PRD - Módulo Licitaciones

0. Control del documento

- **Producto:** Radar y Gestión de Oportunidades Públicas (Licitaciones + Compras Ágiles)
- **Alcance:** Producto funcional de punta a punta (ingesta → detección → gestión → reportes/alertas → auditoría)
- **Fuente de datos:** API oficial de Mercado Público (ChileCompra/DCCP) vigente a diciembre 2025
- **Propósito del PRD:** Definir requisitos funcionales y no funcionales, modelos de datos, flujos, reglas de negocio, integraciones, consideraciones operativas y criterios de aceptación para construir el producto.

1. Resumen ejecutivo

El producto permite:

1. **Adquirir información** desde Mercado Público mediante su API oficial, cubriendo licitaciones y órdenes de compra, incluyendo compras ágiles cuando estén representadas en los datos.
2. **Detectar oportunidades** relevantes usando reglas configurables (palabras clave, categorías, comprador, región, montos, fechas, tipo/estado) y priorizarlas por ventanas de cierre.
3. **Gestionar oportunidades** con pipeline, asignación, notas, checklist, tareas y trazabilidad.
4. **Entregar información** por bandejas, reportes programados y alertas con control de cuota/uso de API.

Definición de éxito:

- Cobertura y actualización confiables dentro de la cuota diaria.
- Alertas y bandejas que reduzcan el tiempo de identificación/priorización.
- Trazabilidad del trabajo del equipo comercial/técnico.

2. Problema, objetivos, y principios

2.1 Problema

- La información de oportunidades públicas existe, pero no se presenta con un flujo eficiente de detección, priorización y ejecución.
- La operación requiere control estricto del consumo de API (cuota y monitoreo).
- Se necesita trazabilidad para equipos (asignación, estados, evidencia, tiempos).

2.2 Objetivos del producto (completo)

- **O1 — Cobertura de datos:** Ingestar todas las entidades y campos disponibles por la API oficial para licitaciones y órdenes de compra.
- **O2 — Detección accionable:** Configurar reglas y generar señales (alertas) con razones interpretables.
- **O3 — Gestión operable:** Pipeline y colaboración con tareas, comentarios y auditoría.
- **O4 — Rendimiento y resiliencia:** Operar confiablemente dentro de cuota, con reintentos y tolerancia a fallas.
- **O5 — Gobernanza:** Seguridad del ticket, privacidad y control de acceso.

2.3 Principios de diseño

- **Control de cuota primero:** toda funcionalidad considera costo en hits (presupuesto diario).
- **Idempotencia:** cualquier job puede re-ejecutarse sin duplicar ni corromper datos.
- **Explicabilidad:** cada match/score debe mostrar razones.
- **Trazabilidad:** acciones del equipo registradas (quién, cuándo, qué cambió).
- **Separación de responsabilidades:** conector API, normalizador, motor de reglas, gestión, notificaciones.

3. Usuarios y permisos

3.1 Roles

- **Admin:** configura integraciones (ticket), usuarios, plantillas de pipeline, políticas de cuota, reglas globales.
- **Manager:** gestiona reglas, asignaciones, reportes y métricas; no ve/edita secretos.
- **Analista/Editor:** opera bandeja, crea oportunidades, cambia estados, comenta, genera reportes.
- **Lector:** consulta bandejas, fichas y reportes, sin modificar.

3.2 Unidades organizacionales

- **Workspace (empresa/equipo):** contiene usuarios, reglas, pipelines y configuraciones.
- **Grupos/Equipos (opcional):** reglas compartidas por línea de negocio.

3.3 Matriz de permisos (resumen)

- Configurar ticket: Admin
- Crear/editar reglas: Admin/Manager
- Ver datos: todos (según restricciones internas)
- Cambiar pipeline/asignar: Admin/Manager/Editor
- Exportar reportes: Admin/Manager/Editor (Lector según policy)
- Ver auditoría: Admin/Manager

4. Alcance funcional detallado

4.1 Integración con API oficial (Conector)

4.1.1 Responsabilidades del conector

- Autenticación por **ticket**.
- Construcción de requests para:
- **Licitaciones:** consultas por código, por fecha, por estado, activas, por organismo, por proveedor.
- **Órdenes de compra:** consultas por código, por fecha, por estado, por organismo, por proveedor.
- **Lookups:** búsqueda de comprador y proveedor (resolución de códigos internos) según endpoints disponibles.
- Soporte a formato **JSON** (mínimo). Soportar XML/JSONP como opcional.
- Gestión de errores HTTP, timeouts y respuestas parciales.

