

BÁO CÁO DỰ ÁN CUỐI KỲ
HỌC KỲ 2, NĂM HỌC 2023-2024
CT484: PHÁT TRIỂN ỨNG DỤNG DI ĐỘNG

- **Tên dự án/ứng dụng:** Ứng dụng review sách
- **Link GitHub mã nguồn:**
<https://github.com/23-24Sem2-Courses/ct48403-project-phamthieuthuongtinh>
- **MSSV 1:** B2014781
- **Họ tên SV 1:** Nguyễn Văn Sơn
- **MSSV 2:** B2005697
- **Họ tên SV 2:** Phạm Thiều Thương Tính
- **Lớp học phần:** 03

I. Tổng quan

- **Miêu tả dự án/ứng dụng:**

Ứng dụng Review Sách là nơi tuyệt vời cho các độc giả đam mê văn học, truyện tranh, khoa học, ... nơi họ có thể tìm kiếm đánh giá về các cuốn sách yêu thích của mình. Với ứng dụng này, người dùng có thể tìm kiếm sách theo thể loại, từ khóa, và tạo danh sách sách yêu thích của riêng mình.

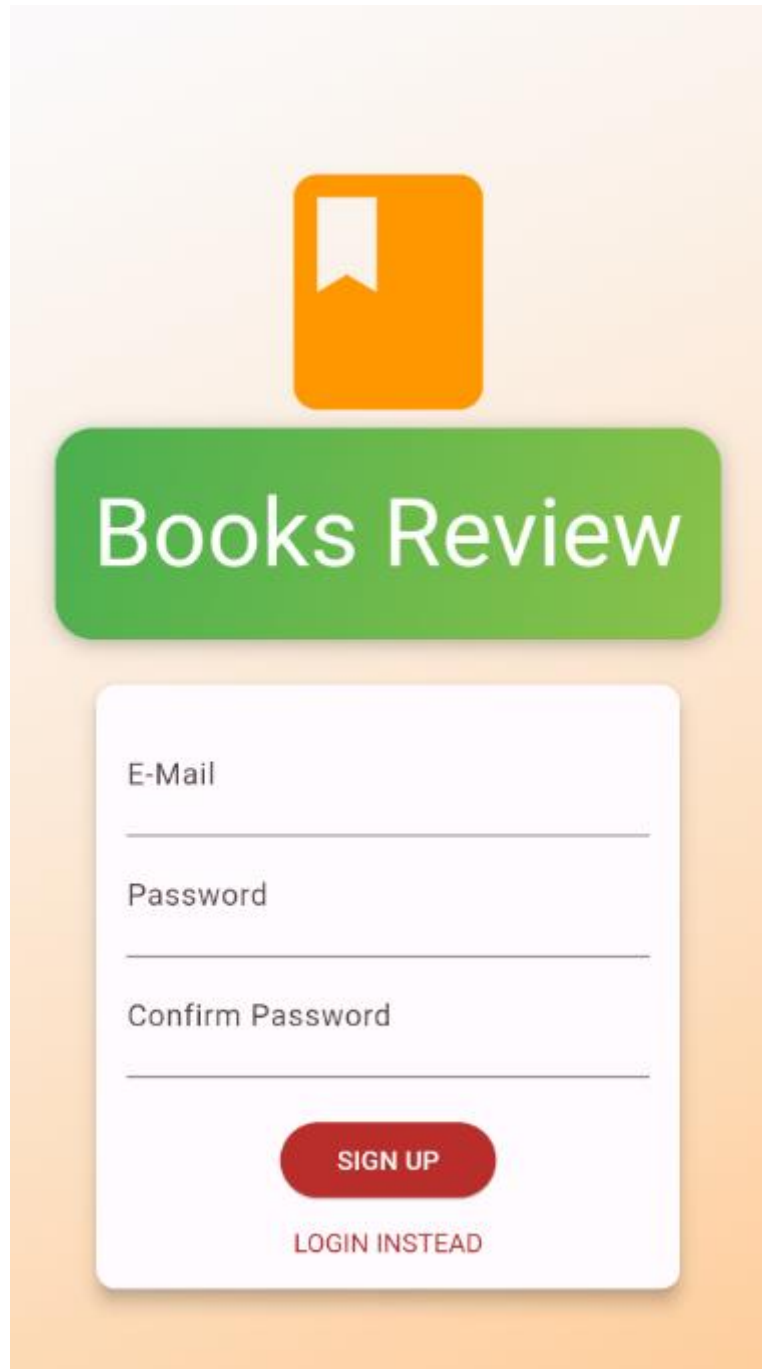
- **Bảng phân công công việc:**

| Họ tên | Công việc |
|------------------------|---|
| Nguyễn Văn Sơn | <ul style="list-style-type: none">- Giao diện và chức năng đăng nhập, đăng ký.- Chức năng đăng xuất.- Kết nối và tương tác với cơ sở dữ liệu Firebase.- Xác thực người dùng.- Định tuyến. |
| Phạm Thiều Thương Tính | <ul style="list-style-type: none">- Giao diện hiển thị sản phẩm.- Thêm, sửa, xóa sản phẩm.- Xem chi tiết sản phẩm.- Thêm và xóa sản phẩm yêu thích.- Tìm kiếm sản phẩm và hiển thị kết quả .- Hiển thị sản phẩm theo thể loại. |

II. Chi tiết các chức năng

1. Chức năng/giao diện 1: Đăng ký

- **Miêu tả chức năng/giao diện:** Người dùng có thể đăng ký tài khoản để sử dụng ứng dụng.
- **Ảnh chức năng/giao diện:**

The image shows a mobile application interface for 'Books Review'. At the top, there is an orange square icon with a white bookmark symbol. Below this is a green rounded rectangle containing the text 'Books Review' in white. Underneath is a white rounded rectangle containing three input fields labeled 'E-Mail', 'Password', and 'Confirm Password'. At the bottom of this white rectangle is a red rounded button labeled 'SIGN UP' and a link labeled 'LOGIN INSTEAD' in red text.

Hình 1: Chức năng và giao diện đăng ký

- **Chi tiết cài đặt:**

+ Các widget được sử dụng:

Container, BoxDecoration, LinearGradient, BoxShadow, Transform.rotate, Text, Card, Form, SingleChildScrollView, ValueListenableBuilder, TextButton, ElevatedButton, ValueNotifier<bool>, TextFormField, Scaffold, Stack, Positioned, Icon, Column, AppBanner, AuthCard

+ Chức năng sử dụng đến các thư viện và plugin sau:

1. flutter/material.dart: Để sử dụng các widget cơ bản trong Flutter.
2. dart:async: cung cấp các công cụ và lớp cho xử lý bất đồng bộ.
3. flutter/foundation.dart: cung cấp các lớp và hàm hỗ trợ cho việc phát triển ứng dụng Flutter.
4. provider/provider.dart: Để sử dụng Provider để quản lý trạng thái và chia sẻ dữ liệu giữa các widget.
5. dart:math: cung cấp các hàm và lớp liên quan đến toán học, bao gồm các phép toán cơ bản, hàm lấy giá trị tuyệt đối, căn bậc hai, lũy thừa, và nhiều hơn nữa.
6. dart:developer: cung cấp các chức năng hỗ trợ gỡ lỗi và logging trong Dart.
7. dart:convert: cung cấp các hàm và lớp liên quan đến mã hóa và giải mã dữ liệu trong Dart.
8. Plugin shared_preferences/shared_preferences.dart: cung cấp cách lưu trữ và truy xuất dữ liệu nhẹ trong ứng dụng của bạn bằng cách sử dụng Shared Preferences, một cơ chế lưu trữ dựa trên cặp key-value.
9. Plugin flutter_dotenv/flutter_dotenv.dart : cung cấp cách tải các biến môi trường từ tệp .env trong ứng dụng, giúp quản lý các cài đặt cụ thể cho môi trường phát triển, thử nghiệm và triển khai của ứng dụng Flutter.
10. http/http.dart: cung cấp các hàm để tạo các yêu cầu HTTP đến các URL được chỉ định và xử lý các phản hồi từ máy chủ.

+ Chức năng có sử dụng giải pháp quản lý trạng thái chia sẻ:

1. AuthManager: quản lý trạng thái xác thực của người dùng. Nó chứa các phương thức để đăng nhập, đăng ký, đăng xuất và thử tự động đăng nhập, tự động đăng xuất.
2. AuthService: xử lý các tác vụ liên quan đến xác thực như gửi yêu cầu đến server để đăng nhập hoặc đăng ký, và lưu trữ thông tin xác thực.
3. Provider: cung cấp AuthManager cho các widget con của nó, thông báo cho các widget con và cập nhật giao diện người dùng tương ứng khi trạng thái AuthManager thay đổi.

