SEARCH OS (Tool 1)

Versión: 2.1 (Definitiva)
Rol: Discovery Engine (Analista Sectorial Automatizado)
Stack Core: Python, Google Gemini 1.5 Pro API, Tavily API.

---

1. VISIÓN Y ALCANCE

El objetivo es construir una aplicación web de **"Copiloto de Inversión"** que automatice las Fases 0, 1 y 2 del proceso de búsqueda de un Search Fund.

La herramienta no es un simple chat. Es un **flujo de trabajo estructurado** que:

1. Recibe una idea de inversión (Sector).
2. Investiga autónomamente en internet (usando Agentes).
3. Genera un **Análisis Sectorial Detallado** (Markdown) en tiempo real.
4. Permite la iteración y refinamiento mediante una interfaz de **Pantalla Dividida (Split-Screen)**.

Restricción Crítica (The Hard Constraint):
El sistema debe operar bajo la **Tesis de Inversión de Emerita** (hardcoded), rechazando o penalizando sectores que no cumplan con métricas de EBITDA, fragmentación o digitalización.

---

2. ARQUITECTURA DEL SISTEMA

2.1 Stack Tecnológico Recomendado

- **Frontend:** Streamlit (Layout `st.columns([1, 2])`) para el MVP.
- **Backend / Orquestador:** Python 3.10+.

- **Inteligencia Artificial (Cerebro):** `google-generative-ai` (Modelo: `gemini-1.5-pro-latest`). Se elige por su ventana de contexto (2M tokens) y capacidad de razonamiento.
- **Búsqueda Externa (Datos):** `tavily-python`. Motor de búsqueda optimizado para RAG (Retrieval-Augmented Generation).

2.2 Diagrama de Flujo de Datos

Fragmento de código

```
[Usuario] --(Input Sector)--> [Orquestador Python]
    |                              |
    |                    [Agente Clasificador (CNAE)]
    |                              |
    |                    [Agente Investigador (Tavily)] <--(Internet)
    |                              |
    |                    [Agente Analista (Gemini)] <--(Tesis Emerita)
    |                              |
    v                              v
[UI Izquierda: Chat] <---> [SESSION STATE (JSON)] <---> [UI Derecha: Markdown]
```

---

3. DEFINICIÓN FUNCIONAL: UI/UX (PANTALLA DIVIDIDA)

La interfaz se divide estrictamente en dos columnas verticales.

3.1 Panel Izquierdo (30-40%): El Centro de Mando

Este panel es dinámico y tiene dos estados lógicos:

Estado A: Configuración (Input Inicial)

- **Campo Texto:** "Nombre del Sector / Nicho" (ej. *Mantenimiento de Ascensores*).
- **Campo Texto (Opcional):** "Contexto Adicional" (ej. *Centrarse en mantenimiento, no instalación*).
- **Botón CTA:** "ARRANCAR ANÁLISIS".

Estado B: Copiloto Interactivo (Chat)
Se activa tras la generación del primer borrador.

- **Historial de Chat:** Muestra la conversación.

- **Input de Chat:** Permite al usuario dar órdenes de refinamiento.
- **Capacidades:**
  - *Consulta (RAG):* "¿Qué margen EBITDA mencionas en la sección 2?".
  - *Mutación (Edit):* "La sección de Competencia está incompleta, añade a Indra y reescríbela".

3.2 Panel Derecho (60-70%): El Lienzo (Canvas)

- **Visor de Documento:** Renderiza el contenido en Markdown limpio.
- **Estructura Modular:** El documento no es un bloque de texto único. Internamente es un objeto JSON, lo que permite actualizar secciones específicas independientemente.
- **Feedback Visual:** Si el usuario pide un cambio en la Sección 3, solo esa sección muestra un spinner de "Actualizando..." mientras el resto permanece estático.

---

4. LÓGICA DE NEGOCIO: EL PIPELINE DE AGENTES

El backend debe ejecutar esta secuencia lógica (Fases 0-2).

