

**TRƯỜNG CAO ĐẲNG KIÊN GIANG  
KHOA CÔNG NGHỆ THÔNG TIN**



**BÁO CÁO ĐỒ ÁN  
MÔN XÂY DỰNG ỨNG DỤNG TRÊN ANDROID**

**ĐỀ TÀI:  
ỨNG DỤNG GIÚP NGƯỜI DÙNG QUẢN LÝ  
CÔNG VIỆC HÀNG NGÀY, ĐẶT LỊCH NHẮC NHỞ.**

**GIÁO VIÊN HƯỚNG DẪN:**

**NGUYỄN VŨ LÂM**

**NHÓM SINH VIÊN THỰC HIỆN:**

**NHÓM 01 – LỚP CNTT K17**

**TRẦN HÀO**

**KTH23CN213**

**LÊ TẤN KHA**

**KTH23CN217**

**ÂU TRIỀU TÂN**

**KTH23CN242**

**NGUYỄN YẾN NHI**

**KTH23CN226**

**Kiên Giang, ngày 22 tháng 05 năm 2025**



## LỜI MỞ ĐẦU

Trong guồng quay hối hả của cuộc sống hiện đại, việc quản lý hiệu quả công việc và thời gian đã trở thành một kỹ năng quan trọng giúp mỗi người duy trì sự cân bằng và nâng cao năng suất làm việc. Tuy nhiên, thực tế cho thấy nhiều cá nhân và tổ chức vẫn đang gặp khó khăn trong việc lập kế hoạch, theo dõi và hoàn thành công việc một cách khoa học, dẫn đến áp lực, căng thẳng và thường xuyên bỏ lỡ các mốc thời gian quan trọng. Mặc dù hiện nay có nhiều công cụ hỗ trợ, nhưng nhu cầu về một ứng dụng quản lý công việc tích hợp tính năng nhắc nhở trực quan, linh hoạt và dễ sử dụng trên nền tảng di động vẫn rất lớn.

Xuất phát từ thực tế đó, nhóm chúng em lựa chọn đề tài **"ỨNG DỤNG GIÚP NGƯỜI DÙNG QUẢN LÝ CÔNG VIỆC HÀNG NGÀY, ĐẶT LỊCH NHẮC NHỞ"** với mục tiêu xây dựng một công cụ hữu ích, hỗ trợ người dùng dễ dàng kiểm soát công việc mọi lúc, mọi nơi. Ứng dụng giúp lập kế hoạch, theo dõi tiến độ và gửi thông báo nhắc nhở kịp thời, từ đó giảm thiểu áp lực công việc, tăng khả năng tập trung và chủ động hơn trong tổ chức cuộc sống.

Về mặt ứng dụng thực tế, sản phẩm có thể được sử dụng rộng rãi cho nhiều đối tượng và trong nhiều môi trường khác nhau. Đối với cá nhân, ứng dụng là một trợ lý đắc lực trong việc quản lý lịch học, công việc, các hoạt động cá nhân hay gia đình. Đối với nhóm làm việc hay doanh nghiệp, nó hỗ trợ quản lý dự án, phân công nhiệm vụ, theo dõi tiến độ và cải thiện sự phối hợp giữa các thành viên. Trong bối cảnh làm việc từ xa và mô hình nhóm phân tán ngày càng phổ biến, ứng dụng này càng trở nên cần thiết để đảm bảo sự kết nối và nâng cao hiệu quả công việc chung.

## LỜI CẢM ƠN

Để hoàn thành bài báo cáo này, chúng em xin chân thành cảm ơn **Tiến sĩ Nguyễn Vũ Lâm – Giảng viên Khoa Công nghệ Thông tin, Trường Cao đẳng Kiên Giang** đã tận tình hướng dẫn, hỗ trợ và tạo điều kiện thuận lợi cho nhóm trong suốt quá trình thực hiện đề tài.

Những kiến thức mà thầy đã truyền đạt không chỉ giúp chúng em nắm vững kiến thức chuyên môn mà còn là nền tảng quý giá, là hành trang để chúng em tự tin hơn trên con đường trở thành những lập trình viên trong tương lai.

Mặc dù đã cố gắng hết sức trong quá trình học tập và thực hiện báo cáo, nhưng do thời gian và kinh nghiệm còn hạn chế, đề tài của nhóm khó tránh khỏi những thiếu sót. Chúng em rất mong nhận được sự đóng góp ý kiến và đánh giá của thầy để bài báo cáo được hoàn thiện hơn.

Cuối cùng, chúng em xin kính chúc thầy dồi dào sức khỏe, luôn tràn đầy nhiệt huyết và thành công trong sự nghiệp giảng dạy cao quý.

Chúng em xin chân thành cảm ơn!

*Rạch Giá, ngày 22 tháng 05 năm 2025*

**Sinh viên thực hiện**

**Trần Hào**

**Lê Tấn Kha**

**Âu Triều Tân**

**Nguyễn Yến Nhi**

**NHẬN XÉT VÀ ĐÁNH GIÁ**  
**(Giáo viên hướng dẫn)**

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

*Rạch Giá, ngày 22 tháng 05 năm 2025*

**Giáo viên hướng dẫn**  
**Nguyễn Vũ Lâm**

# MỤC LỤC

Catal og

<b>LỜI MỞ ĐẦU .....</b>	<b>3</b>
<b>LỜI CẢM ƠN .....</b>	<b>4</b>
<b>CHƯƠNG I: CƠ SỞ LÝ THUYẾT .....</b>	<b>8</b>
1.    Sơ lược về Android Studio .....	8
2.    Thiết lập môi trường .....	8
3.    Cấu trúc dự án Android Studio .....	9
3.1.    Tạo mới một project .....	9
3.2.    Các thành phần trong ứng dụng Android .....	11
<b>CHƯƠNG II: PHÂN TÍCH HỆ THỐNG .....</b>	<b>23</b>
1.    Mô tả hệ thống .....	23
2.    Các chức năng chính .....	23
3.    Phân tích hệ thống .....	24
3.1.    Thành phần giao diện người dùng (UI) .....	24
3.2.    Thành phần xử lý logic (Business Logic) .....	24
3.3.    Thành phần lưu trữ (Data) .....	24
3.4.    Thành phần hệ thống (System) .....	24
4.    Lưu đồ hoạt động (tóm tắt) .....	24
<b>CHƯƠNG III: THIẾT KẾ GIAO DIỆN VÀ CHỨC NĂNG .....</b>	<b>26</b>
1.    Giới thiệu Tổng quan Mục tiêu của thiết kế: .....	26
2.    Thiết kế Giao diện Người dùng (UI) Nguyên tắc thiết kế chung: .....	26
3.    Giao diện và chức năng của ứng dụng .....	26
3.1.    Đăng ký và đăng nhập Màn hình Đăng nhập (bên trái): .....	27
3.2.    Danh sách công việc .....	28
3.3.    Thêm công việc .....	29
3.4.    Xóa công việc .....	29
3.5.    Chỉnh sửa công việc .....	30
3.6.    Lọc công việc .....	32

3.7.	Sắp xếp công việc .....	33
3.8.	Thống kê công việc.....	36
4.	Đánh giá và hướng phát triển .....	37
4.1.	Mục tiêu kiểm thử.....	37
4.2.	Phương pháp kiểm thử.....	37
4.3.	Kịch bản kiểm thử.....	37
<b>CHƯƠNG VI: ĐÁNH GIÁ VÀ HƯỚNG PHÁT TRIỂN .....</b>		<b>39</b>
1.	Đánh giá ứng dụng viết báo cáo .....	39
1.1.	Ưu điểm .....	39
1.2.	Hạn chế / Nhược điểm.....	39
2.	Hướng phát triển cho ứng dụng.....	39
2.1.	Phát triển tính năng thông minh.....	39
2.2.	Nâng cao trải nghiệm người dùng .....	40
2.3.	Phát triển theo hướng giáo dục và doanh nghiệp.....	40
<b>CODE QUAN TRỌNG .....</b>		<b>41</b>
1.	Cấu trúc thư mục .....	41
2.	Code quan trọng.....	42
2.1.	AndroidManifest.xml .....	42
2.2.	TaskAdapter.....	44
2.3.	DatabaseHelper.....	45
2.4.	AlarmReceiver .....	49
2.5.	BroadcastReceiver .....	49
2.6.	AddEditTaskActivity .....	50
2.7.	LoginActivity .....	51
2.8.	MainActivity.....	52
2.9.	RegisterActivity.....	55
<b>TÀI LIỆU THAM KHẢO.....</b>		<b>57</b>
<b>PHỤ LỤC: DANH MỤC CÔNG VIỆC .....</b>		<b>58</b>

# CHƯƠNG I: CƠ SỞ LÝ THUYẾT

## 1. Sơ lược về Android Studio

Google cung cấp một công cụ phát triển ứng dụng Android chính thức trên website của mình, được xây dựng dựa trên nền tảng IntelliJ IDEA, gọi là Android Studio. Android Studio thừa hưởng toàn bộ những tính năng mạnh mẽ từ IntelliJ IDEA – một trong những IDE tốt nhất cho Java hiện nay.

Android Studio hỗ trợ người lập trình Android bằng nhiều công cụ tích hợp như trình giả lập (emulator), giao diện kéo thả để thiết kế layout, kiểm tra hiệu năng, trình quản lý SDK và các công cụ phân tích lỗi. Với những ưu điểm đó, Android Studio được xem là môi trường phát triển ứng dụng tối ưu cho các lập trình viên Android hiện nay.

## 2. Thiết lập môi trường

Để bắt đầu lập trình ứng dụng Android, có hai thành phần cơ bản bắt buộc phải có:

- Bộ phát triển Java (Java Development Kit - JDK): Đây là nền tảng để biên dịch và chạy các ứng dụng Java, đồng thời tạo ra môi trường thực thi máy ảo (JVM) cho hệ điều hành mà lập trình viên đang sử dụng.
- Bộ phát triển phần mềm Android (Android SDK - Software Development Kit): Gồm các thư viện, API, trình giả lập thiết bị, và công cụ dòng lệnh cần thiết cho việc phát triển, kiểm thử và triển khai ứng dụng Android.

Android SDK bao gồm nhiều phiên bản tương ứng với các phiên bản Android khác nhau. Mỗi khi Google phát hành phiên bản Android mới, Android SDK cũng sẽ được cập nhật để bổ sung các API, công cụ và tài nguyên phù hợp.

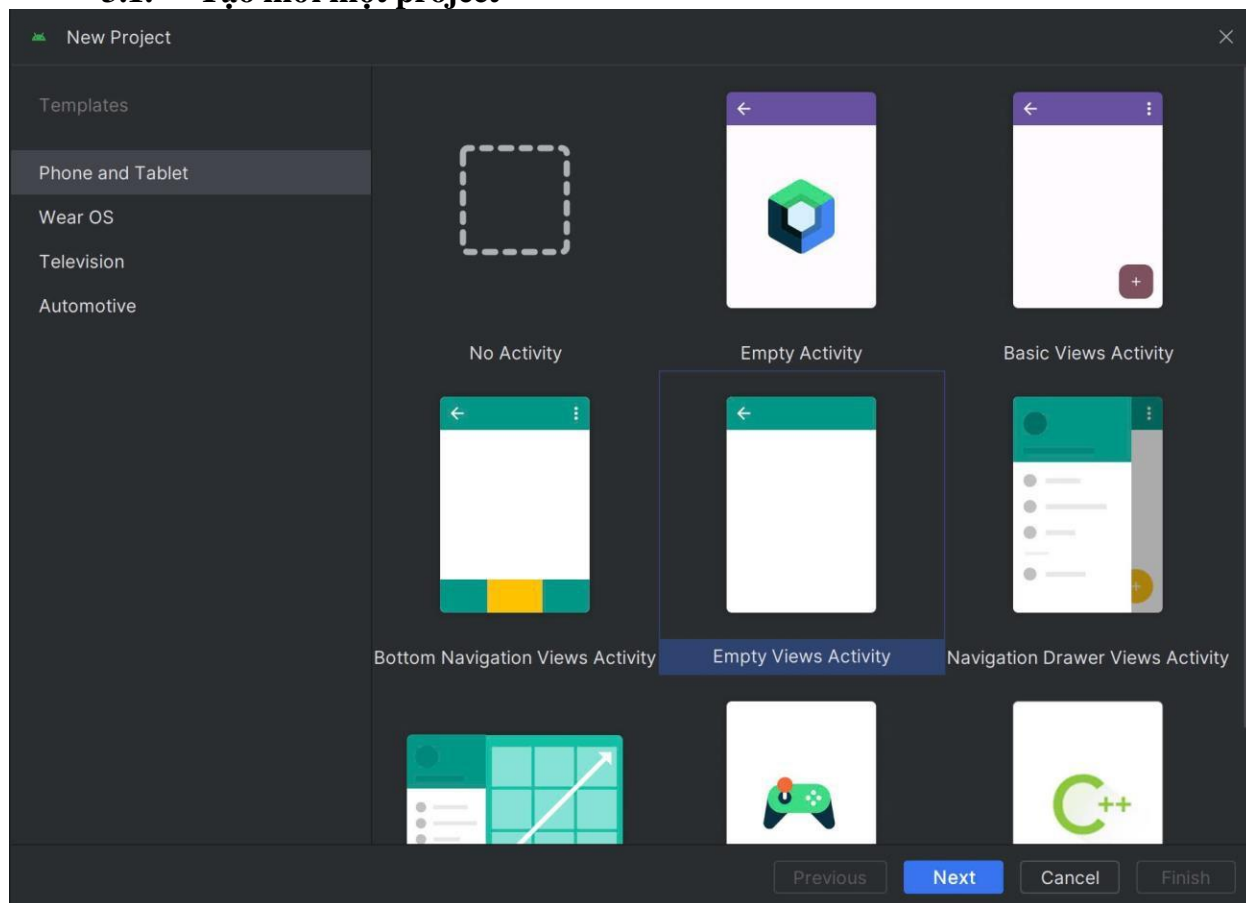
Ngoài ra, để hỗ trợ lập trình viên tối đa, Android Studio còn tích hợp các tiện ích như Gradle để quản lý thư viện và build project, hỗ trợ Git để quản lý mã nguồn, và khả năng tích hợp Firebase – một nền tảng mạnh mẽ hỗ trợ backend cho ứng dụng như cơ sở dữ liệu thời gian thực, xác thực người dùng, gửi thông báo...

Việc thiết lập đúng môi trường phát triển sẽ giúp quá trình lập trình diễn ra mượt mà, giảm thiểu lỗi và đảm bảo tương thích với các thiết bị Android hiện hành.



### 3. Cấu trúc dự án Android Studio

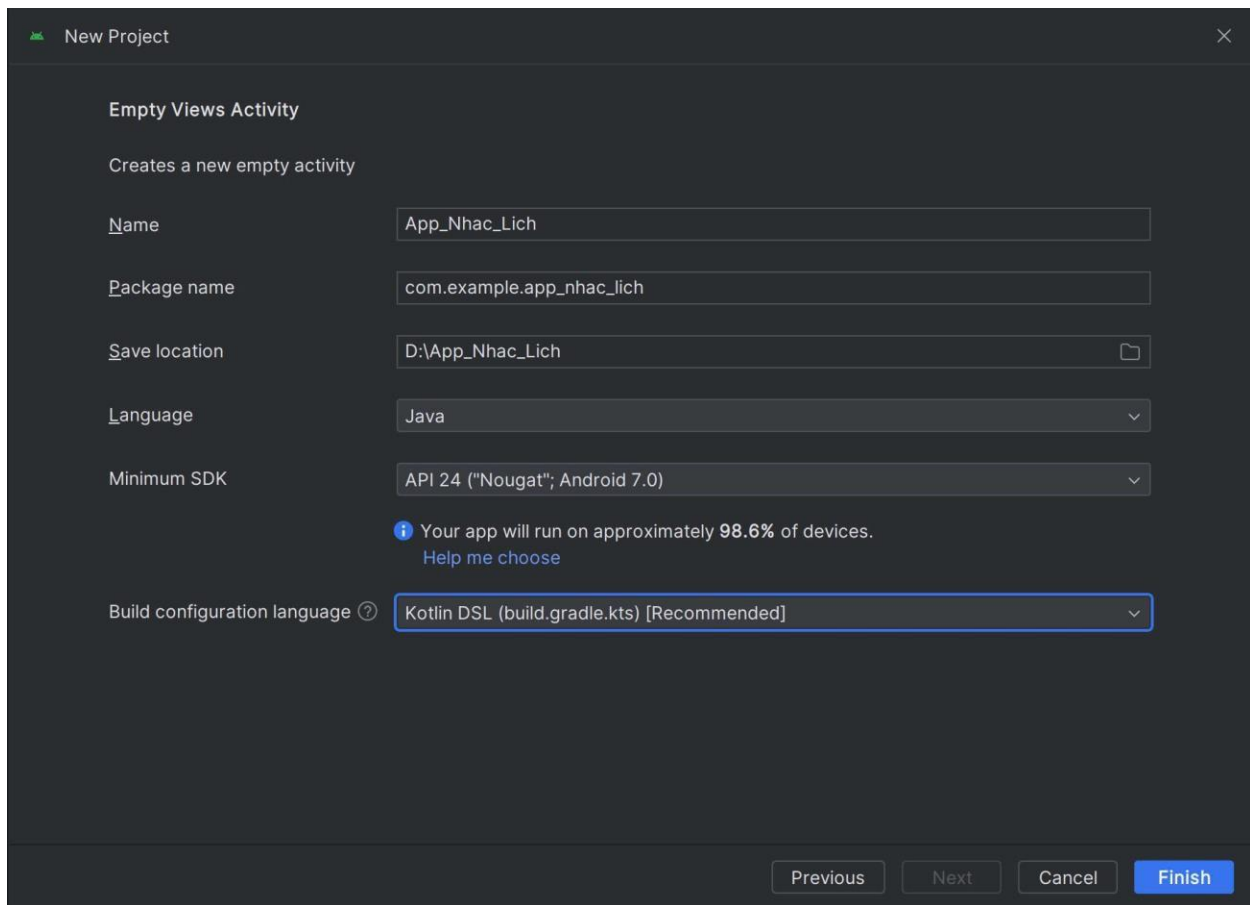
#### 3.1. Tạo mới một project



Hình 3.1.1: Chọn kiểu giao diện

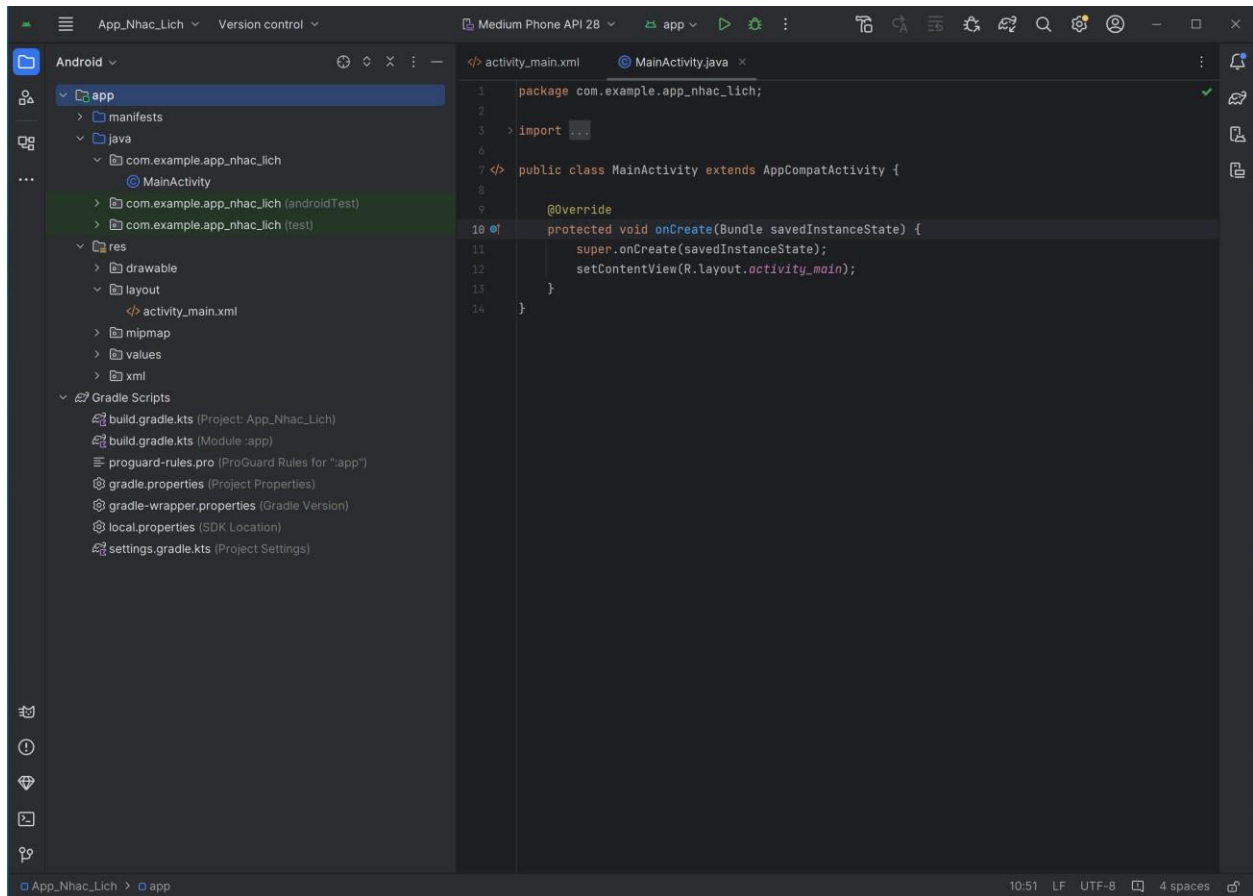
Ở hộp thoại trên cho phép ta lựa chọn là ứng dụng sẽ được viết cho những thiết bị nào (Phone and Tablet, TV, Wear).

