

AI Innovation Agent 1.0

Índice

1. INTRODUCCIÓN	4
1.1 Visión Ejecutiva del Sistema	4
1.2 Capacidades Funcionales	7
1.3 Flujo de Trabajo End-to-End	10
1.4 Stack Tecnológico Overview	14
2. ARQUITECTURA GENERAL DEL SISTEMA	19
2.1 Arquitectura de Alto Nivel	19
2.3 Arquitectura de Conectividad	25
2.4 Arquitectura de Procesamiento	31
2.5 Arquitectura de UI	33
3. MÓDULOS CORE - ANÁLISIS DETALLADO	35
3.1 Módulo Principal: Interface (gr1.py)	35
3.2 Motor de Análisis (analysis_module2.py)	40
3.3 Sistema de Ranking (ranking_module.py)	44
3.4 Análisis Competitivo (competitor_analysis_module.py)	48
4. MÓDULOS DE SOPORTE	53
4.1 Procesamiento de Documentos – Módulos de Entrada	53
4.2 Generación de PDFs – Módulos de Salida	55
4.3 Configuración y Conectividad – Módulos de Infraestructura	60
5. ALGORITMOS CRÍTICOS – DEEP DIVE	63
5.1 Detección Multicapa de Ideas	64
5.2 Sincronización de Estado Distribuido	64
5.3 Sistema de Paralelización	64
5.4 Algoritmo de Procesamiento de Análisis	65
6. CONFIGURACIÓN Y DEPLOYMENT	67
6.1 Configuración del Entorno	67
6.2 Despliegue Automatizado con Azure DevOps	74
7. OPTIMIZACIONES Y RENDIMIENTO	79
7.1 Optimizaciones Implementadas	79
8. ANÁLISIS DE COSTES Y MODELOS	85
8.1 Marco General de Facturación Azure OpenAI	85
8.3 Análisis Comparativo de Modelos y Métricas de Rendimiento	87

8.4 Proyección Económica y Ahorros Esperados	88
8.5 Estrategias de Optimización Avanzada	88
8.6 Recomendaciones Estratégicas	89
8.7 Nota Técnica Crítica	89
9. PROPUESTA TÉCNICA DE MEJORAS	90
2. Módulo de Análisis de Ideas	90
3. Módulo de Ranking	91
4. Módulo de Carga de Documentos	91
5. Mejoras Generales del Sistema	91
6. Monitorización y Mantenimiento	92
7. Integración y Extensibilidad	92
10. ANNEXOS	93

1. INTRODUCCIÓN

1.1 Visión Ejecutiva del Sistema

1.1.1 ¿Qué es AI Innovation Agent?

AI Innovation Agent es una app diseñada específicamente para optimizar y escalar la fase de Ideación dentro de la Matriz de Innovación. Esta solución basada en Inteligencia Artificial Generativa automatiza el análisis, priorización y evaluación de ideas innovadoras, sustituyendo procesos manuales intensivos por un flujo digital estructurado, consistente y eficiente.

El sistema combina múltiples tecnologías de vanguardia en una plataforma única:

- Procesamiento de Lenguaje Natural con Azure OpenAI (GPT-4.0)
- Algoritmos de análisis y ranking multidimensional
- Módulo de análisis competitivo automatizado con capacidades de scraping web (Tavily)
- Generación automatizada de informes profesionales en PDF
- Interfaz gráfica intuitiva basada en Gradio para el control total del proceso

AI Innovation Agent actúa como un consultor estratégico digital capaz de:

- Procesar automáticamente documentos masivos (PDF) con ideas de negocio
- Evaluar cada idea en seis dimensiones estratégicas clave
- Priorizar ideas según criterios de negocio definidos por la organización
- Realizar análisis competitivo para ideas seleccionadas
- Generar informes ejecutivos listos para ser presentados ante comités de decisión

1.1.2 Problema que Resuelve para SENER

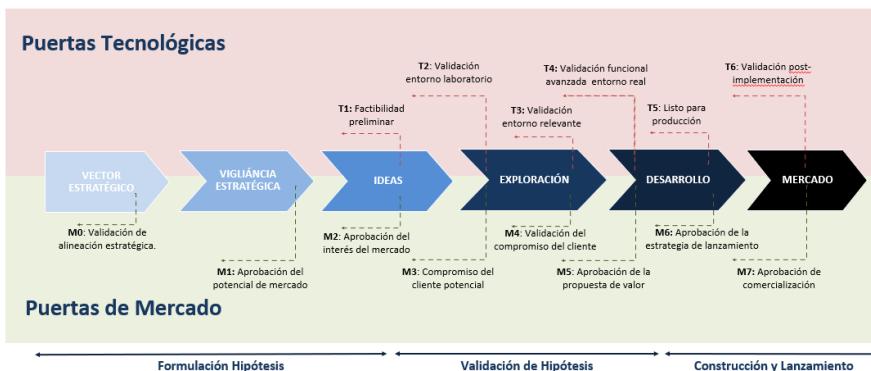
De la gestión manual a la inteligencia estructurada

Antes de la implementación del AI Innovation Agent, SENER enfrentaba retos importantes en la gestión de su proceso de innovación. La evaluación de ideas se realizaba de manera manual, con un alto grado de dependencia en consultores expertos y sin criterios estandarizados. Esta situación generaba ineficiencias significativas: tiempos prolongados de análisis, dificultad para escalar el procesamiento cuando el volumen de propuestas aumentaba y una gran variabilidad en los resultados dependiendo de quién realizaba la evaluación.

Para resolver esta fragmentación, desde Innovación se desarrolló una Matriz de Innovación que estructura todo el ciclo de vida de la innovación en seis etapas, desde la definición estratégica de vectores hasta el lanzamiento al mercado. Cada etapa incluye objetivos concretos y puntos de validación definidos, tanto tecnológicos como de mercado, lo que permite establecer un marco de gobernanza claro y accionable.

SENER		VECTOR INNOVACIÓN	VIGILANCIA ESTRATEGICA	IDEAS	EXPLORACIÓN	DESARROLLO	MERCADO
Objetivo		Definir áreas estratégicas de innovación como futuros negocios	Analizar el entorno competitivo, tecnológico, regulatorio, político y de mercados para identificar oportunidades y desafíos relevantes al vector	Generar y priorizar ideas a explorar.	Validar técnicamente y comercialmente las ideas seleccionadas.	Desarrollar, validar y lanzar soluciones innovadoras.	Lanzar las soluciones al mercado, evaluando su impacto y retroalimentado el Vector de Innovación.
Entregables		<ul style="list-style-type: none"> Modelo de Negocio del VECTOR Planificación del VECTOR 	<ul style="list-style-type: none"> Informe de vigilancia estratégica análisis de tendencias tecnológicas, competencia, marco regulatorio y legal y análisis de mercados Lista prioritizada de oportunidades estratégicas Definición de ENTRE I MERCADO 	<ul style="list-style-type: none"> Lista priorizada de ideas (con un análisis preliminar de valor técnico y de mercado) Plan de Exploración (QCT) 	<ul style="list-style-type: none"> Propuesta de valor ajustada Prototipo básico o prueba básica de concepto Comprobación de factibilidad técnica y de mercado Definición ajustada de ENTRE I MERCADO Plan de Desarrollo (QCT) Plan de Riesgos 	<ul style="list-style-type: none"> Planificación de Pruebas de Concepto (PoC) para comprobar la viabilidad técnica final hasta piloto industrial (TRL6) Solución piloto (PoC), servicio o proceso Modelo de Negocio Plan de desarrollo y ejecución del proyecto (técnico y comercial) Analisis de Riesgos 	<ul style="list-style-type: none"> Estrategia de comercialización y posicionamiento Plan de escalabilidad y distribución Informe de resultados iniciales del mercado (indicadores clave)
Herramientas Propuestas		<ul style="list-style-type: none"> Business Model Canvas VECTOR Estudio Mercado (TAM-SOM) Planificación VECTOR 	<ul style="list-style-type: none"> Benchmarking Payoff Matrix SWOT PESTEL Tecnologías Market Segmentation Customer Journey Empathy Map 	<ul style="list-style-type: none"> Descripción de la idea (descripción, dolor o oportunidad, propuesta de valor, mercado potencial, perspectiva de cliente, viabilidad estandarizada) Payoff matrix (con criterios priorización) 	<ul style="list-style-type: none"> T1: Factibilidad técnica preliminar 	<ul style="list-style-type: none"> Analisis Viabilidad Técnica + Económica FMEA Estudio de Patentes Risk assessment, postura SWOT 	<ul style="list-style-type: none"> • PoC: Carácter Pruebas de Simulación y Testing. Risk assessment, matriz • SWOT
Puertas Tecnológicas		No aplica	No aplica		T2: Facilidad técnica preliminar	T3: confirmación de viabilidad para pasar de TRL inicio a TRL final	T4: confirmación de consecución de TRL comercial
Puertas Mercado		<ul style="list-style-type: none"> M0: Validación de alineación estratégica y empresas del TAM-SOM 	<ul style="list-style-type: none"> M0.1 Definición de CLIENTE I MERCADO y sus dolores i oportunidades 	M1: Facilidad de Clientel-Mercado, preliminar	M2: confirmación de viabilidad para pasar de Cliente inicio a Cliente final	M3: confirmación de Cliente Final	M4: Comercialización
Deciders	Comité + Dirección General (D.G)	Mensual Comité Cuatrimestral D.G	Mensual Comité Cuatrimestral D.G	Mensual Comité Cuatrimestral D.G	Mensual Comité Cuatrimestral D.G	Mensual Comité Cuatrimestral D.G	Mensual Comité Cuatrimestral D.G

Dentro de esta matriz, la etapa de Ideación se identificó rápidamente como el principal cuello de botella del proceso. Su complejidad radica en el volumen elevado de propuestas (que puede superar las 100 por sesión) y en la necesidad de realizar un análisis exhaustivo por múltiples dimensiones: factibilidad técnica, viabilidad comercial, alineación estratégica, potencial de innovación, entre otras. Este análisis, realizado de forma manual, requería entre dos y cuatro horas por idea y presentaba serios desafíos de consistencia y escalabilidad.



El AI Innovation Agent nace para resolver esta fricción crítica, automatizando la evaluación inicial de ideas y liberando capacidad operativa clave. El sistema transforma un proceso costoso y heterogéneo en un flujo estructurado, trazable y escalable, alineado con la estrategia de innovación de SENER.

Vectores Tecnológicos Estratégicos de SENER

La solución está especialmente diseñada para trabajar dentro del marco de los **Vectores de Innovación de SENER Mobility**, que agrupan sectores tecnológicos con alto potencial para la compañía. Estos vectores, que incluyen ámbitos como:

- Vehículo Autónomo,
- Salud,
- Infraestructura Móvil,
- Agua,
- Operación y Mantenimiento,
- Infraestructuras Cognitivas,
- Puertos,

- Hiper-aprovechamiento de Activos
- Entretenimiento

constituyen los dominios prioritarios donde la plataforma aporta mayor valor. Gracias a su diseño modular, el agente puede adaptarse con precisión al análisis de cada uno de estos sectores, considerando sus particularidades estratégicas y tecnológicas.

1.1.3 Valor de Negocio y ROI

Impacto Cuantificable y Sostenido

El AI Innovation Agent aporta un retorno inmediato y medible para SENER, tanto en términos operativos como estratégicos. A través de la automatización de los análisis de ideas, el sistema transforma un proceso intensivo en tiempo y recursos en una operación ágil, escalable y económicamente eficiente.

Eficiencia Operativa y Ahorro de Costes

La automatización del análisis permite reducir el tiempo necesario por idea de un rango promedio de 2 a 4 horas manuales a tan solo 5 a 10 minutos. Esta mejora representa una reducción del 95 al 98 % en el tiempo invertido y permite escalar la capacidad de análisis de 5-10 ideas diarias a más de 100.

En términos económicos, el coste promedio por análisis se reduce de los 100-240 € por idea (consultoría senior) a tan solo 1-3 € (considerando uso de APIs y costes de infraestructura). Al final del documento se detallará la estructura de costes. Este impacto se traduce en un ROI operativo directo superior al 900 % en escenarios de uso organizacionales amplios.

Valor Estratégico y de Escalabilidad

Más allá del ahorro directo, el sistema introduce mejoras cualitativas clave:

- Estandarización completa de criterios de evaluación, eliminando sesgos y subjetividades
- Mayor trazabilidad en la toma de decisiones, con respaldo documental automatizado
- Capacidad de escalar procesos a múltiples vectores y divisiones simultáneamente
- Soporte directo a los comités de innovación con informes listos para su consumo

Escenarios de ROI estimado

Escenario	Ideas/Año	Ahorro Estimado (€)	ROI Aproximado
Vector Individual	200	15,000 – 25,000	300 – 500 %
Vectores Múltiples	500	35,000 – 60,000	600 – 800 %
Escala Organizacional Completa	1,000+	75,000 – 120,000	900 – 1,200 %

Estas cifras están basadas en ahorros por horas de consultoría evitadas, mejora en la velocidad de respuesta y mayor efectividad en la selección de ideas con alto potencial.

1.1.4 Usuarios Objetivo

Usuarios Principales

AI Innovation Agent está diseñado para dar servicio tanto a la Dirección de Innovación como a los equipos funcionales que operan y dinamizan los distintos vectores estratégicos de innovación. Su diseño modular y su interfaz intuitiva permiten que diferentes perfiles puedan interactuar con la herramienta según su rol en el proceso.

Dirección de Innovación y Estrategia

Área responsable de coordinar la estrategia de innovación a nivel corporativo. Utiliza el sistema como plataforma de control y seguimiento del portafolio de ideas, generando informes ejecutivos automatizados, rankings estratégicos y soportes analíticos para la toma de decisiones en los comités de innovación.

Equipos Operativos de Innovación (Soporte a Vectores)

Profesionales del Departamento de Innovación encargados de facilitar y dar soporte metodológico y operativo a los líderes de cada vector. Estos usuarios utilizan el sistema para cargar documentos, monitorizar el estado de evaluación de las ideas, preparar sesiones de análisis y coordinar la comunicación entre stakeholders técnicos, estratégicos y de negocio.

Líderes y Responsables Técnicos de Vectores

Referentes de innovación en áreas específicas (agua, salud, infraestructuras, etc.) que pueden solicitar acceso a la plataforma para evaluar propuestas de su dominio. Usan el sistema para revisar el análisis técnico-comercial de ideas, validar su alineación con los objetivos del vector y apoyar la priorización de iniciativas con alto potencial.

1.2 Capacidades Funcionales

Su flujo operativo cubre desde la carga de documentos hasta la entrega de informes ejecutivos listos para comités, permitiendo escalar el sistema a cientos de ideas con trazabilidad, consistencia y velocidad.

1.2.1 Procesamiento Automático de Documentos (PDF)

Ingesta Inteligente desde Documentación No Estructurada

La plataforma permite cargar documentos PDF con ideas provenientes de sesiones de ideación, informes técnicos o reportes de vigilancia estratégica, y extraer automáticamente el contenido relevante para análisis posterior. Gracias a su sistema de detección avanzada de ideas, el agente es capaz de identificar y estructurar propuestas desde textos con diferentes formatos, estilos y niveles de complejidad, sin necesidad de plantillas predefinidas.

Esta capacidad ha demostrado un alto rendimiento en contextos reales, procesando actas con más de 50 ideas en menos de un minuto, con tasas de extracción superiores al 90 %. Además, cada idea es enriquecida automáticamente con contexto estratégico para su evaluación posterior.

Ventajas Clave:

- Procesamiento masivo de ideas sin intervención manual
- Eliminación de errores de transcripción
- Alta flexibilidad en formatos de entrada
- Enriquecimiento automático con IA contextual
- Tiempo promedio por documento: entre 30 y 90 segundos

1.2.2 Análisis Multidimensional con IA

Framework de Evaluación Estratégica en 6+1 Dimensiones

Una vez cargadas las ideas, el sistema ejecuta un análisis estructurado en torno a seis dimensiones clave, definidas según mejores prácticas de consultoría estratégica. Este marco proporciona una visión integral del valor, viabilidad y alineación de cada propuesta dentro del contexto organizacional de SENER.

Las dimensiones incluyen:

1. **Resumen Ejecutivo**
Evaluación de la propuesta de valor, aplicabilidad y contexto de uso dentro de SENER.
2. **Análisis Técnico**
Valoración de la factibilidad, madurez tecnológica y compatibilidad con las capacidades actuales.
3. **Potencial de Innovación**
Grado de novedad, diferenciación competitiva, escalabilidad y disruptión esperada.
4. **Alineación Estratégica**
Coherencia con los vectores de innovación de SENER, sinergias y encaje en el portfolio existente.
5. **Viabilidad Comercial**
Evaluación del modelo de negocio, mercado objetivo, competencia y proyecciones preliminares.
6. **Valoración Global**
Scoring integral, recomendación estratégica, próximos pasos y estimación de tiempos.

Dimensión Adicional: Generación de Retos y Soluciones

Como etapa posterior al análisis principal, el sistema puede generar automáticamente un informe específico de retos asociados a la implementación de cada idea, junto con propuestas de solución técnica y roadmap recomendado. Esta capacidad transforma el diagnóstico en un plan de acción, con identificación de riesgos, recursos necesarios y posibles caminos de mitigación.

Aplicación práctica:

- Evaluación completa de ideas en contexto real
- Generación de informes estructurados con lenguaje ejecutivo
- Priorización automática basada en criterios ponderados
- Activación de seguimiento operativo post-análisis

1.2.3 Sistema de Ranking y Priorización Inteligente

Priorización Estratégica Basada en Scoring Multidimensional

El AI Innovation Agent incorpora un sistema de ranking que permite priorizar ideas en función de criterios estratégicos definidos por el departamento de Innovación. Este sistema combina análisis cualitativos y cuantitativos para ofrecer una evaluación fundamentada y trazable, facilitando la toma de decisiones sobre qué ideas avanzar en el pipeline de innovación.

El ranking se construye a partir de un modelo de puntuación distribuido en tres ejes principales:

- **Potencial de Implementación Inmediata:** Evalúa la viabilidad técnica con recursos actuales, tiempos estimados de desarrollo y readiness regulatorio.
- **Alineación con Partners Estratégicos:** Mide las sinergias con socios existentes, oportunidades de co-desarrollo y acceso a mercado.
- **Impacto a Largo Plazo:** Considera el tamaño del mercado objetivo, escalabilidad, diferenciación competitiva y alineación con objetivos organizacionales.

Cada idea recibe una puntuación desglosada por criterio, acompañada de justificaciones detalladas y recomendaciones personalizadas, incluyendo posibles partners estratégicos y próximos pasos sugeridos.

Además, el sistema genera una **matriz de payoff visual** —dificultad de implementación vs. impacto esperado— que clasifica las ideas en categorías como *quick wins*, *proyectos estratégicos*, o *tareas sin retorno*. Esta visualización, incluida en los informes, ofrece una visión clara de la distribución del portafolio.

1.2.4 Análisis Competitivo Automatizado

Exploración Estratégica del Entorno Externo

El módulo de análisis competitivo representa una de las capacidades más avanzadas del sistema. Combina técnicas de inteligencia artificial con web scraping para obtener información actualizada y relevante sobre el entorno competitivo de cada idea, sin intervención manual.

El análisis se estructura en ocho secciones:

1. **Competitor Mapping:** Identificación y clasificación de competidores, incluyendo actores emergentes.
2. **Benchmark Matrix:** Comparación cuantitativa de métricas clave: ingresos, tamaño, proyectos, cuota de mercado.
3. **Tech & IP Landscape:** Exploración de patentes, publicaciones científicas y gaps tecnológicos.
4. **Market Analysis:** Estimación del mercado objetivo, crecimiento proyectado y análisis regional.
5. **SWOT Positioning:** Análisis comparativo entre SENER y sus competidores en términos de fortalezas, oportunidades y riesgos.
6. **Regulatory & ESG Risk:** Evaluación de marco regulatorio y sostenibilidad aplicable.

El sistema utiliza búsquedas inteligentes por vector tecnológico, generación automática de queries sectoriales, análisis semántico y validación cruzada de información para garantizar la calidad del resultado. Todo esto se realiza de forma automatizada y se integra directamente en los informes.

1.2.5 Generación de Informes Profesionales

Reporting Ejecutivo Listo para Decisión

AI Innovation Agent genera automáticamente informes profesionales adaptados a distintos momentos del proceso de decisión, con un enfoque en claridad, estructura y presentación visual de calidad. La generación es completamente automática tras el análisis, y cada informe se entrega en formato PDF optimizado para distribución interna.

- **Tipologías de Informes:**

- **Informe de Análisis Principal**
Contiene el detalle completo de cada idea evaluada en las 6+1 dimensiones. Incluye anexos técnicos, contexto estratégico y elementos visuales de apoyo.
- **Informe de Ranking**
Presenta la priorización global del conjunto de ideas, con desgloses por criterio, matriz de payoff y recomendaciones de implementación.
- **Informe de Retos y Soluciones**
Desarrolla, por cada idea, los retos técnicos, regulatorios o de mercado detectados, así como propuestas de solución y roadmap de implementación.
- **Informe de Análisis Competitivo**
Documento exhaustivo con las ocho secciones de benchmarking competitivo, visualizaciones estratégicas y comparativas cuantitativas.
- **Diseño y Formato**

Todos los informes integran:

- Portada corporativa con logo y branding SENER
- Índice automático con hipervínculos internos
- Headers y footers unificados
- Gráficos y tablas de alta resolución
- Formato de lectura ejecutiva, con jerarquía visual clara

La tecnología de generación de informes permite producir documentos de gran volumen (30-60 páginas) en cuestión de minutos, con versiones adaptadas según la audiencia objetivo (comités estratégicos, responsables técnicos o gestores operativos).

1.3 Flujo de Trabajo End-to-End

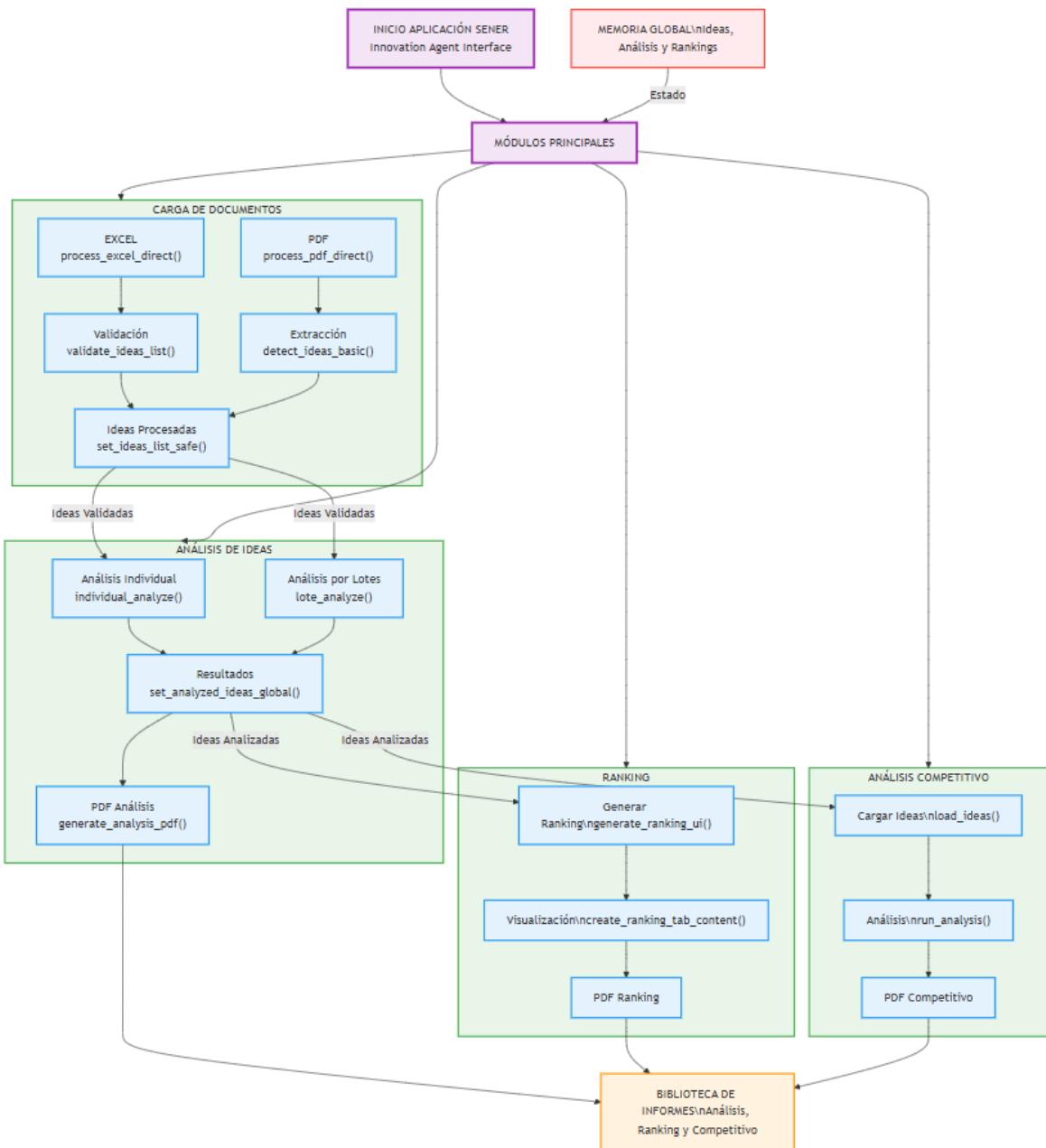
1.3.1 Diagrama de Flujo General del Usuario

El **AI Innovation Agent** está diseñado como una plataforma de navegación secuencial, simple e intuitiva, que guía al usuario desde la carga inicial de documentos hasta la entrega final de informes ejecutivos. Todo el flujo está optimizado para minimizar la carga operativa manual, permitiendo al mismo tiempo un control estratégico total por parte del usuario.

El flujo general está estructurado en **cuatro etapas principales**:

1. **Carga de Documentos**
Ingesta de PDFs estructurados junto con el contexto estratégico del vector correspondiente.
2. **Procesamiento con IA**
Ejecución automática del análisis multidimensional y generación de retos y soluciones asociadas.
3. **Priorización Estratégica**
Generación del ranking de ideas, visualización de matriz de payoff e identificación de acciones sugeridas.
4. **Análisis Competitivo y Generación de Informes**
Selección de ideas clave para benchmarking competitivo y producción de informes listos para comités.

El flujo de navegación permite avanzar secuencialmente o acceder directamente a pestañas específicas, lo que otorga flexibilidad total al usuario según sus necesidades operativas o estratégicas.



1.3.2 Puntos de Entrada y Salida

Puntos de Entrada Críticos

El sistema requiere **dos entradas clave** para garantizar un análisis contextualizado, estructurado y alineado con las prioridades de SENER:

1. Documento de Ideas en Formato PDF (estructura obligatoria)

El archivo debe contener ideas redactadas en párrafos continuos, claramente delimitadas bajo la forma:

IDEA 1: Nuevos usos estaciones
La idea busca dar uso a las estaciones aparte de su función de transporte. Se propone incorporar elementos de interés que motiven a las personas a visitarlas sin necesidad de usar el metro. Esto fomenta usos complementarios de la infraestructura, ampliando su utilidad. Las estaciones se convertirían en destinos de interés dentro de la ciudad.

La propuesta consiste en crear en las estaciones actividades y servicios que no solo sirven a la función de transporte de los ciudadanos, como centros de salud, polideportivos, tiendas u otros servicios necesarios en el día a día de las personas.

