

Sistema de Reportes para Conjuntos Residenciales

Documento de Análisis de Requerimientos
MVP - Producto Mínimo Viable

Versión	1.0
Fecha	04/01/2026
Ubicación	Cali, Colombia
Estado	MVP - Fase de Diseño

Tabla de Contenido

1. Introducción	3
1.1 Propósito del Documento	3
1.2 Alcance del Proyecto	3
1.3 Definiciones y Acrónimos	3
2. Descripción General	4
2.1 Perspectiva del Producto	4
2.2 Usuarios del Sistema	4
3. Requerimientos Funcionales	5
3.1 Módulo de Autenticación	5
3.2 Módulo de Reportes	6
3.3 Módulo de Administración	7
4. Requerimientos No Funcionales	8
5. Modelo de Datos	10
6. Arquitectura del Sistema	12
7. Diagramas	13
8. Stack Tecnológico	15
9. Plan de Implementación	16

1. Introducción

1.1 Propósito del Documento

Este documento presenta el análisis de requerimientos para el desarrollo de un Sistema de Reportes para Conjuntos Residenciales en Cali, Colombia. Define las funcionalidades, restricciones y características técnicas necesarias para construir un Producto Mínimo Viable (MVP) que permita validar la propuesta de valor.

1.2 Alcance del Proyecto

El sistema busca reemplazar el uso informal de grupos de WhatsApp para la gestión de reportes en conjuntos residenciales, proporcionando una plataforma estructurada para que residentes reporten incidencias y administraciones las gestionen eficientemente.

Incluido en el MVP:

- Registro y autenticación de usuarios
- Creación y seguimiento de reportes con imágenes
- Dashboard de administración
- Notificaciones por correo electrónico
- Diseño responsive (mobile-first)

Excluido del MVP:

- Aplicaciones móviles nativas
- Sistema de roles granular
- Integración con proveedores
- Analytics avanzados
- Chat en tiempo real

1.3 Definiciones y Acrónimos

Término	Definición
MVP	Minimum Viable Product - Producto Mínimo Viable
Conjunto	Conjunto residencial, unidad residencial o condominio
Residente	Persona que habita en una unidad del conjunto
Reporte	Registro de una incidencia, problema o solicitud
PWA	Progressive Web App - Aplicación web progresiva
RLS	Row Level Security - Seguridad a nivel de fila en BD

2. Descripción General

2.1 Perspectiva del Producto

El sistema se posiciona como una solución SaaS dirigida a administraciones de conjuntos residenciales en Colombia. Ofrece una alternativa estructurada a los grupos de WhatsApp, proporcionando trazabilidad, organización y métricas de gestión.

Problemas que resuelve:

- Pérdida de información en grupos de WhatsApp saturados
- Falta de trazabilidad en la resolución de problemas
- Imposibilidad de priorizar y categorizar incidencias
- Ausencia de métricas para la toma de decisiones
- Comunicación fragmentada entre residentes y administración

2.2 Usuarios del Sistema

Usuario	Descripción	Necesidades
Residente	Habitante de una unidad del conjunto residencial	Reportar problemas fácilmente, ver estado de reportes, recibir notificaciones
Administrador	Personal administrativo del conjunto	Ver todos los reportes, gestionar estados, comunicarse con residentes
Super Admin	Administrador de la plataforma	Gestionar conjuntos, soporte técnico, monitoreo general

3. Requerimientos Funcionales

3.1 Módulo de Autenticación

ID	Requerimiento	Descripción	Prior.
RF-001	Registro con código	Permitir registro de residentes mediante código único de acceso del conjunto.	Alta
RF-002	Auth Magic Link	Autenticar usuarios mediante enlaces mágicos por email, sin contraseñas.	Alta
RF-003	Validación email	Verificar que el correo electrónico sea válido antes de completar registro.	Alta
RF-004	Perfil de usuario	Permitir completar perfil con nombre y número de unidad.	Media
RF-005	Cierre de sesión	Permitir cerrar sesión de manera segura.	Alta

