

BỘ NÔNG NGHIỆP VÀ MÔI TRƯỜNG
TRƯỜNG ĐẠI HỌC THỦY LỢI



BÀI TẬP LỚN
PHÁT TRIỂN ỨNG DỤNG CHO THIẾT BỊ DI ĐỘNG
ĐỀ TÀI: ỨNG DỤNG TLUContact

Giáo viên hướng dẫn: ThS. Kiều Tuấn Dũng

Sinh viên thực hiện:

Mã sinh viên	Họ và tên	Lớp
2251061894	Nguyễn Thế Toàn	64CNTT1
2251061922	Nguyễn Thị Thanh Vân	64CNTT1
2251061807	Lưu Hiếu Khánh	64CNTT1
2251061863	Lê Hà Phương	64CNTT1

Hà Nội, năm 2025

BỘ NÔNG NGHIỆP VÀ MÔI TRƯỜNG
TRƯỜNG ĐẠI HỌC THỦY LỢI



BÀI TẬP LỚN
PHÁT TRIỂN ỨNG DỤNG CHO THIẾT BỊ DI ĐỘNG
ĐỀ TÀI: ỨNG DỤNG TLUContract

STT	Mã Sinh Viên	Họ và Tên	Ngày Sinh	Điểm	
				Bằng Số	Bằng Chữ
1	2251061894	Nguyễn Thế Toàn	01/09/2004		
2	2251061922	Nguyễn Thị Thanh Vân	20/10/2004		
3	2251061807	Lưu Hiếu Khánh	04/12/2004		
4	2251061863	Lê Hà Phương	02/10/2004		

CÁN BỘ CHẤM THI

Hà Nội, năm 2025

LỜI NÓI ĐẦU

Trong thời đại công nghệ số phát triển mạnh mẽ, việc số hóa và quản lý thông tin một cách hiệu quả đóng vai trò quan trọng trong mọi lĩnh vực, đặc biệt là trong môi trường giáo dục. Nhằm đáp ứng nhu cầu tra cứu, liên hệ và quản lý thông tin cán bộ, giảng viên và sinh viên một cách thuận tiện và chính xác, nhóm chúng em đã thực hiện đề tài "Xây dựng ứng dụng danh bạ điện tử cho Trường Đại học Thủy Lợi" như một phần của bài tập lớn môn học.

Ứng dụng danh bạ này được thiết kế với mục tiêu hỗ trợ nhà trường, cán bộ, giảng viên và sinh viên trong việc tìm kiếm, xem thông tin và liên hệ nhanh chóng với các thành viên trong trường. Ứng dụng không chỉ giúp tiết kiệm thời gian, nâng cao hiệu quả quản lý thông tin, mà còn góp phần hiện đại hóa công tác quản trị và kết nối nội bộ trong nhà trường.

Báo cáo này trình bày quá trình xây dựng dự án, từ phân tích yêu cầu, thiết kế hệ thống, triển khai chức năng đến kiểm thử và đánh giá kết quả. Trong quá trình thực hiện, nhóm đã áp dụng các kiến thức về lập trình Android, cơ sở dữ liệu Firebase cũng như mô hình kiến trúc MVVM để phát triển một ứng dụng thân thiện, dễ sử dụng và đáp ứng nhu cầu thực tế.

Chúng em xin chân thành cảm ơn thầy/cô đã tận tình hướng dẫn, tạo điều kiện để nhóm có cơ hội vận dụng kiến thức vào thực tiễn thông qua dự án này. Mặc dù đã nỗ lực hoàn thiện, nhưng do giới hạn về thời gian và kinh nghiệm, bài làm khó tránh khỏi thiếu sót. Nhóm mong nhận được sự góp ý, đánh giá từ thầy/cô để tiếp tục hoàn thiện kỹ năng và kiến thức của mình.

MỤC LỤC

DANH MỤC CÁC TỪ VIẾT TẮT	6
CHƯƠNG 1. TỔNG QUAN VỀ ĐỀ TÀI	7
1.1. Giới thiệu về đề tài	7
1.2. Mục tiêu của đề tài	7
1.3. Phạm vi của đề tài.....	8
1.4. Phân chia nhiệm vụ	10
CHƯƠNG 2. KIẾN TRÚC VÀ CÔNG NGHỆ	12
2.1. Kiến trúc hệ thống	12
2.2. Giới thiệu về Công nghệ phát triển	13
2.2.1. Công nghệ phát triển ứng dụng android	13
2.2.2. Công nghệ phát triển ứng dụng web.....	13
2.2.3. Firebase.....	14
CHƯƠNG 3. XÂY DỰNG ỨNG DỤNG	16
3.1. Thiết kế Figma.....	16
3.1.1 Màn hình đọc	16
3.1.2. Màn hình ngang	27
3.1.3 Màn hình web	42
3.2. Thiết kế CSDL.....	52
3.2.1. Cơ sở dữ liệu của guest	52
3.2.2. Cơ sở dữ liệu của Student.....	53
3.2.3. Cơ sở dữ liệu của Staff	53
3.2.4. Cơ sở dữ liệu của Department.....	53
3.3. Giao diện ứng dụng	54
3.3.1. Logo ứng dụng.....	54

3.3.2. Màn hình đăng nhập	54
3.3.3. Đăng ký	58
3.3.4. Email xác nhận tài khoản	62
3.3.5. Khôi phục mật khẩu.....	63
3.3.6. Email khôi phục mật khẩu.....	66
3.3.7. Giao diện chỉnh sửa thông tin tài khoản khách	67
3.3.8. Giao diện xem danh bạ sinh viên	68
3.3.9. Giao diện xem danh bạ giảng viên	74
3.3.10. Giao diện xem danh bạ đơn vị.....	80
3.3.11. Giao diện web.....	85
3.4. Code minh họa các chức năng cốt lõi.....	87
KẾT LUÂN	108
1. Kết quả đạt được.....	108
2. Nhược điểm	109
3. Hướng phát triển.....	109

DANH MỤC CÁC TỪ VIẾT TẮT

STT	TỪ VIẾT TẮT	VIẾT ĐẦY ĐỦ
1	MVVM	Model-View-ViewModel
2	MVC	Model-View-Controller
3	Jetpack	Android Jetpack Libraries

CHƯƠNG 1. TỔNG QUAN VỀ ĐỀ TÀI

1.1. Giới thiệu về đề tài

Dù công nghệ thông tin ngày càng phát triển và được ứng dụng rộng rãi trong giáo dục, Trường Đại học Thủy Lợi vẫn đang sử dụng các phương thức truyền thống như tài liệu giấy hoặc file Word để quản lý và tra cứu thông tin liên lạc của cán bộ, giảng viên và sinh viên. Điều này dẫn đến nhiều hạn chế như việc tìm kiếm thông tin kém hiệu quả, cập nhật dữ liệu chậm trễ và nguy cơ cao bị thất lạc dữ liệu. Bên cạnh đó, hiện vẫn chưa có một hệ thống danh bạ tập trung, khiến sinh viên và giảng viên khó có thể liên lạc trực tiếp với nhau. Việc trao đổi thông tin thường phải thông qua giáo viên phụ trách liên chi đoàn, gây tốn thời gian và ảnh hưởng đến hiệu quả giao tiếp.

Xuất phát từ thực tế đó, đề tài “TLUContact – Ứng dụng danh bạ điện tử cho Đại học Thủy Lợi” ra đời với mục tiêu xây dựng một nền tảng hiện đại, an toàn và tiện lợi để quản lý thông tin liên lạc trong nhà trường. Ứng dụng được phát triển trên hệ điều hành Android, sử dụng ngôn ngữ lập trình Kotlin kết hợp với cơ sở dữ liệu NoSQL của Firebase (Cloud Firestore) nhằm đảm bảo khả năng lưu trữ linh hoạt, truy xuất nhanh chóng và đồng bộ dữ liệu theo thời gian thực. Ngoài ra, hệ thống còn tích hợp Firebase Authentication để hỗ trợ đăng ký, xác thực và phân quyền người dùng, giúp đảm bảo tính bảo mật và riêng tư trong quá trình sử dụng.

TLUContact không chỉ đóng vai trò là một danh bạ điện tử, mà còn là cầu nối thông tin linh hoạt giữa các thành viên trong trường. Ứng dụng giúp sinh viên và giảng viên dễ dàng tìm kiếm, liên hệ với nhau một cách trực tiếp, loại bỏ sự phụ thuộc vào các khâu trung gian. Qua đó, TLUContact góp phần nâng cao hiệu quả học tập, giảng dạy và làm việc, đồng thời thúc đẩy quá trình chuyển đổi số trong giáo dục. Đây cũng là bước tiến quan trọng giúp Trường Đại học Thủy Lợi tiến gần hơn đến mô hình quản lý hiện đại và thông minh.

1.2. Mục tiêu của đề tài

Đề tài TLUContact đặt ra những mục tiêu cơ bản và nâng cao nhằm hướng tới một hệ thống quản lý danh bạ hiện đại và an toàn. Cụ thể, các mục tiêu chính bao gồm:

- **Mục tiêu tổng quát:** Xây dựng một ứng dụng danh bạ điện tử trên nền tảng Android, giúp quản lý và tra cứu thông tin liên lạc của các đơn vị, cán bộ giảng viên và sinh viên trong Đại học Thủy Lợi một cách nhanh chóng, chính xác và an toàn.

Mục tiêu cụ thể:

1. Phát triển giao diện người dùng thân thiện và trực quan:

Thiết kế giao diện ứng dụng hiện đại, dễ sử dụng, cho phép người dùng dễ dàng tìm kiếm, lọc và sắp xếp thông tin theo nhu cầu, đồng thời đảm bảo trải nghiệm người dùng mượt mà trên thiết bị di động.

2. Tích hợp và sử dụng hiệu quả Firebase:

Áp dụng Firebase Authentication để thực hiện quy trình đăng ký, đăng nhập và xác thực người dùng; sử dụng Firebase Cloud Firestore làm cơ sở dữ liệu NoSQL nhằm đảm bảo tính linh hoạt, mở rộng và tối ưu hóa hiệu năng truy xuất dữ liệu.

3. Đảm bảo an toàn và bảo mật thông tin:

Xây dựng hệ thống phân quyền dựa trên vai trò của người dùng (CBGV, sinh viên, Admin) cùng với các Firebase Security Rules, từ đó bảo vệ dữ liệu và ngăn ngừa truy cập trái phép.

4. Tạo điều kiện thuận lợi cho việc bảo trì và mở rộng:

Thiết kế cấu trúc hệ thống theo hướng module, dễ dàng bổ sung và thay đổi các chức năng mới khi có yêu cầu, đảm bảo khả năng thích ứng với sự phát triển của công nghệ và nhu cầu thực tiễn.

1.3. Phạm vi của đề tài

Để đảm bảo tính khả thi và tập trung vào các chức năng cốt lõi, phạm vi của đề tài TLUContact được xác định như sau:

- **Phạm vi về công nghệ và nền tảng:**

- Ứng dụng được xây dựng cho hệ điều hành Android, sử dụng ngôn ngữ lập trình Kotlin – một lựa chọn hiện đại và mạnh mẽ cho phát triển ứng dụng di động.

- Sử dụng dịch vụ Firebase của Google, với Cloud Firestore làm cơ sở dữ liệu NoSQL và Firebase Authentication để quản lý quy trình đăng ký, đăng nhập và xác thực người dùng.
- **Phạm vi về chức năng:**
 - **Quản lý tài khoản người dùng:**
 - Cho phép người dùng đăng ký, đăng nhập và xác thực thông qua email được cấp phát bởi trường (phân biệt giữa email của cán bộ giảng viên và sinh viên).
 - Phân quyền truy cập dựa trên loại tài khoản, đảm bảo rằng mỗi đối tượng người dùng chỉ có quyền truy cập và thao tác phù hợp.
 - **Danh bạ đơn vị, cán bộ giảng viên và sinh viên:**
 - **Danh bạ đơn vị:** Quản lý và hiển thị thông tin của các đơn vị trong trường (Khoa, Phòng, Trung tâm...), bao gồm các thông tin như tên, mã, địa chỉ và các liên hệ cần thiết.
 - **Danh bạ cán bộ giảng viên:** Cung cấp danh sách và chi tiết thông tin của cán bộ giảng viên, cho phép tìm kiếm theo tên, chức vụ, đơn vị...; với phân quyền truy cập ưu tiên cho CBGV, Admin
 - **Danh bạ sinh viên:** Hiển thị danh sách sinh viên với thông tin cá nhân cơ bản và chi tiết, cho phép sinh viên xem danh bạ cùng lớp hoặc cùng đơn vị trực thuộc, Admin có quyền truy cập ưu tiên mọi thông tin sinh viên.
 - **Cập nhật thông tin cá nhân:**

Cho phép người dùng tự cập nhật các thông tin cá nhân như ảnh đại diện, số điện thoại, địa chỉ... qua giao diện riêng, đảm bảo tính đồng bộ và an toàn dữ liệu.
- **Phạm vi về bảo mật và hiệu năng:**
 - Áp dụng các Firebase Security Rules nhằm đảm bảo dữ liệu được bảo vệ theo đúng phân quyền của từng đối tượng người dùng.
 - Tối ưu hóa các truy vấn dữ liệu qua các phương thức của Firebase, đảm bảo khả năng xử lý thông tin nhanh chóng dù khi danh sách dữ liệu lớn.

Như vậy, phạm vi đề tài tập trung vào việc xây dựng một hệ thống danh bạ điện tử chuyên dụng cho Đại học Thủy Lợi, với những chức năng chính cần thiết, nhằm hỗ trợ

quản lý thông tin liên lạc hiệu quả, đồng thời tạo nền tảng vững chắc cho các tính năng mở rộng trong tương lai.

1.4. Phân chia nhiệm vụ

Họ và tên	Nhiệm vụ	Đánh giá
Nguyễn Thế Toàn	<ul style="list-style-type: none"> - Code đăng ký tài khoản trên app - Code đăng nhập bằng email/mật khẩu trên app - Code đăng nhập bằng email/mật khẩu trên web - Code đăng nhập tài khoản outlook - Code khôi phục mật khẩu - Code xác nhận tài khoản - Viết security rule - Code đăng xuất tài khoản - Code sửa, lưu thông tin khách và admin - Thiết kế cơ sở dữ liệu khách và admin. - Thiết kế ½ logo app 	Hoàn thành
Nguyễn Thị Thanh Vân	<ul style="list-style-type: none"> - Chính sửa giao diện chung - Hiển thị danh bạ sinh viên - Tìm kiếm theo tên, mã sinh viên - Lọc dữ liệu sinh viên theo lớp, theo tên - Cơ sở dữ liệu sinh viên - Chức năng nhắn tin (qua zalo hoặc sms), gọi điện, gọi video, gửi email - Sắp xếp danh bạ sinh viên (từ A-Z, Z-A) - Xem chi tiết thông tin sinh viên - Hiển thị thông tin người dùng - Xem chi tiết thông tin người dùng là sinh viên. - Chính sửa các trường thông tin của người dùng là sinh viên và lưu vào cloud firestore - Phân quyền người xem là sinh viên lớp nào thì chỉ xem được của lớp tương ứng 	Hoàn thành

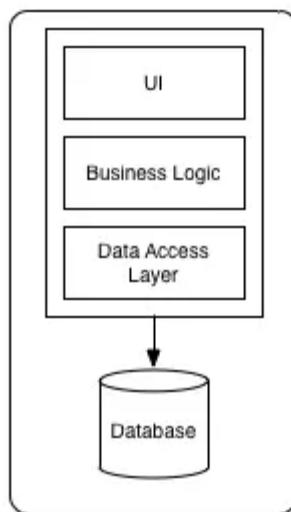
Lưu Hiếu Khánh	<ul style="list-style-type: none"> - Hiển thị danh bạ đơn vị - Xem thông tin chi tiết đơn vị - Tìm kiếm đơn vị (theo tên/ id) - Sắp xếp danh bạ đơn vị theo tên từ A-Z hoặc từ Z-A - Lọc dữ liệu đơn vị (theo Khoa/ Phòng/ Trung tâm/ Viện/ Tất cả) - Cơ sở dữ liệu đơn vị - Chức năng gọi điện thoại, gửi email - Thiết kế ½ logo app 	Hoàn thành
Lê Hà Phương	<ul style="list-style-type: none"> - Tạo HomeScreen chung của 3 danh bạ - Hiển thị danh bạ cán bộ giảng viên - Sửa các trường thông tin người dùng là giảng viên và cập nhật vào firestore - Tìm kiếm giảng viên theo tên - Lọc dữ liệu giảng viên theo đơn vị, theo chức vụ (Giảng viên/ Giảng viên chính/Trưởng bộ môn,...) hoặc theo tất cả (danh sách ban đầu) và Sắp xếp danh sách giảng viên từ A-Z hoặc từ Z-A - Cơ sở dữ liệu cán bộ giảng viên - Chức năng gọi điện, gọi video (nếu không có app gọi video sẽ điều hướng đến CH Play), nhắn tin, gửi email - Xem thông tin chi tiết giảng viên - Hiển thị thông tin giảng viên theo email đăng nhập ở HomeScreen và giao diện sửa thông tin 	Hoàn thành

CHƯƠNG 2. KIẾN TRÚC VÀ CÔNG NGHỆ

2.1. Kiến trúc hệ thống

Hệ thống được xây dựng theo mô hình kiến trúc nguyên khối (Monolithic Architecture). Đây là mô hình kiến trúc phần mềm trong đó tất cả các thành phần của ứng dụng được gộp chung vào một khối duy nhất. Kiến trúc này có các đặc điểm sau:

- **Tích hợp toàn bộ ứng dụng:** Mọi thành phần như giao diện người dùng (UI), xử lý logic nghiệp vụ, truy vấn cơ sở dữ liệu, và giao tiếp với các dịch vụ bên ngoài đều được đặt chung trong một ứng dụng duy nhất.
- **Dễ triển khai:** Ứng dụng có thể được triển khai dưới dạng một tệp tin duy nhất, giúp quá trình triển khai và bảo trì trở nên đơn giản.
- **Hiệu suất cao:** Do không có chi phí giao tiếp giữa các dịch vụ như trong kiến trúc microservices, kiến trúc monolithic thường hoạt động nhanh và hiệu quả hơn trong môi trường nhỏ và vừa.
- **Khó mở rộng:** Một hạn chế của kiến trúc này là khi ứng dụng phát triển lớn mạnh, việc mở rộng và duy trì có thể gặp nhiều thách thức do tất cả các thành phần phụ thuộc chặt chẽ vào nhau.



Monolithic Architecture

Trong hệ thống này, toàn bộ mã nguồn của ứng dụng sẽ được tổ chức thành một khối duy nhất, với các thành phần xử lý dữ liệu, giao diện và logic nghiệp vụ nằm

trong cùng một mã nguồn và được triển khai trên cùng một máy chủ hoặc dịch vụ lưu trữ đám mây.

2.2. Giới thiệu về Công nghệ phát triển

Hệ thống được phát triển dựa trên các công nghệ hiện đại nhằm đảm bảo tính hiệu quả, bảo mật và khả năng mở rộng trong tương lai. Dưới đây là các công nghệ chính được sử dụng:

2.2.1. Công nghệ phát triển ứng dụng android

2.2.1.1. Android Studio

- **Trình biên dịch và giả lập mạnh mẽ:** Cho phép kiểm tra ứng dụng trực tiếp trên máy tính.
- **Hỗ trợ Gradle:** Quản lý thư viện và quá trình biên dịch.
- **Giao diện kéo – thả (Drag and Drop):** Hỗ trợ thiết kế UI nhanh chóng.
- **Tích hợp công cụ kiểm thử:** Đảm bảo hiệu suất và độ ổn định trước khi triển khai.
- **Tương thích đa thiết bị:** Hỗ trợ nhiều phiên bản Android khác nhau.

2.2.1.2. Kotlin

- **Cú pháp ngắn gọn, dễ đọc:** Giảm thiểu số dòng code so với Java.
- **An toàn hơn:** Hạn chế lỗi Null Pointer Exception.
- **Hỗ trợ lập trình hàm:** Cho phép sử dụng lambda, functional programming.
- **Tương thích với Java:** Có thể tích hợp vào các dự án Java hiện có.

2.2.1.3. Jetpack

- **Tích hợp reactive UI:** Giao diện tự động cập nhật theo dữ liệu trong MVVM.
- **UI và logic liền mạch:** Viết UI bằng Kotlin, dễ đọc và dễ debug hơn.

2.2.2. Công nghệ phát triển ứng dụng web

2.2.2.1. Java & Maven

- **Xây dựng theo mô hình MVC:** Tách riêng các thành phần xử lý logic, giao diện và dữ liệu.
- **Quản lý dự án bằng Maven:** Hỗ trợ kiểm soát thư viện và quá trình build hiệu quả.

- **Tích hợp Firebase Admin SDK:** Thao tác trực tiếp với cơ sở dữ liệu Firebase và quản lý người dùng.

2.2.2.2. HTML, CSS và JavaScript

- **Thiết kế giao diện web:** Đảm bảo tính trực quan và thân thiện với người dùng.
- **Kết nối với Firebase SDK for Web:** Đăng nhập, đăng ký, đồng bộ dữ liệu thời gian thực.
- **Triển khai trên Firebase Hosting:** Dễ dàng triển khai và vận hành miễn phí.

2.2.3. Firebase

Firebase là nền tảng dịch vụ đám mây của Google, cung cấp nhiều công cụ mạnh mẽ giúp phát triển ứng dụng một cách nhanh chóng và hiệu quả. Trong hệ thống này, Firebase được sử dụng để quản lý cơ sở dữ liệu, xác thực người dùng và phân quyền truy cập dữ liệu.

Firebase Cloud Firestore (Cơ sở dữ liệu)

Firebase Cloud Firestore là hệ quản trị cơ sở dữ liệu NoSQL, cung cấp các tính năng như:

- **Lưu trữ và truy vấn dữ liệu theo cấu trúc JSON.**
- **Đồng bộ dữ liệu theo thời gian thực giữa các thiết bị.**
- **Tích hợp dễ dàng với các dịch vụ Firebase khác.**
- **Khả năng mở rộng linh hoạt, hiệu suất cao.**

Dữ liệu của ứng dụng được tổ chức dưới dạng bộ sưu tập (collections) và tài liệu (documents), giúp dễ dàng quản lý và truy vấn thông tin cần thiết.

Firebase Authentication (Xác thực người dùng)

Firebase Authentication cung cấp các phương thức đăng nhập bảo mật và tiện lợi, bao gồm:

- **Đăng ký và đăng nhập bằng email/mật khẩu.**
- **Hỗ trợ xác thực bằng tài khoản Google, Facebook, và nhiều dịch vụ bên thứ ba khác.**

- **Tích hợp xác thực hai yếu tố (2FA) để tăng cường bảo mật.**
- **Quên mật khẩu:** Cho phép người dùng đặt lại mật khẩu thông qua email khôi phục.

Việc sử dụng Firebase Authentication giúp hệ thống đảm bảo an toàn và bảo mật thông tin tài khoản người dùng.

Firebase Security Rules (Phân quyền truy cập dữ liệu)

Firebase Security Rules cho phép kiểm soát quyền truy cập vào cơ sở dữ liệu Cloud Firestore. Một số tính năng quan trọng bao gồm:

- **Quyền truy cập dựa trên vai trò:** Hệ thống có thể cấp quyền khác nhau cho từng nhóm người dùng (admin, người dùng thông thường, v.v.).
- **Xác thực dữ liệu:** Chỉ cho phép người dùng hợp lệ truy cập và sửa đổi thông tin theo quyền hạn được cấp.
- **Bảo vệ dữ liệu theo cấp độ tài liệu hoặc bộ sưu tập:** Giúp đảm bảo chỉ những người dùng có quyền mới có thể truy cập dữ liệu quan trọng.

Nhờ vào Firebase Security Rules, ứng dụng có thể ngăn chặn các truy cập trái phép và bảo vệ dữ liệu người dùng một cách hiệu quả.

CHƯƠNG 3. XÂY DỰNG ỨNG DỤNG

3.1. Thiết kế Figma

Link thiết kế figma của dự án: [TLUContact – Figma](#)

3.1.1 Màn hình dọc

3.1.1.1. Đăng nhập

Đăng nhập

Chào mừng đến với TLUContract

Email

Mật khẩu

Quên mật khẩu?

Đăng nhập

Chào mừng đến với TLUContract

Email

Mật khẩu

Quên mật khẩu?

Đăng nhập

Không có tài khoản? [Đăng ký](#)

Hoặc đăng nhập với

ABC space return

Hoặc đăng nhập với

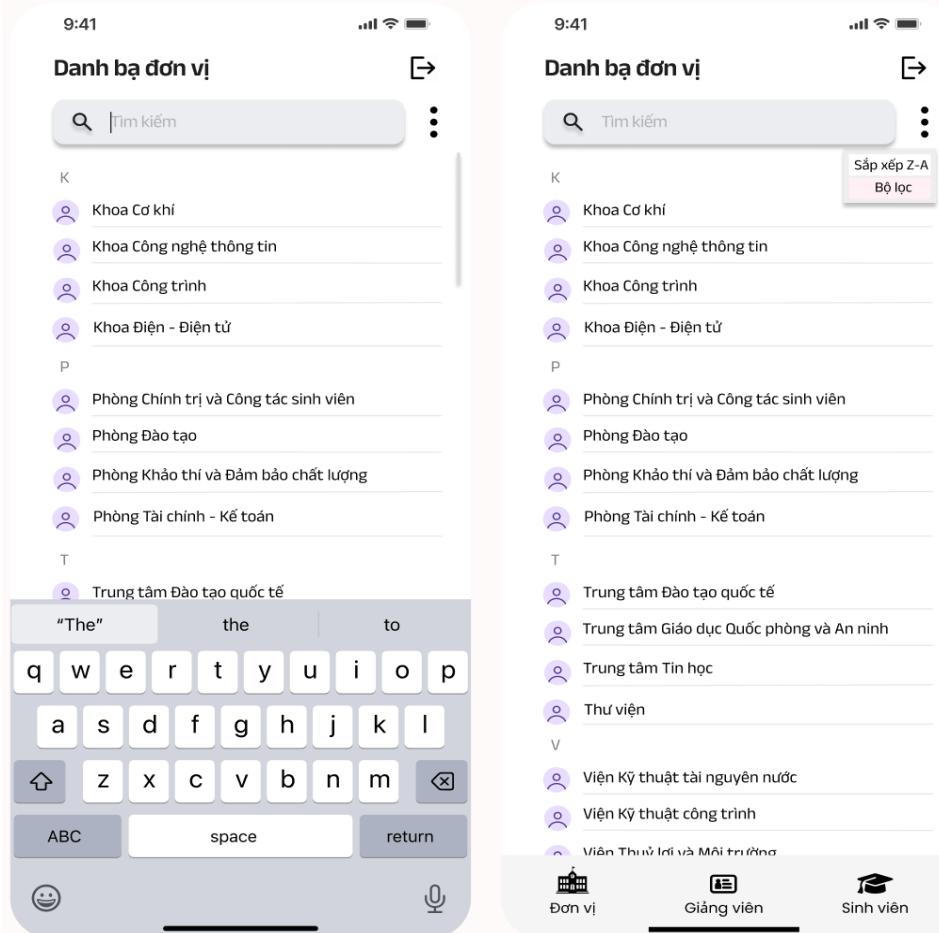
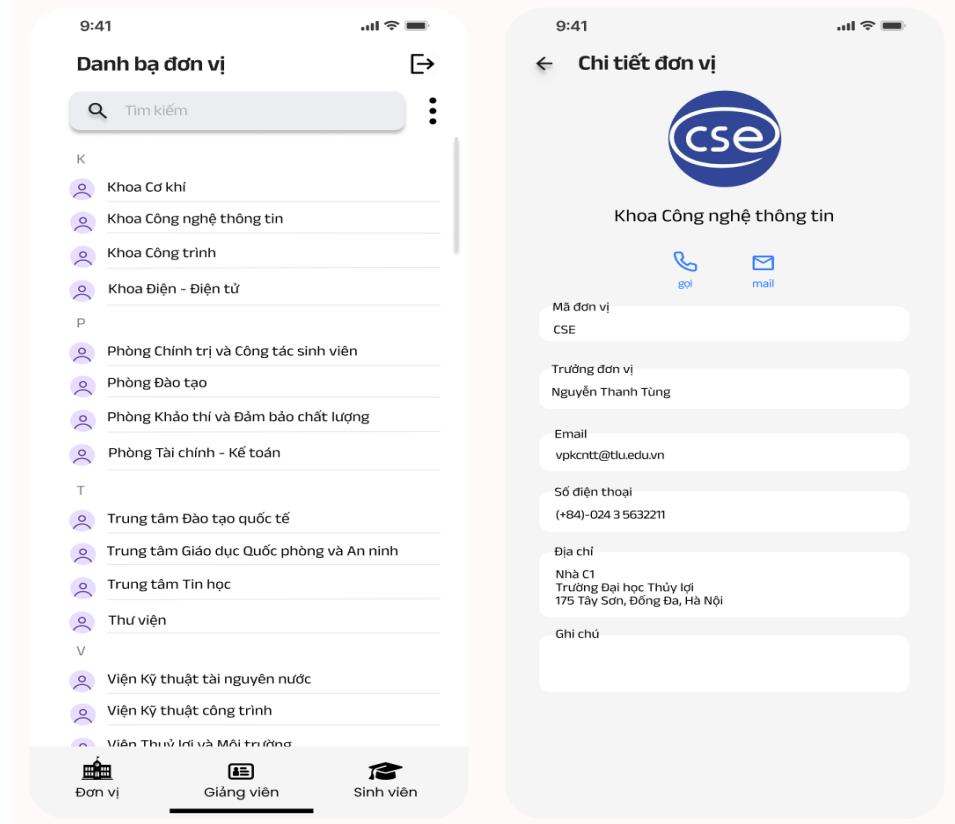
3.1.1.2. Đăng ký

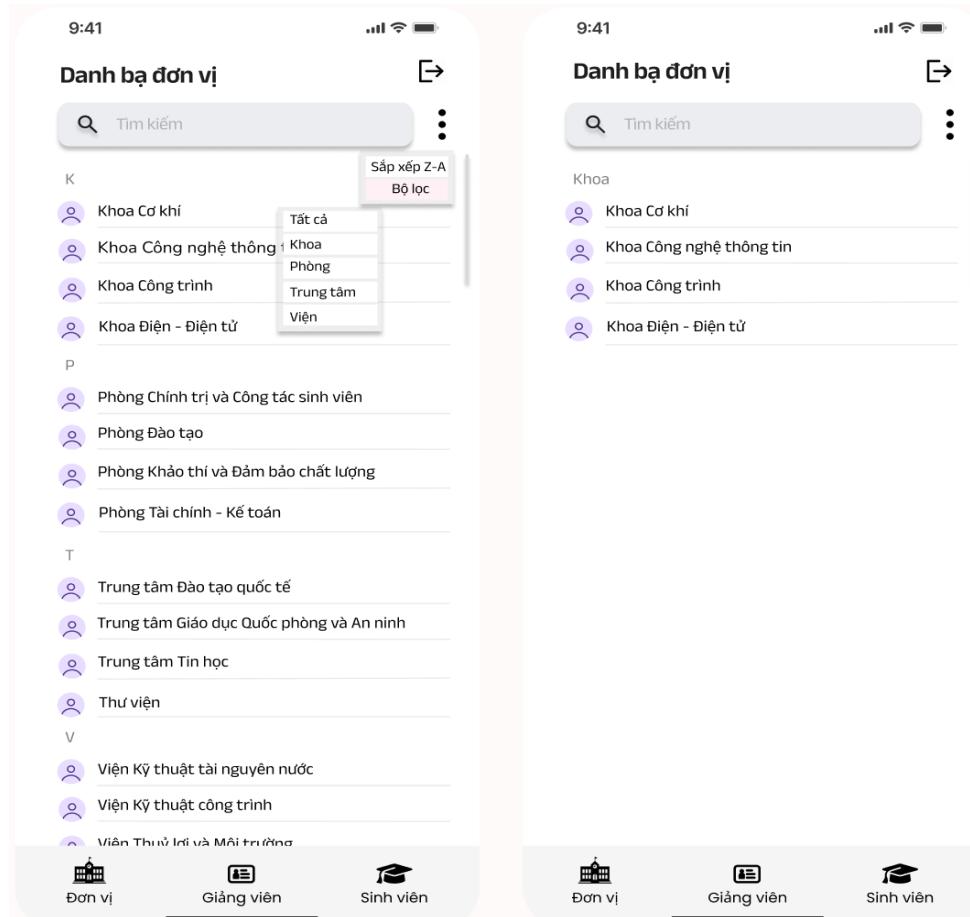
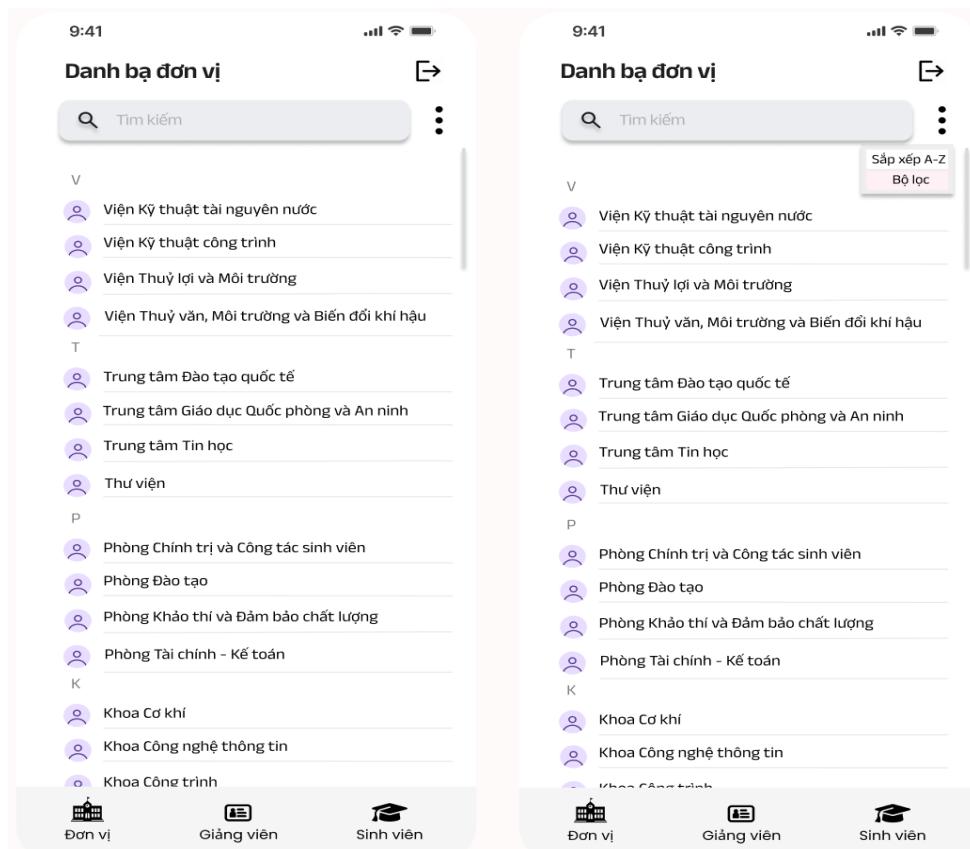
The image displays two side-by-side screenshots of the TLUContact mobile application interface. Both screens are titled "Đăng ký" (Registration) and feature a circular logo at the top left with the text "1959". Below the logo is a message: "Chỉ mất vài giây để kết nối với mọi người" (Only a few seconds to connect with everyone). The registration form consists of several input fields: "Số điện thoại" (Phone number), "Email", "Mật khẩu" (Password), and "Nhập lại mật khẩu" (Re-enter password). Each password field has a blue circular icon with a checkmark to its right. Below these fields is a large black rounded rectangle containing the text "Đăng ký" in white. Underneath this button is the text "Đã có tài khoản? [Đăng nhập](#)" (Already have an account? [Log in](#)). At the bottom of the screen is a virtual keyboard.

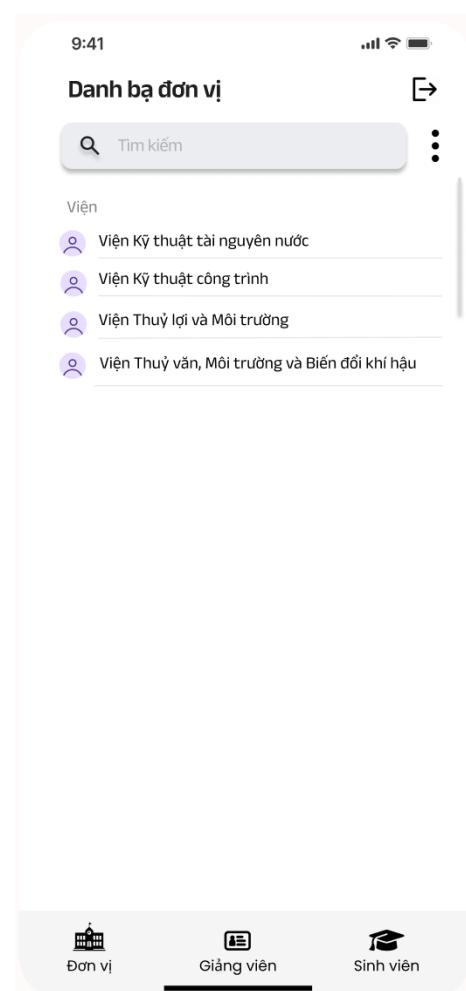
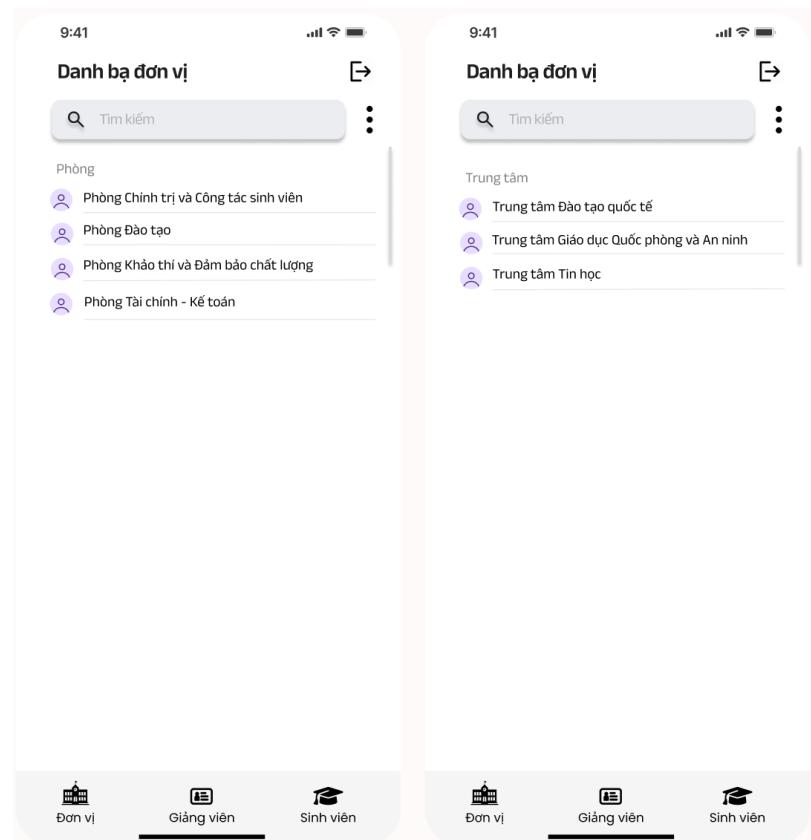
3.1.1.3. Khôi phục mật khẩu

The image displays two side-by-side screenshots of the TLUContact mobile application interface. Both screens are titled "Gửi mã xác minh" (Send verification code) and feature a circular logo at the top left with the text "1959". Below the logo is a message: "Nhập email của bạn" (Enter your email). The main action is a large black rounded rectangle containing the text "Gửi mã xác minh" in white. Underneath this button is the text "Nhập email của bạn" again. At the bottom of the screen is a virtual keyboard.

3.1.1.4. Xem danh bạ đơn vị







3.1.1.5. Xem danh bạ cán bộ giảng viên

The screenshots show the faculty staff directory screen with a search bar at the top. The search results are listed by name, with initials (B, C, D, H) on the left. The results include:

- B**: Bùi Thị Linh (Giảng viên), Bùi Thị A (Giảng viên), Nguyễn An (Giảng viên)
- C**: Chu Văn B (Giảng viên), Chu Thị B (Giảng viên chính), Cường Thị B (Giảng viên), Chí Văn B (Trưởng bộ môn)
- D**: Đào Văn Chung (Trưởng khoa), Đào Văn C (Giảng viên), Đỗ Văn C (Giảng viên), Đỗ Hoài C (Giảng viên chính)
- H**: Hoàng Minh D (Giảng viên), Hoàng Thị D (Giảng viên)

Below the search results are three tabs: **Đơn vị**, **Giảng viên** (highlighted in blue), and **Sinh viên**.

In the middle screenshot, the search bar contains "The" and the keyboard is visible.

