

Desarrollo de un Agente Inteligente con Herramientas Integradas

Documentación Técnica

Programa de Especialización de INTELIGENCIA ARTIFICIAL GENERATIVA III

Paola Aida Soto Salazar

Índice

1. Descripción general del proyecto	2
2. Descripción de las herramientas utilizadas	2
3. Descripción de clases y funciones principales	2
4. Arquitectura	4
5. Flujo general del sistema	5
6. Ejemplo de interacción	6
7. Repositorio del Proyecto	7
8. Reflexión técnica y aprendizajes	7

1. Descripción general del proyecto

El proyecto proyV3.ipynb implementa un sistema inteligente de apoyo al proceso de registro y seguimiento de los exámenes médicos de los alumnos ingresantes. El sistema combina herramientas de inteligencia artificial, manipulación de datos y automatización para asistir tanto al personal médico como al administrativo durante las distintas etapas del examen médico: laboratorio, odontología, psicología y rayos X.

El núcleo del sistema se construye con un **agente inteligente basado en LangChain y LangGraph**, que interactúa con los usuarios para registrar, buscar y actualizar la información almacenada en archivos Excel.

2. Descripción de las herramientas utilizadas

- **Python 3.10+**: Lenguaje de programación principal para la implementación del sistema.
- **LangChain y LangGraph**: Frameworks especializados en la creación de agentes conversacionales con memoria, integrando modelos de lenguaje (LLMs) como OpenAI GPT para ejecutar acciones basadas en lenguaje natural.
- **OpenAI API**: Proporciona la inteligencia conversacional y el razonamiento contextual del agente.
- **Pandas**: Biblioteca fundamental para la manipulación, validación y escritura de datos en formato Excel.
- **SQLAlchemy y LangChain SQLDatabase**: Librerías que permiten la conexión con bases de datos SQL, previstas para futuras ampliaciones del sistema.
- **Pydantic**: Facilita la creación de modelos de datos robustos, garantizando la integridad y el tipo correcto de la información intercambiada.
- **Pathlib y os**: Permiten la manipulación segura de rutas de archivos y configuración de variables de entorno, como la API key.

3. Descripción de clases y funciones principales

3.1. Clase GestorExcel

Es el componente encargado de la **gestión de datos en archivos Excel**. Su objetivo es mantener actualizados los registros de los alumnos y sus exámenes médicos.

- `__init__(ruta_alumnos, ruta_examenes)`: Inicializa el gestor con las rutas de los archivos y verifica que el archivo de exámenes exista.
- `_crear_archivo_examenes_si_no_existe()`: Crea el archivo Excel de exámenes con las columnas necesarias si no existe previamente.
- `cargar_alumnos()`: Carga y normaliza la información de los alumnos desde el Excel de entrada.

- **registrar_examen()**: (En celdas posteriores del notebook) Registra un nuevo examen médico para un alumno, evitando duplicados.
- **buscar_alumno()**: Permite buscar un alumno en la base de datos de acuerdo con su código de matrícula.

3.2. Modelo Pydantic

Define la estructura del examen médico, asegurando que todos los campos (tipo de examen, resultado, estado, responsable, etc.) estén correctamente tipados antes de ser guardados o procesados.

3.3. Agente LangChain

El agente se construye con el método `create_react_agent()` de LangGraph, junto con una memoria persistente (`MemorySaver`). Este agente interpreta instrucciones en lenguaje natural y ejecuta las herramientas (`@tool`) que corresponden a operaciones como:

- Registrar nuevos exámenes.
- Consultar resultados por alumno.
- Actualizar estados de los exámenes.

3.4. Interfaz de usuario

En la versión actual, la interfaz es textual y se ejecuta dentro del entorno de Jupyter Notebook. Sin embargo, el diseño modular del agente permite su integración futura con aplicaciones gráficas (`Flet`, `Streamlit`) o chatbots integrados en la web.

