Code review cho Thắng & Bắc.

Tuần 7–9 (Frontend)

- Xây dựng UI các màn quan trọng: Home, Post, Event, Profile.
- Logic chính của CreatePost, JoinEvent, Realtime.
- Tối ưu hiệu suất Flutter.

Tuần 10–12 (Testing & Deployment)

- Kiểm thử toàn hệ thống.
- Build APK và triển khai.
- Làm slide & demo.

Hoàng Văn Bắc (Frontend Developer – UI/UX)

Tuần 1–3

- Nghiên cứu Flutter UI/UX.
- Thiết kế giao diện bổ sung theo mockup.

Tuần 4–6

- Xây dựng component UI: cards, buttons, lists.
- Hỗ trợ xây dựng trang đăng nhập, đăng ký.

Tuần 7–9

- Xây dựng trang Q&A, Study Group, Clubs.
- Tích hợp API với SupabaseService.
- Kiểm thử giao diện và sửa lỗi UI.

Tuần 10–12

- Test giao diện trên nhiều thiết bị.
- Chuẩn bị ảnh chụp màn hình cho báo cáo.
- Hỗ trợ build & tối ưu cuối cùng.

Lương Văn Thắng (Backend Developer – Supabase Specialist)

Tuần 1–3

- Phân tích dữ liệu cần thiết.
- Thiết kế database 18 bảng.
- Viết supabase_schema.sql.

Tuần 4–6

- Viết SQL Functions: joinEvent, toggleLike, sendMessage...
- Viết triggers & policies RLS.
- Tạo mock data.

Tuần 7–9

- Test API Supabase.
- Kiểm tra realtime events.
- Thủ nghiem security với RLS.

Tuần 10–12

- Debug backend.
- Viết tài liệu kỹ thuật: Database docs, ERD.
- Hỗ trợ triển khai.

4.2. Đảm bảo làm việc nhóm

Cấu trúc nhóm và vai trò

Nhóm Phenikaa Connect (3 thành viên)

1. Quách Gia Bảo – Trưởng nhóm (Leader)

- Quản lý dự án, phân công công việc.
- Thiết kế kiến trúc toàn hệ thống.
- Lập trình chính: Flutter UI + Supabase Integration.
- Code review, đảm bảo chất lượng.

2. Hoàng Văn Bắc – Thành viên

- Phát triển UI giao diện người dùng.
- Thiết kế component và tối ưu Responsive.
- Hỗ trợ tích hợp API, sửa lỗi giao diện.

3. Lương Văn Thắng – Thành viên

- Phát triển backend trên Supabase.
- Viết SQL Schema, Functions, Triggers, RLS.
- Kiểm thử logic backend và bảo mật dữ liệu.

Quy trình làm việc nhóm

Phương pháp Agile – Sprint 2 tuần

- Daily meeting: 10–15 phút mỗi ngày trên nhóm discord, messenger và zalo.
- Sprint Planning: đầu mỗi sprint.
- Sprint Review: cuối sprint.
- Retrospective: đánh giá cải tiến quy trình.

Giao thức giao tiếp

- GitHub Issues & Pull Request để quản lý code.
- Google Drive để quản lý tài liệu.
- Discord/Google Meet/Zalo/Messager để họp và thảo luận.

Công cụ hợp tác & Git Flow

- Branch chính: main
- Branch chức năng:
 - bao/ui-module
 - bac/home-feed-ui
 - thang/supabase-functions
- Pull Request + Code Review trước khi merge.

Giải quyết xung đột

- Bảo quyết định kỹ thuật: Quách Gia Bảo (lead developer).
- Thiết kế UI: Bắc đề xuất – Bảo phê duyệt.
- Cơ sở dữ liệu và chức năng backend: Bảo phụ trách, Thắng hỗ trợ.
- Nếu mâu thuẫn → thảo luận chung → nếu cần, tham khảo leader.

4.3. Đạo đức và làm việc chuyên nghiệp

Bảo vệ dữ liệu và quyền riêng tư

- Supabase RLS đảm bảo dữ liệu đúng người dùng.
- Hash mật khẩu theo cơ chế Auth provider.
- Mã hóa giao tiếp HTTPS và token JWT.
- Chỉ lưu trữ thông tin cần thiết, không thu thập quá mức.

