TRƯỜNG ĐẠI HỌC THỦ DẦU MỘT

VIỆN KỸ THUẬT - CÔNG NGHỆ

TIỂU LUẬN MÔN HỌC

PHÁT TRIỂN ỨNG DỤNG DI ĐỘNG ĐA NỀN TẢNG

ĐỀ TÀI

XÂY DỰNG ỨNG DỤNG DI ĐỘNG QUẢN LÝ THÀNH VIÊN CÂU LẠC BỘ

GVHD: ThS. Trần Văn Hữu

SVTH	MSSV
Nguyễn Minh Quân	1824801030153
Nguyễn Thị Phương Thi	1824801030231

Bình Dương, tháng 04 năm 2021

LỜI CẢM ƠN

Đầu tiên, chúng em xin gửi lời cám ơn chân thành đến thầy Hữu - giảng viên hướng dẫn môn phát triển ứng dụng di động đa nền tảng. Xin cám ơn thầy vì đã giảng dạy nhiệt tình, chi tiết để chúng em có đủ kiến thức và vận dụng chúng vào bài báo cáo này.

Do chưa có nhiều kinh nghiệm làm đề tài, cũng như những hạn chế về kiến thức, trong bài tiểu luận chắc sẽ không tránh khỏi những thiếu sót. Rất mong nhận được sự nhận xét, ý kiến đóng góp, phê bình từ phía Thầy để bài báo cáo của chúng em được hoàn thiện hơn.

Lời cuối cùng, em xin kính chúc thầy nhiều sức khỏe, thành công và hạnh phúc.

NHẬN XÉT VÀ CHẤM ĐIỂM CỦA GIẢNG VIÊN

Họ và tên giảng viên: **Trần Văn Hữu**

Đề tài: **Xây dựng ứng dụng di động quản lý thành viên câu lạc bộ**

Nội dung nhận xét:

.....

.....

.....

.....

.....

.....

.....

.....

Điểm:

Bằng số:……

Bằng chữ:……

Bình Dương, ngày 20 tháng 04 năm 2021

GIẢNG VIÊN

Trần Văn Hữu

LỜI NÓI ĐẦU

Trong những năm gần đây, vai trò của các hệ thống thông tin đang được đẩy mạnh trong cuộc sống. Việc ứng dụng công nghệ thông tin vào các hoạt động đời sống đã giúp nâng cao chất lượng công việc. Trong việc quản lý nói chung và việc quản lý thành viên nói riêng, việc ứng dụng công nghệ thông tin là hết sức cần thiết.

Việc ứng dụng công nghệ thông tin vào việc quản lý mang lại rất nhiều lợi ích. Chẳng hạn, có thể lưu trữ lượng thông tin rất lớn trong khoảng không gian rất nhỏ. Không những thế, việc tìm kiếm thông tin sẽ trở nên thuận tiện, dễ dàng và đạt được hiệu quả cao hơn.

Chính vì những ưu điểm trên, nhóm chúng em quyết định chọn đề tài "Xây dựng ứng dụng di động quản lý thành viên câu lạc bộ". Ứng dụng có thể hỗ trợ người dùng trong việc quản lý thành viên.

MỤC LỤC

DANH SÁCH HÌNH ẢNH

DANH SÁCH BẢNG

CHƯƠNG 1: KHẢO SÁT VÀ THU THẬP YÊU CẦU

1. Thông tin cá nhân

1.1. Nguyễn Minh Quân

Lớp: D18PM03

MSSV: 1824801030153

1.2. Nguyễn Thị Phương Thi

Lớp: D18PM04

MSSV: 1824801030231

2. Công nghệ sử dụng trong đề tài

2.1. React Native

2.1.1. React Native là gì?

React Native là một framework cho phép các lập trình viên xây dựng các ứng dụng native mà chỉ sử dụng ngôn ngữ lập trình javascript. React native cho phép bạn xây dựng các ứng dụng trên android và ios chỉ với một ngôn ngữ thống nhất là javascript nhưng mang lại trải nghiệm native app thực sự. Không như các framework hybrid khác (viết một lần triển khai nhiều nơi), React native tập trung vào việc một lập trình viên làm việc hiệu quả trên môi trường đa nền tảng như thế nào.

2.1.2. Cách hoạt động của React Native

Bằng cách tích hợp 2 thread là Main Thread và JS Thread cho ứng dụng mobile. Với Main Thread sẽ đảm nhận vai trò cập nhật giao diện người dùng(UI). Sau đó sẽ xử lý tương tác người dùng. Trong khi đó, JS Thread sẽ thực thi và xử lý code Javascript. Hai luồng này hoạt động độc lập với nhau.

Để tương tác được với nhau hai Thread sẽ sử dụng một Bridge(cầu nối). Cho phép chúng giao tiếp mà không phụ thuộc lẫn nhau, chuyển đổi dữ liệu từ thread này sang thread khác. Dữ liệu từ hai Thread được vận hành khi tiếp nối dữ liệu cho nhau.

2.1.3. Ưu điểm của React Native

- Tối ưu thời gian.
- Hiệu năng ổn định.
- Tiết kiệm chi phí.
- Đội ngũ phát triển ứng dụng không quá lớn.
- Ứng dụng tin cậy, ổn định.
- Xây dựng ứng dụng ít native code nhất cho nhiều hệ điều hành khác nhau.
- Trải nghiệm người dùng tốt hơn khi so sánh với ứng dụng Hybrid.

2.1.4. Nhược điểm của React Native

- Yêu cầu Native code.
- Hiệu năng kém hơn so với Native App.
- Bảo mật chưa thật sự tốt do dùng JS.
- Quản lý bộ nhớ.
- Tùy biến chưa thật sự tốt ở một số module.

2.2. NodeJS

2.2.1. NodeJS là gì?

NodeJS là một mã nguồn được xây dựng dựa trên nền tảng Javascript V8 Engine và cũng là một mã nguồn mở được sử dụng rộng bởi hàng ngàn lập trình viên trên toàn thế giới.

NodeJS có thể chạy trên nhiều nền tảng hệ điều hành khác nhau từ WIndow cho tới Linux, OS X nên đó cũng là một lợi thế. NodeJS cung cấp các thư viện phong phú ở dạng Javascript Module khác nhau giúp đơn giản hóa việc lập trình và giảm thời gian ở mức thấp nhất.

2.2.2. Một số tính năng của NodeJS

- NodeJS có thể gieo nội dung trang động.

- NodeJS có thể tạo, mở, viết, đọc, xóa, và đóng các tập tin trên server.
- NodeJS có thể thu thập thông tin dữ liệu form.
- NodeJS có thể thêm, xóa, thay đổi dữ liệu trong database của bạn.

2.2.3. Tập tin NodeJS có gì?

Tập tin Node.js chứa các tác vụ được thực thi trên các sự kiện nào đó. Một sự kiện điển hình là ai đó cố gắng truy cập 1 cổng (port) trên server. Tập tin Node.js phải bắt đầu trên server trước khi có bất kỳ ảnh hưởng nào.

2.3. Visual Studio Code

2.3.1. Visual Studio Code là gì?

Visual Studio Code là một trình biên tập lập trình code miễn phí dành cho Windows, Linux và macOS, Visual Studio Code được phát triển bởi Microsoft. Nó được xem là một sự kết hợp hoàn hảo giữa IDE và Code Editor.

Visual Studio Code hỗ trợ chức năng debug, đi kèm với Git, có syntax highlighting, tự hoàn thành mã thông minh, snippets, và cải tiến mã nguồn. Nhờ tính năng tùy chỉnh, Visual Studio Code cũng cho phép người dùng thay đổi theme, phím tắt, và các tùy chọn khác.

2.3.2. Một số tính năng của Visual Studio Code

- Hỗ trợ nhiều ngôn ngữ lập trình.
- Hỗ trợ đa nền tảng.
- Cung cấp kho tiện ích mở rộng.
- Hỗ trợ GitIntellisense
- Màn hình đa nhiệm
- Hỗ trợ thiết bị đầu cuối
- Hỗ trợ viết Code
- Lưu trữ dữ liệu dạng phân cấp

- Hỗ trợ web
- Kho lưu trữ an toàn
- Bình luận

2.4. Firebase

2.4.1. Firebase là gì?

Firebase là dịch vụ cơ sở dữ liệu hoạt động trên nền tảng đám mây – cloud. Kèm theo đó là hệ thống máy chủ cực kỳ mạnh mẽ của Google. Chức năng chính là giúp người dùng lập trình ứng dụng bằng cách đơn giản hóa các thao tác với cơ sở dữ liệu.

Cụ thể là những giao diện lập trình ứng dụng API đơn giản. Mục đích nhằm tăng số lượng người dùng và thu lại nhiều lợi nhuận hơn.

Đặc biệt, còn là dịch vụ đa năng và bảo mật cực tốt. Firebase hỗ trợ cả hai nền tảng Android và IOS. Không có gì khó hiểu khi nhiều lập trình viên chọn Firebase làm nền tảng đầu tiên để xây dựng ứng dụng cho hàng triệu người dùng trên toàn thế giới.

2.4.2. Cách thức hoạt động của Firebase

Firebase Realtime Database là gì?

Khi đăng ký một tài khoản trên **Firebase** để tạo ứng dụng, bạn đã có một cơ sở dữ liệu thời gian thực. Dữ liệu bạn nhận được dưới dạng JSON. Đồng thời nó cũng luôn được đồng bộ thời gian thực đến mọi kết nối client.

Đối với các ứng dụng đa nền tảng, tất cả các client đều sử dụng cùng một cơ sở dữ liệu. Nó được tự động cập nhật dữ liệu mới nhất bất cứ khi nào các lập trình viên phát triển ứng dụng. Cuối cùng, tất cả các dữ liệu này được truyền qua kết nối an toàn SSL có bảo mật với chứng nhận 2048 bit.

Trong trường hợp bị mất mạng, dữ liệu được lưu lại ở local. Vì thế khi có mọi sự thay đổi nào đều được tự động cập nhật lên Server của **Firebase**. Bên cạnh

đó, đối với các dữ liệu ở local cũ hơn với Server thì cũng tự động cập nhật để được dữ liệu mới nhất.

Freebase Authentication là gì?

Hoạt động nổi bật của Firebase là xây dựng các bước xác thực người dùng bằng Email, Facebook, Twitter, GitHub, Google. Đồng thời cũng xác thực nặc danh cho các ứng dụng. Hoạt động xác thực có thể giúp thông tin cá nhân của người sử dụng được an toàn và đảm bảo không bị đánh cắp tài khoản.

