cZr Catering - Documentación Técnica y Funcional

Versión: 1.2.0 (Completa - Rebrandeada)
Autor: Leo Zelvys (Chef & IT Manager)
Fecha: 14 de Diciembre de 2025

1. Visión y Alcance

1.1. Resumen Ejecutivo

cZr Catering es una plataforma de gestión gastronómica integral que articula la **verdad operativa** (costos, mermas, logística interna) con la **propuesta comercial** (experiencia del cliente, precio final). El sistema se centra en la eficiencia de carga ("Keyboard-First"), la inteligencia de datos (Scraping/IA) y la separación estricta de responsabilidades entre el "Back of House" (Cocina) y el "Front of House" (Cliente).

1.2. Problema a Resolver

Los sistemas actuales carecen de una integración fluida entre la ingeniería de menú y la venta comercial. O son ERPs rígidos o son menús digitales "tontos". cZr Catering resuelve la fricción de mantener precios actualizados en una economía volátil sin romper los compromisos comerciales ya asumidos (snapshots).

1.3. Mapa de Módulos del Sistema

El alcance abarca 6 módulos funcionales interconectados:

1. **Seguridad y Usuarios (Auth):** Control de acceso basado en roles (RBAC).
2. **Materia Prima (Almacén Inteligente):** Gestión de insumos con enriquecimiento de datos.
3. **Proveedores y Costos:** Gestión de listas de precios y actualizaciones masivas.
4. **Ingeniería de Recetas (Diseño):** Núcleo de cálculo de costos y armado de platos.
5. **Generador de Propuestas (Comercial):** Salida documental para el cliente (Sanitizada).
6. **Configuración y Reglas de Negocio:** Parámetros globales (márgenes, fuentes de datos).

2. Requerimientos Funcionales (Por Módulo)

Módulo Transversal: Seguridad y Usuarios (AUTH)

- **RF-AUTH-01 - Roles:** El sistema debe manejar al menos 3 roles: Administrador/Chef (Acceso total), Cocina (Solo ver recetas y stock, sin precios), Cliente/Invitado (Solo ver propuestas generadas).
- **RF-AUTH-02 - Acceso Público Seguro:** Las propuestas compartidas deben ser

accesibles mediante tokens únicos (UUID) sin necesidad de login para el cliente final, con expiración configurable.

Módulo 1: Materia Prima (INGREDIENTS)

- **RF-ING-01 - Carga Inteligente:** Autocompletado de datos normativos y de mercado mediante IA/Scraping.
- **RF-ING-02 - Alta Rápida (Quick Add):** Capacidad de crear ingredientes "on-the-fly" desde la receta sin salir del flujo de trabajo.
- **RF-ING-03 - Versionado:** Historial de cambios de precio para análisis de evolución de costos.
- **RF-ING-04 - Atributos Extendidos:** Gestión de sustitutos, mermas por limpieza y cocción, y datos nutricionales.

Módulo 2: Proveedores y Costos (SUPPLIERS)

- **RF-SUP-01 - Gestión de Proveedores:** ABM de proveedores con datos de contacto y condiciones comerciales.
- **RF-SUP-02 - Actualización Masiva:** Posibilidad de importar listas de precios (CSV/Excel) y mapearlas a los ingredientes existentes para actualizar costos en lote.
- **RF-SUP-03 - Alertas de Variación:** Notificar si un insumo clave aumentó más de un X% respecto a la última compra.

Módulo 3: Ingeniería de Recetas (RECIPES)

- **RF-REC-01 - UX Híbrida:** Soporte simultáneo para Drag & Drop y Carga por Grilla (Teclado).
- **RF-REC-02 - Sugerencia Semántica:** El sistema debe proponer ingredientes base al escribir el nombre del plato.
- **RF-REC-03 - Costeo Dinámico:** Cálculo en tiempo real de Costo Materia Prima (CMV) + Margen Bruto = Precio Sugerido.
- **RF-REC-04 - Anidamiento:** Capacidad de usar "Sub-Recetas" (ej: Salsa Pomodoro) como ingrediente de otra receta.