4. AuthCard: widget cho giao diện đăng nhập hoặc đăng ký.
5. AppBanner và AuthScreen: widget được sử dụng để hiển thị giao diện người dùng.

+ Chức năng này có dùng dịch vụ lưu trữ Firebase Realtime Database

Cấu trúc JSON: Gửi đi

```
{  
  "email": "[user@example.com]",  
  "password": "[PASSWORD]",  
  "returnSecureToken": true  
}
```

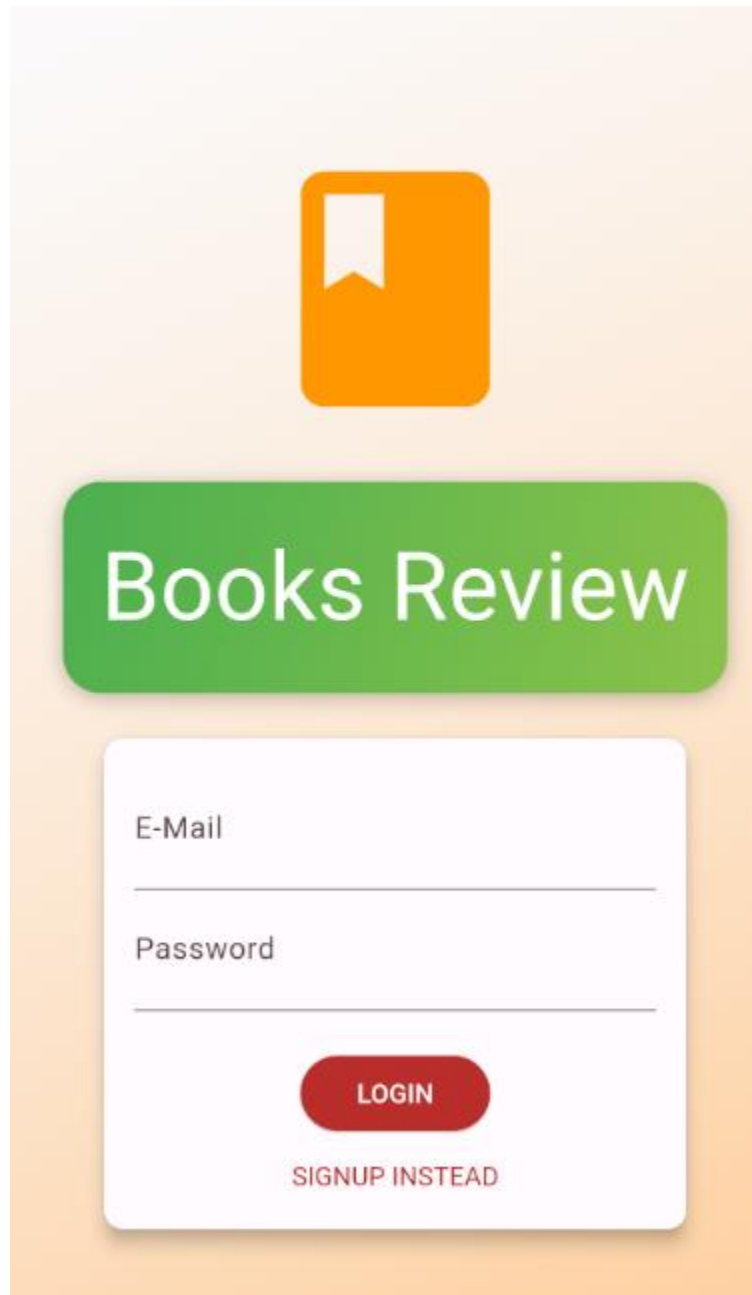
Cấu trúc JSON: trả lời

```
{  
  "idToken": "[ID_TOKEN]",  
  "email": "[user@example.com]",  
  "refreshToken": "[REFRESH_TOKEN]",  
  "expiresIn": "3600",  
  "localId": "tRcfmLH7..."  
}
```

Chức năng có sử dụng API của Firebase Authentication thông qua Web API Key để xác thực người dùng thông qua email và mật khẩu

2. Chức năng/giao diện 2: Đăng nhập

- **Miêu tả chức năng/giao diện:** Người dùng có thể đăng nhập tài khoản để sử dụng ứng dụng.
- **Ảnh chức năng/giao diện:**



Hình 2: Chức năng và giao diện đăng ký

- **Chi tiết cài đặt:**

+ Các widget được sử dụng:

Container, BoxDecoration, LinearGradient, BoxShadow, Transform.rotate, Text, Card, Form, SingleChildScrollView, ValueListenableBuilder, TextButton, ElevatedButton, ValueNotifier<bool>, TextFormField, Scaffold, Stack, Positioned, Icon, Column.

+ Chức năng sử dụng đến các thư viện và plugin sau:

1. flutter/material.dart: Để sử dụng các widget cơ bản trong Flutter.
2. dart:async: cung cấp các công cụ và lớp cho xử lý bất đồng bộ.
3. flutter/foundation.dart: cung cấp các lớp và hàm hỗ trợ cho việc phát triển ứng dụng Flutter.
4. provider/provider.dart: Để sử dụng Provider để quản lý trạng thái và chia sẻ dữ liệu giữa các widget.
5. dart:math: cung cấp các hàm và lớp liên quan đến toán học, bao gồm các phép toán cơ bản, hàm lấy giá trị tuyệt đối, căn bậc hai, lũy thừa, và nhiều hơn nữa.
6. dart:developer: cung cấp các chức năng hỗ trợ gỡ lỗi và logging trong Dart.
7. dart:convert: cung cấp các hàm và lớp liên quan đến mã hóa và giải mã dữ liệu trong Dart.
8. Plugin shared_preferences/shared_preferences.dart: cung cấp cách lưu trữ và truy xuất dữ liệu nhẹ trong ứng dụng của bạn bằng cách sử dụng Shared Preferences, một cơ chế lưu trữ dựa trên cặp key-value.
9. Plugin flutter_dotenv/flutter_dotenv.dart : cung cấp cách tải các biến môi trường từ tệp .env trong ứng dụng, giúp quản lý các cài đặt cụ thể cho môi trường phát triển, thử nghiệm và triển khai của ứng dụng Flutter.
10. http/http.dart: cung cấp các hàm để tạo các yêu cầu HTTP đến các URL được chỉ định và xử lý các phản hồi từ máy chủ.

+ Chức năng có sử dụng giải pháp quản lý trạng thái chia sẻ:

1. AuthManager: quản lý trạng thái xác thực của người dùng. Nó chứa các phương thức để đăng nhập, đăng ký, đăng xuất và thử tự động đăng nhập.
2. AuthService: xử lý các tác vụ liên quan đến xác thực như gửi yêu cầu đến server để đăng nhập hoặc đăng ký, và lưu trữ thông tin xác thực.
3. Provider: cung cấp AuthManager cho các widget con của nó, thông báo cho các widget con và cập nhật giao diện người dùng tương ứng khi trạng thái AuthManager thay đổi.
4. AuthCard: widget cho giao diện đăng nhập hoặc đăng ký.

