Arquitectura recomendada:

[Sensores/Actuadores - Arduino] ←→ [MODbus esclavo]

↑

[Raspberry Pi 4] ←→ [MODbus maestro + OPC Server]

↓

[PC] ←→ [OPC Client (Python)] ←→ [Broker MQTT]

↓

[Web App (interfaz con usuario final)]

**✅ Ejemplo de flujo funcional:**

1. **Arduino (esclavo MODbus)** mide temperatura, humedad y tiene una salida digital.
2. **Raspberry Pi** corre un **maestro MODbus** que consulta al Arduino cada 5 segundos y guarda los datos localmente.
3. Ese mismo dispositivo funciona como **OPC Server** (puede usarse OpenOPC, Kepware, etc.), que publica los datos.
4. **En una PC**, un **cliente OPC en Python** se conecta al servidor y obtiene los datos.
5. Con esos datos, los publica al **broker Mosquitto (MQTT)** usando paho-mqtt.
6. **Una página web** se suscribe a los tópicos MQTT, muestra valores y permite enviar comandos al broker (encender/apagar salida, variar PWM).
7. Esos comandos se procesan en la Raspberry Pi, que actualiza la salida vía MODbus → Arduino.

**⚠️ Requisitos clave a implementar**

* Monitoreo de errores de conexión (cada bloque debe notificar caídas).
  + Ejemplo: si OPC pierde conexión con el server, mostrar un mensaje en consola/web.
* Interacción bidireccional en la web.
  + Lectura: sensores, estado de la entrada digital.
  + Escritura: botón para encendido/apagado, barra para regular intensidad.

**🧪 Tecnologías sugeridas**

| **Parte** | **Tecnología sugerida** |
| --- | --- |
| MODbus esclavo | Arduino + modbus-rtu lib |
| MODbus maestro | Raspberry Pi + pymodbus |
| OPC Server | OpenOPC (con PyOPC + KEPServer si es posible) |
| OPC Client | Python + OpenOPC |
| MQTT Broker | Mosquitto |
| MQTT Publisher | Python + paho-mqtt |
| Frontend Web | HTML/CSS/JS + MQTT.js |

**📄 Informe: ¿Qué incluir?**

* Diagrama funcional como el de arriba.
* Capturas de pantalla del flujo funcionando.
* Código comentado de cada nodo (Arduino, Pi, Python).
* Ejemplos de mensajes MQTT (topic, payload).
* Indicadores de estado: por ejemplo, color verde si hay comunicación activa, rojo si se cayó.
* Dificultades (ej.: sincronización de datos, errores de conexión, delays).