

manual orquestacion agentes ia

SIMULADOR DEPARTAMENTO FINANCIERO EN EMPRESA ALOJAMIENTO ESTUDIANTIL



15 de enero de 2026

**INTRODUCCIÓN**

El proyecto consiste en el desarrollo de un sistema multi-agente de IA que simula el comportamiento y está pensado para dar un soporte al departamento financiero de una empresa de alojamiento estudiantil.

Se ha establecido un criterio que explica la interacción lo más realista posible entre distintos roles del departamento financiero, unido con un modelo LLM, herramientas externas (tools), RAG para recuperar información y MCP.

La aplicación permite al usuario realizar consultas financieras en lenguaje natural y obtener respuestas especializadas gracias a la orquestación de los agentes de IA, es de manera local con Ollama y visualizando con Streamlit.

1. **TEORIA**

Os detallo unos conceptos básicos para entender el proyecto presentado:

**Agente IA** es un sistema autónomo capaz de percibir lo que tiene a su alrededor, razonar sobre cómo resolver el problema planteado y actuar para lograr una solución. A su mano tiene unas herramientas, también denominadas tools, que le permiten dar esa solución con autonomía.

El ciclo que sigue es: Pensar 🡪 Planificar 🡪 Ejecutar 🡪 Observar el Resultado.

**LangChain** es un marco de desarrollo (framework) de código abierto que conecta modelos de lenguaje (LLM – ChatGPT, Claude, Gemini…) con fuentes de datos externas y herramientas digitales para ejecutar acciones. Su función es orquestar secuencias líneas donde la salida de una instrucción sirve como entrada para la siguiente, dotando a la IA de “memoria” y contexto de conversaciones actuales y anteriores. Permite transformar un modelo generador de texto aislado en una aplicación funcional capaz de leer documentos privados, navegar por internet y procesar información.

**Tool (herramienta)** es una función o habilidad al agente de IA para que actúe. Los LLMs requieren de estas tools para poder tener acceso a internet, la hora actual, etc. La asociación entre ambos ocurre por las descripciones que le damos a esas herramientas para que le agente interprete de las herramientas que necesita para la tarea, pero sin necesidad de especificárselo.

**LangGraph** es una extensión avanzada construida sobre LangChain diseñada específicamente para gestionar flujos de trabajo cíclicos y agentes con estado persistente. A diferencia de las cadenas lineales, permite a la IA realizar bucles de razonamiento: planificar, ejecutar, verificar el resultado y corregirse a sí misma si hubo un error. Es la herramienta ideal para coordinar múltiples agentes de IA trabajando juntos y para procesos complejos que requieren mantener el contexto durante largos periodos de tiempo, a este concepto se le llama orquestación de agentes de IA.

**MCP (Model Context Protocol)** es un estándar técnico abierto que funciona como una interfaz universal para conectar asistentes de IA con repositorios de datos y herramientas de software. Elimina la necesidad de crear integraciones personalizadas para cada modelo, permitiendo que una conexión sirva para múltiples aplicaciones de IA diferentes. Garantiza una comunicación segura y estandarizada, facilitando que los modelos accedan al contexto necesario (archivos locales, bases de datos) para ser útiles.

**ROLES FUNCIONALES AGENTES**

**Director Financiero 🡪** Estrategia, supervisión y gestión de riesgos (no ejecuta tareas operativas. Delega análisis detallados al agente correspondiente y orquesta a los agentes).

**Controller 🡪** Contabilidad, control interno (no interpreta fiscalidad, no realiza previsiones futuras y no ejecuta pagos).

**FP&A Analyst 🡪** Presupuestos, forecasts y análisis (no garantiza exactitud de datos de origen, no toma decisiones, solo recomienda).

**Tesorero 🡪** Liquidez y pagos (no negocia condiciones de cobros con clientes. No decide sobre inversiones estratégicas).

**AR Manager 🡪** Facturación, cobros y control de ingresos (no condona deuda sin aprobación y no inicia acciones judiciales sin autorización).

**Fiscalista 🡪** obligaciones tributarias (no decide sobre estructura societaria sin aprobación y no gestiona contabilidad operativa).