Màn hình này hiển thị cho phép chọn loại Activity mặc định. Chọn Empty Views Activity rồi bấm Next.



Hình 3.1.2: Tạo dự án mới

- **Name:** Tên của ứng dụng bạn đặt. Android Studio sẽ dùng tên này để tạo thư mục project và tên file liên quan.
- **Package name:** Tên gói định danh duy nhất ứng dụng của bạn trên Google Play và hệ thống Android. Nó thường có dạng đảo ngược tên miền (com.example).
- **Save location:** Thư mục trong máy tính nơi mã nguồn của project sẽ được lưu.
- **Language:** Ngôn ngữ lập trình bạn chọn cho project. Ở đây là **Java**, có thể chọn Kotlin nếu muốn.
- **Minium SDK:** Phiên bản Android thấp nhất mà ứng dụng sẽ hỗ trợ. API 24 tương đương Android 7.0. Dòng nhỏ bên dưới thông báo: “Ứng dụng của bạn sẽ chạy trên khoảng 98.6% thiết bị”, tức là với mức API này, phần lớn điện thoại Android đều có thể cài đặt được.
- **Build configuration language:** Ngôn ngữ dùng để cấu hình Gradle (công cụ build project). Mặc dù bạn chọn Java làm ngôn ngữ lập trình, cấu hình build có thể dùng Kotlin DSL hoặc Groovy. Kotlin DSL được Google khuyến nghị vì rõ ràng và dễ kiểm soát hơn.
- Khi nhấn **Finish**, Android Studio sẽ tạo project mới với các thiết lập trên.



Hình 3.1.3: Giao diện phiên làm việc của android studio

### 3.2. Các thành phần trong ứng dụng Android

Thành phần ứng dụng là các khối cơ bản để xây dựng một ứng dụng Android. Các thành phần này được liên kết với các ứng dụng bởi tập tin AndroidManifest.xml, tập tin AndroidManifest.xml mô tả mỗi thành phần của ứng dụng và cách chúng tương tác với nhau. Các thành phần có thể được sử dụng trong một ứng dụng Android:

Thành phần	Đặc tả
Activity	Là một màn hình giao diện trong ứng dụng (giống như một trang). Mỗi Activity đại diện cho một tác vụ mà người dùng có thể thực hiện.
Service	Thành phần chạy ngầm, không có giao diện. Dùng để thực hiện các tác vụ nền như phát nhạc, tải dữ liệu, gửi thông báo, v.v.
Broadcast Receiver	Dùng để nhận thông báo (broadcast) từ hệ thống hoặc các ứng dụng khác.

Content Provider	Dùng để chia sẻ dữ liệu giữa các ứng dụng Android. Ví dụ: danh bạ, hình ảnh, video, v.v.
Manifest File	Khai báo các thành phần của ứng dụng như: Activity, Service, BroadcastReceiver,... Permission (quyền truy cập) Tên ứng dụng, icon, theme, v.v.
View	Bao gồm các thành phần như Button, TextView, EditText, ImageView, RecyclerView, v.v. Được khai báo trong file XML (thư mục res/layout/) hoặc tạo bằng code.
Database (SQLite / Room)	Dùng để lưu trữ dữ liệu cục bộ trong ứng dụng. SQLite là hệ quản trị cơ sở dữ liệu tích hợp trong Android. Room là thư viện mới hỗ trợ thao tác SQLite dễ hơn.
SharedPreferences	Lưu trữ dữ liệu đơn giản (dạng key-value), thường dùng để lưu thông tin đăng nhập, cài đặt.
Intent	Cung cấp khả năng giao tiếp giữa các màn hình với nhau.

*Bảng 1: Bảng thành phần của ứng dụng Android*

#### **a. Activity**

Trong ứng dụng Android, Activity đóng vai trò đặc biệt quan trọng, là nơi giúp người dùng tương tác trực tiếp với ứng dụng, ví dụ như gọi điện thoại, chụp ảnh, gửi e-mail hoặc xem bản đồ.

Activity được coi là xương sống của một ứng dụng Android, một ứng dụng có thể có một hoặc nhiều Activity (bất kỳ ứng dụng nào cũng cần có ít nhất 1 Activity).

Activity có thể hiển thị ở chế độ toàn màn hình, dạng cửa sổ hoặc với một kích thước nhất định

Một Activity có thể gọi đến một Activity khác, Activity được gọi đến sẽ tương tác với người dùng tại thời điểm được gọi tới.

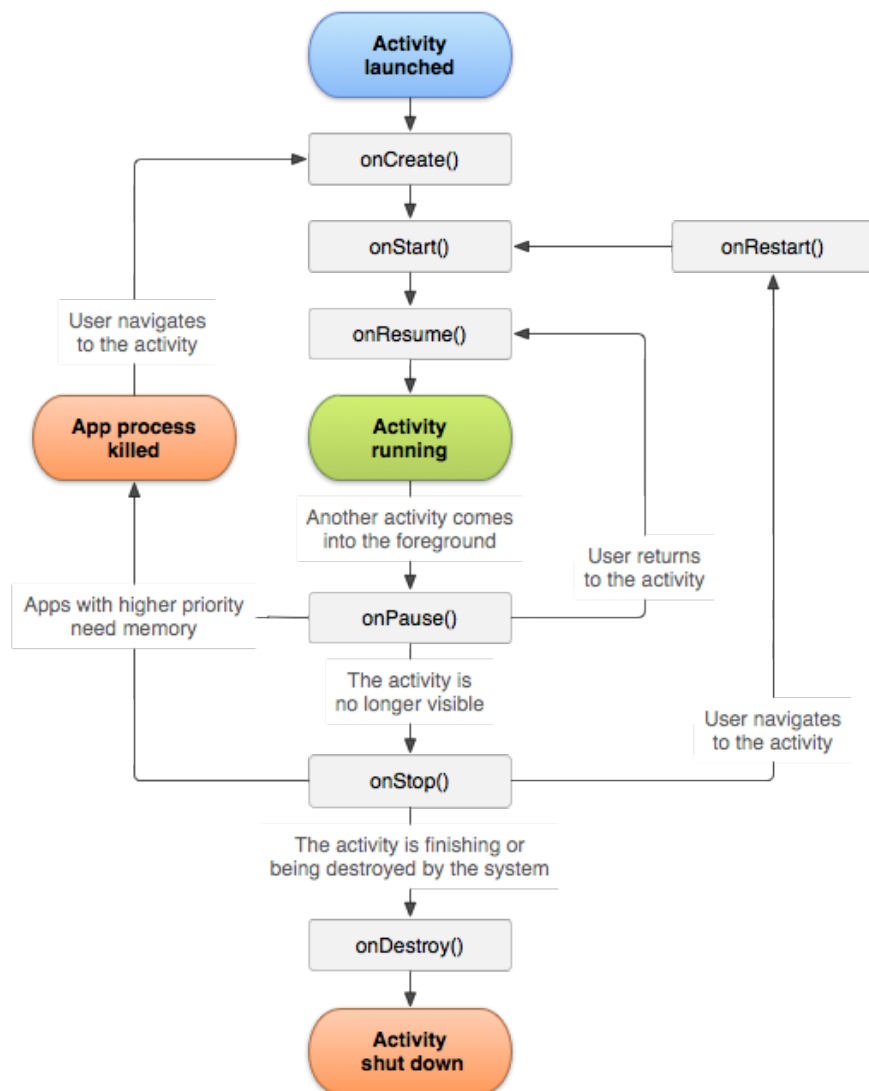
Một ứng dụng bên ngoài có thể gọi tới bất kỳ Activity nào trong ứng dụng (nếu được cấp quyền). Ví dụ: Một ứng dụng chụp ảnh sau khi chụp ảnh xong, sẽ gửi yêu cầu để start một activity có chức năng soạn e-mail trong ứng dụng email nhằm mục đích gửi ảnh vừa chụp đi.

Activity có một vòng đời được quản lý bởi hệ điều hành Android. Các phương

thức chính trong vòng đời bao gồm:

Phương thức	Mô tả
onCreate()	Được gọi khi Activity được tạo lần đầu.
onStart()	Activity sắp hiển thị ra màn hình.
onResume()	Activity sẵn sàng tương tác với người dùng.
onPause()	Activity chuẩn bị chuyển sang trạng thái nền.
onStop()	Activity không còn hiển thị với người dùng.
onDestroy()	Activity bị hủy hoàn toàn khỏi bộ nhớ.
onRestart()	Được gọi khi Activity đang ở trạng thái Stopped và sắp Start lại.

*Bảng 2: Các phương thức của Activity*



*Hình 3.2.1: Vòng đời của Activity*

Khai báo Activity trong AndroidManifest:

```
<activity  
  
    android:name="com.example.myapplication.MainActivity"  
  
    android:configChanges="orientation|screenSize"  
  
    android:exported="false"  
  
    android:screenOrientation="landscape">  
  
    <intent-filter>  
  
        <action android:name="android.intent.action.MAIN" />  
  
        <category android:name="android.intent.category.LAUNCHER" />  
  
    </intent-filter>  
  
</activity>
```

*Hình 3.2.2: Khai báo Activity trong AndroidManifest*

## **b. Service**

**Service** là một thành phần ứng dụng chạy ngầm trên hệ điều hành ví dụ như nghe nhạc, hoặc tương tác với một content provider. Service không tương tác trực tiếp với người dùng, khi service chạy thì người dùng vẫn có thể tương tác với một thành phần khác trong ứng dụng hoặc có thể tương tác với một ứng dụng khác trong hệ thống.

Ví dụ: Chúng ta có thể vừa nghe nhạc, vừa lướt facebook là do ứng dụng nghe nhạc có một service chạy ngầm trong background để phát nhạc trong khi người dùng đang tương tác với ứng dụng facebook.

Theo trang chủ android, Service trong Android được chia thành 3 loại đó là: **Foreground Service**, **Background Service** và **Bound Service**.

Khai báo Service trong AndroidManifest:

```

<service android:name=".ExampleService"

    android:enabled="true"

    android:exported="false"

    android:stopWithTask="true"/>

```

*Hình 3.2.3: Khai báo Service trong AndroidManifest*

### c. Broadcast Receiver

Broadcast Receiver là một thành phần của ứng dụng giúp lắng nghe các sự kiện mà hệ thống phát ra thông qua Intent, hệ thống có thể truyền phát ngay cả khi app không chạy. Broadcast Receiver không có giao diện cụ thể nhưng nó có thể thực hiện thông báo thông qua thanh Notification. Có rất nhiều broadcast được phát ra từ hệ thống, chúng ta có thể lấy ví dụ như một broadcast thông báo rằng màn hình điện thoại đã tắt, hay điện thoại đang ở trạng thái “Battery Low”, “Power Connected”, “Power Disconnected” hoặc một bức ảnh đã được chụp. Cũng có những broadcast được phát ra từ ứng dụng như sau khi download một tệp, ví dụ: Sau khi hoàn thành download một tệp tin, ứng dụng A phát ra thông báo là dữ liệu đã download xong, tệp đã sẵn sàng cho các ứng dụng khác có thể sử dụng.

Khai báo Broadcast Receiver trong AndroidManifest:

```

<receiver android:name=".BatteryLevelReceiver">

    <intent-filter>

        <action android:name="android.intent.action.BATTERY_LOW"/>

        <action android:name="android.intent.action.BATTERY_OKAY"/>

    </intent-filter>

</receiver>

```

*Hình 3.2.4: Khai báo Broadcast Receiver trong AndroidManifest*

#### d. Content Provider

Content Provider là một thành phần giúp các ứng dụng có thể đọc và ghi dữ liệu từ một file hoặc từ SQLite của một ứng dụng khác trong cùng một hệ thống. Bất kỳ ứng dụng nào có quyền (permission) đều có thể truy xuất, chỉnh sửa dữ liệu của một ứng dụng khác.

Content Provider được chia thành 2 loại:

- **Native Content Provider:** Là những Content Provider có sẵn, được tạo ra bởi hệ thống, ví dụ như Contacts, Message, ...
- **Custom Content Provider:** Bao gồm các Content Provider được tạo ra bởi các developer phụ thuộc vào đặc điểm của từng ứng dụng.

Khai báo Content Provider trong AndroidManifest:

```
<provider
    android:name="android.support.v4.content.FileProvider"
    android:authorities="com.example.codelearn.read.fileprovider"
    android:exported="false"
    android:grantUriPermissions="true">
    <meta-data
        android:name="android.support.FILE_PROVIDER_PATHS"
        android:resource="@xml/file_path"/>
</provider>
```

Hình 3.2.5: Khai báo Content Provider trong AndroidManifest:

#### e. Manifest File

AndroidManifest.xml là một **file cấu hình quan trọng** trong mọi ứng dụng Android. Nó cung cấp thông tin cần thiết cho hệ điều hành Android để chạy ứng dụng đúng cách.

##### Vai trò chính

- **Khai báo các thành phần** của ứng dụng:



- Activity, Service, BroadcastReceiver, ContentProvider
- **Cấp quyền truy cập hệ thống:**
  - Ví dụ: INTERNET, CAMERA, READ\_CONTACTS
- **Khai báo thông tin ứng dụng:**
  - Tên, icon, theme, version, min SDK...
- **Xác định Activity khởi chạy chính:**

```

<intent-filter>

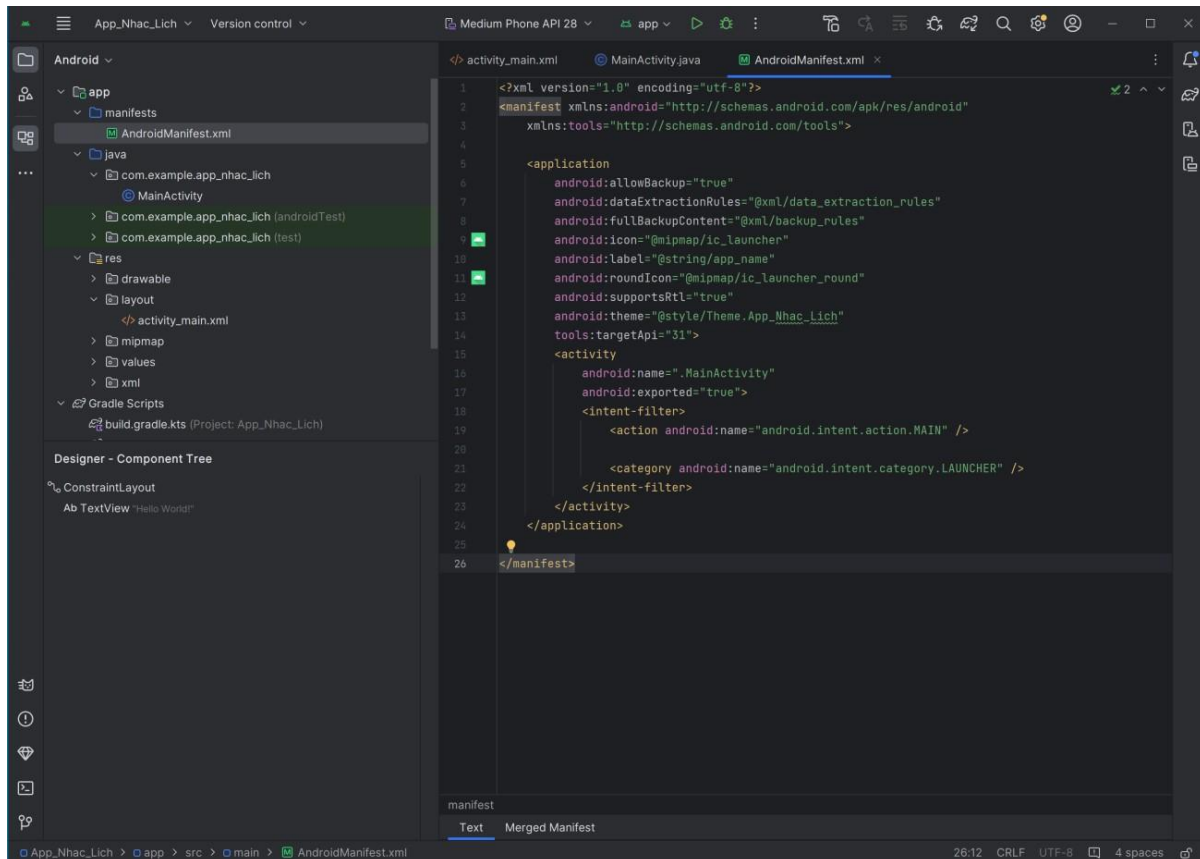
    <action android:name="android.intent.action.MAIN" />

    <category android:name="android.intent.category.LAUNCHER" />

</intent-filter>

```

### Cấu trúc cơ bản:



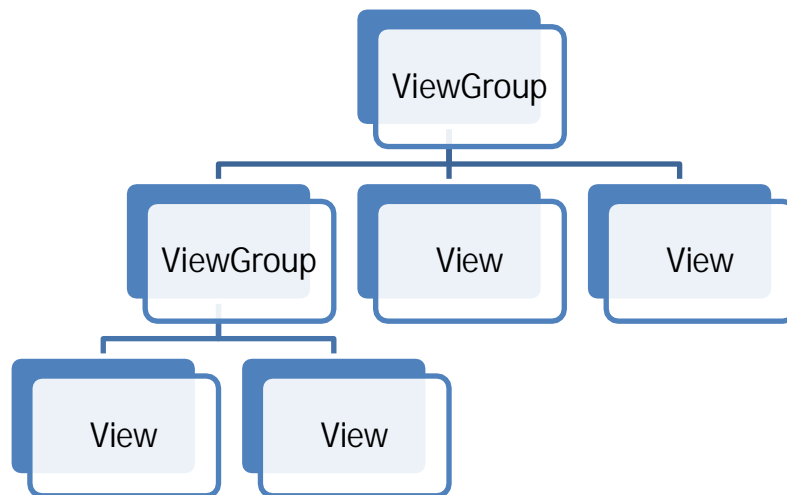
Hình 3.2.6: Cấu trúc cơ bản của Manifest

## f. View

Trong một ứng dụng Android, giao diện người dùng được xây dựng từ các đối tượng View và ViewGroup. Có nhiều kiểu View và ViewGroup. Mỗi một kiểu là một con của class View và tất cả các kiểu đó được gọi là các Widget.

Tất cả mọi widget đều có chung các thuộc tính cơ bản như là cách trình bày vị trí, background, kích thước, lề,... Tất cả những thuộc tính chung này được thể hiện hết ở trong đối tượng View.

