Shape, square

Description automatically generated

**HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG**

**KHOA CÔNG NGHỆ THÔNG TIN**

Logo

Description automatically generated

**BÁO CÁO MÔN THỰC TẬP CƠ SỞ**

**ĐỀ TÀI: APP QUẢN LÝ CÔNG VIỆC TODO LIST**

|  |  |
| --- | --- |
| **Giảng viên:** | **Ths. Nguyễn Hoàng Anh** |
| **Họ & tên:** | **Hoàng Đức Minh** |
| **MSV:** | **B19DCCN428** |
| **Lớp:** | **D19CNPM1** |

Hà Nội, 26 tháng 5 năm 2022

**MỤC LỤC**

[**I.** **Đặt vấn đề** 2](#_Toc104671458)

[**II.** **Giải quyết vấn đề** 4](#_Toc104671459)

[**1.** **Mục đích – phạm vi** 4](#_Toc104671460)

[**2.** **Sơ đồ hệ thống** 4](#_Toc104671461)

[**3.** **Thực hiện dự án** 4](#_Toc104671462)

[3.1 Tổ chức dự án 4](#_Toc104671463)

[3.2 Công cụ 5](#_Toc104671464)

[**4.** **Tìm hiểu mô hình MVVM** 5](#_Toc104671465)

[**5.** **Tìm hiểu Data Binding** 7](#_Toc104671466)

[**6.** **Ngôn ngữ Kotlin** 7](#_Toc104671467)

[**7.** **Tìm hiểu Room Database** 9](#_Toc104671468)

[**8.** **Xây dựng ứng dụng** 9](#_Toc104671469)

[10.1 Thiết kế giao diện ứng dụng: 9](#_Toc104671470)

[10.2 Tạo project mới và xây dựng cây thư mục trên Android Studio: 10](#_Toc104671471)

[10.3 Xây dựng database: 11](#_Toc104671472)

[10.4 Tạo Drawer Navigator: 13](#_Toc104671473)

[10.5 Tạo ViewModel: 14](#_Toc104671474)

[10.6 Tạo các fragment để hiển thị data từ viewmodel và hiển thị lên: 14](#_Toc104671475)

[10.7 Tạo navigation để chuyển giữa các fragment: 14](#_Toc104671476)

[**III.** **Kết quả** 15](#_Toc104671477)

[**1.** **TaskFragment** 15](#_Toc104671478)

[**2.** **DoneFragment** 16](#_Toc104671479)

[**3.** **DeletedFragment** 17](#_Toc104671480)

[**4.** **AddNewTaskDialog** 18](#_Toc104671481)

[**TỔNG KẾT** 20](#_Toc104671482)

[**TÀI LIỆU THAM KHẢO** 21](#_Toc104671483)

1. **Đặt vấn đề**

Trong nhiều năm trở lại đây, hệ điều hành android với sự kết hợp của nhiều công nghệ tiên tiến nhất hiện nay đã nhanh chóng trở thành đối thủ cạnh tranh mạnh mẽ với các hệ điều hành hiện có và đang là hệ điều hành lớn nhất với số lượng người sử dụng chiếm 74,4% trên toàn thế giới còn ở Việt Nam cũng đang chiếm tới 60% thị phần.

Với sự phát triển nhanh chóng của xã hội, số lượng công việc trong một ngày mà một người phải thực hiện ngày một lớn hơn dẫn tới quá tải nếu luôn phải tự quản lý tất cả mọi thứ mà không có công cụ hỗ trợ. Smartphone giờ đây cũng đã trở thành vật gắn liền với cuộc sống của chúng ta vì thế tại sao không tận dụng nó để hỗ trợ quản lý những công việc hằng ngày với ứng dụng TodoList.