5. AppBanner và AuthScreen: widget được sử dụng để hiển thị giao diện người dùng.

+ Chức năng này thực hiện đọc, lưu trữ dữ liệu của dịch vụ lưu trữ Firebase Realtime Database

Cấu trúc JSON: Gửi đi

```
{  
    "email":"[user@example.com]",  
    "password":"[PASSWORD]",  
    "returnSecureToken":true  
}
```

Cấu trúc JSON: trả lời

```
{  
    "idToken": "[ID_TOKEN]",  
    "email": "[user@example.com]",  
    "refreshToken": "[REFRESH_TOKEN]",  
    "expiresIn": "3600",  
    "localId": "tRcfmLH7..."  
}
```

Chức năng có sử dụng API của Firebase Authentication thông qua Web API Key để xác thực người dùng thông qua email và mật khẩu

3. Chức năng/giao diện 3: Trang chủ hiển thị tất cả sản phẩm

- **Miêu tả chức năng/giao diện:** Hiển thị các sản phẩm của ứng dụng
- **Ảnh chức năng/giao diện:**



Hình 3: Giao diện trang chủ

- **Chi tiết cài đặt:**

+ Các widget được sử dụng:

ProductsOverviewScreen, ProductSearchScreen, ProductsManager, AppDrawer, ProductsGrid, PopupMenuButton, appBar, FutureBuilder, ValueListenableBuilder, ProductGridFooter

+ Chức năng sử dụng đến các thư viện và plugin sau:

1. flutter/material.dart: Để sử dụng các widget cơ bản trong Flutter.
2. dart:async: cung cấp các công cụ và lớp cho xử lý bất đồng bộ.
3. flutter/foundation.dart: cung cấp các lớp và hàm hỗ trợ cho việc phát triển ứng dụng Flutter.
4. provider/provider.dart: Để sử dụng Provider để quản lý trạng thái và chia sẻ dữ liệu giữa các widget.
5. dart:developer: cung cấp các chức năng hỗ trợ gỡ lỗi và logging trong Dart.
6. dart:convert: cung cấp các hàm và lớp liên quan đến mã hóa và giải mã dữ liệu trong Dart.
7. Plugin flutter_dotenv/flutter_dotenv.dart : cung cấp cách tải các biến môi trường từ tệp .env trong ứng dụng, giúp quản lý các cài đặt cụ thể cho môi trường phát triển, thử nghiệm và triển khai của ứng dụng Flutter.

+ Chức năng có sử dụng giải pháp quản lý trạng thái chia sẻ:

1. ProductsManager: Class này quản lý danh sách sản phẩm và trạng thái yêu thích của từng sản phẩm. Nó cung cấp các phương thức để tải danh sách sản phẩm và chuyển đổi trạng thái yêu thích của mỗi sản phẩm.
2. ProductsOverviewScreen: Đây là một StatefulWidget được sử dụng để hiển thị danh sách sản phẩm. Trong hàm initState(), nó gọi phương thức fetchProducts() từ ProductsManager để tải danh sách sản phẩm. Khi danh sách sản phẩm được tải xong, nó sẽ hiển thị lên giao diện người dùng thông qua widget ProductsGrid.
3. ProductsGrid: Widget này là một CustomScrollView chứa danh sách các sản phẩm dưới dạng lưới. Nó lấy danh sách sản phẩm từ ProductsManager thông qua Provider. Nếu người dùng chọn chỉ hiển thị các sản phẩm yêu thích, nó sẽ chỉ hiển thị các sản phẩm đã được yêu thích. Nó cũng cung cấp một menu lọc để người dùng có thể chọn xem tất cả sản phẩm hoặc chỉ các sản phẩm yêu thích.
4. ProductGridTile: Widget này là một phần tử trong danh sách sản phẩm. Nó hiển thị một hình ảnh của sản phẩm và các nút để thực hiện các hành

động như thêm vào yêu thích. Nó cũng sử dụng Provider để chuyển đổi trạng thái yêu thích của sản phẩm khi người dùng nhấn vào nút thích.

5. ProductGridFooter: Widget này là footer của mỗi phần tử trong danh sách sản phẩm. Nó hiển thị tiêu đề của sản phẩm và một nút để thực hiện hành động thêm vào yêu thích. Nó sử dụng Provider để lắng nghe sự thay đổi trong trạng thái yêu thích của sản phẩm.

+ Chức năng này có dùng dịch vụ đọc dữ liệu của Firebase Realtime Database

Cấu trúc JSON:

```
{
  "creatorId": " 34HXP9c60EeF7IzJW3lGJNPscf33",
  "description": "Review"
  "imageUrl": "https://upload.wikimedia.org/.../1024px-Cast-Iron-
Pan.jpg",
  "title": "Doraemon"
  "cate": "Truyện tranh"
  "author": "Fuji"
}
```

Chức năng có sử dụng API của Firebase Realtime Database thông qua Web API Key

4. Chức năng/giao diện 4: Hiển thị sản phẩm yêu thích

- **Miêu tả chức năng/giao diện:** Hiển thị các sách sản phẩm yêu thích của người dùng.
- **Ảnh chức năng/giao diện:**



Hình 4: Giao diện chức năng sản phẩm yêu thích

- **Chi tiết cài đặt:**

- + Các widget được sử dụng: ProductsOverviewScreen, ProductSearchScreen, ProductsManager, AppDrawer, ProductsGrid, PopupMenuButton, appBar, FutureBuilder, ValueListenableBuilder, ProductGridFooter
- + Chức năng sử dụng đến các thư viện và plugin sau:
 1. flutter/material.dart: Để sử dụng các widget cơ bản trong Flutter.
 2. dart:async: cung cấp các công cụ và lớp cho xử lý bất đồng bộ.
 3. flutter/foundation.dart: cung cấp các lớp và hàm hỗ trợ cho việc phát triển ứng dụng Flutter.
 4. provider/provider.dart: Để sử dụng Provider để quản lý trạng thái và chia sẻ dữ liệu giữa các widget.

5. `dart:developer`: cung cấp các chức năng hỗ trợ gỡ lỗi và logging trong Dart.
6. `dart:convert`: cung cấp các hàm và lớp liên quan đến mã hóa và giải mã dữ liệu trong Dart.
7. Plugin `flutter_dotenv/flutter_dotenv.dart` : cung cấp cách tải các biến môi trường từ tệp `.env` trong ứng dụng, giúp quản lý các cài đặt cụ thể cho môi trường phát triển, thử nghiệm và triển khai của ứng dụng Flutter.

+ Chức năng sử dụng giải pháp quản lý trạng thái chia sẻ: Có

1. `ProductsManager`: Class này quản lý danh sách sản phẩm và trạng thái yêu thích của từng sản phẩm. Nó cung cấp các phương thức để tải danh sách sản phẩm và chuyển đổi trạng thái yêu thích của mỗi sản phẩm.
2. `ProductsOverviewScreen`: Đây là một `StatefulWidget` được sử dụng để hiển thị danh sách sản phẩm. Trong hàm `initState()`, nó gọi phương thức `fetchProducts()` từ `ProductsManager` để tải danh sách sản phẩm. Khi danh sách sản phẩm được tải xong, nó sẽ hiển thị lên giao diện người dùng thông qua widget `ProductsGrid`.
3. `ProductsGrid`: Widget này là một `CustomScrollView` chứa danh sách các sản phẩm dưới dạng lưới. Nó lấy danh sách sản phẩm từ `ProductsManager` thông qua `Provider`. Nếu người dùng chọn chỉ hiển thị các sản phẩm yêu thích, nó sẽ chỉ hiển thị các sản phẩm đã được yêu thích. Nó cũng cung cấp một menu lọc để người dùng có thể chọn xem tất cả sản phẩm hoặc chỉ các sản phẩm yêu thích.
4. `ProductGridTile`: Widget này là một phần tử trong danh sách sản phẩm. Nó hiển thị một hình ảnh của sản phẩm và các nút để thực hiện các hành động như thêm vào yêu thích. Nó cũng sử dụng `Provider` để chuyển đổi trạng thái yêu thích của sản phẩm khi người dùng nhấn vào nút thích.
5. `ProductGridFooter`: Widget này là footer của mỗi phần tử trong danh sách sản phẩm. Nó hiển thị tiêu đề của sản phẩm và một nút để thực hiện hành động thêm vào yêu thích. Nó sử dụng `Provider` để lắng nghe sự thay đổi trong trạng thái yêu thích của sản phẩm.

+ Chức năng này có dùng dịch vụ đọc dữ liệu của Firebase Realtime Database

Cấu trúc JSON:

```
{
  "creatorId": " 34HXP9c60EeF7IzJW3lGJNPscf33",
  "description": "Review"
```

```

    "imageUrl": "https://upload.wikimedia.org/.../1024px-Cast-Iron-
    Pan.jpg",
    "title": "Doraemon"
    "cate": "Truyện tranh"
    "author": "Fuji"
  }

```

Chức năng có sử dụng API của Firebase Realtime Database thông qua Web API Key

5. Chức năng/giao diện 5: **Hiển thị chi tiết sản phẩm**

- **Miêu tả chức năng/giao diện:** Hiển thị chi tiết của sản phẩm
- **Ảnh chức năng/giao diện:**



Hình 5: Giao diện chi tiết sản phẩm

- **Chi tiết cài đặt:**

+ Các widget được sử dụng:

Scaffold, appBar, IconButton, Appdrawer, Column, Image.network, Text, SnackBar, SingleChildScrollView, Container, SizedBox, Padding.

