

**TỔNG LIÊN ĐOÀN LAO ĐỘNG VIỆT NAM
TRƯỜNG ĐẠI HỌC TÔN ĐỨC THẮNG
KHOA CÔNG NGHỆ THÔNG TIN**



BÁO CÁO MÔN PHÁT TRIỂN ỨNG DỤNG DI ĐỘNG

**NGHIÊN CỨU TOÀN DIỆN VỀ GOOGLE
FIREBASE FIRESTORE ĐỂ PHÁT
TRIỂN ỨNG DỤNG QUẢN LÝ THÔNG
TIN SINH VIÊN THEO THỜI GIAN
THỰC**

Người hướng dẫn: **Th.S VŨ ĐÌNH HỒNG**

Người thực hiện: **Nguyễn Thị Bảo Trân-52200089**

**THÀNH PHỐ HỒ CHÍ MINH, NĂM 2020
TỔNG LIÊN ĐOÀN LAO ĐỘNG VIỆT NAM
TRƯỜNG ĐẠI HỌC TÔN ĐỨC THẮNG
KHOA CÔNG NGHỆ THÔNG TIN**



BÁO CÁO MÔN PHÁT TRIỂN ỨNG DỤNG DI ĐỘNG

**NGHIÊN CỨU TOÀN DIỆN VỀ GOOGLE
FIREBASE FIRESTORE ĐỂ PHÁT
TRIỂN ỨNG DỤNG QUẢN LÝ THÔNG
TIN SINH VIÊN THEO THỜI GIAN
THỰC**

Người hướng dẫn: Th.S NGUYỄN TRỌNG NHÂN

Người thực hiện: NGUYỄN THỊ BẢO TRÂN-52200089

THÀNH PHỐ HỒ CHÍ MINH, NĂM 2020

LỜI CẢM ƠN

Xin chân thành cảm ơn quý thầy/cô đã dành thời gian đọc và đánh giá tiểu luận của em. Em xin gửi lời cảm ơn chân thành vì sự hướng dẫn và hỗ trợ của quý thầy/cô trong quá trình nghiên cứu và thực hiện đề tài.

Em cũng muốn bày tỏ sự biết ơn của mình đến tất cả những nguồn tài liệu, nghiên cứu đã được sử dụng và tham khảo trong tiểu luận này. Đây là nguồn cung cấp kiến thức quý giá giúp em hoàn thiện nội dung và mang lại tính khoa học cho tiểu luận.

Cuối cùng, em xin gửi lời cảm ơn chân thành đến tổ chức giáo dục nhà trường. Nhờ có đề tài này mà em đã có cơ hội được mở mang, nghiên cứu, tìm hiểu về một vấn đề xã hội bổ ích này.

Xin chân thành cảm ơn mọi người đã đồng hành cùng em trong quá trình thực hiện tiểu luận này.

ĐỒ ÁN ĐƯỢC HOÀN THÀNH TẠI TRƯỜNG ĐẠI HỌC TÔN ĐỨC THẮNG

Chúng tôi xin cam đoan đây là sản phẩm đồ án của riêng chúng tôi và được sự hướng dẫn của thầy Nguyễn Trọng Nhân. Các nội dung nghiên cứu, kết quả trong đề tài này là trung thực và chưa công bố dưới bất kỳ hình thức nào trước đây. Những số liệu trong các bảng biểu phục vụ cho việc phân tích, nhận xét, đánh giá được chính tác giả thu thập từ các nguồn khác nhau có ghi rõ trong phần tài liệu tham khảo.

Ngoài ra, trong đồ án còn sử dụng một số nhận xét, đánh giá cũng như số liệu của các tác giả khác, cơ quan tổ chức khác đều có trích dẫn và chú thích nguồn gốc.

Nếu phát hiện có bất kỳ sự gian lận nào chúng tôi xin hoàn toàn chịu trách nhiệm về nội dung đồ án của mình. Trường đại học Tôn Đức Thắng không liên quan đến những vi phạm tác quyền, bản quyền do tôi gây ra trong quá trình thực hiện (nếu có).

TP. Hồ Chí Minh, ngày 03 tháng 11 năm 2024

Tác giả

(ký tên và ghi rõ họ tên)

Nguyễn Thị Bảo Trân

PHẦN XÁC NHẬN VÀ ĐÁNH GIÁ CỦA GIẢNG VIÊN

Phần xác nhận của GV hướng dẫn

Tp. Hồ Chí Minh, ngày tháng năm
(kí và ghi họ tên)

Phần đánh giá của GV chấm bài

Tp. Hồ Chí Minh, ngày tháng năm
(kí và ghi họ tên)

TÓM TẮT

Trong bối cảnh giáo dục hiện nay, việc quản lý thông tin sinh viên một cách hiệu quả và kịp thời đang trở thành một thách thức đối với các trường đại học và tổ chức giáo dục. Các phương pháp quản lý truyền thống, như sử dụng bảng tính hay giấy tờ, không còn đáp ứng được yêu cầu về sự linh hoạt và khả năng cập nhật tức thời khi thông tin sinh viên ngày càng phức tạp và yêu cầu quản lý ngày càng khắt khe hơn. Việc sử dụng các hệ thống quản lý thông tin hiện đại, với khả năng đồng bộ dữ liệu theo thời gian thực, đã trở thành một nhu cầu cấp bách.

Google Firebase Firestore là một trong những nền tảng cơ sở dữ liệu đám mây hiện đại được thiết kế để hỗ trợ các ứng dụng cần lưu trữ và truy xuất dữ liệu theo thời gian thực. Khác với các hệ thống lưu trữ truyền thống, Firestore cung cấp khả năng quản lý dữ liệu một cách linh hoạt, cho phép người dùng truy cập, cập nhật và đồng bộ dữ liệu một cách tức thời, phù hợp với những yêu cầu của hệ thống quản lý thông tin sinh viên. Các đặc điểm nổi bật của Firestore, như khả năng mở rộng, bảo mật dữ liệu cao, và đặc biệt là hỗ trợ tính năng thời gian thực, khiến nó trở thành một lựa chọn lý tưởng cho việc xây dựng các ứng dụng quản lý thông tin sinh viên.

Mục tiêu chính của bài tiểu luận này là nghiên cứu toàn diện về Google Firebase Firestore, bao gồm cấu trúc, tính năng và lợi ích của nó trong việc phát triển một ứng dụng quản lý thông tin sinh viên theo thời gian thực. Cụ thể, bài viết sẽ tập trung phân tích cách thức mà Firestore có thể được ứng dụng để quản lý các thông tin như hồ sơ sinh viên, điểm số, lịch học, và các thông báo liên quan. Qua đó, bài tiểu luận hy vọng sẽ giúp người đọc hiểu rõ hơn về lợi ích của việc sử dụng cơ sở dữ liệu thời gian thực trong môi trường giáo dục và tiềm năng của nó trong việc cải thiện quy trình quản lý thông tin.

MỤC LỤC

LỜI CẢM ƠN	i
PHẦN XÁC NHẬN VÀ ĐÁNH GIÁ CỦA GIẢNG VIÊN	iii
TÓM TẮT	iv
MỤC LỤC	1
DANH MỤC CÁC BẢNG BIỂU, HÌNH VẼ, ĐỒ THỊ	1
DANH MỤC HÌNH	1
DANH MỤC BẢNG	3
CHƯƠNG 1 – MỞ ĐẦU	1
1.1 Trình bày lý do chọn đề tài và tầm quan trọng của việc phát triển ứng dụng quản lý thông tin sinh viên theo thời gian thực.	1
1.2 Mục tiêu nghiên cứu	1
1.3 Phạm vi nghiên cứu	2
CHƯƠNG 2 – GIỚI THIỆU CHUNG	1
2.1 Tổng quan về Google Firebase	1
2.2 Giới thiệu về Firebase Firestore	2
2.3 Các tính năng chính của Firebase	2
2.3.1 Realtime Database	2
2.3.2 Authentication	3
2.3.3 Cloud Storage	3
2.3.4 Cloud Firestore	3
2.3.5 Hosting	3
2.3.6 Cloud Functions	4
2.3.7 Analytics	4

2.3.8 Firebase Crashlytics	4
2.3.9 Firebase Performance Monitoring	4
2.3.10 Firebase A/B Testing	5
2.3.11 Firebase Remote Config	5
2.4 Phân tích kỹ thuật và tính năng của Firebase Firestore (Technical Analysis and Features of Firebase Firestore)	5
2.4.1 Cập nhật theo thời gian thực	5
2.4.2 Hỗ trợ ngoại tuyến	6
2.4.3 Quy tắc bảo mật	6
2.4.4 Khả năng mở rộng và hiệu suất	6
2.5 So sánh Firebase Firestore và Realtime Database	7
2.3.12 Kiến trúc cơ sở dữ liệu	7
2.3.13 Khả năng mở rộng và hiệu suất	7
2.3.14 Tính năng đồng bộ hóa và hỗ trợ hoạt động ngoại	8
2.3.15 Khả năng truy vấn và xử lý dữ liệu	8
2.3.16 Đối tượng sử dụng phù hợp	9
2.3.17 Ưu và nhược điểm của mỗi loại cơ sở dữ liệu	9
2.6 So sánh Firebase Firestore với các cơ sở dữ liệu truyền thống như MySQL	10
2.6.1 Kiến trúc dữ liệu	10
2.6.2 Khả năng mở rộng và hiệu suất	10
2.6.3 Khả năng đồng bộ hóa dữ liệu theo thời gian	10
2.6.4 Tính linh hoạt của mô hình dữ liệu	11
2.6.5 Truy vấn và xử lý dữ liệu	11
2.6.6 Chi phí triển khai và quản lý	12
2.6.7 Trường hợp sử dụng phù hợp	12
2.6.8 Ưu và nhược điểm của mỗi loại cơ sở dữ liệu	13

CHƯƠNG 3 – THIẾT KẾ VÀ XÂY DỰNG ỨNG DỤNG QUẢN LÝ THÔNG TIN	
SINH VIÊN	1
3.1 Phân tích yêu cầu ứng dụng	1
3.1.1 Yêu cầu chức năng	1
3.1.2 Yêu cầu phi chức năng	2
3.2 Thiết kế kiến trúc hệ thống	2
3.2.1 Mô hình tổng quan	2
3.2.2 Thiết kế cơ sở dữ liệu	3
3.3 Xây dựng hệ thống và triển khai	5
3.3.1 Thiết kế giao diện người dùng	5
3.3.1.1 Sơ đồ màn hình (Screen Flow Diagram)	5
3.3.1.2 Wireframes cho ứng dụng	7
3.3.2 Các thành phần giao diện chính	10
3.3.3 Link video thuyết trình + demo : Click here	34
3.3.4 Link source code gitlab : Click here	34
TÀI LIỆU THAM KHẢO	35

DANH MỤC CÁC BẢNG BIỂU, HÌNH VẼ, ĐỒ THỊ

DANH MỤC HÌNH

Hình 1	Sơ đồ màn hình (Screen Flow Diagram)	6
Hình 2	Màn hình đăng nhập	10
Hình 3	Màn hình quên mật khẩu	11
Hình 4	Màn hình chọn dữ liệu quản lý	12
Hình 5	Màn hình quản lý tài khoản	13
Hình 6	Màn hình thêm tài khoản	14
Hình 7	Màn hình cập nhật tài khoản	15
Hình 8	Màn hình xóa tài khoản	16
Hình 9	Màn hình xem lịch sử hoạt động	17
Hình 10	Màn hình quản lý tín chỉ	18
Hình 11	Màn hình thêm tín chỉ	19
Hình 12	Màn hình sửa tín chỉ	20
Hình 13	Màn hình xóa tín chỉ	21
Hình 14	Màn hình thêm / xuất dữ liệu	22
Hình 15	Màn hình quản lý thông tin sinh viên	23
Hình 16	: Màn hình thêm sinh viên	24
Hình 17	: Màn hình cập nhật thông tin sinh viên	26
Hình 18	: Xóa một sinh viên	27
Hình 19	: Sắp xếp sinh viên	28
Hình 20	: Xóa nhiều sinh viên	29
Hình 21	: Nhập xuất sinh viên từ file excel/csv	30
Hình 22	: Xem chi tiết sinh viên	31
Hình 23	: Trang thông tin cá nhân	32

Hình 24 : Sửa thông tin cá nhân	33
---------------------------------------	----

DANH MỤC BẢNG

CHƯƠNG 1 – MỞ ĐẦU

1.1 Trình bày lý do chọn đề tài và tầm quan trọng của việc phát triển ứng dụng quản lý thông tin sinh viên theo thời gian thực.

Google Firebase Firestore là một cơ sở dữ liệu NoSQL mạnh mẽ và được biết đến rộng rãi nhờ khả năng đồng bộ hóa dữ liệu thời gian thực giữa các thiết bị, khả năng mở rộng quy mô và khả năng tích hợp với các công cụ khác của Google như Firebase Authentication, Firebase Hosting, và Firebase Cloud Functions. Trong bối cảnh nhu cầu quản lý thông tin sinh viên ngày càng tăng, đặc biệt trong môi trường học tập số hóa và phân tán, việc xây dựng một ứng dụng quản lý thông tin sinh viên theo thời gian thực là một giải pháp phù hợp và hiệu quả.

Ứng dụng này sẽ giúp các trường học, giáo viên và sinh viên quản lý và truy cập thông tin học tập một cách nhanh chóng và thuận tiện. Firebase Firestore cung cấp nền tảng mạnh mẽ cho các ứng dụng này nhờ vào khả năng cập nhật thời gian thực, tự động đồng bộ hóa và hỗ trợ thiết kế các giải pháp ứng dụng đơn giản và hiệu quả, tiết kiệm thời gian và chi phí so với việc sử dụng các hệ thống quản lý truyền thống.

1.2 Mục tiêu nghiên cứu

Xây dựng và phát triển ứng dụng quản lý thông tin sinh viên theo thời gian thực sử dụng Firebase Firestore, hỗ trợ các tính năng như cập nhật thông tin sinh viên, thông tin chứng chỉ, quản lý lớp học.

Nghiên cứu khả năng đồng bộ hóa dữ liệu và hoạt động offline của Firebase Firestore trong bối cảnh ứng dụng quản lý thông tin sinh viên, từ đó tối ưu hóa trải nghiệm người dùng khi có kết nối mạng không ổn định.

Đánh giá hiệu suất và khả năng mở rộng quy mô của Firebase Firestore trong việc xử lý dữ liệu lớn và đáp ứng nhu cầu của các trường học có số lượng sinh viên lớn.

Phân tích tính bảo mật của Firebase Firestore và áp dụng các biện pháp bảo mật phù hợp nhằm đảm bảo an toàn dữ liệu cá nhân và thông tin nhạy cảm của sinh viên.

Xây dựng tính năng xác thực người dùng bằng Firebase Authentication để đảm bảo chỉ những người dùng có quyền mới có thể truy cập và cập nhật thông tin của hệ thống.

So sánh Firebase Firestore với các cơ sở dữ liệu khác về khả năng đồng bộ hóa dữ liệu, hiệu suất, và bảo mật nhằm xác định mức độ phù hợp của Firestore với ứng dụng quản lý thông tin sinh viên.

Xác định các thách thức và giải pháp trong việc triển khai ứng dụng quản lý thông tin sinh viên sử dụng Firebase Firestore trong môi trường giáo dục, bao gồm tích hợp với các hệ thống hiện có và tối ưu hóa trải nghiệm người dùng. Phạm vi nghiên cứu

1.3 Phạm vi nghiên cứu

Bài tiểu luận này tập trung nghiên cứu về việc ứng dụng Google Firebase Firestore trong việc phát triển một ứng dụng quản lý thông tin sinh viên theo thời gian thực. Phạm vi nghiên cứu bao gồm việc phân tích kiến trúc của Firestore, các tính năng hỗ trợ dữ liệu thời gian thực, và khả năng tích hợp của nó trong xây dựng một hệ thống quản lý thông tin sinh viên.

