

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: ỨNG DỤNG QUẢN LÝ CÔNG VIỆC (TODO LIST)**

Giáo viên hướng dẫn: ThS. Kiều Tuấn Dũng

Sinh viên thực hiện:

STT	Mã sinh viên	Họ và tên	Lớp
1	2151062830	Trần Quang Minh	63CNTT4
2	2151062828	Lưu Công Minh	63CNTT4

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: ỨNG DỤNG**

STT	Mã Sinh Viên	Họ và Tên	Ngày Sinh	Điểm	
				Bảng Số	Bảng Chữ
1	2151062830	Trần Quang Minh	13/10/2003		
2	2151062828	Lưu Công Minh	14/03/2003		

CÁN BỘ CHẤM THI

Hà Nội, năm 2025

LỜI NÓI ĐẦU

Trong bối cảnh cách mạng công nghiệp 4.0 đang phát triển mạnh mẽ, công nghệ thông tin đã và đang đóng vai trò quan trọng trong việc nâng cao năng suất lao động và chất lượng cuộc sống. Sự phổ biến của các thiết bị di động thông minh cùng với kết nối internet tốc độ cao đã tạo điều kiện thuận lợi cho việc ứng dụng công nghệ vào nhiều lĩnh vực khác nhau, trong đó có quản lý thời gian và công việc cá nhân.

Với nhịp sống hiện đại ngày càng hối hả, con người luôn phải đối mặt với khối lượng công việc lớn và áp lực về thời gian. Việc ghi nhớ và sắp xếp công việc một cách khoa học, hợp lý trở thành nhu cầu thiết yếu. Tuy nhiên, phương pháp ghi chép truyền thống trên giấy hoặc trí nhớ cá nhân thường thiếu tính linh hoạt và dễ dẫn đến bỏ sót. Từ thực tế đó, việc phát triển một ứng dụng hỗ trợ quản lý công việc trên nền tảng di động không chỉ là một giải pháp tiện lợi mà còn thể hiện sự ứng dụng hiệu quả của công nghệ vào đời sống.

Ứng dụng “Quản lý Công Việc - Todo List” được xây dựng nhằm mục tiêu giúp người dùng tổ chức công việc hằng ngày một cách khoa học, thông minh và cá nhân hóa. Ứng dụng cho phép người dùng tạo và quản lý danh sách công việc theo từng danh mục như Sức khỏe, Công việc, Sức khỏe tinh thần và Khác. Ngoài ra, người dùng còn có thể đặt lịch nhắc nhở, nhận thông báo đúng giờ, phân loại công việc theo mức độ ưu tiên, chia sẻ và cộng tác công việc với người khác, đồng bộ với Google Calendar, và hiển thị lịch theo ngày/tháng một cách trực quan.

Không chỉ chú trọng đến chức năng, ứng dụng còn được thiết kế với giao diện đơn giản, hiện đại, dễ sử dụng, phù hợp với mọi đối tượng người dùng từ học sinh, sinh viên đến người đi làm. Với sự hỗ trợ của ứng dụng, người dùng có thể dễ dàng kiểm soát lịch trình, nâng cao hiệu suất làm việc, hạn chế tình trạng quên việc và giảm bớt áp lực trong cuộc sống hằng ngày.

Chúng em hy vọng rằng, với ứng dụng “Quản lý Công Việc - Todo List”, việc sắp xếp và hoàn thành công việc sẽ trở nên dễ dàng, hiệu quả và chủ động hơn. Từ đó góp phần hình thành thói quen quản lý thời gian khoa học và lối sống năng suất cho cộng đồng trong thời đại số hiện nay.

Mục lục

LỜI NÓI ĐẦU	3
DANH MỤC HÌNH ẢNH	6
DANH MỤC CÁC TỪ VIẾT TẮT	7
Chương 1. TỔNG QUAN VỀ ĐỀ TÀI	8
1.1. Giới thiệu về đề tài	8
1.2. Mục tiêu của đề tài	8
1.3. Phạm vi của đề tài	8
1.4 Phân chia nhiệm vụ	9
Chương 2. KIẾN TRÚC VÀ CÔNG NGHỆ	9
2.1. Phân tích yêu cầu	9
a) Xác định người dùng:	9
b) Thu thập yêu cầu:	9
c) Phân tích yêu cầu:	10
2.2. Thiết kế hệ thống	10
a) Xác định các lớp:	10
b) Môi quan hệ giữa các lớp:	11
c) Biểu đồ lớp cho mô hình miền:	11
d) Thiết kế giao diện:	12
2.3. Triển khai	12
2.4. Vận hành và bảo trì	12
Chương 3. XÂY DỰNG ỨNG DỤNG	13
3.1. Thiết kế Figma	13
3.2. Thiết kế CSDL	18
3.3. Giao diện ứng dụng	18
3.3.1. Màn hình giao diện đăng nhập	18

3.3.2. Màn hình đăng ký.....	20
3.3.3. Màn hình giao diện quên mật khẩu	21
3.3.4. Màn hình giao diện trang chủ.....	23
3.3.5. Màn hình giao diện thêm nội dung công việc	24
3.3.6. Màn hình giao diện đăng xuất.....	26
3.4. Code minh họa các chức năng cốt lõi.....	27
3.4.1. chức năng quên mật khẩu	27
3.4.2. chức năng đăng nhập.....	27
3.4.3. chức năng đăng ký.....	29
3.4.4. chức năng hẹn giờ và thông báo.....	31
3.4.5. chức năng tạo nội dung công việc	33
3.4.6. chức năng xóa công việc.....	36
3.4.7. chức năng phân loại công việc theo từng mục	38
KẾT LUẬN	41
1. Kết quả đạt được.....	41
2. Nhược điểm	41
3. Hướng phát triển	41
TÀI LIỆU THAM KHẢO	43

DANH MỤC HÌNH ẢNH

Hình 1: Giao diện màn hình đăng nhập.....	19
Hình 2: Giao diện màn hình đăng ký.....	21
Hình 3: Giao diện quên mật khẩu.....	22
Hình 4: Giao diện trang chủ.....	24
Hình 5: Giao diện thêm công việc.....	25
Hình 6: Giao diện màn hình đăng xuất.....	27
Hình 7: Code minh họa quên mật khẩu.....	27
Hình 8: Code minh họa đăng nhập.....	29
Hình 9: Code minh họa đăng ký.....	31
Hình 10: Code minh họa hẹn giờ.....	33
Hình 11: Code minh họa tạo nội dung.....	36
Hình 12: Code minh họa xóa công việc.....	38
Hình 13: Code minh họa phân loại.....	40

DANH MỤC CÁC TỪ VIẾT TẮT

STT	TỪ VIẾT TẮT	VIẾT ĐẦY ĐỦ
1	RWD	Responsive Web Design
2	UI	User Interface
3	UX	User Experience
4	IDE	Integrated Development Environment
5	API	Application Programming Interface
6	DB	Database
7	MVC	Model - View - Controller
8	CRUD	Create – Read - Update - Delete
9	JSON	JavaScript Object Notation
10	SDK	Software Development Kit
11	XML	eXtensible Markup Language

Chương 1. TỔNG QUAN VỀ ĐỀ TÀI

1.1. Giới thiệu về đề tài

Ứng dụng “Quản lý công việc” là một công cụ hỗ trợ người dùng quản lý thời gian và công việc, trong thời đại mà khối lượng công việc của mỗi người ngày càng lớn mà thời gian vẫn chỉ có 24 tiếng 1 ngày. Ứng dụng này được xây dựng bằng Java và phát triển trên nền tảng Android, nhằm cung cấp cho người dùng một phương tiện quản lý tiện lợi, hiệu quả và chính xác.

1.2. Mục tiêu của đề tài

Đề tài “Ứng dụng Quản lý công việc” được thực hiện với các mục tiêu chính sau:

- Xây dựng một ứng dụng di động trên nền tảng Android giúp người dùng dễ dàng tạo, theo dõi và quản lý các công việc hàng ngày.
- Hỗ trợ người dùng sắp xếp công việc theo từng danh mục cụ thể như: Sức khỏe, Công việc, Sức khỏe tinh thần và Khác, từ đó tăng hiệu suất làm việc và cải thiện chất lượng cuộc sống.
- Tích hợp các chức năng thông báo và nhắc nhở công việc đúng giờ bằng AlarmManager và Notification, giúp người dùng không bỏ lỡ các nhiệm vụ quan trọng.
- Cho phép người dùng đồng bộ công việc với Google Calendar, từ đó mở rộng khả năng kết nối và quản lý đa nền tảng.
- Áp dụng kiến thức lập trình Java, Firebase (Authentication và Realtime Database), AlarmManager, và kỹ thuật xây dựng giao diện người dùng (UI/UX) trên Android Studio để hoàn thiện ứng dụng.