4.1.2 Limitaciones operacionales (hard constraints)

- **Cuota diaria de hits por ticket**.

• **Monitoreo por IP** y mitigación de ráfagas.
• Posibles interrupciones de servicio.

4.1.3 Reglas de consumo y presupuesto de hits

• Definir un **presupuesto diario** por workspace (por defecto 80% jobs, 20% interacción).
• **Rate limiting** (p. ej., X req/min configurable) y **backoff exponencial** ante errores.
• **Circuit breaker**: si se detecta degradación (errores repetidos), pausar y reintentar más tarde.
• **Caché de detalle**: el detalle por código solo se pide si:
• se abrió la ficha,
• se marcó como "en revisión",
• se incluyó en reporte,
• o el registro está stale (TTL configurable).

4.1.4 Observabilidad del conector

• Métricas: hits consumidos, error rate, latencias, jobs exitosos/fallidos.
• Logs estructurados sin secretos.

4.2 Ingesta y sincronización (Scheduler/Jobs)

4.2.1 Objetivo

Mantener una base actualizada de licitaciones/órdenes de compra con historial de cambios y campos normalizados.

4.2.2 Estrategia de sincronización (por defecto)

1. **Carga diaria (D+0):**
2. Licitaciones del día (todos los estados del día actual o por fecha).
3. Órdenes de compra del día (por fecha y estado "todos" si aplica).
4. **Refresco de activas:**
5. Consultar licitaciones activas varias veces al día (configurable).
6. **Detalle bajo demanda:**
7. Traer detalle por código cuando el usuario lo requiera o cuando reglas lo definan.
8. **Backfill controlado (opcional):**
9. Sincronización de N días hacia atrás (configurable) con límites para no exceder cuota.

4.2.3 Horarios recomendados

• Permitir configurar ventanas de ejecución (p. ej., nocturnas para cargas pesadas).

4.2.4 Idempotencia y deduplicación

• Claves naturales:
• Licitación: `codigo_externo` (o equivalente).
• Orden de compra: `codigo`.
• Cada ejecución:
• upsert en entidades normalizadas,
• guardar raw payload,
• registrar un "snapshot" de campos relevantes para auditoría.

4.2.5 Detección de cambios

- Comparar hashes de campos críticos (estado, cierre, items, montos) y crear eventos de cambio.
- Eventos alimentan alertas y timeline de oportunidad.

4.3 Normalización, enriquecimiento y calidad de datos

4.3.1 Normalización

- Fechas: almacenar en UTC y mostrar en zona local.
- Estados: mapear códigos a enumeraciones internas.
- Tipos: mapear tipos de licitación y tipos de OC (incluyendo compra ágil cuando aplique).
- Montos: normalizar moneda y totales.

4.3.2 Enriquecimiento

- Resolver códigos internos de comprador/proveedor mediante lookups.
- Indexar categorías (por ejemplo UNSPSC) a partir de ítems cuando esté presente.
- Campos derivados:
- `days_to_close`, `close_bucket` (0–10, 11–30, 31+)
- `is_urgent` (0–10)
- `recency` (edad desde publicación)

4.3.3 Validación y QA de datos

- Reglas de integridad:
- fechas válidas,
- montos no negativos,
- estados reconocidos,
- ítems con estructura consistente.
- Registros inválidos se marcan con `data_quality_flag` y se almacenan para revisión.

4.4 Motor de reglas (Detección/Matching)

4.4.1 Objetivo

Convertir el universo de registros en una lista priorizada de oportunidades relevantes para el equipo.

4.4.2 Componentes

- **Rule Engine:** evalúa reglas contra licitaciones/OC.
- **Scoring:** asigna puntaje basado en coincidencias.
- **Explainability:** genera “razones” por coincidencia.

4.4.3 Tipos de reglas (mínimo)

- Palabras clave:
- contiene, contiene-todas, exacta, frase.
- operadores AND/OR, grupos.

• lista de exclusión (negative keywords).
• Comprador:
• por código de organismo, nombre, región, unidad.
• Proveedor (para análisis de OCs/histórico):
• por código interno o RUT.
• Categorías:
• por código(s) de categoría de ítem.
• Fechas:
• ventana de cierre (0–30 por defecto), umbrales (0–10 destacado).
• Montos:
• rango mínimo/máximo (si el campo está disponible).
• Tipo/estado:
• licitación activa/publicada, etc.
• tipo de OC (incl. compra ágil si aplica).