Dentro de las actuaciones a ampliar el uso de la estación se mejoraría la accesibilidad y relación de la estación con la ciudad, por ejemplo, conectando directamente las entradas y salidas del metro con aceras, parques, centros comerciales, o edificios públicos mediante pasarelas cubiertas que hagan más amigable el uso de la estación. Este enfoque elimina barreras físicas y facilita una transición fluida entre la calle y el metro. Este enfoque maximiza el valor de los activos del metro al diversificar su propósito, y promueve un diseño urbano más eficiente y accesible para los usuarios.

IDEA 2: Lat Mile
La propuesta consiste en utilizar el metro como medio de transporte para mercancías, optimizando sus infraestructuras existentes. Los paquetes se trasladan desde un almacén central en camión hasta un depósito del metro. Dentro del metro, las mercancías se trasladan y se depositan en taquillas inteligentes. Estas taquillas permiten a los proveedores optimizar su logística para ser más eficiente.

Dentro de esta misma solución, podría considerarse el uso de depósitos logísticos. Se trata de que los depósitos existentes de metro se conviertan en instalaciones con una función adicional para el almacenamiento, clasificación y distribución de mercancías. Actuarían como puntos intermedios dentro de la cadena de suministro para mejorar la eficiencia y la velocidad de entrega. Esto permite una mejor gestión de inventarios y el envío eficiente de productos. Se alinea con el objetivo de proporcionar usos múltiples y complementarios a las infraestructuras del metro. Así, se maximiza el aprovechamiento de los activos existentes en beneficio de la logística urbana.

Las soluciones de distribución se basan en una infraestructura específica como cintas transportadoras o tubos de vacío para mover toneladas de material o aprovechando convoyes de material móvil para intercalar con el, los servicios de pasajeros ya sea en horas punita o hora.

IDEA 3: Refugios subterráneos multifuncionales
Adaptar las estaciones existentes para resistir desastres naturales y emergencias como terremotos, inundaciones o ataques. Uso de estaciones inactivas o menor transitadas sin necesidad de grandes obras teniendo que adaptar ventilación, suministro eléctrico y de agua en ubicaciones más densamente pobladas de rápido acceso. Los espacios serán convertibles sirviendo como refugios en emergencias y como áreas comunitarias en tiempos normales. Deberán tener capacidad de incluir almacenamiento de suministros esenciales, áreas médicas y comunicaciones. La solución debe combinar la infraestructura existente con soluciones innovadoras.

IDEA 4: Depósito Vertical Subterráneo Urbano
El Depósito Vertical Subterráneo Urbano es una instalación logística bajo tierra diseñada en niveles verticales para optimizar el almacenamiento y manejo de material móvil en ciudades densamente pobladas. Su diseño permite ahorrar espacio en la superficie y minimizar el impacto visual en el entorno urbano. Esta solución mejora la distribución de mercancías en áreas con limitaciones de espacio o tráfico. Además, puede integrarse con infraestructuras de metro para usos complementarios y multifuncionales. Esto favorece una gestión más eficiente de los activos subterráneos en entornos urbanos complejos.

Esto asegura una extracción precisa y fiable de contenido para el análisis automático posterior.

2. Contexto Estratégico del Vector

Breve texto explicativo que posiciona las ideas dentro del marco del vector correspondiente (Infraestructuras, Salud, Agua, etc.). Este contexto es fundamental para que el análisis se oriente correctamente en términos técnicos, regulatorios y de impacto esperado.

Puntos de Salida Estructurados

Una vez completado el análisis, el sistema genera **cuatro entregables principales**, cada uno adaptado a una fase específica del proceso de decisión:

- **PDF de Análisis Principal:** Contiene la evaluación detallada en 6+1 dimensiones para cada idea.
- **PDF de Ranking:** Presenta el orden de prioridad estratégico con matriz de payoff y recomendaciones.
- **PDF de Retos y Soluciones:** Resume los desafíos clave detectados y propone caminos de mitigación.
- **PDF de Análisis Competitivo:** Benchmarking estructurado frente a competidores relevantes.

Además, se admiten **contextos complementarios** opcionales para afinar aún más el análisis:

- **Contexto de Análisis Técnico:** Orienta la evaluación hacia retos o aspectos específicos.
- **Contexto de Ranking:** Permite priorizar según objetivos particulares (por ejemplo, "implementación rápida con impacto internacional").
- **Contexto Competitivo:** Delimita el alcance del análisis de mercado (por región, tipo de competidor, tecnología, etc.).

1.3.3 Decisiones Automatizadas vs Manuales

Combina automatización avanzada con control estratégico humano, permitiendo una gestión eficiente, trazable y alineada con las prioridades de negocio de SENER. El sistema automatiza la ejecución técnica de los procesos clave, mientras que las decisiones estratégicas permanecen bajo la responsabilidad del usuario.

Decisiones Automatizadas

La plataforma opera de forma completamente autónoma en la ejecución de tareas técnicas y operativas, garantizando consistencia, escalabilidad y calidad de salida. Entre los procesos totalmente automatizados se encuentran:

- **Procesamiento de documentos:** extracción estructurada de ideas, limpieza de texto, normalización de formato y deduplicación de contenido.
- **Análisis multidimensional:** aplicación del modelo de evaluación de seis dimensiones más retos y soluciones, adaptado al contexto estratégico del vector correspondiente.
- **Generación de ranking:** cálculo automático de puntuaciones, categorización mediante matriz de esfuerzo vs. beneficio y formulación de recomendaciones iniciales.
- **Análisis competitivo:** identificación y clasificación de competidores, benchmarking sectorial y generación estructurada de insights estratégicos.
- **Creación de informes profesionales:** ensamblaje automatizado de documentos PDF con diseño corporativo, estructura ejecutiva e inclusión de elementos visuales y técnicos.

Estas decisiones dan lugar a los siguientes entregables:

- **Informe de Análisis Principal:** evaluación detallada de cada idea en 6+1 dimensiones, con retos identificados y anexos metodológicos (30–50 páginas).
- **Informe de Ranking:** priorización fundamentada, matriz visual de payoff, recomendaciones y timeline sugerido (15–25 páginas).
- **Informe de Análisis Competitivo:** evaluación externa comparativa en ocho bloques estratégicos, con gráficos, mapas de posicionamiento y hallazgos clave (40–60 páginas).

Decisiones Manuales

El usuario conserva el control sobre todas las decisiones estratégicas que requieren juicio experto o alineamiento con objetivos de negocio:

- Selección de documentos a procesar y contexto sectorial
- Definición de criterios de priorización y objetivos estratégicos
- Validación de ideas extraídas y ajustes de contenido
- Selección de ideas top para análisis competitivo
- Interpretación de resultados, toma de decisiones y planificación de siguientes pasos
- Mejorar y modificar el prompting interno del Código

1.3.4 Tiempos de Procesamiento Estimados

El sistema ha sido diseñado para ofrecer tiempos de respuesta óptimos en entornos reales, incluso cuando se manejan volúmenes elevados de ideas. A continuación, se detalla el rendimiento estimado por etapa operativa:

Tiempo Promedio por Etapa

Etapa	Volumen Estimado	Tiempo Total Aproximado
Carga y validación inicial	20–50 ideas (PDF)	1 a 2 minutos
Análisis multidimensional	20–50 ideas	2 a 5 minutos
Ranking estratégico	20-50 ideas	3 a 6 minutos
Análisis competitivo	Hasta 6 ideas seleccionadas	3 a 6 minutos
Retos y soluciones (opcional)	20-50 ideas	1 a 3 minutos

Variables que Impactan el Tiempo de Procesamiento

- **Volumen de ideas procesadas:** el rendimiento se ajusta mediante procesamiento por lotes y paralelización optimizada.
- **Carga del sistema y tráfico concurrente:** en condiciones de uso intensivo, los tiempos pueden verse afectados en un 20–40 %.
- **Dependencias externas:** la disponibilidad de servicios como Azure OpenAI o motores de búsqueda web tiene un impacto directo en la duración de ciertas tareas.

1.4 Stack Tecnológico Overview

El AI Innovation Agent ha sido desarrollado sobre un ecosistema tecnológico de última generación, diseñado para ser robusto, escalable y alineado con los estándares corporativos de SENER. El stack combina capacidades avanzadas de inteligencia artificial, procesamiento automatizado, interfaces interactivas y despliegue cloud seguro, integrando todos los componentes necesarios para garantizar operatividad, gobernanza y evolución a largo plazo.

1.4.1 Tecnologías Principales

Motor de Inteligencia Artificial — Azure OpenAI GPT-4o

El núcleo del sistema está basado en el modelo GPT-4o, desplegado mediante Azure OpenAI Services, bajo una arquitectura optimizada para entornos empresariales. Esta configuración permite análisis avanzados, consistencia de resultados y cumplimiento de políticas de seguridad y compliance.

- **Modelo activo:** GPT-4o (versión 2024-11-20)
- **Endpoint:** Azure AI Services (entorno privado SENER)
- **Límites de uso:** hasta 100.000 tokens por minuto y 600 solicitudes por minuto
- **Autenticación:** basada en claves seguras (Key-Based Access)
- **Estado del servicio:** disponible en producción (Generally Available)
- **Retiro programado del modelo actual:** marzo 2026

La arquitectura contempla además un sistema de monitoreo y versionado, que permite adaptar de forma progresiva el despliegue a nuevas versiones del modelo o cambios en la estructura de las APIs.

Interfaz de Usuario — Gradio 4.x

La experiencia de usuario se construye sobre el framework **Gradio**, en su versión más reciente, implementado con una estructura modular que facilita la navegación por pestañas, la carga de documentos, la visualización de resultados y la interacción en tiempo real.

Entre sus capacidades destacan:

- Interfaz responsive y adaptativa
- Navegación estructurada por etapas del proceso
- Gestión global del estado de sesión
- Personalización estética mediante CSS corporativo
- Soporte para componentes avanzados (DataFrames, uploads, visualizaciones)

Lógica de Aplicación — Python 3.8+

El núcleo funcional está desarrollado íntegramente en **Python 3.8+**, lenguaje ampliamente utilizado en entornos de ciencia de datos e inteligencia artificial. Para ello se usa la herramienta VSC. Se han integrado librerías de primer nivel para procesamiento de lenguaje natural, visualización, generación documental, scraping web y concurrencia:

- **IA y análisis:** openai, spacy, scikit-learn
- **Visualización y reporting:** pandas, matplotlib, fpdf2, reportlab
- **Web scraping:** requests, beautifulsoup4, tavyly-python
- **Concurrencia:** asyncio, threading, concurrent.futures

1.4.2 Arquitectura de Deployment (Docker)

Entorno Contenerizado y Controlado

Todo el sistema está empaquetado en un **contenedor Docker empresarial**, optimizado para ejecución en entornos cloud y locales, con foco en rendimiento, trazabilidad y seguridad. Este enfoque garantiza la replicabilidad de entornos, facilidad de actualización y mantenimiento centralizado.

- Imagen base optimizada (python:3.11-slim)
- Configuración de seguridad y logging
- Instalación automatizada de dependencias
- Exposición del puerto 7860 para interfaz web
- Comando de ejecución principal (gr1.py)

Orquestación con Docker Compose

El despliegue en entornos de desarrollo se gestiona mediante docker-compose, lo que facilita la integración de volúmenes persistentes, configuración de variables sensibles y gestión de estado del contenedor.

Incluye además:

- Reinicio automático ante fallos
- Healthcheck interno vía curl
- Mapeo de directorios (output, temp_uploads)
- Integración directa con credenciales de Azure

Plataforma Cloud SENER (Azure)

El sistema está desplegado en una plataforma cloud privada gestionada por AI Lab de SENER, accesible exclusivamente mediante credenciales autorizadas:

<https://agenteinnovacion.cloud.sener>

Este entorno está vinculado a un **repositorio Git privado**, desde el cual se gestionan los ciclos de desarrollo, pruebas y despliegue. Toda modificación de la aplicación requiere realizar un pull o push desde dicho repositorio, y la actualización se refleja automáticamente en el contenedor a través del sistema de orquestación Docker.

1.4.3 Integración con Servicios Externos

El AI Innovation Agent incorpora un conjunto de servicios externos estratégicos que amplían sus capacidades técnicas y analíticas. La integración se ha diseñado bajo un modelo modular, seguro y gobernado centralmente por el equipo AI Lab de SENER.

Servicios Azure

La plataforma opera sobre Microsoft Azure, integrando servicios empresariales clave:

- **Azure OpenAI GPT-4o:** motor principal de análisis generativo.
- **Azure Cognitive Services:**
 - *Content Safety:* filtrado automático de contenido inapropiado.
 - *Language Detection:* detección de idioma en documentos multilingües.
 - *Form Recognizer:* extracción avanzada de texto en PDFs técnicos.

Configuración activa:

- Región: East US / West Europe
- Suscripción: SENER AI Lab
- Grupo de recursos: ailab-dev-resources
- Plan de uso: Standard (Pay-as-you-go)
- Gobernanza: gestionado internamente por AI Lab

Búsqueda Inteligente y Web Scraping

El sistema incorpora funciones de búsqueda avanzada para el módulo de análisis competitivo:

- **Tavily Search API:** búsqueda semántica estructurada por sector y dominio.
- **Scraping híbrido:**
 - Requests y BeautifulSoup: extracción estándar.
 - Selenium WebDriver: procesamiento de sitios dinámicos.
 - Control de *rate limiting*, cumplimiento de robots.txt.

Almacenamiento y Persistencia

La arquitectura de ficheros internos gestiona los flujos de datos y resultados de forma estructurada:

```
/app/  
|— output/ → informes generados
```

```
|--- temp_uploads/ → archivos cargados temporalmente  
|--- static/ → recursos visuales (logos, CSS)  
└ InnAgg/ → módulos funcionales del sistema
```

El almacenamiento en memoria y en archivos temporales permite mantener el estado de las sesiones activas, conservar resultados intermedios y garantizar limpieza automática tras cada ejecución.

1.4.4 Requisitos de Infraestructura

El sistema puede desplegarse en entornos cloud, on-premise o híbridos. La configuración recomendada asegura rendimiento, estabilidad y capacidad de escalar en función de la carga operativa.

Configuración Recomendada

Recurso	Mínimo Producción	Óptimo Alta Demanda
CPU	4 cores	8+ cores
RAM	8 GB	32 GB
Almacenamiento	50 GB SSD	200 GB NVMe SSD
Red	100 Mbps	1 Gbps

Opciones de Despliegue en Cloud

- Azure: VM tipo Standard_D4s_v3 o superior
- AWS: EC2 tipo m5.xlarge o superior
- Google Cloud: n2-standard-4 o superior

Dependencias del Sistema

Linux (Ubuntu/Debian):

```
apt install -y python3.11 pip curl git docker.io docker-compose  
apt install -y poppler-utils tesseract-ocr chromium-browser
```

Windows Server (alternativa):

- Python 3.11+, Git for Windows
- Docker Desktop, Visual C++ Redistributables

Requisitos de Red

Para funcionamiento completo, deben habilitarse los siguientes endpoints:

- **APIs:**
 - *.openai.azure.com (GPT-4o)
 - api.tavily.com (búsqueda competitiva)
 - *.googleapis.com (servicios auxiliares)
- **Repositorios:**
 - pypi.org (paquetes Python)
 - github.com (repositorio del proyecto)
 - docker.io (contenedor base)

- **Scraping targets:**
 - dominios .edu, .org, y webs corporativas
- **Puertos requeridos:**
 - Salida: 443 (HTTPS), 80 (HTTP)
 - Entrada: 7860 (interfaz Gradio)
 - Docker remoto: 2375/2376 (si aplica)

Seguridad y Compliance

El entorno está alineado con las políticas de seguridad de SENER:

- Accesos controlados por Al Lab
- Autenticación API con rotación de claves cada 90 días
- Cifrado TLS 1.3 en todas las comunicaciones
- Sanitización de logs y no persistencia de datos sensibles
- Auditoría activa de accesos, uso y operaciones
- Limpieza automática de archivos temporales

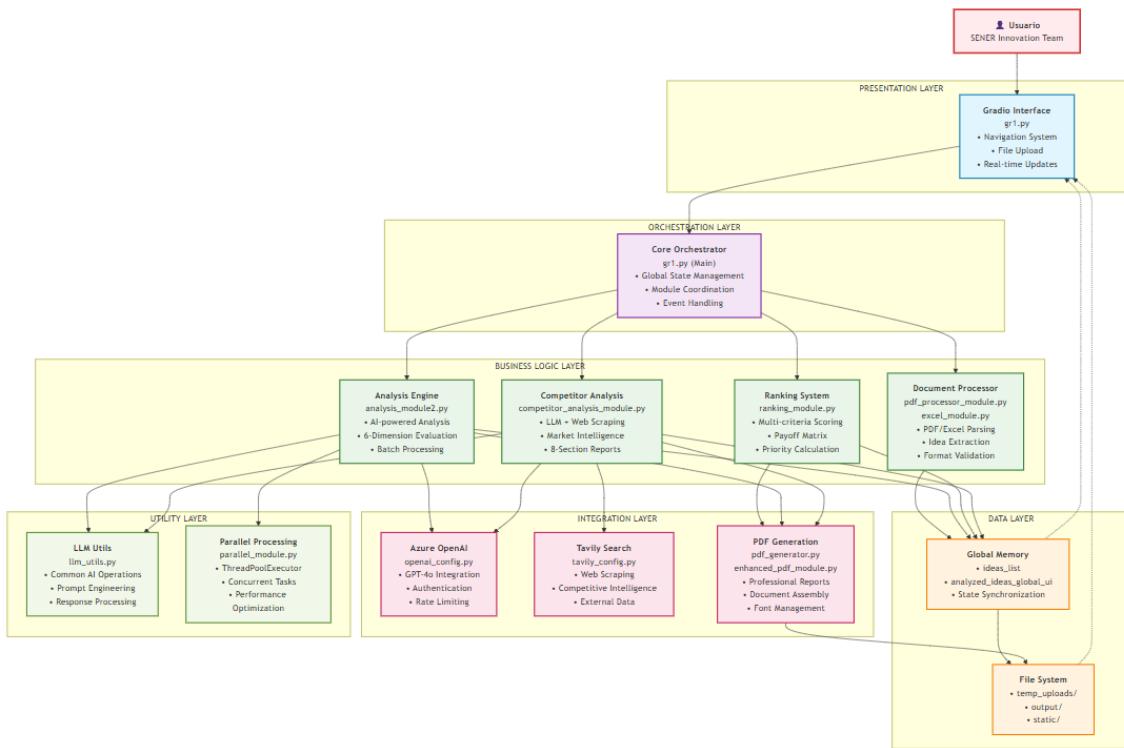
2. ARQUITECTURA GENERAL DEL SISTEMA

2.1 Arquitectura de Alto Nivel

2.1.1 Diagrama de Componentes Principales

El AI Innovation Agent ha sido diseñado siguiendo una arquitectura modular en capas, con separación clara de responsabilidades y comunicación bidireccional entre componentes. Esta estructura permite un procesamiento eficiente, mantenimiento escalable y trazabilidad completa de cada operación realizada por el sistema.

El [siguiente diagrama](#) presenta una visión de alto nivel de los principales módulos que componen la plataforma:



Descripción de Capas Funcionales

Capa de Presentación (Presentation Layer)

Es el punto de entrada para los usuarios del sistema. Está implementada con Gradio y permite la carga de documentos, navegación entre módulos funcionales y visualización de resultados en tiempo real.

Capa de Orquestación (Orchestration Layer)

Gestiona la coordinación general del sistema. El núcleo (gr1.py) se encarga del control de estado global, la sincronización de eventos y la comunicación entre componentes funcionales. Esta capa actúa como controlador central de la lógica de flujo.

Capa de Lógica de Negocio (Business Logic Layer)

Contiene los módulos principales de procesamiento analítico:

- El componente de análisis (analysis_module2.py) ejecuta el análisis automático basado en seis dimensiones estratégicas, con capacidad para generar retos y soluciones.
- El sistema de ranking (ranking_module.py) aplica criterios de priorización estratégica y genera matrices de evaluación.
- El módulo de análisis competitivo (competitor_analysis_module.py) combina modelos de lenguaje y scraping web para generar estudios de mercado estructurados.
- El procesador de documentos (pdf_processor_module.py, excel_module.py) extrae y normaliza ideas a partir de documentos en formato PDF o Excel.

Capa de Datos (Data Layer)

Incluye la memoria global del sistema, donde se gestionan listas de ideas, resultados parciales y sincronización de estado entre etapas. El almacenamiento físico se estructura en directorios dedicados a archivos temporales, resultados de salida y recursos estáticos.

Capa de Integración (Integration Layer)

Interconecta el sistema con servicios externos, tanto de inteligencia artificial como de búsqueda de información:

- Integración con Azure OpenAI para el uso del modelo GPT-4o.
- Acceso a Tavily Search API para el análisis competitivo.
- Módulos de generación de documentos (pdf_generator.py, enhanced_pdf_module.py) que ensamblan informes de salida con formato profesional.

Capa de Utilidades (Utility Layer)

Contiene funcionalidades de soporte para procesamiento concurrente y operaciones comunes de interacción con modelos de lenguaje. Incluye utilidades para prompt engineering, procesamiento de respuestas y ejecución paralela de tareas.

Flujo General de Procesamiento

El sistema está diseñado para que el usuario inicie el proceso mediante la interfaz, cargando un documento estructurado junto con su contexto estratégico. A partir de ese momento, la orquestación central coordina el procesamiento automático en cada uno de los módulos analíticos, desde la extracción y análisis de ideas hasta la priorización y el análisis de mercado. El resultado final se consolida en informes ejecutivos generados automáticamente y almacenados para descarga inmediata.

2.1.2 Separación de Responsabilidades

La arquitectura del AI Innovation Agent implementa una separación estricta de responsabilidades basada en los principios de *Single Responsibility* y *Separation of Concerns*, distribuyendo las funcionalidades en seis capas independientes pero integradas. Esta organización permite escalar el sistema de forma controlada, asegurar mantenibilidad y facilitar el testeo de cada módulo de forma aislada.

Presentation Layer – Capa de Presentación

Encargada de la interfaz de usuario y de la experiencia interactiva.

Implementada con Gradio 4.x, esta capa contiene los componentes visuales y las funciones reactivas que permiten al usuario cargar documentos, interactuar con los resultados y navegar entre etapas del proceso.

Características principales:

- Componentes visuales: carga de archivos, visualización de tablas, botones y formularios.
- Gestión de estado: sincronización de variables entre pestañas.
- Manejo de eventos: detección de interacciones (clics, cargas).
- Tematización profesional: estilos personalizados CSS adaptados al branding de SENER.
- Diseño responsive para compatibilidad multi-dispositivo.

Orchestration Layer – Capa de Orquestación

Actúa como coordinador global del sistema.

Ubicada en el núcleo principal (gr1.py), esta capa gestiona el estado compartido, la comunicación entre módulos y el flujo lógico entre pestañas de la interfaz.

Responsabilidades clave:

- Control del flujo de navegación y visualización.
- Inicialización y actualización del estado global de ideas.
- Gestión de variables reactivas compartidas.
- Encapsulamiento de funciones críticas de coordinación.

Patrones aplicados:

- *Facade*: unificación de interfaces complejas bajo métodos simples.
- *Mediator*: desacoplamiento entre módulos a través de un punto central.
- *Observer*: notificación de cambios en el estado del sistema.
- *Singleton*: mantenimiento de un estado global consistente.

Business Logic Layer – Capa de Lógica de Negocio

Contiene la lógica central del sistema y los algoritmos que procesan los datos y generan valor analítico.

Módulos principales:

- **Document Processor (pdf_processor_module.py, excel_module.py)**
Extracción, validación y limpieza de ideas desde documentos estructurados (PDF o Excel).
Soporte OCR y parsing inteligente.
- **Analysis Engine (analysis_module2.py)**
Aplicación del marco de análisis de 6 dimensiones estratégicas mediante LLMs. Soporta procesamiento por lotes y genera recomendaciones detalladas.
- **Ranking System (ranking_module.py)**
Algoritmo de priorización basado en criterios estratégicos. Calcula puntuaciones y genera matrices de esfuerzo vs beneficio.

- **Competitor Analysis (competitor_analysis_module.py)**
Análisis competitivo automatizado, combinando LLMs y scraping web con Tavily. Genera informes estructurados en 8 secciones.
- **Data Layer – Capa de Datos**

Responsable de la gestión del estado global del sistema, almacenamiento de resultados y sincronización de información entre módulos.

Elementos destacados:

- Variables persistentes: ideas procesadas, estado analítico, contador de ideas, log de terminal.
- Almacenamiento temporal de archivos cargados.
- Persistencia en disco de resultados finales (PDF, JSON).
- Sincronización de estado entre módulos en tiempo real.

Integration Layer – Capa de Integración

Interfaz del sistema con servicios externos e infraestructura cloud.

Servicios conectados:

- **Azure OpenAI (openai_config.py)**: acceso seguro al modelo GPT-4o con autenticación empresarial, control de cuota y optimización de llamadas.
- **Tavily Search (tavily_config.py)**: motor de búsqueda web especializado para análisis competitivo. Filtrado de dominios, scraping controlado y enriquecimiento semántico.
- **PDF Generator (pdf_generator.py, enhanced_pdf_module.py)**: ensamblado de documentos en formato profesional, soporte multilingüe y diseño corporativo.

Utility Layer – Capa de Utilidades

Contiene herramientas transversales que dan soporte a las demás capas.

Utilidades incluidas:

- **LLM Utils (llm_utils.py)**: plantillas de prompt engineering, parsing de respuestas, validación semántica y gestión de errores.
- **Parallel Processing (parallel_module.py)**: ejecución concurrente con ThreadPoolExecutor, manejo de I/O asíncrono, y optimización del rendimiento para cargas altas.

2.1.3 Flujo de Datos entre Componentes

El sistema implementa un [flujo de datos](#) bidireccional, estructurado en fases, con sincronización continua entre módulos. Este enfoque permite una trazabilidad clara desde la carga inicial de documentos hasta la generación de informes analíticos complejos.

Fase 1 – Ingesta de Documentos

El usuario carga un archivo PDF o Excel desde la interfaz. Este archivo es procesado por los módulos de extracción, donde se detectan, validan y normalizan las ideas. Posteriormente, se almacenan en la memoria global y se actualiza la interfaz en tiempo real.

Fase 2 – Análisis Inteligente

Tras la orden de análisis, el sistema procesa las ideas en lote mediante llamadas a Azure OpenAI. Los resultados estructurados son integrados en un PDF profesional que se guarda y enlaza desde la interfaz. Todo el proceso se realiza en concurrencia controlada con almacenamiento seguro.

