

**BỘ NÔNG NGHIỆP VÀ MÔI TRƯỜNG
TRƯỜNG ĐẠI HỌC THỦY LỢI**



BÀI TẬP LỚN

**PHÁT TRIỂN ÚNG DỤNG CHO THIẾT BỊ DI ĐỘNG
ĐỀ TÀI: ÚNG DỤNG FRUIT STORE**

Giáo viên hướng dẫn: ThS. Kiều Tuấn Dũng

Sinh viên thực hiện:

STT	Mã sinh viên	Họ và tên	Lớp
1	2251061817	Phạm Văn Lâm	64CNTT1
2	2251061756	Trần Ánh Dương	64CNTT1
3	2251061896	Đỗ Thị Quỳnh Trang	64CNTT1
4	2251061831	Đào Cẩm Ly	64CNTT1

Hà Nội, năm 2025

**BỘ NÔNG NGHIỆP VÀ MÔI TRƯỜNG
TRƯỜNG ĐẠI HỌC THỦY LỢI**



BÀI TẬP LỚN

**PHÁT TRIỂN ÚNG DỤNG CHO THIẾT BỊ DI ĐỘNG
ĐỀ TÀI: ỦNG DỤNG FRUIT STORE**

STT	Mã Sinh Viên	Họ và Tên	Ngày Sinh	Điểm	
				Bằng Số	Bằng Chữ
1	2251061817	Phạm Văn Lâm	31-07-2004		
2	2251061756	Trần Ánh Dương	13-06-2004		
3	2251061896	Đỗ Thị Quỳnh Trang	03-06-2004		
4	2251061831	Đào Cẩm Ly	17-02-2004		

CÁN BỘ CHẤM THI

Hà Nội, năm 2025

LỜI NÓI ĐẦU

Trong thời đại công nghệ 4.0, khi mà điện thoại thông minh và Internet đã trở thành một phần không thể thiếu trong đời sống hàng ngày, các hoạt động mua sắm truyền thống đang dần được thay thế bởi các nền tảng mua sắm trực tuyến. Người tiêu dùng ngày nay có xu hướng tìm đến sự tiện lợi, nhanh chóng, và trải nghiệm cá nhân hóa trong quá trình mua sắm. Chính vì thế, việc áp dụng công nghệ thông tin vào hoạt động bán hàng là một xu hướng tất yếu và mang tính cấp thiết đối với các cửa hàng, đặc biệt là các cửa hàng thực phẩm tươi sống như hoa quả, rau củ.

Trong bối cảnh đó, nhóm chúng em đã triển khai xây dựng ứng dụng Fruit Store – một ứng dụng di động hỗ trợ bán hoa quả online dành riêng cho các cửa hàng bán lẻ trong khu vực. Ứng dụng được xây dựng với mục tiêu giúp cửa hàng có thể tiếp cận nhiều khách hàng hơn, tối ưu hóa quy trình bán hàng, đồng thời mang đến cho khách hàng sự tiện lợi khi đặt hàng ngay tại nhà thông qua điện thoại thông minh.

Fruit Store hướng đến việc đơn giản hóa trải nghiệm người dùng với các chức năng cơ bản như: đăng ký tài khoản, đăng nhập, chỉnh sửa thông tin cá nhân (Profile), xem danh sách sản phẩm (trái cây), thêm sản phẩm vào giỏ hàng, và trong các phiên bản tương lai sẽ tích hợp chức năng đặt hàng, thanh toán trực tuyến và theo dõi đơn hàng. Ứng dụng sử dụng Firebase làm nền tảng lưu trữ và quản lý dữ liệu người dùng, đảm bảo độ tin cậy, an toàn và đồng bộ hóa trên nhiều thiết bị.

Việc triển khai ứng dụng không chỉ là cơ hội để chúng em vận dụng những kiến thức đã học về lập trình Android, thiết kế giao diện người dùng, tích hợp Firebase mà còn giúp hiểu sâu hơn về cách xây dựng một hệ thống phần mềm có khả năng phục vụ thực tế và tiềm cận với nhu cầu đời sống. Đồng thời, đây cũng là tiền đề cho việc phát triển các ứng dụng thương mại điện tử chuyên nghiệp hơn trong tương lai.

Mặc dù ứng dụng hiện tại vẫn còn một số hạn chế nhất định như giao diện còn đơn giản, chưa hỗ trợ thanh toán trực tuyến, chưa phù hợp cho mô hình chuỗi cửa hàng, nhưng đây là một bước đi quan trọng trong quá trình học tập và rèn luyện kỹ năng phát triển ứng dụng di động của chúng em.

Với tất cả những lý do trên, nhóm em xin trình bày báo cáo đồ án với chủ đề:

“Xây dựng ứng dụng bán hoa quả trực tuyến trên nền tảng Android sử dụng Firebase”

Rất mong nhận được sự góp ý và đánh giá từ thầy/cô để hoàn thiện sản phẩm ngày càng tốt hơn.

MỤC LỤC

LỜI NÓI ĐẦU	3
MỤC LỤC	4
DANH MỤC CÁC TỪ VIẾT TẮT	5
Chương 1. TỔNG QUAN VỀ ĐỀ TÀI	6
1.1. Giới thiệu về đề tài	6
1.2. Mục tiêu của đề tài	6
1.3. Phạm vi của đề tài	6
1.4. Phân chia nhiệm vụ	6
Chương 2. KIẾN TRÚC VÀ CÔNG NGHỆ	9
2.1. Kiến trúc hệ thống	9
2.2. Giới thiệu về Công nghệ phát triển	10
Chương 3. XÂY DỰNG ỨNG DỤNG	12
3.1. Thiết kế Figma	12
3.2. Thiết kế CSDL	12
3.3. Giao diện ứng dụng	13
3.4. Code minh họa các chức năng cốt lõi	27
KẾT LUẬN	66
TÀI LIỆU THAM KHẢO	68

DANH MỤC CÁC TỪ VIẾT TẮT

STT	TỪ VIẾT TẮT	VIẾT ĐẦY ĐỦ
1	RWD	Responsive Web Design
2	CSDL	Cơ sở dữ liệu
3	MVC	Model – View – Controller
4	MVVM	Model – View – View Model
5	FCM	Firebase Cloud Messaging

Chương 1. TỔNG QUAN VỀ ĐỀ TÀI

1.1.Giới thiệu về đề tài

Fruit Store là một ứng dụng di động giúp người dùng dễ dàng mua sắm hoa quả trực tuyến và quản lý đơn hàng một cách tiện lợi. Ứng dụng cung cấp trải nghiệm mượt mà với giao diện thân thiện, tích hợp nhiều tính năng hỗ trợ cho cả khách hàng (User) và người quản trị (Admin). Với các tính năng đa dạng và hiệu suất mạnh mẽ, ứng dụng hứa hẹn mang đến trải nghiệm tuyệt vời cho cả khách hàng và người quản lý.

1.2.Mục tiêu của đề tài

- Xây dựng một ứng dụng mua sắm hoa quả trực tuyến với giao diện thân thiện, dễ sử dụng cho cả khách hàng và người quản trị.
- Tối ưu hóa trải nghiệm mua sắm của khách hàng :
 - Cho phép khách hàng dễ dàng tìm kiếm, lựa chọn và đặt mua các loại hoa quả trực tuyến.
- Cung cấp thông tin chi tiết về sản phẩm, bao gồm nguồn gốc, xuất xứ, giá cả.
- Nâng cao hiệu quả quản lý và vận hành cửa hàng
 - Quản lý sản phẩm: Cung cấp hệ thống giúp cửa hàng cập nhật, quản lý thông tin sản phẩm như tên, giá cả, số lượng tồn kho, hình ảnh và mô tả sản phẩm.
 - Quản lý đơn hàng: Hỗ trợ nhân viên cửa hàng trong việc tiếp nhận, xử lý và vận chuyển đơn hàng một cách nhanh chóng, chính xác.

1.3.Phạm vi của đề tài

- Ứng dụng này được phát triển dành cho một cửa hàng bán hoa quả trực tuyến, phục vụ khách hàng trong một khu vực cụ thể. Người dùng có thể dễ dàng đặt hàng ngay trên ứng dụng, cập nhật thông tin cá nhân, và tận hưởng trải nghiệm mua sắm tiện lợi.

1.4.Phân chia nhiệm vụ

<<Bảng phân chia nhiệm vụ>>

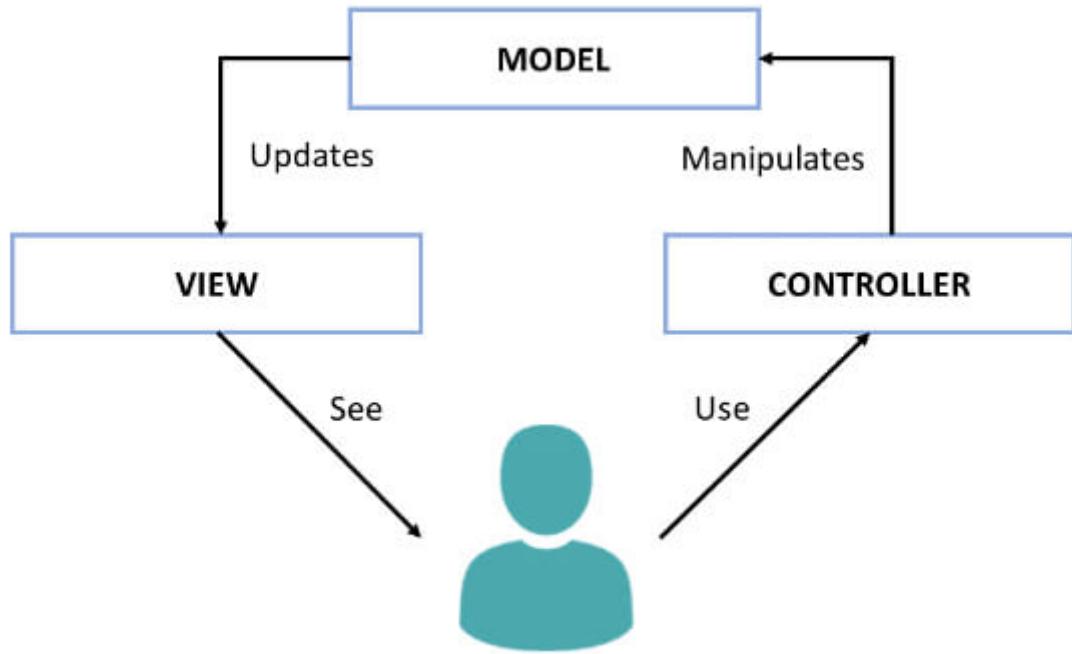
STT	Thành viên	Nhiệm vụ/Chức năng	Mô tả
1	Phạm Văn Lâm	Trang chủ , Đơn hàng , Phân quyền người dùng	<ul style="list-style-type: none"> • Trang chủ: Hiển thị danh sách quả cửa hàng đang bán • Đơn hàng : tạo hóa đơn khi người dùng xác nhận mua hàng , xóa hóa đơn khi nhân viên xác nhận đã thanh toán. • Phân quyền: Cho phép hệ thống phân biệt tài khoản user và admin . User có thể chỉnh sửa thông tin cá nhân , mua hàng. Admin có thêm quyền chỉnh sửa kho hàng và quản lý đơn hàng
2	Trần Ánh Dương	Quản lý giỏ hàng, xem thông tin mặt hàng	<ul style="list-style-type: none"> • Thêm sản phẩm: Người dùng thêm sản phẩm vào giỏ hàng. • Xóa sản phẩm: Người dùng xóa sản phẩm khỏi giỏ hàng • Mua sản phẩm: Người dùng mua sản phẩm, hiển thị giao hàng • Xem thông tin mặt hàng : Hiển

			thị thông tin chi tiết sản phẩm khi người dùng chọn vào sản phẩm
3	Đỗ Thị Quỳnh Trang	Quản lý kho hàng	<ul style="list-style-type: none"> Thêm sản phẩm : quản lý thêm sản phẩm mới Cập nhật sản phẩm: quản lý cập nhật tên, số lượng, giá sản phẩm Xóa sản phẩm : quản lý xóa sản phẩm
4	Đào Cẩm Ly	Đăng ký, Đăng nhập, Thông tin cá nhân	<ul style="list-style-type: none"> Đăng ký: Người dùng nhập thông tin vào tài khoản mới. Đăng Nhập: Người dùng đăng nhập bằng email và mật khẩu Thông tin cá nhân: Hiển thị và cập nhập thông tin cá nhân, thay đổi ảnh đại diện

Chương 2. KIẾN TRÚC VÀ CÔNG NGHỆ

2.1. Kiến trúc hệ thống

Mô hình MVC



1. Model (Dữ liệu)

- Định nghĩa các đối tượng dữ liệu, truy xuất và lưu trữ thông tin từ Firebase
- Vd các lớp dữ liệu : FruitModel, BillModel, UserModel,...
- FruitModel chứa các thuộc tính như: name, price, img_url, description, quantity, unit.

2. View (Giao diện người dùng)

- Hiển thị thông tin giao diện và nhận tương tác từ người dùng như nhập dữ liệu, nhấn nút,...
- Vd : là các activity / Fragment / layout XML như activity_main.xml : hiển thị danh sách các loại trái cây,...

3. Controller (Xử lý yêu cầu và điều phối)

- Xử lý các sự kiện do người dùng thực hiện

- Gọi đến model để xử lý và cập nhật View tương ứng
- Vd : AddFruit, updateFruit,...
- updateFruit.java :
 - Nhận dữ liệu từ View
 - Kiểm tra hợp lệ
 - Cập nhật lên Firebase

2.2. Giới thiệu về Công nghệ phát triển

- Java là một trong những ngôn ngữ lập trình phổ biến nhất, mạnh mẽ và đa nền tảng. Với khả năng “viết một lần, chạy mọi nơi”, Java là lựa chọn hàng đầu trong phát triển ứng dụng Android và các hệ thống backend doanh nghiệp.
- Firebase là nền tảng phát triển ứng dụng toàn diện do Google cung cấp, hỗ trợ mạnh mẽ cho việc xây dựng ứng dụng thời gian thực, bảo mật và dễ mở rộng. Firebase cung cấp nhiều dịch vụ như:
 - Firebase Authentication – Xác thực người dùng (Email, Google, Facebook, v.v.)
 - Firebase Realtime Database – Cơ sở dữ liệu thời gian thực, đồng bộ dữ liệu giữa các client nhanh chóng.
 - Firebase Firestore – Cơ sở dữ liệu NoSQL nâng cao, dễ quản lý và linh hoạt hơn.
 - Firebase Cloud Storage – Lưu trữ file (ảnh, video, tài liệu, v.v.)
 - Firebase Cloud Messaging (FCM) – Gửi thông báo đẩy đến thiết bị.
 - Firebase Analytics, Crashlytics – Theo dõi hành vi người dùng, lỗi ứng dụng.