3.2 Módulo de Reportes

ID	Requerimiento	Descripción	Prior.
RF-006	Crear reporte	Permitir crear reportes con título, descripción, categoría y ubicación.	Alta
RF-007	Adjuntar imágenes	Permitir adjuntar hasta 3 fotos por reporte con optimización automática.	Alta
RF-008	Categorización	Ofrecer categorías: Infraestructura, Seguridad, Aseo, Convivencia, Otro.	Alta
RF-009	Ubicación	Permitir seleccionar ubicación: Parqueadero, Piscina, Lobby, Ascensor, etc.	Media
RF-010	Ver mis reportes	Mostrar lista de reportes del usuario con estado y última actualización.	Alta
RF-011	Detalle reporte	Mostrar detalle completo incluyendo historial de estados y comentarios.	Alta
RF-012	Comentar	Permitir agregar comentarios tanto por residentes como administradores.	Media

3.3 Módulo de Administración

ID	Requerimiento	Descripción	Prior.
RF-013	Dashboard	Mostrar dashboard con todos los reportes, filtros por estado, categoría y fecha.	Alta
RF-014	Cambiar estado	Permitir cambiar estado: Abierto, En Progreso, Resuelto, Cerrado.	Alta
RF-015	Nota interna	Permitir agregar comentarios internos visibles solo para administradores.	Media
RF-016	Responder	Permitir responder al residente notificándole por email.	Alta
RF-017	Código acceso	Permitir regenerar el código de acceso del conjunto.	Media
RF-018	Lista residentes	Mostrar lista de residentes registrados en el conjunto.	Baja
RF-019	Crear conjunto	Permitir crear nuevo conjunto con nombre, dirección y contacto.	Alta

4. Requerimientos No Funcionales

4.1 Rendimiento

ID	Requerimiento	Métrica Objetivo
RNF-001	Tiempo de carga inicial	< 3 segundos en conexión 4G
RNF-002	Tiempo de respuesta operaciones	< 500ms para CRUD
RNF-003	Carga de imágenes optimizadas	< 2 segundos
RNF-004	Usuarios concurrentes	Mínimo 50 por conjunto

4.2 Seguridad

ID	Requerimiento	Descripción
RNF-005	Autenticación segura	Tokens JWT con expiración de 7 días
RNF-006	Aislamiento de datos	Row Level Security (RLS) para acceso solo a datos del conjunto propio
RNF-007	Encriptación	HTTPS/TLS 1.3 en todas las comunicaciones
RNF-008	Validación entrada	Sanitización de inputs para prevenir XSS e inyección SQL
RNF-009	URLs firmadas	URLs de imágenes con firma y expiración

4.3 Usabilidad

ID	Requerimiento
RNF-010	Interfaz intuitiva, utilizable sin capacitación previa
RNF-011	Diseño responsive, funcional en dispositivos móviles
RNF-012	Mensajes de error claros que orienten hacia la solución
RNF-013	Proceso de crear reporte en menos de 60 segundos
RNF-014	Funcionar como PWA con acceso desde pantalla de inicio

4.4 Disponibilidad

ID	Requerimiento
----	---------------

RNF-015	Disponibilidad mínima del 99.5%
RNF-016	Escalabilidad a múltiples conjuntos sin cambios de arquitectura
RNF-017	Backups de base de datos diarios
RNF-018	Recuperación de fallos en menos de 4 horas

5. Modelo de Datos

El modelo de datos está diseñado para PostgreSQL (Supabase) con las entidades principales necesarias para el MVP. Se usa UUID como identificador primario.

5.1 Tabla: conjuntos

Campo	Tipo	Descripción
id	UUID (PK)	Identificador único
nombre	VARCHAR(255)	Nombre del conjunto
direccion	VARCHAR(500)	Dirección física
ciudad	VARCHAR(100)	Ciudad
codigo_acceso	VARCHAR(10)	Código para registro de residentes
created_at	TIMESTAMP	Fecha de creación

5.2 Tabla: usuarios

Campo	Tipo	Descripción
id	UUID (PK)	Identificador único
auth_id	UUID (FK)	Referencia a Supabase Auth
email	VARCHAR(255)	Correo electrónico
nombre	VARCHAR(255)	Nombre completo
conjunto_id	UUID (FK)	Referencia al conjunto
unidad	VARCHAR(50)	Número de apartamento/casa
rol	ENUM	residente admin super_admin