Trong Android Platform, các screen luôn được bố trí theo một kiểu cấu trúc phân cấp như hình dưới. Một màn hình là một tập hợp các Layout và các widget được bố trí có thứ tự. Để thể hiện một màn hình thì trong hàm onCreate của mỗi Activity cần phải được gọi một hàm là setContentView(R.layout.main), hàm này sẽ load giao diện từ file XML lên để phân tích thành mã bytecode.



Hình 3.2.7: Mô hình ViewGroup

## g. Database SQLite

SQLite là một hệ quản trị cơ sở dữ liệu quan hệ (RDBMS) **nhẹ** và **nhúng (embedded)**, được thiết kế để sử dụng trực tiếp trong các ứng dụng. Không giống như các hệ cơ sở dữ liệu máy chủ như MySQL hay PostgreSQL, SQLite hoạt động hoàn toàn độc lập, không cần một tiến trình hoặc dịch vụ riêng biệt để vận hành. Mỗi cơ sở dữ liệu được lưu trữ trong một tệp đơn trên hệ thống tệp.

Trong Android, **SQLite được tích hợp sẵn trong nền tảng SDK**, cho phép lập trình viên lưu trữ dữ liệu cục bộ (local) một cách hiệu quả, không cần phụ thuộc vào dịch vụ bên ngoài hoặc kết nối Internet.

Trong môi trường Android, SQLite được sử dụng để:

- Lưu trữ dữ liệu người dùng, cấu hình, nội dung, cài đặt ứng dụng.
- Quản lý dữ liệu dạng bảng như danh bạ, ghi chú, sản phẩm, hóa đơn...
- Thay thế cho việc lưu trữ bằng file hoặc SharedPreferences khi cần cấu trúc dữ liệu phức tạp hơn.

SQLite là lựa chọn phù hợp cho các ứng dụng nhỏ đến trung bình, không cần cơ sở dữ liệu phân tán hay đồng bộ hóa trên máy chủ.

Trên Android, việc sử dụng SQLite thường thông qua lớp tiện ích SQLiteOpenHelper. Lớp này hỗ trợ:

- Tạo và cập nhật cơ sở dữ liệu khi nâng cấp phiên bản ứng dụng.
- Quản lý kết nối tới cơ sở dữ liệu.
- Hỗ trợ thực thi các câu truy vấn SQL.

## **h. SharedPreferences**

SharedPreferences là một cơ chế lưu trữ dữ liệu nhẹ (lightweight data storage) được Android cung cấp sẵn, cho phép lưu trữ **các cặp giá trị dạng khóa – giá trị (key-value)** vào bộ nhớ trong của ứng dụng. Cấu trúc lưu trữ này rất thích hợp để lưu các thông tin đơn giản như: tên người dùng, trạng thái đăng nhập, tùy chọn hiển thị, cấu hình ứng dụng, v.v. Dữ liệu được lưu dưới **dạng tệp XML** và chỉ có thể truy cập bởi chính ứng dụng đã lưu nó (trừ khi sử dụng chế độ chia sẻ có chủ đích).

SharedPreferences thường được sử dụng khi ứng dụng cần lưu một lượng nhỏ thông tin mà không cần đến cơ sở dữ liệu hoặc tệp tin phức tạp. Đặc biệt, nó thường được dùng để **"ghi nhớ" trạng thái đăng nhập, lưu cài đặt của người dùng, lưu chủ đề hiển thị, hoặc thiết lập ban đầu của ứng dụng.**

### **Cách sử dụng SharedPreferences**

Để sử dụng SharedPreferences, Android cung cấp các API cho phép truy xuất và chỉnh sửa dữ liệu như sau:

- **Truy cập đến SharedPreferences:**

```
SharedPreferences prefs = getSharedPreferences("MyPrefs", MODE_PRIVATE);
```

- **Ghi dữ liệu vào SharedPreferences:**

```
SharedPreferences.Editor editor = prefs.edit();  
editor.putString("username", "admin");  
editor.putBoolean("isLoggedIn", true);  
editor.apply(); // hoặc editor.commit();
```

- **Đọc dữ liệu từ SharedPreferences:**

```
String user = prefs.getString("username", "default");  
boolean loggedIn = prefs.getBoolean("isLoggedIn", false);
```

## **Quy trình sử dụng SharedPreferences**

Quy trình cơ bản gồm 3 bước chính:

- **Bước 1:** Truy cập đối tượng SharedPreferences với tên tùy chọn.
- **Bước 2:** Thực hiện ghi hoặc đọc dữ liệu bằng Editor hoặc các phương thức get.
- **Bước 3:** Áp dụng thay đổi với apply() (khuyến dùng, không đồng bộ) hoặc commit() (đồng bộ, có trả về kết quả boolean).

## **Lưu ý quan trọng khi sử dụng SharedPreferences**

- **Không nên dùng SharedPreferences để lưu dữ liệu lớn hoặc phức tạp như danh sách dài, ảnh, hoặc đối tượng dạng JSON nặng.**
- **Không lưu dữ liệu nhạy cảm (mật khẩu, token truy cập) nếu không có mã hóa bổ sung, vì dữ liệu có thể bị truy cập nếu thiết bị bị root.**
- Nên sử dụng apply() thay vì commit() để tránh chặn luồng UI, trừ khi cần kiểm tra kết quả lưu thành công.
- Dữ liệu lưu bằng SharedPreferences **sẽ bị mất nếu ứng dụng bị gỡ cài đặt hoặc người dùng xóa bộ nhớ ứng dụng.**

**Tóm lại,** SharedPreferences là một công cụ lưu trữ đơn giản, hiệu quả và rất hữu ích trong nhiều trường hợp cần lưu trữ dữ liệu nhẹ, không có cấu trúc phức tạp. Nắm vững cách sử dụng SharedPreferences giúp các lập trình viên xây dựng các chức năng như ghi nhớ trạng thái người dùng, cấu hình cá nhân, hoặc lưu thông tin nhanh một cách tiện lợi và an toàn trong môi trường Android.

## i. Intent

Content Provider được kích hoạt khi chúng được gọi từ một ContentResolver. Ba thành phần khác (hoạt động, dịch vụ và bộ nhận quảng bá) được kích hoạt bởi thông điệp không đồng bộ từ các Intent. Một Intent là một đối tượng có kiểu Intent chứa nội dung của thông điệp. Với các hoạt động và dịch vụ, nó gọi tên hành động được yêu cầu và xác định URI của dữ liệu tác động tới ở giữa. Ví dụ, nó có thể truyền tải một yêu cầu cho một hoạt động hiển thị một ảnh cho người dùng hay cho phép người dùng sửa văn bản. Với bộ nhận quảng bá, đối tượng Intent gọi tên của hành động được thông báo. Ví dụ, bộ nhận quảng bá có thể thông báo các phần nó quan tâm là nút chụp ảnh đã được bấm.

Có vài phương thức cho việc kích hoạt mỗi thành phần:

Một hoạt động được khởi chạy thông qua một đối tượng Intent Context.startActivity() hay Activity.startActivityForResult(). Hoạt động đáp lại có thể theo dõi Intent được tạo ra đó bằng phương thức getIntent() và cập nhật thông qua phương thức setIntent(Intent). Android gọi phương thức onNewIntent() để bỏ qua các Intent đến trước nó.

Một hoạt động thường bắt đầu hoạt động khác. Nếu nó muốn trả lại kết quả hoạt động nó đã khởi chạy, nó sẽ gọi phương thức startActivityForResult() thay cho startActivity(). Ví dụ, nếu nó khởi chạy một hoạt động cho phép người dùng lấy một ảnh, nó có thể muốn lấy kết quả của ảnh được chọn. Kết quả được trả về trong một đối tượng Intent thông qua phương thức onActivityResult().

Một dịch vụ được bắt đầu thông qua một đối tượng Intent là `Context.startService()` và Android gọi phương thức `onStart()` của dịch vụ và thông qua đối tượng Intent của nó.

Tương tự, một Intent có thể thông qua `Context.bindService()` để thiết lập một kết nối liên tục giữa các thành phần và dịch vụ đích. Dịch vụ nhận đối tượng Intent qua lời gọi `onBind()` (nếu dịch vụ chưa được chạy, `bindService()` có thể chọn bắt đầu nó). Cho ví dụ, một hoạt động có thể thiết lập kết nối với dịch vụ chơi nhạc đề cập ở phần trước để nó có thể cung cấp cho người dùng giao diện sử dụng để điều khiển chơi lại. Hoạt động sẽ gọi `bindService` để thiết lập kết nối và sau đó gọi phương thức đã định nghĩa bởi dịch vụ để áp dụng chơi lại ca khúc.

Một ứng dụng có thể khởi tạo một quảng bá thông qua đối tượng Intent bằng phương thức như: `Context.setBroadcast()`, `Context.setOrderedBroadcast()` và `Context.sendStickyBroadcast()`. Android chuyển những Intent tới tất cả các bộ nhận quảng bá nào quan tâm bằng việc gọi phương thức `onReceive()` của nó.

## CHƯƠNG II: PHÂN TÍCH HỆ THỐNG

### 1. Mô tả hệ thống

Ứng dụng hỗ trợ người dùng quản lý các công việc cá nhân hàng ngày bằng cách thêm, sửa, xoá và hiển thị danh sách công việc. Ngoài ra, người dùng có thể thiết lập lịch nhắc nhở để được thông báo đúng giờ. Ứng dụng cũng cung cấp tính năng **đăng nhập và đăng ký tài khoản**, nhằm bảo vệ dữ liệu và cho phép người dùng sử dụng ứng dụng một cách cá nhân hoá.

### 2. Các chức năng chính

- **Đăng ký tài khoản**
  - o Người dùng tạo tài khoản mới bằng email và mật khẩu.
  - o Dữ liệu được lưu vào SQLite Database.
- **Đăng nhập**
  - o Xác thực người dùng từ thông tin đã đăng ký.
  - o Dữ liệu người dùng được tải lên khi đăng nhập thành công.
- **Thêm công việc**
  - o Nhập tiêu đề, nội dung, thời gian, và có thể thiết lập nhắc nhở.
  - o Dữ liệu lưu trữ vào SQLite.
- **Hiển thị danh sách công việc**
  - o Dùng RecyclerView để hiển thị danh sách công việc theo ngày hoặc toàn bộ.
- **Chỉnh sửa / Xoá công việc**
  - o Cho phép cập nhật thông tin công việc hoặc xoá công việc không còn cần thiết.
- **Đặt nhắc nhở công việc**
  - o Sử dụng AlarmManager để tạo nhắc nhở đúng giờ đã đặt.
  - o Khi đến giờ, hiển thị thông báo cho người dùng thông qua Notification.
- **Lọc công việc**
  - o Hiển thị công việc theo trạng thái hoàn thành hoặc chưa hoàn thành.
- **Sắp xếp công việc**
  - o Theo tên (A-Z hoặc Z-A) hoặc theo ngày (tăng/giảm dần).

- **Thống kê**

- o Đếm số công việc hoàn thành, chưa hoàn thành, tổng số công việc.

### 3. Phân tích hệ thống

#### 3.1. Thành phần giao diện người dùng (UI)

- o **Activity/Fragment:** LoginActivity, RegisterActivity, MainActivity, AddTaskActivity, EditTaskActivity
- o **Các thành phần UI chính:** EditText, Button, RecyclerView, TimePicker, Notification

#### 3.2. Thành phần xử lý logic (Business Logic)

- o Xử lý đăng nhập, đăng ký
- o Thêm/sửa/xoá công việc
- o Lọc công việc
- o Sắp xếp công việc
- o Thông kê công việc
- o Thiết lập nhắc nhở
- o Kiểm tra hợp lệ dữ liệu (validate input)

#### 3.3. Thành phần lưu trữ (Data)

- o SQLite Database với hai bảng:
  - User: chứa thông tin người dùng (id, email, password)
  - Task: chứa công việc (id, title, description, datetime, user\_id)
- o Dùng SQLiteOpenHelper để thao tác với cơ sở dữ liệu

#### 3.4. Thành phần hệ thống (System)

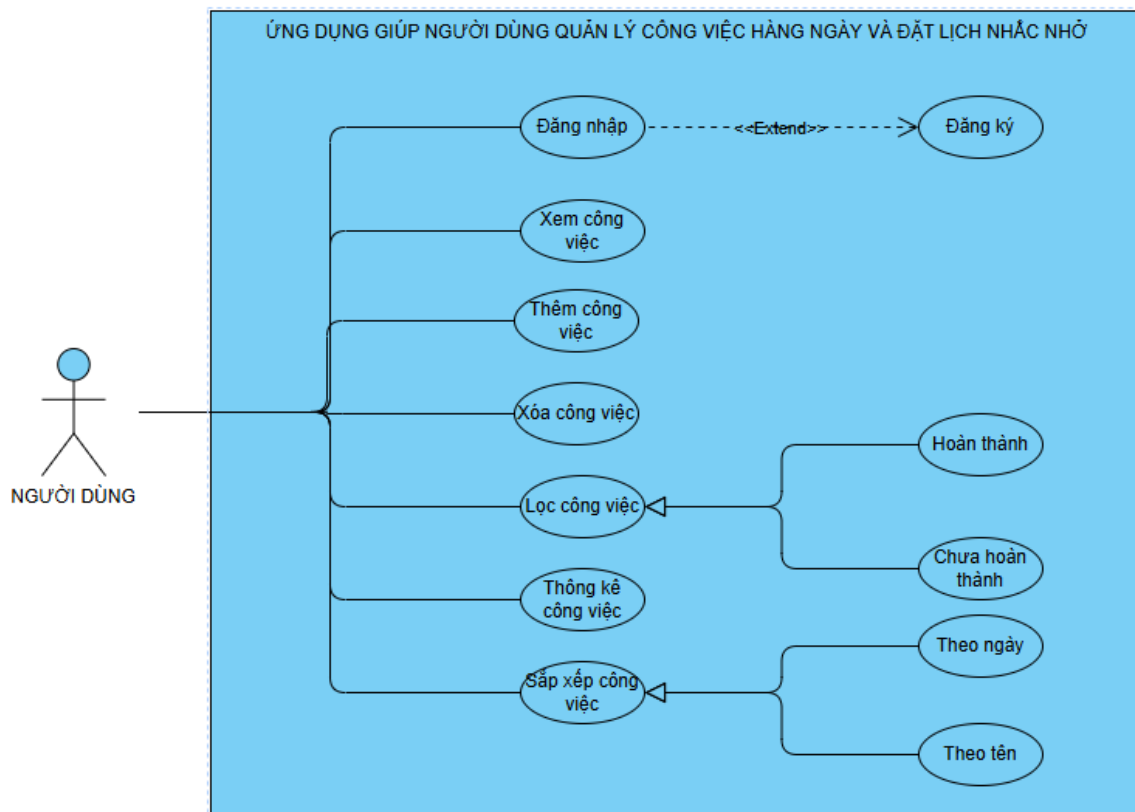
- o SharedPreferences: dùng để lưu trạng thái đăng nhập
- o AlarmManager + BroadcastReceiver: dùng để xử lý lịch nhắc nhở
- o NotificationManager: hiển thị thông báo nhắc công việc

### 4. Lưu đồ hoạt động (tóm tắt)

- Người dùng mở app:
  - Nếu đã đăng nhập → chuyển đến danh sách công việc
  - Nếu chưa → hiển thị màn hình đăng nhập
- Người dùng có thể:
  - Thêm công việc mới
  - Sửa hoặc xoá công việc
  - Thiết lập lịch nhắc nhở
  - Lọc/sắp xếp công việc.



- Xem thống kê tổng quan.
- Đến thời gian nhắc nhở:
  - AlarmManager kích hoạt BroadcastReceiver → Gửi Notification



Hình 4: Biểu đồ Use case

## **CHƯƠNG III: THIẾT KẾ GIAO DIỆN VÀ CHỨC NĂNG**

### **1. Giới thiệu Tổng quan**

#### **Mục tiêu của thiết kế:**

Cung cấp một nền tảng trực quan và dễ sử dụng để người dùng dễ dàng theo dõi, quản lý công việc và đặt lịch nhắc nhở.

Tối ưu hóa năng suất cá nhân bằng cách giảm thiểu việc bỏ lỡ deadline và tăng cường tính chủ động trong công việc hàng ngày.

Tạo trải nghiệm người dùng liền mạch và thân thiện, giúp họ cảm thấy kiểm soát được lịch trình của mình.

#### **Đối tượng người dùng:**

Cá nhân bận rộn: Sinh viên, nhân viên văn phòng, freelancer, hoặc bất kỳ ai muốn sắp xếp công việc và thời gian hiệu quả hơn.

Người dùng có nhu cầu nhắc nhở thường xuyên: Đặt lịch hẹn, uống thuốc, tập thể dục, v.v.

#### **Phạm vi:**

Báo cáo này tập trung vào thiết kế giao diện người dùng (UI) và các chức năng cốt lõi của ứng dụng, bao gồm quản lý công việc, tạo nhắc nhở, và các cài đặt liên quan.

### **2. Thiết kế Giao diện Người dùng (UI)**

#### **Nguyên tắc thiết kế chung:**

Đơn giản và gọn gàng: Ưu tiên không gian trống, giảm thiểu các yếu tố gây xao nhãng để người dùng tập trung vào công việc.

Trực quan và dễ học: Các biểu tượng và bố cục rõ ràng, giúp người dùng mới dễ dàng làm quen mà không cần hướng dẫn chi tiết.

Nhất quán: Duy trì cùng một phong cách thiết kế, font chữ, và hệ thống màu sắc trên toàn bộ ứng dụng để tạo sự chuyên nghiệp và dễ nhận diện.

Thân thiện với di động: Đảm bảo giao diện hiển thị tốt và dễ thao tác trên các kích thước màn hình điện thoại khác nhau.

### **3. Giao diện và chức năng của ứng dụng**

### 3.1. Đăng ký và đăng nhập

#### Màn hình Đăng nhập (bên trái):

- Nhập tên đăng nhập và mật khẩu
- Nút Đăng nhập
- Liên kết chuyển sang màn hình đăng ký

#### Màn hình Đăng ký (bên phải):

- Nhập tên đăng nhập, mật khẩu và xác nhận mật khẩu
- Nút Đăng ký
- Liên kết chuyển sang màn hình đăng nhập

The image displays two mobile application screens side-by-side. The left screen, titled "Quản lý công việc", is the login interface. It features a status bar at the top with the time 8:00, 4G+ signal, and 100% battery. Below the title bar, there is a large grey arrow pointing right. The main content area contains two input fields: "Tên đăng nhập" and "Mật khẩu", each with a toggle icon. A blue button labeled "ĐĂNG NHẬP" is positioned below the fields. At the bottom, a link reads "Chưa có tài khoản? Đăng ký ngay". The right screen, titled "Đăng ký", also has a status bar with 8:00, 4G+ signal, and 100% battery. It features a blue arrow pointing left. The main content area contains three input fields: "Tên đăng nhập", "Mật khẩu", and "Xác nhận mật khẩu", each with a toggle icon. A blue button labeled "ĐĂNG KÝ" is positioned below the fields. At the bottom, a link reads "Đã có tài khoản? Đăng nhập".

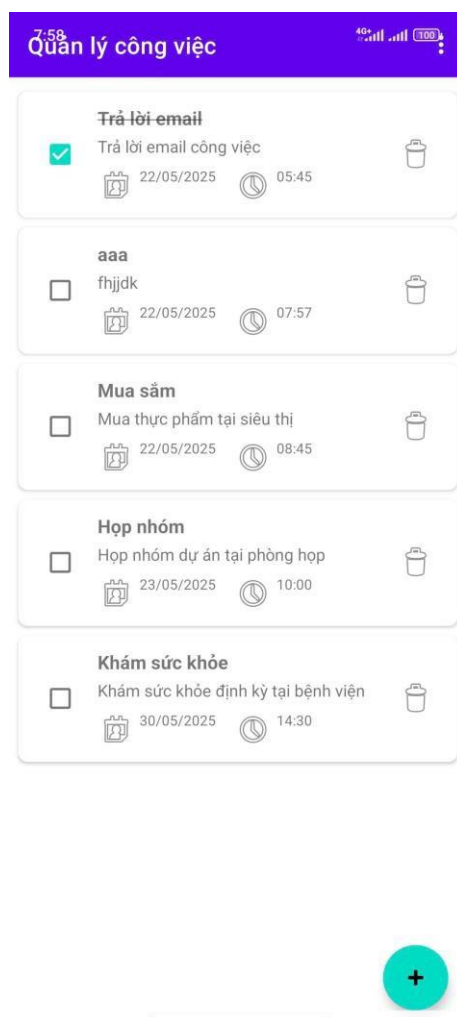
Hình 3.1.1: Giao diện đăng nhập và đăng ký

Ứng dụng có hai màn hình chính là **Đăng nhập** và **Đăng ký**. Ở màn hình Đăng nhập, người dùng nhập tên đăng nhập và mật khẩu để kiểm tra và truy cập vào ứng dụng.