Lý do chọn Java + Firebase

- Java được hỗ trợ chính thức cho phát triển Android, cộng đồng lớn và tài nguyên phong phú.
- Firebase giúp phát triển ứng dụng nhanh hơn, không cần tự triển khai backend phức tạp.
- Kết hợp Java + Firebase cho phép xây dựng ứng dụng thời gian thực, an toàn và dễ mở rộng chỉ với vài dòng code.

Ứng dụng quản lý người dùng, todo-list, trò chuyện (chat app)

Ứng dụng bán hàng online, tích hợp giỏ hàng và thanh toán

Hệ thống thông báo đẩy và theo dõi hoạt động người dùng.

Ứng dụng vào Bài Tập Lớn

- Ứng dụng cửa hàng bán hoa quả được phát triển thông qua Android studio bằng 100% ngôn ngữ java
- Firebase Authentication: Tạo tài khoản thông qua Email & Password , đăng nhập thông qua Email & Password , xác định tài khoản đang đăng nhập để cập nhật dữ liệu cho người dùng đó
- Firebase Realtime Database: Lưu thông tin người dùng sau khi đăng kí , truy vấn và sử dụng dữ liệu thông tin người dùng theo thời gian thực.
- Firebase Firestore: chứa các csdl (Fruits , Bills , CurrentUser , AddToCart). Sử dụng để lưu trữ , cập nhật , tải xuống dữ liệu khi người dùng sử dụng ứng dụng

Firebase Cloud Storage : lưu trữ các thư mục chứa ảnh(profile_picture , fruit_img).

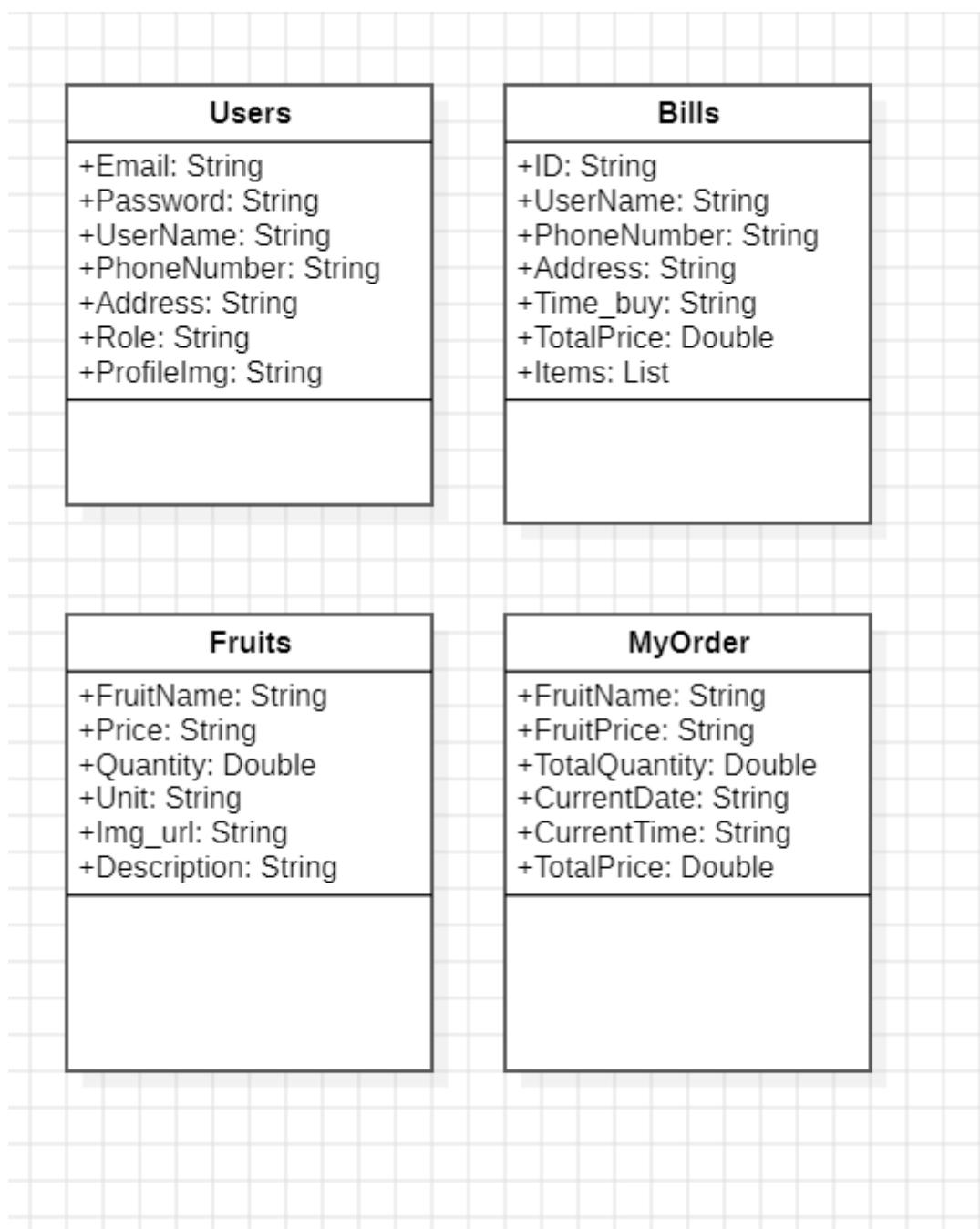
Ứng dụng trong việc cập nhật ảnh đại diện người dùng, thêm ảnh của trái cây.

Chương 3. XÂY DỰNG ỦNG DỤNG

3.1.Thiết kế Figma

<https://www.figma.com/design/ok2viy1dGI2mP4RIPKIfyi/Untitled?node-id=48-68&t=81f5CRLonqiJZFN0-0>

3.2.Thiết kế CSDL



3.3.Giao diện ứng dụng

3.3.1. Màn hình

FMart



Cảm ơn bạn đã ghé thăm! Chúng
tôi cung cấp các loại trái cây tươi
mới theo mùa!

3.3.2. Màn hình đăng ký



Họ Tên

Email

Password

Số Điện Thoại

Địa Chỉ

Đăng Kí

Bạn đã có tài khoản ? **Đăng nhập ngay**

3.3.3. Màn hình đăng nhập



Email

Password

Đăng Nhập

Bạn chưa có tài khoản ? [Đăng kí ngay](#)

3.3.4. Màn hình trang chủ (Admin)

Trang Chủ

Liên hệ chúng tôi nếu có vấn đề
nhé !
Hotline: 0968686868

Trang Chủ

Thông tin tài khoản

Giỏ Hàng

Kho Hàng

Đơn Hàng

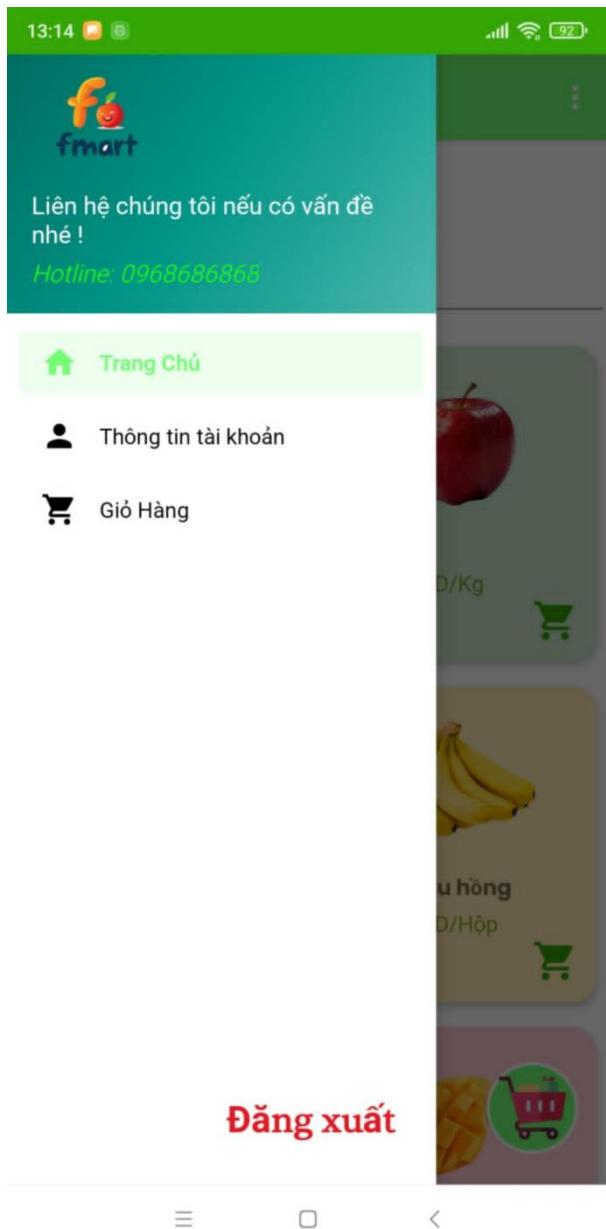
Đăng xuất

Trang Chủ

Danh sách trái cây

Trái Cây	Giá	Mô Tả
Lê Sa Pa	25000 VND/Kg	Lê Sa Pa
Táo Mỹ	55000 VND/Kg	Táo Mỹ
Vải thiều lục ngạn	40000 VND/Kg	Vải thiều lục ngạn
Chuối tiêu hồng	25000 VND/Hộp	Chuối tiêu hồng
Dứa		Dứa
Mango		Mango

Màn hình trang chủ (User)



3.3.5. Màn hình thông tin quả

01:43

75%

← Thông Tin Chi Tiết

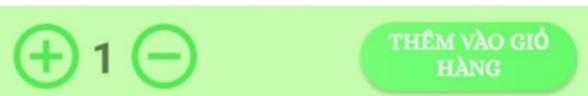


25000 VNĐ/Kg

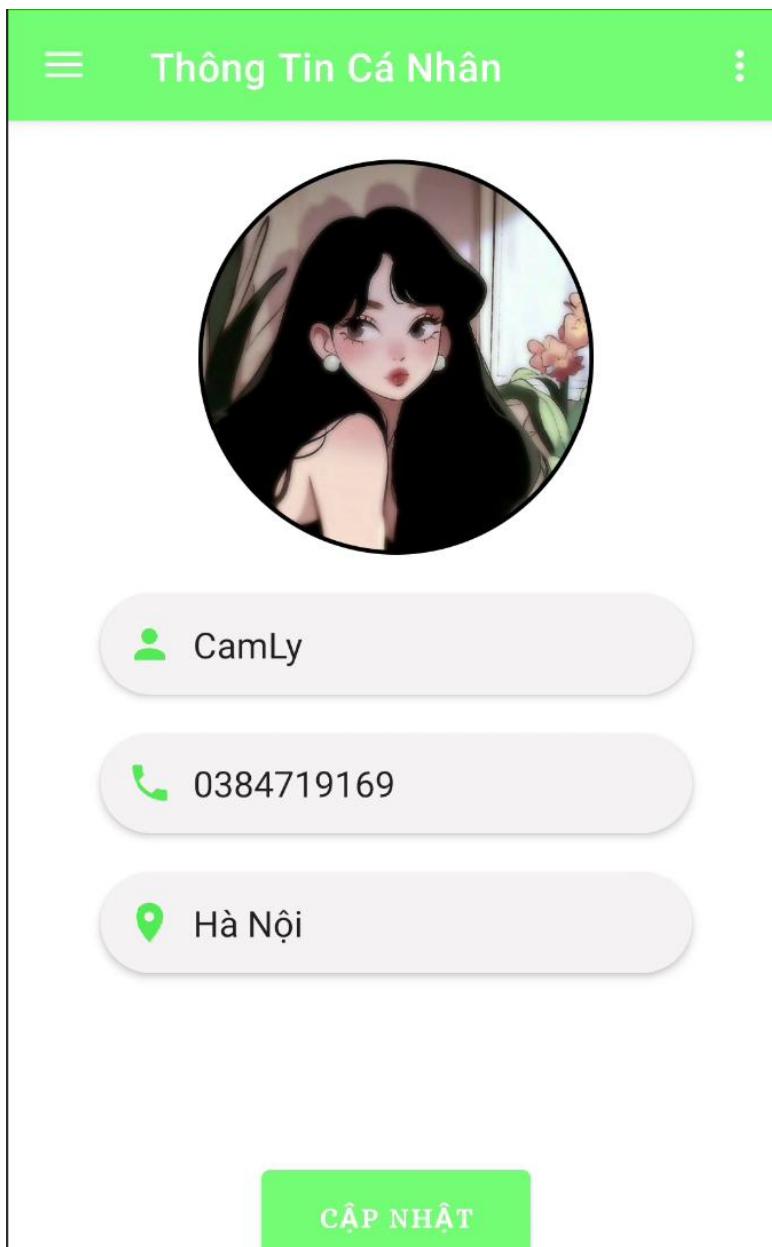
Miêu tả

Ê tơi mọng nước, vỏ mỏng, ruột giòn ngọt và mát lành.
Thích hợp dùng trực tiếp, ép nước, làm salad hoặc tráng miệng.