Firebase Hosting là gì?

Cách thức hoạt động cuối cùng của Firebase được đề cập trong bài viết này là cung cấp các hosting. Hosting được phân phối qua tiêu chuẩn công nghệ bảo mật SSL từ mạng CDN.

2.4.3. Ưu điểm của Firebase

- Tạo tài khoản và sử dụng dễ dàng
- Tốc độ phát triển nhanh
- Nhiều dịch vụ trong một nền tảng
- Được cung cấp bởi Google
- Tập trung vào phát triển giao diện người dùng
- Firebase không có máy chủ
- Học máy (Machine Learning)
- Tạo lưu lượng truy cập
- Theo dõi lỗi
- Sao lưu

2.4.4. Nhược điểm của Firebase

- Không phải là mã nguồn mở

- Người dùng không có quyền truy cập mã nguồn
- Firebase không hoạt động ở nhiều quốc gia
- Chỉ hoạt động với Cơ sở dữ liệu NoSQL
- Truy vấn chậm
- Không phải tất cả các dịch vụ Firebase đều miễn phí
- Firebase khá đắt và giá không ổn định
- Chỉ chạy trên Google Cloud
- Thiếu Dedicated Servers và hợp đồng doanh nghiệp
- Không cung cấp các API GraphQL

3. Khảo sát hiện trạng

3.1. Mô tả bài toán

Hiện nay, hầu hết các câu lạc bộ thường quản lý thông tin bằng sổ sách. Việc này mang lại rất nhiều bất cập. Đối với lượng thông tin lớn, việc lưu trữ sẽ trở thành vấn đề cực kì nan giải.

Thông tin được lưu trữ bằng sổ sách có thể thất lạc hoặc hỏng do các điều kiện khách quan. Không những vậy, việc tìm kiếm thông tin sẽ tốn rất nhiều thời gian. Hơn thế nữa, việc chỉnh sửa thông tin nhiều lần hoặc lượng thông tin chỉnh sửa quá lớn cũng sẽ mang đến khó khăn cho các người quản lý.

Qua quá trình tìm hiểu và khảo sát thì đa phần các câu lạc bộ thường chỉ quản lý bằng cách thông thường. Chính vì vậy, chúng em quyết định xây dựng ứng dụng này để hỗ trợ các quản lý của câu lạc bộ.

3.2. Mô tả yêu cầu

Quản lý có thể đăng nhập để thực hiện các thao tác:

- Quản lý thông tin thành viên: thêm, xóa, sửa, xem
- Quản lý thông tin nhóm: thêm, xóa, sửa, xem

3.3. Các chức năng của bài toán

3.3.1. Yêu cầu chức năng

- Quản lý có thể thêm thông tin thành viên
- Quản lý có thể xóa thông tin thành viên
- Quản lý có thể sửa thông tin thành viên
- Quản lý có thể xem thông tin thành viên
- Quản lý có thể thêm thông tin nhóm
- Quản lý có thể xóa thông tin nhóm
- Quản lý có thể sửa thông tin nhóm
- Quản lý có thể xem thông tin nhóm

3.3.2. Yêu cầu phi chức năng

- Tính hiệu dụng:

Ứng dụng tổ chức theo mô hình giống như một trang quản lý thông tin với các chức năng và giao diện đơn giản, dễ sử dụng.

- Phương án xây dựng:

Ngôn ngữ sử dụng: JavaScript

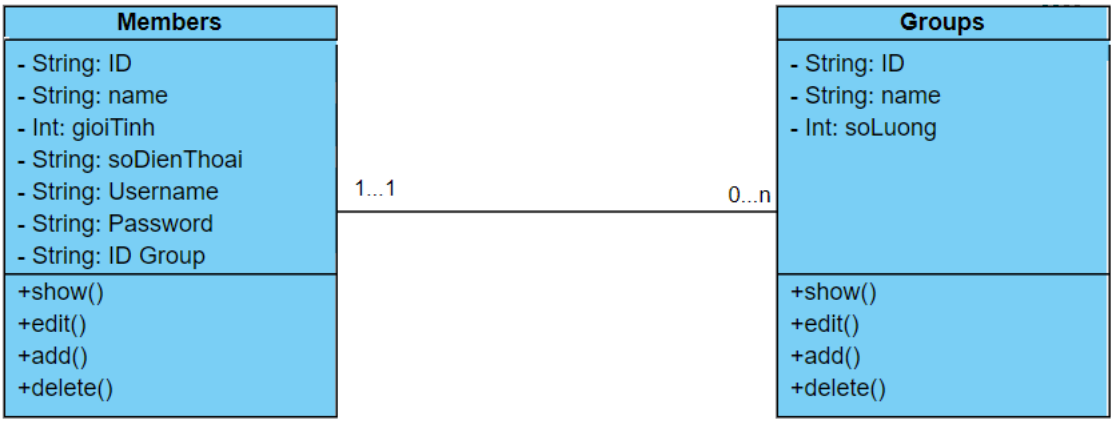
Hệ quản trị cơ sở dữ liệu: Firebase

- Hệ điều hành:

Đa nền tảng (Android và iOS)

CHƯƠNG 2: PHÂN TÍCH THIẾT KẾ HỆ THỐNG

1. Biểu đồ lớp



Hình 2.1. Biểu đồ lớp

2. Biểu đồ UseCase

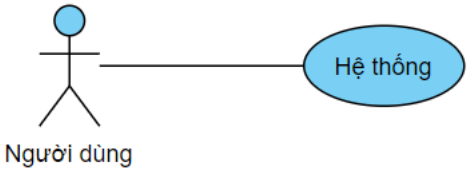
2.1. Danh sách Actor

Một actor hay tác nhân ngoài là một vai trò của một hay nhiều người hay vật thể trong sự tương tác với hệ thống

STT	Tác nhân	Mô tả
1	Người dùng	Là người sử dụng hệ thống, có quyền truy cập vào hệ thống

Bảng 2.2.2.1. Danh sách Actor

2.2. Biểu đồ hệ thống



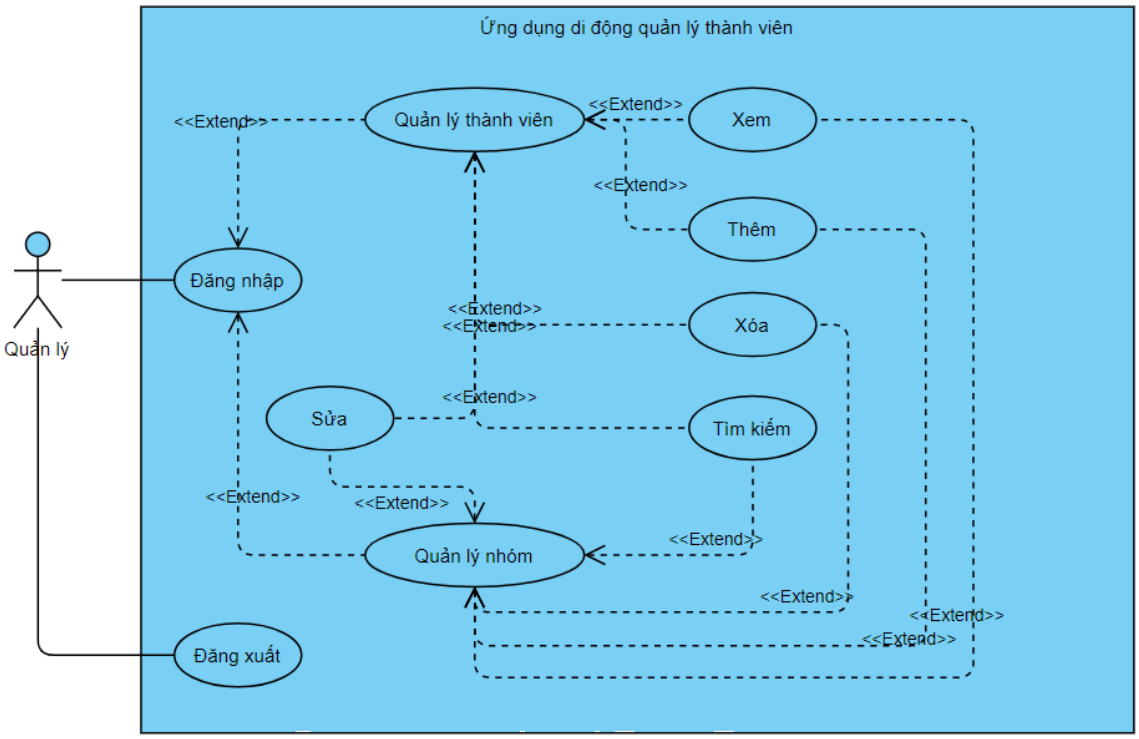
Hình 2.2.2.2. Biểu đồ hệ thống

2.3. Danh sách các UseCase

STT	Tên UseCase	Ý nghĩa	Actor
1	Đăng nhập	Cho phép người dùng truy cập vào hệ thống	Quản lý
2	Đăng xuất	Cho phép người dùng đăng xuất ra khỏi hệ thống	Quản lý
3	Xem thông tin	Cho phép người dùng biết được thông tin chi tiết	Quản lý
4	Thêm thông tin	Cho phép người dùng thêm thông tin vào hệ thống	Quản lý
5	Xóa thông tin	Cho phép người dùng xóa thông tin khỏi hệ thống	Quản lý
6	Sửa thông tin	Cho phép người dùng sửa thông tin hiện có trong hệ thống	Quản lý
7	Tìm thông tin	Cho phép người dùng tìm kiếm thông tin	Quản lý

Bảng 2.2.2.3. Danh sách các UseCase

2.4. Biểu đồ UseCase toàn hệ thống



Hình 2.2.4. Biểu đồ UseCase toàn hệ thống

2.5. Đặc tả UseCase

2.5.1. UseCase Đăng nhập

Tên UseCase	Đăng nhập
Actor chính	Quản lý
Actor phụ	Hệ thống
Tiền điều kiện	Đã có tài khoản trong hệ thống
Hậu điều kiện	Vào trang chủ của hệ thống
Mô tả chung	Cho phép người dùng truy cập vào hệ thống
Dòng sự kiện chính	B1: Người dùng nhập tên tài khoản B2: Người dùng nhập mật khẩu B3: Người dùng chọn đăng nhập
Dòng sự kiện phụ	Thông báo sai thông tin