1.3. Phạm vi của đề tài

Phạm vi của đề tài tập trung vào việc phát triển một ứng dụng di động chạy trên hệ điều hành Android với các chức năng chính sau:

- Tạo, chỉnh sửa và xóa công việc.
- Phân loại công việc theo 4 danh mục: Sức khỏe, Công việc, Sức khỏe tinh thần, Khác.
- Thiết lập thời gian nhắc nhở công việc, hiển thị thông báo khi đến giờ hẹn.
- Giao diện lịch (Calendar View) để người dùng dễ dàng xem công việc theo ngày.
- Đồng bộ dữ liệu công việc với Firebase Realtime Database.
- Xác thực người dùng bằng Firebase Authentication.
- Tính năng chia sẻ công việc cho người khác (ở mức cơ bản).
- Giao diện thân thiện, dễ sử dụng, phù hợp với nhiều đối tượng người dùng.

1.4 Phân chia nhiệm vụ

PHÂN CÔNG CÔNG VIỆC

Thành Viên	Công Việc
Trần Quang Minh (Nhóm trưởng)	<ul style="list-style-type: none">- Thực hiện thiết kế giao diện và code các chức năng:<ul style="list-style-type: none">• Màn hình chính• Đăng ký, đăng nhập• Thêm công việc, giao diện thêm công việc• Liên kết firebase authentication
Lưu Công Minh	<ul style="list-style-type: none">- Vẽ Figma- Code các chức năng<ul style="list-style-type: none">• Liên kết Cloud firebase• Đồng bộ với Google Calendar• Phân loại nội dung lưu công việc• Hẹn giờ thông báo nội dung công việc

Chương 2. KIẾN TRÚC VÀ CÔNG NGHỆ

2.1. Phân tích yêu cầu

a) Xác định người dùng:

- Người dùng cuối là bất kỳ ai cần sắp xếp, quản lý công việc trong cuộc sống hằng ngày như sinh viên, nhân viên văn phòng, người nội trợ, v.v...
- Người dùng có thể có các mức độ am hiểu công nghệ khác nhau, vì vậy ứng dụng được thiết kế đơn giản, dễ sử dụng và giao diện thân thiện.

b) Thu thập yêu cầu:

- Dựa trên nhu cầu quản lý thời gian và công việc, các chức năng chính được xác định:
 - Tạo công việc mới theo ngày, giờ cụ thể.
 - Phân loại công việc theo các danh mục: Sức khỏe, Công việc, Sức khỏe tinh thần, Khác.
 - Đặt lịch nhắc nhở bằng thông báo và rung/chuông thông qua AlarmManager.
 - Đồng bộ dữ liệu với Firebase.
 - Hiển thị lịch (Calendar View) để theo dõi công việc trong tháng.
 - Chia sẻ công việc với người khác.
- Yêu cầu phi chức năng:
 - Ứng dụng hoạt động ổn định.
 - Giao diện đơn giản, dễ thao tác.
 - Bảo mật thông tin người dùng khi sử dụng Firebase Authentication.

c) Phân tích yêu cầu:

- **Tạo công việc mới:** Người dùng có thể thêm công việc, chọn ngày/giờ thực hiện, phân loại và lưu trữ vào Firebase.
- **Quản lý danh mục:** Hệ thống cho phép người dùng chọn 1 trong 4 danh mục để sắp xếp công việc rõ ràng.
- **Lịch nhắc nhở:** Sử dụng AlarmManager và NotificationHelper để gửi thông báo đúng thời gian người dùng đặt.
- **Đồng bộ dữ liệu:** Firebase Realtime Database giúp lưu trữ và truy cập dữ liệu từ nhiều thiết bị.
- **Chia sẻ công việc:** Cho phép người dùng gửi lời mời người khác cùng quản lý một công việc chung.
- **Hiển thị lịch:** Dễ dàng theo dõi và kiểm tra các công việc theo từng ngày trên CalendarView.

2.2. Thiết kế hệ thống

a) Xác định các lớp:

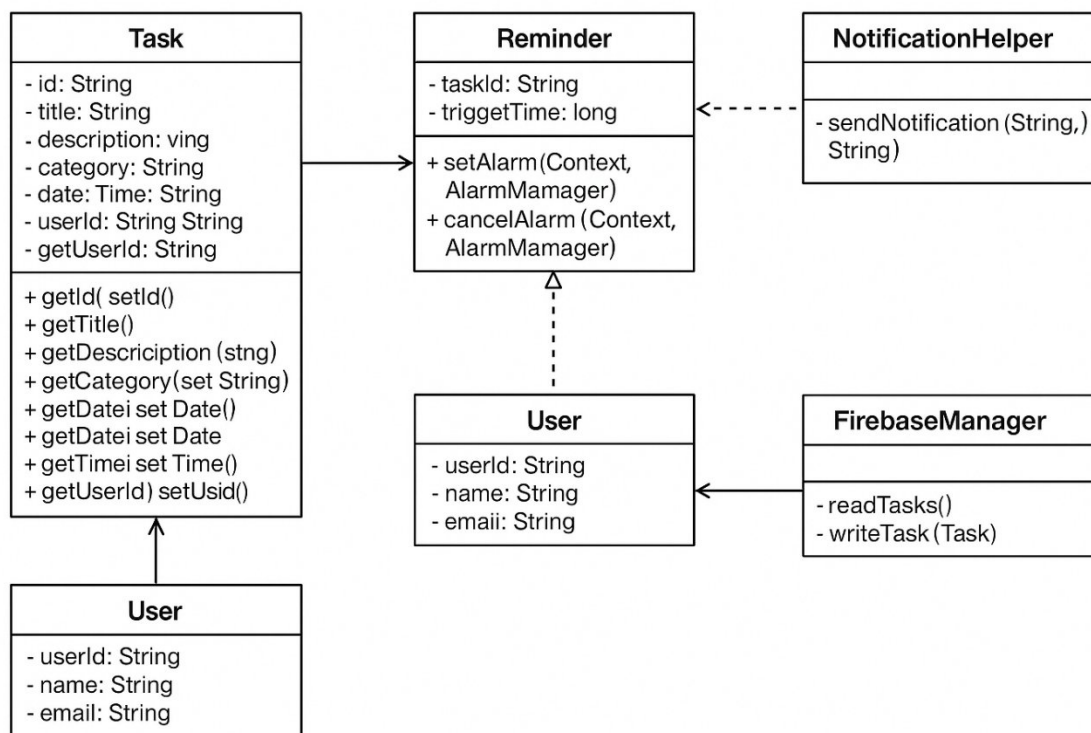
- **Lớp Task:**
 - String id, title, description, category, date, time, userId
 - Phương thức: getter/setter cho tất cả các thuộc tính.
- **Lớp Reminder:**
 - String taskId, long triggerTime
 - Phương thức: setAlarm, cancelAlarm liên kết với AlarmManager.
- **Lớp NotificationHelper:**
 - Gửi thông báo dựa trên thông tin từ lớp Reminder.
- **Lớp User:**

- String userId, name, email
- Quản lý đăng nhập, đăng ký bằng Firebase Authentication.
- **Lớp FirebaseManager:**
 - Đọc/ghi dữ liệu Task và User từ Firebase Realtime Database.

b) Môi quan hệ giữa các lớp:

- **User – Task:** Một người dùng (User) có thể tạo nhiều công việc (Task) → Quan hệ Một-Nhiều.
- **Task – Reminder:** Mỗi công việc (Task) có thể có một lời nhắc (Reminder).
- **Reminder – NotificationHelper:** Reminder kích hoạt NotificationHelper để gửi thông báo.
- **FirebaseManager – Các lớp dữ liệu:** FirebaseManager là cầu nối giữa ứng dụng và cơ sở dữ liệu đám mây.

c) Biểu đồ lớp cho mô hình miền:



Biểu đồ sẽ gồm các lớp: Task, Reminder, User, NotificationHelper, FirebaseManager, thể hiện các thuộc tính và phương thức, cùng quan hệ giữa các lớp theo mô tả ở mục b).

d) Thiết kế giao diện:

- **Màn hình chính:** Hiển thị ngày hiện tại, danh sách công việc, nút "+" để thêm công việc mới.
- **Màn hình thêm công việc:** Gồm tiêu đề, mô tả, ngày/giờ, danh mục.
- **Calendar View:** Hiển thị công việc theo tháng.
- **Thanh điều hướng:** Truy cập các danh mục công việc, Google Calendar, cài đặt.
- **Giao diện chia sẻ:** Màn hình gửi lời mời cộng tác.

2.3. Triển khai

- **Ngôn ngữ và nền tảng:** Sử dụng Java trên Android Studio.
- **Cơ sở dữ liệu:** Firebase Realtime Database để lưu trữ công việc và người dùng.
- **Xử lý thông báo:** AlarmManager và NotificationHelper được dùng để gửi thông báo đúng thời điểm.
- **Tính năng nâng cao:** Kết nối Google Calendar (nếu người dùng cho phép), chia sẻ công việc thông qua ID hoặc Email.
- **Giao diện:** Thiết kế bằng XML với Material Design, sử dụng RecyclerView để hiển thị danh sách công việc.