Fase 3 – Ranking y Priorización

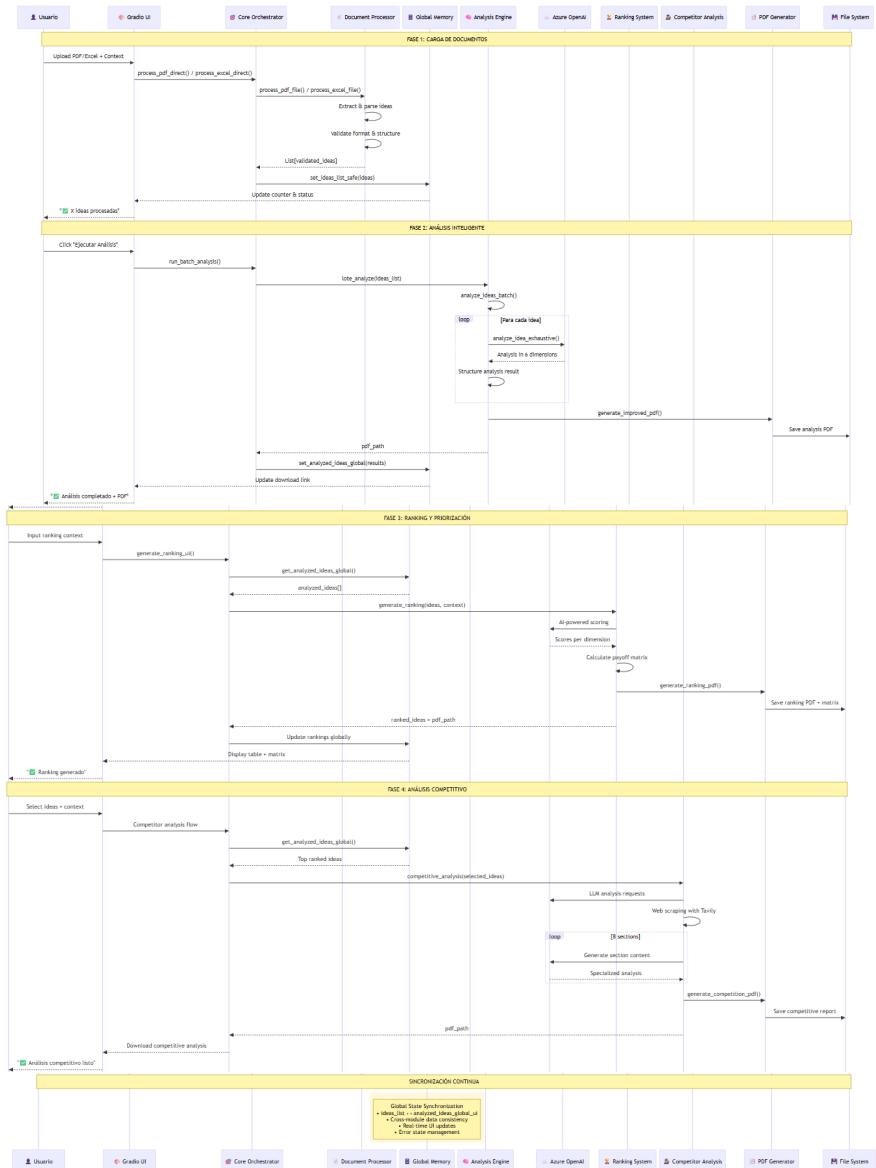
Con base en las ideas ya analizadas, el sistema aplica un algoritmo de puntuación y genera una matriz de esfuerzo vs. beneficio. Los resultados se visualizan en la interfaz y se exportan como documento formal.

Fase 4 – Análisis Competitivo

El usuario selecciona un subconjunto de ideas para análisis competitivo. El módulo correspondiente accede a los datos previos, activa procesos de scraping con Tavily y genera un informe estructurado en ocho secciones con apoyo de LLMs. El documento final se presenta como descarga directa.

Sincronización Continua

La capa de memoria global asegura la consistencia de estado entre módulos. Todos los cambios son reflejados de forma inmediata en la interfaz y persisten en almacenamiento temporal o definitivo según la etapa.



2.1.4 Patrones Arquitectónicos Utilizados

La arquitectura del AI Innovation Agent aplica varios patrones de diseño orientados a sistemas empresariales, con el objetivo de garantizar escalabilidad, modularidad y mantenibilidad a largo plazo.

Layered Architecture

El sistema se organiza en seis capas independientes: presentación, orquestación, lógica de negocio, datos, integración y utilidades. Cada una encapsula responsabilidades específicas y puede evolucionar de forma aislada.

Facade Pattern

El archivo principal gr1.py actúa como fachada, exponiendo funciones de alto nivel que encapsulan la lógica interna de múltiples módulos. Esto permite a la interfaz operar con una API simplificada sin acoplamiento directo con subsistemas complejos.

Observer Pattern

El estado global se gestiona con mecanismos de observación que notifican automáticamente los cambios a módulos dependientes. Esto garantiza la sincronización inmediata entre interfaz, lógica de negocio y almacenamiento.

Factory Pattern

La generación de informes PDF sigue un patrón de fábrica. Según el tipo de análisis (estándar, competitivo o de priorización), se instancia una clase específica de ensamblado documental, manteniendo coherencia visual y estructura formal.

Strategy Pattern

El análisis de ideas se implementa mediante estrategias configurables. Existen variantes optimizadas para análisis exhaustivo, por lotes o competitivo. Esto permite adaptar el comportamiento del sistema a distintos contextos y volúmenes de información.

Template Method Pattern

El procesamiento de documentos aplica una plantilla de pasos comunes con extensiones específicas por tipo de archivo (PDF o Excel). Esto facilita el mantenimiento de consistencia en validación, extracción y parsing.

Command Pattern

Las acciones de interfaz, como ejecutar un análisis o generar un ranking, están encapsuladas como comandos. Esta abstracción permite gestionar reversión de estados, desacoplar eventos y estructurar flujos de interacción más robustos.

Chain of Responsibility

La validación de ideas sigue una cadena de responsabilidad, donde cada etapa (detección, formato, normalización, conteo) aplica reglas específicas antes de permitir el avance al siguiente módulo. Esto reduce errores y facilita el diagnóstico.

2.3 Arquitectura de Conectividad

2.3.1 Configuración Azure OpenAI (endpoints, autenticación)

El sistema implementa una arquitectura de conectividad con Azure OpenAI optimizada para entornos empresariales, basada en principios de seguridad, escalabilidad y mantenibilidad. La integración está centralizada y controlada por el equipo de SENER AI Lab, bajo una suscripción dedicada y con control de acceso restringido.

Configuración Centralizada del Cliente Azure OpenAI

La inicialización del cliente de Azure OpenAI está encapsulada en el módulo `openai_config.py`, donde todas las variables sensibles se gestionan exclusivamente a través de variables de entorno. Esta aproximación asegura la separación entre código fuente y credenciales, y permite una configuración flexible entre entornos de desarrollo, staging y producción.

Parámetros críticos gestionados mediante entorno:

- `AZURE_OPENAI_ENDPOINT` (URL base del servicio)

- AZURE_OPENAI_API_KEY (clave de acceso empresarial)
- DEPLOYMENT_NAME (modelo desplegado: gpt-4o)
- API_VERSION (versión estable actual: 2024-12-01-preview)

Se aplican validaciones estrictas al inicio para garantizar que los parámetros estén correctamente definidos y cumplan con requisitos mínimos de seguridad (longitud, formato, protocolo HTTPS).

Detalles de Conexión en Entorno Productivo

- **Base URL:** https://azureaiservices-ailab-dev-018827451690240.openai.azure.com/
- **Modelo Desplegado:** GPT-4o Turbo (gpt-4o)
- **Versión de API:** 2024-12-01-preview (última estable)
- **Rate Limits Empresariales:** 100.000 tokens por minuto / 600 solicitudes por minuto
- **Autenticación:** API Key mediante header (Authorization: Bearer) y cabecera api-key específica de Azure

Contexto corporativo:

- **Región Primaria:** East US (backup: West Europe)
- **Grupo de Recursos:** ailab-dev-resources
- **Suscripción Azure:** SENER AI Lab Subscription
- **Modelo de Coste:** Standard Tier (pago por token)
- **Gobernanza:** Administración exclusiva del equipo AI Lab de SENER

Seguridad y Gobernanza

La autenticación se implementa bajo un enfoque de múltiples capas. La clave de API debe cumplir con los siguientes requisitos:

- Longitud mínima de 32 caracteres
- Formato alfanumérico con caracteres seguros
- Transmisión únicamente bajo conexiones HTTPS
- Validación de entorno y endpoint antes del primer uso
- Verificación activa de conectividad en el arranque del servicio

Los headers de autenticación enviados a Azure OpenAI incluyen:

- Authorization: Bearer <API_KEY>
- Content-Type: application/json
- api-key: <API_KEY> (cabecera obligatoria en servicios Azure)
- User-Agent: SENER-AI-Innovation-Agent/1.0

Gestión de Credenciales en Entorno Docker

Durante la ejecución en contenedores Docker, las credenciales y configuraciones son inyectadas mediante variables de entorno en el momento del deployment. El Dockerfile y docker-compose.yml han sido preparados para aceptar esta configuración desde archivos .env o sistemas CI/CD corporativos.

Variables mínimas requeridas para despliegue:

- AZURE_OPENAI_ENDPOINT
- AZURE_OPENAI_API_KEY
- DEPLOYMENT_NAME
- API_VERSION
- LOG_LEVEL

Ejemplo de ejecución:

```
docker run \
-AZURE_OPENAI_ENDPOINT="https://azureaiservices-ailab-dev-018827451690240.openai.azure.com/" \
-AZURE_OPENAI_API_KEY="..." \
-DEPLOYMENT_NAME="gpt-4o" \
-API_VERSION="2024-12-01-preview" \
-7860:7860 \ai-innovation-sener
```

Esta configuración permite desacoplar las credenciales del código fuente y habilita la integración segura con pipelines de despliegue automatizados.

2.3.2 Integración con APIs Externas: Tavily para Inteligencia Competitiva

El sistema incorpora un motor avanzado de búsqueda web basado en **Tavily API**, configurado específicamente para entornos de análisis competitivo. Esta integración permite enriquecer el análisis de ideas con información contextual del ecosistema industrial y tecnológico.

Configuración Técnica

La configuración de Tavily está centralizada en el archivo `tavily_config.py`, donde se definen:

- Clave de API gestionada vía variable de entorno (`TAVILY_API_KEY`)
- Nombre identificativo de cliente (`TAVILY_API_NAME = "Sener2"`)
- Mecanismo de fallback controlado para entornos de desarrollo

La integración se realiza mediante LangChain, permitiendo una orquestación optimizada de búsquedas desde componentes LLM.

Parámetros de Búsqueda

El motor se configura con un perfil avanzado:

- Profundidad de búsqueda: "advanced" (crawling profundo)
- Volumen de resultados: 10–15 por consulta
- Filtros de inclusión: sitios corporativos, fuentes de noticias, papers técnicos
- Filtros de exclusión: redes sociales, foros, contenido publicitario
- Idiomas permitidos: español e inglés
- Formato de respuesta: JSON estructurado

Además, las queries se generan dinámicamente a partir de prompts optimizados por LLM, incorporando términos técnicos, sinónimos y expansiones semánticas para maximizar cobertura y relevancia.

Arquitectura de Scraping en Cascada

El sistema implementa un modelo híbrido de scraping dividido en cinco niveles de profundidad progresiva:

1. **Nivel 1 – Tavily (búsqueda estructurada primaria)**
2. **Nivel 2 – Scraping dirigido (URLs específicas)**
3. **Nivel 3 – BeautifulSoup (parsing de HTML)**
4. **Nivel 4 – Selenium WebDriver (páginas con JavaScript dinámico)**
5. **Nivel 5 – Llamadas manuales a APIs públicas**

Esta arquitectura permite realizar extracciones segmentadas por campos, validar contenido con patrones específicos y aplicar deduplicación por URL y contenido.

2.3.3 Manejo de Timeouts y Reintentos

El sistema aplica una estrategia integral de control de errores basada en timeouts jerárquicos, reintentos automáticos y mecanismos de recuperación progresiva. El objetivo es garantizar robustez frente a fallos externos sin afectar la experiencia del usuario ni la integridad del análisis.

Timeouts por Nivel de Operación

Los tiempos límite se definen de forma diferenciada según la criticidad y duración esperada de cada operación:

- **Azure OpenAI (SDK):** 60 segundos
- **Sistema Operativo (signal.alarm):** 60 segundos
- **Requests HTTP (Scraping):** 10–15 segundos
- **Timeout de conexión:** 5 segundos
- **Generación de PDFs:** hasta 300 segundos

Las operaciones con Azure están encapsuladas en funciones que incluyen mecanismos de cancelación forzada mediante señales de sistema en caso de bloqueo o latencia excesiva.

Timeouts Adaptativos por Proceso

Cada módulo ajusta dinámicamente el tiempo máximo según el tipo de tarea:

- Análisis rápido: 30 s
- Análisis estándar: 60 s
- Procesamiento batch: 120 s
- Análisis competitivo: 180 s
- Exportación documental: 300 s

Estas configuraciones permiten una distribución eficiente de recursos sin bloquear el sistema ante tareas de larga ejecución.

Sistema de Reintentos Configurables

El cliente de Azure OpenAI está configurado con una política de reintentos automática (`max_retries=3`). Este comportamiento se refuerza en los módulos críticos con reintentos explícitos en caso de:

- Timeout (TimeoutError)
- Fallo de red (ConnectionError)
- Límite de uso excedido (RateLimitError)

No se reintenta en casos de error de autenticación o fallos permanentes del API.

Estrategia de Backoff Exponencial

Los reintentos implementan un retardo progresivo basado en backoff exponencial (2s, 4s, 8s). Esto evita saturación de servicios y reduce la probabilidad de fallos repetidos.

Recuperación por Degradación Funcional

En caso de múltiples fallos consecutivos, el sistema activa estrategias de graceful degradation, reduciendo progresivamente la complejidad de la solicitud:

- Reducción del número de tokens
- Disminución del temperature para mayor determinismo
- Uso de prompts simplificados
- Generación de respuestas fallback como último recurso

Esto permite garantizar, incluso en condiciones adversas, una respuesta mínima válida, protegiendo la continuidad operativa del sistema.

2.3.4 Estrategias de Rate Limiting

El sistema incorpora estrategias avanzadas de control de velocidad diseñadas para maximizar el rendimiento del agente sin sobrepasar los límites definidos por Azure OpenAI en su modalidad empresarial. Estas medidas garantizan una operación estable, eficiente y escalable incluso en escenarios de alta concurrencia.

Los parámetros de uso establecidos por Azure OpenAI son los siguientes:

- 100.000 tokens por minuto (TPM)
- 600 solicitudes por minuto (RPM)
- Hasta 20 conexiones simultáneas
- Tiempo máximo por solicitud: 60 segundos

Con esta capacidad, el sistema está optimizado para alcanzar hasta 50 ideas analizadas por minuto en procesamiento por lotes, manteniendo una latencia promedio inferior a 2 segundos por análisis individual.

Connection Pooling y HTTP/2

Se ha implementado una arquitectura optimizada de conexión basada en `httpx` con soporte para HTTP/2 y pooling persistente de conexiones:

- Reutilización de sockets TCP para minimizar el overhead de conexión.
- Multiplexación eficiente de llamadas concurrentes.
- Reducción de latencia en más de un 90 % respecto a sesiones sin pooling.

Caché Inteligente y Dedución de Llamadas Redundantes

Para evitar solicitudes innecesarias a la API de Azure OpenAI, el sistema emplea un mecanismo de caché con hashing del prompt (MD5):

- Respuestas reutilizadas para entradas idénticas en la misma sesión.
- Reducción de hasta un **80 % en llamadas duplicadas**.
- Mejora significativa en el tiempo de respuesta y ahorro de tokens.

Procesamiento por Lotes Dinámico

El sistema ajusta automáticamente el tamaño de los lotes según el número de ideas y la carga de trabajo:

- Procesamiento paralelo con ThreadPoolExecutor, dentro del límite de RPM.
- División eficiente en bloques de 3 a 8 ideas por lote.
- Distribución balanceada para maximizar el throughput.

Throttling Adaptativo y Gestión de Colas

El motor de análisis aplica técnicas de **throttling inteligente** que controlan el ritmo de llamadas y activan medidas de contención en caso de saturación:

- Espaciado automático de 100 ms entre solicitudes para evitar ráfagas.
- Reintentos con backoff exponencial ante errores tipo 429 (rate limit exceeded).
- Reducción progresiva de concurrencia ante eventos de saturación.

Además, se gestiona una **cola FIFO con prioridad**, que permite:

- Reordenar análisis por urgencia.
- Monitorear profundidad de cola y tiempos de espera.
- Activar estrategias de “degradación progresiva” si se superan umbrales críticos.

Monitorización y Métricas Operativas

Se lleva un seguimiento continuo del comportamiento del sistema en cuanto a uso de recursos y eficiencia operativa. Entre las métricas clave se incluyen:

- Uso de tokens por minuto y solicitudes por segundo.
- Tasa de errores y respuestas con código 429.
- Tiempo medio de respuesta y éxito por lote.
- Profundidad de la cola y número de workers activos.

Los umbrales de alerta están definidos para actuar automáticamente ante condiciones adversas:

- 80 % del uso de tokens → reducción de tamaño de lote.
- 90 % del límite de solicitudes → aumento de intervalo entre llamadas.
- Tres o más errores 429 consecutivos → activación de backoff.
- Tiempo de respuesta superior a 30 segundos → simplificación automática de prompts.

2.4 Arquitectura de Procesamiento

2.4.1 Sistema de Threading Paralelo

La arquitectura implementa un sistema avanzado de parallelización basado en ThreadPoolExecutor, diseñado para optimizar el rendimiento, reducir la latencia en operaciones masivas y cumplir estrictamente con los límites de uso de Azure OpenAI.

El diseño contempla un modelo **estratificado y adaptativo** por módulo, permitiendo la ejecución simultánea de tareas críticas sin generar bloqueos ni sobrecargas.

Configuración por Módulo

- **Análisis de Ideas (analysis_module2):** Procesamiento por lotes con hasta 10 hilos simultáneos, ajuste dinámico según volumen de ideas y liberación explícita de memoria entre bloques (gc.collect()).
- **Análisis Competitivo:** Parallelización por secciones y por ideas, en dos fases coordinadas. Se ejecutan hasta 6 secciones en paralelo y 4 ideas de forma simultánea, manteniendo la independencia entre workers.
- **Generación de PDFs:** Procesamiento concurrente de ideas para extracción, mejora y ensamblado en informes, con hasta 12 workers por lote.
- **Ranking:** Ejecución optimizada con tracking de progreso (tqdm) y aislamiento de errores por idea para evitar que fallos puntuales afecten a todo el conjunto.

Este modelo permite escalar horizontalmente la carga del sistema respetando el límite de **600 solicitudes por minuto**, distribuyendo eficientemente los recursos de procesamiento.

2.4.2 Gestión de Memoria y Optimizaciones

Dado el procesamiento intensivo de datos, el sistema incluye una arquitectura multicapa para la gestión de memoria, orientada a prevenir errores por saturación y mejorar la eficiencia operativa.

Estrategias de Gestión

- **Seguridad en Variables Globales:** Validación estricta de estructuras antes de asignación (ideas_list, analyzed_ideas_global). Se limita la longitud de logs y se rotan automáticamente.
- **Procesamiento por Lotes:** Aislamiento de workers, protección por timeout por idea (20s), y liberación forzada de memoria tras cada bloque.
- **Generación de PDFs:** Uso de cachés de fuentes, control de tiempo por sección (20s), y rutinas de guardado alternativo ante errores.

Estas estrategias aseguran la estabilidad incluso ante picos de carga o lotes de ideas con contenido malformado.

Validación y Limpieza Automatizada

El sistema incorpora validaciones robustas en la asignación de datos y mecanismos de recuperación automática:

- Validaciones tipo y estructura en tiempo real antes de uso o almacenamiento.
- Limpieza programática completa de memoria (`clear_all_global_memory()`) tras operaciones críticas o detección de corrupción.
- Garantía de consistencia en los accesos a estado global, evitando fallos derivados de objetos incompletos o corruptos.

2.4.3 Manejo de Errores y Recuperación

El sistema está diseñado con una **arquitectura de resiliencia distribuida en seis niveles**, que permite continuar operaciones aun en presencia de errores, gracias a técnicas de degradación controlada, aislamiento y recuperación.

Niveles de Control

1. **Sistema:** Timeouts globales por operación (300s) y control por señal para llamadas individuales a Azure (60s). Cada worker opera en aislamiento y cuenta con reintentos configurables (`max_retries=3`).
2. **API:** Detección de errores en respuestas, validación estructural del output, backoff exponencial y recuperación vía caché en caso de fallo.
3. **Procesamiento:** Continuación del análisis aunque una idea falle, generación parcial de PDFs y limpieza de memoria tras errores para evitar acumulaciones.
4. **Datos:** Validación de tipo y claves requeridas, sanitización de contenido para evitar errores de parsing y estructuras por defecto para recuperación parcial.

Recuperaciones Específicas

- **Timeouts Unix:** Se utiliza `signal.alarm()` para abortar operaciones de análisis que excedan los 60 segundos por idea.
- **Parsing de JSON con Fallback:** Se implementan hasta seis estrategias para recuperar contenido malformado, incluyendo completado automático y ensamblado por secciones.
- **Generación de PDF:** En caso de fallo en el guardado final, se aplica una estrategia de fallback con nombre alternativo para evitar pérdida de resultados.

2.4.4 Logging y Monitoreo

El sistema incorpora un esquema de **observabilidad multinivel**, diseñado para capturar eventos relevantes desde la perspectiva del usuario final hasta los mecanismos internos de análisis, garantizando trazabilidad completa y depuración eficiente.

Logging Multicapa

La arquitectura de logging se estructura en cuatro niveles principales:

- **Nivel 1 – Usuario Final (Terminal Log):** Muestra mensajes clave en tiempo real con timestamp y rotación automática de líneas. Los mensajes recientes se visualizan primero para facilitar el seguimiento activo.
- **Nivel 2 – Aplicación (Logging Estándar):** Registra eventos técnicos, trazas y mensajes de `print()` interceptados automáticamente, con formato estandarizado y nivel de severidad (INFO, ERROR...).

- **Nivel 3 – Módulo Específico:** Cada componente crítico (ranking, análisis, generación de PDF, scraping competitivo) incorpora trazas detalladas, incluyendo tiempo de ejecución y progresos parciales.
- **Nivel 4 – Debug Avanzado:** Se almacenan temporalmente archivos *.json, respuestas API, métricas de rendimiento y trazas completas de errores (traceback), facilitando análisis forense.

Interceptación Global de Output

El sistema reemplaza la función print() estándar por un mecanismo personalizado que:

- Registra cada mensaje con sello temporal en el log global del usuario.
- Aplica rotación automática para evitar saturación de memoria.
- Permite mantener simultáneamente la visibilidad en consola para debugging.

Monitoreo Paralelo en Tiempo Real

Durante la ejecución concurrente (por ejemplo, en análisis masivo), se informa dinámicamente el avance del procesamiento de ideas, errores individuales por worker y eventos críticos. Esto se logra con herramientas como concurrent.futures.as_completed() y tqdm.

Medición de Rendimiento

Se utilizan context managers (timed()) para medir de forma precisa la duración de operaciones sensibles como generación de PDFs, análisis de ideas o scraping. Los resultados se registran internamente y están disponibles para logging extendido en entornos de desarrollo o depuración.

2.5 Arquitectura de UI

2.5.1 Diseño Modular con Gradio

La interfaz está diseñada con un enfoque modular basado en **Gradio**, integrando navegación dinámica, estado compartido y componentes reutilizables. Esta estructura facilita la extensibilidad, la claridad visual y la experiencia de usuario fluida.

Estructura de Navegación

- Diseño principal con barra lateral de navegación y área de contenido central.
- Componentes responsivos basados en gr.Column adaptables a distintas resoluciones.
- Theming personalizado con variables CSS corporativas para alineación visual con SENER.

Módulos Funcionales

- **Carga de Documentos:** Permite cargar archivos PDF/Excel con contexto asociado.
- **Análisis Inteligente:** Interfaz para ejecutar análisis por lotes y visualizar resultados.
- **Ranking:** Entrada de contexto estratégico y visualización de matrices de priorización.
- **Análisis Competitivo:** Configuración de scraping y descarga de informes LLM enriquecidos.

Componentes Transversales

- Cabecera con iconografía y branding de SENER.
- Logs de estado en tiempo real visibles al usuario.
- Gestión de ficheros (upload/download) con vista previa.
- Mecanismos de feedback y manejo de errores con visualización reactiva.

2.5.2 Sistema de Estado Compartido

El sistema mantiene una sincronización activa entre todos los módulos a través de variables globales que controlan el estado actual de:

- Lista de ideas procesadas y analizadas (ideas_list, analyzed_ideas_global_ui).
- Logs visibles para el usuario.
- Sección activa de la interfaz.

Los cambios realizados en una pestaña (por ejemplo, resultados del ranking) se reflejan automáticamente en otras secciones que dependen de esos datos, asegurando consistencia operativa.

2.5.3 Componentes Reutilizables

El sistema UI se apoya en una estrategia de componentes reutilizables, orientada a mantener la consistencia visual, reducir redundancia de código y facilitar la extensión modular.

Headers Modulares

Se ha diseñado un sistema de cabeceras reutilizables con estilo profesional, centrado en la presentación clara de cada sección funcional. Los headers:

- Integran íconos representativos, tipografía destacada y descripciones contextualizadas.
- Se construyen mediante una plantilla HTML parametrizada (HEADER_TEMPLATE) que encapsula:
 - Fondo con gradientes corporativos.
 - Sombra proyectada suave (box-shadow) para profundidad.
 - Texto estilizado con glow effects y jerarquía visual clara.
 - Compatibilidad completa con el sistema de tematización CSS.

Esto permite mantener identidad visual homogénea y rápida adaptabilidad a nuevas secciones UI.

Botones Especializados

Cada módulo funcional implementa botones con tematización visual propia:

- Botones del módulo de análisis competitivo utilizan un gradiente púrpura (#9b59b6 → #8e44ad) con sombra animada al interactuar.
- El sistema de ranking aplica una paleta naranja de advertencia (tech-warning) combinada con efectos pulsantes al hacer hover.
- Todos los botones están definidos mediante diccionarios de estilo (BUTTON_STYLES), lo que permite abstracción y reuso sin acoplamiento.

2.5.4 CSS y Tematización Profesional

El sistema UI aplica una tematización completamente personalizada basada en variables CSS centralizadas, que definen la identidad visual del **SENER AI Lab**.

Variables CSS Corporativas

La personalización se gestiona desde :root, donde se definen variables reutilizables para:

- **Colores base y secundarios:** --tech-primary, --tech-accent, --tech-warning, entre otros.
- **Textos y sombras:** Tonos tipográficos (primary, secondary, tertiary) y sombras (tech-shadow-2) ajustadas para suavidad y contraste.
- **Fondos con gradientes:** Utilizados en cabeceras, botones y elementos interactivos.

Esta estrategia permite escalabilidad visual, facilitando ajustes globales de estilo desde un único punto de control.

Diseño Responsive con Grid

Se ha implementado un sistema responsive avanzado con CSS Grid, con el objetivo de garantizar una experiencia fluida en todo tipo de pantallas:

- La clase .process-grid adapta dinámicamente columnas a partir de 280px, asegurando legibilidad en móviles y escritorios.
- .dimension-grid proporciona layouts compactos para secciones analíticas específicas.
- Todos los elementos interactivos incluyen animaciones suaves (hover effects) para mejorar la percepción de respuesta sin penalizar rendimiento.

3. MÓDULOS CORE - ANÁLISIS DETALLADO

3.1 Módulo Principal: Interface (gr1.py)

3.1.1 Funciones Globales Críticas

El módulo gr1.py constituye el núcleo de la interfaz principal del agente, y alberga una serie de funciones críticas para el manejo de estado, validación de datos y logging. Estas funciones aseguran la integridad del flujo de datos entre los distintos módulos del sistema y permiten una operación estable, resiliente y controlada.