In the bottom screenshots, the search bar contains "Nguyễn Thị Mai Hương" and the results are filtered to show:

- B**: Bùi Thị Linh (Giảng viên), Bùi Thị A (Giảng viên), Nguyễn An (Giảng viên)
- C**: Chu Văn B (Giảng viên), Chu Thị B (Giảng viên chính), Cường Thị B (Giảng viên), Chí Văn B (Trưởng bộ môn)
- D**: Đào Văn Chung (Trưởng khoa), Đào Văn C (Giảng viên), Đỗ Văn C (Giảng viên), Đỗ Hoài C (Giảng viên chính)
- H**: Hoàng Minh D (Giảng viên), Hoàng Thị D (Giảng viên)
- Y**: Yên Ngọc Linh (Giảng viên)
- X**: Xuân Thị Thanh (Giảng viên)
- V**: Vũ Văn Nguyệt (Giảng viên), Vũ Thế Minh (Giảng viên), Vũ Thực Anh (Giảng viên)
- T**: Trần Văn Tú (Giảng viên), Tạ Thị Tuyền (Giảng viên), Trần Thị Tuyền (Giảng viên)
- Q**: Quyển Văn Thực Quyển (Giảng viên), Quý Văn Quyển (Giảng viên), Quỳnh Đức Quân (Giảng viên)
- P**: Phạm Văn Phong (Giảng viên)

Danh bạ cán bộ giảng viên

Tìm kiếm

Hồ sơ của bạn

Nguyễn Thị Mai Hương

Sắp xếp Z-A
Bộ lọc

Khoa Công nghệ thông tin

Nguyễn Thu Thủy
Giảng viên

Trần Văn Nam
Giảng viên

Nguyễn Thị Vân
Giảng viên

Khoa Cơ khí

Phạm Văn Bình
Giảng viên

Nguyễn Thị Bưởi
Giảng viên

Phạm Thị Trang
Giảng viên

Lê Văn Huy
Giảng viên

Khoa Kỹ thuật tài nguyên nước

Nguyễn Chung
Giảng viên

Lê Văn Cường
Giảng viên

Vũ Văn Chung
Giảng viên

Đỗ Hoài Nam
Giảng viên

Khoa công trình

Nguyễn Thị Dung
Giảng viên

Nguyễn Văn Hoàn
Giảng viên

Đơn vị Giảng viên Sinh viên

← Chính sửa thông tin

Nguyễn Thị Mai Hương

Họ và tên
Nguyễn Thị Mai Hương

Mã giảng viên
2251061111

Chức vụ
Giảng viên

Số điện thoại
0337725310

Email
mai.huong@example.com

Đơn vị trực thuộc
Khoa Công nghệ thông tin

Hủy Lưu

Danh bạ giảng viên

Tìm kiếm

Hồ sơ của bạn

Nguyễn Thị Mai Hương

Sắp xếp Z-A
Bộ lọc

Giảng viên

Nguyễn Thu Thủy
Giảng viên

Trần Văn Nam
Giảng viên

Nguyễn Thị Vân
Giảng viên

Giảng viên chính

Phạm Chí Thanh
Giảng viên chính

Nguyễn Thị Nhưng
Giảng viên chính

Phạm Như Trang
Giảng viên chính

Đào Văn Huy
Giảng viên chính

Trưởng bộ môn

Lê Đào Chung
Trưởng bộ môn

Lê Thị Cường
Trưởng bộ môn

Trần Văn Chung
Trưởng bộ môn

Đỗ Văn Nam
Trưởng bộ môn

KS

Lê Thị Kim Dung
KS

Nguyễn Văn Nhường
KS

Đơn vị Giảng viên Sinh viên

Danh bạ giảng viên

→

Tất cả
Đơn vị
Chức vụ

B

Bùi Thị Linh
Giảng viên

Bùi Thị A
Giảng viên

Nguyễn An
Giảng viên

C

Chu Văn B
Giảng viên

Chu Thị B
Giảng viên chính

Cường Thị B
Giảng viên

Chí Văn B
Trưởng bộ môn

D

Đào Văn Chung
Trưởng khoa

Đào Văn C
Giảng viên

Đỗ Văn C
Giảng viên

Đỗ Hoài C
Giảng viên chính

H

Hoàng Minh D
Giảng viên

Hoàng Thị D
Giảng viên

Đơn vị Giảng viên Sinh viên

9:41

← Chính sửa thông tin


Nguyễn Thị Mai Hương

Họ và tên	Nguyễn Thị Mai Hương
Mã giảng viên	22510611111
Chức vụ	Giảng viên
Số điện thoại	0337725310
Email	mai.huong@example.com
Ban vị trực thuộc	Khoa Công nghệ thông tin

Hủy **Lưu**

✔ Cập nhật thông tin thành công!

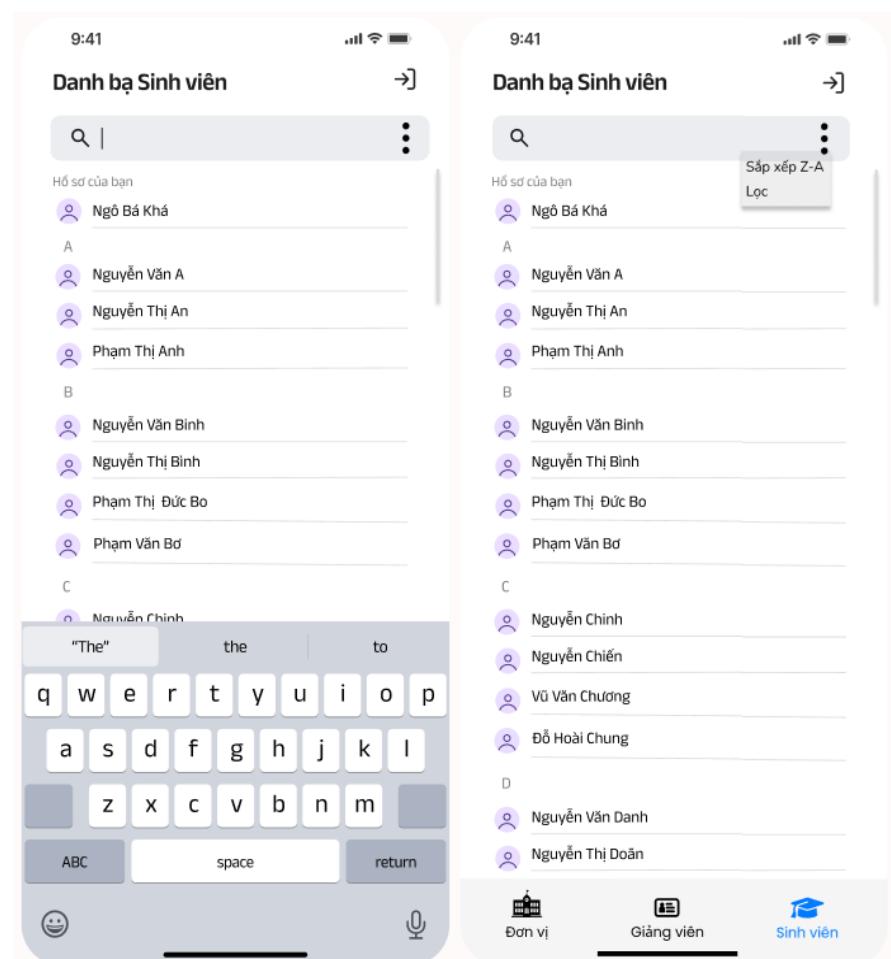
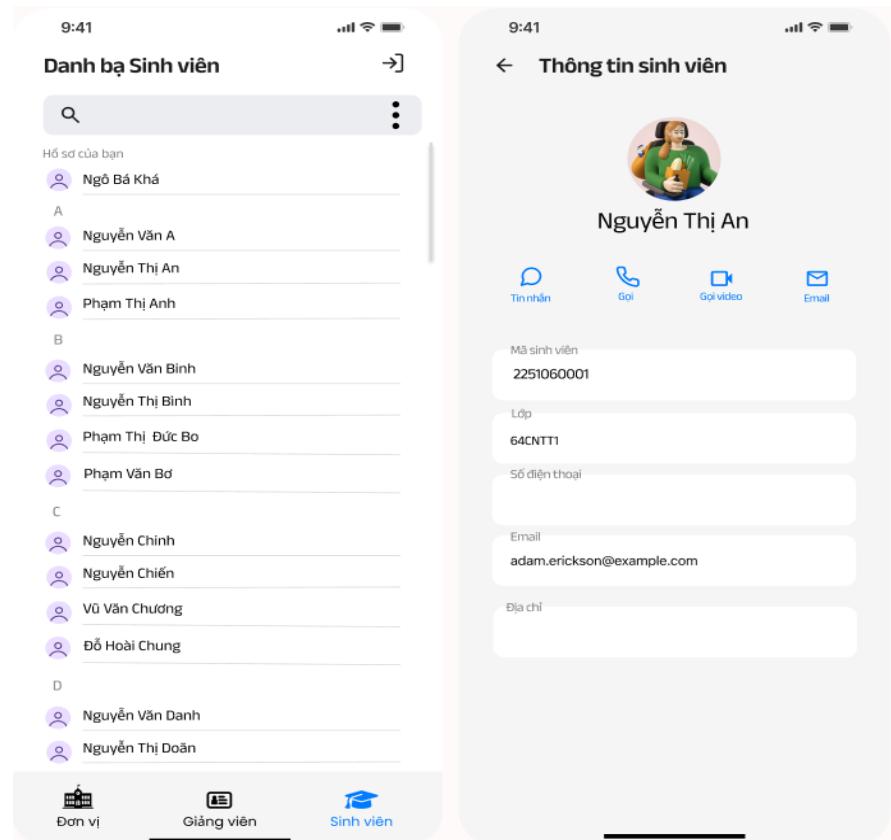
9:41

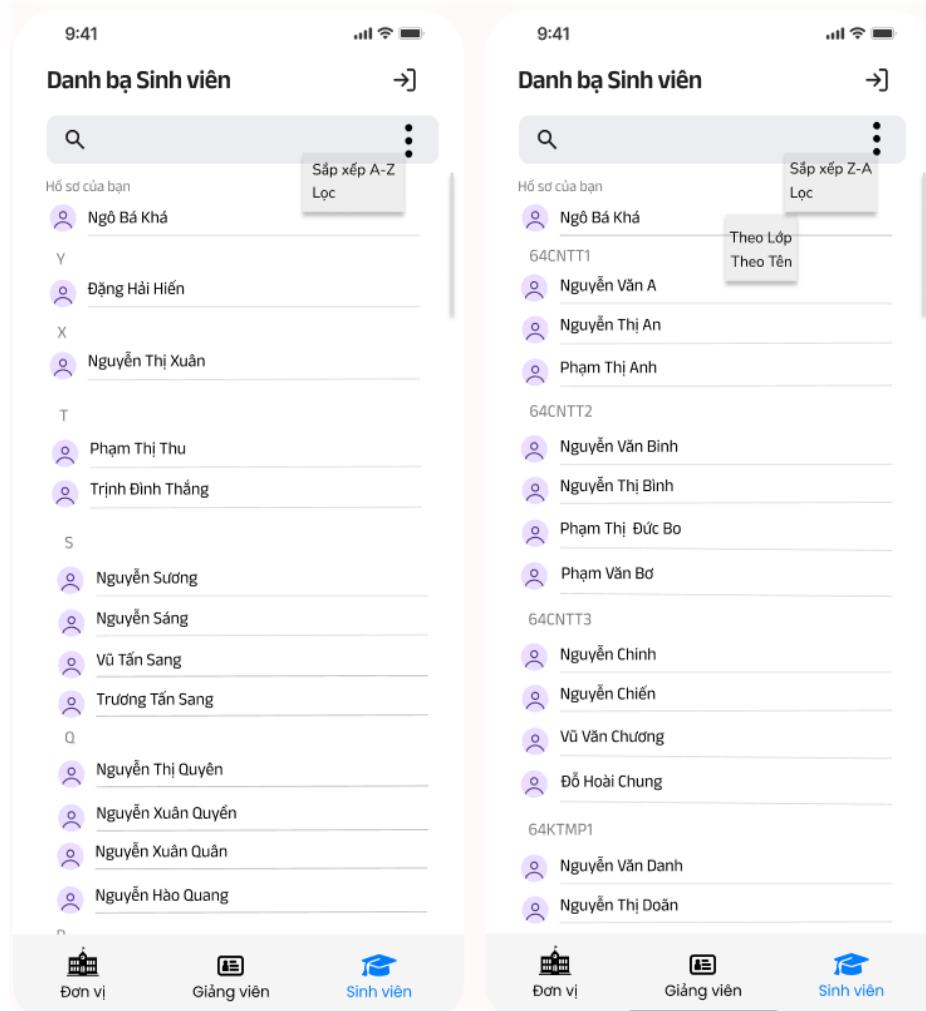
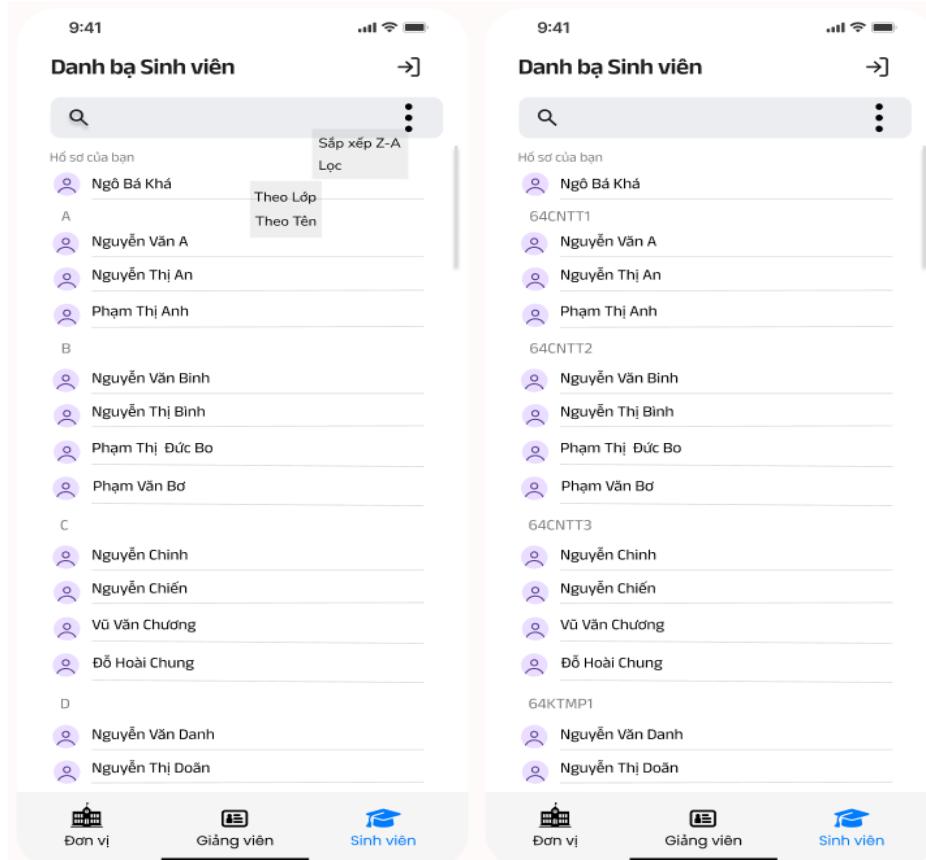
← Thông tin giảng viên

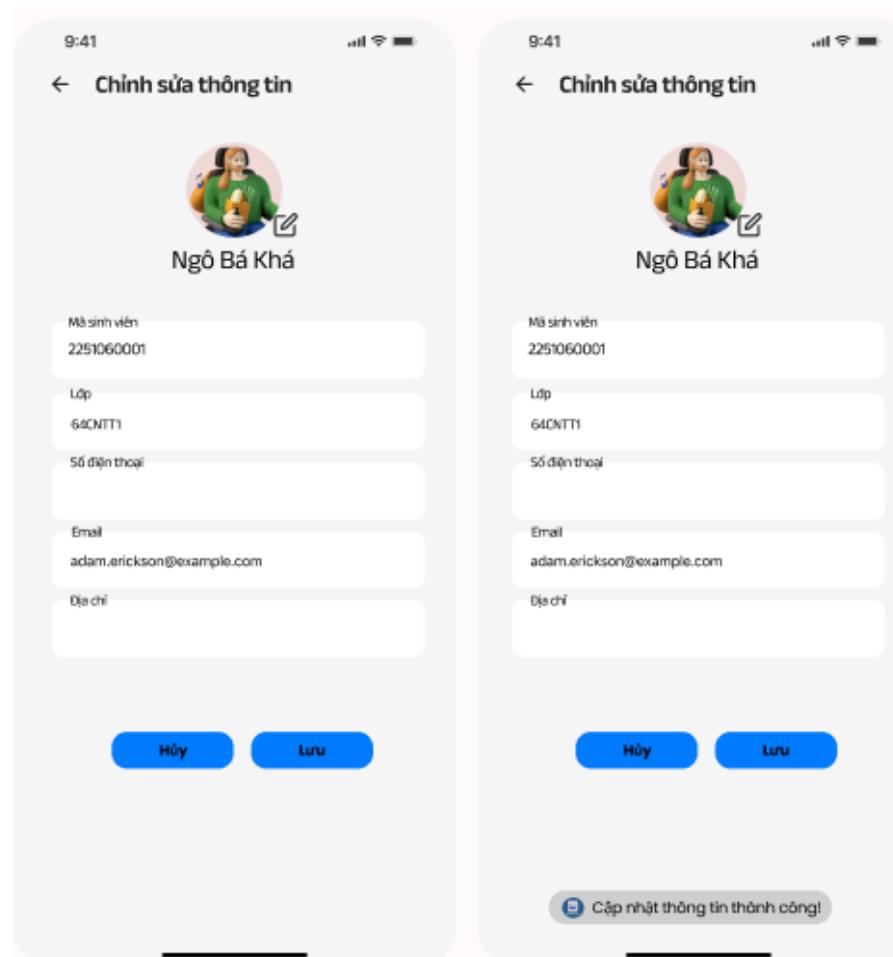
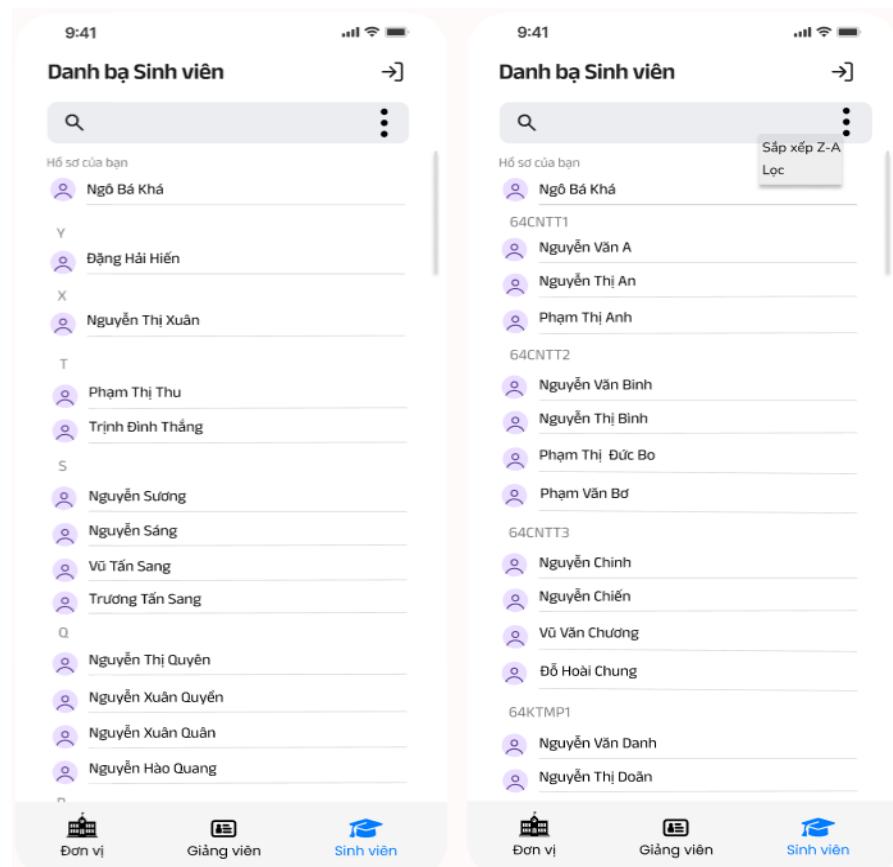

Bùi Thị A

 tin nhắn	 gọi	 gọi video	 mail
Mã giảng viên	225100000		
Chức vụ	Giảng viên		
Số điện thoại	0943628261		
Email	letchia@gmail.com		
Ban vị trực thuộc	Khoa Cơ Khí		
Ghi chú	-		

3.1.1.6. Xem danh bạ sinh viên





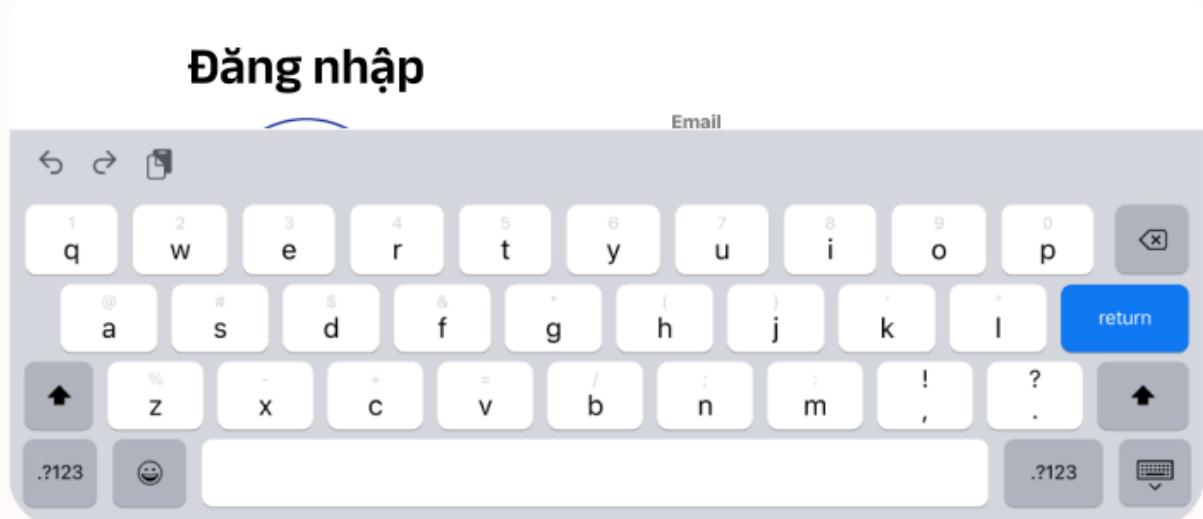


3.1.2. Màn hình ngang

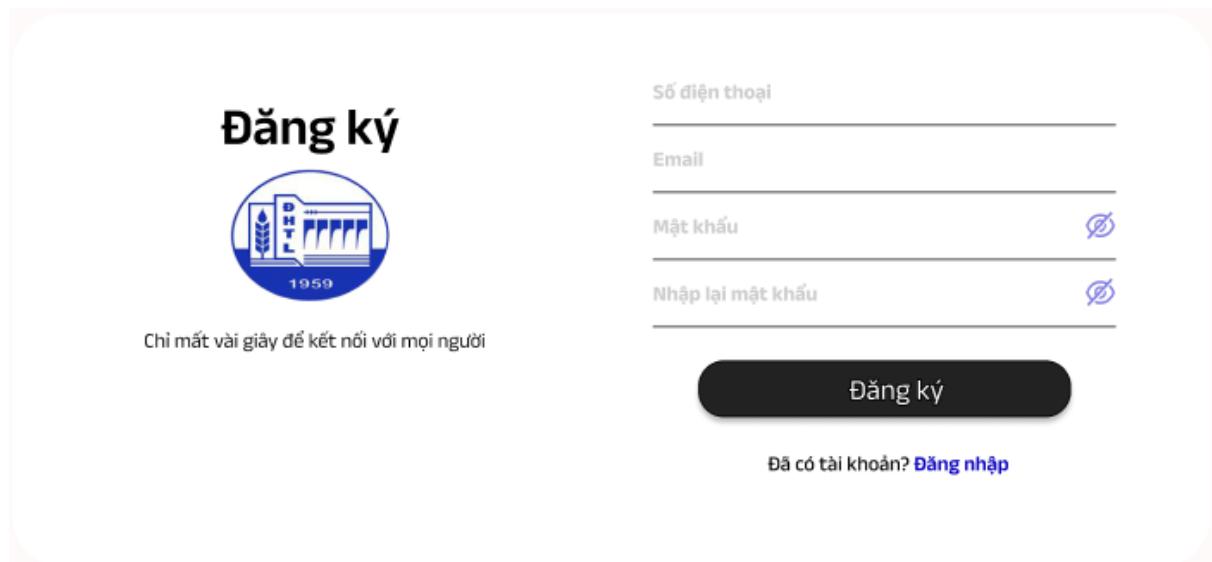
3.1.2.1. Đăng nhập

The screenshot shows the login screen of the TLUContract mobile application. At the top center is the text "Đăng nhập". Below it is the university logo. To the right are fields for "Email" and "Mật khẩu" (Password), each with a placeholder icon. Below the password field is a "Forgot password?" link. A large blue "Đăng nhập" button is centered at the bottom. To the left of the button, there is a link for users without an account: "Không có tài khoản? Đăng ký".

Sign_in



3.1.2.2. Đăng ký



The image shows a registration form titled "Đăng ký" (Register) with a logo of the University of Technology and Education (HCMC) from 1959. The form includes fields for phone number, email, password, and password confirmation, each with a blue eye icon for visibility. A "Đăng ký" button is at the bottom, and a link for existing users to log in ("Đã có tài khoản? Đăng nhập") is also present.

Đăng ký

Số điện thoại

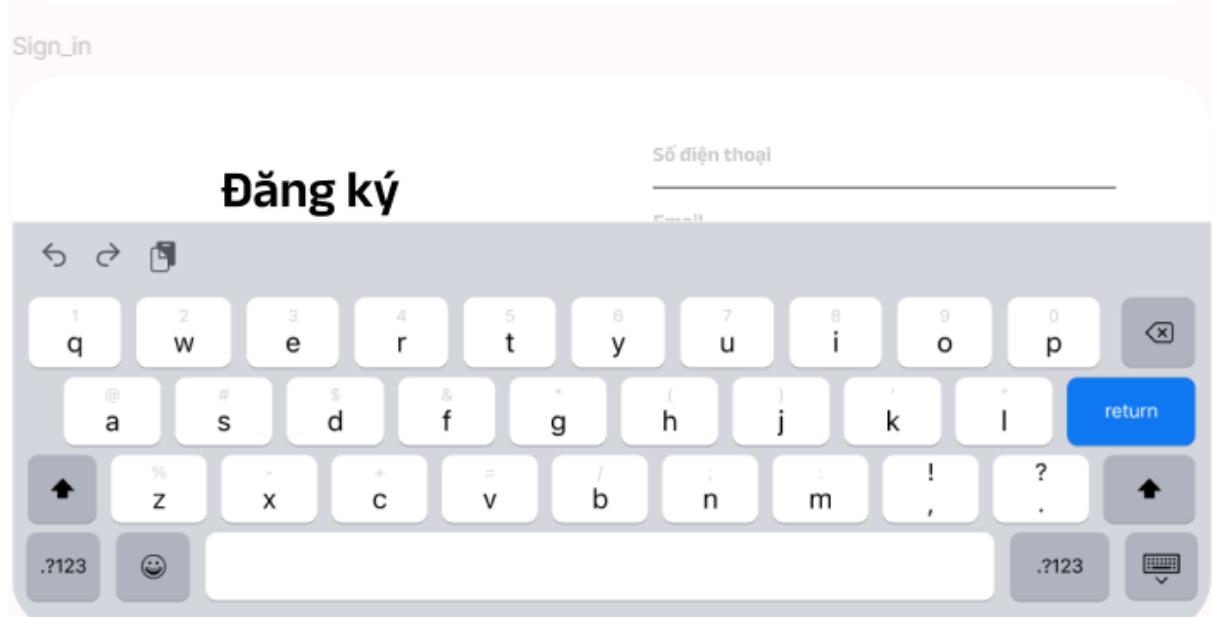
Email

Mật khẩu

Nhập lại mật khẩu

Đăng ký

Đã có tài khoản? [Đăng nhập](#)



The image shows a virtual keyboard interface with a "Sign_in" label at the top left. The keyboard has standard QWERTY layout with additional symbols like @, #, \$, &, *, (,), :, !, ?, ., and ,. There are also numeric keys (1-0), a return key, and special keys for punctuation and symbols. The entire interface is contained within a rounded rectangle.

Sign_in

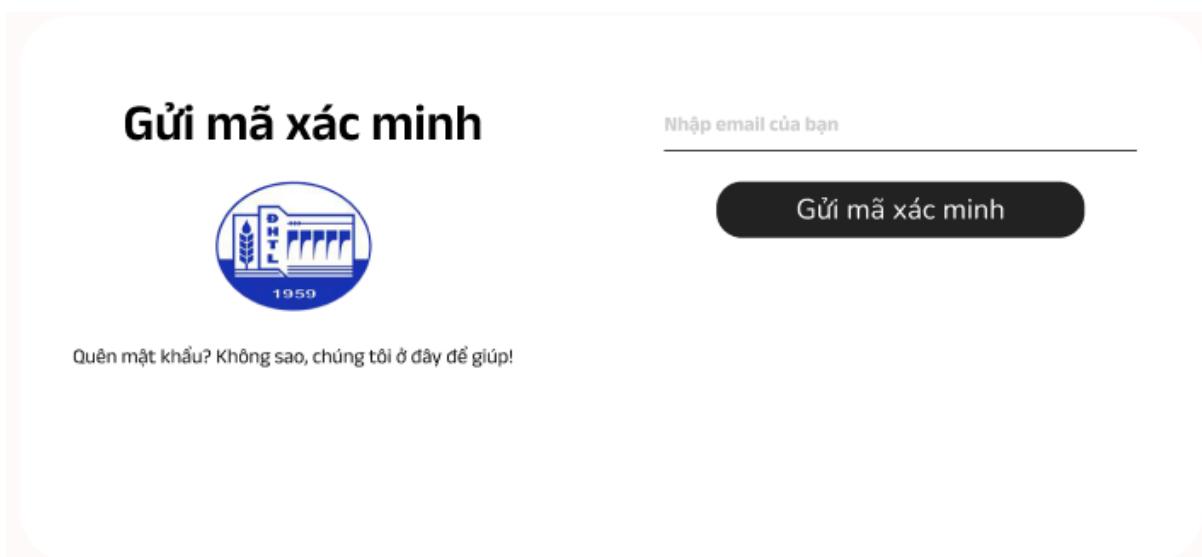
Đăng ký

Số điện thoại

return

?.123

3.1.2.3. Khôi phục mật khẩu



Reset_password_1



3.1.2.4. Danh bạ đơn vị

9:41

Danh bạ đơn vị

Tìm kiếm

K

-  Khoa Cơ khí
-  Khoa Công nghệ thông tin
-  Khoa Công trình
-  Khoa Điện - Điện tử

 Đơn vị  Giảng viên  Sinh viên

9:41

Danh bạ đơn vị

Tìm kiếm

K

-  Khoa Cơ khí
-  Khoa Công nghệ thông tin
-  Khoa Công trình
-  Khoa Điện - Điện tử

 Đơn vị  Giảng viên  Sinh viên

Sắp xếp
Bộ lọc

9:41

Danh bạ đơn vị

Tìm kiếm

V

-  Viện Kỹ thuật tài nguyên nước
-  Viện Kỹ thuật công trình
-  Viện Thuỷ lợi và Môi trường
-  Viện Thuỷ văn, Môi trường và Biến đổi khí hậu

 Đơn vị  Giảng viên  Sinh viên

9:41

Danh bạ đơn vị

Tim kiếm

V

Viện Kỹ thuật tài nguyên nước
Viện Kỹ thuật công trình
Viện Thuỷ lợi và Môi trường
Viện Thuỷ văn, Môi trường và Biến đổi khí hậu

Sắp xếp
Bộ lọc

Đơn vị Giảng viên Sinh viên

9:41

Danh bạ đơn vị

Tim kiếm

K

Khoa Cơ khí
Khoa Công nghệ thông tin
Khoa Công trình
Khoa Điện - Điện tử

Tất cả
Khoa
Phòng
Trung tâm
Viện

Đơn vị Giảng viên Sinh viên

9:41

Danh bạ đơn vị

Tim kiếm

Khoa

Khoa Cơ khí
Khoa Công nghệ thông tin
Khoa Công trình
Khoa Điện - Điện tử

Đơn vị Giảng viên Sinh viên

9:41

Danh bạ đơn vị

Tim kiếm

⋮

Phòng

Phòng Chính trị và Công tác sinh viên

Phòng Đào tạo

Phòng Khảo thí và Đảm bảo chất lượng

Phòng Tài chính - Kế toán

Đơn vị

Giảng viên

Sinh viên

9:41

Danh bạ đơn vị

Tim kiếm

⋮

Trung tâm

Trung tâm Đào tạo quốc tế

Trung tâm Giáo dục Quốc phòng và An ninh

Trung tâm Tin học

Thư viện

Đơn vị

Giảng viên

Sinh viên

9:41

Danh bạ đơn vị

Tim kiếm

⋮

Viện

Viện Kỹ thuật tài nguyên nước

Viện Kỹ thuật công trình

Viện Thuỷ lợi và Môi trường

Viện Thuỷ văn, Môi trường và Biến đổi khí hậu

Đơn vị

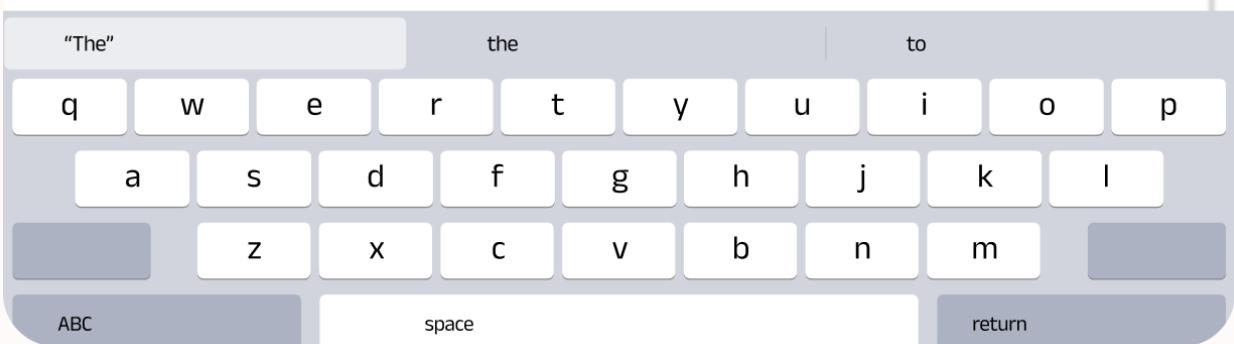
Giảng viên

Sinh viên

9:41



Danh bạ đơn vị



9:41



← Chi tiết đơn vị



Khoa Công nghệ thông tin



goi

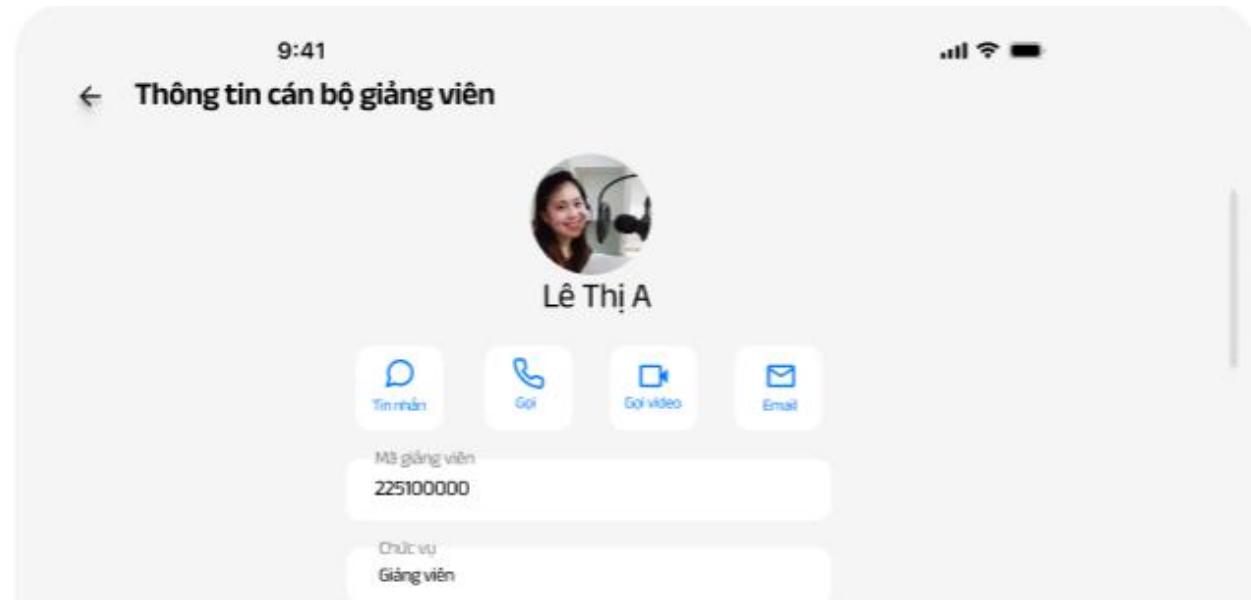
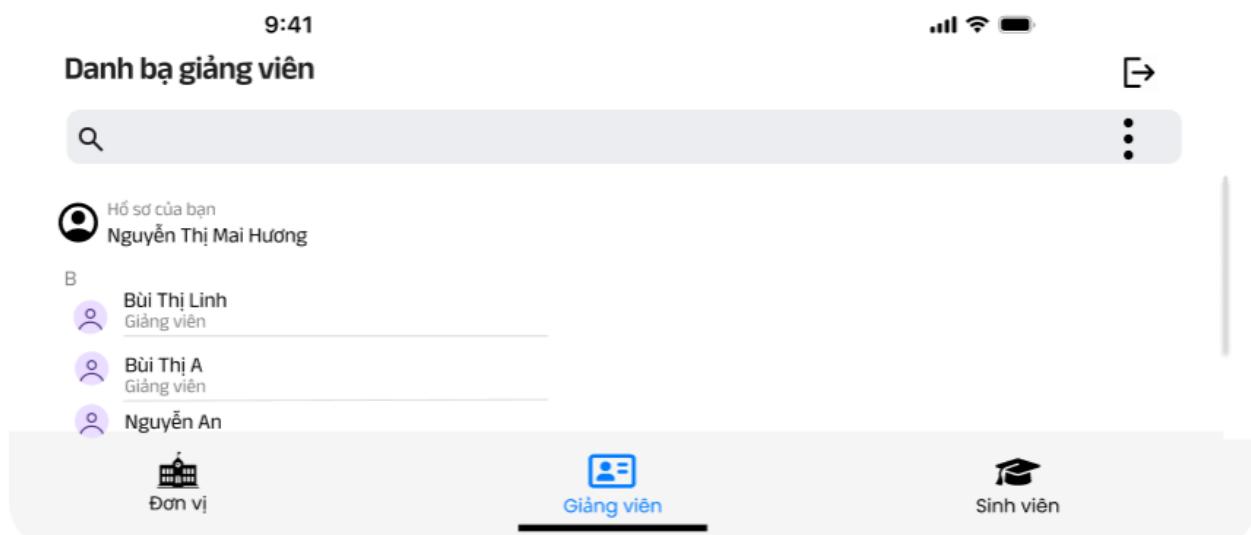


mail

Mã đơn vị

CSE

3.1.2.5. Danh bạ cán bộ giảng viên



9:41
← **Chỉnh sửa thông tin**



Nguyễn Thị Mai Hương

Mã giảng viên
2251061111

Chức vụ
Giảng viên

Số điện thoại
03947646732

9:41
← **Chỉnh sửa thông tin**



Nguyễn Thị Mai Hương

Mã giảng viên
2251061111

Chức vụ
Giảng viên Cập nhật thông tin thành công!

