BÁO CÁO CÁC BƯỚC THỰC HIỆN ỨNG DỤNG REACT NATIVE VỚI FIREBASE

# 1. Giới thiệu

Ứng dụng này là một ứng dụng React Native kết hợp với Firebase để quản lý sản phẩm. Người dùng có thể thực hiện các chức năng như thêm, sửa, xóa sản phẩm, tìm kiếm và lọc sản phẩm theo loại. Các dữ liệu sản phẩm được lưu trữ trên Firebase Realtime Database và cập nhật theo thời gian thực. Ứng dụng sử dụng Expo để đơn giản hóa việc phát triển và thử nghiệm.

# 2. Các bước thực hiện

## 2.1 Thiết lập Firebase

- Tạo một dự án Firebase trên trang Firebase Console (https://console.firebase.google.com).  
- Kết nối Firebase với ứng dụng React Native bằng cách thêm file cấu hình google-services.json cho Android.  
- Thiết lập Firebase trong ứng dụng thông qua việc cài đặt thư viện Firebase SDK cho React Native:  
  
*```bash  
npm install firebase  
```*

## Cài đặt Expo và các thư viện cần thiết

### 2.2.1 Cài đặt Expo

#### Bước 1: Cài đặt Expo CLI

*npm install -g expo-cli*

#### Bước 2: Sử dụng Expo CLI để tạo và chạy ứng dụng: ```bash npm install -g expo-cli expo init my-app cd my-app npx expo start ``` - Expo giúp tối ưu hóa quá trình phát triển mà không cần cấu hình quá phức tạp cho các môi trường iOS và Android.

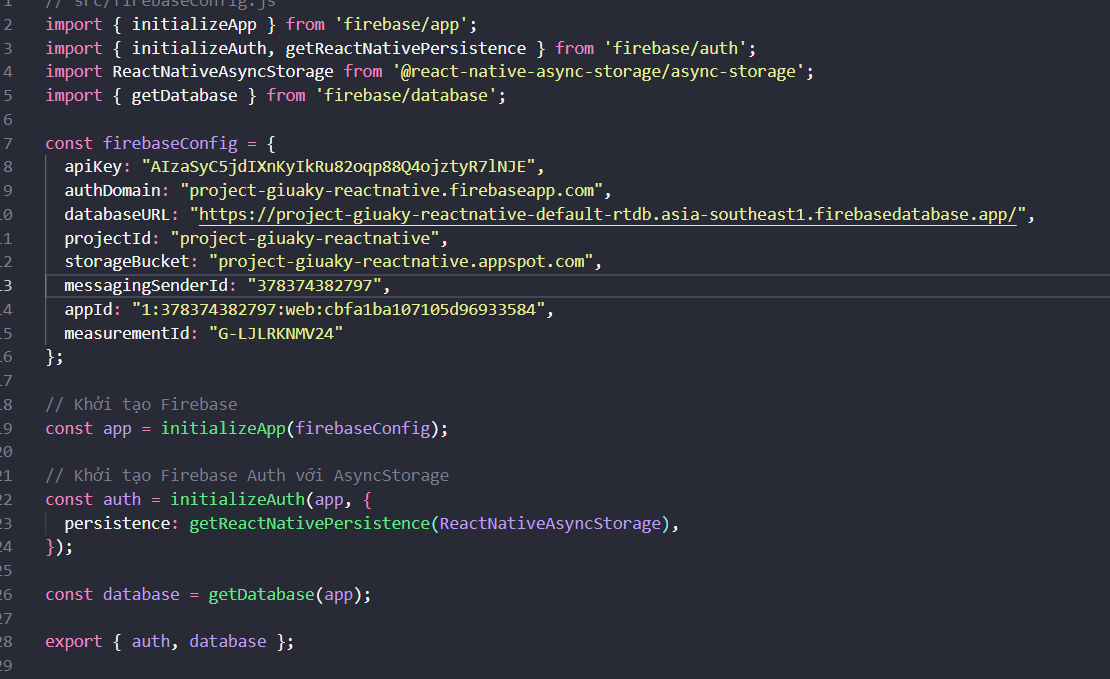
* **Điện thoại thật**: Quét mã QR bằng ứng dụng **Expo Go** (có trên Google Play và App Store).
* **Máy ảo (Android Studio):** Nhấn a để chạy trên **Android emulator**

### 2.2.2 Cài đặt các thư viện

- Cài đặt các thư viện Firebase và React Navigation để hỗ trợ các tính năng cần thiết:  
  
*```bash  
npm install @react-navigation/native @react-navigation/stack  
npm install firebase  
```*