Cụ thể, nghiên cứu sẽ tập trung vào các yếu tố sau:

Firestore và tính năng thời gian thực: Nghiên cứu cách Firestore tổ chức và quản lý dữ liệu dưới dạng NoSQL, bao gồm **Collections**, **Documents**, **Fields**, và **Subcollections**. Nghiên cứu sẽ phân tích chi tiết về tính năng đồng bộ hóa thời gian thực và cách chúng hỗ trợ cho quản lý thông tin sinh viên.

Ứng dụng vào quản lý thông tin sinh viên: Phạm vi nghiên cứu sẽ bao gồm các khía cạnh như quản lý điểm số, thông tin cá nhân, lịch học, và thông báo của

sinh viên. Mục tiêu là chứng minh cách Firestore có thể cải thiện quy trình quản lý thông tin này một cách hiệu quả hơn so với các hệ thống truyền thống.

Bảo mật và phân quyền: Nghiên cứu cũng sẽ đề cập đến các phương thức bảo mật dữ liệu của Firestore, bao gồm **Firebase Security Rules** và cách phân quyền để đảm bảo dữ liệu sinh viên được an toàn trong quá trình truy cập và đồng bộ.

Không đi sâu vào các nền tảng cơ sở dữ liệu khác: Bài tiểu luận này sẽ không đi sâu vào các hệ thống cơ sở dữ liệu khác ngoài Firebase Firestore. Mặc dù có thể có một số so sánh với các hệ thống truyền thống để làm nổi bật ưu điểm của Firestore, nhưng mục tiêu chính vẫn là cung cấp cái nhìn sâu sắc về Firestore và ứng dụng của nó trong quản lý thông tin sinh viên.

Phạm vi về kỹ thuật: Các khía cạnh kỹ thuật được nghiên cứu sẽ tập trung vào việc tích hợp Firestore vào các ứng dụng di động hoặc web để cung cấp giải pháp quản lý thời gian thực. Việc lập trình, xây dựng giao diện người dùng và tích hợp các tính năng thời gian thực sẽ là trọng tâm của nghiên cứu.

CHƯƠNG 2 – GIỚI THIỆU CHUNG

2.1 Tổng quan về Google Firebase

Firebase là nền tảng phát triển ứng dụng toàn diện của Google, cung cấp nhiều tính năng và thành phần để phát triển các ứng dụng web hoặc di động. Hiện nay, Firebase được nhiều người sử dụng do tính hữu dụng mà nó mang lại. Trong Firebase có hai cơ sở dữ liệu khác nhau là Realtime Database và Cloud Firestore. Nó cung cấp lưu trữ các file bảo mật cho người dùng và cho phép sử dụng Google Cloud Storage. Đồng thời, dịch vụ cũng sử dụng các tính năng xác thực và bảo mật nâng cao nhằm mục đích kiểm soát quyền điều khiển và hạn chế các mối đe dọa.

Firebase cung cấp một hệ sinh thái đa dạng bao gồm các dịch vụ chính như:

- **Firebase Authentication:** Giúp quản lý xác thực người dùng, hỗ trợ các phương thức đăng nhập phổ biến như Google, Facebook, Email và Số điện thoại.
- **Firebase Hosting:** Cung cấp dịch vụ lưu trữ và phân phối nội dung tĩnh cho các ứng dụng web, hỗ trợ tốc độ và bảo mật tốt.
- **Firebase Cloud Functions:** Cho phép viết mã chạy trên đám mây khi có sự kiện từ các dịch vụ khác của Firebase, giúp mở rộng tính năng mà không cần quản lý máy chủ.
- **Firebase Firestore:** Một trong những cơ sở dữ liệu chính của Firebase, hỗ trợ lưu trữ dữ liệu theo thời gian thực với khả năng mở rộng và tích hợp cao.

2.2 Giới thiệu về Firebase Firestore

Firebase Firestore là một cơ sở dữ liệu NoSQL, nghĩa là nó lưu trữ dữ liệu dưới dạng các tài liệu thay vì bảng như SQL. Cấu trúc này giúp Firestore dễ dàng mở rộng và hỗ trợ dữ liệu linh hoạt, phù hợp với các ứng dụng có yêu cầu cao về tốc độ và khả năng thay đổi dữ liệu liên tục. Được xây dựng trên nền tảng đám mây, chuyên hỗ trợ cho các ứng dụng có nhu cầu đồng bộ dữ liệu theo thời gian thực. Khác với **Firebase Realtime Database**, Firestore được thiết kế với cấu trúc dữ liệu linh hoạt và mạnh mẽ hơn, hỗ trợ truy vấn nâng cao, và tích hợp tốt với các dịch vụ khác của Firebase.

Firestore sử dụng cấu trúc **NoSQL** dạng tài liệu, với các thành phần chính như:

- **Collections:** Là tập hợp các tài liệu cùng loại, chứa các nhóm dữ liệu liên quan.
- **Documents:** Là đơn vị lưu trữ dữ liệu chính, nằm trong Collections, và chứa các thông tin chi tiết về sinh viên hoặc đối tượng khác.
- **Fields:** Là các giá trị cụ thể trong mỗi tài liệu, lưu giữ thông tin như tên sinh viên, điểm số, lớp học.
- **Subcollections:** Là các nhóm con của một tài liệu, cho phép lưu trữ thông tin chi tiết và phân cấp.

2.3 Các tính năng chính của Firebase

2.3.1 *Realtime Database*

Realtime Database là một cơ sở dữ liệu thời gian thực. Ngay sau khi bạn đăng ký tài khoản trên Firebase, bạn sẽ nhận được Realtime Database được lưu trữ dưới dạng JSON và được đồng bộ hóa theo thời gian thực đối với mọi kết nối.

Đối với các ứng dụng được xây dựng trên đa nền tảng như Android, IOS và WebApp, tất cả client sẽ cùng sử dụng một cơ sở dữ liệu. Bên cạnh đó, hệ

thông dữ liệu này sẽ tự động cập nhật khi lập trình viên phát triển ứng dụng. Sau đó, tất cả dữ liệu này sẽ được truyền tải thông qua các kết nối SSL có 2048 bit.

2.3.2 Authentication

Authentication là tính năng giúp xác thực danh tính của người dùng ứng dụng. Firebase cung cấp các bước xác thực thông qua Email, Facebook, Twitter, GitHub hay Google. Điều này giúp cho các thông tin cá nhân của user được bảo vệ một cách tốt nhất, hạn chế được tình trạng bị hacker đánh cắp. Đồng thời việc xác thực danh tính qua Firebase sẽ giúp người dùng tiếp cận sản phẩm nhanh chóng và an toàn hơn.

2.3.3 Cloud Storage

Cloud Storage là tính năng cho phép lưu trữ và quản lý nội dung đã tạo ra như ảnh, video, nội dung, văn bản,... Firebase Storage cung cấp các API hỗ trợ bạn upload và download các file từ ứng dụng một cách trơn tru mà không cần quan tâm đến chất lượng đường truyền mạng với độ bảo mật cao.

2.3.4 Cloud Firestore

Cloud Firestore được phát triển từ tính năng Realtime Database. Trải qua nhiều lần nâng cấp, Cloud Firestore có giao diện trực quan và khả năng mở rộng ưu việt hơn so với Realtime Database. Tính năng này của Firebase giúp đồng bộ mọi dữ liệu trên các ứng dụng thông qua việc đăng ký thời gian thực và cung cấp hỗ trợ ngoại tuyến cho thiết bị di động cũng như website.

2.3.5 Hosting

Hosting được phân phối thông qua tiêu chuẩn công nghệ bảo mật SSL từ hệ thống mạng CDN. CDN là một mạng lưới máy chủ giúp lưu trữ các bản sao của các nội dung tĩnh trên website. Thông qua CDN, người

dùng có thể truy cập và sử dụng các dịch vụ trên web khi cài Firebase Hosting một cách nhanh chóng và ổn định hơn.

2.3.6 Cloud Functions

Cloud Functions giúp chạy code backend tự động nhằm phản hồi các sự kiện được kích hoạt bởi tính năng của Firebase và HTTPS request. Cloud Functions có tính bảo mật cao và luôn thực hiện chính xác những gì mà bạn muốn.

2.3.7 Analytics

Analytics giúp bạn có thể phân tích hành vi của người sử dụng ứng dụng của bạn. Qua đó, bạn sẽ biết được khách hàng thường xuyên truy cập tính năng nào và các thông tin về hiệu quả quảng cáo, tình trạng trả phí,... để có thể đưa ra được chiến lược phát triển phù hợp. Để thực hiện tính năng Analytics của Firebase, bạn cần cài đặt Software Development Kit (SDK).

2.3.8 Firebase Crashlytics

Crashlytics là dịch vụ giám sát và báo cáo lỗi giúp lập trình viên theo dõi và xử lý các lỗi trong ứng dụng. Nó cung cấp các báo cáo chi tiết về các sự cố xảy ra, giúp lập trình viên có thể phát hiện và sửa lỗi một cách nhanh chóng, từ đó nâng cao trải nghiệm người dùng và duy trì sự ổn định của ứng dụng.

2.3.9 Firebase Performance Monitoring

Performance Monitoring là công cụ giúp giám sát hiệu suất ứng dụng, cung cấp các thông tin về hiệu suất mạng, thời gian xử lý các tác vụ quan trọng, v.v. Điều này giúp lập trình viên dễ dàng phát hiện và tối ưu hóa những vấn đề ảnh hưởng đến hiệu suất của ứng dụng, từ đó nâng cao tốc độ và chất lượng trải nghiệm cho người dùng.

2.3.10 Firebase A/B Testing

A/B Testing là tính năng giúp lập trình viên tiến hành các thử nghiệm A/B nhằm kiểm tra các thay đổi trên ứng dụng và xem ảnh hưởng của chúng đối với hành vi người dùng. Từ đó có thể quyết định triển khai những thay đổi phù hợp nhất, giúp tăng tỷ lệ chuyển đổi và cải thiện trải nghiệm người dùng.

2.3.11 Firebase Remote Config

Remote Config cho phép cập nhật giao diện hoặc các hành vi của ứng dụng mà không cần phát hành bản cập nhật mới. Nó giúp lập trình viên có thể dễ dàng kiểm soát và điều chỉnh các thông số của ứng dụng từ xa, giúp thay đổi nội dung, trải nghiệm mà không cần gửi lại ứng dụng cho người dùng.

2.4 Phân tích kỹ thuật và tính năng của Firebase Firestore (Technical Analysis and Features of Firebase Firestore)

2.4.1 Cập nhật theo thời gian thực

Giải thích cơ chế: Firebase Firestore sử dụng **realtime listeners** để theo dõi và tự động cập nhật khi có sự thay đổi trong cơ sở dữ liệu. Mỗi khi dữ liệu thay đổi, Firestore đảm bảo cập nhật tức thời trên tất cả các thiết bị đang kết nối. Điều này rất hữu ích khi cần duy trì thông tin đồng nhất giữa các sinh viên, giáo viên và nhà trường trong thời gian thực.

Ứng dụng thực tiễn: Điều này giúp nâng cao tính hiệu quả và minh bạch trong việc quản lý thông tin sinh viên, như cập nhật điểm số, lịch học, và thông báo mới.

2.4.2 Hỗ trợ ngoại tuyến

Khả năng làm việc ngoại tuyến: Firebase Firestore cho phép người dùng truy cập và chỉnh sửa dữ liệu ngay cả khi không có kết nối internet. **Local cache** lưu trữ các thay đổi và sẽ tự động đồng bộ hóa khi có kết nối trở lại, giúp đảm bảo không có thông tin nào bị mất mát.

Lợi ích cho giáo dục: Tính năng này đặc biệt quan trọng khi sinh viên hoặc giáo viên cần truy cập thông tin trong môi trường có kết nối mạng yếu hoặc không ổn định, giúp duy trì sự mượt mà trong quá trình học tập và giảng dạy.

2.4.3 Quy tắc bảo mật

Cơ chế bảo mật: Firebase Firestore sử dụng **Firebase Security Rules** để quản lý quyền đọc và ghi của người dùng đối với cơ sở dữ liệu. Những quy tắc này cho phép quản lý chi tiết quyền truy cập, đảm bảo chỉ những người dùng đã xác thực có thể truy cập dữ liệu phù hợp với quyền hạn của họ.

Xác thực người dùng: Firestore tích hợp với **Firebase Authentication** để xác thực người dùng, cho phép phân quyền và đảm bảo chỉ người dùng được phép mới có thể truy cập và chỉnh sửa dữ liệu. Điều này giúp bảo mật thông tin nhạy cảm, chẳng hạn như thông tin cá nhân của sinh viên hoặc kết quả học tập.

2.4.4 Khả năng mở rộng và hiệu suất

Khả năng mở rộng: Firebase Firestore hỗ trợ **mở rộng theo chiều ngang** (horizontal scaling), tức là có thể phân bổ dữ liệu qua nhiều máy chủ để đáp ứng lượng người dùng lớn mà không ảnh hưởng đến hiệu suất. Điều này giúp hệ thống duy trì được khả năng phản hồi tốt ngay cả khi số lượng sinh viên và giáo viên tăng lên.

Hiệu suất: Firestore tự động tạo **chỉ mục** cho các trường dữ liệu, giúp tăng tốc độ truy vấn. Đối với các yêu cầu truy vấn phức tạp hoặc xử lý nhiều dữ liệu, việc sử dụng **chỉ mục tùy chỉnh** sẽ giúp tối ưu hóa hiệu suất và đảm bảo ứng dụng hoạt động trơn tru ngay cả khi số lượng yêu cầu cao.

2.5 So sánh Firebase Firestore và Realtime Database

Firebase cung cấp hai dịch vụ cơ sở dữ liệu thời gian thực là **Realtime Database** và **Firestore**. Mặc dù cả hai đều được thiết kế để lưu trữ dữ liệu NoSQL và hỗ trợ đồng bộ hóa theo thời gian thực, nhưng chúng có những điểm khác biệt quan trọng về cấu trúc, tính năng, và khả năng mở rộng. Mục này sẽ phân tích sự khác biệt giữa Firestore và Realtime Database để giúp chọn lựa công nghệ phù hợp với từng trường hợp sử dụng.

2.3.12 Kiến trúc cơ sở dữ liệu

Realtime Database: Sử dụng mô hình cơ sở dữ liệu **NoSQL dạng cây** (tree structure), dữ liệu được lưu trữ dưới dạng **JSON**. Điều này phù hợp cho các ứng dụng có cấu trúc dữ liệu đơn giản, nhưng khi dữ liệu trở nên phức tạp, việc quản lý và duy trì cấu trúc này có thể gây khó khăn.

Firestore: Sử dụng mô hình cơ sở dữ liệu **NoSQL dạng tài liệu** (document model), với cấu trúc **Collections** chứa các **Documents**. Mỗi tài liệu có thể chứa **Fields** và **Subcollections**. Cấu trúc này giúp dữ liệu được tổ chức rõ ràng hơn và dễ dàng mở rộng khi ứng dụng phát triển.

2.3.13 Khả năng mở rộng và hiệu suất

Realtime Database: Hỗ trợ quy mô **nhỏ và trung bình**, nhưng khi số lượng người dùng và dữ liệu tăng lên, hiệu suất của Realtime Database có thể bị ảnh

hưởng. Do cấu trúc dữ liệu dạng cây, các truy vấn phức tạp có thể gây ra tình trạng giảm hiệu suất khi dữ liệu lớn.

Firestore: Được thiết kế với khả năng **mở rộng tốt hơn**, giúp dễ dàng phục vụ các ứng dụng lớn với số lượng người dùng tăng cao. Firestore hỗ trợ khả năng xử lý các truy vấn phức tạp và tối ưu hóa cho việc mở rộng quy mô với tính năng **tự động tạo chỉ mục (automatic indexing)**.