**Gestor de activos 🡪** gestión de los activos y depreciaciones (no decide sobre compra o venta de activos, no realiza tasaciones, pero si solicita valoraciones externas si es necesario).

**FLUJO ENTRE AGENTES**

Diagrama, Esquemático

El contenido generado por IA puede ser incorrecto.

**Director Financiero:**

* Recibe: informes y alertas de todos los agentes.
* Envía: directrices y decisiones.

**Controller:**

* Recibe: facturas (AR), pagos (Tesorero), amortizaciones (Gestor
* Envía: datos a FP&A, Fiscalista y CFO

**FP&A Analyst**

* Recibe: datos reales del Controller y AR
* Envía: análisis y previsiones al CFO y Tesorero

**Tesorero**

* Recibe: previsión de cobros y forecasts
* Envía: pagos ejecutados y alertas de liquidez

**AR Manager**

* Recibe: confirmación de cobros.
* Envía: previsiones, facturas y alertas de impago

**Fiscalista**

* Recibe: estados financieros y amortizaciones
* Envía: ajustes fiscales y calendario de impuestos

**Gestor de Activos**

* Recibo: aprobaciones del CFO
* Envía: amortizaciones y valoraciones

**CONEXIONES CRÍTICAS**

1. **AR Manager 🡪 Tesorero:** previsión de cobros semanal
2. **AR Manager 🡪 Controller:** Facturas emitidas (diarios/semanal)
3. **Controller 🡪 FP&A:** Datos de cierre mensual
4. **Gestor de Activos 🡪 Controller:** Amortizaciones mensuales
5. **Tesorero 🡪 Controller:** Pagos ejecutados
6. **Todos 🡪 Director Financiero:** Alertas y escalados

**INSTALACIÓN Y PROGRAMACIÓN PASO A PASO**

Como queremos una versión gratuita, montaremos lo siguiente:

**Paso 1. INSTALACIÓN OLLAMA**

Ollama es una herramienta de IA que permite a los usuarios hacer uso de modelos de lenguaje.

A través del enlace <https://ollama.com/download> descargamos la versión para un ordenador con una RAM de mínimo 16GB y CPU moderna.

Mis características son:

* 32GB de RAM y un i9-13900H

Comprobamos que se haya instalado correctamente





**Paso 2. DESCARGA DEL MODELO**

Con el siguiente comando descargamos el modelo *qwen2.5:7b* de parámetros que es con el que correremos nuestros agentes. Este proceso puede durar unos cuantos minutos.



Una vez que se ha descargado el modelo, lo podemos ejecutar:



Compruebas que todo funciona correctamente y cuando acabes escribe /bye para salir.

Texto, Logotipo

El contenido generado por IA puede ser incorrecto.

Tenemos que verificar la versión del Python para ver que tenemos instalado (mínimo versión 3.10) y en caso de no tenerlo instalado, desde la web <https://www.python.org/downloads/>

**Paso 3. CREAR ENTORNO DEL PROYECTO**



Creamos la carpeta AgentesFinancieros en la ruta indicada.



Accedemos a la carpeta



Generemos un entorno virtual.



Necesitamos ejecutar ese comando para que nos permita utilizar los Scripts en un entorno virtual. Debes indicar “S”.



Ahora ya puedes trabajar en este entorno, si se quiere, se puede desactivar de nuevo con el comando anterior e indicar “N”.



**Paso 4. INSTALAR DEPENDENCIAS**

Necesitaremos las siguientes librerías para que funcione todo el proyecto, lo tendremos conjuntamente en un archivo *requirements.txt* o se pueden instalar cada librería una a una manualmente dentro de venv:



Librerías principales:

* LangChain
* LangGraph
* Streamlit
* Pandas
* MCP
* Ollama adapters

**Paso 5. CREAR ESTRUCTURA DEL PROYECTO**

Creamos las siguientes carpetas:

Texto

El contenido generado por IA puede ser incorrecto.

**Dentro de las carpetas agents y tools necesitamos generar un archivo \_\_init\_\_.py vacío para que reconozca que es un módulo importable.**

1. **EXPLICACION CÓDIGO**

A continuación, se explica el funcionamiento del sistema bloque a bloque, indicando una breve descripción, del por qué existe y cómo lo integramos con el resto del sistema.