## 2.3 Cấu hình Firebase trong ứng dụng Expo

**2.3.1** **Tạo file firebaseConfig.js trong thư mục dự án để lưu cấu hình Firebase:**



**2.3.2 Sử dụng Firebase trong file App.js**

- Import auth từ file firebaseConfig:



- Trong useEffect, sử dụng auth.onAuthStateChanged để theo dõi trạng thái đăng nhập của người dùng từ Firebase:

A screen shot of a computer

Description automatically generated

- Đăng xuất, dùng signOut(auth) trong hàm handleLogout:

A computer screen with text

Description automatically generated

## 2.4 Xây dựng giao diện người dùng:

**2.4.1 Màn hình Đăng Nhập (SignIn)**

Màn hình đăng nhập là một trong những thành phần quan trọng trong ứng dụng mạng xã hội. Để xây dựng giao diện và xử lý đăng nhập, sử dụng React Native kết hợp với Firebase Authentication.

* + - 1. **Cấu trúc giao diện**

Màn hình đăng nhập bao gồm các thành phần cơ bản:

* **Logo**: Hiển thị biểu tượng của ứng dụng, giúp người dùng dễ dàng nhận diện thương hiệu. Được đặt tại vị trí trung tâm trên màn hình, với kích thước và bóng đổ tạo cảm giác chuyên nghiệp.
* **Tiêu đề**: Dòng chữ **"PUNGUIN"** và **"Đăng Nhập"** được hiển thị nổi bật, với màu sắc chủ đạo là hồng (#FF69B4).
* **Các trường nhập liệu**: Hai trường nhập liệu cho email và mật khẩu với placeholder rõ ràng. Cả hai trường được thiết kế có khung viền hồng, giúp phân biệt với phần nền.
* **Nút Đăng Nhập**: Nút bấm có màu hồng chủ đạo, người dùng có thể nhấn vào để đăng nhập. Sau khi bấm, ứng dụng sẽ thực hiện yêu cầu đăng nhập thông qua Firebase Authentication.
  + - 1. **Xử lý logic đăng nhập**

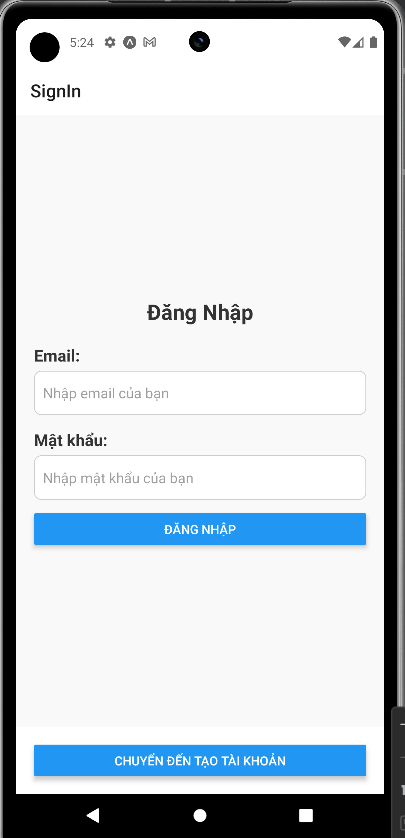
Khi người dùng nhập thông tin và nhấn vào nút **Đăng Nhập**, hàm handleLogin được gọi. Hàm này sử dụng Firebase Authentication để xác thực thông tin người dùng:

* **signInWithEmailAndPassword**: Phương thức này nhận email và mật khẩu từ người dùng và gửi yêu cầu đăng nhập lên Firebase. Nếu đăng nhập thành công, trạng thái người dùng sẽ được cập nhật qua hàm onSignIn, và người dùng sẽ được chuyển hướng đến màn hình chính (Home).
* Nếu đăng nhập thất bại (ví dụ như sai mật khẩu), một thông báo lỗi sẽ được hiển thị bằng hàm Alert.alert với nội dung lỗi rõ ràng.