2.3.14 Tính năng đồng bộ hóa và hỗ trợ hoạt động ngoại

Realtime Database: Hỗ trợ **đồng bộ hóa thời gian thực** giữa các thiết bị và người dùng, cho phép dữ liệu được cập nhật ngay lập tức khi có thay đổi. Tuy nhiên, tính năng **hoạt động offline** chỉ được hỗ trợ cho các ứng dụng di động, và không hỗ trợ tốt cho các ứng dụng web.

Firestore: Ngoài tính năng **đồng bộ hóa thời gian thực**, Firestore còn cung cấp **hỗ trợ hoạt động offline** cho cả ứng dụng di động và ứng dụng web. Người dùng có thể truy cập và cập nhật dữ liệu ngay cả khi mất kết nối mạng, và dữ liệu sẽ tự động được đồng bộ hóa khi có kết nối trở lại.

2.3.15 Khả năng truy vấn và xử lý dữ liệu

Realtime Database: Cung cấp **khả năng truy vấn cơ bản**. Việc truy vấn dữ liệu từ Realtime Database có thể trở nên phức tạp và ít linh hoạt khi cần thực hiện các truy vấn sâu hoặc lọc dữ liệu với nhiều điều kiện.

Firestore: Firestore cho phép **truy vấn phức tạp hơn**, hỗ trợ các truy vấn với nhiều điều kiện, sắp xếp, và phân trang. Khả năng truy vấn mạnh mẽ của Firestore giúp việc quản lý và lấy dữ liệu từ cơ sở dữ liệu trở nên linh hoạt hơn.

2.3.16 Đối tượng sử dụng phù hợp

Realtime Database: Phù hợp với các **ứng dụng nhỏ**, có yêu cầu **đồng bộ hóa đơn giản**, ví dụ như ứng dụng chat cơ bản, quản lý dữ liệu nhỏ theo thời gian thực, hoặc khi cần một giải pháp nhanh và dễ triển khai.

Firestore: Phù hợp với các ứng dụng **lớn hơn**, có cấu trúc dữ liệu **phức tạp**, và yêu cầu **khả năng mở rộng**. Firestore đặc biệt phù hợp với các hệ thống quản lý thông tin phức tạp, như **quản lý thông tin sinh viên**, nơi cần truy vấn linh hoạt và quản lý dữ liệu theo nhiều cấp độ.

2.3.17 Ưu và nhược điểm của mỗi loại cơ sở dữ liệu

- **Realtime Database:**

Ưu điểm: Đồng bộ hóa tức thì, dễ triển khai, phù hợp với các ứng dụng có cấu trúc đơn giản và ít người dùng.

Nhược điểm: Khả năng mở rộng hạn chế, khó quản lý khi cấu trúc dữ liệu phức tạp, thiếu hỗ trợ tốt cho hoạt động offline trên web.

- **Firestore:**

Ưu điểm: Kiến trúc linh hoạt, hỗ trợ truy vấn phức tạp, mở rộng quy mô tốt, hỗ trợ hoạt động offline cho cả di động và web.

Nhược điểm: Chi phí có thể tăng cao khi có nhiều hoạt động đọc và ghi, yêu cầu quản lý chặt chẽ để tối ưu chi phí.

2.6 So sánh Firebase Firestore với các cơ sở dữ liệu truyền thống như MySQL

2.6.1 Kiến trúc dữ liệu

Firebase Firestore: Là một cơ sở dữ liệu **NoSQL** với cấu trúc **document**. Dữ liệu được lưu trữ dưới dạng **Collections** và **Documents**, rất linh hoạt cho các ứng dụng cần thay đổi cấu trúc dữ liệu thường xuyên. Kiến trúc này cho phép lưu trữ và truy vấn dữ liệu một cách dễ dàng mà không cần tuân thủ một sơ đồ dữ liệu cứng nhắc.

MySQL: Là một cơ sở dữ liệu **SQL** truyền thống với mô hình **quan hệ**. Dữ liệu được lưu trữ dưới dạng **bảng** (tables) có liên kết với nhau thông qua **khóa ngoại** (foreign keys). Điều này phù hợp với các ứng dụng có cấu trúc dữ liệu phức tạp và cần đảm bảo tính toàn vẹn dữ liệu.

2.6.2 Khả năng mở rộng và hiệu suất

Firestore: Được thiết kế với khả năng **mở rộng theo chiều ngang** (horizontal scaling). Khi lượng dữ liệu và số lượng người dùng tăng, Firestore có thể tự động phân phối dữ liệu qua các cụm máy chủ để đảm bảo hiệu suất. Cách này giúp tăng khả năng mở rộng và giảm thiểu tắc nghẽn khi có nhiều yêu cầu đồng thời.

MySQL: Chủ yếu được thiết kế với **mở rộng theo chiều dọc** (vertical scaling), tức là cần nâng cấp phần cứng để nâng cao hiệu suất khi lượng dữ liệu và số lượng người dùng tăng. Mặc dù có thể áp dụng **replication** và **sharding**, nhưng việc mở rộng quy mô vẫn phức tạp và đòi hỏi nhiều công sức.

2.6.3 Khả năng đồng bộ hóa dữ liệu theo thời gian

Firestore: Cung cấp khả năng **đồng bộ hóa thời gian thực**. Bất kỳ thay đổi nào được thực hiện đều được cập nhật ngay lập tức trên tất cả các thiết bị liên quan. Tính năng này đặc biệt phù hợp cho các ứng dụng cần chia sẻ thông tin và cộng tác, như quản lý lớp học, thông tin sinh viên, hoặc ứng dụng chat.

MySQL: Không có khả năng đồng bộ hóa dữ liệu thời gian thực mà cần sử dụng các giải pháp bổ sung, như **polling** hoặc **trigger**, để phát hiện và xử lý thay đổi. Điều này có thể gây ra độ trễ và tốn thêm công sức cho lập trình viên để đảm bảo dữ liệu luôn được cập nhật kịp thời.

2.6.4 Tính linh hoạt của mô hình dữ liệu

Firestore: Do là cơ sở dữ liệu NoSQL, Firestore có mô hình dữ liệu rất **linh hoạt**, phù hợp cho các ứng dụng có yêu cầu thay đổi cấu trúc dữ liệu liên tục mà không cần thay đổi toàn bộ hệ thống. Cách lưu trữ dữ liệu không cần phải theo một sơ đồ cố định, giúp việc phát triển nhanh hơn.

MySQL: Cần một **sơ đồ dữ liệu cố định** và rõ ràng. Mô hình dữ liệu quan hệ yêu cầu phải xác định bảng, trường dữ liệu và quan hệ giữa chúng trước khi sử dụng, điều này đảm bảo tính toàn vẹn và hợp nhất của dữ liệu nhưng thiếu tính linh hoạt khi cần thay đổi cấu trúc.

2.6.5 Truy vấn và xử lý dữ liệu

Firestore: Sử dụng các **truy vấn đơn giản** để truy cập dữ liệu, phù hợp với cấu trúc dạng document. Các truy vấn có thể được thực hiện với nhiều điều kiện khác nhau nhưng không hỗ trợ liên kết phức tạp giữa các Collections. Điều này làm cho việc thực hiện các truy vấn phức tạp trở nên khó khăn hơn so với cơ sở dữ liệu quan hệ.

MySQL: Sử dụng **ngôn ngữ truy vấn có cấu trúc (SQL)** để truy vấn dữ liệu, có khả năng thực hiện các truy vấn phức tạp với nhiều phép nối (join) giữa các bảng khác nhau. Điều này giúp MySQL rất phù hợp khi cần truy vấn dữ liệu có mối liên hệ phức tạp.

2.6.6 Chi phí triển khai và quản lý

Firestore: Là dịch vụ được quản lý hoàn toàn bởi Google, nên chi phí quản lý và triển khai ban đầu thấp. Tuy nhiên, chi phí sử dụng sẽ phụ thuộc vào số lượng **lần đọc, ghi, và lưu trữ dữ liệu**, và có thể tăng lên khi số lượng người dùng và dữ liệu lớn.

MySQL: Là một cơ sở dữ liệu mã nguồn mở, chi phí triển khai có thể thấp, nhưng yêu cầu **cơ sở hạ tầng riêng** và đội ngũ quản trị để duy trì, cập nhật và quản lý. Chi phí quản lý có thể tăng khi mở rộng quy mô hệ thống, nhưng người dùng có thể kiểm soát tốt hơn về mặt chi phí phần cứng và triển khai.

2.6.7 Trường hợp sử dụng phù hợp

Firestore: Phù hợp với các ứng dụng cần **đồng bộ hóa thời gian thực**, có cấu trúc dữ liệu **linh hoạt** và thay đổi liên tục. Ví dụ: ứng dụng quản lý thông tin sinh viên, ứng dụng học trực tuyến, ứng dụng chat, và ứng dụng cần tính năng offline.

MySQL: Phù hợp với các ứng dụng cần **cấu trúc dữ liệu phức tạp và cố định**, như hệ thống quản lý nhân sự, hệ thống thương mại điện tử với các bảng sản phẩm, khách hàng, đơn hàng có mối liên hệ phức tạp, hoặc hệ thống yêu cầu truy vấn dữ liệu có tính liên kết chặt chẽ.

2.6.8 Ưu và nhược điểm của mỗi loại cơ sở dữ liệu

- **Firestore:**

Ưu điểm: Đồng bộ hóa thời gian thực, hỗ trợ hoạt động offline, khả năng mở rộng linh hoạt, dễ triển khai và quản lý.

Nhược điểm: Chi phí có thể tăng nhanh khi số lượng người dùng và dữ liệu tăng; giới hạn trong khả năng truy vấn phức tạp; phụ thuộc vào hạ tầng của Google.

- **MySQL:**

Ưu điểm: Cấu trúc dữ liệu rõ ràng, hỗ trợ truy vấn phức tạp, khả năng quản lý dữ liệu tốt với tính toàn vẹn cao, chi phí triển khai thấp nếu sử dụng hạ tầng riêng.

Nhược điểm: Khả năng mở rộng không linh hoạt như NoSQL, không hỗ trợ đồng bộ hóa thời gian thực và hoạt động offline, thiếu tính linh hoạt khi thay đổi cấu trúc dữ liệu.

CHƯƠNG 3 – THIẾT KẾ VÀ XÂY DỰNG ỨNG DỤNG QUẢN LÝ THÔNG TIN SINH VIÊN

3.1 Phân tích yêu cầu ứng dụng

Ứng dụng cần cung cấp ba vai trò người dùng: quản trị viên (admin), quản lý (manager) và nhân viên (employee).

3.1.1 Yêu cầu chức năng

- Quản lý thông tin sinh viên
 - Xem danh sách sinh viên
 - Thêm sinh viên mới
 - Xóa sinh viên
 - Cập nhật thông tin sinh viên
 - Sắp xếp danh sách sinh viên dựa trên các tiêu chí khác nhau
 - Tìm kiếm sinh viên theo nhiều tiêu chí
 - Truy cập trang chi tiết sinh viên
 - Xem thông tin đầy đủ của sinh viên trên màn hình chi tiết
 - Nhập danh sách sinh viên từ tệp
 - Xuất danh sách sinh viên sang Excel/CSV
- Quản lý tài khoản người dùng
 - Đăng nhập (cho tất cả người dùng)
 - Thay đổi ảnh đại diện (cho tất cả người dùng)
 - Xem danh sách người dùng hệ thống
 - Thêm người dùng mới: Tên, Tuổi, Số điện thoại, Trạng thái (Bình thường/Đã khóa)
 - Xóa người dùng
 - Chỉnh sửa thông tin người dùng

- Xem lịch sử đăng nhập của người dùng
- Quản lý danh sách chứng chỉ cho sinh viên:
 - Xem danh sách chứng chỉ
 - Thêm chứng chỉ mới
 - Xóa chứng chỉ
 - Cập nhật thông tin chứng chỉ
 - Nhập danh sách chứng chỉ cho sinh viên từ tệp
 - Xuất danh sách chứng chỉ sang Excel/CSV

3.1.2 Yêu cầu phi chức năng

- **Bảo mật:** Đảm bảo dữ liệu được bảo mật, quyền truy cập được kiểm soát theo từng cấp độ người dùng, cụ thể như quản trị viên có thể thực hiện tất cả các chức năng, quản lý có thể thực hiện tất cả các chức năng liên quan đến sinh viên, còn tài khoản nhân viên chỉ có thể xem nội dung. Nhân viên không được phép chỉnh sửa bất kỳ nội dung nào ngoại trừ việc cập nhật ảnh đại diện của họ.
- **Khả năng mở rộng:** Hệ thống cần có khả năng mở rộng để đáp ứng số lượng lớn sinh viên và giáo viên.
- **Hiệu suất:** Ứng dụng phải hoạt động nhanh chóng và đảm bảo tính ổn định khi có nhiều người dùng truy cập đồng thời.
- **Khả năng hoạt động ngoại tuyến:** Cho phép người dùng truy cập dữ liệu và cập nhật thông tin ngay cả khi không có kết nối internet.

3.2 Thiết kế kiến trúc hệ thống

3.2.1 Mô hình tổng quan

- **Frontend:**

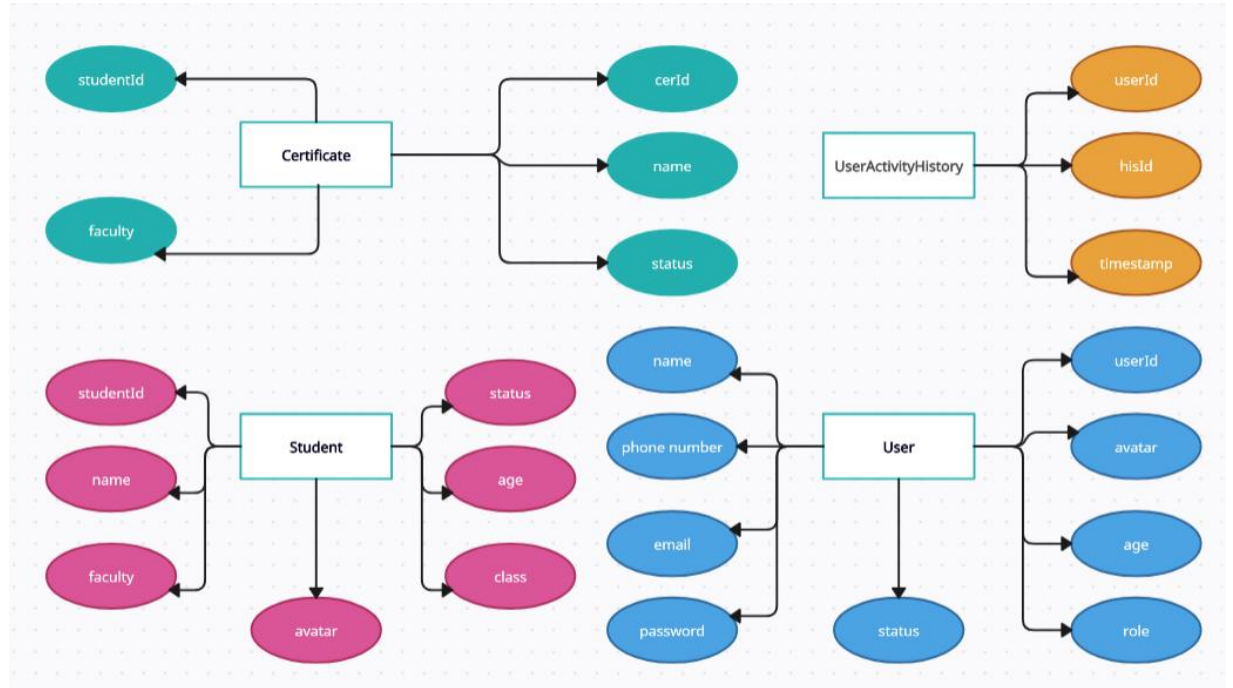
Android Application: Thiết kế giao diện người dùng bằng **Java** trên **Android Studio** để tạo ra ứng dụng Android với trải nghiệm tương tác thân thiện, dễ sử dụng cho giáo viên và các quản trị viên. Giao diện được thiết kế đơn giản và rõ ràng, giúp người dùng dễ dàng thao tác các chức năng như quản lý thông tin sinh viên, chứng chỉ.