**Archivo financial\_graph.py (BACKEND)**

Texto

El contenido generado por IA puede ser incorrecto.

Con este código estamos indicando el modelo de LLM estamos utilizando en la configuración.

Texto

El contenido generado por IA puede ser incorrecto.

AGENT\_CONFIG: Es un diccionario donde estamos definiendo la configuración de cada agente de IA dentro del sistema.

Para cada agente estamos especificando:

* Clave interna que se utilizará para identificarlo en el código (por ejemplo, director\_financiero)
* Nombre: identifica que se verá en el entorno visual.
* Icono: el dibujo o imagen que lo representará.
* System\_prompt: es el prompt que lo hace experto en…

Existe este bloque para:

* Separar lógica del negocio/departamento de la configuración restante.
* Evita tener que duplicar código al crear los agentes.
* Facilita la escalabilidad, porque con la misma plantilla se pueden hacer nuevos agentes.

Permite que cada agente esté personalizado en su rol.

Su integración en el sistema se hace:

* El system\_prompt se inyecta como “SystemMessage” en cada llamada al LLM.
* El nombre e icono se le devuelven a Streamlit para que lo muestre en la respuesta.
* La clave del agente (agent\_key) se usa en todo el grafo para enrutarlo correctamente.

Texto

El contenido generado por IA puede ser incorrecto.

Esta clase encapsula como definimos la estructura del estado global que se comparte entre todos los nodos del grafo de LangGraph.

Existe porque LangGraph trabaja con un estado persistente, que se va actualizando conforme los nodos (agentes) se van ejecutando en el proceso.

Actúa como una memoria común que permite que los agentes tengan un histórico de la conversación, saber que agente está interviniendo, decir que agente de actuar en el siguiente paso.

AgentState se pasa para construir el grafo (StateGraph(AgentState)), que garantiza coherencia en todo el flujo.

Texto

El contenido generado por IA puede ser incorrecto.

Esta función es la que inicializa el modelo de LLM que llamamos con Ollama en local y que utilizarán todos nuestros agentes de IA.

Existe porque centraliza la creación del LLM en una única función que permite:

* Reutilizar el código para configurar cada agente actual como futuros.
* Se evita duplicidad de código.
* Es fácil de sustituir el modelo que se utiliza sin tocar el resto.

Para integrarlo, esta función se llama cada vez que se crea un agente dentro del grafo, así existe una consistencia entre ellos.

El parámetro *temperature* se le indica el grado de creatividad que del modelo. Cuanto más bajo más determinista y cuando más alto más creativo.

Se indica 0.7 para que tenga respuestas flexibles, sin perder el rigor que se necesita en el área financiera.

Texto

El contenido generado por IA puede ser incorrecto.

Esta función define la creación de un agente de IA completamente funcional, incluye:

* Modelo de LLM que utilizará
* Las herramientas propias que va a tener (agent\_key)
* Herramientas RAG (si fuese necesario su uso)
* Herramientas MCP (si fuese necesario su uso)

Existe permite:

* Centralizar la lógica de creación de agentes
* Evita duplicidad en el código
* Garantiza que cada agente tenga las herramientas que necesita y llamarlas.

Esta función es invocada cada vez que se crea un nodo de agente en LangGraph

El agente se convierte en un nuevo nodo del grafo.

Texto

El contenido generado por IA puede ser incorrecto.Captura de pantalla de un celular

El contenido generado por IA puede ser incorrecto.

Parte por parte el código:



Esta función define el nodo supervisor del grafo para decidir que agente financiero debe responder.

El nodo no genera la respuesta final, sino que hacer de controlador del flujo.

Existe en un sistema multi-agente, porque:

* No todos los agentes deben responder cada interacción.
* Para hacer un enrutamiento inteligente

Se integra siendo el primer nodo del grafo y siempre se ejecuta antes de asignar el agente especializado.



Extrae el último mensaje del usuario desde el estado compartido del grafo.

Existe porque el supervisor dependiendo de la última consulta, no de todo el histórico.

Se integra con *message* que se ha definido previamente en AgentState.

Texto

El contenido generado por IA puede ser incorrecto.