4.4.4 Score (MVP y escalable)

• Puntos por dimensión:
• keyword +X
• categoría +Y
• comprador +Z
• región +W
• urgencia (0–10) multiplicador
• Score final = suma ponderada × factor de urgencia.
• Debe ser configurable por workspace.

4.4.5 Salidas del motor

• `MatchResult`:
• entidad referenciada (licitación/OC)
• score
• razones (lista)
• regla(s) que dispararon
• timestamp
• Persistencia de resultados para auditoría y reportes.

4.5 Bandejas y búsqueda (UX principal)

4.5.1 Bandeja de oportunidades (Inbox)

• Vistas predeterminadas:
• **Urgentes (0–10 días)**
• **Próximas (11–30 días)**
• **Más adelante (31+)**
• **Sin match** (opcional, para exploración)
• Orden:
• por `days_to_close` ascendente, luego score desc.
• Tarjeta de registro:
• código, título, comprador, región, fecha cierre, días a cierre, score, badges (urgente/nuevo), top 2 razones.

- Acciones rápidas:
- Guardar como oportunidad
- Descartar
- Asignar
- Añadir etiqueta

4.5.2 Búsqueda y filtros

- Filtros:
- estado, tipo, región, comprador, rango de fechas, rango de montos, categorías, palabras clave.
- Búsqueda full-text sobre título/descripcion/items.
- Guardar búsquedas como “vistas” (producto completo).

4.6 Fichas detalladas (Licitación y Orden de Compra)

4.6.1 Ficha de licitación

Secciones:

- Resumen: código, estado, tipo, fechas, cierre, días a cierre.
- Comprador: organismo/unidad, región, datos de contacto si disponibles.
- Ítems: listado con categoría/código, descripción, cantidad/unidad, montos si disponibles.
- Cronograma/hitos (si disponible).
- Historial de cambios (eventos detectados por sincronización).
- Acciones: crear/abrir oportunidad, actualizar detalle (controlado), exportar.

4.6.2 Ficha de orden de compra

Secciones:

- Resumen: código, estado, tipo (incl. compra ágil si aplica), fechas relevantes.
- Comprador y proveedor: códigos internos, RUT/razón social si existe.
- Totales: neto, impuestos, descuentos/cargos, moneda.
- Ítems: detalle de líneas.
- Historial de cambios.
- Acciones: vincular a oportunidad, etiquetar, exportar.

4.7 Gestión (Pipeline y colaboración)

4.7.1 Entidad “Oportunidad”

- Es la unidad de trabajo.
- Puede originarse desde una licitación o desde una OC (según estrategia comercial).
- Puede vincular múltiples entidades (por ejemplo, licitación principal + aclaraciones + OCs relacionadas), si el dato existe.

4.7.2 Pipeline

- Estados por defecto (editable por workspace):

• Nuevo → En revisión → Califica → Preparando → Enviado → Ganada → Perdida → Archivada
• Reglas:
• cada cambio de estado registra evento.
• estados finales bloquean ediciones de campos clave salvo permisos.

4.7.3 Asignación

• Asignación a un usuario responsable.
• Bandeja “Mis oportunidades”.
• Reasignación con comentario obligatorio (configurable).

4.7.4 Notas, comentarios y evidencias

• Notas libres.
• Comentarios tipo hilo.
• Adjuntos (producto completo): enlaces/documentos de soporte.
• Checklist por etapa (plantillas configurables):
• revisar bases
• validar requerimientos
• planificar visitas/consultas
• preparar propuesta
• enviar

4.7.5 Tareas y recordatorios

• Tareas vinculadas a oportunidad con fecha/hora.
• Recordatorios automáticos basados en cierre (p. ej., T-7, T-3, T-1).