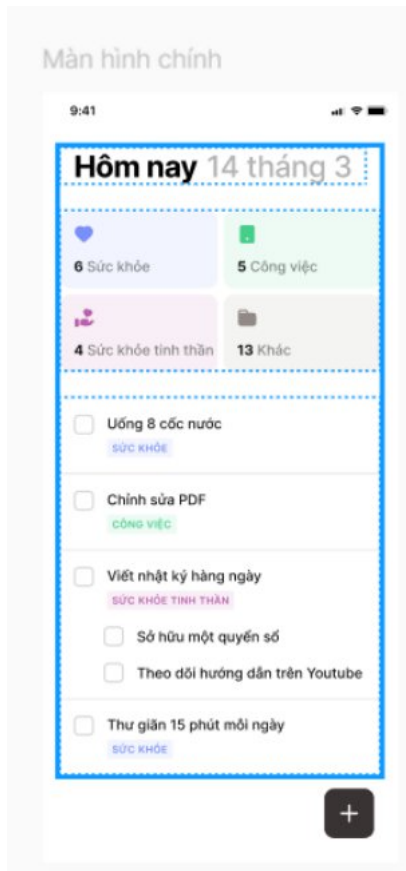
2.4. Vận hành và bảo trì

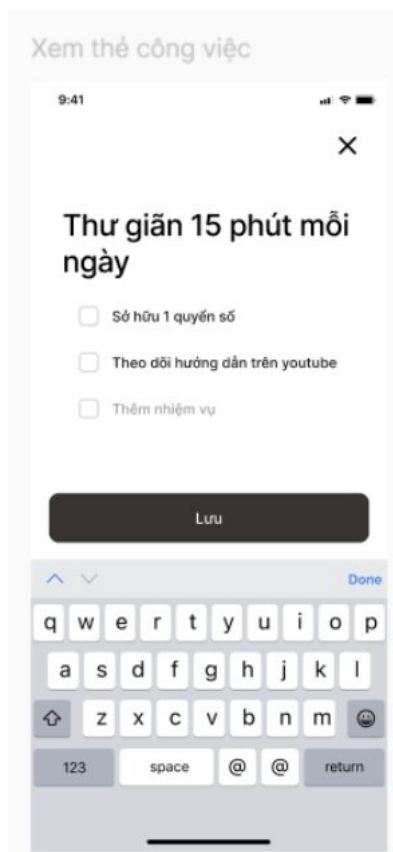
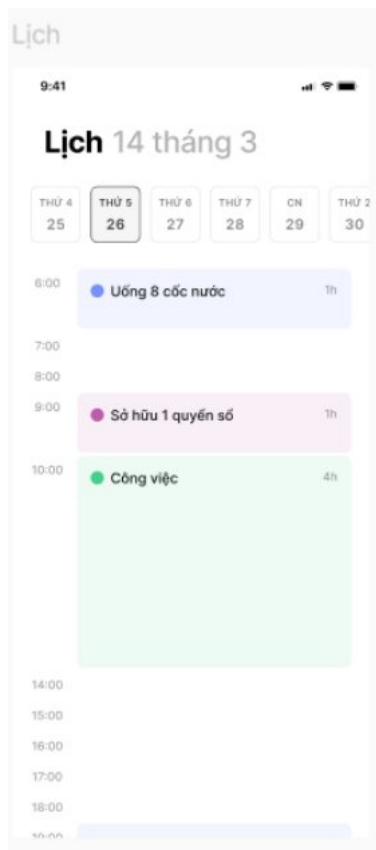
- **Cài đặt và triển khai:**
 - Ứng dụng được triển khai trên môi trường Android Studio.
 - Người dùng chỉ cần kết nối Internet và đăng nhập để bắt đầu sử dụng.
- **Bảo trì:**
 - Ứng dụng có thể mở rộng thêm tính năng như lọc/sắp xếp công việc theo trạng thái.
 - Sửa lỗi hiển thị, tối ưu truy xuất dữ liệu từ Firebase.
 - Cập nhật theo phản hồi người dùng và xu hướng công nghệ mới

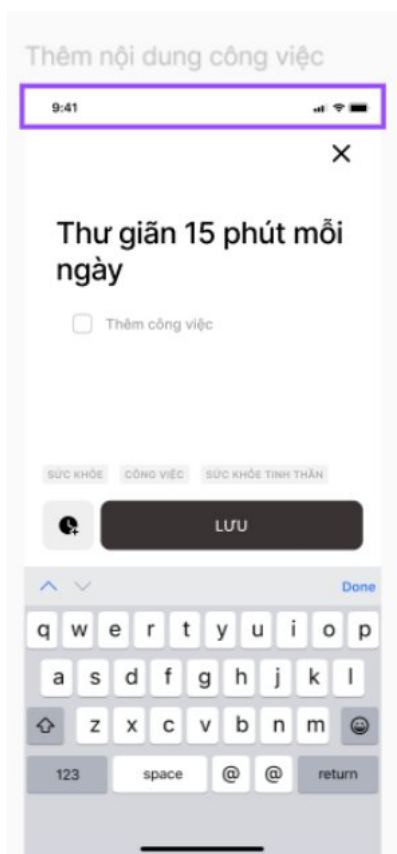
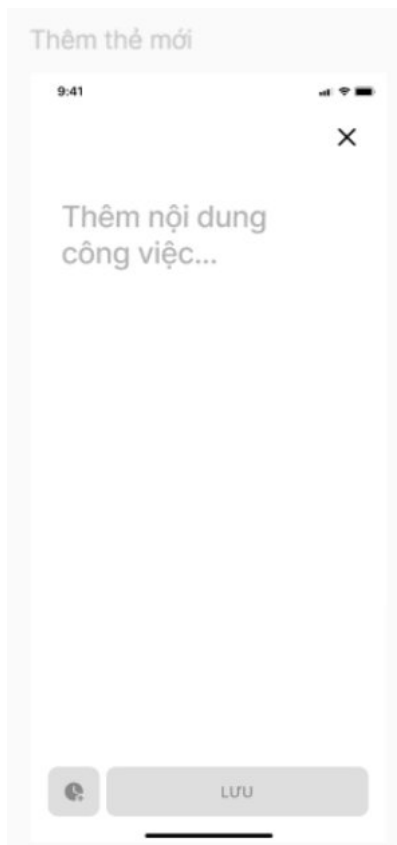
Chương 3. XÂY DỰNG ỨNG DỤNG

3.1. Thiết kế Figma

[Minhdeptrai1310/Finaltask](#)







Danh sách đơn giản

9:41

Hôm nay 14 tháng 3

- ☐ Uống 8 cốc nước
SỨC KHỎE
- ☐ Chỉnh sửa PDF
CÔNG VIỆC
- ☐ Viết nhật ký hàng ngày
SỨC KHỎE TÌNH THẦN
- ☐ Thư giãn 15 phút mỗi ngày
SỨC KHỎE

+

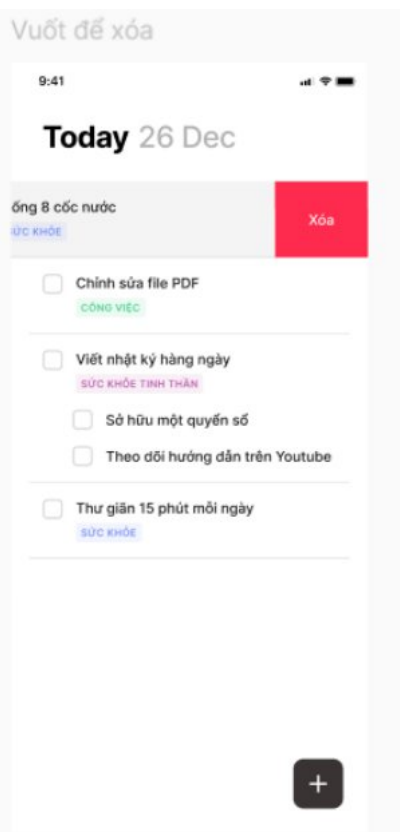
Có gắn thẻ

9:41

Hôm nay 14 tháng 3

- ☐ Uống 8 cốc nước
SỨC KHỎE
- ☐ Chỉnh sửa PDF
CÔNG VIỆC
- ☐ Viết nhật ký hàng ngày
SỨC KHỎE TÌNH THẦN
- ☐ Thư giãn 15 phút mỗi ngày
SỨC KHỎE