+ Chức năng sử dụng đến các thư viện và plugin sau:

1. flutter/material.dart: Để sử dụng các widget cơ bản trong Flutter.
2. dart:async: cung cấp các công cụ và lớp cho xử lý bất đồng bộ.
3. flutter/foundation.dart: cung cấp các lớp và hàm hỗ trợ cho việc phát triển ứng dụng Flutter.
4. provider/provider.dart: Để sử dụng Provider để quản lý trạng thái và chia sẻ dữ liệu giữa các widget.
5. dart:developer: cung cấp các chức năng hỗ trợ gỡ lỗi và logging trong Dart.
6. dart:convert: cung cấp các hàm và lớp liên quan đến mã hóa và giải mã dữ liệu trong Dart.
7. Plugin flutter_dotenv/flutter_dotenv.dart : cung cấp cách tải các biến môi trường từ tệp .env trong ứng dụng, giúp quản lý các cài đặt cụ thể cho môi trường phát triển, thử nghiệm và triển khai của ứng dụng Flutter.

+ Chức năng có sử dụng giải pháp quản lý trạng thái chia sẻ:

1. ProductsManager: Đây là một class được tạo ra để quản lý danh sách sản phẩm và trạng thái yêu thích của từng sản phẩm. Class này sử dụng Provider để cung cấp trạng thái này cho toàn bộ ứng dụng.
2. ProductDetailScreen: Đây là một StatefulWidget được sử dụng để hiển thị thông tin chi tiết của một sản phẩm cụ thể. Trong hàm initState(), nó sử dụng Provider để lấy trạng thái yêu thích của sản phẩm từ ProductsManager và cập nhật trạng thái cục bộ của nó (isFavorite). Khi người dùng nhấn vào nút thêm vào yêu thích, nó gọi phương thức addToFavorites để cập nhật trạng thái yêu thích của sản phẩm và hiển thị một SnackBar thông báo kết quả.
3. Provider: Cung cấp trạng thái yêu thích của sản phẩm cho ProductDetailScreen thông qua ProductsManager. Khi trạng thái yêu thích thay đổi, Provider thông báo cho tất cả các widget đang lắng nghe về sự thay đổi này để cập nhật giao diện người dùng.

+ Chức năng này có dùng dịch vụ đọc và lưu trữ dữ liệu của Firebase Realtime Database

Cấu trúc JSON:

```

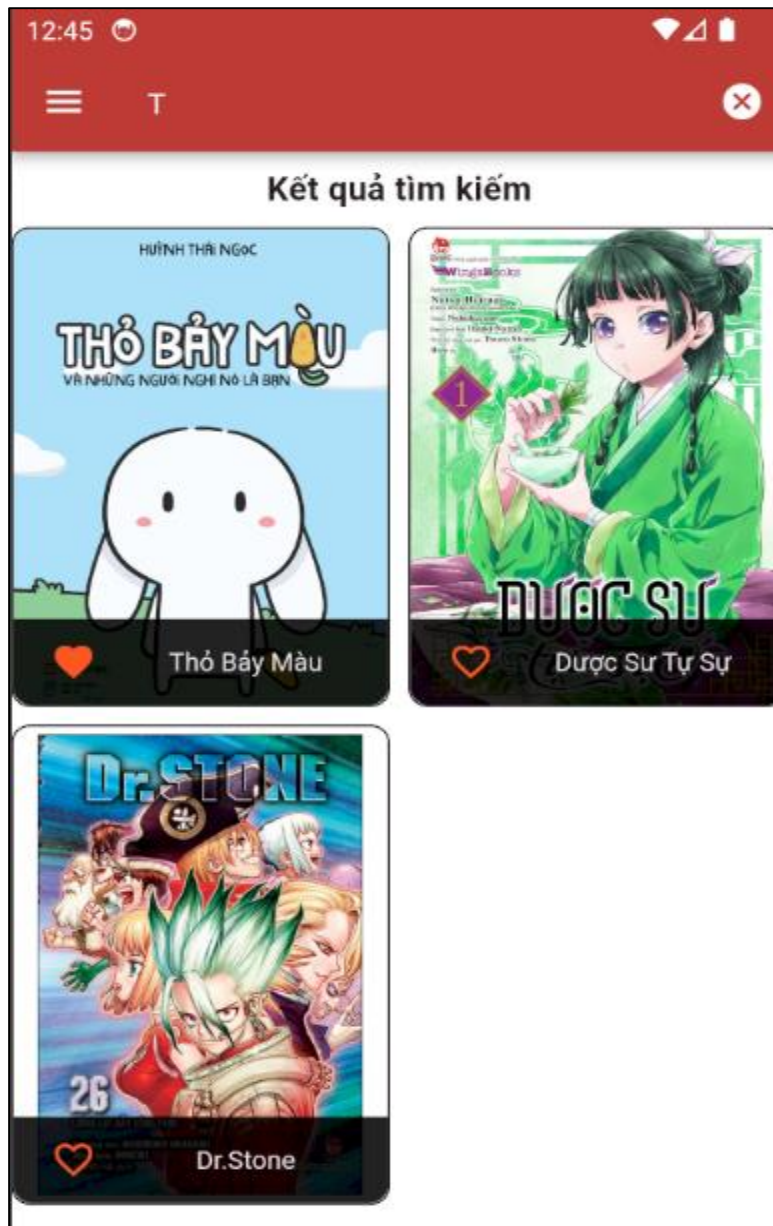
{
    "creatorId": " 34HXP9c60EeF7lzJW3lGJNPscf33",
    "description": "Review"
    "imageUrl": "https://upload.wikimedia.org/.../1024px-Cast-Iron-
Pan.jpg",
    "title": "Doraemon"
    "cate": "Truyện tranh"
    "author": "Fuji"
}

```

Chức năng có sử dụng API của Firebase Realtime Database thông qua Web API Key

6. Chức năng/giao diện 6: Tìm kiếm sản phẩm

- **Miêu tả chức năng/giao diện:** Tìm kiếm sản phẩm bằng cách nhập từ khóa vào ô tìm kiếm, ấn vào icon x để xóa từ khóa tìm kiếm.
- **Ảnh chức năng/giao diện:**



Hình 6: Giao diện tìm kiếm sản phẩm

– **Chi tiết cài đặt:**

+ Các widget được sử dụng:

Scaffold, appBar, IconButton, Appdrawer, Container, SizedBox, TextField, SingleChildScrollView, Image.network, Text, GridView.builder, ProductGridTile.

+ Chức năng sử dụng đến các thư viện và plugin sau:

1. flutter/material.dart: Để sử dụng các widget cơ bản trong Flutter.
2. dart:async: cung cấp các công cụ và lớp cho xử lý bất đồng bộ.

3. flutter/foundation.dart: cung cấp các lớp và hàm hỗ trợ cho việc phát triển ứng dụng Flutter.
4. provider/provider.dart: Để sử dụng Provider để quản lý trạng thái và chia sẻ dữ liệu giữa các widget.
5. dart:developer: cung cấp các chức năng hỗ trợ gỡ lỗi và logging trong Dart.
6. dart:convert: cung cấp các hàm và lớp liên quan đến mã hóa và giải mã dữ liệu trong Dart.
7. Plugin flutter_dotenv/flutter_dotenv.dart : cung cấp cách tải các biến môi trường từ tệp .env trong ứng dụng, giúp quản lý các cài đặt cụ thể cho môi trường phát triển, thử nghiệm và triển khai của ứng dụng Flutter.

+ Chức năng có sử dụng giải pháp quản lý trạng thái chia sẻ:

1. ProductsManager: Class này quản lý danh sách sản phẩm và trạng thái yêu thích của từng sản phẩm. Nó cung cấp các phương thức để tải danh sách sản phẩm và chuyển đổi trạng thái yêu thích của mỗi sản phẩm.
2. ProductGridTile: Widget này là một phần tử trong danh sách sản phẩm. Nó hiển thị một hình ảnh của sản phẩm và các nút để thực hiện các hành động như thêm vào yêu thích. Nó cũng sử dụng Provider để chuyển đổi trạng thái yêu thích của sản phẩm khi người dùng nhấn vào nút thích.
3. ProductGridFooter: Widget này là footer của mỗi phần tử trong danh sách sản phẩm. Nó hiển thị tiêu đề của sản phẩm và một nút để thực hiện hành động thêm vào yêu thích. Nó sử dụng Provider để lắng nghe sự thay đổi trong trạng thái yêu thích của sản phẩm.