**2.4.1.3 Giao diện và mã nguồn**

- Mã nguồn: https://github.com/TruyenGau/gk-react-native/tree/develop

- Kết quả thực hiện:



Người dùng nhập email và mật khẩu để đăng nhập vào hệ thống. Sau khi nhập thông tin email và mật khẩu, và bấm nút đăng nhập, người dùng được chuyển hướng đến màn hình chính nếu đăng nhập thành công.

**2.4.2 Màn Hình Đăng Ký (SignUp)**

Màn hình đăng ký là một phần quan trọng để người dùng tạo tài khoản mới trong ứng dụng mạng xã hội. Để xây dựng giao diện và xử lý việc đăng ký tài khoản, React Native được sử dụng cùng với Firebase Authentication.

* + - 1. **Cấu trúc giao diện**

Màn hình đăng ký bao gồm các thành phần cơ bản:

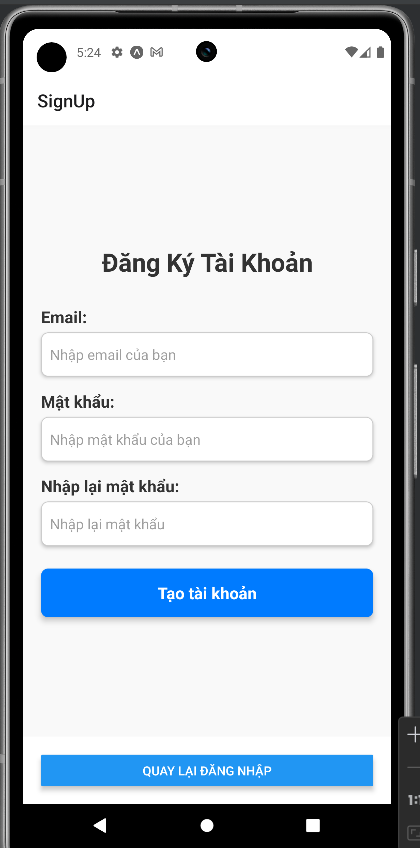
* **Logo**: Hiển thị biểu tượng của ứng dụng, giúp người dùng dễ dàng nhận diện thương hiệu. Logo được đặt ở vị trí trung tâm với kích thước vừa phải và có hiệu ứng bóng đổ để tăng tính thẩm mỹ.
* **Tiêu đề**: Dòng chữ "PUNGUIN" và "Đăng Ký Tài Khoản" được hiển thị nổi bật với thiết kế của ứng dụng.
* **Các trường nhập liệu**: Gồm hai trường nhập liệu cho email và mật khẩu. Các trường này có placeholder để hướng dẫn người dùng. Khung viền màu hồng giúp phân biệt rõ ràng các trường nhập liệu với nền.
* **Nút Tạo Tài Khoản**: Nút bấm được thiết kế với màu hồng chủ đạo, khi người dùng nhấn, ứng dụng sẽ thực hiện yêu cầu đăng ký qua Firebase Authentication.
  + - 1. **Xử lý logic đăng ký**

Khi người dùng nhập thông tin và nhấn vào nút Tạo Tài Khoản, hàm handleSignUp sẽ được gọi. Hàm này sử dụng Firebase Authentication để xử lý đăng ký tài khoản:

* **createUserWithEmailAndPassword**: Phương thức này nhận email và mật khẩu từ người dùng, sau đó gửi yêu cầu lên Firebase để tạo tài khoản. Nếu tài khoản được tạo thành công, người dùng sẽ nhận thông báo qua Alert.alert và trạng thái người dùng sẽ được cập nhật qua hàm onSignUp.
* Nếu việc đăng ký thất bại (ví dụ email không hợp lệ), một thông báo lỗi sẽ được hiển thị với nội dung rõ ràng để thông báo cho người dùng.

**2.4.2.3 Giao diện và mã nguồn**

- Mã nguồn: <https://github.com/TruyenGau/gk-react-native/tree/develop>

- Kết quả thực hiện:  


Người dùng có thể nhập email và mật khẩu để tạo tài khoản mới. Sau khi nhấn nút Tạo Tài Khoản, nếu thông tin hợp lệ, người dùng sẽ nhận được thông báo tài khoản đã được tạo thành công và trạng thái người dùng sẽ được cập nhật.