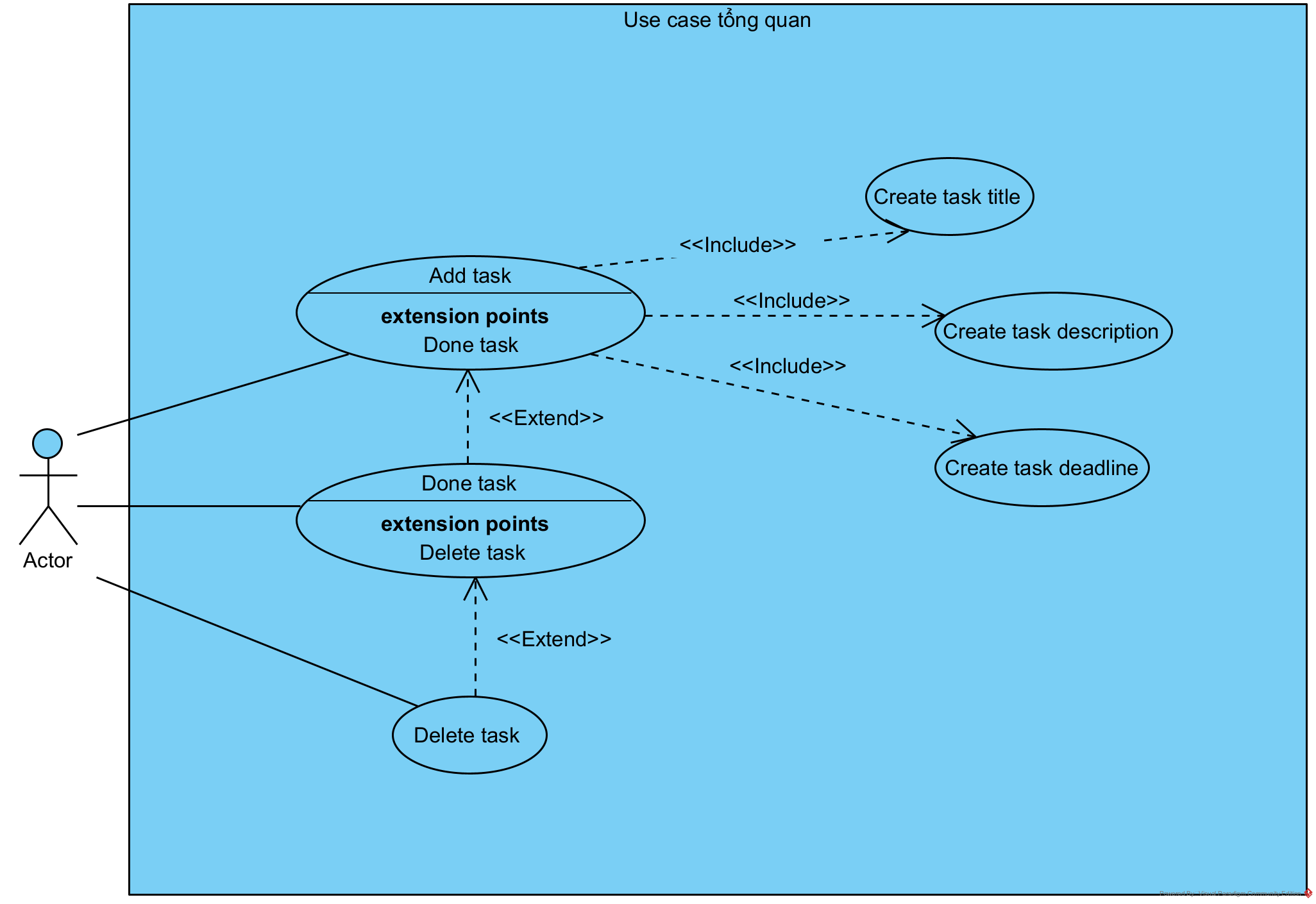
1. **Giải quyết vấn đề**

* Từ vấn đề trên, ứng dụng quản lý công việc TodoList ra đời.

1. **Mục đích – phạm vi**

* **Mục đích:**
* Ứng dụng cho phép thêm công việc trên nền tảng android, tạo checklist các công việc, thay đổi trạng thái các công việc trong list.
* **Phạm vi:**
* Là ứng dụng chạy trên điện thoại cá nhân, sử dụng độc lập.

1. **Sơ đồ hệ thống**

****

1. **Thực hiện dự án**
   1. Tổ chức dự án

* Sử dụng mô hình MVVM kết hợp Data Binding
* Quản lý dự án bằng Git
  1. Công cụ
* Phát triển ứng dụng trên nền tảng Android
* Công cụ phát triển Android Studio
* Ngôn ngữ: Kotlin
* Lưu trữ database: Room Database

1. **Tìm hiểu mô hình MVVM**

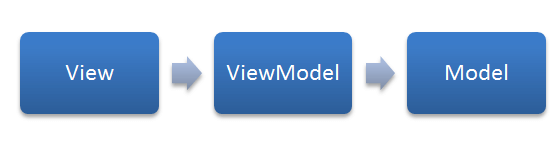
*Mô hình MVVM là gì ?*

MVVM không phải là framework hay thư viện, api… nó chỉ đơn thuần là hướng dẫn bạn định nghĩa cấu trúc ứng dụng của bạn. MVVM được phát triển dựa trên kiến trúc MVP.

Mô hình mvvm cho phép tách biệt dữ liệu (Model), mã thực thi (logic hay ViewModel) và giao diện người dùng (View).

Trong các mô hình truyền thống, chúng ta thường xử lý sự kiện Click và viết mã thực thi trực tiếp ở trên một Button nhưng với mô hình MVVM không cho phép làm điều này.

