Informe de Sistema inteligente de investigación y redacción automatizada.

Asignatura: Ingeniería de Soluciones con Inteligencia Artificial (ISY0101)

Estudiante: Martín Carrasco

Descripción del caso:

En el panorama corporativo actual, los equipos de análisis y gestión de conocimiento se enfrentan a una carga de trabajo que es, paradójicamente, rutinaria pero crítica. A diario, existe una demanda constante de informes de investigación simples sobre temas específicos desde tecnologías emergentes hasta perfiles biográficos o respecto a cualquier tema de tu interés.

La naturaleza de estos informes es particular: se trata de documentos "simples pero detallados". No requieren necesariamente un análisis estratégico complejo de alto nivel, sino una recopilación rigurosa, factual y bien estructurada de datos clave.

Requerimientos

Autonomía: Capacidad de recibir un tema y ejecutar el ciclo de investigación sin intervención de uno, solo escribir el tema a buscar.

Calidad de Información: Uso de fuentes externas (Wikipedia en esta iteración)

Seguridad: Filtros de seguridad para evitar la generación de informes con contenido inapropiado o peligroso

Observabilidad: Trazabilidad completa de las acciones de los agentes y métricas de rendimiento.

Proximos Desafios

Coordinacion: Asegurar que el redactor utilice examen lo que el investigador encontró

Seguridad: Mejorar la seguridad en sí, analizar los logs de ejecución para anomalías. además la lista de security manager se puede evadir poniendo números en la búsqueda por ejemplo b0mba lo cual genera un informe pero del error.Agregar una etapa de validación post-ejecución donde un agente "Auditor" o un algoritmo verifique que el informe final no contenga datos sensibles, lenguaje ofensivo o alucinaciones evidentes antes de mostrárselo al usuario.

Memoria: Implementar una memoria de largo plazo, ya que la memoria solo dura la ejecución del script

Frontend: Mejorar el frontend a que sea visualmente más intuitivo y más Bonito

Frontend: Optimizar la interfaz de usuario para que sea más intuitiva y atractiva visualmente.

Diseño de la solución

Se diseñaron Prompts específicos para cada rol utilizando la metodología de asignación de Rol, Objetivo y Contexto disponible en el framework CrewAI:

- **Agente Investigador:**
  - Rol: "Investigador"
  - Prompt: "Experto en buscar datos y filtrar informacion".
- **Agente Redactor:**
  - Rol: "Redactor Profesional"
  - Prompt: "Escritor experto en transformar datos en reportes completos.".

Implementación de pipelines

La solución implementa un patrón RAG modular. En lugar de confiar en el conocimiento pre-entrenado del modelo, el sistema inyecta contexto externo mediante herramientas:

- **Recuperación:** El agente investigador utiliza la herramienta BuscadorWikipedia para extraer fragmentos de texto real.
- **Generación:** El agente redactor recibe estos fragmentos como contexto en su ventana de memoria para generar el informe final en Markdown.

Diseño de arquitectura

La arquitectura integra tres capas:

1. **Capa de Seguridad:** Clase SecurityManager que intercepta el prompt del usuario antes de procesarlo.
2. **Capa de Orquestación:** Framework CrewAI gestionando un proceso secuencial donde la salida del Agente Investigador es la entrada del Agente Redactor.
3. **Capa de Herramientas:** Conexión externa vía API (Wikipedia wrapper).

Justificación de decisiones de diseño

- **Modelo GPT-4o:** Elegido por su alta capacidad de razonamiento lógico para seguir instrucciones complejas de seguridad y formato, superior a modelos más pequeños.
- **Separación de Roles:** Dividir la tarea en dos agentes evita que el modelo "olvide" instrucciones; uno se enfoca solo en buscar y otro solo en escribir.

Desarrollo de Agnete Funcional

Integración de Herramientas

Se desarrolló una herramienta personalizada WikipediaSearchTool heredando de BaseTool. Esta herramienta encapsula la lógica de búsqueda y manejo de excepciones, asegurando que el agente pueda recuperarse de búsquedas fallidas sin que colapse el flujo.

Configuración de Memoria y Recuperación de Contexto

El sistema utiliza Memoria de Corto Plazo proporcionada por el framework. El contexto recuperado por el Investigador se pasa automáticamente al Redactor a través del Task Context. Esto asegura continuidad en el flujo de trabajo sin necesidad de gestión manual del historial de chat.

Estrategias de Planificación

Se implementó una estrategia de Planificación Secuencial

1. **Investigador:** Definir puntos clave, fechas y hechos.
2. **Dependencia:** La Tarea 2 no inicia hasta que la Tarea 1 finaliza con éxito.
3. **Redactor:** Estructurar el informe en Markdown basándose estrictamente en el output de la Tarea 1.

Orquestación y Flujo Automatizado

La orquestación se centraliza en la clase ResearchAssistantSystem y el flujo es: Usuario - Validación Seguridad - Crew (Investigador y Redactor) - Output Final - Logs.

Implementación de Observabilidad, Trazabilidad y Seguridad

Métricas de Observabilidad

Se implementó un sistema de logging robusto que captura métricas clave en tiempo real:

- **Latencia:** Medición del tiempo total de ejecución
- **Estado:** Registro de éxito, fallo técnico o bloqueo de seguridad.
- **Consumo:** Trazabilidad de tokens.

Análisis de Registros

Los registros se almacenan en agente_investigador.log. El análisis de estos logs permite identificar:

- Temas más buscados que generan errores de desambiguación.
- Intentos de uso malicioso (filtros de seguridad activados).
- Tiempos de respuesta anómalos para optimización de prompts.

Integración de Protocolos de Seguridad

Se aplicó un enfoque de Defensa en Profundidad con dos niveles:

1. **Filtrado Local:** Lista negra de términos prohibidos (hackear, bomba, terrorismo, etc.) que bloquea la solicitud antes de llamar a la API, ahorrando costos y protegiendo la ética.
2. **Manejo de Errores de Proveedor:** Captura de excepciones de Azure OpenAI para manejar bloqueos por políticas de contenido del modelo.

Conclusion

El sistema que se desarrolló cumple con todo lo esperado de un asistente inteligente moderno: puede realizar trabajo complejo (investigar y redactar) de forma autónoma, segura y completamente transparente. Su diseño modular nos permitirá agregar fácilmente nuevas funcionalidades en el futuro, como buscar en archivos PDF o añadir un agente traductor, lo que lo convierte en una base sólida para cualquier solución de negocio. Con este proyecto me quedo con cosas buenas y negativas. A pesar de haberme integrado tarde a la clase siento que aprendí bastante y me di cuenta de mis errores, siento al 100% que podría haber entregado un proyecto mucho mejor, siento que en la parte del frontend es donde más me faltó ya que siento que es muy simple.