**2.4.3 Màn Hình Trang Chủ (HomeScreen)**

**Màn hình Trang Chủ** hiển thị danh sách sản phẩm của người dùng và cho phép họ tìm kiếm, quản lý các sản phẩm (sửa, xóa sản phẩm). Màn hình này được phát triển bằng React Native và kết nối với Firebase Realtime Database để hiển thị dữ liệu.

* + - 1. **Cấu trúc giao diện**
* **Tiêu đề:**
  + Hiển thị tiêu đề "Dữ liệu sản phẩm của bạn".
  + **Thanh tìm kiếm:**
  + Một thanh nhập liệu cho phép người dùng tìm kiếm sản phẩm theo tên.
  + Viền thanh tìm kiếm có màu hồng, phù hợp với thiết kế tổng thể của ứng dụng.
* **Danh sách sản phẩm:**
  + Hiển thị danh sách sản phẩm của người dùng được lấy từ Firebase.
  + Mỗi sản phẩm bao gồm tên, giá và hình ảnh (nếu có).
* **Phân loại sản phẩm:**
  + Danh sách sản phẩm có thể được lọc theo loại. Người dùng có thể chọn "Tất cả" hoặc loại cụ thể (ví dụ: "Điện tử", "Thời trang").
  + Nút chọn loại sản phẩm hiển thị dạng dropdown và được thiết kế với màu nền hồng.
* **Nút hành động:**
  + **Sửa sản phẩm:** Nút "Sửa sản phẩm" sẽ điều hướng người dùng đến màn hình chỉnh sửa sản phẩm.
  + **Xóa sản phẩm:** Nút "Xóa sản phẩm" màu đỏ, khi nhấn sẽ xóa sản phẩm từ Firebase.
  + **Thêm sản phẩm:** Nút "Thêm sản phẩm" nằm ở cuối màn hình, khi nhấn sẽ điều hướng đến màn hình thêm sản phẩm.
* **Chọn loại sản phẩm:**
  + Giao diện dropdown cho phép người dùng lọc sản phẩm theo loại.
  + Nút "Chọn loại" giúp người dùng lựa chọn sản phẩm theo các nhóm loại như “Đồ ăn”, “Đồ uống”
* **Đăng xuất:**
  + Một nút "Đăng xuất" nằm ở góc trên cùng bên phải của màn hình, cho phép người dùng đăng xuất khỏi tài khoản.

**2.4.3.2 Xử lý logic hiển thị**

**- Firebase Realtime Database:**

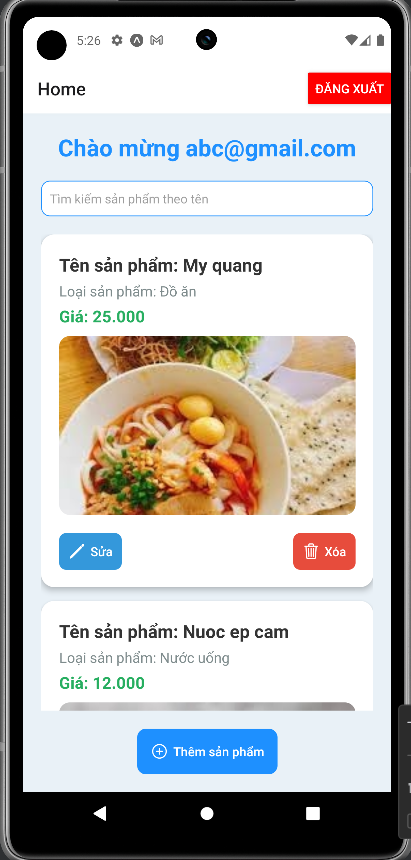
* + Dữ liệu sản phẩm được lấy từ Firebase Realtime Database và hiển thị trên màn hình. Dữ liệu được tải về và cập nhật tự động mỗi khi có sự thay đổi.
* **Tìm kiếm và lọc sản phẩm:**
  + Khi người dùng nhập từ khóa vào thanh tìm kiếm, danh sách sản phẩm sẽ được lọc và chỉ hiển thị những sản phẩm có tên chứa từ khóa đó.
* **Xóa sản phẩm:**
  + Khi người dùng nhấn nút "Xóa sản phẩm", một yêu cầu sẽ được gửi đến Firebase để xóa sản phẩm đó. Nếu thành công, danh sách sản phẩm sẽ được cập nhật và hiển thị thông báo thành công.