Trong mô hình MVVM, các điều khiển(control) như Button, ListView, SearchBar, v.v. không thể kết buộc trực tiếp đến dữ liệu mà phải thông qua thuộc tính Command – là một thuộc tính kiểu ICommand.



*MVVM được hiểu như thế nào ?*

***View:***

* Thành phần giao diện của ứng dụng. Tương tự như ở trong mô hình MVC, View là thành phần duy nhất mà người dùng có thể tương tác được trong chương trình, nó chính là thành phần mô tả dữ liệu.
* Một điểm khác biệt so với các mô hình khác là View trong mô hình này tích cực hơn. Nó có khả năng thực hiện các hành vi và phản hồi lại người dùng thông qua tính năng là: binding, command.

***Model:***

* Cũng tương tự như trong mô hình MVC. Model là các đối tượng giúp truy xuất và thao tác trên dữ liệu thực sự.

***ViewModel:***

* Lớp trung gian giữa View và Model. ViewModel có thể được xem là thành phần thay thế cho Controller trong mô hình MVC. Nó chứa các mã lệnh cần thiết để thực hiện data binding, command.

*Ưu nhược điểm của MVVM: Model – View – ViewModel?*

***Ưu điểm:***

* Thực hiện Unit testing bây giờ sẽ rất dễ dàng, vì bạn thực sự không phụ thuộc vào view.
* MVVM sẽ tạo sự tương tác hiệu quả giữa designer và developer.
* Tăng khả năng sử dụng lại các thành phần hay việc thay đổi giao diện chương trình mà không cần phải viết lại code quá nhiều.
* Phát triển ứng dụng nhanh, đơn giản, dễ nâng cấp, bảo trì…

***Nhược điểm :***

* Khả năng duy trì khi view có thể gán cả biến và biểu thức, các logic không liên quan sẽ tăng dần theo thời gian, ảnh hưởng đến việc thêm code vào XML.
* Đối với dự án nhỏ việc áp dụng mô hình MVVM gây cồng kềnh, tốn thời gian trong quá trình phát triển. Tốn thời gian trung chuyển dữ liệu của các thành phần.
* Đối với dự án lớn hơn, nó gây khó khăn và mất thời gian để thiết kế các ViewModel.
* Việc liên kết dữ liệu cho tất cả các thành phần gây khó khăn trong việc debug khi cơ sở dữ liệu phức tạp.

1. **Tìm hiểu Data Binding**

* Data Binding là kĩ thuật dùng để tạo gắn kết giữa phần giao diện (UI) và dữ liệu thông qua phần business logic. Nhờ Data Binding, UI có thể tự động cập nhật lại để hiển thị các thay đổi trong dữ liệu. Ngoài ra, Data Binding còn hỗ trợ các chiều khác nhau, nghĩa là các thay đổi có thể cập nhật từ UI vào dữ liệu. Kĩ thuật binding trong mô hình mvvm thực sự là một bước tiến mới, thỏa mãn những điều mà hầu hết lập trình viên mong đợi.

1. **Ngôn ngữ Kotlin**

*Kotlin là gì ?*

Kotlin là ngôn ngữ lập trình dành cho các ứng dụng đa nền tảng hiện đại, tương thích hoàn toàn với Android và được phát triển bởi JetBrains, công ty đứng sau IntelliJ Java IDE (bộ công cụ và ứng dụng dành cho nhà phát triển) mà Android Studio dựa trên. Tuy nhiên, đây không phải là ngôn ngữ lập trình riêng biệt mà được xem là một phương pháp mới để sử dụng và phát triển code của Java. Kotlin hoạt động liền mạch với Java, nhờ đó giúp các nhà phát triển yêu thích Java dễ dàng sử dụng thêm một ngôn ngữ lập trình tuyệt vời là Kotlin cũng như tận dụng thêm các thư viện hữu ích từ ngôn ngữ này.



Cũng giống như nhiều ngôn ngữ lập trình khác, Kotlin chạy trên Java Virtual Machine (JVM) và sử dụng các công cụ và thư viện sẵn có trên Java.

*Kotlin có gì vượt trội hơn Java?*

Java có thể được xem là ngôn ngữ lập trình xây dựng các ứng dụng chạy trên hầu hết mọi hệ điều hành, với mọi phần cứng. Java vô cùng mạnh mẽ, bạn có thể sử dụng Java để viết các chương trình thực hiện nhiều thứ khác nhau, như trò chơi điện tử, các tiện ích công nghệ, điều khiển phần cứng hoặc bất cứ thứ gì khác. Đó cũng là ngôn ngữ lập trình chính để phát triển các ứng dụng trên hệ điều hành Android.

Java nhìn chung là rất phức tạp. Và Kotlin ra đời để “loại bỏ” một số thành phần rườm rà, giúp chương trình trở nên ngắn gọn và rõ ràng hơn.

