Cuentas claras

**Período:** 22/08/2025 — 05/12/2025  
**Metodología:** Scrum — Sprints iterativos e incrementales

# **1.** Análisis del Caso

## 1.1 Contexto General

En Chile, la administración de comunidades enfrenta desafíos significativos en términos de **transparencia, trazabilidad y eficiencia**. La **Ley 21.442 de Copropiedad Inmobiliaria** exige procesos digitalizados y una gestión clara de gastos comunes, lo que ha evidenciado carencias en los sistemas actuales:

* Uso de **planillas Excel y registros manuales**.
* **Poca integración** con pasarelas de pago y sistemas externos.
* Ausencia de **prorrateo automatizado** en gastos comunes.
* **Falta de reporterías** para toma de decisiones en tiempo real.

El proyecto **Cuentas Claras 2.0** busca dar solución a estas problemáticas con una **plataforma integral**, interoperable y con alto nivel de seguridad.

## 1.2 Problemas Identificados

* **Inconsistencias en pagos**: errores frecuentes en conciliaciones manuales.
* **Baja trazabilidad**: registros dispersos dificultan auditorías.
* **Limitada transparencia**: propietarios carecen de acceso en tiempo real.
* **Emisiones lentas**: procesos mensuales poco automatizados.
* **Escasa integración** con sistemas tributarios y pasarelas de pago.

## 1.3 Objetivos del Proyecto

* Desarrollar un sistema **web multi-tenant**(multi-clientes) para administración de comunidades.
* Digitalizar procesos de **emisión, pago y reporterías** de gastos comunes.
* Integrar **pagos en línea** y conciliación automática.
* Garantizar **cumplimiento legal** y seguridad de la información.

## 1.4 Alcance del Proyecto

El proyecto considera los siguientes módulos:

1. **Autenticación y Seguridad:**
   * Login, roles, JWT, control de acceso y auditoría.
2. **Comunidades y Unidades:**
   * CRUD de comunidades, unidades, coeficiente de copropiedad, propietarios y arrendatarios.
3. **Gastos Comunes y Emisión:**
   * Registro y prorrateo de gastos (coeficiente, partes iguales, consumo, fijo).
   * Emisión mensual con estados (borrador, emitido, pagado).
4. **Pagos y Conciliación:**
   * Pagos manuales y electrónicos con Webpay/Khipu.
   * Conciliación automática vía webhooks y exportación de reportes.
5. **Integraciones y Reporterías:**
   * Exportación CSV/PDF, interoperabilidad con SII y sistemas externos.

## 2. Squad y Responsabilidades

| **Integrante** | **Rol / Especialidad** | **Responsabilidades Detalladas** | **Tecnologías Clave** |
| --- | --- | --- | --- |
| Patricio Quintanilla | Desarrollo Frontend | -Desarrollo del panel de **Cuentas claras** en React/Angular  -Diseño de interfaces modernas, responsive y accesible.  -Consumo de **APIs REST**, integración con autenticación JWT y **2FA(TOPT).**  -Gestión de estados y validación de formularios  -Pruebas de usabilidad, experiencia de usuario y mejoras de interacción. | React, Angular, typeScript, JavaScript(ES6+), Framer Motion, Lucide-react, Vite, JWT, TOPT(otpauth),Docker |
| Matías Román | QA & Coordinación | -Coordinación con stakeholders y gestión de backlog ágil.  -Diseño y ejecución de pruebas unitarias, funcionales e integración.  -Automatización de regresiones con pipelines CI/CD.  -Reportería de errores y seguimiento de métricas de calidad.  -Validación de contratos de API con Swagger y colecciones Postaman. | Postman, Swagger/OpenAPI, Github Actions, Jira/Confuence, Vitest/Jest, Cypress/Playwright, Eslint, Prettier, SonarQube. |
| Frank Vogt | Desarrollo Backend | -Diseño e implementación de **API REST** para módulos de cuentas claras.  -Modelado de datos en MySQL (3FN) y uso Redis para caché y colas.  -Autenticación JWT, 2FA (TOTP), roles y permisos por comunidad.  -Integración con pasarelas de pago y conciliaciones bancarias.  -Seguridad: CORS, rate limiting, auditoria e idempotencia.  -Implementación de CI/CD y despliegues dockerizados | Node.js (Express/NestJS), MySQL, Sequelize/TypeORM, Redis, JWT, Bcrypt/Argon2, TOTP(otplib/speajeasy), Swagger/ OpenAPI, Docker/composes, Nginx/PM2 |