Bảng 2.2.5.1. Đặc tả UseCase Đăng nhập

2.5.2. UseCase Đăng xuất

Tên UseCase	Đăng xuất
Actor chính	Quản lý
Actor phụ	Hệ thống
Tiền điều kiện	Đã đăng nhập vào hệ thống
Hậu điều kiện	Quay về màn hình đăng nhập
Mô tả chung	Cho phép người dùng đăng xuất khỏi hệ thống
Dòng sự kiện chính	B1: Người dùng chọn đăng xuất
Dòng sự kiện phụ	Không có

Bảng 2.2.5.2. Đặc tả UseCase Đăng xuất

2.5.3. UseCase Xem thông tin

Tên UseCase	Xem thông tin
Actor chính	Quản lý
Actor phụ	Hệ thống
Tiền điều kiện	Đã đăng nhập vào hệ thống
Hậu điều kiện	Hiển thị thông tin chi tiết
Mô tả chung	Cho phép người dùng xem thông tin chi tiết của đối tượng
Dòng sự kiện chính	B1: Người dùng chọn mục cần xem thông tin B2: Người dùng nhấn giữ vào đối tượng cần xem thông tin B3: Người dùng chọn xem chi tiết
Dòng sự kiện phụ	Không có

Bảng 2.2.5.3. Đặc tả UseCase Xem thông tin

2.5.4. UseCase Thêm thông tin

Tên UseCase	Thêm thông tin
Actor chính	Quản lý
Actor phụ	Hệ thống
Tiền điều kiện	Đã đăng nhập vào hệ thống
Hậu điều kiện	Thông báo đã thêm thông tin
Mô tả chung	Cho phép người dùng thêm thông tin vào hệ thống
Dòng sự kiện chính	B1: Người dùng chọn mục thêm B2: Người dùng nhập các thông tin cần thiết B3: Người dùng chọn thêm
Dòng sự kiện phụ	Thông báo thông tin không hợp lệ

Bảng 2.2.5.4. Đặc tả UseCase Thêm thông tin

2.5.5. UseCase Xóa thông tin

Tên UseCase	Xóa thông tin
Actor chính	Quản lý
Actor phụ	Hệ thống
Tiền điều kiện	Đã đăng nhập vào hệ thống
Hậu điều kiện	Thông báo đã xóa thông tin khỏi hệ thống
Mô tả chung	Cho phép người dùng xóa thông tin được lưu trữ trong hệ thống
Dòng sự kiện chính	B1: Người dùng chọn thông tin cần xóa B2: Người dùng chọn xóa B3: Người dùng xác nhận muốn xóa
Dòng sự kiện phụ	Thông báo thông tin không xóa được

Bảng 2.2.5.5. Đặc tả UseCase Xóa thông tin

2.5.6. UseCase Sửa thông tin

Tên UseCase	Sửa thông tin
Actor chính	Quản lý
Actor phụ	Hệ thống
Tiền điều kiện	Đã đăng nhập vào hệ thống
Hậu điều kiện	Thông báo thông tin đã được chỉnh sửa
Mô tả chung	Cho phép người dùng sửa thông tin trong hệ thống
Dòng sự kiện chính	B1: Người dùng chọn thông tin cần sửa B2: Người dùng nhập thông tin cần thiết B3: Người dùng chọn sửa
Dòng sự kiện phụ	Thông báo thông tin không hợp lệ

Bảng 2.2.5.6. Đặc tả UseCase Sửa thông tin

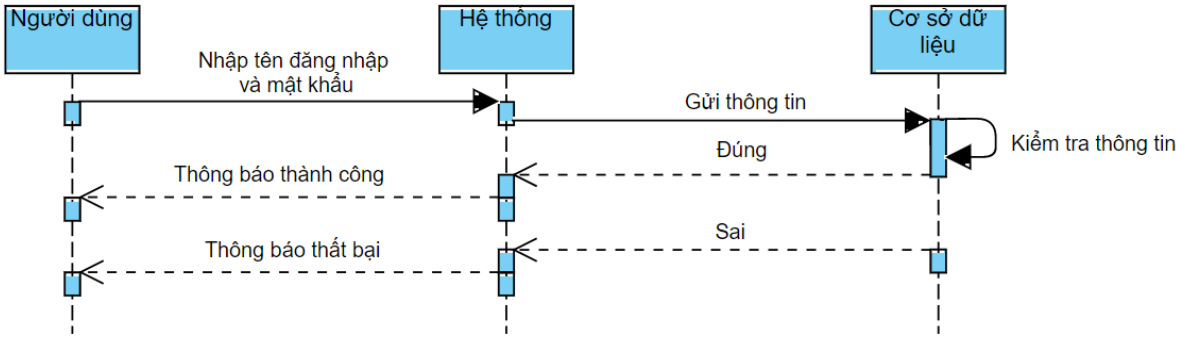
2.5.7. UseCase Tìm thông tin

Tên UseCase	Tìm thông tin
Actor chính	Quản lý
Actor phụ	Hệ thống
Tiền điều kiện	Đã đăng nhập vào hệ thống
Hậu điều kiện	Hiển thị thông tin đã tìm được
Mô tả chung	Cho phép người dùng tìm kiếm các thông tin được lưu trong hệ thống
Dòng sự kiện chính	B1: Người dùng nhập thông tin vào ô tìm kiếm B2: Người dùng chọn tìm kiếm
Dòng sự kiện phụ	Thông báo thông tin không tồn tại

Bảng 2.2.5.7. Đặc tả UseCase Tìm thông tin

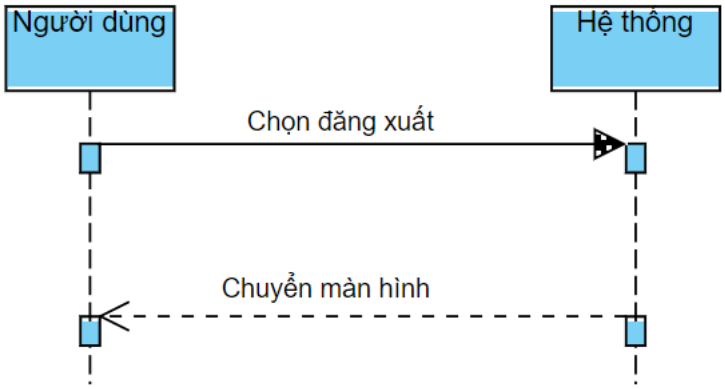
3. Biểu đồ tuần tự

3.1. Biểu đồ tuần tự Đăng nhập



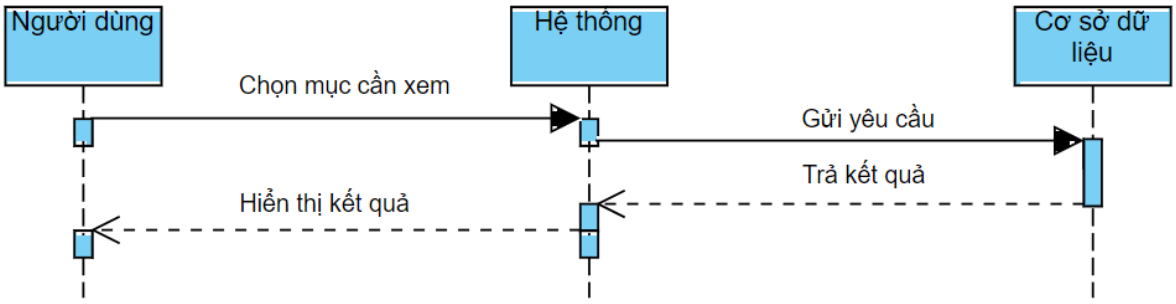
Hình 2.3.1. Biểu đồ tuần tự Đăng nhập

3.2. Biểu đồ tuần tự Đăng xuất



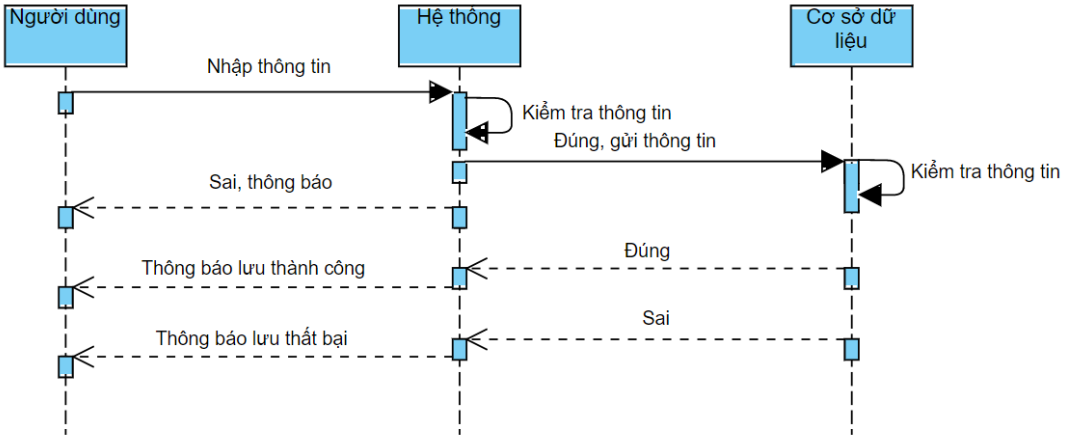
Hình 2.3.2. Biểu đồ tuần tự Đăng xuất

3.3. Biểu đồ tuần tự Xem thông tin



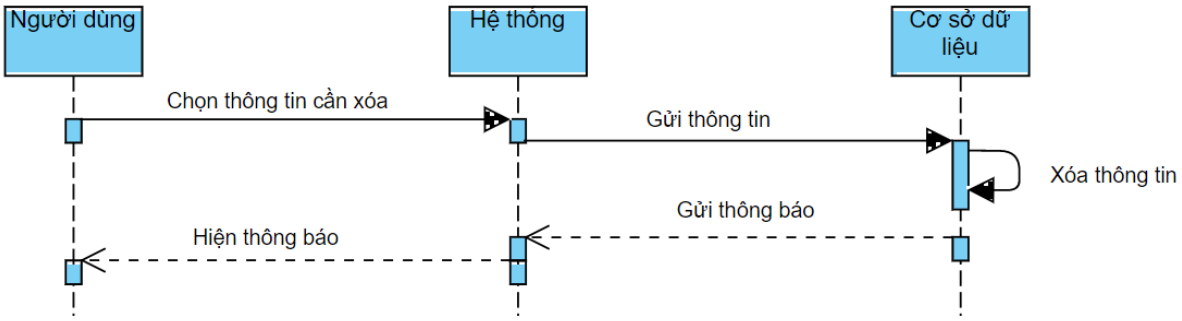
Hình 2.3.3. Biểu đồ tuần tự Xem thông tin