5.3 Tabla: reportes

Campo	Tipo	Descripción
id	UUID (PK)	Identificador único
conjunto_id	UUID (FK)	Referencia al conjunto

usuario_id	UUID (FK)	Usuario creador
titulo	VARCHAR(255)	Título del reporte
descripcion	TEXT	Descripción detallada
categoria	ENUM	infraestructura seguridad aseo convivencia otro
ubicacion	VARCHAR(100)	Ubicación dentro del conjunto
estado	ENUM	abierto en_progreso resuelto cerrado
es_anonimo	BOOLEAN	Indica si es anónimo
created_at	TIMESTAMP	Fecha de creación

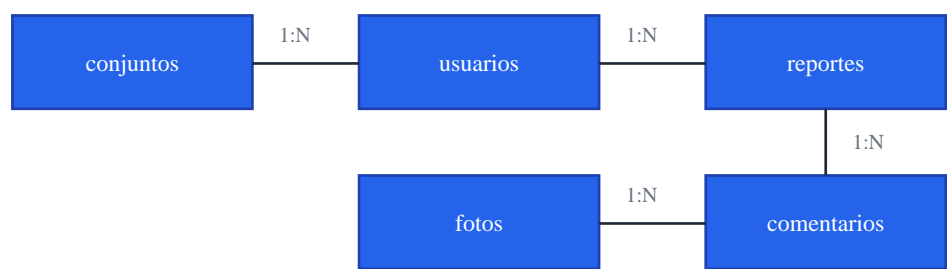
5.4 Tabla: reporte_fotos

Campo	Tipo	Descripción
id	UUID (PK)	Identificador único
reporte_id	UUID (FK)	Referencia al reporte
cloudinary_id	VARCHAR(255)	ID público en Cloudinary
url	TEXT	URL de la imagen
created_at	TIMESTAMP	Fecha de subida

5.5 Tabla: comentarios

Campo	Tipo	Descripción
id	UUID (PK)	Identificador único
reporte_id	UUID (FK)	Referencia al reporte
usuario_id	UUID (FK)	Usuario autor
contenido	TEXT	Contenido del comentario
es_interno	BOOLEAN	Solo visible para admins si es true
created_at	TIMESTAMP	Fecha de creación

