

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



## **BÀI TẬP LỚN**

**PHÁT TRIỂN ỨNG DỤNG CHO THIẾT BỊ DI ĐỘNG**  
**ĐỀ TÀI: XÂY DỰNG ỨNG DỤNG QUẢN LÝ CHI TIÊU CÁ NHÂN**  
**TRÊN NỀN TẢNG ANDROID SỬ DỤNG FIREBASE**

**Giáo viên hướng dẫn: ThS. Kiều Tuấn Dũng**

**Sinh viên thực hiện:**

<b>STT</b>	<b>Mã sinh viên</b>	<b>Họ và tên</b>	<b>Lớp</b>
1	2251061900	Lê Vạn Bảo Trọng	64CNTT1
2	2251061695	Chu Công Đức Anh	64CNTT1
3	2251061765	Đào Thành Hà	64CNTT1
4	2251061802	Phùng Thị Thu Huyền	64CNTT1

**Hà Nội, năm 2025**

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



## **BÀI TẬP LỚN**

**PHÁT TRIỂN ỨNG DỤNG CHO THIẾT BỊ DI ĐỘNG**  
**ĐỀ TÀI: XÂY DỰNG ỨNG DỤNG QUẢN LÝ CHI TIÊU CÁ NHÂN**  
**TRÊN NỀN TẢNG ANDROID SỬ DỤNG FIREBASE**

STT	Mã Sinh Viên	Họ và Tên	Ngày Sinh	Điểm	
				Bảng Số	Bảng Chữ
1	2251061900	Lê Vạn Bảo Trọng	26/03/2004		
2	2251061695	Chu Công Đức Anh	21/01/2004		
3	2251061765	Đào Thành Hà	22/01/2004		
4	2251061802	Phùng Thị Thu Huyền	23/11/2004		

CÁN BỘ CHẤM THI

Hà Nội, năm 2025

## LỜI NÓI ĐẦU

Trong bối cảnh công nghệ phát triển mạnh mẽ, các ứng dụng di động ngày càng đóng vai trò quan trọng trong việc hỗ trợ con người quản lý công việc và cuộc sống hiệu quả hơn. Việc quản lý tài chính cá nhân là một nhu cầu thiết yếu, đòi hỏi sự tiện lợi, nhanh chóng và chính xác. Xuất phát từ thực tế đó, nhóm chúng em đã lựa chọn đề tài: **"Xây dựng ứng dụng quản lý chi tiêu cá nhân trên nền tảng Android sử dụng Firebase"** nhằm hỗ trợ người dùng ghi chép, theo dõi và phân tích các khoản thu chi một cách dễ dàng.

Trong quá trình thực hiện đề tài, nhóm đã ứng dụng kiến thức đã học về lập trình Android, cơ sở dữ liệu, và dịch vụ Firebase để xây dựng nên một ứng dụng có giao diện thân thiện, chức năng thiết thực như: quản lý giao dịch, ngân sách, thống kê chi tiêu và nhắc nhở thanh toán.

Mặc dù đã cố gắng hoàn thiện tốt nhất trong khả năng, nhưng do thời gian có hạn và kinh nghiệm còn hạn chế, ứng dụng vẫn có thể tồn tại những thiếu sót. Nhóm rất mong nhận được sự góp ý, đánh giá từ thầy/cô để cải thiện và phát triển sản phẩm hoàn thiện hơn trong tương lai.

Nhóm xin chân thành cảm ơn **quý thầy/cô** đã tận tình hướng dẫn và tạo điều kiện thuận lợi để nhóm thực hiện đề tài này.

# MỤC LỤC

<b>LỜI NÓI ĐẦU .....</b>	<b>3</b>
<b>MỤC LỤC .....</b>	<b>4</b>
<b>Chương 1. TỔNG QUAN VỀ ĐỀ TÀI .....</b>	<b>6</b>
<b>1.1. Giới thiệu về đề tài.....</b>	<b>6</b>
<b>1.2. Mục tiêu của đề tài .....</b>	<b>6</b>
<b>1.3. Phạm vi của đề tài.....</b>	<b>6</b>
<b>1.4. Phân chia nhiệm vụ .....</b>	<b>7</b>
<b>Chương 2. KIẾN TRÚC VÀ CÔNG NGHỆ .....</b>	<b>8</b>
<b>2.1. Kiến trúc hệ thống.....</b>	<b>8</b>
2.1.1. Ứng dụng Client (Android App) .....	8
2.1.2. Firebase Authentication (Xác thực người dùng) .....	8
2.1.3. Firebase Cloud Firestore (Cơ sở dữ liệu) .....	8
2.1.4. Firebase GenKit (AI chat bot) .....	8
<b>2.2. Giới thiệu về Công nghệ phát triển.....</b>	<b>8</b>
2.2.1. Firebase.....	8
2.2.1. Android SDK.....	9
<b>Chương 3. XÂY DỰNG ỨNG DỤNG .....</b>	<b>9</b>
<b>3.1. Thiết kế Figma .....</b>	<b>9</b>
<b>3.2. Thiết kế CSDL .....</b>	<b>13</b>
<b>3.2. Giao diện ứng dụng .....</b>	<b>13</b>
3.2.1. Màn hình đăng kí, đăng nhập, thông tin tài khoản .....	13
3.2.2. Màn hình chính (Tổng quan).....	14
3.2.3. Màn hình giao dịch.....	14
3.2.4. Màn hình ngân sách.....	15

3.2.5. Màn hình thống kê.....	16
3.2.6. Màn hình nhắc nhở .....	16
3.2.7. Màn hình quản lý danh mục .....	17
3.2.8. Màn hình mục tiêu dài hạn .....	18
3.2.9. Màn hình tư vấn tài chính (chat bot) .....	19
3.2.9. Màn hình admin quản lý.....	20
<b>3.3. Code minh họa các chức năng cốt lõi.....</b>	<b>20</b>
3.3.1. Code xử lý lấy các giao dịch .....	20
3.3.2. Code xử lý thêm giao dịch.....	21
3.3.3. Code xử cập nhật giao dịch .....	21
3.3.4. Code xử lý xóa giao dịch.....	22
3.3.5. Code xử lý ngân sách (thêm/sửa/xóa/xem) .....	23
3.3.6. Code xử lý nhắc nhở (thêm/sửa/xóa/xem) .....	25
3.3.7. Code xử lý các thông báo .....	27
<b>KẾT LUẬN .....</b>	<b>27</b>
<b>1. Kết quả đạt được .....</b>	<b>27</b>
<b>2. Nhược điểm .....</b>	<b>28</b>
<b>3. Hướng phát triển .....</b>	<b>28</b>

# Chương 1. TỔNG QUAN VỀ ĐỀ TÀI

## 1.1. Giới thiệu về đề tài

Trong bối cảnh nhu cầu quản lý tài chính cá nhân ngày càng được quan tâm, việc xây dựng một ứng dụng hỗ trợ người dùng theo dõi và kiểm soát chi tiêu trở nên thiết thực và hữu ích. Đề tài này tập trung phát triển một ứng dụng **quản lý chi tiêu cá nhân** với các tính năng như ghi chép giao dịch, quản lý ngân sách, thống kê chi tiêu và nhắc nhở định kỳ. Ứng dụng được xây dựng trên nền tảng di động, sử dụng dịch vụ **Firestore Authentication** để xác thực người dùng và **Firestore** để lưu trữ dữ liệu thời gian thực, đảm bảo khả năng truy cập và đồng bộ hóa dữ liệu nhanh chóng, linh hoạt.

## 1.2. Mục tiêu của đề tài

- **Xây dựng ứng dụng hỗ trợ quản lý tài chính cá nhân** với giao diện thân thiện, dễ sử dụng, phù hợp với nhiều đối tượng người dùng.
- Cho phép người dùng thực hiện các thao tác **thêm, sửa, xóa giao dịch** chi tiêu/thu nhập, phân loại theo danh mục và thời gian.
- **Có mục tiêu lâu dài, AI hỗ trợ tư vấn tài chính**, có khả năng phân tích dữ liệu chi tiêu, lịch sử giao dịch, mục tiêu tài chính để đưa ra lời khuyên cá nhân hóa cho người dùng, giúp họ quản lý tiền bạc thông minh hơn.
- **Thiết lập và theo dõi ngân sách** chi tiêu theo từng danh mục hoặc tổng ngân sách tháng, cảnh báo khi gần đạt hoặc vượt mức.
- Cung cấp các **biểu đồ thống kê trực quan** giúp người dùng dễ dàng nắm bắt tình hình tài chính của mình.
- Hỗ trợ **nhắc nhở các khoản chi định kỳ** và lập lịch tự động cho các giao dịch lặp lại như tiền thuê nhà, hóa đơn...
- Ứng dụng sử dụng **Firestore** để đảm bảo an toàn dữ liệu và khả năng mở rộng trong tương lai.

## 1.3. Phạm vi của đề tài

### • Chức năng chính:

- **Quản lý giao dịch:** Cho phép người dùng ghi chép chi tiết giao dịch (số tiền, ngày, danh mục, ghi chú), phân loại theo nhóm chi tiêu như Ăn uống, Mua sắm, Giải trí, v.v. Hỗ trợ xem lịch sử và lọc theo thời gian/danh mục.
- **Quản lý ngân sách:** Thiết lập ngân sách cho từng danh mục hoặc tổng ngân sách tháng, theo dõi mức chi tiêu thực tế, hiển thị cảnh báo khi sắp vượt giới hạn.
- **Mục tiêu dài hạn:** Cho phép người dùng đặt mục tiêu tiết kiệm như mua nhà, du lịch, quỹ khẩn cấp,... và theo dõi tiến độ đạt mục tiêu theo thời gian.
- **AI Chatbot tư vấn tài chính:** Tích hợp trợ lý ảo sử dụng trí tuệ nhân tạo, có khả năng phân tích dữ liệu chi tiêu, lịch sử giao dịch, mục tiêu tài chính để đưa ra lời khuyên cá nhân hóa cho người dùng, giúp họ quản lý tiền bạc thông minh hơn.
- **Thống kê – Báo cáo:** Hiển thị biểu đồ chi tiêu (tròn, cột), tạo báo cáo tuần/tháng/năm giúp người dùng đánh giá tình hình tài chính.
- **Nhắc nhở – Lập lịch:** Tạo nhắc nhở thanh toán hóa đơn, giao dịch định kỳ, hỗ trợ lập lịch tự động hàng ngày, hàng tuần, hàng tháng.

