Diseño de solución

Objetivo: dejar definido un diseño técnico completo para que un equipo de desarrollo pueda implementar la app sin ambigüedades.

---

1) Tipo de sistema y alcance

- **Aplicación móvil** multiplataforma (Android/iOS).
- **Back-end gestionado** (BaaS) para acelerar desarrollo y reducir costes.
- **Consola ligera de administración** (web) para validación de recursos y moderación.
- **Integraciones externas**: mapas, notificaciones push y enlaces a canales oficiales de denuncias (Policía Nacional / Guardia Civil).

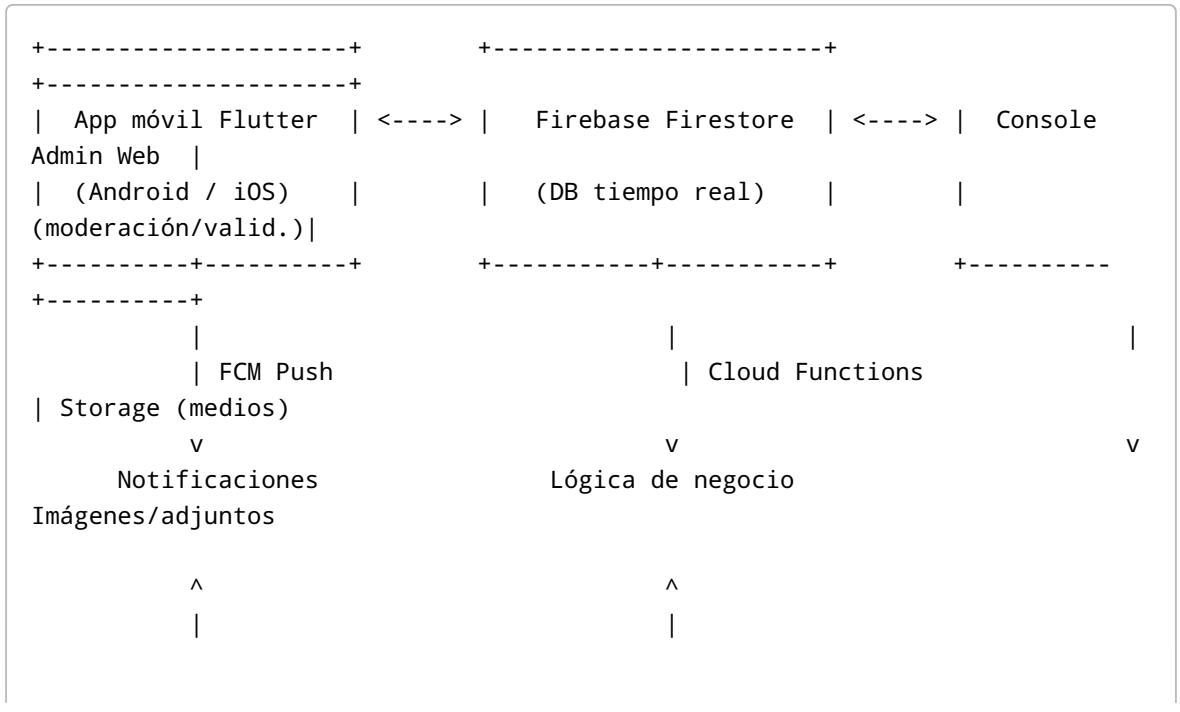
Elección base (costo 0€ / tiempo):

- **Frontend móvil:** Flutter (Dart). Alternativa: React Native (Expo).
- **Back-end:** Firebase (Auth, Firestore, Storage, Cloud Functions, FCM). Alternativa open-source: Supabase + Edge Functions.
- **Mapas:** MapLibre GL + tiles OSM mediante proveedor gratuito (ej. MapTiler plan free) o **Google Maps SDK** (cuota gratuita inicial).
- **Analítica básica:** Firebase Analytics (anonimizada).

Reglas: minimizar datos personales, anonimato por defecto para familias; datos públicos verificables para entidades (ONG/parroquias/centros).