Màn hình Đăng ký cho phép người dùng tạo tài khoản mới bằng cách nhập tên đăng nhập, mật khẩu và xác nhận lại mật khẩu. Cả hai màn hình đều có liên kết giúp chuyển qua lại dễ dàng giữa Đăng nhập và Đăng ký. Dữ liệu tài khoản được lưu trữ trong cơ sở dữ liệu để xác thực người dùng.

### 3.2. Danh sách công việc

Đây là giao diện chính của ứng dụng Quản lý công việc, nơi hiển thị danh sách các công việc mà người dùng đã thêm. Mỗi công việc được trình bày dưới dạng một ô với đầy đủ thông tin như tiêu đề, mô tả, ngày và giờ thực hiện. Người dùng có thể đánh dấu hoàn thành công việc bằng cách tick vào ô vuông bên trái, hoặc xóa công việc bằng biểu tượng thùng rác bên phải. Dưới góc phải màn hình có nút dấu cộng màu xanh để thêm công việc mới. Giao diện được thiết kế đơn giản, dễ sử dụng, giúp người dùng quản lý và theo dõi công việc hằng ngày một cách hiệu quả.



Hình 3.1.2: Giao diện trang chủ

### 3.3. Thêm công việc

Đây là màn hình để bạn **thêm một công việc mới** vào danh sách. Bạn chỉ cần nhập **tên công việc** và (nếu muốn) có thể thêm phần **mô tả** chi tiết. Sau đó, chọn **ngày** và **giờ thực hiện** bằng cách cuộn các cột. Nếu công việc đã hoàn thành, bạn có thể đánh dấu vào ô “**Đã hoàn thành**”. Cuối cùng, nhấn nút **LƯU** để lưu lại công việc. Thông tin sẽ được hiển thị trong danh sách công việc chính để bạn dễ dàng theo dõi và quản lý.

Hình 3.1.3: Giao diện thêm công việc

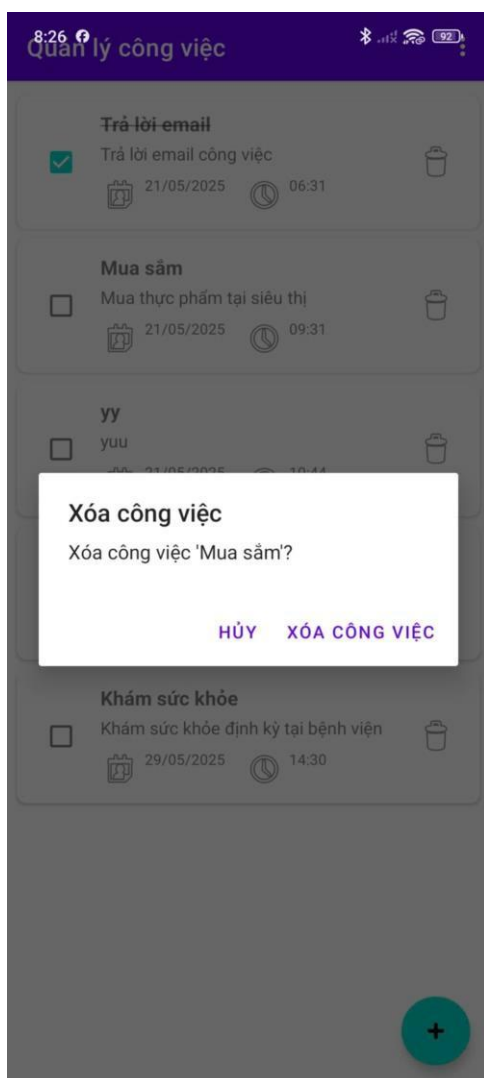
### 3.4. Xóa công việc

Đây là **hộp thoại xác nhận xóa công việc** trong ứng dụng. Khi người dùng nhấn vào biểu tượng **thùng rác** bên cạnh một công việc (ví dụ: "Mua sắm"), ứng dụng sẽ hiển thị hộp thoại hỏi lại để chắc chắn rằng bạn muốn xóa công việc đó.

Người dùng có thể chọn:

- **HỦY** nếu không muốn xóa nữa.
- **XÓA CÔNG VIỆC** để xác nhận và xóa công việc khỏi danh sách.

Chức năng này giúp tránh việc xóa nhầm công việc và đảm bảo người dùng có thể kiểm soát tốt hơn dữ liệu của mình.



Hình 3.1.4: Giao diện xóa công việc

### 3.5. Chỉnh sửa công việc

Đây là giao diện **sửa công việc** trong ứng dụng quản lý công việc. Người dùng có thể chỉnh sửa thông tin của một công việc đã tạo trước đó.

Cụ thể, người dùng có thể:

- Cập nhật **tên công việc**.
- Thay đổi hoặc thêm **mô tả chi tiết**.
- Chọn lại **ngày** và **giờ** thực hiện công việc.
- Đánh dấu vào ô "**Đã hoàn thành**" nếu công việc đã được thực hiện xong.
- Sau khi chỉnh sửa xong, nhấn nút **LƯU** để cập nhật lại thông tin.

Chức năng này giúp người dùng linh hoạt điều chỉnh công việc khi có thay đổi trong lịch trình.

The screenshot shows a mobile application interface for managing tasks. At the top, there's a status bar with the time 7:58 and signal/battery icons. Below it is a purple header bar with a back arrow and the text 'Quản lý công việc'. The main section is titled 'Sửa công việc' (Edit Task). It includes a 'Task ID:' field with the value '1'. There are two text input fields: 'Tên công việc' (Task Name) with the value 'Mua sắm' (Shopping) and 'Mô tả (tùy chọn)' (Description (optional)) with the value 'Mua thực phẩm tại siêu thị' (Buy food at the supermarket). Below these is a date picker labeled 'Ngày' (Date) showing a calendar grid with the date 21 selected for the month of April (thg 4) in the year 2024. Underneath is a time picker labeled 'Thời gian' (Time) showing a grid with the time 7:44 selected. At the bottom, there is a checkbox labeled 'Đã hoàn thành' (Completed) which is currently unchecked, and a large purple button labeled 'LƯU' (Save).

Hình 3.1.5: Giao diện chỉnh sửa công việc

### 3.6. Lọc công việc

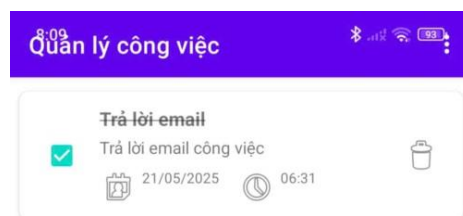
Chức năng **lọc công việc** trong ứng dụng giúp người dùng dễ dàng quản lý danh sách công việc bằng cách phân loại theo **trạng thái hoàn thành**.

Cụ thể, người dùng có thể chọn:

- **Hiển thị công việc chưa hoàn thành:** chỉ hiện các công việc đang chờ xử lý.
- **Hiển thị công việc đã hoàn thành:** chỉ hiện các công việc đã được đánh dấu là hoàn tất.
- **Hiển thị tất cả công việc:** bao gồm cả công việc đã và chưa hoàn thành.

Chức năng này giúp người dùng tập trung vào những việc cần làm trước, đồng thời dễ dàng theo dõi tiến độ công việc của mình.

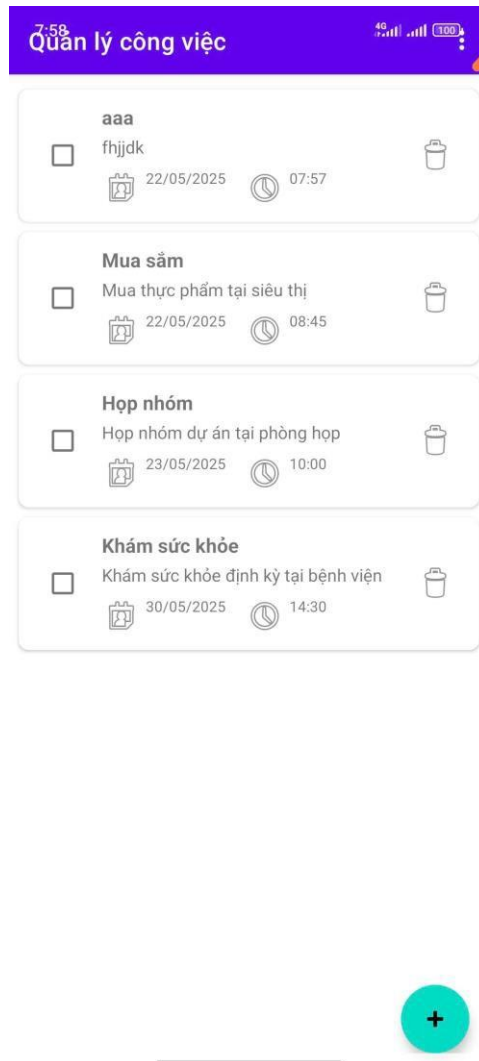
#### a. Hoàn thành





Hình 3.1.6: Sau khi đã lọc công việc hoàn thành

**b. Chưa hoàn thành**



Hình 3.1.7: Sau khi đã lọc công việc chưa hoàn thành

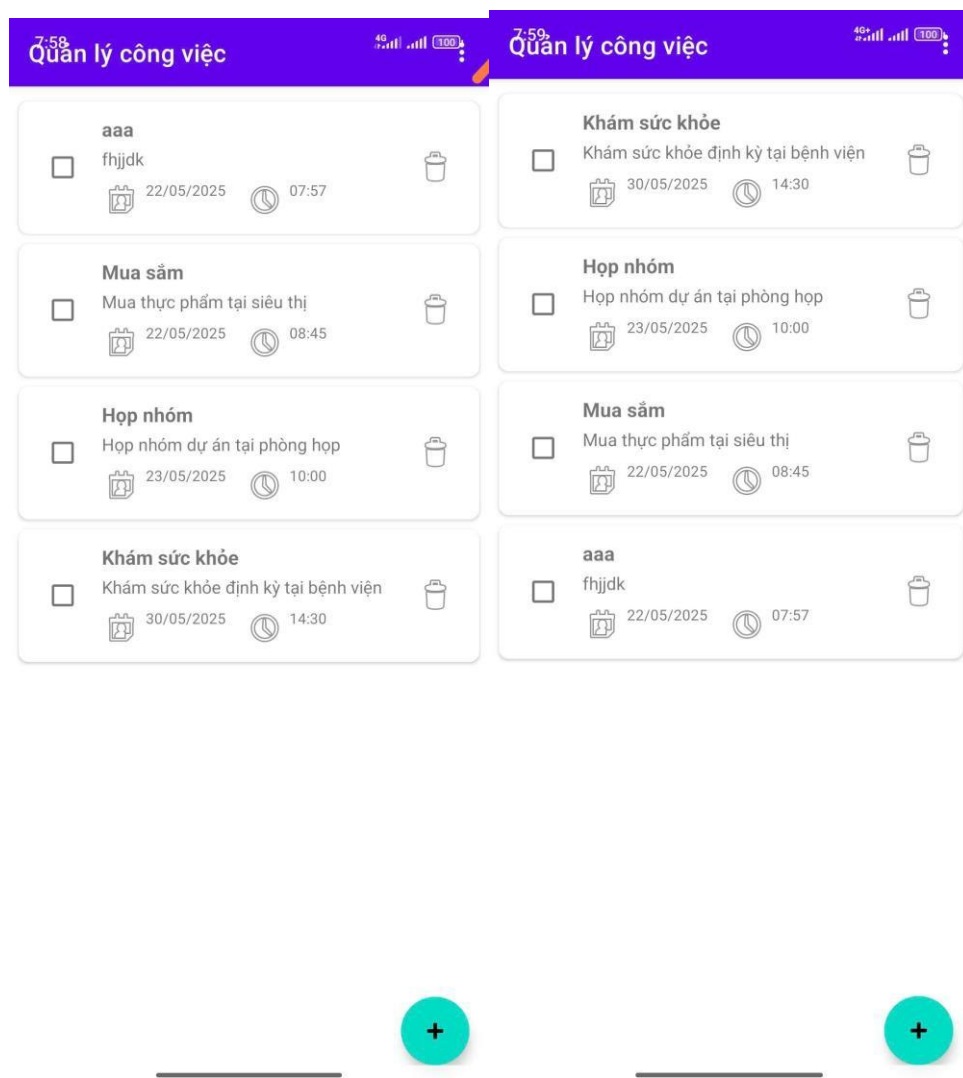
### 3.7. Sắp xếp công việc

Chức năng **sắp xếp công việc** trong ứng dụng giúp người dùng quản lý danh sách công việc một cách khoa học và tiện lợi hơn. Người dùng có thể lựa chọn sắp xếp theo:

- **Tên công việc:** Sắp xếp danh sách theo thứ tự bảng chữ cái (A–Z hoặc Z–A), giúp dễ dàng tìm kiếm theo tên.
- **Ngày thực hiện:** Sắp xếp công việc theo thời gian (từ sớm đến muộn hoặc ngược lại), giúp người dùng ưu tiên xử lý công việc theo đúng lịch trình.

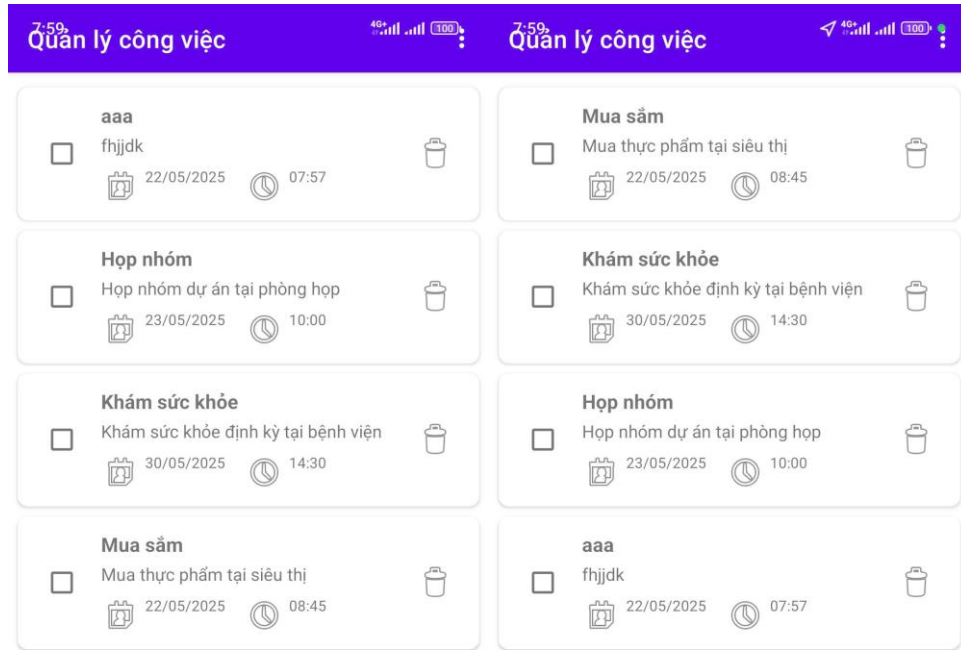
Việc sắp xếp này hỗ trợ người dùng tối ưu hóa quá trình làm việc và quản lý thời gian hiệu quả hơn.

**a. Theo ngày**



*Hình 3.1.8: Công việc được sắp xếp theo ngày*

## b. Theo tên



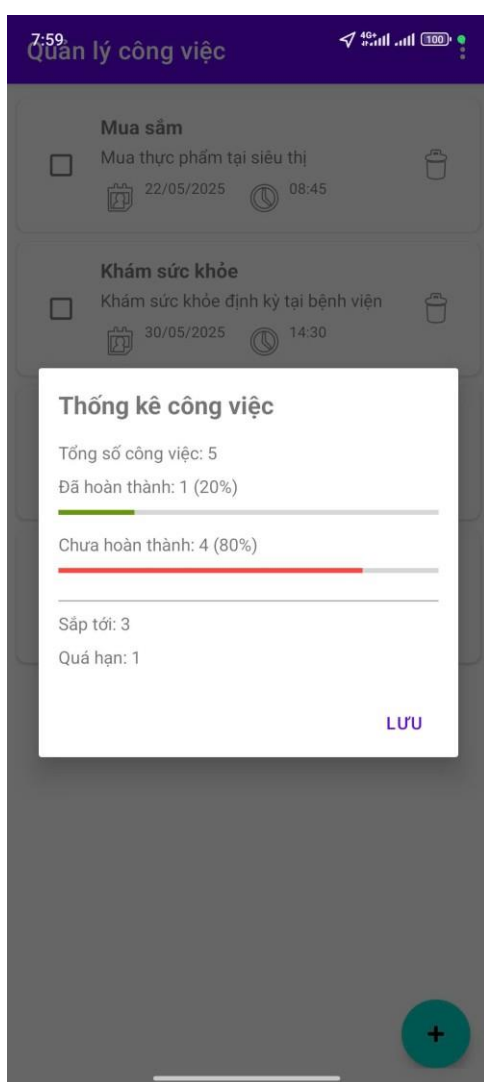
Hình 3.1.9: Công việc được sắp xếp theo tên

### 3.8. Thống kê công việc

**Chức năng thống kê công việc** giúp người dùng có cái nhìn tổng quan về quá trình quản lý công việc của mình. Giao diện này có thể hiển thị:

- **Tổng số công việc** đã tạo.
- **Số công việc đã hoàn thành** – giúp đánh giá mức độ hoàn thành mục tiêu.
- **Số công việc chưa hoàn thành** – nhắc nhở người dùng những việc còn dang dở.
- **Tỉ lệ hoàn thành (%)** – giúp người dùng theo dõi hiệu suất công việc của bản thân.

Nhờ chức năng thống kê, người dùng dễ dàng theo dõi tiến độ công việc và điều chỉnh kế hoạch làm việc hợp lý hơn.



Hình 3.1.10: Giao diện của thông kê công việc

## 4. Đánh giá và hướng phát triển

### 4.1. Mục tiêu kiểm thử

Mục tiêu của kiểm thử là đảm bảo rằng ứng dụng **quản lý công việc hằng ngày** hoạt động ổn định, chính xác và đáp ứng đúng yêu cầu đã đặt ra. Đồng thời, kiểm thử cũng giúp phát hiện và khắc phục các lỗi phát sinh trong quá trình sử dụng, cải thiện chất lượng phần mềm và nâng cao trải nghiệm người dùng.

### 4.2. Phương pháp kiểm thử

Sử dụng máy ảo và máy thật chạy hệ điều hành Android để kiểm tra các chức năng chính của ứng dụng, bao gồm:

- Đăng ký tài khoản
- Đăng nhập
- Thêm, sửa, xóa công việc
- Đặt lịch nhắc nhở
- Lọc và sắp xếp công việc
- Thống kê công việc

### 4.3. Kịch bản kiểm thử

STT	Chức năng	Dữ liệu kiểm thử	Kết quả mong đợi	Kết quả thực tế	Đạt/Không đạt
1	Đăng ký	Tài khoản mới, mật khẩu hợp lệ	Thông báo đăng ký thành công, chuyển trang	Đúng	Đạt
2	Đăng nhập	Tài khoản đúng	Đăng nhập thành công	Đúng	Đạt
3	Thêm công việc	Nhập đầy đủ tên, ngày, trạng thái	Công việc hiển thị trong danh sách	Đúng	Đạt
4	Sửa công việc	Chỉnh sửa tên hoặc ngày	Công việc được cập nhật lại	Đúng	Đạt
5	Xóa công việc	Nhấn xóa một công việc trong danh sách	Công việc bị xóa khỏi danh sách	Đúng	Đạt
6	Đặt lịch nhắc nhở	Chọn ngày giờ nhắc	Thông báo hiện đúng thời gian	Đúng	Đạt
7	Lọc theo trạng thái	Chọn “Hoàn thành” hoặc	Danh sách lọc chính xác theo	Đúng	Đạt

		“Chưa hoàn thành”	trạng thái		
8	Sắp xếp theo tên/ngày	Chọn tiêu chí sắp xếp	Danh sách sắp xếp đúng thứ tự	Đúng	Đạt
9	Thông kê công việc	Gồm tổng số công việc, hoàn thành, chưa xong	Thông kê đúng số lượng	Đúng	Đạt

o **Kết quả và nhận xét**

Sau khi tiến hành kiểm thử các chức năng chính, ứng dụng **hoạt động ổn định, không phát sinh lỗi nghiêm trọng**. Các tính năng như thêm công việc, nhắc nhở, lọc và sắp xếp được xử lý nhanh chóng, chính xác. Thông báo nhắc công việc hoạt động đúng thời gian, giúp người dùng không bỏ lỡ các công việc quan trọng.