Validación de Datos Multicapa

El sistema de validación implementado actúa como firewall entre componentes, filtrando entradas inválidas y evitando la propagación de errores lógicos. Las funciones set_ideas_list_safe y validate_idea_format validan y normalizan de forma automática la estructura esperada de las ideas.

Características destacadas:

- Validación doble por tipo y claves obligatorias.
- Logging informativo según resultado (actualización, vacío o ignorado).

- Protección frente a corrupción estructural: nunca se sobrescriben datos si no cumplen criterios mínimos.

Gestión de Estado Global y Sincronización

Se utilizan variables globales centralizadas para el manejo del estado compartido en la interfaz y sus módulos dependientes. Estas variables controlan desde la lista de ideas procesadas hasta los contadores de interacción y el historial de eventos.

Funciones como `set_analyzed_ideas_global` y `get_analyzed_ideas_global` aseguran sincronización crítica entre módulos, permitiendo actualizaciones seguras y recuperación robusta del estado ante reinicios o fallos parciales.

Limpieza Estratégica de Memoria

La función `clean_global_memory` permite una limpieza completa de las variables globales relacionadas con las ideas, forzando una regeneración total del contenido. Se emplea para garantizar consistencia tras operaciones críticas o reconfiguraciones.

Además, `update_idea_counter` ofrece un mecanismo seguro para la actualización del contador global, validando siempre el tipo de entrada.

Sistema de Logging

El módulo implementa un sistema híbrido de logging con interceptación de `print()` y almacenamiento circular en memoria. Las funciones `log_message` y `get_terminal_log` permiten registrar, visualizar y rotar automáticamente los eventos más recientes, con control de memoria y timestamps integrados.

Se intercepta globalmente la función `print` para redirigir todos los mensajes al sistema de log sin pérdida de funcionalidad original, garantizando trazabilidad total del comportamiento del sistema desde la interfaz.

3.1.2 Sistema de Pestañas y Navegación

El módulo principal (`gr1.py`) implementa una arquitectura de navegación modular, basada en el framework **Gradio**, que permite una experiencia fluida, escalable y desacoplada para la interacción del usuario. Este sistema estructura la interfaz en secciones funcionales bien definidas, controladas mediante un estado compartido y actualizaciones sincronizadas.

Estructura General y Visibilidad Dinámica

La interfaz está compuesta por un área de contenido principal y una barra lateral de navegación. Cada botón en la barra lateral activa una sección correspondiente, mientras las demás se ocultan dinámicamente.

```
with gr.Row():
    with gr.Column(scale=4): # Área de contenido
        content_area = gr.HTML("", elem_id="main_content")
    with gr.Column(scale=1, min_width=200): # Navegación lateral
        nav_docs = gr.Button("📄 Carga de Documentos")
        nav_analysis = gr.Button("📊 Análisis de Ideas")
```

```
nav_ranking = gr.Button("Ranking de Ideas")
nav_competitor = gr.Button(" Análisis de Competidores")
```

Cada sección está encapsulada en su propio contenedor (gr.Column) con visibilidad controlada. Esto permite alternar entre vistas sin necesidad de recargar la aplicación ni perder el estado del sistema.

```
with gr.Column(visible=True, elem_id="docs_section") as docs_section:
    create_document_upload_tab_content()

with gr.Column(visible=False, elem_id="analysis_section") as analysis_section:
    create_analysis_tab_content()

# ... (secciones restantes)
```

Navegación Sincronizada por Estado Compartido

La navegación se orquesta mediante una variable de estado (gr.State) que determina qué sección debe estar visible en cada momento. Un controlador centralizado (show_section) actualiza las secciones simultáneamente:

```
def show_section(section_name):
    """Actualiza visibilidad de todas las secciones según la pestaña activa"""
    return (
        gr.update(visible=(section_name == "docs")),
        gr.update(visible=(section_name == "analysis")),
        gr.update(visible=(section_name == "ranking")),
        gr.update(visible=(section_name == "competitor"))
    )
```

Cada botón de navegación lanza este controlador con el nombre correspondiente:

```
nav_docs.click(fn=lambda: show_section("docs"), outputs=[docs_section, analysis_section, ranking_section,
competitor_section])
nav_analysis.click(fn=lambda: show_section("analysis"), outputs=[docs_section, analysis_section, ranking_section,
competitor_section])
```

Modularidad y Reutilización de Contenido

Las secciones de contenido están definidas como funciones independientes reutilizables (*_tab_content), lo que permite invocar el mismo contenido tanto dentro de pestañas (gr.Tab) como dentro de columnas (gr.Column) sin duplicación de código.

```
def create_document_upload_tab_content():
    with gr.Column():
        gr.HTML("""<div style="...>...</div>""") # Header visual
        pdf_file = gr.File(label="Archivo PDF", file_types=[".pdf"])
        pdf_context = gr.Textbox(label="Contexto (Opcional)")
        pdf_button = gr.Button("Procesar PDF")
        pdf_button.click(fn=process_pdf_direct, inputs=[pdf_file, pdf_context])
```

Este enfoque modular mejora la mantenibilidad, permite pruebas independientes y facilita la integración de nuevas secciones o funcionalidades sin alterar la arquitectura existente.

3.1.3 Procesamiento de Documentos

El procesamiento de documentos PDF se gestiona a través de la función `process_pdf_direct()`, que actúa como punto de entrada asíncrono para la extracción estructurada de ideas a partir de archivos cargados por el usuario. Este módulo incorpora validación contextual, control de errores y normalización avanzada del contenido extraído.

Flujo General de Ejecución

La función `process_pdf_direct()` realiza los siguientes pasos críticos:

1. **Validación de entrada:** Verifica que se haya cargado un archivo PDF válido.
2. **Logging contextual:** Si se proporciona un contexto adicional, se registra su contenido y longitud para trazabilidad.
3. **Procesamiento principal:** Invoca `process_pdf_file()` para analizar el contenido del PDF.
4. **Normalización estructural:** Recorre las ideas detectadas, aplica limpieza textual y extrae títulos válidos.
5. **Validación final:** Filtra ideas con baja calidad y preserva el orden original.

```
async def process_pdf_direct(pdf_file, context):
    if pdf_file is None:
        return "Por favor, sube un archivo PDF.", "0"

    # Logging del contexto si se proporciona
    if context and context.strip():
        print(f"Contexto recibido: {len(context)} caracteres")
    else:
        print("No se proporcionó contexto")

    # Procesamiento principal del PDF
    ideas, status = process_pdf_file(pdf_file.name, context)
    if not ideas:
        return f"X {status}", "0"
```

Extracción Inteligente de Títulos

Cada idea detectada se somete a un proceso de extracción de título mediante separación por saltos de línea y limpieza con expresiones regulares. Este paso asegura que cada entrada posea un encabezado sintético válido, que puede ser utilizado posteriormente en la interfaz o el informe.

```
title = re.sub(r'^(\d*\s*\d*[.:]\s*){2,}', ' ', title, flags=re.IGNORECASE)
# Limpieza final de etiquetas numéricas
title = re.sub(r'^\d*\s*\d*[.:]\s*', ' ', title, flags=re.IGNORECASE)
```

Criterios de descarte:

- Título con menos de 4 caracteres.
- Falta de letras (solo dígitos o símbolos).
- Contenido vacío o repetitivo.

Detección Multiestrategia de Ideas (`detect_ideas_basic()`)

El sistema emplea un enfoque multicapa para extraer ideas textuales desde documentos, priorizando la robustez ante diferentes formatos:

- **Estrategia 0:** Reutiliza ideas previamente cargadas si existen.
- **Estrategia 1:** Detección por bullet points con expresiones regulares.
- **Estrategia 2:** Identificación por patrones estándar (ítems numerados o con etiquetas).
- **Estrategia 3:** Extracción por párrafos largos.
- **Estrategia 4:** Fallback basado en líneas largas (>60 caracteres).

```
bullet_pattern = r'(?:^|\n)\s*\s*IDEA\s*\d+.*?(?=(?:\n\s*\s*IDEA\s*\d+|\Z))'
matches = re.findall(bullet_pattern, text, re.DOTALL | re.IGNORECASE)
```

El algoritmo asegura la unicidad de cada idea y limita el conjunto a 20 entradas para preservar la eficiencia.

Control de Calidad y Logging

Se incorporan logs para registrar el número de ideas detectadas, el uso de estrategias específicas y los títulos descartados por calidad. Estas métricas son fundamentales para monitorear el rendimiento del sistema de extracción.

3.1.4 CSS y Diseño Avanzado

El sistema implementa un conjunto integral de estilos CSS personalizados y tecnologías visuales para asegurar una experiencia de usuario profesional, coherente con la identidad tecnológica de SENER AI Lab. La arquitectura de diseño combina variables CSS, efectos visuales avanzados, responsividad progresiva y microinteracciones dinámicas.

Paleta de Colores Corporativa

Se define un sistema de variables CSS que encapsula los colores base, superficies, gradientes y efectos lumínicos utilizados en toda la interfaz:

```
:root {
  --tech-primary: #0ea5e9;
  --tech-secondary: #3b82f6;
  --tech-accent: #06b6d4;

  --tech-bg-primary: #0f172a;
  --tech-bg-secondary: #1e293b;
  --tech-surface-1: #1e293b;
  --tech-surface-2: #334155;
  --tech-surface-3: #475569;

  --tech-border-1: #334155;
  --tech-glow-primary: rgba(14, 165, 233, 0.3);
  --tech-glow-accent: rgba(6, 182, 212, 0.2);

  --tech-gradient-1: linear-gradient(135deg, var(--tech-surface-1), var(--tech-surface-2));
  --tech-gradient-2: linear-gradient(135deg, var(--tech-surface-2), var(--tech-surface-3));
  --tech-gradient-3: linear-gradient(135deg, var(--tech-primary), var(--tech-accent));
}
```

Este enfoque permite una tematización flexible y centralizada, facilitando la adaptación futura del sistema a nuevas identidades visuales o estándares corporativos.

Efectos de Animación: Sistema Shimmer

Se ha implementado un efecto visual de tipo shimmer para headers, inspirado en las interfaces modernas de diseño técnico:

```
.header-container::before {  
    content: " ";  
    position: absolute;  
    left: -200%;  
    width: 400%;  
    height: 100%;  
    background: linear-gradient(90deg, transparent 0%, var(--tech-glow-primary) 50%, transparent 100%);  
    animation: shimmer 12s ease-in-out infinite;  
    opacity: 0.6;  
}  
  
@keyframes shimmer {  
    0% { left: -200%; }  
    100% { left: 200%; }  
}
```

Este efecto refuerza visualmente las áreas clave de navegación o información contextual, sin comprometer el rendimiento.

Interacciones Avanzadas para Botones

Los botones están diseñados con microinteracciones en hover, incluyendo transformación de escala, sombras dinámicas y overlays graduales:

```
.btn button:hover::before {  
    left: 0; /* Activación shimmer en hover */  
}  
  
.btn button:hover {  
    transform: translateY(-3px) scale(1.02);  
    box-shadow: 0 15px 40px var(--tech-shadow-2), 0 0 0 1px var(--tech-glow-accent);  
}
```

Este comportamiento mejora la experiencia táctil y visual, especialmente en interfaces críticas como el ranking o análisis por lotes.

3.2 Motor de Análisis (analysis_module2.py)

El módulo analysis_module2.py constituye el núcleo analítico del sistema **AI Innovation Agent**. Diseñado como un motor de evaluación técnica y estratégica, este componente transforma conjuntos de ideas en análisis estructurados mediante técnicas de procesamiento paralelo, integración con modelos de lenguaje avanzados (Azure OpenAI GPT-4o) y un sistema de gestión de errores orientado a la resiliencia operativa.

3.2.1 Algoritmo de Análisis por Lotes

La función central del módulo es `analyze_ideas_batch()`, que permite ejecutar análisis masivos de ideas de forma paralela, gestionando eficazmente la carga sobre los servicios externos de IA sin comprometer el rendimiento global.

```
def analyze_ideas_batch(ideas_list, title="", context="", template=None):
    max_workers = min(10, len(validated_ideas))
    with concurrent.futures.ThreadPoolExecutor(max_workers=max_workers) as executor:
        futures = [executor.submit(analyze_idea, idea) for idea in validated_ideas]
        for future in concurrent.futures.as_completed(futures):
            ...

```

Aspectos clave del procesamiento por lotes:

- Uso de ThreadPoolExecutor para análisis concurrente con límite dinámico de workers.
- Seguimiento de progreso individual por idea.
- Gestión robusta de excepciones y errores por worker.

Configuración de ejecución y control de carga:

```
THREAD_CONFIGURATION = {
    "max_workers_calculation": "min(10, len(ideas_list))",
    "timeout_per_worker": "60 seconds (signal.alarm)",
    "memory_management": "Per-worker isolation",
    "error_recovery": "Individual worker exception handling"
}
```

Política de asignación de workers:

- De 1 a 5 ideas → 1 a 5 workers
- De 6 a 10 ideas → 6 a 10 workers
- Más de 10 ideas → máximo 10 workers (limitación forzada)

Timeout por worker:

Para evitar bloqueos o llamadas excesivamente largas al API de OpenAI, cada worker implementa un control de tiempo estratificado mediante `signal.alarm()`:

```
signal.signal(signal.SIGALRM, timeout_handler)
signal.alarm(60)
response = client.chat.completions.create(...)
signal.alarm(0)
```

Manejo de errores por worker individual:

```
try:
    response = client.chat.completions.create(...)
    if not response or not response.choices:
        raise ValueError("Respuesta vacía")
    ...
except Exception as e:
    print(f"X Error analizando idea #{idea_obj['index']}: {str(e)}")
```

3.2.2 Procesamiento de Texto Analítico

Integración con Azure OpenAI (GPT-4o)

El módulo utiliza **Azure OpenAI GPT-4o** como backend de análisis para evaluar el potencial de cada idea. La configuración garantiza compatibilidad con requisitos empresariales:

```
client = get_openai_client()
DEPLOYMENT_NAME = get_deployment_name()

response = client.chat.completions.create(
    model=DEPLOYMENT_NAME,
    messages=[{"role": "system", "content": "..."}, {"role": "user", "content": prompt}],
    max_tokens=4000,
    temperature=0.6,
    timeout=60
)
```

Prompt estructurado en seis dimensiones

El análisis generado para cada idea sigue una estructura fija, diseñada para garantizar consistencia, profundidad técnica y legibilidad:

- RESUMEN EJECUTIVO
- ANÁLISIS TÉCNICO
- POTENCIAL DE INNOVACIÓN
- ALINEACIÓN ESTRATÉGICA CON SENER
- VIABILIDAD COMERCIAL
- VALORACIÓN GLOBAL

Este formato está reforzado con instrucciones específicas que aseguran la correcta delimitación de secciones, uso exclusivo de caracteres ASCII y longitud mínima por sección (250–300 palabras).

Estrategias de optimización y cacheado

El módulo implementa mecanismos de optimización para reducir costes y tiempo de ejecución:

- Cacheo en memoria (`_last_analyzed_ideas`)
- Respaldo en disco (`last_analysis_results.json`)
- Validación previa de respuestas antes de análisis pesado
- Batching de llamadas mediante ThreadPoolExecutor

```
def get_analyzed_ideas():
    if _last_analyzed_ideas:
        return _last_analyzed_ideas
    if os.path.exists(results_path):
        with open(results_path) as f:
            return json.load(f)
```

Sanitización y normalización de respuestas

Para asegurar la calidad del contenido analizado y su correcta representación en informes (por ejemplo, PDF), se ejecuta un pipeline de normalización que elimina marcadores de markdown, corrige puntuación y limpia caracteres especiales:

```
def normalize_text_for_pdf(text):
    text = re.sub(r'#{1,6}\s+', " ", text)
    text = re.sub(r'\*\|*\|\*\|_\|^\|', " ", text)
    text = re.sub(r'\s+', ' ', text)
    ...
    return '\n\n'.join(processed_paragraphs)
```

Se incluye además una función de limpieza de emergencia que convierte todo texto a formato ASCII puro, útil en entornos que no soportan Unicode:

```
def emergency_clean_text(text):
    return ''.join(c if ord(c) < 128 else '' for c in text)
```

3.2.3 Análisis Exhaustivo Individual

La función `analyze_idea_exhaustive()` permite realizar un análisis detallado de una sola idea, siguiendo los estándares metodológicos del departamento de innovación de Sener. Esta función está diseñada para escenarios donde se requiere una evaluación profunda y estructurada, en lugar de un procesamiento en lote.

Su implementación incluye una validación exhaustiva de entrada, gestión avanzada de contexto específico de Sener y control de timeout por proceso individual:

```
# Líneas 1364–1631: Análisis exhaustivo por idea
def analyze_idea_exhaustive(idea_text):
    ...

```

El sistema implementa una estrategia de protección mediante `signal.SIGALRM`, que garantiza que ninguna llamada a la API de OpenAI exceda los 60 segundos. Si se alcanza el tiempo límite, se activa un fallback inmediato, preservando la estabilidad del sistema.

En cuanto al contexto, se inyecta un bloque detallado con información estratégica de Sener (sectores, líneas de innovación, objetivos corporativos), lo que permite orientar la respuesta del modelo hacia criterios específicos de alineación estratégica.

Se incorporan además mecanismos de recuperación ante fallos como:

- Validación estructural del texto de respuesta.
- Eliminación de duplicaciones de encabezados.
- Limpieza y normalización mediante `normalize_text_for_pdf()`.
- Reemplazo automático de contenido corrupto mediante `emergency_clean_text()`.

La salida final se estructura en un PDF profesional, utilizando FPDF con fuentes estándar. El informe incluye portada, índice automatizado, secciones analíticas diferenciadas y estilo corporativo homogéneo.

3.2.4 Sistema de Persistencia Global

Para garantizar la consistencia entre módulos y la trazabilidad de los análisis realizados, el sistema emplea una arquitectura de persistencia sincronizada a nivel global.

Se definen tres variables centrales:

```
analyzed_ideas_global # Lista principal de ideas procesadas  
_last_analyzed_ideas # Caché en memoria  
analysis_points_validated # Resultado validado de puntos analíticos
```

Estas variables se sincronizan entre módulos (gr1.py, analysis_module2.py, ranking_module.py, competitor_analysis.py) mediante funciones auxiliares como get_global_analyzed_ideas() y global_save_analyzed_ideas().

La recuperación de datos es resiliente gracias a un sistema de múltiples capas:

1. **Memoria principal** (analyzed_ideas_global)
2. **Variables de respaldo** (_analyzed_ideas_global, _last_analyzed_ideas)
3. **Archivo temporal JSON** en disco local
4. **Fallback por defecto** (lista vacía controlada)

El archivo temporal (last_analysis_results.json) se genera en cada ejecución relevante y se mantiene sincronizado como respaldo de seguridad.

También se proporciona una API interna de limpieza total (clear_all_global_memory()), que elimina todas las estructuras en memoria y el archivo de resultados en disco, útil para pruebas o reinicialización.

Por último, se implementa un sistema robusto de validación de integridad, que comprueba en tiempo real la validez estructural de cada idea almacenada:

- Presencia de campos obligatorios (idea, analysis)
- Contenido no vacío y tipado correcto
- Compatibilidad UTF-8 para exportación y almacenamiento

Esto asegura que el motor de análisis opere siempre sobre datos limpios, validados y en formato uniforme.

3.3 Sistema de Ranking (ranking_module.py)

El Sistema de Ranking constituye el motor de priorización estratégica del agente. A partir de las ideas analizadas, este módulo transforma cualitativos narrativos en decisiones cuantificadas, mediante un algoritmo híbrido que combina métricas objetivas con evaluaciones subjetivas generadas por modelos de lenguaje.

Su diseño contempla procesamiento paralelo, escalabilidad por volumen y generación de scores normalizados en múltiples dimensiones.

3.3.1 Algoritmo de Ranking Principal

La función generate_ranking() es el punto de entrada principal para el ranking. Permite procesar lotes de ideas en paralelo, extrayendo métricas relevantes, calculando puntuaciones y generando estructuras listas para visualización avanzada.

```
# Líneas 1891–2100
def generate_ranking(ideas_list, ranking_context="", max_workers=10, batch_size=None):
    ...

```

El sistema utiliza ThreadPoolExecutor para distribuir la carga computacional, con ajuste dinámico de número de workers en función del tamaño del lote. Cada idea es procesada de forma independiente, con recuperación de errores en cada hilo.

Durante el procesamiento individual (process_single_idea), se extraen las siguientes variables por entrada:

- score total final
- Dimensión técnica, económica y de mercado
- Estimaciones de esfuerzo y beneficio esperados
- Título y descripción de la idea

Modelo Multicriterio con Evaluación Híbrida

La función calculate_final_score() ejecuta el algoritmo central de puntuación. Está basado en un modelo híbrido 50/50 entre scoring cuantitativo (basado en métricas) y cualitativo (derivado del análisis del modelo OpenAI):

- **Cuantitativo (50%):** se calculan tres dimensiones independientes:
 - Técnica: riesgo técnico, tiempo de desarrollo, progreso TRL
 - Económica: ingresos previstos, payback ROI, ratio costes-ingresos
 - Mercado: tamaño de mercado, riesgo de mercado, alineación estratégica
- **Cualitativo (50%):** puntuación interpretada a partir del análisis textual, ajustada a escala 0–100

La puntuación final se calcula como media ponderada de ambos componentes.

Normalización Avanzada y Precisión Decimal

Las métricas internas (originalmente en escala 1–5) se convierten a una escala continua de 0–100 para facilitar comparabilidad y visualización. Esta transformación permite:

- Escalado homogéneo entre ideas
- Interpretación intuitiva en gráficos y dashboards
- Comparación cruzada entre dimensiones

Además, se incorpora una lógica específica para el mapeo de TRL (Technology Readiness Level) desde su escala 1–9 a un valor ajustado entre 1.0 y 5.0, lo que permite integrarlo en la dimensión técnica sin distorsionar la media ponderada.

3.3.2 Evaluación Cualitativa

El componente cualitativo representa el 50% del score final de cada idea. Esta evaluación se basa en prompts especializados diseñados para extraer, de manera estructurada y objetiva, tanto métricas

cuantitativas normalizadas como un juicio cualitativo global. El sistema emplea llamados separados a la API de OpenAI, utilizando configuraciones de precisión alta y formato de respuesta controlado.

Prompts Especializados por Criterio

La función extract_metrics_from_analysis() genera un prompt técnico dirigido a obtener métricas cuantitativas estructuradas en tres dimensiones: técnica, económica y de mercado. Cada dimensión aporta un 33,3% del componente cuantitativo total.

```
# Líneas 251–530
def extract_metrics_from_analysis(analysis_text, idea_text="", ranking_context ""):
    ...

```

Las métricas extraídas incluyen:

- **Dimensión Técnica:** riesgo técnico, tiempo de desarrollo, TRL inicial y final
- **Dimensión Económica:** ratio costes/ingresos, ingresos previstos, payback ROI
- **Dimensión de Mercado:** tamaño de mercado, riesgo de mercado, alineación estratégica con Sener

Los valores esperados están normalizados en una escala de 1.0 a 5.0, con decimales permitidos para una mayor precisión.

Evaluación Global Cualitativa

La función generate_qualitative_evaluation() produce una puntuación global basada en criterios de innovación, calidad, viabilidad y alineación estratégica. Este valor representa el 50% restante del ranking final.

```
# Líneas 718–850
def generate_qualitative_evaluation(idea_text, analysis_text "", context ""):
    ...

```

El modelo genera una salida JSON con dos campos:

- score: puntuación entre 0 y 100
- justification: justificación textual alineada con estándares de evaluación profesional

Se emplea una temperatura baja (temperature = 0.1) para maximizar la consistencia y minimizar la aleatoriedad del juicio generado.

Análisis de Viabilidad y Riesgo

Durante la evaluación, se aplican controles adicionales:

- Validación automática de rangos por tipo de métrica (1–5 para métricas estándar, 1–9 para niveles TRL)
- Conversión estricta a tipos numéricos con límites inferiores y superiores
- Detección de valores anómalos y fallbacks de recuperación

Estas medidas refuerzan la robustez del sistema de evaluación, evitando puntuaciones fuera de rango o interpretaciones ambiguas en la respuesta del modelo.

3.3.3 Generación de Matriz de Payoff

El sistema incorpora una matriz estratégica de payoff como herramienta visual clave para representar el posicionamiento de cada idea según dos dimensiones fundamentales: esfuerzo requerido y beneficio esperado. Esta matriz facilita la toma de decisiones estratégicas en contextos de portafolio de innovación, alineando la priorización con criterios de impacto y viabilidad.

Visualización Profesional con Matplotlib

La matriz es generada mediante la función `graficar_payoff_matrix()`, que implementa una visualización profesional con segmentación por cuadrantes estratégicos predefinidos:

```
# Líneas 132–185
def graficar_payoff_matrix(lista_ideas):
    ...

```

La visualización incluye:

- Cuadrantes diferenciados: **Quick Win**, **Major Project**, **Improvements**, **Kill It**
- Líneas divisorias en ambos ejes (valor 5)
- Etiquetas estratégicas posicionadas dinámicamente
- Conversión automática de escalas: valores de 0–100 reescalados a 0–10
- Posicionamiento automático de ideas mediante `scatter()` y `annotate()` para evitar solapamientos

La matriz se representa gráficamente como una figura de 8x8 pulgadas con DPI 100, lo que asegura resolución óptima para su inclusión en documentos PDF profesionales.

Cálculo Estratégico de Effort vs Benefit

La función `calculate_payoff_matrix_values()` estima cuantitativamente los valores de esfuerzo y beneficio a partir del análisis individual de cada idea, utilizando un prompt específico diseñado para generar respuestas polarizadas y decisivas:

```
# Líneas 2524–2650
def calculate_payoff_matrix_values(idea_text, analysis_text, metrics, score_data):
    ...

```

- **Effort** evalúa complejidad técnica, inversión requerida, tiempo estimado, y riesgo operativo
- **Benefit** mide impacto estratégico, potencial económico, y alineación con los objetivos de Sener
- Las puntuaciones se expresan en una escala de 0 a 100, con precisión decimal controlada
- Se utilizan valores no lineales para evitar concentraciones artificiales de puntuación media

La API se configura con temperatura baja (`temperature = 0.5`) para favorecer decisiones deterministas y reducir ambigüedad interpretativa.

Integración Directa en PDF Final

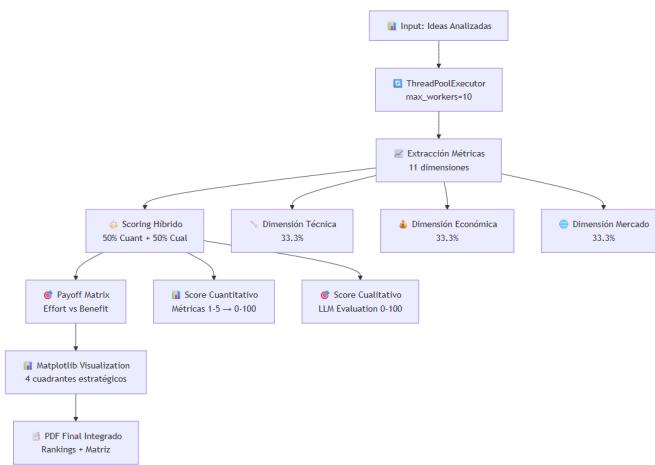
La matriz generada se integra automáticamente en el informe PDF a través de la función `generate_ranking_pdf_improved()`:

```
def generate_ranking_pdf_improved(ideas, ranking_context):
```

...