- **Backend:**

Firebase Firestore: Sử dụng **Firestore Database** để lưu trữ và quản lý dữ liệu chính của hệ thống như thông tin sinh viên, giáo viên, danh sách tín chỉ. Firestore cung cấp khả năng lưu trữ linh hoạt với mô hình **Collections** và **Documents** phù hợp cho việc quản lý thông tin sinh viên và cập nhật dữ liệu theo thời gian thực.

Firebase Storage: Dùng **Firebase Storage** để lưu trữ ảnh đại diện của sinh viên và người dùng, giúp quản lý nội dung đa phương tiện một cách dễ dàng và hiệu quả. Firebase Storage hỗ trợ upload và download file với độ bảo mật cao, giúp lưu trữ các hình ảnh với khả năng truy cập nhanh chóng.

3.2.2 Thiết kế cơ sở dữ liệu



Hình 1 Sơ đồ cơ sở dữ liệu

Student Collection: Lưu trữ thông tin sinh viên, mỗi sinh viên là một **Document** trong **Collection**, bao gồm các trường như họ tên, tuổi, mã số sinh viên, lớp, ảnh đại diện, trạng thái, khoa và sẽ được lưu trữ dưới dạng các **Fields** của **Document**.

User Collection: Lưu trữ thông tin về giáo viên, mỗi user là một **Document** chứa các trường như mã user, avatar, age, role, name, phone number, email, password, status và sẽ được lưu trữ dưới dạng các **Fields** của **Document**.

UserActivityHistory Collection: Lưu trữ thông tin về lịch sử hoạt động của các tài khoản user, mỗi lịch sử hoạt động của một tài khoản tại một thời điểm là một **Document** chứa các trường như mã userId, mã hisId,

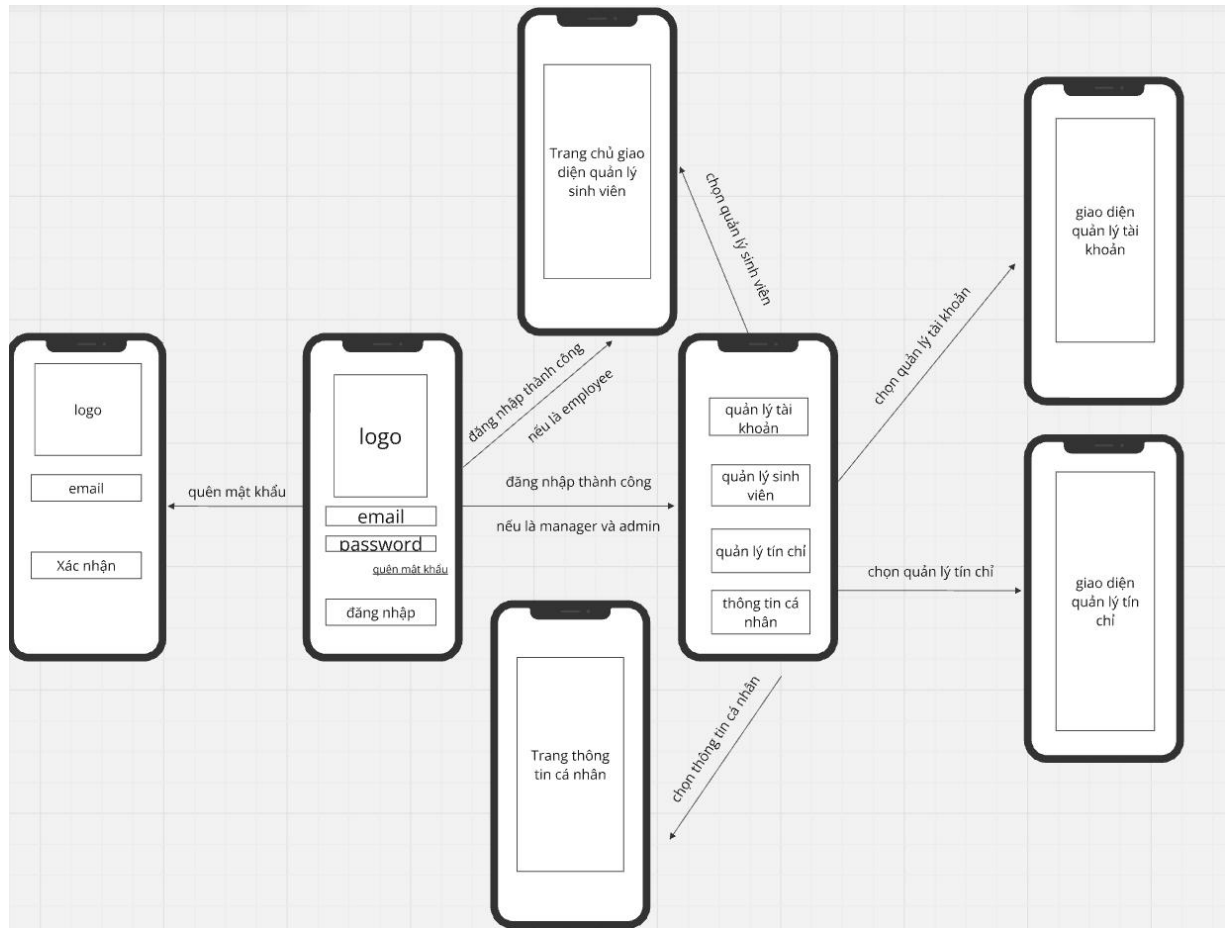
thời gian tại thời điểm truy cập và sẽ được lưu trữ dưới dạng các **Fields** của **Document**.

Certificate Collection: Lưu trữ thông tin về các chứng chỉ của sinh viên, mỗi **Document** bao gồm tên chứng chỉ, mã tín chỉ, khoa, trạng thái, và mã số sinh viên để liên kết với sinh viên và sẽ được lưu trữ dưới dạng các **Fields** của **Document**.

3.3 Xây dựng hệ thống và triển khai

3.3.1 *Thiết kế giao diện người dùng*

3.3.1.1 *Sơ đồ màn hình (Screen Flow Diagram)*



Hình 1 Sơ đồ màn hình (Screen Flow Diagram)

Màn hình Đăng nhập : Đây là màn hình đầu tiên mà người dùng sẽ gặp khi truy cập vào ứng dụng. Người dùng có thể đăng nhập nếu đã có tài khoản hoặc chọn quên mật khẩu nếu người dùng quên mật khẩu .

Màn hình quên mật khẩu : Sau khi bấm quên mật khẩu màn hình này sẽ hiện ra , yêu cầu người dùng nhập email đã được đăng ký tài khoản và hệ thống sẽ tự động gửi mật khẩu về email đấy .

Màn hình Chọn dữ liệu quản lý: Sau khi đăng nhập, nếu người dùng sử dụng tài khoản có quyền hạn là admin hoặc manager thì sẽ chuyển sang màn hình chọn dữ liệu để quản lý gồm quản lý tài khoản , quản lý sinh viên , quản lý

tín chỉ , riêng đối với manager thì chỉ được phép sử dụng chức năng quản lý sinh viên , quản lý tín chỉ.

Màn hình Quản lý Sinh viên: Sau khi đăng nhập , nếu người dùng sử dụng tài khoản có quyền hạn là employee thì sẽ vào thẳng giao diện quản lý sinh viên , và xem các nội dung liên quan đến thông tin sinh viên , và không được phép sử dụng bất kỳ tính năng nào(thêm , sửa , xóa, nhập , xuất, excel,..) trừ chức năng thay đổi ảnh đại diện.

Màn hình Quản lý Tài khoản Người dùng: Cho phép quản trị viên xem, thêm, xóa và chỉnh sửa thông tin người dùng hệ thống, xem lịch sử hoạt động của các tài khoản, tìm kiếm.

Màn hình Quản lý Chứng chỉ: Hiển thị danh sách chứng chỉ của sinh viên, cho phép thêm mới, cập nhật hoặc xóa chứng chỉ, nhập xuất tín chỉ từ file excel/csv.

3.3.1.2 Wireframes cho ứng dụng

- **Màn hình Đăng nhập (Login Screen)**

- *Thành phần chính:*

- **Trường nhập Email:** Nhập địa chỉ email của người dùng.
 - **Trường nhập Mật khẩu:** Nhập mật khẩu của người dùng.
 - **Nút Đăng nhập:** Nhấn để đăng nhập vào hệ thống.
 - **Nút Quên Mật khẩu:** Nhấn để di chuyển đến màn hình Quên Mật khẩu.

- *Mô tả:* Màn hình đăng nhập đơn giản với hai trường nhập và các nút chức năng rõ ràng, giúp người dùng dễ dàng truy cập.

- **Màn hình Quên Mật khẩu**

- *Thành phần chính:*

- **Trường nhập Email:** Yêu cầu người dùng nhập địa chỉ email đã đăng ký.
 - **Nút Gửi Yêu Cầu(Xác nhận) :** Khi nhấn vào, hệ thống sẽ gửi thông tin đặt lại mật khẩu tới email của người dùng.

- **Thông báo (Message Box):** Hiện thị thông báo thành công khi email đặt lại mật khẩu đã được gửi và chuyển về trang đăng nhập
 - *Mô tả:* Màn hình đơn giản với trường nhập email và nút gửi yêu cầu, nhằm hướng dẫn người dùng khôi phục mật khẩu.
- **Màn hình Chọn Dữ liệu Quản lý**
 - *Thành phần chính:*
 - **Danh sách các lựa chọn quản lý (Selection List):**
 - **Quản lý Tài khoản** (chỉ dành cho Admin).
 - **Quản lý Sinh viên.**
 - **Quản lý Chứng chỉ.**
 - **Nút Chọn (Select Button):** Nhân để di chuyển đến màn hình tương ứng với chức năng được chọn.
 - *Mô tả:* Màn hình này có các lựa chọn chức năng rõ ràng dành cho **admin** hoặc **manager** để tiếp tục quản lý hệ thống.
- **Màn hình Quản lý Sinh viên**
 - *Thành phần chính:*
 - **Danh sách Sinh viên (Student List):** Hiện thị danh sách sinh viên dưới dạng bảng, bao gồm các thông tin như **Họ tên, Mã số sinh viên, Lớp.**
 - **Nút Tìm kiếm (Search Bar):** Cho phép tìm kiếm sinh viên dựa trên các tiêu chí như **Tên, Mã số sinh viên.**
 - **Nút Chi tiết (Detail Button):** Để xem chi tiết thông tin sinh viên.
 - **Nút Thay đổi Ảnh Đại diện (Change Avatar Button):** Dành cho **employee** để thay đổi ảnh đại diện của sinh viên.
 - **Các Nút Thêm, Sửa, Xóa, Nhập/Xuất (Add, Edit, Delete, Import/Export Buttons):** Chỉ dành cho **admin** và **manager**, không khả dụng cho **employee.**
 - *Mô tả:* Màn hình này có đầy đủ các nút thao tác quản lý sinh viên, trong đó **employee** chỉ có thể xem và thay đổi ảnh đại diện.
- **Màn hình Quản lý Tài khoản Người dùng**
 - *Thành phần chính:*

- **Danh sách Người dùng (User List):** Hiện thị danh sách tất cả người dùng với các thông tin như **Tên, Vai trò, Trạng thái,..**
- **Nút Thêm Người dùng (Add User Button):** Cho phép thêm người dùng mới.
- **Nút Xóa (Delete Button):** Cho phép xóa người dùng khỏi hệ thống.
- **Nút Sửa (Delete Button):** Cho phép sửa thông tin người dùng
- **Nút Tìm kiếm Người dùng (Search User Bar):** Tìm kiếm người dùng trong hệ thống.
- **Nút Xem Lịch sử Hoạt động (View Activity Log Button):** Hiện thị lịch sử hoạt động của các tài khoản truy cập vào hệ thống.
- **Mô tả:** Màn hình này cho phép **admin** quản lý tài khoản người dùng, bao gồm các thao tác thêm mới, chỉnh sửa, xóa và xem lịch sử hoạt động.

- **Màn hình Quản lý Chứng chỉ (Certificate Management Screen)**

- **Thành phần chính:**
 - **Danh sách Chứng chỉ (Certificate List):** Hiện thị danh sách chứng chỉ của sinh viên, bao gồm các thông tin như **Tên Chứng chỉ, Ngày Cấp, Sinh viên,..**
 - **Nút Thêm, Xóa, Sửa Chứng chỉ (Add Certificate Button):** Cho phép người dùng thêm xóa sửa tin chỉ .
 - **Nút Nhập/Xuất Chứng chỉ (Import/Export Certificate Button):** Cho phép nhập hoặc xuất danh sách chứng chỉ dưới dạng file **Excel/CSV**.
- **Mô tả:** Màn hình này cung cấp khả năng thêm, sửa, xóa và nhập/xuất danh sách chứng chỉ, giúp dễ dàng quản lý các chứng chỉ của sinh viên.

3.3.2 Các thành phần giao diện chính

- **Đăng nhập**



3:01 Sun, Nov 3

QUẢN LÝ THÔNG TIN SINH VIÊN

Email

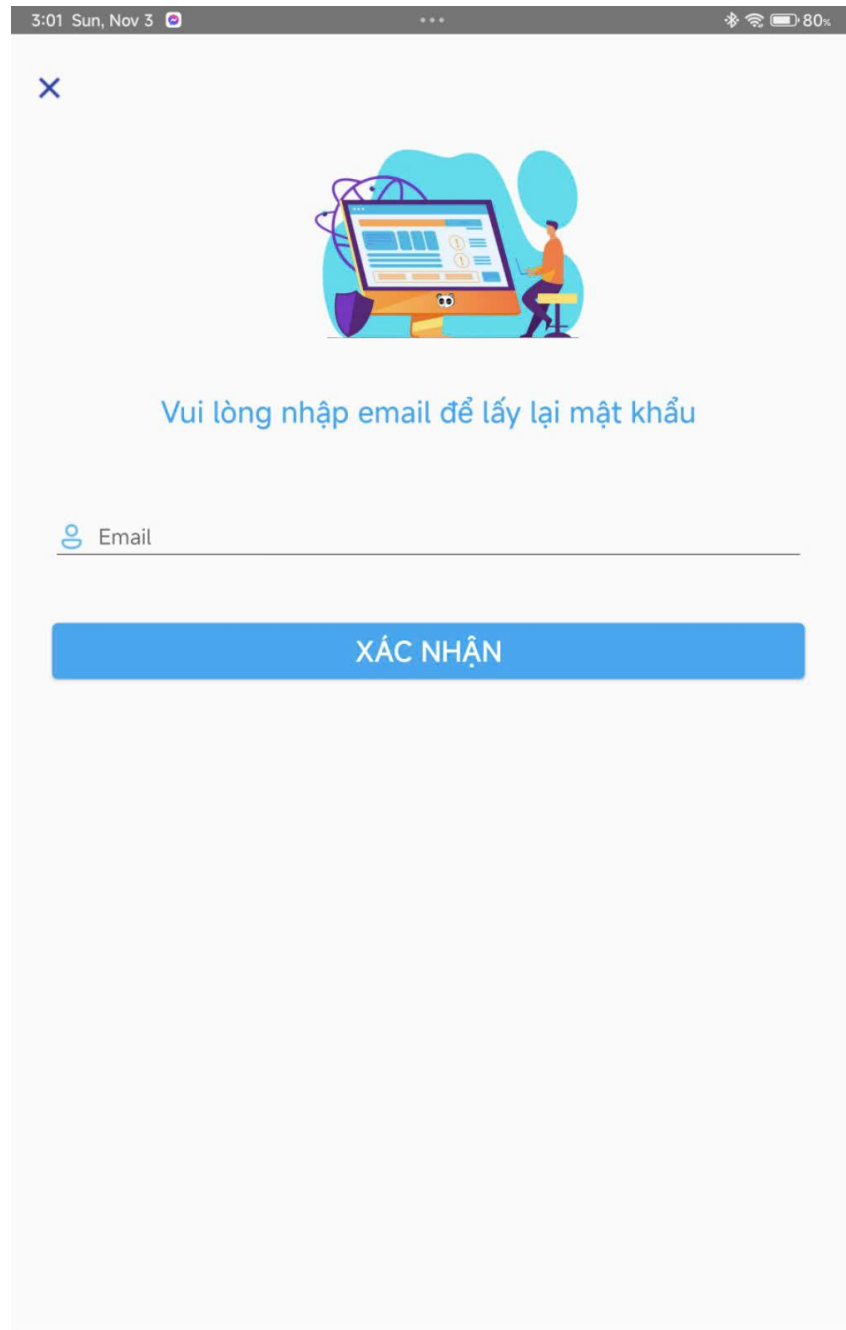
Password

Quên mật khẩu?