Về giao diện, ứng dụng có bố cục hợp lý, dễ sử dụng. Các thao tác đơn giản, phù hợp với mọi đối tượng người dùng. Thời gian phản hồi của ứng dụng nhanh, không gây giật lag hay thoát đột ngột.

o **Kết luận kiểm thử**

Thông qua quá trình kiểm thử, có thể kết luận rằng ứng dụng **quản lý công việc hằng ngày** đã đáp ứng đầy đủ các yêu cầu chức năng đặt ra ban đầu. Ứng dụng có thể triển khai sử dụng thực tế hoặc tiếp tục phát triển thêm các tính năng nâng cao như đồng bộ dữ liệu, chia sẻ công việc, phân loại công việc theo nhóm, v.v.

## CHƯƠNG VI: ĐÁNH GIÁ VÀ HƯỚNG PHÁT TRIỂN

### 1. Đánh giá ứng dụng viết báo cáo

#### 1.1. Ưu điểm

- **Giao diện thân thiện, dễ sử dụng:** Người dùng có thể dễ dàng tiếp cận và thao tác viết báo cáo mà không cần học quá nhiều. Các biểu tượng, menu, chức năng được bố trí rõ ràng.
- **Chức năng soạn thảo cơ bản đầy đủ:** Có thể định dạng chữ (in đậm, nghiêng, gạch chân), căn lề, chèn hình ảnh, bảng biểu – đáp ứng được yêu cầu của phần lớn báo cáo học tập hoặc công việc.
- **Lưu trữ cục bộ hoặc trực tuyến:** Tùy vào thiết kế, người dùng có thể lưu trữ trên thiết bị hoặc đồng bộ với Google Drive, OneDrive, v.v. Giúp bảo vệ dữ liệu và truy cập mọi lúc mọi nơi.
- **Tích hợp mẫu báo cáo:** Cung cấp sẵn các template như: báo cáo thực tập, báo cáo tiến độ, báo cáo tài chính, giúp người dùng tiết kiệm thời gian khởi tạo.
- **Hỗ trợ xuất file:** Có thể xuất ra PDF hoặc Word, phục vụ cho in ấn hoặc nộp online.

#### 1.2. Hạn chế / Nhược điểm

- **Chưa hỗ trợ làm việc nhóm:** Trong môi trường học nhóm hoặc công việc, việc không thể cùng chỉnh sửa khiến thao tác cộng tác gặp khó khăn.
- **Thiếu chức năng kiểm tra lỗi:** Không có tính năng kiểm tra chính tả hoặc ngữ pháp, dễ dẫn đến lỗi trong báo cáo chính thức.
- **Không hỗ trợ dữ liệu phức tạp:** Chưa có khả năng xử lý số liệu, vẽ biểu đồ tự động từ bảng tính hoặc thống kê.
- **Không có chức năng gợi ý nội dung thông minh:** Người dùng phải viết hoàn toàn thủ công, mất thời gian và dễ bí ý tưởng.
- **Chưa tối ưu hóa trải nghiệm người dùng trên di động:** Nếu ứng dụng chưa có phiên bản mobile hoặc chưa tối ưu, việc viết báo cáo trên điện thoại sẽ gặp hạn chế.

### 2. Hướng phát triển cho ứng dụng

#### 2.1. Phát triển tính năng thông minh

- **Tích hợp AI hỗ trợ nội dung:**
  - Gợi ý bố cục báo cáo phù hợp với chủ đề.
  - Tự động sinh tiêu đề, phần mở đầu, kết luận dựa trên nội dung đã viết.
  - Gợi ý trích dẫn, tài liệu tham khảo.
- **Kiểm tra lỗi ngữ pháp, chính tả đa ngôn ngữ:** Đảm bảo nội dung chuẩn mực, chuyên nghiệp hơn.
- **Chèn biểu đồ và phân tích số liệu thông minh:** Nhập dữ liệu và ứng dụng tự động tạo biểu đồ cột, đường, tròn theo chuẩn báo cáo.

## 2.2. Nâng cao trải nghiệm người dùng

- **Hỗ trợ đa nền tảng:** Phát triển bản web, Android, iOS, và cả ứng dụng desktop (Windows/Mac).
- **Đồng bộ hóa đám mây:** Tự động lưu nội dung, đồng bộ với tài khoản Google, Microsoft hoặc Dropbox để tránh mất dữ liệu.
- **Tự động lưu và khôi phục phiên bản trước:** Cho phép khôi phục lại các phiên bản trước đó nếu người dùng chỉnh sửa nhầm.
- **Hỗ trợ làm việc nhóm:** Tạo nhóm làm việc, chia sẻ quyền chỉnh sửa/bình luận, gán thẻ thành viên, phân chia mục cần viết.
- **Tạo báo cáo theo mẫu có sẵn:** Cho phép chọn mẫu báo cáo theo từng ngành nghề, lĩnh vực (CNTT, giáo dục, kinh doanh, y tế...).

## 2.3. Phát triển theo hướng giáo dục và doanh nghiệp

- **Dành cho sinh viên:** Cung cấp tài nguyên như hướng dẫn viết báo cáo, bài mẫu, phân tích điểm mạnh/yếu của bài viết.
- **Dành cho doanh nghiệp:** Hỗ trợ báo cáo tiến độ, báo cáo doanh thu, xuất bảng tổng hợp từ file Excel.
- **Tích hợp lịch và nhắc nhở deadline:** Giúp người dùng theo dõi tiến độ viết và nhắc hạn nộp.



## CODE QUAN TRỌNG

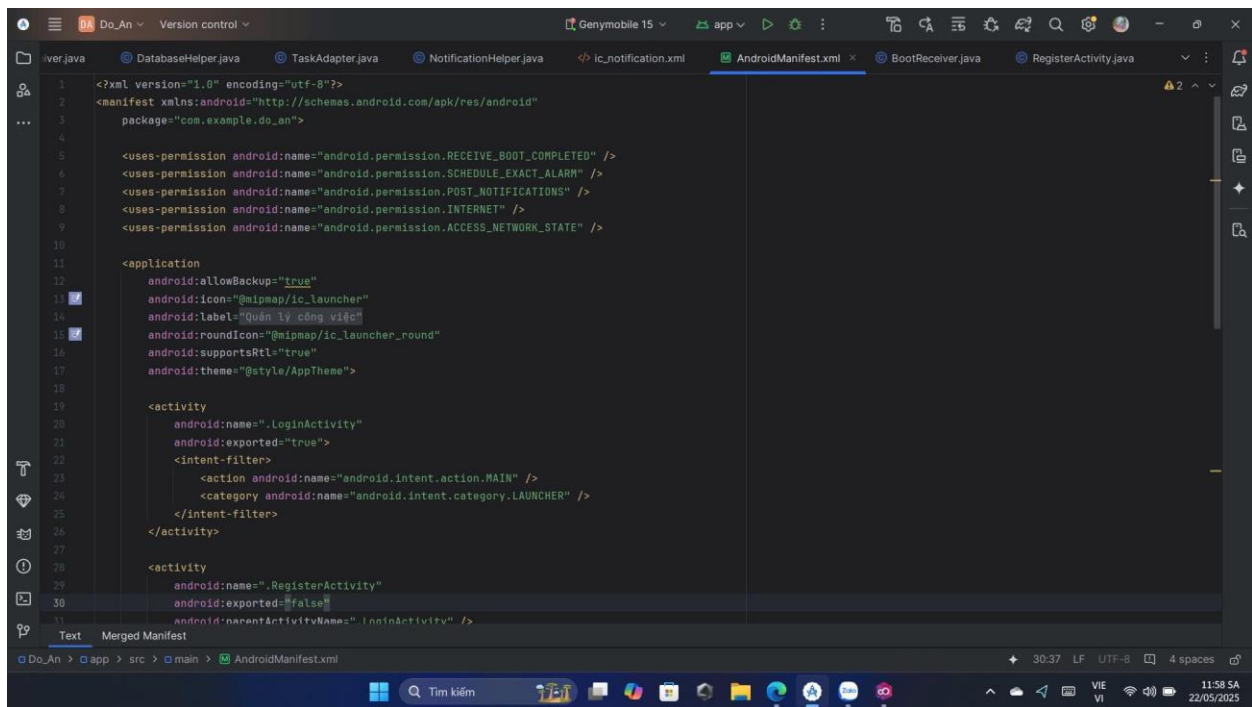
### 1. Cấu trúc thư mục

- ❖ manifests
  - AndroidManifest.xml
- ❖ java
  - com.example.do\_an
    - ◆ adapter
      - TaskAdapter
    - ◆ database
      - DatabaseHelper
    - ◆ model
      - Task
      - User
    - ◆ receiver
      - AlarmReceiver
      - BootReceiver
    - ◆ util
      - AddEditTaskActivity
      - CloudSyncActivity
      - LoginActivity
      - MainActivity
      - RegisterActivity
- ❖ Layout
  - </> activity\_add\_edit\_task.xml
  - </> activity\_cloud\_sync.xml
  - </> activity\_login.xml
  - </> activity\_main.xml
  - </> activity\_register.xml
  - </> dialog\_statistics.xml
  - </> item\_task.xml

## 2. Code quan trọng

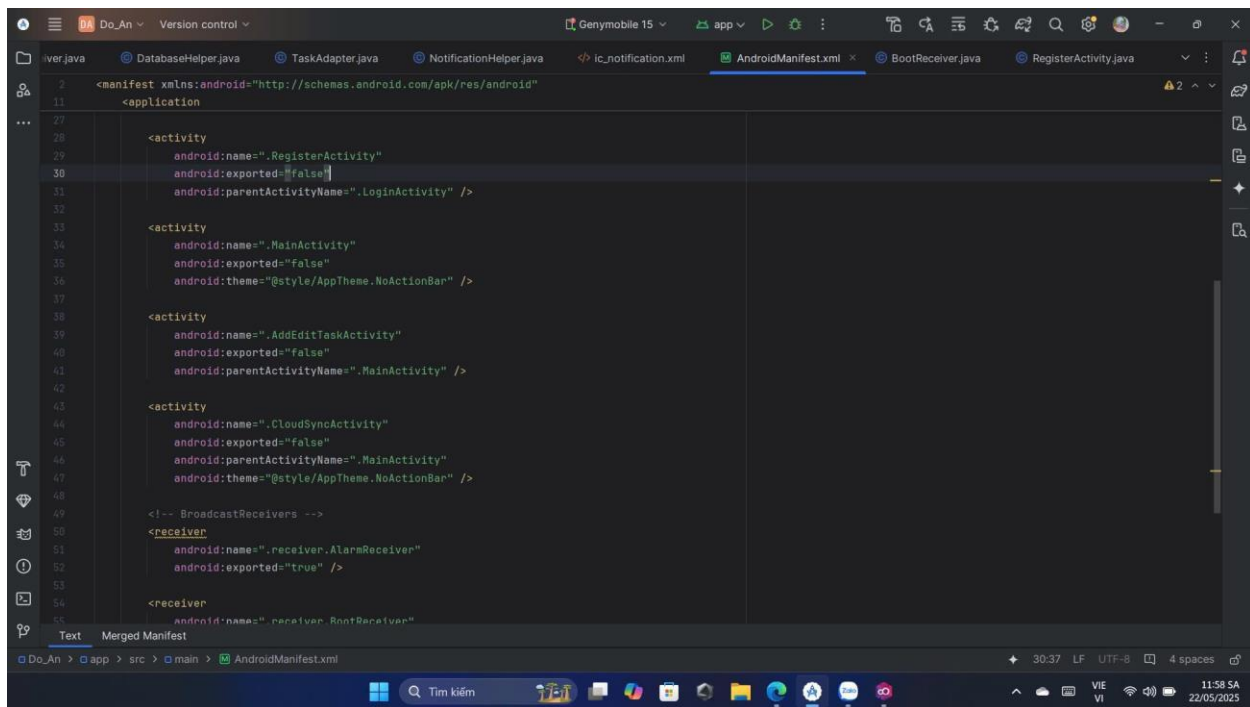
### 2.1. AndroidManifest.xml

File AndroidManifest.xml là nơi cấu hình tổng thể cho ứng dụng Android. Không có file này, ứng dụng không thể chạy được. Là nơi cấu hình để các Activity chuyển đổi qua lại với nhau.

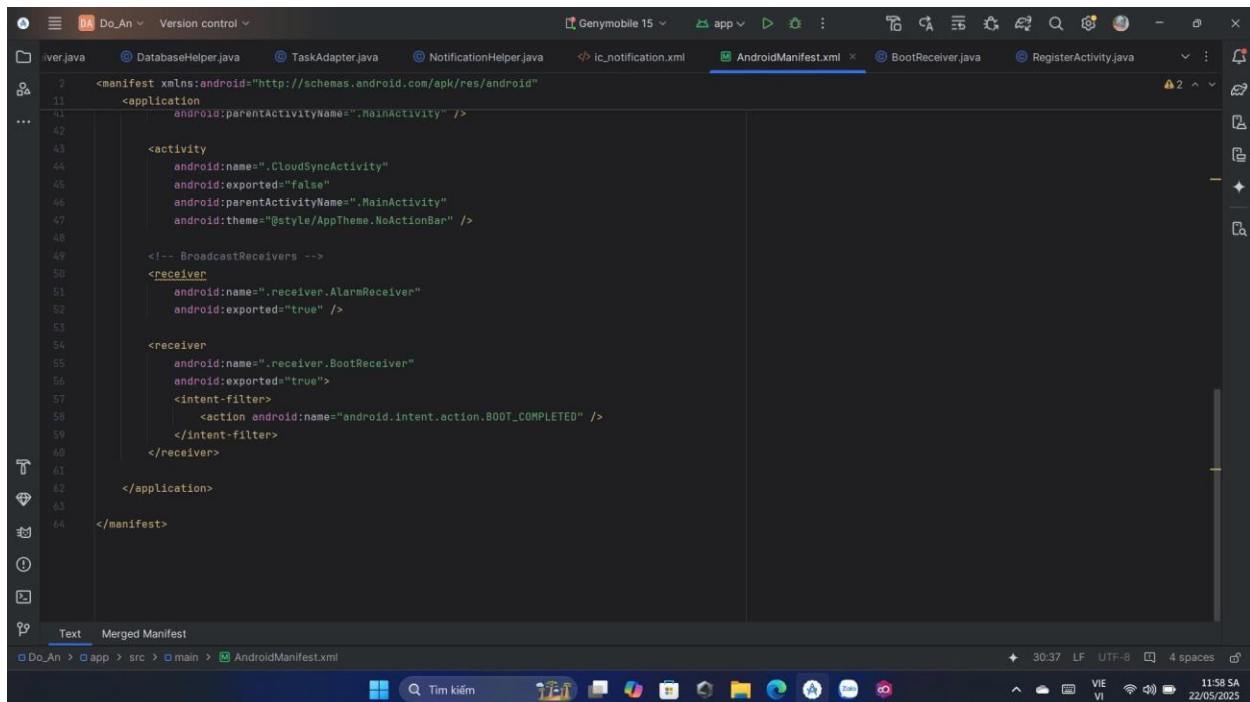


```
1 <?xml version="1.0" encoding="utf-8"?>
2 <manifest xmlns:android="http://schemas.android.com/apk/res/android"
3     package="com.example.do_an">
4
5     <uses-permission android:name="android.permission.RECEIVE_BOOT_COMPLETED" />
6     <uses-permission android:name="android.permission.SCHEDULE_EXACT_ALARM" />
7     <uses-permission android:name="android.permission.POST_NOTIFICATIONS" />
8     <uses-permission android:name="android.permission.INTERNET" />
9     <uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />
10
11     <application
12         android:allowBackup="true"
13         android:icon="@mipmap/ic_launcher"
14         android:label="@string/app_name"
15         android:roundIcon="@mipmap/ic_launcher_round"
16         android:supportRtl="true"
17         android:theme="@style/AppTheme">
18
19         <activity
20             android:name=".LoginActivity"
21             android:exported="true">
22             <intent-filter>
23                 <action android:name="android.intent.action.MAIN" />
24                 <category android:name="android.intent.category.LAUNCHER" />
25             </intent-filter>
26         </activity>
27
28         <activity
29             android:name=".RegisterActivity"
30             android:exported="false">
31             <intent-filter>
32                 <action android:name="android.intent.action.VIEW" />
33                 <category android:name="android.intent.category.BROWSABLE" />
34                 <data android:scheme="http" />
35             </intent-filter>
36         </activity>
37     </application>
38 </manifest>
```

Hình 2.1.1: Code File AndroidManifestb đoạn 1



Hình 2.1.2: Code File AndroidManifestb đoạn 2



Hình 2.1.3: Code File AndroidManifestb đoạn 3

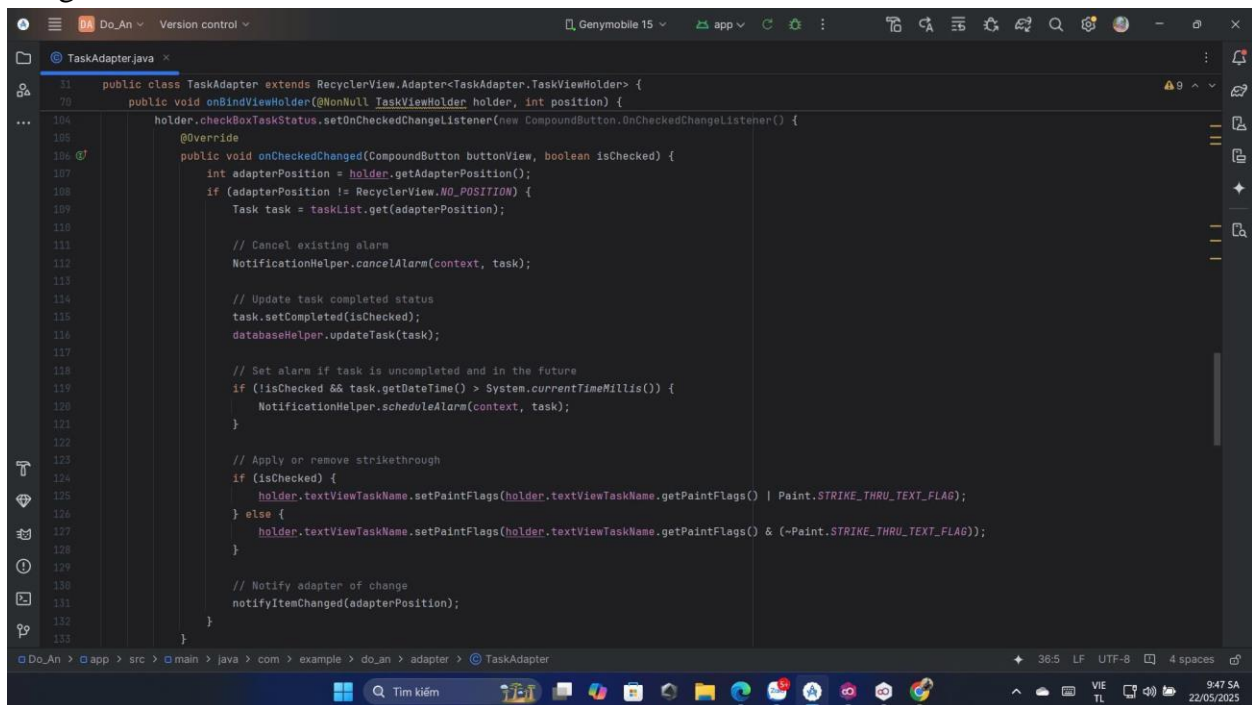
## 2.2. TaskAdapter

Thay đổi trạng thái quan trọng của công việc: Việc hoàn thành hay chưa hoàn thành là trọng tâm trong quản lý công việc.