Paso 1: Mapping y Clasificación

- **Acción:** El LLM consulta su base de conocimiento interna para mapear el sector a códigos industriales **CNAE 2009** (España).
- **Output:** Código primario y secundarios (Ej. *CNAE 4322 - Fontanería*).

Paso 2: Investigación Profunda (Market Research)

- **Acción:** Generación de queries de búsqueda vía Tavily.
- **Objetivos de Búsqueda (Hardcoded):**
  1. *Tamaño:* "Tamaño mercado [Sector] España 2024 facturación".
  2. *Fragmentación:* "Cuota de mercado líderes [Sector] España" (Alerta si Top 3 > 50%).
  3. *Rentabilidad:* "Márgenes EBITDA sector [Sector] España".

4. *Tendencias:* "Normativa y Riesgos [Sector] España".

Paso 3: Análisis Financiero (Inferencia)

- **Reglas de Tesis (Filtros):**
  - Margen EBITDA objetivo: **>15%**.
  - Margen Bruto objetivo: **>40%**.
  - Conversión Caja: **>70%**.
- *Nota:* Si los datos encontrados indican márgenes bajos, el sistema debe marcarlo como "ROJO" o "ÁMBAR" en el reporte final.

Paso 4: Generación y Estructuración

El LLM redacta el informe siguiendo estrictamente la plantilla JSON definida en la Sección 5.

---

5. MODELO DE DATOS Y PLANTILLA

El estado de la aplicación (`SessionState`) debe mantener esta estructura JSON para el documento. Esto es fundamental para que la UI funcione por módulos.

JSON

```
{
 "meta": {
  "sector_name": "Logística de Frío",
  "cnae_codes": ["5210"],
  "verdict": "ÁMBAR",
  "timestamp": "2025-01-08"
 },
 "sections": {
  "1_executive_summary": { "title": "1. Resumen Ejecutivo y Semáforo", "content": "..." },
  "2_financials": { "title": "2. Unit Economics y Financieros", "content": "..." },
  "3_market_size": { "title": "3. Tamaño y Segmentación", "content": "..." },
  "4_value_chain": { "title": "4. Cadena de Valor", "content": "..." },
  "5_competition": { "title": "5. Estructura Competitiva", "content": "..." },
  "6_regulations": { "title": "6. Regulación y Riesgos", "content": "..." },
  "7_opportunities": { "title": "7. Oportunidades (Tesis Emerita)", "content": "..." },
  "8_gtm_targets": { "title": "8. Hipótesis y Targets", "content": "..." },
  "9_conclusion": { "title": "9. Conclusión Estructurada", "content": "..." },
  "10_sourcing_signals": { "title": "10. Señales de Búsqueda", "content": "..." }
 }
}
```

---

6. LÓGICA DE INTERACCIÓN (CHAT & EDICIÓN)

Esta sección define cómo funciona el Panel Izquierdo (Chat).

6.1 Prompt del Sistema para el Chat

El LLM del chat debe tener acceso a herramientas (`function calling`) para manipular el JSON del estado.

Python

```
SYSTEM_PROMPT_CHAT = """
Eres el Copiloto de Inversión de Emerita.
Tienes acceso de lectura/escritura al documento de análisis que el usuario ve a su derecha.

TUS HERRAMIENTAS (TOOLS):
1. `read_full_document()`: Lee el contenido actual.
2. `search_internet(query)`: Usa Tavily para buscar datos frescos.
3. `update_section(section_key, user_instruction)`: Llama a un sub-agente para reescribir UNA sección específica.

REGLAS DE COMPORTAMIENTO:
- Si el usuario pide un cambio ("Añade esto", "Corrige esto"), USA `update_section`. NO respondas solo con texto.
- Si el usuario pregunta ("¿Qué significa esto?"), responde conversacionalmente.
- Mantén siempre el rigor del Manifiesto Emerita (EBITDA >15%, B2B, España).
"""
```

6.2 Función de Actualización (`update_section`)

Lógica:

1. **Recibir:** `section_key` (ej. '6_regulations') y `instruction` (ej. "Añade impacto de normativa NIS2").
2. **Leer:** Obtener el `content` actual de esa sección.
3. **Generar:** LLM reescribe *solo* esa sección integrando la nueva instrucción y datos.
4. **Guardar:** Actualizar el JSON del estado.
5. **Refrescar:** La UI derecha detecta el cambio en el JSON y re-renderiza solo ese bloque.