**3. Mapa Mental (Descripción Conceptual)**

El mapa mental de **Cuentas Claras 2.0** se organiza en **cinco ramas principales**:

* **Seguridad y Acceso:** Login, roles, JWT, auditorías.
* **Gestión de Comunidades:** CRUD, unidades, coeficiente, propietarios.
* **Gastos y Emisiones:** Registro de gastos, prorrateo, emisión mensual.
* **Pagos y Conciliación:** Pagos manuales, Webpay/Khipu, conciliación automática.
* **Reporterías e Integración:** Exportaciones, SII, webhooks, dashboards.

**4. Mapa de Actores**

| **Actor** | **Rol en el Sistema** | **Interacciones Clave** |
| --- | --- | --- |
| Administrador | Gestiona comunidad y gastos | CRUD comunidades, emisión de gastos, reporterías, conciliación. |
| Propietario/Arrendatario | Consulta y paga gastos comunes | Portal de pagos, visualización de cargos y estados. |
| Backend API | Lógica de negocio y datos | Endpoints REST, autenticación, auditoría. |
| Pasarelas de Pago | Webpay/Khipu | Procesamiento y confirmación de pagos vía webhooks. |

## 5. Visión del Proyecto + 4 Pilares

**Visión del Proyecto**

"Cuentas Claras 2.0 será la plataforma más confiable y moderna para la **gestión de comunidades y gastos comunes en Chile**, integrando tecnología, seguridad y facilidad de uso para todos los actores."

**4 pilares Estratégicos**

1. **Digitalización Total:** Migración completa de procesos manuales a una plataforma web.
2. **Seguridad y Cumplimiento Legal:** Autenticación robusta, roles, auditorías y normativa Ley 21.442.
3. **Automatización e Integración:** Procesos automáticos de emisión, prorrateo, pagos y reporterías.
4. **Experiencia de Usuario:** Interfaz intuitiva y accesible, con datos en tiempo real.