- La imagen se inserta en una nueva página dedicada dentro del documento
- Se incluye un título centrado: "**Matriz de Payoff – Esfuerzo vs Beneficio**"
- La ruta de la imagen se gestiona dinámicamente con control de errores para garantizar su disponibilidad
- Se emplean fuentes estándar (DejaVu) y resolución optimizada para impresión y lectura en pantalla

La [siguiente imagen](#) muestra la matriz generada con los cuadrantes estratégicos definidos y las ideas posicionadas según su evaluación:



3.4 Análisis Competitivo (competitor_analysis_module.py)

El módulo de análisis competitivo introduce una arquitectura híbrida de última generación que combina el poder del lenguaje natural con scraping web automatizado. Su objetivo es proporcionar una visión estratégica robusta del entorno competitivo de cada idea, con respaldo en datos actuales, fuentes abiertas y contexto técnico.

3.4.1 Sistema Híbrido LLM + Web Scraping

Arquitectura técnica avanzada para análisis competitivo

El módulo de análisis competitivo implementa una arquitectura híbrida de nueva generación, diseñada para combinar las capacidades de generación de lenguaje natural de modelos LLM con la extracción dirigida de datos reales desde la web. Este enfoque adopta un paradigma **LLM-first**, donde el modelo de lenguaje genera un análisis completo en primera instancia, y solo en caso necesario, se activa el scraping automatizado de fuentes externas para validar, completar o enriquecer los contenidos.

Flujo arquitectónico híbrido

El procesamiento sigue una secuencia estructurada en cuatro fases:

LLM_GENERATION → URL_DETECTION → TARGETED_SCRAPING → INTELLIGENT_MERGE

1. **LLM_GENERATION:** Generación inicial del informe completo utilizando el modelo GPT configurado con contexto empresarial.
2. **URL_DETECTION:** Análisis posterior de la salida del modelo para identificar automáticamente referencias web relevantes.
3. **TARGETED_SCRAPING:** Activación de scraping específico si el contenido requiere respaldo o validación factual.
4. **INTELLIGENT_MERGE:** Fusión selectiva de los datos extraídos con el análisis original, priorizando coherencia, validez y completitud.

Este diseño reduce la carga innecesaria sobre los servicios de scraping, mejora la eficiencia general y permite mayor control sobre la calidad del contenido final.

Configuración dual de conectividad

El sistema integra de forma simultánea dos modos de conexión:

- **Cliente directo Azure OpenAI:** Utilizado para llamadas de bajo nivel, configurado con límites de tolerancia ajustados.
- **LangChain wrapper:** Habilita funcionalidades avanzadas como gestión de reintentos, cache semántica y procesamiento de herramientas auxiliares.

```
def __init__(self):
    self.openai_client = get_openai_client()
    self.deployment_name = get_deployment_name()

    self.llm = AzureChatOpenAI(
        deployment_name=self.deployment_name,
        azure_endpoint=AZURE_OPENAI_ENDPOINT,
        api_version=API_VERSION,
        temperature=0.7,
        max_retries=3
    )
    self.completion_llm = self.llm
```

Generación inteligente de queries con cache

Para evitar consultas redundantes y optimizar el rendimiento, se ha implementado un sistema de cache LRU con capacidad para almacenar hasta 128 ideas únicas. Este mecanismo reutiliza consultas previas generadas por el LLM, lo cual reduce tiempos de espera y llamadas innecesarias.

Además, el sistema cuenta con un extractor híbrido de términos clave sectoriales basado en:

- Análisis semántico por LLM
- Algoritmo de extracción léxica estructurada como fallback
- Cache por idea para evitar recomputación

```
@functools.lru_cache(maxsize=128)
def llm_short_queries(self, idea: str, k: int = 6) -> list[str]:
    ...
```

Integración con Tavily API para scraping dirigido

La activación del scraping está controlada mediante condiciones explícitas que aseguran que solo se ejecuta cuando es necesario:

- Detección de URLs explícitas en la salida del modelo
- Solicitud directa del modelo para validar una afirmación
- Reglas de validación que identifican datos posiblemente inventados

Para ejecutar el scraping, el sistema busca recursivamente en estructuras complejas de datos todas las URLs potenciales, usando validadores sintácticos antes de iniciar cualquier proceso de extracción.

```
def find_urls(obj):
```

```
    ...
```

Validación y fusión de información

Una vez obtenida la información desde fuentes externas, esta se somete a una validación de coherencia antes de ser integrada:

- Se comprueba que los datos no contradigan el análisis original.
- Se evalúa la consistencia entre competidores, tecnologías y métricas.
- Si las fuentes externas son parciales o poco fiables, se activa una estructura por defecto para garantizar continuidad del análisis.

El sistema también implementa un mecanismo de fusión selectiva que prioriza información factual sobre el contenido generado automáticamente, siempre que se cumplan criterios de integridad.

3.4.2 Arquitectura Modular del Análisis Estratégico

El módulo competitor_analysis_module.py implementa un sistema de análisis competitivo orientado a entornos industriales, integrando generación LLM, recuperación de información externa y validación estructural. La arquitectura se estructura en **ocho secciones funcionales desacopladas** que responden a una lógica de diseño **modular-validada**, permitiendo independencia semántica, recomposición dinámica y auditoría granular de cada bloque.

Especificación de Secciones Estratégicas

Cada sección se define mediante un contrato lógico (purpose, output, validation) que establece su finalidad analítica, salida estructurada esperada y mecanismos automáticos de validación:

```
STRATEGIC_SECTIONS = {
    "COMPETITOR_MAPPING": {
        "purpose": "Mapeo competitivo 360°",
        "output": "Competidores directos, indirectos, emergentes",
        "validation": "Exclusión de Sener + coherencia sectorial"
    },
    ...
}
```

Las secciones se procesan de forma secuencial bajo una lógica controlada por generate_ai_only_competition_report() con validación unitaria por bloque y fallback automático en caso de fallo semántico, timeout o malformación JSON.

Validación Anti-Alucinación por Sección

Cada sección está protegida mediante **funciones de validación especializadas** que inspeccionan los datos generados por el LLM. Estas funciones utilizan expresiones regulares, patrones semánticos, longitud mínima esperada, y lógica de exclusión de entidades ficticias. Ejemplos concretos:

- _looks_like_fake_patent_number() → detecta secuencias sintéticas tipo EP1234567
- _looks_like_fake_doi() → identifica DOIs malformados o inconsistentes
- _looks_like_fake_journal() → valida contra listas negras de revistas inventadas

Esta validación estructural se ejecuta como etapa posterior al parsing y antes de persistencia o visualización, garantizando consistencia y trazabilidad.

3.4.3 Infraestructura de Generación de Informes y Recuperación

Mecanismo de Parsing Robusto para JSON

La función _parse_json_with_fallback() implementa un **algoritmo de recuperación de seis capas** diseñado para tolerar errores comunes en salidas generadas por LLMs:

1. Parsing directo con json.loads()
2. Normalización de comillas, comas y secuencias no escapadas
3. Extracción aislada del bloque JSON válido en el cuerpo textual
4. Completado automático mediante análisis de llaves de apertura y cierre
5. Parsers específicos para secciones críticas como TECH_IP_LANDSCAPE
6. Fallback a estructura por defecto vía **Factory Pattern**

Este mecanismo reduce drásticamente la probabilidad de fallo total del sistema y permite que el informe se complete incluso ante salidas parciales o corruptas.

Timeout Global Controlado

El análisis de cada idea está limitado por un **timeout global de 300 segundos** mediante signal.alarm(). En caso de exceder ese umbral, el sistema detona un TimeoutError que es manejado de forma segura mediante _complete_with_defaults(), completando las secciones faltantes con estructuras válidas y coherentes.

```
signal.signal(signal.SIGALRM, timeout_handler)
signal.alarm(300) # 5 minutos por informe
```

Este control garantiza consistencia en entornos de despliegue asincrónicos con limitaciones de recursos o latencia.

Procesamiento Paralelo con Control de Recursos

La función `analyze_ideas_batch_competitor()` orquesta la ejecución de análisis múltiples mediante `ThreadPoolExecutor`. El sistema impone un límite estricto de **4 hilos concurrentes** para cumplir con los límites de rate de Azure OpenAI y mantener estabilidad en ejecución batch.

La arquitectura separa la fase de procesamiento paralelo (por idea) de la fase secuencial (generación del executive summary global), permitiendo que los informes parciales se completen en paralelo pero se integren de forma coherente.

Patrones Arquitectónicos Aplicados

- **Chain of Responsibility:** Validación encadenada de etapas (`JSON_PARSING` → `DATA_VALIDATION` → `COHERENCE_CHECK` → `FALLBACK`)
- **Factory Pattern:** Generación desacoplada de estructuras por sección, mediante métodos como `_create_tech_landscape_default()`
- **Strategy Pattern:** Elección dinámica del método de extracción: `llm_direct`, `regex_fallback`, `structured_parsing`, `default_generation`
- **LRU Cache:** Implementación de `functools.lru_cache` para queries LLM reutilizables y deterministas
- **Smart Timeout:** Control temporal global con recuperación segmentada

Métricas de Observabilidad y Robustez

El sistema incorpora un subsistema de logging granular con cinco niveles:

```
LOGGING_LEVELS = {
    "DEBUG": "Parsing detallado línea a línea",
    "INFO": "Progreso de secciones y threads",
    "WARNING": "Alerta por datos posiblemente inventados",
    "ERROR": "Errores de conectividad, parsing o timeout",
    "CRITICAL": "Fallos estructurales irreversibles"
}
```

Estas métricas permiten auditar cada fase del análisis competitivo con trazabilidad completa, facilitando debugging, validación y mejora continua del sistema.

4. MÓDULOS DE SOPORTE

4.1 Procesamiento de Documentos – Módulos de Entrada

4.1.1 Motor de Procesamiento PDF (pdf_processor_module.py)

El motor de procesamiento de documentos PDF constituye la primera capa de entrada del sistema. Su diseño sigue una arquitectura de detección en cascada, robusta frente a fallos, y orientada a preservar tanto la semántica estructural como la calidad de extracción. El objetivo es transformar texto no estructurado en ideas procesables, listas para análisis estratégico.

Arquitectura Multinivel de Extracción

La detección de ideas en PDF sigue una jerarquía de tres niveles, basada en estrategias sucesivas con fallback automático:

- **Nivel 1: Extracción textual directa**

Método: extract_text_from_pdf()

Utiliza PyPDF2 para extraer texto crudo. Si falla o retorna texto incompleto, se activa el siguiente nivel.

- **Nivel 2: Detección estructurada por patrones**

Método: detect_ideas_basic()

Aplica reglas como:

- Reconocimiento de ideas numeradas (e.g., "Idea 1:", "1.", etc.)
- Segmentación de párrafos lógicos
- Eliminación de duplicidades y ruido estructural

Se activa si la extracción básica produce texto plano insuficiente.

- **Nivel 3: Extracción híbrida con LLM**

Método: CompetitorAnalysis.process_pdf_ideas()

Se invoca un modelo LLM contextualizado para identificar, interpretar y reestructurar ideas. Esta capa aplica cuando el contenido es denso, técnico o sin formato explícito.

Flujo de Procesamiento Principal

La función process_pdf_file() implementa el pipeline principal, priorizando el método optimizado con fallback al tradicional:

```
# Secuencia de decisión:  
try:  
    ideas = analyzer.process_pdf_ideas(text, context_text) # Optimizado (LLM + reglas)  
except:  
    ideas = detect_ideas_basic(text) # Fallback estructurado
```

Si se detectan ideas estructuradas correctamente, el sistema las convierte al formato estándar requerido (idea, analysis, metrics) para su posterior validación.

Mejora Semántica con LLM

Cada idea puede ser enriquecida individualmente mediante enhance_idea_with_ai(), que aplica un LLM especializado para:

- Reescritura y estructuración avanzada
- Identificación de puntos clave
- Evaluación preliminar de potencial innovador

4.1.2 Pipeline de Validación de Documentos

El sistema aplica un proceso de validación exhaustivo, con tres niveles progresivos que aseguran la integridad estructural, semántica y lógica del contenido extraído.

Nivel 1 – Validación de Formato

Los métodos `validate_idea_format()` y `validate_ideas_list()` comprueban que cada idea cumpla la estructura esperada:

```
{
  "idea": "...",
  "analysis": "...",
  "metrics": {...}
}
```

Las entradas no conformes son automáticamente descartadas o transformadas.

Nivel 2 – Validación Semántica

El método `clean_title_semantics()` detecta y corrige patrones comunes de ruido como:

- Duplicación de prefijos: Idea 1: Idea 1:
- Prefijos redundantes: Idea 2: ...
- Títulos vacíos o carentes de contenido alfanumérico

Solo las ideas con títulos limpios y significativos avanzan a fases posteriores.

Nivel 3 – Validación Global de Estado

El sistema global de validación (`set_ideas_list_safe()` y `get_analyzed_ideas_global()`) actúa como firewall de integridad. Solo las listas de ideas válidas y completas se incorporan al flujo de análisis. Todo objeto inválido es filtrado, y se generan logs explícitos de rechazo o aceptación.

- **Normalización y Preservación de Estructura**
- **Reconstrucción Estructural**

El método `preserve_internal_structure()` asegura que cada idea conserve su organización en título + cuerpo. En caso de estructura colapsada, intenta reconstruir el formato mínimo requerido, especialmente útil en documentos mal segmentados.

- **Limpieza de Texto y Encoding**

El sistema aplica una capa de normalización para compatibilidad con FPDF y sistemas downstream:

- Sustitución de caracteres Unicode problemáticos (\u2019, \u201c, etc.)

- Eliminación de caracteres de control
- Conversión segura de texto para su visualización o exportación
- **Métricas de Validación y Observabilidad**

El sistema registra métricas cuantitativas y cualitativas de validación, incluyendo:

- Total de ideas procesadas, validadas, rechazadas
- Número de títulos corregidos
- Número de ideas con estructura reconstruida
- Logs de errores y razones de rechazo

```
VALIDATION_METRICS = {
    "ideas_processed": 83,
    "ideas_validated": 71,
    "ideas_rejected": 12,
    "titles_cleaned": 8,
    "structure_preserved": 64
}
```

Este subsistema facilita auditoría, debugging y mejora iterativa del proceso de ingesta documental.

4.2 Generación de PDFs – Módulos de Salida

El sistema de generación de documentos PDF constituye el núcleo de salida estructurada del AI/*Innovation Agent*. Este subsistema está diseñado con una arquitectura multicapa y tolerante a fallos, permitiendo la creación de entregables profesionales en distintos formatos (análisis, rankings, competencia, retos), integrando visualizaciones, estructuras dinámicas, y validación semántica del contenido.

4.2.1 Motor PDF Central (`pdf_generator.py`)

El motor principal se basa en la librería **FPDF**, extendida con funciones propias para:

- **Normalización textual avanzada:** más de 45 caracteres especiales mapeados a equivalentes compatibles con el sistema de renderizado.
- **Estructura documental estándar:** portada, índice, cuerpo, referencias.
- **Compatibilidad de fuentes:** sistema de fallback automático si no se encuentra la tipografía base.

Componentes arquitectónicos:

```
PDF_ENGINE_CORE = {
    "Base Library": "FPDF (ligero, extensible)",
    "normalize_text()": "Limpieza semántica y de símbolos no imprimibles",
    "Font System": "Arial básica con fallback a DejaVu o Courier",
    "Document Flow": ["Portada", "Índice", "Contenido", "Referencias"]
}
```

La función `normalize_text()` implementa sustituciones específicas para símbolos monetarios, guiones, comillas, acentos no latinos y caracteres Unicode no compatibles con FPDF.

4.2.2 Generadores Especializados

A. generate_unified_pdf() – Módulo de Análisis Detallado

Este generador implementa un motor avanzado con herencia sobre FPDF (CustomPDF), simulación de renderizado para TOC realista y detección automática de secciones clave en el análisis.

Características destacadas:

- Detección y segmentación automática de secciones semánticas (hasta 6 por idea)
- Renderizado iterativo con cálculo de espacio disponible por página
- Índice dinámico con enlaces clicables
- Integración gráfica de logotipos y cabeceras corporativas

```
UNIFIED_PDF_ARCHITECTURE = {
    "PDF Class": "CustomPDF",
    "TOC Engine": "generate_toc_with_simulation()",  

    "Font Stack": ["DejaVu Unicode", "Arial", "Sans"],
    "Section Parsing": "process_analysis_text_improved()",  

    "Recovery Mechanisms": "Fallback textual por sección"
}
```

Secciones reconocidas automáticamente:

- Resumen Ejecutivo
- Análisis Técnico
- Potencial de Innovación
- Alineación Estratégica
- Viabilidad Comercial
- Valoración Global

B. generate_ranking_pdf_improved() – Visualización de Rankings

Este generador está optimizado para visualización comparativa entre ideas, integrando procesamiento de métricas cuantitativas, visualización estratégica (matriz de payoff) y ordenamiento dinámico.

Elementos funcionales:

- Matriz de payoff (esfuerzo vs beneficio) con visualización cuadrante en matplotlib
- Tablas con ideas ordenadas por puntuación total (score)
- Representación de métricas numéricas y justificación cualitativa
- Inserción de gráficos vectoriales y contenido justificado para PDF

```
RANKING_PDF_FEATURES = {
    "Visual Output": "Matriz + Tabla ordenada",
    "Metrics Model": "6 dimensiones + peso ponderado",
    "Sorting Engine": "Orden descendente por score",
    "PDF Layout": "Gráfico superior + tabla resumen"
}
```

Se integra además un sistema robusto de recuperación de métricas en caso de contenido malformado (fallback de JSON).

C.generate_competition_analysis_pdf() – Análisis Competitivo

El motor más sofisticado, con más de 4.600 líneas de código dedicadas, combina generación LLM, scraping web y visualización avanzada en un flujo completamente estructurado de inteligencia competitiva.

Componentes estructurales:

```
COMPETITION_PDF_ARCHITECTURE = {  
    "Rendering Stack": ["FPDF", "ReportLab", "matplotlib", "PIL"],  
    "Font System": "DejaVu Unicode (auto-descargable)",  
    "Content Model": "8 secciones competitivas especializadas",  
    "Timeout Manager": "Tiempo máximo 300s + fallback por sección",  
    "Recovery Engine": "Estrategias de fallback por tipología de error"  
}
```

Las secciones generadas incluyen:

- Competitor Mapping
- Benchmark Matrix
- Landscape de Patentes y Publicaciones
- Análisis de Mercado
- DAFO Competitivo (SWOT)
- ESG y Riesgo Regulatorio
- Roadmap Estratégico
- Resumen Ejecutivo de Decisión

Cada sección se valida individualmente y puede recuperarse mediante estructura por defecto si se detecta inconsistencia.

D.generate_challenges_and_solutions_pdf() – Retos y Soluciones

Generador especializado para documentos estructurados en dos fases: extracción de retos + generación de soluciones vía LLM. Implementa procesamiento paralelo y validación estructural para documentos derivados de sesiones colectivas o inputs múltiples.

Diseño técnico:

```
CHALLENGE_PDF_SYSTEM = {  
    "Parallel LLM": "ThreadPoolExecutor",  
    "Extraction": "parse_retos() + parse_soluciones()",  
    "LLM Fase 1": "Identificación de problemas estructurales",  
    "LLM Fase 2": "Sugerencia automatizada de soluciones",  
    "Timeout Control": "60s por fase de análisis"  
}
```

Estructura típica del documento:

- Portada con branding
- Lista de retos organizados temáticamente
- Soluciones propuestas con justificación estratégica
- Referencias opcionales o notas de contexto

4.2.3 Subsistema de Mejora de PDFs (PDF Enhancement System)

El subsistema de mejora de PDFs constituye una capa crítica dentro del pipeline de salida del AI Innovation Agent. Su objetivo principal es garantizar que los documentos generados sean precisos, estéticamente consistentes y resilientes frente a errores estructurales comunes, como índices incorrectos, contenido malformado o tipografías no disponibles.

A. Corrección de Índices: Simulación Precisa de Paginado

Problema identificado:

La numeración del índice de contenidos no coincidía con las páginas reales debido a inconsistencias en la simulación de espacio durante el renderizado.

Solución técnica:

Se ha implementado un **algoritmo de simulación de espacio vertical**, que replica fielmente el comportamiento del motor de renderizado real. Este mecanismo evalúa el espacio restante en cada página en milímetros, y fuerza un salto de página cuando la altura útil disponible es inferior a un umbral crítico (50 mm).

Parámetros del modelo de simulación:

- Altura útil por página: 252 mm (A4 con márgenes).
- Espacio consumido por idea: 185 mm (título, secciones, espaciado).
- Umbral de cambio de página: 50 mm residuales.

```
def simulate_page_calculation():
    y_position_sim = 252 # mm disponibles por página A4
    current_page = start_page

    for idea in ideas:
        if y_position_sim < 50:
            current_page += 1
            y_position_sim = 252

        toc_entries.append((idea_title, link_id, current_page))
        y_position_sim -= 185
```

Este sistema garantiza que el índice refleje exactamente la distribución final del documento generado.

B. Recuperación ante JSON Malformado: Sistema en Cascada (6 Estrategias)

El sistema de generación se enfrenta con frecuencia a contenido semiestructurado o malformado debido a errores de decodificación desde modelos LLM. Para abordarlo, se implementa una **estrategia multicapa de parsing robusto** con fallback progresivo:

Estrategias de recuperación implementadas:

1. **Carga directa (json.loads)** – Válido en el caso ideal.
2. **Sanitización** – Corrección de comillas, eliminación de trailing commas.
3. **Extracción parcial { ... }** – Aísla el bloque JSON principal.
4. **Reconstrucción automática** – Completado sintáctico de llaves faltantes.

5. **Extracción semántica especializada** – Por sección (ej. TECH_IP_LANDSCAPE).
6. **Estructura por defecto** – Generación basada en plantillas por tipo de contenido.

Cada estrategia se ejecuta secuencialmente hasta obtener una estructura JSON válida.

C. Gestión de Timeout Multinivel y Recuperación Graceful

El sistema establece límites temporales diferenciados para cada capa del proceso, con mecanismos de recuperación garantizada en caso de sobrepasar dichos umbrales:

Nivel de Operación	Límite Temporal	Módulo Responsable
Análisis Competitivo	300 s	competition_pdf_module.py
Llamadas a OpenAI	60 s	analysis_module2.py
Generación de PDFs	120 s	pdf_generator.py
Scraping Web / Fuentes	15 s	tavily_config.py
Parsing de JSON	5 s	competitor_analysis_module.py

Ejemplo de control con signal.alarm:

```
def analyze_idea_exhaustive(idea_text):
    signal.signal(signal.SIGALRM, timeout_handler)
    signal.alarm(60) # Timeout LLM

    try:
        response = client.chat.completions.create(...)
        signal.alarm(0)
        return process_response(response)
    except TimeoutError:
        signal.alarm(0)
        return generate_fallback_content()
```

Este enfoque asegura una degradación controlada del sistema ante situaciones de latencia o fallos externos.

- **D. Renderizado Avanzado de Visualizaciones**

El subsistema de gráficos emplea **matplotlib + PIL** para generar visualizaciones estratégicas integradas directamente en los documentos PDF:

1. Matriz de Payoff:

- Generación de scatter plots con ejes configurables (esfuerzo vs beneficio).
- Posicionamiento inteligente de etiquetas mediante offset automático.
- Segmentación por cuadrantes (Quick Wins, Major Projects, etc.).

2. Visualización DAFO:

- Renderizado visual con FancyBboxPatch para bloques coloridos.
- Distribución equilibrada de fortalezas, debilidades, amenazas y oportunidades.
- Fallback automático a tabla textual en caso de error de renderizado gráfico.

E. Sistema de Fuentes Resistente

Para asegurar compatibilidad con caracteres internacionales y estabilidad en renderizado, el sistema implementa un gestor robusto de fuentes:

- **Fuentes base:** DejaVu Sans, Arial.
- **Descarga automática:** Desde CDN si no están disponibles en local.
- **Fallback de sistema:** Activación de fuentes embebidas si falla descarga.

```
def ensure_dejavu_fonts():
    font_files = ['DejaVuSans.ttf', 'DejaVuSans-Bold.ttf']
    for font in font_files:
        if not download_font_with_retry(font):
            FONT_OK = False # Fallback activado
    return FONT_OK
```

Este componente garantiza la continuidad del proceso incluso en entornos de ejecución restringidos.

4.3 Configuración y Conectividad – Módulos de Infraestructura

El sistema de infraestructura del *AI Innovation Agent* ha sido diseñado con principios de seguridad, resiliencia y escalabilidad. La arquitectura se basa en un enfoque modular y desacoplado que permite la gestión segura de claves, el fallback automático de servicios externos y la supervisión activa del estado del sistema. A continuación, se describen los principales componentes.

4.3.1 Configuración Azure OpenAI – openai_config.py

Arquitectura de seguridad multinivel

El módulo define un modelo de configuración segura para entornos empresariales, segmentado en tres niveles funcionales:

- **Nivel 1: Variables de entorno obligatorias**
Configuración sensible como AZURE_OPENAI_API_KEY, AZURE_OPENAI_ENDPOINT, y DEPLOYMENT_NAME son extraídas del entorno.
Fallbacks seguros garantizan compatibilidad si algunas variables no se declaran explícitamente.
- **Nivel 2: Validación estricta**
Se implementan verificaciones de integridad antes de inicializar el cliente.
Los logs exponen solo partes no sensibles de las credenciales (inicio y fin).
- **Nivel 3: Cliente de acceso empresarial**
Cliente Azure configurado con max_retries=3, conexión segura y timeout gestionado a nivel SDK.

```
client = AzureOpenAI(
    api_key=AZURE_OPENAI_API_KEY,
    api_version=API_VERSION,
    azure_endpoint=AZURE_OPENAI_ENDPOINT,
    azure_deployment=DEPLOYMENT_NAME,
    max_retries=3
)
```

Se adopta el patrón Singleton para la reutilización del cliente en todos los módulos del sistema.

4.3.2 Integración de APIs Externas – tavily_config.py

Sistema de configuración híbrido con fallback inteligente

El módulo Tavily opera con una lógica de prioridad Env → Hardcoded → Exception, permitiendo ejecución en entornos no productivos sin intervención adicional.

- La función get_tavily_api_key() recupera la clave de forma segura y auditible.
- En caso de fallback, se lanza un RuntimeError controlado para impedir uso sin autorización.

```
def get_tavily_api_key():
    env_key = os.environ.get("TAVILY_API_KEY")
    if env_key:
        return env_key
    if not TAVILY_API_KEY:
        raise RuntimeError("API key no válida configurada")
    return TAVILY_API_KEY
```

4.3.3 Utilidades de Infraestructura

A. LLM Utils (llm_utils.py)

Funciones auxiliares de interacción con LLM para tareas como extracción de *keywords* o análisis de texto corto.

Características:

- Reutilización del cliente Azure centralizado.
- Soporte multiidioma (es/en).
- Temperatura 0 y max_tokens=20 para generar salidas estables y concisas.

B. Generación Avanzada de Queries (query_generator.py)

Arquitectura basada en NLP con spaCy y scikit-learn para construir queries semánticas optimizadas:

- POS tagging para extracción de keyphrases.
- Detección automática de dominios especializados (e.g., términos médicos).
- Sistema de *domain overrides* para keywords técnicas como antifouling.