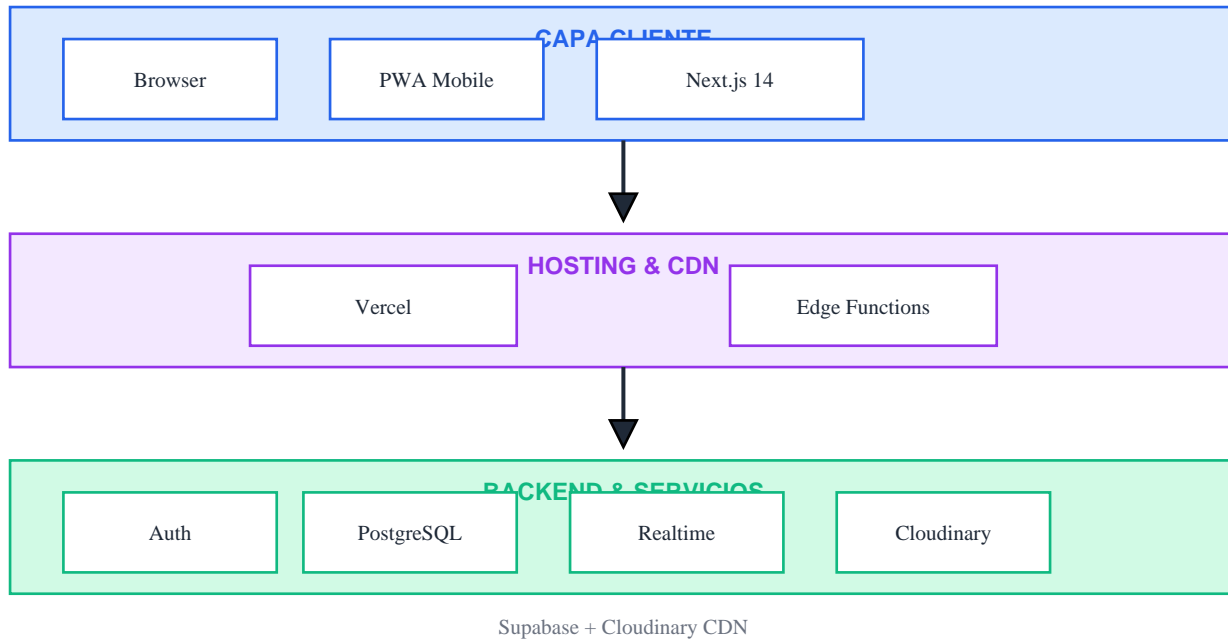
5.6 Diagrama Entidad-Relación



6. Arquitectura del Sistema

La arquitectura sigue un patrón de aplicación web moderna con separación entre frontend y backend, aprovechando servicios en la nube para reducir complejidad y acelerar el desarrollo.

6.1 Diagrama de Arquitectura



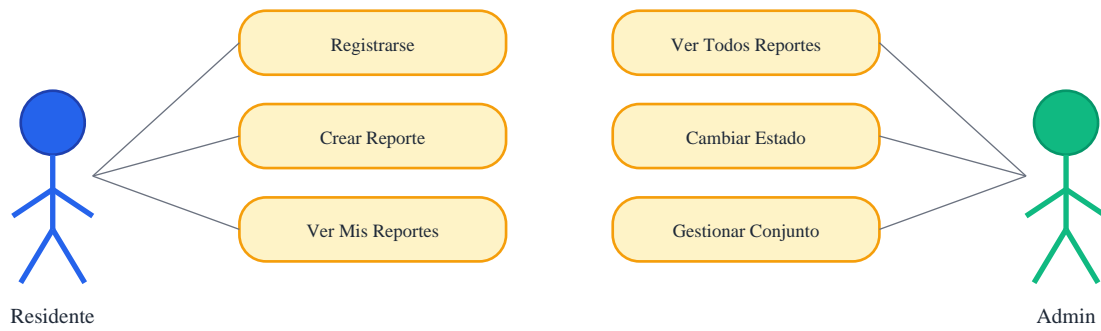
6.2 Flujo de Datos

Crear Reporte: Usuario completa formulario → Imágenes suben a Cloudinary (cliente-side) → Se obtienen URLs optimizadas → Datos van a Supabase → RLS valida pertenencia al conjunto → Se guarda y notifica vía Realtime.

Consultar Reportes: Usuario solicita reportes → Next.js consulta Supabase → RLS filtra por conjunto_id automáticamente → Se retornan solo reportes visibles → Imágenes se sirven desde CDN de Cloudinary.