- **Công nghệ sử dụng:**
  - **Firebase Authentication:** Đăng nhập/xác thực người dùng qua **email/password** hoặc phương thức khác (**Google**).
  - **Firebase Firestore:** Lưu trữ dữ liệu người dùng, giao dịch, ngân sách, nhắc nhở dưới dạng cloud database, hỗ trợ đồng bộ nhanh và bảo mật.
  - **Firebase Genkit:** Hỗ trợ triển khai chatbot dùng model Gemini.
- **Phạm vi triển khai:** Ứng dụng di động trên hệ điều hành Android, hướng đến cá nhân hoặc hộ gia đình quản lý chi tiêu hàng ngày.

**1.4. Phân chia nhiệm vụ**

Thành viên	Nhiệm vụ chính
Lê Vạn Bảo Trọng	- Thiết kế và triển khai cơ sở dữ liệu Firestore - Tích hợp Firebase Authentication, bảo mật - Triển khai chức năng Nhắc nhở - Triển khai AI chat bot - Quản lý của admin (web-server)
Chu Công Đức Anh	- Triển khai chức năng và màn hình Giao dịch - Triển khai chức năng Thông tin tài khoản - Mục tiêu dài hạn
Đào Thành Hà	- Triển khai chức năng và màn hình quản lý Ngân sách - Triển khai màn hình Dashboard - Xử lý chức năng đăng nhập và đăng xuất
Phùng Thị Thu Huyền	- Triển khai chức năng và màn hình Thống kê - Triển khai chức năng xuất file báo cáo - Thiết kế giao diện người dùng (UI/UX)

## Chương 2. KIẾN TRÚC VÀ CÔNG NGHỆ

### 2.1. Kiến trúc hệ thống

Ứng dụng quản lý chi tiêu được xây dựng theo mô hình **client-server** với sự hỗ trợ của dịch vụ **Firebase Backend-as-a-Service (BaaS)**. Kiến trúc tổng thể của hệ thống gồm các thành phần chính sau:

#### 2.1.1. Ứng dụng Client (Android App)

- Là nơi người dùng tương tác trực tiếp để thực hiện các chức năng như ghi giao dịch, theo dõi ngân sách, xem biểu đồ, thiết lập nhắc nhở.
- Giao diện người dùng (UI) được phát triển với ngôn ngữ **Java**, tuân thủ nguyên tắc Material Design, tối ưu trải nghiệm trên thiết bị di động.
- Ứng dụng sử dụng **Firebase SDK** để kết nối với các dịch vụ backend (Authentication, Firestore...).

#### 2.1.2. Firebase Authentication (Xác thực người dùng)

- Cung cấp cơ chế xác thực an toàn, hỗ trợ đăng ký/đăng nhập bằng email, Google.
- Đảm bảo rằng mỗi người dùng có không gian lưu trữ dữ liệu riêng biệt, bảo mật.

#### 2.1.3. Firebase Cloud Firestore (Cơ sở dữ liệu)

- Là cơ sở dữ liệu **Realtime NoSQL**, lưu trữ thông tin người dùng, giao dịch, ngân sách, nhắc nhở dưới dạng **collection – document**.
- Hỗ trợ **subcollection**, cho phép tổ chức dữ liệu linh hoạt, phù hợp với việc lưu trữ các giao dịch hoặc ngân sách riêng biệt cho từng người dùng.
- Cung cấp khả năng **đồng bộ hóa thời gian thực (Realtime Sync)**, hỗ trợ trải nghiệm liền mạch khi sử dụng đa thiết bị.

#### 2.1.4. Firebase GenKit (AI chat bot)

- Hỗ trợ triển khai chat bot tư vấn tài chính sử dụng Gemini dựa trên dữ liệu tài chính của người dùng.

### 2.2. Giới thiệu về Công nghệ phát triển

#### 2.2.1. Firebase

- **Firebase Authentication:** Cung cấp hệ thống xác thực người dùng mạnh mẽ, hỗ trợ đa nền tảng như Android, iOS, Web. Firebase Authentication dễ dàng tích hợp với Android SDK, hỗ trợ đăng nhập bằng Email/Password, Google, Facebook, v.v.
- **Firebase Firestore:** Là cơ sở dữ liệu NoSQL thời gian thực, hỗ trợ lưu trữ dữ liệu dưới dạng collection/document. Firestore dễ dàng mở rộng, hỗ trợ truy vấn linh hoạt và đảm bảo an toàn nhờ vào hệ thống **bảo mật Security Rules**. Ngoài ra, Firestore còn hỗ trợ đồng bộ hóa dữ liệu theo thời gian thực giữa các thiết bị.



- **GenKit:** Genkit tích hợp mượt mà với Firebase, hỗ trợ triển khai chatbot, hệ thống trả lời tự động, hoặc các tính năng thông minh sử dụng mô hình ngôn ngữ lớn (LLM) như Gemini, hoặc các mô hình mã nguồn mở như Ollama. Genkit còn cung cấp công cụ CLI, hệ thống quan sát (observability), và hỗ trợ pipeline để kiểm soát và theo dõi toàn bộ luồng xử lý dữ liệu AI.

### 2.2.1. Android SDK

- **Ngôn ngữ lập trình:** Java.
- **Bộ công cụ phát triển chính: Android Studio** – môi trường phát triển chính thức do Google cung cấp, hỗ trợ lập trình giao diện (UI), quản lý vòng đời ứng dụng (Lifecycle), và tích hợp trực tiếp với Firebase qua Plugin.
- **Thư viện hỗ trợ:**
  - **Material Design Components:** Thư viện giao diện tiêu chuẩn, giúp ứng dụng có giao diện hiện đại, thân thiện với người dùng.
  - **MPAndroidChart:** Hỗ trợ vẽ các loại biểu đồ (tròn, cột, đường), phục vụ thống kê và báo cáo chỉ tiêu.
  - **CircleImageView:** Hiển thị hình ảnh đại diện người dùng dưới dạng hình tròn, tăng tính thẩm mỹ cho giao diện.
  - **Glide:** Thư viện tải ảnh mạnh mẽ, hỗ trợ caching, load ảnh nhanh, tối ưu hiệu suất.
  - **Facebook Shimmer:** Tạo hiệu ứng loading mượt mà, nâng cao trải nghiệm người dùng khi chờ dữ liệu hiển thị.

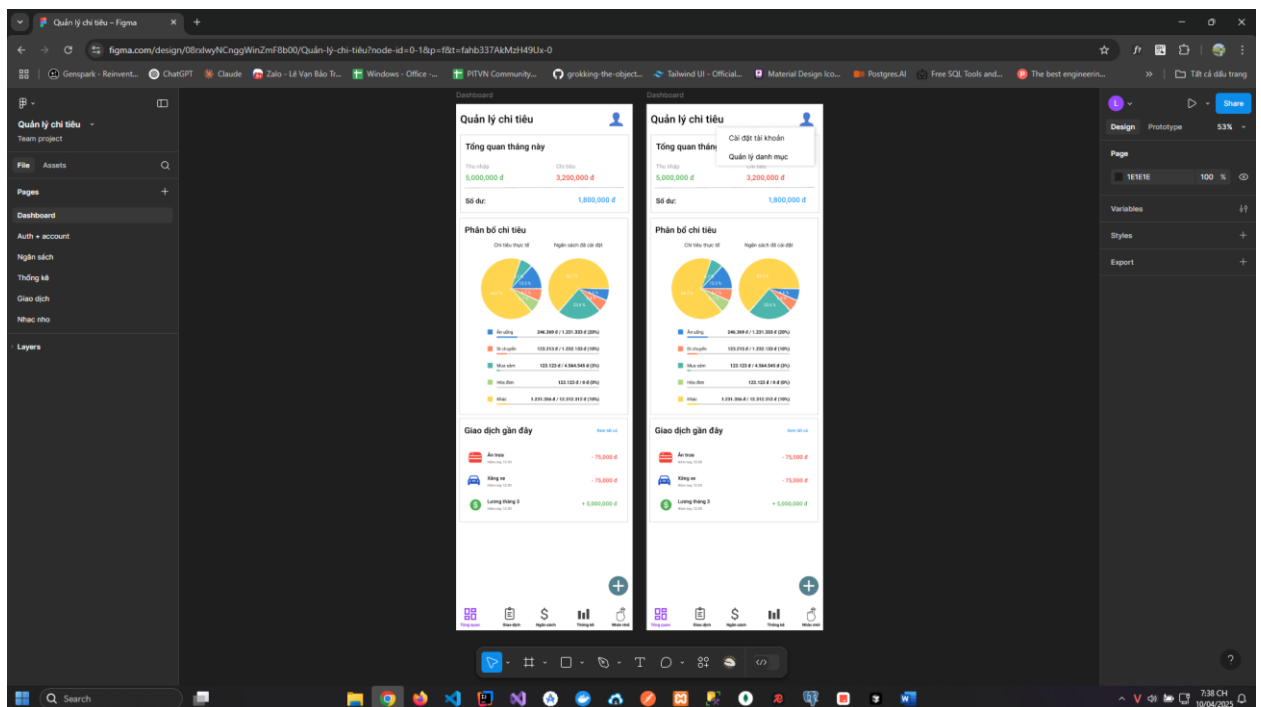
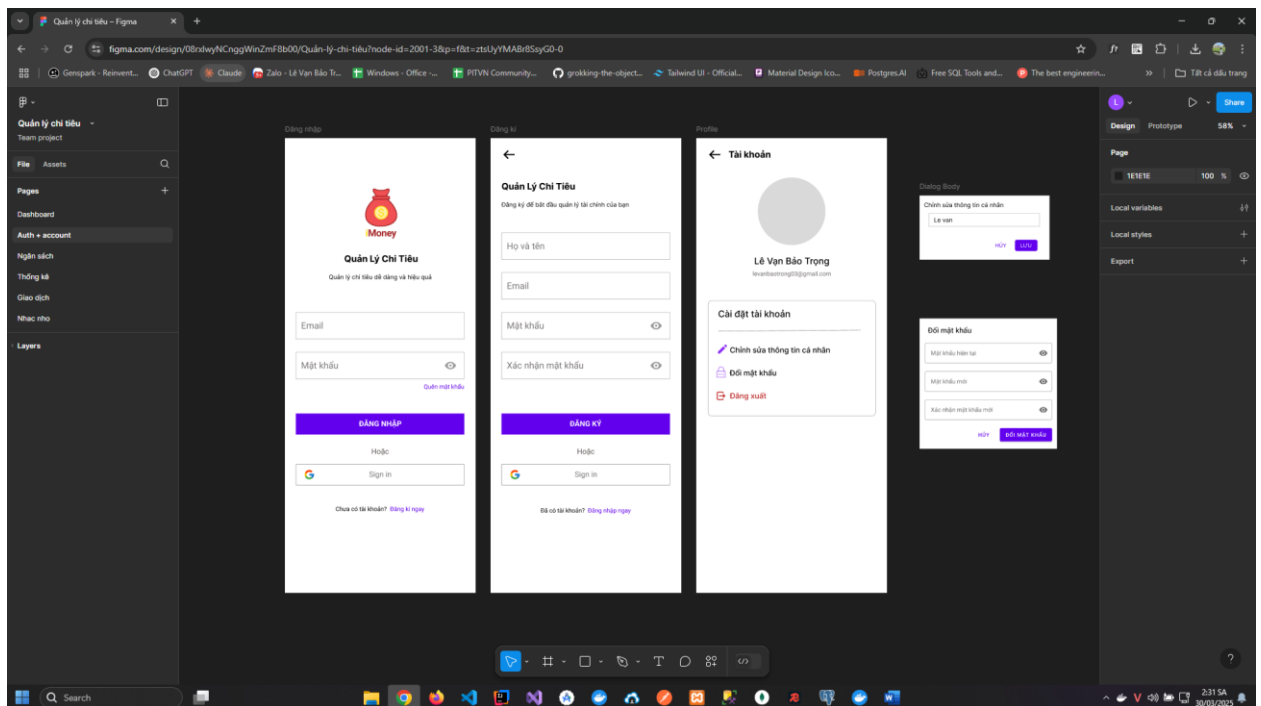
## Chương 3. XÂY DỰNG ỨNG DỤNG