+

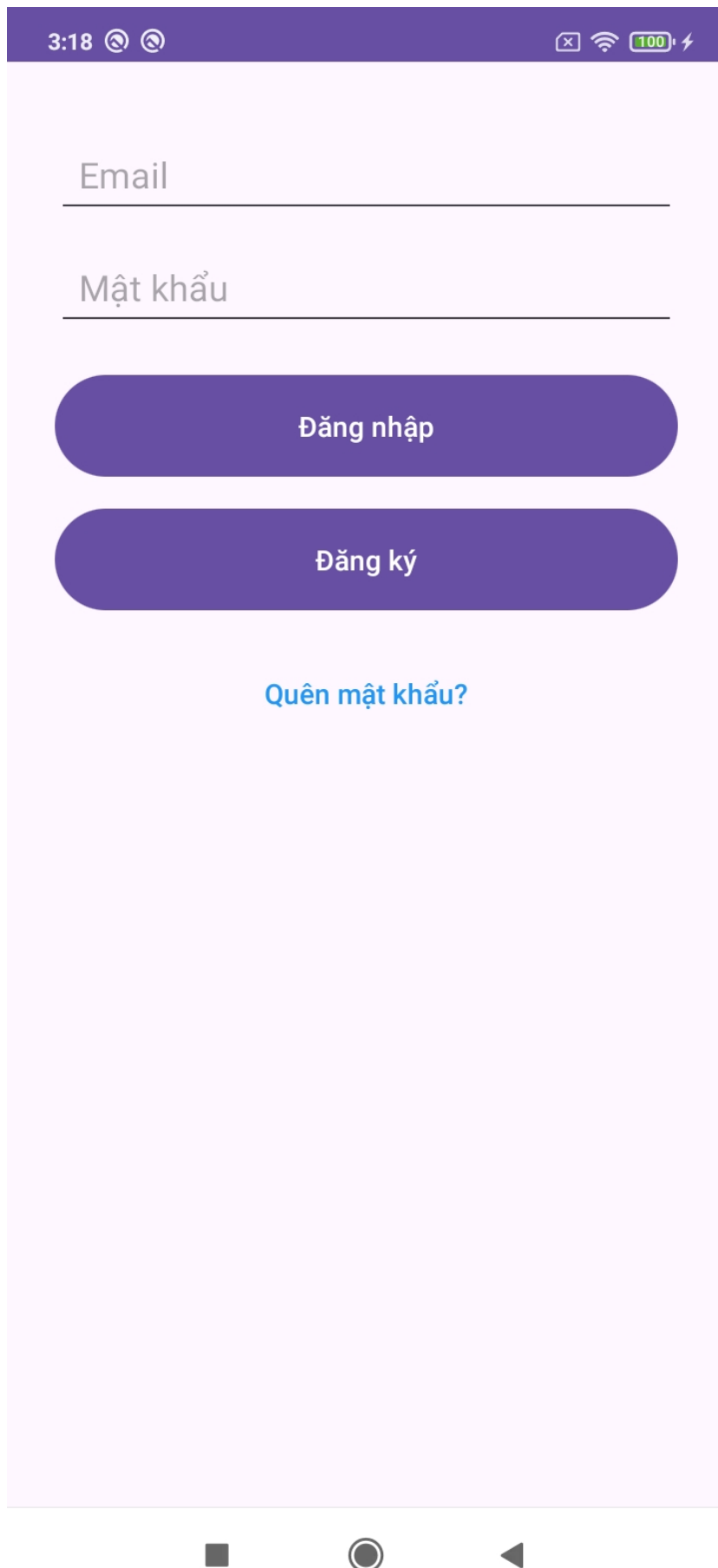


3.2. Thiết kế CSDL



3.3. Giao diện ứng dụng

3.3.1. Màn hình giao diện đăng nhập

A mobile application login screen with a light purple background. At the top is a dark purple status bar showing the time 3:18, signal strength, Wi-Fi, and 100% battery. Below the status bar are two input fields: 'Email' and 'Mật khẩu' (Password), each with a horizontal line underneath. Under the password field are two rounded purple buttons: 'Đăng nhập' (Login) and 'Đăng ký' (Register). Below these buttons is a blue link 'Quên mật khẩu?' (Forgot password?). At the bottom of the screen are three Android navigation icons: a square, a circle, and a triangle.

3:18

Email

Mật khẩu

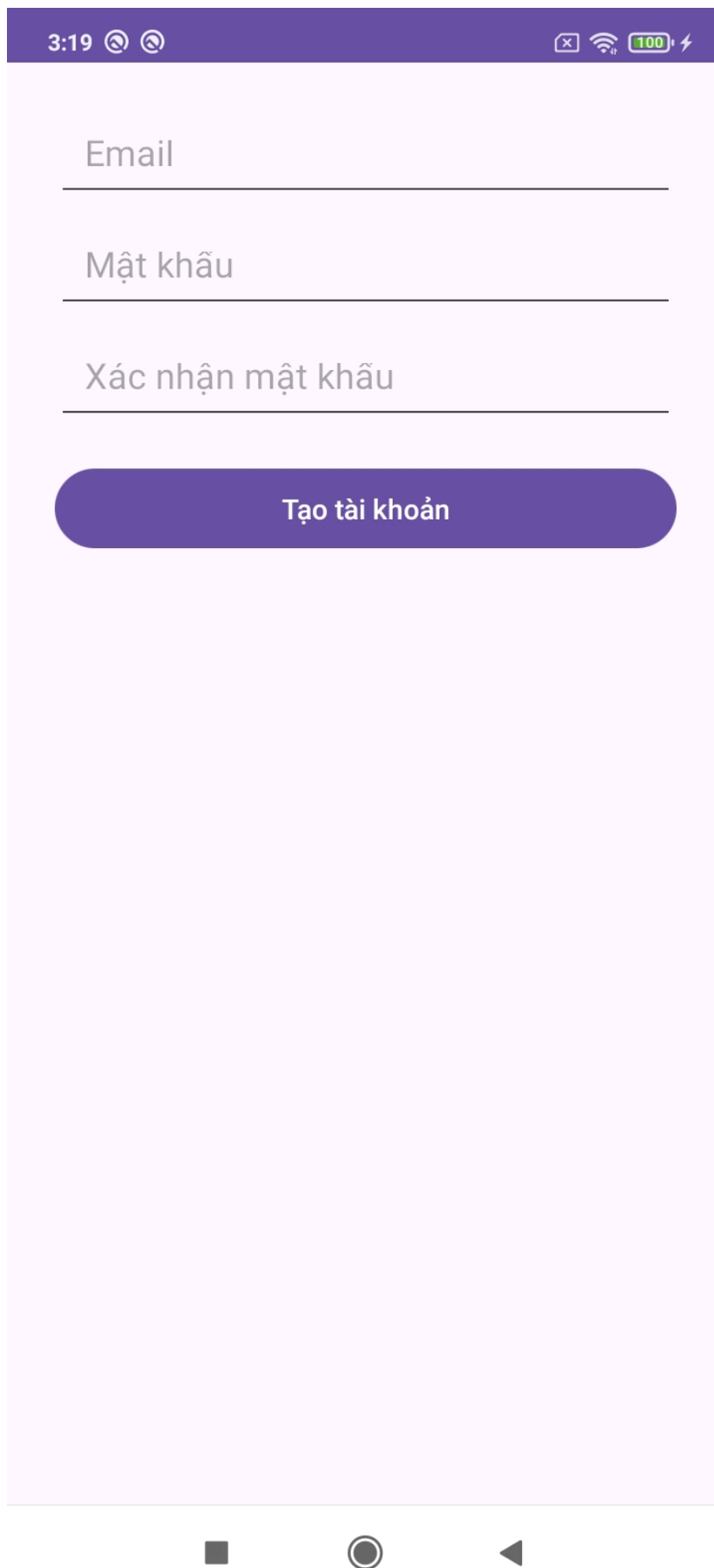
Đăng nhập

Đăng ký

[Quên mật khẩu?](#)

Hình 1: Giao diện màn hình đăng nhập

3.3.2. Màn hình đăng ký



The image shows a mobile application registration screen. At the top, there is a purple status bar with the time 3:19, two notification icons, a close button, a Wi-Fi icon, a 100% battery icon, and a charging icon. The main area has a light purple background. It contains three input fields with labels: 'Email', 'Mật khẩu' (Password), and 'Xác nhận mật khẩu' (Confirm password). Below these fields is a large purple button with the text 'Tạo tài khoản' (Create account). At the bottom of the screen, there are three Android navigation icons: a square, a circle, and a triangle.