2.3. Biểu đồ tuần tự Thêm thông tin



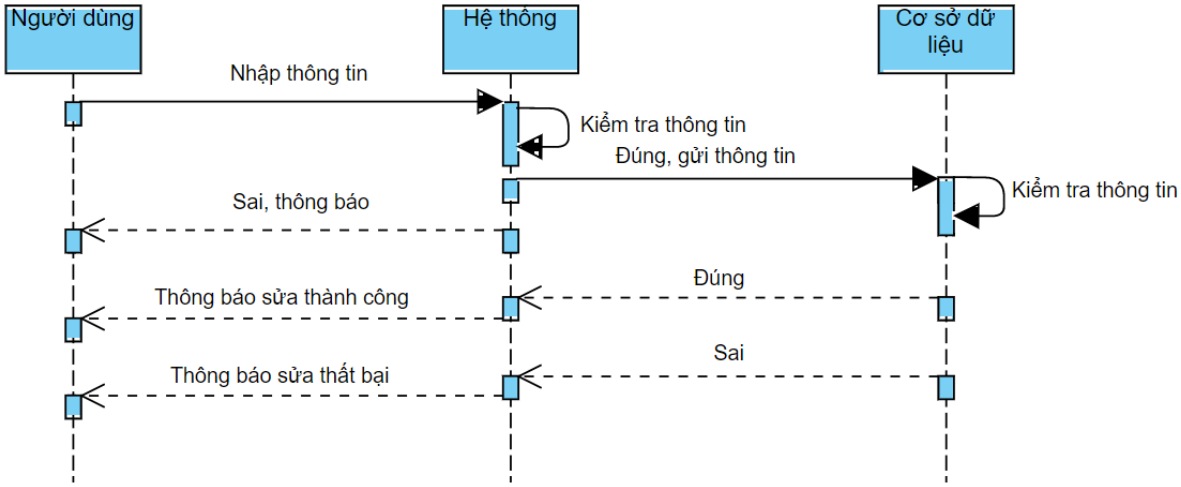
Hình 2.3.4. Biểu đồ tuần tự Thêm thông tin

3.5. Biểu đồ tuần tự Xóa thông tin



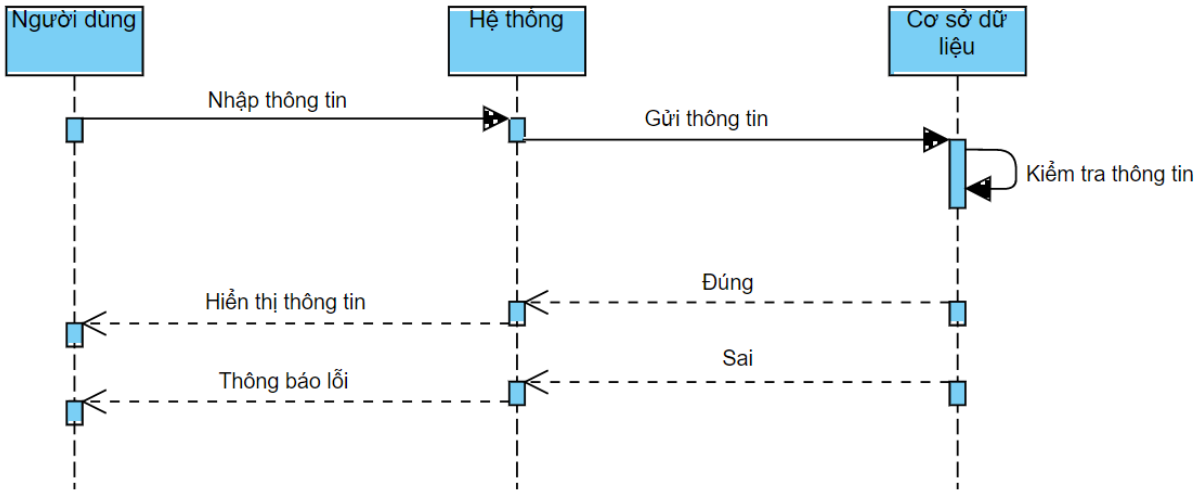
Hình 2.3.5. Biểu đồ tuần tự Xóa thông tin

3.6. Biểu đồ tuần tự Sửa thông tin



Hình 2.3.6. Biểu đồ tuần tự Sửa thông tin

3.7. Biểu đồ tuần tự Tìm thông tin



Hình 2.3.7. Biểu đồ tuần tự Tìm thông tin

CHƯƠNG 3: THIẾT KẾ CƠ SỞ DỮ LIỆU

1. Thiết kế Database

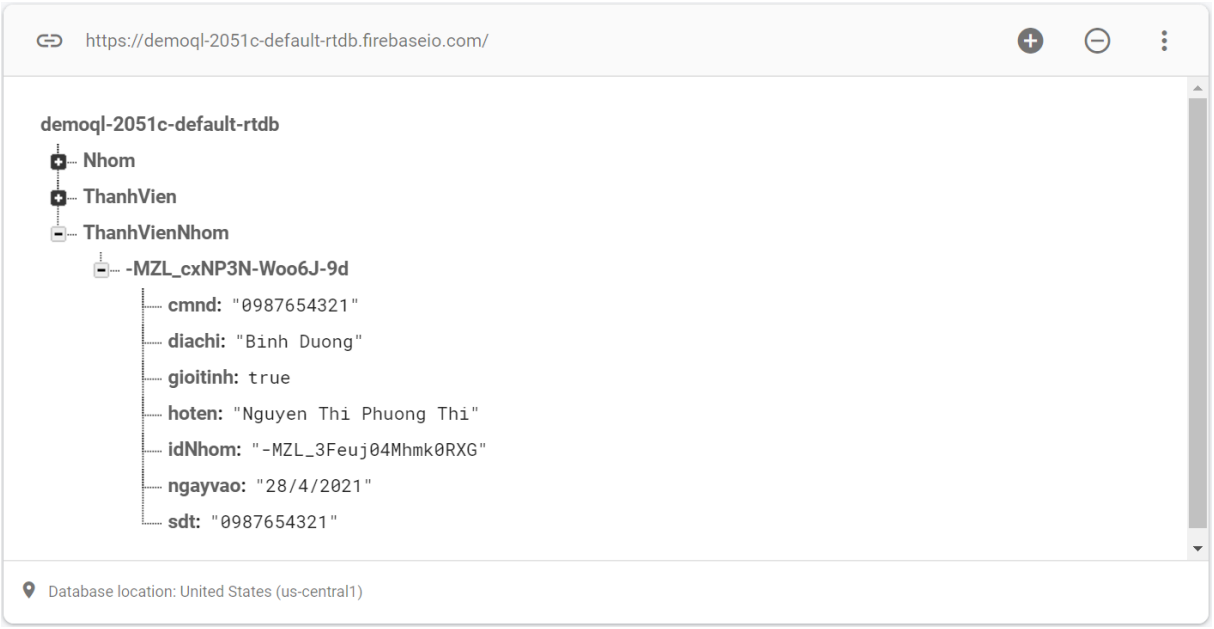
1.1. Cấu trúc Nhóm

Bảng 2.5.1. Cấu trúc Nhóm

1.2. Cấu trúc Thành viên quản trị

Bảng 2.5.2. Cấu trúc Thành viên quản trị

1.3. Cấu trúc Thành viên



Bảng 2.5.3. Cấu trúc Thành viên nhóm

CHƯƠNG 4: CÀI ĐẶT ỨNG DỤNG

1. Giao diện các màn hình

1.1. Giao diện Màn hình đăng nhập

Hình 4.1. Giao diện Màn hình đăng nhập

Chức năng: Cho phép người dùng nhập thông tin đăng nhập để truy cập và hệ thống

1.2. Giao diện Màn hình chính

Hình 4.2. Giao diện Màn hình chính

Chức năng: Hiển thị cho người dùng cái nhìn tổng quan về chương trình

1.3. Giao diện Màn hình danh sách quản trị viên

Hình 4.3. Giao diện Màn hình danh sách quản trị viên

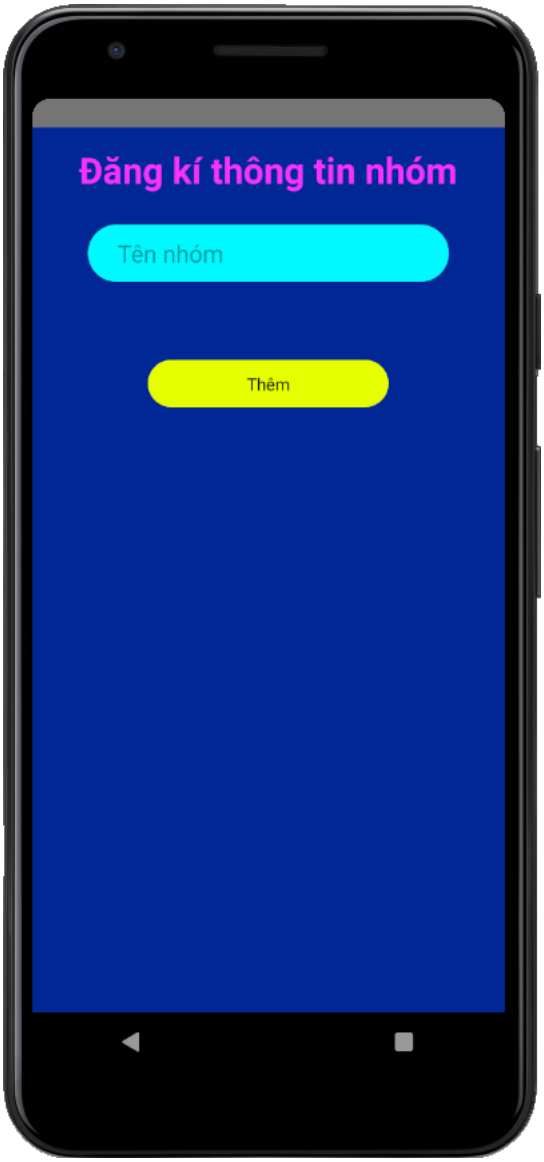
Chức năng: Cung cấp thông tin của các quản trị viên cho người dùng biết