ĐĂNG NHẬP

Hình 2 Màn hình đăng nhập

- ***Quên mật khẩu***



3:01 Sun, Nov 3

✕

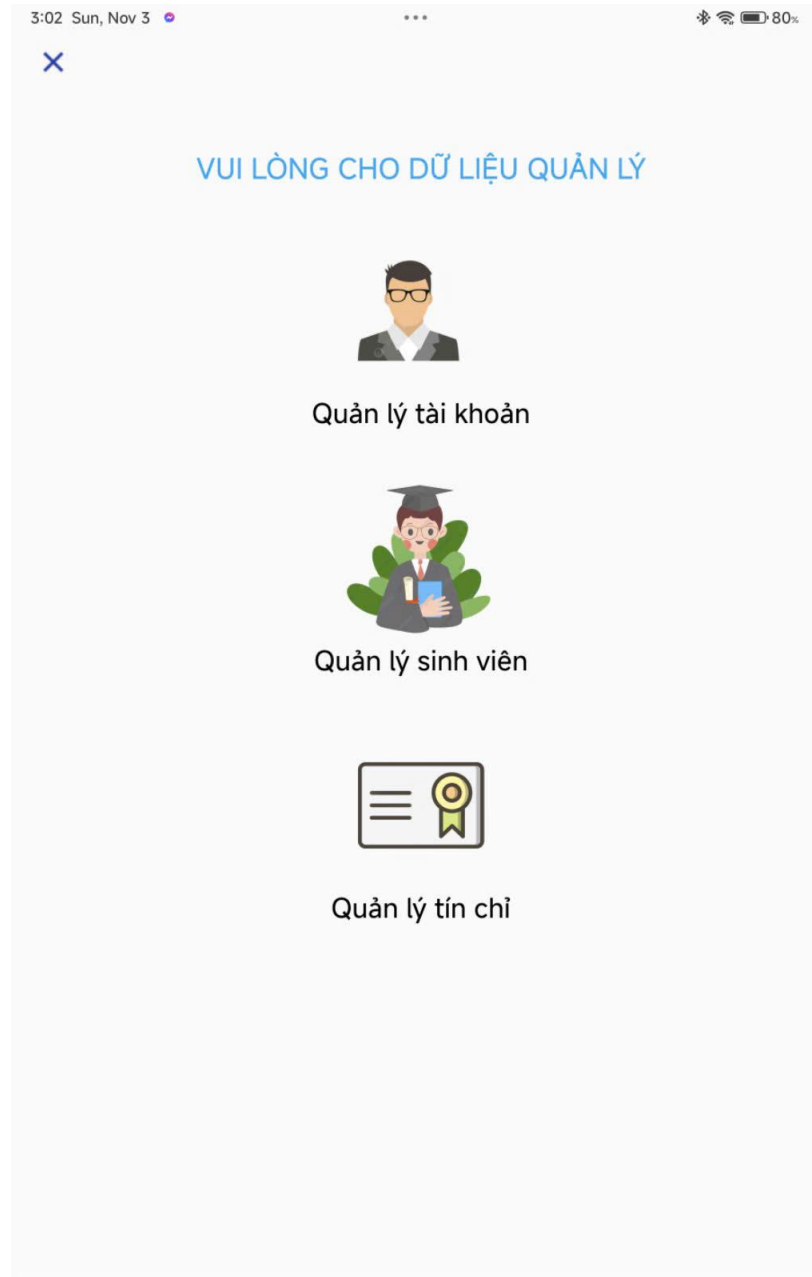
Vui lòng nhập email để lấy lại mật khẩu

Email

XÁC NHẬN

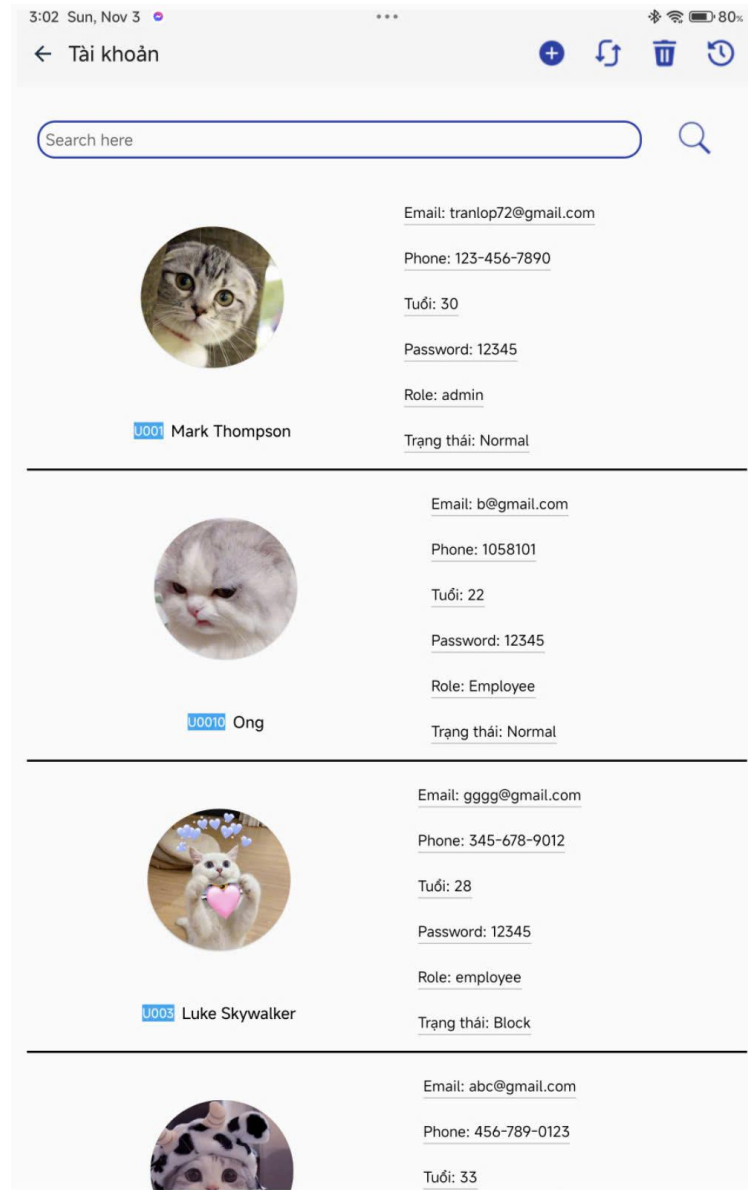
Hình 3 Màn hình quên mật khẩu

- ***Chọn dữ liệu quản lý***



Hình 4 Màn hình chọn dữ liệu quản lý

- **Quản lý tài khoản**



Hình 5 Màn hình quản lý tài khoản

- ***Thêm tài khoản***

3:02 Sun, Nov 3

← Tài khoản

Search here

Thêm tài khoản mới

Mã tài khoản

Họ và tên

Tuổi

Email

Phone

Mật khẩu

Role

Status

HỦY THÊM

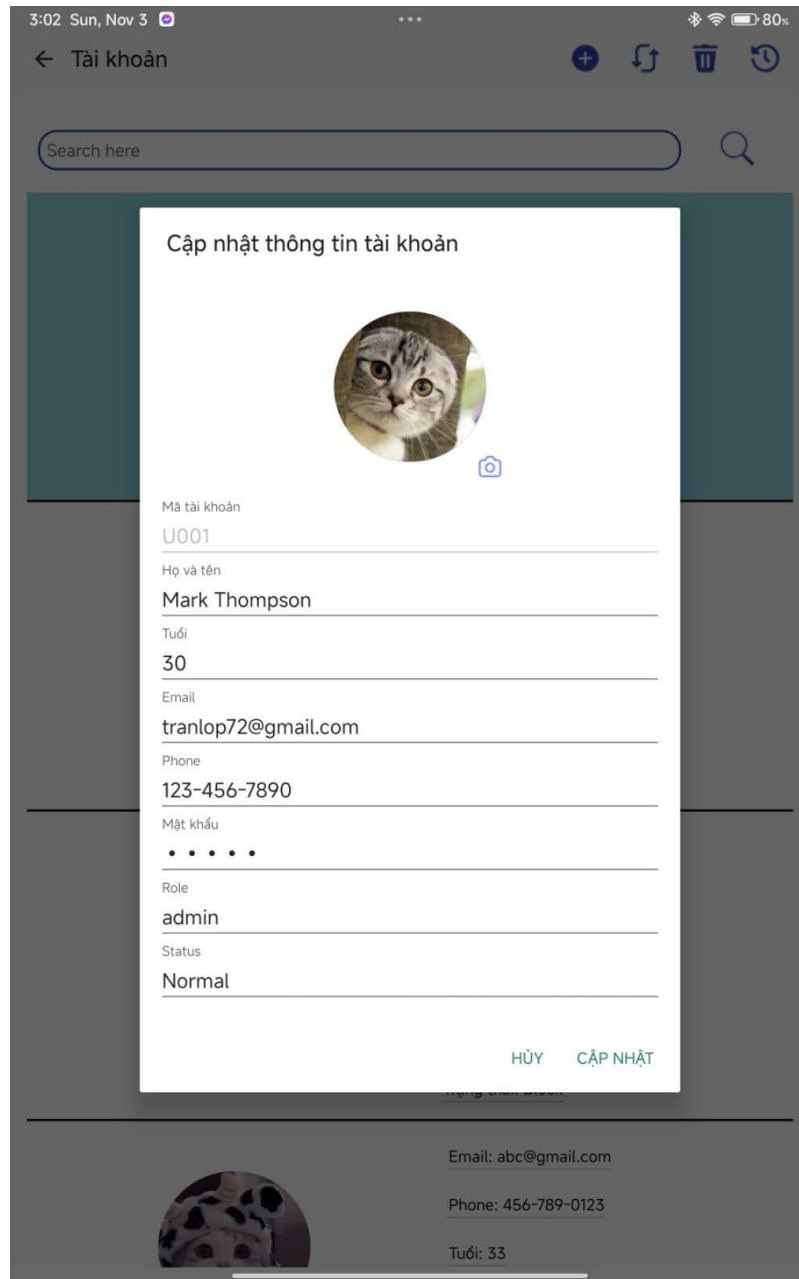
Email: abc@gmail.com

Phone: 456-789-0123

Tuổi: 33

Hình 6 Màn hình thêm tài khoản

- ***Sửa tài khoản***

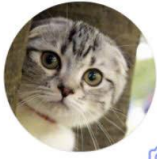


3:02 Sun, Nov 3

← Tài khoản

Search here

Cập nhật thông tin tài khoản



Mã tài khoản
U001

Họ và tên
Mark Thompson

Tuổi
30

Email
tranlop72@gmail.com

Phone
123-456-7890

Mật khẩu
• • • • •

Role
admin

Status
Normal

HỦY CẬP NHẬT

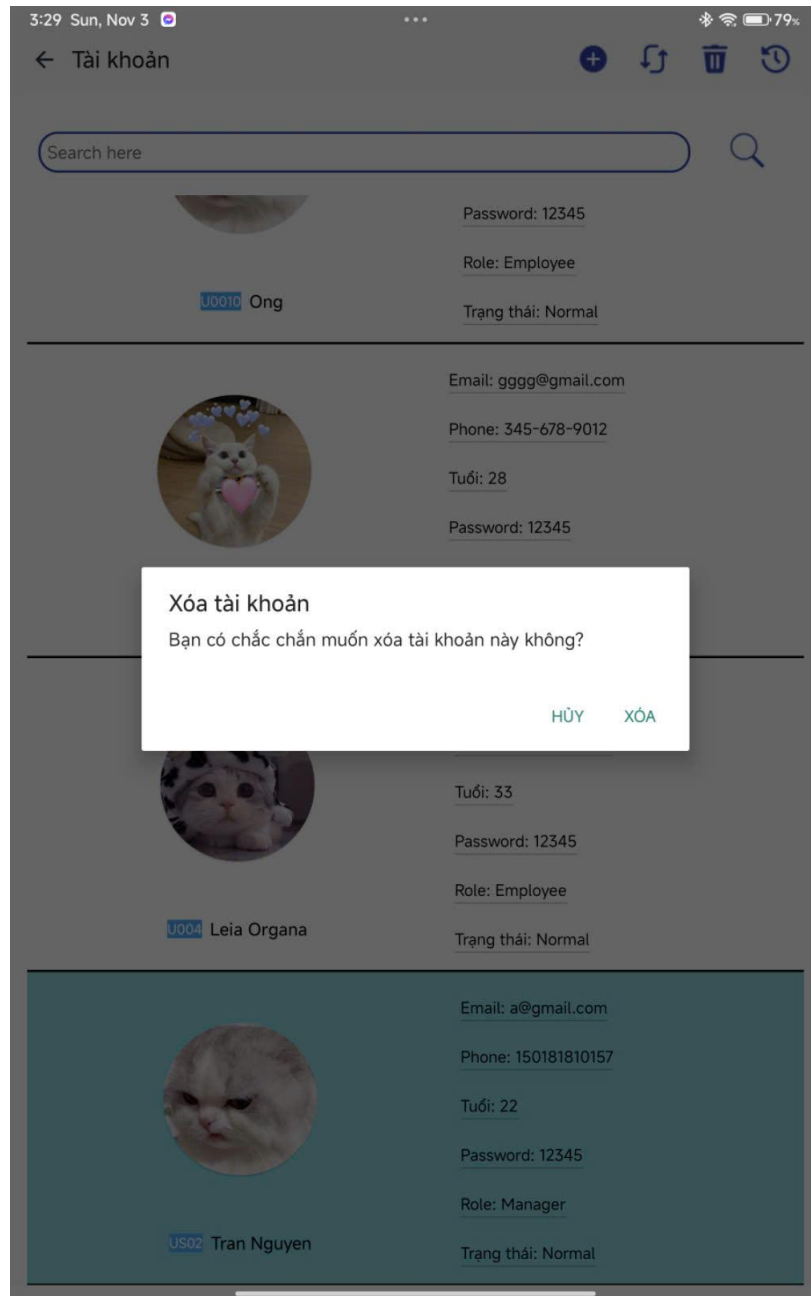
Email: abc@gmail.com

Phone: 456-789-0123

Tuổi: 33

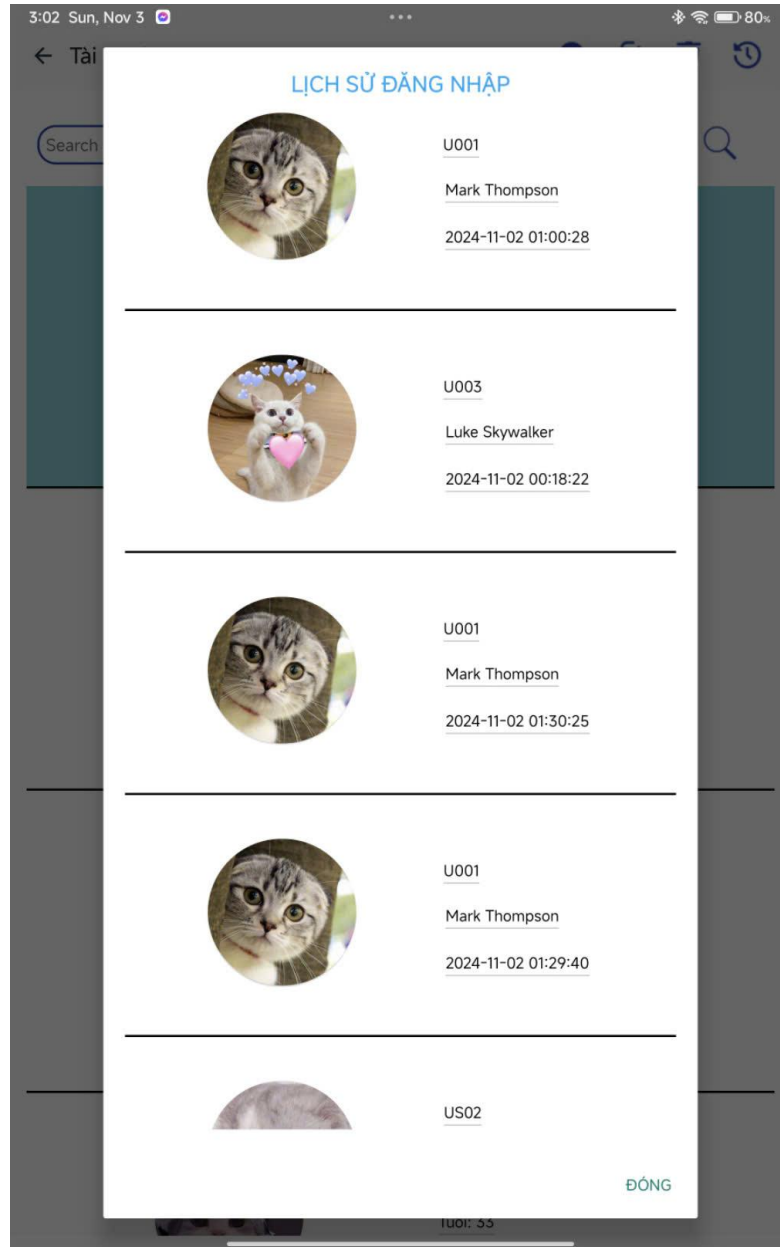
Hình 7 Màn hình cập nhật tài khoản

- **Xóa tài khoản**



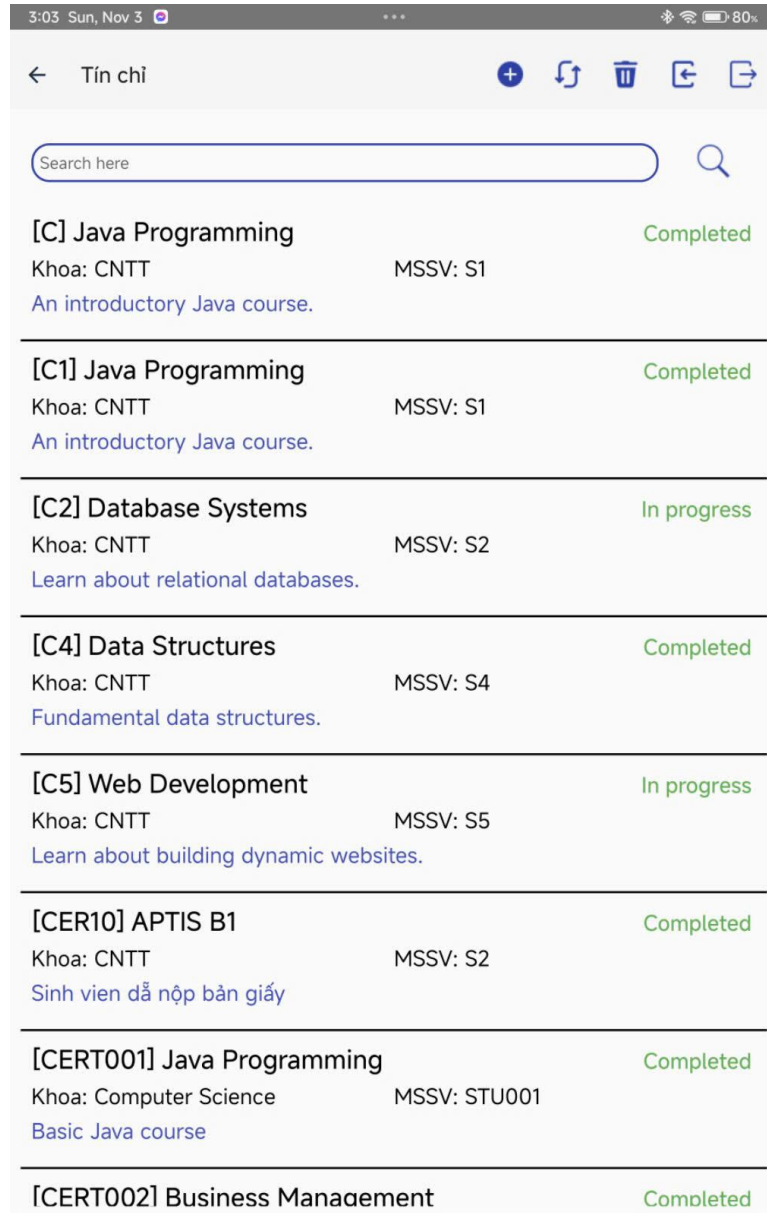
Hình 8 Màn hình xóa tài khoản

- *Xem lịch sử hoạt động*



Hình 9 Màn hình xem lịch sử hoạt động

- ***Quản lý tín chỉ***



3:03 Sun, Nov 3		
Tin chỉ		
Search here		
[C] Java Programming		Completed
Khoa: CNTT	MSSV: S1	
An introductory Java course.		
[C1] Java Programming		Completed
Khoa: CNTT	MSSV: S1	
An introductory Java course.		