4. Arquitectura

El sistema está diseñado con una arquitectura modular basada en agentes especializados que interactúan con herramientas definidas y una capa de datos persistente. A continuación se describe cada componente y su interacción:

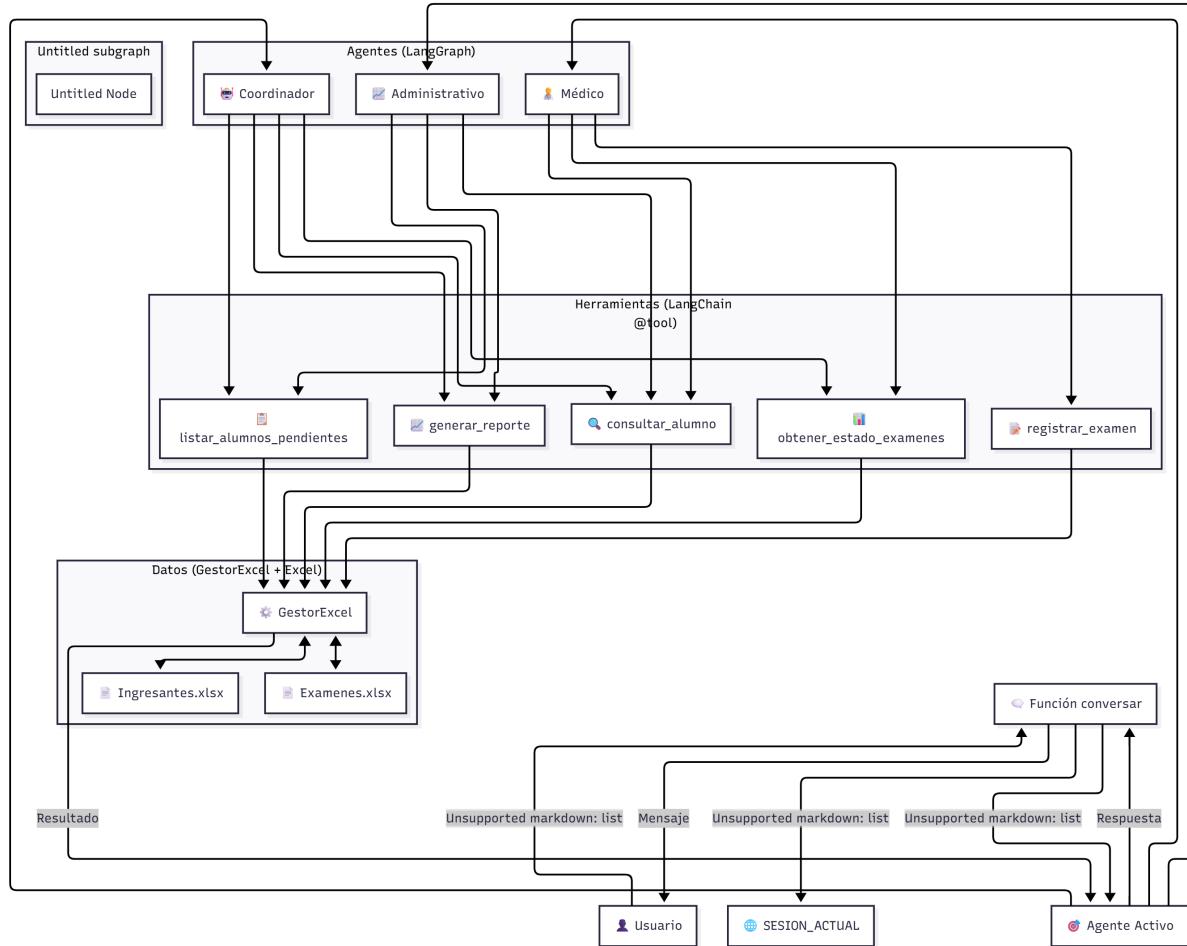


Figura 1: Diagrama de arquitectura

Interacción del Usuario

El flujo comienza con el **usuario**, quien realiza una consulta mediante la función principal **conversar**. Esta función verifica el estado de la sesión a través de la variable global **SESION_ACTUAL**, que determina qué agente está activo en ese momento.