**2.4.3.3 Giao diện và mã nguồn**

- Mã nguồn: https://github.com/TruyenGau/gk-react-native/tree/develop

- Kết quả thực hiện:

* Giao diện HomeScreen sẽ hiển thị danh sách sản phẩm của người dùng, cho phép người dùng tìm kiếm, chỉnh sửa hoặc xóa sản phẩm:
* Chức năng tìm kiếm sản phẩm theo tên, hiển thị kết quả tìm kiếm dựa trên từ khóa mà người dùng nhập:
* Chức năng lọc sản phẩm theo loại, người dùng có thể chọn loại sản phẩm để xem các sản phẩm cùng loại:



* Ngoài ra, người dùng có thể thêm sản phẩm mới hoặc đăng xuất khỏi tài khoản của mình thông qua các nút tương ứng.

**2.4.4 Màn Hình Thêm Sản Phẩm (AddProductScreen)**

Màn hình Thêm Sản Phẩm cho phép người dùng nhập thông tin sản phẩm mới và lưu vào cơ sở dữ liệu Firebase. Màn hình này được phát triển bằng React Native và sử dụng Expo Image Picker để chọn hình ảnh cho sản phẩm.

* + - 1. **Cấu trúc giao diện**
* **Tiêu đề:**
* Hiển thị tiêu đề "Thêm Sản Phẩm".
* **Ô nhập thông tin sản phẩm:**
  + **Tên sản phẩm:**
    - Ô nhập liệu cho phép người dùng nhập tên sản phẩm.
    - Viền ô nhập có màu hồng, phù hợp với thiết kế tổng thể của ứng dụng.
  + **Giá sản phẩm:**
    - Ô nhập liệu cho phép người dùng nhập giá sản phẩm, hỗ trợ định dạng số.
  + **Loại sản phẩm:**
    - Dropdown Picker cho phép người dùng chọn loại sản phẩm từ danh sách.
    - Viền cho Picker được thiết kế với màu hồng và có các tùy chọn như “Đồ ăn” và “Đồ uống”
* **Chọn hình ảnh:**
  + Người dùng có thể chọn hình ảnh cho sản phẩm từ thư viện ảnh.
  + Ô vuông trống được sử dụng để hiển thị hình ảnh đã chọn hoặc thông báo "Chưa chọn hình ảnh".
* **Nút hành động:**
  + **Chọn hình ảnh:**
    - Nút "Chọn hình ảnh" cho phép người dùng mở thư viện ảnh để chọn hình ảnh cho sản phẩm.
  + **Thêm sản phẩm:**
    - Nút "Thêm sản phẩm" cho phép người dùng lưu thông tin sản phẩm vào Firebase.