Mục đích chính của Kotlin là kế thừa những tốt đẹp và giải quyết những vấn đề mà Java gặp phải. Vì vậy câu hỏi "Kotlin hay Java?" đã xuất hiện rất nhiều trong cộng đồng phát triển Android kể từ sự kiện Google I/O khi mà Kotlin được gọi tên trong phần "IDE Android", trở thành ngôn ngữ được hỗ trợ số 1 trên nền tảng hệ điều hành này. Câu trả lời là code của Kotlin ngắn gọn và rõ ràng hơn Java, và các tệp của cả hai ngôn ngữ này có thể cùng tồn tại trong các ứng dụng Android. Do đó, Kotlin không chỉ hữu ích cho các ứng dụng mới mà còn mở rộng các ứng dụng Java hiện có.

Một số ưu điểm của Kotlin có thể kể đến như:

* Code ngắn gọn, giảm thiểu các đoạn không cần thiết, cấu trúc ngắn, trực quan, dễ học, dễ làm, dễ sử dụng, tiết kiệm thời gian. Theo ước tính của Ovechkin thì với việc sử dụng Kotlin, code base có thể được thu nhỏ hơn từ 15-25%.
* An toàn, tránh lỗi NullPointerException. NullPointerException xảy ra khi code tham chiếu đến một biến, nhưng biến lại không có giá trị khi ứng dụng đang chạy. Kết quả là ứng dụng sẽ không có bất kỳ tùy chọn nào ngoại trừ việc báo lỗi. Và tất nhiên, với Kotlin, đảm bảo các nhà phát triển sẽ không gặp phải vấn đề này.
* Tool làm việc thân thiện, có thể lựa chọn các Java IDE để làm việc hoặc làm việc với command line

Tất cả những điều trên chính là lý do tại sao nhà phát triển yêu rất thích Kotlin. Code nhanh hơn, ít quy tắc hơn và đơn giản hơn so với Java. Bạn có thể đọc code mà chính bạn hoặc người khác viết một cách dễ chịu hơn hơn. Vẫn là code, nhưng không còn quá phức tạp đau đầu.

1. **Tìm hiểu Room Database**

Room database được phát triển và cải tiến từ sqlite. Room database giúp đơn giản hoá việc code,và giảm thiểu các công đoạn liên quan đến cơ sở dữ liệu.

Bản chất Room database là abstract layer gồm cơ sở dữ liệu chuẩn SQLite được Android thông qua.

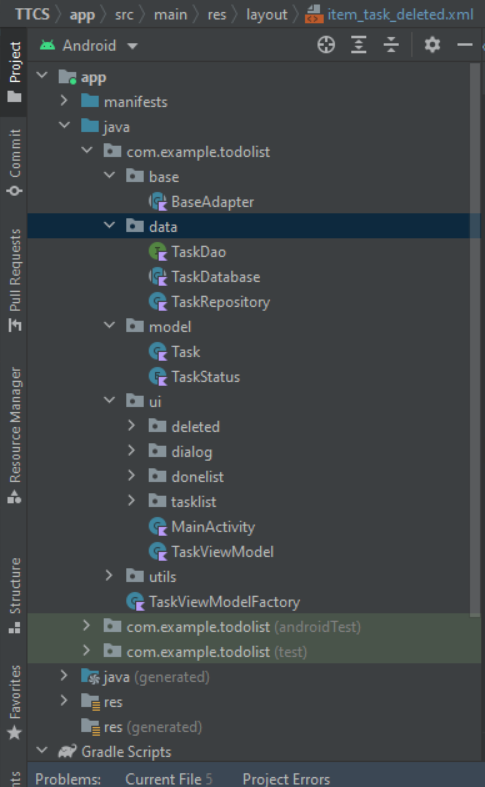
Với 3 thành phần chính là: Database, DAO(Data Access Object) và entity. Mỗi thành phần đều có nhiệm vụ và chức năng riêng.

1. **Xây dựng ứng dụng**
   1. Thiết kế giao diện ứng dụng:

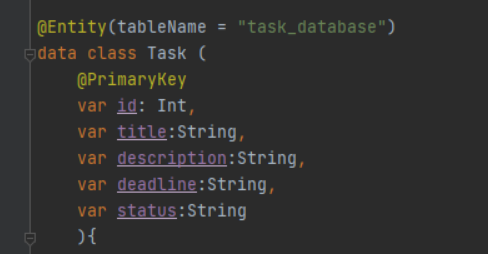
* Công cụ: Figma
* Kết quả: https://www.figma.com/file/YvZDJOoi1D1Vtj4CQZ4RrJ/Untitled?node-id=0%3A1