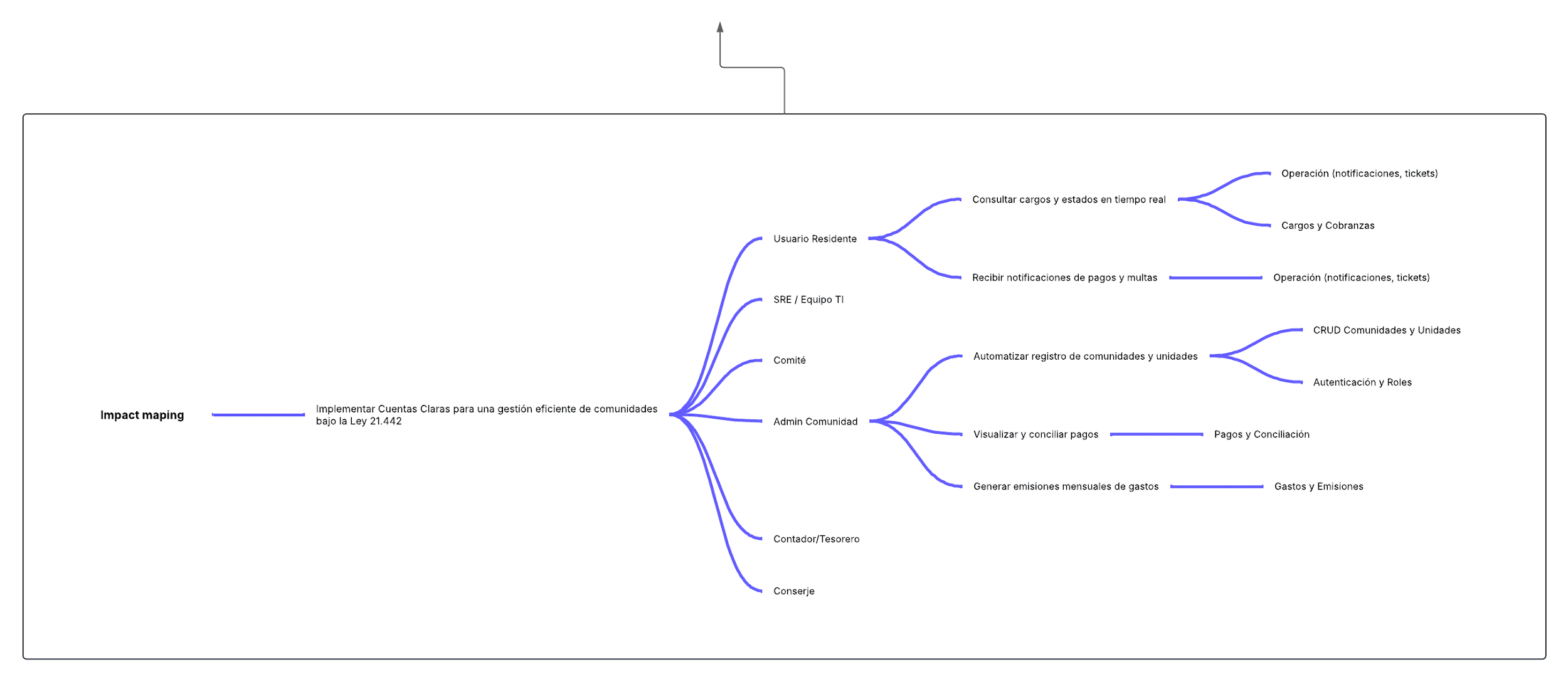
## Épicas

| **Epic ID** | **Épica** | **Objetivo** |
| --- | --- | --- |
| E1 | Autenticación y Cuentas | JWT, refresh, usuarios, perfiles vinculados a persona. |
| E2 | Comunidades y Unidades | CRUD comunidad, edificio, torre, unidad, tenencias, membresías. |
| E3 | Gastos y Emisiones | Registro de gastos, emisiones mensuales, prorrateo por regla. |
| E4 | Cargos y Cobranzas | Cargos por unidad, recálculo de interés, notificaciones. |
| E5 | Pagos y Conciliación | Pagos, aplicaciones, webhooks (Webpay/Khipu), conciliación bancaria. |
| E6 | Medidores y Consumos | Medidores, lecturas, tarifas, consumos compartidos. |
| E7 | Multas e Intereses | Multas, configuración de interés simple/compuesto con tope. |
| E8 | Operación | Amenidades, reservas, tickets, notificaciones, documentos, bitácora. |
| E9 | Reporterías y BI | Export /reportes, hechos/dimensiones básicas, métricas clave. |
| E10 | Fondos y Contabilidad | Fondos, asientos contables, plan de cuentas por comunidad. |
| E11 | Presupuesto | Presupuesto mensual por categoría y seguimiento. |
| E12 | Seguridad y Observabilidad | Roles, auditoría, rate limiting, logs/metrics/tracing, health checks. |

