SEARCH OS (Tool 2)

Master Technical Specification Document

Metadatos	Detalle
Versión	**7.0 (Final Release Candidate)**
Propósito	Plataforma SaaS de Cualificación y Gestión de Dealflow (CRM Analítico)
Core Value	Transformar listados estáticos (SABI) en inteligencia accionable mediante IA flexible.
Stack	Python, Streamlit (AgGrid), LangChain, APIs (Groq/DeepInfra/OpenAI/Google).

1. VISIÓN EJECUTIVA Y GUÍA DE USO

1.1 ¿Qué es la Tool 2?

Es un **"Excel Inteligente con Cerebros Intercambiables"**. Su función es ingerir listados masivos de empresas (SABI), enriquecerlos automáticamente con datos de internet (Scoring IA) y permitir su gestión como un CRM (Etiquetas, Filtros, Chat).

1.2 Flujo de Usuario (The Golden Path)

1. **Ingesta (Import):** El usuario arrastra un archivo `.csv` o `.xlsx` de SABI. El sistema detecta cabeceras clave (Nombre, Web, EBITDA).
2. **Enriquecimiento (AI Columns):**
   - El usuario crea columnas inteligentes (ej. *"¿Existe Relevo Generacional?"*).
   - Selecciona el modelo de IA según la necesidad: **Velocidad** (Groq) vs. **Profundidad** (GPT-4o) vs. **Volumen** (DeepInfra).

- El sistema busca en internet y rellena la celda con un Score (0-10) y una explicación.

3. **Gestión (CRM):**
   - El usuario etiqueta empresas ("Interesante", "Descartado") usando tags visuales.
   - Mueve empresas entre listas ("Longlist" → "Shortlist").
4. **Consulta (Data Chat):** Un asistente lateral permite "hablar" con la tabla ("Fíltrame las empresas de Valencia con EBITDA > 1M€").

---

2. ESPECIFICACIÓN DE UI/UX (FRONTEND)

2.1 La Pantalla Principal: "The Intelligent Grid"

Diseño visual estilo **Airtable** o **Linear**, implementado con **AgGrid**.

- **Estructura:**
  - **Barra Superior (Toolbar):**
    - Botones: `[+ Nueva Columna IA]`, `[+ Nueva Etiqueta]`.
    - Tabs de Navegación: `[📥 Longlist]` `[⭐ Shortlist]` `[🗑️ Descartados]`.
  - **Barra Flotante (Action Bar):** Aparece al seleccionar filas (checkboxes) en la parte inferior. Permite acciones en lote: `Mover`, `Borrar`, `Ejecutar IA`.
- **Anatomía de las Columnas (Cell Renderers):**
  - **Identidad (Fija Izq):** Nombre en negrita + Link Web (↗️) + Logo/Favicon.
  - **Financieras:** Formato numérico compacto y monoespaciado (`1.5M €`). Alineación derecha.
  - **AI Scores (Semáforo):** Badges visuales coloreados según el valor:
    - 🟢 **8-10:** Verde (Target Ideal).
    - 🟡 **5-7:** Amarillo (Duda).
    - 🔴 **0-4:** Rojo (Descarte).
    - *Tooltip:* Al pasar el mouse, muestra la justificación generada por la IA.

- **Tags:** Estilo "Notion" (Badges pasteles). Editables con un clic (Dropdown nativo).

2.2 Modal de Creación de Columna IA

El punto crítico donde el usuario configura la inteligencia.

- **Input:** Título de la Columna (ej. "Riesgo Regulatorio").
- **Input:** Prompt (Instrucciones para la IA).
- **Selector de Modelo (Dropdown):**
  - ⚡ **Instantáneo:** Groq (Llama-3-70B). *Default.*
  - 🏭 **Batch / Económico:** DeepInfra (Llama-3/Mixtral).
  - 🧠 **Razonamiento Complejo:** OpenAI (GPT-4o).
  - 📚 **Contexto Largo:** Google (Gemini 1.5 Pro).
- **Switch:** `Auto-optimizar columnas (Smart Context)` (Activado por defecto).

2.3 Asistente Lateral (Data Chat)

Panel desplegable a la derecha. Usa **Groq** por defecto para respuestas instantáneas sobre los datos visibles en la tabla actual.

---

3. ARQUITECTURA TÉCNICA (BACKEND)

3.1 LLM Factory (El Enrutador)

Patrón de diseño *Factory* para instanciar el cliente LLM correcto según la elección del usuario en la UI.