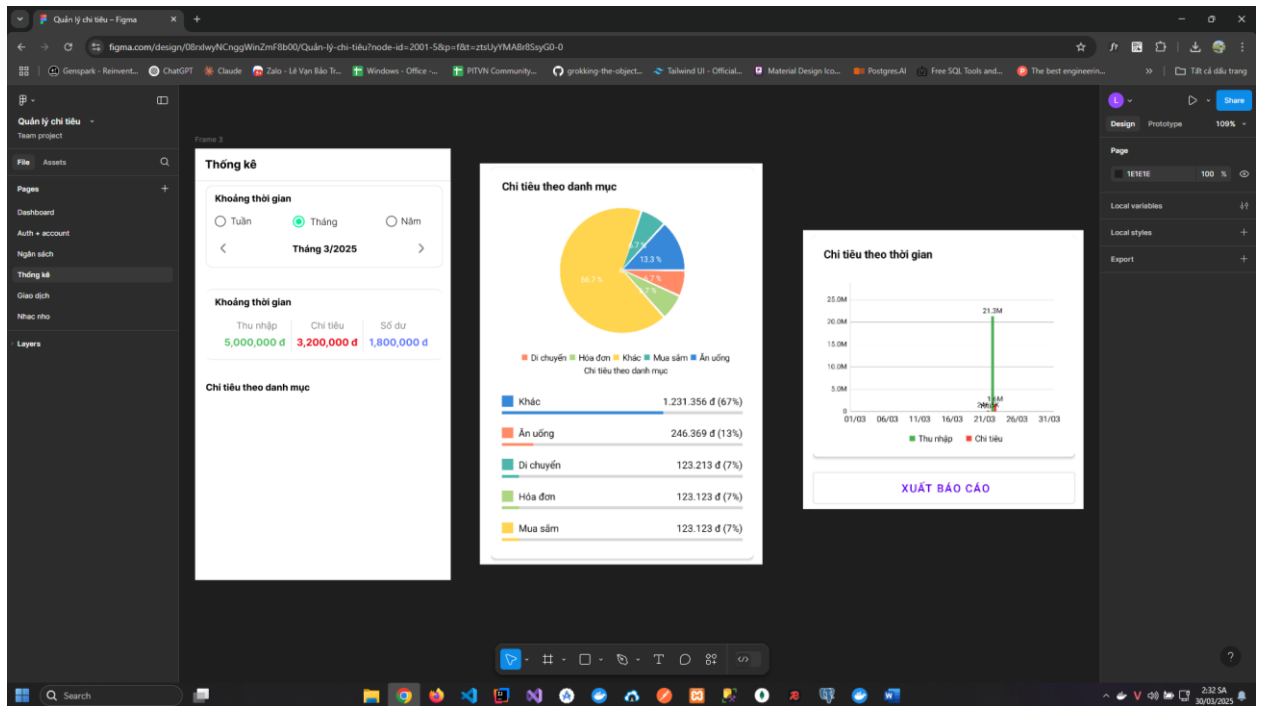
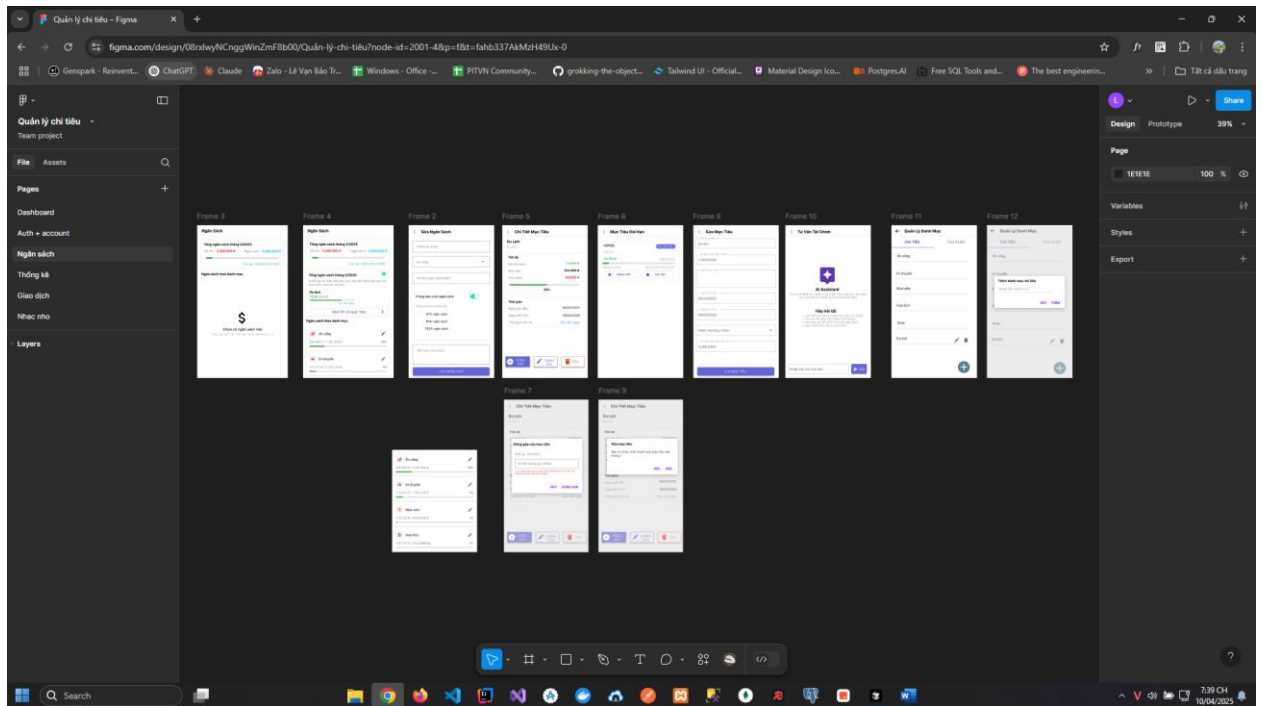
### 3.1. Thiết kế Figma