+ Chức năng này có dùng dịch vụ đọc và lưu trữ dữ liệu của Firebase Realtime Database

Cấu trúc JSON:

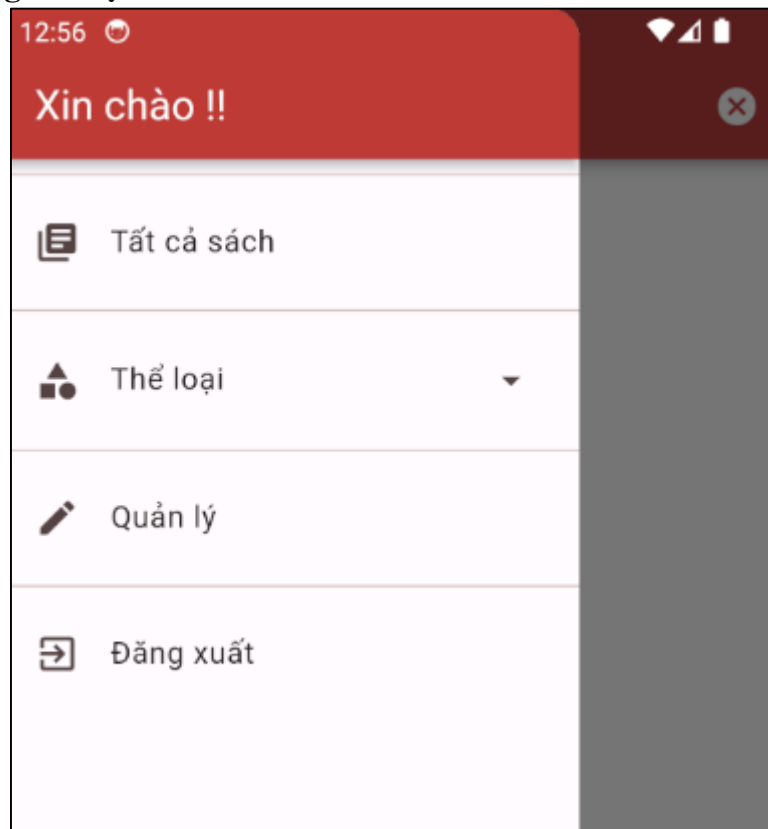
```
{
  "creatorId": " 34HXP9c60EeF7IzJW3lGJNPscf33",
  "description": "Review"
  "imageUrl": "https://upload.wikimedia.org/.../1024px-Cast-Iron-Pan.jpg",
  "title": "Doraemon"
  "cate": "Truyện tranh"
```

```
"author": "Fuji"  
}
```

Chức năng có sử dụng API của Firebase Realtime Database thông qua Web API Key

7. Chức năng/giao diện 7: Điều hướng

- **Miêu tả chức năng/giao diện:** Đi đến các trang khác hoặc đăng xuất
- **Ảnh chức năng/giao diện:**



Hình 7: Giao diện bộ điều hướng

- Chi tiết cài đặt:

+ Các widget được sử dụng:

Drawer, Column, AppBar, Divider, ListTile, ExpansionTile

+ Chức năng sử dụng đến các thư viện và plugin sau:

1. flutter/material.dart: Để sử dụng các widget cơ bản trong Flutter.
2. dart:async: cung cấp các công cụ và lớp cho xử lý bất đồng bộ.

3. flutter/foundation.dart: cung cấp các lớp và hàm hỗ trợ cho việc phát triển ứng dụng Flutter.
4. provider/provider.dart: Để sử dụng Provider để quản lý trạng thái và chia sẻ dữ liệu giữa các widget.
5. dart:developer: cung cấp các chức năng hỗ trợ gỡ lỗi và logging trong Dart.
6. dart:convert: cung cấp các hàm và lớp liên quan đến mã hóa và giải mã dữ liệu trong Dart.
7. Plugin flutter_dotenv/flutter_dotenv.dart : cung cấp cách tải các biến môi trường từ tệp .env trong ứng dụng, giúp quản lý các cài đặt cụ thể cho môi trường phát triển, thử nghiệm và triển khai của ứng dụng Flutter.

+ Chức năng có sử dụng giải pháp quản lý trạng thái chia sẻ:

1. ProductsManager: Được sử dụng để quản lý danh sách sản phẩm và thể loại sản phẩm. Nó cung cấp các phương thức để tải dữ liệu sản phẩm, lấy danh sách thể loại, và thực hiện các thao tác khác liên quan đến sản phẩm. Trạng thái quản lý bởi ProductsManager bao gồm danh sách sản phẩm, danh sách thể loại và trạng thái yêu thích của các sản phẩm.
2. AuthManager: Quản lý trạng thái xác thực của người dùng, bao gồm việc đăng nhập và đăng xuất. Nó cung cấp các phương thức để xử lý việc đăng nhập và đăng xuất, cũng như kiểm tra trạng thái xác thực hiện tại của người dùng.

+ Chức năng này có dùng dịch vụ đọc dữ liệu của Firebase Realtime Database

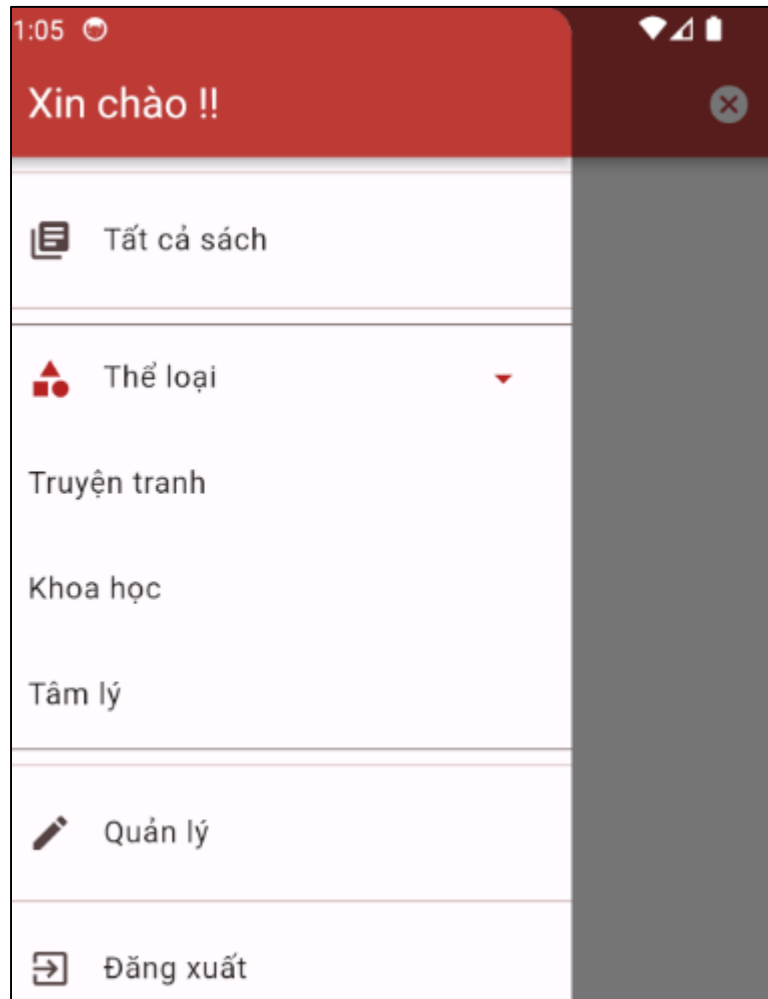
Cấu trúc JSON:

```
{
  "creatorId": " 34HXP9c60EeF7IzJW3lGJNPscf33",
  "description": "Review"
  "imageUrl": "https://upload.wikimedia.org/.../1024px-Cast-Iron-
Pan.jpg",
  "title": "Doraemon"
  "cate": "Truyện tranh"
  "author": "Fuji"
}
```

Chức năng có sử dụng API của Firebase Realtime Database thông qua Web API Key

8. Chức năng/giao diện 8: Hiển thị sản phẩm theo loại

- **Miêu tả chức năng/giao diện:** Hiển thị các sản phẩm theo loại
- **Ảnh chức năng/giao diện:**



Hình 8: Giao diện điều hướng đến loại sản phẩm



Hình 9: Giao diện hiển thị sản phẩm theo loại

– **Chi tiết cài đặt:**

+ Các widget được sử dụng:

Scaffold, appBar, IconButton, Appdrawer, CircularProgressIndicator
SingleChildScrollView,Column,GridView.builder,SizedBox,
FutureBuilder.