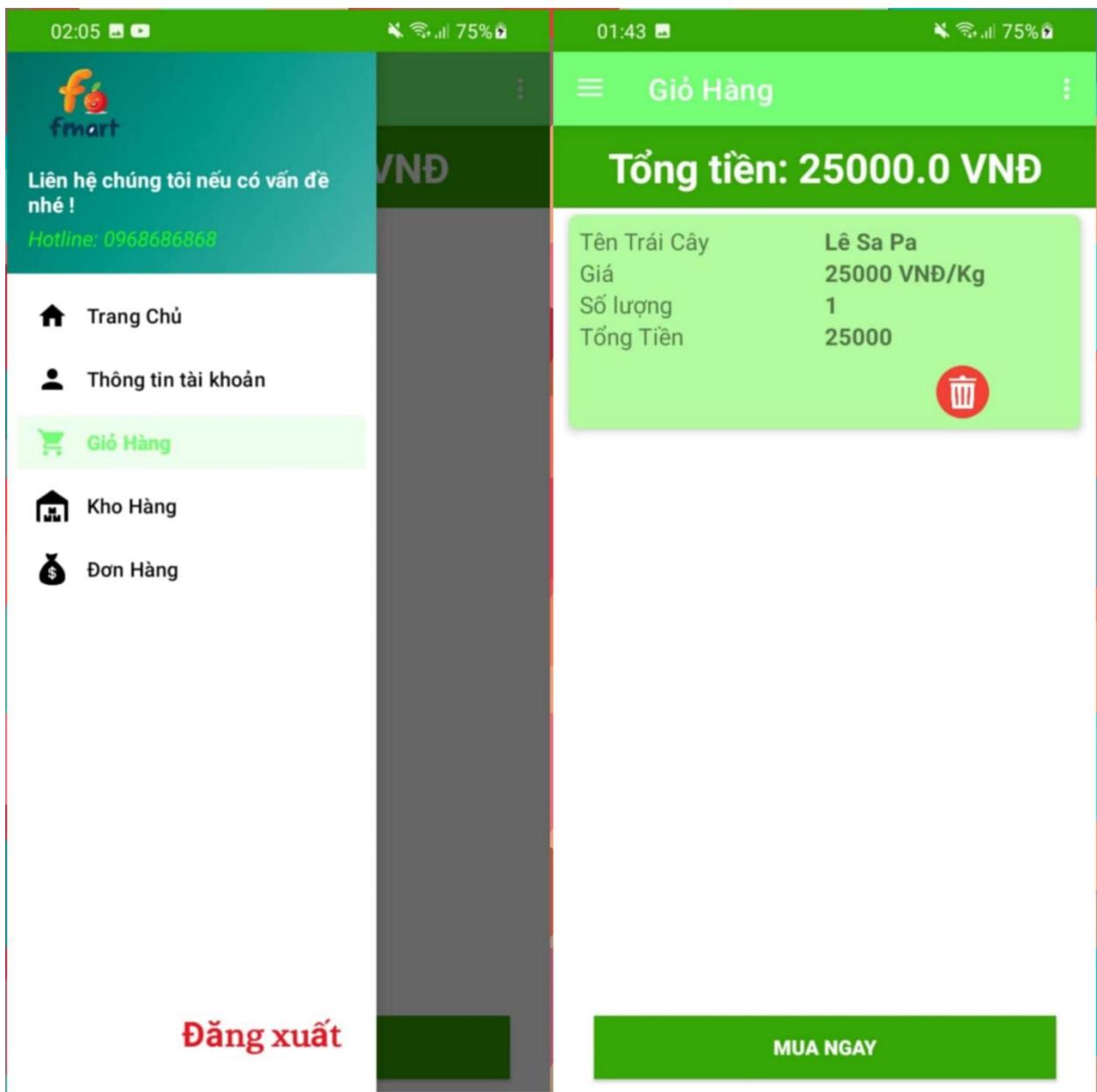
1kg khoảng 3-5 quả
📍 Nguồn gốc: Lê Sa Pa (Lào Cai)



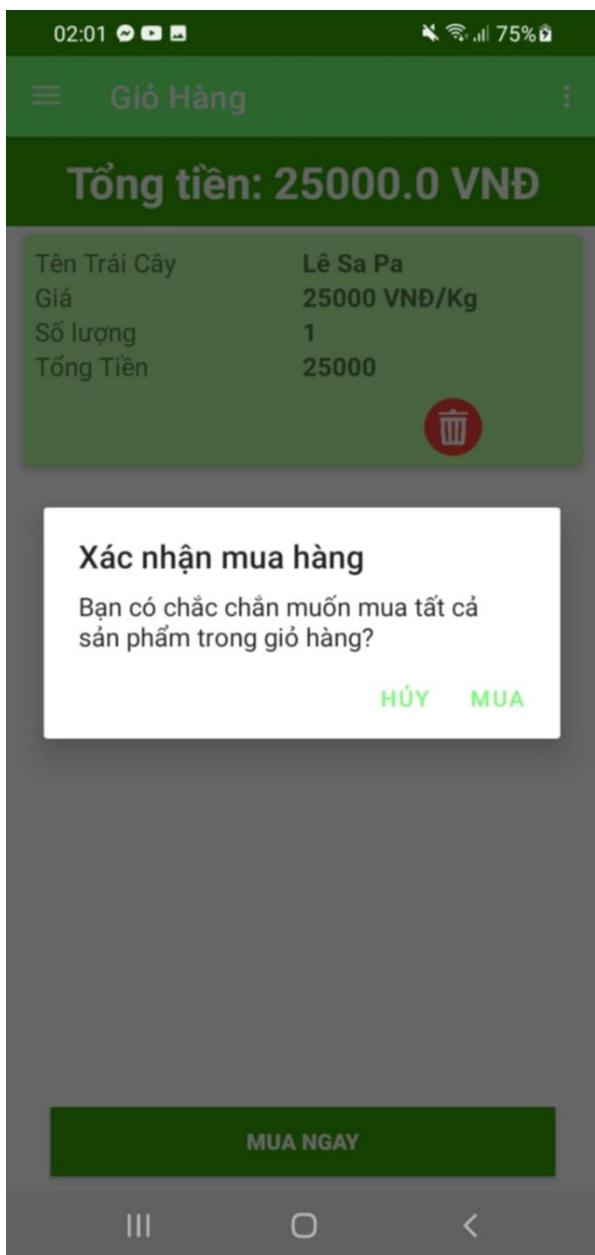
3.3.6. Màn hình thông tin cá nhân



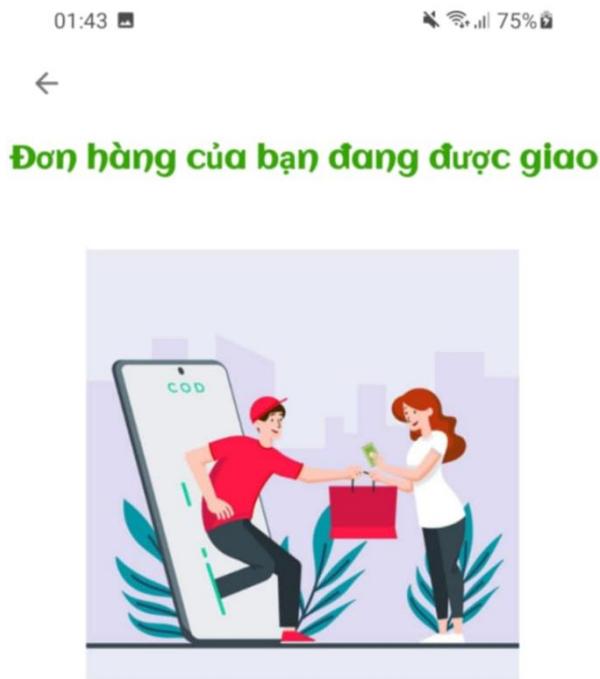
3.3.7. Màn hình giỏ hàng



- Màn hình xác nhận mua hàng:



- Màn hình mua hàng thành công:



Cảm ơn đã lựa chọn sản phẩm của chúng
tôi
Hãy thường xuyên ghé thăm nhé !

3.3.8. Màn hình kho hàng

The screenshot displays the Fimart mobile application interface for managing inventory. The top navigation bar is green with the title "Kho Hàng". On the left side, there's a sidebar with various menu items: "Trang Chủ", "Thông tin tài khoản", "Giỏ Hàng", "Kho Hàng" (which is highlighted in green), and "Đơn Hàng". A red button labeled "Đăng xuất" is located at the bottom of the sidebar. The main content area shows four items in the warehouse:

- Lê Sa Pa**: Price 25000VND/Kg, Quantity 22. Actions: Edit, Delete.
- Táo Mỹ**: Price 55000VND/Kg, Quantity 90. Actions: Edit, Delete.
- Vải thiều lục ngạn**: Price 40000VND/Kg, Quantity 200. Actions: Edit, Delete.
- Chuối tiêu hồng**: Price 25000VND/Hộp. Actions: Edit, Delete.

Each item card includes a small image of the fruit.

- Màn hình thêm sản phẩm mới

Thêm sản phẩm

Tên sản phẩm

Giá

Số lượng Kg ▾

Mô tả

[THÊM](#)

- Màn hình cập nhật sản phẩm
-

← Cập nhật sản phẩm



Lê Sa Pa

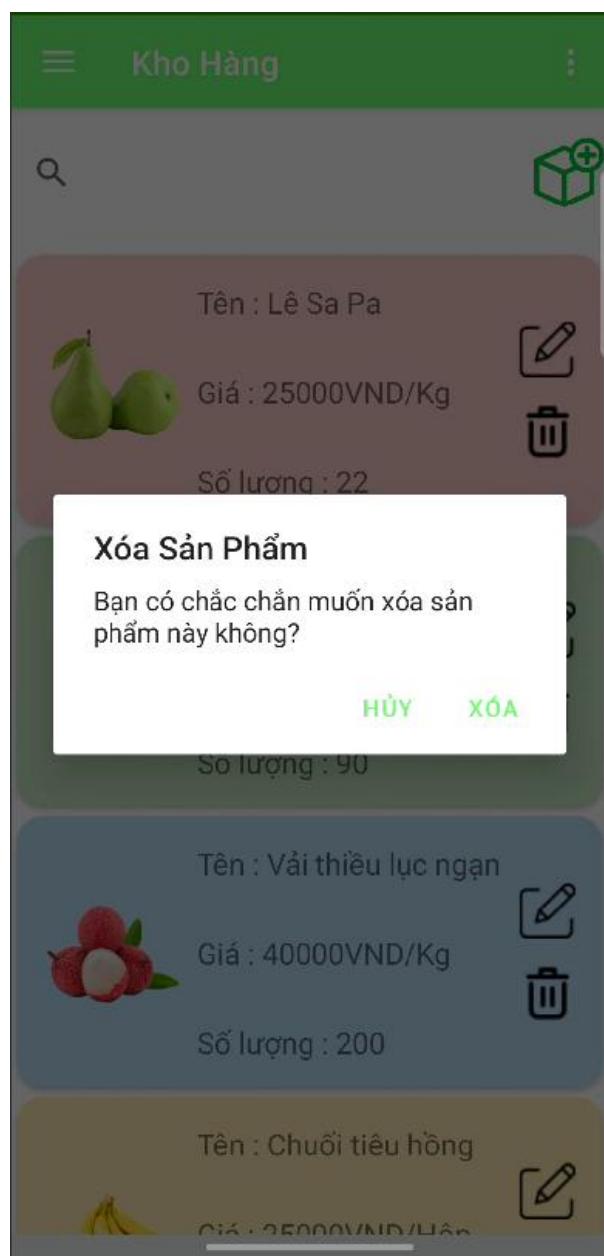
25000

22 Kg ▾

Lê tươi mọng nước, vỏ
mỏng, ruột giòn ngọt và
mát lành. Thích hợp dùng

CẬP NHẬT

- Màn hình xóa sản phẩm



3.3.9. Màn hình đơn hàng

The screenshot displays a mobile application interface for managing fruit orders. At the top, a green header bar contains the title "Đơn Hàng". Below the header, the order number "#000002" is shown. To the right, a section titled "Trái cây đã chọn" (Selected Fruits) lists the items and their details:

Tên trái cây	SL	ĐG	T.Tiền
Táo	1	55000 VNĐ/ Kg	55000 VNĐ
Lê	1	25000 VNĐ/ Kg	25000 VNĐ

A total amount of "Tổng cộng: 80000.0 VNĐ" is displayed at the bottom of this section. On the left side of the screen, there are two large buttons: "Thanh Toán" (Pay) and "Hủy Đơn" (Cancel Order), each accompanied by an empty checkbox. Below these buttons is a green button labeled "Xem chi tiết" (View Details). Further down, another order entry is visible with the number "#000003" and similar fields for customer information and fruit selection.

3.4. Code minh họa các chức năng cốt lõi

3.4.1. Chức năng đăng ký tài khoản

```
1 package com.example.fruit_store.activities;
2
3 > import ...
4
5
6 D>< public class RegistrationActivity extends AppCompatActivity {
7     2 usages
8     private Button bt_sign_up;
9     2 usages
10    private EditText txt_Name;
11    2 usages
12    @O private EditText txt_Email;
13    2 usages
14    private EditText txt_Password;
15    2 usages
16    private EditText txt_Phone_Number;
17    2 usages
18    private EditText txt_Address;
19    2 usages
20    private TextView txt_sign_in;
21
22    2 usages
23    private FirebaseAuth auth;
24    2 usages
25    private FirebaseDatabase database;
26
27     8 usages
28    private ProgressBar progressBar;
29
30
31 @Override
32     protected void onCreate(Bundle savedInstanceState) {
33         super.onCreate(savedInstanceState);
34         EdgeToEdge.enable( $this$enableEdgeToEdge: this );
35         setContentView(R.layout.activity_registration);
36
37         auth = FirebaseAuth.getInstance();
38         database = FirebaseDatabase.getInstance();
39
40         progressBar = (ProgressBar) findViewById(R.id.progressbar);
41         progressBar.setVisibility(View.GONE);
42
43         bt_sign_up = (Button) findViewById(R.id.bt_sign_up);
44         txt_Name = (EditText) findViewById(R.id.txt_name);
45         txt_Email = (EditText) findViewById(R.id.txt_email);
46         txt_Password = (EditText) findViewById(R.id.txt_password);
47         txt_Phone_Number = (EditText) findViewById(R.id.txt_phone_number);
48         txt_Address = (EditText) findViewById(R.id.txt_address);
49         //Khai báo các biến
50         txt_sign_in = (TextView) findViewById(R.id.txt_sign_in);
51         txt_sign_in.setOnClickListener(new View.OnClickListener() {
52
53             @Override
54             public void onClick(View v) {
55                 Intent sign_in_intent = new Intent( packageContext: RegistrationActivity.this, LoginActivity.class );
56                 startActivity(sign_in_intent);
57             }
58         });
59
60         //bắt sự kiện click vào đăng nhập
61         bt_sign_up.setOnClickListener(new View.OnClickListener() {
62             @Override
63             public void onClick(View v) {
64                 Intent sign_in_intent = new Intent( packageContext: RegistrationActivity.this, LoginActivity.class );
65                 startActivity(sign_in_intent);
66             }
67         });
68
69         //bắt sự kiện click vào đăng nhập
70         bt_sign_up.setOnClickListener(new View.OnClickListener() {
71             @Override
72             public void onClick(View v) {
73                 progressBar.setVisibility(View.VISIBLE);
74                 createUser();
75             }
76         });
77         View rootView = findViewById(android.R.id.content);
78         rootView.setOnTouchListener(new View.OnTouchListener() {
79             @Override
80             public boolean onTouch(View v, MotionEvent event) {
81                 hideKeyboard();
82                 return false;
83             }
84         });
85     });
86
87 }
88 1 usage
89 private void createUser() {
90     String userName = txt_Name.getText().toString();
91     String userEmail = txt_Email.getText().toString();
92     String userPassword = txt_Password.getText().toString();
93     String userPhoneNumber = txt_Phone_Number.getText().toString();
94     String userAddress = txt_Address.getText().toString();
95     String userRole = "user";
96     if(TextUtils.isEmpty(userName)){
97         Toast.makeText( context: this, text: "Họ tên đang rỗng", Toast.LENGTH_SHORT).show();
98     }
99 }
```

```

99         //kiểm tra tên người dùng
100        if(TextUtils.isEmpty(userEmail)){
101            Toast.makeText( context: this, text: "Email đang rỗng", Toast.LENGTH_SHORT).show();
102            return;
103        }
104        if(TextUtils.isEmpty(userPassword)){
105            Toast.makeText( context: this, text: "Password đang rỗng", Toast.LENGTH_SHORT).show();
106            return;
107        }
108        //kiểm tra email
109        if(TextUtils.isEmpty(userPhoneNumber)){
110            Toast.makeText( context: this, text: "SDT đang rỗng", Toast.LENGTH_SHORT).show();
111            return;
112        }
113        if(TextUtils.isEmpty(userAddress)){
114            Toast.makeText( context: this, text: "Địa chỉ đang rỗng", Toast.LENGTH_SHORT).show();
115            return;
116        }
117        //kiểm tra địa chỉ
118        if(userPassword.length() < 6){
119            Toast.makeText( context: this, text: "Password phải nhiều hơn 5 ký tự", Toast.LENGTH_SHORT).show();
120            return;
121        }
122        //kiểm tra mật khẩu
123        auth.createUserWithEmailAndPassword(userEmail, userPassword).addOnCompleteListener(new OnCompleteListener<AuthResult>() {
124
125            @Override
126            public void onComplete(@NonNull Task<AuthResult> task) {
127                if(task.isSuccessful()){
128                    UserModel userModel = new UserModel(userName,userEmail,userPassword,userPhoneNumber,userAddress,userRole);
129                    String id = task.getResult().getUser().getUid();
130                    database.getReference().child( pathString: "Users").child(id).setValue(userModel);
131                    progressBar.setVisibility(View.GONE);
132                    Toast.makeText( context: RegistrationActivity.this, text: "Đăng kí thành công", Toast.LENGTH_SHORT).show();
133                }
134                // nếu đăng ký thành công
135                else if(task.isSuccessful()){
136                    progressBar.setVisibility(View.GONE);
137                    Toast.makeText( context: RegistrationActivity.this, text: "Đăng kí thất bại", Toast.LENGTH_SHORT).show();
138                // nếu đăng ký thất bại
139                else if(task.getException() != null){
140                    progressBar.setVisibility(View.GONE);
141                    Toast.makeText( context: RegistrationActivity.this, text: "Email đã tồn tại", Toast.LENGTH_SHORT).show();
142                // nếu email đã tồn tại
143                else if(task.getException() != null){
144                    progressBar.setVisibility(View.GONE);
145                    Toast.makeText( context: RegistrationActivity.this, text: "Đăng kí thất bại", Toast.LENGTH_SHORT).show();
146                // nếu đăng ký thất bại
147                else {
148                    progressBar.setVisibility(View.GONE);
149                    Toast.makeText( context: RegistrationActivity.this, text: "Error" + task.getException(), Toast.LENGTH_SHORT).show();
150                }
151            }
152        });
153    }
154
155    1 usage
156    private void hideKeyboard() {
157        View view = this.getCurrentFocus();
158        if (view != null) {
159            InputMethodManager imm = (InputMethodManager) getSystemService(INPUT_METHOD_SERVICE);
160            imm.hideSoftInputFromWindow(view.getWindowToken(), flags: 0);
161        }
162        // Đưa con trỏ về cuối
163    }