Cập nhật vào cơ sở dữ liệu: Trạng thái được ghi vào SQLite giúp giữ dữ liệu bền vững.

Tích hợp thông báo (AlarmManager): Nếu người dùng đánh dấu là chưa hoàn thành, thì app sẽ tự động lên lịch lại thông báo nhắc nhở.

Cập nhật giao diện (strikethrough + notifyItemChanged): Giúp người dùng thấy ngay tác động khi tick/untick.



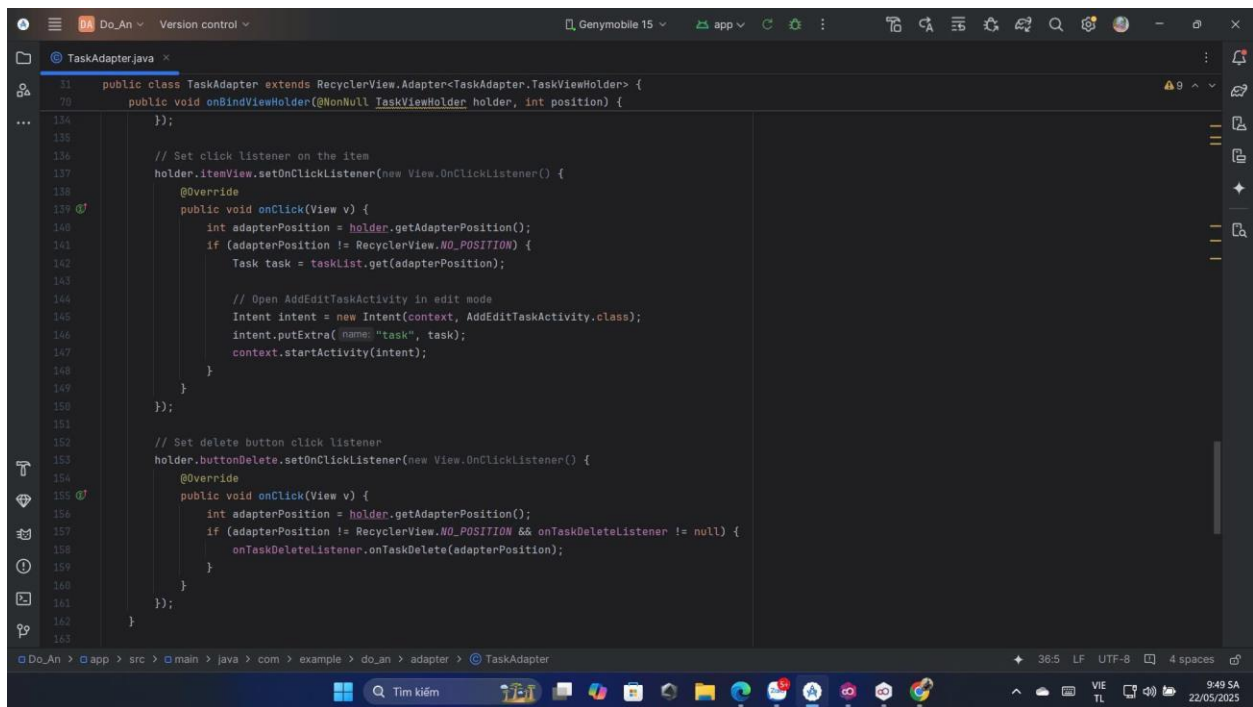
```
11 public class TaskAdapter extends RecyclerView.Adapter<TaskAdapter.TaskViewHolder> {
79 public void onBindViewHolder(@NonNull TaskViewHolder holder, int position) {
...
104 holder.checkBoxTaskStatus.setOnCheckedChangeListener(new CompoundButton.OnCheckedChangeListener() {
105     @Override
106     public void onCheckedChanged(CompoundButton buttonView, boolean isChecked) {
107         int adapterPosition = holder.getAdapterPosition();
108         if (adapterPosition != RecyclerView.NO_POSITION) {
109             Task task = taskList.get(adapterPosition);
110
111             // Cancel existing alarm
112             NotificationHelper.cancelAlarm(context, task);
113
114             // Update task completed status
115             task.setCompleted(isChecked);
116             databaseHelper.updateTask(task);
117
118             // Set alarm if task is uncompleted and in the future
119             if (!isChecked && task.getDateTime() > System.currentTimeMillis()) {
120                 NotificationHelper.scheduleAlarm(context, task);
121             }
122
123             // Apply or remove strikethrough
124             if (isChecked) {
125                 holder.textViewTaskName.setPaintFlags(holder.textViewTaskName.getPaintFlags() | Paint.STRIKE_THRU_TEXT_FLAG);
126             } else {
127                 holder.textViewTaskName.setPaintFlags(holder.textViewTaskName.getPaintFlags() & (~Paint.STRIKE_THRU_TEXT_FLAG));
128             }
129
130             // Notify adapter of change
131             notifyItemChanged(adapterPosition);
132         }
133     }
134 }
```

Hình 2.2.1: Phần code này quan trọng trong TaskAdapter

Ngoài ra, các phần cũng quan trọng nhưng hỗ trợ: itemView.setOnClickListener: Mở màn hình chỉnh sửa công việc.

buttonDelete.setOnClickListener: Gọi callback để xóa task.

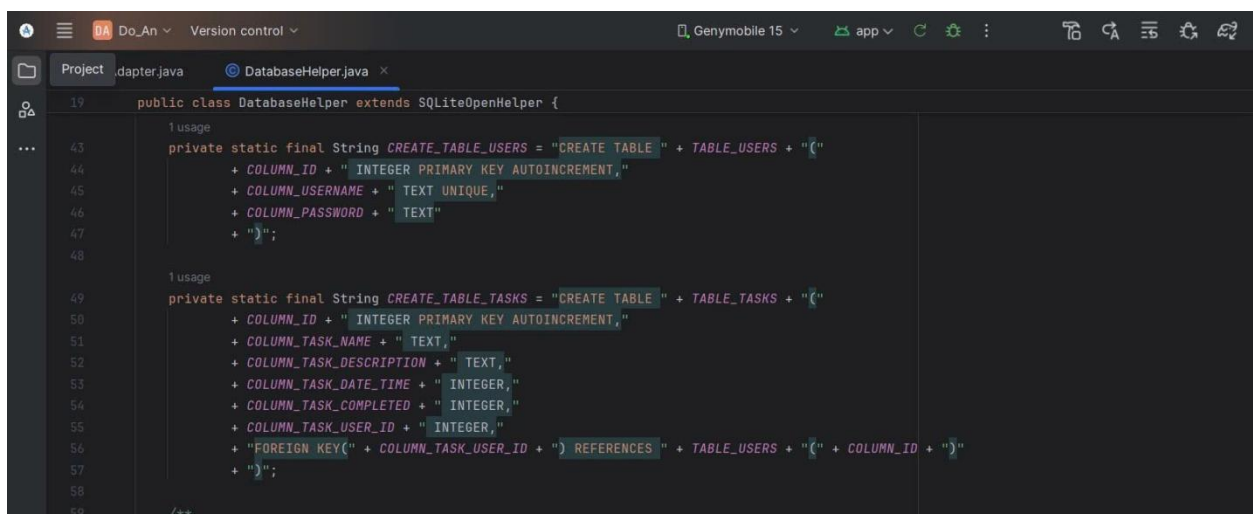
updateData(): Dùng để cập nhật toàn bộ danh sách khi có thay đổi.



Hình 2.2.2: Phần code mở chỉnh sửa và xóa công việc

### 2.3. DatabaseHelper

Là các đoạn code liên quan đến truy xuất dữ liệu:



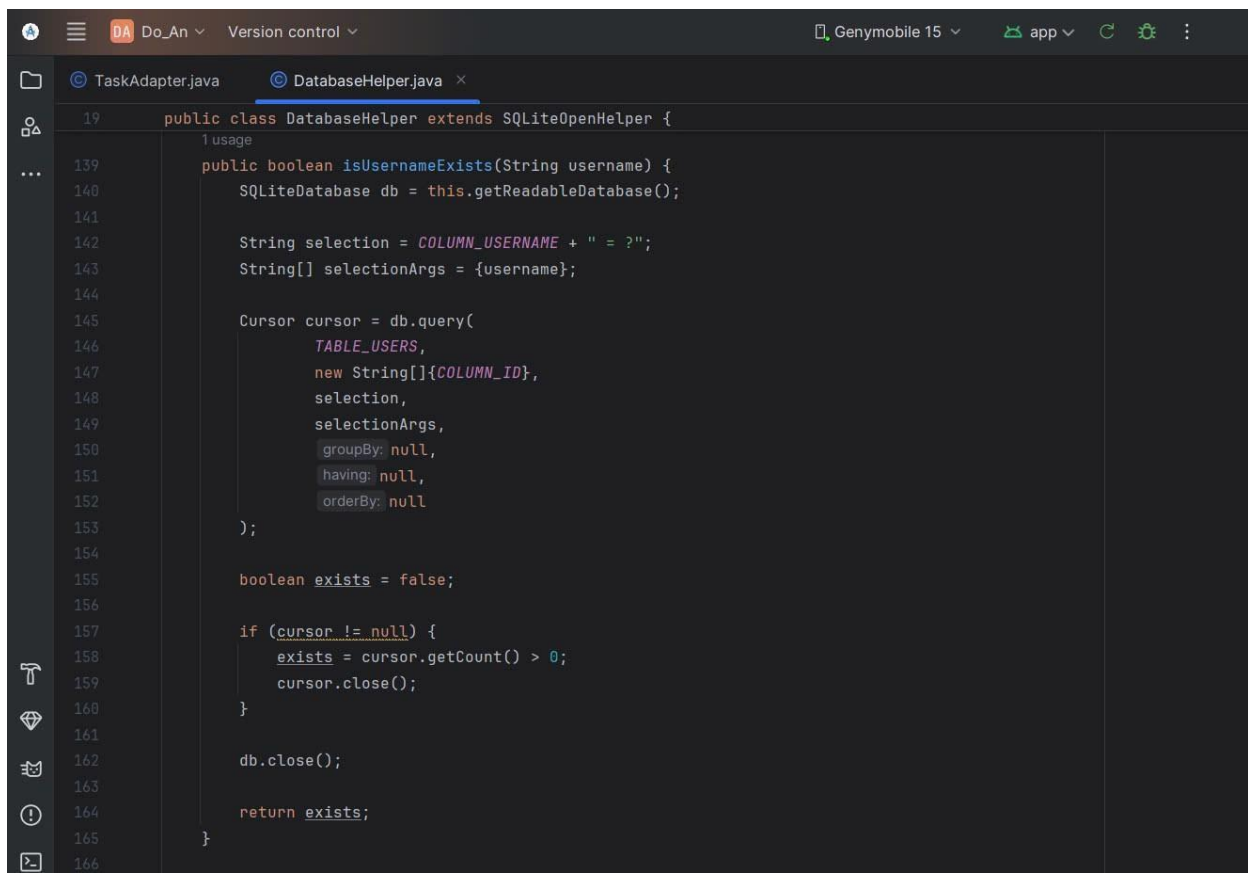
Hình 2.3.1: Tạo cơ sở dữ liệu

```
84      * Add a new user to the database
85      */
86      @usage
87      public long addUser(User user) {
88          SQLiteDatabase db = this.getWritableDatabase();
89
90          ContentValues values = new ContentValues();
91          values.put(COLUMN_USERNAME, user.getUsername());
92          values.put(COLUMN_PASSWORD, user.getPassword());
93
94          // Insert row
95          long id = db.insert(TABLE_USERS, nullColumnHack: null, values);
96
97          db.close();
98
99          return id;
100      }
```

Hình 2.3.2: Thêm người dùng

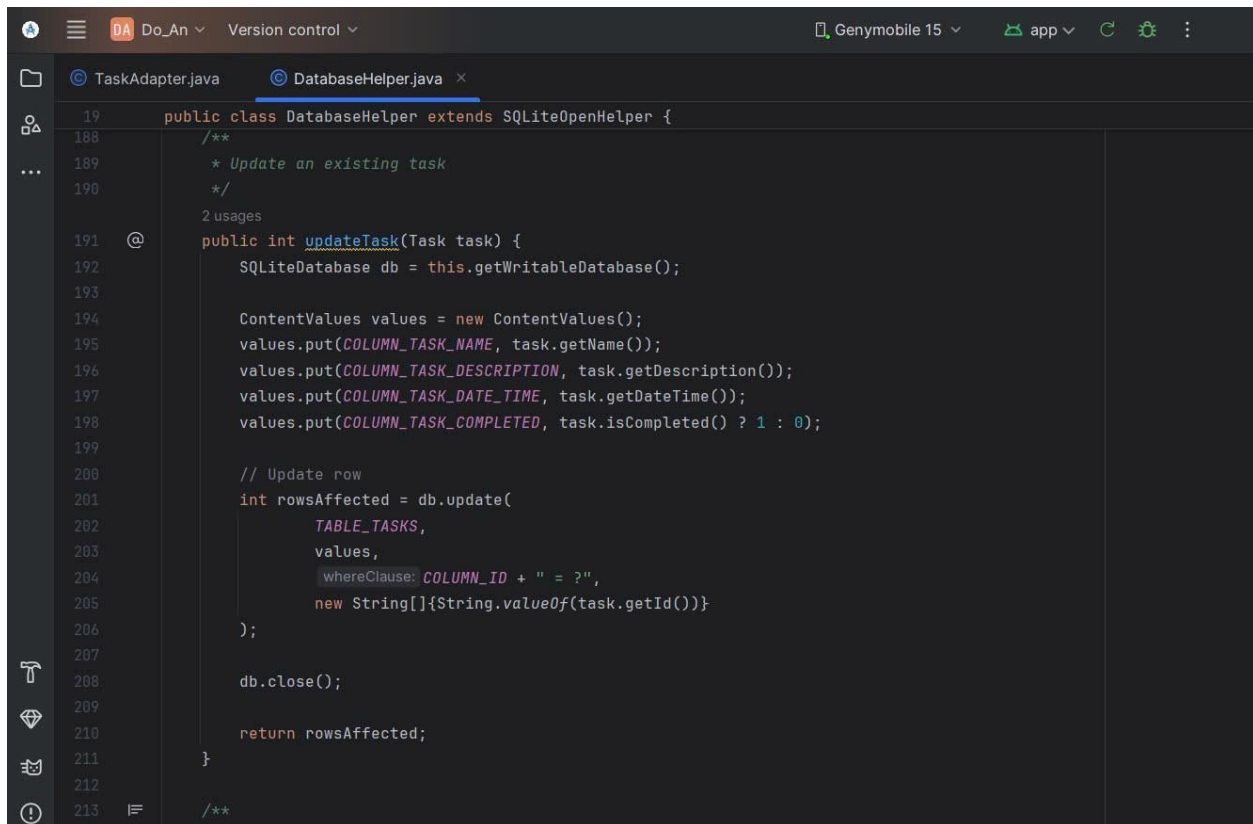
```
19      public class DatabaseHelper extends SQLiteOpenHelper {
20          @usage
21          public User getUser(String username, String password) {
22              SQLiteDatabase db = this.getReadableDatabase();
23
24              String selection = COLUMN_USERNAME + " = ? AND " + COLUMN_PASSWORD + " = ?";
25              String[] selectionArgs = {username, password};
26
27              Cursor cursor = db.query(
28                  TABLE_USERS,
29                  new String[]{COLUMN_ID, COLUMN_USERNAME, COLUMN_PASSWORD},
30                  selection,
31                  selectionArgs,
32                  null,
33                  null,
34                  null,
35                  null);
36
37              User user = null;
38
39              if (cursor != null && cursor.moveToFirst()) {
40                  user = new User(
41                      cursor.getInt(cursor.getColumnIndexOrThrow(COLUMN_ID)),
42                      cursor.getString(cursor.getColumnIndexOrThrow(COLUMN_USERNAME)),
43                      cursor.getString(cursor.getColumnIndexOrThrow(COLUMN_PASSWORD))
44                  );
45                  cursor.close();
46              }
47
48              db.close();
49
50              return user;
51          }
```

Hình 2.3.3: Người dùng đăng nhập



Hình 2.3.4: Kiểm tra người dùng để đăng nhập vào ứng dụng





The screenshot shows an IDE window with two tabs: TaskAdapter.java and DatabaseHelper.java. The DatabaseHelper.java tab is active, displaying the `updateTask` method. The code is as follows:

```
19 public class DatabaseHelper extends SQLiteOpenHelper {
188 /**
189  * Update an existing task
190  */
191 @
192 public int updateTask(Task task) {
193     SQLiteDatabase db = this.getWritableDatabase();
194
195     ContentValues values = new ContentValues();
196     values.put(COLUMN_TASK_NAME, task.getName());
197     values.put(COLUMN_TASK_DESCRIPTION, task.getDescription());
198     values.put(COLUMN_TASK_DATE_TIME, task.getDateTime());
199     values.put(COLUMN_TASK_COMPLETED, task.isCompleted() ? 1 : 0);
200
201     // Update row
202     int rowsAffected = db.update(
203         TABLE_TASKS,
204         values,
205         whereClause: COLUMN_ID + " = ?",
206         new String[]{String.valueOf(task.getId())}
207     );
208
209     db.close();
210
211     return rowsAffected;
212 }
213 /**
```

Hình 2.3.5: Chỉnh sửa công việc



The screenshot shows the `deleteTask` method in the DatabaseHelper.java file. The code is as follows:

```
216 public void deleteTask(int taskId) {
217     SQLiteDatabase db = this.getWritableDatabase();
218
219     db.delete(
220         TABLE_TASKS,
221         whereClause: COLUMN_ID + " = ?",
222         new String[]{String.valueOf(taskId)}
223     );
224
225     db.close();
226 }
227
228 /**
```

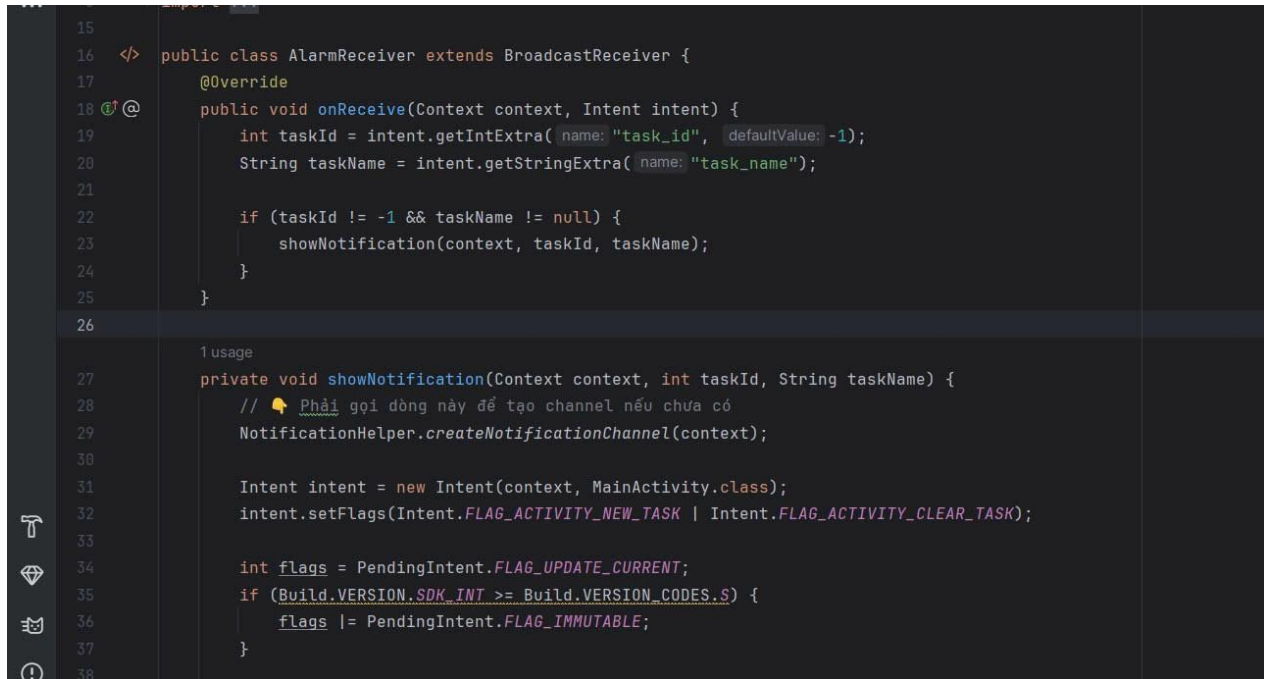
Hình 2.3.6: Code để xóa 1 công việc



## 2.4. AlarmReceiver

Nó nhận tín hiệu báo thức từ AlarmManager (đặt ở nơi khác trong app).