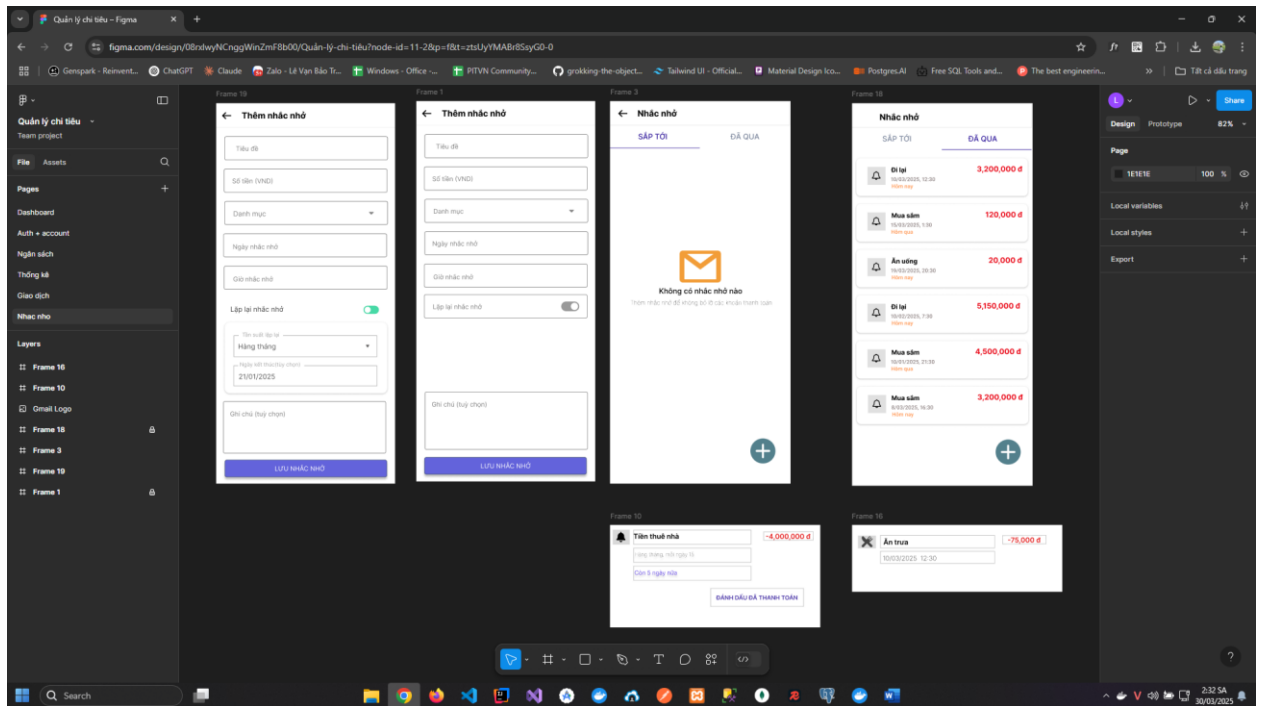
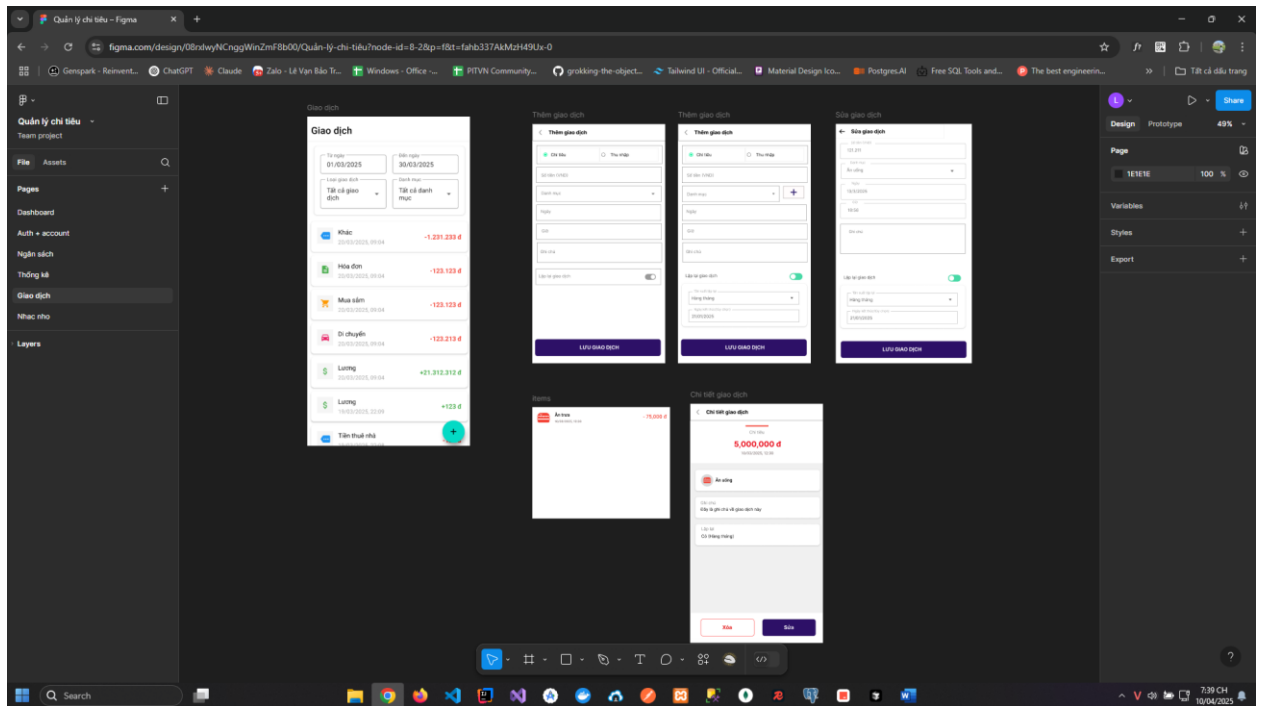
#### Link Figma:

<https://www.figma.com/design/08rxlwYNcnggWinZmF8b00/Qu%E1%BA%A3n-l%C3%BD-chi-ti%C3%AAu?node-id=0-1&p=f>

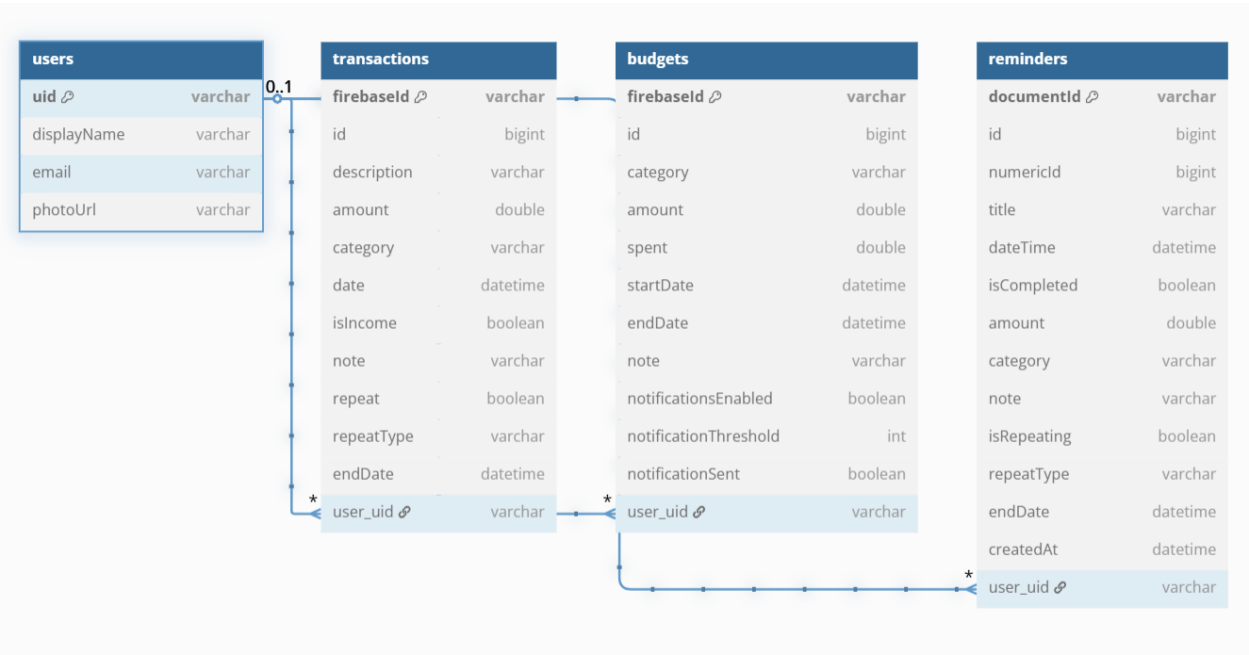
**Link Github:** <https://github.com/chuanghiduoc/quanlychitieu>








3.2. Thiết kế CSDL



3.2. Giao diện ứng dụng

3.2.1. Màn hình đăng kí, đăng nhập, thông tin tài khoản




**Quản Lý Chi Tiêu**

Quản lý chi tiêu dễ dàng và hiệu quả

[Quên mật khẩu?](#)

ĐĂNG NHẬP

HOẶC

 Sign in

Chưa có tài khoản? [Đăng ký ngay](#)


←

**Tạo tài khoản**

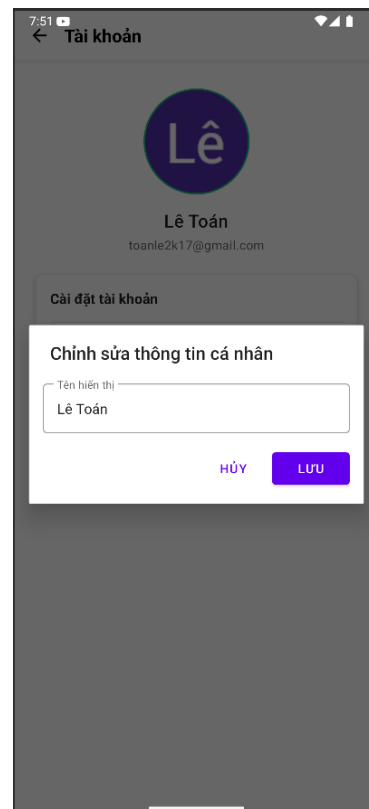
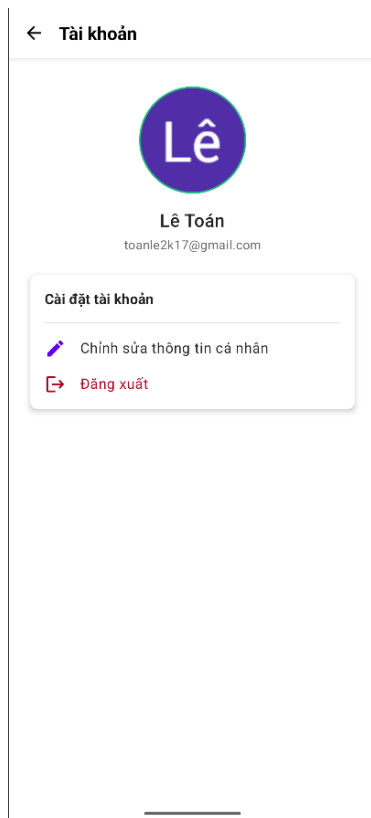
Đăng ký để bắt đầu quản lý tài chính của bạn