3:19

Email

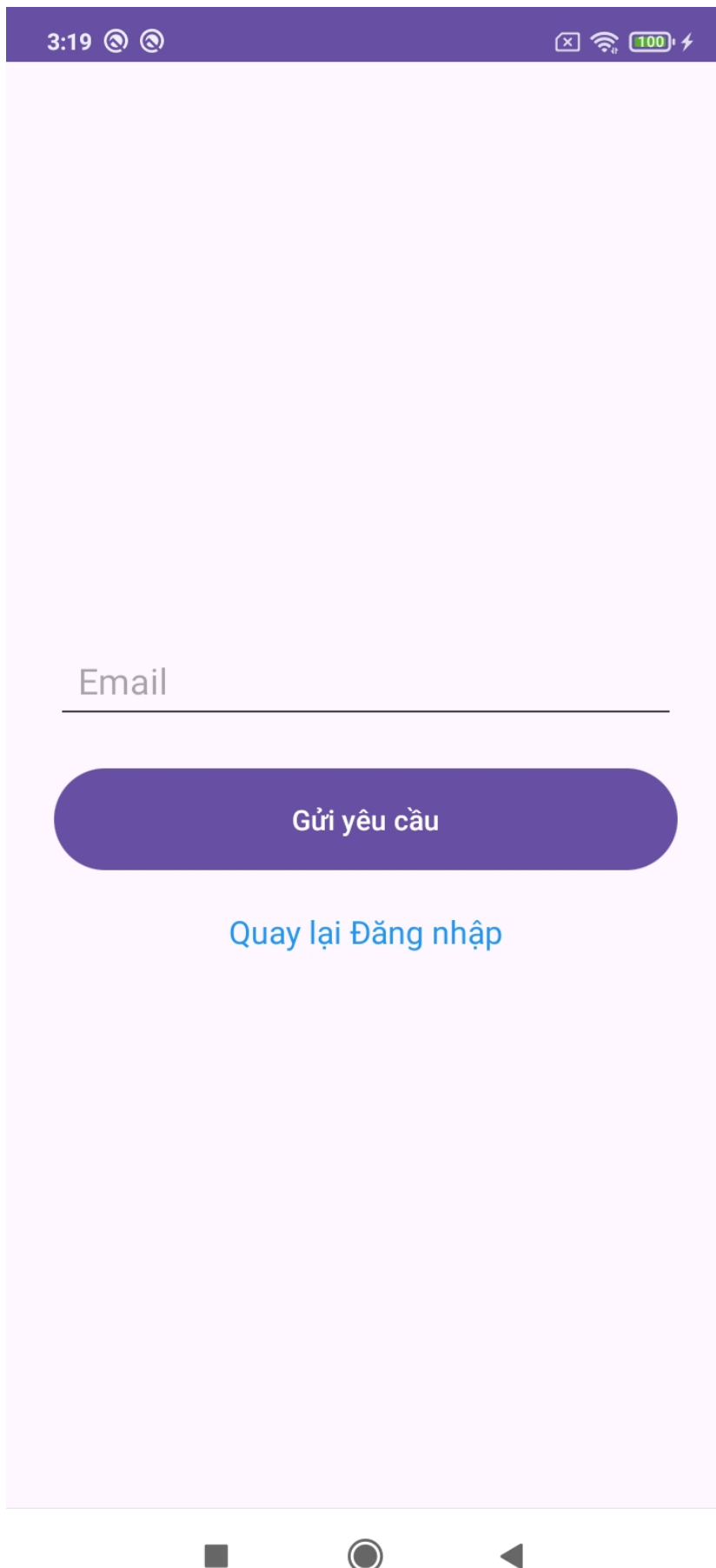
Mật khẩu

Xác nhận mật khẩu

Tạo tài khoản

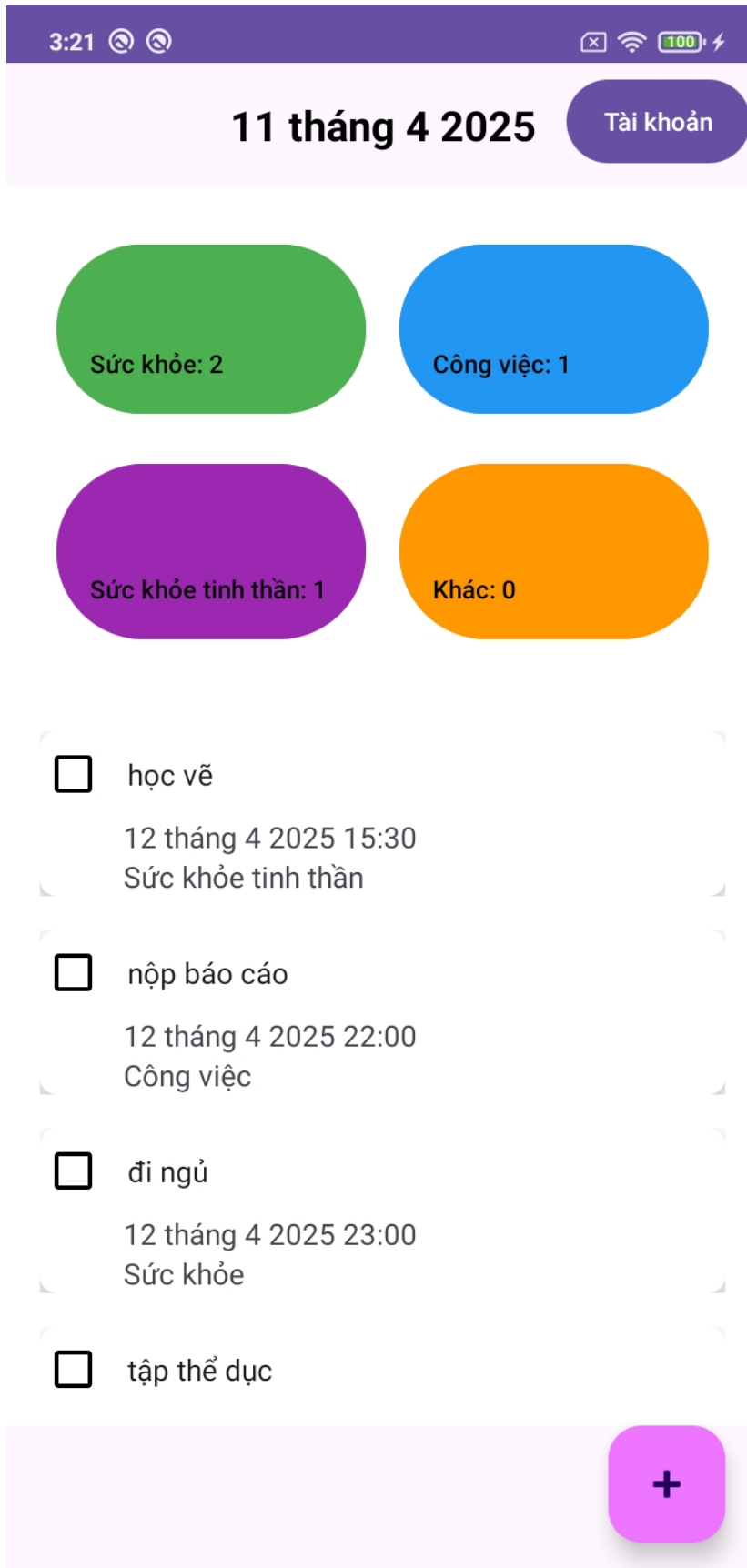
Hình 2: Giao diện màn hình đăng ký

3.3.3. Màn hình giao diện quên mật khẩu



Hình 3: Giao diện quên mật khẩu

3.3.4. Màn hình giao diện trang chủ



Hình 4: Giao diện trang chủ

3.3.5. Màn hình giao diện thêm nội dung công việc

3:22      100 

Nhập nội dung công việc

Chọn ngày

Sức khỏe 

LƯU CÔNG VIỆC

Hình 5: Giao diện thêm công việc

3.3.6. Màn hình giao diện đăng xuất



Hình 6: Giao diện màn hình đăng xuất

3.4. Code minh họa các chức năng cốt lõi

3.4.1. chức năng quên mật khẩu

```
public class ForgetActivity extends AppCompatActivity {
    2 usages
    private EditText etEmail;
    2 usages
    private Button btnResetPassword;
    2 usages
    private TextView tvBackToLogin;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_forget);

        // Khởi tạo các view
        etEmail = findViewById(R.id.etEmail);
        btnResetPassword = findViewById(R.id.btnResetPassword);
        tvBackToLogin = findViewById(R.id.tvBackToLogin);

        // Đặt sự kiện cho nút gửi yêu cầu reset mật khẩu
        btnResetPassword.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                String email = etEmail.getText().toString().trim();
                if (email.isEmpty()) {
                    Toast.makeText(context, ForgetActivity.this, text: "Vui lòng nhập email của bạn", Toast.LENGTH_SHORT).show();
                } else {
                    // Xử lý yêu cầu đặt lại mật khẩu
                    // Bạn có thể thêm logic để xử lý gửi email reset mật khẩu ở đây

                    Toast.makeText(context, ForgetActivity.this, text: "Yêu cầu đặt lại mật khẩu đã được gửi", Toast.LENGTH_SHORT).show();
                    // Quay lại màn hình đăng nhập
                    startActivity(new Intent(context, ForgetActivity.this, LoginActivity.class));
                }
            }
        });

        // Đặt sự kiện cho TextView quay lại trang đăng nhập
        tvBackToLogin.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                // Quay lại màn hình đăng nhập
                startActivity(new Intent(context, ForgetActivity.this, LoginActivity.class));
            }
        });
    }
}
```