[C2] Database Systems		In progress
Khoa: CNTT	MSSV: S2	
Learn about relational databases.		
[C4] Data Structures		Completed
Khoa: CNTT	MSSV: S4	
Fundamental data structures.		
[C5] Web Development		In progress
Khoa: CNTT	MSSV: S5	
Learn about building dynamic websites.		
[CER10] APTIS B1		Completed
Khoa: CNTT	MSSV: S2	
Sinh viên đã nộp bản giấy		
[CERT001] Java Programming		Completed
Khoa: Computer Science	MSSV: STU001	
Basic Java course		
[CERT002] Business Manaaement		Completed

Hình 10 Màn hình quản lý tín chỉ

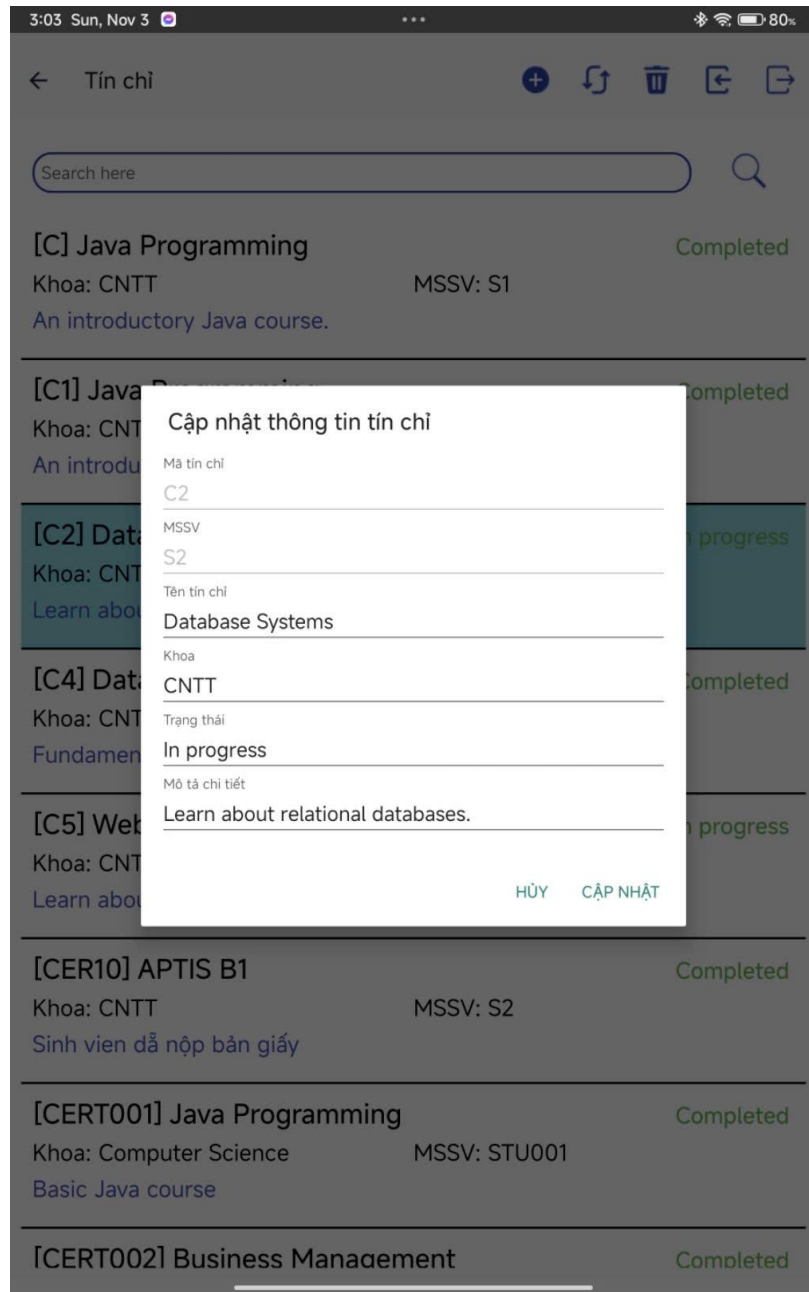
- ***Thêm tín chỉ***

The screenshot shows a mobile application interface for managing credits. At the top, there's a header bar with the title "Tín chỉ" and navigation icons. Below the header is a search bar labeled "Search here". The main content area displays a list of courses, each with a title, department (Khoa), student ID (MSSV), and status (Completed or progress). A modal form titled "Thêm chứng chỉ mới" is overlaid on the screen, containing input fields for "Mã tín chỉ", "MSSV", "Tên tín chỉ", "Khoa", "Trạng thái", and "Mô tả chi tiết". At the bottom of the modal are two buttons: "HỦY" and "THÊM".

Tên tín chỉ	Khoa	MSSV	Trạng thái
[C] Java Programming	CNTT	S1	Completed
[C1] Java Programming	CNTT		Completed
[C2] Data Structures	CNTT		progress
[C4] Data Structures	CNTT		Completed
[C5] Web Development	CNTT		progress
[CER10] APTIS B1	CNTT	S2	Completed
[CERT001] Java Programming	Computer Science	STU001	Completed
[CERT002] Business Management			Completed