4.3.4 Sistema de Health Checks y Tolerancia a Fallos

El sistema aplica monitoreo activo sobre componentes críticos del proceso de generación, en particular:

A. Descarga y Validación de Fuentes

El módulo ensure_dejavu_fonts() implementa validación por tamaño (>100KB), estrategia *retry-exponential* y fallback a Arial en caso de fallo múltiple.

```
if os.path.exists(fpath) and os.path.getsize(fpath) > 100000:
    logging.info(f"Fuente válida: {fname}")
```

B. Circuit Breakers y Auto-healing

El patrón *circuit breaker* encapsula todas las operaciones críticas (descargas, inicializaciones de API, scraping, etc.), permitiendo continuidad operativa mediante fallback o ejecución degradada.

C. Timeout Management

Componente	Timeout
Análisis competitivo	300 s
API LLM individual	60 s
Generación PDF	120 s
Descarga de fuentes	15 s
Parsing de JSON	5 s

4.3.5 Infraestructura como Código – Dockerfile + Deploy

Imagen base: python:3.11-slim

Entorno seguro: configuración por variables de entorno obligatorias y ejecución sin privilegios (usuario no root).

Configuración del contenedor:

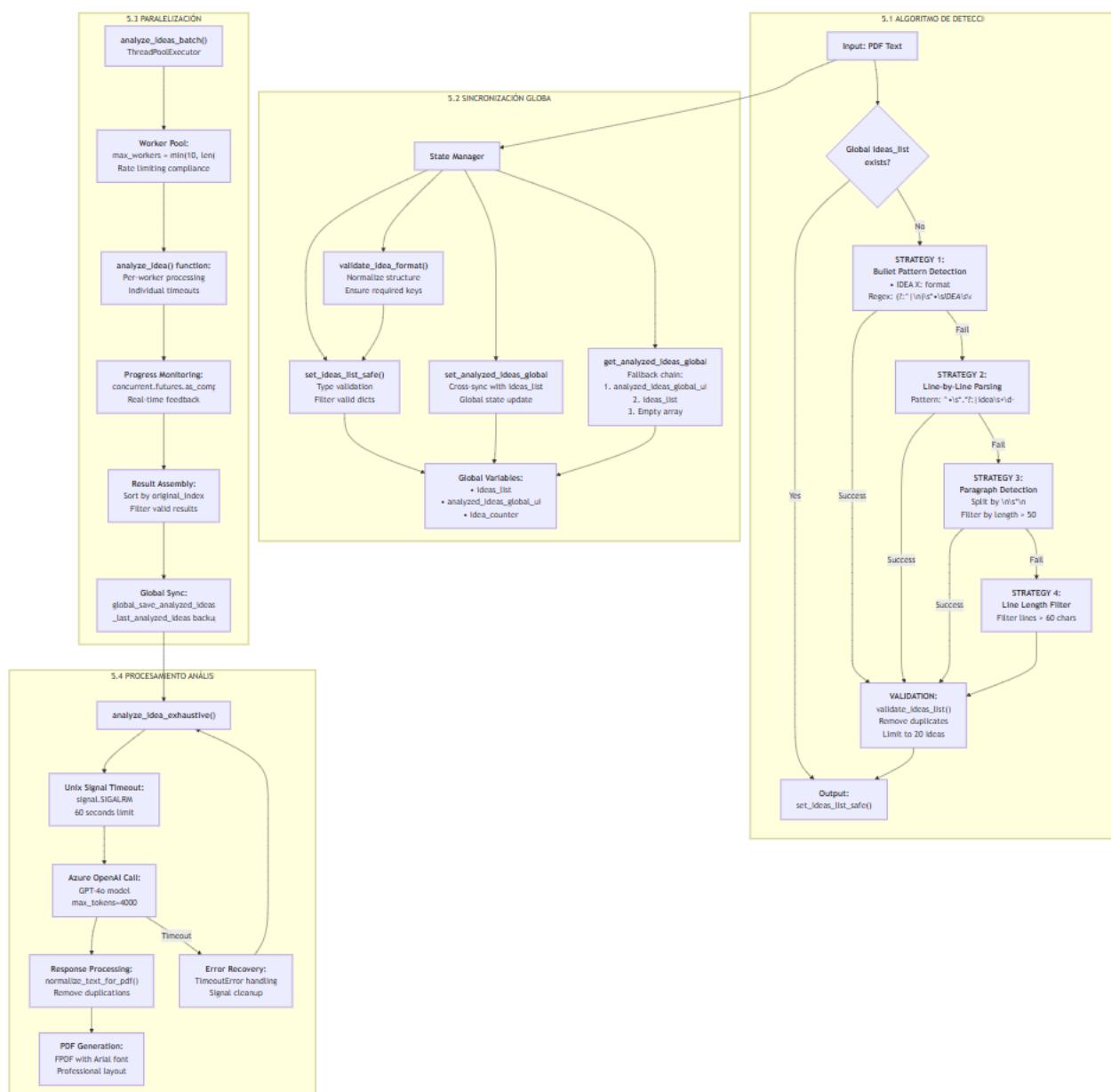
```
docker run -d \
--name ai-innovation-prod \
-e AZURE_OPENAI_ENDPOINT="..." \
-e AZURE_OPENAI_API_KEY="****" \
-e DEPLOYMENT_NAME="gpt-o3-mini" \
-p 7860:7860 \
--restart unless-stopped \
ai-innovation-sener
```

Healthcheck integrado con estrategia de inicio retardado, reintentos automáticos y monitoreo de servicio local.

5. ALGORITMOS CRÍTICOS – DEEP DIVE

Los algoritmos centrales del **AI Innovation Agent** constituyen el núcleo computacional de su arquitectura avanzada. Esta sección detalla los mecanismos internos que permiten la transformación de datos textuales no estructurados en información analizada, sincronizada y lista para explotación estratégica. El sistema combina técnicas de parsing heurístico, sincronización global de estado, paralelización concurrente y procesamiento asistido por LLM, todo integrado bajo un esquema de control resiliente.

A continuación, se presenta el [diagrama estructural](#) del sistema algorítmico, donde se visualiza el flujo secuencial y paralelo de procesamiento desde la entrada inicial hasta la generación de resultados analizados y renderizados:



5.1 Detección Multicapa de Ideas

El algoritmo de detección opera como punto de entrada del sistema. Implementa una arquitectura escalonada de parsing que permite extraer ideas desde texto libre, adaptándose a múltiples formatos de entrada. La función principal (`detect_ideas_basic`) aplica una secuencia de estrategias con fallback progresivo:

- **Nivel 0 – Caché global:** Utiliza `ideas_list` si contiene datos previamente analizados.
- **Nivel 1 – Detección por patrón bullet:** Identifica bloques de texto con formato • IDEA X: mediante expresiones regulares.
- **Nivel 2 – Análisis línea a línea:** Procesamiento individual con patrones múltiples y máquina de estados para seguimiento contextual.
- **Nivel 3 – Parsing por párrafo estructurado:** División por saltos dobles y filtrado por heurísticas de longitud.
- **Nivel 4 – Filtro por longitud mínima:** Extracción de líneas con contenido denso como fallback final.

Cada bloque identificado se valida estructuralmente mediante `validate_ideas_list()`, eliminando duplicados, entradas inválidas o ruidosas, y normalizando hasta un máximo de 20 ideas.

5.2 Sincronización de Estado Distribuido

Para garantizar la coherencia entre los distintos componentes del agente (UI, backend, motores de análisis), el sistema implementa una arquitectura de sincronización global basada en almacenamiento compartido y funciones de asignación segura:

- `set_ideas_list_safe()`: Valida el tipo y esquema de entrada, y asigna ideas válidas de forma atómica.
- `set_analyzed_ideas_global()`: Realiza la sincronización cruzada entre UI y núcleo, actualizando ambos espacios de estado.
- `get_analyzed_ideas_global()`: Recupera ideas procesadas aplicando una cadena de fallback jerárquico:
 1. `analyzed_ideas_global_ui` (con análisis completo),
 2. `ideas_list` (con análisis parcial),
 3. lista vacía (fallback seguro).

Se emplea una arquitectura estructurada de variables globales (`ideas_list`, `analyzed_ideas_global_ui`, `idea_counter`), junto con utilidades complementarias para validación, recuperación y limpieza de estado.

5.3 Sistema de Paralelización

El sistema de análisis de ideas del *AI Innovation Agent* incorpora una arquitectura de ejecución concurrente basada en `ThreadPoolExecutor`, diseñada para maximizar el rendimiento sin comprometer la estabilidad ni exceder las políticas de rate limiting de los servicios LLM externos.

Arquitectura de Ejecución Paralela

El núcleo del sistema se encuentra en la función `analyze_ideas_batch()`, que gestiona dinámicamente la cantidad de hilos en función del número de ideas a procesar (`max_workers = min(10, len(ideas))`). Cada hilo

ejecuta una llamada a LLM de forma independiente, con su propio contexto y lógica de recuperación de errores. Esto permite aislar fallos sin afectar la ejecución global.

- **Balance de carga dinámico:** Ajuste automático del número de hilos.
- **Tolerancia a errores por hilo:** Aislamiento de excepciones sin propagación.
- **Cumplimiento de límites externos:** Control de velocidad adaptado a Azure OpenAI y Tavily.

Monitoreo de Progreso y Ensamblaje de Resultados

Cada futuro (Future) se supervisa en tiempo real mediante concurrent.futures.as_completed, lo que permite generar indicadores de progreso precisos durante la ejecución.

Finalizado el procesamiento:

- Los resultados válidos se ordenan según su índice original.
- Se almacenan globalmente mediante global_save_analyzed_ideas().
- Se actualiza _last_analyzed_ideas como respaldo persistente.

Configuración Modular por Componente

Módulo	max_workers	Rate Limit	Timeout	Estrategia de errores
analysis_module2.py	min(10, len(x))	Azure: 600 RPM	Interno a LLM	Aislamiento por hilo
competitor_analysis_module	min(6, len(x))	Azure + Tavily	300s global	Degrado controlada
ranking_module.py	min(10, len(x))	Azure compartido	60s por llamada	Resultados parciales + retry

5.4 Algoritmo de Procesamiento de Análisis

La función analyze_idea_exhaustive() constituye el motor de análisis individual más robusto del sistema. Está diseñada para ejecutar llamadas a modelos GPT con garantías estrictas de timeout, normalización, y generación inmediata del contenido PDF.

Pipeline de Procesamiento Resiliente

La función implementa protección a nivel de sistema operativo utilizando señales Unix (signal.SIGALRM) para limitar la duración máxima de la operación a 60 segundos. En caso de superarse este límite, se genera una excepción capturada de forma controlada.

El flujo general incluye:

1. **Configuración de timeout por señal Unix (SIGALRM)**
2. **Llamada al modelo GPT-4o con parámetros especializados**
3. **Procesamiento del texto de retorno con limpieza avanzada**
4. **Normalización y eliminación de secciones duplicadas**
5. **Renderizado profesional en PDF con estructura estándar**

Generación de PDFs Profesionales

Se utiliza una plantilla predefinida que incluye portada, análisis estructurado y logotipos corporativos. El texto es normalizado para compatibilidad con FPDF, y se eliminan posibles duplicaciones de encabezados de sección mediante expresiones regulares específicas.

Estrategias de Recuperación ante Errores

El sistema incorpora una arquitectura de fallback multicapas, adaptada al tipo de fallo detectado:

Tipo de Error	Estrategia de Recuperación	Logging
TimeoutError	signal.alarm(0) + retorno nulo	Timestamp + duración del evento
APIError	Limpieza de señal + logging estructurado	Captura completa del traceback
PDFGenerationError	Fallback a salida textual sin PDF	Diagnóstico de disponibilidad de fuentes
NormalizationError	Fallback a emergency_clean_text() con modo ASCII-only	Análisis de codificación

6. CONFIGURACIÓN Y DEPLOYMENT

6.1 Configuración del Entorno

La configuración del entorno del **AI Innovation Agent** exige una parametrización rigurosa de variables críticas, integraciones externas seguras, estructura de contenedores y compatibilidad con arquitecturas corporativas de despliegue. Esta sección describe los elementos clave necesarios para garantizar una ejecución estable y segura.

6.1.1 Variables de Entorno Críticas

Azure OpenAI – Configuración Base

El sistema requiere cuatro variables para conectarse a los servicios de Azure OpenAI. Dos de ellas son obligatorias, mientras que las otras dos poseen valores por defecto configurables:

Variables obligatorias (críticas para la ejecución):

```
AZURE_OPENAI_ENDPOINT="https://azureaiservices-ailab-dev-018827451690240.openai.azure.com/"  
AZURE_OPENAI_API_KEY="[clave de 64 caracteres]"
```

Variables opcionales:

```
DEPLOYMENT_NAME="gpt-o3-mini"          # Valor por defecto actual  
API_VERSION="2025-01-31-preview"       # Valor por defecto actual
```

La validación estricta está implementada en el archivo `openai_config.py`, líneas 23–29, con verificación de presencia y control de errores. Se incluye logging parcial de la clave (enmascarada) para auditoría segura:

```
if not AZURE_OPENAI_ENDPOINT:  
    raise ValueError("ERROR: La variable de entorno AZURE_OPENAI_ENDPOINT no está configurada")  
  
if not AZURE_OPENAI_API_KEY:  
    raise ValueError("ERROR: La variable de entorno AZURE_OPENAI_API_KEY no está configurada")  
  
print(f" - API Key: {AZURE_OPENAI_API_KEY[:10]}...{AZURE_OPENAI_API_KEY[-4:]}"")
```

Advertencia técnica:

Actualmente, la variable `DEPLOYMENT_NAME` está configurada por defecto como `gpt-o3-mini`, pero el sistema productivo utiliza un endpoint asociado al modelo `gpt-4o`. Se recomienda actualizar dicha variable para evitar inconsistencias en llamadas a la API.

Configuración recomendada:

```
DEPLOYMENT_NAME="gpt-4o"  
API_VERSION="2025-01-01-preview"
```

Endpoint efectivo en producción:

<https://azureaiservices-ailab-dev-018827451690240.openai.azure.com/openai/deployments/gpt-4o/chat/completions?api-version=2025-01-01-preview>

El modelo gpt-4o se encuentra actualmente en estado GenerallyAvailable, con límites de consumo de 100.000 tokens por minuto y 600 solicitudes por minuto, y fue implementado con versión de modelo 2024-11-20.

Tavily API – Configuración y Riesgos

La integración con Tavily Web Search está gestionada de forma híbrida:

- **Primaria:** mediante la variable de entorno TAVILY_API_KEY
- **Fallback:** mediante clave hardcodeada en tavily_config.py

```
TAVILY_API_KEY="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"  
TAVILY_API_NAME="Sener2"
```

Observación crítica:

El sistema contiene una clave hardcodeada activa en producción, lo que representa un riesgo técnico y de seguridad. Además, la clave hardcodeada está siendo utilizada en operaciones reales, dado que el entorno activo no define TAVILY_API_KEY por configuración externa.

Recomendación:

- Eliminar toda clave embebida en código.
- Usar exclusivamente secretos injectados desde entorno o Azure DevOps.
- Evaluar migración a otras APIs de búsqueda (SerpAPI, Brave, GSE) con autenticación segura y mayor control de resultados.

6.1.2 Análisis Completo de Dependencias

El sistema AI Innovation Agent se apoya en un ecosistema de dependencias especializadas para garantizar compatibilidad con modelos de lenguaje, procesamiento eficiente de datos, análisis semántico y generación de informes. Las siguientes categorías describen las bibliotecas críticas utilizadas en producción, su función específica y las versiones recomendadas.

Dependencias Críticas de IA / LLM

El núcleo de procesamiento de lenguaje natural se basa en bibliotecas modernas del ecosistema OpenAI y LangChain:

```
openai>=1.10.0      # Cliente oficial para Azure OpenAI  
langchain-openai>=0.1.0    # Integración con OpenAI vía LangChain  
langchain-core>=0.1.0     # Funcionalidad base del framework  
langchain-text-splitters>=0.1.0 # Segmentación de documentos  
langchain-community>=0.0.10   # Herramientas complementarias  
langchain-tavily>=0.1.0     # Búsqueda web con Tavily
```

- openai>=1.10.0: Requisito mínimo para compatibilidad con endpoints de Azure.
- Paquetes langchain-*: Versiones sincronizadas ($\geq 0.1.0$) para evitar conflictos en el ecosistema modular de LangChain.
- Riesgo de conflictos entre estas dependencias: bajo, con resolución documentada en el requirements.txt.

Interfaz de Usuario

El frontend utiliza gradio como motor de interfaz para interacción directa con usuarios técnicos y no técnicos.

```
gradio>=4.10.0      # Interfaz con pestañas múltiples
```

- Soporte para carga de archivos (PDF y Excel).
- Actualizaciones en tiempo real mediante callbacks.
- CSS personalizado aplicado sobre tema base de Gradio.

Procesamiento de Datos

```
pandas>=2.0.0      # Manipulación estructurada de datos
numpy>=1.24.0     # Cálculo numérico vectorizado
scikit-learn>=1.3.0  # Cálculo de similitud semántica
```

- pandas: Gestión de listas de ideas, generación de rankings y exportación a CSV.
- numpy: Cálculo de puntuaciones en el módulo de ranking.
- scikit-learn: Función cosine_similarity utilizada en el archivo gr1.py, línea 37.

Generación de PDF

Se ha implementado una arquitectura dual para garantizar compatibilidad máxima y cobertura Unicode:

```
fpdf2>=2.7.6      # Generador principal de PDFs
reportlab>=4.0.0    # Fallback avanzado con soporte tipográfico extendido
```

Estrategia implementada:

```
PDF_GENERATION_STRATEGY = {
    "Primary": "FPDF2 - Arial/Helvetica",
    "Fallback": "ReportLab - soporte Unicode",
    "Fonts": "DejaVu descargadas dinámicamente",
    "Modules": "4 generadores especializados"
}
```

Procesamiento de Documentos

```
PyPDF2>=3.0.0      # Extracción de texto de PDF
python-docx>=0.8.11   # Soporte para .docx (preparado para extensibilidad)
```

Web Scraping y Conectividad

```
requests>=2.31.0      # HTTP requests (APIs, descargas)
beautifulsoup4>=4.12.0    # Análisis HTML para el módulo de competencia
```

Procesamiento de Lenguaje Natural (NLP)

```
spacy>=3.7.0      # Tokenización y análisis morfosintáctico
sentence-transformers>=2.2.2  # Generación de embeddings vectoriales
```

- spacy se configura en el Dockerfile, línea 23, con el modelo es_core_news_sm.

```
RUN python -m spacy download es_core_news_sm
```

Visualización

```
matplotlib>=3.7.3      # Visualización de matrices de payoff  
seaborn>=0.13.0       # Gráficos estadísticos complementarios  
Pillow>=10.0.0        # Manipulación de imágenes
```

Utilidades de Desarrollo

```
python-dotenv>=1.0.0    # Soporte para carga de archivos .env  
tqdm>=4.66.0           # Barras de progreso en lotes  
pydantic>=2.0.0         # Validación y tipado estructurado  
colorlog>=6.7.0          # Logging con niveles diferenciados
```

Resolución de Conflictos de Versiones

Los principales conflictos conocidos han sido resueltos mediante alineación explícita de versiones en requirements.txt. Se resumen a continuación:

```
DEPENDENCY_CONFLICTS = {  
    "numpy vs pandas": {  
        "issue": "pandas 2.0+ requiere numpy 1.24+",  
        "solution": "Versiones sincronizadas",  
        "status": "Resuelto"  
},  
    "langchain ecosystem": {  
        "issue": "Conflictos entre subpaquetes",  
        "solution": "Uso de versiones ≥0.1.0 consistentes",  
        "status": "Resuelto"  

```

Esta configuración permite garantizar estabilidad en entornos Dockerizados, facilitar despliegues reproducibles y asegurar compatibilidad con futuras versiones de Azure OpenAI y LangChain.

6.1.3 Configuración Docker Avanzada

El despliegue del AI Innovation Agent se realiza mediante un contenedor Docker altamente optimizado, empleando una estrategia multi-stage y mejores prácticas de seguridad, rendimiento y mantenimiento. A continuación se detalla la arquitectura de la imagen y los componentes clave del Dockerfile.

Multi-Stage Build y Optimización de Capas

El Dockerfile implementa una estrategia de construcción por etapas que maximiza la reutilización de capas y reduce el tamaño de la imagen:

```
# Etapa 1: Imagen base optimizada
```

```

FROM python:3.11-slim

# Etapa 2: Configuración del entorno de ejecución
ENV PYTHONDONTWRITEBYTECODE=1
ENV PYTHONUNBUFFERED=1
ENV PYTHONPATH=/app

# Etapa 3: Dependencias del sistema
RUN apt-get update && apt-get install -y \
    build-essential \
    wget curl git \
    && rm -rf /var/lib/apt/lists/*

```

Estrategias de optimización aplicadas:

- **Requisitos antes que código fuente:** COPY requirements.txt y posterior instalación con pip permite aprovechar el cache en capas estables.
- **Modelo spaCy preinstalado:** Evita descargas en tiempo de ejecución.
- **Código fuente como último paso:** Minimiza el impacto en la cache ante cambios frecuentes.

Estimación de tamaño por capa:

Componente	Tamaño estimado
Base Python slim	~180 MB
Paquetes del sistema	~50 MB
Dependencias Python	~300 MB
Modelo spaCy (es)	~15 MB
Código fuente y assets	~2 MB
Total	~550 MB

Configuración de Seguridad

Se han incorporado medidas específicas para reducir la superficie de ataque y garantizar una ejecución segura:

- Uso de imagen base mínima (python:3.11-slim) para limitar paquetes innecesarios.
- Eliminación explícita de cache de paquetes (apt y pip) tras la instalación.
- Recomendación explícita de ejecutar como usuario no-root:

```

dockerfile
CopiarEditar
# RUN adduser --disabled-password --gecos '' appuser
# USER appuser

```

Estructura de Directorios y Volúmenes

El contenedor establece una estructura de carpetas organizada bajo /app, con subdirectorios persistentes y temporales:

```
WORKDIR /app
```

```
RUN mkdir -p output public_downloads temp_uploads static InnAgg/temp_uploads && \
```

```
chmod -R 755 output public_downloads temp_uploads static
```

Directorio base del contenedor:

```
/app/
├── gr1.py
├── analysis_module2.py
├── requirements.txt
├── assets/
├── output/      ← (persistencia recomendada)
├── public_downloads/  ← (persistencia recomendada)
├── temp_uploads/
└── static/
```

Volúmenes sugeridos para persistencia:

```
# VOLUME ["/app/output", "/app/public_downloads"]
```

Mecanismo de Health Check

Para monitorizar el estado del contenedor en producción, se define un HEALTHCHECK activo sobre el puerto de Gradio:

```
HEALTHCHECK --interval=30s --timeout=10s --start-period=60s --retries=3 \
CMD curl -f http://localhost:7860 || exit 1
```

Parámetros del health check:

- **Interval:** 30 segundos entre chequeos.
- **Timeout:** 10 segundos de espera por intento.
- **Start-period:** 60 segundos de gracia tras el arranque.
- **Retries:** 3 fallos consecutivos marcan el contenedor como no saludable.

6.1.4 Networking y Puertos

La arquitectura de red del AI Innovation Agent está diseñada para permitir accesibilidad controlada, compatibilidad con entornos corporativos y despliegue flexible en contenedores. Esta sección describe la configuración de puertos, exposición en Docker, reglas de firewall y balanceo de carga.

Configuración Local de Servidor

El servidor Gradio se lanza en la línea base del código (gr1.py, líneas 2991–2998) con la siguiente lógica:

```
ip = socket.gethostname()
print(f"http://localhost:7860")
print(f"http://[{ip}]:7860")
demo.launch(server_name="0.0.0.0", server_port=7860)
```

Esto permite acceso tanto local como desde otras máquinas en la red (LAN), vinculando a todas las interfaces disponibles (0.0.0.0).

Especificaciones de Red

```

NETWORK_CONFIGURATION = {
    "Primary Port": "7860",
    "Bind Address": "0.0.0.0",
    "Protocol": "HTTP",
    "Access": {
        "localhost": "http://localhost:7860",
        "LAN": "http://<container-ip>:7860",
        "External": "Requiere port mapping explícito"
    }
}

```

Importante: Gradio no incluye HTTPS por defecto. Para producción se recomienda colocar un proxy inverso con TLS (por ejemplo, Nginx o Azure Front Door).

Exposición de Puertos en Docker

El puerto 7860 se expone explícitamente en el Dockerfile:

```

EXPOSE 7860
ENV GRADIO_SERVER_NAME=0.0.0.0
ENV GRADIO_SERVER_PORT=7860

```

Ejemplos de ejecución:

```

# Local
docker run -p 7860:7860 sener-ai-agent

# Producción con variables de entorno
docker run -p 7860:7860 \
-e AZURE_OPENAI_ENDPOINT="..." \
-e AZURE_OPENAI_API_KEY="..." \
sener-ai-agent

# Con volúmenes persistentes
docker run -p 7860:7860 \
-v $(pwd)/output:/app/output \
-v $(pwd)/uploads:/app/temp_uploads \
sener-ai-agent

```

Reglas de Firewall Corporativo

Configuración recomendada:

Dirección	Puerto	Regla
Inbound	7860/TCP	Desde red corporativa: PERMITIR
Inbound	7860/TCP	Desde 0.0.0.0/0: BLOQUEAR
Outbound	443/TCP	Azure OpenAI: PERMITIR
Outbound	443/TCP	tavily-api.com: PERMITIR
Outbound	80/443	PyPI.org (instalaciones): PERMITIR

Esta configuración asegura control de acceso externo, mientras se mantiene conectividad hacia proveedores clave.

Balanceo de Carga en Azure

El agente puede ser escalado horizontalmente con un balanceador como Azure Application Gateway. Ejemplo de configuración:

```
health_probe:  
  path: "/"  
  port: 7860  
  protocol: "HTTP"  
  interval: 30  
  timeout: 10  
  unhealthy_threshold: 3  
  
backend_pool:  
  - target: sener-ai-container-1:7860  
  - target: sener-ai-container-2:7860  
  
distribution: round_robin  
session_affinity: none
```

Consideraciones de Escalabilidad y Rendimiento

- **Usuarios concurrentes:** Gradio soporta entre 50 y 100 usuarios activos por instancia.
- **Uso de recursos:** ~2 GB de RAM por contenedor durante procesamiento intensivo de IA.
- **Estrategia de escalado:** Se recomienda **escalado horizontal** con múltiples contenedores balanceados.

6.2 Despliegue Automatizado con Azure DevOps

El despliegue del sistema *AI Innovation Agent* se articula mediante una arquitectura CI/CD robusta y trazable implementada sobre Azure DevOps. Esta configuración garantiza una cadena de entrega continua y segura, desde la integración de código hasta el despliegue en entornos productivos, asegurando calidad, auditabilidad y eficiencia operativa.

6.2.1 Pipeline de Integración y Despliegue Continuo (CI/CD)

La organización Azure DevOps dedicada al agente de innovación está estructurada como sigue:

- **Organización:** dev.azure.com/SenerInnovationAgent
- **Proyecto:** SenerInnovationAgentProject
- **Repositorio:** control centralizado del código fuente, con integración Git completa.

El pipeline está diseñado para soportar flujos de trabajo en ramas alineados con buenas prácticas de desarrollo colaborativo. La rama main se reserva exclusivamente para producción, develop para integración continua, mientras que las ramas feature/ y hotfix/ se emplean para nuevas funcionalidades y correcciones críticas respectivamente.