Hình 7: Code minh họa quên mật khẩu

3.4.2. chức năng đăng nhập

```

1 package com.example.finaltask.login;
2
3 import android.content.Intent;
4 import android.os.Bundle;
5 import android.widget.Button;
6 import android.widget.EditText;
7 import android.widget.Toast;
8 import androidx.appcompat.app.AppCompatActivity;
9 import com.example.finaltask.MainActivity;
10 import com.example.finaltask.R;
11 import com.google.firebase.auth.FirebaseAuth;
12
13 public class LoginActivity extends AppCompatActivity {
14
15     private EditText etEmail, etPassword;
16     private Button btnLogin, btnRegister, btnForgotPassword;
17     private FirebaseAuth mAuth;
18
19     @Override
20     protected void onCreate(Bundle savedInstanceState) {
21         super.onCreate(savedInstanceState);
22         setContentView(R.layout.activity_login);
23
24         // Ánh xạ view
25         etEmail = findViewById(R.id.etEmail);
26         etPassword = findViewById(R.id.etPassword);
27         btnLogin = findViewById(R.id.btnLogin);
28         btnRegister = findViewById(R.id.btnRegister);
29         btnForgotPassword = findViewById(R.id.btnForgotPassword);
30
31         mAuth = FirebaseAuth.getInstance();
32
33         // Xử lý sự kiện đăng nhập
34         btnLogin.setOnClickListener( View v -> {
35             String email = etEmail.getText().toString().trim();
36             String password = etPassword.getText().toString().trim();
37
38             if (email.isEmpty() || password.isEmpty()) {
39                 Toast.makeText( context: this, text: "Vui lòng nhập đầy đủ thông tin", Toast.LENGTH_SHORT).show();
40             } else {
41                 mAuth.signInWithEmailAndPassword(email, password)
42                     .addOnSuccessListener( AuthResult authResult -> {
43                         Toast.makeText( context: this, text: "Đăng nhập thành công", Toast.LENGTH_SHORT).show();
44                         startActivity(new Intent( packageContext: this, MainActivity.class));
45                     });
46             }
47         });
48     }
49 }

```

```

31 mAuth = FirebaseAuth.getInstance();
32
33 // Xử lý sự kiện đăng nhập
34 btnLogin.setOnClickListener( View v -> {
35     String email = etEmail.getText().toString().trim();
36     String password = etPassword.getText().toString().trim();
37
38     if (email.isEmpty() || password.isEmpty()) {
39         Toast.makeText( context: this, text: "Vui lòng nhập đầy đủ thông tin", Toast.LENGTH_SHORT).show();
40     } else {
41         mAuth.signInWithEmailAndPassword(email, password)
42             .addOnSuccessListener( AuthResult authResult -> {
43                 Toast.makeText( context: this, text: "Đăng nhập thành công", Toast.LENGTH_SHORT).show();
44                 startActivity(new Intent( packageContext: this, MainActivity.class));
45                 finish(); // Đóng LoginActivity
46             })
47             .addOnFailureListener( Exception e ->
48                 Toast.makeText( context: this, text: "Lỗi: " + e.getMessage(), Toast.LENGTH_SHORT).show()
49             );
50     }
51 });
52
53 // Xử lý sự kiện đăng ký
54 btnRegister.setOnClickListener( View v -> {
55     startActivity(new Intent( packageContext: this, RegisterActivity.class));
56 });
57
58 // Xử lý sự kiện quên mật khẩu
59 btnForgotPassword.setOnClickListener( View v -> {
60     startActivity(new Intent( packageContext: this, ForgetActivity.class));
61 });
62 }
63
64 // THÊM CODE KIỂM TRA ĐĂNG NHẬP TẠI ĐÂY
65 @Override
66 protected void onStart() {
67     super.onStart();
68
69     // Nếu người dùng đã đăng nhập, tự động chuyển đến MainActivity
70     if (mAuth.getCurrentUser() != null) {
71         startActivity(new Intent( packageContext: this, MainActivity.class));
72         finish(); // Đóng LoginActivity để không thể quay lại bằng nút back
73     }
74 }
75 }

```

Hình 8: Code minh họa đăng nhập

3.4.3. chức năng đăng ký

```

1 package com.example.finaltask.login;
2
3 import android.content.Intent;
4 import android.os.Bundle;
5 import android.text.TextUtils;
6 import android.widget.Button;
7 import android.widget.EditText;
8 import android.widget.Toast;
9
10 import androidx.appcompat.app.AppCompatActivity;
11
12 import com.example.finaltask.R;
13 import com.google.firebase.auth.FirebaseAuth;
14
15 public class RegisterActivity extends AppCompatActivity {
16
17     2 usages
18     private EditText etEmail, etPassword, etConfirmPassword;
19     2 usages
20     private Button btnCreateAccount;
21     3 usages
22     private FirebaseAuth mAuth;
23
24     @Override
25     protected void onCreate(Bundle savedInstanceState) {
26         super.onCreate(savedInstanceState);
27         setContentView(R.layout.activity_register);
28
29         etEmail = findViewById(R.id.etRegisterEmail);
30         etPassword = findViewById(R.id.etRegisterPassword);
31         etConfirmPassword = findViewById(R.id.etConfirmPassword);
32         btnCreateAccount = findViewById(R.id.btnCreateAccount);
33
34         mAuth = FirebaseAuth.getInstance();
35
36         btnCreateAccount.setOnClickListener( View v -> registerUser());
37     }
38
39     1 usage
40     private void registerUser() {
41         String email = etEmail.getText().toString().trim();
42         String password = etPassword.getText().toString().trim();
43         String confirmPassword = etConfirmPassword.getText().toString().trim();
44
45         if (TextUtils.isEmpty(email) || TextUtils.isEmpty(password) || TextUtils.isEmpty(confirmPassword)) {
46             Toast.makeText( context this, text: "Vui lòng điền đầy đủ thông tin", Toast.LENGTH_SHORT).show();
47             return;
48         }
49     }
50 }

```

```

29     btnCreateAccount = findViewById(R.id.btnCreateAccount);
30
31     mAuth = FirebaseAuth.getInstance();
32
33     btnCreateAccount.setOnClickListener( View v -> registerUser());
34 }
35
36 1 usage
37 private void registerUser() {
38     String email = etEmail.getText().toString().trim();
39     String password = etPassword.getText().toString().trim();
40     String confirmPassword = etConfirmPassword.getText().toString().trim();
41
42     if (TextUtils.isEmpty(email) || TextUtils.isEmpty(password) || TextUtils.isEmpty(confirmPassword)) {
43         Toast.makeText( context this, text "Vui lòng điền đầy đủ thông tin", Toast.LENGTH_SHORT).show();
44         return;
45     }
46
47     if (!password.equals(confirmPassword)) {
48         Toast.makeText( context this, text "Mật khẩu không khớp", Toast.LENGTH_SHORT).show();
49         return;
50     }
51
52     if (password.length() < 6) {
53         Toast.makeText( context this, text "Mật khẩu phải có ít nhất 6 ký tự", Toast.LENGTH_SHORT).show();
54         return;
55     }
56
57     mAuth.createUserWithEmailAndPassword(email, password)
58         .addOnSuccessListener( AuthResult authResult -> {
59             mAuth.signOut(); // Đăng xuất ngay sau khi tạo tài khoản
60             Toast.makeText( context this, text "Tạo tài khoản thành công. Vui lòng đăng nhập lại.", Toast.LENGTH_SHORT).show();
61             startActivity(new Intent( packageContext RegisterActivity.this, LoginActivity.class)); // Chuyển về màn hình đăng nhập
62             finish();
63         })
64         .addOnFailureListener( Exception e ->
65             Toast.makeText( context this, text "Lỗi: " + e.getMessage(), Toast.LENGTH_SHORT).show());
66     }
67 }

```

Hình 9: Code minh họa đăng ký

3.4.4. chức năng hẹn giờ và thông báo


```

public class AlarmReceiver extends BroadcastReceiver {
    @SuppressWarnings
    private static MediaPlayer mediaPlayer;

    @Override
    public void onReceive(Context context, Intent intent) {
        // Kiểm tra quyền thông báo trên Android 13+
        if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.TIRAMISU) {
            if (ContextCompat.checkSelfPermission(context, Manifest.permission.POST_NOTIFICATIONS)
                != PackageManager.PERMISSION_GRANTED) {
                Log.d( tag: "AlarmReceiver", msg: "Không có quyền thông báo");
                return;
            }
        }

        // Lấy thông tin từ Intent
        String taskName = intent.getStringExtra( name: "taskName");
        String documentId = intent.getStringExtra( name: "documentId");

        // Tạo Intent mở ứng dụng
        Intent mainIntent = new Intent(context, MainActivity.class);
        PendingIntent pendingIntent = PendingIntent.getActivity(
            context,
            requestCode: 0,
            mainIntent,
            flags: PendingIntent.FLAG_IMMUTABLE | PendingIntent.FLAG_UPDATE_CURRENT
        );

        // Tạo Intent cho nút "TẮT"
        Intent dismissIntent = new Intent(context, DismissReceiver.class);
        dismissIntent.putExtra( name: "documentId", documentId);
        PendingIntent dismissPendingIntent = PendingIntent.getBroadcast(
            context,
            documentId.hashCode(),
            dismissIntent,
            PendingIntent.FLAG_IMMUTABLE
        );

        // Tạo thông báo với Channel ID đúng
        Notification notification = new NotificationCompat.Builder(context, channelId: "task_notifications_channel")
            .setSmallIcon(R.drawable.ic_launcher_foreground)
            .setContentTitle("Đến giờ: " + taskName)
            .setContentText("Hãy kiểm tra công việc!")
    }
}