4.8 Alertas, reportes y exportación

4.8.1 Alertas

Tipos:

• Nueva coincidencia (nuevo match).
• Cambio relevante:
• cambio de estado,
• cambio de fecha de cierre,
• nuevos ítems/cambios en items (si aplica).
• Alertas por umbral de cierre:
• 0–10 destacado
• T-7/T-3/T-1 configurable

Canales (producto completo):

• Email (mínimo)
• Web notifications
• Integración con mensajería corporativa (opcional)

4.8.2 Reportes

• **Reporte diario programado** (por defecto):
• incluye cierres 0–30 días
• sección destacada 0–10
• Reportes personalizados:
• por regla, por comprador, por región, por categoría.
• Contenidos mínimos por fila:
• tipo entidad, código, título, comprador, región, estado, cierre, días a cierre, score, razones.

4.8.3 Exportación

• CSV y XLSX (mínimo)
• API interna de exportación para integraciones futuras.

4.9 Administración y configuración

4.9.1 Configuración de ingesta

• Frecuencia de jobs (diaria, activas, backfill)
• Ventanas horarias
• Presupuesto de hits
• TTL de caché de detalle

4.9.2 Configuración de reglas

• Editor de reglas (UI)
• Prioridades por regla
• Versionado y rollback (producto completo)

4.9.3 Configuración de pipeline

• Estados y transiciones
• Plantillas de checklist
• Campos personalizados (producto completo)

4.9.4 Auditoría

• Registro de:
• cambios de configuración
• cambios en oportunidades
• accesos y exportaciones

---

5. Interrelación de datos y flujo de información

5.1 Diagrama conceptual (texto)

1. **Conector API** trae **RawPayload** de:
2. Licitación

3. Orden de compra
4. Comprador/Proveedor (lookup)
5. **Normalizador** transforma a entidades internas:
6. `Tender`
7. `PurchaseOrder`
8. `Party` (Buyer/Supplier)
9. **Motor de reglas** evalúa `Tender` / `PurchaseOrder` → genera `MatchResult`
10. **Inbox** muestra `MatchResult` priorizado
11. Usuario crea `Opportunity` desde `Tender` / `PurchaseOrder`
12. `Opportunity` se gestiona en `Pipeline`, con `Task`, `Comment`, `Attachment`
13. **Notificador** genera `Alert` basado en eventos (nuevo match, cambios, umbrales)
14. **Report Engine** genera `ReportRun` y exporta/entrega.

5.2 Reglas de vinculación

- Un `Tender` puede tener 0..N `MatchResult` (por diferentes reglas).
- Un `PurchaseOrder` puede tener 0..N `MatchResult`.
- Un `Opportunity` referencia exactamente 1 entidad origen primaria (Tender u OC) y puede vincular N entidades secundarias.
- Un `Party` se vincula como comprador/proveedor en entidades.

5.3 Eventos y triggers

- Evento `ENTITY_UPSERTED` cuando cambia hash de entidad.
- Evento `MATCH_CREATED` cuando un match es nuevo.
- Evento `OPPORTUNITY_STATE_CHANGED`.
- Evento `DEADLINE_THRESHOLD` al cruzar umbral.

---

6. Modelo de datos (detallado)

> Nota: los nombres de campos exactos de la API pueden variar; el sistema guarda raw payload para trazabilidad y adapta el mapeo.

6.1 Tablas/colecciones principales

6.1.1 `raw_payloads`

- `id` (uuid)
- `source` (tender|purchase_order|lookup)
- `source_key` (codigo_externo/codigo/etc.)
- `fetched_at`
- `payload` (json)
- `parser_version`
- `request_fingerprint` (para deduplicar requests)

6.1.2 `tenders`

- `id` (uuid)
- `external_code` (string, unique)

- `name`
- `description`
- `status_code` / `status_text`
- `type_code` / `type_text`
- `published_at` (si existe)
- `close_at`
- `days_to_close`
- `close_bucket` (enum)
- `buyer_party_id` (fk)
- `buyer_org_code` (string)
- `buyer_region`
- `currency`
- `items` (jsonb o tabla `tender_items`)
- `awards` (jsonb o tabla `tender_awards`)
- `last_seen_at` (control de stale)
- `hash` (para cambios)
- `data_quality_flag` (ok|warning|error)
- índices: (close_at), (buyer_org_code), (status_code), full-text (name/description)

6.1.3 `tender_items`

- `id`
- `tender_id`
- `line_number`
- `category_code`
- `product_code` (si existe)
- `description`
- `quantity`
- `unit`
- `amount` (si existe)
- `currency` (si existe)