---

7. INPUTS DE SISTEMA (HARDCODED THESIS)

Para asegurar la calidad "Emerita", estas constantes deben inyectarse en cada llamada al LLM (tanto en la generación inicial como en el chat).

Variables de Tesis (Search Fund DNA):

- **TARGET_GEO:** "España (Prioritario), Europa Occidental (Secundario)".
- **TARGET_SIZE:** "Ventas 5-40M€, EBITDA 1-5M€".
- **TARGET_MARGINS:** "EBITDA >= 15%, Bruto >= 40%".
- **DEAL_KILLERS:** "Riesgo tecnológico alto, Dependencia de un solo cliente, Sector en declive, Márgenes muy bajos (<10%)".
- **VALUE_LEVERS (Palancas):** "Digitalización (pasar de papel a software), Profesionalización comercial (Outbound), Eficiencia operativa".

---

8. DETALLE DE IMPLEMENTACIÓN (HANDOFF DEV)

8.1 Prompt Maestro (Análisis Inicial)

Este prompt genera el JSON inicial en la Fase 4 del pipeline.

Python

```
INITIAL_SYSTEM_PROMPT = f"""
ERES EL CIO (DIRECTOR DE INVERSIONES) DE 'EMERITA', UN SEARCH FUND DE ÉLITE.
{EMERITA_THESIS_CONSTRAINTS}

TU TAREA:
Analizar los datos proporcionados y generar un INFORME SECTORIAL PRELIMINAR.
Tu tono debe ser escéptico, profesional y basado en datos. Debes juzgar, no solo describir.

ESTRUCTURA DE RESPUESTA (JSON OBLIGATORIO):
Debes devolver un JSON válido donde las claves sean exactamente:
"1_executive_summary", "2_financials", "3_market_size", "4_value_chain",
"5_competition", "6_regulations", "7_opportunities", "8_gtm_targets",
"9_conclusion", "10_sourcing_signals".

CONTENIDO POR SECCIÓN:
1. SUMMARY: Veredicto final (VERDE/ÁMBAR/ROJO) justificado.
2. FINANCIALS: Estructura de costes, márgenes reales vs tesis, ciclo de caja.
... (Resto de secciones según especificación) ...
10. SOURCING: Qué buscar en Google/LinkedIn para encontrar al target (Señales).

INSTRUCCIONES CLAVE:
- Si el margen es <15%, el semáforo debe ser ROJO o ÁMBAR.
- Prioriza datos de España.
```

- Formato del contenido dentro del JSON: Markdown.
"""

PROMPT MAESTRO: GENERADOR DE ANÁLISIS INICIAL (Fase 1)

INITIAL_ANALYSIS_SYSTEM_PROMPT = """
ERES EL DIRECTOR DE INVERSIONES (CIO) DE 'EMERITA', UN SEARCH FUND DE ÉLITE EN ESPAÑA.

TU MISIÓN:
Tu objetivo no es solo informar, sino JUZGAR. Debes redactar un Análisis Sectorial Preliminar basado en los datos que te proporciono, determinando si el sector es apto para adquirir una PYME.

1. EL ADN DE EMERITA (CRITERIOS INNEGABLES)

Debes evaluar el sector usando ESTRICTAMENTE estos filtros. Si el sector falla en alguno, debes destacarlo como un RIESGO CRÍTICO (Red Flag).

- **TAMAÑO TARGET:** Buscamos nichos donde existan empresas facturando 5-40M€ con EBITDA 1-5M€.
- **RENTABILIDAD:** Margen Bruto >40% y Margen EBITDA >15% (Esencial).
- **MODELO:** B2B, ingresos recurrentes, bajo riesgo tecnológico (evitar startups/disrupción).
- **CAJA:** Negocios "Asset Light". Capex bajo (<5% ventas). Conversión de caja >70%.
- **DINÁMICA:** Sectores fragmentados (sin un líder con >20% cuota) y poco digitalizados.