+ Chức năng sử dụng đến các thư viện và plugin sau:

1. flutter/material.dart: Để sử dụng các widget cơ bản trong Flutter.
2. dart:async: cung cấp các công cụ và lớp cho xử lý bất đồng bộ.
3. flutter/foundation.dart: cung cấp các lớp và hàm hỗ trợ cho việc phát triển ứng dụng Flutter.
4. provider/provider.dart: Để sử dụng Provider để quản lý trạng thái và chia sẻ dữ liệu giữa các widget.
5. dart:developer: cung cấp các chức năng hỗ trợ gỡ lỗi và logging trong Dart.
6. dart:convert: cung cấp các hàm và lớp liên quan đến mã hóa và giải mã dữ liệu trong Dart.
7. Plugin flutter_dotenv/flutter_dotenv.dart : cung cấp cách tải các biến môi trường từ tệp .env trong ứng dụng, giúp quản lý các cài đặt cụ thể cho môi trường phát triển, thử nghiệm và triển khai của ứng dụng Flutter.

+ Chức năng có sử dụng giải pháp quản lý trạng thái chia sẻ:

1. ProductsManager: Được sử dụng để quản lý danh sách sản phẩm và thể loại sản phẩm. Nó cung cấp các phương thức để tải dữ liệu sản phẩm, lấy danh sách thể loại, và thực hiện các thao tác khác liên quan đến sản phẩm. Trạng thái quản lý bởi ProductsManager bao gồm danh sách sản phẩm, danh sách thể loại và trạng thái yêu thích của các sản phẩm.
2. AuthManager: Quản lý trạng thái xác thực của người dùng, bao gồm việc đăng nhập và đăng xuất. Nó cung cấp các phương thức để xử lý việc đăng nhập và đăng xuất, cũng như kiểm tra trạng thái xác thực hiện tại của người dùng.
3. ProductGridTile: Widget này là một phần tử trong danh sách sản phẩm. Nó hiển thị một hình ảnh của sản phẩm và các nút để thực hiện các hành động như thêm vào yêu thích. Nó cũng sử dụng Provider để chuyển đổi trạng thái yêu thích của sản phẩm khi người dùng nhấn vào nút thích.
4. ProductGridFooter: Widget này là footer của mỗi phần tử trong danh sách sản phẩm. Nó hiển thị tiêu đề của sản phẩm và một nút để thực hiện hành động thêm vào yêu thích. Nó sử dụng Provider để lắng nghe sự thay đổi trong trạng thái yêu thích của sản phẩm.

+ Chức năng này có dùng dịch vụ đọc dữ liệu của Firebase Realtime Database

Cấu trúc JSON:

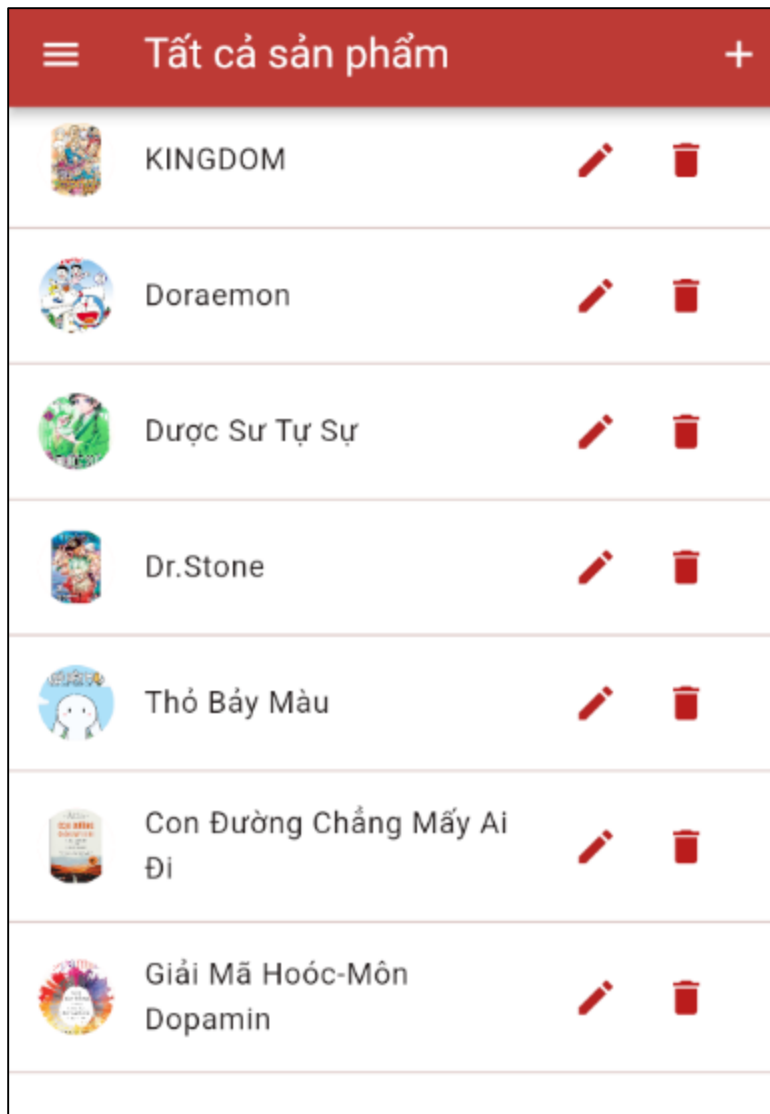
```
{
```

```
"creatorId": " 34HXP9c60EeF7IzJW3lGJNPscf33",  
"description": "Review"  
"imageUrl": "https://upload.wikimedia.org/.../1024px-Cast-Iron-  
Pan.jpg",  
"title": "Doraemon"  
"cate": "Truyện tranh"  
"author": "Fuji"  
}
```

Chức năng có sử dụng API của Firebase Realtime Database thông qua Web API Key

9. Chức năng/giao diện 9: Quản lý sản phẩm

- **Miêu tả chức năng/giao diện:** Hiện thị các sản phẩm, các chức năng sửa, xóa sản phẩm
- **Ảnh chức năng/giao diện:**



Hình 10: Giao diện trang quản lý sản phẩm

– **Chi tiết cài đặt:**

+ Các widget được sử dụng:

Scaffold, appBar, IconButton, Appdrawer, ListTile, CircleAvatar
RefreshIndicator, ListView.builder, Consumer.

+ Chức năng sử dụng đến các thư viện và plugin sau:

1. flutter/material.dart: Để sử dụng các widget cơ bản trong Flutter.
2. dart:async: cung cấp các công cụ và lớp cho xử lý bất đồng bộ.
3. flutter/foundation.dart: cung cấp các lớp và hàm hỗ trợ cho việc phát triển ứng dụng Flutter.
4. provider/provider.dart: Để sử dụng Provider để quản lý trạng thái và chia sẻ dữ liệu giữa các widget.

5. dart:developer: cung cấp các chức năng hỗ trợ gỡ lỗi và logging trong Dart.
6. dart:convert: cung cấp các hàm và lớp liên quan đến mã hóa và giải mã dữ liệu trong Dart.
7. Plugin flutter_dotenv/flutter_dotenv.dart : cung cấp cách tải các biến môi trường từ tệp .env trong ứng dụng, giúp quản lý các cài đặt cụ thể cho môi trường phát triển, thử nghiệm và triển khai của ứng dụng Flutter.

+ Chức năng có sử dụng giải pháp quản lý trạng thái chia sẻ:

1. ProductsManager: Đây là một class được tạo ra để quản lý danh sách sản phẩm và trạng thái yêu thích của từng sản phẩm. Class này sử dụng Provider để cung cấp trạng thái này cho toàn bộ ứng dụng.
2. Provider: được sử dụng để cung cấp dữ liệu sản phẩm và quản lý các thay đổi trong dữ liệu này, truy cập và gọi phương thức trên 'ProductsManager'.