## Historias de Usuario (con criterios de aceptación y estimación)

| **ID** | **Épica** | **Historia** | **Criterios de Aceptación** | **Prioridad** | **SP** |
| --- | --- | --- | --- | --- | --- |
| HU1 | E1 | Como usuario quiero registrarme e iniciar sesión para acceder al sistema. | POST /auth/register, /auth/login; valida email único, password hash (bcrypt). | Alta | **5** |
| HU2 | E2 | Como admin quiero crear comunidades con RUT y razón social para iniciar la operación. | POST /comunidades; valida RUT/DV; rut único. | Alta | **8** |
| HU3 | E2 | Como admin quiero registrar unidades y propietarios/arrendatarios con vigencias. | POST /unidades, /unidades/{id}/tenencias; regla: ≥1 propietario activo. | Alta | **8** |
| HU4 | E3 | Como contador quiero cargar gastos y emitir el gasto común del periodo. | POST /gastos; POST /emisiones; reglas de prorrateo: coeficiente, partes iguales, consumo, fijo. | Alta | **13** |
| HU5 | E4 | Como residente quiero ver mis cargos y su estado para pagar a tiempo. | GET /unidades/{id}/cargos; estados: pendiente/pagado/vencido/parcial | Alta | **5** |
| HU6 | E5 | Como tesorero quiero registrar pagos y aplicarlos a cargos según prioridad. | POST /pagos; POST /pagos/{id}/aplicar; soporte webhooks. | Alta | **8** |
| HU7 | E6 | Como admin quiero registrar lecturas de medidores y calcular consumo mensual. | POST /medidores; POST /medidores/{id}/lecturas; tarifa vigente por periodo. | Media | **8** |
| HU8 | E7 | Como comité quiero emitir multas y configurar intereses por mora. | POST /unidades/{id}/multas; POST /comunidades/{id}/configuracion-interes. | Media | 5 |
| HU9 | E8 | Como conserje quiero registrar eventos en bitácora y gestionar tickets. | POST /comunidades/{id}/bitacora; POST /comunidades/{id}/tickets. | Media | 5 |
| HU10 | E9 | Como admin quiero exportar reportes de cargos y pagos. | GET /reportes/export (CSV) con filtros por periodo/comunidad. | Media | 5 |
| HU11 | E10 | Como contador quiero registrar movimientos de fondos y asientos contables. | POST /fondos; POST /fondos/{id}/movimientos; POST /asientos. | Media | 8 |
| HU12 | E11 | Como admin quiero definir presupuesto mensual por categoría y compararlo con real. | POST /presupuestos; /presupuestos/{id}/detalles; vistas de desvíos. | Baja | 8 |
| HU13 | E12 | Como SRE quiero monitorear métricas, logs y health checks. | GET /healthz, /readyz, /metrics; logs JSON y trazas. | Baja | 5 |

## Impact mapping



| **Actor** | **Objetivo (Por qué)** | **Impactos / Cambios Esperados (Cómo)** | **Entregables / Funcionalidades (Qué)** |
| --- | --- | --- | --- |
| Usuario Residente | Acceder y pagar sus cargos de manera transparente | - Consultar cargos y estados en tiempo real- Recibir notificaciones de pagos y multas | - Portal de pagos- Notificaciones automáticas |
| Admin Comunidad | Gestionar comunidades y asegurar la correcta emisión de gastos | - Automatizar registro de comunidades y unidades- Generar emisiones mensuales- Visualizar y conciliar pagos | - CRUD Comunidades y Unidades- Gastos y Emisiones- Cargos y Cobranzas- Pagos y Conciliación |
| Contador / Tesorero | Llevar registro contable y reporte financiero | - Gestionar pagos y conciliaciones- Exportar reportes confiables | - Export / Reporterías- Fondos y Contabilidad |
| Comité | Supervisar operaciones y cumplimiento | - Emitir multas- Configurar intereses de mora | - Multas e Intereses |
| Conserje | Registrar eventos y gestionar solicitudes | - Registrar eventos de bitácora- Gestionar tickets con prioridad | - Bitácora y Tickets |
| Equipo TI / SRE | Mantener la plataforma estable y segura | - Monitorear métricas y logs- Controlar accesos y roles | - Seguridad y Observabilidad (roles, auditoría, health checks, logging) |
| Pasarelas de Pago (Webpay / Khipu) | Procesar pagos en línea de manera confiable | - Confirmar pagos automáticamente- Integrar conciliación bancaria | - Pagos y Conciliación / Webhooks |
| SII | Cumplimiento tributario | - Interoperar con sistemas tributarios | - Reporterías / Export CSV/PDF |

(Por si no se entiende bien la imagen)

[Impacct mapping](https://lucid.app/lucidchart/587c3af1-cc7f-4a4f-8cea-9b9e719901a4/edit?invitationId=inv_a3a7a2f7-21dc-4f54-879a-6bb01de81119&page=0_0)