1.4. Giao diện Màn hình danh sách nhóm

Hình 4.4. Giao diện Màn hình danh sách nhóm

Chức năng: Cung cấp thông tin tất cả các nhóm có trong hệ thống

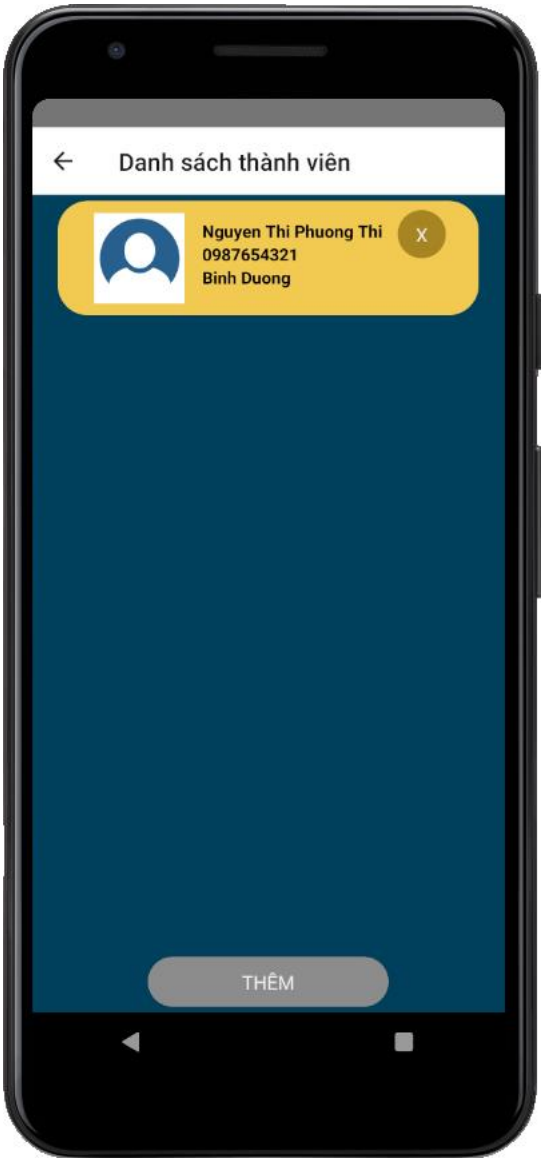
1.5. Giao diện Màn hình thêm thông tin nhóm



Hình 4.5. Giao diện Màn hình thêm thông tin nhóm

Chức năng: Cho phép người dùng tạo nhóm mới

1.6. Giao diện Màn hình danh sách thành viên



Hình 4.6. Giao diện Màn hình danh sách thành viên

Chức năng: Cung cấp thông tin thành viên có trong nhóm

1.7. Giao diện Màn hình thêm thông tin thành viên

Hình 4.7. Giao diện Màn hình thêm thông tin thành viên

Chức năng: Cho phép người dùng thêm thành viên mới vào nhóm

1.8. Giao diện Màn hình đăng kí

Hình 4.8. Giao diện Màn hình đăng kí

Chức năng: Cho phép người dùng đăng kí tài khoản để trở thành quản trị viên

2. Cài đặt thư viện

2.1. Yarn

> npm install yarn

2.2. Navigation

> yarn add @react-navigation/native

> npx react-native link

> yarn add react-native-reanimated react-native-gesture-handler react-native-screens react-native-safe-area-context @react-native-community/masked-view

> npx react-native link

> yarn add @react-navigation/stack

> npx react-native link

2.3. Community

> yarn add @react-native-community/checkbox

> npx react-native link

> yarn add @react-native-community/datetimepicker

> npx react-native link

2.4. Firebase

> yarn add @react-native-firebase/database

> npx react-native link

> yarn add @react-native-firebase/app

> npx react-native link

3. Cài đặt chương trình

3.1.Login.js

import React, {useEffect, useState} from 'react'

```
import
{View,Image,Text,ImageBackground,StyleSheet,TextInput,TouchableOpacity,
Alert}
from 'react-native'
import database from '@react-native-firebase/database'
const Login = ({navigation}) => {
    const [data, setdata] = useState([])
    const [userName, setUsername] = useState('')
    const [passWord, setPassword] = useState('')
    useEffect(() => {
        const getData = async () => {
            try {
                await database().ref("/ThanhVien")
                    .on('value', snapshot => {
                        let arr = [];
                        snapshot.forEach(element => {
                            let temp = {
                                id: element.key,
                                hoTen: element.val().hoTen,
                                soDienThoai: element.val().soDienThoai,
                                tenDangNhap: element.val().tenDangNhap,
                                matKhau: element.val().matKhau,
                                maNhom: element.val().maNhom,
```

```
                    chucVu: element.val().chucVu
                }
                arr.push(temp)
            })
            setdata(arr)
        });
    }
    catch (e) {
        Alert.alert("Lỗi: " + e);
    }
  }
  getData();
}, [])
const nen = require('../img/background.jpg');
const logo = require('../img/logo-frame.png');
const dangnhap = ()=>{
  if(userName===''){
    Alert.alert("Tên đăng nhập trống.")
    return
  }
  if(passWord ===''){
    Alert.alert("Mật khẩu trống.")
```

```
        return
    }
    let user=data.find(value=>{
        if(value.tenDangNhap===userName && value.matKhau===passWord){
            return value;
        }
    })
    if(user != null){
        navigation.replace('Home', {user: user})
    }
    else{
        Alert.alert("Thông tin đăng nhập không hợp lệ.")
    }
}
const dangky=()=>{
    navigation.navigate('Signup',{data: data})
}
const quenmatkhau=()=>{
    Alert.alert("Chức năng trong quá trình hoàn thiện.")
}
return (
    <ImageBackground source={nen} style={styles.nen}>
```

```
        <View>
            <Image source={logo} style={styles.logo} />
            <TextInput style={styles.input}
                placeholder='Tên đăng nhập'
                placeholderTextColor='#003f5c'
                onChangeText={text => setUsername(text)}
            />
            <TextInput style={styles.input}
                placeholder='Mật khẩu'
                placeholderTextColor='#003f5c'
                secureTextEntry={true}
                onChangeText={text => setPassword(text)}
            />
            <TouchableOpacity style={{ alignItems: 'center' }}
onPress={quenmatkhau}>
                <Text style={styles.forgot}>Quên mật khẩu?</Text>
            </TouchableOpacity>
            <TouchableOpacity onPress={dangnhap} style={styles.loginBtn}>
                <Text style={styles.loginText}>Đăng nhập</Text>
            </TouchableOpacity>
            <TouchableOpacity style={{ alignItems: 'center' }}
onPress={dangky}>
                <Text style={styles.loginText}>Đăng ký</Text>
```

```
            </TouchableOpacity>
          </View>
        </ImageBackground>
    )
}
export default Login
const styles = StyleSheet.create({
    nen: {
        flex: 1,
        resizeMode: 'contain',
        justifyContent: 'center',
        alignItems: 'center'
    },
    logo: {
        resizeMode: 'contain',
        width: 200,
        height: 200
    },
    input: {
        backgroundColor: "#0ED3DA",
        borderRadius: 25,
        height: 55,
```

```
    marginBottom: 20,
    justifyContent: "center",
    padding: 20
},
forgot: {
    color: "red",
    fontSize: 11
},
loginBtn: {
    backgroundColor: "#0D44F6",
    borderRadius: 25,
    height: 50,
    alignItems: "center",
    justifyContent: "center",
    marginTop: 40,
    marginBottom: 10
},
hinhmuc: {
    width: 75,
    height: 75
},
loginText:{
```

```
        color: "white"
    }
})
```

3.2. Signup.js

```
import React, { useState } from 'react'
import { StyleSheet, Text, View, TouchableOpacity, TextInput, Alert } from 'react-native'
import database from '@react-native-firebase/database'
const Signup = ({ navigation, route }) => {
    const data=route.params.data
    const [hoTen, setHoten] = useState('')
    const [soDienThoai, setSodienthoai] = useState('')
    const [tenDangNhap, setTenDangNhap] = useState('')
    const [matKhau, setMatkhau] = useState('')
    const [nhapLaiMatKhau, setNhaplaimatkhau] = useState('')
    const dangky = async () => {
        if(hoTen === '')
        {
            Alert.alert("Họ tên không được để trống.")
            return
        }
        if(soDienThoai === '')
        {
```

```
    Alert.alert("Số điện thoại không được để trống.")
    return
}
if(tenDangNhap === '')
{
    Alert.alert("Tên đăng nhập không được trống.")
    return
}
if(matKhau === '')
{
    Alert.alert("Mật khẩu không được để trống.")
    return
}
if(nhapLaiMatKhau === '')
{
    Alert.alert("Nhập lại mật khẩu không được để trống.")
    return
}
if(matKhau!=nhapLaiMatKhau){
    Alert.alert("Nhập lại mật khẩu không khớp.")
    return
}
```

```
let taiKhoan=data.find(value=>{
    if(value.tenDangNhap===tenDangNhap){
        return value;
    }
})
if(taiKhoan!=null){
    Alert.alert("Tài khoản đã tồn tại.")
    return
}
let user = {
    hoTen: hoTen,
    soDienThoai: soDienThoai,
    tenDangNhap: tenDangNhap,
    matKhau: matKhau,
    chucVu: 3
}
try {
    await database().ref("/ThanhVien").push(user);
    Alert.alert("Đăng ký thành công.")
    navigation.navigate('Login')
}
catch (e) {
```

```
            Alert.alert('Loi' + e)
        }
    }

    return (
        <View style={styles.tong1}>
            <View style={styles.vtieude}>
                <Text style={styles.tieude}>Đăng kí thành viên</Text>
            </View>
            <View style={styles.tong2}>
                <TextInput style={styles.txtInput} placeholder={"Họ và tên"}
onChangeText={(text) => { setHoten(text) }} />
            </View>
            <View style={styles.tong2}>
                <TextInput style={styles.txtInput} placeholder={"Số điện thoại"}
onChangeText={(text) => { setSodienthoai(text) }} />
            </View>
            <View style={styles.tong2}>
                <TextInput style={styles.txtInput} placeholder={"Tên đăng nhập"}
onChangeText={(text) => { setTenDangNhap(text) }} />
            </View>
            <View style={styles.tong2}>
```

```
            <TextInput style={styles.txtInput} placeholder={"Mật khẩu"}
secureTextEntry={true} onChangeText={(text) => { setMatkhau(text) }} />
        </View>
        <View style={styles.tong2}>
            <TextInput style={styles.txtInput} placeholder={"Nhập lại mật
khẩu"} secureTextEntry={true} onChangeText={(text) =>
{ setNhaplaimatkhau(text) }} />
        </View>
        <View style={styles.vbtn}>
            <TouchableOpacity style={styles.btn} onPress={dangky}>
                <Text style={styles.txtbtn}>Đăng kí</Text>
            </TouchableOpacity>
        </View>
    </View>
  )
}
export default Signup
const styles = StyleSheet.create({
  tong1: {
    flex: 1,
    backgroundColor: '#002695'
  },
  vtieude: {
```

```
    margin: 15,
    alignItems: 'center'
},
tong2: {
    flexDirection: 'row',
    justifyContent: 'center'
},
tieude: {
    fontSize: 30,
    fontWeight: 'bold',
    color: '#F633FF'
},
txtInput: {
    backgroundColor: '#0CF9FF',
    justifyContent: 'flex-end',
    width: 300,
    fontSize: 20,
    paddingTop: 10,
    paddingBottom: 10,
    paddingLeft: 25,
    paddingRight: 25,
    margin: 10,
```

```
    borderRadius: 25
  },
  vbtn: {
    marginTop: 50,
    flex: 1,
    alignItems: 'center'
  },
  btn: {
    backgroundColor: '#E6FF0C',
    width: 200,
    justifyContent: 'center',
    alignItems: 'center',
    margin: 2.5,
    margin: 5,
    padding: 10,
    borderRadius: 25,
  }
})
```