Esta función sirve para crear el nodo agente.

Existe porque facilita la creación y permite tener una uniformidad entre ellos que facilita el poder incrementar agentes.

Se usa al construir el grafo para crear un nodo por cada agente financiero.

Texto

El contenido generado por IA puede ser incorrecto.

Esta función define el router que decide con cierto dinamismo qué nodo del grafo debe ejecutarse a continuación, basándose en el estado compartido (AgentState).

Existe para que LangGraph permita definir edges condicionales, pero necesita una función que:

* Lea el estado
* Devuelva una clave
* Use usa clave para decidir la transición

Se integra utilizando el grafo como router y conectando el nodo supervisor con los distintos nodos de los agentes especializados.

Texto

El contenido generado por IA puede ser incorrecto.

Se define la función que se encarga de construir el grafo completo (workflow) de agentes y devolverlo ya compilado, listo para ejecutarlo.

Existe por que unifica:

* La creación de nodos
* La definición de edges
* El punto de entrada
* La compilación final

La integración se experimenta ya que todos los nodos del grafo reciben y devuelven este estado.

Texto

El contenido generado por IA puede ser incorrecto.

Esta función define que:

* Construye/obtiene el grafo
* Prepara el estado inicial
* Ejecuta el workflow completo
* Devuelve el resultado en formato útil para la UI

Existe para separar la parte lógica de orquestación (LangGraph) de la capa de presentación (Streamlit). Se encapsula todo el proceso y permite que Streamlit solo tenga que llamar un único método.

Se integra llamándola desde app.py cuando el usuario escribe una consulta en el chat o selecciona un agente.

**Archivo app.py (FRONTEND)**

Texto

El contenido generado por IA puede ser incorrecto.

Se importan todas las librerías y funcionalidades que requerimos para que el proyecto funcione en Streamlit.

import streamlit as st 🡪 para crear la interfaz web

import sys 🡪 gestionar rutas del proyecto

import os 🡪 gestionar rutas del proyecto

from datetime import datetime, timedelta 🡪 para filtros de temporalidad, análisis de cobros, cálculos por vencimiento, dashboards

import pandas as pd 🡪 manipulación de dataframe y extracción de información.

import numpy as np 🡪 manipulación de números

import io 🡪 generación de archivos de memoria

import re 🡪 tratamiento de texto, validaciones.

import plotly.express as px 🡪 representación gráfica y dashboards

import plotly.graph\_objects as go 🡪 representación gráfica y dashboards

from plotly.subplots import make\_subplots 🡪 representación gráfica y dashboards

from graphs.financial\_graph import run\_agent\_query, AGENT\_CONFIG, MODEL\_NAME 🡪 para ejecutar las funciones definidas en el fichero graphs.financial.py

Existen para que se pueda procesar el código y representar la información o valores que queremos plasmar.

Captura de pantalla de computadora

El contenido generado por IA puede ser incorrecto.

Sirve para configurar la app, la parte frontal (tema, layout, título).

Texto

El contenido generado por IA puede ser incorrecto.

st.session\_state es el mecanismo que utilizamos con Streamlit para que la información circule entre interacciones del usuario.

Streamlit ejecuta el script completo cada vez que el usuario:

* Escribe un input
* Pulsa un botón
* Cambia de pestaña

Sin este comando, se perdería en cada interacción con el entorno.

Texto

El contenido generado por IA puede ser incorrecto.

Con este decorador se encapsula la función cargar\_datos. Ayuda a que se carguen en caché los archivos csv para evitar leerlos cada vez que el usuario hace clic en un botón o hace una solicitud sobre ellos.

Esto se justifica porque optimiza el rendimiento de la app.

Captura de pantalla de computadora

El contenido generado por IA puede ser incorrecto.

Esta función se configura el entorno visual que tendremos al darle al botón.

Se muestra como se transforman los datos en *Pandas* en información visual con *Plotly*.

* Se destaca cómo se calculan las métricas en tiempo real
* Se menciona el uso de plotly.graph\_objects y plotly.express para los gráficos interactivos.

Texto

El contenido generado por IA puede ser incorrecto.

Esta función se configura como se muestra el chat con los agentes.