6.1.4 `purchase_orders`

- `id` (uuid)
- `external_code` (string, unique)
- `status`
- `type_code` (incluye compra ágil si aplica)
- `created_at`
- `sent_at` / `accepted_at` / `cancelled_at` (si existen)
- `buyer_party_id` (fk)
- `supplier_party_id` (fk)
- `total_amount`
- `tax_amount`
- `net_amount`
- `discounts` / `charges` (si existen)
- `currency`
- `items` (jsonb o tabla `purchase_order_items`)
- `hash`
- `last_seen_at`

6.1.5 `purchase_order_items`

- `id`
- `purchase_order_id`
- `line_number`
- `description`
- `quantity`
- `unit`
- `unit_price`
- `line_total`
- `category_code` (si existe)

6.1.6 `parties`

- `id` (uuid)
- `party_type` (buyer|supplier|both)
- `internal_code` (codigo empresa)
- `rut` (si existe)
- `name`
- `region` (si existe)
- `metadata` (jsonb)
- índices: (internal_code), (rut)

6.1.7 `rules`

- `id`
- `workspace_id`
- `name`
- `is_enabled`
- `priority`
- `definition` (json): keywords, negatives, categories, buyers, regions, types, statuses, date windows, amount ranges
- `weights` (json)
- `created_by`, `created_at`, `updated_at`, `version`

6.1.8 `match_results`

- `id`
- `workspace_id`
- `entity_type` (tender|purchase_order)
- `entity_id`
- `rule_id`
- `score`
- `reasons` (json array)
- `created_at`
- `is_new` (boolean)
- índices: (entity_type, entity_id), (rule_id), (created_at)

6.1.9 `opportunities`

- `id`

- `workspace_id`
- `origin_entity_type` (tender|purchase_order)
- `origin_entity_id`
- `title` (por defecto heredado)
- `owner_user_id`
- `pipeline_state`
- `probability` (0–100)
- `estimated_value`
- `tags` (array)
- `notes` (text)
- `created_at`, `updated_at`
- `closed_at` (si aplica)
- `outcome` (won|lost|archived|none)

6.1.10 `opportunity_links`

- `id`
- `opportunity_id`
- `entity_type` (tender|purchase_order)
- `entity_id`
- `relation` (supporting|related|reference)

6.1.11 `tasks`

- `id`
- `opportunity_id`
- `title`
- `due_at`
- `status` (open|done|cancelled)
- `assignee_user_id`
- `created_at`

6.1.12 `comments`

- `id`
- `opportunity_id`
- `author_user_id`
- `body`
- `created_at`

6.1.13 `alerts`

- `id`
- `workspace_id`
- `type` (new_match|entity_changed|deadline_threshold|assignment)
- `payload` (json)
- `channel` (email|web|...)
- `status` (queued|sent|failed)
- `created_at`, `sent_at`

6.1.14 `report_runs`

- `id`
- `workspace_id`
- `report_type` (daily|custom)
- `parameters` (json)
- `generated_at`
- `storage_ref` (ruta archivo)
- `delivery_status`

6.2 Reglas de consistencia

- Un `match_result` no debe duplicarse para (workspace, entity, rule) si el score/razones no cambian.
- Una `opportunity` por defecto es única por (workspace, origin_entity_type, origin_entity_id) salvo override.

---

7. Arquitectura de alto nivel

7.1 Componentes

- **API Backend:** expone endpoints internos para UI, reglas, oportunidades, reportes.
- **Job Worker/Scheduler:** ejecuta sincronizaciones, refrescos, backfills y generación de reportes.
- **Conector Mercado Público:** módulo aislado, con manejo de cuota, caché y reintentos.
- **DB:** Postgres (recomendado) + JSONB para raw payload y arrays.
- **Search (opcional):** motor de búsqueda si full-text nativo no alcanza.
- **Notificador:** envío email y notificaciones web.
- **Frontend:** web app (Inbox, fichas, pipeline, configuración).

7.2 Consideraciones operativas

- Configurar **secrets manager** para ticket.
- Observabilidad (metrics + logs + trazas).
- Feature flags para rollout.

---

8. API interna (contratos mínimos)

8.1 Autenticación

- JWT/SAML/OAuth corporativo (definir). MVP: JWT con login.