## Product Backlog Priorizado

Esta dentro de un Excel mejor explicado

## User Story Mapping

| Epica/Historia de usuario | | E1: Seguridad | E2: gestión de comunidades | | | E3: gasto y emisiones | | E4: Consulta de cargos | | E5: Pagos y conciliaciones | | E6: medidores y consumo |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| HU1 | | Registrarme e iniciar sesión |  | | |  | |  | |  | |  |
| HU2 | |  | Crear comunidades con RUT y razón social | | |  | |  | |  | |  |
| HU3 | |  | Registrar unidades y propietarios/arrendatarios con vigencias | | |  | |  | |  | |  |
| HU4 | |  |  | | | Cargar gastos y emitir gasto común del periodo | |  | |  | |  |
| HU5 | |  |  | | |  | | Ver mis cargos y su estado para pagar a tiempo | |  | |  |
| HU6 | |  |  | | |  | |  | | Registrar pagos y aplicarlos a cargos según prioridad | |  |
| HU7 | |  |  | | |  | |  | |  | | lecturas de medidores y calcular consumo mensual |
| ----------- | **E7: Multas e Intereses** | | | **E8: Bitácora y Tickets** | **E9: Reporterías** | | **E10: Fondos y Contabilidad** | | **E11: Presupuestos** | | **E12: Monitoreo y Métricas** | |
| HU8 | Emitir multas y configurar intereses por mora | | |  |  | |  | |  | |  | |
| HU9 |  | | | Registrar eventos en bitácora y gestionar tickets |  | |  | |  | |  | |
| HU10 |  | | |  | Exportar reportes de cargos y pagos | |  | |  | |  | |
| HU11 |  | | |  |  | | Registrar movimientos de fondos y asientos contables | |  | |  | |
| HU12 |  | | |  |  | |  | | Definir presupuesto mensual por categoría y compararlo con real | |  | |
| HU13 |  | | |  |  | |  | |  | | Monitorear métricas, logs y health checks | |

## Retrospectiva del Proyecto

**Resumen del Proyecto**

El proyecto **Cuentas Claras** tuvo como objetivo principal desarrollar un sistema integral para la gestión de comunidades, incluyendo autenticación segura, manejo de gastos comunes, pagos en línea, reporterías y control de accesos. A lo largo del ciclo de desarrollo, se trabajó con metodologías ágiles, priorizando funcionalidades críticas y buscando siempre la satisfacción del usuario final.

**Lo que salió bien**

* **Definición clara de requisitos:** La recopilación inicial de historias de usuario, épicas y criterios de aceptación fue exhaustiva y permitió orientar correctamente el desarrollo.
* **Uso de metodologías ágiles:** La planificación mediante backlog priorizado e iteraciones cortas mejoró la visibilidad y el control del avance.
* **Colaboración del equipo:** El trabajo coordinado entre desarrolladores, área de negocio y responsables de QA facilitó la resolución de dudas y la toma de decisiones rápidas.

**Lo que se podría mejorar**

* **Pruebas más amplias:** Aunque se realizaron validaciones funcionales, incorporar pruebas automatizadas y de integración continua habría permitido detectar errores antes.
* **Gestión del tiempo:** Ajustar mejor la carga de trabajo por sprint para evitar tareas pendientes al cierre de cada iteración.
* **Comunicación con interesados externos:** Se identificaron retrasos por falta de información de terceros, lo que podría mejorarse con reuniones de alineación más frecuentes.

**Conclusiones y acciones para tomar**

* Establecer **un plan de pruebas automatizadas** para garantizar la calidad del software.
* Mejorar la **planificación de sprints** con estimaciones más precisas y gestión proactiva de bloqueos.
* Mantener **canales de comunicación abiertos** con todas las áreas, incluyendo stakeholders externos.
* Documentar de manera más detallada las **lecciones aprendidas** para futuros proyectos similares.

Con estos ajustes, futuros desarrollos podrán ejecutarse de manera más eficiente, con mayor calidad y alineación entre todos los involucrados.