* St.session\_state sirve para guardar el historial del chat, para evitar que se borre.
* Run\_agent\_query(prompt\_to\_send, agente) es donde el Streamlit llama al grafo que se ha definido en el archivo financial\_graph.py
* St.spinner se configura como se visualiza los iconos del agente específico que responde.

Texto

El contenido generado por IA puede ser incorrecto.

Con esta función se puede exportar en PDF. Para ello se requiere de la librería fpdf.

Existe para facilitar la exportación de la información en este formato.

Texto

El contenido generado por IA puede ser incorrecto.

Esta función se define lo que vamos a ver en el entorno principal de la app.

Existe para poder llevar un control sobre lo que visualizaremos y configurarlo a nuestro modo.

Se integra como controlador maestro. Orquesta todas las visualizaciones que hemos definido previamente.

**SERVIDOR MCP (DATA LAYER)**

**Archivo collections\_server.py**

**Archivo financial\_data\_server.py**

Texto

El contenido generado por IA puede ser incorrecto.

Esta función sirve para cargar los archivos CSV desde la carpeta /data.

Existe porque ayuda a centralizar la conexión con la base de datos. Al usar load\_csv, evitamos repetir la lógica de rutas en cada herramienta, haciendo que el código quede más limpio y fácil de mantener.

Se integra siendo la base del resto de herramientas. Si no existiese el MCP no podría arrancar ni registrar.

Texto

El contenido generado por IA puede ser incorrecto.

Con este decorador se insta el listado de herramientas disponibles para los agentes. Se define el nombre, la descripción que le damos, el esquema de como lo utilizaremos y que se necesita para llamarla.

Existe por que es una obligación para los MCP. Los LLMs necesitan saber qué pueden pedir y cómo deben pedirlo antes de que se haya ejecutado algo.

Su integración es cuando se formula una pregunta, este busca y le responde.

Texto

El contenido generado por IA puede ser incorrecto.

Este decorador es esencial para el funcionamiento. Es quien recibe la petición del agente, ejecuta la parte lógica usando la librería que corresponda y lo devuelve en un JSON con el resultado que después es transformado para que sea en lenguaje natural.

Existe para separar la definición de la ejecución es un clásico en los MCP.

Se integra en el orden siguiente: recibe la orden del Agente 🡪 procesa los datos 🡪 devuelve la respuesta.

Texto

El contenido generado por IA puede ser incorrecto.

Esta función arranca el servidor utilizando la entrada/salida estándar (STDIO) para la comunicación.

Existe porque el MCP funciona mediante el protocolo cliente-servidor.

Se integra desde el punto de entrada que permite que se conecte el microservicio al grafo principal o a cualquier cliente MCP compatible.

**Archivo mcp\_client.py**

Texto

El contenido generado por IA puede ser incorrecto.

Con esta función definimos como queremos que salgan los valores de los importes. Es necesario porque así nos aseguramos que todos tienen el mismo formato.

Texto

El contenido generado por IA puede ser incorrecto.

Este decorador está encapsulando la función donde se replica la lógica definida al leer los CSVs.

Existe porque permite al agente “Tesorero” o “Controller” obtener los datos financieros estructurados en un JSON.

Esto se integra en formato JSON string para que el LLM lo entienda mejor para dar la respuesta.

Texto

El contenido generado por IA puede ser incorrecto.

Este decorador encapsula una función que nos da la lógica utilizada para poder filtrar los morosos. Se utiliza para calculas los días de retraso en tiempo real.

Existe porque es el núcleo operativo del agente “AR Manager” para responder a las dudas sobre la morosidad.

Se integra desde una exportación en la lista MCP\_COLLECTIONS\_TOOLS, que luego se importa en financial\_graph.py para asignárselas al agente correspondiente.

Texto

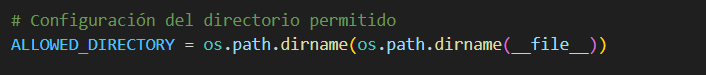
El contenido generado por IA puede ser incorrecto.

Encapsula la función para el calculo de los intereses comparando con el mercado.

Permite al Director Financiero dar consejos basados en datos actuales.

