Preguntas Importantes Antes de Pedir Código (ADAPTIA)

• ¿Qué quiero que haga mi aplicación?

Gestionar y monitorear rutinas de entrenamiento personalizadas organizadas por deportes (fútbol, natación, etc.) que contienen múltiples ejercicios con series, repeticiones y tiempos de descanso. Al marcar una rutina como completada, debe **integrarse automáticamente** con sistemas externos para enviar notificaciones motivacionales.

• ¿Cuál es la funcionalidad clave que necesito?

La funcionalidad clave es el **seguimiento del estado de completado de las rutinas**. Específicamente, la capacidad de los usuarios para **marcar rutinas como completadas**, lo cual desencadena el envío de una petición **HTTP POST** (usando HttpURLConnection en un hilo secundario) a un *webhook* de n8n para enviar notificaciones a Telegram.

¿Qué sí debe resolver y qué no?

- **Debe Resolver:**
  - **Registro y persistencia local** de rutinas y su estado de completado.
  - **Visualización** de rutinas y ejercicios usando RecyclerView.
  - **Integración con sistemas externos** (n8n/Telegram) mediante HttpURLConnection en un hilo secundario.
  - Manejo robusto de errores de conexión (*try-catch*, *timeouts*, validación de códigos HTTP).
- **No Debe Resolver:**
  - Sustituir entrenadores profesionales.
  - Ofrecer diagnósticos médicos.

• ¿Quién usará esta funcionalidad y en qué escenario real?

Usuarios que buscan **registrar, seguir y motivar** su progreso deportivo personal desde casa o el gimnasio. El proyecto también sirve como **demostración universitaria** de integración entre Android, *webhooks*, n8n y Telegram.

• ¿Qué datos necesito recibir y qué resultados espero devolver o mostrar?

- **Entradas:** El usuario proporciona los datos de la rutina (nombre, deporte, descripción) y la acción de **marcar una rutina como completada**.
- **Salidas:**
  - Visualización del detalle de la rutina y ejercicios en DetalleRutinaActivity.

    - **Notificaciones motivacionales automáticas** enviadas a Telegram.
    - **Retroalimentación visual** al usuario mediante Toast messages diferenciados (éxito, error de conexión, o sin internet).

¿Debe hacerse en Java (Android Studio)? ¿Requiere conexión a internet, base de datos, API externa?

- **Plataforma/Lenguaje:** Sí, la aplicación es **Android** y está desarrollada en **Java**.
- **Conexión/BD/API:**
    - **Conexión a internet: Requerida** para enviar la petición HTTP POST al *webhook* de n8n.
    - **Base de datos:** Utiliza **persistencia local de datos** a través de RutinaManager.
    - **API Externa:** Depende del **webhook de n8n** alojado en la nube para la automatización de notificaciones a Telegram.

¿Cómo debería mostrarse en pantalla (UI/UX)?

La interfaz principal es **DetalleRutinaActivity**, la cual muestra el nombre, deporte, descripción completa y la lista de ejercicios (en un RecyclerView). Incluye botones para **iniciar la rutina**, **marcarla como completada** y volver atrás.

• ¿Qué validaciones y manejo de errores necesito?

- **Manejo de Errores:** Implementación de **try-catch**, **timeouts de conexión**, **validación de códigos de respuesta HTTP**, y retroalimentación visual al usuario con Toast messages.
- **Depuración:** Uso de *logs* en **Logcat** con la etiqueta **N8N_LOGRO**.

• ¿Será algo pequeño o crecerá a futuro?

Inicialmente, es una aplicación funcional con la integración n8n/Telegram para una demostración universitaria.

A futuro podría crecer con:

- Generación de **reportes PDF con análisis de progreso mediante Gemini AI**.
- Sincronización en la nube.
- Sistema de gamificación con logros y *badges*.
- Integración con *wearables*.
- Exportación a Google Fit.
- Compartir rutinas entre usuarios.

PROMPT ESTRUCTURADO PARA GITHUB

1. CONTEXTO Y OBJETIVO

Contexto: Una aplicación de fitness para Android enfocada en la gestión y seguimiento de rutinas de entrenamiento. **Objetivo:** Permitir a los usuarios registrar rutinas y utilizar la finalización de una rutina para desencadenar una automatización externa (n8n a Telegram) con fines motivacionales.

2. REQUISITOS FUNCIONALES

Entradas: Datos de rutinas (deporte, descripción, ejercicios, series, repeticiones) y la acción del usuario de marcar una rutina como completada. **Procesos:** Persistencia local de datos de rutina con RutinaManager, y envío de petición **HTTP POST** al *webhook* de n8n en un hilo secundario (HttpURLConnection). **Salidas:** Visualización de rutinas en DetalleRutinaActivity con RecyclerView, notificaciones motivacionales automáticas a Telegram, y mensajes *Toast* de retroalimentación de conexión.

3. REQUISITOS TÉCNICOS

Plataforma: Android (Android Studio). **Lenguaje:** Java. **APIs/Librerías:** HttpURLConnection (para la petición POST), RecyclerView (para la lista de ejercicios). **Permisos:** Permiso de Internet.

4. ESTRUCTURA DE LA SOLUCIÓN

Arquitectura: Modular (para facilitar la escalabilidad futura). **Componentes principales:**

- DetalleRutinaActivity (pantalla principal de la rutina).
- Adaptador personalizado para RecyclerView (ejercicios).
- RutinaManager (servicio de persistencia local).
- Hilo secundario para la comunicación HTTP. **Patrones de diseño:** N/A explícitamente mencionado, pero se sigue un diseño modular.

5. DETALLES DE IMPLEMENTACIÓN

Validaciones: Validación del código de respuesta HTTP y del estado de conexión a Internet. **Manejo de errores:** Uso de try-catch, *timeouts* de conexión, y *Toast messages* diferenciados. **Estado de la app:** Persistencia del estado de completado de las rutinas. **Depuración:** Uso de *logs* en Logcat con etiqueta N8N_LOGRO.

6. CONSIDERACIONES ESPECÍFICAS

Escalabilidad: Diseñado para futura integración con **Gemini AI** (reportes PDF), sincronización en la nube y *wearables*. **Propósito:** Demostración universitaria de integración tecnológica, sin ofrecer diagnósticos médicos. **Conexión:** La integración n8n requiere que el *webhook* esté **alojado en la nube**.