2. INSTRUCCIONES DE REDACCIÓN

- **Tono:** Profesional, escéptico, directo. Evita adjetivos vacíos ("interesante", "bueno"). Usa datos.
- **Formato:** Markdown estricto. Usa negritas para cifras clave.
- **Idioma:** Español de Negocios (España).

3. ESTRUCTURA DEL INFORME (OBLIGATORIA)

Debes generar un JSON con las siguientes secciones. NO omitas ninguna.

1. **RESUMEN EJECUTIVO Y SEMÁFORO:** * Conclusión directa.
    - VEREDICTO: [VERDE / ÁMBAR / ROJO]. Justifica el color basándote en márgenes y fragmentación.
2. **UNIT ECONOMICS Y FINANCIEROS:** * ¿Cómo gana dinero una empresa aquí? Desglose de costes típicos.
    - Márgenes medios observados (Bruto y EBITDA).
    - Ciclo de caja (¿Quién financia a quién?).
3. **TAMAÑO Y SEGMENTACIÓN:**
    - Estimación TAM/SAM en España. Tendencia (CAGR).

- Segmentos más atractivos para un Search Fund.

4. **CADENA DE VALOR:**
   - Diagrama textual (Proveedor → Fabricante → Distribuidor → Cliente).
   - ¿Quién tiene la sartén por el mango (poder de negociación)?
5. **ESTRUCTURA COMPETITIVA:**
   - Grado de concentración.
   - Menciona nombres de competidores reales encontrados en el contexto.
   - Busca el "Arquetipo Ideal": Empresa familiar, 20+ años historia, dueños >60 años.
6. **REGULACIÓN Y RIESGOS:**
   - Barreras de entrada reales. Normativas críticas (ej. ISOs, Marcado CE).
   - Deal Killers (Riesgos que harían cancelar la operación).
7. **OPORTUNIDADES (PALANCAS DE VALOR):**
   - ¿Dónde está la ineficiencia? (Ej. Procesos en papel, sin equipo comercial).
   - ¿Cómo puede Emerita multiplicar el EBITDA?
8. **HIPÓTESIS Y TARGETS:**
   - Perfil de la empresa ideal a buscar.
   - Lista de 3-5 empresas reales identificadas en la búsqueda (si las hay).
9. **SEÑALES DE BÚSQUEDA (SOURCING SIGNALS):**
   - Instrucciones para el analista junior/scrapper.
   - Ej: "Buscar empresas con webs Copyright 2015", "Buscar en asociaciones de instaladores".

4. INPUT DE DATOS (CONTEXTO)

Utiliza EXCLUSIVAMENTE la siguiente información recopilada por el Agente de Investigación.
Si falta un dato crítico, indícalo como "DATO NO DISPONIBLE - REQUIERE ENTREVISTA EXPERTA", no te lo inventes.

[AQUÍ SE INYECTARÁN LOS RESULTADOS DE TAVILY/GOOGLE]
"""

8.2 Snippet de Lógica de Edición

Este código maneja la solicitud de edición del usuario desde el chat.

Python

```
def update_section_logic(current_doc, section_id, user_prompt):
  # 1. Contexto actual
  old_content = current_doc['sections'][section_id]['content']

  # 2. Decisión de búsqueda (Agente ligero)
  # (Omitido por brevedad: aquí iría lógica check_if_search_needed)

  # 3. Reescribir (Agente Editor)
  prompt = f"""
  ACTÚA COMO UN EDITOR SENIOR DE PRIVATE EQUITY.
  TAREA: Actualizar la sección '{section_id}' del informe.

  CONTENIDO ORIGINAL:
  {old_content}

  INSTRUCCIÓN DEL USUARIO:
  "{user_prompt}"

  REGLA: Devuelve la sección completa reescrita en Markdown. Mantén el tono profesional.
  """

  new_content = gemini_model.generate_content(prompt).text
  return new_content
```