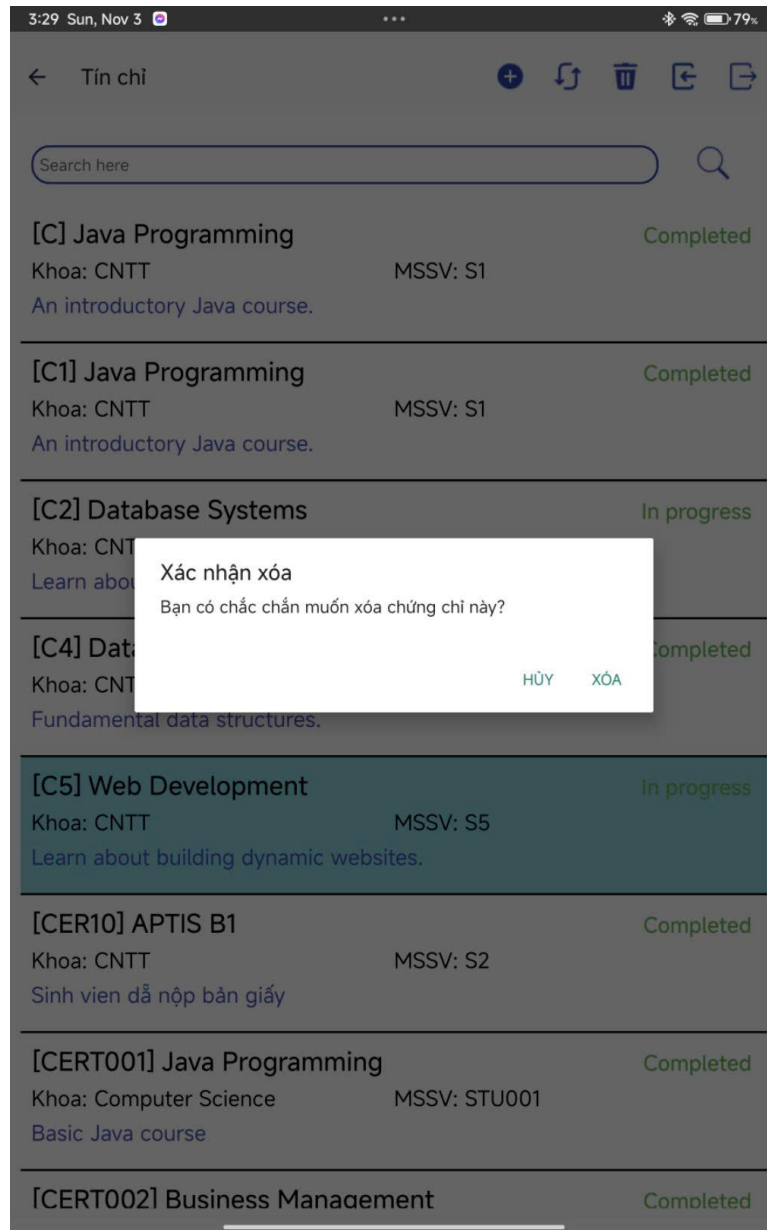
Hình 11 Màn hình thêm tín chỉ

- *Sửa tin chỉ*



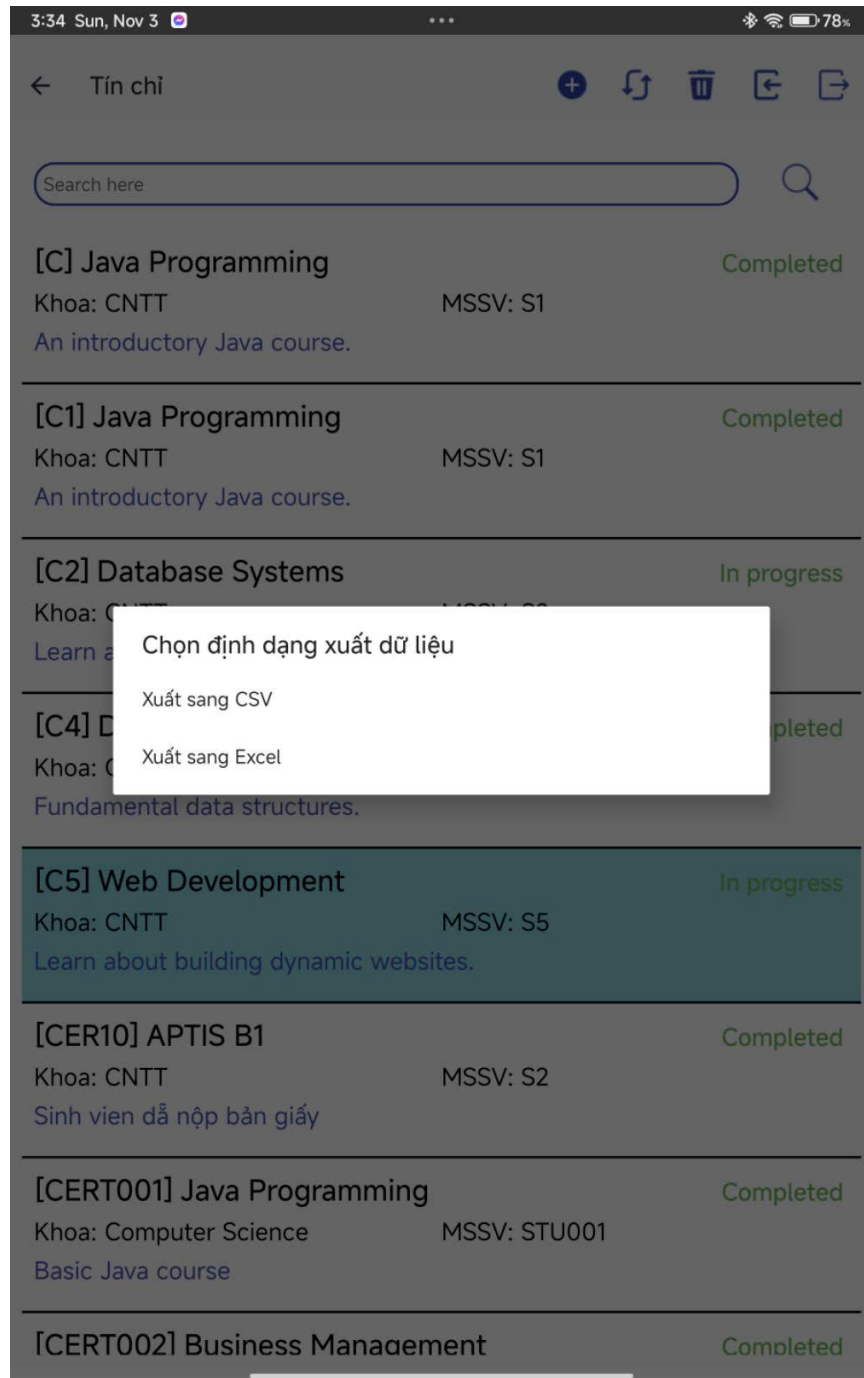
Hình 12 Màn hình sửa tin chỉ

- *Xóa tín chỉ*



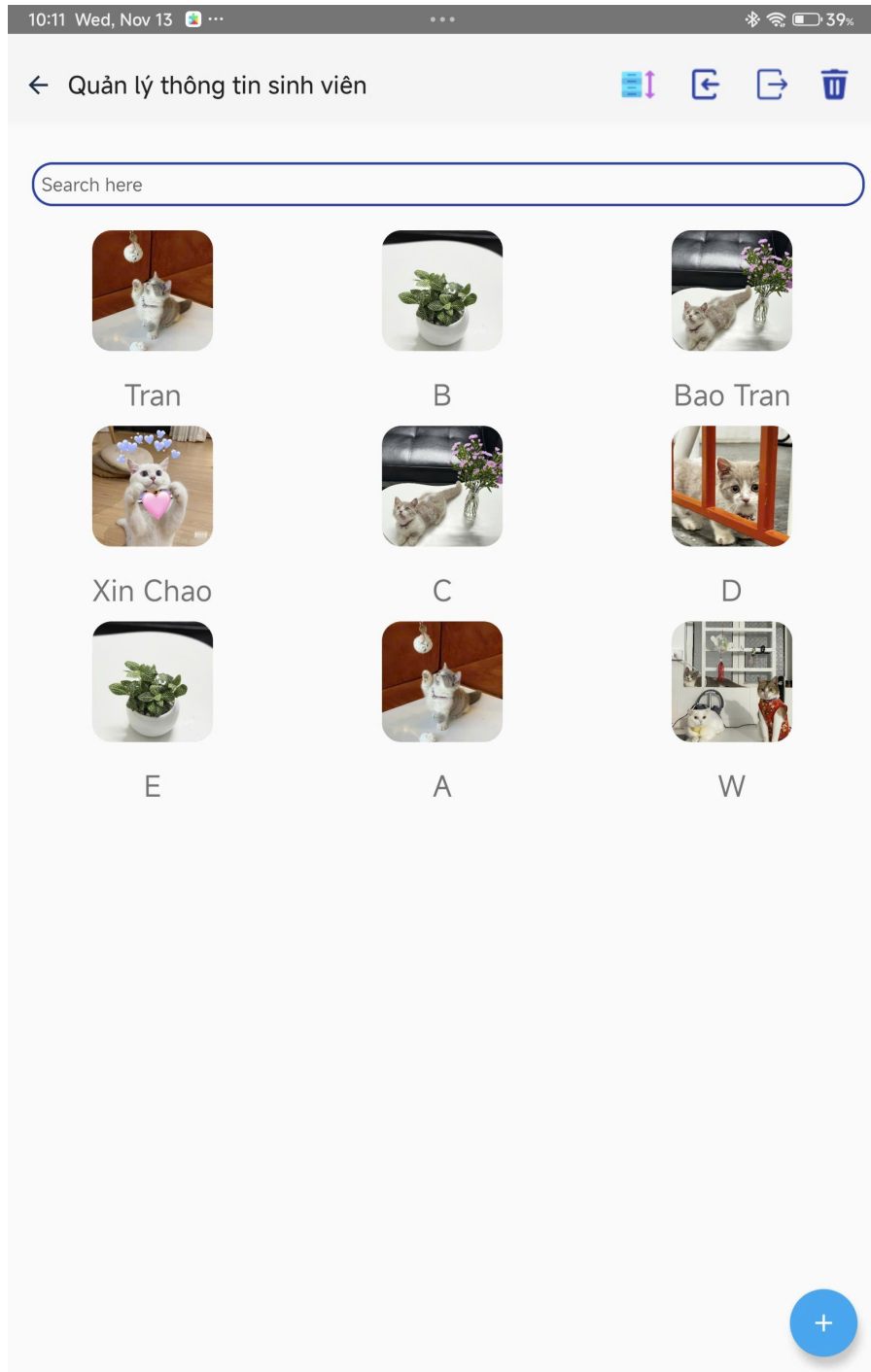
Hình 13 Màn hình xóa tín chỉ

- *Thêm / Xuất dữ liệu thành file excel/csv*



Hình 14 Màn hình thêm / xuất dữ liệu

- ***Quản lý sinh viên***



Hình 15 Màn hình quản lý thông tin sinh viên


- ***Thêm sinh viên***

10:11 Wed, Nov 13 39%

← Quản lý thông tin sinh viên

Search here

Thêm sinh viên mới



STU003

Mã ID STU386

Họ và Tên Nguyễn Văn A

Ngày sinh dd/mm/yyyy

Lớp hoa hong

Khoa Công nghệ thông tin

Giáo viên Nguyen Van A

HỦY THÊM

Hình 16 : Màn hình thêm sinh viên


- ***Cập nhật sinh viên***

12:21 Thu, Nov 14 22%

← Quản lý thông tin sinh viên

Search here

Cập nhật thông tin sinh viên



Mã ID STU003 STU010

Họ và Tên Nguyễn Văn A Tran

Ngày sinh dd/mm/yyyy 30/11/2024

Lớp hoa hong TD

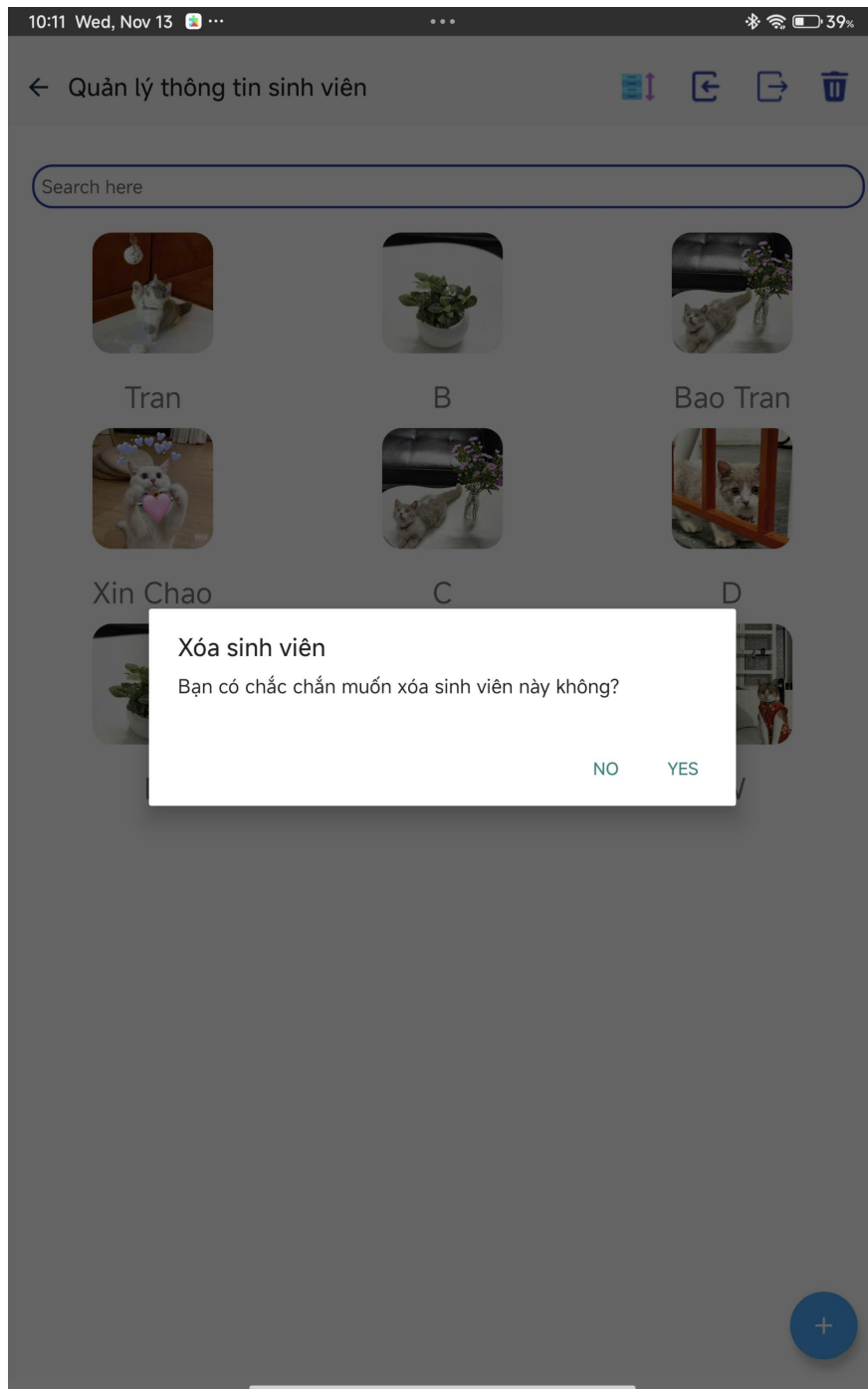
Khoa Công nghệ thông tin CNTT

Giáo viên Nguyen Van A TD

HỦY CẬP NHẬT

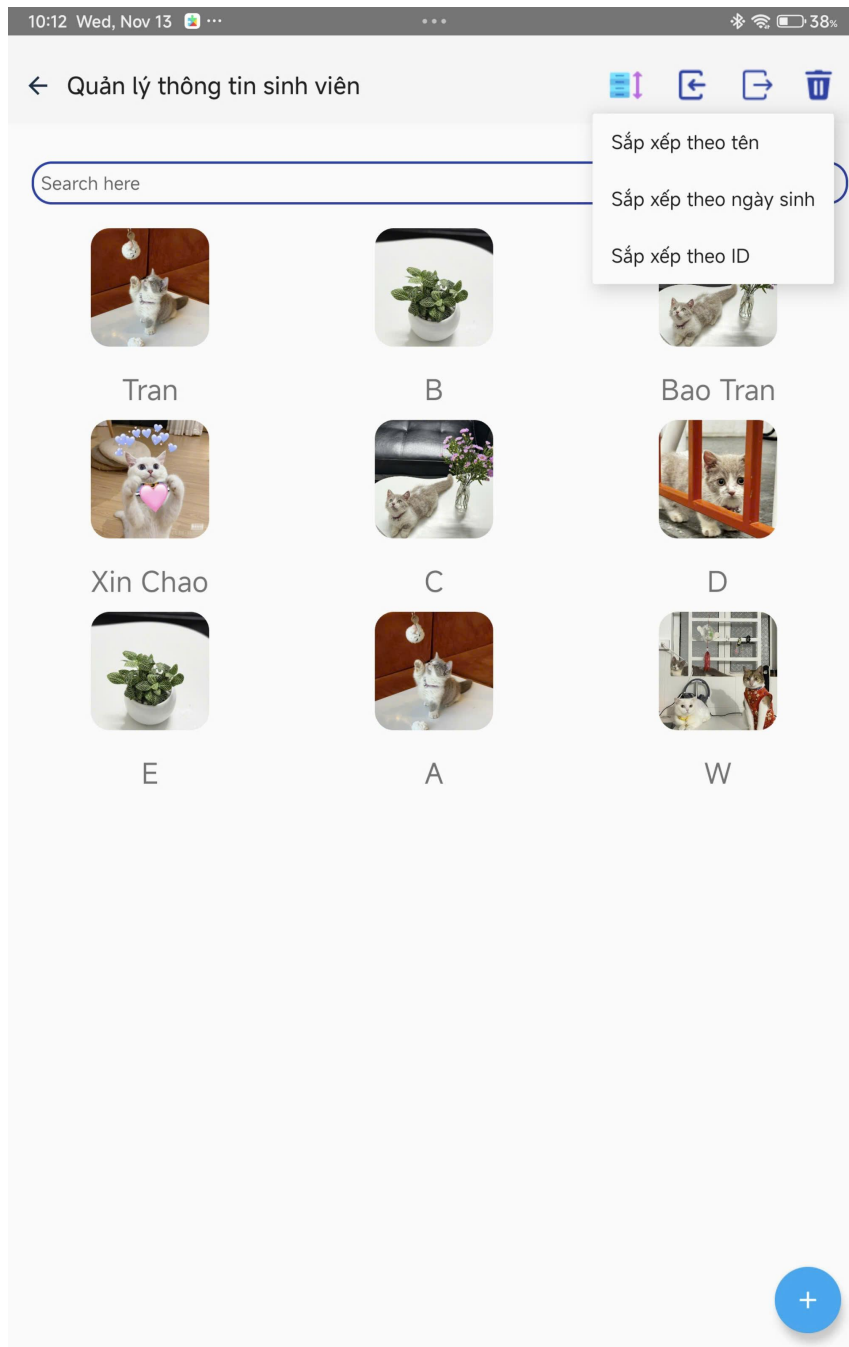
Hình 17 : Màn hình cập nhật thông tin sinh viên

