**✅ Resumen de VenceYa**

**✨ Funcionalidad principal:**

Una app para **agendar recordatorios de vencimientos importantes**, con notificaciones push, categorías, frecuencia y un dashboard claro.

**🎯 Alcance:**

* Login con Google o email (Firebase Auth)
* Dashboard con lista de recordatorios
* Alta de recordatorios (título, fecha, categoría, frecuencia, switch de notificación)
* Ver detalle con opción de editar/eliminar
* Notificaciones push (Firebase Messaging)
* Mini sistema de gamificación (racha de uso y puntos)

**🔒 Límites:**

* No es app bancaria, no hay integración con pagos ni finanzas reales.
* Solo recordatorios personales, sin sincronización con calendarios externos (aunque podrías agregarlo como plus).
* Se enfoca en UX simple y mobile-only (sin versión web por ahora).

**🚀 Mejor forma de encarar este proyecto (RÁPIDO + LIMPIO)**

**📦 1. Crea el proyecto en local (VS Code o Android Studio)**

Ventajas:

* Control total
* Puedes testear rápido con hot reload
* Mejor para probar Firebase, UI, y flujos sin necesidad de deploy

Después lo subís a GitHub y si querés escalar, conectás con Firebase Hosting, Cloud Functions, etc.

**🛠️ 2. Estructura inicial recomendada (folder system)**

plaintext

CopiarEditar

/lib

│

├── main.dart // punto de entrada

├── app.dart // MaterialApp y rutas

│

├── core/ // constantes, temas, helpers

│ ├── constants.dart

│ └── theme.dart

│

├── models/ // clases Reminder, User, etc.

│

├── services/ // Firebase, auth, notificaciones

│ ├── auth\_service.dart

│ ├── firestore\_service.dart

│ └── notification\_service.dart

│

├── ui/

│ ├── screens/ // Pantallas principales

│ │ ├── splash\_screen.dart

│ │ ├── login\_screen.dart

│ │ ├── dashboard\_screen.dart

│ │ ├── add\_reminder\_screen.dart

│ │ └── detail\_screen.dart

│ ├── widgets/ // Componentes reutilizables

│ └── styles/ // colores, textos, paddings

│

└── config/ // configuración inicial

└── firebase\_options.dart

**🧱 3. Pasos para comenzar YA (sin vueltas)**

**✅ Paso 1: Inicializar Flutter + Firebase**

bash

CopiarEditar

flutter create venceya

cd venceya

flutterfire configure

**✅ Paso 2: Instalar paquetes clave**

yaml

CopiarEditar

# pubspec.yaml

dependencies:

firebase\_core:

firebase\_auth:

cloud\_firestore:

firebase\_messaging:

provider:

flutter\_local\_notifications:

intl:

**✅ Paso 3: Setear Firebase en main**

dart

CopiarEditar

void main() async {

WidgetsFlutterBinding.ensureInitialized();

await Firebase.initializeApp();

runApp(const MyApp());

}

**✅ Paso 4: Crea los modelos base**

* Reminder
* UserData
* NotificationRecord

**✅ Paso 5: Ruteo con Navigator 2.0 o GoRouter (para crecer limpio)**

**🎨 4. Luego comenzá las pantallas (reutilizables)**

1. **Splash + Auth Check**
2. **Login (Firebase Auth)**
3. **Dashboard (lista + botón + filtros)**
4. **Agregar recordatorio (formulario)**
5. **Detalle (info + editar + eliminar)**

Usá Provider para manejar estado (rápido, limpio, escalable).

**🧠 Como senior te digo:**

* Empieza simple y **modular** cada funcionalidad
* Usa Firestore con colección reminders y users/{uid}/reminders si querés escalar por usuario
* Las **notificaciones** usalas con firebase\_messaging + flutter\_local\_notifications para background y foreground
* Documenta desde el principio: qué hace cada servicio, cada widget

¡Excelente idea, Fabricio! Un buen resumen nos ayudará a mantenernos orientados en la estructura y el progreso.