3.3. Home.js

```
import React from 'react'
import { Text, View, StyleSheet, Image, TouchableOpacity, Alert} from 'react-native'
const Home = ({navigation, route}) => {
```

```
    const USER= route.params.user
    const logo = require('../img/manga.png')
    const thanhvien = require('../img/member.png')
    const nhom = require('../img/group.jpg')
    var today = new Date()
    var date = today.getDate() + '/' + (today.getMonth() + 1) + '/' +
today.getFullYear();
    const Thoat=()=>{
        Alert.alert(
            "Xác nhận đăng xuất!",
            "Bạn có muốn đăng xuất?",
            [
                {
                    text: "Cancel"
                },
                { text: "OK", onPress: () => navigation.replace('Login')}
            ]
        );

    }
    return (
        <View style={styles.tong}>
                <Image source={logo} style={styles.logo}/>
```

```
            <View style={{marginTop: 25}}>
               <View style={styles.row}>
                  <View style={styles.muc}>
                     <TouchableOpacity onPress={() =>
{ navigation.navigate("QLNhom") }} style={styles.muc}>
                        <Image source={nhom} style={styles.hinhmuc} />
                        <Text style={styles.txtmuc}>Nhóm</Text>
                     </TouchableOpacity>
                  </View>
                  <View style={styles.muc}>
                     <TouchableOpacity onPress={() =>
{ navigation.navigate("QLThanhVien") }} style={styles.muc}>
                        <Image source={thanhvien} style={styles.hinhmuc} />
                        <Text style={styles.txtmuc}>Thành viên quản trị</Text>
                     </TouchableOpacity>
                  </View>
               </View>
            </View>
            <View style={{height: 180}}></View>
            <TouchableOpacity style={styles.btnThoat} onPress={Thoat}>
               <Text>Đăng xuất tài khoản</Text>
            </TouchableOpacity>
            <Text style={styles.txtmuc}>Thông tin quản trị viên </Text>
```

```
            <Text style={styles.txtmuc}>Họ và tên: {USER.hoTen}</Text>
            <Text style={styles.txtmuc}>Số điện thoại:
{USER.soDienThoai}</Text>
            <Text style={styles.txtmuc}>Hôm nay: {date}</Text>
        </View>
    )
}
export default Home
const styles = StyleSheet.create({
    tong: {
        flex: 1,
        backgroundColor: '#003f5c',
        justifyContent: 'flex-start',
        alignItems: 'center'
    },
    logo: {
        width: 250,
        height: 250,
        resizeMode: 'stretch'

    },
    row: {
        flexDirection: 'row'
```

```
},
muc: {
    justifyContent: 'center',
    alignItems: 'center',
    padding: 10
},
hinhmuc: {
    width: 75,
    height: 75
},
txtmuc: {
    color: '#fff'
},
btnThoat:{
        backgroundColor: '#8C8C8C',
        width: 350,
        justifyContent: 'center',
        alignItems: 'center',
        margin: 5,
        padding: 10,
        borderRadius: 25,
}
```

```
})
```

3.4. QLThanhVien.js

```
import React, { useEffect, useState } from 'react'
import { StyleSheet, Text, View, FlatList, TouchableOpacity, Alert } from 'react-native'
import database from '@react-native-firebase/database'
const QLThanhVien = ({ navigation }) => {
    const [data, setdata] = useState([])
    useEffect(() => {
        const getData = async () => {
            try {
                await database().ref("/ThanhVien")
                    .on('value', snapshot => {
                        let arr = [];
                        snapshot.forEach(element => {
                            let temp = {
                                id: element.key,
                                hoTen: element.val().hoTen,
                                soDienThoai: element.val().soDienThoai,
                                tenDangNhap: element.val().tenDangNhap,
                                matKhau: element.val().matKhau,
                                maNhom: element.val().maNhom,
                                chucVu: element.val().chucVu
```

```
                    }
                    arr.push(temp)
                })
                setdata(arr)
            });
        }
        catch (e) {
            Alert.alert("Lỗi:",e);
        }
    }
    getData();
}, []);
const XoaThanhVien = async (id) => {
    await database().ref(`/ThanhVien/${id}`).remove()
}
const chitiet = (id) => {
    let thongtin=data.find(value=>{
        if(value.id===id){
            return value;
        }
    })
    if(thongtin.chucVu === 1)
```

```
    {
        Alert.alert("Trưởng ban quản trị")
    }
    else    if(thongtin.chucVu === 2)
    {
        Alert.alert("Phó ban quản trị")
    }
    else
    {
        Alert.alert("Thành viên ban quản trị")
    }
}
const xoa = (id) => {
    try {
        Alert.alert(
            "Xác nhận xóa thành viên!!!",
            "Bạn có muốn xóa quản trị viên này?",
            [
                {
                    text: "Cancel"
                },
                { text: "OK", onPress: () => XoaThanhVien(id) }
```

```
            ]
        );
    }
    catch (e) {
        Alert.alert('Err:',e)
    }
  }
  const renderItem = (item) => {
    return (
        <TouchableOpacity style={styles.item1} onPress={() =>
chitiet(item.id)}>
            <Text style={styles.tenitem}>{item.hoTen}</Text>
            <Text style={styles.tenitem}>{item.soDienThoai}</Text>
            <View style={styles.listCN}>
            <TouchableOpacity style={styles.btnitem} onPress={() =>
xoa(item.id)}>
              <Text style={styles.txtitem}>Xóa </Text>
            </TouchableOpacity>
            </View>
        </TouchableOpacity>
    )

  }
```

```
    return (
        <View style={styles.tong1}>
            <FlatList style={styles.list}
                data={data}
                renderItem={({ item }) => renderItem(item)}
                keyExtractor={item => item.id}
            />
        </View>
    )
}
export default QLThanhVien
const styles = StyleSheet.create({
    tong1:{
        flex: 1,
        justifyContent: 'center',
        alignItems: 'center',
        backgroundColor: '#003f5c'
    },
    tieude:{
        fontSize: 30,
        fontWeight: 'bold',
        color: '#F2D43D'
```

```
},

list:{

},
listCN:{
    flexDirection: 'row'
},
listHT:{
    alignItems: 'flex-start'
},
item:{
    backgroundColor:'#F2CA52',
    margin: 5,
    justifyContent: 'center',
    alignItems: 'center',
    paddingLeft: 25,
    paddingRight: 25,
    paddingTop: 5,
    paddingBottom: 5,
    borderRadius: 25
},
```

```
item1:{
    backgroundColor:'#F2F2F2',
    margin: 5,
    justifyContent: 'center',
    alignItems: 'center',
    paddingLeft: 25,
    paddingRight: 25,
    paddingTop: 5,
    paddingBottom: 5,
    borderRadius: 25
},
tenitem:{
    fontSize: 24,
    fontWeight: 'bold',
    color: '#8C8C8C'
},
btnitem:{
    width: 100,
    justifyContent:'center',
    alignItems: 'center',
    backgroundColor:'#A68524',
    margin: 2.5,
```

```
        padding: 3.5,
        borderRadius: 25
    },
    txtitem:{
        fontSize: 16,
        color: '#F2F2F2'
    },
    txtitem1:{
        fontSize: 16,
        color: '#8C8C8C'
    },
    btn:{
        backgroundColor: '#8C8C8C',
        width: 200,
        justifyContent:'center',
        alignItems: 'center',
        margin: 2.5,
        margin: 5,
        padding: 10,
        borderRadius: 25
    },
})
```