- ***Xóa một sinh viên (nhấn giữ)***



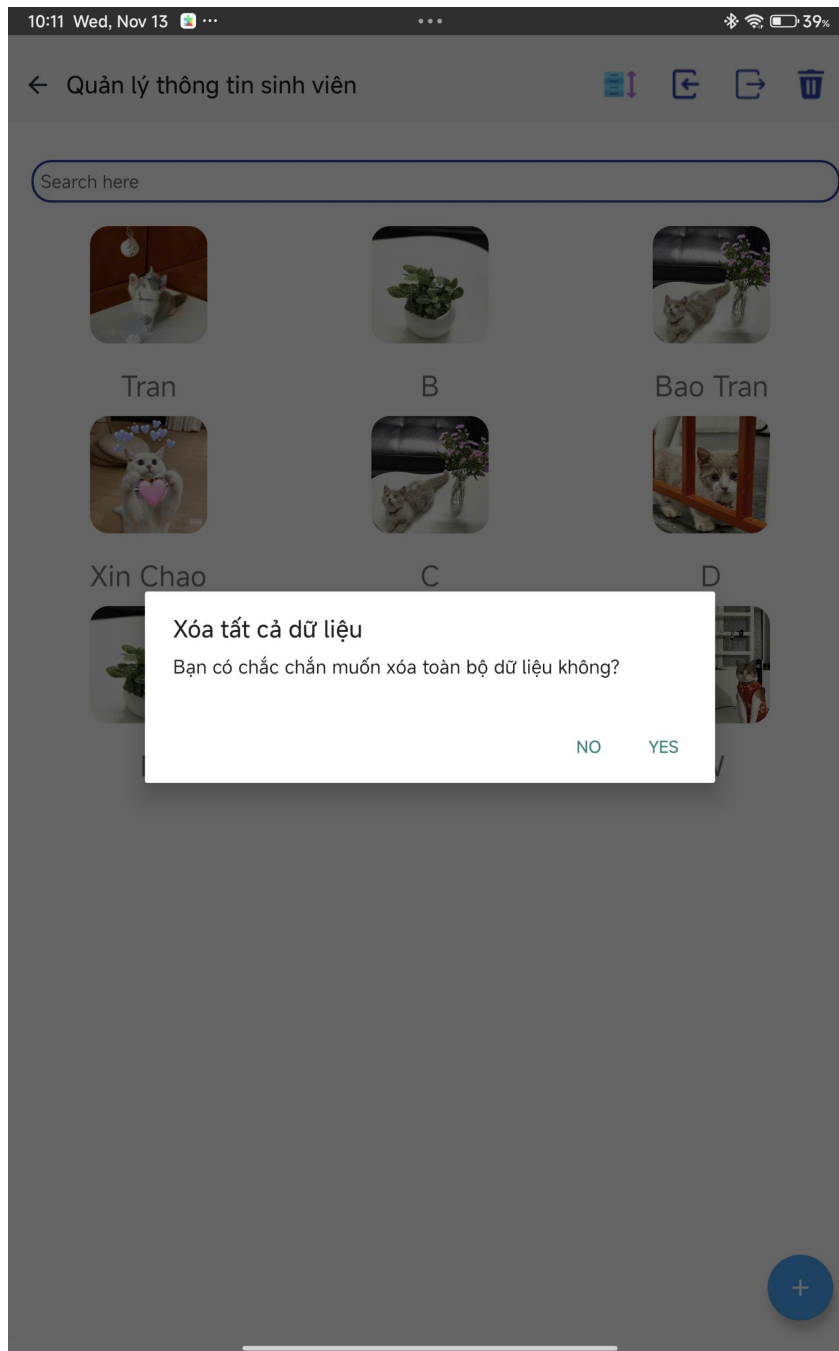
Hình 18 : Xóa một sinh viên

- **Sắp xếp sinh viên**



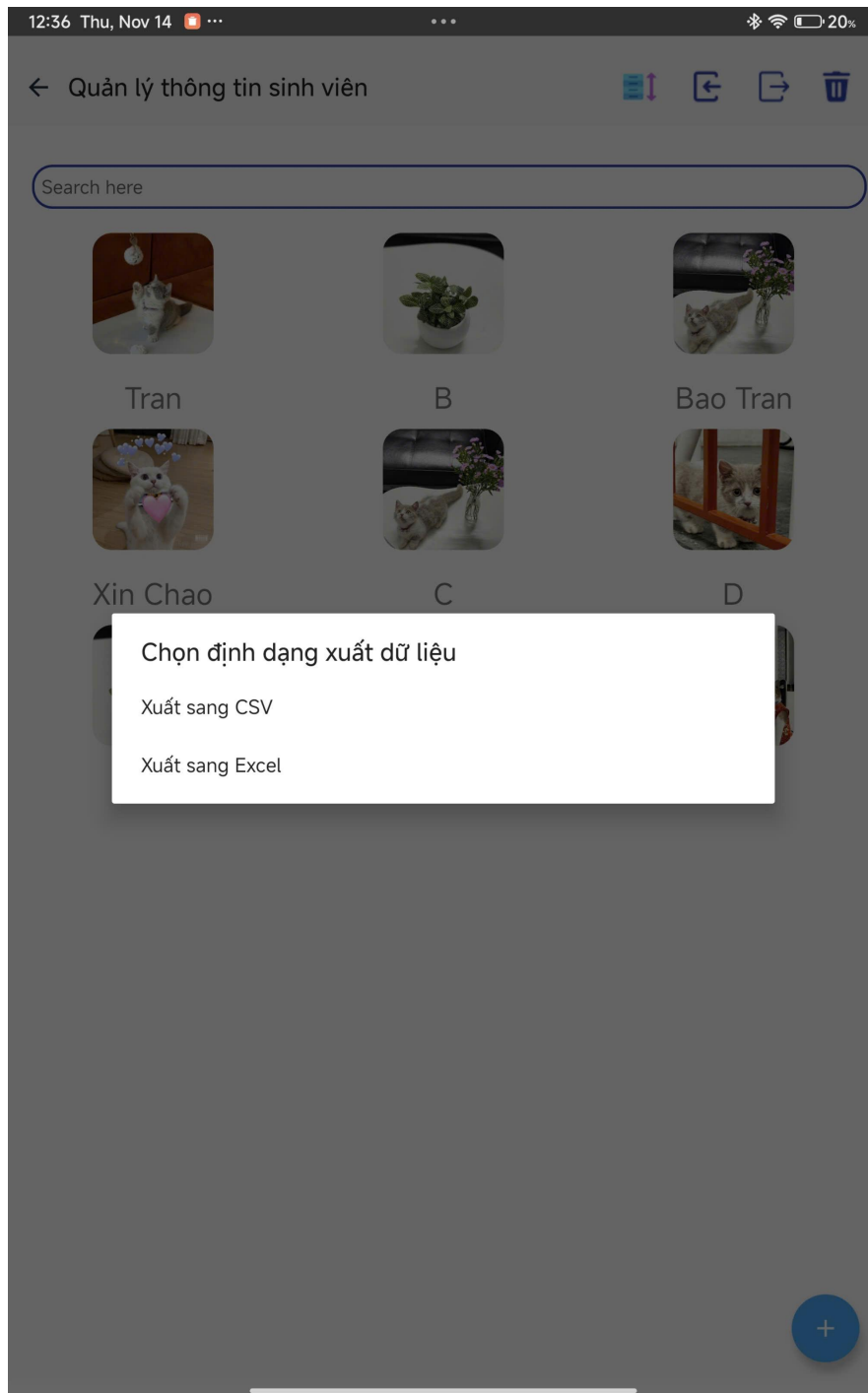
Hình 19 : Sắp xếp sinh viên

- ***Xóa nhiều sinh viên***



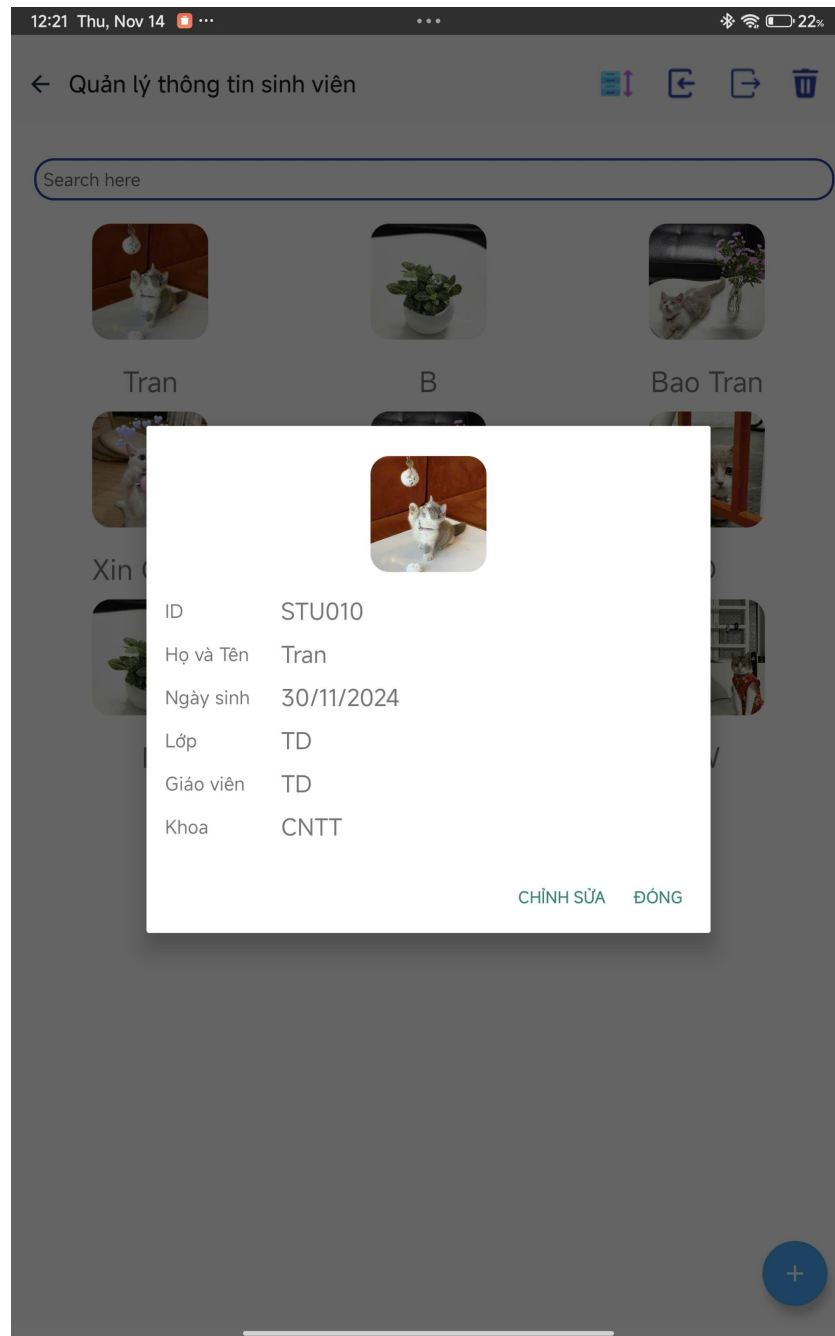
Hình 20 : Xóa nhiều sinh viên

- ***Nhập/xuất sinh viên từ file excel/csv***



Hình 21 : Nhập xuất sinh viên từ file excel/csv

- *Xem chi tiết sinh viên*




Hình 22 : Xem chi tiết sinh viên

- **Trang thông tin cá nhân**

10:13 Wed, Nov 13 38%

X

Power



Mã tài khoản

Mã tài khoản U001

Họ và tên

Họ và tên Mark Thompson

Tuổi

Tuổi 35

Email

Email tranlop72@gmail.com

Phone

Phone 123-456-7890

Mật khẩu

Mật khẩu

Quyền hạn

Quyền hạn admin

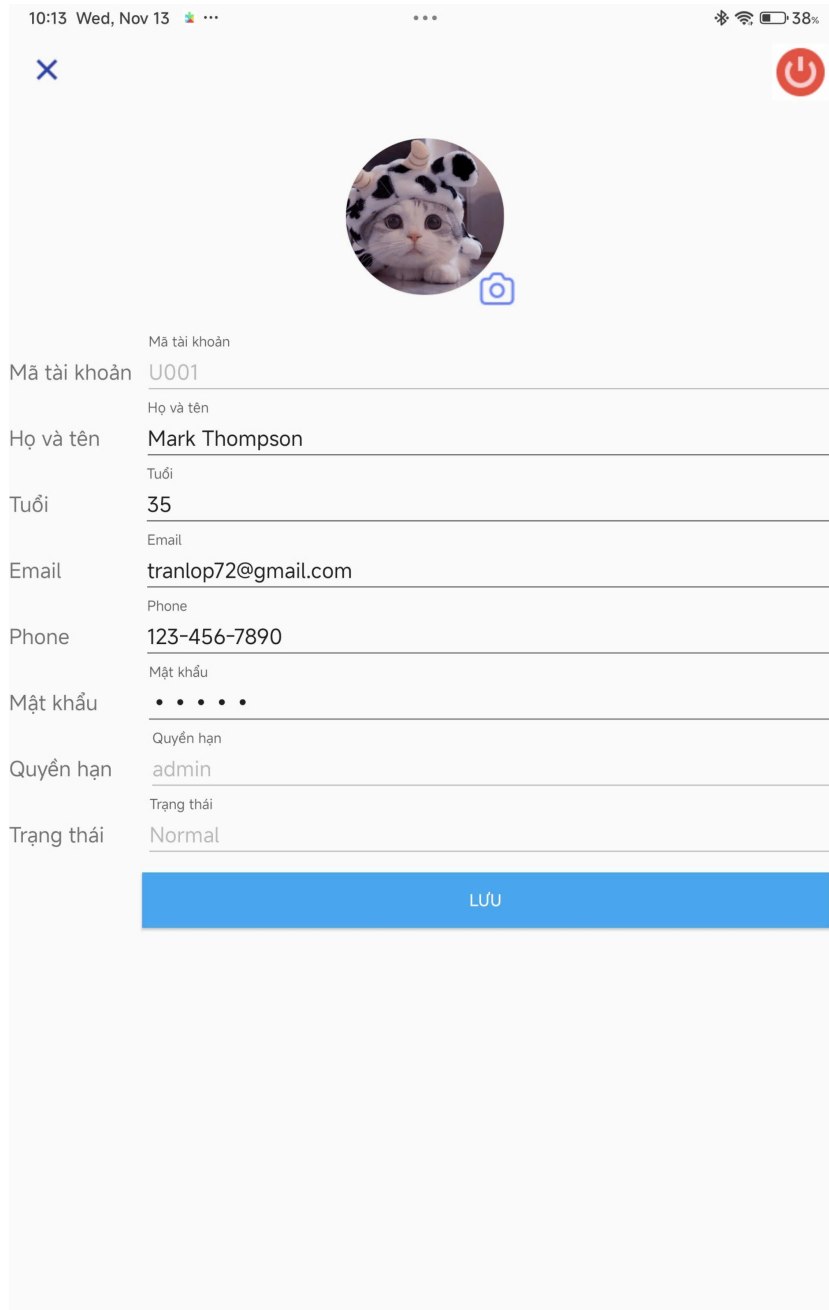
Trạng thái

Trạng thái Normal

SỬA

Hình 23 : Trang thông tin cá nhân

- **Sửa thông tin cá nhân**



10:13 Wed, Nov 13 38%

X

Power

Profile picture

Mã tài khoản U001

Họ và tên Mark Thompson

Tuổi 35

Email tranlop72@gmail.com

Phone 123-456-7890

Mật khẩu • • • • •

Quyền hạn admin

Trạng thái Normal

LƯU

Hình 24 : Sửa thông tin cá nhân

3.3.3 *Link video thuyết trình + demo : [Click here](#)*

3.3.4 *Link source code gitlab : [Click here](#)*

TÀI LIỆU THAM KHẢO

Tiếng Việt

[1] “Wikipedia” Firebase. [Trực tuyến]. Available: <https://en.wikipedia.org/wiki/Firebase>. [Trực tuyến] [Đã truy cập 03-11-2024].

[2] “Phạm Xuân Nam” Tìm hiểu sơ lược về Firebase. [Trực tuyến]. Available: <https://viblo.asia/p/tim-hieu-so-luoc-ve-firebase-Eb85oeOmZ2G>. [Đã truy cập 03-11-2024].

[3] “FPT CLOUD,” Firebase là gì? Ưu nhược điểm & Các dịch vụ của Firebase. [Trực tuyến]. Available: <https://fptcloud.com/firebase-la-gi/>. [Đã truy cập 03-11-2024].

[4] “MATBAO” Firebase là gì? Giải pháp lập trình không cần Backend từ Google [Trực tuyến]. Available : <https://wiki.matbao.net/firebase-la-gi-giai-phap-lap-trinh-khong-can-backend-tu-google/> [Đã truy cập 03-11-2024].