---

2) Arquitectura general



```
        SDK Maps <---> Tiles OSM/Google     |      Enlaces externos
(Denuncias oficiales)
```

Módulos principales (app)

- **Autenticación y perfil** (familia/ONG).
- **Mapa de recursos** (ONG, parroquias, centros; validación previa).
- **Botón de pánico** (alerta a contactos de confianza vía push y/o deeplink).
- **Comercios solidarios** (contacto directo del comercio con entidades).
- **Denuncias anónimas** (enlace directo a webs oficiales; la app no almacena datos).
- **Espacio "Entre Familias"** (tablón + comentarios + mensajes directos opcionales).

---

3) Modelos de datos (Firestore — colecciones y campos)

> Convención: siempre campos mínimos, timestamps `createdAt/updatedAt`, `ownerId` cuando aplique.

- **users** (todas las cuentas)
- `uid` (string, auth id)
- `role` (enum: `family` | `org`)
- `email` (string, privado)
- `realName` (string, privado, solo para `family`)
- `publicName` (string, visible para `family` — seudónimo)
- `orgData` (obj, solo si `role=org`: nombre, tel, email público, web, dirección, geoPoint, descripción, `verified:boolean`)
- `dmEnabled` (boolean, mensajes directos permitidos)
- `trustedContacts` (array de `uid` para botón de pánico)
- `consents` (obj: TOS/privacidad/ubicación)

- **resources** (recursos del mapa: parroquias/ONG/centros)

- `name`, `type` (enum: `parroquia|ong|centro|otros`)
- `geo` (geoPoint), `address`
- `services` (array: alimentación, escucha, atención psicológica, etc.)
- `contact` (tel, email, web)
- `ownerOrgId` (uid)
- `status` (enum: `pending|approved|rejected`)

- **posts** (tablón "Entre Familias")

- `authorId` (uid — familia por defecto)
- `text` (string)
- `visibility` (enum: `public|anon` – mostrará `publicName` o “Anónimo”)
- `commentsCount` (int)

- **comments**

- `postId`, `authorId`, `text`

- **dmThreads** (mensajes privados)

- `memberIds` (array[2])
- `lastMessageAt`

- **dmMessages**

- `threadId`, `senderId`, `text`, `attachments[]`

- **panicAlerts**

- `ownerId`
- `location` (geoPoint)
- `status` (`sent|acked|closed`)
- `notifiedContacts[]` (uids)

- **donations** (comercios → entidades)

- `ownerUserId` (comercio)
- `items[]` (tipo, cantidad, caducidad aprox)
- `orgContactedIds[]` (uids)
- `status` (`open|taken|closed`)

Denuncias: no se almacena nada; es un enlace directo a canales oficiales.

---

4) Reglas de seguridad (Firestore)

- `users`: solo el propio usuario puede leer/escribir sus datos privados. Campos públicos: `publicName`, `orgData` de entidades verificadas.
- `resources`: lectura pública; escritura por `ownerOrgId` (entidad) y publicación solo si `status=approved` (moderación por Cloud Function).
- `posts/comments`: lectura pública; escritura por usuarios autenticados; filtro de lenguaje/longitud en Cloud Function.
- `dm*`: solo miembros del hilo pueden leer; tope de tamaño por mensaje; opción de bloqueo/denuncia.
- `panicAlerts`: solo creador y contactos notificados pueden leer; caducidad/auto-borrado (TTL).
- `donations`: lectura por entidades; escritura por “comercio” (usuario normal); sin datos personales.