```



```

63     .setSmallIcon(R.drawable.ic_launcher_foreground)
64     .setContentTitle("Đến giờ: " + taskName)
65     .setContentText("Hãy kiểm tra công việc!")
66     .addAction(R.drawable.ic_close, "TẮT", dismissPendingIntent)
67     .setPriority(NotificationCompat.PRIORITY_HIGH)
68     .setContentIntent(pendingIntent)
69     .setAutoCancel(true)
70     .build();
71
72     // Hiển thị thông báo
73     NotificationManagerCompat notificationManager = NotificationManagerCompat.from(context);
74     notificationManager.notify(documentId.hashCode(), notification);
75
76     // Phát âm thanh
77     playAlarmSound(context);
78 }
79
80 1 usage
81 private void playAlarmSound(Context context) {
82     Uri alarmUri = RingtoneManager.getDefaultUri(RingtoneManager.TYPE_ALARM);
83     if (alarmUri == null) {
84         alarmUri = RingtoneManager.getDefaultUri(RingtoneManager.TYPE_NOTIFICATION);
85     }
86
87     try {
88         mediaPlayer = MediaPlayer.create(context, alarmUri);
89         if (mediaPlayer != null) {
90             mediaPlayer.setLooping(true);
91             mediaPlayer.start();
92         }
93     } catch (Exception e) {
94         e.printStackTrace();
95     }
96
97     1 usage
98     public static void stopAlarm() {
99         if (mediaPlayer != null && mediaPlayer.isPlaying()) {
100             mediaPlayer.stop();
101             mediaPlayer.release();
102             mediaPlayer = null;
103             Log.d("AlarmReceiver", "Âm thanh đã dừng");
104         }
105     }

```

Hình 10: Code minh họa hẹn giờ

3.4.5. chức năng tạo nội dung công việc

```

28 public class AddNewTask extends Activity {
    2 usages
29     private EditText etTaskName, etTaskDate;
    3 usages
30     private Spinner spCategory;
    2 usages
31     private Button btnSave;
    13 usages
32     private Calendar calendar;
    2 usages
33     private FirebaseFirestore db;
34
35     @Override
36     protected void onCreate(Bundle savedInstanceState) {
37         super.onCreate(savedInstanceState);
38         setContentView(R.layout.add_new_task);
39
40         // Anh xạ view
41         etTaskName = findViewById(R.id.etTaskName);
42         etTaskDate = findViewById(R.id.etTaskDate);
43         spCategory = findViewById(R.id.spinnerTaskCategory);
44         btnSave = findViewById(R.id.btnSave);
45
46         calendar = Calendar.getInstance();
47         db = FirebaseFirestore.getInstance();
48
49         setupSpinner();
50         etTaskDate.setOnClickListener( View v -> showDateTimePicker());
51         btnSave.setOnClickListener( View v -> saveTaskToFirestore());
52     }
53
54     1 usage
55     private void setupSpinner() {
56         ArrayAdapter<CharSequence> adapter = ArrayAdapter.createFromResource(
57             context: this,
58             R.array.task_categories,
59             android.R.layout.simple_spinner_item
60         );
61         adapter.setDropDownViewResource(android.R.layout.simple_spinner_dropdown_item);
62         spCategory.setAdapter(adapter);
63     }

```

```

1 usage
64 private void showDateTimePicker() {
65     new DatePickerDialog(
66         context: this,
67         ( DatePicker view, int year, int month, int dayOfMonth) -> {
68             calendar.set(year, month, dayOfMonth);
69             new TimePickerDialog(
70                 context: this,
71                 ( TimePicker timeView, int hourOfDay, int minute) -> {
72                     calendar.set(Calendar.HOUR_OF_DAY, hourOfDay);
73                     calendar.set(Calendar.MINUTE, minute);
74                     updateDateTimeDisplay();
75                 },
76                 calendar.get(Calendar.HOUR_OF_DAY),
77                 calendar.get(Calendar.MINUTE),
78                 is24HourView: true
79             ).show();
80         },
81         calendar.get(Calendar.YEAR),
82         calendar.get(Calendar.MONTH),
83         calendar.get(Calendar.DAY_OF_MONTH)
84     ).show();
85 }
86
1 usage
87 private void updateDateTimeDisplay() {
88     String dateTimeFormat = "dd MMMM yyyy HH:mm";
89     SimpleDateFormat sdf = new SimpleDateFormat(dateTimeFormat, Locale.getDefault());
90     etTaskDate.setText(sdf.format(calendar.getTime()));
91 }
92
1 usage
93 private void saveTaskToFirestore() {
94     String taskName = etTaskName.getText().toString().trim();
95     String taskCategory = spCategory.getSelectedItem().toString();
96     String taskDateTime = new SimpleDateFormat( pattern: "dd MMMM yyyy HH:mm", Locale.getDefault())
97         .format(calendar.getTime());
98
99     if (taskName.isEmpty()) {
100         Toast.makeText( context: this, text: "Vui lòng nhập tên công việc", Toast.LENGTH_SHORT).show();
101         return;
102     }

```

```

104 // Tạo task với timestamp
105 Task newTask = new Task();
106 newTask.setTaskName(taskName);
107 newTask.setTaskCategory(taskCategory);
108 newTask.setTaskDateTime(taskDateTime);
109 newTask.setTaskTimestamp(calendar.getTimeInMillis());
110
111 // Lưu task lên Firestore và lấy documentId
112 db.collection(collectionPath: "tasks") CollectionReference
113     .add(newTask) Task<DocumentReference>
114     .addOnSuccessListener(DocumentReference documentReference -> {
115         String documentId = documentReference.getId();
116         setAlarm(calendar, documentId, taskName); // Truyền thêm taskName
117         Toast.makeText(context: this, text: "Task đã được lưu!", Toast.LENGTH_SHORT).show();
118         finish();
119     })
120     .addOnFailureListener(Exception e -> {
121         Toast.makeText(context: this, text: "Lỗi khi lưu task", Toast.LENGTH_SHORT).show();
122         Log.e(tag: "AddNewTask", msg: "Error saving task", e);
123     });
124 }
125
126 1 usage
127 private void setAlarm(Calendar calendar, String documentId, String taskName) {
128     Intent intent = new Intent(packageContext: this, AlarmReceiver.class);
129     intent.putExtra(name: "taskName", taskName);
130     intent.putExtra(name: "documentId", documentId); // Truyền documentId
131
132     // Tạo requestCode từ documentId
133     int requestCode = documentId.hashCode();
134
135     PendingIntent pendingIntent = PendingIntent.getBroadcast(
136         context: this,
137         requestCode,
138         intent,
139         PendingIntent.FLAG_IMMUTABLE
140     );
141
142     AlarmManager alarmManager = (AlarmManager) getSystemService(ALARM_SERVICE);
143     if (alarmManager != null) {
144         alarmManager.setExact(AlarmManager.RTC_WAKEUP, calendar.getTimeInMillis(), pendingIntent);
145         Log.d(tag: "AddNewTask", msg: "Alarm set for document: " + documentId);
146     }
147 }

```

Hình 11: Code minh họa tạo nội dung

3.4.6. chức năng xóa công việc

```

//Xử lý sự kiện cho các nút và swipe

1 usage
private void setupEventListeners() {
    fabAddTask.setOnClickListener( View v ->
        startActivityForResult(new Intent( packageContext: this, AddNewTask.class), requestCode: 1)
    );
    btnAccount.setOnClickListener( View v ->
        startActivity(new Intent( packageContext: this, AccountActivity.class))
    );
    setupSwipeToDelete();
    setupCategoryButtons();
}

//hiết lập chức năng swipe để xóa task

1 usage
private void setupSwipeToDelete() {
    new ItemTouchHelper(new ItemTouchHelper.SimpleCallback( dragDirs: 0, ItemTouchHelper.LEFT) {
        @Override
        public boolean onMove(@NonNull RecyclerView recyclerView,
                               @NonNull RecyclerView.ViewHolder viewHolder,
                               @NonNull RecyclerView.ViewHolder target) {
            return false;
        }

        3 usages
        @Override
        public void onSwiped(@NonNull RecyclerView.ViewHolder viewHolder, int direction) {
            int position = viewHolder.getAdapterPosition();
            DocumentSnapshot snapshot = taskAdapter.getSnapshots().getSnapshot(position);
            handleTaskDeletion(position, snapshot);
        }

        6 usages
        @Override
        public void onChildDraw(@NonNull Canvas c, @NonNull RecyclerView recyclerView,
                                @NonNull RecyclerView.ViewHolder viewHolder,
                                float dx, float dy, int actionState, boolean isCurrentlyActive) {
            new RecyclerViewSwipeDecorator.Builder(c, recyclerView, viewHolder, dx, dy, actionState, isCurrentlyActive)
                .addSwipeLeftBackgroundColor(ContextCompat.getColor( context: MainActivity.this, android.R.color.holo_red_light)) Builder
                .addSwipeLeftActionIcon(R.drawable.ic_delete)
                .addSwipeLeftLabel("XÓA")
                .setSwipeLeftLabelColor(ContextCompat.getColor( context: MainActivity.this, android.R.color.white))
            }
        }
    }
}