Python

```
from langchain_groq import ChatGroq
from langchain_openai import ChatOpenAI
from langchain_google_genai import ChatGoogleGenerativeAI

class LLMFactory:
    @staticmethod
    def get_model(ui_selection, api_keys):
        # 1. INSTANTÁNEO (GROQ) - Ideal UX interactiva
        if ui_selection == "instant":
            return ChatGroq(model_name="llama3-70b-8192", groq_api_key=api_keys['GROQ'], temperature=0.1)
```

```
    # 2. BATCH (DEEPINFRA) - Ideal volumen (usa cliente OpenAI compatible)
    elif ui_selection == "batch":
        return ChatOpenAI(model="meta-llama/Meta-Llama-3-70B-Instruct",
                          api_key=api_keys['DEEPINFRA'],
                          base_url="https://api.deepinfra.com/v1/openai", temperature=0.1)

    # 3. RAZONAMIENTO (OPENAI)
    elif ui_selection == "complex":
        return ChatOpenAI(model="gpt-4o", api_key=api_keys['OPENAI'])

    # 4. CONTEXTO (GOOGLE)
    elif ui_selection == "long_context":
        return ChatGoogleGenerativeAI(model="gemini-1.5-pro", google_api_key=api_keys['GOOGLE'])
```

3.2 Optimizador de Contexto (Smart Context)

Agente intermedio que reduce el consumo de tokens seleccionando solo las columnas necesarias del Excel antes de enviar el prompt al LLM principal.

- **Modelo:** Usa `Llama-3-8B` (Groq) por ser casi gratuito e instantáneo.
- **Prompt del Sistema:**Python

  ```
  CONTEXT_OPTIMIZER_SYSTEM_PROMPT = """
  ERES UN INGENIERO DE DATOS OPTIMIZADOR.
  TU OBJETIVO: Seleccionar del dataset únicamente las columnas estrictamente necesarias para responder al prompt del usuario.
  INPUTS: Prompt Usuario + Lista de Headers del CSV.
  REGLA: Sé minimalista. Incluye siempre 'Nombre' y 'Web'. Devuelve JSON Array de strings.
  """
  ```

3.3 Pipeline de Ejecución (Lógica Batch)

Cuando el usuario ejecuta una columna sobre 1.000 filas:

1. **Planificación:** El *Smart Context* determina qué columnas leer (ej. solo `EBITDA` y `Nombre`).
2. **Construcción:** Se crea un JSON ligero por fila.
3. **Enriquecimiento:** (Opcional) Se buscan datos en Tavily.
4. **Inferencia:** Se llama al modelo seleccionado en la `LLMFactory`.
5. **Resultado:** Se actualiza el `SessionState` y la Grid se refresca.

---

4. MODELO DE DATOS (JSON STATE)

La estructura de datos persistente en `st.session_state` para soportar columnas dinámicas y configuración de modelos.

JSON

```
{
 "project_id": "Sector_Vitivinicola_2025",
 "data_lists": {
  "longlist": [ {"id": "B123", "Nombre": "Bodegas X", "EBITDA": 500000, "ai_relevo": 8} ],
  "shortlist": [],
  "discarded": []
 },
 "column_definitions": [
  {
   "field": "ai_relevo",
   "title": "Relevo Generacional",
   "type": "ai_score",
   "config": {
    "user_prompt": "Analiza edad administradores...",
    "model_selected": "instant", // Guardamos la elección
    "smart_context": true
   }
  },
  {
   "field": "tag_estado",
   "title": "Estado",
   "type": "tag_select",
   "options": ["Pendiente", "Contactado", "NDA"]
  }
 ]
}
```

5. CONFIGURACIÓN VISUAL (SNIPPET AGGRID)

Configuración crítica para los desarrolladores Frontend (Streamlit) para lograr el look & feel deseado.

Python

```
grid_options = {
  "rowHeight": 50,
  "headerHeight": 45,
  "animateRows": True,
  "rowSelection": "multiple", # Permite selección múltiple con checkboxes
  "suppressCellFocus": True,
  "columnDefs": [
    {
      "field": "Nombre",
      "pinned": "left",
      "width": 250,
```

```
        "cellRenderer": "CompanyLogoRenderer"
    },
    {
        "field": "[AI] Relevo",
        "cellRenderer": "ScoreBadgeRenderer", # Renderiza el Badge de color
        "tooltipField": "relevo_reason"      # Tooltip con la explicación
    },
    {
        "field": "Estado",
        "editable": True,
        "cellEditor": "agSelectCellEditor",
        "cellEditorParams": {"values": ["Longlist", "Contactado", "NDA"]}
    }
  ]
}
```

---

6. ROADMAP DE IMPLEMENTACIÓN

1. **Fase 1 (Core Data):** Ingesta de CSV + Renderizado de AgGrid básico + Persistencia de columnas manuales (Tags).
2. **Fase 2 (AI Engine):** Implementar `LLMFactory` y conectividad con las 4 APIs.
3. **Fase 3 (Optimization):** Implementar el `SmartContextOptimizer` para filtrar JSONs antes de enviar.
4. **Fase 4 (UX Polish):** Badges de colores (Semáforo), Tooltips, Barra de Acciones Flotante y Asistente Lateral (Groq).