---

5) Flujos clave (diagramas de secuencia)

5.1 Autenticación (familia anónima)

```
Usuario -> App: Registro email + contraseña + seudónimo
App -> Firebase Auth: createUser
App -> Firestore: users/{uid} (role=family, publicName=seudónimo)
App: Nunca muestra email/realName públicamente
```

5.2 Publicar en “Entre Familias”

```
Usuario -> App: Escribe post (modo anónimo o con seudónimo)
App -> Firestore: posts.add()
Moderación (CF): valida longitud/contenido; marca aprobado
Usuarios -> App: ven el post y comentan
```

5.3 Botón de pánico

```
Usuario -> App: Pulsar botón (confirmación antitoque)
App -> Geolocalización: obtener posición
App -> Firestore: panicAlerts.create
Cloud Function -> FCM: enviar push a trustedContacts[]
Contactos -> App: abrir alerta (mapa + opciones de llamada)
Usuario/Contacto -> App: cerrar alerta
```

5.4 Comercios solidarios → Entidades

```
Comercio (usuario normal) -> App: Crear oferta de donación
Entidades -> App: Visualizar y aceptar contacto
Comercio <-> Entidad: contacto directo (teléfono/email públicos de la
entidad)
```

5.5 Denuncias anónimas

```
Usuario -> App: Abrir pantalla Denuncias
App: Botones “Policía Nacional” / “Guardia Civil”
App -> Navegador: abre enlace oficial (la app no guarda datos)
```

---

6) Diseño de GUI (pantallas principales)

- **Onboarding + permisos** (ubicación para mapa/pánico; notificaciones).
- **Registro/Iniciar sesión** (familia anónima / entidad verificada).
- **Home** (accesos: Mapa, Entre Familias, Botón de pánico, Comercios, Denuncias).

- **Mapa** (filtros por tipo de recurso; ficha con datos públicos de entidades).
- **Entre Familias** (feed de posts, detalle con comentarios, perfil básico y DM opcional).
- **Pánico** (botón + configuración de contactos de confianza).
- **Comercios** (crear/gestionar donaciones; ver entidades y contactar).
- **Denuncias** (enlaces oficiales).
- **Perfil** (cambiar seudónimo; activar/desactivar DM; gestionar contactos de confianza).
- **Admin web** (moderación de posts, aprobación de recursos, verificación entidades).

---

7) API / Cloud Functions (especificación breve)

- `onCreateUser(uid)`: crear doc `users` inicial, normalizar seudónimo.
- `moderateText(type, id)`: validar `posts/comments` (longitud, palabras prohibidas, links).
- `approveResource(resourceId)`: cambiar `status=pending->approved` (admin).
- `triggerPanic(alertId)`: enviar FCM a contactos; marcar `status=sent`.
- `ackPanic(alertId)`: contacto confirma recepción; `status=acked`.
- `closePanic(alertId)`: cerrar y programar borrado TTL.
- `statsPublic()`: métricas agregadas sin datos personales.

Formato genérico (HTTP Callable):

```
POST /functions/moderateText { type: "post|comment", id: "..." }
=> { ok: true }
```

---

8) Configuración y despliegue

- **Firebase**: 1 proyecto (dev/prod con colecciones separadas o prefijos).
- **Variables**: claves Maps (entorno), FCM, dominios permitidos.
- **Privacidad**: políticas RGPD/LOPDGDD; logs mínimos; retención limitada (p. ej., 30 días para `panicAlerts`).
- **Backups**: export Firestore diario (gratis local + opcional).
- **Store**: Play Store / App Store (texto de privacidad, permisos justificados).

---

9) Consideraciones legales y éticas

- Anonimato para familias por defecto; advertencias de seguridad en DM.
- Botón de pánico no sustituye al **112**.
- Denuncias: solo **enlace** a canales oficiales; no almacenamos contenido.
- Moderación previa en “Entre Familias” para proteger a la comunidad.

---

10) Roadmap (resumen de sprints)

- **Sprint 1**: Auth + perfiles + home + políticas.
- **Sprint 2**: Mapa (lectura) + validación básica de recursos.

- **Sprint 3**: Entre Familias (posts/comentarios + moderación).
- **Sprint 4**: Botón de pánico (push + flujo cierre).
- **Sprint 5**: Comercios → Entidades (contacto directo).
- **Sprint 6**: DM opcional + admin web + pulido general.

Con esto, un equipo técnico puede implementar directamente: stack, datos, reglas, flujos y pantallas quedan definidos con suficiente detalle.