+ Chức năng này có dùng dịch vụ đọc dữ liệu của Firebase Realtime Database

Cấu trúc JSON:

```
{
  "creatorId": " 34HXP9c60EeF7IzJW3IGJNPscf33",
  "description": "Review"
  "imageUrl": "https://upload.wikimedia.org/.../1024px-Cast-Iron-
Pan.jpg",
  "title": "Doraemon"
  "cate": "Truyện tranh"
  "author": "Fuji"
}
```

Chức năng có sử dụng API của Firebase Realtime Database thông qua Web API Key

10. Chức năng/giao diện 10: Thêm sản phẩm

- **Miêu tả chức năng/giao diện:** Thêm sản phẩm vào ứng dụng
- **Ảnh chức năng/giao diện:**

The screenshot shows a mobile application interface for editing a product. At the top is a red header bar with a white back arrow on the left, the text 'Hiệu chỉnh' (Edit) in the center, and a white save icon on the right. Below the header is a white form area. The form contains several input fields: a text field labeled 'Tên' (Name) with a red underline, a text field labeled 'Thể loại' (Category), a text field labeled 'Tác giả' (Author), a text field labeled 'Mô tả' (Description), and a text field labeled 'URL hình ảnh' (Image URL). There is also a small rectangular box with the placeholder text 'Enter a URL'.

Hình 11: Giao diện thêm sản phẩm

- **Chi tiết cài đặt:**
 - + Các widget được sử dụng:
Scaffold, appBar, IconButton, Form, TextFormField, ListView, FutureBuilder, CircularProgressIndicator

+ Chức năng sử dụng đến các thư viện và plugin sau:

1. flutter/material.dart: Để sử dụng các widget cơ bản trong Flutter.
2. dart:async: cung cấp các công cụ và lớp cho xử lý bất đồng bộ.
3. flutter/foundation.dart: cung cấp các lớp và hàm hỗ trợ cho việc phát triển ứng dụng Flutter.
4. provider/provider.dart: Để sử dụng Provider để quản lý trạng thái và chia sẻ dữ liệu giữa các widget.
5. dart:developer: cung cấp các chức năng hỗ trợ gỡ lỗi và logging trong Dart.
6. dart:convert: cung cấp các hàm và lớp liên quan đến mã hóa và giải mã dữ liệu trong Dart.
7. Plugin flutter_dotenv/flutter_dotenv.dart : cung cấp cách tải các biến môi trường từ tệp .env trong ứng dụng, giúp quản lý các cài đặt cụ thể cho môi trường phát triển, thử nghiệm và triển khai của ứng dụng Flutter.

+ Chức năng có sử dụng giải pháp quản lý trạng thái chia sẻ

1. ProductsManager: Đây là một class được tạo ra để quản lý danh sách sản phẩm và trạng thái yêu thích của từng sản phẩm. Class này sử dụng Provider để cung cấp trạng thái này cho toàn bộ ứng dụng.
2. Provider: được sử dụng để cung cấp dữ liệu sản phẩm và quản lý các thay đổi trong dữ liệu này, truy cập và gọi phương thức trên 'ProductsManager'.
3. TextEditingController: Được sử dụng để theo dõi và cập nhật nội dung của trường văn bản "URL hình ảnh".
4. Validation: Thực hiện kiểm tra dữ liệu đầu vào từ người dùng và hiển thị thông báo lỗi nếu cần.
5. Form và GlobalKey: Sử dụng để quản lý và xác thực biểu mẫu chỉnh sửa sản phẩm.

+ Chức năng này có dùng dịch vụ lưu trữ dữ liệu của Firebase Realtime Database

Cấu trúc JSON:

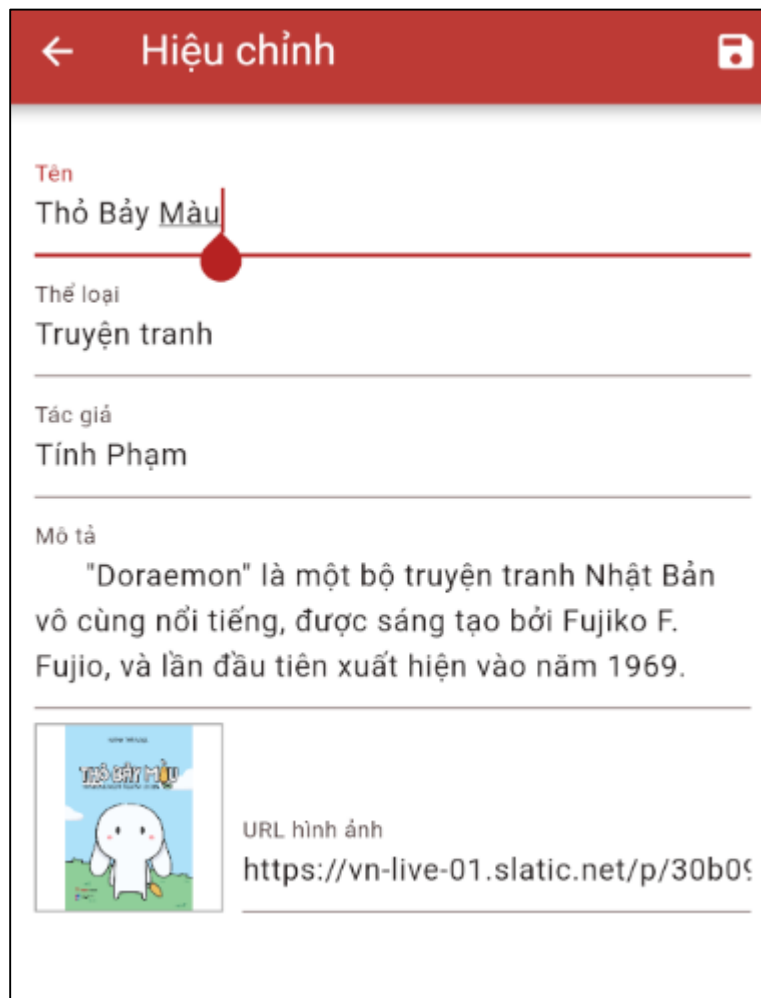
```
{  
  "creatorId": " 34HXP9c60EeF7IzJW3IGJNPscf33",  
  "description": "Review"  
  "imageUrl": "https://upload.wikimedia.org/.../1024px-Cast-Iron-Pan.jpg",
```

```
"title": "Doraemon"
"cate": "Truyện tranh"
"author": "Fuji"
}
```

Chức năng có sử dụng API của Firebase Realtime Database thông qua Web API Key

11. Chức năng/giao diện 11: Chỉnh sửa sản phẩm

- **Miêu tả chức năng/giao diện:** Sửa thông tin sản phẩm
- **Ảnh chức năng/giao diện:**



Hình 12: Giao diện sửa sản phẩm

- **Chi tiết cài đặt:**

+ Các widget được sử dụng:

Scaffold, appBar, IconButton, Form, TextFormField, ListView, FutureBuilder, CircularProgressIndicator

+ Chức năng sử dụng đến các thư viện và plugin sau:

1. flutter/material.dart: Để sử dụng các widget cơ bản trong Flutter.
2. dart:async: cung cấp các công cụ và lớp cho xử lý bất đồng bộ.
3. flutter/foundation.dart: cung cấp các lớp và hàm hỗ trợ cho việc phát triển ứng dụng Flutter.
4. provider/provider.dart: Để sử dụng Provider để quản lý trạng thái và chia sẻ dữ liệu giữa các widget.
5. dart:developer: cung cấp các chức năng hỗ trợ gỡ lỗi và logging trong Dart.
6. dart:convert: cung cấp các hàm và lớp liên quan đến mã hóa và giải mã dữ liệu trong Dart.
7. Plugin flutter_dotenv/flutter_dotenv.dart : cung cấp cách tải các biến môi trường từ tệp .env trong ứng dụng, giúp quản lý các cài đặt cụ thể cho môi trường phát triển, thử nghiệm và triển khai của ứng dụng Flutter.

+ Chức năng sử dụng giải pháp quản lý trạng thái chia sẻ: Có

1. ProductsManager: Đây là một class được tạo ra để quản lý danh sách sản phẩm và trạng thái yêu thích của từng sản phẩm. Class này sử dụng Provider để cung cấp trạng thái này cho toàn bộ ứng dụng.
2. Provider: được sử dụng để cung cấp dữ liệu sản phẩm và quản lý các thay đổi trong dữ liệu này, truy cập và gọi phương thức trên 'ProductsManager'.
3. TextEditingController: Được sử dụng để theo dõi và cập nhật nội dung của trường văn bản "URL hình ảnh".
4. Validation: Thực hiện kiểm tra dữ liệu đầu vào từ người dùng và hiển thị thông báo lỗi nếu cần.
5. Form và GlobalKey: Sử dụng để quản lý và xác thực biểu mẫu chỉnh sửa sản phẩm.