7. Diagramas

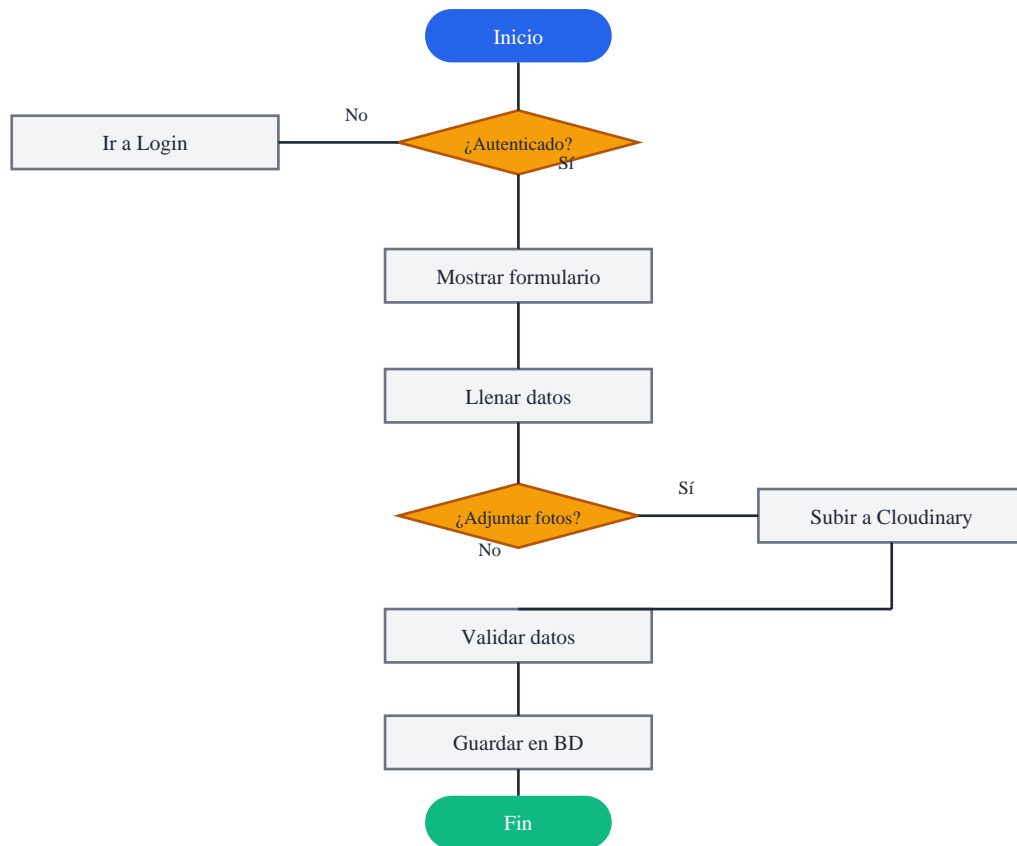
7.1 Casos de Uso



7.2 Estados del Reporte



7.3 Flujo: Crear Reporte



8. Stack Tecnológico

El stack prioriza velocidad de desarrollo, bajo costo operacional y escalabilidad. Todas las tecnologías cuentan con tiers gratuitos generosos para la fase de validación.

Capa	Tecnología	Justificación
Frontend	Next.js 14	Framework React con SSR/SSG, optimizado para SEO
UI	shadcn/ui + Tailwind	Componentes accesibles, estilos rápidos
Backend	Supabase	PostgreSQL con Auth, Storage, Realtime, API auto
Auth	Magic Link	Sin contraseña, reduce fricción de usuarios
Base Datos	PostgreSQL	Robusta, soporta RLS nativamente
Imágenes	Cloudinary	CDN global, optimización automática
Hosting	Vercel	Deploy automático, Edge Functions incluidas
Email	Resend	API moderna, 3000 emails/mes gratis

8.1 Estimación de Costos

Servicio	Tier Gratis	Si Escala	Notas
Vercel	100GB BW	\$20/mes	Plan Pro si necesario
Supabase	500MB DB	\$25/mes	Plan Pro
Cloudinary	25GB	\$0	Suficiente para MVP
Resend	3K emails	\$20/mes	Si supera 3K
Dominio .co	-	\$30/año	Registro inicial
TOTAL	\$0	~\$65/mes	Al escalar

9. Plan de Implementación

Plan diseñado para entregar un MVP funcional en 3-4 semanas, permitiendo iniciar validación con usuarios reales lo antes posible.

9.1 Cronograma

Sem.	Fase	Entregables
1	Setup	Config Next.js + Supabase, Schema BD, Auth Magic Link, Estructura base
2	Residente	Flujo registro con código, Formulario reporte, Cloudinary, Vista mis reportes
3	Admin	Dashboard reportes, Cambio estados, Comentarios, Config conjunto
4	Deploy	Notificaciones email, Responsive, Testing básico, Producción, Piloto

9.2 Criterios de Éxito del Piloto

Métrica	Objetivo	Medición
Adopción	> 30% residentes registrados	Registrados / Total unidades
Uso	> 10 reportes/conjunto/mes	Conteo reportes creados
Resolución	> 70% reportes resueltos	Resueltos / Total
Respuesta	< 48 horas primer response	Tiempo creación a comentario
NPS	> 30	Encuesta usuarios mes 1

9.3 Riesgos y Mitigaciones

Riesgo	Prob.	Imp.	Mitigación
Baja adopción inicial	Media	Alto	Trabajar con admin para promover activamente
Usuarios no técnicos	Alta	Medio	UX simple, onboarding guiado, soporte WhatsApp
Competencia WhatsApp	Alta	Alto	Enfatizar trazabilidad, historial, sin ruido
Escalabilidad técnica	Baja	Medio	Arquitectura cloud-native desde inicio

Próximos Pasos: Con este documento como base, comenzar desarrollo del MVP siguiendo el cronograma propuesto. En paralelo, trabajar en conseguir el primer conjunto piloto para validar con usuarios reales.

Documento generado para fines de planificación y desarrollo. Versión 1.0 - Enero 2025