9:41
Danh bạ giảng viên



Hồ sơ của bạn
Nguyễn Thị Mai Hương

Sắp xếp A-Z
Bộ lọc

Y
 Yên Ngọc Linh
Giảng viên

X
 Xuân Thị Thanh
Giảng viên



Đơn vị



Giảng viên



Sinh viên

9:41



Danh bạ giảng viên



Sắp xếp Z-A

Bộ lọc



Hồ sơ của bạn

Nguyễn Thị Mai Hương

B



Bùi Thị Linh

Giảng viên



Bùi Thị A

Giảng viên



Nguyễn An



Đơn vị



Giảng viên



Sinh viên

9:41



Danh bạ giảng viên



Sắp xếp Z-A

Bộ lọc



Hồ sơ của bạn

Nguyễn Thị Mai Hương

Khoa Công nghệ thông tin

Nguyễn Thu Thủy

Giảng viên



Trần Văn Nam

Giảng viên



Nguyễn Thị Vân

Giảng viên



Đơn vị



Giảng viên



Sinh viên

9:41



Danh bạ giảng viên



Sắp xếp Z-A

Bộ lọc



Hồ sơ của bạn

Nguyễn Thị Mai Hương

Giảng viên

Nguyễn Thu Thủy

Giảng viên



Trần Văn Nam

Giảng viên



Nguyễn Thị Vân

Giảng viên



Đơn vị



Giảng viên



Sinh viên

9:41



Danh bạ giảng viên



Sắp xếp Z-A

Bộ lọc

Tất cả

Đơn vị

Chức vụ



Hồ sơ của bạn

Nguyễn Thị Mai Hương

B



Bùi Thị Linh

Giảng viên



Bùi Thị A

Giảng viên



Nguyễn An



Đơn vị



Giảng viên



Sinh viên

3.1.2.6. Danh bạ sinh viên

9:41



Danh bạ Sinh viên



Hồ sơ của bạn



Ngô Bá Khá

A



Nguyễn Văn A



Nguyễn Thị An



Phạm Thị Anh



Đơn vị



Giảng viên



Sinh viên

9:41



← Thông tin sinh viên



Nguyễn Thị An



Tin nhắn



Gọi

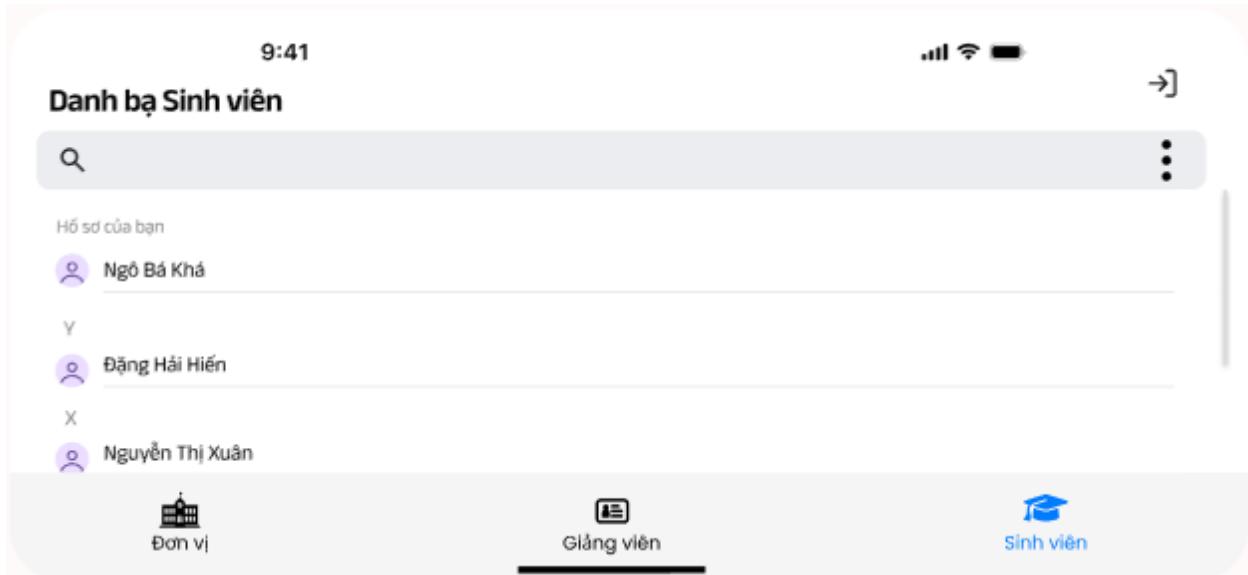
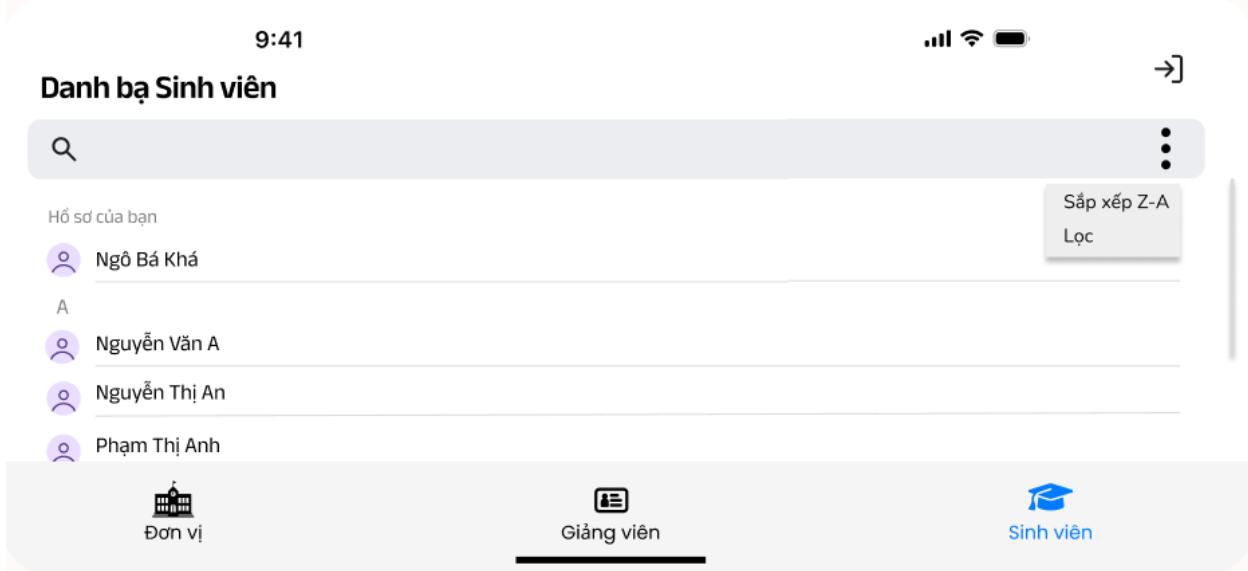
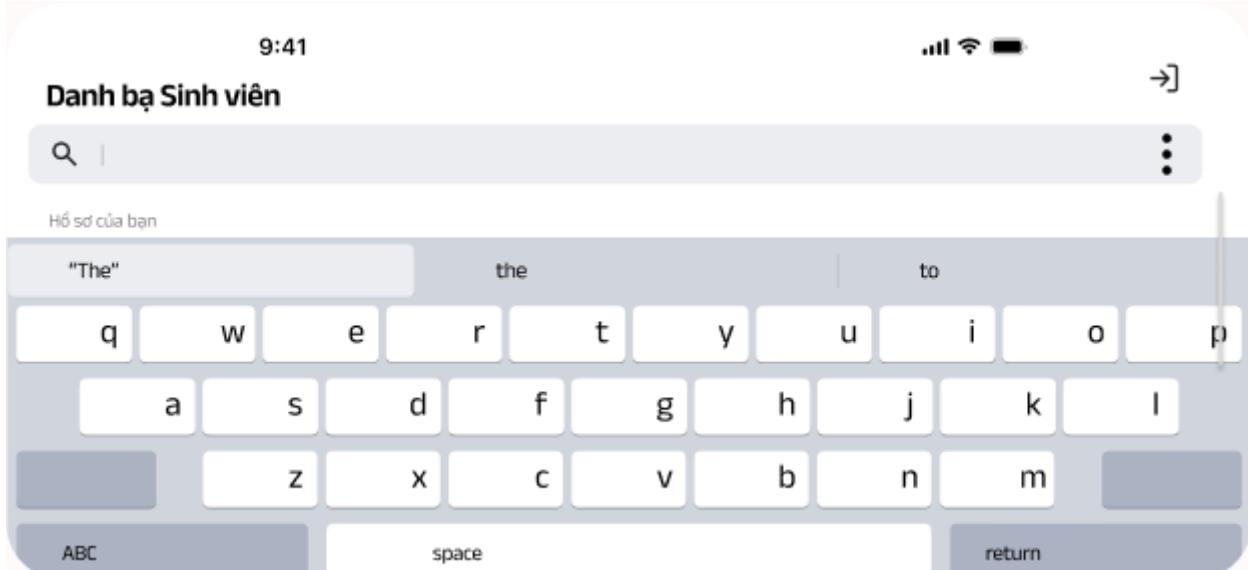


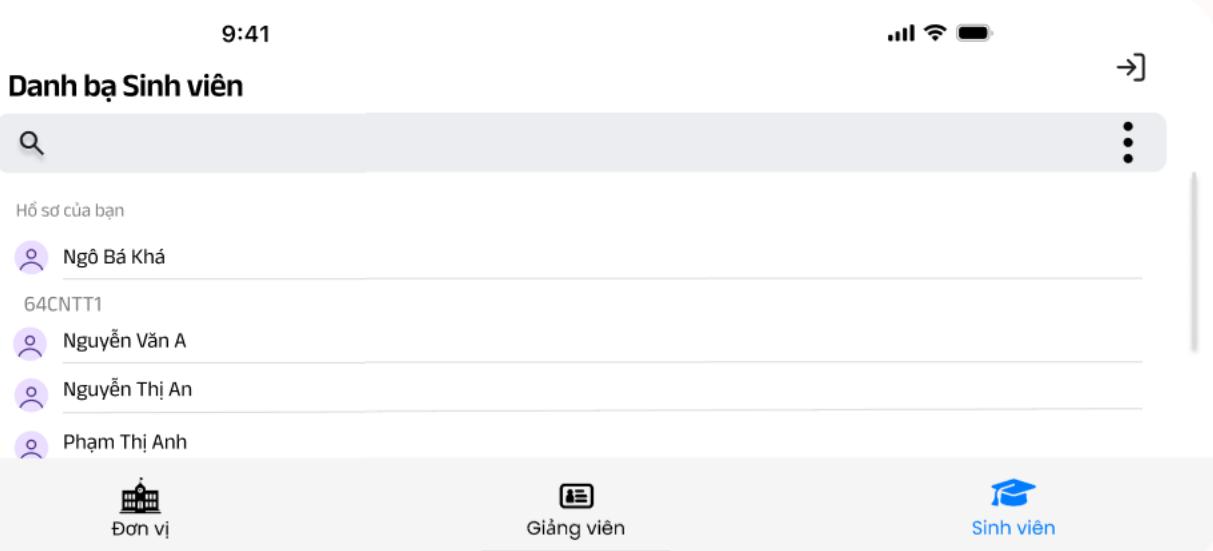
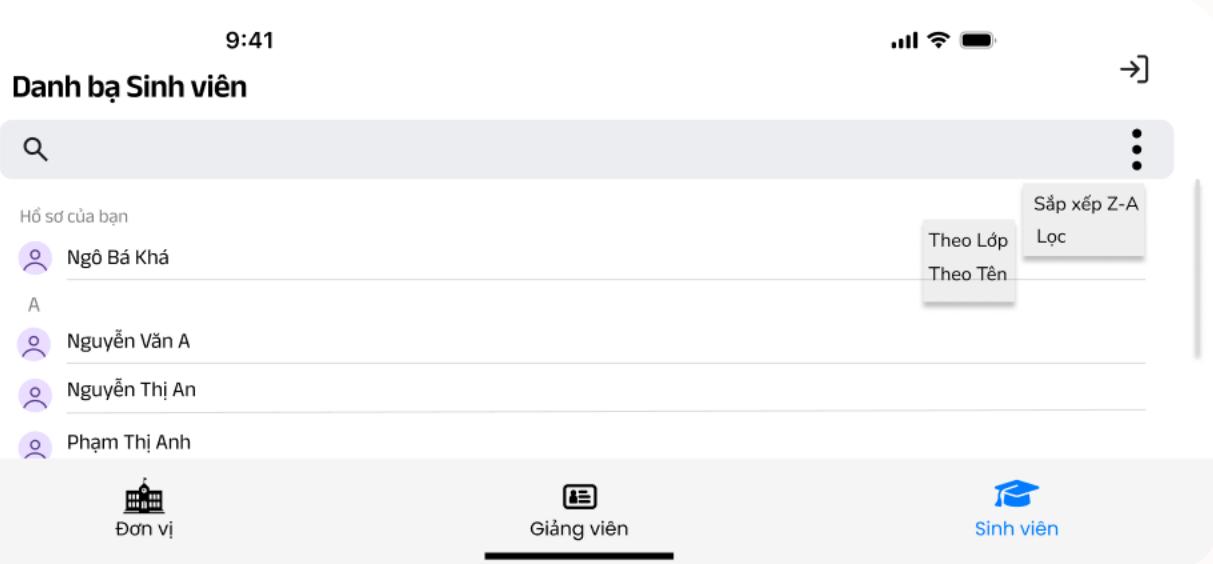
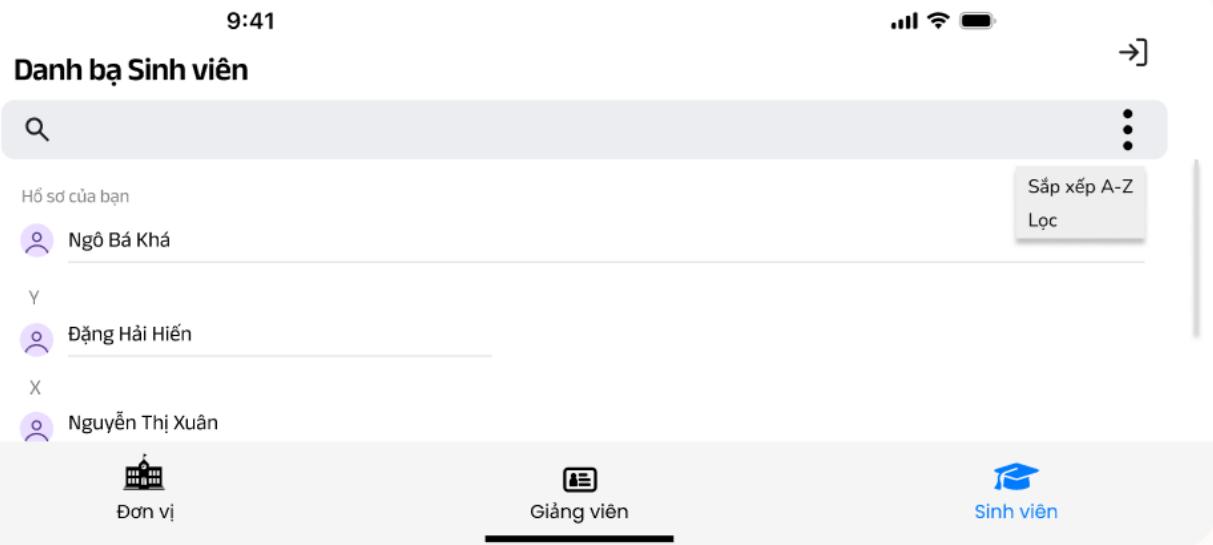
Gọi video



Email

Mã sinh viên
2251060001





9:41

Danh bạ Sinh viên

Hồ sơ của bạn

Sắp xếp Z-A
Lọc

Ngô Bá Khá
64CNTT1

Nguyễn Văn A

Nguyễn Thị An

Phạm Thị Anh

Đơn vị Giảng viên Sinh viên

This screenshot shows a mobile application interface for managing student contacts. At the top, there's a search bar with a magnifying glass icon and a menu icon with three dots. Below the search bar, there's a section for 'Hồ sơ của bạn' (Your profile) with sorting and filtering options ('Sắp xếp Z-A', 'Lọc'). The main list contains four entries: 'Ngô Bá Khá' (with '64CNTT1' below it), 'Nguyễn Văn A', 'Nguyễn Thị An', and 'Phạm Thị Anh'. At the bottom, there are three tabs: 'Đơn vị' (Department), 'Giảng viên' (Lecturer, which is selected and highlighted in black), and 'Sinh viên' (Student).

9:41

Danh bạ Sinh viên

Hồ sơ của bạn

Sắp xếp Z-A
Lọc

Theo Tên
Theo Lớp

Ngô Bá Khá
64CNTT1

Nguyễn Văn A

Nguyễn Thị An

Phạm Thị Anh

Đơn vị Giảng viên Sinh viên

This screenshot is similar to the one above, showing the student contact list. It includes a search bar, sorting/filtering options ('Sắp xếp Z-A', 'Lọc', 'Theo Tên', 'Theo Lớp'), and a list of four students: 'Ngô Bá Khá' (with '64CNTT1' below it), 'Nguyễn Văn A', 'Nguyễn Thị An', and 'Phạm Thị Anh'. The 'Giảng viên' tab is selected at the bottom.

9:41

← Chính sửa thông tin

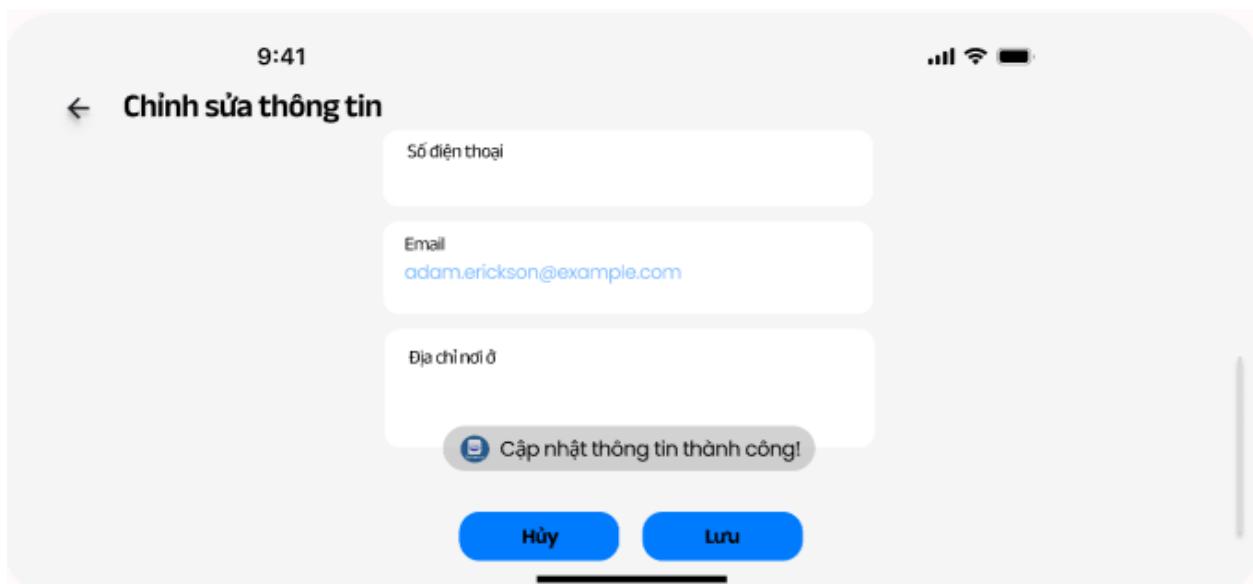
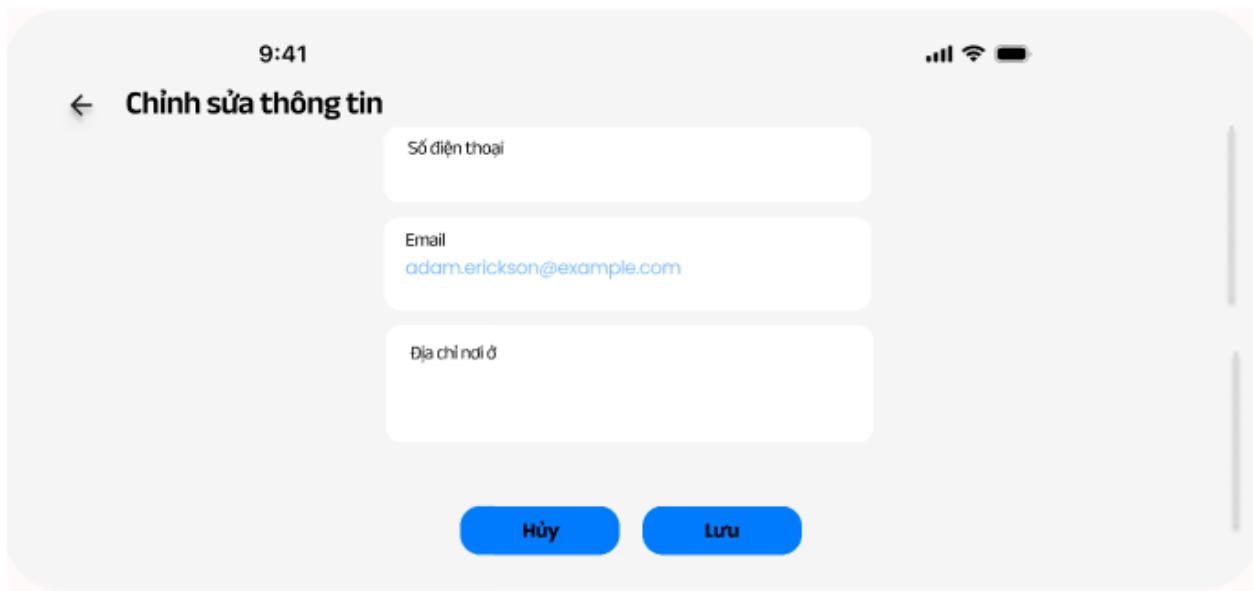
Ngô Bá Khá

Mã sinh viên
2251060001

Lớp
64CNTT1

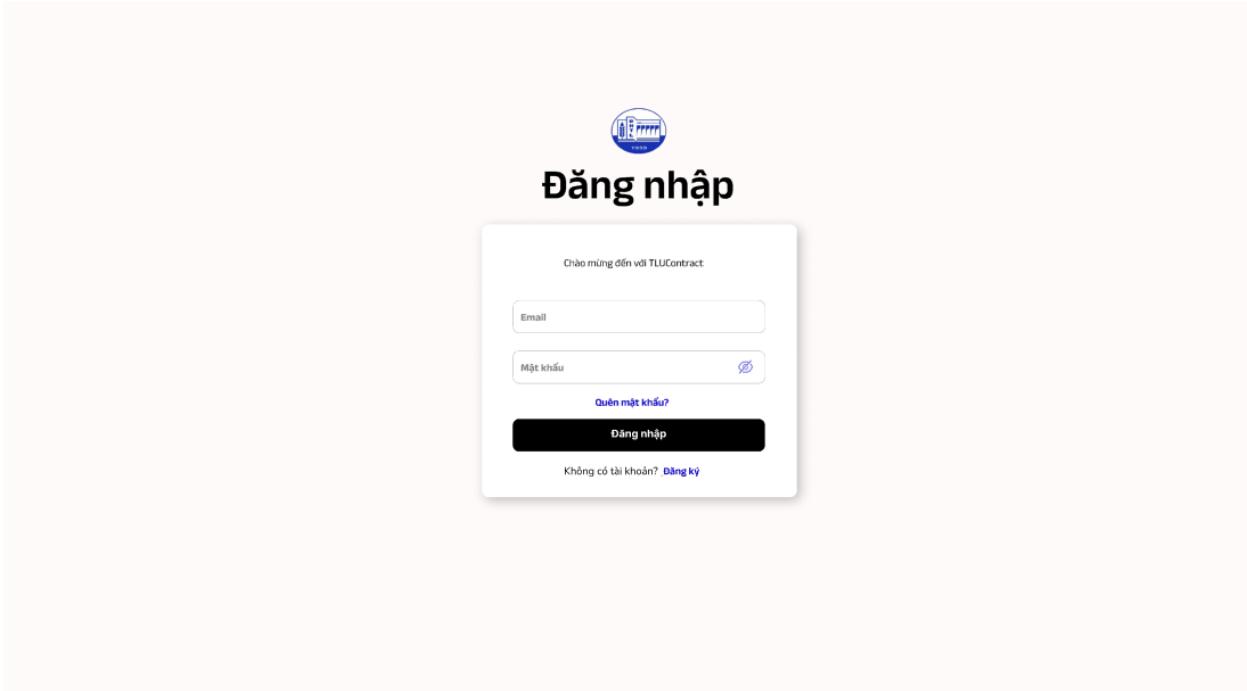
Số điện thoại

This screenshot shows the 'Chỉnh sửa thông tin' (Edit information) screen for 'Ngô Bá Khá'. It features a circular profile picture of a person holding a book and a pen. Below the name, there are three input fields: 'Mã sinh viên' (Student ID) containing '2251060001', 'Lớp' (Class) containing '64CNTT1', and 'Số điện thoại' (Phone number), which is currently empty. There is also a back arrow icon on the left.



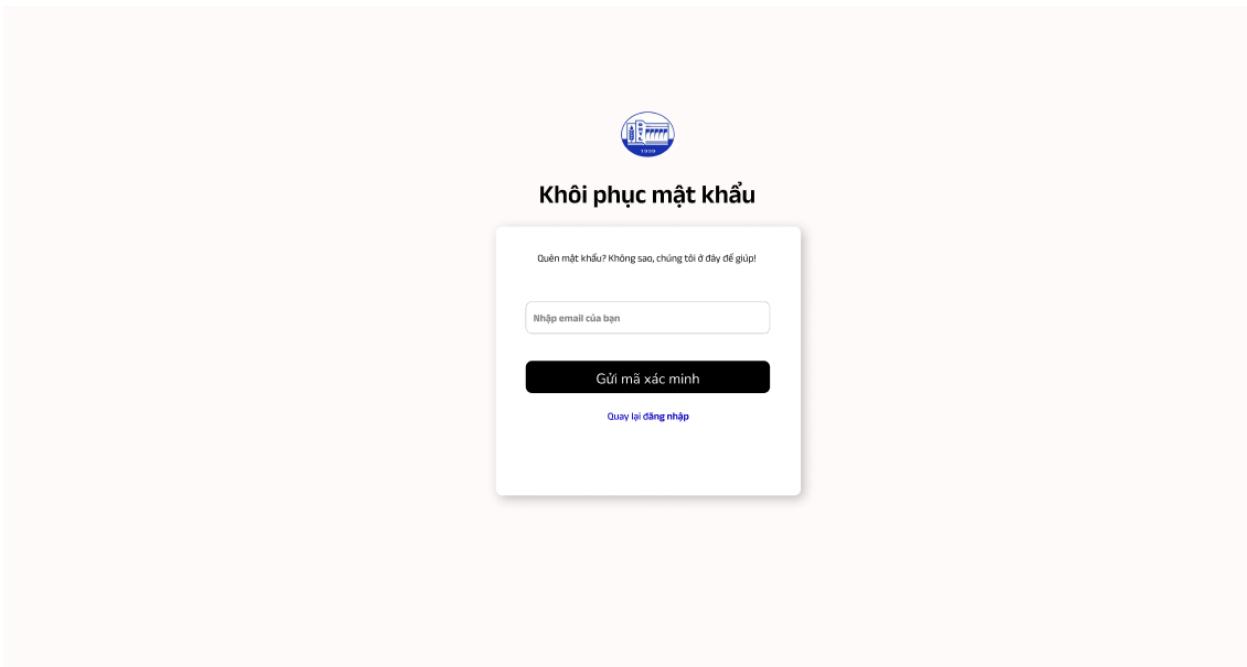
3.1.3 Màn hình web

3.1.3.1. Đăng nhập



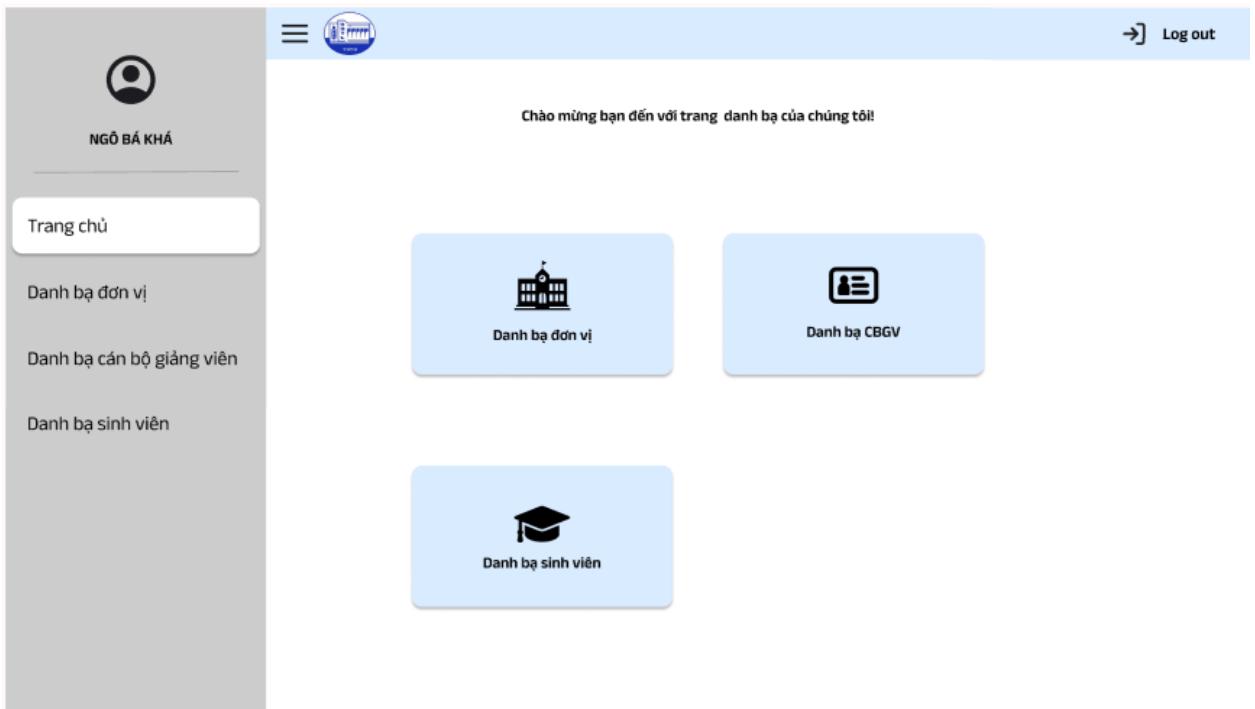
The screenshot shows the login page for TLUContract. At the top center is the university's logo. Below it, the word "Đăng nhập" (Login) is displayed in a large, bold font. A sub-header "Chào mừng đến với TLUContract" (Welcome to TLUContract) is present above the input fields. There are two input fields: one for "Email" and one for "Mật khẩu" (Password), which includes a small icon of an eye for password visibility. Below these fields are two buttons: "Quên mật khẩu?" (Forgot password?) and a large black button labeled "Đăng nhập" (Login). At the bottom of the form, a link "Không có tài khoản? Đăng ký" (Don't have an account? Register) is visible.

3.1.3.2. Quên mật khẩu

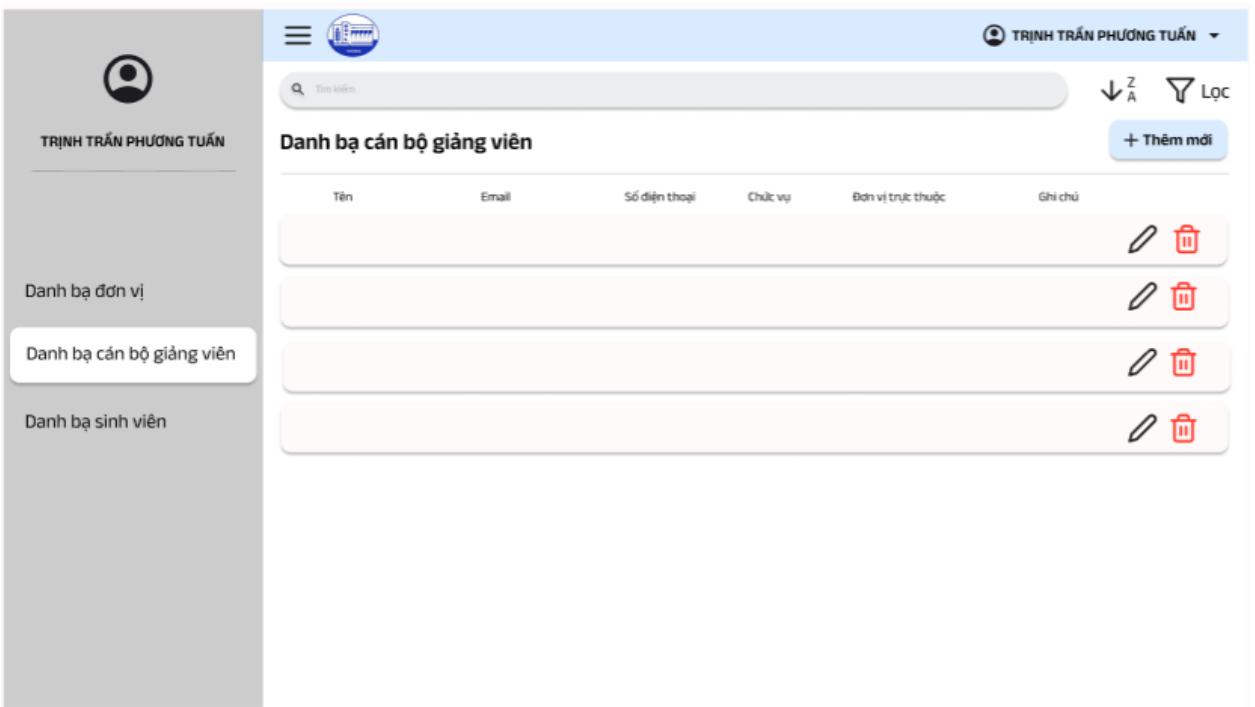


The screenshot shows the password recovery page for TLUContract. At the top center is the university's logo. Below it, the title "Khôi phục mật khẩu" (Reset password) is centered. A sub-header "Quên mật khẩu? Không sao, chúng tôi ở đây để giúp!" (Forgot password? No problem, we're here to help!) is displayed above the input field. There is a single input field labeled "Nhập email của bạn" (Enter your email). Below this is a large black button labeled "Gửi mã xác minh" (Send verification code). At the bottom of the form, a link "Quay lại đăng nhập" (Return to login) is visible.

3.1.3.3. Trang chủ



3.1.3.4. Danh bạ cán bộ giảng viên



The screenshot shows the 'Danh bạ cán bộ giảng viên' (Faculty Staff Directory) page. At the top right, the user's name 'NGUYỄN THỊ MAI HƯƠNG' is displayed with a profile icon. To the right of the name are icons for sorting ('↓ A'), filtering ('Lọc'), and a search bar with placeholder text 'Tìm kiếm'. On the far right, there are buttons for '+ Thêm mới' (Add new) and a file upload dialog box labeled 'Tải tệp lên' (Upload file). The main content area displays a table with columns: Tên (Name), Email, Số điện thoại (Phone number), Chức vụ (Position), and Đơn vị trực thuộc (Directly affiliated unit). Below the table are five rows, each with edit and delete icons. The first row has a gray background, indicating it is selected or highlighted.

This screenshot shows the same 'Danh bạ cán bộ giảng viên' (Faculty Staff Directory) page, but with a file upload dialog box overlaid. The dialog box has a light orange background and contains the text 'Kéo thả tệp vào đây' (Drag and drop files here). It includes a 'Tên' (Name) input field, a 'Ghi chú' (Note) section with edit and delete icons, and two buttons at the bottom: 'Hủy' (Cancel) and 'Thêm mới' (Add new). The rest of the interface is visible in the background.



NGUYỄN THỊ MAI HƯƠNG

- [Trang chủ](#)
- [Danh bạ đơn vị](#)
- [Danh bạ cán bộ giảng viên](#)
- [Danh bạ sinh viên](#)

 Tim kiếm
↓ A
Lọc

Danh bạ cán bộ giảng viên

Thêm cán bộ giảng viên mới

Tên	Mã cán bộ giảng viên
<input type="text"/>	<input type="text"/>
Email	Số điện thoại
<input type="text"/>	<input type="text"/>
Chức vụ	Ghi chú
<input type="text"/>	<input type="text"/>
Đơn vị trực thuộc	
Hủy Lưu lại	



NGUYỄN THỊ MAI HƯƠNG

- [Trang chủ](#)
- [Danh bạ đơn vị](#)
- [Danh bạ cán bộ giảng viên](#)
- [Danh bạ sinh viên](#)

 Tim kiếm
↓ A
Lọc

Danh bạ cán bộ giảng viên

Sửa thông tin

Tên	Mã cán bộ giảng viên
<input type="text"/>	<input type="text"/>
Email	Số điện thoại
<input type="text"/>	<input type="text"/>
Chức vụ	Ghi chú
<input type="text"/>	<input type="text"/>
Đơn vị trực thuộc	
Hủy Lưu lại	

NGUYỄN THỊ MAI HƯƠNG

Danh bạ cán bộ giảng viên

+ Thêm mới

Tên Email Số điện thoại Chức vụ Đơn vị trực thuộc Ghị chú

Bạn có muốn xóa cán bộ giảng viên này?

Hủy Xóa

3.1.3.5. Danh bạ sinh viên

NGÔ BÁ KHÁ

Danh bạ sinh viên

+ Thêm sinh viên mới

Tên Lớp Email Mã sinh viên Địa chỉ Số điện thoại

Bạn có muốn xóa sinh viên này?

Hủy Xóa

The screenshot shows the 'Danh bạ sinh viên' (Student Directory) page. At the top right are 'Log out' and search/filter icons. Below is a table with columns: Tên (Name), Lớp (Class), Email, Mã sinh viên (Student ID), and Địa chỉ (Address). A grey button bar at the top right contains 'Tải tệp lên' (Upload file) and 'Thêm SV mới' (Add new student). On the left sidebar, under 'Danh bạ sinh viên', there is a sub-menu item 'Danh bạ sinh viên'.

This screenshot shows a modal dialog for file upload. It has a dashed rectangular area in the center with the text 'Kéo thả tệp vào đây' (Drag and drop files here). At the bottom are 'Hủy' (Cancel) and 'Thêm mới' (Add new) buttons. The background shows the same student directory interface as the first screenshot.

The screenshot shows the 'Danh bạ sinh viên' (Student Address Book) section. On the left, a sidebar menu includes 'Trang chủ', 'Danh bạ đơn vị', 'Danh bạ cán bộ giảng viên', and 'Danh bạ sinh viên', with 'Danh bạ sinh viên' being the active tab. The main area displays a form titled 'Sửa thông tin sinh viên' (Edit Student Information). It contains fields for 'Tên' (Name), 'Mã sinh viên' (Student ID), 'Lớp' (Class), 'Email', 'Số điện thoại' (Phone Number), and 'Địa chỉ nơi ở' (Address). At the bottom right are 'Hủy' (Cancel) and 'Lưu lại' (Save) buttons.

The screenshot shows the 'Danh bạ sinh viên' (Student Address Book) section. The sidebar menu is identical to the previous screenshot. The main area displays a list of student information with columns for 'Tên' (Name), 'Lớp' (Class), 'Email', 'Số điện thoại' (Phone Number), 'Địa chỉ' (Address), and 'Ghi chú' (Notes). Each row has edit and delete icons. A modal dialog box is centered over the list, asking 'Bạn muốn xóa sinh viên này?' (Do you want to delete this student?). It includes 'Hủy' (Cancel) and 'Xóa' (Delete) buttons.

NGÔ BÁ KHÁ

Danh bạ sinh viên

Thêm sinh viên mới

Tên	Mã sinh viên
Lớp	Email
Số điện thoại	Địa chỉ nơi ở

Hủy Lưu lại

3.1.3.6. Danh bạ đơn vị

Châu Thiên Mẫn

Danh bạ đơn vị

Tên	Email	Số điện thoại	Địa chỉ	Đơn vị con	Ghi chú

+ Thêm mới

The screenshot shows the 'Danh bạ đơn vị' (Unit Contact Book) page. At the top right are 'Logout' and search/filter buttons. Below is a table with columns: Tên (Name), Email, Số điện thoại (Phone Number), Địa chỉ (Address), and Đơn vị con (Sub-unit). A modal window is open in the bottom right corner with buttons 'Tải tệp lên' (Upload file) and 'Thêm DV mới' (Add new unit).

The screenshot shows the same 'Danh bạ đơn vị' page. A large orange modal window is centered over the table, containing a dashed box with the text 'Kéo thả tệp vào đây' (Drag and drop file here). At the bottom of the modal are 'Hủy' (Cancel) and 'Thêm mới' (Add new) buttons.