```

3.4.2. Chức năng đăng nhập

```

1 package com.example.fruit_store.activities;
2
3 > import ...
27
28 <> public class LoginActivity extends AppCompatActivity {
29     2 usages
30     private Button bt_sign_in;
31     2 usages
32     private EditText txt_email_login;
33     6 usages
34     private EditText txt_password_login;
35     2 usages
36     private TextView txt_sign_up;
37
38     5 usages
39     private ProgressBar progressBar;
40     2 usages
41     private FirebaseAuth auth;
42
43     4 usages
44     private ImageView iv_toggle_password;
45     3 usages
46     private boolean isPasswordVisible = false;
47     @Override
48     protected void onCreate(Bundle savedInstanceState) {
49         super.onCreate(savedInstanceState);
50         EdgeToEdge.enable( $this$enableEdgeToEdge: this);
51         setContentView(R.layout.activity_login);
52
53         auth = FirebaseAuth.getInstance();
54         progressBar = (ProgressBar) findViewById(R.id.progressbar);
55         progressBar.setVisibility(View.GONE);
56
57         bt_sign_in = (Button) findViewById(R.id.bt_sign_in);
58         txt_email_login = (EditText) findViewById(R.id.txt_email_login);
59         txt_password_login =(EditText) findViewById(R.id.txt_password_login);
60         txt_sign_up =(TextView) findViewById(R.id.txt_sign_up);
61         //khai báo các biến
62
63         txt_sign_up.setOnClickListener(new View.OnClickListener() {
64             @Override
65             public void onClick(View v) {
66                 Intent sign_up_intent = new Intent( packageContext: LoginActivity.this, RegistrationActivity.class);
67                 //bắt sự kiện click vào đăng ký
68                 startActivity(sign_up_intent);
69             }
70         });
71
72         bt_sign_in.setOnClickListener(new View.OnClickListener() {
73             @Override
74             public void onClick(View v) {
75                 progressBar.setVisibility(View.VISIBLE);
76                 //bắt sự kiện click vào đăng nhập
77                 loginUser();
78             }
79         });
80
81         View rootView = findViewById(android.R.id.content);
82         rootView.setOnTouchListener(new View.OnTouchListener() {
83             @Override
84             public boolean onTouch(View v, MotionEvent event) {
85                 hideKeyboard();
86                 // Án bàn phím khi chạm ra ngoài EditText
87                 return false;
88             }
89
90         });
91

```

```

82     iv_toggle_password = findViewById(R.id.iv_toggle_password);
83     iv_toggle_password.setOnClickListener(new View.OnClickListener() {
84         @Override
85         public void onClick(View v) {
86             if (isPasswordVisible) {
87                 // Ẩn mật khẩu
88                 txt_password_login.setInputType(InputType.TYPE_CLASS_TEXT | InputType.TYPE_TEXT_VARIATION_PASSWORD);
89                 iv_toggle_password.setImageResource(R.drawable.eye); // icon mắt đóng
90             } else {
91                 // Hiển thị mật khẩu
92                 txt_password_login.setInputType(InputType.TYPE_CLASS_TEXT | InputType.TYPE_TEXT_VARIATION_VISIBLE_PASSWORD);
93                 iv_toggle_password.setImageResource(R.drawable.hidden); // icon mắt mở
94             }
95             isPasswordVisible = !isPasswordVisible;
96             txt_password_login.setSelection(txt_password_login.getText().length()); // đưa con trỏ về cuối
97         }
98     });
99 }
100 }
101 usage
102 private void hideKeyboard() {
103     View view = this.getCurrentFocus();
104     if (view != null) {
105         InputMethodManager imm = (InputMethodManager) getSystemService(INPUT_METHOD_SERVICE);
106         imm.hideSoftInputFromWindow(view.getWindowToken(), flags: 0);
107     }
108 }
109
110 private void loginUser() {
111     String userEmail = txt_email_login.getText().toString(); // Lấy email từ EditText
112     String userPassword = txt_password_login.getText().toString(); // Lấy mật khẩu từ EditText
113
114     if(TextUtils.isEmpty(userEmail)){
115         Toast.makeText(context: this, text: "Email đang rỗng", Toast.LENGTH_SHORT).show();
116         return;
117     }
118     // Kiểm tra định dạng email
119     if(TextUtils.isEmpty(userPassword)){
120         Toast.makeText(context: this, text: "Password đang rỗng", Toast.LENGTH_SHORT).show();
121         return;
122     }
123     // Kiểm tra định dạng mật khẩu
124     if(userPassword.length() < 6){
125         Toast.makeText(context: this, text: "Password phải chứa nhiều hơn 5 ký tự", Toast.LENGTH_SHORT).show();
126         return;
127     }
128     // Kiểm tra độ dài mật khẩu
129 }
130
131 auth.signInWithEmailAndPassword(userEmail, userPassword)
132     .addOnCompleteListener(new OnCompleteListener<AuthResult>() {
133         @Override
134         public void onComplete(@NonNull Task<AuthResult> task) {
135             if(task.isSuccessful()){
136                 progressBar.setVisibility(View.GONE);
137                 startActivity(new Intent(packageContext: LoginActivity.this, MainActivity.class));
138                 Toast.makeText(context: LoginActivity.this, text: "Đăng nhập thành công", Toast.LENGTH_SHORT).show();
139             }
140             // Nếu đăng nhập thành công, chuyển đến MainActivity
141             else{
142                 progressBar.setVisibility(View.GONE);
143                 String error = task.getException() != null ? task.getException().getMessage() : "Đăng nhập thất bại";
144                 Toast.makeText(context: LoginActivity.this, text: "Vui lòng kiểm tra lại thông tin tài khoản", Toast.LENGTH_SHORT).show();
145             }
146             // Nếu đăng nhập thất bại, hiển thị thông báo lỗi
147         }
148     });
149 }
150 no usages
151 private void test() {
152     // Test code here
153     // This is just a placeholder for your test code
154     Toast.makeText(context: this, text: "Test code executed", Toast.LENGTH_SHORT).show();
155 }

```

3.4.3. Thông tin cá nhân

```
1 package com.example.fruit_store.ui.Profile;                                     ▲2
2
3 > import ...
40
41 <> public class ProfileFragment extends Fragment {
42     7 usages
43     private CircleImageView img_profile;
44     5 usages
45     private EditText txt_name;
46     5 usages
47     private EditText txt_phone;
48     5 usages
49     private EditText txt_address;
50     2 usages
51     private Button bt_update;
52     2 usages
53     private FirebaseStorage storage;
54     1 usage
55     private FirebaseAuth auth;
56     6 usages
57     private FirebaseDatabase database;
58     2 usages
59     private ActivityResultLauncher<Intent> imagePickerLauncher;
60     3 usages
61     private ProgressBar progressBar;
62
63     public View onCreateView(@NonNull LayoutInflater inflater,
64         ViewGroup container, Bundle savedInstanceState) {
65         View root = inflater.inflate(R.layout.fragment_profile, container, attachToRoot: false);
66         // bat su kien click vao nut cap nhat
67         img_profile = (CircleImageView) root.findViewById(R.id.profile_img);
68         img_profile.setVisibility(View.GONE);
69         txt_name = (EditText) root.findViewById(R.id.profile_name);
70         txt_name.setVisibility(View.GONE);
71         txt_phone = (EditText) root.findViewById(R.id.profile_phone);
72         txt_phone.setVisibility(View.GONE);
73         txt_address = (EditText) root.findViewById(R.id.profile_address);
74         txt_address.setVisibility(View.GONE);
75         bt_update = (Button) root.findViewById(R.id.bt_update);
76         progressBar = (ProgressBar) root.findViewById(R.id.prf_progressbar);
77         progressBar.setVisibility(View.VISIBLE);
78         storage = FirebaseStorage.getInstance();
79         auth = FirebaseAuth.getInstance();
80         database = FirebaseDatabase.getInstance();
81
82         imagePickerLauncher = registerForActivityResult(
83             new ActivityResultContracts.StartActivityForResult(), result -> {
84                 if(result.getResultCode() == Activity.RESULT_OK && result.getData() != null){
85                     Uri profileUri = result.getData().getData();
86                     img_profile.setImageURI(profileUri);
87                     // upload hinh anh len firebase
88                     final StorageReference reference = storage.getReference().child( pathString: "profile_picture")
89                         .child(FirebaseAuth.getInstance().getUid());
90                     // lay duong dan hinh anh
91                     reference.putFile(profileUri).addOnSuccessListener(new OnSuccessListener<UploadTask.TaskSnapshot>() {
92                         @Override
93                         public void onSuccess(UploadTask.TaskSnapshot taskSnapshot) {
94                             Toast.makeText(getApplicationContext() , text: "Uploaded" , Toast.LENGTH_SHORT).show();
95                             reference.getDownloadUrl().addOnSuccessListener(new OnSuccessListener<Uri>() {// lay duong dan hinh anh
96                                 @Override
97                                 public void onSuccess(Uri uri) {
98                                     database.getReference().child( pathString: "Users").child(FirebaseAuth.getInstance().getUid())
99                                         .child( pathString: "profileImg").setValue(uri.toString());
100                                     Toast.makeText(getApplicationContext() , text: "Profile picture uploaded" , Toast.LENGTH_SHORT).show();
101                                 }
102                             // lay duong dan hinh anh
103                         });
104                     });
105                 }
106             });
107     }
108 }
```

```

101    bt_update.setOnClickListener(new View.OnClickListener() {
102        @Override
103        public void onClick(View v) { updateUserProfile(); }
104    });
105    // chon hinh anh lam avatar tu may
106    img_profile.setOnClickListener(new View.OnClickListener() {
107        @Override
108        public void onClick(View v) {
109            Intent intent = new Intent();
110            intent.setAction(Intent.ACTION_GET_CONTENT);
111            intent.setType("image/*");
112            imagePickerLauncher.launch(intent);
113        }
114        // bat su kien click vao hinh anh
115    });
116    // load du lieu tu firebase
117    database.getReference().child( pathString: "Users").child(FirebaseAuth.getInstance().getUid())
118        .addValueEventListener(new ValueEventListener() {
119            2 usages
120            @Override
121            public void onDataChange(@NonNull DataSnapshot snapshot) {
122                UserModel userModel = snapshot.getValue(UserModel.class);
123                if (userModel.getProfileImg() != null && !userModel.getProfileImg().isEmpty()) {
124                    Glide.with(getApplicationContext()).load(userModel.getProfileImg()).into(img_profile);
125                } else {
126                    img_profile.setImageResource(R.drawable.profile);
127                }
128            }
129            // set du lieu vao cac edittext
130            txt_name.setText(userModel.getName());
131            txt_phone.setText(userModel.getPhoneNumber());
132            txt_address.setText(userModel.getAddress());
133            progressBar.setVisibility(View.GONE);
134            img_profile.setVisibility(View.VISIBLE);
135            txt_name.setVisibility(View.VISIBLE);
136            txt_phone.setVisibility(View.VISIBLE);
137            txt_address.setVisibility(View.VISIBLE);
138        }
139        // lay duong dan hinh anh
140
141        @Override
142        public void onCancelled(@NonNull DatabaseError error) {
143
144        });
145    // bat su kien click vao nut cap nhat
146    root.setOnClickListener(new View.OnClickListener() {
147        @Override
148        public boolean onTouch(View v, MotionEvent event) {
149            hideKeyboard(v);
150            return false;
151        }
152    });
153    // bat su kien click vao nut cap nhat
154    return root;
155 }
156
157 }

129 // set du lieu vao cac edittext
130 txt_name.setText(userModel.getName());
131 txt_phone.setText(userModel.getPhoneNumber());
132 txt_address.setText(userModel.getAddress());
133 progressBar.setVisibility(View.GONE);
134 img_profile.setVisibility(View.VISIBLE);
135 txt_name.setVisibility(View.VISIBLE);
136 txt_phone.setVisibility(View.VISIBLE);
137 txt_address.setVisibility(View.VISIBLE);
138
139 // lay duong dan hinh anh
140
141 @Override
142 public void onCancelled(@NonNull DatabaseError error) {
143
144 });
145
146 // bat su kien click vao nut cap nhat
147
148 root.setOnClickListener(new View.OnClickListener() {
149     @Override
150     public boolean onTouch(View v, MotionEvent event) {
151         hideKeyboard(v);
152         return false;
153     }
154 });
155 // bat su kien click vao nut cap nhat
156 return root;
157