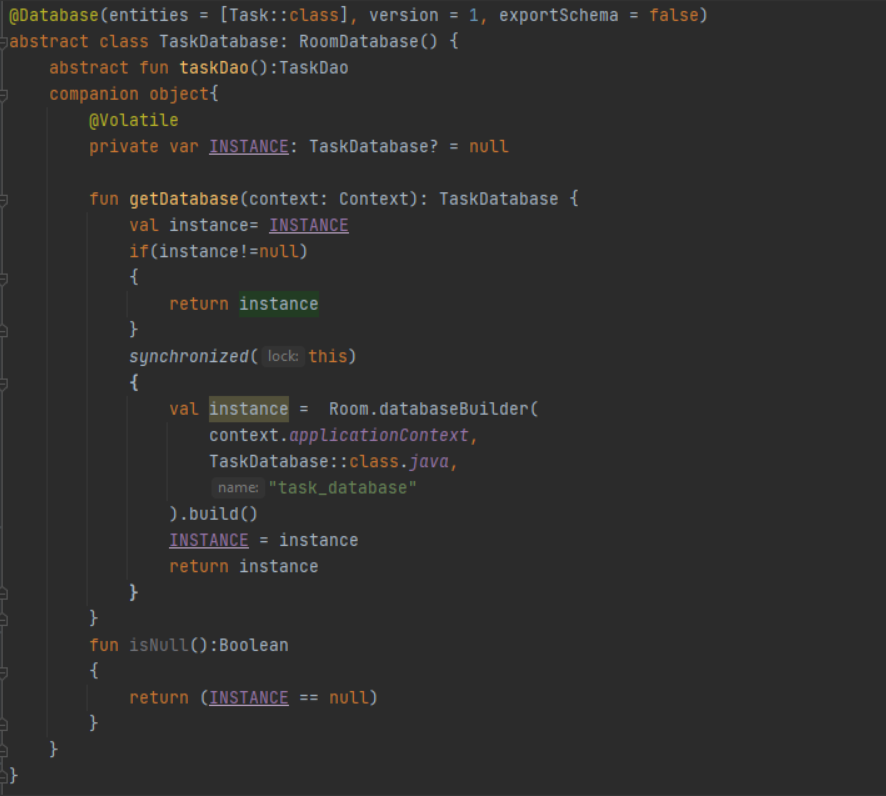
* 1. Tạo project mới và xây dựng cây thư mục trên Android Studio:



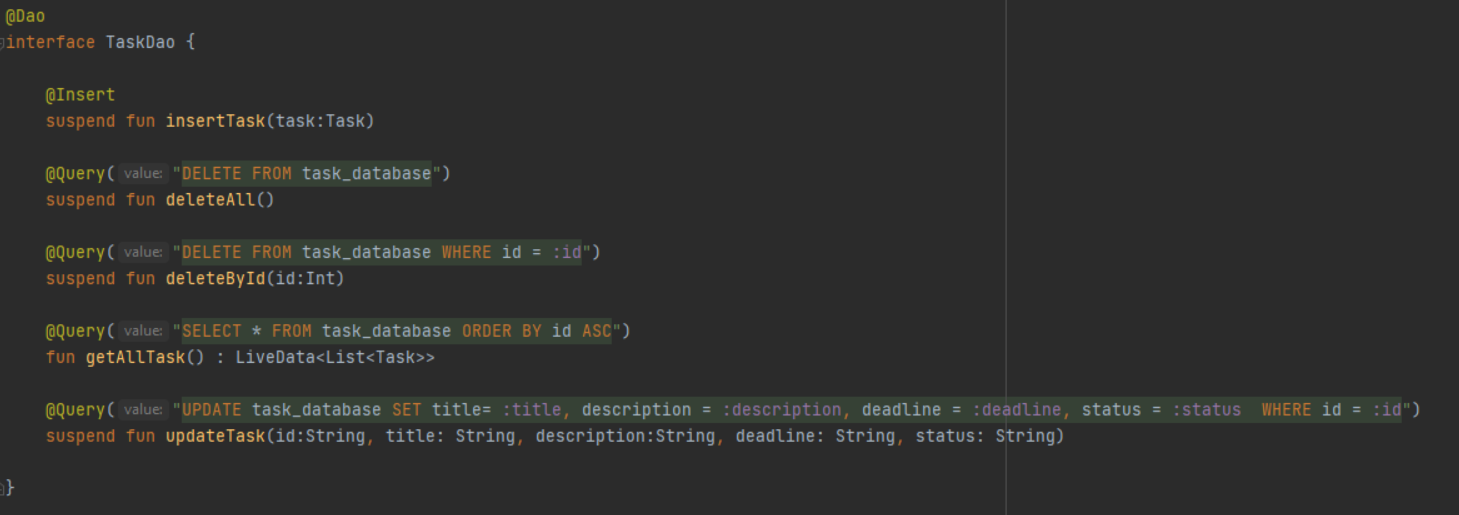
* 1. Xây dựng database:
* Bảng Task:

****

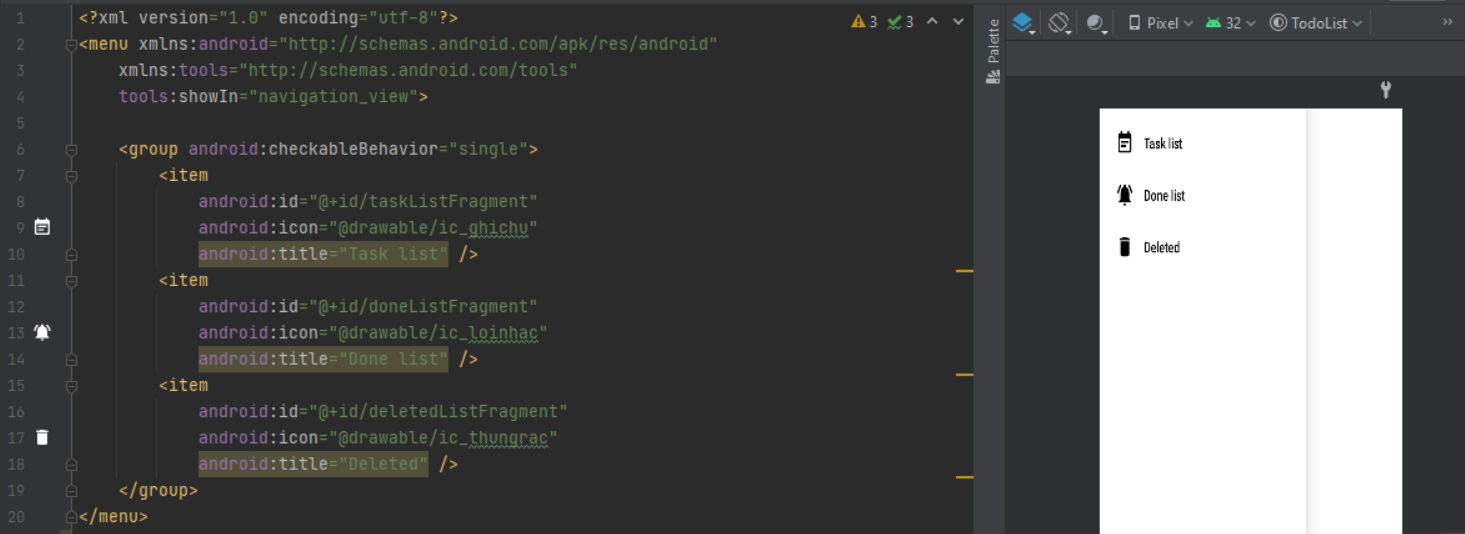
* Thiết lập database:

****

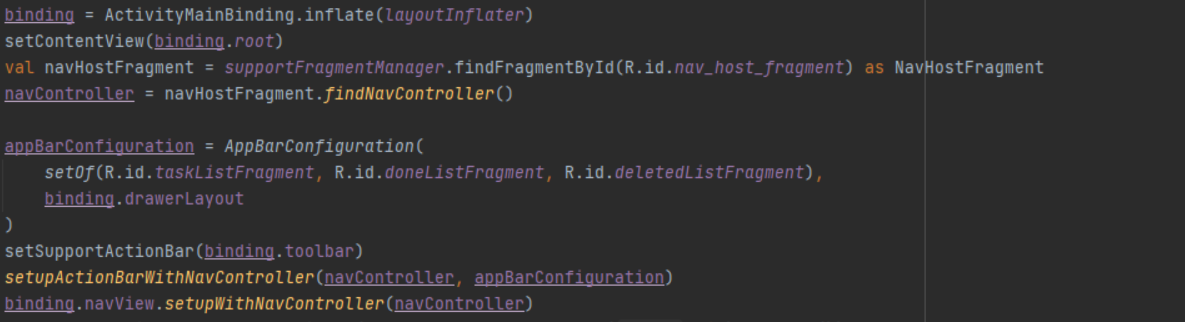
* Ta có 1 class TaskDao chịu trách nhiệm cho việc thao tác với database:

****

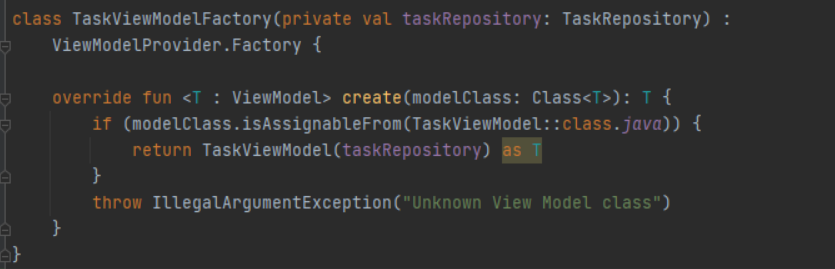
* 1. Tạo Drawer Navigator:
* Tạo 1 menu

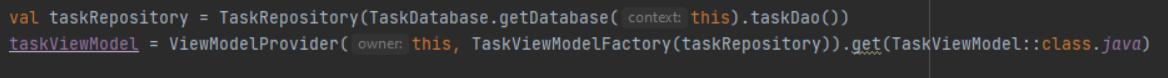
****

* Cấu hình ở MainActivity

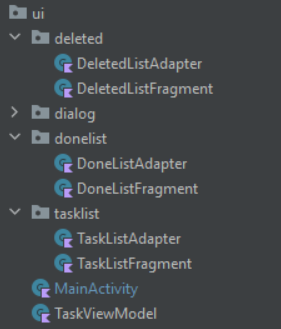


* 1. Tạo ViewModel:
* 3 màn hình trong ứng dụng có dữ liệu chung nên sẽ sử dụng viewmodel của Activity để dễ sàng trong việc xử lý