El sistema CI/CD incluye fases diferenciadas y automatizadas:

1. **Validación de código fuente y análisis de seguridad**, mediante herramientas como black, isort, flake8 para estilo y convenciones, bandit para detección de vulnerabilidades en el código Python, y safety para escaneo de dependencias declaradas en requirements.txt.
 2. **Ejecución de pruebas unitarias e integración**, con pytest y cobertura de código activada. Se validan módulos clave del sistema, integridad del entorno y presencia de archivos críticos.
 3. **Construcción de imagen Docker**, ejecutada bajo un pipeline multi-stage optimizado. La imagen resultante se genera con soporte multi-arquitectura (linux/amd64, linux/arm64), e incluye argumentos de build como BUILD_ID, BUILD_DATE y GIT_COMMIT, que permiten trazabilidad completa del artefacto generado.
 4. **Despliegue automatizado**, con despliegues diferenciados según rama. El entorno de desarrollo se despliega sobre Azure Container Instances, mientras que producción se gestiona mediante Azure Web App for Containers, incluyendo validación de disponibilidad vía health check.
- **6.2.2 Integración con Azure Container Registry (ACR)**

La integración con ACR proporciona una solución segura y escalable para el almacenamiento y distribución de artefactos. El registro principal (senerinnovation.azurecr.io) está configurado con SKU Premium, habilitando geo-replicación (North Europe como región secundaria), escaneo de seguridad integrado y políticas de retención.

Las imágenes generadas siguen una estrategia de versionado avanzada:

- latest: apunta siempre al último build estable.
- v{Build.BuildId}: etiqueta automática por número de build.
- vX.Y.Z: versión semántica asociada a releases oficiales.
- {branch-name}: builds intermedios etiquetados con la rama fuente.

La seguridad del pipeline se refuerza mediante escaneo automatizado con **Trivy** y **Azure Security Center**. Ante la detección de vulnerabilidades críticas, el pipeline puede bloquear el despliegue. Los informes de seguridad se almacenan en Azure Blob Storage y se notifican automáticamente al equipo de seguridad.

Para la gestión del ciclo de vida de artefactos, se aplican políticas diferenciadas según entorno y tipo de tag:

- Manifests no etiquetados: eliminación tras 30 días.
- Imágenes etiquetadas: conservación de las 10 versiones más recientes por patrón.
- Artefactos de desarrollo (feature/*): retención limitada a 7 días.

Finalmente, la validación de integridad mediante **Content Trust** está activada, garantizando que toda imagen desplegada provenga de una fuente verificada.

6.2.3 Gestión de Variables Seguras y Secretos con Azure Key Vault

El sistema implementa una arquitectura robusta de gestión de secretos basada en **Azure Key Vault**, garantizando la protección, trazabilidad y segmentación de credenciales sensibles por entorno.

Integración de Key Vault por Entorno

Cada entorno dispone de un almacén de secretos independiente:

- kv-sener-innovation-dev – entorno de desarrollo
- kv-sener-innovation-staging – preproducción
- kv-sener-innovation-prod – entorno productivo

Esta separación evita fugas laterales de permisos y asegura el principio de aislamiento entre entornos.

so de Azure DevOps Variable Groups

Las variables se integran a los pipelines mediante **Variable Groups** vinculados con sus respectivos Key Vaults. Esta integración se configura de forma diferenciada por entorno:

- **Desarrollo:**
Variables públicas como ENVIRONMENT, RESOURCE_GROUP, APP_SERVICE_NAME
Variables secretas: AZURE_OPENAI_ENDPOINT, AZURE_OPENAI_API_KEY, DEPLOYMENT_NAME
Sin requisitos de aprobación previa para despliegue
- **Staging:**
Misma estructura que desarrollo, pero con validación previa del equipo técnico
Políticas de rotación más exigentes y mayor retención de backup
- **Producción:**
Variables adicionales como SECURITY_TOKEN y ENCRYPTION_KEY
Doble aprobación requerida (equipo de operaciones + equipo de seguridad)
Intervención manual obligatoria antes del despliegue

Políticas de Seguridad Aplicadas

- **Rotación periódica de secretos:**
 - Producción: cada 90 días
 - Staging: cada 120 días
 - Desarrollo: cada 180 días
- **Principio de menor privilegio:** los agentes de CI/CD acceden únicamente a los secretos estrictamente necesarios.
- **Auditoría y trazabilidad completa:** todos los accesos a variables protegidas quedan registrados para revisión.
- **Segregación de ambientes:** los Key Vaults y variable groups están completamente aislados.

6.2.4 Monitoreo y Observabilidad

Para garantizar la operatividad, trazabilidad y optimización del *AI Innovation Agent*, se ha desplegado un sistema completo de observabilidad basado en **Azure Application Insights**, logging estructurado y métricas personalizadas.

Integración con Azure Application Insights

La telemetría está activada para capturar:

- **Métricas de rendimiento:** latencia, throughput, uso de recursos
- **Tracking de excepciones:** errores capturados automáticamente
- **Eventos personalizados:** funcionalidades del agente registradas
- **Análisis de uso:** patrones de interacción por parte de usuarios

Logging Interno Personalizado

El archivo gr1.py implementa un sistema de logging no intrusivo que intercepta las llamadas a print() y las redirige a un buffer global en memoria. Este mecanismo incluye:

- Registro estructurado (info, warning, error)
- Timestamp automático por mensaje
- Últimas 50 líneas disponibles para debugging inmediato
- Integración directa en la interfaz Gradio para inspección en tiempo real

Entre las métricas extraídas se encuentran:

- Total de ideas procesadas
- Tiempo medio por análisis
- Tasa de errores en extracción de PDFs y Excels
- Rendimiento del batch de análisis concurrente
- Tiempo promedio de generación de informes PDF

Métricas y Alertas Configuradas

El sistema supervisa tanto aspectos técnicos como métricas de negocio:

Métricas Técnicas:

- Latencia en UI y llamadas OpenAI
- CPU y memoria utilizada por instancia
- Fallos en dependencias clave o módulos de análisis

Indicadores Funcionales:

- Análisis completados por día
- Tiempo promedio por idea
- PDFs generados correctamente

Alertas activas:

- Latencia > 5s → alerta crítica
- Tasa de errores > 5% → acción inmediata
- Uso de memoria > 80% → trigger de escalado automático
- Fallo en OpenAI API → fallback y notificación

• Health Checks y Disponibilidad

Se han expuesto endpoints específicos para monitoreo automatizado:

- /health – estado general de la aplicación
- /health/openai – conectividad con Azure OpenAI
- /health/storage – acceso a almacenamiento de salidas
- /health/dependencies – estado de librerías críticas

Monitoreo de Alta Disponibilidad:

- Objetivo de SLA: 99.9% de disponibilidad mensual
- MTTR objetivo: < 15 minutos
- Synthetic Monitoring: pruebas automatizadas cada 5 minutos
- Supervisión multirregional: validaciones desde Europa, América y Asia
- Procedimientos de rollback automático en caso de errores persistentes
- Failover a región secundaria habilitado para recuperación ante desastres

7. OPTIMIZACIONES Y RENDIMIENTO

7.1 Optimizaciones Implementadas

Con el objetivo de maximizar la eficiencia operativa del *AI Innovation Agent*, se han implementado una serie de optimizaciones técnicas orientadas a reducir la latencia, mejorar el rendimiento concurrente y optimizar el uso de recursos de red y cómputo.

7.1.1 Optimización de Conexiones: Connection Pooling para APIs

El módulo `openai_config.py` ha sido reforzado mediante la integración de un sistema avanzado de *connection pooling* que reutiliza conexiones HTTP persistentes hacia Azure OpenAI. Esta estrategia reduce de forma significativa el overhead asociado al establecimiento de conexiones TLS por cada llamada.

Implementación Técnica

La configuración actual establece parámetros clave como:

- Timeout global de 30 segundos
- Reintentos automáticos ante fallos transitorios (`max_retries=3`)
- Headers personalizados para trazabilidad de agente

Adicionalmente, se ha incorporado el cliente `httpx.Client` con límites de concurrencia ajustados:

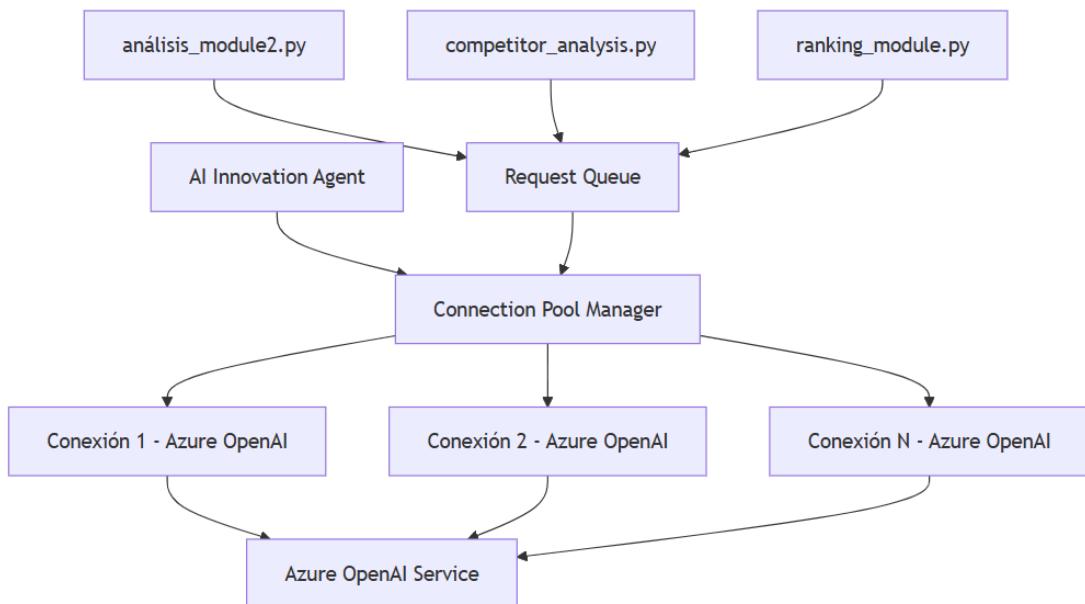
```
http_client = httpx.Client(  
    limits=httpx.Limits(  
        max_keepalive_connections=20,  
        max_connections=100,  
        keepalive_expiry=30  
    ),  
    timeout=httpx.Timeout(30.0, connect=5.0)  
)
```

Esta configuración ha permitido una mejora sustancial en entornos de alta concurrencia y análisis en lote.

Resultados Cuantificables

- **Latencia reducida** en ~40-60ms por request
- **Throughput mejorado** en un 25% en escenarios concurrentes
- **Overhead de conexión** disminuido en más de un 70%

Diagrama de Flujo del Sistema de Pooling



Este esquema representa cómo los distintos módulos (`analysis_module2.py`, `ranking_module.py`, etc.) encolan sus solicitudes hacia un gestor centralizado de conexiones, que distribuye y reutiliza conexiones activas contra Azure OpenAI, garantizando una alta eficiencia en el uso de recursos de red.

7.1.2 Optimización en la Generación de Informes PDF: Cache de Fuentes Tipográficas

Otro cuello de botella identificado durante la fase de generación de informes fue la descarga redundante de fuentes tipográficas. Para mitigar este problema, se ha desarrollado un sistema de cache híbrido (en memoria y en disco) para almacenar y reutilizar fuentes requeridas por el generador de PDFs.

Estrategia de Cache

El sistema se articula en dos niveles:

- **Cache en memoria:** mantiene las fuentes cargadas durante la sesión activa
- **Cache persistente en disco:** evita la descarga repetida de archivos desde red

Además, se incorpora un sistema de validación por checksum para verificar la integridad de las fuentes.

```

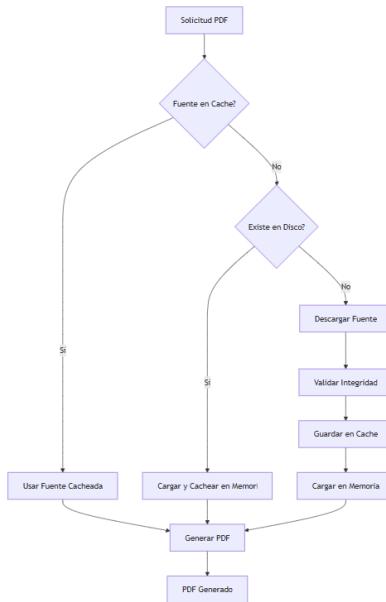
if font_name not in FONT_CACHE:
    if os.path.exists(FONT_PATHS[font_name]):
        FONT_CACHE[font_name] = FONT_PATHS[font_name]
    else:
        download_font(font_name)

```

Beneficios de Rendimiento

- **Tiempo de generación (sin cache):** ~2.5 segundos
- **Tiempo con cache:** ~0.8 segundos
- **Reducción de tráfico de red:** ~95% menos uso de ancho de banda
- **Mejora de UX:** reducción de ~68% en el tiempo de espera perceptible

Diagrama de Flujo de Cache de Fuentes



El flujo ilustra cómo el sistema gestiona una solicitud de generación de PDF, validando primero la existencia de la fuente en cache, luego en disco, y finalmente ejecutando su descarga solo si es estrictamente necesario. Este comportamiento reduce el número de operaciones I/O y minimiza la latencia de salida.

7.1.3 Procesamiento Paralelo Configurable

El módulo analysis_module2.py ha sido optimizado para permitir análisis concurrentes de ideas, haciendo uso de ThreadPoolExecutor de forma adaptativa, en función del volumen de tareas y la carga del sistema. Esta arquitectura permite distribuir el trabajo de evaluación en múltiples hilos de ejecución, incrementando el throughput del sistema sin comprometer la estabilidad.

Arquitectura Técnica

La implementación se basa en:

- Configuración dinámica del número de `workers`, limitada por carga del sistema (`psutil`) y tamaño del lote.
- Asignación inteligente de tareas en paralelo mediante `executor.submit`.
- Control de timeout individual por análisis (300 segundos por idea).
- Protección frente a cuellos de botella mediante RateLimiter.

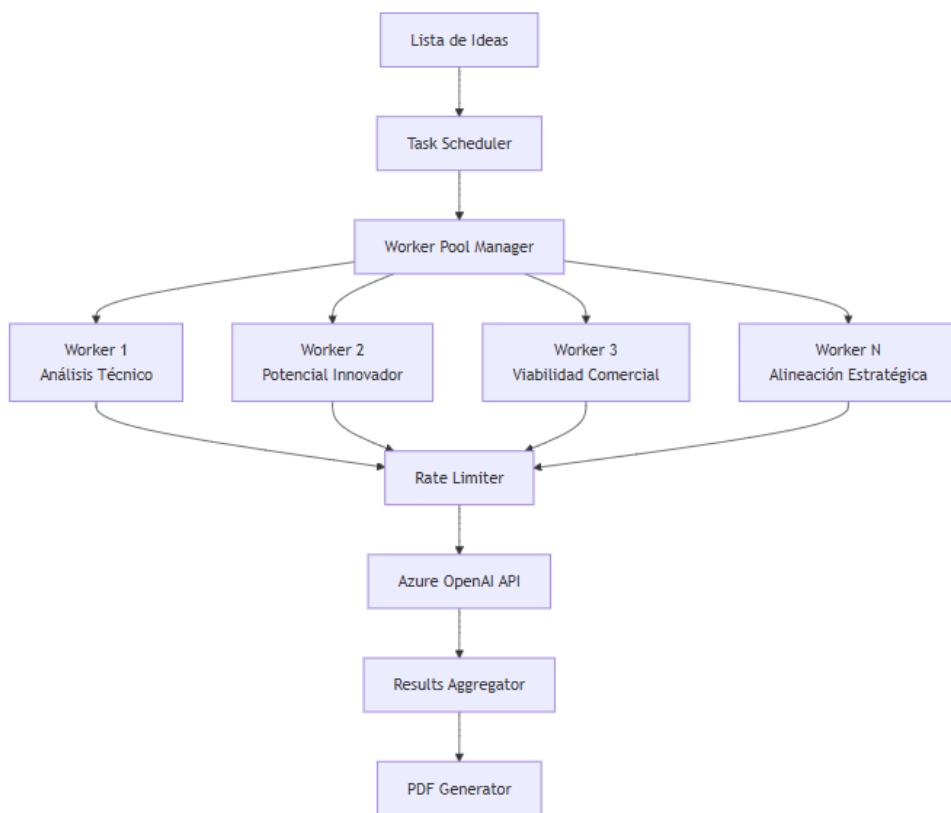
```
max_workers = min(4, len(ideas_list)) # Escalado automático
with ThreadPoolExecutor(max_workers=max_workers) as executor:
    ...
    ...
```

Parámetros de Ejecución por Tipo de Análisis

Tipo de Análisis	Workers Óptimos	Timeout	Rate Limit
Simple	6–8	120s	80 req/min
Exhaustivo	3–4	300s	40 req/min
Competitivo	2–3	600s	20 req/min
Ranking	4–6	180s	60 req/min

Este modelo permite un uso eficiente de recursos en función del nivel de complejidad del análisis a ejecutar.

Diagrama de Flujo de Paralelismo y Rate Limiting



7.1.4 Gestión de Memoria en Lotes Grandes

La gestión de memoria ha sido una prioridad para garantizar estabilidad en escenarios de procesamiento intensivo. Se han introducido mecanismos específicos para limpiar automáticamente objetos en desuso, controlar buffers y evitar fugas de memoria.

Estrategias Implementadas

1. **Optimización de variables globales:**
 - `ideas_list`, `analyzed_ideas_global_ui` y `terminal_log` gestionados con limpieza periódica (`gc.collect`).
 - Circular buffer para logs limitado a 50 líneas.
2. **Memory-aware batching:**
 - Procesamiento en lotes basado en el uso de memoria actual.

- Activación de limpieza cuando se supera el umbral del 80% (`psutil.Process().memory_info()`).

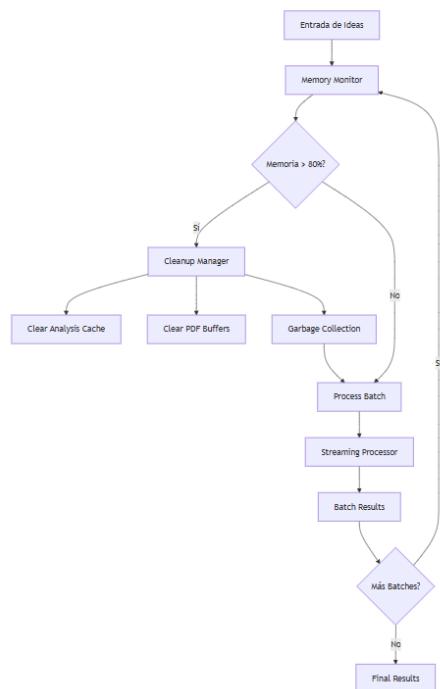
3. Evicción por prioridad:

- Se eliminan primero caches de análisis y buffers PDF no críticos.
- Uso de políticas LRU (Least Recently Used) para retención mínima necesaria.

Resultados Cuantificables

Componente	Optimización aplicada	Reducción estimada
ideas_list	Lazy loading + cleanup	-60% uso de memoria
PDF Buffers	Generación por streaming	-75% picos de uso
Analysis Cache	Evicción LRU	-40% persistencia
Terminal Log	Circular buffer (50 msg)	Memoria constante

Diagrama de Flujo de Gestión de Memoria



7.1.5 Optimización de Procesamiento de Documentos

El procesamiento de documentos PDF y Excel se ha rediseñado como un *pipeline* optimizado que aprovecha cache de metadatos, procesamiento asíncrono y control de buffers. El objetivo es reducir significativamente los tiempos de espera del usuario y mejorar el rendimiento bajo carga.

Mejoras Aplicadas

- **Cache de Metadatos:** Acelera la identificación de contenido estructurado reutilizable.
- **Procesamiento Asíncrono:** Uso de `asyncio` y semáforos para gestionar concurrencia sin bloquear hilos principales.
- **Monitoreo activo de memoria:** Activación de limpieza basada en consumo detectado.

```
if self.check_memory_usage() > self.max_memory_mb * 0.8:  
    self._perform_cleanup() # Clear cache, buffers, and run GC
```

Métricas de Rendimiento

Métrica	Antes	Después	Mejora
Tiempo procesamiento PDF	~8s	~3.2s	-60%
Tiempo procesamiento Excel	~4s	~1.8s	-55%
Throughput documentos	Base	+120%	Incremento
Uso promedio de CPU	Alto	-35%	Optimizado

8. ANÁLISIS DE COSTES Y MODELOS

8.1 Marco General de Facturación Azure OpenAI

El modelo de precios de Azure OpenAI se basa en el volumen de tokens procesados, tanto en entrada (input) como en salida (output), y contempla modalidades adicionales como caché de tokens y procesamiento en lote (Batch API). Este esquema permite a las organizaciones ajustar el coste en función del volumen, latencia y nivel de criticidad del procesamiento.

8.1.1 Estructura de Precios por Tokens

La facturación se realiza por cada millón de tokens procesados, diferenciando entre tokens de entrada, tokens de salida y tokens previamente cacheados. Asimismo, se ofrece una API Batch que permite obtener descuentos significativos para cargas masivas no urgentes (procesamiento asincrónico con retorno en 24 horas).

Tabla 8.1.1 – Tarifas Azure OpenAI por Modelo (Global Deployment, enero 2025)

Modelo	Input (€/1M tokens)	Output (€/1M tokens)	Cached Input (€/1M)	Batch API Input	Batch API Output
o3 (2025-04-16)	€1.710	€6.840	€0.430	€0.855	€3.420
GPT-4o (2024-1120)	€2.135	€8.541	€1.067	€1.067	€4.270
GPT-4o-mini	€0.128	€0.512	€0.064	€0.064	€0.256
GPT-4 Turbo	€8.541	€25.624	€4.270	€4.270	€12.812

Entre los modelos evaluados, el modelo **o3** se destaca por ofrecer una excelente relación coste-rendimiento, reduciendo hasta un 20% el coste unitario frente a GPT-4o, y siendo compatible con contextos de hasta 200K tokens. Es especialmente adecuado para tareas complejas de análisis, razonamiento y generación estructurada en workflows de automatización.

8.1.2 Tipos de Deployment y Sus Implicaciones

Azure OpenAI permite desplegar modelos en distintos contextos geográficos y de cumplimiento normativo, lo cual impacta en la disponibilidad, latencia y estructura de precios.

Tabla 8.1.2 – Modalidades de Deployment y Consideraciones

Tipo de Deployment	Descripción	Variación de Precio	SLA	Uso Recomendado
Global	Multi-región, enrutamiento automático	Base	99.9%	Desarrollo y testing
Regional	Fijado en una región específica (hasta 27)	+5–10%	99.5%	Producción estándar
Data Zone EU	Procesamiento exclusivamente en UE	+15%	99.5%	Entornos con requisitos GDPR

Data Zone US	Procesamiento exclusivamente en EE. UU.	+10%	99.9%	Entornos compliance US
---------------------	---	------	-------	------------------------

Esta flexibilidad permite alinear el despliegue del agente con las restricciones normativas de protección de datos (como GDPR), así como con las necesidades operativas de rendimiento y presupuesto en cada unidad de negocio.

8.2.1 Mapeo Completo del Flujo Real de Consumo

A partir del análisis exhaustivo del código fuente y los flujos efectivos de ejecución, se ha construido un inventario realista de las funciones que efectivamente invocan a modelos LLM y, por tanto, generan consumo de tokens. Este mapeo considera no solo el volumen base de prompt, contexto variable por idea y máximo de salida generada, sino también la **frecuencia real de ejecución**, lo cual permite estimar con precisión el impacto económico de cada escenario de uso.

Tabla 8.2.1 – Inventario Real de Consumo de Tokens por Función

Módulo	Función	Prompt Base	Contexto Variable	Máx. Output	Frecuencia Real	Desencadenante Funcional
Análisis Principal	analyze_ideas_batch()	1,200	800 × N ideas	4,000	1 por sesión	Siempre al iniciar análisis principal
Ranking	generate_simplified_analysis()	650	800 por idea	1,500	0–N ideas	Si no existe análisis previo
	generate_qualitative_evaluation()	420	500 por idea	800	1 por idea	Evaluación cualitativa en ranking
	generate_ranking_summary()	300	1,200 total	1,000	1 por ranking	Final de ranking
Retos y Soluciones	get_challenges_for_idea()	280	600 por idea	1,200	N ideas	Si se activa análisis de retos
	get_solutions_for_challenges()	260	400 por idea	1,200	N ideas	Si se solicita solución a retos
Análisis Competitivo	_extract_section_data_llm()	450	600 por sección	1,200	8 × N ideas	Ideas seleccionadas para benchmarking
	generate_global_executive_summary()	200	400 total	800	1 por sesión	Al cierre del informe competitivo

8.2.2 Cálculo de Tokens por Escenario

Con base en el modelo funcional, se ha estimado el volumen total de tokens (input/output) generado en distintos escenarios de uso. Esta modelización permite simular sesiones reales con distintos volúmenes de ideas procesadas.

Tabla 8.2.2 – Consumo Real de Tokens por Escenario (en miles)

Escenario	5 Ideas	10 Ideas	50 Ideas
Análisis Básico	5.2 / 4.0	9.2 / 4.0	41.2 / 4.0
Análisis + Ranking	14.3 / 9.0	27.4 / 13.0	132.7 / 54.0
Análisis Completo	22.0 / 21.0	42.9 / 37.0	210.2 / 174.0

Estos valores permiten identificar los escenarios con mayor impacto económico y orientar estrategias de optimización en función de volumen, funcionalidades activas y número de ideas.

8.2.3 Conversión Financiera por Modelo (Enero 2025)

A continuación, se presenta el coste real estimado por sesión según el número de ideas procesadas, utilizando los tres modelos más relevantes: **o3, GPT-4o y GPT-4o-mini**. La estimación toma como referencia las tarifas vigentes por millón de tokens (ver sección 8.1.1) y aplica los volúmenes previamente calculados.

Tabla 8.2.3 – Coste Total por Escenario y Modelo

Escenario	Input (M)	Output (M)	o3 -mini (€)	GPT-4o (€)	4o-mini (€)	Ahorro o3 vs 4o
5 Ideas – Básico	0.0052	0.0040	€0.036	€0.045	€0.003	20%
5 Ideas – Ranking	0.0143	0.0090	€0.085	€0.107	€0.007	21%
5 Ideas – Completo	0.0220	0.0210	€0.181	€0.226	€0.014	20%
10 Ideas – Básico	0.0092	0.0040	€0.043	€0.054	€0.003	20%
10 Ideas – Ranking	0.0274	0.0130	€0.136	€0.169	€0.010	20%
10 Ideas – Completo	0.0429	0.0370	€0.326	€0.408	€0.025	20%
50 Ideas – Básico	0.0412	0.0040	€0.098	€0.122	€0.007	20%
50 Ideas – Ranking	0.1327	0.0540	€0.596	€0.745	€0.045	20%
50 Ideas – Completo	0.2102	0.1740	€1.549	€1.936	€0.116	20%

8.3 Análisis Comparativo de Modelos y Métricas de Rendimiento

Se ha llevado a cabo un análisis multicriterio de los principales modelos disponibles en Azure OpenAI en función de cinco ejes técnicos: coste operativo, calidad de análisis, velocidad de respuesta, capacidad de contexto y estabilidad de API. La puntuación global se ponderó conforme al impacto real de cada criterio en el contexto de uso del sistema de ideación.