```

```

159     ^
160     private void uploadImageToFireBase(Uri profileUri) {
161     }
162
163     1 usage
164     private void updateUserProfile() {
165         String name = txt_name.getText().toString();
166         String phone = txt_phone.getText().toString();
167         String address = txt_address.getText().toString();
168         // kiểm tra xem cac edittext co rong hay khong
169         if (name.isEmpty() || phone.isEmpty() || address.isEmpty()) {
170             Toast.makeText(getApplicationContext(), text: "Cần nhập đủ dữ liệu", Toast.LENGTH_SHORT).show();
171             return;
172         }
173         // kiểm tra xem cac edittext co rong hay khong
174         database.getReference().child( pathString: "Users").child(FirebaseAuth.getInstance().getUid())
175             .child( pathString: "name").setValue(name);
176         database.getReference().child( pathString: "Users").child(FirebaseAuth.getInstance().getUid())
177             .child( pathString: "phoneNumber").setValue(phone);
178         database.getReference().child( pathString: "Users").child(FirebaseAuth.getInstance().getUid())
179             .child( pathString: "address").setValue(address);
180
181         Toast.makeText(getApplicationContext(), text: "Cập nhật thành công", Toast.LENGTH_SHORT).show();
182
183     } // bat su kien click vao nut cap nhat
184     1 usage
185     private void hideKeyboard(View view) {
186         InputMethodManager imm = (InputMethodManager) getSystemService(Context.INPUT_METHOD_SERVICE);
187         if (imm != null) {
188             imm.hideSoftInputFromWindow(view.getWindowToken(), flags: 0);
189         }
190         // Ẩn bàn phím
191     }

```

3.4.4. Trang chủ

3.4.3.1 Khởi tạo biến

```
public class HomeFragment extends Fragment {

    5 usages
    private RecyclerView recyclerView_fruit;
    4 usages
    private FruitAdapters fruitAdapter;
    6 usages
    private List<FruitModel> fruitModelList , filteredList;
    2 usages
    private FirebaseFirestore db;
    2 usages
    private SearchView searchBox;
    3 usages
    private ProgressBar progressBar;
    2 usages
    private ImageButton bt_go_to_cart;

    public View onCreateView(@NonNull LayoutInflater inflater,
                           ViewGroup container, Bundle savedInstanceState) {

        View root = inflater.inflate(R.layout.fragment_home, container, attachToRoot: false);
        db = FirebaseFirestore.getInstance();
        searchBox = root.findViewById(R.id.search_box);
        bt_go_to_cart = root.findViewById(R.id.btn_go_to_cart);
        progressBar = root.findViewById(R.id.home_progressbar);
        progressBar.setVisibility(View.VISIBLE);
        filteredList = new ArrayList<>();
        recyclerView_fruit = root.findViewById(R.id.fruit_recyclerView);
        recyclerView_fruit.setVisibility(View.GONE);
        recyclerView_fruit.setLayoutManager(new GridLayoutManager(getContext(), spanCount: 2));
        fruitModelList = new ArrayList<>();
        fruitAdapter = new FruitAdapters(getActivity(), filteredList);
        recyclerView_fruit.setAdapter(fruitAdapter);
    }
}
```

3.4.3.2. Load dữ liệu từ firebase hiển thị lên recyclerview thông qua fruitAdapter

```
// lay du lieu tu firebase truyen vao fruitModelList , filterList de loc tim kiem
db.collection( collectionPath: "Fruits" ) CollectionReference
    .get() Task<QuerySnapshot>
    .addOnCompleteListener(new OnCompleteListener<QuerySnapshot>() {
        @Override
        public void onComplete(@NonNull Task<QuerySnapshot> task) {
            if(task.isSuccessful()){
                fruitModelList.clear();
                filteredList.clear();
                for(QueryDocumentSnapshot document : task.getResult()){
                    FruitModel fruitModel = document.toObject(FruitModel.class);
                    fruitModelList.add(fruitModel);

                }
                filteredList.addAll(fruitModelList);
                fruitAdapter.notifyDataSetChanged();
                progressBar.setVisibility(View.GONE);
                recyclerView_fruit.setVisibility(View.VISIBLE);
            }
            else {
                Toast.makeText(getApplicationContext() , text: "Error" + task.getException(),Toast.LENGTH_SHORT);
            }
        }
    });
}
```

3.4.3.3 Tìm kiếm quả

```
// Bắt sự kiện nhập vào SearchView
searchBox.setOnQueryTextListener(new SearchView.OnQueryTextListener() {
    1 usage
    @Override
    public boolean onQueryTextSubmit(String query) {
        filter(query);
        return true;
    }

    1 usage
    @Override
    public boolean onQueryTextChange(String newText) {
        filter(newText);
        return true;
    }
});

2 usages
private void filter(String text) {
    filteredList.clear();
    if (text.isEmpty()) {
        filteredList.addAll(fruitModelList);
    } else {
        for (FruitModel fruit : fruitModelList) {
            if (fruit.getName().toLowerCase().contains(text.toLowerCase())) {
                filteredList.add(fruit);
            }
        }
    }
    fruitAdapter.notifyDataSetChanged();
}
```

3.4.4.4. Tạo nút để người dùng bấm chuyển đến trang giỏ hàng

```
    ...
    bt_go_to_cart.setOnClickListener(v -> reloadCartFragment());

private void reloadCartFragment() {
    if (getActivity() instanceof androidx.fragment.app.FragmentActivity) {
        androidx.fragment.app.FragmentActivity activity = (androidx.fragment.app.FragmentActivity) getActivity();
        NavController navController = Navigation.findNavController(activity, R.id.nav_host_fragment_content_main);

        // Xóa toàn bộ backstack trước khi điều hướng
        navController.popBackStack(R.id.nav_home, inclusive: true);

        // Điều hướng đến giỏ hàng
        navController.navigate(R.id.nav_my_cart);
    }
}
```

3.4.4.5. fruitAdapter

```
1 package com.example.fruit_store.Adapters;
2
3 > import ...
22
23 public class FruitAdapters extends RecyclerView.Adapter<FruitAdapters.FruitViewHolder>{
24     7 usages
25     private List<FruitModel> fruitsList;
26     4 usages
27     private Context context;
28     2 usages
29     private int[] colors = {
30         Color.parseColor(colorString: "#FFCDD2"), // Đỏ nhạt
31         Color.parseColor(colorString: "#C8E6C9"), // Xanh lá nhạt
32         Color.parseColor(colorString: "#BBDEFB"), // Xanh dương nhạt
33         Color.parseColor(colorString: "#FFECB3"), // Vàng nhạt
34         Color.parseColor(colorString: "#D1C4E9") // Tím nhạt
35     };
36
37     1 usage
38     public FruitAdapters(Context context, List<FruitModel> fruitsList) {
39         this.context = context;
40         this.fruitsList = fruitsList;
41     }
42
43     4 usages
44     public static class FruitViewHolder extends RecyclerView.ViewHolder {
45         2 usages
46         ImageView fruitImage;
47         2 usages
48         TextView fruitName, fruitPrice;
49         2 usages
50         View fruit_item_background;
51
52         1 usage
53         public FruitViewHolder(View itemView) {
54             super(itemView);
55             fruitImage = itemView.findViewById(R.id.fruitImage);
56             fruitName = itemView.findViewById(R.id.fruitName);
57             fruitPrice = itemView.findViewById(R.id.fruitPrice);
58             fruit_item_background = itemView.findViewById(R.id.fruit_linearlayout);
59         }
60     }
61 }
```

```
55     @Override
56     public FruitViewHolder onCreateViewHolder(@NotNull ViewGroup parent, int viewType) {
57         View view = LayoutInflater.from(parent.getContext()).inflate(R.layout.item_fruit, parent, attachToRoot: false)
58         return new FruitViewHolder(view);
59     }
60
61     @Override
62     public void onBindViewHolder(@NotNull FruitViewHolder holder, int position) {
63         Glide.with(context).load(fruitsList.get(position).getImg_url()).into(holder.fruitImage);
64         holder.fruitName.setText(fruitsList.get(position).getName());
65         holder.fruitPrice.setText( fruitsList.get(position).getPrice() + " VND/" + fruitsList.get(position).getUnit())
66         int colorIndex = position % colors.length;
67         holder.fruit_item_background.setBackgroundColor(colors[colorIndex]);
68
69         holder.itemView.setOnClickListener(new View.OnClickListener() {
70             @Override
71             public void onClick(View v) {
72                 int pos = holder.getAdapterPosition();
73                 Intent intent = new Intent(context, DetailActivity.class);
74                 intent.putExtra(name: "detail", fruitsList.get(pos));
75                 context.startActivity(intent);
76             }
77         });
78     }
79 }
```

@Override

```
> public int getItemCount() { return fruitsList.size(); }
```

3.4.5. Chức năng hiển thị chi tiết thông tin quả (DetailActivity.java)

3.4.4.1 Khởi tạo

```
//Hàm khởi tạo
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    EdgeToEdge.enable( $this$enableEdgeToEdge: this);
    //Khởi tạo các biến và liên kết với XML
    setContentView(R.layout.activity_detail);

    detailImg = (ImageView) findViewById(R.id.detail_img);
    txt_price = (TextView) findViewById(R.id.detail_price);
    txt_description = (TextView) findViewById(R.id.content_description);
    bt_addToCart = (Button) findViewById(R.id.add_to_cart);
    img_addFruit = (ImageView) findViewById(R.id.add_fruit);
    img_removeFruit = (ImageView) findViewById(R.id.remove_fruit);
    firestore = FirebaseFirestore.getInstance();
    auth = FirebaseAuth.getInstance();

    toolbar = (Toolbar) findViewById(R.id.detail_toolbar);

    // bat nut back
    setSupportActionBar(toolbar);
    getSupportActionBar().setDisplayHomeAsUpEnabled(true);
    toolbar.setNavigationOnClickListener(view -> onBackPressed());

    txt_quantity = (TextView) findViewById(R.id.quantity);
```

```
// lay du lieu tu intent = key detail
final Object object = getIntent().getSerializableExtra( name: "detail");
if(object instanceof FruitModel){
    fruitModel = (FruitModel) object;
}
```

3.4.4.2 Hiển thị thông tin lên giao diện

```
//Hiển thị thông tin sản phẩm lên giao diện
if(fruitModel != null){
    Glide.with(getApplicationContext()).load(fruitModel.getImg_url()).into(detailImg);
    txt_description.setText(fruitModel.getDescription());
    txt_price.setText(fruitModel.getPrice() + " VND/" + fruitModel.getUnit());
    Max_quantity =fruitModel.getQuantity();
}
```

3.4.4.3 Thay đổi số lượng quả khi nhấn cộng trừ

```

//Thay đổi giá trị số lượng sản phẩm khi nhấn nút cộng hoặc trừ
img_addFruit.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        total_quantity++;
        txt_quantity.setText(String.valueOf(total_quantity));
    }
});

img_removeFruit.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        if(total_quantity > 1){
            total_quantity--;
            txt_quantity.setText(String.valueOf(total_quantity));
        }
    }
});

```

3.4.4.4 Xử lý nút thêm vào giỏ hàng

```

//Xử lí nút Thêm sản phẩm vào giỏ hàng
bt_addToCart.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        if(total_quantity < Max_quantity){
            total_price = Integer.parseInt(fruitModel.getPrice()) * total_quantity;
            added_to_cart();
        }
        else{
            cant_buy(fruitModel.getName(), Max_quantity, fruitModel.getUnit());
        }
    }
});

```

3.4.4.5 Hàm thêm vào giỏ hàng, đẩy dữ liệu lên firebase firestore

```

private void added_to_cart() {
    String saveCurrentDate, savecurrentTime;
    Calendar calForDate = Calendar.getInstance();

    // dd là ngày trong tháng , MM là tháng , DD là ngày trong năm , mm là phút
    SimpleDateFormat currentDate = new SimpleDateFormat( pattern: "dd/MM/yyyy" , Locale.getDefault());
    saveCurrentDate = currentDate.format(calForDate.getTime());

    SimpleDateFormat currentTime = new SimpleDateFormat( pattern: "HH:mm:ss");
    savecurrentTime = currentTime.format(calForDate.getTime());

    // bảng bam chưa du lieu cac thuoc tinh cua fruitModel
    final HashMap<String , Object> cartMap = new HashMap<>();

    //Thêm dữ liệu vào Firestore
    cartMap.put("fruitName", fruitModel.getName());
    cartMap.put("fruitPrice", txt_price.getText().toString());
    cartMap.put("currentDate", saveCurrentDate);
    cartMap.put("currentTime", savecurrentTime);
    cartMap.put("totalQuantity", total_quantity);
    cartMap.put("totalPrice", total_price);

    // day du lieu cua bang bam len firebase firestore
    firestore.collection( collectionPath: "CurrentUser").document(auth.getCurrentUser().getUid())
        .collection( collectionPath: "AddToCart").add(cartMap).addOnCompleteListener(
            new OnCompleteListener<DocumentReference>() {
                @Override
                public void onComplete(@NonNull Task<DocumentReference> task) { finish(); }
            });
}

```

3.4.4.5 Hiển thị thông báo khi số lượng hàng không đủ

```

//Hàm hiển thị thông báo khi không đủ số lượng sản phẩm trong kho
1 usage
private void cant_buy(String fruit_name , int fruit_quantity, String fruit_unit) {
    new AlertDialog.Builder( context: this)
        .setTitle("Xin lỗi quý khách")
        .setMessage("Sản phẩm " + fruit_name + " không đủ số lượng trong kho. " +
            "Số lượng hiện có là " + fruit_quantity + " " + fruit_unit + ".")
        .setPositiveButton( text: "OK", (dialog, which) -> dialog.dismiss())
        .show();
}

```

3.4.6. Chức năng mua hàng thông qua giỏ hàng

3.4.6.1 Thêm sản phẩm vào giỏ hàng

(*DetailActivity.java*)

```

//Xử lý nút Thêm sản phẩm vào giỏ hàng
bt_addToCart.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        if(total_quantity < Max_quantity){
            total_price = Integer.parseInt(fruitModel.getPrice()) * total_quantity;
            added_to_cart();
        }
        else{
            cant_buy(fruitModel.getName(), Max_quantity,fruitModel.getUnit());
        }
    }
});
```

3.4.6.2 Hiển thị từng sản phẩm, xóa sản phẩm khỏi giỏ, cập nhật tổng tiền sau khi xóa
(*MyCartAdapter.java*)

- Hiển thị từng sản phẩm

```

public void onBindViewHolder(@NonNull ViewHolder holder, int position) {

    holder.name.setText(myCartModelList.get(position).getFruitName());
    holder.price.setText(String.valueOf(myCartModelList.get(position).getFruitPrice()));

    holder.quantity.setText(String.valueOf(myCartModelList.get(position).getTotalQuantity()));
    holder.total_price.setText(String.valueOf(myCartModelList.get(position).getTotalPrice()));

    holder.img_delete.setOnClickListener(new View.OnClickListener() {
```