**2.4.4.2 Xử lý logic hiển thị**

**- Quản lý trạng thái:**

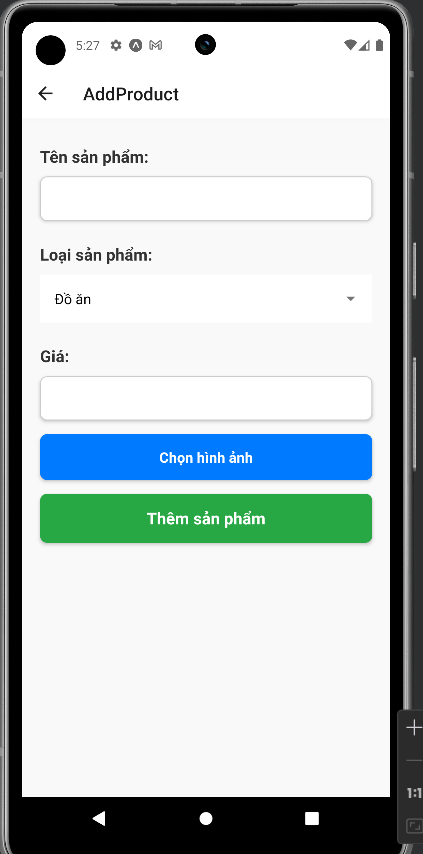
* + Sử dụng useState để quản lý các thông tin như tên sản phẩm, loại, giá và hình ảnh.
* **Chọn hình ảnh:**
  + Khi người dùng nhấn vào ô chọn hình ảnh, ứng dụng yêu cầu quyền truy cập vào thư viện ảnh và mở thư viện để người dùng có thể chọn hình ảnh.
  + Nếu không có hình ảnh nào được chọn, một thông báo sẽ hiện ra yêu cầu người dùng chọn hình ảnh.
* **Thêm sản phẩm:**
  + Khi người dùng nhấn nút "Thêm sản phẩm", ứng dụng sẽ kiểm tra xem tất cả các trường thông tin đã được điền đầy đủ chưa.
  + Nếu đầy đủ, thông tin sản phẩm sẽ được lưu vào Firebase Realtime Database.
  + Nếu có lỗi xảy ra trong quá trình lưu dữ liệu, một thông báo lỗi sẽ được hiển thị cho người dùng.

**2.4.4.3 Giao diện và mã nguồn**

- Mã nguồn: https://github.com/TruyenGau/gk-react-native/tree/develop

- Kết quả thực hiện:

* Giao diện AddProductScreen sẽ cho phép người dùng nhập thông tin sản phẩm mới và chọn hình ảnh từ thư viện ảnh. Người dùng có thể thêm sản phẩm mới vào hệ thống và quay lại màn hình chính sau khi hoàn thành:



**2.4.5 Màn Hình Sửa Sản Phẩm (EditProductScreen)**

Màn hình Sửa Sản Phẩm cho phép người dùng chỉnh sửa thông tin của một sản phẩm đã có trong danh sách sản phẩm. Màn hình này cũng được phát triển bằng React Native và tích hợp với Firebase Realtime Database để cập nhật dữ liệu sản phẩm.

* + - 1. **Cấu trúc giao diện**
* **Tiêu đề:**
  + Hiển thị tiêu đề "Sửa sản phẩm".
* **Ô nhập liệu:**
  + **Tên sản phẩm:** Ô nhập liệu cho phép người dùng chỉnh sửa tên sản phẩm.
  + **Giá sản phẩm:** Ô nhập liệu cho phép người dùng chỉnh sửa giá sản phẩm, với định dạng số.
* **Phân loại sản phẩm:**
  + Sử dụng một dropdown (Picker) để người dùng chọn loại sản phẩm.
* Các lựa chọn bao gồm: “Đồ ăn”, “Đồ uống”.
* **Chọn hình ảnh:**
  + Một ô vuông trống cho phép người dùng xem và chọn hình ảnh mới cho sản phẩm.
  + Nếu đã có hình ảnh, hình ảnh sẽ được hiển thị trong ô, nếu không sẽ có thông báo "Chưa chọn hình ảnh".
  + Nút "Chọn hình ảnh" nằm dưới ô hình ảnh cho phép người dùng mở thư viện ảnh.
* **Nút lưu thay đổi:**
  + Nút "Lưu thay đổi" cho phép người dùng lưu thông tin đã chỉnh sửa và điều hướng trở lại màn hình chính sau khi thành công.