Khi được kích hoạt, nó sẽ hiển thị thông báo tương ứng với công việc được nhắc.



```
15
16 </> public class AlarmReceiver extends BroadcastReceiver {
17     @Override
18     @
19     public void onReceive(Context context, Intent intent) {
20         int taskId = intent.getIntExtra("task_id", -1);
21         String taskName = intent.getStringExtra("task_name");
22
23         if (taskId != -1 && taskName != null) {
24             showNotification(context, taskId, taskName);
25         }
26     }
27
28     1 usage
29     private void showNotification(Context context, int taskId, String taskName) {
30         // 📌 Phải gọi dòng này để tạo channel nếu chưa có
31         NotificationHelper.createNotificationChannel(context);
32
33         Intent intent = new Intent(context, MainActivity.class);
34         intent.setFlags(Intent.FLAG_ACTIVITY_NEW_TASK | Intent.FLAG_ACTIVITY_CLEAR_TASK);
35
36         int flags = PendingIntent.FLAG_UPDATE_CURRENT;
37         if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.S) {
38             flags |= PendingIntent.FLAG_IMMUTABLE;
39         }
40     }
41 }
```

Hình 2.4.1: Code đặt lời nhắc

## 2.5. BroadcastReceiver

Mục đích: Đặt lại báo thức (alarm) cho các công việc chưa hoàn thành sau khi thiết bị khởi động lại.

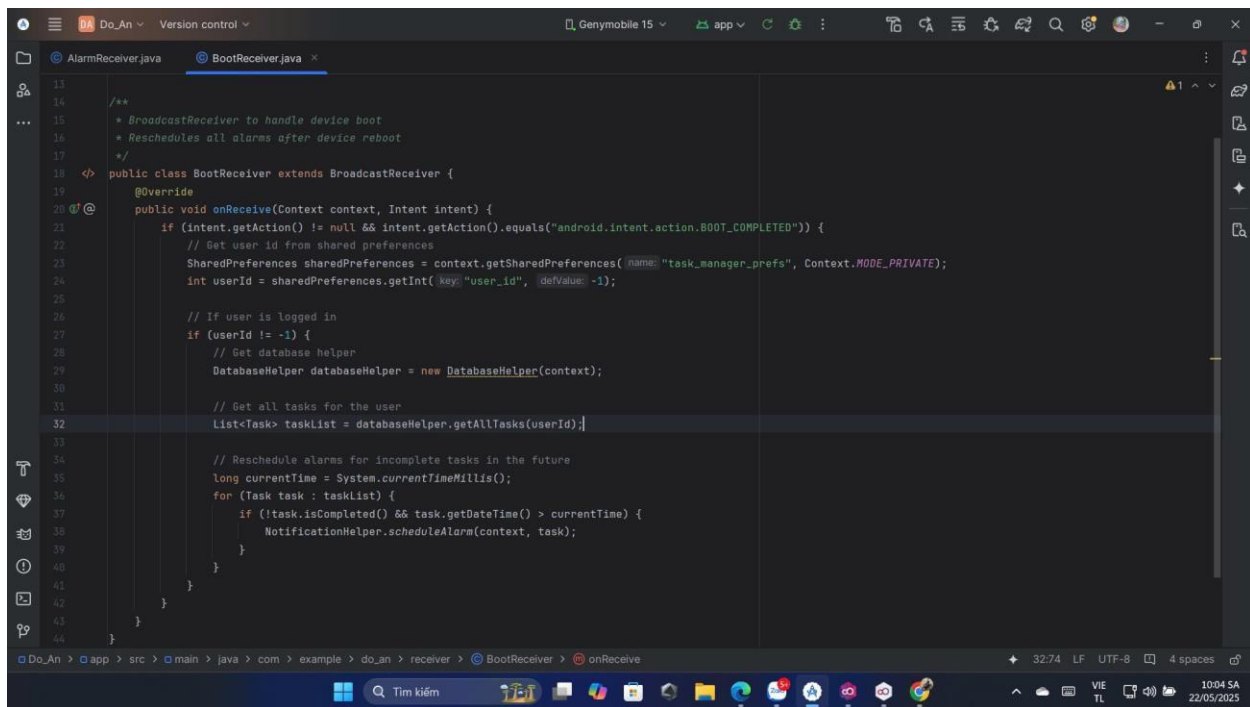
Cách hoạt động:

Nhận sự kiện BOOT\_COMPLETED.

Lấy user\_id từ SharedPreferences.

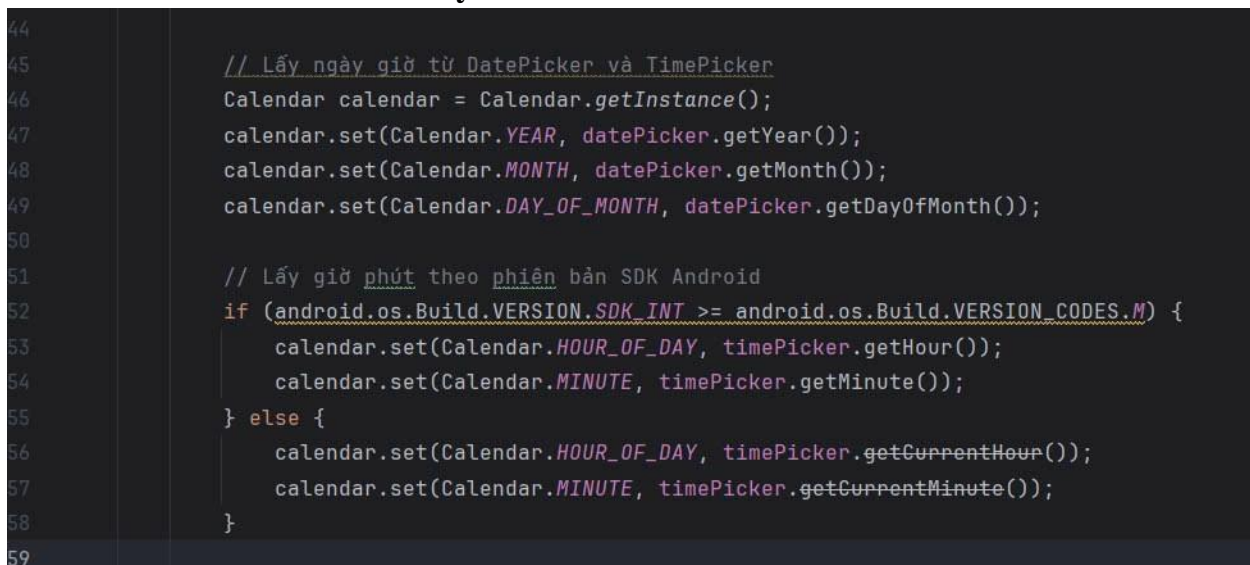
Truy vấn danh sách công việc từ SQLite.

Với mỗi công việc chưa hoàn thành và còn thời gian, gọi scheduleAlarm() để đặt lại thông báo nhắc nhở.

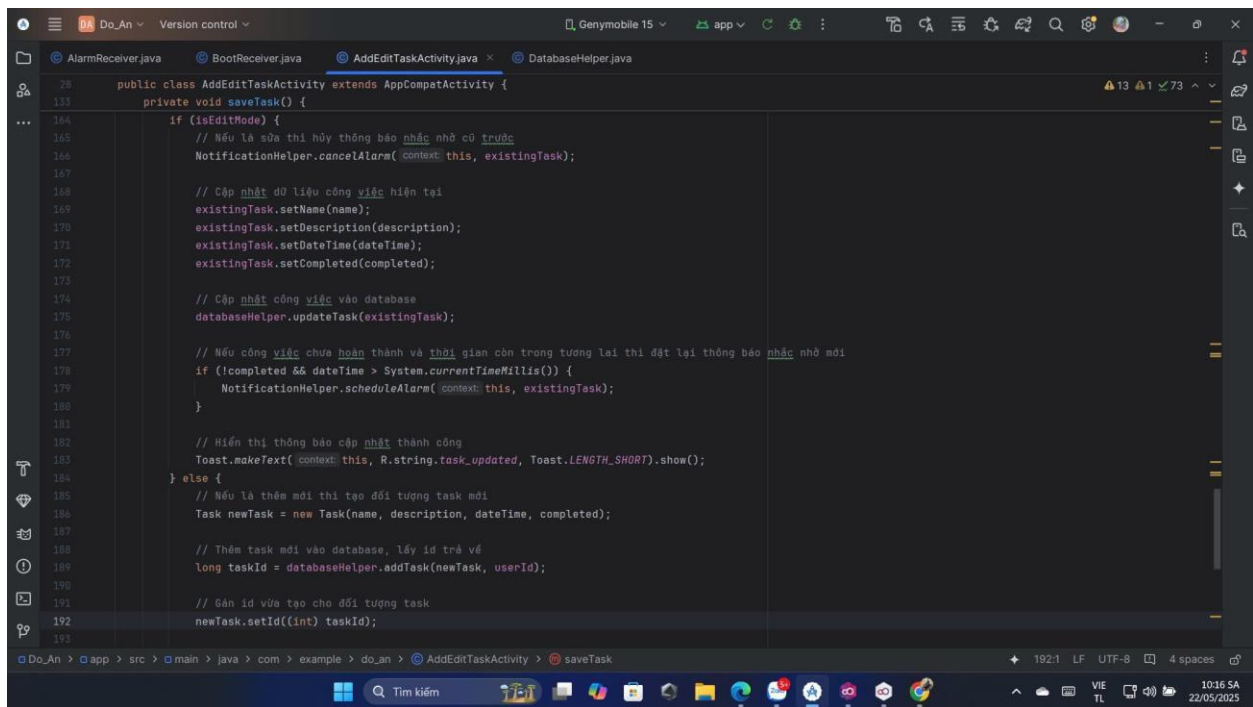


Hình 2.5.1: Code đặt lịch nhắc nhở

## 2.6. AddEditTaskActivity

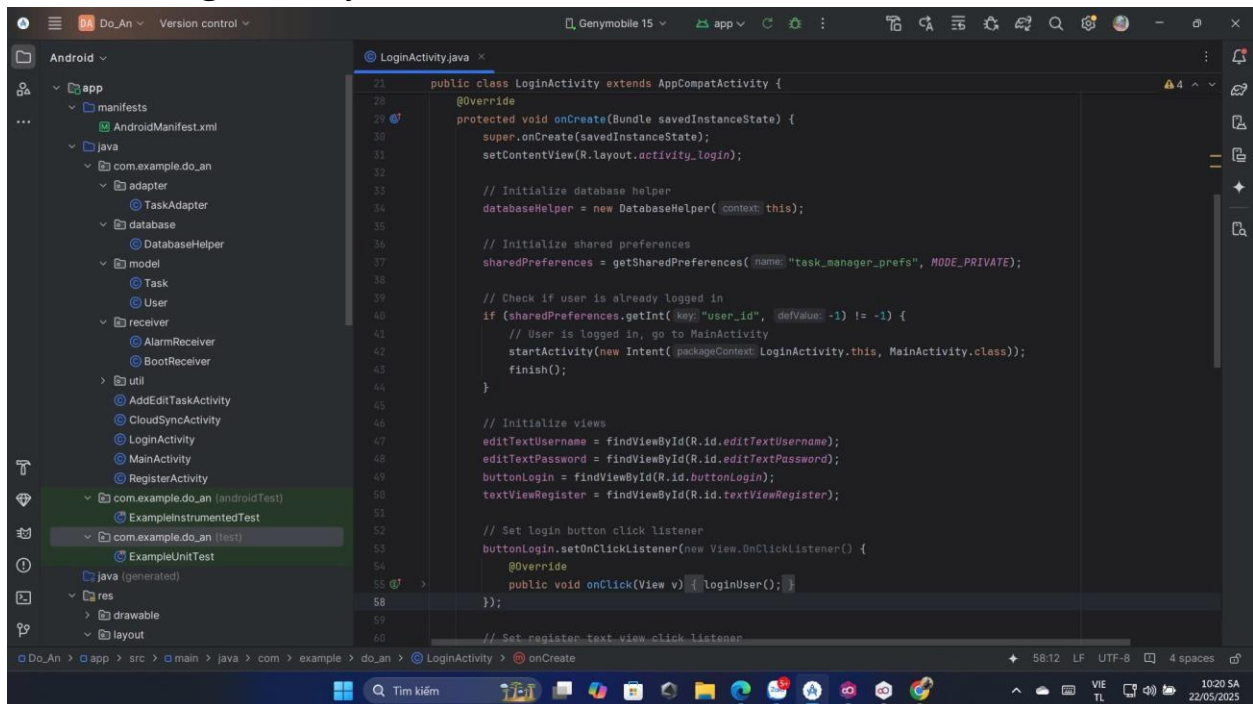


Hình 2.6.1: Lấy Ngày và Giờ từ DatePicker & TimePicker java

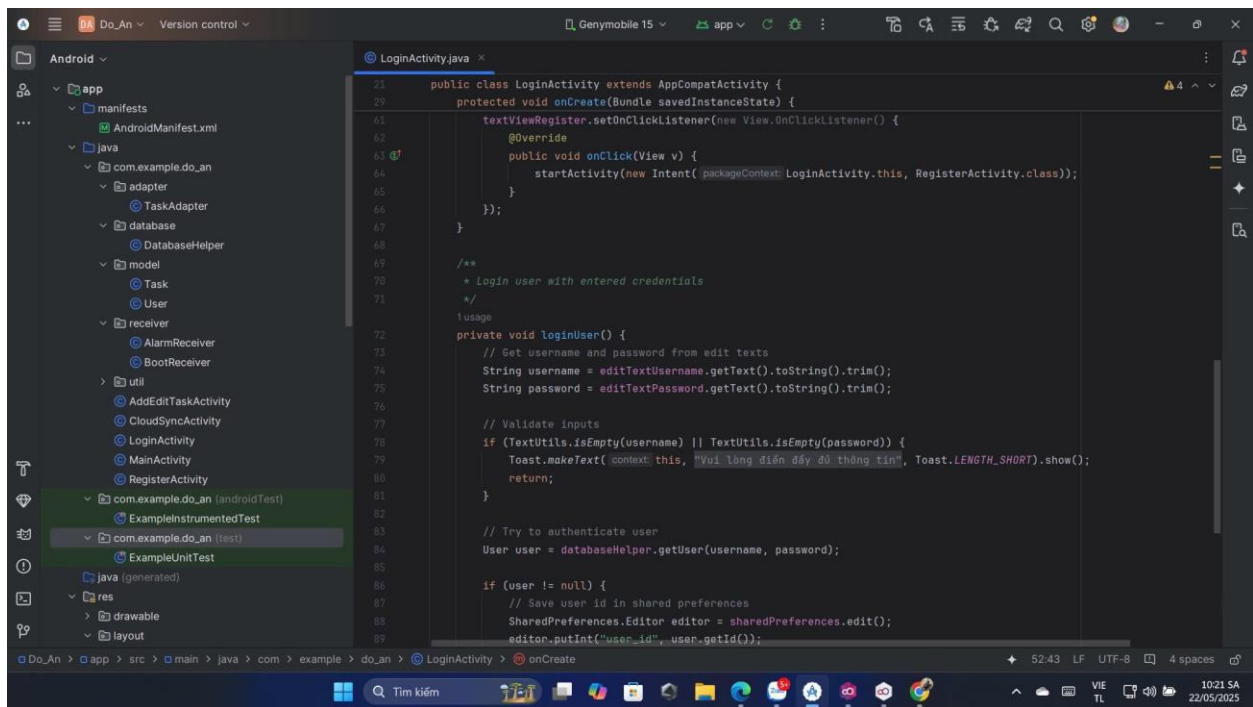


Hình 2.6.2: Xử lý cả 2 trường hợp thêm mới và chỉnh sửa task. Giao tiếp với SQLite và hệ thống nhắc nhở.