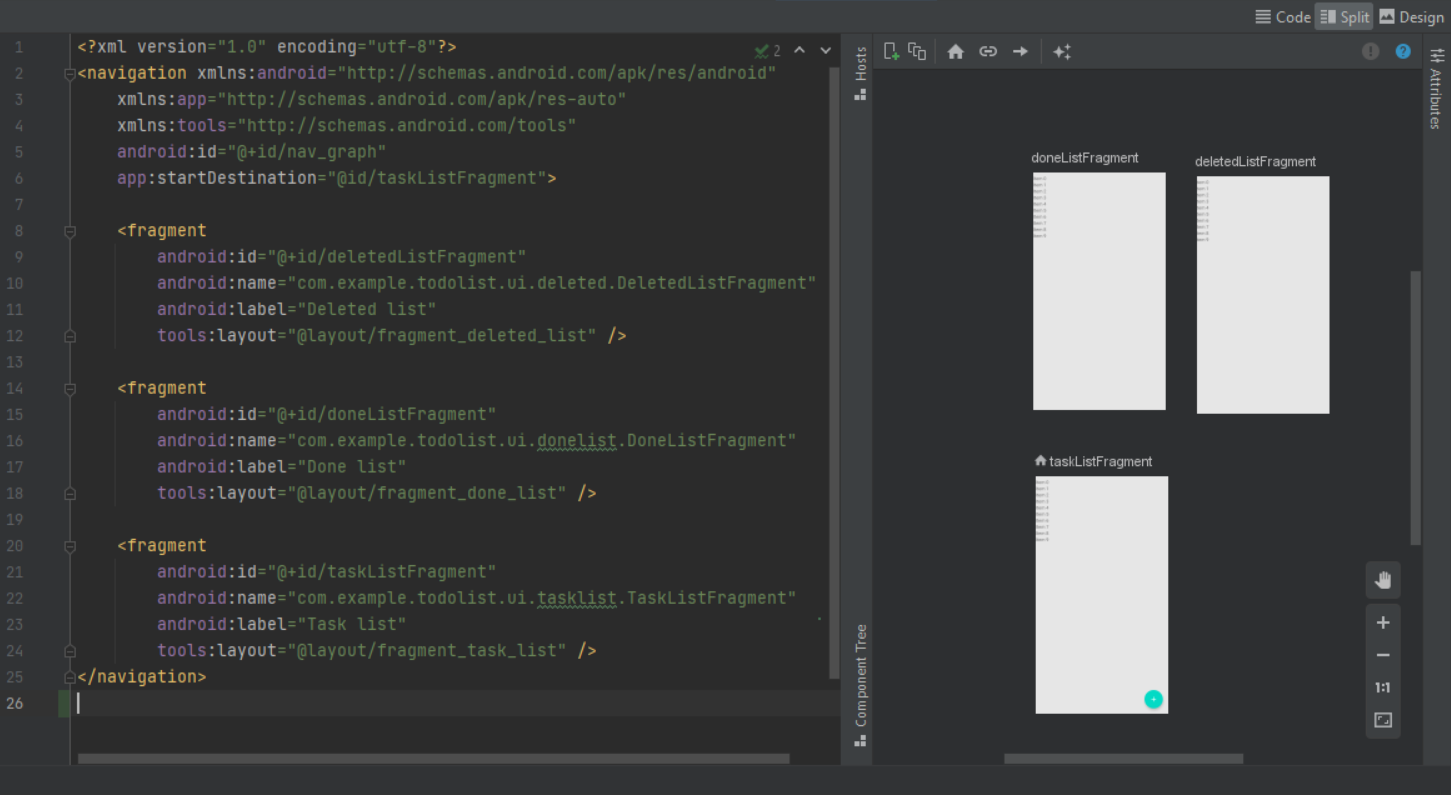


****

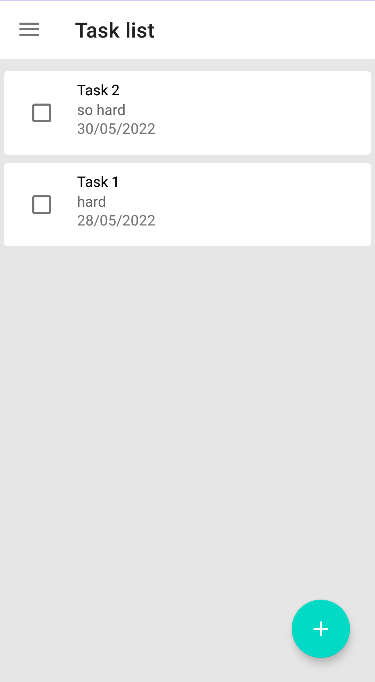
* 1. Tạo các fragment để hiển thị data từ viewmodel và hiển thị lên:



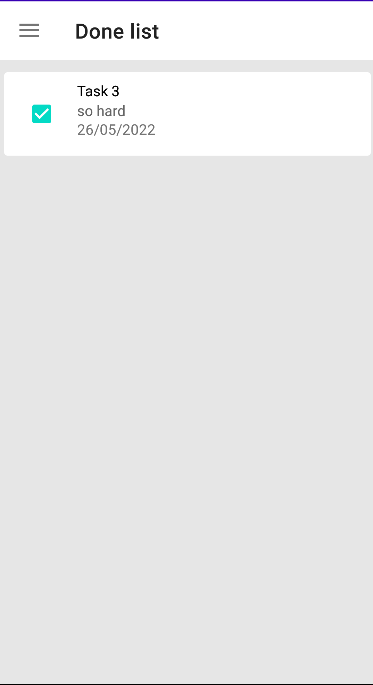
* 1. Tạo navigation để chuyển giữa các fragment:



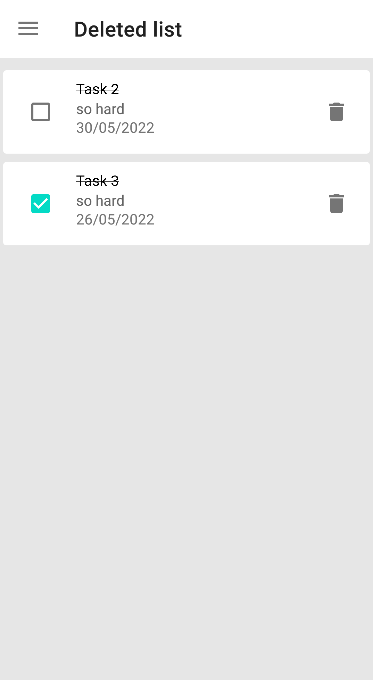
1. **Kết quả**
2. **TaskFragment**



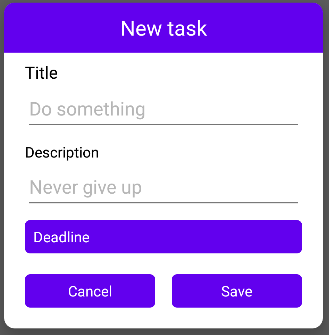
1. **DoneFragment**



1. **DeletedFragment**



1. **AddNewTaskDialog**



**TỔNG KẾT**

Project đã thực hiện các nhiệm vụ sau:

* Tìm hiểu Android.
* Tìm hiểu Android Studio.
* Tìm hiểu Git.
* Tìm hiểu ngôn ngữ Kotlin.
* Tìm hiểu Room Database.
* Tìm hiểu MVVM và Databinding.
* Nắm bắt quy trình làm một phần mềm Android.

Trong tương lại, em mong muốn ứng dụng phát triển hơn nữa về giao diện, chức năng (công việc trong ngày, theo tuần tháng, thông báo khi đến deadline,… ), và trở nên phổ biến để giúp mọi người quản lý công việc của mình.

**TÀI LIỆU THAM KHẢO**

[1] https://kotlinlang.org/

[2] https://product.hubspot.com/blog/git-and-github-tutorial-for-beginners

[3] https://developer.android.com/

[4] https://viblo.asia/p/tim-hieu-ve-mo-hinh-mvvm-maGK7vW95j2

[5] https://viblo.asia/p/databinding-trong-android-phan-1-3Q75wpxJKWb