- Xử lý nút Xóa,

Khi người dùng nhấn xóa, dữ liệu trong Firestore cũng sẽ bị xóa.

```

holder.img_delete.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        int pos = holder.getAdapterPosition();
        firestore.collection("CurrentUser").document(auth.getCurrentUser().getUid()).DocumentReference
            .collection("AddToCart") CollectionReference
            .document(myCartModelList.get(pos).getDocumetnId()) DocumentReference
            .delete() Task<Void>
            .addOnCompleteListener(new OnCompleteListener<Void>() {
                @Override
                public void onComplete(@NonNull Task<Void> task) {
                    if(task.isSuccessful()){
                        myCartModelList.remove(myCartModelList.get(pos));
                        notifyDataSetChanged(); //Làm mới Recycleview
                        updateTotalAmount(); //cập nhật tổng tiền
                        FragmentManager fragmentManager = ((AppCompatActivity) context).getSupportFragmentManager();
                        MyCartFragment myCartFragment = (MyCartFragment) fragmentManager.findFragmentByTag("MY_CART_FRAGMENT_TAG");
                        if (myCartFragment != null) {
                            myCartFragment.calculatorTotalAmount(myCartModelList);
                        } else {
                            Log.e("MyCartAdapter", "myCartFragment is null");
                        }
                        Toast.makeText(context, text: "Đã xóa" , Toast.LENGTH_SHORT).show();
                    }
                    else {
                        Toast.makeText(context, text: "Error: " + task.getException() , Toast.LENGTH_SHORT).show();
                    }
                }
            });
    }
});

```

- Cập nhật tổng tiền trong giỏ hàng

```

1 usage
private void updateTotalAmount() {
    double totalAmount = 0.0;
    for (MyCartModel myCartModel : myCartModelList) {
        totalAmount += myCartModel.getTotalPrice();
    }
    overTotalAmount.setText("Tổng tiền: " + totalAmount + " VND");
}

```

- Lưu trữ các view con trong mỗi item giỏ hàng.

```

public class ViewHolder extends RecyclerView.ViewHolder {
    2 usages
    TextView name , price , date , time , quantity , total_price;
    2 usages
    ImageView img_delete;
    1 usage
    public ViewHolder(@NonNull View itemView) {
        super(itemView);
        name = (TextView) itemView.findViewById(R.id.txt_cart_fruitName);
        price = (TextView) itemView.findViewById(R.id.txt_cart_price);
        quantity = (TextView) itemView.findViewById(R.id.txt_cart_total_quantity);
        total_price = (TextView) itemView.findViewById(R.id.txt_cart_total_price);
        img_delete = (ImageView) itemView.findViewById(R.id.img_delete);
    }
}

```

3.4.6.3 Hiển thị giỏ hàng, tính tổng tiền, mua hàng, tạo hóa đơn , xóa giỏ hàng sau khi mua (*MyCartFragment.java*)

- Load giỏ hàng từ Firestore:

```

if (auth.getCurrentUser() != null) {
    // Lấy danh sách sản phẩm trong giỏ hàng từ Firestore
    firestore.collection( collectionPath: "CurrentUser" ).document(auth.getCurrentUser().getUid())
        .collection( collectionPath: "AddToCart" ).get() Task<QuerySnapshot>
        .addOnCompleteListener(new OnCompleteListener<QuerySnapshot>() {

```

- Tính tổng tiền:

```

public void calculateTotalAmount(List<MyCartModel> myCartModelList) {
    // Tính toán tổng tiền từ danh sách sản phẩm trong giỏ hàng
    double totalAmount = 0.0;
    for (MyCartModel myCartModel: myCartModelList){
        totalAmount += myCartModel.getTotalPrice();
    }
    overTotalAmount.setText("Tổng tiền: " + totalAmount + " VNĐ");
}

```

- Xử lí khi nhấn nút mua:

Nếu giỏ hàng rỗng → hiển thị thông báo.

Nếu có sản phẩm sẽ hiển thị hộp thoại xác nhận.

```
bt_buy.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        if(myCartModelList.isEmpty()){
            new android.app.AlertDialog.Builder(getContext())
                .setTitle("Giỏ hàng trống")
                .setMessage("Bạn chưa có sản phẩm nào trong giỏ hàng!")
                .setPositiveButton( text: "OK", listener: null)
                .show();
            return;
        }
        else{
            // Nếu giỏ hàng không trống, hiển thị thông báo xác nhận mua hàng
            new android.app.AlertDialog.Builder(getContext())
                .setTitle("Xác nhận mua hàng")
                .setMessage("Bạn có chắc chắn muốn mua tất cả sản phẩm trong giỏ hàng?")
                .setPositiveButton( text: "Mua", (dialog, which) -> {
                    // Khi người dùng xác nhận mua
                    Intent intent = new Intent(getContext(), ThanksActivity.class);
                    updateToBills();
                    getTime();
                    intent.putExtra( name: "fruitList", (Serializable) myCartModelList);
                    startActivityForResult(intent);
                })
                .setNegativeButton( text: "Hủy", listener: null)
                .show();
        }
    }
});
```

- Xóa toàn bộ sản phẩm trong giỏ sau khi tạo hóa đơn thành công:

```

// Xóa tất cả sản phẩm trong giỏ hàng
1 usage
private void deleteAllCart(){
    String userId = auth.getCurrentUser().getUid();
    firestore.collection( collectionPath: "CurrentUser" ).document(userId) DocumentRe
        .collection( collectionPath: "AddToCart" ) CollectionReference
        .get() Task<QuerySnapshot>
        .addOnSuccessListener(queryDocumentSnapshots -> {
            for (QueryDocumentSnapshot document : queryDocumentSnapshots){
                document.getReference().delete();
            }
            myCartModelList.clear();
            calculatorTotalAmount(myCartModelList);
            myCartAdapter.notifyDataSetChanged();
        });
}

```

- Cập nhật thông tin hóa đơn:

```

private void updateToBills() {
    String userID = auth.getCurrentUser().getUid();
    totalPrice = 0.0; // Khởi tạo tránh lỗi null

    database.getReference( path: "Users" ).child(userID).addListenerForSingleValueEvent(new ValueEventListener() {
        2 usages
        @Override
        public void onDataChange(@NonNull DataSnapshot snapshot) {
            if (snapshot.exists()) {
                name = snapshot.child( path: "name" ).getValue(String.class);
                phone = snapshot.child( path: "phoneNumber" ).getValue(String.class);
                address = snapshot.child( path: "address" ).getValue(String.class);

                // Khi dữ liệu đã được tải về, ta mới tạo bill
                createBill(name, phone, address);
            } else {
                Log.e( tag: "updateToBills", msg: "Không tìm thấy dữ liệu người dùng!" );
            }
        }

        @Override
        public void onCancelled(@NonNull DatabaseError error) {
            Log.e( tag: "updateToBills", msg: "Lỗi truy vấn dữ liệu: " + error.getMessage() );
        }
    });
}

```

- Tạo hóa đơn lưu vào Firebase Firestore:

```

// tạo hóa đơn
1 usage
private void createBill(String name, String phone, String address) {
    List<Map<String, Object>> items = new ArrayList<>();
    Log.d( tag: "createBill", msg: "Số lượng sản phẩm trong giỏ hàng: " + myCartModelList.size());
    for (MyCartModel cartItem : myCartModelList) {
        Map<String, Object> item = new HashMap<>();
        item.put("fruitName", cartItem.getFruitName());
        item.put("fruitPrice", cartItem.getFruitPrice());
        item.put("totalQuantity", cartItem.getTotalQuantity());
        item.put("totalPrice", cartItem.getTotalPrice());
        item.put("currentDate", cartItem.getCurrentDate());
        item.put("currentTime", cartItem.getCurrentTime());
        items.add(item);

        totalPrice += cartItem.getTotalPrice();
    }

}

Log.d( tag: "createBill", msg: "Tổng giá trị đơn hàng: " + totalPrice);
Log.d( tag: "createBill", msg: "Số lượng items trong danh sách: " + items.size());
// Lấy bill ID cuối cùng từ Firestore
firestore.collection( collectionPath: "Bills" ) CollectionReference
    .orderBy( field: "id", com.google.firebaseio.Query.Direction.DESCENDING ) Query
    .limit(1)
    .get() Task<QuerySnapshot>
    .addOnSuccessListener(queryDocumentSnapshots -> {
        String newBillId;
        if (!queryDocumentSnapshots.isEmpty()) {
            String lastBillId = queryDocumentSnapshots.getDocuments().get(0).getString( field: "id");
            int lastNumber = Integer.parseInt(lastBillId);
            newBillId = String.format("%06d", lastNumber + 1); // Định dạng thành 000001, 000002
        } else {
            newBillId = "000001"; // Nếu chưa có hóa đơn nào, bắt đầu từ 000001
        }
    }

    // Tạo BillModel với ID mới
    BillModel bill = new BillModel(newBillId, name, phone, address, totalPrice, items , time_buy);

    firestore.collection( collectionPath: "Bills").document(newBillId) DocumentReference
        .set(bill) Task<Void>
        .addOnSuccessListener(unused ->{
            Log.d( tag: "Firestore", msg: "Bill added successfully");
            deleteAllCart();
        })
        .addOnFailureListener(e -> Log.e( tag: "Firestore", msg: "Lỗi khi thêm bill: " + e.getMessage()));
    }
    .addOnFailureListener(e -> Log.e( tag: "Firestore", msg: "Lỗi lấy ID bill: " + e.getMessage()));
}

```

- Lấy thời gian hiện tại:

```

// Hàm này dùng để lấy thời gian hiện tại
1 usage
public void getTime(){
    String saveCurrentDate, savecurrentTime;
    Calendar calForDate = Calendar.getInstance();
    // dd là ngày trong tháng , MM là tháng , DD là ngày trong năm , mm là phút
    SimpleDateFormat currentDate = new SimpleDateFormat( pattern: "dd/MM/yyyy" , Locale.getDefault());
    saveCurrentDate = currentDate.format(calForDate.getTime());
    SimpleDateFormat currentTime = new SimpleDateFormat( pattern: "HH:mm:ss");
    savecurrentTime = currentTime.format(calForDate.getTime());
    time_buy = saveCurrentDate + " " + savecurrentTime;
}

```

3.4.7. Chức năng đơn hàng

3.4.5.1 Load dữ liệu các đơn hàng từ firebase thông qua BillsAdapter

```

public void loadBills() {
    firestore.collection( collectionPath: "Bills")
        .addSnapshotListener(new EventListener<QuerySnapshot>() {
            @Override
            public void onEvent(@Nullable QuerySnapshot value, @Nullable FirebaseFirestoreException error) {
                list_bill.clear();
                if (value != null) {
                    for (QueryDocumentSnapshot doc : value) {
                        try {
                            // do cù bao lỗi không convert được sang Map nên phải theo này
                            String id = doc.getString( field: "id");
                            String name = doc.getString( field: "name");
                            String phone = doc.getString( field: "phone");
                            String address = doc.getString( field: "address");
                            Double totalPrice = doc.getDouble( field: "totalPrice");
                            List<Map<String, Object>> items = (List<Map<String, Object>>) doc.get("items");
                            String time_buy = doc.getString( field: "time_buy");
                            if (items == null) {
                                items = new ArrayList<>();
                            }

                            BillModel bill_info = new BillModel(id, name, phone, address, totalPrice, items, time_buy);
                            list_bill.add(bill_info);
                        } catch (Exception e) {
                            Log.e( tag: "Firestore" , msg: "Lỗi khi chuyển đổi dữ liệu hóa đơn: " + e.getMessage());
                        }
                    }
                    billsAdapter.notifyDataSetChanged();
                }
                progressBar.setVisibility(View.GONE);
                rcv_bill.setVisibility(View.VISIBLE);
            }
        });
}

```

3.4.5.2 BillsAdapter

```
1 package com.example.fruit_store.Adapters;
2
3 > import ...
30
31     6 usages
32     public class BillsAdapter extends RecyclerView.Adapter<BillsAdapter.BillsViewHolder> {
33         7 usages
34         private List<BillModel> list_bill;
35         9 usages
36         private Context context;
37         5 usages
38         private FirebaseFirestore firestore;
39         no usages
40         private BillFragment billFragment;
41         1 usage
42         public BillsAdapter(Context context, List<BillModel> list_bill) {
43             this.context = context;
44             this.list_bill = list_bill;
45             this.firestore = FirebaseFirestore.getInstance();
46         }
47
48     @NotNull
49     @Override
50     public BillsAdapter.BillsViewHolder onCreateViewHolder(@NotNull ViewGroup parent, int viewType) {
51         View view = LayoutInflater.from(parent.getContext()).inflate(R.layout.item_bill, parent, attachToRoot: false
52         return new BillsViewHolder(view);
53     }
54
55     ...
56
57     @Override
58     public void onBindViewHolder(@NotNull BillsAdapter.BillsViewHolder holder, int position) {
59         BillModel bill = list_bill.get(position);
60
61         holder.txtBillId.setText("#" + bill.getId());
62         holder.txtBillName.setText("Tên Khách Hàng: " + bill.getName());
63         holder.txtBillPhone.setText("SĐT: " + bill.getPhone());
64         holder.txtBillAddress.setText("Địa chỉ giao hàng: " + bill.getAddress());
65         holder.txtBillPrice.setText("Tổng tiền: " + bill.getTotalPrice() + " VND");
66         holder.txtTimeBuy.setText(" " + bill.getTime_buy());
67         // Tạo intent để truyền dữ liệu chi tiết hóa đơn vào billInfoActivity
68         holder.itemView.setOnClickListener(v -> {
69             Intent intent = new Intent(context, BillInfoActivity.class);
70             intent.putExtra(name: "bill_items", (Serializable) bill.getItems());
71             intent.putExtra(name: "total_price", bill.getTotalPrice());
72             context.startActivity(intent);
73         });
74         // Xử lý CheckBox thanh toán
75         holder.checkBoxPayment.setOnCheckedChangeListener(null);
76         holder.checkBoxPayment.setChecked(false);
77         holder.checkBoxPayment.setOnCheckedChangeListener((buttonView, isChecked) -> {
78
79     }
```

```

if (isChecked) {
    String billId = bill.getId();
    updateFruitStock(bill.getItems() , () ->{
        firestore.collection( collectionPath: "Bills").document(billId) DocumentReference
            .delete() Task<Void>
            .addOnSuccessListener(aVoid -> {
                Log.d( tag: "DeleteBill" , msg: "delete: " + billId);
                int removedPosition = holder.getAdapterPosition();
                if (removedPosition != RecyclerView.NO_POSITION && removedPosition < list_bill.size()) {
                    list_bill.remove(removedPosition);
                    notifyItemRemoved(removedPosition);

                    Toast.makeText(context, text: "Đã thanh toán và cập nhật kho!", Toast.LENGTH_SHORT).show();
                }
                // phai reload tai cu xoa het cac hoadon sau cua hoa don can xoa, nhung tren firebase lai khong xoa
                reloadBillFragment();
            })
            .addOnFailureListener(e -> {
                Toast.makeText(context, text: "Lỗi khi xóa hóa đơn!", Toast.LENGTH_SHORT).show();
                holder.checkBoxPayment.setChecked(false); // Bỏ chọn nếu xóa thất bại
            });
}