Minh bạch – Trung thực

- Mã nguồn rõ ràng, có ghi chú đầy đủ.
- Báo cáo lỗi trung thực, không che giấu.
- Không sử dụng dữ liệu thật của sinh viên ngoài phạm vi demo.

Tuân thủ chuẩn nghề nghiệp IEEE

- Mang lại lợi ích cho cộng đồng sinh viên.
- Tôn trọng quyền riêng tư và bảo vệ dữ liệu cá nhân.
- Liên tục học tập nâng cao kỹ năng.

Sở hữu trí tuệ

- Sử dụng Supabase, Flutter theo giấy phép mã nguồn mở.
- Tài nguyên bên ngoài đều ghi nguồn rõ ràng.
- Không sao chép code từ dự án khác.

Khả năng tiếp cận

- UI đơn giản, dễ dàng cho mọi đối tượng.
- Hỗ trợ đa ngôn ngữ (Vietnamese/English – tùy cấu hình Flutter).
- Thiết kế dễ đọc, màu sắc an toàn cho người khiếm thị nhẹ.

4.4. Tác động xã hội

Tác động tích cực

- Đối với sinh viên
 - Kết nối thông tin dễ dàng với trường.
 - Tham gia sự kiện, xem thông báo nhanh chóng.
 - Tương tác cộng đồng (posts, Q&A, clubs).
- Đối với nhà trường
 - Kênh truyền thông hiệu quả đến sinh viên.
 - Quản lý sự kiện – hoạt động sinh viên tốt hơn.
 - Thu thập dữ liệu phục vụ phân tích – báo cáo.

Tác động cộng đồng

- Tạo môi trường gắn kết sinh viên Phenikaa.
- Góp phần thúc đẩy văn hóa học thuật – chia sẻ.
- Tạo nền tảng cho việc mở rộng quy mô trường.

Tính bền vững lâu dài

- Hosting cloud thân thiện môi trường.
- Ít tiêu tốn tài nguyên (Supabase PostgreSQL tối ưu).
- Giảm sử dụng giấy (thông báo số – hoạt động số).

4.5. Kế hoạch học tập và phát triển kỹ năng

Đánh giá kỹ năng hiện tại

- Bảo: Flutter, UI/UX, State Management.
- Bắc: Giao diện Flutter, component design.
- Thắng: PostgreSQL, Supabase, SQL Functions.

Tài nguyên học tập

- Supabase Academy
- Flutter Docs
- YouTube (Code With Andrea, Reso Coder)
- Courses Udemy, Coursera, freeCodeCamp

Chiến lược chuyển giao kiến thức

- Document đầy đủ trong GitHub Wiki.
- Lập nhóm study session mỗi tuần.
- Mentor nội bộ: Bảo hướng dẫn Bắc & Thắng về Flutter; Bảo chia sẻ kiến thức Supabase.

Kế hoạch phát triển tương lai

- Version 2.0: push notification, file upload, advanced analytics.
- Version 3.0: AI recommendation, chat GPT-based assistant.
- Mobile release: build lên iOS + Android store.

5 Tài liệu tham khảo

- [1] Dennis, A., Wixom, B. H., & Tegarden, D. (2020). *Systems Analysis & Design: An Object-Oriented Approach with UML* (6th ed.). Wiley.
- [2] Gamma, E., Helm, R., Johnson, R., & Vlissides, J. (1995). *Design Patterns: Elements of Reusable Object-Oriented Software*. Addison-Wesley.
- [3] Pressman, R. S., & Maxim, B. R. (2019). *Software Engineering: A Practitioner's Approach* (9th ed.). McGraw-Hill Education.
- [4] Jorgensen, P. C. (2014). *Software Testing: A Craftsman's Approach* (4th ed.). CRC Press.
- [5] Node.js Foundation. (2023). *Node.js Documentation*. Available at: <https://nodejs.org/docs/>
- [6] Express.js. (2023). *Express.js - Node.js Web Application Framework*. Available at: <https://expressjs.com/>
- [7] World Wide Web Consortium (W3C). (2020). *Web Content Accessibility Guidelines (WCAG) 2.1*. Available at: <https://www.w3.org/TR/WCAG21/>
- [8] Oracle. (2023). *MySQL 8.0 Reference Manual*. Available at: <https://dev.mysql.com/doc/refman/8.0/en/>
- [9] Google Developers. (2023). *Google Identity Platform - OAuth 2.0*. Available at: <https://developers.google.com/>