+ Chức năng này có dùng dịch vụ đọc và lưu trữ dữ liệu của Firebase Realtime Database

Cấu trúc JSON:

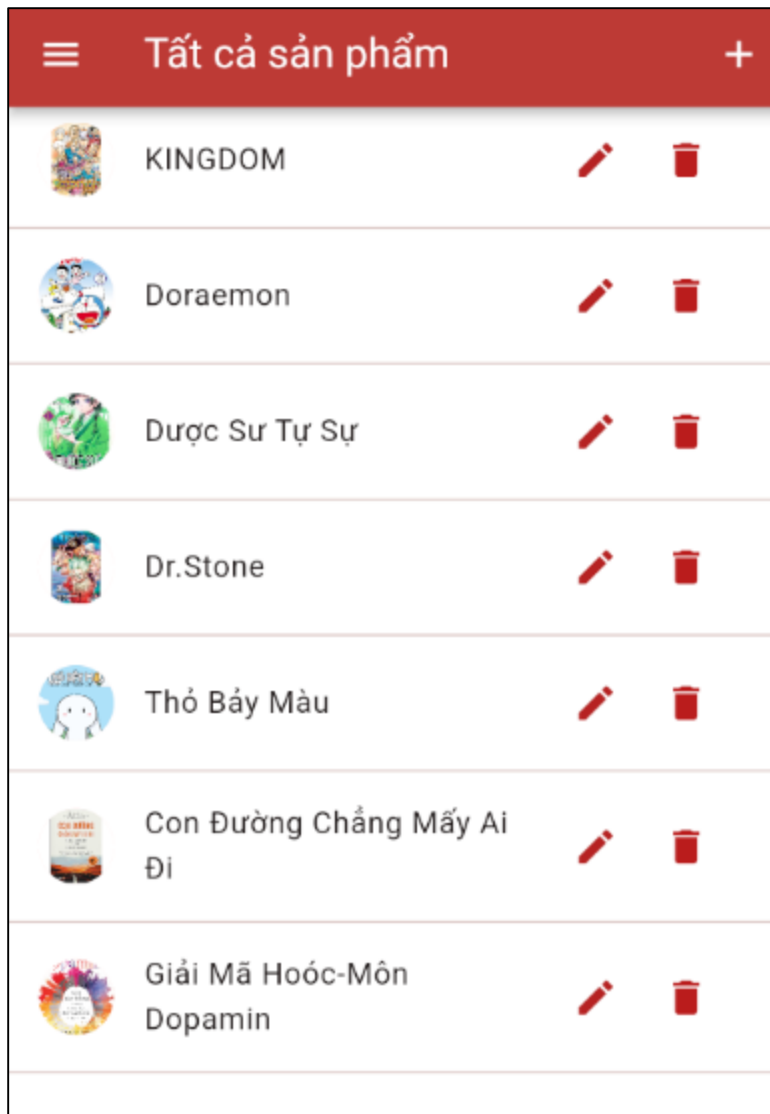
{

```
"creatorId": " 34HXP9c60EeF7IzJW3lGJNPscf33",  
"description": "Review"  
"imageUrl": "https://upload.wikimedia.org/.../1024px-Cast-Iron-  
Pan.jpg",  
"title": "Doraemon"  
"cate": "Truyện tranh"  
"author": "Fuji"  
}
```

Chức năng có sử dụng API của Firebase Realtime Database thông qua Web API Key

12. Chức năng/giao diện 12: Xóa sản phẩm

- **Miêu tả chức năng/giao diện:** Xóa sản phẩm
- **Ảnh chức năng/giao diện:**



Hình 13: Giao diện chức năng quản lý sản phẩm

– **Chi tiết cài đặt:**

- + Các widget được sử dụng: IconButton, ListTile.
- + Chức năng sử dụng đến các thư viện và plugin sau:
 1. flutter/material.dart: Để sử dụng các widget cơ bản trong Flutter.
 2. dart:async: cung cấp các công cụ và lớp cho xử lý bất đồng bộ.
 3. flutter/foundation.dart: cung cấp các lớp và hàm hỗ trợ cho việc phát triển ứng dụng Flutter.
 4. provider/provider.dart: Để sử dụng Provider để quản lý trạng thái và chia sẻ dữ liệu giữa các widget.
 5. dart:developer: cung cấp các chức năng hỗ trợ gỡ lỗi và logging trong Dart.

6. `dart:convert`: cung cấp các hàm và lớp liên quan đến mã hóa và giải mã dữ liệu trong Dart.
7. Plugin `flutter_dotenv/flutter_dotenv.dart` : cung cấp cách tải các biến môi trường từ tệp `.env` trong ứng dụng, giúp quản lý các cài đặt cụ thể cho môi trường phát triển, thử nghiệm và triển khai của ứng dụng Flutter.

+ Chức năng sử dụng giải pháp quản lý trạng thái chia sẻ: Có

1. `ProductsManager`: Đây là một class được tạo ra để quản lý danh sách sản phẩm và trạng thái yêu thích của từng sản phẩm. Class này sử dụng `Provider` để cung cấp trạng thái này cho toàn bộ ứng dụng.
2. `Provider`: được sử dụng để cung cấp dữ liệu sản phẩm và quản lý các thay đổi trong dữ liệu này, truy cập và gọi phương thức trên `'ProductsManager'`.

+ Chức năng này có dùng dịch vụ đọc và lưu trữ dữ liệu của Firebase Realtime Database

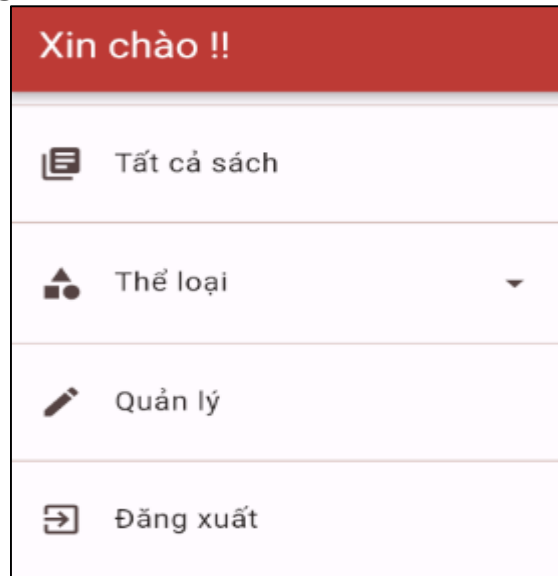
Cấu trúc JSON:

```
{
  "creatorId": " 34HXP9c60EeF7IzJW3IGJNPscf33",
  "description": "Review"
  "imageUrl": "https://upload.wikimedia.org/.../1024px-Cast-Iron-Pan.jpg",
  "title": "Doraemon"
  "cate": "Truyện tranh"
  "author": "Fuji"
}
```

Chức năng có sử dụng API của Firebase Realtime Database thông qua Web API Key

13. Chức năng/giao diện 13: Đăng xuất

- **Miêu tả chức năng/giao diện:** Đăng xuất tài khoản người dùng
- **Ảnh chức năng/giao diện:**



Hình 14: Giao diện chức năng đăng xuất

- **Chi tiết cài đặt:**
 - + Các widget được sử dụng: IconButton, ListTile.
 - + Chức năng sử dụng đến các thư viện và plugin sau:
 1. flutter/material.dart: Để sử dụng các widget cơ bản trong Flutter.
 2. dart:async: cung cấp các công cụ và lớp cho xử lý bất đồng bộ.
 3. flutter/foundation.dart: cung cấp các lớp và hàm hỗ trợ cho việc phát triển ứng dụng Flutter.
 4. provider/provider.dart: Để sử dụng Provider để quản lý trạng thái và chia sẻ dữ liệu giữa các widget.
 5. dart:developer: cung cấp các chức năng hỗ trợ gỡ lỗi và logging trong Dart.
 6. dart:convert: cung cấp các hàm và lớp liên quan đến mã hóa và giải mã dữ liệu trong Dart.
 7. Plugin flutter_dotenv/flutter_dotenv.dart : cung cấp cách tải các biến môi trường từ tệp .env trong ứng dụng, giúp quản lý các cài đặt cụ thể cho môi trường phát triển, thử nghiệm và triển khai của ứng dụng Flutter.
 - + Chức năng có sử dụng giải pháp quản lý trạng thái chia sẻ:
Provider: được sử dụng để cung cấp dữ liệu sản phẩm và quản lý các thay đổi trong dữ liệu này.
 - + Chức năng này không dùng dịch vụ đọc và lưu trữ dữ liệu.