Se provee de datos macroeconómicos a las respuestas. Cuando el usuario pregunta sobre hipotecas, el agente consulta y ve que el Euribor está bajando, el agente recomienda acciones.

**Archivo third\_party\_mcp.py**



Esta constante es donde configuramos las rutas raíz del proyecto y validación de seguridad.

Existe para dar capacidad a la IA para leer los archivos del ordenador, es crítico porque da seguridad para restringir el acceso a la información. Este bloque hace que el agente solo pueda leer los archivos de dentro de la carpeta del proyecto y no pueda acceder a otro lugar.

Actúa como un middleware de seguridad. Todas las funciones que se indican posteriormente verifican primero que la ruta solicitada comience por el prefijo autorizado para ejecutar la acción.

Texto

El contenido generado por IA puede ser incorrecto.

Este decorador está encapsulando una herramienta para tener el directorio de todas las rutas.

Existe para cumplir el requisito de integración con herramientas de terceros. Permite capacidades al “Director Financiero” verificar si existen los archivos CSV de datos (/data) o comprobar si se ha generado un reporte PDF correctamente.

Devuelve un JSON estructurado que el LLM utiliza para ubicarse en el entorno y poder navegar por las carpetas antes de decidir que hacer archivo necesito leer.

Texto

El contenido generado por IA puede ser incorrecto.

Este decorador tiene una función que da una herramienta de lectura de contenido. Limita la lectura de los ficheros para no desbordar al LLM.

Existe para permitir al agente leer información no estructurada o logs que no están cubiertos con el resto de las herramientas financieras. Es la capacidad de “read” del MCP.

Para su integración se utiliza un conjunto con filesystem\_search\_files. Si el agente necesita buscar un dato en un archivo de configuración o en un README.md, utiliza esta herramienta para extraer el texto crudo.

Texto

El contenido generado por IA puede ser incorrecto.

En una variable se lista las herramientas de exportación. Se agrupan las 4 funciones de sistema de archivos en una lista única para ser inyectada.

Existe para facilitar el modularidad. En lugar de una a una, el grafo importa el conjunto de capacidades.

Para integrarse, el archivo financial\_graph.py, estas herramientas las asigna al “Director Financiero”. Esto le otorga al rol permisos de alto rango para auditar el sistema, diferenciándolo de los roles operativos que solo ven los datos del negocio.

**RAG (DATA LAYER)**

**Archivo rag\_system.py**

Texto

El contenido generado por IA puede ser incorrecto.

Con esta clase se implementa un motor de búsqueda basado en el algoritmo BM25, en lugar de configurar una base de datos vectorial compleja que requiere dependencias muy pesadas.

Existe para mayor portabilidad, y eficiencia.

Para integrar, se instancia un objeto global rag\_system = RAGSystem() al final de la clase, que actúa como un Singleton para mantener el índice cargado en memoria y disponible para todas las herramientas.

Texto

El contenido generado por IA puede ser incorrecto.

Esta función se utiliza para dividir el texto. No corta el texto arbitrariamente por número fijo de caracteres. Utiliza la estructura del Markdown para separar el contenido por secciones temáticas.

Existe porque al dividir por encabezados, se asegura que cada fragmento de información que recupera el agente tenga sentido completo.

Texto

El contenido generado por IA puede ser incorrecto.

Con estos decoradores @tool se definen las funciones a LangChain. Se envuelve la funcionalidad de búsqueda de la clase RAGSystem en funciones decoradas y se le añaden instrucciones para el LLM.

* Buscar\_normativa: herramienta para el fiscalista.
* Buscar\_procedimiento\_cobros: herramienta para el AR Manager.

Se integran estas funciones devolviendo un string formateado con los resultados más relevantes. El agente recibe este texto y lo utiliza para generar una respuesta fundamentada, así se evitan alucinaciones sobre leyes o plazos.

**AGENTS (BUSINESS LOGIC)**

**Archivo ar\_manager\_tools.py**

Texto

El contenido generado por IA puede ser incorrecto.

Se encapsulan dentro del decorador @tool ciertas funciones.

En este caso, la función definida es consultar\_facturas, donde se configura como leer el archivo csv y como mostrar los resultados.