Châu Thiên Mẫn

Trang chủ

Danh bạ đơn vị

Danh bạ cán bộ giảng viên

Danh bạ sinh viên

Danh bạ đơn vị

Thêm đơn vị mới

Tên	Mã đơn vị
<input type="text"/>	<input type="text"/>
Email	Số điện thoại
<input type="text"/>	<input type="text"/>
Địa chỉ	Ghi chú
<input type="text"/>	<input type="text"/>
Đơn vị con (nếu có)	
<input type="text"/>	

Hủy **Lưu lại**

Châu Thiên Mẫn

Trang chủ

Danh bạ đơn vị

Danh bạ cán bộ giảng viên

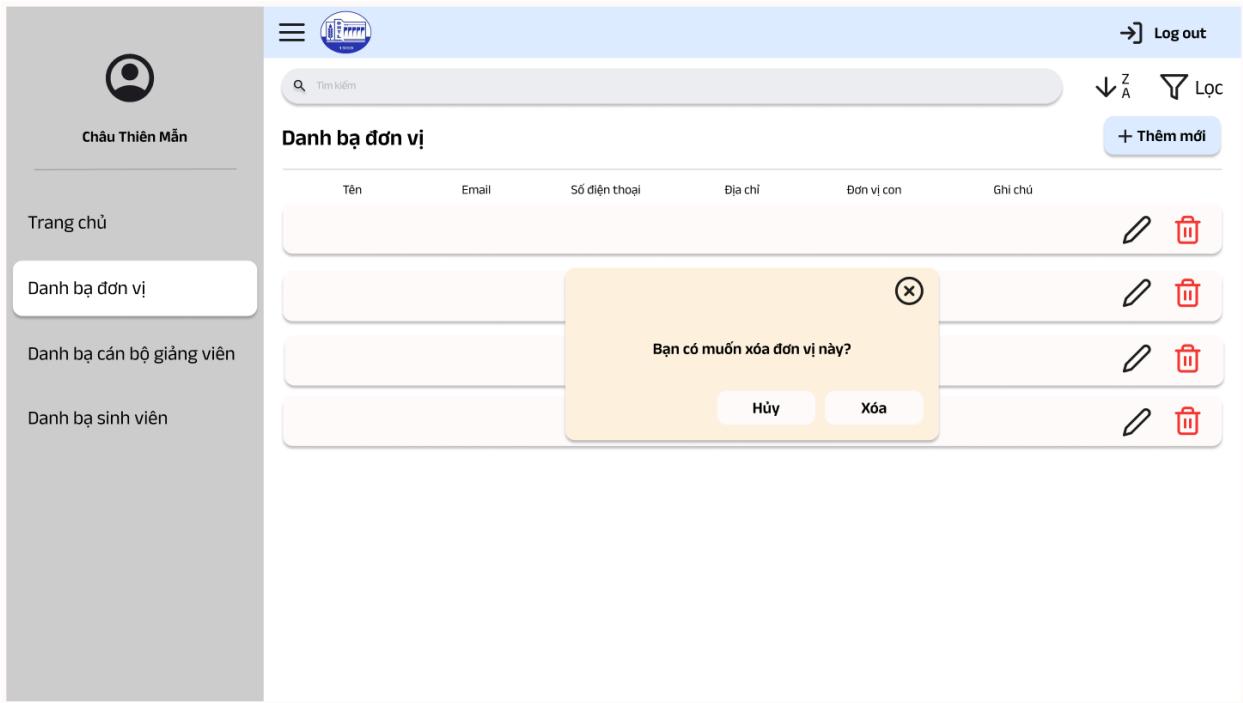
Danh bạ sinh viên

Danh bạ đơn vị

Sửa thông tin

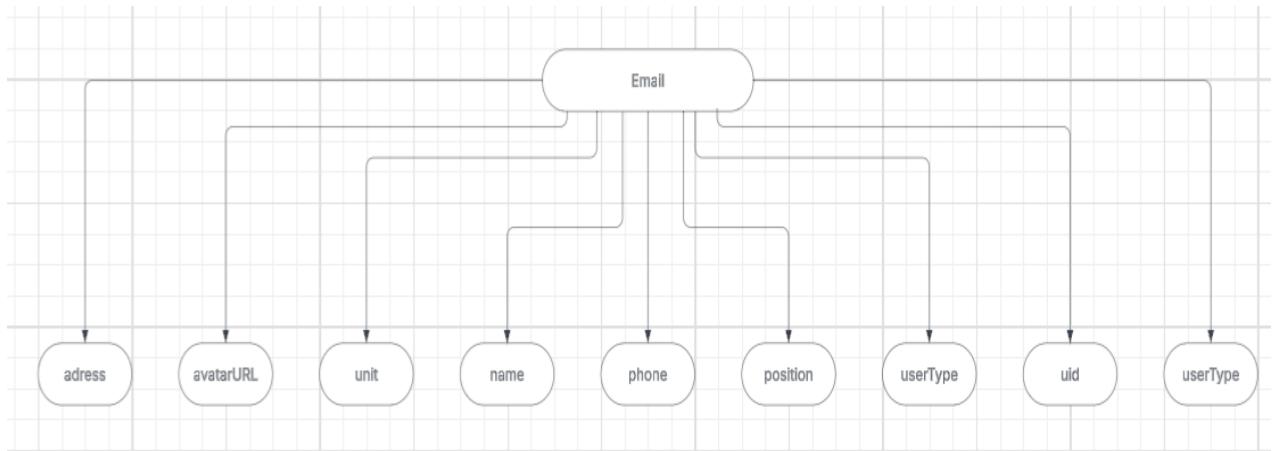
Tên	Mã đơn vị
<input type="text"/>	<input type="text"/>
Email	Số điện thoại
<input type="text"/>	<input type="text"/>
Địa chỉ	Ghi chú
<input type="text"/>	<input type="text"/>
Đơn vị con (nếu có)	
<input type="text"/>	

Hủy **Lưu lại**

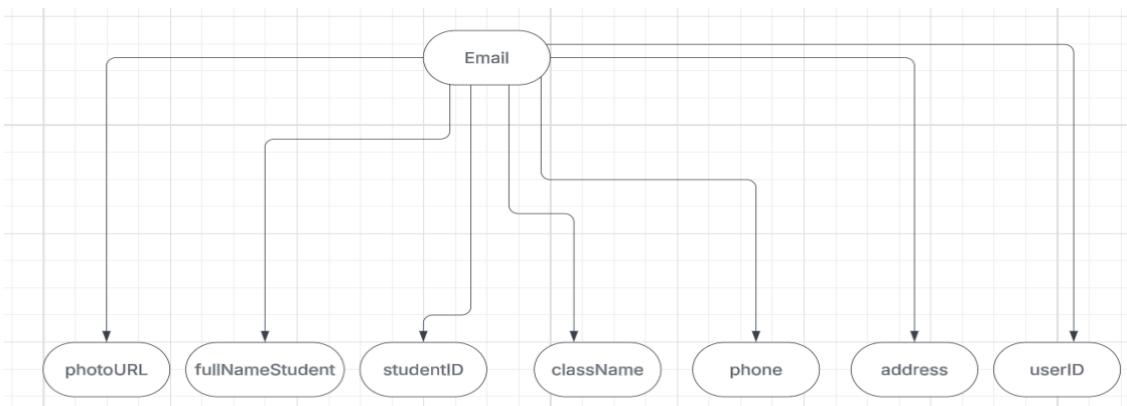


3.2. Thiết kế CSDL

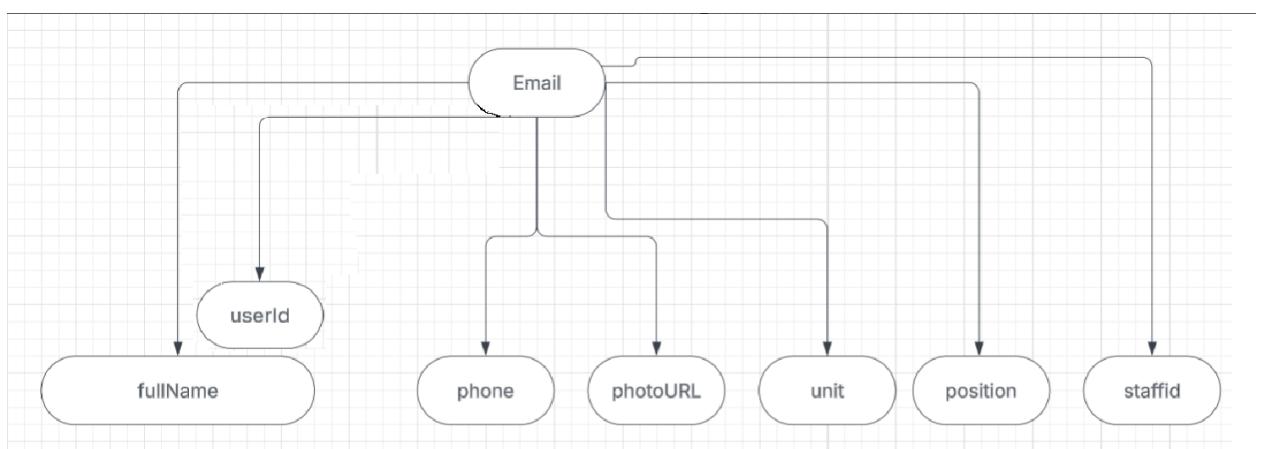
3.2.1. Cơ sở dữ liệu của guest



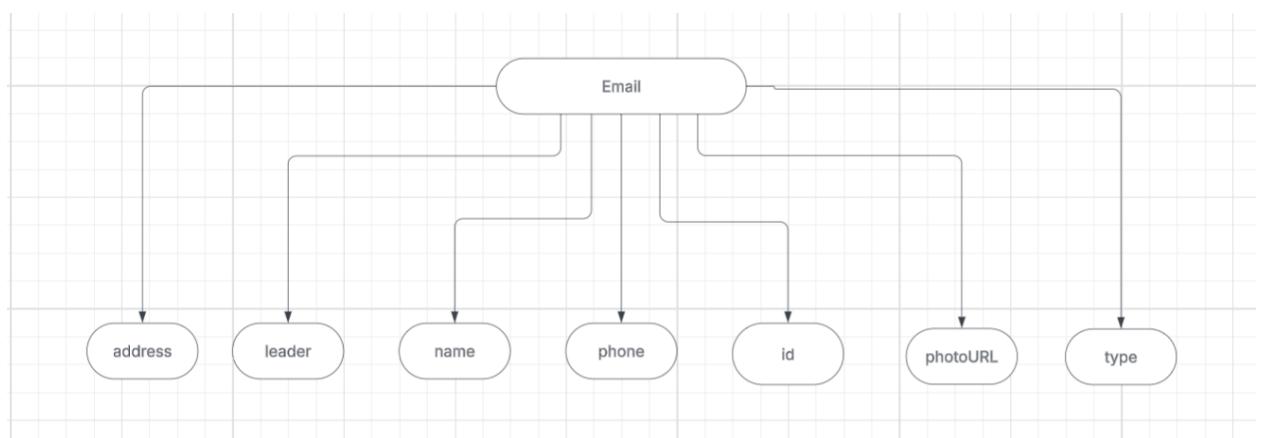
3.2.2. Cơ sở dữ liệu của Student



3.2.3. Cơ sở dữ liệu của Staff



3.2.4. Cơ sở dữ liệu của Department

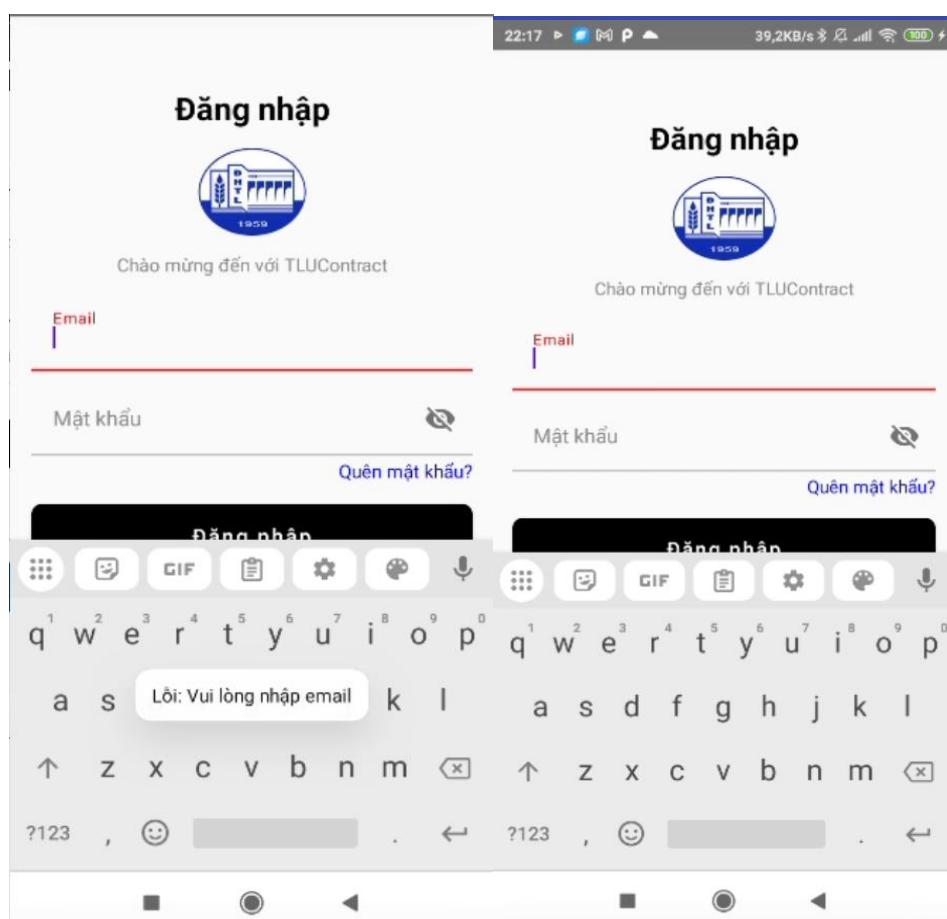
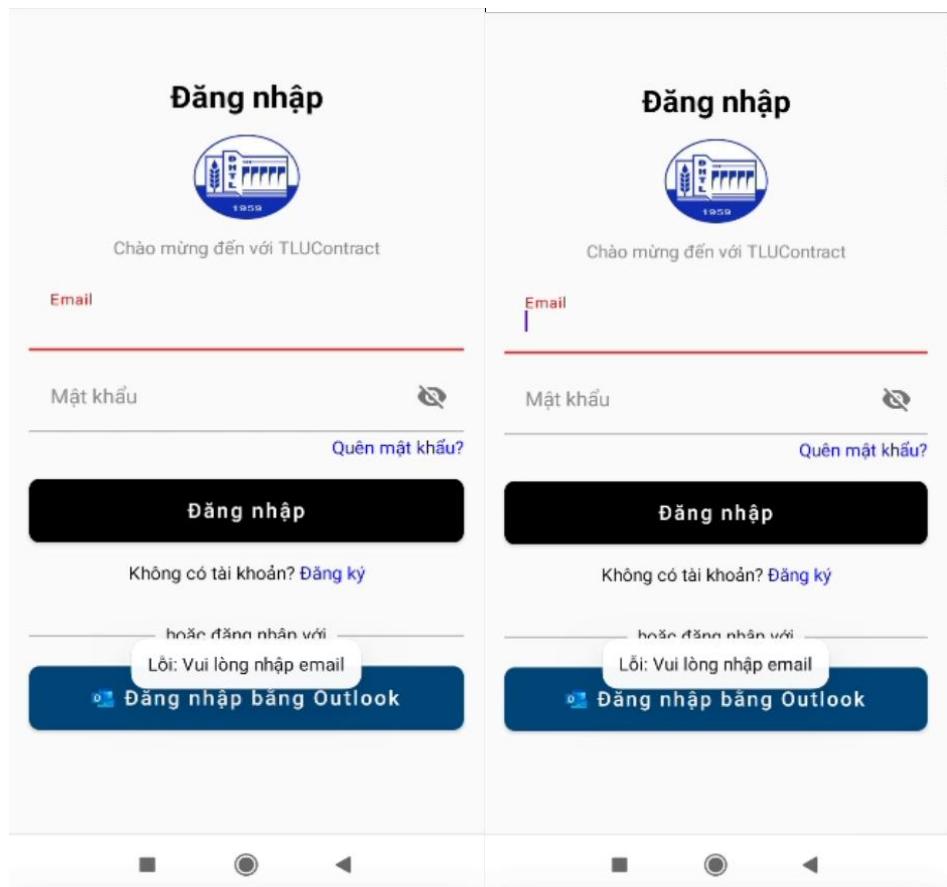


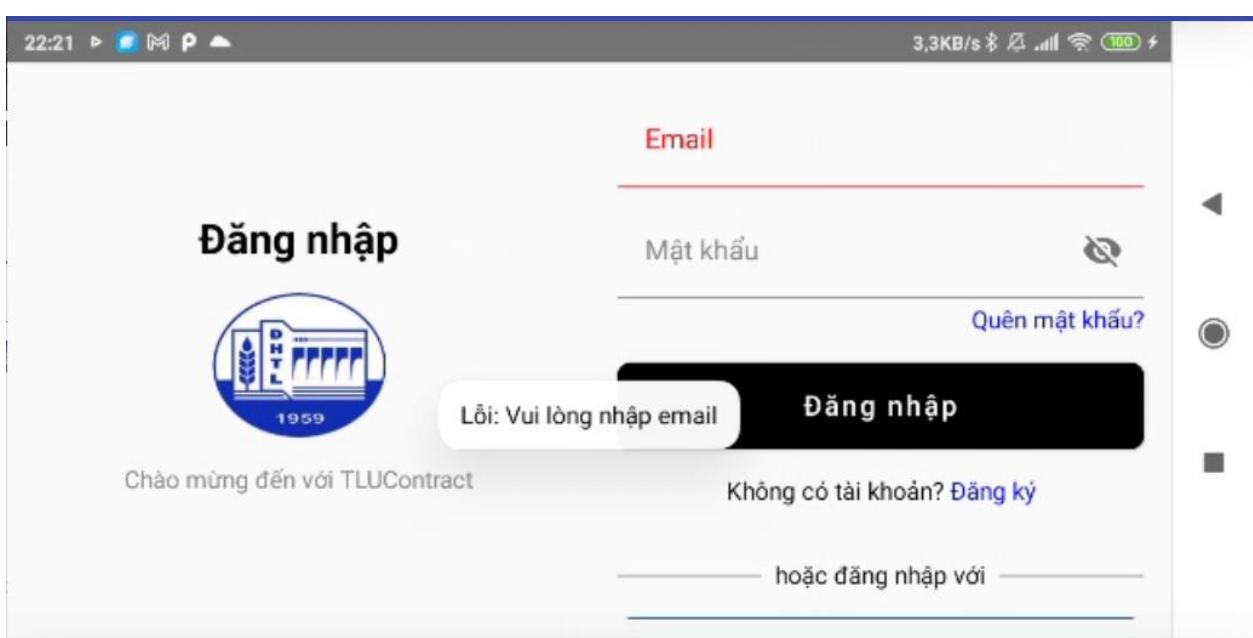
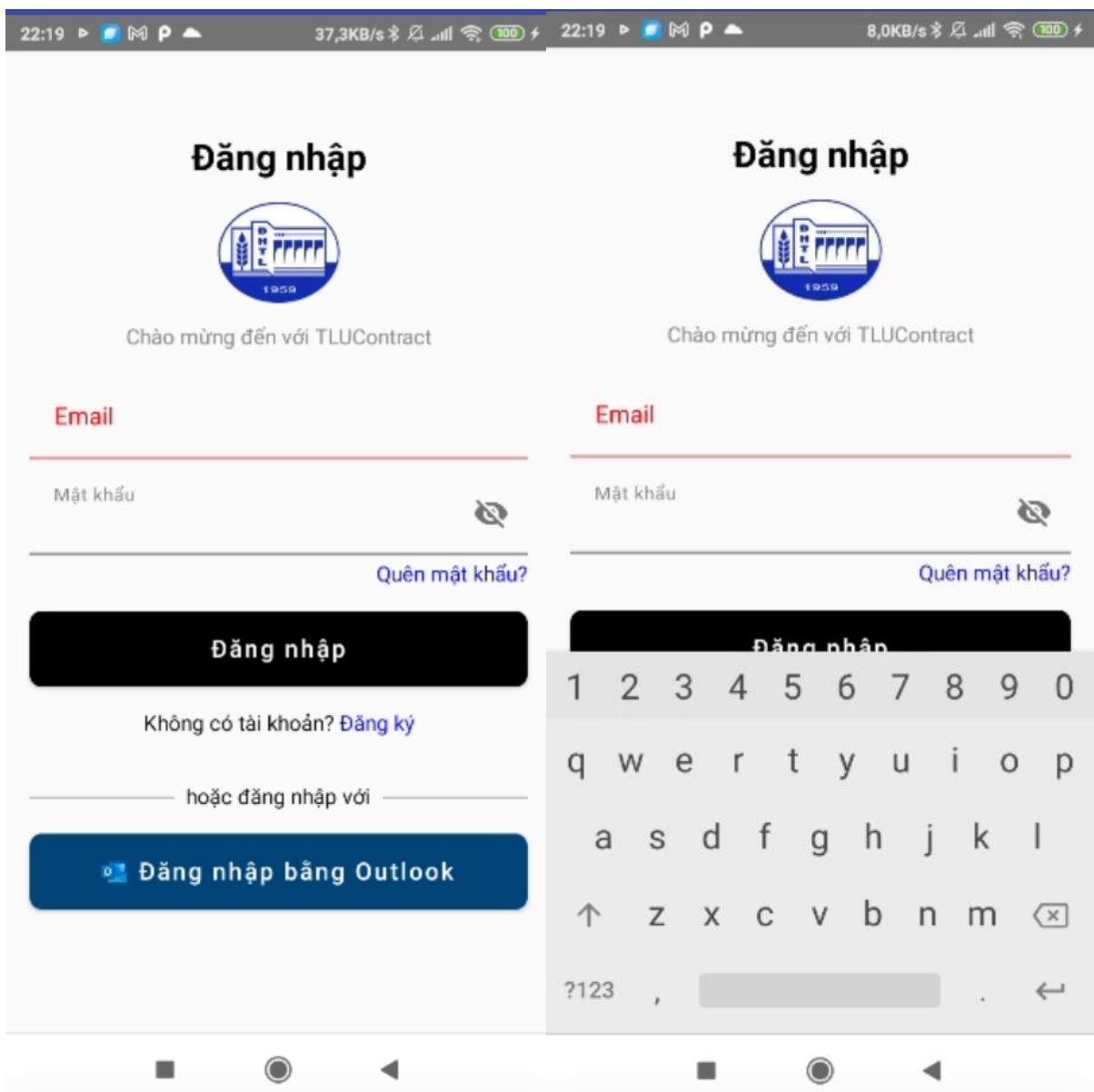
3.3. Giao diện ứng dụng

3.3.1. Logo ứng dụng



3.3.2. Màn hình đăng nhập





22:23 ▶ 129KB/s ⚡ 4G 100% ↗

Đăng nhập



Chào mừng đến với TLUContract

Email _____

Mật khẩu _____

[Quên mật khẩu?](#)

Đăng nhập

[Không có tài khoản? Đăng ký](#)

hoặc đăng nhập với _____

22:21 ▶ 3,5KB/s ⚡ 4G 100% ↗

Đăng nhập



Chào mừng đến với TLUContract

Email _____

Mật khẩu _____

[Quên mật khẩu?](#)

Đăng nhập

[Không có tài khoản? Đăng ký](#)

hoặc đăng nhập với _____

22:22 ▶ 3,0KB/s ⚡ 4G 100% ↗

Đăng nhập



Chào mừng đến với TLUContract

Mật khẩu _____

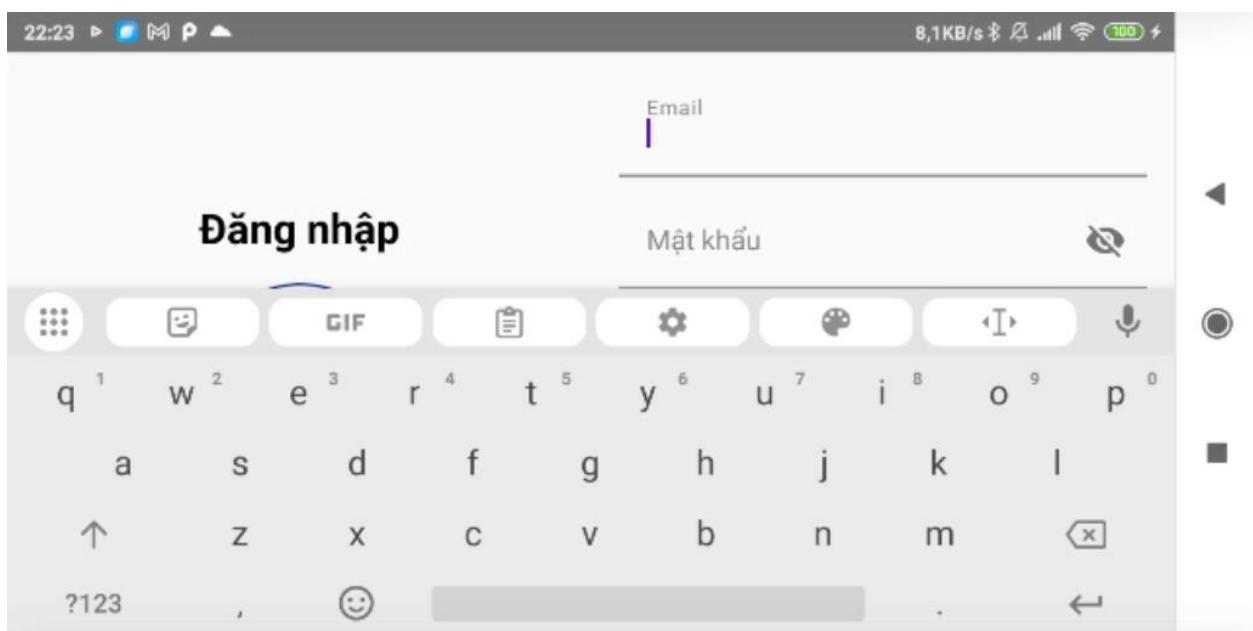
[Quên mật khẩu?](#)

Đăng nhập

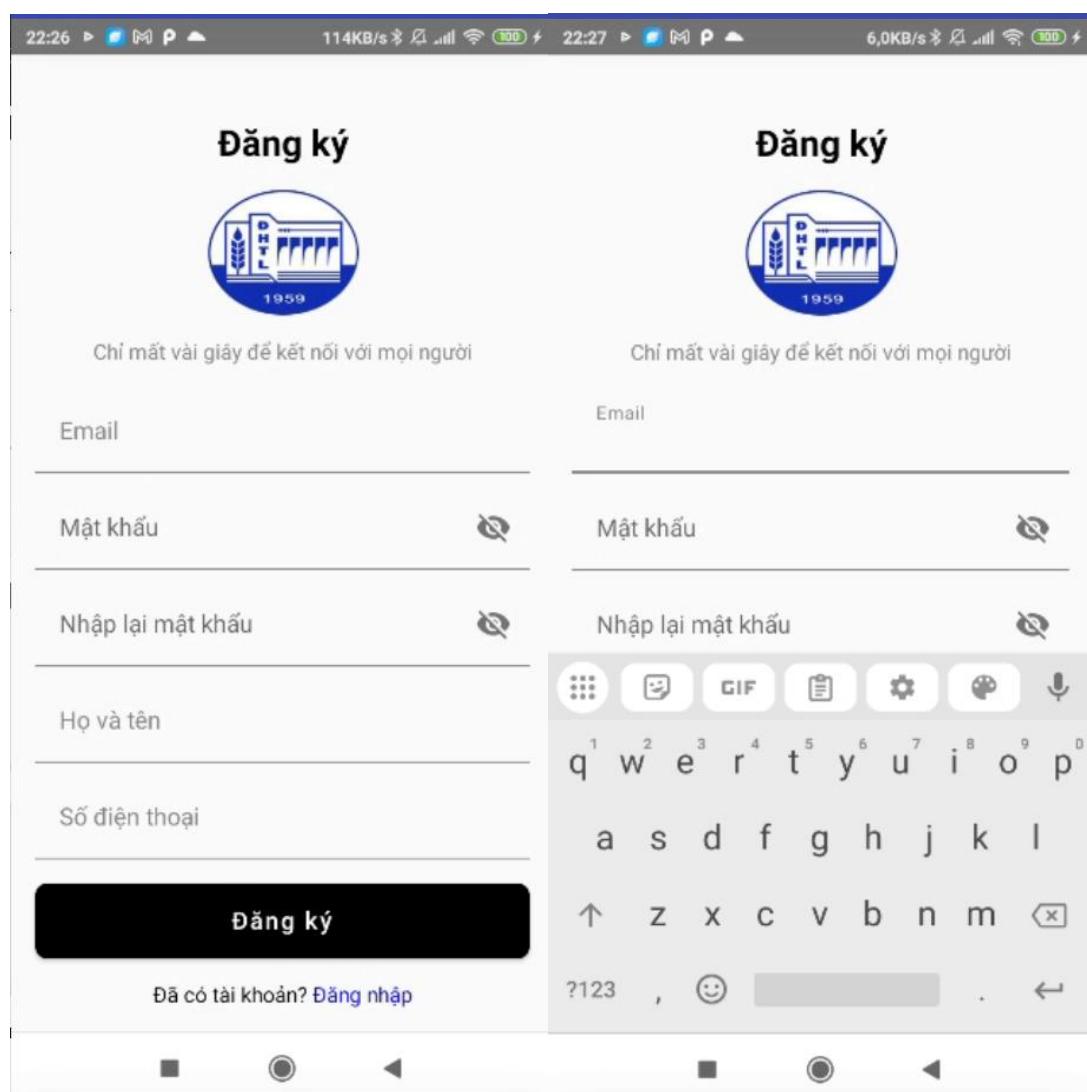
[Không có tài khoản? Đăng ký](#)

hoặc đăng nhập với _____

Đăng nhập bằng Outlook



3.3.3. Đăng ký



<p>Đăng ký</p>  <p>Chỉ mất vài giây để kết nối với mọi người</p> <p>Email toannt204@gmail.com</p> <hr/> <p>Mật khẩu ***** <input type="password"/></p> <hr/> <p>Nhập lại mật khẩu ***** <input type="password"/></p> <hr/> <p>Họ và tên Nguyen The Toan</p> <hr/> <p>Số điện thoại 0111111111</p> <hr/> <p style="background-color: black; color: white; padding: 5px;">Đăng ký</p> <p>Đã có tài khoản? Đăng nhập</p>	<p>Đăng ký</p>  <p>Chỉ mất vài giây để kết nối với mọi người</p> <p>Email toannt204@gmail.com</p> <hr/> <p>Mật khẩu 123456 <input type="password"/></p> <hr/> <p>Nhập lại mật khẩu 123456 <input type="password"/></p> <hr/> <p>Họ và tên Nguyen The Toan</p> <hr/> <p>Số điện thoại 0111111111</p> <hr/> <p style="background-color: black; color: white; padding: 5px;">Đăng ký</p> <p>Đã có tài khoản? Đăng nhập</p>
--	--

<p>Đăng ký</p>  <p>Chỉ mất vài giây để kết nối với mọi người</p> <p>Email 2251061894@e.tlu.edu.vn</p> <hr/> <p>Mật khẩu ***** <input type="password"/></p> <hr/> <p>Nhập lại mật khẩu ***** <input type="password"/></p> <hr/> <p style="background-color: black; color: white; padding: 5px;">Đăng ký</p> <p>Đã có tài khoản? Đăng nhập</p>	<p>Đăng ký</p>  <p>Chỉ mất vài giây để kết nối với mọi người</p> <p>Email 2251061894@e.tlu.edu.vn</p> <hr/> <p>Mật khẩu ***** <input type="password"/></p> <hr/> <p>Nhập lại mật khẩu ***** <input type="password"/></p> <hr/> <p style="background-color: black; color: white; padding: 5px;">Đăng ký</p> <p>Đã có tài khoản? Đăng nhập</p>
---	---

Đăng ký



Chỉ mất vài giây để kết nối với mọi người

Email

Mật khẩu

Nhập lại mật khẩu

Họ và tên

Số điện thoại

Đăng ký (Dark mode button)

Đã có tài khoản? [Đăng nhập](#)

Đăng ký



Chỉ mất vài giây để kết nối với mọi người

Email

Mật khẩu

Nhập lại mật khẩu

grid camera GIF list gear paint mic

q w e r t y u i o p
a s d f g h j k l
z x c v b n m X

?123
,
😊
.
⬅

22:34 1,2KB/s ⚡ 4G ⚡ 100% ↗

Đăng ký



Chỉ mất vài giây để kết nối với mọi người

Email

Mật khẩu

Nhập lại mật khẩu

Họ và tên

Số điện thoại

◀ ● ■

22:35 19,0KB/s

Nhập lại mật khẩu

Họ và tên
Nguyen The Toan

Số điện thoại
0111111111

Đăng ký

Chỉ mất vài giây để kết nối với mọi người

Đã có tài khoản? [Đăng nhập](#)



22:35 3,0KB/s

Nhập lại mật khẩu

Họ và tên
Nguyen The Toan

Toán

q 1 w 2 e 3 r 4 t 5 y 6 u 7 i 8 o 9 p 0

a s d f g h j k l

↑ z x c v b n m ←

?123 , ☺ . ←



22:37 4,2KB/s

Email

Mật khẩu

Nhập lại mật khẩu

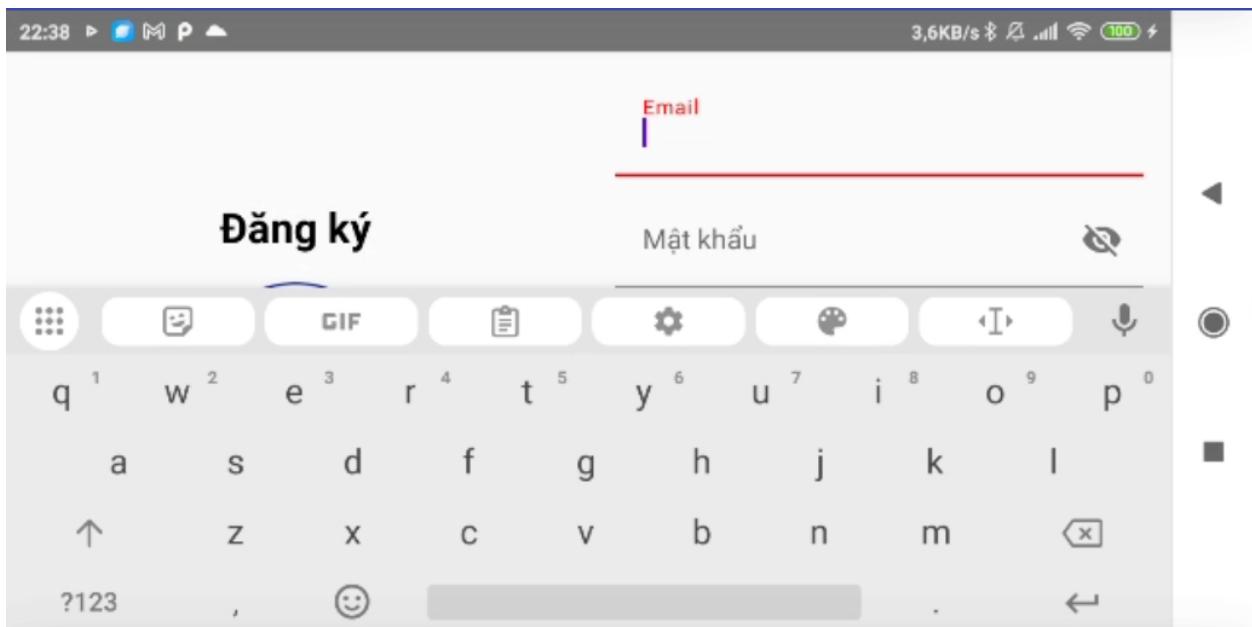
Họ và tên

Số điện thoại

Đăng ký

Chỉ mất vài giây để kết nối với mọi người





3.3.4. Email xác nhận tài khoản

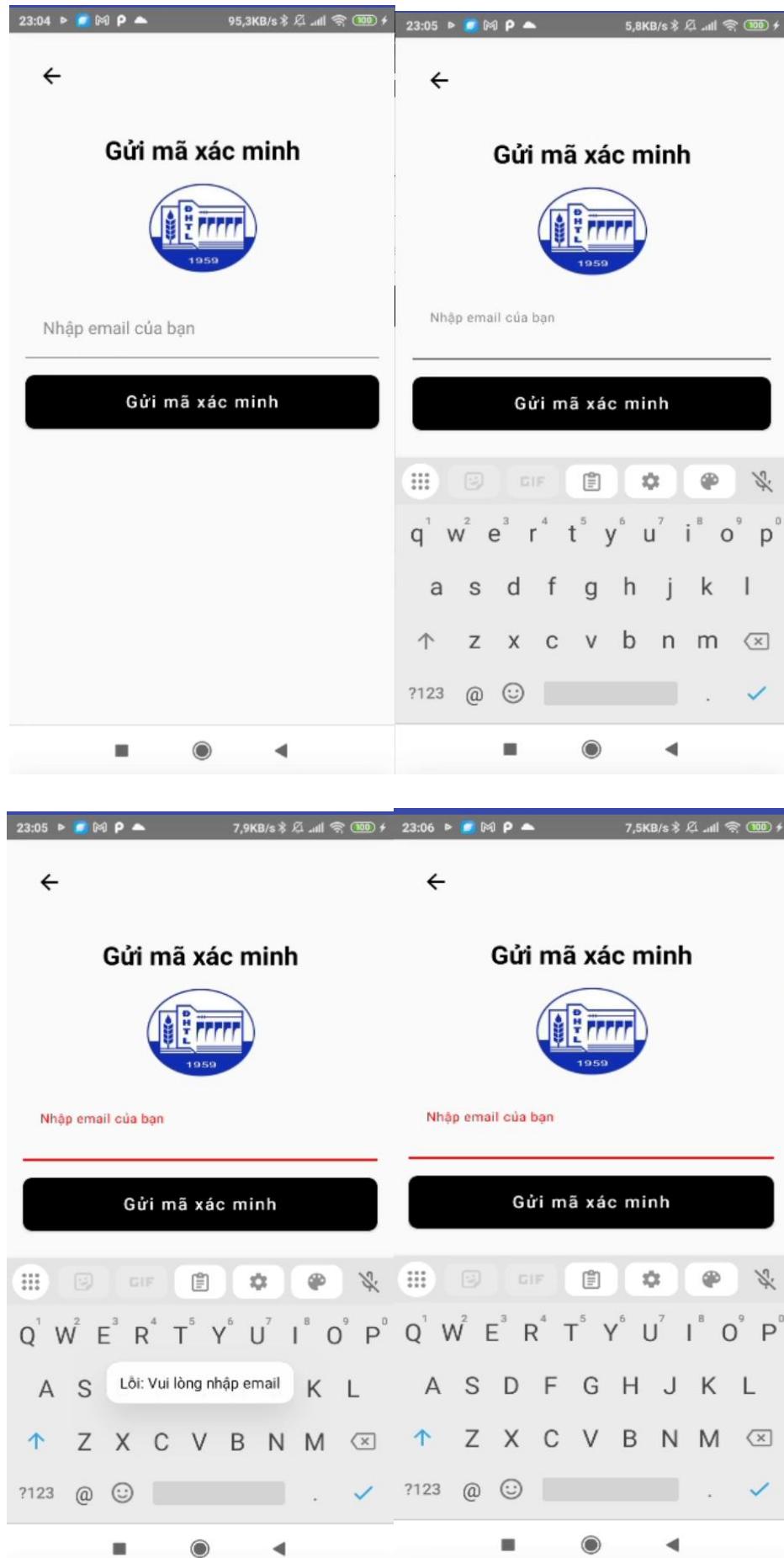
The screenshot shows an email inbox and a specific email message. The inbox at the top has a blue header with a back arrow, trash, reply, and more options. It lists an incoming email from 'TLUContract' with the subject 'Đến Bạn' and a timestamp of 18:00. The message body starts with 'Xin chào!' followed by instructions to click a link to verify the email address. The link is:

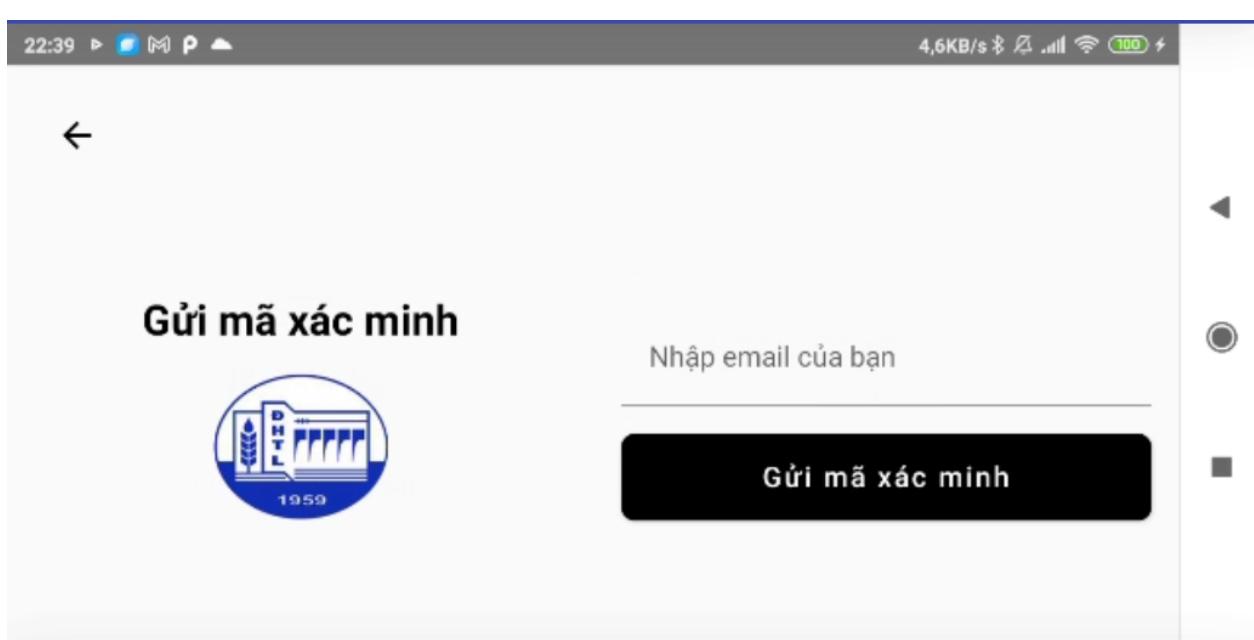
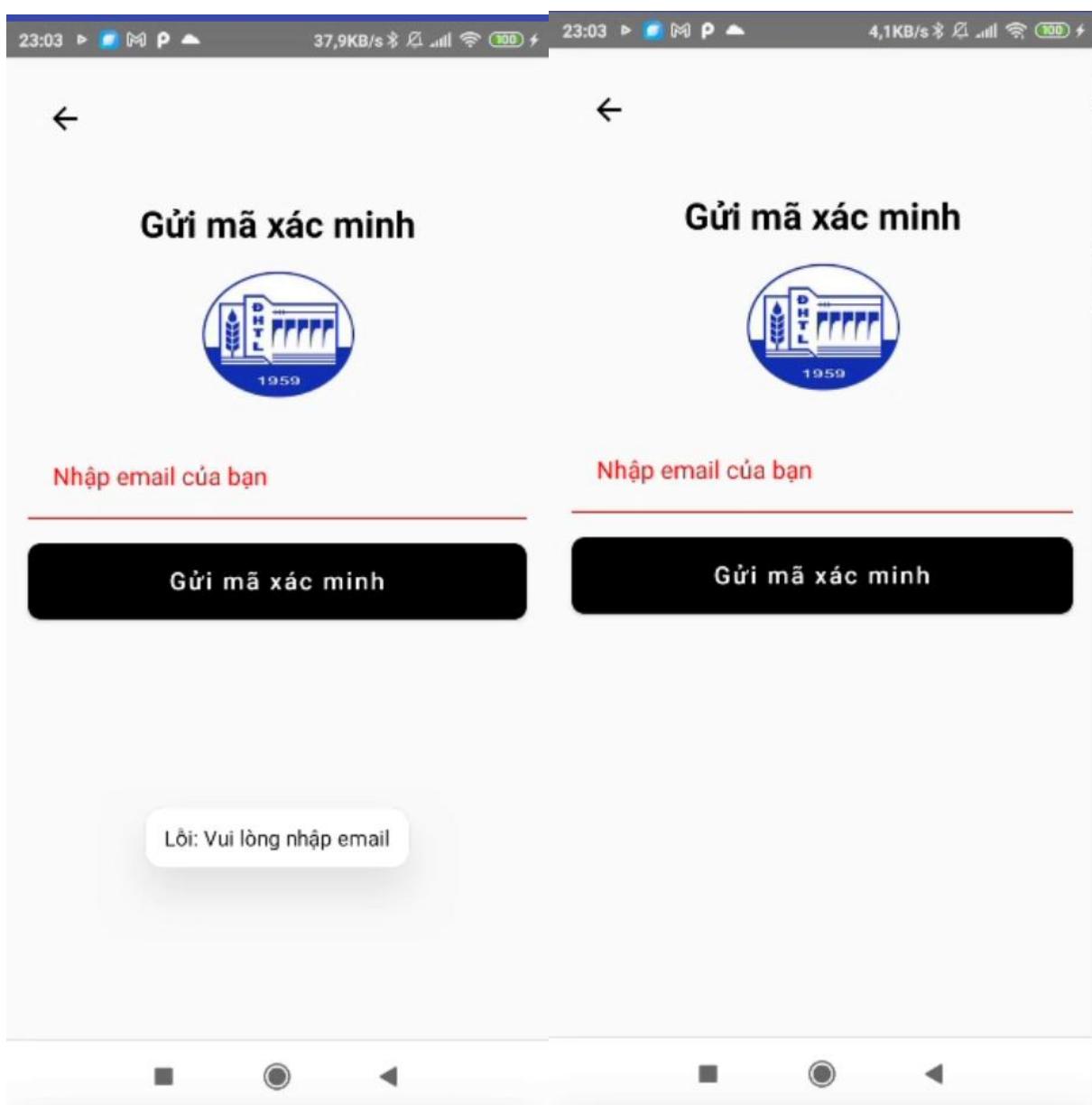
https://tlucontract.firebaseio.com/_/auth/action?mode=verifyEmail&oobCode=iYy_d1yCt6ZKACEc5AgyKf3m11MCDybNw-1xNVxbbSMAAGWJG21A&apiKey=AIzaSyDpvZoCFTHq1ysemKbnf0rbVj3WEMJCqE&lang=vi

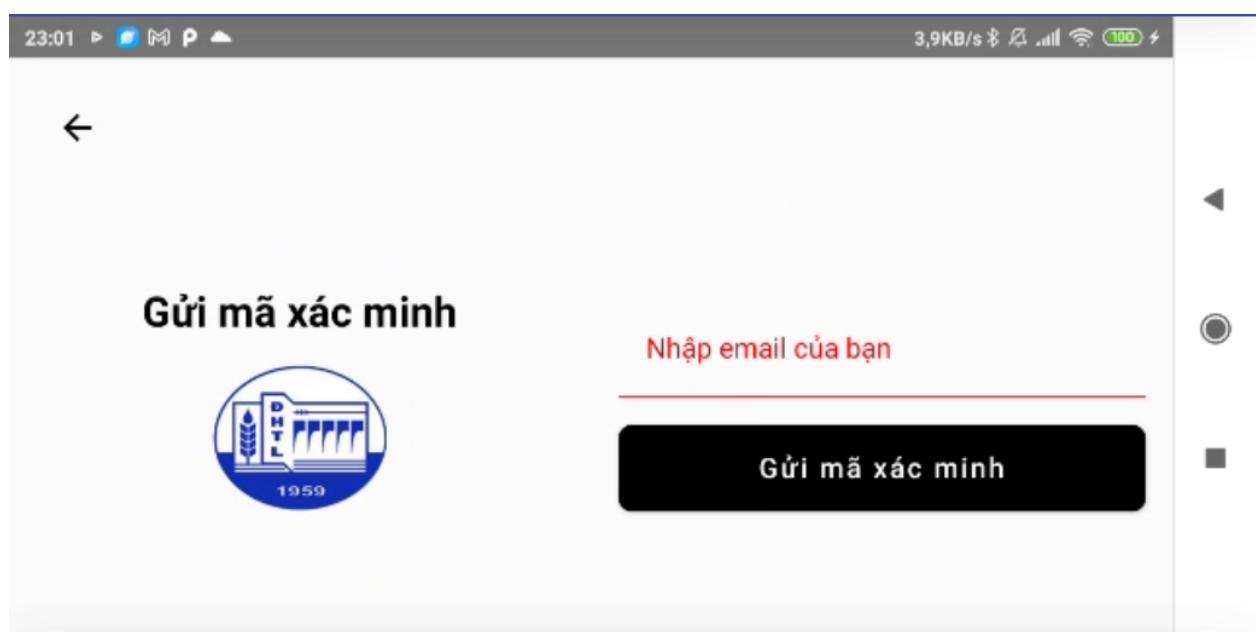
If you do not want to verify the email address, you can delete the email.

The message ends with 'Cảm ơn bạn!' (Thank you!).

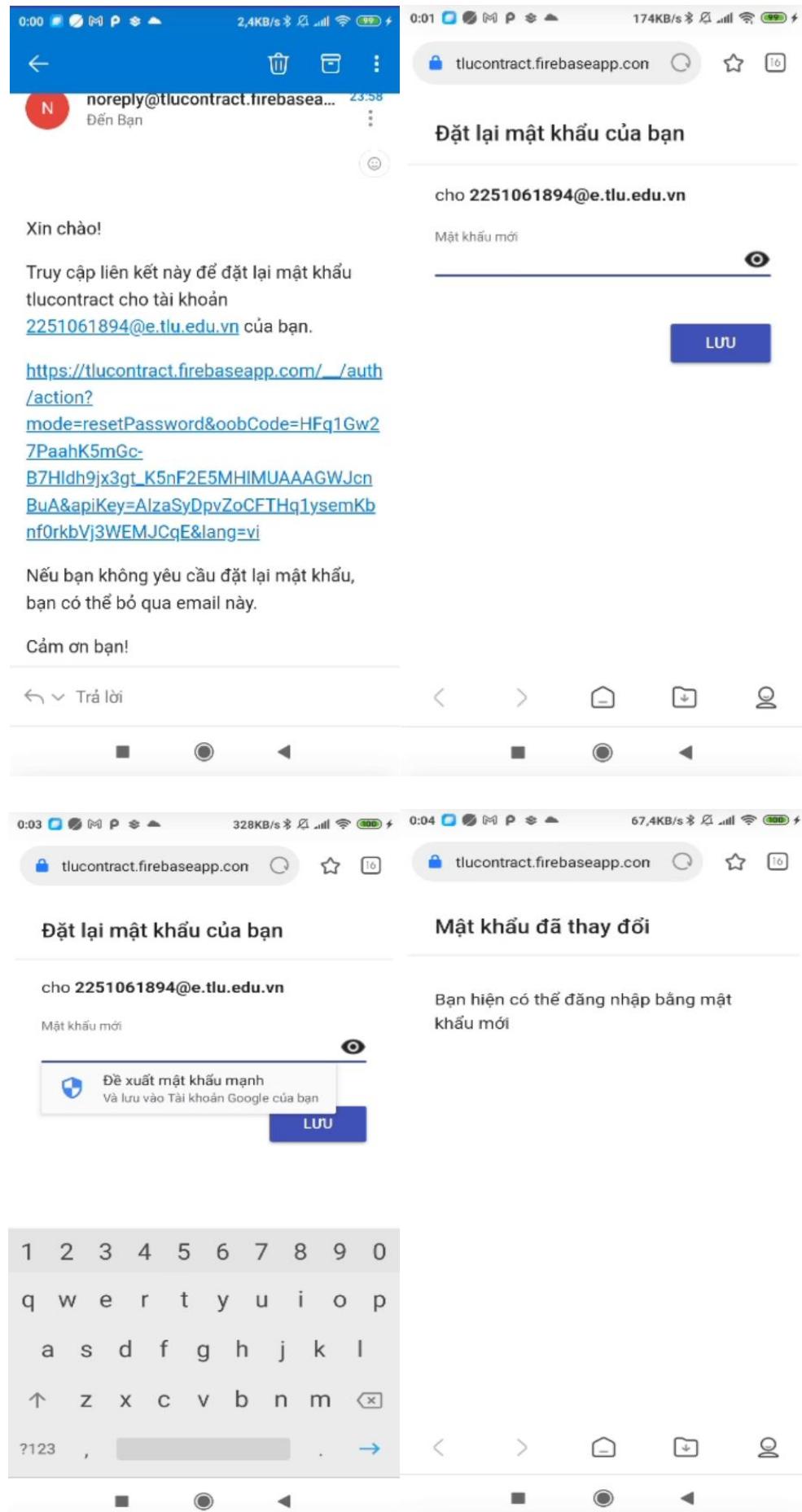
3.3.5. Khôi phục mật khẩu



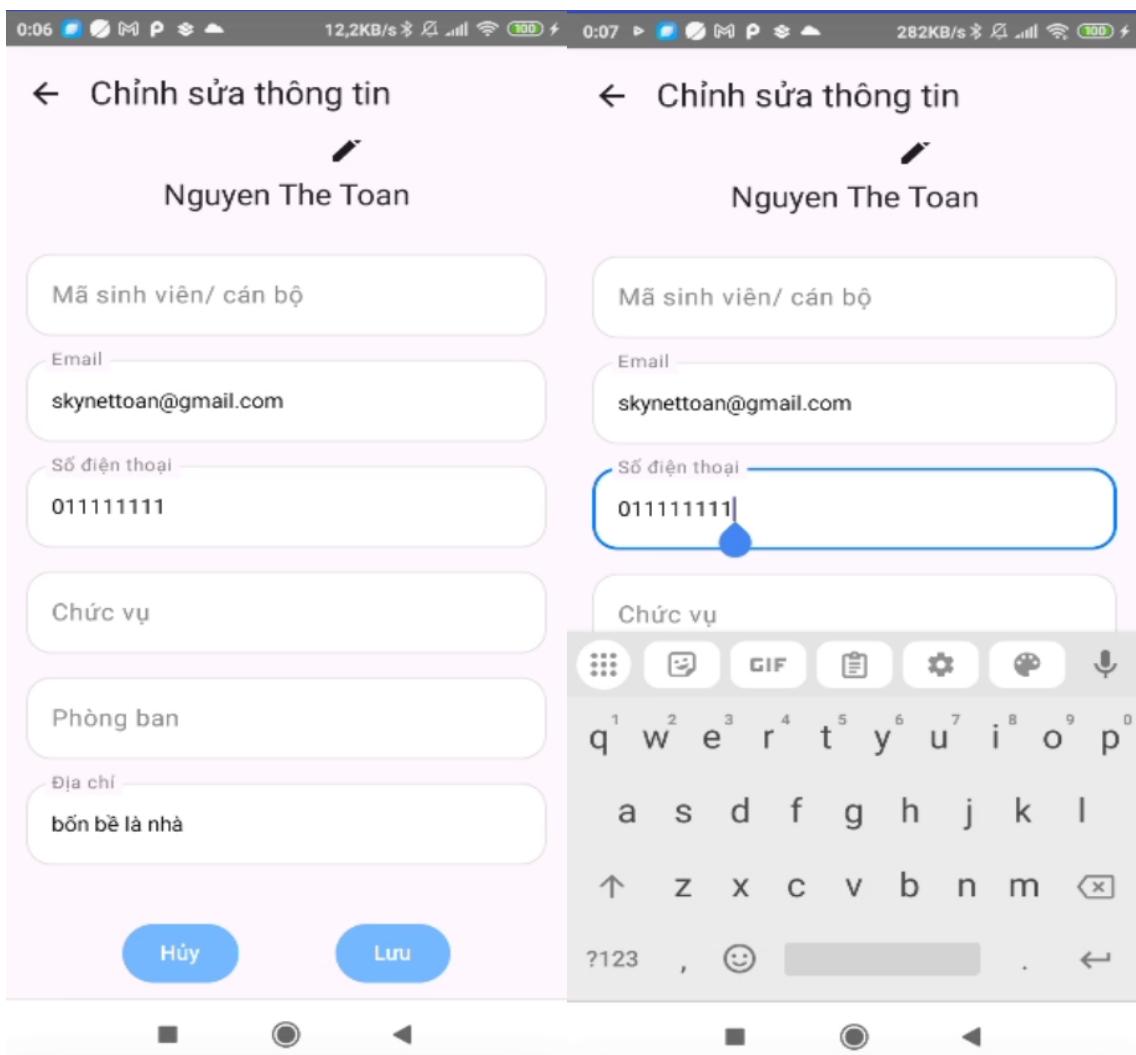




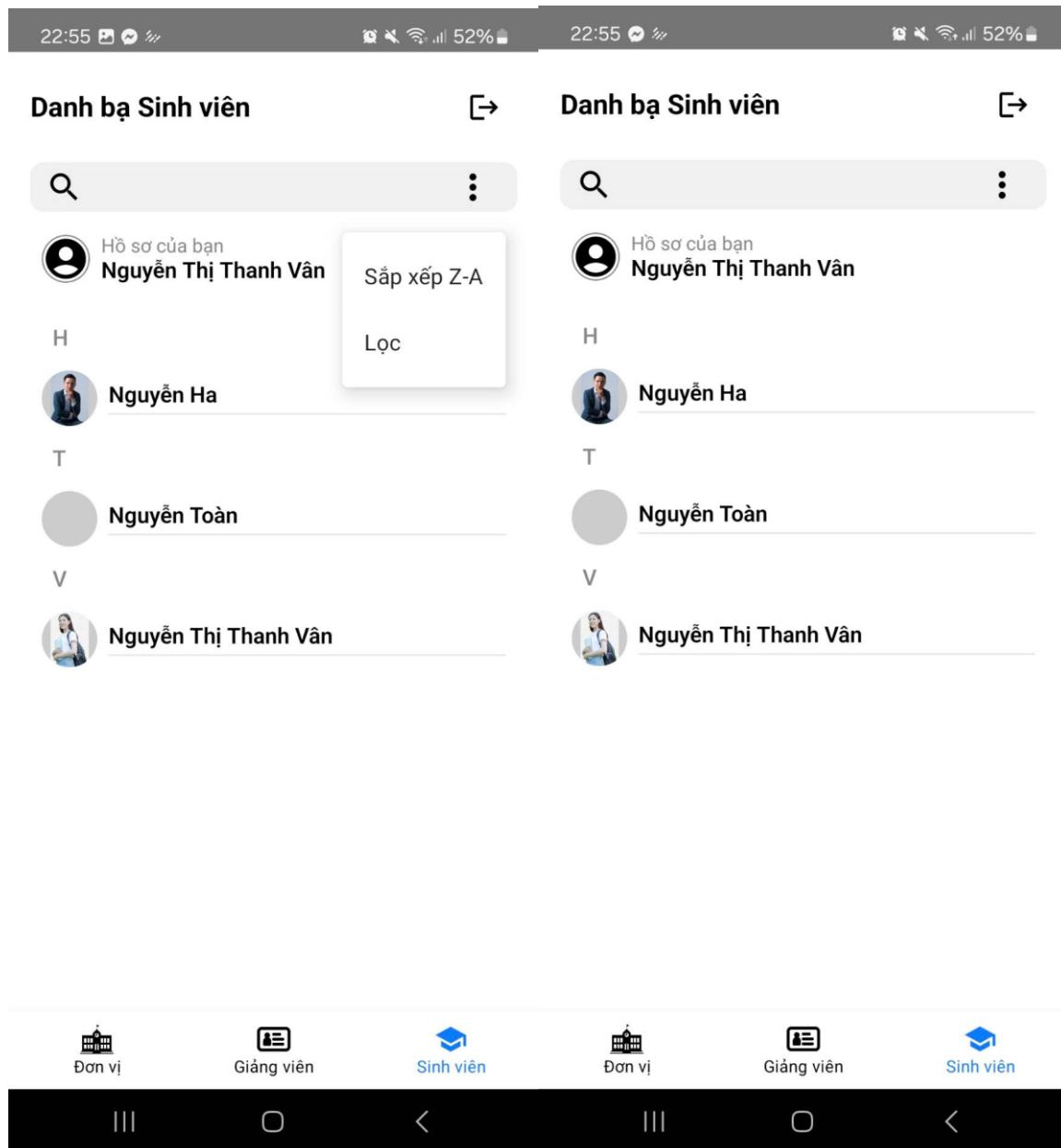
3.3.6. Email khôi phục mật khẩu



3.3.7. Giao diện chỉnh sửa thông tin tài khoản khách



3.3.8. Giao diện xem danh bạ sinh viên





Danh bạ Sinh viên

Hồ sơ của bạn
Nguyễn Thị Thanh Vân

V

 **Nguyễn Thị Thanh Vân**

T

 **Nguyễn Toàn**

H

 **Nguyễn Ha**

Danh bạ Sinh viên

Hồ sơ của bạn
Nguyễn Thị Thanh Vân

Sắp xếp A-Z

Lọc

V

 **Nguyễn Thị Thanh Vân**

T

 **Nguyễn Toàn**

H

 **Nguyễn Ha**



Danh bạ Sinh viên

Danh bạ Sinh viên

Hồ sơ của bạn
Nguyễn Thị Thanh Vân

Sắp xếp Z-A

H

 **Nguyễn Ha**

Lọc

T

 **Nguyễn Toàn**

V

 **Nguyễn Thị Thanh Vân**

Theo Lớp

Theo Tên

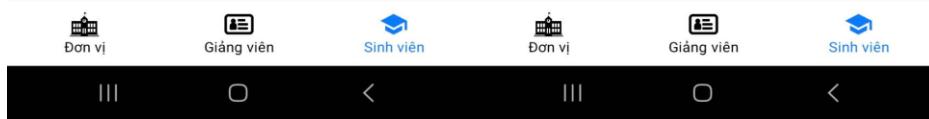
Hồ sơ của bạn
Nguyễn Thị Thanh Vân

64CNTT1

 **Nguyễn Ha**

 **Nguyễn Toàn**

 **Nguyễn Thị Thanh Vân**



02:38 94% 02:38 94%

Danh bạ Sinh viên

← Thông tin sinh viên

Hồ sơ của bạn
Nguyễn Thị Thanh Vân

64CNTT1

Lê Thị Bình
Nguyễn Hà
Trần Thị Lụa
Nguyễn Toàn
Nguyễn Thị Thanh Vân

Sắp xếp Z-A
Lọc
Theo Lớp
Theo Tên

Nguyễn Toàn

Tin nhắn
Gọi
Gọi video
Mail

Mã sinh viên: 2251061894
Lớp: 64CNTT1
Địa chỉ: 1111111
Số điện thoại: 2121212
Email: 2251061894@e.tlu.edu.vn

Đơn vị
Giảng viên
Sinh viên

22:57 52% 22:57 52%

← Chính sửa thông tin

← Chính sửa thông tin

Nguyễn Thị Thanh Vân

Mã sinh viên: 2251061922
Lớp: 64CNTT1
Số điện thoại: 0986765767
Email: 2251061922@e.tlu.edu.vn
Địa chỉ nơi ở: Bắc Ninh quê tôi

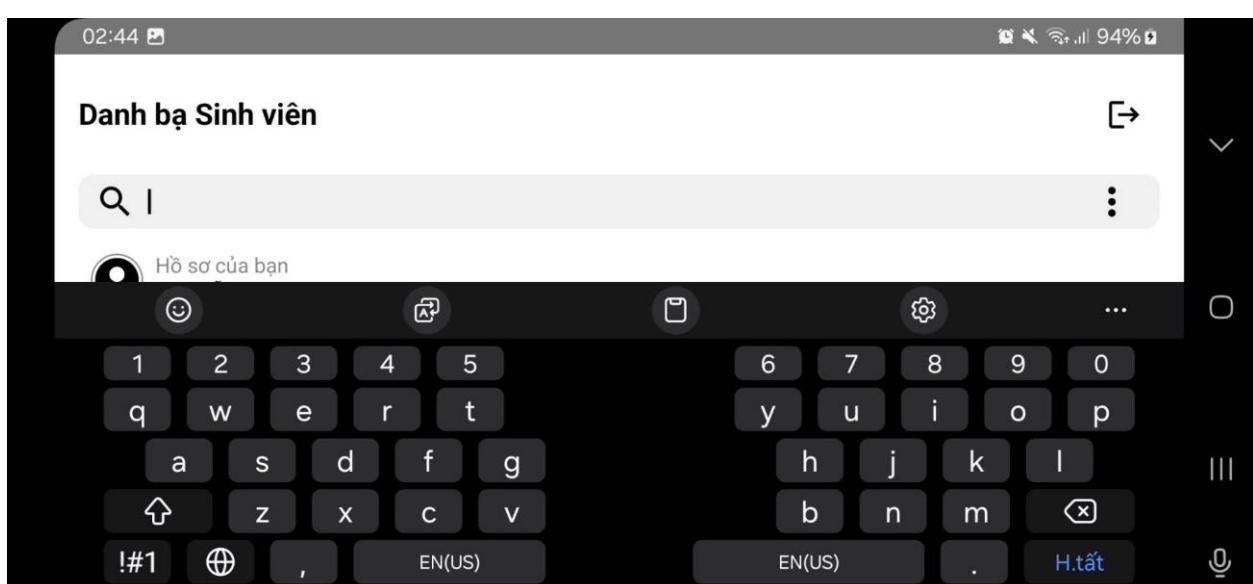
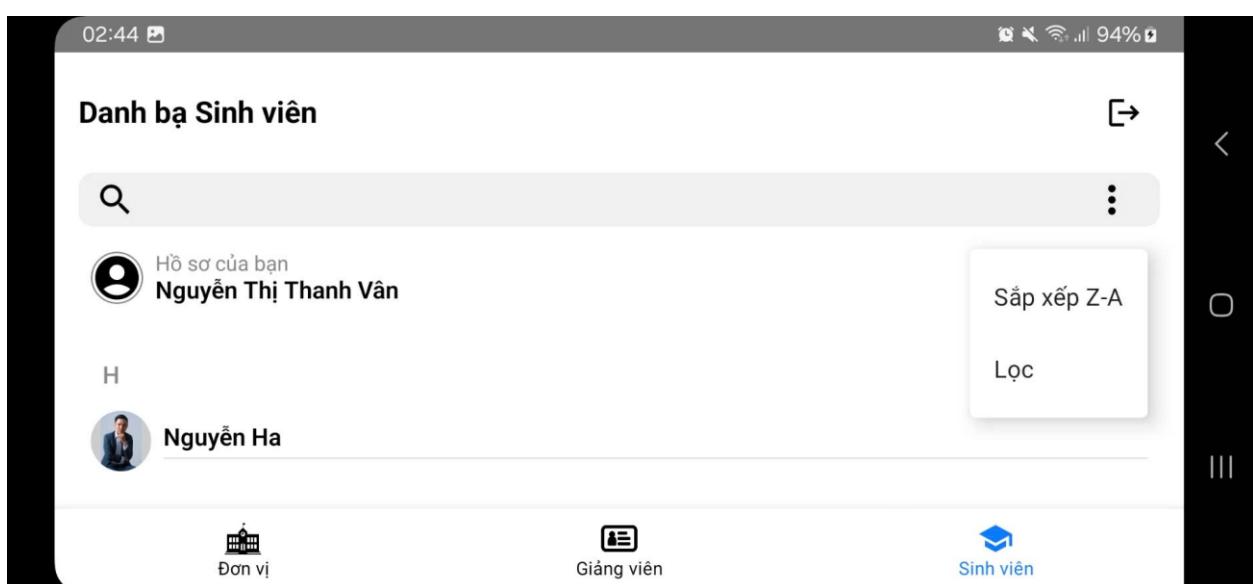
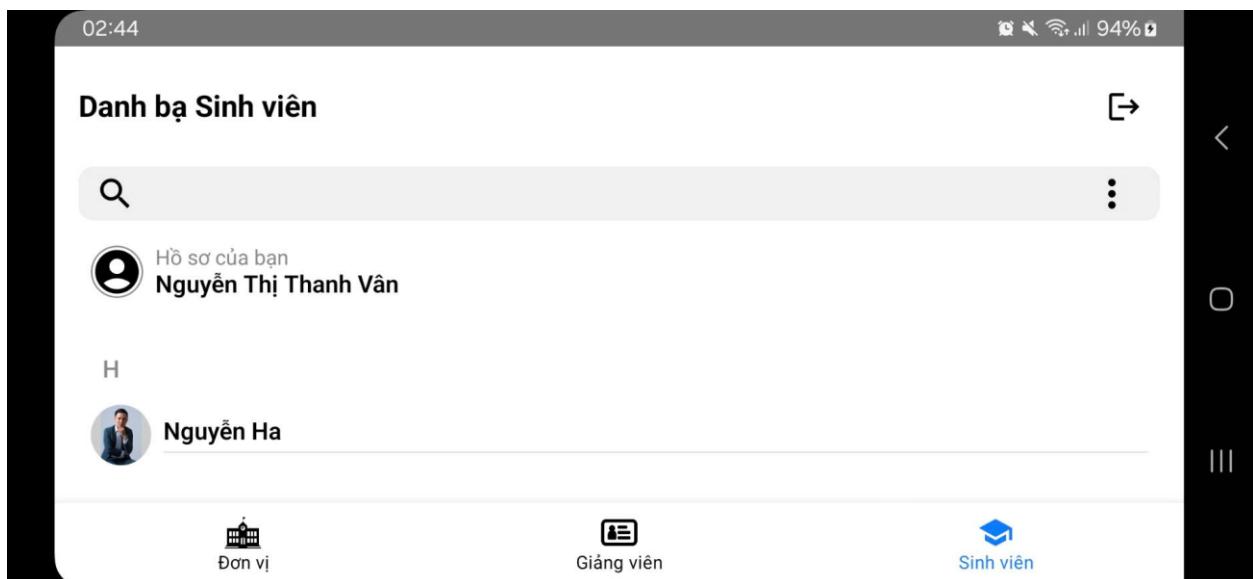
Hủy Lưu

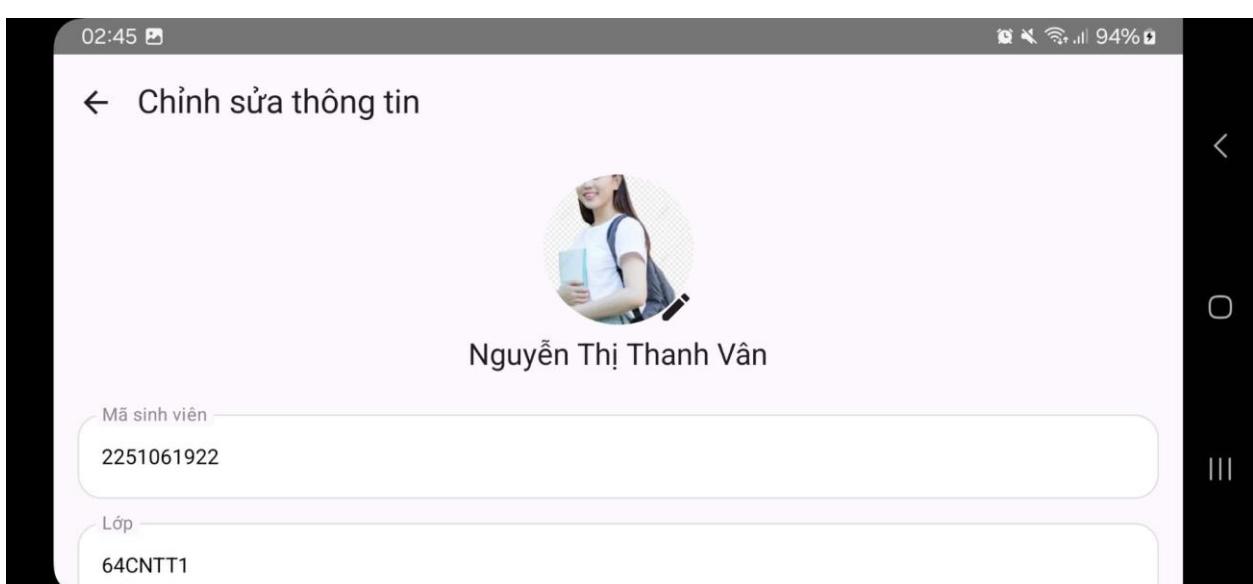
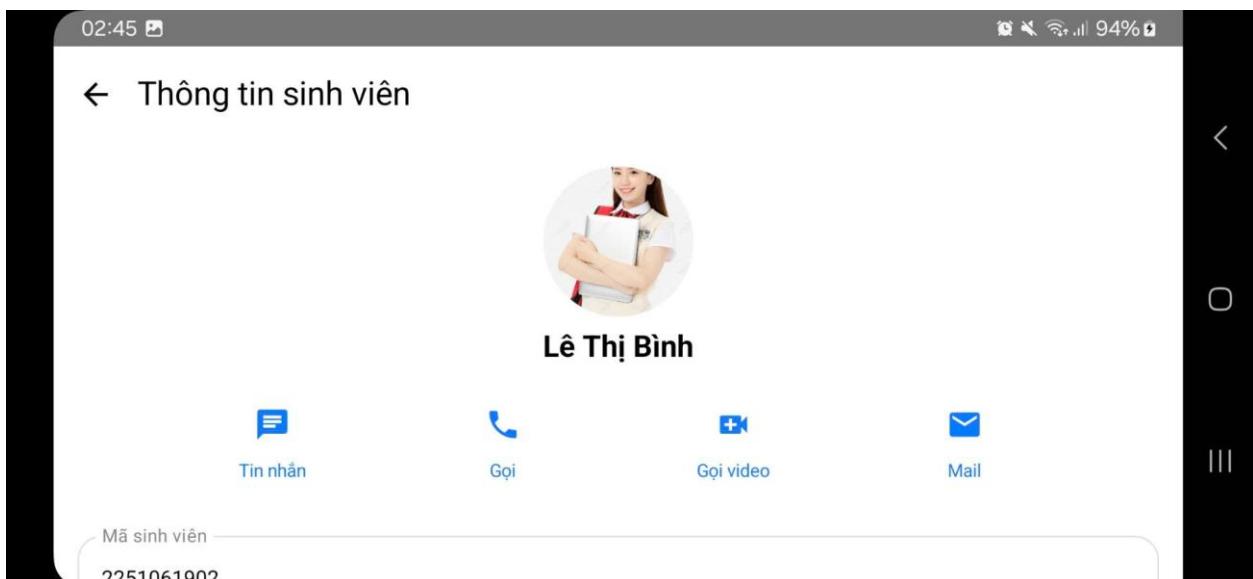
Nguyễn Thị Thanh Vân

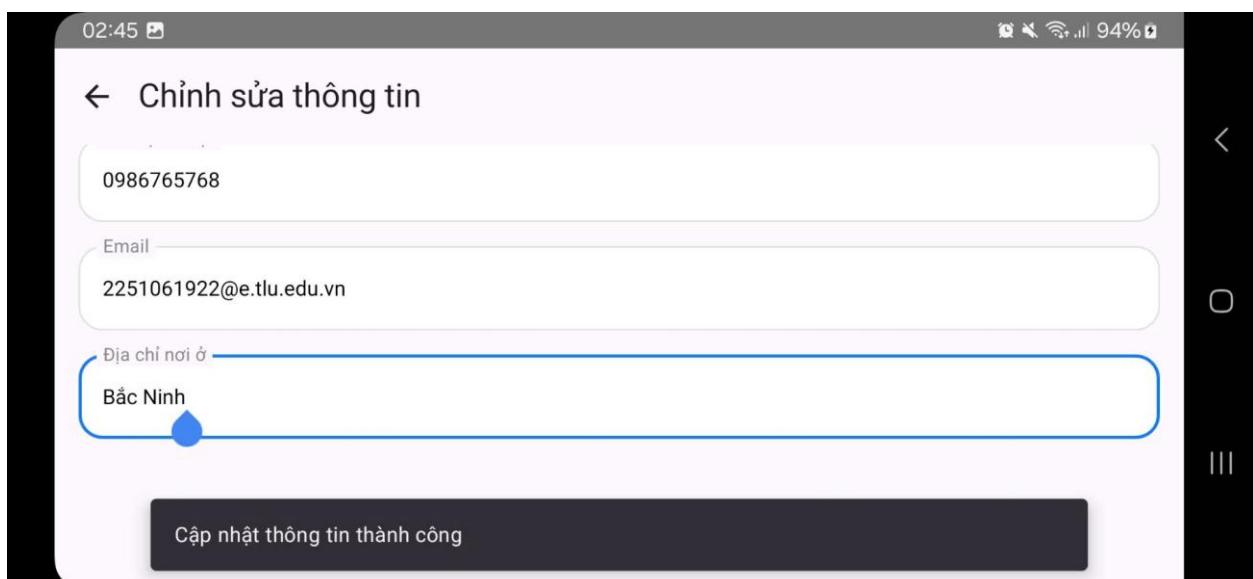
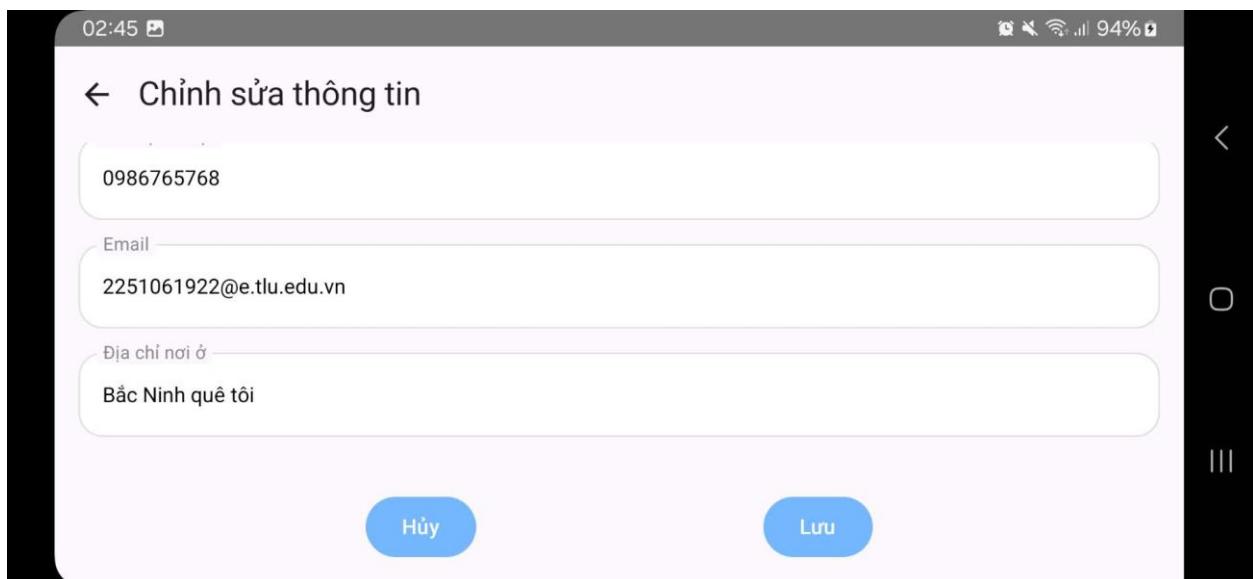
Mã sinh viên: 2251061922
Lớp: 64CNTT1
Số điện thoại: 0986765768
Email: 2251061922@e.tlu.edu.vn
Địa chỉ nơi ở: Bắc Ninh quê tôi

Hủy Lưu

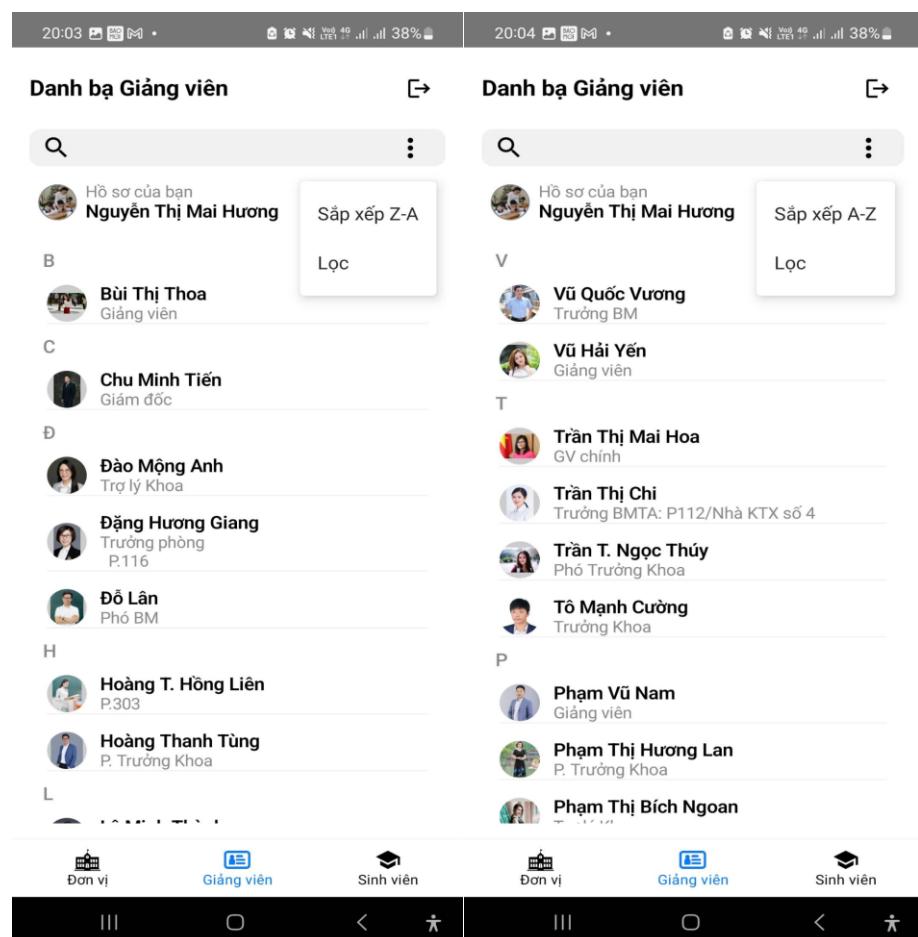
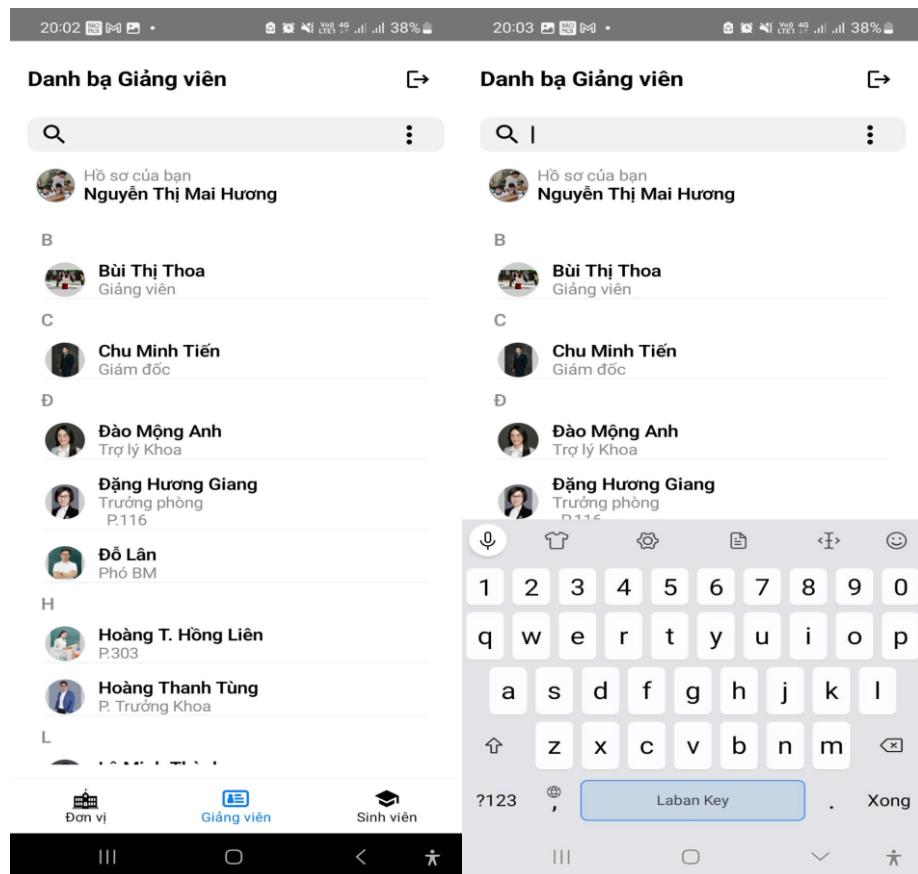
Cập nhật thông tin thành công