ĐĂNG KÝ

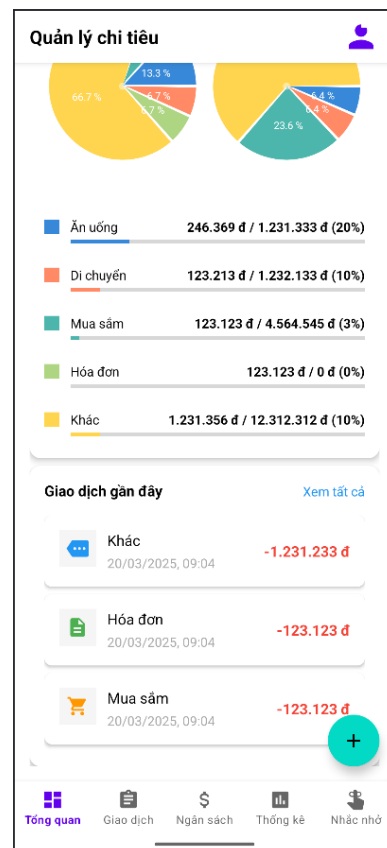
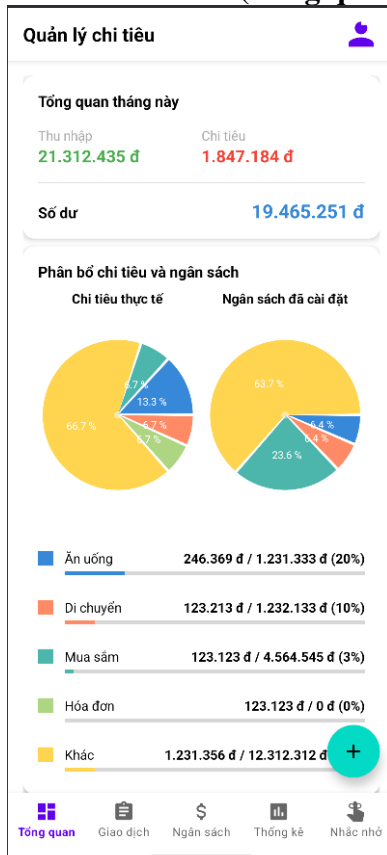
HOẶC

 Sign in

Đã có tài khoản? [Đăng nhập ngay](#)



### 3.2.2. Màn hình chính (Tổng quan)



### 3.2.3. Màn hình giao dịch

Từ ngày

01/03/2025

Đến ngày

30/03/2025

Loại giao dịch

Tất cả giao dịch

Danh mục

Tất cả danh mục

Khác

20/03/2025, 09:04

-1.231.233 đ

Hóa đơn

20/03/2025, 09:04

-123.123 đ

Mua sắm

20/03/2025, 09:04

-123.123 đ

Di chuyển

20/03/2025, 09:04

-123.213 đ

Lương

20/03/2025, 09:04

+21.312.312 đ

Lương

19/03/2025, 22:09

+123 đ

Tiền thuê nhà

19/03/2025, 22:08

-123 đ

Tổng quan

Giao dịch

Ngân sách

Thống kê

Nhắc nhở

Thêm giao dịch

Chi tiêu

Thu nhập

Số tiền (VND)

Danh mục

Ngày

Giờ

Ghi chú

Lặp lại giao dịch

LƯU GIAO DỊCH

Tổng quan

Giao dịch

Ngân sách

Thống kê

Nhắc nhở

Chi tiết giao dịch

Chi tiêu

1.231.233 đ

20/03/2025, 09:04

Khác

XÓA

SỬA

Tổng quan

Giao dịch

Ngân sách

Thống kê

Nhắc nhở

Sửa giao dịch

Chi tiêu

Thu nhập

Số tiền (VND)

100.000.000

Danh mục

Thưởng

Ngày

06/04/2025

Giờ

19:09

Ghi chú

Lặp lại giao dịch

LƯU GIAO DỊCH

Tổng quan

Giao dịch

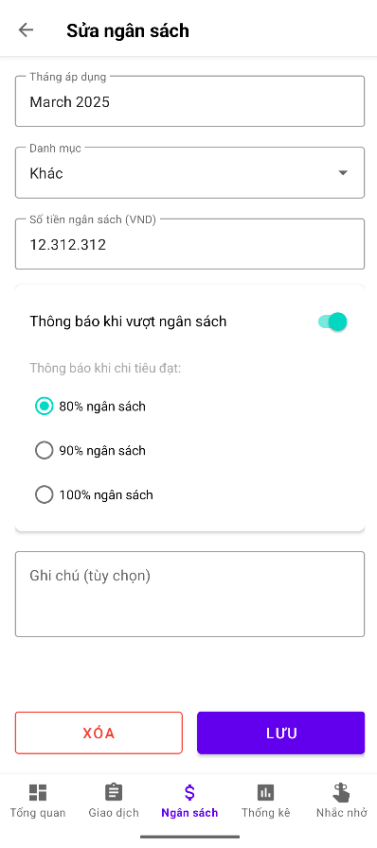
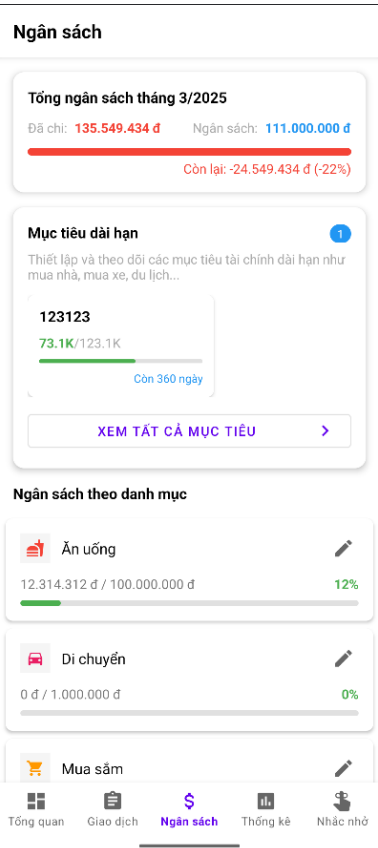
Ngân sách

Thống kê

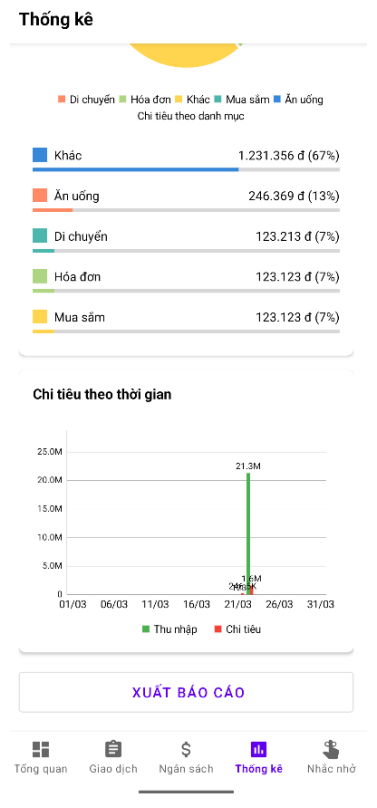
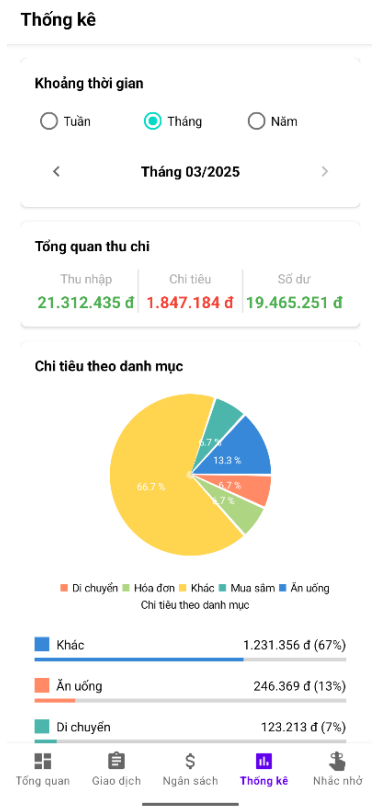
Nhắc nhở

3.2.4. Màn hình ngân sách

Phát triển ứng dụng cho thiết bị di động



3.2.5. Màn hình thống kê



3.2.6. Màn hình nhắc nhở



Nhắc nhở

SẮP TỚI

ĐÃ QUA

123

12/04/2025 lúc 19:45

Còn 1 ngày nữa

ĐÁNH DẤU ĐÃ THANH TOÁN

Tổng quan

Giao dịch

Ngân sách

Thống kê

Nhắc nhở

Nhắc nhở

SẮP TỚI

ĐÃ QUA

123123

Hàng tháng, ngày 31 lúc 02:57

Quá hạn 10 ngày

ĐÁNH DẤU ĐÃ THANH TOÁN

123

30/03/2025 lúc 02:57

Quá hạn 11 ngày

ĐÁNH DẤU ĐÃ THANH TOÁN

123

19/03/2025 lúc 22:08

Đã thanh toán

Tổng quan

Giao dịch

Ngân sách

Thống kê

Nhắc nhở

Thêm nhắc nhở

Tiêu đề

Số tiền (VND)

Danh mục

Ngày nhắc nhở

Giờ nhắc nhở

Lặp lại nhắc nhở

Ghi chú (tùy chọn)

LƯU NHẮC NHỞ

Tổng quan

Giao dịch

Ngân sách

Thống kê

Nhắc nhở

Chỉnh sửa nhắc nhở

Tiêu đề

Số tiền (VND)

Danh mục

Ngày nhắc nhở

Giờ nhắc nhở

Lặp lại nhắc nhở

Ghi chú (tùy chọn)