**2.4.5.2 Xử lý logic hiển thị**

**- Firebase Realtime Database:**

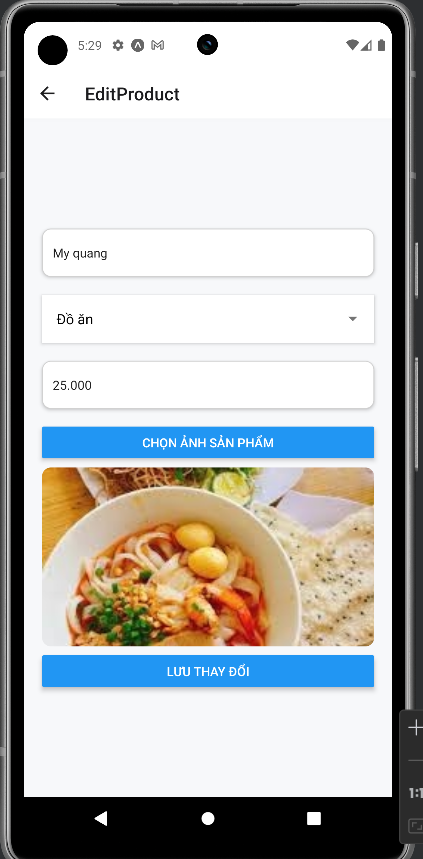
* + Khi người dùng nhấn nút "Lưu thay đổi", thông tin sản phẩm sẽ được gửi đến Firebase Realtime Database để cập nhật.
  + Nếu quá trình cập nhật thành công, màn hình sẽ hiển thị thông báo thành công và điều hướng trở lại màn hình chính.
* **Kiểm tra thông tin:**
  + Trước khi cập nhật sản phẩm, màn hình sẽ kiểm tra xem người dùng đã điền đầy đủ thông tin (tên, loại, giá và hình ảnh) hay chưa.
  + Nếu thiếu thông tin, một thông báo sẽ được hiển thị để nhắc nhở người dùng.
* **Chọn hình ảnh:**
  + Khi người dùng nhấn nút chọn hình ảnh, ứng dụng sẽ mở thư viện ảnh.
  + Người dùng có thể chọn hình ảnh, và hình ảnh đã chọn sẽ hiển thị trong ô vuông trống.

**2.4.5.3 Giao diện và mã nguồn**

- Mã nguồn: https://github.com/TruyenGau/gk-react-native/tree/develop

- Kết quả thực hiện:

* Giao diện EditProductScreen cho phép người dùng dễ dàng chỉnh sửa thông tin sản phẩm, chọn loại sản phẩm, và tải hình ảnh lên:



* Người dùng có thể cập nhật thông tin sản phẩm một cách nhanh chóng và hiệu quả, đồng thời nhận thông báo về tình trạng cập nhật sản phẩm.

# 3. Kết luận

Ứng dụng React Native kết hợp với Firebase đã hoàn thiện các chức năng cốt lõi của một hệ thống quản lý sản phẩm. Nhờ tích hợp Firebase Realtime Database, ứng dụng đảm bảo khả năng lưu trữ và truy xuất dữ liệu theo thời gian thực, mang đến trải nghiệm nhanh chóng, mượt mà cho người dùng. Các chức năng như đăng nhập, đăng ký, thêm mới, chỉnh sửa và quản lý sản phẩm được xây dựng hiệu quả, giúp người dùng dễ dàng quản lý danh sách sản phẩm cá nhân một cách trực quan và thuận tiện. Một trong những điểm nổi bật của ứng dụng là việc xử lý hình ảnh sản phẩm thông qua thư viện expo-image-picker, cho phép người dùng dễ dàng chọn ảnh từ thư viện cá nhân, tăng tính tương tác và tính thẩm mỹ cho giao diện. Sự kết hợp giữa Firebase Authentication và Firebase Realtime Database đảm bảo tính bảo mật cao, đồng thời cung cấp khả năng theo dõi trạng thái đăng nhập của người dùng một cách hiệu quả. Việc quản lý dữ liệu sản phẩm được thực hiện một cách nhanh chóng, đồng bộ, và luôn cập nhật theo thời gian thực. Điều này giúp người dùng theo dõi sát sao thông tin sản phẩm, đảm bảo tính chính xác và kịp thời trong quá trình quản lý.

Tổng thể, ứng dụng đã hoàn thành tốt các mục tiêu đề ra, cung cấp một hệ thống quản lý sản phẩm đầy đủ chức năng với khả năng mở rộng trong tương lai. Nhờ sự linh hoạt của Firebase, ứng dụng có thể dễ dàng nâng cấp để bổ sung thêm các tính năng phức tạp hơn, như phân tích dữ liệu, tối ưu hóa trải nghiệm người dùng, và tích hợp thêm các dịch vụ khác. Đây là nền tảng vững chắc để phát triển thêm các tính năng mới và nâng cao hiệu suất của ứng dụng trong các phiên bản tiếp theo.

LINK SOURCE CODE: https://github.com/TruyenGau/gk-react-native/tree/develop