Esto ayuda a establecer un resultado más fiel y ordenado para facilitar la extracción y rellenado al agente.

Texto

El contenido generado por IA puede ser incorrecto.

En este caso, la función definida es consultar\_morosos, donde se configura como leer el archivo csv y como mostrar los resultados. Todas las facturas que tienen pendiente los clientes y ordenar por por días de retraso y los datos de contacto para su reclamación.

Es importante para tener la información necesaria e iniciar actos legales en caso de ser necesario y que no se llegue a una resolución amistosa.

Texto

El contenido generado por IA puede ser incorrecto.

En este caso, la función definida es consultar\_estudiantes donde se configura como leer el archivo csv y como mostrar los resultados. Todos los datos de los estudiantes se muestran en el formato que se ha preestablecido en esta función.

Facilita un orden para que sea más sencillo ver la información.

Texto

El contenido generado por IA puede ser incorrecto.

Esta función genera un informe de Antigüedad de deuda, clasificando los impagos en cubos temporales, de 1-30 días, 31-60 días, >90 días.

Es importante, porque ayuda a calcular la provisión de insolvencia. La función devuelve una tabla pre-formateada que el Agente puede presentar directamente al Director Financiero sin necesidad de post-procesado.

**Archivo controller\_tools.py**

Texto

El contenido generado por IA puede ser incorrecto.

Es una función auxiliar. Implementa una capa de robustez. Esta función estandariza las etiquetas antes de sumar, garantizando que el total del Activo sea correcto sin importar cómo se introdujeron los datos.

Texto

El contenido generado por IA puede ser incorrecto.Texto

El contenido generado por IA puede ser incorrecto.

Estas funciones encapsuladas en el decorador calculan las partidas derivadas, si el Patrimonio Neto no está explícito en el CSV, lo deduce por la ecuación fundamental de la contabilidad (Activo = Patrimonio Neto + Pasivo)

Entrega el informe en el formato establecido en la función listo para presentar, con tablas alineadas y verificación de cuadre incluida.

Texto

El contenido generado por IA puede ser incorrecto.

La función encapsulada cruza los datos del Balance y de la Cuenta de Resultados para generar KPIs complejos:

* Liquidez (solvencia a corto plazo)
* Endeudamiento (% de deuda sobre activos)
* ROE (Rentabilidad para el accionista)

Se ha incluido un sistema de semáforos comparando el dato calculado contra benchmarks del sector bancario español. Esto aporta valor añadido para interpretar.

**Archivo director\_financiero\_tools.py**

Texto

El contenido generado por IA puede ser incorrecto.

Esta función encapsulada actúa como un “hub” de información. En una ejecución, lee 4 archivos CSV distintos.

Existe para calcular KPIs cruzados que ningún departamento individual tiene.

Su integración devuelve un informe en Markdown estructurado que el LLM puede presentar directamente al usuario, proporcionando la “foto fija” del estado de la empresa.

Texto

El contenido generado por IA puede ser incorrecto.

Esta función encapsulada genera un resumen ejecutivo listo para presentar al Consejo de Administración haciendo un informe resumido. Calcula métricas de alto nivel como la Solvencia (Patrimonio / Activo)

Existe para demostrar la capacidad de la IA para realizar “Reporting Cualitativo”. No solo vuelca número, sino que estructura recomendaciones estratégicas basadas en los datos de ocupación y resultados.

**Archivo fiscalista\_tools.py**

Texto

El contenido generado por IA puede ser incorrecto.

Esta función encapsulada, lee el archivo de obligaciones\_fiscales.csv para presentar un Calendario Fiscal dinámico.

Utiliza lógica condicional para mostrar el estado de cada modelo con icono visuales (verde para presentado y rojo para pendiente)

La integración de esta herramienta permite al Director Financiero preguntar y recibir una lista precisa de los modelos a presentar.

Texto

El contenido generado por IA puede ser incorrecto.

Esta función encapsulada simula la liquidación trimestral del IVA aplicando la fórmula básica:

IVA a Ingresar = IVA Repercutido – IVA Soportado

Existe porque da robustez al código porque get\_col lo hace resistente a cambios .

* Ayuda a visualizar el resultado rápidamente y a facilitar el pago a las a la administración correspondiente o la reclamación.