```

```

91     // Xử lý CheckBox hủy hóa đơn
92     holder.checkBoxCancleBill.setOnCheckedChangeListener(null);
93     holder.checkBoxCancleBill.setChecked(false);
94     holder.checkBoxCancleBill.setOnCheckedChangeListener((buttonView, isChecked) -> {
95         if (isChecked) {
96             String billId = bill.getId();
97             firestore.collection( collectionPath: "Bills").document(billId) DocumentReference
98                 .delete() Task<Void>
99                 .addOnSuccessListener(aVoid -> {
100                     Log.d( tag: "DeleteBill" , msg: "delete: " + billId);
101                     int removedPosition = holder.getAdapterPosition();
102                     if (removedPosition != RecyclerView.NO_POSITION && removedPosition < list_bill.size()) {
103                         list_bill.remove(removedPosition);
104                         notifyItemRemoved(removedPosition);
105                         Toast.makeText(context, text: "Đã hủy hóa đơn!", Toast.LENGTH_SHORT).show();
106                     }
107                     reloadBillFragment();
108                 })
109                 .addOnFailureListener(e -> {
110                     Toast.makeText(context, text: "Lỗi khi xóa hóa đơn!", Toast.LENGTH_SHORT).show();
111                     holder.checkBoxCancleBill.setChecked(false); // Bỏ chọn nếu xóa thất bại
112                 });
113     });
114 }

```

```

@Override
public int getItemCount() { return list_bill.size(); }

```

```
+-- dagger
public class BillsViewHolder extends RecyclerView.ViewHolder{
    2 usages
    TextView txtBillId, txtBillName, txtBillPhone, txtBillAddress, txtBillPrice, txtInfo , txtTimeBuy;
    5 usages
    CheckBox checkBoxPayment , checkBoxCancleBill;
    1 usage
    public BillsViewHolder(@NotNull View itemView) {
        super(itemView);
        txtBillId = itemView.findViewById(R.id.txt_bill_id);
        txtBillName = itemView.findViewById(R.id.txt_bill_name_customer);
        txtBillPhone = itemView.findViewById(R.id.txt_bill_phone);
        txtBillAddress = itemView.findViewById(R.id.txt_bill_address);
        txtBillPrice = itemView.findViewById(R.id.txt_bill_price);
        txtTimeBuy = itemView.findViewById(R.id.txt_time);
        txtInfo = itemView.findViewById(R.id.txt_info);
        checkBoxPayment = itemView.findViewById(R.id.chb_payment);
        checkBoxCancleBill = itemView.findViewById(R.id.chb_cancle_bill);
    }
}

139 // Cập nhật tồn kho trái cây
1 usage
140 @ private void updateFruitStock(List<Map<String, Object>> items, Runnable onSuccess) {
141     for (Map<String, Object> item : items) {
142         String fruitName = (String) item.get("fruitName");
143         Object totalQuantityObj = item.get("totalQuantity");
144         int totalQuantity;
145         if (totalQuantityObj instanceof Number) {
146             totalQuantity = ((Number) totalQuantityObj).intValue();
147         } else {
148             Log.e( tag: "UpdateStock", msg: "Lỗi: totalQuantity không hợp lệ!");
149             return;
150         }

151         firestore.collection( collectionPath: "Fruits").whereEqualTo( field: "name", fruitName) Query
152             .get() Task<QuerySnapshot>
153                 .addOnSuccessListener(queryDocumentSnapshots -> {
154                     if (!queryDocumentSnapshots.isEmpty()) {
155                         String documentId = queryDocumentSnapshots.getDocuments().get(0).getId();
156                         Long maxQuantity = queryDocumentSnapshots.getDocuments().get(0).getLong( field: "quantity")
157                         if (maxQuantity != null && maxQuantity >= totalQuantity) {
```

```
160     int newQuantity = maxQuantity.intValue() - totalQuantity;
161
162     firestore.collection(collectionPath: "Fruits").document(documentId) DocumentReference
163         .update( field: "quantity", newQuantity) Task<Void>
164             .addOnSuccessListener(aVoid -> {
165                 Log.d( tag: "UpdateStock", msg: "Cập nhật tồn kho thành công cho " + fruitName);
166                 onSuccess.run(); // ✅ chạy tiếp khi cập nhật kho thành công
167             })
168             .addOnFailureListener(e ->
169                 Log.e( tag: "UpdateStock", msg: "Lỗi cập nhật tồn kho: " + e.getMessage()));
170         } else {
171             Log.e( tag: "UpdateStock", msg: "Số lượng tồn kho không hợp lệ cho " + fruitName);
172         }
173     }
174 }
175 .addOnFailureListener(e -> Log.e( tag: "UpdateStock", msg: "Lỗi truy vấn sản phẩm: " + e.getMessage()));
176
177
178 // Reload lại BillFragment
179 2 usages
180 private void reloadBillFragment() {
181     if (context instanceof androidx.fragment.app.FragmentActivity) {
182         androidx.fragment.app.FragmentActivity activity = (androidx.fragment.app.FragmentActivity) context;
183         NavController navController = Navigation.findNavController(activity, R.id.nav_host_fragment_content_main);
184         navController.navigate(R.id.nav_bill); // Điều hướng lại về chính nó
185     }
186 }
187
188 }
```

3.4.5.3 Hiển thị chi tiết đơn hàng khi bấm vào đơn hàng : load dữ liệu vào recyclerview trong BillInfoActivity thông qua Bill_Info_Adapter

```
1 package com.example.fruit_store.activities;
2
3 > import ...
16
17 >< public class BillInfoActivity extends AppCompatActivity {
18
19     3 usages
20     private RecyclerView rcvBillItems;
21     2 usages
22     private Bill_Info_Adapter billInfoAdapter;
23     2 usages
24     private List<Map<String, Object>> billItems;
25     2 usages
26     private TextView txtTotalPrice;
27     3 usages
28     private Toolbar toolbar;
29
30     @Override
31     protected void onCreate(Bundle savedInstanceState) {
32         super.onCreate(savedInstanceState);
33         setContentView(R.layout.activity_bill_info);
34
35         rcvBillItems = findViewById(R.id.rcv_bill_items);
36         txtTotalPrice = findViewById(R.id.txt_total_price);
37         toolbar = (Toolbar) findViewById(R.id.tb_bill_info);
38         // bat nut back
39         setSupportActionBar(toolbar);
40         getSupportActionBar().setDisplayShowTitleEnabled(false);
41         getSupportActionBar().setDisplayHomeAsUpEnabled(true);
42         toolbar.setNavigationOnClickListener(view -> onBackPressed());
43
44         // Nhận dữ liệu từ Intent ở billsAdapter
45         billItems = (List<Map<String, Object>>) getIntent().getSerializableExtra( name: "bill_items");
46         double totalPrice = getIntent().getDoubleExtra( name: "total_price", defaultValue: 0);
47
48         // Hiển thị tổng tiền
49         txtTotalPrice.setText("Tổng cộng: " + totalPrice + " VND");
50
51         // Cấu hình RecyclerView
52         rcvBillItems.setLayoutManager(new LinearLayoutManager( context: this));
53         billInfoAdapter = new Bill_Info_Adapter( context: this, billItems);
54         rcvBillItems.setAdapter(billInfoAdapter);
55
56     }
57 }
```

3.4.5.4 Bill_Info_Adapter

```
1 package com.example.fruit_store.Adapters;
2
3 > import ...
4 usages
5
6 public class Bill_Info_Adapter extends RecyclerView.Adapter<Bill_Info_Adapter.BillItemViewHolder> {
7     2 usages
8     private Context context;
9     3 usages
10    private List<Map<String, Object>> billItems;
11
12    1 usage
13    public Bill_Info_Adapter(Context context, List<Map<String, Object>> billItems) {
14        this.context = context;
15        this.billItems = billItems;
16    }
17
18
19
20    @NonNull
21    @Override
22    public BillItemViewHolder onCreateViewHolder(@NonNull ViewGroup parent, int viewType) {
23        View view = LayoutInflater.from(context).inflate(R.layout.item_bill_info, parent, attachToRoot: false);
24        return new BillItemViewHolder(view);
25    }
26
27
28    @Override
29    public void onBindViewHolder(@NonNull BillItemViewHolder holder, int position) {
30        Map<String, Object> item = billItems.get(position);
31
32        String name = (String) item.get("fruitName");
33        Number totalQuantityObj = (Number) item.get("totalQuantity");
34        // neu totalQuantityObj != null thi lay gia tri = 0
35        int totalQuantity = totalQuantityObj != null ? totalQuantityObj.intValue() : 0;
36        String price = (String) item.get("fruitPrice");
37        Number totalPriceObj = (Number) item.get("totalPrice");
38        int totalPrice = totalPriceObj != null ? totalPriceObj.intValue() : 0;
39
40        holder.txtName.setText(name);
41        holder.txtQuantity.setText(String.valueOf(totalQuantity));
42        holder.txtPrice.setText(price);
43        holder.txtTotal.setText(String.format("%d VND", totalPrice));
44    }
45
46
47
48
49
```

```
49
50     @Override
51     <@t> > public int getItemCount() { return billItems.size(); }
54
55     4 usages
56     <@t> > public static class BillItemViewHolder extends RecyclerView.ViewHolder {
57         2 usages
58             TextView txtName, txtQuantity, txtPrice, txtTotal ;
59
60             1 usage
61             public BillItemViewHolder(@NonNull View itemView) {
62                 super(itemView);
63                 txtName = itemView.findViewById(R.id.txt_name_info);
64                 txtQuantity = itemView.findViewById(R.id.txt_quantity_info);
65                 txtPrice = itemView.findViewById(R.id.txt_price_info);
66                 txtTotal = itemView.findViewById(R.id.txt_total_info);
67             }
68         }
69     }
```

3.4.8. Chức năng thêm sản phẩm mới

3.4.6.1 Bắt sự kiện thêm

```
// Button thêm sản phẩm
btnAdd = root.findViewById(R.id.btnAdd);
btnAdd.setOnClickListener(v -> {
    Intent intent = new Intent(getActivity(), addFruit.class);
    startActivityForResult(intent, REQUEST_CODE_ADD_PRODUCT);
});
```

3.4.6.2 Kiểm tra thông tin đầu vào

```
// Kiểm tra thông tin đầu vào
1 usage
private boolean validateInputs() {
    if (edtName.getText().toString().trim().isEmpty() ||
        edtPrice.getText().toString().trim().isEmpty() ||
        edtQuantity.getText().toString().trim().isEmpty() ||
        edtDescription.getText().toString().trim().isEmpty()) {

        Toast.makeText(context: this, text: "Vui lòng nhập đầy đủ thông tin!", Toast.LENGTH_SHORT).show();
        return false;
    }
    if (imageUri == null) {
        Toast.makeText(context: this, text: "Vui lòng chọn ảnh!", Toast.LENGTH_SHORT).show();
        return false;
    }
    return true;
}

// Chọn ảnh từ thư viện
1 usage
private void openFileChooser() {
    Intent intent = new Intent();
    intent.setType("image/*");
    intent.setAction(Intent.ACTION_GET_CONTENT);
    startActivityForResult(intent, PICK_IMAGE_REQUEST);
}
```

3.4.6.3 Nhận ảnh và upload ảnh lên Firebase Storage

```
// Nhận ảnh sau khi chọn
@Override
protected void onActivityResult(int requestCode, int resultCode, Intent data) {
    super.onActivityResult(requestCode, resultCode, data);
    if (requestCode == PICK_IMAGE_REQUEST && resultCode == RESULT_OK && data != null && data.getData() != null) {
        imageUri = data.getData();
        try {
            Bitmap bitmap = MediaStore.Images.Media.getBitmap(getContentResolver(), imageUri);
            imgProduct.setImageBitmap(bitmap);
        } catch (IOException e) {
            e.printStackTrace();
        }
    }
}

// Upload ảnh lên Firebase Storage
1 usage
private void uploadImageToFirebase() {
    progressDialog.setMessage("Đang tải ảnh...");
    progressDialog.show();

    String imageName = UUID.randomUUID().toString();
    StorageReference fileReference = storageReference.child(imageName);

    fileReference.putFile(imageUri).addOnSuccessListener(taskSnapshot ->
        fileReference.getDownloadUrl().addOnSuccessListener(uri -> {
            String imageUrl = uri.toString();
            addProductToFirestore(imageUrl);
        })
    ).addOnFailureListener(e -> {
        progressDialog.dismiss();
        Toast.makeText(context: addFruit.this, text: "Lỗi tải ảnh!", Toast.LENGTH_SHORT).show();
    });
}
```

3.4.6.4 Thêm sản phẩm vào Firebase FireStore

```
private void addProductToFirestore(String imageUrl) {
    progressDialog.setMessage("Đang thêm sản phẩm...");
    progressDialog.show();

    HashMap<String, Object> productMap = new HashMap<>();
    productMap.put("img_url", imageUrl);
    productMap.put("name", edtName.getText().toString().trim());
    productMap.put("price", edtPrice.getText().toString().trim());
    productMap.put("quantity", Integer.parseInt(edtQuantity.getText().toString().trim()));
    productMap.put("description", edtDescription.getText().toString().trim());
    productMap.put("unit", spinnerUnit.getSelectedItem().toString());

    firestore.collection( collectionPath: "Fruits").document().set(productMap)
        .addOnCompleteListener(task -> {
            progressDialog.dismiss();
            if (task.isSuccessful()) {
                Toast.makeText( context: addFruit.this, text: "Thêm sản phẩm thành công!", Toast.LENGTH_SHORT).show();

                // Gửi kết quả về Fragment trước đó
                Intent resultIntent = new Intent();
                setResult(Activity.RESULT_OK, resultIntent);

                // Đóng activity
                finish();
            } else {
                Toast.makeText( context: addFruit.this, text: "Lỗi khi thêm sản phẩm!", Toast.LENGTH_SHORT).show();
            }
        });
}
```