Módulo 4: Generador de Propuestas (PROPOSALS)

- **RF-PROP-01 - Snapshot (Congelamiento):** Al emitir una propuesta, se genera una copia estática de los datos. Los aumentos posteriores de costos NO deben afectar propuestas enviadas.
- **RF-PROP-02 - Sanitización:** Eliminación total de datos de costos, proveedores y notas internas en la vista del cliente.
- **RF-PROP-03 - Personalización:** Selección de plantilla (Catering, Menú Carta, Evento Corporativo) y visualización de semáforo nutricional.

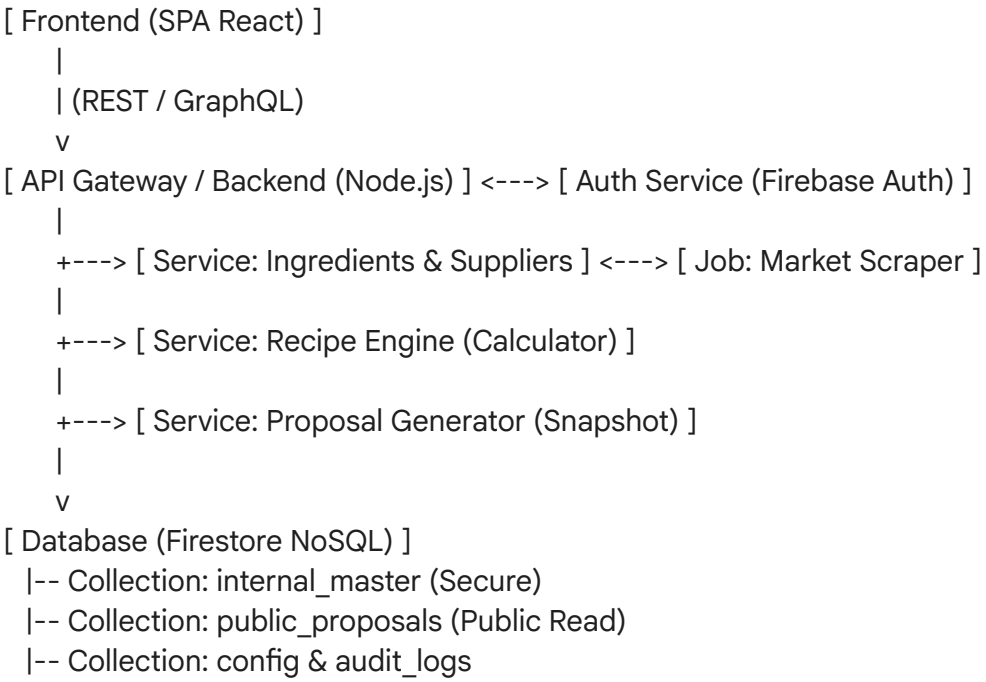
Módulo 5: Configuración (SETTINGS)

- **RF-SET-01 - Variables Globales:** Definición de márgenes por defecto, moneda base y

reglas de redondeo.
- **RF-SET-02 - Fuentes de Datos:** Configuración de APIs de scraping o endpoints de proveedores conectados.

3. Arquitectura del Sistema

3.1. Diagrama de Componentes



3.2. Estrategia de Persistencia

- **Master Data:** Los ingredientes y recetas vivas residen en colecciones privadas.
- **Transactional Data:** Las propuestas son documentos inmutables una vez enviadas.
- **Caching:** Uso de Redis o caché en memoria para las búsquedas de ingredientes frecuentes y resultados de scraping.

