**1. Elección de componentes**

* **Interfaz de Usuario (Web/App)**:  
  Proporciona acceso al sistema desde navegadores o dispositivos móviles. Es el punto de interacción del usuario con el sistema.
* **Módulo de Planificación de Rutas**:  
  Procesa los datos topográficos y de fertilización para generar rutas óptimas para los drones.
* **Drones con Sensores**:  
  Son la parte operativa, ejecutan las rutas y capturan información del campo.
* **Módulo de Análisis de Efectividad**:  
  Procesa la información que envían los drones para determinar si la distribución fue eficiente.
* **Base de Datos (PostgreSQL)**:  
  Centraliza el almacenamiento de mapas, configuraciones, rutas y resultados.
* **Interfaz de Monitoreo en Tiempo Real**:  
  Permite observar el estado y posición de los drones durante la operación.
* **Sistema de Monitoreo de Cultivos (Externo)**:  
  Fuente de datos adicional para enriquecer la planificación de rutas y mejorar la precisión de fertilización.

**2. Integración entre componentes**

La comunicación entre los módulos sigue un flujo **semi-lineal**, pero con soporte para datos en tiempo real:

1. El **usuario** solicita la planificación desde la **UI**.
2. El **Planificador** consulta la **Base de Datos** y el **Sistema Externo** de monitoreo para obtener información actualizada.
3. El **Planificador** envía las rutas a los **Drones**.
4. Los **Drones** envían datos al **Módulo de Análisis** y actualizan la **Base de Datos**.
5. Los **Drones** también transmiten información en vivo a la **Interfaz de Monitoreo**.

**3. Tipo de arquitectura**

Opté por una **arquitectura basada en microservicios** y no por un sistema monolítico, por las siguientes razones:

* **Escalabilidad**:  
  Permite escalar cada módulo según la carga (por ejemplo, más instancias del módulo de planificación en temporadas altas).
* **Flexibilidad tecnológica**:  
  Podemos implementar la interfaz en JavaScript/React, el backend en Python, y mantener servicios independientes para análisis y planificación.
* **Resiliencia**:  
  Si un servicio falla (por ejemplo, el módulo de análisis), el resto del sistema puede seguir funcionando parcialmente.
* **Facilidad de integración con servicios externos**:  
  Los microservicios facilitan conectarse con APIs como la del sistema de monitoreo de cultivos.

**4. Justificación**

En un proyecto agrícola con drones, el tiempo de respuesta y la confiabilidad son críticos. La elección de microservicios, un flujo de datos en tiempo real y una separación clara de responsabilidades permite:

* Minimizar riesgos ante fallos.
* Actualizar módulos de forma independiente.
* Reducir el tiempo de desarrollo y pruebas.
* Integrar nuevos sensores o sistemas externos en el futuro sin reescribir todo.