def generate_initial_report(sector_name, search_data_context):
"""
Genera el primer borrador del análisis completo.
"""

```
# 1. Preparar el Prompt final inyectando los datos
final_prompt = f"""
SECTOR A ANALIZAR: {sector_name}

DATOS DE INVESTIGACIÓN (CONTEXTO):
{search_data_context}

Genera el informe completo siguiendo la estructura JSON definida en las instr
ucciones del sistema.
"""

# 2. Llamada al Modelo (Gemini 1.5 Pro)
response = model.generate_content(
```

```
    contents=final_prompt,
    system_instruction=INITIAL_ANALYSIS_SYSTEM_PROMPT, # ←—— AQUÍ V
A EL PROMPT MAESTRO
    generation_config={
        "temperature": 0.3, # Baja temperatura para ser factual y riguroso
        "response_mime_type": "application/json" # Forzamos JSON para facilita
r el parsing en la UI
    }
)

return response.text
```

---

9. ENTREGABLES FINALES (OUTPUTS)

1. **Documento Markdown (.md):** Descargable, formateado para Notion.
2. **JSON Estructurado:** Para alimentar futuras herramientas del SEARCH OS (como el Scrapper de targets).
3. **Semáforo de Inversión:** Un veredicto claro al final del reporte:
    - 🔴 **ROJO:** Descartar (Deal Killers encontrados).
    - 🟡 **ÁMBAR:** Dudoso (Requiere hablar con expertos).
    - 🟢 **VERDE:** Avanzar a Fase 3 (Análisis Detallado / Sourcing).

---

10. LÓGICA DE IMPLEMENTACIÓN DEL PIPELINE

Paso 1: Investigación (Tavily)

El sistema no debe buscar a ciegas. Debe usar estas *queries* base para cada sector:

Python

```
def get_research_queries(sector):
  return [
    f"Tamaño mercado {sector} España 2024 facturación",
```

```
    f"Márgenes EBITDA sector {sector} España",
    f"Principales empresas {sector} España cuota mercado",
    f"Asociación nacional empresas {sector} España",
    f"Normativa y regulación {sector} España riesgos",
    f"Tendencias M&A {sector} Europa 2024"
]
```

Paso 2: Generación del Borrador

Python

```
def generate_initial_report(sector, web_context):
    prompt = f"SECTOR: {sector}\n\nDATOS WEB:\n{web_context}"

    response = model.generate_content(
        prompt,
        system_instruction=INITIAL_SYSTEM_PROMPT,
        generation_config={"response_mime_type": "application/json"}
    )
    return json.loads(response.text)
```

Paso 3: Función de Actualización (Chat)

Python

```
def handle_user_edit(current_json, section_key, user_instruction):
    # 1. Recuperar contenido actual
    old_text = current_json['sections'][section_key]['content']

    # 2. Re-generar solo esa sección
    prompt = f"""
    TAREA: Actualizar sección '{section_key}'.
    TEXTO ORIGINAL: {old_text}
    CAMBIO SOLICITADO: {user_instruction}

    Devuelve solo el nuevo Markdown.
    """

    new_text = model.generate_content(prompt).text

    # 3. Actualizar estado
    current_json['sections'][section_key]['content'] = new_text
    return current_json
```

---

11. ENTREGABLES Y ROADMAP

Fase 1: Backend Core (Día 1-2)

- Implementar `analyst.py` con la lógica de Gemini + Tavily.

- Validar que el JSON de salida respeta la estructura.

Fase 2: UI Skeleton (Día 3)

- Montar Streamlit con dos columnas.
- Columna Izquierda: Input simple.
- Columna Derecha: `st.markdown()` iterando sobre el JSON.

Fase 3: Interactividad (Día 4-5)

- Conectar el Chat de la izquierda a la función `handle_user_edit`.
- Implementar el refresco de estado (`st.session_state`) para que la derecha se actualice al instante.

Fase 4: Refinamiento (Día 6)

- Ajustar la temperatura del modelo (0.2 recomendado).
- Mejorar los *System Prompts* con ejemplos de "Deal Killers" reales.