```

```

156         super.onChildDraw(c, recyclerView, viewHolder, dX, dY, actionState, isCurrentlyActive);
157     }
158     }).attachToRecyclerView(recyclerView);
159 }
160
161 //Xử lý việc xóa task khi người dùng swipe
162
163 @usage
164 private void handleTaskDeletion(int position, DocumentSnapshot snapshot) {
165     String documentId = snapshot.getId();
166     String taskName = snapshot.getString(field: "taskName");
167     showDeleteConfirmationDialog(position, documentId, taskName);
168 }
169
170 //Hiển thị hộp thoại xác nhận xóa task
171
172 @usage
173 private void showDeleteConfirmationDialog(int position, String documentId, String taskName) {
174     new AlertDialog.Builder(context: this)
175         .setTitle("Xác nhận xóa")
176         .setMessage("Bạn có chắc muốn xóa công việc: " + taskName + "?")
177         .setPositiveButton(text: "Có", (DialogInterface dialog, int which) -> deleteTask(position, documentId))
178         .setNegativeButton(text: "Không", (DialogInterface dialog, int which) -> taskAdapter.notifyItemChanged(position))
179         .show();
180 }
181
182 //Xóa task từ Firestore và hủy thông báo liên quan
183
184 @usage
185 private void deleteTask(int position, String documentId) {
186     cancelAlarm(documentId);
187     db.collection(collectionPath: "tasks").document(documentId).delete()
188         .addOnSuccessListener(Void aVoid -> {
189             Toast.makeText(context: this, text: "Đã xóa công việc", Toast.LENGTH_SHORT).show();
190             updateCategoryCounts();
191         })
192         .addOnFailureListener(Exception e -> {
193             Toast.makeText(context: this, text: "Lỗi khi xóa công việc", Toast.LENGTH_SHORT).show();
194             taskAdapter.notifyItemChanged(position);
195         });
196 }

```

Hình 12: Code minh họa xóa công việc

3.4.7. chức năng phân loại công việc theo từng mục


```

214 //Cập nhật số lượng task theo từng danh mục
215 3 usages
216 private void updateCategoryCounts() {
217     Map<String, Integer> counts = new HashMap<>();
218     String[] categories = {"Sức khỏe", "Công việc", "Sức khỏe tinh thần", "Khác"};
219     for (String category : categories) counts.put(category, 0);
220
221     db.collection( collectionPath: "tasks").get().addOnSuccessListener( QuerySnapshot queryDocumentSnapshots -> {
222         for (DocumentSnapshot snapshot : queryDocumentSnapshots) {
223             String category = snapshot.getString( field: "taskCategory");
224             counts.put(category, counts.getOrDefault(category, defaultValue: 0) + 1);
225         }
226         updateCategoryButtons(counts);
227     });
228
229 //Cập nhật giao diện các nút danh mục
230
231 1 usage
232 @ private void updateCategoryButtons(Map<String, Integer> counts) {
233     ((Button) findViewById(R.id.btnHealth)).setText("Sức khỏe: " + counts.get("Sức khỏe"));
234     ((Button) findViewById(R.id.btnWork)).setText("Công việc: " + counts.get("Công việc"));
235     ((Button) findViewById(R.id.btnMental)).setText("Sức khỏe tinh thần: " + counts.get("Sức khỏe tinh thần"));
236     ((Button) findViewById(R.id.btnOthers)).setText("Khác: " + counts.get("Khác"));
237 }
238
239 //Thiết lập sự kiện cho các nút danh mục
240
241 1 usage
242 private void setupCategoryButtons() {
243     findViewById(R.id.btnHealth).setOnClickListener( View v -> toggleCategory("Sức khỏe"));
244     findViewById(R.id.btnWork).setOnClickListener( View v -> toggleCategory("Công việc"));
245     findViewById(R.id.btnMental).setOnClickListener( View v -> toggleCategory("Sức khỏe tinh thần"));
246     findViewById(R.id.btnOthers).setOnClickListener( View v -> toggleCategory("Khác"));
247 }

```

```

247 //Chuyển đổi giữa các chế độ lọc danh mục
248
249 @
250 private void toggleCategory(String category) {
251     if (category.equals(selectedCategory)) {
252         selectedCategory = null;
253         showAllTasks();
254     } else {
255         selectedCategory = category;
256         filterTasksByCategory(category);
257     }
258     updateButtonStates(category);
259 }
260
261 //Cập nhật trạng thái selected cho các nút danh mục
262 1 usage
263 private void updateButtonStates(String category) {
264     int[] buttonIds = {R.id.btnHealth, R.id.btnWork, R.id.btnMental, R.id.btnOthers};
265     for (int id : buttonIds) findViewById(id).setSelected(false);
266     switch (category) {
267         case "Sức khỏe": findViewById(R.id.btnHealth).setSelected(true); break;
268         case "Công việc": findViewById(R.id.btnWork).setSelected(true); break;
269         case "Sức khỏe tinh thần": findViewById(R.id.btnMental).setSelected(true); break;
270         case "Khác": findViewById(R.id.btnOthers).setSelected(true); break;
271     }
272 }
273
274 //Hiển thị tất cả tasks không lọc
275 1 usage
276 private void showAllTasks() {
277     taskAdapter.updateOptions(new FirestoreRecyclerOptions.Builder<Task>()
278         .setQuery(db.collection(collectionPath "tasks").orderBy(field "taskDateTime"), Task.class)
279         .build());
280 }
281
282 //Lọc tasks theo danh mục
283 1 usage
284 private void filterTasksByCategory(String category) {
285     taskAdapter.updateOptions(new FirestoreRecyclerOptions.Builder<Task>()
286         .setQuery(db.collection(collectionPath "tasks").orderBy(field "taskDateTime"), Task.class)
287         .build());
288 }
289
290 @Override
291 protected void onActivityResult(int requestCode, int resultCode, Intent data) {
292     super.onActivityResult(requestCode, resultCode, data);
293     if (requestCode == 1 && resultCode == RESULT_OK && data != null) {
294         Task newTask = (Task) data.getSerializableExtra(name: "new_task");
295         if (newTask != null) {
296             db.collection(collectionPath "tasks").add(newTask)
297                 .addOnSuccessListener(documentReference -> updateCategoryCounts())
298                 .addOnFailureListener(exception e -> Log.e(tag: "MainActivity", msg: "Lỗi thêm task", e));
299         }
300     }
301 }

```

Hình 13: Code minh họa phân loại

KẾT LUẬN

1. Kết quả đạt được

Sau quá trình nghiên cứu và phát triển, nhóm đã hoàn thành một ứng dụng **Quản lý Công Việc - Todo List** với các tính năng chính như:

- Tạo, chỉnh sửa và xóa công việc theo từng danh mục: Sức khỏe, Công việc, Sức khỏe tinh thần và Khác.
- Thiết lập thời gian nhắc nhở công việc thông qua AlarmManager và Notification, giúp người dùng không bỏ lỡ các nhiệm vụ quan trọng.
- Giao diện lịch (Calendar View) hỗ trợ xem công việc theo ngày và tháng một cách trực quan.
- Đồng bộ dữ liệu với Firebase Realtime Database, đảm bảo lưu trữ và truy cập mọi lúc mọi nơi.
- Xác thực người dùng bằng Firebase Authentication, giúp bảo mật thông tin và phân quyền người dùng.
- Tính năng chia sẻ công việc, cho phép người dùng cộng tác trong quản lý nhiệm vụ.
- Thiết kế giao diện bằng XML theo phong cách Material Design, đơn giản, dễ sử dụng, phù hợp với nhiều đối tượng người dùng.

2. Nhược điểm

- Chưa hỗ trợ tính năng phân quyền chi tiết trong việc chia sẻ công việc (chỉ dừng ở mức chia sẻ cơ bản).
- Chưa có chức năng phân loại công việc theo trạng thái hoàn thành hoặc độ ưu tiên nâng cao.
- Chưa tích hợp tính năng thông báo lặp lại định kỳ (hàng ngày, hàng tuần...).
- Việc đồng bộ với Google Calendar chỉ mới triển khai ở mức đơn giản, cần hoàn thiện hơn.

3. Hướng phát triển

- Phát triển thêm tính năng lọc và sắp xếp công việc theo độ ưu tiên, trạng thái, thời gian hoàn thành.
- Bổ sung nhắc nhở định kỳ, hỗ trợ các công việc lặp lại như uống thuốc, học tập...
- Tối ưu hiệu năng ứng dụng, cải thiện tốc độ load dữ liệu từ Firebase.

- Thiết kế giao diện đa nền tảng, hướng tới khả năng chạy được trên iOS (Flutter, React Native...).
- Nâng cấp tính năng chia sẻ: phân quyền chi tiết (người xem, người chỉnh sửa).
- Thêm chế độ tối (Dark Mode) và tùy chỉnh giao diện người dùng.
- Hướng tới tích hợp AI để gợi ý sắp xếp lịch trình tối ưu.

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.