¿Qué quiero que haga mi aplicación?
Una aplicación Android que permita crear registros de pacientes en un centro médico, almacenando:

- Datos personales (nombre, edad, sexo, contacto, antecedentes médicos).
- Historial de consultas, diagnósticos y tratamientos.
- Archivos médicos (resultados de laboratorio, imágenes médicas, recetas).
- Sincronización entre una base de datos local y un servidor remoto, para que la información esté disponible incluso sin conexión.

¿Cuál es la funcionalidad clave que necesito?

- Crear, leer, actualizar y eliminar registros médicos de pacientes (CRUD).
- Adjuntar resultados médicos (archivos o fotos).
- Sincronización automática cuando haya internet.
- Seguridad y privacidad de la información.

¿Qué sí debe resolver y qué no?

- Sí: manejo de expedientes médicos, almacenamiento seguro, historial de atenciones.
- No: no sustituye al diagnóstico médico ni integra dispositivos médicos especializados (como ECG en tiempo real).

¿Quién usará esta funcionalidad y en qué escenario real?

- Personal autorizado de un centro médico (médicos, enfermeros, administrativos).
- Escenario: atención en clínicas, hospitales o consultorios.

¿Qué datos necesito recibir y qué resultados espero mostrar?

- **Entradas:** datos del paciente, historial clínico, archivos adjuntos (laboratorios, imágenes).
- **Salidas:** expediente médico digital, historial de visitas, reportes de cada paciente.

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

- Sí, en Java con Android Studio.
- Debe funcionar offline con base local (Room) y sincronizar con una base remota mediante API (Retrofit).

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

- Pantalla de inicio con menú principal.
- Pantalla para registrar paciente.
- Pantalla de lista de pacientes con opción de búsqueda.

- Pantalla de detalles del paciente (historial médico y adjuntos).

¿Qué validaciones y manejo de errores necesito?

- Validar que los datos obligatorios no estén vacíos.
- Validar formato de edad, correo o teléfono.
- Evitar registros duplicados del mismo paciente.
- Mensajes de error claros al usuario.

¿Será algo pequeño o crecerá a futuro?

- Inicia como MVP en una clínica pequeña.
- Puede escalar a hospitales y múltiples dispositivos con sincronización.

COMPLEMENTO

1. Comprender el Problema

¿Qué problema necesito resolver?
Evitar el uso de expedientes médicos en papel, centralizar la información clínica de los pacientes y permitir un acceso rápido y seguro a su historial.

¿Cuál es el objetivo principal de la funcionalidad?
Crear un **sistema de expediente médico digital** que:

- Guarde datos del paciente.
- Permita registrar consultas y tratamientos.
- Almacene documentos médicos digitalizados.
- Sincronice con un servidor para respaldo.

2. Definir el Alcance

¿Qué debe hacer exactamente la aplicación?

- Registrar pacientes con sus datos personales y antecedentes médicos.
- Guardar consultas médicas (fecha, síntomas, diagnóstico, tratamiento).
- Adjuntar documentos (recetas, exámenes, imágenes).
- Generar reportes de historial médico.
- Funcionar offline y sincronizar cuando haya internet.

Casos de uso principales:

1. Registrar un nuevo paciente.

2. Buscar paciente en la base de datos.
3. Editar o actualizar información del paciente.
4. Registrar una nueva consulta médica.
5. Adjuntar resultados médicos (PDF, fotos).
6. Ver historial médico completo.
7. Sincronizar datos con el servidor.

Usuarios/escenarios:

- 1 usuario administrativo (recepción).
- Médicos y enfermeros que acceden a los expedientes.
- Pacientes indirectamente (cuando el médico consulta su expediente).

3. Entender a los Usuarios

¿Quién va a usar la app?

- Médicos, enfermeras y personal autorizado.

Nivel técnico de los usuarios:

- Manejo básico de dispositivos móviles.

¿Qué esperan obtener?

- Expediente médico digital, rápido de consultar y seguro.