3.3.9. Giao diện xem danh bạ giảng viên



Danh bạ Giảng viên

Chỉnh sửa thông tin

Nguyễn Thị Mai Hương

Họ và tên: Nguyễn Thị Mai Hương
Mã giảng viên: 2343724567
Chức vụ: Giảng viên chính 1
Số điện thoại: 0337725310hhehe
Email: lehaphuongquack@gmail.com
Đơn vị trực thuộc: Khoa Công nghệ thông tin1

Hủy **Lưu**

Danh bạ Giảng viên

Sắp xếp A-Z

Tất cả **Theo Đơn vị** **Theo Chức vụ**

Khoa công trình

- Vũ Quốc V Trưởng BM
- Nguyễn T. Phó trưởng
- Nguyễn Q. Giảng viên

Trung tâm giáo dục Quốc phòng và An ninh

- Vũ Hải Yến Giảng viên
- Lê Thị Diên Giảng viên
- Bùi Thị Thoa Giảng viên

Khoa hóa & Môi trường

- Trần Thị Mai Hoa GV chính
- Lê Thị Thắng Giảng viên
- Lê Minh Thành

Đơn vị **Giảng viên** **Sinh viên**

Danh bạ Giảng viên

Sắp xếp A-Z

Tất cả **Theo Đơn vị** **Theo Chức vụ**

Trưởng BM

- Vũ Quốc V Trưởng BM
- Nguyễn H. Trưởng BM
- Nguyễn Đ. Trưởng BM

Giảng viên

- Vũ Hải Yến Giảng viên
- Phạm Vũ Nam Giảng viên
- Nguyễn Quang Phú Giảng viên
- Lê Thị Thắng Giảng viên
- Lê Thị Diên Giảng viên
- Bùi Thị Thoa Giảng viên

Đơn vị **Giảng viên** **Sinh viên**

Danh bạ Giảng viên

Sắp xếp Z-A

Tất cả **Theo Đơn vị** **Theo Chức vụ**

B

- Bùi Thị Th Giảng viên

C

- Chu Minh Giám đốc

Đ

- Đào Mộng Anh Trợ lý Khoa
- Đặng Hương Giang Trưởng phòng P.116
- Đỗ Lân Phó BM

H

- Hoàng T. Hồng Liên P.303
- Hoàng Thanh Tùng P. Trưởng Khoa

L

Đơn vị **Giảng viên** **Sinh viên**

20:05  VoLTE 4G  38% 

← Chính sửa thông tin



Nguyễn Thị Mai Hương

Họ và tên
Nguyễn Thị Mai Hương

Mã giảng viên
2343724567

Chức vụ
Giảng viên chính 1

Số điện thoại
0337725310 1

Email
lehaphuongquack@gmail.com

Đơn vị trực thuộc
Khoa Công nghệ thông tin1

  Cập nhật thông tin thành công

← Thông tin cán bộ giảng viên



Bùi Thị Thoa

 Tin nhắn

 Gọi

 Gọi video

 Mail

Mã giảng viên
CB0025

Chức vụ
Giảng viên

Số điện thoại
0984.951.159

Email
thoabt@tlu.edu.vn

Đơn vị
Trung tâm giáo dục Quốc phòng và An ninh

Ghi chú
-

20:11  VoLTE 4G  37% 

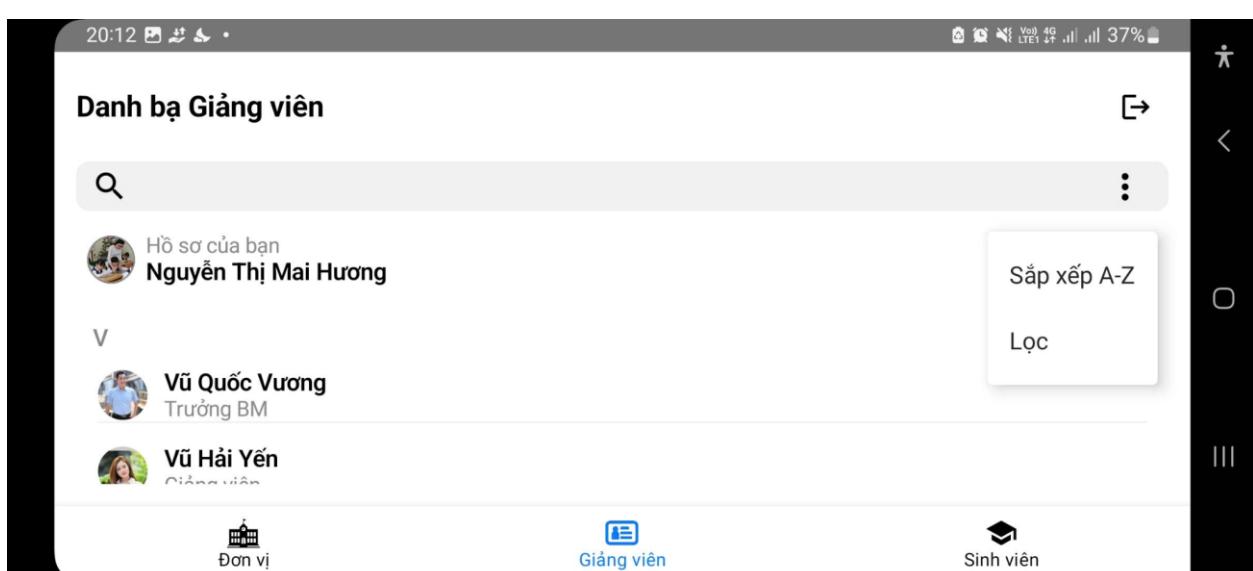
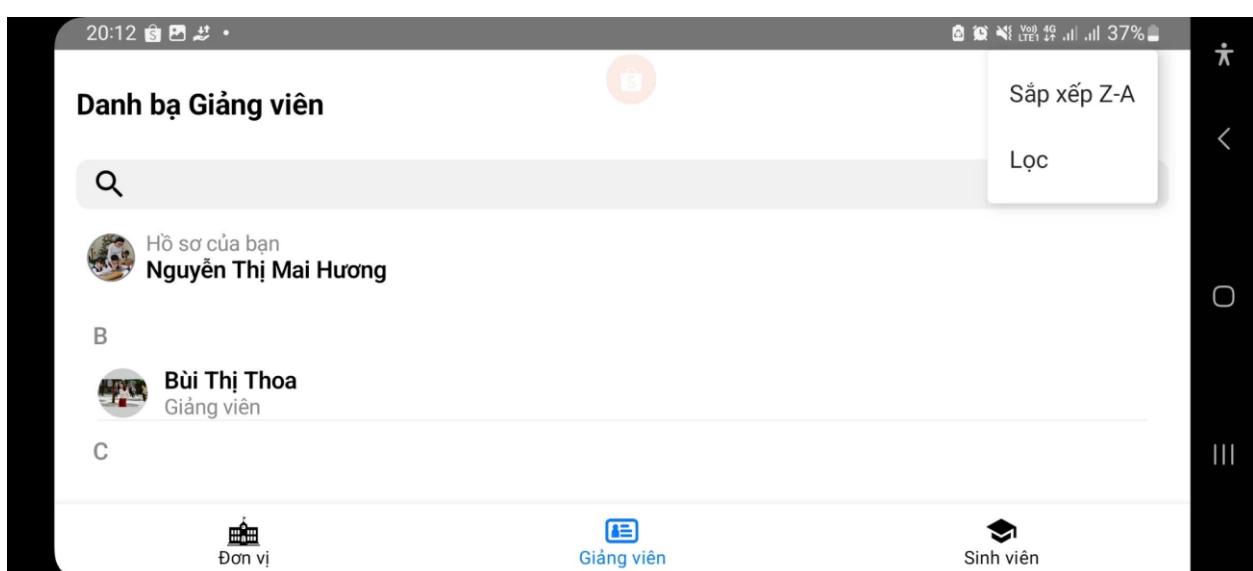
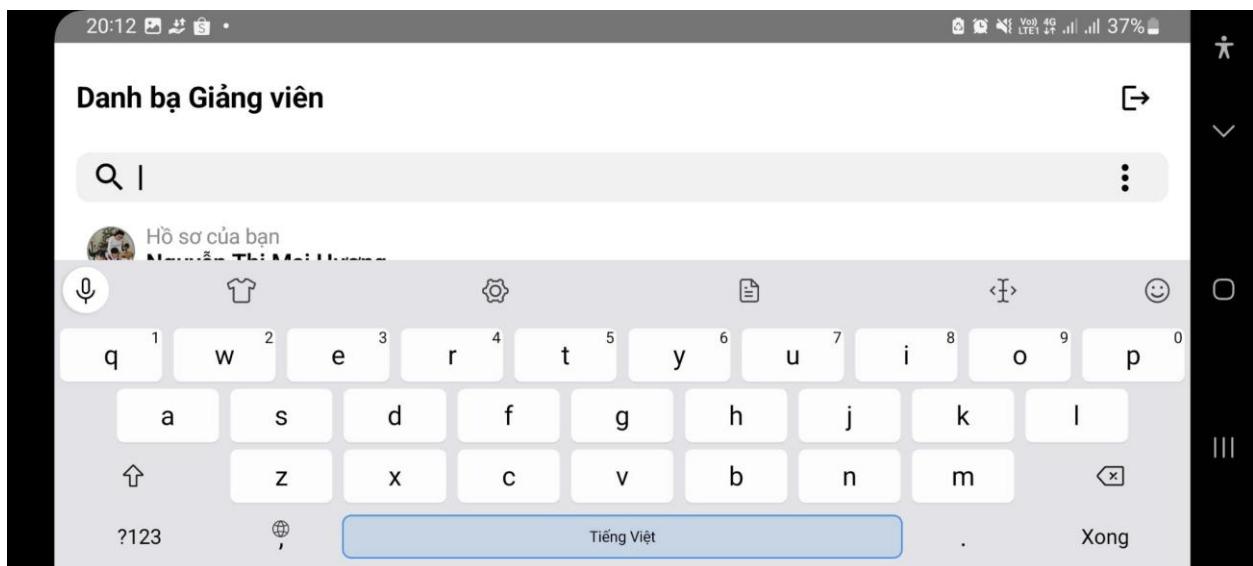
Danh bạ Giảng viên

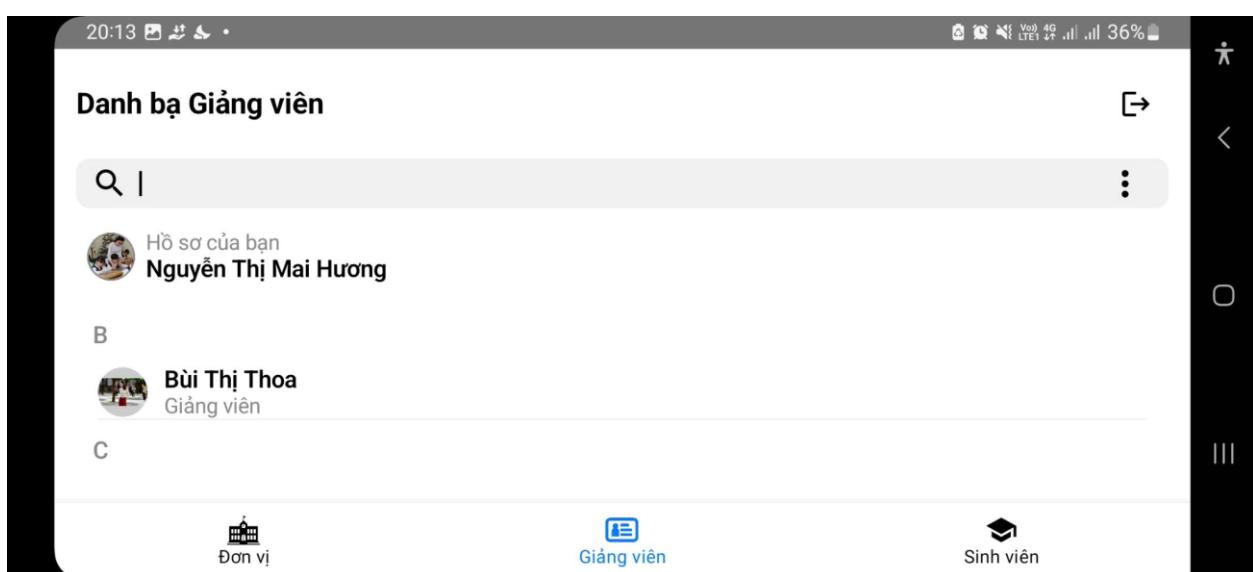
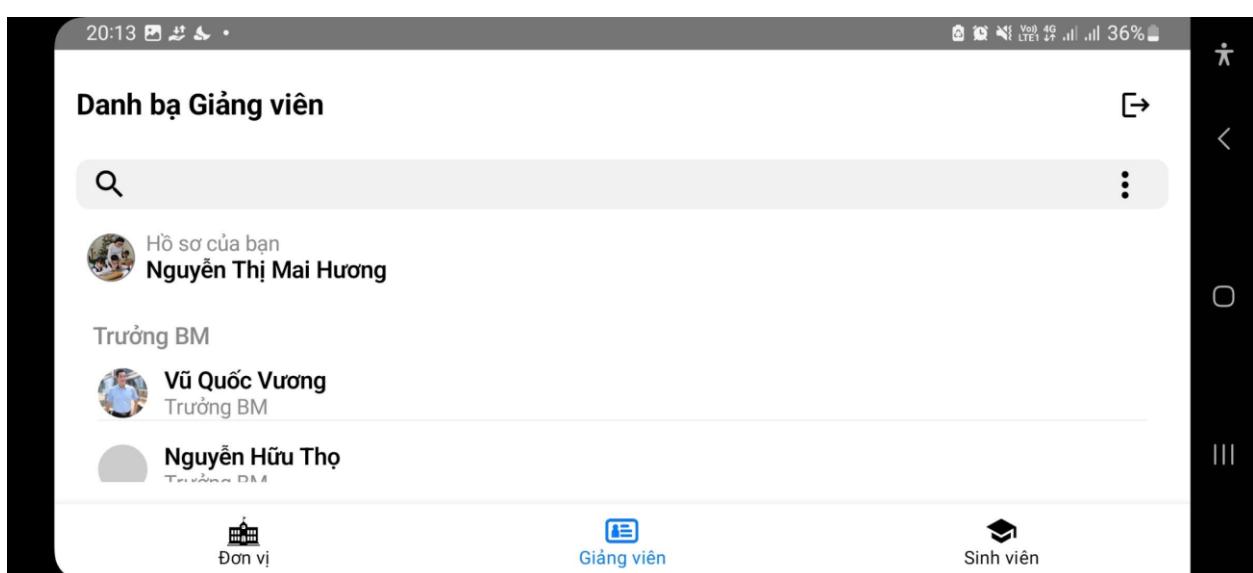
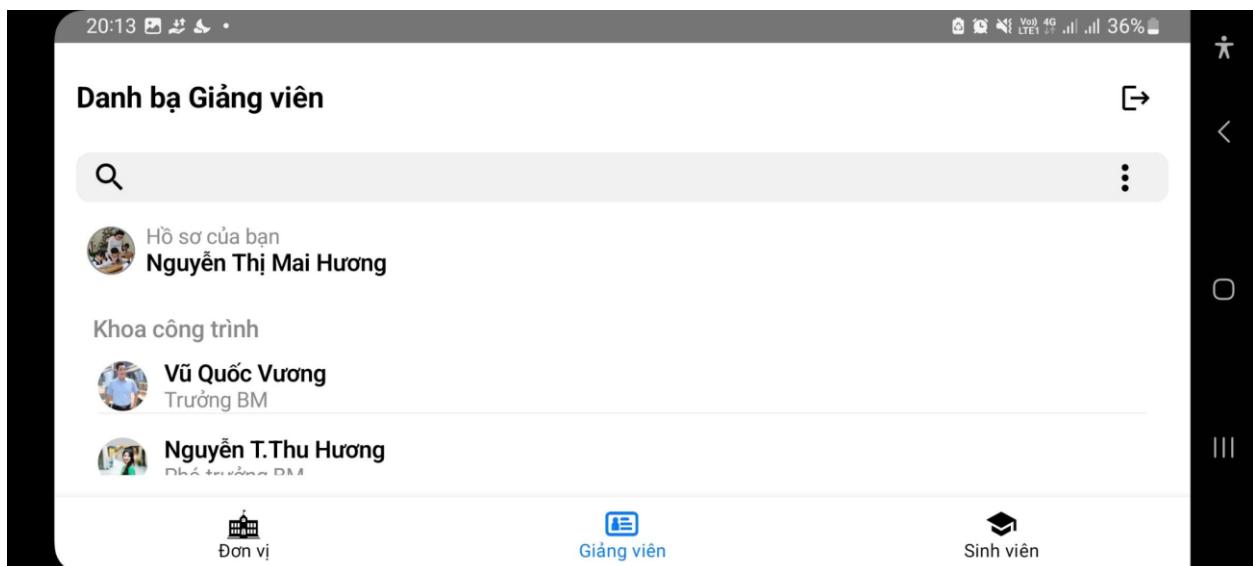
 

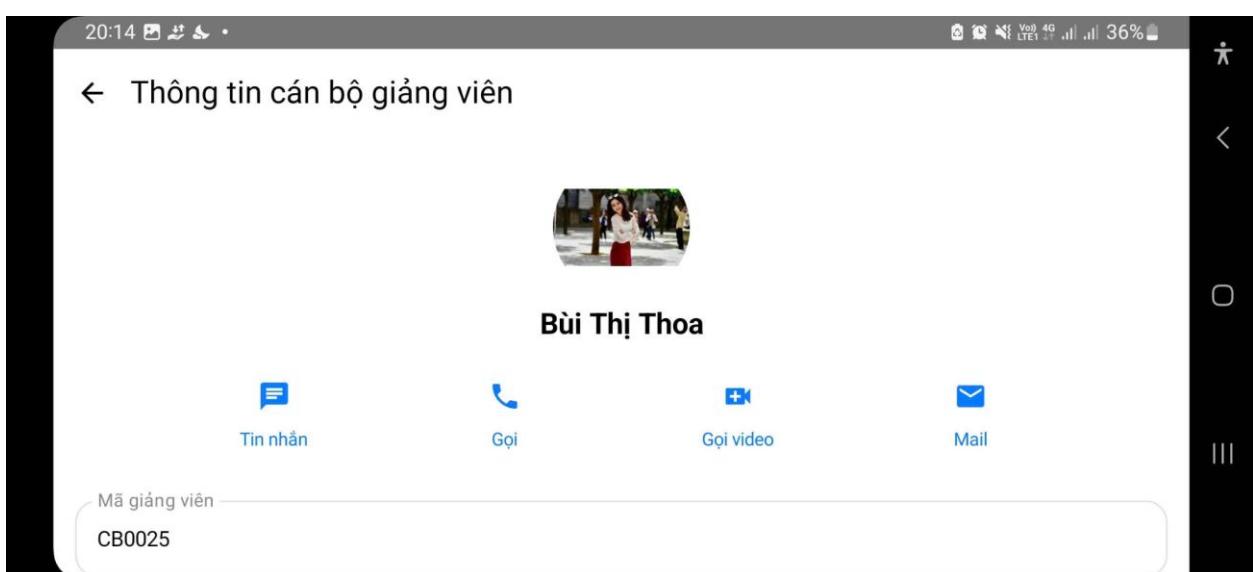
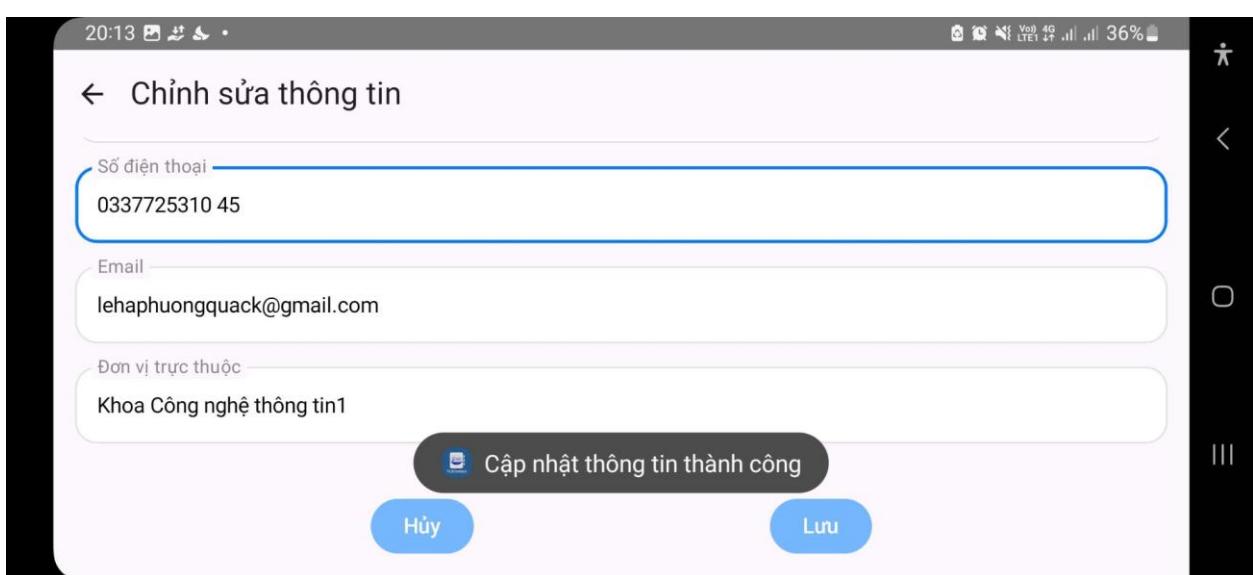
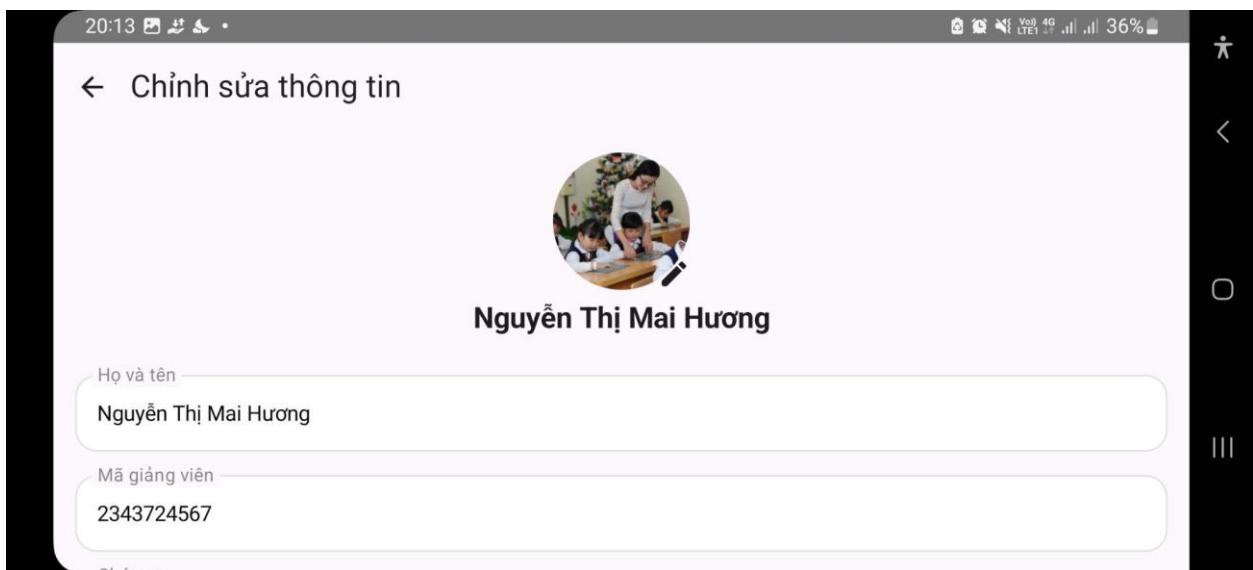
 Hồ sơ của bạn
Nguyễn Thị Mai Hương

B  **Bùi Thị Thoa**
Giảng viên

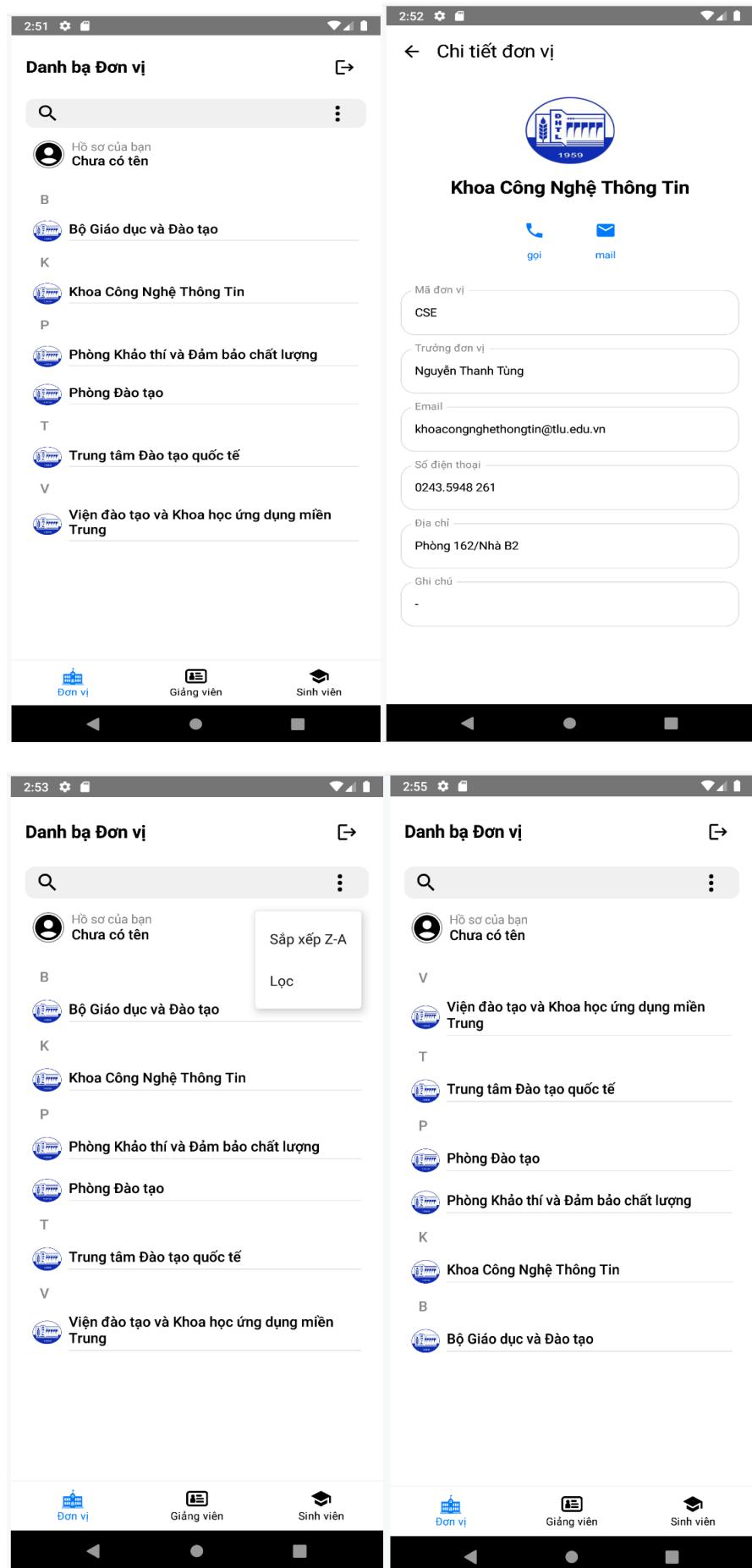
C  Đơn vị  Giảng viên  Sinh viên

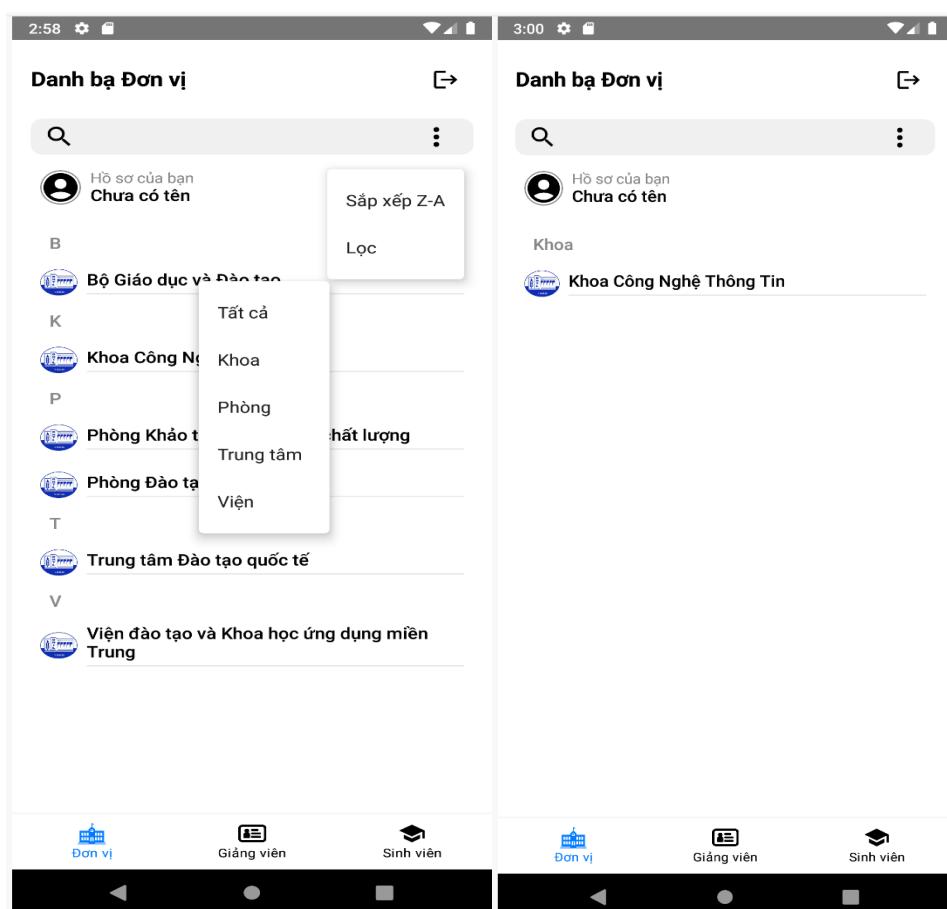
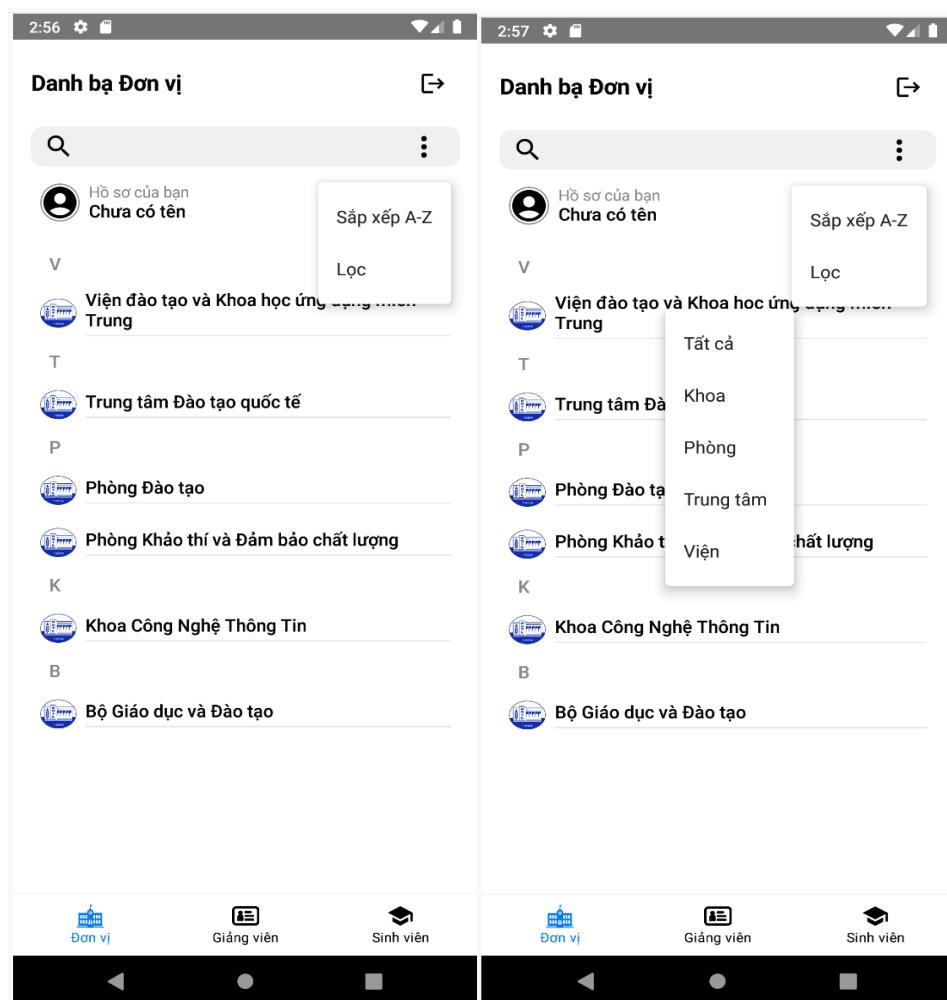


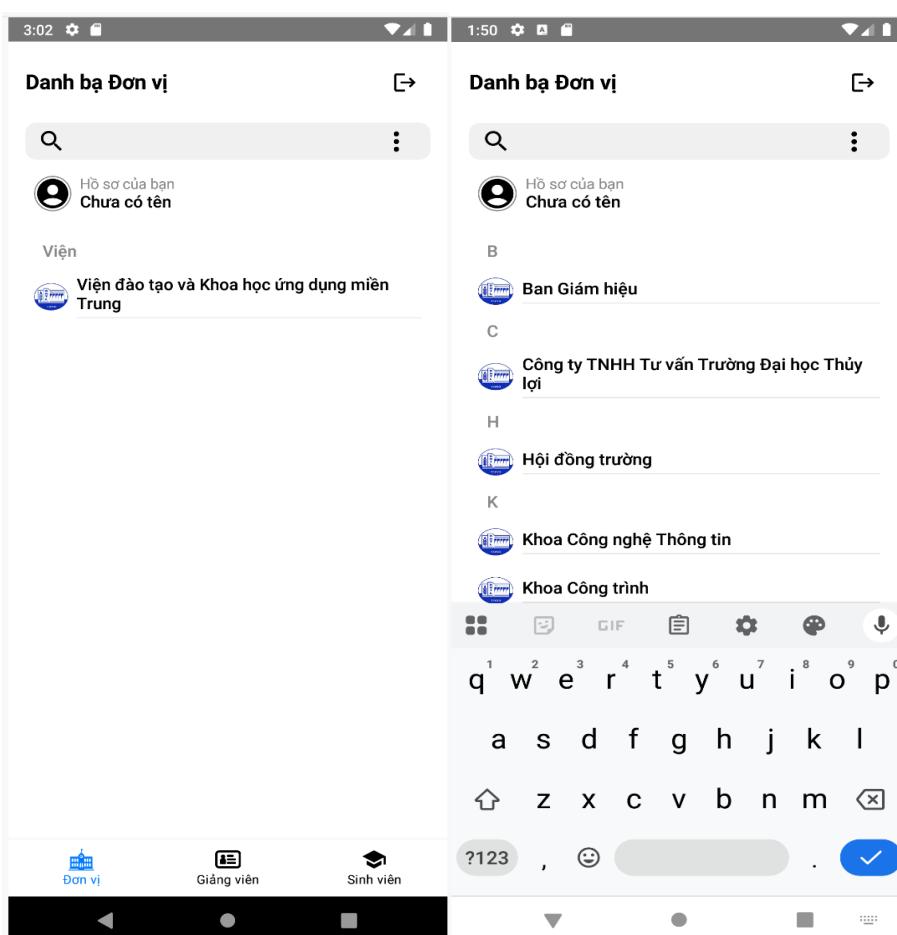
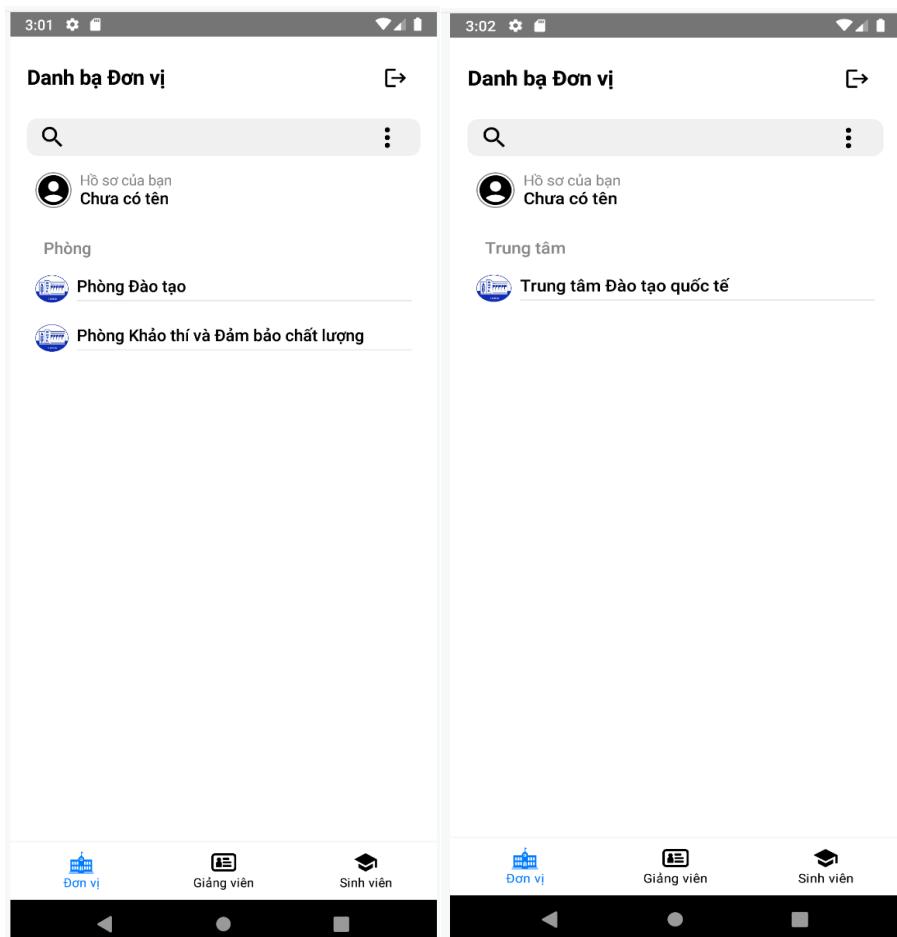