LƯU NHẮC NHỞ

Tổng quan

Giao dịch

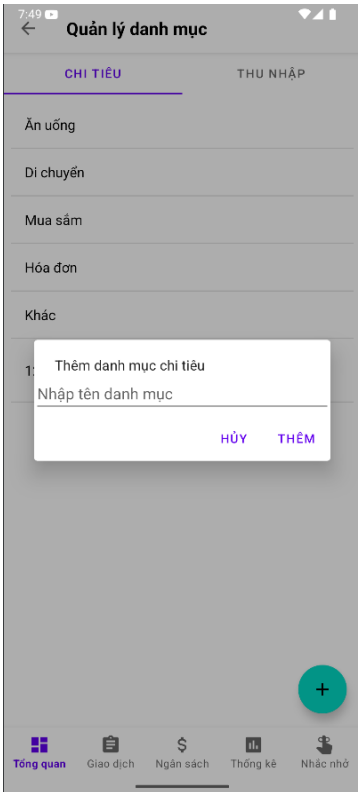
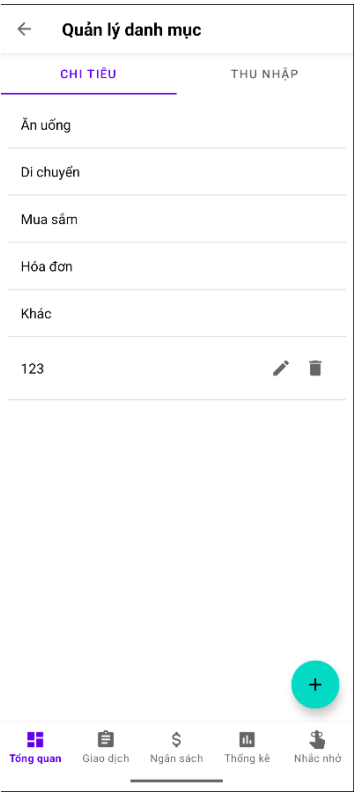
Ngân sách

Thống kê

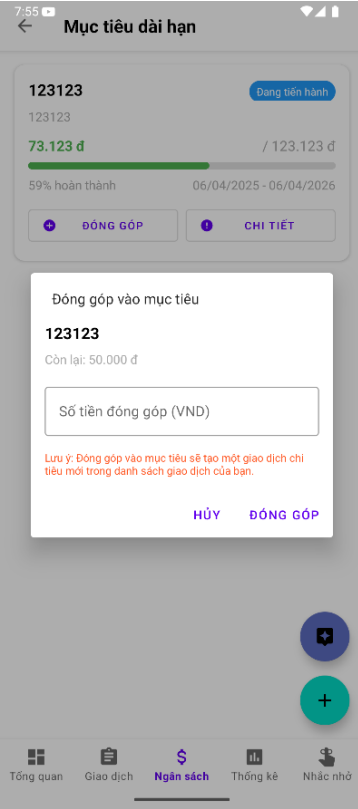
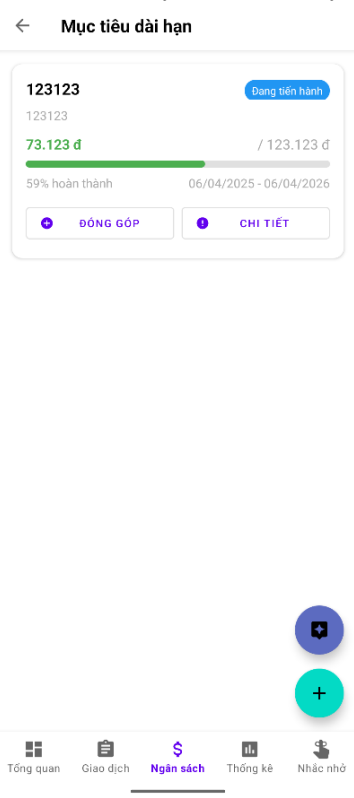
Nhắc nhở

3.2.7. Màn hình quản lý danh mục

Phát triển ứng dụng cho thiết bị di động



3.2.8. Màn hình mục tiêu dài hạn



←

Chi tiết mục tiêu

123123

Đang tiến hành

123123

Tiến độ

Đã tiết kiệm:73.123 đ

Mục tiêu:123.123 đ

Còn thiếu:50.000 đ

59%

Thời gian

Ngày bắt đầu:06/04/2025

Ngày kết thúc:06/04/2026

Thời gian còn lại:Còn 360 ngày

+

ĐÓNG GÓP

CHỈNH SỬA

XÓA

Tổng quan

Giao dịch

\$

Ngân sách

📊

Thống kê

👤

Nhắc nhở

←

Sửa mục tiêu

Tên mục tiêu

123123

Số tiền mục tiêu (VND)

123.123

Mô tả mục tiêu

123123

Ngày bắt đầu

06/04/2025

Ngày kết thúc

06/04/2026

Danh mục (tùy chọn)

Số tiền ban đầu (nếu có)

73.123

LƯU MỤC TIÊU

Tổng quan

Giao dịch

\$

Ngân sách

📊

Thống kê

👤

Nhắc nhở

### 3.2.9. Màn hình tư vấn tài chính (chat bot)

←

Tư vấn tài chính

AI Assistant

Tôi có thể giúp bạn quản lý tài chính hiệu quả hơn, đạt được các mục tiêu tài chính và tối ưu hóa tiết kiệm.

Hãy hỏi tôi:

• Làm thế nào để đạt được mục tiêu tài chính?

• Tôi nên tiết kiệm bao nhiêu mỗi tháng?

• Làm thế nào để quản lý chi tiêu hiệu quả?

• Các chiến lược đầu tư phù hợp?

Nhập câu hỏi của bạn

➤

GỬI

Tổng quan

Giao dịch

\$

Ngân sách

📊

Thống kê

👤

Nhắc nhở

←

Tư vấn tài chính

tu van giúp toi

Tuyệt vời! Dưới đây là phân tích chi tiết và lời khuyên dựa trên dữ liệu tài chính bạn cung cấp.

1. Phân tích tình hình tài chính hiện tại

• **Tổng quan:** Bạn đang có một tình hình tài chính rất tốt với thu nhập ròng (doanh thu trừ chi phí) là 358,709,053. Điều này cho thấy bạn đang kiếm được nhiều tiền hơn số tiền bạn chi tiêu.

• **Thu nhập:** Tổng thu nhập 221,312,435 là một con số đáng kể. Cần tìm hiểu thêm về nguồn gốc thu nhập để có thể đưa ra lời khuyên về việc tối ưu hóa.

• **Chi tiêu:** Tổng chi tiêu 137,396,618, mặc dù nhỏ hơn thu nhập, vẫn là một con số lớn. Việc phân tích chi tiết hơn về các danh mục chi tiêu là rất quan trọng.

• **Tiết kiệm:** Khoản tiết kiệm -110,876 là quá nhỏ so với thu nhập của bạn. Đây là một điểm cần cải thiện lớn.

• **Mục tiêu tài chính:** Bạn đã đạt được 59.4% mục tiêu tài chính "123123".

2. Những điểm cần lưu ý hoặc cải thiện

• **Chi tiêu danh mục "123":** Khoản chi tiêu khổng lồ -123,124,246 ở danh mục "123" là một dấu hiệu lớn. Cần làm rõ danh mục này là gì và liệu có thể cắt giảm được hay không. Đây có thể là một khoản đầu tư lớn.

Nhập câu hỏi của bạn

➤

GỬI

Tổng quan

Giao dịch

\$

Ngân sách

📊

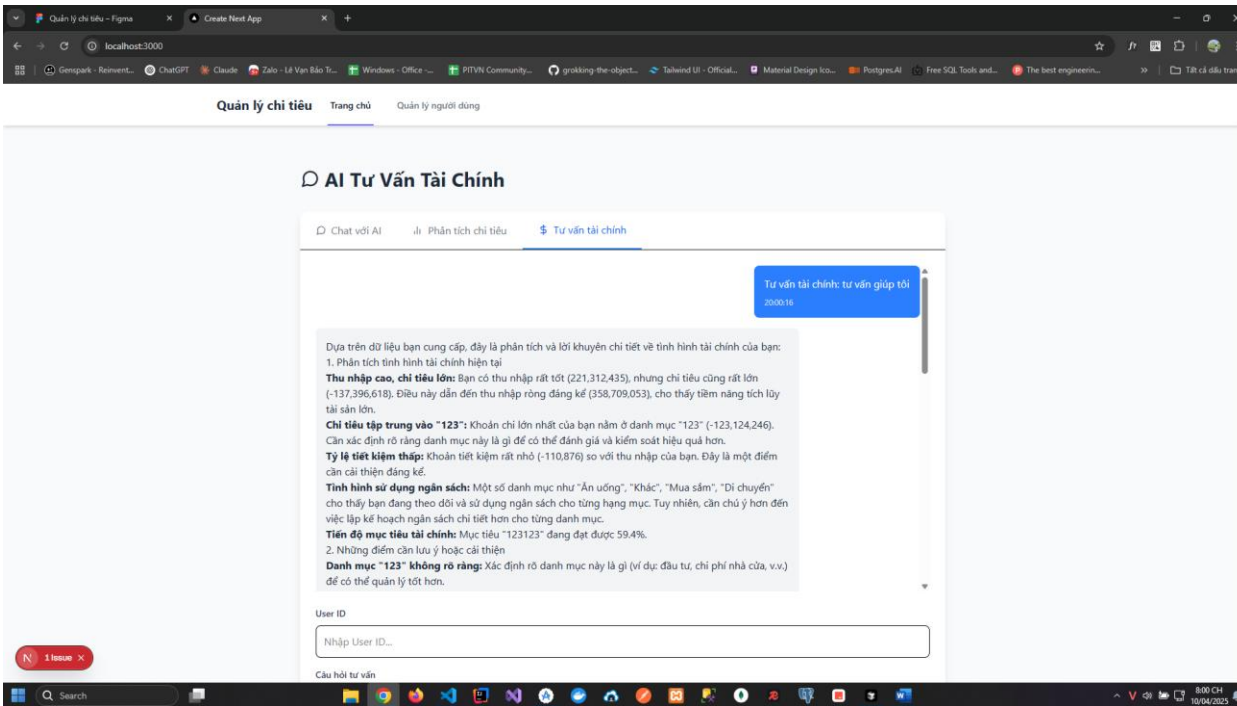
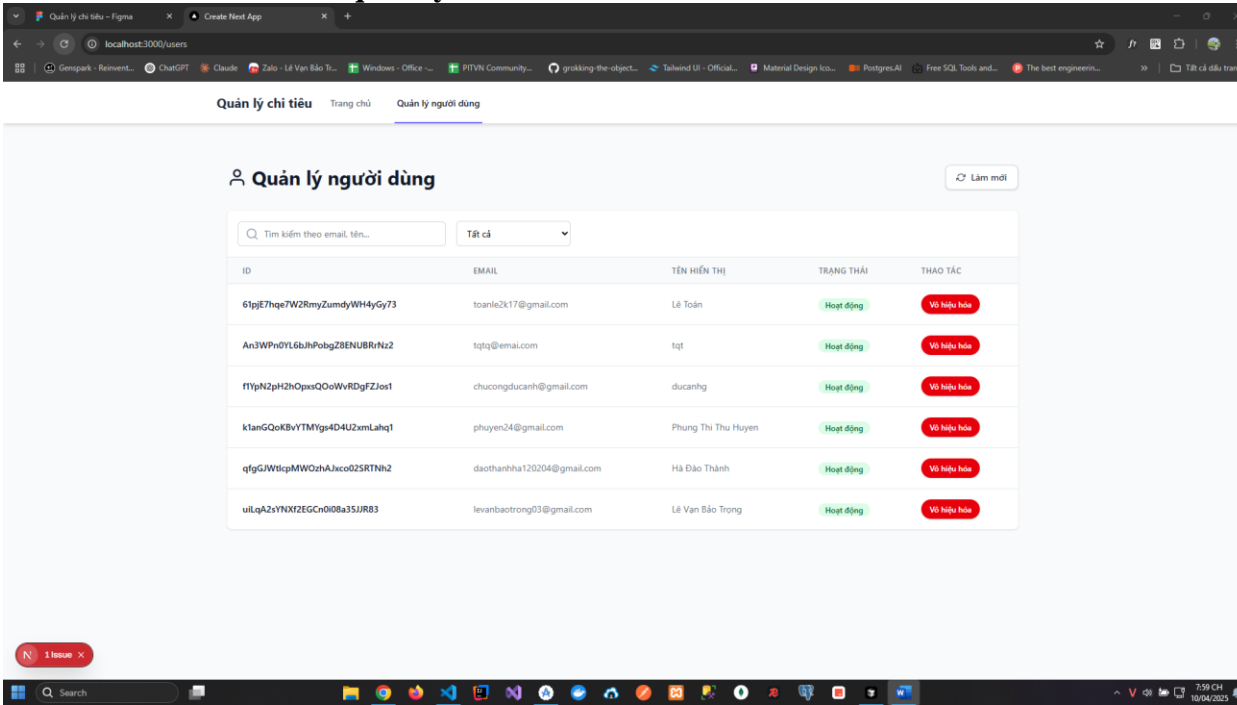
Thống kê