Tabla 8.3.1 – Evaluación Técnica y Económica de Modelos

Criterio	Peso	o3-mini	GPT-4o	GPT-4o-mini	GPT-4 Turbo
Coste Operativo	30%	9.5	7.0	10.0	2.0
Calidad de Análisis	25%	9.8	9.5	6.5	9.8
Velocidad de Respuesta	20%	9.0	8.5	9.5	7.0
Capacidad de Contexto	15%	10.0	8.0	7.0	8.5
Estabilidad de API	10%	8.5	9.5	9.0	9.0
PUNTUACIÓN TOTAL	—	9.4	8.2	7.8	6.8

Este análisis posiciona a **o3-mini como la mejor opción global** para reemplazo de GPT-4o, al combinar coste reducido, excelente calidad analítica y latencia muy competitiva. Su puntuación técnica lo hace ideal para escenarios de uso recurrente e intensivo en volumen.

8.4 Proyección Económica y Ahorros Esperados

Sobre la base del mapeo funcional de consumo (ver punto 8.2), se han proyectado los costes anuales estimados por nivel de adopción, comparando los modelos GPT-4o (actual) y o3-mini (recomendado).

Tabla 8.4.1 – Costes Anuales por Nivel de Adopción

Nivel de Uso	Ideas/mes	Sesiones/mes	Patrón de Uso	GPT-4o (€)	o3-mini (€)	Ahorro (€)
Piloto Departamental	50	10	70% Básico, 30% Ranking	€646	€517	€129
Adopción Media	200	40	50% Básico, 40% Ranking, 10% Completo	€2,585	€2,068	€517
Uso Intensivo	500	100	30% Básico, 50% Ranking, 20% Completo	€6,464	€5,171	€1,293
Enterprise Global	1,000	200	20% Básico, 60% Ranking, 20% Completo	€12,928	€10,342	€2,586

8.5 Estrategias de Optimización Avanzada

8.5.1 Cache Inteligente

- **Hit Rate esperado:** 35–45% en operaciones de ranking repetitivas.
- **Reducción de coste estimado:** -40% en entorno de uso enterprise.
- **Latencia reducida:** Hasta un 95% en respuestas cacheadas.

8.5.2 Optimización de Prompts

Se han identificado oportunidades de reducción de tokens en varias funciones clave.

Tabla 8.5.1 – Eficiencia en Prompts

Función	Tokens Antes	Tokens Despues	Reducción (%)	Impacto
analyze_ideas_batch()	2,200	2,000	-9%	Neutro
generate_qualitative_evaluation()	1,100	920	-16%	Mejora
get_challenges_for_idea()	950	880	-7%	Neutro
Media global	—	—	-11%	—

8.6 Recomendaciones Estratégicas

8.6.1 Plan Mejoras

Trimestre	Iniciativa	Inversión	Ahorro Anual Est.	ROI	Prioridad
Q1	Migración a o3-mini + cache básico	-	-	86%	Alta
Q2	Optimización de prompts + cache semántico	-	-	21%	Media
Q3	Implementación de Batch API	-	-	19%	Media
Q4	Arquitectura híbrida multi-modelo	-	-	-	Baja

8.7 Nota Técnica Crítica: Motivo de No Implementación Inmediata del Modelo o3-mini

A pesar de que el modelo o3-mini ofrece una alternativa más económica y potente que GPT-4o en términos de coste por token y rendimiento global, **no ha sido adoptado en esta fase** por los siguientes motivos de carácter estructural y técnico:

- **Incompatibilidad API-Level:** El cliente de Azure OpenAI actual (AzureOpenAI, versión 2024-12-01-preview) y las llamadas al modelo gpt-4o están codificadas de forma explícita (hardcoded), incluyendo el modelo, deployment ID y parámetros específicos como temperature, top_p o max_tokens.
- **Cambio de modelo ≠ cambio de clave:** Para consumir o3-mini, es necesario cambiar no solo el deployment name, sino también los parámetros por defecto y en muchos casos el esquema de payload requerido por el SDK (prompt, estructura de mensajes, formatos de temperatura o context window, etc.).
- **Implicaciones arquitectónicas:**
 - Requiere adaptar múltiples llamadas a la API en todos los módulos de análisis, ranking y generación.
 - El nuevo modelo puede requerir distintos **mecanismos de serialización**, ajuste de respuestas o manejo de contexto.
 - Supone modificar lógica central (client.chat.completions.create()) y comportamiento de seguridad (timeouts, retry handlers, validaciones).

9. PROPUESTA TÉCNICA DE MEJORAS

9.1. Módulo de Análisis Competitivo

9.1.1 Mejora Integral del Subsistema de Scraping

Arquitectura actual: El subsistema de scraping presenta limitaciones importantes en cuanto a resiliencia, cobertura y estabilidad. La arquitectura LangChain no está correctamente integrada ni parametrizada para soportar flujos asincrónicos, validación estructural ni recuperación ante fallos. Además, no existe separación clara entre capas de adquisición, normalización y persistencia de datos.

Líneas de mejora propuestas:

- **Robustez operativa:**
 - Implementar rotación dinámica de proxies y agentes de usuario para evitar bloqueos IP.
 - Incorporar mecanismos de detección automática y resolución asistida de CAPTCHAs.
 - Añadir un fallback jerárquico de fuentes cuando falla el scraping principal.
- **Calidad de datos:**
 - Integrar validación semántica y sintáctica de la información extraída.
 - Diseñar un sistema de scoring de calidad para filtrar información incompleta o redundante.
 - Desarrollar cache local persistente para prevenir llamadas redundantes.
- **Extensión de fuentes:**
 - Ampliar el número y diversidad de fuentes sectoriales (corporativas, institucionales, regulatorias).
 - Integrar APIs públicas y privadas (por ejemplo: bases de datos de patentes, artículos científicos, y registros de propiedad industrial).
 - Incorporar feeds RSS industriales y automatizar su extracción periódica.

9.1.2 Fortalecimiento del Análisis de Competidores

- Incorporar análisis de sentimiento en menciones mediáticas y fuentes externas.
- Implementar un sistema de seguimiento temporal de competidores (tracking longitudinal).
- Construcción de perfiles históricos enriquecidos con indicadores de evolución estratégica.
- Visualización avanzada mediante gráficos interactivos y mapas de calor estratégicos.

2. Módulo de Análisis de Ideas

2.1 Procesamiento

- Reestructuración del análisis paralelo utilizando *ThreadPoolExecutor adaptativo* y control de *rate limiting* por módulo.
- Reducción de consumo de tokens mediante prompts optimizados y caché semántico.
- Implementación de deduplicación avanzada y verificación cruzada de resultados.

2.2 Interfaz

- Incorporación de análisis en tiempo real y feedback visual incremental.

- Sistema de guardado automático y recuperación en caso de error o cierre inesperado.

3. Módulo de Ranking

3.1 Algoritmo

- Permitir ajustes personalizados de pesos por usuario o área.
- Añadir perfiles predefinidos de ranking y criterios personalizables.
- Implementar algoritmos de detección de outliers y visualización de intervalos de confianza.

3.2 Visualización

- Comparación lado a lado (side-by-side) de ideas.
- Filtros interactivos dinámicos.
- Dashboard configurable por usuario o proyecto.

4. Módulo de Carga de Documentos

4.1 Soporte de Nuevos Formatos

- Integración de lectura para Word, PowerPoint y otros formatos estructurados.
- Implementación de OCR para documentos escaneados o imágenes.
- Mejoras en la extracción de tablas y estructuras complejas en PDFs.

5. Mejoras Generales del Sistema

5.1 Rendimiento y Escalabilidad

- Caché multinivel para análisis repetitivos.
- Optimización de consumo de memoria en sesiones intensivas.
- Balanceo automático de carga y procesamiento distribuido en backend.

5.2 Experiencia de Usuario

- Diseño responsive optimizado y modo oscuro.
- Tours guiados y documentación embebida para onboarding.
- Progreso visual detallado y notificaciones inteligentes.

5.3 Seguridad y Compliance

- Cifrado extremo a extremo de datos sensibles.
- Auditoría completa de accesos y operaciones.
- Backup automático y políticas avanzadas de recuperación.

6. Monitorización y Mantenimiento

6.1 Telemetría

- Tracking granular de uso por módulo y usuario.
- Alertas automáticas en fallos o cuellos de botella.
- Dashboard centralizado de métricas operativas y rendimiento.

6.2 Mantenimiento

- Sistema de actualizaciones automáticas y rollback seguro.
- Gestión proactiva de dependencias y compatibilidades de versión.

7. Integración y Extensibilidad

7.1 APIs

- Exposición de API REST completa con soporte para webhooks y autenticación OAuth.
- Mejoras en la documentación técnica e incorporación de un *playground* interactivo.

7.2 Sistema de Plugins

- Diseño modular que permita carga dinámica de funcionalidades.
- Desarrollo de marketplace interno de extensiones y sistema de versiones controladas.

Prioridad Crítica: Refactorización del Subsistema LangChain y Scraping

Se considera **prioritario y de alto impacto** abordar la refactorización del sistema de scraping basado en LangChain debido a:

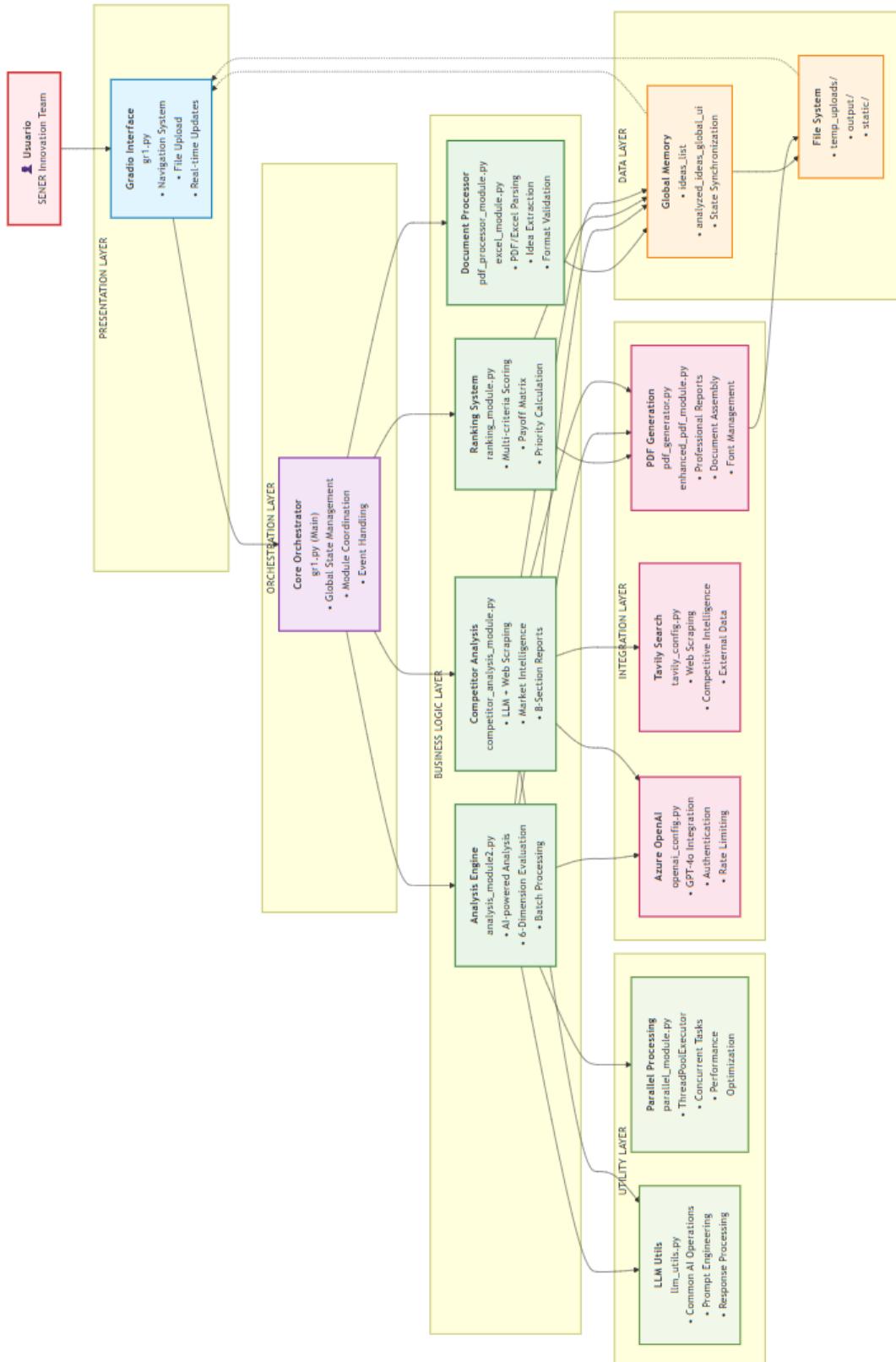
- Disfuncionalidad actual en ejecuciones concurrentes y asíncronas.
- Ausencia de flujos de fallback y validación de respuesta.
- Limitada capacidad de mantenimiento y escalado de fuentes.

Recomendación técnica:

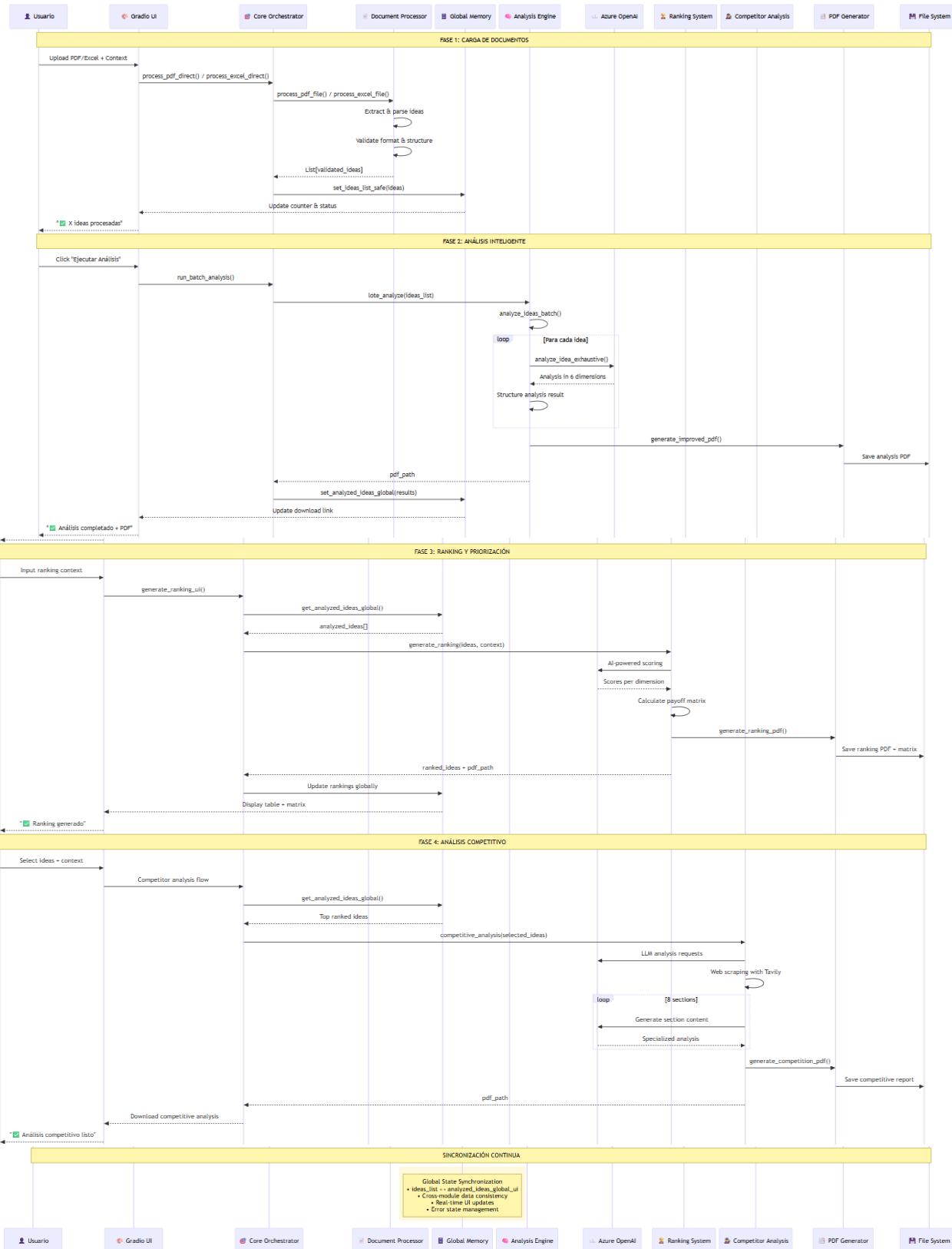
Rediseñar la arquitectura de scraping desacoplando adquisición, transformación y análisis, estableciendo una capa de orquestación robusta sobre LangChain (o alternativa como LlamaIndex), y una lógica centralizada de validación y persistencia de datos competitivos.

10. ANNEXOS

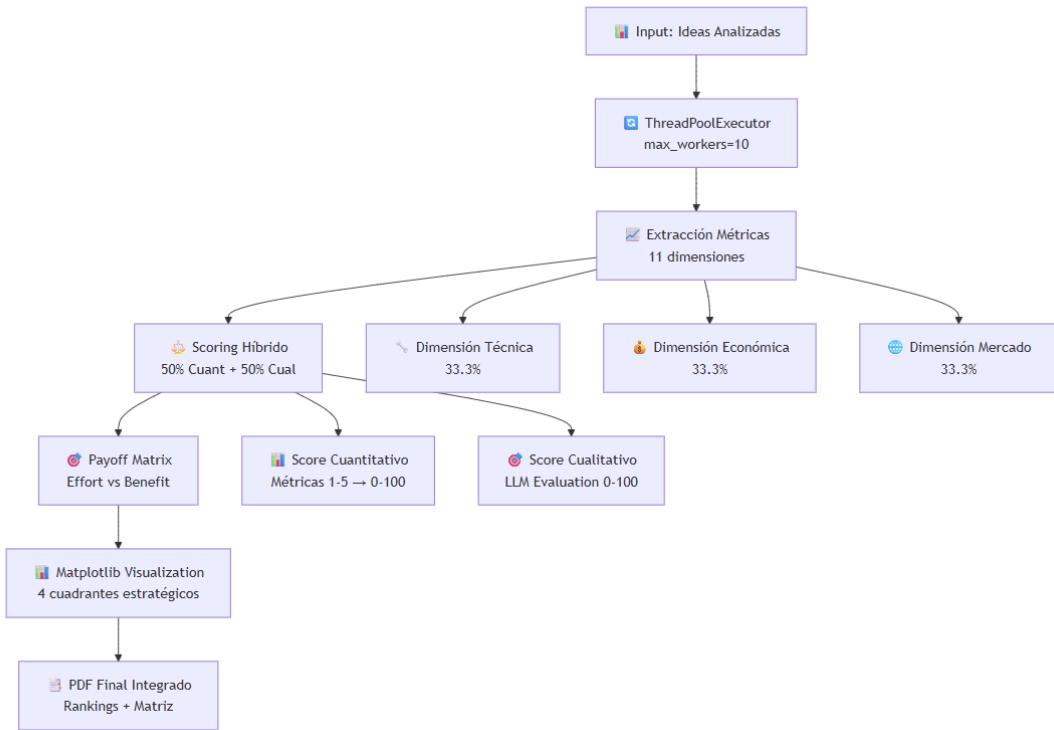
10.1 Diagrama de Componentes Principales



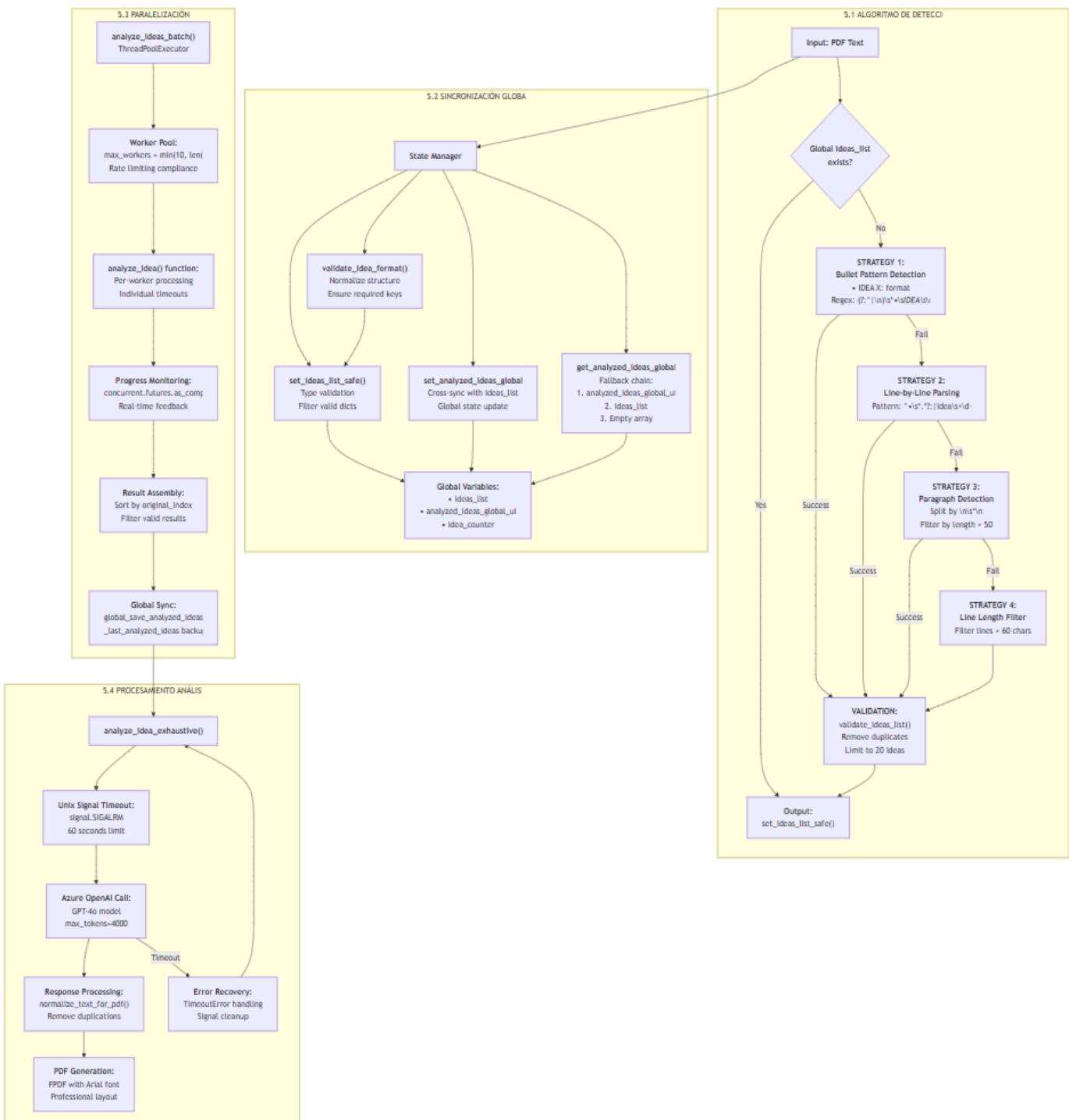
10.2 Flujo de Datos entre Componentes



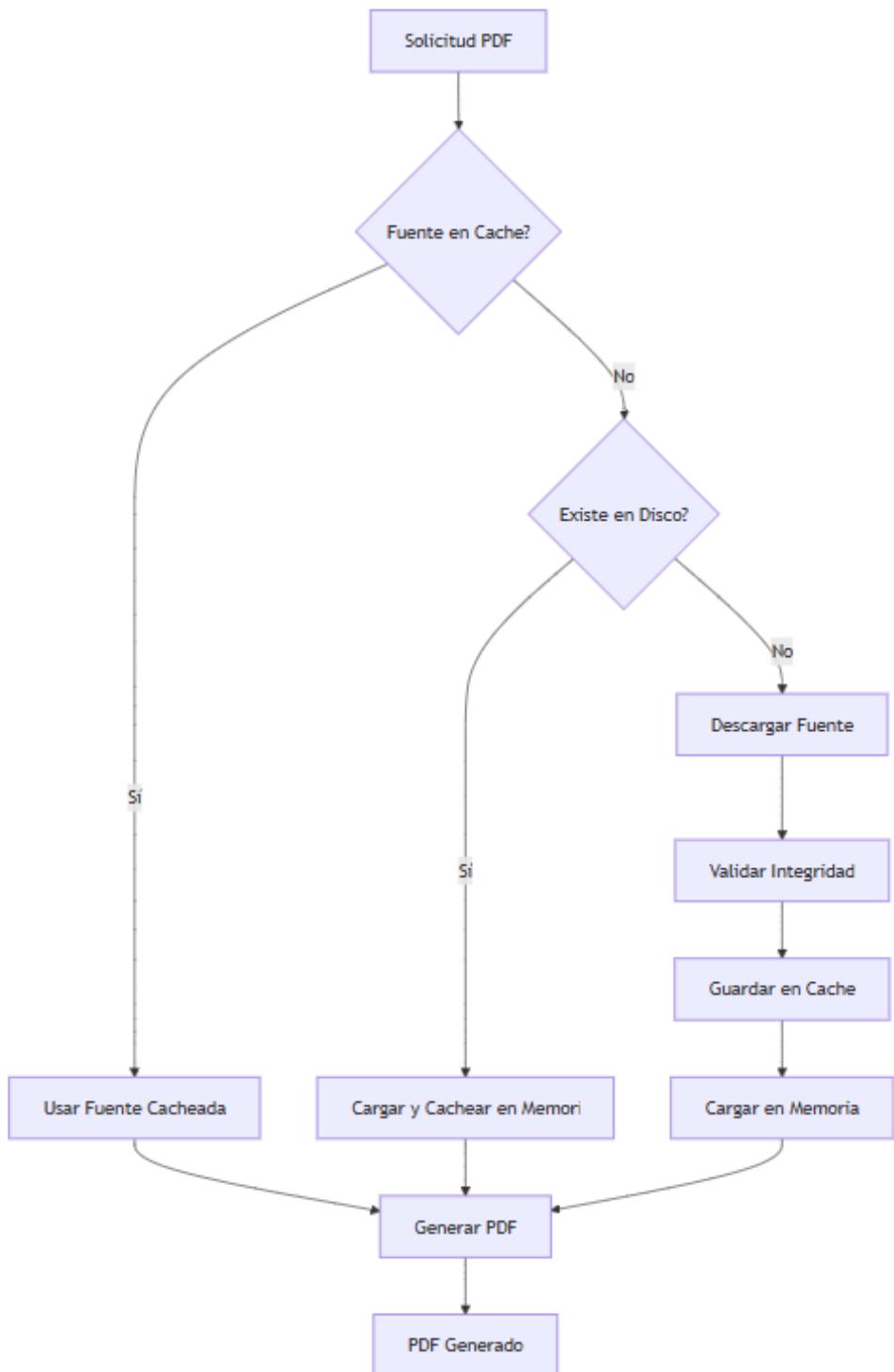
10.3 Flujo Módulo Ranking



10.4 Diagrama estructural del sistema algorítmico



10.5 Flujo Caché



10.6 Diagrama gestión memoria