8.2 Endpoints (resumen)

- `GET /inbox?bucket=urgent|next|later&filters...`
- `GET /tenders/{id}` / `POST /tenders/{id}/refresh_detail`
- `GET /purchase-orders/{id}` / `POST /purchase-orders/{id}/refresh_detail`
- `POST /opportunities` (desde tender/oc)
- `PATCH /opportunities/{id}` (estado, owner, notas, tags)
- `POST /opportunities/{id}/tasks`

- `POST /opportunities/{id}/comments`
- `GET /rules` / `POST /rules` / `PATCH /rules/{id}`
- `POST /reports/daily/run` / `GET /reports/{id}/download`
- `GET /admin/quota` / `PATCH /admin/quota`

8.3 Respuestas y paginación

- Paginación por cursor.
- Respuestas con `meta` (total, page_size, cursor).

---

9. UI/UX: pantallas y flujos

9.1 Pantallas

1. **Dashboard/Inbox**
2. **Ficha Licitación**
3. **Ficha Orden de Compra**
4. **Pipeline**
5. **Oportunidad (detalle)**
6. **Reglas (editor y listado)**
7. **Reportes (historial y parámetros)**
8. **Admin (cuota, jobs, usuarios, seguridad)**

9.2 Flujos clave

- Configuración inicial:
- Admin ingresa ticket → define cuota y cronograma → crea reglas → invita usuarios.
- Operación diaria:
- Jobs corren → Inbox se llena → analista guarda oportunidades → asigna → ejecuta pipeline → reporta.
- Gestión de cierre:
- umbrales generan tareas/alertas → responsable actúa → registra outcome.

---

10. Requisitos no funcionales

10.1 Seguridad

- Ticket cifrado, nunca en logs.
- RBAC estricto.
- Auditoría de exportaciones.

10.2 Rendimiento

- Inbox debe cargar en < 2 s para 1.000 resultados (paginado).
- Fichas deben cargar en < 2 s usando caché; refresh bajo demanda.

10.3 Resiliencia

- Reintentos con backoff.

- Jobs idempotentes.
- Degradación controlada: si cuota baja, priorizar urgentes.

10.4 Cumplimiento

- Atribución de fuente donde corresponda.
- Política de retención de datos (configurable).

---

11. Métricas y analítica

- Cobertura: (# entidades ingeridas / # estimadas consultadas) por día.
- Time-to-signal: publicación/actualización → match/alert.
- Consumo: hits/día, hits por job, costo por usuario.
- Conversión: oportunidades creadas, tasas por estado, ganadas/perdidas.

---

12. Calidad, pruebas y criterios de aceptación

12.1 Pruebas

- Unit tests: parser/normalizador, motor de reglas, score.
- Integration tests: conector (mock), rate limit, reintentos.
- E2E: jobs → inbox → crear oportunidad → pipeline → reporte.

12.2 Criterios de aceptación (producto funcional)

- El sistema ingesta licitaciones y OCs diariamente y refresca activas sin exceder cuota configurada.
- Cada registro en Inbox muestra bucket (0–10/11–30/31+), score y razones.
- Crear oportunidad desde una licitación/OC en 1 clic y gestionarla en pipeline.
- Alertas por nueva coincidencia y por umbral de cierre funcionan (al menos por email).
- Reporte diario programado genera archivo descargable con columnas mínimas.
- Auditoría registra cambios de estado/asignación y exportaciones.

---

13. Riesgos y mitigaciones

- **Cuota insuficiente** al escalar: mitigación con caché, priorización, reducción de refrescos, optimización de backfill.
- **Cambios en schema de API:** mitigación guardando raw payload + parser versionado + pruebas de contrato.
- **Datos incompletos (p. ej., montos/categorías faltantes):** mitigación con flags de calidad y UI de “dato no disponible”.
- **Requerimientos de compra ágil pre-OC:** si el proceso requiere detectar antes de emitirse una OC y la API no lo expone, se deberá definir fuente oficial adicional o estrategia alternativa.

---

14. Entregables técnicos esperados

• Repositorio con:
• módulo conector
• esquema DB + migraciones
• workers/jobs
• backend API
• frontend
• Documentación:
• guía de despliegue
• runbook de cuota/operación
• manual de usuario
• especificación de reglas

---

15. Glosario

• **Entidad:** licitación u orden de compra ingerida.
• **Match:** resultado de evaluar reglas sobre entidad.
• **Oportunidad:** registro de trabajo interno asociado a una entidad.
• **Bucket de cierre:** grupo por días a cierre (0–10 / 11–30 / 31+).
• **Cuota:** límite diario de consultas (hits) del ticket.