Hasta el momento, hemos construido una base sólida para tu aplicación "VenceYa" con Flutter y Firebase. Aquí tienes un breve resumen:

**Recapitulacón de "VenceYa" hasta la fecha:**

1. **Configuración del Proyecto (pubspec.yaml):**
   * Instalamos y configuramos todas las dependencias clave (Firebase, freezed, go\_router, provider, notificaciones locales, timezone, intl, Google Sign-In).
   * Declaramos el asset de tu logo (venceya\_logo.png).
   * (Opcional, si lo hiciste) Configuramos la fuente "Inter" para un estilo moderno.
2. **Estructura de Carpetas (lib/):**
   * Organizamos el proyecto en carpetas modulares: core/, models/, services/, ui/, config/.
   * **Importante:** Hemos corregido la ubicación de las pantallas, ahora están correctamente en lib/ui/screens/.
   * Archivos principales en lib/: main.dart, app.dart, app\_router.dart, firebase\_options.dart.
3. **Modelos de Datos (lib/models/):**
   * Creamos reminder.dart (para los recordatorios) y user\_data.dart (para el perfil y gamificación del usuario).
   * Usamos freezed para generar automáticamente el código (.freezed.dart y .g.dart) para que estos modelos sean inmutables y se puedan guardar/leer fácilmente de Firestore.
4. **Servicios Principales (lib/services/):**
   * auth\_service.dart: Maneja toda la lógica de autenticación (login, registro, Google Sign-In).
   * firestore\_service.dart: Interactúa con la base de datos Firestore (agregar, actualizar, eliminar, obtener recordatorios y datos de usuario).
   * local\_notification\_service.dart: Se encarga de programar y mostrar las notificaciones locales en tu dispositivo Android.
5. **Tema Global de la Aplicación (lib/core/theme.dart):**
   * Creamos theme.dart con una paleta de colores moderna y limpia (azules, grises, blancos) inspirada en tus diseños.
   * Definimos estilos consistentes para botones, campos de texto, tarjetas y tipografía (usando la fuente "Inter" si la configuraste).
   * Este tema se aplica globalmente en app.dart.
6. **Navegación de la Aplicación (lib/app\_router.dart):**
   * Configuramos go\_router para definir todas las rutas de la aplicación (/splash, /login, /signup, /reset-password, /dashboard, etc.).
   * Implementamos la lógica de redirección: si estás logueado te envía al dashboard, si no, al login (excepto la splash screen que tiene su propia lógica de temporizador).
7. **Punto de Entrada y Configuración General (main.dart y app.dart):**
   * main.dart: Inicializa Firebase, configura el manejador de notificaciones en segundo plano, crea las instancias de los servicios y las pone a disposición de toda la app usando MultiProvider.
   * app.dart: Es el widget raíz que usa tu app\_router.dart y aplica el AppTheme a toda la interfaz.
8. **Pantallas de UI (Iniciales, lib/ui/screens/):**
   * splash\_screen.dart: La pantalla de carga inicial con tu logo, que espera unos segundos y luego redirige según el estado de autenticación. Ya usa tu nuevo tema.
   * login\_screen.dart: La pantalla de inicio de sesión, con campos para email/contraseña y un botón para Google Sign-In. Incluye botones para navegar a las futuras pantallas de registro y restablecer contraseña, y ya usa tu nuevo tema.
9. **Configuración en la Consola de Firebase:**
   * Habilitamos los métodos de autenticación (Email/Contraseña y Google).
   * Establecimos las Reglas de Seguridad de Firestore para proteger los datos de los usuarios.

¡Eso es todo lo que tenemos hasta ahora! La base está muy sólida y organizada.

**El siguiente paso es crear las pantallas que faltan para la autenticación:**

* **lib/ui/screens/signup\_screen.dart** (pantalla de registro).
* **lib/ui/screens/reset\_password\_screen.dart** (pantalla de restablecimiento de contraseña).