PROMPT PROYECTO – RETOS PERSONALES

Contexto del proyecto

Estoy desarrollando una app Android en Java, en Android Studio, llamada *Retos Personales*.

La app permitirá a los usuarios registrar, gestionar y completar retos personales diarios o semanales, mostrando el progreso en tiempo real y guardando los datos de forma persistente.

Objetivo claro

Necesito que la aplicación permita:

1. Crear, editar y eliminar retos.

2. Marcar retos como completados.

3. Mostrar listas actualizadas de retos pendientes y completados automáticamente.

Detalles técnicos

- Lenguaje: Java

- Arquitectura: MVVM

- Librerías: Room (persistencia), RecyclerView (listas dinámicas), LiveData (observables)

- Patrones: Observer (para actualización automática de UI), Singleton (para la base de datos)

- Permisos: Acceso a almacenamiento local

- Restricciones: Compatible con Android 8.0+ (API 26+) y diferentes resoluciones de pantalla.

Entradas y Salidas

- Entradas: Nombre del reto, descripción, fecha límite, estado (pendiente/completado)

- Salidas: Lista de retos actualizada, notificaciones de logros completados

Formato esperado

- Clases Java completas para:

  - Modelo `Reto`

  - DAO y Database de Room

  - ViewModel

  - Adapter para RecyclerView

  - Fragmento de UI para agregar/editar retos

- Comentarios claros explicando cada función y manejo de errores.

Extras opcionales

- Validar campos vacíos y fechas incorrectas

- Manejo de excepciones en la base de datos

- Código limpio y organizado

- Futuras mejoras: filtros por categoría, estadísticas, recordatorios, sincronización en la nube

Instrucciones para Copilot

1. Divide el desarrollo en pasos: modelo → DAO → Database → ViewModel → Adapter → UI.

2. Sé específico con validaciones, UI mínima y librerías.

3. Evita ambigüedades y pide funciones concretas.

4. Itera y corrige el prompt según las respuestas de Copilot.

5. Revisa que el código funcione antes de usarlo en producción.

Ejemplo de Prompt Adaptado

> Necesito un fragmento de código que registre un reto con nombre, descripción, fecha límite y estado.

> Debe usarse Room como base de datos local.

> El formulario tendrá campos de texto para nombre y descripción, un selector de fecha y un botón Guardar.

> Valida que los campos no estén vacíos y que la fecha sea válida.

> Genera la clase `Reto`, DAO, Database, ViewModel, Adapter y un ejemplo de inserción con comentarios claros.

PREGUNTAS.

1. ¿Qué quiero que haga mi aplicación?
Quiero que mi aplicación me permita registrar, gestionar y completar retos personales diarios o semanales, y que pueda ver de manera clara mi progreso con todos los retos guardados de forma persistente en el dispositivo.

2. ¿Cuál es la funcionalidad clave que necesito?
La funcionalidad principal que necesito es poder crear, editar, eliminar y marcar los retos como completados, mostrando siempre las listas actualizadas de retos pendientes y completados.

3. ¿Qué sí debe resolver y qué no?
Sí debe permitirme gestionar mis retos personales de manera completa y mantenerlos guardados localmente.
No necesita tener conexión a internet, sincronización en la nube ni análisis avanzado de datos por ahora.

4. ¿Quién usará esta funcionalidad y en qué escenario real?
Yo la usaré principalmente para organizar mis metas y retos diarios o semanales. También puede ser útil para cualquier persona que quiera llevar un control de sus objetivos personales.

5. ¿Qué datos necesito recibir y qué resultados espero devolver o mostrar?
Necesito ingresar el nombre del reto, una breve descripción, la fecha límite y el estado (pendiente o completado).
Espero que la app me muestre una lista organizada de mis retos, indicando cuáles están pendientes y cuáles completados, y me notifique cuando haya logros completados.

6. ¿Debe hacerse en Java (Android Studio)? ¿Requiere conexión a internet, base de datos, API externa?
Sí, quiero que se haga en Java usando Android Studio.
La aplicación debe usar una base de datos local (Room) para guardar los retos, y no necesita internet ni APIs externas en esta versión inicial.

7. ¿Cómo debería mostrarse en pantalla (UI/UX)?
La app debería mostrar los retos en listas separadas para pendientes y completados usando RecyclerView.
También debería tener botones claros para agregar, editar y marcar retos como completados, con una interfaz limpia y fácil de usar.

8. ¿Qué validaciones y manejo de errores necesito?
Quiero que la app valide que los campos de nombre y descripción no estén vacíos, que la fecha límite sea correcta y que no ocurra ningún error al guardar,

editar o eliminar un reto.
Si ocurre algún error, que se muestre un mensaje claro al usuario.

9. ¿Será algo pequeño o crecerá a futuro?
Por ahora es un proyecto pequeño, pero quiero que tenga la posibilidad de crecer con funciones futuras como recordatorios, estadísticas, filtros por categoría y sincronización en la nube.