3.5. QLNhom.js

```
import React, { useEffect, useState } from 'react'
import { StyleSheet, Text, View, FlatList, TouchableOpacity, Alert } from 'react-native'
import database from '@react-native-firebase/database'
const QLPhong = ({ navigation }) => {
    const [data, setdata] = useState([])
    useEffect(() => {
        const getData = async () => {
            try {
                await database().ref("/Nhom")
                    .on('value', snapshot => {
                        let arr = [];
                        snapshot.forEach(element => {
                            let temp = {
                                id: element.key,
                                tenNhom: element.val().tenNhom,
                                soLuong: element.val().soLuong
                            }
                            arr.push(temp)
                        })
                        setdata(arr)
                    });
```

```
        }
        catch (e) {
            Alert.alert("Lỗi:",e);
        }
    }
    getData();
}, []);
const them = () => {
    navigation.navigate('AddNhom', {data: data})
}
const XoaNhom = async (id) => {
    await database().ref(`/Nhom/${id}`).remove()
}
const DSThanhVienNhom = (value) => {
    navigation.navigate("QLThanhVienNhom", {value: value})
}
const xoa = (id) => {
    try {
        Alert.alert(
            "Xác nhận xóa nhóm?",
            "Bạn có muốn xóa nhóm này hay không ?",
            [
```

```
                {
                    text: "Cancel"
                },
                { text: "OK", onPress: () => XoaNhom(id) }
            ]
        );
    }
    catch (e) {
        Alert.alert('Err:',e)
    }
  }
  const renderItem = (item) => {
    return (
        <TouchableOpacity style={styles.item} onPress={() =>
DSThanhVienNhom(item)}>
            <Text style={styles.tenitem}>{item.tenNhom}</Text>
            <View style={styles.listCN}>
            <TouchableOpacity style={styles.btnitem}>
                <Text style={styles.txtitem}>Số lượng: {item.soLuong}</Text>
            </TouchableOpacity>
            <TouchableOpacity style={styles.btnitem} onPress={() =>
DSThanhVienNhom(item)}>
                <Text style={styles.txtitem}>Chi tiết</Text>
```

```
            </TouchableOpacity>
            <TouchableOpacity style={styles.btnitem} onPress={() =>
xoa(item.id)}>
                <Text style={styles.txtitem}>Xóa</Text>
            </TouchableOpacity>
            </View>
        </TouchableOpacity>
      )

  }
  return (
    <View style={styles.tong1}>
      <FlatList style={styles.list}
        data={data}
        renderItem={({ item }) => renderItem(item)}
        keyExtractor={item => item.id}
      />
      <TouchableOpacity style={styles.btn} onPress={() => them()}>
        <Text style={styles.txtitem}>Thêm nhóm</Text>
      </TouchableOpacity>
    </View>
  )
}
```

```
export default QLPhong
const styles = StyleSheet.create({
    tong1:{
        flex: 1,
        justifyContent: 'center',
        alignItems: 'center',
        backgroundColor: '#003f5c'
    },
    tieude:{
        fontSize: 30,
        fontWeight: 'bold',
        color: '#F2D43D'
    },

    list:{

    },
    listCN:{
        flexDirection: 'row'
    },
    listHT:{
        alignItems: 'flex-start'
```

```
},
item:{
    backgroundColor:'#F2CA52',
    margin: 5,
    justifyContent: 'center',
    alignItems: 'center',
    paddingLeft: 25,
    paddingRight: 25,
    paddingTop: 5,
    paddingBottom: 5,
    borderRadius: 25
},
item1:{
    backgroundColor:'#F2F2F2',
    margin: 5,
    justifyContent: 'center',
    alignItems: 'center',
    paddingLeft: 25,
    paddingRight: 25,
    paddingTop: 5,
    paddingBottom: 5,
    borderRadius: 25
```

```
},
tenitem:{
    fontSize: 24,
    fontWeight: 'bold',
    color: '#8C8C8C'
},
btnitem:{
    width: 100,
    justifyContent:'center',
    alignItems: 'center',
    backgroundColor:'#A68524',
    margin: 2.5,
    padding: 3.5,
    borderRadius: 25
},
txtitem:{
    fontSize: 16,
    color: '#F2F2F2'
},
txtitem1:{
    fontSize: 16,
    color: '#8C8C8C'
```

```
    },
    btn:{
        backgroundColor: '#8C8C8C',
        width: 200,
        justifyContent:'center',
        alignItems: 'center',
        margin: 2.5,
        margin: 5,
        padding: 10,
        borderRadius: 25
    },
})
```

3.6. AddNhom.js

```
import React, { useState } from 'react'
import { StyleSheet, Text, View, TouchableOpacity, TextInput, Alert } from 'react-native'
import database from '@react-native-firebase/database'
const ThemNhom = ({ navigation, route }) => {
    const data=route.params.data
    const [tenNhom, settenNhom] = useState('')
    const dangky = async () => {
        if(tenNhom === '')
        {
```

```
    Alert.alert("Tên nhóm trống.")
    return
}

let taiKhoan=data.find(value=>{
    if(value.tenNhom===tenNhom){
        return value;
    }
})
if(taiKhoan!=null){
    Alert.alert("Tên nhóm đã tồn tại.")
    return
}
let group = {
    tenNhom: tenNhom,
    soLuong: 0
}
try {
    await database().ref("/Nhom").push(group);
    Alert.alert("Thêm nhóm thành công.")
    navigation.navigate('Home')
}
```

```
        catch (e) {
            Alert.alert('Loi' + e)
        }
    }

    return (
        <View style={styles.tong1}>
            <View style={styles.vtieude}>
                <Text style={styles.tieude}>Đăng kí thông tin nhóm</Text>
            </View>
            <View style={styles.tong2}>
                <TextInput style={styles.txtInput} placeholder={"Tên nhóm"}
onChangeText={(text) => { settenNhom(text) }} />
            </View>
            <View style={styles.vbtn}>
                <TouchableOpacity style={styles.btn} onPress={dangky}>
                    <Text style={styles.txtbtn}>Thêm</Text>
                </TouchableOpacity>
            </View>
        </View>
    )
}
export default ThemNhom
```

```
const styles = StyleSheet.create({
    tong1: {
        flex: 1,
        backgroundColor: '#002695'
    },
    vtieude: {
        margin: 15,
        alignItems: 'center'
    },
    tong2: {
        flexDirection: 'row',
        justifyContent: 'center'
    },
    tieude: {
        fontSize: 30,
        fontWeight: 'bold',
        color: '#F633FF'
    },
    txtInput: {
        backgroundColor: '#0CF9FF',
        justifyContent: 'flex-end',
        width: 300,
```

```
    fontSize: 20,
    paddingTop: 10,
    paddingBottom: 10,
    paddingLeft: 25,
    paddingRight: 25,
    margin: 10,
    borderRadius: 25
},
vbtn: {
    marginTop: 50,
    flex: 1,
    alignItems: 'center'
},
btn: {
    backgroundColor: '#E6FF0C',
    width: 200,
    justifyContent: 'center',
    alignItems: 'center',
    margin: 2.5,
    margin: 5,
    padding: 10,
    borderRadius: 25,
```

```
    }
})
```

3.7. QLThanhVienNhom.js

```
import React, { useState, useEffect } from 'react'
import { StyleSheet, Text, View, TouchableOpacity, FlatList, Alert, Image } from 'react-native'
import database from '@react-native-firebase/database'
const QLKhachThue = ({ navigation, route }) => {
    const nhom = route.params.value
    const [data, setdata] = useState([])
    const logo = require('../img/user.png')
    useEffect(() => {
        const getData = async () => {
            try {
                await database().ref("/ThanhVienNhom")
                    .on('value', snapshot => {
                        let arr = []
                        snapshot.forEach(element => {
                            let temp = {
                                id: element.key,
                                hoten: element.val().hoten,
                                sdt: element.val().sdt,
                                cmnd: element.val().cmnd,
```

```
                    gioitinh: element.val().gioitinh,
                    diachi: element.val().diachi,
                    ngayvao: element.val().ngayvao,
                    idnhom: element.val().idNhom
                }
                arr.push(temp)
            })
            let arrnhom = arr.filter((value) => {
                return value.idnhom === nhom.id
            })
            setdata(arrnhom)
        });
    }
    catch (e) {
        Alert.alert("Lỗi: " + e);
    }
  }
  getData()
}, [])

const them = (tt) => {
  navigation.navigate('CTThanhVien', { tt: tt, nhom: nhom })
```

```
    }
    const XoaKT = async (id) => {
        await database().ref(`/ThanhVienNhom/${id}`).remove()
        await database().ref(`/Nhom/${nhom.id}`).update({soLuong:
nhom.soLuong - 1,});
    }
    const capnhat = (tt, value) => {
        navigation.navigate('CTThanhVien', { tt: tt, value: value, nhom: nhom })
    }
    const xoa = (id) => {
        try {
            Alert.alert(
                "Xóa thành viên",
                "Bạn có muốn xóa thành viên này hay không ?",
                [
                    {
                        text: "Cancel"
                    },
                    { text: "OK", onPress: () => XoaKT(id) }
                ]
            );
        }
        catch (e) {
```

```
            Alert.alert('Err: ' + e)
        }
    }
    const renderItem = (item) => {
        return (
            <TouchableOpacity style={styles.item} onPress={() => { capnhat('1',
item) }}>
                <View style={styles.vItem}>
                    <Image source={logo} style={styles.hinhmuc} />
                    <View style={styles.vItemTXT}>
                        <Text style={styles.txtmuc}>{item.hoten}</Text>
                        <Text style={styles.txtmuc}>{item.sdt}</Text>
                        <Text style={styles.txtmuc}>{item.diachi}</Text>
                    </View>
                    <TouchableOpacity style={styles.btnitem} onPress={() =>
{ xoa(item.id) }}>
                        <Text style={styles.txtitem}>X</Text>
                    </TouchableOpacity>
                </View>
            </TouchableOpacity>
        )
    }
    return (
```

```
        <View style={styles.tong1}>
            <FlatList style={styles.list}
                data={data}
                renderItem={({ item }) => renderItem(item)}
                keyExtractor={item => item.id}
            />
            <TouchableOpacity style={styles.btn} onPress={() => them('0')}>
                <Text style={styles.txtitem}>THÊM</Text>
            </TouchableOpacity>
        </View>
    )
}
export default QLKhachThue
const styles = StyleSheet.create({
    tong1: {
        flex: 1,
        justifyContent: 'center',
        alignItems: 'center',
        backgroundColor: '#003f5c'
    },
    tieude: {
        fontSize: 30,
```

```
    fontWeight: 'bold',
    color: '#F2D43D'
},
list: {
},
listCN: {
    flexDirection: 'row'
},
listHT: {
    alignItems: 'flex-start'
},
item: {
    width: 350,
    backgroundColor: '#F2CA52',
    margin: 5,
    paddingLeft: 25,
    paddingRight: 25,
    paddingTop: 5,
    paddingBottom: 5,
    borderRadius: 25
},
item1: {
```

```
    backgroundColor: '#F2F2F2',
    margin: 5,
    justifyContent: 'center',
    alignItems: 'center',
    paddingLeft: 25,
    paddingRight: 25,
    paddingTop: 5,
    paddingBottom: 5,
    borderRadius: 25
},
tenitem: {
    fontSize: 24,
    fontWeight: 'bold',
    color: '#8C8C8C'
},
btnitem: {
    width: 40,
    height: 40,
    justifyContent: 'center',
    alignItems: 'center',
    backgroundColor: '#A68524',
    margin: 2.5,
```

```
    padding: 3.5,
    borderRadius: 50
},
txtitem: {
    fontSize: 16,
    color: '#F2F2F2'
},
txtitem1: {
    fontSize: 16,
    color: '#8C8C8C'
},
btn: {
    backgroundColor: '#8C8C8C',
    width: 200,
    justifyContent: 'center',
    alignItems: 'center',
    margin: 2.5,
    margin: 5,
    padding: 10,
    borderRadius: 25
},
hinhmuc: {
```

```
        width: 75,
        height: 75,
        margin: 5
    },
    vItem: {
        flexDirection: 'row',
    },
    vItemTXT: {
        width: 150,
        margin: 10
    },
    txtmuc: {
        fontWeight: 'bold'
    }
})
```