3.4.9. Chức năng Cập nhật sản phẩm

3.4.7.1 Bắt sự kiện cập nhật

```
// Bắt sự kiện cập nhật
holder.imgUpdate.setOnClickListener(v -> {
    Intent intent = new Intent(context, updateFruit.class);
    intent.putExtra( name: "name", fruit.getName());
    intent.putExtra( name: "price", fruit.getPrice());
    intent.putExtra( name: "quantity", String.valueOf(fruit.getQuantity()));
    intent.putExtra( name: "description", fruit.getDescription());
    intent.putExtra( name: "img_url", fruit.getImg_url());
    intent.putExtra( name: "unit", fruit.getUnit());
    //
    updateLauncher.launch(intent);

});
```

3.4.7.2 Nhận dữ liệu từ Intent

```
// Nhận dữ liệu từ Intent
productName = getIntent().getStringExtra( name: "name");
String productPrice = getIntent().getStringExtra( name: "price");
String productQuantity = getIntent().getStringExtra( name: "quantity");
String productDescription = getIntent().getStringExtra( name: "description");
String productImage = getIntent().getStringExtra( name: "img_url");
String unit = getIntent().getStringExtra( name: "unit");

if (productName == null) {
    Toast.makeText( context: this,  text: "Lỗi: Không tìm thấy sản phẩm!", Toast.LENGTH_SHORT).show();
    finish();
    return;
}
```

3.4.7.3 Hiển thị dữ liệu lên các EditText, ImageView, Spinner

```
ArrayAdapter<CharSequence> adapter = ArrayAdapter.createFromResource( context: this,
    R.array.unit_array, android.R.layout.simple_spinner_item);
adapter.setDropDownViewResource(android.R.layout.simple_spinner_dropdown_item);
spinnerUnit.setAdapter(adapter);

if (unit != null) {
    int spinnerPosition = adapter.getPosition(unit);
    spinnerUnit.setSelection(spinnerPosition);
}
edtName.setText(productName);
edtPrice.setText(productPrice);
edtQuantity.setText(productQuantity);
edtDescription.setText(productDescription);

if (productImage != null && !productImage.isEmpty()) {
    Glide.with( activity: updateFruit.this).load(productImage).into(imgUpdate);
} else {
    imgUpdate.setImageResource(R.drawable.image);
}
```

3.4.7.4 Cập nhật thông tin :

```
private void updateProduct() {
    String name = edtName.getText().toString().trim();
    String price = edtPrice.getText().toString().trim();
    int quantity = Integer.parseInt(edtQuantity.getText().toString().trim());
    String description = edtDescription.getText().toString().trim();
    String unit = spinnerUnit.getSelectedItem().toString();

    if (name.isEmpty() || price.isEmpty() || edtQuantity.getText().toString().trim().isEmpty() || description.isEmpty()) {
        Toast.makeText(context: this, text: "Vui lòng nhập đầy đủ thông tin!", Toast.LENGTH_SHORT).show();
        return;
    }

    progressDialog = new ProgressDialog(context: this);
    progressDialog.setMessage("Đang cập nhật...");
    progressDialog.show();

    Map<String, Object> updates = new HashMap<>();
    updates.put("name", name);
    updates.put("price", price);
    updates.put("quantity", quantity);
    updates.put("description", description);
    updates.put("unit", unit);

    db.collection(collectionPath: "Fruits").whereEqualTo(field: "name", productName) Query
        .get() Task<QuerySnapshot>
        .addOnSuccessListener(queryDocumentSnapshots -> {
            if (!queryDocumentSnapshots.isEmpty()) {
                String documentId = queryDocumentSnapshots.getDocuments().get(0).getId();
                db.collection(collectionPath: "Fruits").document(documentId) DocumentReference
                    .update(updates) Task<Void>
                    .addOnSuccessListener(aVoid -> {
                        if (imageUri == null) {
                            progressDialog.dismiss();
                            Toast.makeText(context: this, text: "Cập nhật thành công!", Toast.LENGTH_SHORT).show();

                            // Gửi kết quả về Fragment trước đó
                            Intent resultIntent = new Intent();
                            setResult(Activity.RESULT_OK, resultIntent);
                            finish();
                        } else {
                            uploadImage(documentId);
                        }
                    })
                    .addOnFailureListener(e -> {
                        progressDialog.dismiss();
                        Toast.makeText(context: updateFruit.this, text: "Lỗi khi cập nhật dữ liệu!", Toast.LENGTH_SHORT).show();
                    });
            } else {
                progressDialog.dismiss();
                Toast.makeText(context: updateFruit.this, text: "Lỗi: Không tìm thấy sản phẩm trong cơ sở dữ liệu!", Toast.LENGTH_SHORT).show();
            }
        })
        .addOnFailureListener(e -> {
            progressDialog.dismiss();
            Toast.makeText(context: updateFruit.this, text: "Lỗi truy vấn dữ liệu!", Toast.LENGTH_SHORT).show();
        });
}
}
```

3.4.7.5 Tải ảnh mới lên Firebase

```
private void uploadImage(String documentId) {
    storageRef.putFile(imageUri).addOnSuccessListener(taskSnapshot ->
        storageRef.getDownloadUrl().addOnSuccessListener(uri -> {
            db.collection(collectionPath: "Fruits_img").document(documentId).DocumentReference
                .update( field: "img_url", uri.toString()) Task<Void>
                .addOnSuccessListener(aVoid -> {
                    progressDialog.dismiss();
                    Toast.makeText(context: this, text: "Cập nhật thành công!", Toast.LENGTH_SHORT).show();
                    // Gửi kết quả về Fragment trước đó
                    Intent resultIntent = new Intent();
                    setResult(Activity.RESULT_OK, resultIntent);
                    finish();
                });
        })
    ).addOnFailureListener(e -> {
        progressDialog.dismiss();
        Toast.makeText(context: updateFruit.this, text: "Lỗi khi tải ảnh!", Toast.LENGTH_SHORT).show();
    });
}
```

3.4.10. Chức năng xóa sản phẩm

3.4.8.1 Bắt sự kiện xóa sản phẩm

```
// Bắt sự kiện xóa
holder.imgDelete.setOnClickListener(v -> deleteProduct(fruit.getName()));
```

3.4.8.2 Xóa sản phẩm khỏi FireStore và Firebase Storage

```
private void deleteProduct(String productName) {
    new AlertDialog.Builder(context)
        .setTitle("Xóa Sản Phẩm")
        .setMessage("Bạn có chắc chắn muốn xóa sản phẩm này không?")
        .setPositiveButton( text: "Xóa", (dialog, which) -> {
            ProgressDialog progressDialog = new ProgressDialog(context);
            progressDialog.setMessage("Đang xóa sản phẩm...");
            progressDialog.show();

            FirebaseFirestore db = FirebaseFirestore.getInstance();
            FirebaseStorage storage = FirebaseStorage.getInstance();

            // Tìm sản phẩm theo tên
            db.collection( collectionPath: "Fruits" ) CollectionReference
                .whereEqualTo( field: "name", productName ) Query
                .get() Task<QuerySnapshot>
                .addOnSuccessListener(queryDocumentSnapshots -> {
                    if (!queryDocumentSnapshots.isEmpty()) {
                        for (QueryDocumentSnapshot document : queryDocumentSnapshots) {
                            String foundProductId = document.getId();
                            String imageUrl = document.getString( field: "img_url"); // Lấy đường dẫn ảnh từ Firestore

                            // Xóa sản phẩm từ Firestore
                            db.collection( collectionPath: "Fruits").document(foundProductId) DocumentReference
                                .delete() Task<Void>
                                .addOnSuccessListener(aVoid -> {
                                    if (imageUrl != null && !imageUrl.isEmpty()) {
                                        // Tạo reference từ URL để xóa ảnh trong Storage
                                        StorageReference imageRef = storage.getReferenceFromUrl(imageUrl);
                                        imageRef.delete()
                                            .addOnSuccessListener(unused -> {
                                                progressDialog.dismiss();
                                                Toast.makeText(context, text: "Sản phẩm đã bị xóa!", Toast.LENGTH_SHORT).show();
                                                //
                                                removeItemFromList(productName);
                                            })
                                            .addOnFailureListener(e -> {
                                                progressDialog.dismiss();
                                                Toast.makeText(context, text: "Lỗi khi xóa ảnh!", Toast.LENGTH_SHORT).show();
                                            });
                                    } else {
                                        progressDialog.dismiss();
                                        Toast.makeText(context, text: "Sản phẩm đã bị xóa nhưng không có ảnh!", Toast.LENGTH_SHORT).show();
                                        //
                                        removeItemFromList(productName);
                                    }
                                })
                                .addOnFailureListener(e -> {
                                    progressDialog.dismiss();
                                    Toast.makeText(context, text: "Lỗi khi xóa sản phẩm!", Toast.LENGTH_SHORT).show();
                                });
                    } else {
                        progressDialog.dismiss();
                        Toast.makeText(context, text: "Không tìm thấy sản phẩm!", Toast.LENGTH_SHORT).show();
                    }
                })
                .addOnFailureListener(e -> {
                    progressDialog.dismiss();
                    Toast.makeText(context, text: "Lỗi khi tìm sản phẩm!", Toast.LENGTH_SHORT).show();
                });
            }

            .setNegativeButton( text: "Hủy", listener: null )
            .show();
        });
}
```

3.4.8.3 Cập nhật lại danh sách sau xóa

```
private void removeItemFromList(String productName) {
    for (int i = 0; i < listWare.size(); i++) {
        if (listWare.get(i).getName().equals(productName)) {
            listWare.remove(i);
            notifyItemRemoved(i);
            notifyItemRangeChanged(i, listWare.size()); // Cập nhật lại danh sách sau khi xóa
            Toast.makeText(context, text: "Sản phẩm đã bị xóa!", Toast.LENGTH_SHORT).show();
            return;
        }
    }
}
```

3.4.11. Phân quyền người dùng

- Kiểm tra User Role để hiển thị chức năng Kho hàng và đơn hàng

```
| usage
private void checkUserRole(NavigationView navigationView) {
    FirebaseUser user = FirebaseAuth.getInstance().getCurrentUser();
    if (user != null) {
        String userId = user.getUid();
        DatabaseReference databaseReference = FirebaseDatabase.getInstance().getReference(path: "Users").child(userId);

        databaseReference.addValueEventListener(new ValueEventListener() {
            2 usages
            @Override
            public void onDataChange(DataSnapshot snapshot) {
                if (snapshot.exists()) {
                    String role = snapshot.child(path: "role").getValue(String.class);
                    Log.d(tag: "FIREBASE_ROLE", msg: "User Role: " + role); // Debug log để kiểm tra

                    Menu menu = navigationView.getMenu();
                    MenuItem warehouseItem = menu.findItem(R.id.nav_warehouse);
                    MenuItem billItem = menu.findItem(R.id.nav_bill);

                    if ("admin".equals(role)) {
                        warehouseItem.setVisible(true);
                        billItem.setVisible(true);
                    } else {
                        warehouseItem.setVisible(false);
                        billItem.setVisible(false);
                    }
                }
            }
        });
    }
}
```

3.4.12. Đăng Xuất

```
TextView txt_log_out = findViewById(R.id.txt_logout);
txt_log_out.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        FirebaseAuth.getInstance().signOut();
        Intent intent = new Intent(packageContext: MainActivity.this, WelcomeActivity.class);
        intent.setFlags(Intent.FLAG_ACTIVITY_NEW_TASK | Intent.FLAG_ACTIVITY_CLEAR_TASK); // Xóa lịch sử activity
        startActivity(intent);
        finish();
    }
});
```

KẾT LUẬN

1. Kết quả đạt được

- Hoàn thiện một ứng dụng bán hoa quả online cơ bản phục vụ khách hàng trong khu vực.
- Cho phép người dùng đăng ký, đăng nhập, chỉnh sửa thông tin cá nhân.
- Lưu trữ thông tin người dùng vào Firebase Realtime Database.
- Tích hợp Firebase Authentication để xác thực tài khoản.
- Hỗ trợ cập nhật ảnh đại diện thông qua Firebase Storage.

Xây dựng giao diện người dùng trực quan, phân chia rõ ràng thành các Fragment chức năng :

- Trang chủ : hiển thị danh sách các sản phẩm hiện có.
- Giỏ hàng : quản lý các sản phẩm chọn mua.
- Thông tin cá nhân : quản lý thông tin và chỉnh sửa thông tin cá nhân.
- Kho hàng : giao diện quản lý sản phẩm (dành cho người bán)
- Đơn hàng : theo dõi các đơn hàng của người mua (dành cho người bán)

2. Nhược điểm

Mặc dù đã đạt được những kết quả ban đầu, ứng dụng vẫn tồn tại một số hạn chế cần được cải thiện để mang lại trải nghiệm tốt hơn cho người dùng và mở rộng khả năng phát triển:

- Giao diện chưa bắt mắt:
- Màu sắc, bố cục còn đơn giản, chưa thu hút người dùng.

- Thiếu hình ảnh động, hiệu ứng mượt mà khi chuyển Fragment.
- Phù hợp cho cửa hàng bán lẻ nhỏ.
- Cấu trúc dữ liệu, tính năng hiện tại chưa đủ cho hệ thống nhiều chi nhánh.
- Người dùng chưa thể thanh toán bằng ví điện tử hoặc thẻ ngân hàng.
- Chưa hỗ trợ theo dõi đơn hàng sau khi đặt hàng.
- Chưa có chức năng đánh giá sản phẩm.
- Chưa có chức năng lọc và sắp xếp sản phẩm theo giá, loại, ...

3. Hướng phát triển

Để nâng cao chất lượng ứng dụng và đáp ứng tốt hơn nhu cầu của người dùng, các hướng phát triển tiềm năng trong tương lai bao gồm:

- Về tính năng:
 - Thanh toán online: Tích hợp cổng thanh toán như Momo, ZaloPay, hoặc VNPay.
 - Theo dõi đơn hàng: Cho phép người dùng跟踪 theo dõi trạng thái đơn hàng.
 - Đánh giá sản phẩm: Hệ thống rating và review.
 - Lọc sản phẩm: lọc loại trái cây, theo giá, ...
- Về giao diện (UI/UX):
 - Sử dụng Material Design, hiệu ứng mượt mà.
 - Cải thiện trải nghiệm người dùng trên cả điện thoại và máy tính bảng.
 - Tối ưu hóa giao diện cho người lớn tuổi dễ sử dụng.
- Về kỹ thuật:
 - Chuyển đổi kiến trúc: Nghiên cứu và triển khai kiến trúc MVVM kết hợp với Repository pattern để tăng tính module hóa, khả năng bảo trì và kiểm thử của ứng dụng. Cân nhắc sử dụng máy chủ trung gian để quản lý logic nghiệp vụ phức tạp và tương tác với Firebase hiệu quả hơn.
 - Tối ưu hiệu suất ứng dụng, tránh tải dữ liệu thừa.

TÀI LIỆU THAM KHẢO

- [1] David Flanagan, JavaScript: The Definitive Guide, 7th Edition, O'Reilly Media, 2020.
- [2] Adam Freeman, “Pro jQuery”, Apress, 2018.
- [3] Benjamin Jakobus, “Mastering Bootstrap 5”, Packt Publishing, 2018.