**Archivo fpa\_analyst\_tools.py**

Texto

El contenido generado por IA puede ser incorrecto.

Esta función encapsulada calcula el porcentaje de ocupación en tiempo real por residencia y el total ponderado. Da un detalle visual implementando semáforos visuales (verde >90% ocupación excelente, amarillo 75-90% alerta preventiva, rojo <75% situación crítica.

Texto

El contenido generado por IA puede ser incorrecto.

Esta función encapsulada lee el archivo kpis.csv y evalúa el cumplimiento de objetivos.

Es importante, porque incluye una lógica condicional avanzada para determinar el éxito o fracaso según la naturaleza del indicador.

Texto

El contenido generado por IA puede ser incorrecto.

Esta función encapsulada realiza el análisis de varianza. Cruza el Presupuesto (Budget) con el Real y calcula el porcentaje de desviación.

Su integración provee al Director Financiero de la información necesaria para justificar correcciones. Clasifica las desviaciones en tres niveles de gravedad (<5%, 5-10%, <10&) permitiendo “gestión por excepción”.

**Archivo gestor\_activos\_tools.py**

Texto

El contenido generado por IA puede ser incorrecto.

Esta función nos sirve para consultar los activos fijos que tenemos en la actualidad en la empresa y ver que está amortizando, el valor por el que se adquirió, valor restante de amortizar.

Es importante, porque nos ayuda a ver que activos tenemos a punto de llegar a su vida útil y necesario para el EBITDA.

Texto

El contenido generado por IA puede ser incorrecto.

Esta función nos muestra el resultado de la amortización mensual de los activos.

Texto

El contenido generado por IA puede ser incorrecto.

Esta función nos permite ver los costes que pueden ocasionar nuestros activos y así nos podamos anticipar a posibles gastos extraordinarios. Por ejemplo, si hay una reparación costosa para el mes que viene, el sistema avisa al Tesorero para que prepare liquidez.

Texto

El contenido generado por IA puede ser incorrecto.

Esta función permite al Tesorero tener la información de las cuentas de un vistazo sin necesidad de entrar cuenta a cuenta para así planificarse y ayudar a preparar el informe de tesoreria.

Texto

El contenido generado por IA puede ser incorrecto.

Esta función, consulta los pagos pendientes y los clasifica según cercanía y prioridad con un sistema de semáforos.

Existe para que el Tesorero puede anticiparse a todos los posibles pagos y poder tener liquidez suficiente para hacer frente a todos ellos.

Texto

El contenido generado por IA puede ser incorrecto.

Esta función es otra parte de la información que necesita el Tesorero para hacer el informe de tesorería.

Texto

El contenido generado por IA puede ser incorrecto.

Esta herramienta calcula cuántos meses de vida tiene la empresa con el dinero actual, si el resultado es menor de 2 meses señala un semáforo rojo para que el Tesorero pueda informar al Director Financiero de la necesidad de liquidez y tomar medidas.

Existe como modo de alerta.

La integración sirve para trabajar junto al AR Manager y ver si se puede adelantar algún cobro pendiente de los clientes o recuperar de aquellos morosos.

**Archivo utils.py**

Texto

El contenido generado por IA puede ser incorrecto.

Estas tres funciones centralizan el formato de visualización de datos numéricos y monetarios.

Existe porque Python por defecto trabaja con el formato inglés y nuestra información se trabaja en formato español (sepador de miles “.” y separador de decimales “,”).

Se integra dentro de este fichero y se rescata en otros ficheros porque en caso de algún cambio, solo se debe hacer en un único fichero.

**DESPLIEGUE EN STREAMLIT**

Ejemplo de dashboard principal:

Interfaz de usuario gráfica

El contenido generado por IA puede ser incorrecto.

Ejemplo de prompt:

“¿Cuál es el estado general de la empresa? Necesito un resumen para el consejo”

Texto

El contenido generado por IA puede ser incorrecto.

Texto

El contenido generado por IA puede ser incorrecto.

Interfaz de usuario gráfica, Texto, Aplicación

El contenido generado por IA puede ser incorrecto.

Texto

El contenido generado por IA puede ser incorrecto.