👤

Nhắc nhở

Phát triển ứng dụng cho thiết bị di động

### 3.2.9. Màn hình admin quản lý



### 3.3. Code minh họa các chức năng cốt lõi

#### 3.3.1. Code xử lý lấy các giao dịch

```
private void loadTransactions() {
    FirebaseAuth currentUser = auth.getCurrentUser();
    if (currentUser == null) {
        transactionsLiveData.setValue(new ArrayList<>());
        return;
    }

    db.collection(COLLECTION_USERS)
        .document(currentUser.getUid())
```

```

        .collection(COLLECTION_TRANSACTIONS)
        .orderBy("date", Query.Direction.DESENDING)
        .get()
        .addOnSuccessListener(queryDocumentSnapshots -> {
            List<Transaction> transactions = new ArrayList<>();
            for (QueryDocumentSnapshot document : queryDocumentSnapshots) {
                Transaction transaction = documentToTransaction(document);
                transactions.add(transaction);
            }
            transactionsLiveData.setValue(transactions);
        })
        .addOnFailureListener(e -> {
            // Handle failure
            transactionsLiveData.setValue(new ArrayList<>());
        });
    }
}

```

### 3.3.2. Code xử lý thêm giao dịch

```

public void addTransaction(Transaction transaction) {
    FirebaseUser currentUser = auth.getCurrentUser();
    if (currentUser == null) {
        return;
    }
    double amount = transaction.getAmount();
    if (!transaction.isIncome() && amount > 0) {
        // Nếu là chi tiêu nhưng số tiền là dương, chuyển thành số âm
        transaction.setAmount(-amount);
    } else if (transaction.isIncome() && amount < 0) {
        // Nếu là thu nhập nhưng số tiền là âm, chuyển thành số dương
        transaction.setAmount(Math.abs(amount));
    }

    // Convert transaction to Map
    Map<String, Object> transactionMap = transactionToMap(transaction);

    // Add to Firestore
    db.collection(COLLECTION_USERS)
        .document(currentUser.getId())
        .collection(COLLECTION_TRANSACTIONS)
        .add(transactionMap)
        .addOnSuccessListener(documentReference -> {
            transaction.setFirebaseId(documentReference.getId());
            loadTransactions(); // ✂ Cập nhật danh sách sau khi thêm

            // Cập nhật giao dịch tháng hiện tại nếu giao dịch thuộc tháng hiện
            tại
            if (isCurrentMonth(transaction.getDate())) {
                loadCurrentMonthTransactions();
            }
            // Kiểm tra ngân sách nếu là chi tiêu
            if (!transaction.isIncome()) {
                checkBudgetThresholdAfterTransaction(transaction.getCategory());
            }
        });
}

```

### 3.3.3. Code xử cập nhật giao dịch

```

public void updateTransaction(Transaction transaction) {
    FirebaseUser currentUser = auth.getCurrentUser();
    if (currentUser == null) return;
}

```

```

        double amount = transaction.getAmount();
        transaction.setAmount(transaction.isIncome() ? Math.abs(amount) : -
Math.abs(amount));

        Map<String, Object> transactionMap = transactionToMap(transaction);

        db.collection(COLLECTION_USERS)
            .document(currentUser.getUid())
            .collection(COLLECTION_TRANSACTIONS)
            .document(transaction.getFirebaseId())
            .set(transactionMap)
            .addOnSuccessListener(aVoid -> {
                loadTransactions();

                // Cập nhật giao dịch tháng hiện tại nếu giao dịch thuộc tháng hiện
                tại
                if (isCurrentMonth(transaction.getDate())) {
                    loadCurrentMonthTransactions();
                }
                if (!transaction.isIncome()) {
                    checkBudgetThresholdAfterTransaction(transaction.getCategory());
                }
            });
    }
}

```

### 3.3.4. Code xử lý xóa giao dịch

```

public Task<Void> deleteTransaction(String transactionId) {
    FirebaseUser currentUser = auth.getCurrentUser();
    if (currentUser == null) return Tasks.forException(new Exception("User not logged
in"));

    // Trước khi xóa, kiểm tra xem giao dịch có thuộc tháng hiện tại không
    boolean isCurrentMonthTransaction = false;
    Transaction transactionToDelete = null;
    List<Transaction> currentTransactions = transactionsLiveData.getValue();
    if (currentTransactions != null) {
        for (Transaction t : currentTransactions) {
            if (t.getFirebaseId().equals(transactionId)) {
                transactionToDelete = t;
                break;
            }
        }
    }
    final Transaction finalTransactionToDelete = transactionToDelete;

    final boolean needsCurrentMonthUpdate = isCurrentMonthTransaction;

    return db.collection(COLLECTION_USERS)
        .document(currentUser.getUid())
        .collection(COLLECTION_TRANSACTIONS)
        .document(transactionId)
        .delete()
        .addOnSuccessListener(aVoid -> {
            // Update the local cache
            List<Transaction> currentList = transactionsLiveData.getValue();
            if (currentList != null) {
                List<Transaction> updatedList = new ArrayList<>(currentList);
                updatedList.removeIf(t ->
t.getFirebaseId().equals(transactionId));
                transactionsLiveData.setValue(updatedList);
            }
        })
    }
}

```

```

        // Cập nhật giao dịch tháng hiện tại nếu cần
        if (needsCurrentMonthUpdate) {
            loadCurrentMonthTransactions();
            if (finalTransactionToDelete != null &&
!finalTransactionToDelete.isIncome()) {
                checkBudgetThresholdAfterTransaction(finalTransactionToDelete.getCategory());
            }
        }
    });
}

```

### 3.3.5. Code xử lý ngân sách (thêm/sửa/xóa/xem)

```

private void loadActiveBudgets() {
    FirebaseUser currentUser = auth.getCurrentUser();
    if (currentUser == null) {
        activeBudgetsLiveData.setValue(new ArrayList<>());
        totalBudgetLiveData.setValue(0.0);
        return;
    }

    // Lấy tháng hiện tại
    Calendar calendar = Calendar.getInstance();
    calendar.set(Calendar.DAY_OF_MONTH, 1);
    calendar.set(Calendar.HOUR_OF_DAY, 0);
    calendar.set(Calendar.MINUTE, 0);
    calendar.set(Calendar.SECOND, 0);
    calendar.set(Calendar.MILLISECOND, 0);
    Date startOfMonth = calendar.getTime();

    calendar.add(Calendar.MONTH, 1);
    calendar.add(Calendar.MILLISECOND, -1);
    Date endOfMonth = calendar.getTime();

    db.collection(COLLECTION_USERS)
        .document(currentUser.getId())
        .collection(COLLECTION_BUDGETS)
        .whereGreaterThanOrEqualTo("startDate", startOfMonth)
        .whereLessThanOrEqualTo("endDate", endOfMonth)
        .orderBy("startDate", Query.Direction.DESENDING)
        .addSnapshotListener((value, error) -> {
            if (error != null) {
                Log.e("BudgetRepo", "Error loading active budgets", error);
                return;
            }

            if (value != null) {
                List<Budget> budgets = new ArrayList<>();
                double totalBudgetAmount = 0.0;

                for (QueryDocumentSnapshot document : value) {
                    Budget budget = documentToBudget(document);
                    budgets.add(budget);
                    totalBudgetAmount += budget.getAmount();
                }

                // Cập nhật danh sách ngân sách
                activeBudgetsLiveData.setValue(budgets);
                totalBudgetLiveData.setValue(totalBudgetAmount);

                // Cập nhật chỉ tiêu cho mỗi ngân sách
            }
        });
}