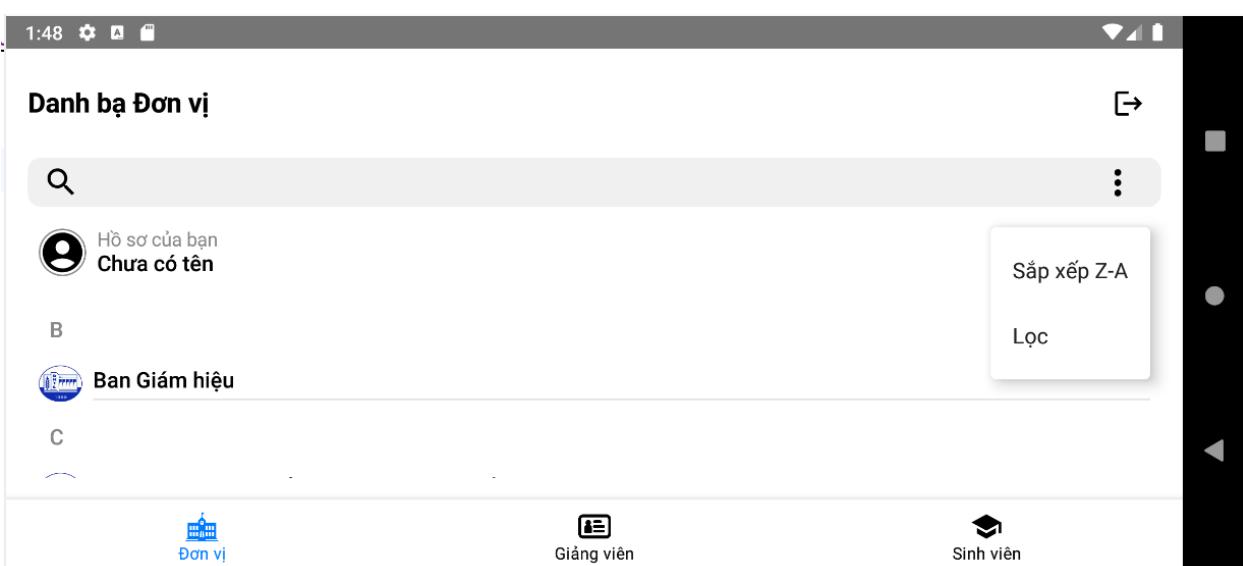
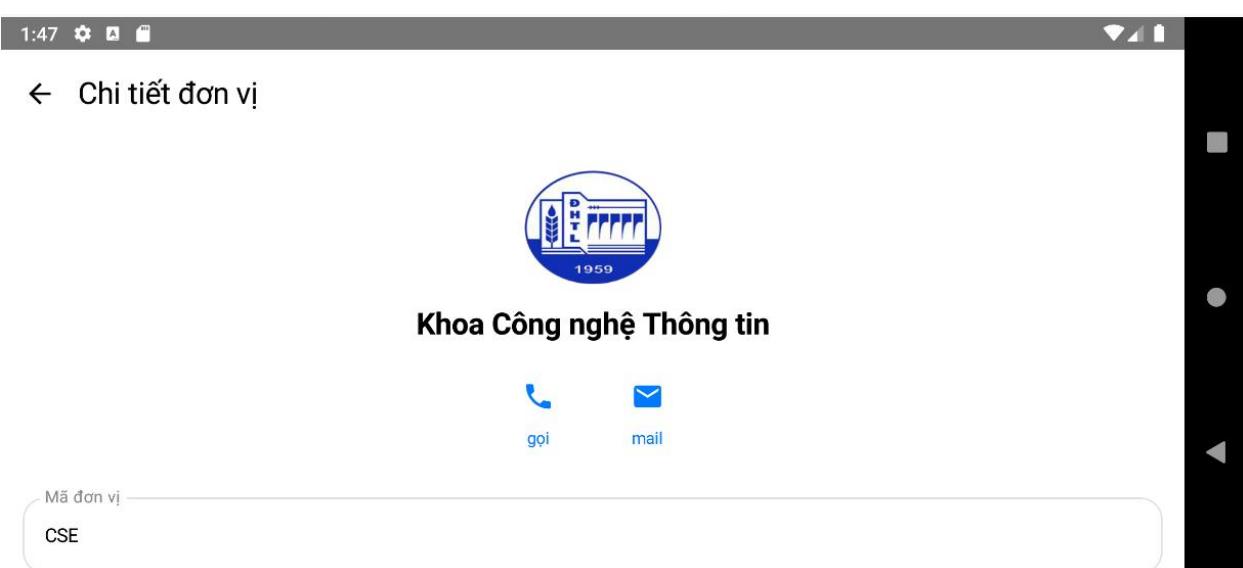
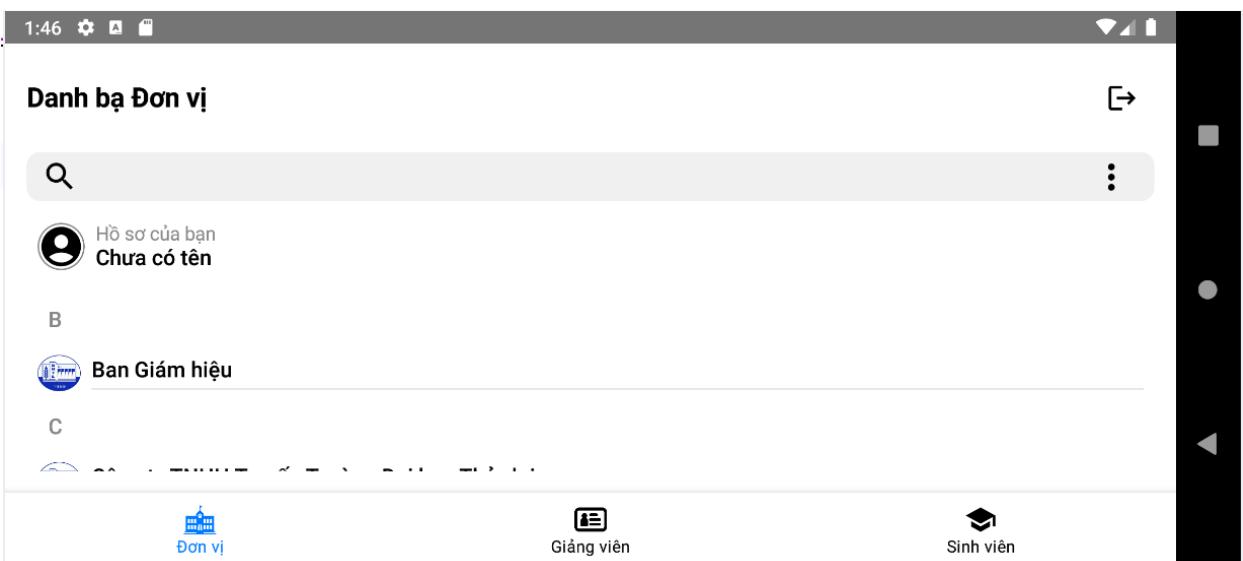


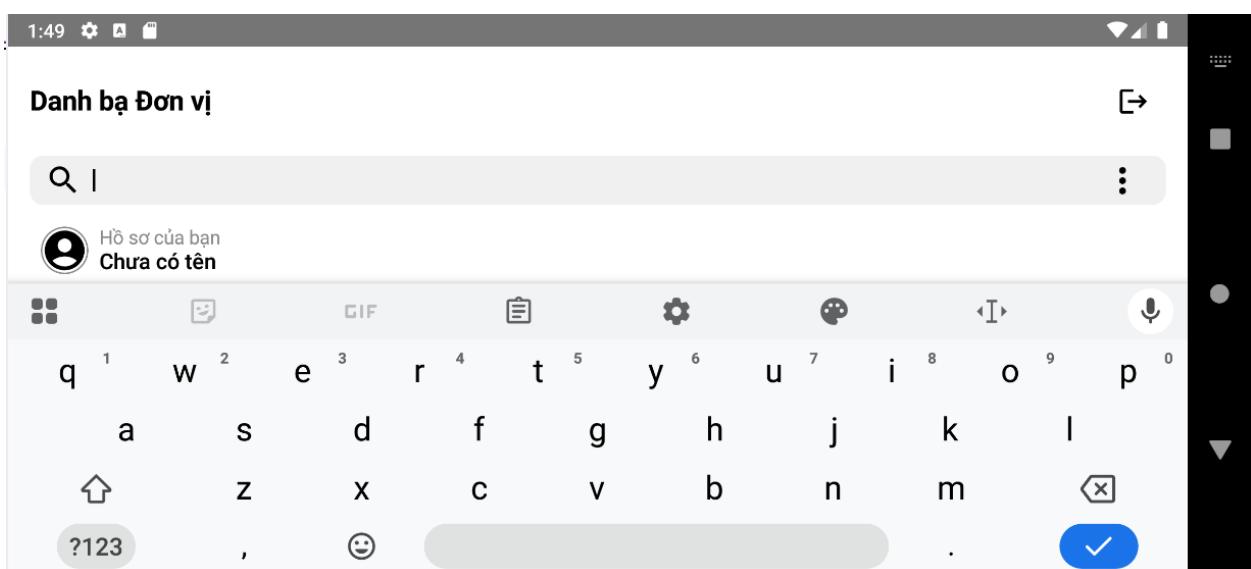
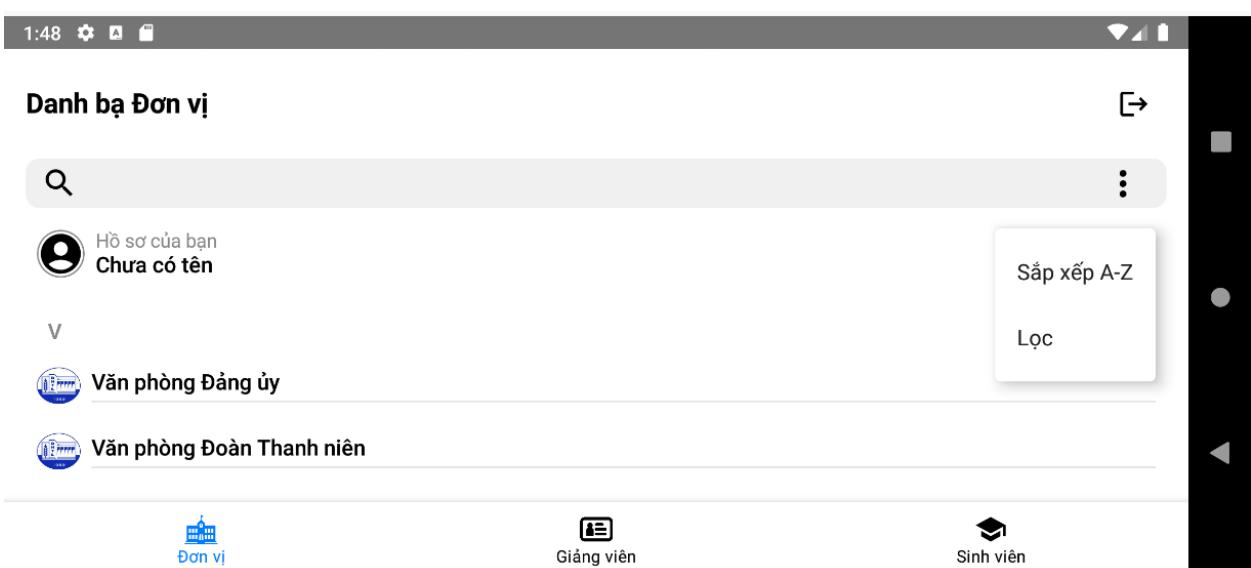
3.3.10. Giao diện xem danh bạ đơn vị



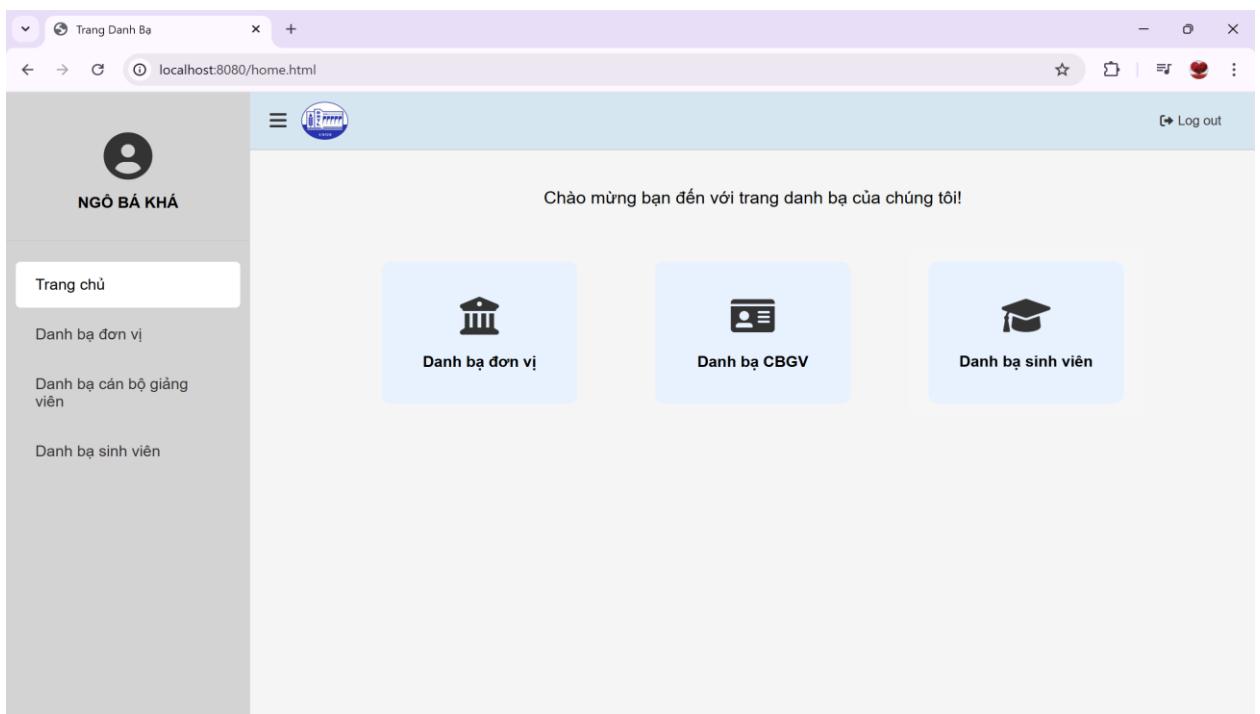
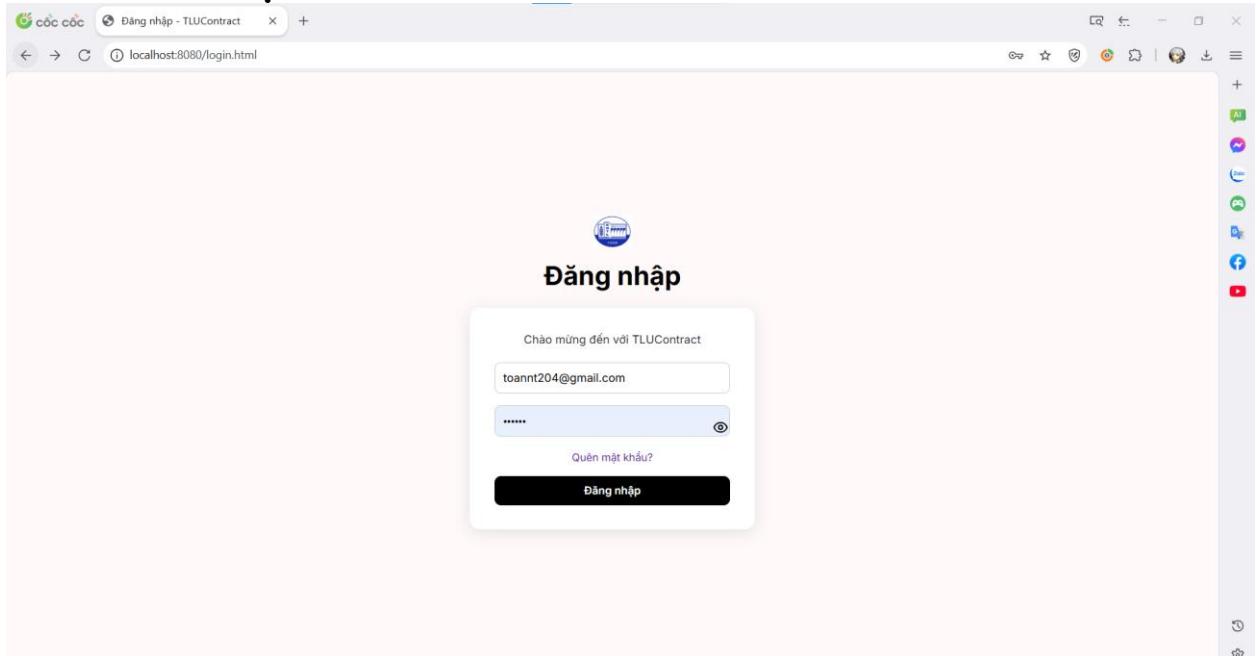


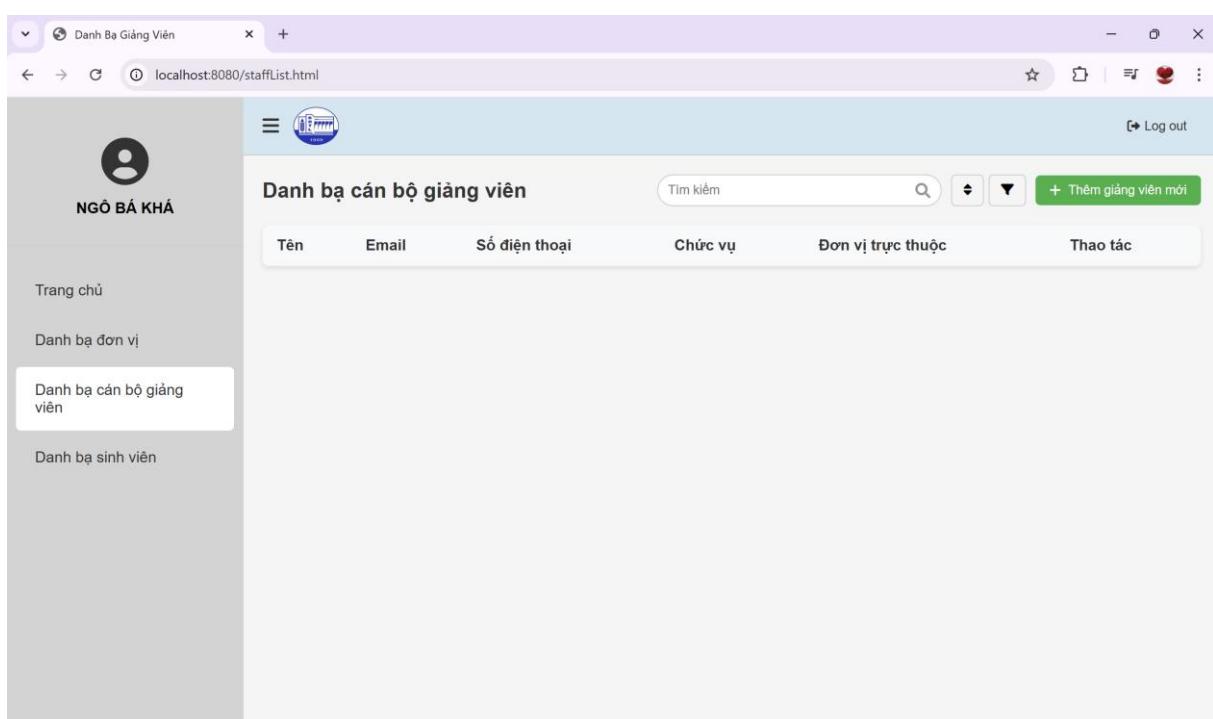
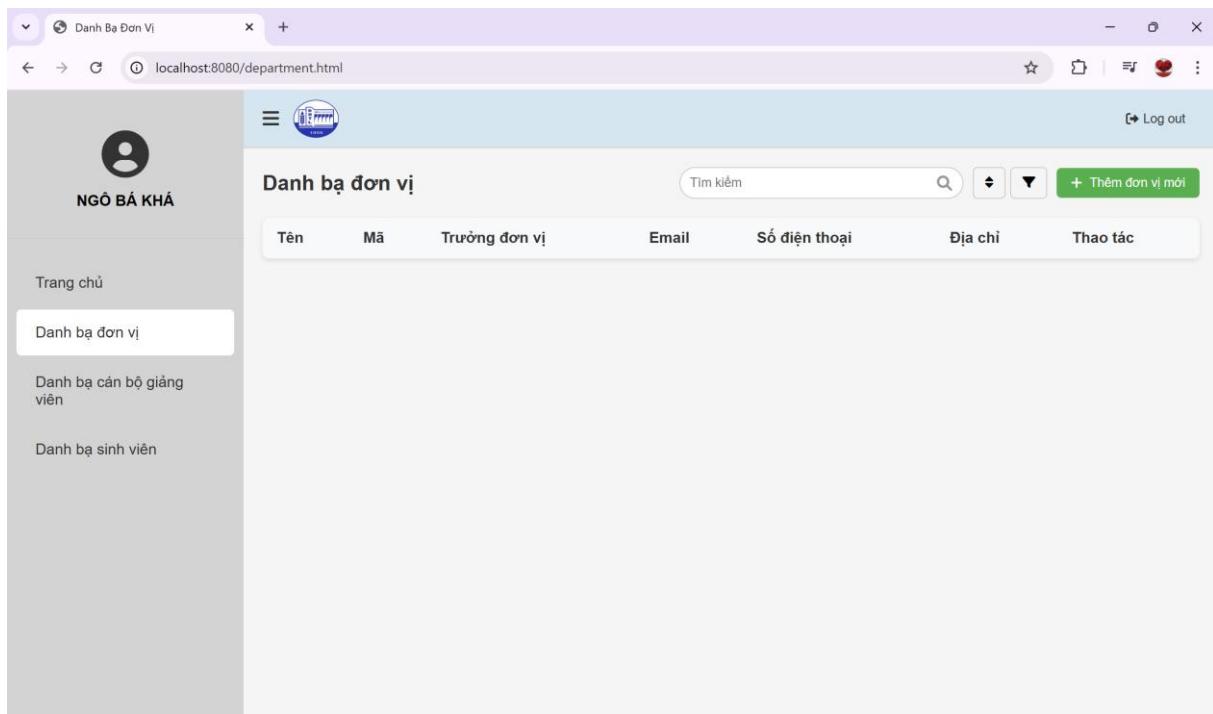


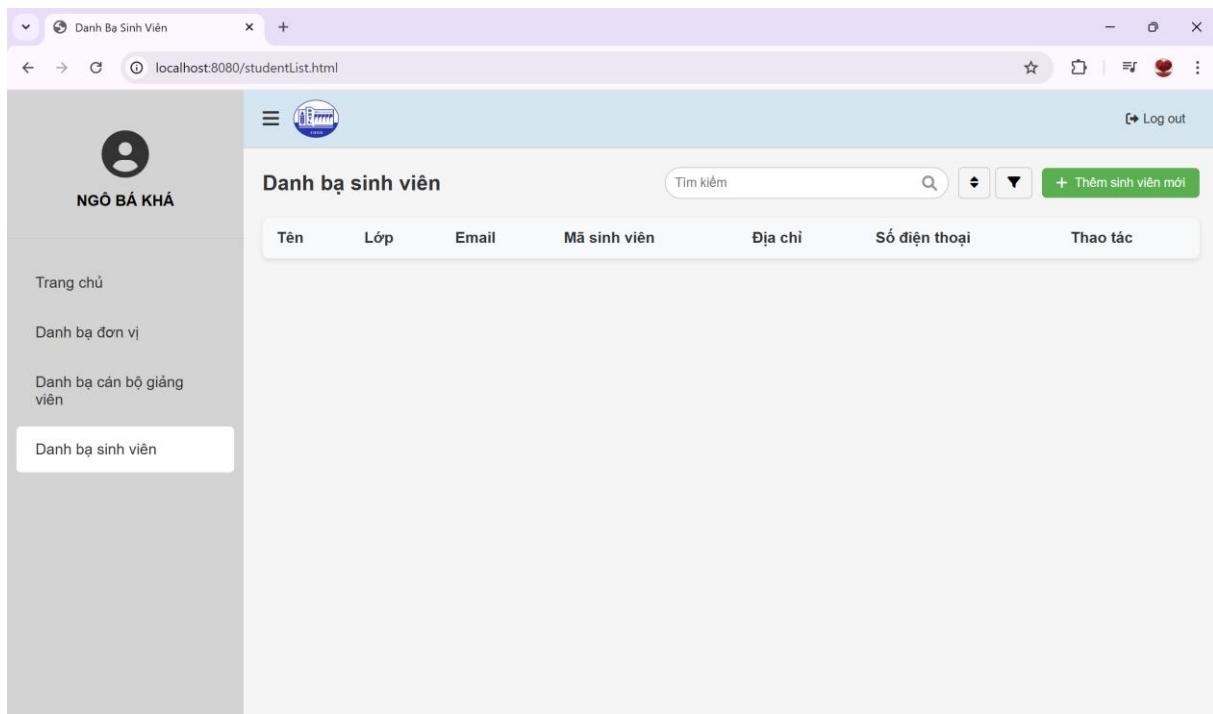




3.3.11. Giao diện web







3.4. Code minh họa các chức năng cốt lõi

Đăng nhập

```
10 < class LogInRepository(private val context: Context) {
11     private val auth = FirebaseAuth.getInstance() // Lấy instance của FirebaseAuth để sử dụng các chức năng xác thực
12
13     // Hàm xử lý đăng nhập với email và mật khẩu
14     fun login(email: String, password: String, onResult: (Boolean, String?) -> Unit) {
15         val trimmedEmail = email.trim() // Xóa khoảng trắng ở đầu và cuối email
16
17         // Gọi FirebaseAuth để đăng nhập bằng email và mật khẩu
18         auth.signInWithEmailAndPassword(trimmedEmail, password)
19             .addOnCompleteListener { task -> // Lắng nghe kết quả của quá trình đăng nhập
20                 if (task.isSuccessful) { // Nếu đăng nhập thành công
21                     val firebaseUser = auth.currentUser // Lấy thông tin người dùng hiện tại từ FirebaseAuth
22                     // Kiểm tra xem email đã được xác minh chưa
23                     if (firebaseUser != null && !firebaseUser.isEmailVerified) {
24                         onResult(false, "Tài khoản chưa được xác minh. Vui lòng kiểm tra email để xác minh tài khoản.") // Thông báo nếu email chưa
25                         return@addOnCompleteListener // Thoát khỏi hàm
26                     }
27
28                     // Lấy token của người dùng
29                     FirebaseAuth.getInstance().currentUser?.getIdToken(false)
30                         ?.addOnCompleteListener { tokenTask -> // Lắng nghe kết quả của quá trình lấy token
31                             if (tokenTask.isSuccessful) { // Nếu lấy token thành công
32                                 tokenTask.result?.token?.let { firebaseToken -> // Lấy token từ kết quả
33                                     // Lưu token bằng SessionManager
34                                     SessionManager(context).saveUserToken(firebaseToken) // Lưu token vào SessionManager
35                                     SessionManager(context).saveUserLoginEmail(trimmedEmail) // Lưu email vào SessionManager
36                                     onResult(true, firebaseToken) // Trả kết quả thành công kèm token
37                                 } ?: onResult(false, "Token is null") // Trả kết quả thất bại nếu token null
38                         }
39                     }
40                 }
41             }
42         }
43     }
44 }
```

```

38         } else {
39             onResult(false, tokenTask.exception?.message) // Trả kết quả thất bại kèm thông báo lỗi
40         }
41     }
42     } else {
43         onResult(false, task.exception?.message) // Trả kết quả thất bại kèm thông báo lỗi
44     }
45 }
46
47 // Hàm kiểm tra email có phải email của trường không
48 private fun isValidSchoolEmail(email: String): Boolean {
49     return email.endsWith("@tlu.edu.vn") || email.endsWith("@e.tlu.edu.vn") // Trả về true nếu email thuộc domain của trường
50 }
51
52
53 // Hàm đăng nhập bằng tài khoản Microsoft (Outlook)
54 fun loginWithMicrosoft(activity: Activity, callback: (Result<FirebaseUser>) -> Unit) {
55     val provider = OAuthProvider.newBuilder("microsoft.com").apply { // Tạo provider OAuth cho Microsoft
56         addCustomParameter("tenant", "tlu.edu.vn") // Thêm tham số tenant domain
57         scopes = listOf("openid", "profile", "User.Read", "email") // Thêm các scope cần thiết
58     }
59
60     if (auth.pendingAuthResult != null) { // Kiểm tra nếu có phiên đăng nhập Microsoft đang chờ xử lý
61         auth.pendingAuthResult?.addOnSuccessListener { authResult ->
62             Log.d("PROFILE_RAW", "Raw profile: ${authResult.additionalUserInfo?.profile ?: "null"}) // Xử lý thành công
63
64             handleMicrosoftAuthResult(authResult.user, callback) // Gọi hàm xử lý kết quả
65         }?.addOnFailureListener { exception -> // Xử lý thất bại
66             callback(Result.failure(exception)) // Trả về lỗi
67         }
68     } else {
69         auth.startActivityForSignInWithProvider(activity, provider.build()) // Bắt đầu đăng nhập với provider Microsoft
70         .addOnSuccessListener { authResult ->
71             Log.d("PROFILE_RAW", "Raw profile: ${authResult.additionalUserInfo?.profile ?: "null"}) // Xử lý thành công
72             handleMicrosoftAuthResult(authResult.user, callback) // Gọi hàm xử lý kết quả
73         }
74         .addOnFailureListener { exception -> // Xử lý thất bại
75             callback(Result.failure(exception)) // Trả về lỗi
76         }
77     }
78
79 // Hàm xử lý kết quả từ đăng nhập Microsoft
80 private fun handleMicrosoftAuthResult(
81     firebaseUser: FirebaseUser?, // Đối tượng người dùng Firebase trả về từ Microsoft
82     callback: (Result<FirebaseUser>) -> Unit // Callback để trả kết quả
83 ) {
84     if (firebaseUser == null) { // Kiểm tra nếu người dùng là null
85         callback(Result.failure(Exception("Đăng nhập không thành công."))) // Trả về lỗi đăng nhập không thành công
86         return // Thoát khỏi hàm
87     }
88     val userEmail = firebaseUser.email?.trim() ?: "" // Lấy email người dùng và xóa khoảng trắng
89     Log.d("USEREMAIL", "X UserEmail lef: ${userEmail}")
90     if (!isValidSchoolEmail(userEmail)) { // Kiểm tra nếu email không hợp lệ với định dạng trường
91         Log.d("USEREMAIL", "X UserEmail lef: ${userEmail}")
92         firebaseUser.delete() // Xóa tài khoản người dùng
93
94         callback(Result.failure(Exception("Email không hợp lệ. Vui lòng sử dụng email của trường."))) // Trả về lỗi email không hợp lệ
95         return // Thoát khỏi hàm
96     }
97
98     // Lấy token của người dùng và lưu lại
99     firebaseUser.getIdToken(false)
100     .addOnCompleteListener { tokenTask -> // Lắng nghe kết quả lấy token
101         if (tokenTask.isSuccessful) { // Nếu lấy token thành công
102             tokenTask.result?.token?.let { firebaseToken -> // Lấy token từ kết quả
103                 SessionManager(context).saveUserToken(firebaseToken) // Lưu token vào SessionManager
104                 SessionManager(context).saveUserLoginEmail(userEmail) // Lưu email vào SessionManager
105                 callback(Result.success(firebaseUser)) // Trả kết quả thành công với người dùng Firebase
106             } ?: callback(Result.failure(Exception("Token is null"))) // Trả kết quả thất bại nếu token null
107         } else {
108             callback(Result.failure(tokenTask.exception ?: Exception("Không thể lấy token"))) // Trả kết quả thất bại kèm thông báo lỗi
109         }
110     }
111 }

```

Đăng ký

```
7  class SignUpRepository() {
8      private val auth = FirebaseAuth.getInstance() // Lấy instance của FirebaseAuth để sử dụng các chức năng xác thực
9      private val firestore = FirebaseFirestore.getInstance() // Lấy instance của FirebaseFirestore để thao tác với database
10
11     // Hàm xử lý đăng ký tài khoản mới
12     fun signup(email: String, password: String, name: String, phone: String, onResult: (Boolean, String?) -> Unit) {
13         // Gọi FirebaseAuth để tạo tài khoản mới
14         auth.createUserWithEmailAndPassword(email, password)
15             .addOnCompleteListener { task -> // Lắng nghe kết quả của quá trình đăng ký
16                 if (task.isSuccessful) { // Nếu đăng ký thành công
17                     val uid = auth.currentUser?.uid ?: return@addOnCompleteListener // Lấy UID của người dùng đăng ký
18                     saveUserData(uid, email, name, phone) { success, error -> // Lưu dữ liệu người dùng vào Firestore
19                         if (success) { // Nếu lưu dữ liệu thành công
20                             auth.currentUser?.sendEmailVerification() // Gửi email xác thực
21                             onResult(true, null) // Trả kết quả thành công
22                         } else {
23                             onResult(false, error) // Trả kết quả thất bại kèm thông báo lỗi
24                         }
25                     }
26                 } else {
27                     onResult(false, task.exception?.message) // Trả kết quả thất bại kèm thông báo lỗi
28                 }
29             }
30         }
31     }
32
33     // Hàm xác định loại tài khoản dựa trên email
34     private fun detectUserType(email: String): String {
35         return when { // Sử dụng when để xác định loại tài khoản
36             email.endsWith("@tlu.edu.vn") -> "staff" // Nếu email kết thúc bằng @tlu.edu.vn thì loại là staff
37             email.endsWith("@e.tlu.edu.vn") -> "student" // Nếu email kết thúc bằng @e.tlu.edu.vn thì loại là student
38             else -> "guest" // Các trường hợp còn lại là guest
39         }
40     }
41
42     // Lưu dữ liệu người dùng dựa trên loại tài khoản
43     private fun saveUserData(uid: String, email: String, name: String, phone: String, onResult: (Boolean, String?) -> Unit) {
44         val userType = detectUserType(email) // Xác định loại tài khoản từ email
45
46         if (userType == "staff" || userType == "student") { // Nếu là staff hoặc student thì không lưu dữ liệu
47             onResult(true, null) // Trả kết quả thành công mà không lưu dữ liệu
48         } else {
49             val guest = Guest(email, phone, name, userType) // Tạo đối tượng Guest với thông tin người dùng
50             // Chuẩn bị dữ liệu để lưu vào Firestore
51             val userData = mapOf(
52                 "uid" to guest.uid, // UID của người dùng
53                 "name" to guest.name, // Tên của người dùng
54                 "phone" to guest.phone, // Số điện thoại của người dùng
55                 "avatarURL" to guest.avatarURL, // URL ảnh đại diện của người dùng
56                 "position" to guest.position,
57                 "department" to guest.department, // Chức vụ của người dùng
58                 "userId" to guest.userId, // ID người dùng
59                 "address" to guest.address, // Địa chỉ của người dùng
60                 "userType" to "guest" // Loại tài khoản (guest)
61             )
62             firestore.collection("guests") // Chọn collection "guests" trong Firestore
63                 .document(email) // Tạo document với ID là email
64                 .set(userData) // Lưu dữ liệu vào document
65                 .addOnCompleteListener { task -> // Lắng nghe kết quả của quá trình lưu
66                     if (task.isSuccessful) { // Nếu lưu thành công
67                         onResult(true, null) // Trả kết quả thành công
68                     } else {
69                         onResult(false, task.exception?.message) // Trả kết quả thất bại kèm thông báo lỗi
70                     }
71                 }
72         }
73     }
74 }
```

Khôi phục mật khẩu

```
5   <class> ForgotPasswordRepository() {
6       private val auth = FirebaseAuth.getInstance() // Lấy instance của FirebaseAuth để sử dụng các chức năng xác thực
7
8       // Hàm xử lý đặt lại mật khẩu
9       fun resetPassword(email: String, onResult: (Boolean, String?) -> Unit) {
10           val trimmedEmail = email.trim() // Xóa khoảng trắng ở đầu và cuối email
11           if (trimmedEmail.isEmpty()) { // Kiểm tra nếu email rỗng
12               onResult(false, "Vui lòng nhập email!") // Trả kết quả thất bại với thông báo email rỗng
13               return // Thoát khỏi hàm
14           }
15           auth.sendPasswordResetEmail(trimmedEmail) // Gửi email đặt lại mật khẩu qua Firebase
16           .addOnCompleteListener { task -> // Lắng nghe kết quả của quá trình gửi email
17               if (task.isSuccessful) { // Nếu gửi email thành công
18                   onResult(true, "Email đặt lại mật khẩu đã được gửi!") // Trả kết quả thành công
19               } else {
20                   onResult(false, task.exception?.message) // Trả kết quả thất bại kèm thông báo lỗi
21               }
22           }
23       }
24   }
```

Đăng xuất

```
6   class LogOutRepository(private val context: Context) {
7       private val auth = FirebaseAuth.getInstance() // Lấy instance của FirebaseAuth để sử dụng các chức năng xác thực
8
9       // Hàm xử lý đăng xuất
10      fun logout(onComplete: (Boolean, String?) -> Unit) {
11          try {
12              auth.signOut() // Đăng xuất khỏi Firebase
13              SessionManager(context).clearSession() // Xóa dữ liệu phiên làm việc
14              onComplete(true, null) // Thành công
15          } catch (e: Exception) {
16              onComplete(false, e.message) // Thất bại
17          }
18      }
19  }
```

Truy xuất thông tin guest

```
8  class GuestRepository {
9      private val db = Firebase Firestore.getInstance() // Lấy instance của Firebase Firestore
10     private val guestCollection = db.collection("guests") // Truy cập collection "guests" trong Firestore
11
12     // Hàm lấy thông tin của guest dựa vào email
13     fun getGuestByEmail(email: String, onResult: (Guest?) -> Unit) {
14         db.collection("guests").document(email).get() // Truy cập document của email trong collection "guests"
15             .addOnSuccessListener { doc -> // Xử lý khi lấy dữ liệu thành công
16                 if (doc.exists()) { // Kiểm tra xem document có tồn tại không
17                     val guest = Guest( // Tạo đối tượng Guest từ dữ liệu trong document
18                         userId = doc.getString("userId") ?: "", // Lấy userId, nếu null gán giá trị mặc định ""
19                         name = doc.getString("name") ?: "", // Lấy name, nếu null gán giá trị mặc định ""
20                         email = doc.id, // Lấy email, id của document là email
21                         phone = doc.getString("phone") ?: "", // Lấy phone, nếu null gán giá trị mặc định ""
22                         avatarURL = doc.getString("avatarURL") ?: "", // Lấy avatarURL, nếu null gán giá trị mặc định ""
23                         position = doc.getString("position") ?: "", // Lấy position, nếu null gán giá trị mặc định ""
24                         department = doc.getString("department") ?: "", // Lấy department, nếu null gán giá trị mặc định ""
25                         address = doc.getString("address") ?: "", // Lấy address, nếu null gán giá trị mặc định ""
26                         userType = doc.getString("userType") ?: "" // Lấy userType, nếu null gán giá trị mặc định ""
27                     )
28                     onResult(guest) // Trả kết quả là đối tượng Guest qua callback
29                 } else { // Nếu document không tồn tại
30                     onResult(null) // Trả kết quả null qua callback
31                 }
32             }
33             .addOnFailureListener { // Xử lý khi có lỗi xảy ra khi lấy dữ liệu
34                 onResult(null) // Trả kết quả null qua callback
35             }
36     }
37
38     // Hàm cập nhật thông tin guest
39     suspend fun updateGuest(guest: Guest): Result<Unit> { // Định nghĩa hàm cập nhật thông tin khách trả về Result
40         return try {
41             // Tạo một bản sao của đối tượng Guest mà không bao gồm email
42             val guestData = mapOf( // Tạo Map chứa dữ liệu của guest để lưu vào Firestore
43                 "userId" to guest.userId, // Ánh xạ trường userId từ đối tượng guest
44                 "name" to guest.name, // Ánh xạ trường name từ đối tượng guest
45                 "phone" to guest.phone, // Ánh xạ trường phone từ đối tượng guest
46                 "avatarURL" to guest.avatarURL, // Ánh xạ trường avatarURL từ đối tượng guest
47                 "position" to guest.position, // Ánh xạ trường position từ đối tượng guest
48                 "department" to guest.department, // Ánh xạ trường department từ đối tượng guest
49                 "address" to guest.address, // Ánh xạ trường address từ đối tượng guest
50                 "userType" to guest.userType // Ánh xạ trường userType từ đối tượng guest
51             )
52
53             // Lưu dữ liệu vào Firestore mà không bao gồm email
54             guestCollection.document(guest.email).set(guestData).await() // Cập nhật dữ liệu vào document có id là email
55             Result.success(Unit) // Trả về kết quả thành công nếu không có lỗi
56         } catch (e: Exception) { // Bắt ngoại lệ nếu có lỗi xảy ra
57             Result.failure(e) // Trả về kết quả thất bại với thông tin lỗi
58         }
59     }
60 }
```