4. Modelo de Dominio (Entidades Principales)

Entidad: Proveedor

```
{
  "id": "uuid",
  "razon_social": "string",
  "rubro": "Carniceria | Verduleria",
  "lista_precios_url": "string (opcional)",
  "condicion_pago": "30 dias"
}
```

Entidad: Ingrediente

```
{
  "id": "uuid",
  "nombre": "string",
  "unidad_medida": "kg | lt | uni",
  "costo_actual": 1500.00,
  "ultima_actualizacion": "ISO8601",
  "proveedor_preferido_id": "uuid_ref",
  "historial_precios": [{ "fecha": "...", "precio": 1400 }],
  "mermas": { "limpieza": 0.10, "coccion": 0.20 },
  "datos_nutricionales": { ... }
}
```

Entidad: Receta

```
{
  "id": "uuid",
  "nombre": "string",
  "tipo": "Plato | Sub-receta",
  "componentes": [
    { "tipo": "ingrediente", "id": "uuid", "cantidad": 0.5 },
    { "tipo": "sub-receta", "id": "uuid", "cantidad": 1 }
  ],
  "costo_teorico": 5000,
  "margen_aplicado": 300,
  "precio_carta": 20000,
  "tags": ["Sin Tacc", "Verano"]
}
```

Entidad: Usuario

```
{
  "uid": "firebase_uid",
  "email": "leo@leocozina.com",
  "rol": "admin",
  "preferencias": { "teclado_virtual": false, "tema": "dark" }
}
```

5. Análisis Técnico

5.1. Stack Tecnológico

- **Frontend:** React 18 + Vite. Framework CSS: Tailwind.
- **Backend:** Node.js (Express o NestJS) o Serverless Functions (Firebase Functions).
- **BD:** Firestore (ideal para estructuras anidadas de recetas y snapshots).
- **Auth:** Firebase Authentication.
- **Infra:** Vercel o Netlify para Frontend, Google Cloud para Backend/BD.

5.2. Desafíos Técnicos

1. **Consistencia de Costos:** Cuando se actualiza un proveedor, ¿se recalcula todo el menú en tiempo real o batch nocturno? *Decisión: Recálculo asíncrono (Job)*.
2. **Performance del Buscador:** El "Quick Add" y la búsqueda deben ser instantáneos (<100ms). Requiere indexación local o Algolia/Meilisearch.

6. Contrato de APIs (Endpoints Clave)

Auth & Usuarios

- POST /auth/login
- GET /users/me

Proveedores

- GET /suppliers
- POST /suppliers/{id}/import-prices (Upload CSV)

Ingredientes

- GET /ingredients?q={term}
- POST /ingredients (Alta simple y Quick Add)
- GET /ingredients/{id}/history

Recetas

- POST /recipes (Crea receta y calcula costos)
- PUT /recipes/{id}/recalc (Fuerza recálculo de costos con precios actuales)

Propuestas

- POST /proposals (Genera Snapshot)
- GET /public/proposals/{token} (Acceso cliente)

7. Decisiones Clave de Arquitectura (ADRs)

ADR-001: NoSQL para Recetas

- **Decisión:** Usar Firestore.
- **Por qué:** Las recetas son documentos jerárquicos que varían mucho en estructura. NoSQL permite flexibilidad.

ADR-002: Snapshotting para Propuestas

- **Decisión:** Duplicar datos al crear una propuesta.
- **Por qué:** Es un requisito legal y comercial que el presupuesto enviado no cambie aunque suba la carne al día siguiente.

ADR-003: Auth Híbrida

- **Decisión:** Usuarios internos con login fuerte, Clientes con acceso por Token (Link mágico).
- **Por qué:** Reduce fricción para el cliente final (no quiere crearse una cuenta para ver un menú).

ADR-004: Cálculo de Costos Asíncrono

- **Decisión:** Al subir una lista de precios, el impacto en las 500 recetas que usan esos insumos se procesa en segundo plano (Cloud Task).
- **Por qué:** Evita timeout en el request de actualización de precios y mantiene la UI fluida.