```

```

        updateBudgetsWithSpentAmounts();
    }
});
}

public LiveData<Budget> getBudgetById(String budgetId) {
    MutableLiveData<Budget> budgetLiveData = new MutableLiveData<>();

    FirebaseUser currentUser = auth.getCurrentUser();
    if (currentUser == null) return budgetLiveData;

    db.collection(COLLECTION_USERS)
        .document(currentUser.getId())
        .collection(COLLECTION_BUDGETS)
        .document(budgetId)
        .get()
        .addOnSuccessListener(documentSnapshot -> {
            if (documentSnapshot.exists()) {
                Budget budget = documentSnapshotToBudget(documentSnapshot);

                // Cập nhật số tiền đã chi tiêu từ dữ liệu giao dịch
                Map<String, Double> categorySpentAmounts =
categorySpentAmountsLiveData.getValue();
                if (categorySpentAmounts != null) {
                    Double spentAmount =
categorySpentAmounts.getOrDefault(budget.getCategory(), 0.0);
                    budget.setSpent(spentAmount);
                }

                budgetLiveData.setValue(budget);
            }
        });

    return budgetLiveData;
}

public void addBudget(Budget budget) {
    FirebaseUser currentUser = auth.getCurrentUser();
    if (currentUser == null) return;

    // Đặt userId cho ngân sách
    budget.setUserId(currentUser.getId());

    // Cập nhật số tiền đã chi tiêu từ dữ liệu giao dịch
    Map<String, Double> categorySpentAmounts =
categorySpentAmountsLiveData.getValue();
    if (categorySpentAmounts != null) {
        Double spentAmount = categorySpentAmounts.getOrDefault(budget.getCategory(),
0.0);
        budget.setSpent(spentAmount);
    }

    // Chuyển đổi thành Map
    Map<String, Object> budgetMap = budgetToMap(budget);

    // Thêm vào Firestore
    db.collection(COLLECTION_USERS)
        .document(currentUser.getId())
        .collection(COLLECTION_BUDGETS)
        .add(budgetMap)
        .addOnSuccessListener(documentReference -> {
            budget.setFirebaseId(documentReference.getId());
        });
}

```



```

        loadActiveBudgets();
    });
}

public void updateBudget(Budget budget) {
    FirebaseUser currentUser = auth.getCurrentUser();
    if (currentUser == null) return;

    // Đảm bảo có userId
    if (budget.getUserId() == null) {
        budget.setUserId(currentUser.getId());
    }

    // Cập nhật số tiền đã chi tiêu từ dữ liệu giao dịch
    Map<String, Double> categorySpentAmounts =
categorySpentAmountsLiveData.getValue();
    if (categorySpentAmounts != null) {
        Double spentAmount = categorySpentAmounts.getOrDefault(budget.getCategory(),
0.0);
        budget.setSpent(spentAmount);
    }

    Map<String, Object> budgetMap = budgetToMap(budget);

    db.collection(COLLECTION_USERS)
        .document(currentUser.getId())
        .collection(COLLECTION_BUDGETS)
        .document(budget.getFirebaseId())
        .set(budgetMap)
        .addOnSuccessListener(aVoid -> loadActiveBudgets());
}

public void deleteBudget(String budgetId) {
    FirebaseUser currentUser = auth.getCurrentUser();
    if (currentUser == null) return;

    db.collection(COLLECTION_USERS)
        .document(currentUser.getId())
        .collection(COLLECTION_BUDGETS)
        .document(budgetId)
        .delete()
        .addOnSuccessListener(aVoid -> loadActiveBudgets());
}
}

```

### 3.3.6. Code xử lý nhắc nhở (thêm/sửa/xóa/xem)

```

// Lấy danh sách nhắc nhở sắp tới (chưa hoàn thành)
public Task<QuerySnapshot> getUpcomingReminders() {
    String userId = auth.getCurrentUser().getId();
    Date now = new Date();

    Log.d("ReminderRepository", "Getting upcoming reminders, current time: " + now);

    return getRemindersCollection()
        .whereEqualTo("userId", userId)
        .whereEqualTo("isCompleted", false)
        .whereGreaterThanOrEqualTo("dateTime", now) // Chỉ lấy nhắc nhở có thời
gian từ hiện tại trở đi
        .orderBy("dateTime", Query.Direction.ASCENDING)
        .get();
}

// Lấy danh sách nhắc nhở đã qua (đã hoàn thành hoặc quá hạn)
public Task<QuerySnapshot> getPastReminders() {

```

```

String userId = auth.getCurrentUser().getUid();

Log.d("ReminderRepository", "Getting past reminders for user: " + userId);

return getRemindersCollection()
    .whereEqualTo("userId", userId)
    .whereEqualTo("isCompleted", true)
    .get()
    .addOnSuccessListener(querySnapshot -> {
        Log.d("ReminderRepository", "Found " + querySnapshot.size() + "
completed reminders");
        for (DocumentSnapshot doc : querySnapshot.getDocuments()) {
            Log.d("ReminderRepository", "Completed reminder: " + doc.getId()
+ ", title: " + doc.getString("title"));
        }
    })
    .addOnFailureListener(e -> {
        Log.e("ReminderRepository", "Error getting completed reminders", e);
    });
}

public Task<QuerySnapshot> getOverdueReminders() {
    String userId = auth.getCurrentUser().getUid();
    Date now = new Date();

    return getRemindersCollection()
        .whereEqualTo("userId", userId)
        .whereEqualTo("isCompleted", false)
        .whereLessThan("dateTime", now)
        .orderBy("dateTime", Query.Direction.DESCENDING)
        .get();
}

// Thêm nhắc nhở mới
public Task<DocumentReference> addReminder(Reminder reminder) {
    // Đảm bảo reminder có userId
    if (reminder.getUserId() == null && auth.getCurrentUser() != null) {
        reminder.setUserId(auth.getCurrentUser().getUid());
    }

    Map<String, Object> reminderMap = convertReminderToMap(reminder);
    return getRemindersCollection().add(reminderMap);
}

// Cập nhật nhắc nhở
public Task<Void> updateReminder(String documentId, Reminder reminder) {
    Map<String, Object> reminderMap = convertReminderToMap(reminder);
    return getRemindersCollection().document(documentId).update(reminderMap);
}

// Đánh dấu nhắc nhở là đã hoàn thành
public Task<Void> markReminderAsCompleted(String documentId) {
    return getRemindersCollection()
        .document(documentId)
        .update("isCompleted", true);
}

// Xóa nhắc nhở
public Task<Void> deleteReminder(String documentId) {
    return getRemindersCollection().document(documentId).delete();
}

// Lấy nhắc nhở theo ID

```

```

public Task<DocumentSnapshot> getReminderById(String documentId) {
    return getRemindersCollection().document(documentId).get();
}

// Lấy nhắc nhở theo ID số
public Task<QuerySnapshot> getReminderByNumericId(long numericId) {
    return getRemindersCollection()
        .whereEqualTo("numericId", numericId)
        .limit(1)
        .get();
}

```

### 3.3.7. Code xử lý các thông báo

Vì để thực hiện được chức năng này, cần code các service bao gồm xử lý các Action Receiver, Broadcast Receiver, ... nên khá là dài, tóm tắt sẽ có các code để thực hiện những việc sau:

- Tạo kênh thông báo
- Lên lịch thông báo
- Chức năng hiển thị, xóa thông báo
- Xử lý nhắc nhở chi tiêu bằng cách nhận Broadcast và hiển thị thông báo
- ...

## KẾT LUẬN

### 1. Kết quả đạt được

Sau quá trình nghiên cứu và phát triển, ứng dụng quản lý chi tiêu đã hoàn thành với các tính năng cốt lõi bao gồm:

- **Ghi chép giao dịch:** Cho phép người dùng thêm, sửa, xóa và xem lịch sử chi tiêu.
- **Phân loại chi tiêu:** Hỗ trợ nhóm giao dịch theo danh mục như ăn uống, mua sắm, hóa đơn,...
- **Mục tiêu lâu dài:** Cho phép người dùng đặt mục tiêu tiết kiệm như mua nhà, du lịch, quỹ khẩn cấp,... và theo dõi tiến độ đạt mục tiêu theo thời gian.
- **AI chat bot tư vấn tài chính:** Tích hợp trợ lý ảo sử dụng trí tuệ nhân tạo, có khả năng phân tích dữ liệu chi tiêu, lịch sử giao dịch, mục tiêu tài chính để đưa ra lời khuyên cá nhân hóa cho người dùng, giúp họ quản lý tiền bạc thông minh hơn.
- **Nhắc nhở thanh toán:** Hệ thống gửi thông báo nhắc nhở khi đến thời điểm cần thanh toán.
- **Tự động tạo giao dịch:** Khi người dùng xác nhận thanh toán nhắc nhở, ứng dụng tự động cập nhật giao dịch vào lịch sử chi tiêu.
- **Quản lý ngân sách:** Người dùng có thể thiết lập ngân sách cho từng danh mục hoặc tổng ngân sách hàng tháng, theo dõi mức tiêu dùng so với ngân sách.

- **Thống kê & Báo cáo:** Hiện thị biểu đồ trực quan về chi tiêu theo thời gian, danh mục, đồng thời cung cấp báo cáo định kỳ theo tuần/tháng.
- **Lưu trữ dữ liệu:** Ứng dụng sử dụng Firebase để lưu trữ thông tin giao dịch và nhắc nhở, giúp người dùng theo dõi tài chính một cách hiệu quả.

## 2. Nhược điểm

Mặc dù ứng dụng đã đáp ứng các yêu cầu cơ bản, nhưng vẫn còn một số hạn chế:

- **Chưa hỗ trợ đa nền tảng:** Hiện tại, ứng dụng chỉ chạy trên Android, chưa có phiên bản iOS hoặc Web.
- **Chưa tối ưu giao diện thống kê:** Biểu đồ và báo cáo hiện tại còn đơn giản, chưa hỗ trợ tùy chỉnh theo nhu cầu người dùng.
- **Hiệu suất xử lý dữ liệu lớn:** Khi số lượng giao dịch nhiều, hiệu suất truy vấn có thể bị ảnh hưởng.

## 3. Hướng phát triển

Trong tương lai, ứng dụng có thể được mở rộng và nâng cấp với các tính năng sau:

- **Phát triển phiên bản đa nền tảng:** Hỗ trợ iOS và Web để người dùng có thể sử dụng trên nhiều thiết bị khác nhau.
- **Cải thiện giao diện thống kê & báo cáo:** Tích hợp thêm các biểu đồ chi tiết hơn, hỗ trợ tùy chỉnh báo cáo theo nhu cầu.