Capa de Agentes (LangGraph)

Los agentes están organizados en una capa lógica que simula roles profesionales:

- **Coordinador:** Encargado de consultas generales y generación de reportes.
- **Médico:** Responsable del registro de exámenes clínicos.
- **Administrativo:** Especializado en gestión masiva y reportería.

Cada agente tiene acceso limitado a un conjunto de herramientas según su rol.

Capa de Herramientas (LangChain)

Las herramientas son funciones definidas con el decorador `@tool` y validadas con Pydantic. Entre ellas se encuentran:

- `consultar_alumno`: Busca alumnos por código, documento o apellido.
- `registrar_examen`: Registra o actualiza exámenes clínicos.
- `obtener_estado_examenes`: Muestra el progreso de un alumno.
- `listar_alumnos_pendientes`: Lista alumnos con exámenes faltantes.
- `generar_reporte`: Genera estadísticas agregadas.

Cada agente decide qué herramienta utilizar según la consulta recibida.

Capa de Datos (GestorExcel + Archivos Excel)

Las herramientas no interactúan directamente con los archivos Excel. En su lugar, utilizan la clase `GestorExcel`, que actúa como un ORM simplificado para manejar la lectura y escritura de datos en los archivos:

- `Ingresantes.xlsx`: Contiene datos personales de los alumnos.
- `Examenes.xlsx`: Registra los resultados de los exámenes.

`GestorExcel` garantiza la persistencia y consistencia de los datos compartidos entre agentes.

Flujo de Retorno

Una vez que el gestor procesa la solicitud, devuelve los resultados al agente correspondiente. El agente formatea la respuesta en lenguaje natural y la función `conversar` la entrega al usuario, cerrando el ciclo de interacción.

Resumen

Esta arquitectura permite una clara separación de responsabilidades, facilita la escalabilidad del sistema y asegura que cada agente opere dentro de sus límites funcionales, manteniendo la integridad de los datos y la lógica del negocio.

5. Flujo general del sistema

1. Carga de configuración y claves de API.
2. Inicialización del gestor de archivos Excel.
3. Validación de los datos de alumnos y exámenes.
4. Construcción del agente inteligente con memoria.
5. Interacción del usuario con el agente para registrar o consultar información.
6. Actualización de los resultados en los archivos Excel.

6. Ejemplo de interacciòn

```
conversar("Buscar el alumno con código 25010008")  
  
'He encontrado la información del alumno con código de matrícula 25010008:\n\n- **Nombre:** Joselyn Atencio Aguilar\n- **Tipo de Documento:** D.N.I.\n- **Número de Documento:** 71483771\n- **Fecha de Nacimiento:** 12 de marzo de 2004\n- **Sexo:** Femenino\n- **Código de Escuela:** 110\n- **Nombre de Escuela:** Medicina Humana\n- **Año Base:** 2025  
\n\n## Estado de Exámenes Medicos:\n- **Exámenes Completados:** 1\n- **Tipo de Examen:** Laboratorio\n- **Fecha de Examen:** 26 de octubre de 2025\n- **Resultado:** Resultados normales y fue evaluado por Dr. Pérez.\n- **Estado:** Pendiente de Evaluación\n- **Exámenes Pendientes:** 3\n\nSi necesitas más información o detalles adicionales, no dudes en preguntar.'  
  
# Cambiar a agente médico  
# Verificar estado de exámenes  
cambiar_agente("medico")  
conversar("¿Qué exámenes tiene registrados?")  
  
"Por favor, proporcioname el código de matrícula, número de documento o apellido del alumno para poder consultar los exámenes registrados."  
  
conversar("ok, su codigo es 25010008")  
  
'La alumna **Joselyn Atencio Aguilar** con código de matrícula **25010008** tiene registrados los siguientes exámenes:\n- **Exámenes Completados:** 1\n- **Tipo:** Laboratorio\n- **Fecha de Examen:** 26 de octubre de 2025\n- **Resultado:** Resultados normales y fue evaluado por Dr. Pérez.\n- **Estado:** Pendiente de Evaluación\n- **Exámenes Pendientes:** 3\n- **Tipo:** Psicológico\n- **Nombre:** Evaluación Psicológica\n- **Tipo:** Odontología\n- **Nombre:** Examen Odontológico\n- **Tipo:** Rayos X\n- **Nombre:** Rayos X de Tórax\nEl progreso del proceso médico es del **25%** y hay un examen completado de un total de cuatro. Actualmente, no hay exámenes que requieran seguimiento. Si necesitas más información o realizar alguna acción, no dudes en decírmelo.'
```