3.8. CTThanhVien.js

```
import React, { useState, useEffect } from 'react'
import { Alert, ScrollView, StyleSheet, Text, TextInput, View } from 'react-native'
import CheckBox from '@react-native-community/checkbox'
import { TouchableOpacity } from 'react-native-gesture-handler'
import DateTimePicker from '@react-native-community/datetimepicker'
import database from '@react-native-firebase/database'
```

```
const CTThanhVien = ({ navigation, route }) => {
    const nhom =route.params.nhom
    const thanhvien=route.params.value
    const tt=route.params.tt
    const [id, setid] = useState('')
    const [idNhom, setidNhom] = useState('')
    const [hoten, setHoten] = useState('')
    const [sdt, setSdt] = useState('')
    const [cmnd, setCmnd] = useState('')
    const [diachi, setDiachi] = useState('')
    const [date, setDate] = useState(new Date());
    const [mode, setMode] = useState('date');
    const [show, setShow] = useState(false);
    const [trangthai, setTrangthai] = useState(true);
    var today = new Date()
    var ngay = today.getDate() + '/' + (today.getMonth() + 1) + '/' +
today.getFullYear();
    const [ngayvao, setngayvao] = useState(ngay)
    useEffect(() => {
        if(tt==='1'){
            setid(thanhvien.id)
            setHoten(thanhvien.hoten)
            setSdt(thanhvien.sdt)
```

```
            setCmnd(thanhvien.cmnd)
            setDiachi(thanhvien.diachi)
            setTrangthai(thanhvien.gioitinh)
            setngayvao(thanhvien.ngayvao)
            setidNhom(thanhvien.idNhom)
        }
    }, [])
    const onChange = (event, selectedDate) => {
        const currentDate = selectedDate || date;
        setShow(false);
        setDate(currentDate);
        setngayvao(date.getDate() + '/' + (date.getMonth() + 1) + '/' +
date.getFullYear())
    };
    const showMode = (currentMode) => {
        setShow(true);
        setMode(currentMode);
    };
    const showDatepicker = () => {
        showMode('date');
    };
    const Save = async () => {
        if(hoten==''){
```

```
    Alert.alert("Cảnh Báo","Vui lòng nhập họ và tên.")
    return
}
if(sdt==''){
    Alert.alert("Cảnh Báo","Vui lòng nhập số điện thoại.")
    return
}
if(cmnd==''){
    Alert.alert("Cảnh Báo","Vui lòng nhập số CMND.")
    return
}
if(diachi==''){
    Alert.alert("Cảnh Báo","Vui lòng nhập địa chỉ.")
    return
}
if(ngayvao==''){
    Alert.alert("Cảnh Báo","Vui lòng nhập ngày vào nhóm.")
    return
}
let chitiet = {
    hoten: hoten,
    sdt: sdt,
```

```
            cmnd: cmnd,
            gioitinh: trangthai,
            diachi: diachi,
            ngayvao: ngayvao,
            idNhom: nhom.id
        }
        try {
            console.log(tt)
            if (tt === '0') {
                await database().ref("/ThanhVienNhom").push(chitiet);
                await database().ref(`/Nhom/${nhom.id}`).update({soLuong:
nhom.soLuong + 1,});
                Alert.alert("Thêm thành công.")
            }
            else {
                await database().ref(`/ThanhVienNhom/${id}`).set(chitiet);
                Alert.alert("Sửa thành công.")
            }
            navigation.navigate('QLThanhVienNhom')
        }
        catch (e) {
            Alert.alert('Err:', e)
        }
```

```
    }
    return (
        <ScrollView style={styles.tong1}>
            <View style={styles.tong2}>
                <Text style={styles.txtTieuDe}>Họ và tên:</Text>
                <TextInput style={styles.txtInput} placeholder={"Tên thành viên"}
value={hoten} onChangeText={(value) => setHoten(value)} />
            </View>
            <View style={styles.tong2}>
                <Text style={styles.txtTieuDe}>Liên lạc: </Text>
                <TextInput style={styles.txtInput} placeholder={"Số điện thoại"}
value={sdt} onChangeText={(value) => setSdt(value)} />
            </View>
            <View style={styles.tong2}>
                <Text style={styles.txtTieuDe}>Số CMND:</Text>
                <TextInput style={styles.txtInput} placeholder={"Số CMND"}
value={cmnd} onChangeText={(value) => setCmnd(value)} />
            </View>
            <View style={styles.tong2}>
                <Text style={styles.txtTieuDe}>Giới Tính: </Text>
                <CheckBox style={styles.checkbox}
                    disabled={false}
```

```
            value={trangthai}
            onValueChange={(newValue) => setTrangthai(newValue)}
          />
          <Text style={styles.txtTieuDeC} >Nam</Text>
        </View>
        <View style={styles.tong2}>
          <Text style={styles.txtTieuDe}>Địa chỉ:</Text>
          <TextInput style={styles.txtInput} placeholder={"Địa chỉ"}
value={diachi} onChangeText={(value) => setDiachi(value)} />
        </View>
        <View style={styles.tong2}>
          <Text style={styles.txtTieuDe}>Ngày vào: </Text>
          <TextInput style={styles.txtInput} value={ngayvao}
onChangeText={(value) => { setngayvao(value) }} />
        </View>
        <View style={styles.tong2}>
          <Text style={styles.txtTieuDe}></Text>
          {show && (
            <DateTimePicker
              testID="dateTimePicker"
              value={date}
              mode={mode}
              is24Hour={true}
```

```
                display="default"
                onChange={onChange}
              />
            )}
            <TouchableOpacity style={styles.btnDate}
onPress={showDatepicker}>
              <Text style={styles.txt}>Chọn ngày</Text>
            </TouchableOpacity>
          </View>
          <View style={styles.vbtn}>
            <TouchableOpacity style={styles.btn} onPress={Save}>
              <Text style={styles.txt}>LƯU</Text>
            </TouchableOpacity>
          </View>
        </ScrollView>
    )
}
export default CTThanhVien
const styles = StyleSheet.create({
    tong1: {
        flex: 1,
        backgroundColor: '#003f5c'
    },
```

```
tong2: {
    flexDirection: 'row'
},
tieude: {
    fontSize: 30,
    fontWeight: 'bold',
    color: '#F2D43D'
},
vtieude: {
    margin: 15,
    alignItems: 'center'
},
txtTieuDe: {
    fontSize: 20,
    padding: 10,
    margin: 10,
    textAlign: 'center',
    width: 120,
    color: '#A68524'
},
txtTieuDeC: {
    fontSize: 18,
```

```
    paddingTop: 10,
    paddingBottom: 10,
    paddingRight: 10,
    marginTop: 10,
    marginRight: 10,
    marginBottom: 10,
    textAlign: 'center',
    width: 120,
    color: '#A68524'
},
txtInput: {
    backgroundColor: '#F2CA52',
    justifyContent: 'flex-end',
    width: 250,
    fontSize: 20,
    paddingTop: 10,
    paddingBottom: 10,
    paddingLeft: 25,
    paddingRight: 25,
    margin: 10,
    borderRadius: 25
},
```

```
txt: {
    fontSize: 16,
    color: '#F2F2F2'
},
vbtn: {
    marginTop: 50,
    flex: 1,
    alignItems: 'center'
},
btn: {
    backgroundColor: '#8C8C8C',
    width: 200,
    justifyContent: 'center',
    alignItems: 'center',
    margin: 5,
    padding: 10,
    borderRadius: 25,
},
btnDate: {
    backgroundColor: '#8C8C8C',
    width: 100,
    justifyContent: 'center',
```

```
    alignItems: 'center',
    margin: 5,
    padding: 10,
    borderRadius: 25,
},
checkbox: {
    alignSelf: "center",
    paddingTop: 10,
    paddingBottom: 10,
    paddingLeft: 25,
    marginLeft: 10,
    marginTop: 10,
    marginBottom: 10,
    margin: 10,
},
txtcheckbox: {
    alignSelf: "center",
    paddingTop: 10,
    paddingBottom: 10,
    paddingRight: 25,
    margin: 10,
}
```

```
})
```

3.9. App.js

```
import * as React from 'react';
import { NavigationContainer } from '@react-navigation/native';
import { createStackNavigator } from '@react-navigation/stack';
import database from '@react-native-firebase/database';
import Login from './components/Login';
import Home from './components/Home';
import Signup from './components/Signup';
import QLThanhVien from './components/QLThanhVien';
import QLThanhVienNhom from './components/QLThanhVienNhom';
import QLNhom from './components/QLNhom';
import AddNhom from './components/AddNhom';
import CTThanhVien from './components/CTThanhVien';
const Stack = createStackNavigator();
function App() {
  return (
    <NavigationContainer>
      <Stack.Navigator>
        <Stack.Screen name="Login" component={Login}
options={{headerShown: false}} />
        <Stack.Screen name="Home" component={Home}
options={{ headerShown: false}} />
```

```
        <Stack.Screen name="Signup" component={Signup}
options={{headerShown: false}} />

        <Stack.Screen name="QLThanhVien" component={QLThanhVien}
options={{ title: 'Danh sách quản trị viên' }}/>

        <Stack.Screen name="QLThanhVienNhom"
component={QLThanhVienNhom} options={{ title: 'Danh sách thành viên'}}/>

        <Stack.Screen name="QLNhom" component={QLNhom} options={{ title:
'Danh sách nhóm' }}/>

        <Stack.Screen name="AddNhom" component={AddNhom}
options={{headerShown: false}}/>

        <Stack.Screen name="CTThanhVien" component={CTThanhVien}
options={{title: 'Thông tin thành viên'}}/>

      </Stack.Navigator>

    </NavigationContainer>

  );

}

export default App;
```

CHƯƠNG 5: TỔNG KẾT

1. Đánh giá kết quả

1.1. Kết quả đạt được

- Giao diện có tính tương tác
- Có các chức năng cơ bản của hệ thống
- Dễ thao tác và sử dụng

1.2. Hạn chế của đề tài

- Thiếu chức năng chuyển thành viên từ nhóm này sang nhóm khác
- Việc xóa nhóm không ảnh hưởng đến các thành viên trong nhóm

2. Hướng phát triển

- Xây dựng hàm chuyển thông tin của thành viên từ nhóm này sang nhóm khác
- Xây dựng hàm xóa các thành viên khi xóa nhóm của các thành viên đó

TÀI LIỆU THAM KHẢO

Tiếng Anh

[1] Bonnie Eisenman (), Learning React Native: Building Native Mobile Apps with JavaScript (2nd Edition)

[2] Richard Kho (2017), React Native by example

[3] Ethan Holmes, Tom Bray (), Getting Started with React Native

Website

[4] reactnative.dev

[5] console.firebase.google.com

[6] reactnavigation.org