## 2.7. LoginActivity

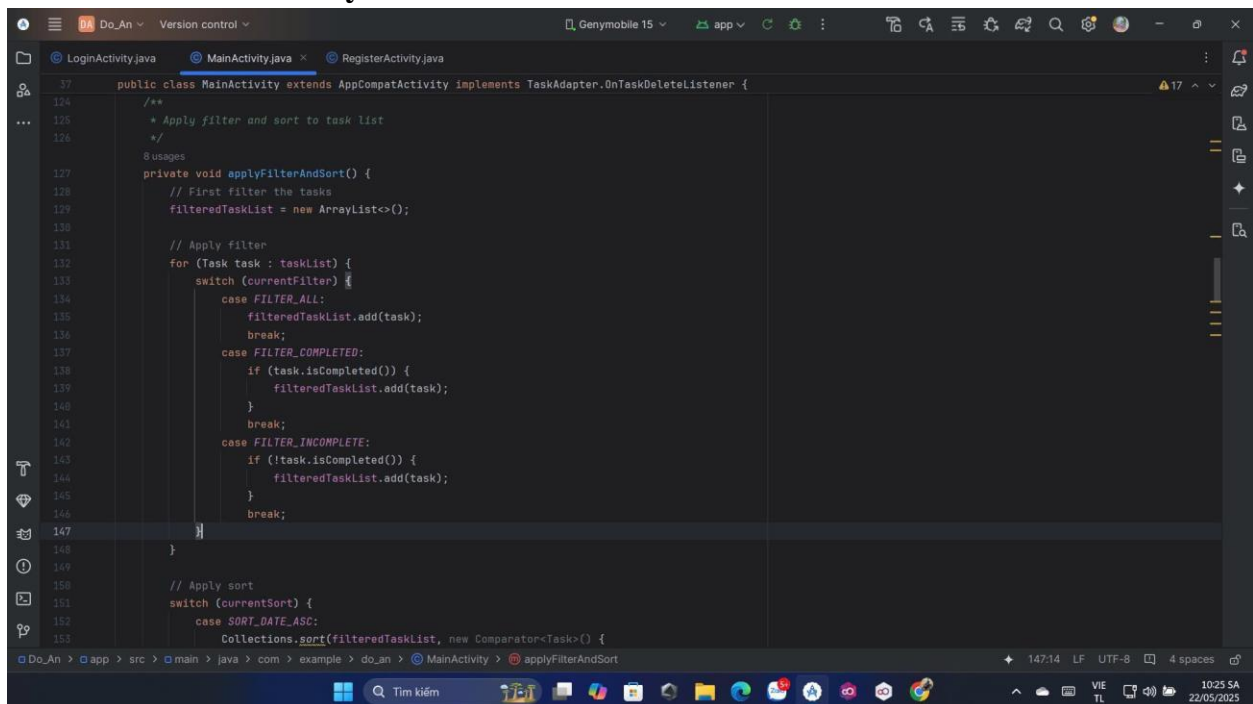


Hình 2.7.1: Code để người dùng đăng nhập vào ứng dụng đoạn 1

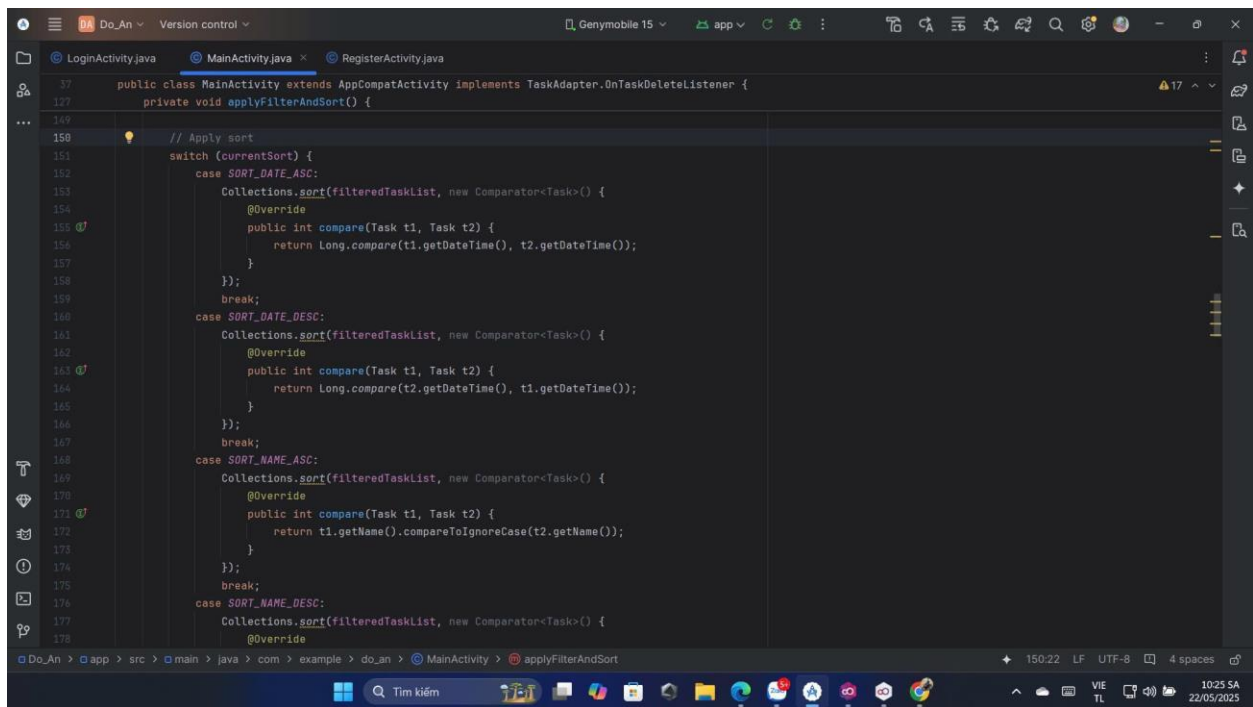


hình 2.7.1: Code để người dùng đăng nhập vào ứng dụng đoạn 1

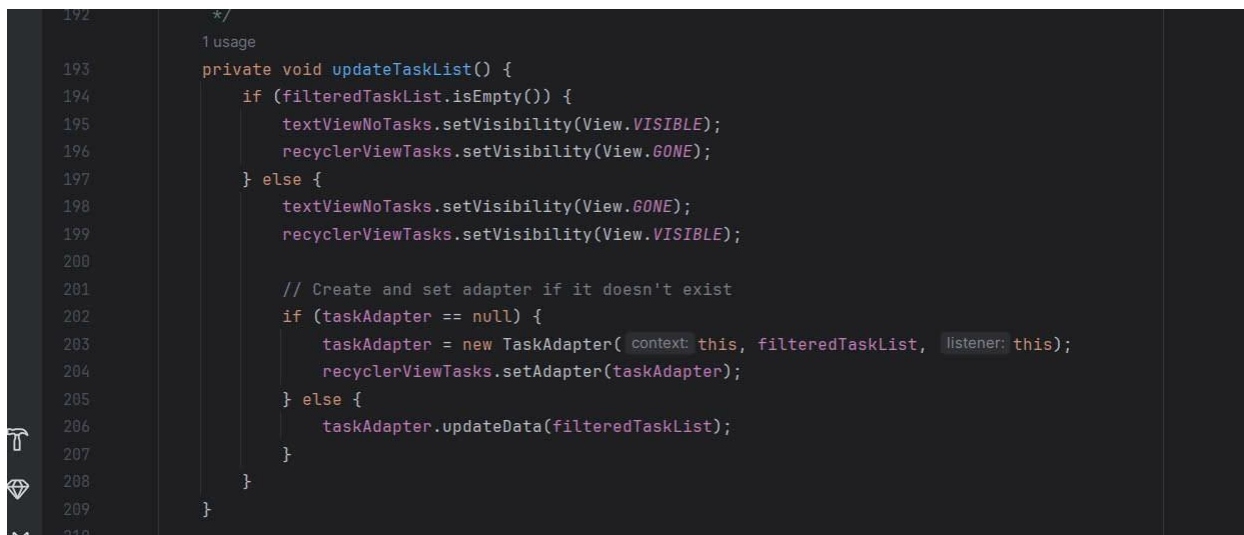
## 2.8. MainActivity



Hình 2.8.1: Lọc task theo trạng thái (tất cả, đã hoàn thành, chưa hoàn thành).



Hình 2.8.2: Sắp xếp task theo ngày hoặc tên, tăng hoặc giảm dần.



Hình 2.8.3: Khởi tạo hoặc cập nhật adapter với danh sách task mới

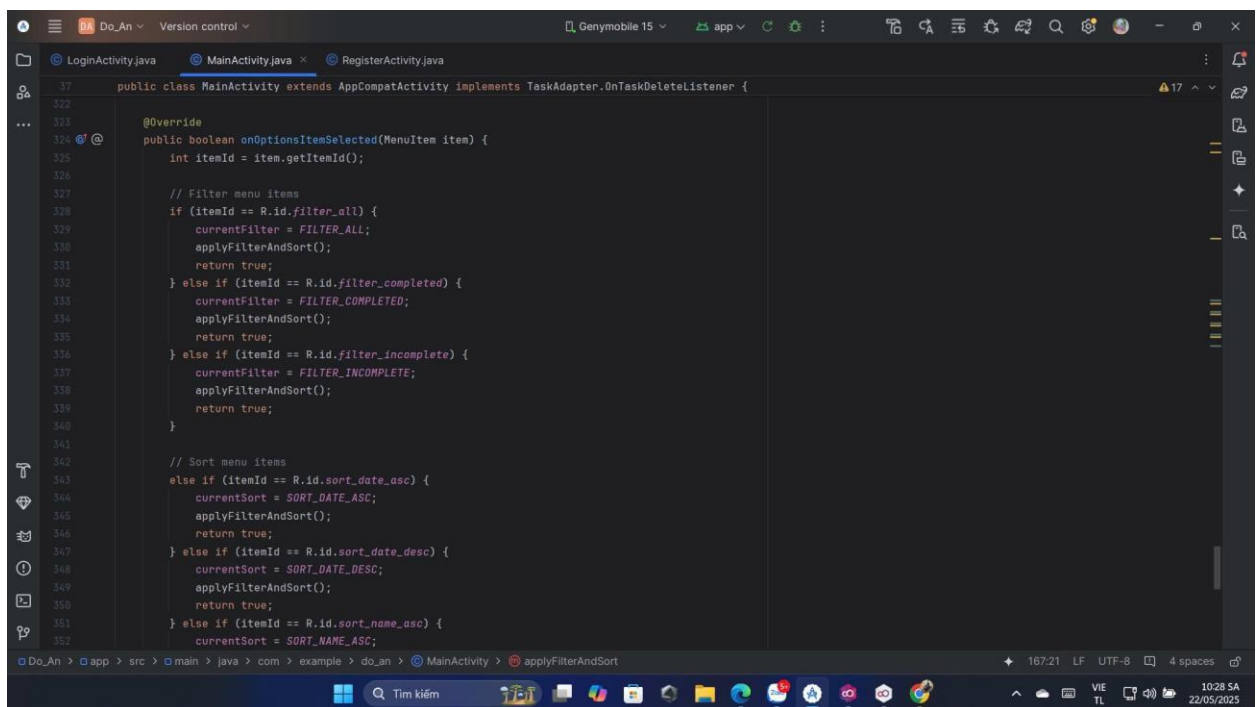


```

210
211     @Override
212     protected void onActivityResult(int requestCode, int resultCode, @Nullable Intent data) {
213         super.onActivityResult(requestCode, resultCode, data);
214
215         if (resultCode == RESULT_OK) {
216             if (requestCode == REQUEST_ADD_TASK || requestCode == REQUEST_EDIT_TASK) {
217                 // Reload tasks list
218                 loadTasks();
219             }
220         }
221     }
222

```

Hình 2.8.4: Xử lý kết quả từ màn hình thêm/sửa task

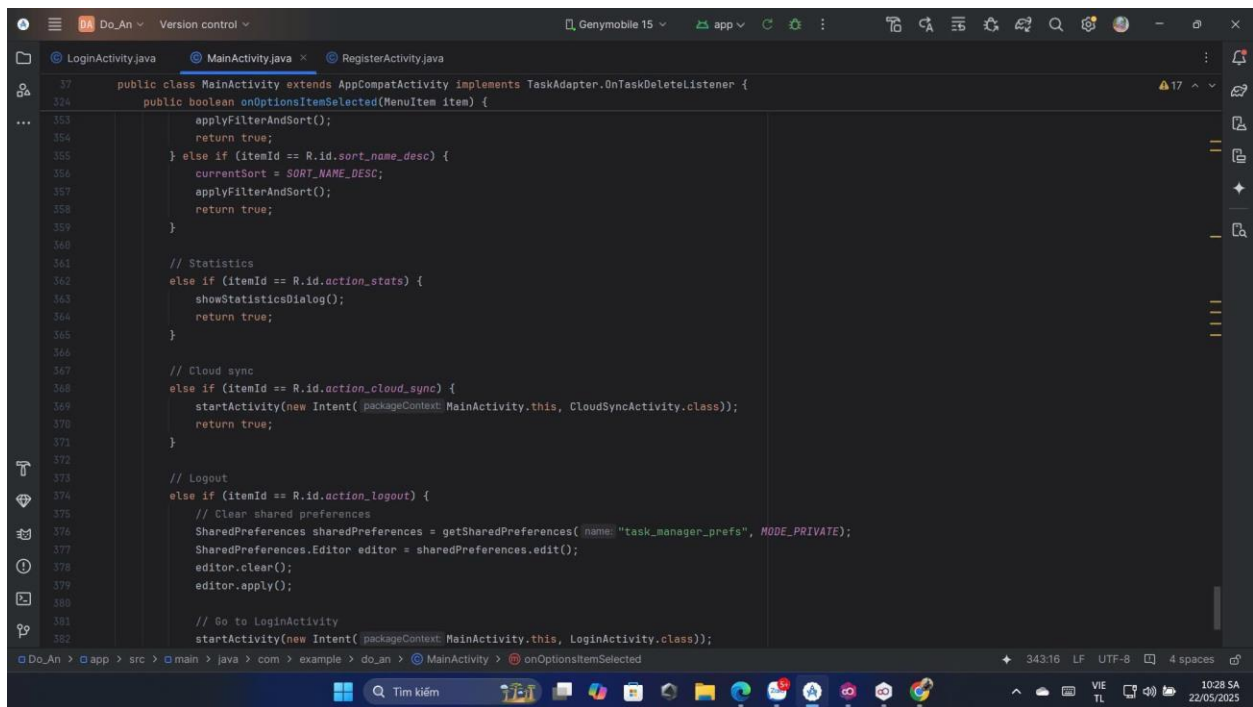


```

37 public class MainActivity extends AppCompatActivity implements TaskAdapter.OnTaskDeleteListener {
38
39     @Override
40     public boolean onOptionsItemSelected(MenuItem item) {
41         int itemId = item.getItemId();
42
43         // Filter menu items
44         if (itemId == R.id.filter_all) {
45             currentFilter = FILTER_ALL;
46             applyFilterAndSort();
47             return true;
48         } else if (itemId == R.id.filter_completed) {
49             currentFilter = FILTER_COMPLETED;
50             applyFilterAndSort();
51             return true;
52         } else if (itemId == R.id.filter_incomplete) {
53             currentFilter = FILTER_INCOMPLETE;
54             applyFilterAndSort();
55             return true;
56         }
57
58         // Sort menu items
59         else if (itemId == R.id.sort_date_asc) {
60             currentSort = SORT_DATE_ASC;
61             applyFilterAndSort();
62             return true;
63         } else if (itemId == R.id.sort_date_desc) {
64             currentSort = SORT_DATE_DESC;
65             applyFilterAndSort();
66             return true;
67         } else if (itemId == R.id.sort_name_asc) {
68             currentSort = SORT_NAME_ASC;
69

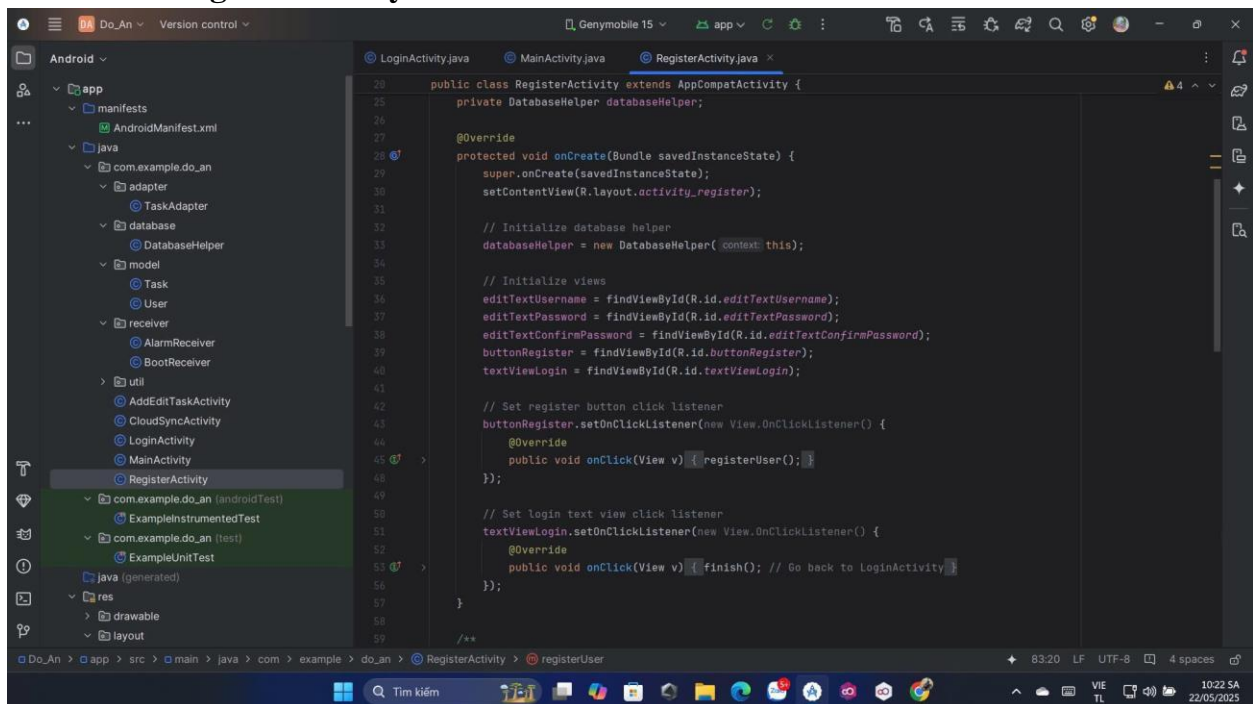
```

Hình 2.8.5: Menu xử lý lọc, sắp xếp, thống kê, đồng bộ, đăng xuất đoạn 1

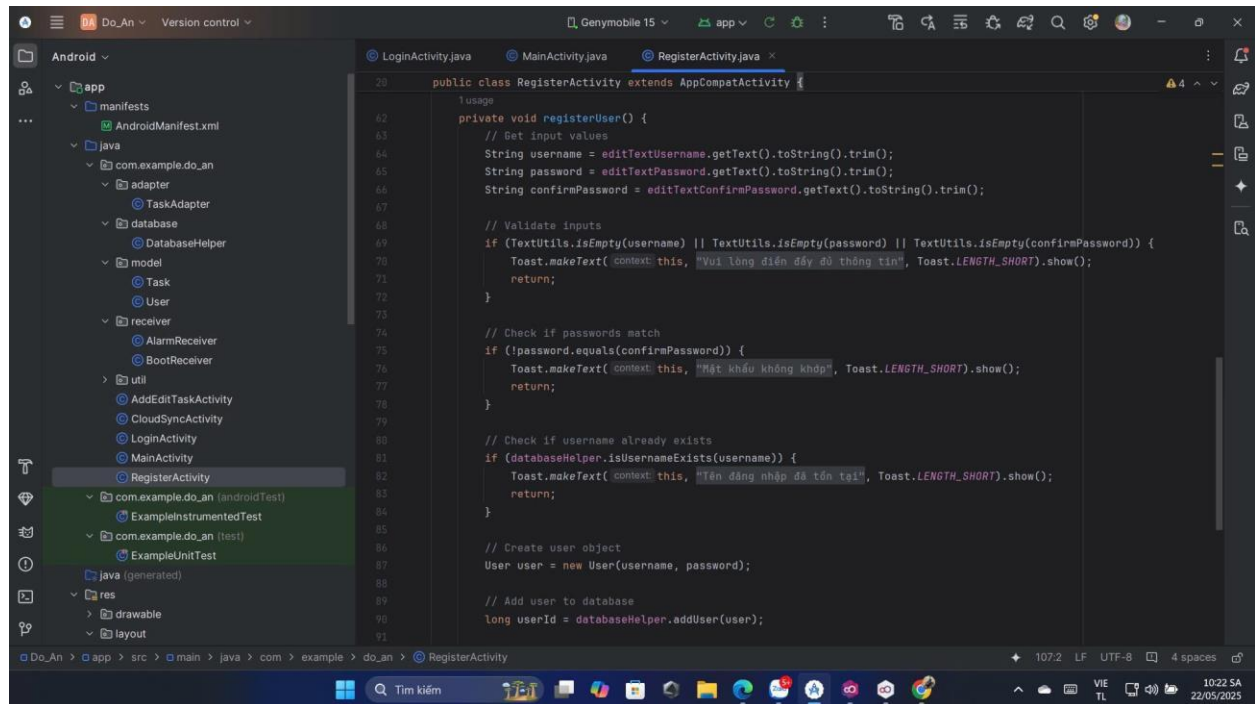


Hình 2.8.5: Menu xử lý lọc, sắp xếp, thống kê, đồng bộ, đăng xuất đoạn 2

## 2.9. RegisterActivity



Hình 2.9.1: Đoạn 1 code đăng ký người dùng mới



Hình 2.9.2: Đoạn 2 code đăng ký người dùng mới



## TÀI LIỆU THAM KHẢO

1. GIÁO TRÌNH MÔ ĐUN: XÂY DỰNG ỨNG DỤNG TRÊN ANDROID – TRƯỜNG CAO ĐẲNG KIÊN GIANG
2. ANDROID DEVELOPERS - <https://developer.android.com>
3. ĐỒ ÁN TỐT NGHIỆP XÂY DỰNG ỨNG DỤNG ANDROID NGHE NHẠC TRÊN INTERNET - <https://www.slideshare.net/slideshow/n-tt-nghiep-xy-dng-ng-dng-android-nghe-nhc-trn-internetdoc/258492881>
4. CÁC THÀNH PHẦN CƠ BẢN TRONG ANDROID LÀ GÌ? - [Các Thành Phần Cơ Bản Trong Android Là Gì? | CodeLearn](#)
5. Chatgpt - <https://chatgpt.com>

### PHỤ LỤC: DANH MỤC CÔNG VIỆC

STT	HỌ VÀ TÊN	CÔNG VIỆC
1	TRẦN HÀO	Lập trình ứng dụng Nâng cấp và bảo trì hệ thống
2	LÊ TẤN KHA	
3	ÂU TRIỀU TÂN	Viết báo cáo - thuyết trình Góp ý tính năng và giao diện
4	NGUYỄN YẾN NHI	Làm Powerpoint Kiểm thử ứng dụng

- Tài liệu cả toàn bộ code của đồ án nhóm đã thực hiện:  
[https://github.com/letankha/Do\\_An.git](https://github.com/letankha/Do_An.git)