Figura 2: Ejemplo 1

ejemplo con otro alumno de codigo 25110777

```
cambiar_agente("medico")
conversar("Registra laboratorio para alumno 25110777:
Hemoglobina: 15.2 g/dL
Leucocitos: 7500/µL
Plaquetas: 250000/µL
Resultados normales
Estado: completado
Responsable: Dr. María Torres")
```

'La alumna **Thais Bretha Gomez Asencios** con código de matrícula **25110777** tiene los siguientes exámenes pendientes:\n\n- **Evaluación Psicológica**\n- **Rayos X de Tórax** (ya registrado)\nHasta el momento, ha completado un total de **2 exámenes** de un total de **4**. Los exámenes completados son:\n- **Laboratorio** - Resultado: Hemoglobina: 15.2 g/dL, Leucocitos: 7500/µL, Plaquetas: 250000/µL, Resultados normales\n- **Estado: Completado** - Responsable: Dr. María Torres\n- **Odontología** - Resultado: Higiene dental adecuada, sin caries. Se recomienda limpieza dental preventiva\n- **Estado: Completado** - Responsable: Dr. Ana Sánchez\nSi necesitas realizar alguna otra acción o tienes más consultas, no dudes en decírmelo.'

```
conversar("si, claro. registra los resultados dados del alumno 25110777")
```

'El examen de laboratorio para la alumna **Thais Bretha Gomez Asencios** con código de matrícula **25110777** ha sido registrado con éxito.\n\n### Detalles del examen registrado:\n- **Tipo de Examen:** Laboratorio\n- **Resultado:** Hemoglobina: 15.2 g/dL, Leucocitos: 7500/µL, Plaquetas: 250000/µL, Resultados normales\n- **Estado:** Completado\n- **Responsable:** Dr. María Torres\n\nSi necesitas realizar alguna otra acción o tienes más consultas, no dudes en decírmelo.'

```
conversar("Registra odontología para alumno 25110777:
Higiene dental adecuada, sin caries
Se recomienda limpieza dental preventiva
Estado: completado
Responsable: Dr. Ana Sánchez")
```

'El examen de odontología para la alumna **Thais Bretha Gomez Asencios** con código de matrícula **25110777** ha sido registrado con éxito.\n\n### Detalles del examen registrado:\n- **Tipo de Examen:** Examen Odontológico\n- **Resultado:** Higiene dental adecuada, sin caries. Se recomienda limpieza dental preventiva\n- **Estado:** Completado\n- **Responsable:** Dr. Ana Sánchez\n\nSi necesitas realizar alguna otra acción o tienes más consultas, no dudes en decírmelo.'