¿En qué contexto usarán la app?

- En clínicas, hospitales y consultorios.
- En entornos con y sin conexión a internet.

4. Requisitos Funcionales

Entradas del sistema:

- Texto: nombre, edad, sexo, contacto, antecedentes médicos.
- Selecciones: diagnóstico, estado del paciente, medicamentos.
- Archivos: imágenes médicas, PDFs de exámenes, recetas.
- Fecha/hora de consultas.

Salidas:

- Listado de pacientes registrados.
- Historial clínico de cada paciente.

- Reportes médicos exportables (PDF).

Procesos principales:

1. Crear paciente.
2. Actualizar información personal o médica.
3. Registrar consulta (diagnóstico, tratamiento).
4. Adjuntar y consultar documentos médicos.
5. Sincronización con servidor remoto.

Acciones del usuario:

- Crear, leer, actualizar, eliminar registros médicos.
- Subir y descargar archivos médicos.
- Buscar paciente por nombre o ID.
- Ver historial médico.

5. Requisitos No Funcionales

- **Rendimiento:** debe funcionar en móviles estándar sin lentitud.
- **Seguridad:** datos médicos cifrados (privacidad sensible).
- **Escalabilidad:** comenzar en una clínica, escalar a hospital.
- **Disponibilidad offline:** trabajar sin internet y sincronizar luego.

6. Contexto Técnico

- **Versión Android mínima:** API 24 (Nougat 7.0) para compatibilidad amplia.
- **Lenguaje:** Java.
- **Permisos requeridos:**
    - android.permission.CAMERA (fotos de exámenes médicos).
    - android.permission.READ_EXTERNAL_STORAGE.
    - android.permission.WRITE_EXTERNAL_STORAGE.
    - android.permission.INTERNET.

Librerías y frameworks:

- Persistencia: androidx.room:room-runtime.
- Cámara: androidx.camera:camera-core.
- Sincronización: com.squareup.retrofit2:retrofit.

- Seguridad: javax.crypto para cifrado de datos médicos.

7. Consideraciones de Diseño

- **UI/UX:** Material Design, navegación clara (Inicio → Nuevo paciente / Lista pacientes / Historial).
- **Patrones de navegación:** menú principal y subpantallas.
- **Compatibilidad:** usar ConstraintLayout para pantallas responsivas.
- **Estilo visual:** interfaz limpia y profesional, priorizando legibilidad.

8. Estructura de la Solución

Arquitectura: MVVM (Model-View-ViewModel).

- **Model:**
    - Paciente (Entidad Room con datos personales y médicos).
    - Consulta (Entidad para historial de visitas).
    - Documento (Entidad para adjuntos médicos).
    - DAO y AppDatabase.
- **View:**
    - RegistroPacienteActivity.
    - ListaPacientesActivity (RecyclerView).
    - DetallePacienteActivity (expediente médico).
- **ViewModel:**
    - PacienteViewModel.
    - ConsultaViewModel.
- **Otros componentes:**
    - SyncWorker (WorkManager para sincronización en segundo plano).
    - PacienteRepository (manejo de datos entre local y remoto).

9. Detalles de Implementación

Validaciones:

- Todos los campos básicos obligatorios.
- Edad numérica válida.
- Evitar pacientes duplicados.
- Verificar que se carguen archivos correctamente.

Manejo de errores:

- Mensajes claros en caso de fallar guardado, permisos o sincronización.
- Retries automáticos de sincronización en caso de fallo de red.

Persistencia:

- Room Database para almacenamiento local.
- Cifrado de datos médicos antes de guardarlos.

10. Consideraciones Específicas

- **Seguridad:**
    - Cifrado AES con javax.crypto.
    - Sincronización vía HTTPS con autenticación.
- **UI/UX:**
    - Botones claros para nuevo paciente, consulta e historial.
    - Pantallas minimalistas para no saturar al usuario.
- **Testing:**
    - Pruebas unitarias sobre ViewModel y Repository.
    - Simulación de fallos de red y permisos.