Truy xuất dữ liệu sinh viên

```
// Hàm lấy danh sách sinh viên từ Firebase dựa trên email của người dùng hiện tại
public fun fetchStudents(currentUserEmail: String) {
    if (currentUserEmail.endsWith("@e.tlu.edu.vn")) {
        // Nếu là sinh viên → truy xuất className trước từ chính bản thân
        db.collection("student").document(currentUserEmail).get()
            .addOnSuccessListener { doc ->
                // Lấy tên lớp của sinh viên hiện tại
                val className = doc.getString("className") ?: ""
                if (className.isNotEmpty()) {
                    // Nếu có tên lớp, truy vấn tất cả sinh viên cùng lớp
                    db.collection("student")
                        .whereEqualTo("className", className)
                        .get()
                        .addOnSuccessListener { result ->
                            // Chuyển đổi các document thành đối tượng Student
                            val studentItems = result.map { studentDoc ->
                                Student(
                                    studentID = studentDoc.getString("studentID") ?: "",
                                    fullNameStudent = studentDoc.getString("fullNameStudent") ?: "Không có tên",
                                    photoURL = studentDoc.getString("photoURL") ?: "",
                                    email = studentDoc.getString("email") ?: "",
                                    phone = studentDoc.getString("phone") ?: "",
                                    address = studentDoc.getString("address") ?: "",
                                    className = studentDoc.getString("className") ?: "",
                                    userID = studentDoc.getString("userID") ?: ""
                                )
                            }
                            // Cập nhật danh sách sinh viên
                            _studentList.value = studentItems
                        }
                }
                // Cập nhật danh sách sinh viên
                _studentList.value = studentItems
            }
            .addOnFailureListener { e ->
                // Xử lý lỗi khi truy vấn danh sách sinh viên
                println("Lỗi lấy danh sách theo lớp: ${e.message}")
            }
        } else {
            // Xử lý trường hợp không tìm thấy tên lớp
            println("Không tìm thấy className của sinh viên $currentUserEmail")
        }
    }
    .addOnFailureListener { e ->
        // Xử lý lỗi khi lấy thông tin sinh viên hiện tại
        println("Lỗi lấy thông tin sinh viên hiện tại: ${e.message}")
    }
}
```

```

} else {
    // Nếu là staff → lấy toàn bộ sinh viên
    db.collection("student").get()
        .addOnSuccessListener { result ->
            // Chuyển đổi tất cả document thành đối tượng Student
            val studentItems = result.map { doc ->
                Student(
                    studentID = doc.getString("studentID") ?: "",
                    fullNameStudent = doc.getString("fullNameStudent") ?: "Không có tên",
                    photoURL = doc.getString("photoURL") ?: "",
                    email = doc.getString("email") ?: "",
                    phone = doc.getString("phone") ?: "",
                    address = doc.getString("address") ?: "",
                    className = doc.getString("className") ?: "",
                    userID = doc.getString("userID") ?: ""
                )
            }
            // Cập nhật danh sách sinh viên
            _studentList.value = studentItems
        }
        .addOnFailureListener { e ->
            // Xử lý lỗi khi lấy toàn bộ sinh viên
            println("Lỗi lấy toàn bộ sinh viên: ${e.message}")
        }
}

// Hàm lấy thông tin chi tiết của sinh viên theo email
fun setStudentByEmail(emailUser: String) {
    // Ghi log để debug
    Log.d("setStudentByEmail", "Bat dau vao ham: $emailUser")
    // Truy vấn document của sinh viên theo email
    db.collection("student").document(emailUser).get()
        .addOnSuccessListener { doc ->
            // Ghi log khi lấy dữ liệu thành công
            Log.d("setStudentByEmail", "lay xong du lieu: $emailUser")
            if (doc.exists()) {
                // Ghi log khi document tồn tại
                Log.d("setStudentByEmail", "Du lieu ton tai: $emailUser")
                try {
                    // Tạo đối tượng Student từ document
                    _selectedStudent.value = Student(
                        studentID = doc.getString("studentID") ?: "",
                        fullNameStudent = doc.getString("fullNameStudent") ?: "Không có tên",
                        photoURL = doc.getString("photoURL") ?: "",
                        email = doc.getString("email") ?: "",
                        phone = doc.getString("phone") ?: "",
                        address = doc.getString("address") ?: "",
                        className = doc.getString("className") ?: "",
                        userID = doc.getString("userID") ?: ""
                    )
                } catch (e: Exception) {
                    // Xử lý lỗi khi tạo đối tượng Student
                    println("Lỗi khi tạo đối tượng Student: ${e.message}")
                }
            } else {
                // Ghi log khi document không tồn tại
                Log.d("setStudentByEmail", "Du lieu khong ton tai: $emailUser")
            }
        }
}

```

```

        // Đặt giá trị null cho sinh viên được chọn
        _selectedStudent.value = null
    }
}

.addOnFailureListener { exception ->
    // Xử lý lỗi khi lấy dữ liệu
    println("Lỗi lấy dữ liệu setStudentByEmail: ${exception.message}")
}

// Hàm cập nhật thông tin sinh viên trong Firebase
fun updateStudentInfo(updatedStudent: Student) {
    // Cập nhật document sinh viên theo email
    db.collection("student").document(updatedStudent.email)
        .set(
            // Tạo map chứa các thông tin cần cập nhật
            mapOf(
                "fullNameStudent" to updatedStudent.fullNameStudent,
                "phone" to updatedStudent.phone,
                "studentID" to updatedStudent.studentID,
                "userID" to updatedStudent.userID,
                "email" to updatedStudent.email,
                "address" to updatedStudent.address,
                "className" to updatedStudent.className,
                "photoURL" to updatedStudent.photoURL
            )
        )
        .addOnSuccessListener {
            // Cập nhật đối tượng sinh viên đã chọn
            _selectedStudent.value = updatedStudent
            // Hiển thị thông báo thành công
            _snackBarMessage.value = "Cập nhật thông tin thành công"
        }
        .addOnFailureListener { exception ->
            // Hiển thị thông báo lỗi
            _snackBarMessage.value = "Lỗi cập nhật: ${exception.message}"
        }
}

```

Chức năng gọi, tin nhắn, gọi video, email cho Sinh viên

```
{  
    // Tạo nút "Tin nhắn" với biểu tượng chat  
    StudentActionButton(icon = Icons.Filled.Chat, label = "Tin nhắn") {  
        // Tạo Intent để mở ứng dụng nhắn tin  
        val intent = Intent(Intent.ACTION_VIEW).apply {  
            data = Uri.parse("sms:${student.phone}") // Cấu hình URI với số điện thoại của sinh viên  
        }  
        context.startActivity(intent) // Khởi chạy ứng dụng nhắn tin  
    }  
  
    // Tạo nút "Gọi" với biểu tượng điện thoại  
    StudentActionButton(icon = Icons.Filled.Phone, label = "Gọi") {  
        // Tạo Intent để mở ứng dụng gọi điện  
        val intent = Intent(Intent.ACTION_DIAL).apply {  
            data = Uri.parse("tel:${student.phone}") // Cấu hình URI với số điện thoại của sinh viên  
        }  
        context.startActivity(intent) // Khởi chạy ứng dụng gọi điện  
    }  
  
    // Tạo nút "Gọi video" với biểu tượng cuộc gọi video  
    StudentActionButton(icon = Icons.Filled.VideoCall, label = "Gọi video") {  
        // Bắt đầu khối try để xử lý các ngoại lệ có thể xảy ra  
        try {  
            // Tạo Intent để mở ứng dụng gọi video  
            val intent = Intent(Intent.ACTION_VIEW).apply {  
                data = Uri.parse("tel:${student.phone}") // Cấu hình URI với số điện thoại của sinh viên  
            }  
            context.startActivity(intent) // Khởi chạy ứng dụng gọi video  
        } catch (e: ActivityNotFoundException) {  
            // Bắt ngoại lệ khi không tìm thấy ứng dụng phù hợp  
            Toast.makeText(context, "Thiết bị không hỗ trợ gọi video", Toast.LENGTH_SHORT).show()  
        } catch (e: Exception) {  
            // Bắt các ngoại lệ khác có thể xảy ra  
            Toast.makeText(context, "Không thể thực hiện cuộc gọi video", Toast.LENGTH_SHORT).show()  
        }  
    }  
  
    // Tạo nút "Mail" với biểu tượng email  
    StudentActionButton(icon = Icons.Filled.Email, label = "Mail") {  
        // Tạo Intent để mở ứng dụng email  
        val intent = Intent(Intent.ACTION_SENDTO).apply {  
            data = Uri.parse("mailto:${student.email}") // Cấu hình URI với địa chỉ email của sinh viên  
        }  
        context.startActivity(intent) // Khởi chạy ứng dụng email  
    }  
}
```

```

// Hàm lấy chữ cái đầu của tên riêng
fun extractFirstLetterOfLastName(fullName: String): Char {
    return fullName.trim()
        .split(" ")
        .lastOrNull()
        ?.firstOrNull()
        ?.uppercaseChar() ?: '#'
}

// Hàm dùng để sắp xếp theo tên riêng
fun extractLastNameForSort(fullName: String): String {
    return fullName.trim()
        .split(" ")
        .lastOrNull()
        ?.lowercase() ?: ""
}

// Hàm đảo ngược trang thái sắp xếp (từ tăng dần sang giảm dần hoặc ngược lại)
fun toggleSortOrder() {
    _sortAscending.value = !_sortAscending.value
}

// Sắp xếp danh sách sinh viên theo tên
val sortedStudents = if (sortAscending) {
    filteredStudents.sortedBy { extractLastNameForSort(it.fullNameStudent) }
} else {
    filteredStudents.sortedByDescending { extractLastNameForSort(it.fullNameStudent) }
}

```

Lọc cho Student

```

// Hàm cập nhật chế độ lọc (ByName hoặc ByClass)
fun setFilterMode(mode: String) {
    _filterMode.value = mode
    // Không cần lấy lại dữ liệu, chỉ thay đổi cách hiển thị trên UI
}

val filterMode by studentViewModel.filterMode.collectAsState() // Lấy chế độ lọc từ ViewModel
// Lọc danh sách sinh viên theo tên
val filteredStudents = students.filter { student ->
    student.fullNameStudent.contains(query, ignoreCase = true) || // Tìm kiếm theo tên
        student.studentID.equals(query, ignoreCase = true) // Tìm kiếm theo mã sinh viên
}

```

Truy xuất dữ liệu giảng viên

```
// ...  
fun fetchStaffs(query: String?) { // query là từ khóa tìm kiếm (? là có thể null)  
    viewModelScope.launch { // Khởi chạy coroutine trong viewModelScope // Để xử lý bất đồng bộ để tải dữ liệu Firestore hoặc  
    // giúp chạy các coroutine bất đồng bộ trong ViewModel một cách dễ dàng và an toàn mà không phải lo lắng về việc hủy ch  
    // Kiểm tra xem có kết nối Internet không  
    var ref: Query = db.collection(collectionPath: "staffs") // Truy cập collection "staffs"  
  
    // Thêm điều kiện sắp xếp theo tên  
    ref = if (_sortAscending.value) { // Nếu đang sắp xếp tăng dần  
        ref.orderBy(field: "fullName") // Sắp xếp theo tên giảng viên  
    } else { // Nếu đang sắp xếp giảm dần  
        ref.orderBy(field: "fullName", Query.Direction.DESCENDING) // Sắp xếp theo tên giảng viên giảm dần  
    } //SD ORDERBY()  
  
    // Áp dụng điều kiện lọc theo chế độ  
    when (_filterMode.value) { // Kiểu lọc: All/ByDepartment/ByPosition  
        "ByDepartment" -> { // Lọc theo đơn vị  
            if (!query.isNullOrEmpty()) { // Nếu query không null hoặc rỗng  
                ref = ref.whereGreaterThanOrEqualTo(field: "unit", query) // tim tương đối  
                    .whereLessThanOrEqualTo(field: "unit", value: query + '\uf8ff') // tim tương đối  
            } // Nếu không có query, sẽ lấy tất cả giảng viên  
        }  
        // Lọc theo vị trí  
        "ByPosition" -> { // Lọc theo vị trí  
            if (!query.isNullOrEmpty()) { // Nếu query không null hoặc rỗng  
                ref = ref.whereGreaterThanOrEqualTo(field: "position", query)  
                    .whereLessThanOrEqualTo(field: "position", value: query + '\uf8ff')  
            }  
        }  
        "ByAll" -> {  
            if (!query.isNullOrEmpty()) {  
                ref = ref.whereGreaterThanOrEqualTo(field: "fullName", query)  
                    .whereLessThanOrEqualTo(field: "fullName", value: query + '\uf8ff')  
            }  
        }  
    }  
  
    // Giới hạn số kết quả trả về  
    ref = ref.limit(100) // Giới hạn số lượng kết quả trả về tối đa là 100  
  
    // Lấy dữ liệu từ Firestore  
    ref.get() // Trả về một Task chưa kết quả  
        .addOnSuccessListener { result -> // Khi lấy dữ liệu thành công  
            // Chuyển dữ liệu từ Firestore thành danh sách Staff  
            val staffItems = result.map { doc -> // Chuyển đổi từng document thành Staff  
                Staff( // Khởi tạo đối tượng Staff  
                    staffId = doc.getString(field: "staffid") ?: "",  
                    staffIdFB = doc.getString(field: "staffid") ?: "", // Lấy staffId từ document  
                    name = doc.getString(field: "fullName") ?: "Không có tên", // Lấy tên từ document  
                    email = doc.id, // Lấy email từ document ID  
                    phone = doc.getString(field: "phone") ?: "", // Lấy số điện thoại từ document  
                    department = doc.getString(field: "unit") ?: "", // Lấy đơn vị từ document  
                    position = doc.getString(field: "position") ?: "", // Lấy vị trí từ document  
                    avatarURL = doc.getString(field: "photoURL") ?: "" // Lấy URL ảnh từ document  
                )  
            }  
            _staffList.value = staffItems // Cập nhật StateFlow  
        }  
        .addOnFailureListener { e -> // Khi có lỗi xảy ra  
            Log.e(tag: "Firestore", msg: "Lỗi lấy danh sách giảng viên: ${e.message}") // Ghi log lỗi để dễ kiểm tra  
        }  
    }  
}
```

```

// Hàm tìm giảng viên theo email (dùng khi vào trang chỉnh sửa)
fun setStaffByEmail(emailUser: String) { // emailUser là email của giảng viên
    db.collection( collectionPath: "staffs").document(emailUser).get() // Lấy document theo email
        .addOnSuccessListener { doc -> // Khi lấy dữ liệu thành công
            if (doc.exists()) { // Kiểm tra xem document có tồn tại không
                _selectedStaff.value = Staff( // Khởi tạo đối tượng Staff
                    staffId = doc.id,
                    name = doc.getString( field: "fullName") ?: "Không có tên",
                    staffIdFB = doc.getString( field: "staffid") ?: "",
                    email = doc.getString( field: "email") ?: "",
                    phone = doc.getString( field: "phone") ?: "",
                    department = doc.getString( field: "unit") ?: "",
                    position = doc.getString( field: "position") ?: "",
                    avatarURL = doc.getString( field: "photoURL") ?: ""
                )
            }
        }
    }
    .addOnFailureListener { e -> // Khi có lỗi xảy ra
        Log.e( tag: "Firestore", msg: "Lỗi lấy giảng viên: ${e.message}") // Ghi log lỗi để dễ kiểm tra
    }
}

// Cập nhật thông tin giảng viên lên Firestore
fun updateStaffInfo(updatedStaff: Staff) { // updatedStaff là đối tượng Staff đã được chỉnh sửa
    db.collection( collectionPath: "staffs").document(updatedStaff.staffId) // Truy cập document theo staffId
        .set( // Cập nhật dữ liệu trong document
            mapOf( // Sử dụng map để truyền dữ liệu
                "fullName" to updatedStaff.name, // Tên giảng viên
                "phone" to updatedStaff.phone, // Số điện thoại
                "unit" to updatedStaff.department, // Đơn vị
                "position" to updatedStaff.position, // Vị trí
                "staffid" to updatedStaff.staffIdFB, // ID giảng viên
                "userId" to updatedStaff.userId, // ID người dùng
                "photoURL" to updatedStaff.avatarURL // URL ảnh đại diện
            )
        )
        .addOnSuccessListener { // Khi cập nhật thành công
            _selectedStaff.value = updatedStaff // Cập nhật dữ liệu trong ViewModel
            _updateMessage.value = "Cập nhật thông tin thành công" // Thông báo cập nhật thành công
            _isUpdateSuccessful.value = true // Đánh dấu cập nhật thành công
        }
        .addOnFailureListener { e -> // Khi có lỗi xảy ra
            _updateMessage.value = "Lỗi cập nhật: ${e.message}" // Thông báo lỗi
            _isUpdateSuccessful.value = false // Đánh dấu cập nhật thất bại
        }
}

```

```

// Upload ảnh avatar lên Firebase Storage, gọi callback khi thành công hoặc lỗi
fun uploadImageToStorage( // Hàm này dùng để upload ảnh lên Firebase Storage
    uri: Uri?, // Uri của ảnh cần upload
    onSuccess: (String) -> Unit, // Callback khi upload thành công, trả về URL ảnh
    onFailure: (Exception) -> Unit
) {
    if (uri == null) {
        onFailure(IllegalArgumentException("Uri ảnh không được null"))
        return
    }

    val storageRef = FirebaseStorage.getInstance().reference
    val imageRef = storageRef.child( pathString: "avatars/${UUID.randomUUID()}.jpg") // Đặt tên file ngẫu nhiên

    imageRef.putFile(uri) // Upload ảnh lên Firebase, uri là kiểu dữ liệu cho phép lưu đường dẫn đến ảnh
        .addOnSuccessListener { // Khi upload thành công
            imageRef.downloadUrl.addOnSuccessListener { url -> // Lấy URL của ảnh vừa upload
                onSuccess(url.toString()) // Trả về đường dẫn URL khi thành công
            }.addOnFailureListener { onFailure(it) } // Nếu lấy URL thất bại
        }
        .addOnFailureListener { onFailure(it) } // Nếu upload thất bại
    }
}

```

Chức năng nhắm tin, gửi mail, gọi video, gọi điện của danh bạ giảng viên

```

// Nút chức năng với icon và nhắm
fun ActionButton(icon: ImageVector, label: String, phoneNumber: String? = null) { // Hàm tạo nút chức năng
    val context = LocalContext.current // Lấy context để gọi intent

    Column(horizontalAlignment = Alignment.CenterHorizontally) { // Cột chứa icon và nhắm
        IconButton(onClick = { // Xử lý khi nhấn nút
            // Xử lý hành động tương ứng với nút được nhấn
            when (label) { // Kiểm tra nhắm của nút
                "Gọi" -> phoneNumber?.let { makePhoneCall(context, it) } // Gọi điện thoại
                "Tin nhắn" -> phoneNumber?.let { sendSMS(context, it) } // Gửi tin nhắn
                "Mail" -> phoneNumber?.let { sendEmail(context, it) } // Gửi email
                "Gọi video" -> phoneNumber?.let { openVideoCallApp(context, it) } // Gọi video
            }
        })
        // Icon hiển thị trên nút
        Icon(imageVector = icon, contentDescription = label, tint = Color( color: 0xFF007AFF))
    }

    Text( // Label dưới nút
        text = label, // Nhắm của nút
        fontSize = 12.sp, // cỡ chữ 12
        color = Color( color: 0xFF007AFF), // Màu chữ
        modifier = Modifier.clickable { // Xử lý khi nhấn vào chữ
            // Cũng xử lý khi người dùng nhấn vào chữ
            when (label) { // Kiểm tra nhắm
                "Gọi" -> phoneNumber?.let { makePhoneCall(context, it) } // Gọi điện thoại
                "Tin nhắn" -> phoneNumber?.let { sendSMS(context, it) } // Gửi tin nhắn
                "Mail" -> phoneNumber?.let { sendEmail(context, it) } // Gửi email
                "Gọi video" -> phoneNumber?.let { openVideoCallApp(context, it) } // Gọi video
            }
        }
    )
}

```

```

fun sendSMS(context: Context, phoneNumber: String) { // Gửi tin nhắn SMS
    val intent = Intent(Intent.ACTION_VIEW, Uri.parse("sms:$phoneNumber")) // Mở app SMS với số đã điền
    context.startActivity(intent) // Bắt đầu activity với intent
}

fun sendEmail(context: Context, email: String) { // Gửi email
    val intent = Intent(Intent.ACTION_SENDTO).apply { // Tạo intent gửi email
        data = Uri.parse("mailto:$email") // Mở app email với email đã điền
    }
    context.startActivity(intent) // Bắt đầu activity với intent
}

fun openVideoCallApp(context: Context, phoneNumber: String) { // Mở ứng dụng gọi video
    val intent = context.packageManager.getLaunchIntentForPackage("com.google.android.apps.meetings") // Gọi Google Meet
    if (intent != null) { // Nếu ứng dụng đã cài
        context.startActivity(intent) // Bắt đầu activity với intent
    } else { // Nếu chưa cài
        // Nếu chưa cài, mở Google Play để cài
        val playStoreIntent = Intent(Intent.ACTION_VIEW, Uri.parse("market://details?id=com.google.android.apps.meetings")) // Mở Google Play để cài
        context.startActivity(playStoreIntent) // Bắt đầu activity với intent
    }
}

```

```

// Hàm này sẽ được gọi khi người dùng nhấn vào nút gọi điện thoại
fun makePhoneCall(context: Context, phoneNumber: String) { // Gọi điện thoại
    // Kiểm tra xem ứng dụng có quyền gọi điện hay không
    if (ContextCompat.checkSelfPermission(context, Manifest.permission.CALL_PHONE) != PackageManager.PERMISSION_GRANTED) {
        // Nếu chưa được cấp quyền gọi điện, yêu cầu quyền
        ActivityCompat.requestPermissions((context as Activity), // Chuyển đổi context thành Activity
            arrayOf(Manifest.permission.CALL_PHONE), // Yêu cầu quyền gọi điện
            requestCode: 1 // Mã yêu cầu quyền
        )
    } else { // Nếu đã có quyền
        val intent = Intent(Intent.ACTION_CALL) // Gọi trực tiếp
        intent.data = Uri.parse("tel:$phoneNumber") // Đặt số điện thoại vào intent
        context.startActivity(intent) // Bắt đầu activity với intent
    }
}

```

Chức năng lọc và sắp xếp

Quy tắc lọc

```

// Cập nhật chế độ lọc và làm mới danh sách
fun setFilterMode(mode: String) { // mode là kiểu lọc: All, ByDepartment, ByPosition
    _filterMode.value = mode // cập nhật chế độ lọc
    fetchStaffs(query = null) // gọi lại hàm fetchStaffs để cập nhật danh sách
}

```

Đảo cách sắp xếp:

```

// Hàm đảo ngược trạng thái sắp xếp, đồng thời làm mới danh sách
fun toggleSortOrder() {
    _sortAscending.value = !_sortAscending.value // đảo ngược giá trị hiện tại
    fetchStaffs(query = null) // gọi lại hàm fetchStaffs để cập nhật danh sách
}

```

Logic lọc và sắp xếp được định nghĩa trong fun StaffList

```
@Composable
fun Stafflist(
    staffs: List<Staff>, // Danh sách tất cả giảng viên
    query: String, // Chuỗi tìm kiếm (tên giảng viên)
    navController: NavController, // Dung để điều hướng đến màn hình chi tiết
    staffViewModel: StaffViewModel = viewModel() // ViewModel quản lý danh sách và trang thái lọc/sắp xếp
) {
    // Lấy trang thái sắp xếp từ ViewModel (tăng hay giảm dần theo tên)
    val sortAscending by staffViewModel.sortAscending.collectAsState()
    val filterMode by staffViewModel.filterMode.collectAsState() // Lấy chế độ lọc từ ViewModel

    val collator = Collator.getInstance(Locale(language: "vi", country: "VN"))
    collator.strength = Collator.PRIMARY // Bỏ qua phân biệt hoa thường và dấu

    // Sắp xếp danh sách giảng viên theo tên (tăng/giảm dần)
    val sortedStaffs = if (sortAscending) { // sắp xếp theo collator để có chữ tiếng việt
        staffs.sortedWith(compareBy(collator) { it.name }) // sx giảm dần
    } else {
        staffs.sortedWith(compareByDescending(collator) { it.name }) // sắp xếp tăng dần
    }

    // Lọc danh sách giảng viên theo tên và filterMode
    // Duyệt qua từng phần tử trong danh sách sortedStaffs và chỉ giữ lại các phần tử thỏa mãn điều kiện
    val filteredStaffs = sortedStaffs.filter { // khai báo biến chứa danh sách đã lọc
        // kiểm tra xem tên của staff có chứa chuỗi query không (không phân biệt chữ hoa và thường)
        it.name.contains(query, ignoreCase = true) //
    }

    // Nhóm danh sách theo chế độ lọc
    val groupedStaffs = when (filterMode) {
        // khai báo 1 biến và gán cho nó kết quả phân nhóm (groupBy) danh sách filteredStaffs theo tiêu chí được chọn trong filterMode
        "ByAll" -> filteredStaffs.groupBy { it.name.firstOrNull()?.uppercaseChar() }
        "ByDepartment" -> filteredStaffs.groupBy { it.department }
    }
}

fun Stafflist(
    // Nhóm danh sách theo chế độ lọc
    val groupedStaffs = when (filterMode) {
        // khai báo 1 biến và gán cho nó kết quả phân nhóm (groupBy) danh sách filteredStaffs theo tiêu chí được chọn trong filterMode
        "ByAll" -> filteredStaffs.groupBy { it.name.firstOrNull()?.uppercaseChar() }
        "ByDepartment" -> filteredStaffs.groupBy { it.department }
        "ByPosition" -> filteredStaffs.groupBy { it.position } // it đại diện cho từng staff, name là tên của staff,
        // firstOrNull lấy kí tự đầu tiên trong tên, nếu rỗng trả về null, nếu có ký tự đầu tiên, thì chuyển nó thành chữ in hoa.

        else -> filteredStaffs.groupBy { it.name.firstOrNull()?.uppercaseChar() } // Nếu key không khớp với key hiện có
    }

    // Hiển thị danh sách dạng LazyColumn (cuộn được)
    LazyColumn {
        groupedStaffs.forEach { (key, staffList) -> // Nhóm staff theo key là chức vụ hoặc đơn vị trong danh sách staff
            item { // Đây là tên của group (ví dụ: Khoa công nghệ thông tin hay Giảng viên)
                Text(
                    text = key.toString(), // Hiển thị tên nhóm (bộ môn hoặc chức vụ)
                    fontSize = 16.sp, // cỡ chữ 16dp
                    fontWeight = FontWeight.Medium, // độ đậm của chữ
                    color = Color.Gray, // màu chữ xám
                    modifier = Modifier // điều chỉnh
                        .fillMaxWidth() // giãn hết cỡ
                        .padding(horizontal = 12.dp, vertical = 4.dp) // đệm theo chiều ngang 12dp, chiều dọc 4dp
                )
            }
        }

        // Hiển thị giảng viên thuộc nhóm đó
        items(staffList) { staff -> // duyệt qua danh sách staff, với mỗi staff thi vẽ 1 Staffitem như dưới
            Staffitem( // toàn bộ một item dạng staff trong danh sách
                staff = staff, // gán là đối tượng staff
                isSelected = false, // gán là chưa chọn
                onClick = {

```

Phần logic lọc trong SearchBar

```
2     fun Searchbar(
3         ) {
4             ) {
5                 Box {
6                     ) {
7
8                         "Giảng viên" -> { // Lọc theo giảng viên
9                             DropdownMenuItem(onClick = {
10                                 staffViewModel.setFilterMode("ByAll") // Lọc theo tất cả
11                                 expanded = false
12                                 expandedFilter = false
13                             }) {
14                                 Text( text: "Tất cả") // Tiêu đề lọc theo all
15                             }
16                             DropdownMenuItem(onClick = {
17                                 staffViewModel.setFilterMode("ByDepartment") // Lọc theo đơn vị
18                                 expanded = false
19                                 expandedFilter = false
20                             }) {
21                                 Text( text: "Theo Đơn vị") // Tiêu đề lọc theo đơn vị
22                             }
23                             DropdownMenuItem(onClick = { // Lọc theo chức vụ
24                                 staffViewModel.setFilterMode("ByPosition") // Lọc theo chức vụ
25                                 expanded = false // Đóng menu
26                                 expandedFilter = false // Đóng menu
27                             }) {
28                                 Text( text: "Theo Chức vụ") // Tiêu đề lọc theo chức vụ
29                             }
30                         )
31                     )
32                 )
33             )
34         )
35     )
36 }
```

Truy xuất dữ liệu đơn vị

```
// Hàm này sẽ được gọi khi bạn muốn lấy danh sách đơn vị từ Firestore
private fun fetchDepartments() {
    // Lấy danh sách đơn vị từ Firestore
    db.collection("department").get()
        .addOnSuccessListener { result ->
            val departmentItems = result.map { doc ->
                // Chuyển đổi tài liệu Firestore thành đối tượng Department
                Department(
                    id = doc.getString("id") ?: "",
                    name = doc.getString("name") ?: "Không có tên",
                    leader = doc.getString("leader") ?: "",
                    email = doc.getString("email") ?: "",
                    phone = doc.getString("phone") ?: "",
                    address = doc.getString("address") ?: "",
                    photoURL = doc.getString("photoURL") ?: "",
                    type = doc.getString("type") ?: ""
                )
            }
            // Sắp xếp danh sách theo tên
            _departmentList.value = departmentItems
            applyFilters() // Gọi applyFilters ngay sau khi lấy dữ liệu
        }
        .addOnFailureListener { exception ->
            println("Lỗi lấy dữ liệu: ${exception.message}")
            // Xử lý lỗi nếu cần
        }
    }
}
```

Lọc danh bạ đơn vị

```
//Hàm lọc danh sách đơn vị
fun applyFilters(query: String = "") {
    // Lọc danh sách đơn vị theo tên hoặc mã
    val filteredList = _departmentList.value.filter { department ->
        // Kiểm tra xem tên hoặc mã đơn vị có chứa chuỗi tìm kiếm hay không
        val matchesQuery = department.name.contains(query, ignoreCase = true) ||
            department.id.contains(query, ignoreCase = true)
        // Kiểm tra xem loại đơn vị có khớp với chế độ lọc hay không
        val matchesFilter = when (_filterMode.value) {
            "Khoa" -> department.type == "Khoa"
            "Phòng" -> department.type == "Phòng"
            "Trung tâm" -> department.type == "Trung tâm"
            "Viện" -> department.type == "Viện"
            "Tất cả" -> true
            else -> true
        }
        // Trả về true nếu đơn vị khớp với cả hai điều kiện
        matchesQuery && matchesFilter
    }
    // Sắp xếp danh sách theo tên
    _filteredDepartmentList.value = filteredList
}
```

```

// Hàm này sẽ được gọi khi người dùng chọn một đơn vị
fun toggleSortOrder() {
    _sortAscending.value = !_sortAscending.value
}
// Bộ lọc danh sách đơn vị
fun setFilterMode(mode: String) {
    _filterMode.value = mode
    applyFilters()
}

// Lấy đúng sort đang áp dụng theo tab được chọn
val currentSortAscending = when (selectedTab) {
    "Sinh viên" -> studentSortAscending
    "Giảng viên" -> staffSortAscending
    "Đơn vị" -> departmentSortAscending
    else -> true
}

"Đơn vị" -> {
    DropdownMenuItem(onClick = {
        departmentViewModel?.setFilterMode("Tất cả")
        departmentViewModel?.applyFilters() // Áp dụng bộ lọc
        expandedFilter = false
        expanded = false
        onFilterClick()
    }) {
        Text("Tất cả")
    }

    DropdownMenuItem(onClick = {
        departmentViewModel?.setFilterMode("Khoa")
        departmentViewModel?.applyFilters() // Áp dụng bộ lọc
        expandedFilter = false
        expanded = false
        onFilterClick()
    }) {
        Text("Khoa")
    }
}

```

```
DropdownMenuItem(onClick = {
    departmentViewModel?.setFilterMode("Phòng")
    departmentViewModel?.applyFilters() // Áp dụng bộ lọc
    expandedFilter = false
    expanded = false
    onFilterClick()
}) {
    Text("Phòng")
}

DropdownMenuItem(onClick = {
    departmentViewModel?.setFilterMode("Trung tâm")
    departmentViewModel?.applyFilters() // Áp dụng bộ lọc
    expandedFilter = false
    expanded = false
    onFilterClick()
}) {
    Text("Trung tâm")
}

DropdownMenuItem(onClick = {
    departmentViewModel?.setFilterMode("Viện")
    departmentViewModel?.applyFilters() // Áp dụng bộ lọc
    expandedFilter = false
    expanded = false
    onFilterClick()
}) {
    Text("Viện")
}
}
```

Sắp xếp danh bạ đơn vị

```
// Menu xổ xuống (Dropdown) chính
DropdownMenu(
    expanded = expanded,
    onDismissRequest = { expanded = false },
    offset = dropdownOffset
) {
    // Đảo thứ tự sắp xếp tùy theo tab
    DropdownMenuItem(onClick = {
        when (selectedTab) {
            "Sinh viên" -> studentViewModel.toggleSortOrder()
            "Giảng viên" -> staffViewModel.toggleSortOrder() // Đảo thứ tự giảng viên
            "Đơn vị" -> departmentViewModel?.toggleSortOrder()
        }
        expanded = false // Đóng menu sau khi chọn
    }) {
        val label = when (selectedTab) { // Xác định nhãn cho nút sắp xếp
            "Sinh viên" -> if (studentSortAscending) "Sắp xếp Z-A" else "Sắp xếp A-Z"
            "Giảng viên" -> if (staffSortAscending) "Sắp xếp Z-A" else "Sắp xếp A-Z"
            "Đơn vị" -> if (departmentSortAscending) "Sắp xếp Z-A" else "Sắp xếp A-Z"
            else -> "Sắp xếp" // Giá trị mặc định nếu không có tab nào được chọn
        }
        Text(text = label) // Hiển thị chữ "Sắp xếp" với thứ tự tương ứng
    }
}
```

Chức năng gọi điện thoại và gửi email cho đơn vị

```
// Hiển thị các nút gọi điện thoại và gửi email
DepartmentActionButton(
    icon = Icons.Filled.Phone,
    label = "gọi",
    onClick = { openDialer(context, department.phone) }
)
Spacer(modifier = Modifier.width(32.dp))
DepartmentActionButton(
    icon = Icons.Filled.Email,
    label = "mail",
    onClick = { openEmail(context, department.email) }
)
```

```

// Hàm mở ứng dụng gọi điện thoại
fun openDialer(context: Context, phoneNumber: String) {
    val intent = Intent(Intent.ACTION_DIAL).apply {
        data = Uri.parse("tel:$phoneNumber")
    }
    context.startActivity(intent)
}

// Hàm mở ứng dụng gửi email
fun openEmail(context: Context, emailAddress: String) {
    val intent = Intent(Intent.ACTION_SENDTO).apply {
        data = Uri.parse("mailto:$emailAddress")
    }
    context.startActivity(intent)
}

```

Tìm kiếm đơn vị

```

// Hàm cập nhật danh sách khi query thay đổi
fun setQuery(query: String) {
    applyFilters(query)
}

// Ô nhập text tìm kiếm
BasicTextField( // BasicTextField cho phép tùy biến nhiều hơn
    value = query, // Nội dung ô tìm kiếm
    onValueChange = {
        onQueryChange(it)
        if (selectedTab == "Đơn vị") {
            departmentViewModel?.setQuery(it) // Gọi hàm setQuery khi query thay đổi
        }
    },
    //onValueChange = onQueryChange, // Cập nhật nội dung khi người dùng nhập
    modifier = Modifier.weight(1f), // Chiếm toàn bộ chiều rộng còn lại
    singleLine = true // Chỉ cho phép nhập một dòng
)

```

KẾT LUẬN

1. Kết quả đạt được

Phát triển hoàn thiện ứng dụng TLUContact – Ứng dụng danh bạ điện tử cho Đại học Thủy Lợi với đầy đủ các tính năng.

Hệ thống quản lý tài khoản hoàn chỉnh:

- Xây dựng hệ thống đăng nhập/đăng ký an toàn thông qua Firebase Authentication
- Phát triển chức năng khôi phục mật khẩu qua email với quy trình xác thực bảo mật
- Phân quyền người dùng rõ ràng giữa sinh viên và nhân viên, với các chức năng phù hợp cho từng đối tượng

Cơ sở dữ liệu thông tin đầy đủ:

- Thiết kế và triển khai cơ sở dữ liệu Firestore với cấu trúc tối ưu
- Lưu trữ thông tin sinh viên bao gồm mã SV, họ tên, lớp, email, số điện thoại và địa chỉ
- Hỗ trợ hiển thị ảnh đại diện cho mỗi người dùng

Giao diện người dùng hiện đại:

- Thiết kế UI theo nguyên tắc Material Design 3, tạo trải nghiệm người dùng thống nhất
- Sử dụng Jetpack Compose để xây dựng giao diện phản hồi nhanh và linh hoạt
- Tối ưu hóa hiển thị trên nhiều kích thước màn hình khác nhau

Tính năng kết nối đa dạng:

- Tích hợp liền mạch với các ứng dụng liên lạc của hệ thống
- Cho phép gọi điện thoại trực tiếp từ thông tin liên hệ

- Hỗ trợ gửi tin nhắn SMS với số điện thoại được điền sẵn
- Tích hợp chức năng gọi video call thông qua ứng dụng mặc định
- Cho phép soạn và gửi email trực tiếp từ ứng dụng

Quản lý danh bạ tiện lợi:

- Chức năng tìm kiếm thông minh theo tên và mã sinh viên
- Sắp xếp danh sách theo thứ tự tăng/giảm dần
- Lọc danh sách theo lớp hoặc theo bảng chữ cái
- Hiển thị danh sách có phân nhóm để dễ dàng theo dõi

Quản lý thông tin cá nhân:

- Cho phép người dùng xem và chỉnh sửa thông tin cá nhân
- Cập nhật thông tin liên lạc nhanh chóng và thuận tiện
- Xác thực dữ liệu nhập vào để đảm bảo tính chính xác

2. Nhược điểm

- Phần sửa thông tin cá nhân không sửa được Avatar và Email vì firebase cần trả phí và Email là Document ID
- Chưa có phần kết hợp tài khoản có cùng email lại do hạn chế về trình độ
- Phần màn hình xoay ngang, hiển thị box lọc vẫn chưa đúng với thiết kế giao diện figma.
- Thông báo hiển thị khi chỉnh sửa thông tin thành công vẫn chưa đúng với thiết kế giao diện.
- Ứng dụng web của nhóm vẫn chưa được hoàn chỉnh

3. Hướng phát triển

Nâng cao khả năng liên lạc trong hệ sinh thái TLUContact:

- Phát triển hệ thống nhắn tin nội bộ để sinh viên và giảng viên có thể liên lạc trực tiếp trong ứng dụng

- Xây dựng tính năng gọi điện thoại/video tích hợp sử dụng WebRTC, loại bỏ sự phụ thuộc vào ứng dụng bên thứ ba
- Tạo phòng chat nhóm theo lớp/khoa giúp trao đổi thông tin học tập hiệu quả

Mở rộng quy mô và tích hợp hệ thống:

- Tích hợp với hệ thống quản lý đào tạo của trường để đồng bộ thông tin sinh viên, lịch học, điểm số
- Phát triển API để các ứng dụng khác trong hệ sinh thái của trường có thể kết nối và trao đổi dữ liệu
- Mở rộng đối tượng người dùng bao gồm cựu sinh viên, nhà tuyển dụng để tạo mạng lưới kết nối toàn diện

Cải thiện trải nghiệm người dùng:

- Phát triển phiên bản web để truy cập trên nhiều thiết bị khác nhau
- Triển khai tính năng cá nhân hóa giao diện người dùng (theme, chế độ tối/sáng)
- Bổ sung các widget màn hình chính để truy cập nhanh các liên hệ thường xuyên sử dụng

Nâng cao bảo mật và quyền riêng tư:

- Triển khai hệ thống phân quyền chi tiết, cho phép người dùng kiểm soát thông tin hiển thị
- Tích hợp xác thực hai lớp để tăng cường bảo mật tài khoản
- Phát triển tính năng mã hóa đầu cuối cho các tin nhắn trong ứng dụng

Ứng dụng công nghệ tiên tiến:

- Tích hợp AI để đề xuất liên hệ thông minh dựa trên lịch sử tương tác
- Xây dựng trợ lý ảo hỗ trợ sinh viên tìm kiếm thông tin liên hệ bằng giọng nói

- Phát triển tính năng nhận diện khuôn mặt để đơn giản hóa việc tìm kiếm và cập nhật thông tin

Kế hoạch phát triển dài hạn:

- Mở rộng thành nền tảng kết nối cho toàn bộ hệ thống giáo dục đại học Việt Nam
- Xây dựng hệ sinh thái ứng dụng xung quanh nền tảng cốt lõi đáp ứng nhu cầu đa dạng của sinh viên, giảng viên
- Phát triển các giải pháp phân tích dữ liệu để nâng cao hiệu quả giao tiếp và hợp tác trong môi trường học thuật

TÀI LIỆU THAM KHẢO

- [1] [Add Firebase to your Android project | Firebase for Android](#)
- [2] [Add Firebase to your JavaScript project | Firebase for Web platforms](#)
- [3] [Đôi nét về Microservice architecture và Monolithic architecture](#)
- [4] [Microservices vs. monolithic architecture | Atlassian](#)