Figura 3: Ejemplo 2 parte 1

```

conversar(''Registra rayos_x para alumno 25110777:
Radiografía de tórax sin alteraciones
Campos pulmonares normales
Estado: completado
Responsable: Dr. Podesta'')

'El examen de rayos X para la alumna **Thais Bretha Gomez Asencios** con código de matrícula **25110777** ha sido registrado con éxito. \n\n### Detalles del examen registrado:\n- **Tipo de Examen:** Rayos X de Tórax\n- **Resultado:** Radiografía de tórax sin alteraciones. Campos pulmonares normales\n- **Estado:** Completado\n- **Responsable:** Dr. Podesta\n\nActualmente, la alumna ha completado un total de **2 exámenes** de un total de **4**. Los exámenes pendientes son:\n- Evaluación Psicológica\n- Rayos X de Tórax (ya registrado)\n\nSi necesitas realizar alguna otra acción o tienes más consultas, no dudes en decírmelo.'


conversar("al alumno 25110777 le falta pasar por algun examen?")

'La alumna **Thais Bretha Gomez Asencios** con código de matrícula **25110777** tiene los siguientes exámenes pendientes:\n- **Evaluación Psicológica**\n- **Rayos X de Tórax** (ya registrado)\nHasta el momento, ha completado un total de **2 exámenes** de un total de **4**. Los exámenes completados son:\n1. **Laboratorio**\n - Resultado: Hemo globina: 15.2 g/dL, Leucocitos: 7500/μL, Plaquetas: 250000/μL, Resultados normales\n - Estado: Completado\n - Responsable: Dr. María Torres\n2. **Odontología**\n - Resultado: Higiene dental adecuada, sin caries. Se recomienda limpieza dental preventiva\n - Estado: Completado\n - Responsable: Dr. Ana Sánchez\n\nSi necesitas realizar alguna otra acción adicional o tienes más consultas, no dudes en decírmelo.'


cambiar_agente("administrativo")
conversar("Genera reporte de los alumnos que hayan pasado por todos o al menos un examen médico")

'Aquí tienes el reporte general del estado actual y la lista de los primeros alumnos con exámenes pendientes:\n\nReporte General\n- **Fecha:** 26 de octubre de 2025\n- **Total de Alumnos:** 7,237\n- **Proceso Completo:** 0\n- **Porcentaje Completado:** 0.0%\n- **Requieren Seguimiento:** 0\n- **Exámenes Realizados por Tipo:**\n  - Laboratorio: 2\n  - Odontología: 1\n  - Rayos X: 1\nAlumnos con Exámenes Pendientes (Primeros 10):\n1. **Código de Matrícula:** 25010257\n  - **Nombre Completo:** GABRIELA ALESSANDRA ACUÑA TORRES\n  - **Escuela:** MEDICINA HUMANA\n  - **Exámenes Pendientes:** Laboratorio, Psicológico, Odontología, Rayos X\n  - **Cantidad Pendientes:** 4\n2. **Código de Matrícula:** 25010258\n  - **Nombre Completo:** ANYER ALCANTARA CABREJOS\n  - **Escuela:** MEDICINA HUMANA\n  - **Exámenes Pendientes:** Laboratorio, Psicológico, Odontología, Rayos X\n  - **Cantidad Pendientes:** 4\n3. **Código de Matrícula:** 25010001\n  - **Nombre Completo:** CAMIL...'

```

Figura 4: Ejemplo 2 parte 2

7. Repositorio del Proyecto

El código fuente, documentación y recursos adicionales del sistema están disponibles en el siguiente repositorio de GitHub:

<https://github.com/AidaPSS1/Proyecto1--IA-generativa>

Este repositorio contiene:

- Archivos del notebook para ejecución en Google Colab
- Scripts de configuración de agentes y herramientas
- Archivos de ejemplo en Excel para pruebas
- Instrucciones detalladas de instalación y uso

Se recomienda revisar el archivo README.md para comenzar rápidamente con la implementación.

8. Reflexión técnica y aprendizajes

El desarrollo del proyecto permitió reforzar los siguientes aprendizajes técnicos:

- **Integración de agentes LangChain:** La práctica con LangGraph demostró la importancia de estructurar el flujo conversacional mediante herramientas definidas, manteniendo un equilibrio entre autonomía del agente y control de operaciones.
- **Diseño modular y escalabilidad:** La separación de capas (datos, lógica, agente) facilita la extensibilidad hacia otros dominios o sistemas, como bases de datos SQL o interfaces web.
- **Validación estructural con Pydantic:** Garantiza la integridad de los datos antes de almacenarlos, previniendo errores comunes de formato en los archivos Excel.

Posibles mejoras futuras

- Integrar una **interfaz visual** (por ejemplo, con Flet o Streamlit) para la interacción con el agente.
- Conectar el sistema a una **base de datos relacional** (PostgreSQL o MySQL) para reemplazar los archivos Excel.
- Incorporar un **módulo de análisis estadístico** que genere reportes automáticos de desempeño y resultados médicos.
- Mejorar la **seguridad y privacidad** de los datos mediante autenticación de usuarios y encriptación de los archivos.