**PROYECTO A1**

Darío González Sánchez

Alonso Montalvo Arias

# Resumen

El presente proyecto consiste en el diseño, implementación y puesta en marcha de un sistema de monitorización ambiental IoT (Internet of Things) de bajo coste, aplicado a un entorno crítico simulado: una Base Lunar. El objetivo principal ha sido desarrollar un prototipo capaz de medir, transmitir y visualizar en tiempo real variables vitales para la supervivencia en un hábitat cerrado: Temperatura, Humedad Relativa, Presión Atmosférica y Radiación Ultravioleta (UVI).

La solución técnica se basa en una arquitectura de tres capas. En la **capa física**, se ha utilizado un microcontrolador ESP32-S3 con conectividad WiFi, encargado de la adquisición de datos mediante sensores de precisión (BME280 y LTR390) y de la gestión de alertas visuales in-situ mediante un indicador LED RGB. En la **capa de comunicación**, se ha implementado el protocolo MQTT, ligero y robusto, garantizando la transmisión eficiente de tramas de datos JSON a través de un broker en la nube. Finalmente, en la **capa de aplicación**, se ha desarrollado un Dashboard de control en Python utilizando el framework Streamlit, que permite la visualización remota y el análisis de históricos.

Entre los principales **logros obtenidos**, destaca la consecución de un sistema estable y resiliente ante fallos de conexión, capaz de reconectarse automáticamente. Se han resuelto desafíos técnicos significativos, como la optimización del buffer de transmisión para paquetes de datos extensos y la calibración dinámica de la ganancia del sensor UV para lecturas precisas en exteriores. El resultado final es un prototipo funcional "llave en mano" que cumple con todos los requisitos de monitorización, filtrado de señales y alerta temprana ante condiciones ambientales adversas.

# Diseño preliminar

## Caso de uso

El sistema tiene como finalidad **monitorizar las condiciones ambientales** de distintas zonas de un hábitat lunar, garantizando la seguridad y confort de sus ocupantes.  
Para ello, cada nodo-sensor (formada por un **microcontrolador ESP32-S3** y un **sensor BME280**) mide de forma continua la **presión atmosférica, la temperatura y la humedad relativa** en su entorno inmediato.

Cuando alguno de estos parámetros se sale de los valores nominales definidos, el sistema activa una **alerta visual mediante el LED RGB integrado** en el microcontrolador, indicando el nivel de severidad:

* 🟢 Verde fijo: condiciones normales.
* 🟡 Amarillo fijo: valores fuera del rango óptimo (alerta).
* 🔴 Rojo parpadeante: condiciones críticas (alarma).

Los sensores se distribuyen en diferentes salas del hábitat (por ejemplo, un invernadero y las zonas comunes), lo que permite **comparar las condiciones ambientales entre áreas con distintos requerimientos**.

El usuario o el sistema central pueden así **identificar anomalías locales** y tomar medidas de corrección (ajuste térmico, humidificación, control de presión, etc.).

## Arquitectura funcional y Física

## Requisitos

Para el desarrollo del sistema de monitorización ambiental de la Base Lunar simulada, se han definido los siguientes requisitos de hardware y software:

* **Microcontrolador:** Se requiere un **ESP32-S3** (versión con LED RGB NeoPixel integrado en el GPIO 48) por su capacidad de procesamiento dual-core y conectividad WiFi nativa.
* **Sensores Ambientales:**
  + **BME280:** Para la medición precisa de Temperatura (-40 a 85 °C), Humedad Relativa (0-100%) y Presión Atmosférica (300-1100 hPa). Comunicación vía protocolo I2C.
  + **LTR390:** Para la medición específica de radiación ultravioleta (Índice UVI) y luz ambiente, crucial para la seguridad en entornos sin atmósfera. Comunicación vía I2C.
* **Indicadores:** Uso del LED RGB integrado para feedback visual de estados (Verde=Nominal, Naranja=Aviso, Rojo=Peligro, Azul=Sincronización).
* **Conectividad:** Capacidad de conexión a redes WiFi WPA2 y transmisión de datos mediante protocolo MQTT sobre TCP/IP.
* **Interfaz de Usuario:** Un Dashboard desarrollado en Python (Streamlit) capaz de graficar datos en tiempo real, filtrar mensajes por ID de dispositivo y exportar históricos.

# Diseño detallado

## Sistema de Sensores

El sistema de adquisición de datos se basa en el bus I2C. El ESP32-S3 actúa como maestro I2C, interrogando a los sensores periódicamente (cada 2 segundos según el firmware):

1. **BME280 (Dirección 0x76/0x77):** Conectado a los pines SDA (11) y SCL (10). Proporciona los datos de variables termodinámicas.
2. **LTR390 (Dirección 0x53):** Conectado en un segundo bus I2C o compartido (pines SDA 20, SCL 21 definidos en el código). Se ha configurado con una ganancia de 18x y resolución de 20 bits para asegurar sensibilidad suficiente ante la luz solar, convirtiendo la lectura bruta a índice UVI mediante un factor de calibración.

El LED NeoPixel (GPIO 48) no es un sensor, sino un actuador visual que procesa la lógica de "semáforo", cambiando de color según umbrales predefinidos en el código (ej. Temp > 35°C activa color Rojo).

## Plataforma

La arquitectura de la plataforma sigue un modelo IoT de publicación/suscripción:

1. **Nodo Edge (ESP32-S3):** Recopila datos, formatea un paquete JSON (incluyendo ID "A1", Timestamp sincronizado, y valores de sensores) y lo publica en el tópico Enviromental Sensors Network.
2. **Broker MQTT (EMQX):** Actúa como intermediario en la nube, recibiendo los mensajes y distribuyéndolos.
3. **Cliente Software (Dashboard):** Una aplicación Python ejecutada en PC que se suscribe al mismo tópico. Utiliza una arquitectura multihilo para separar la recepción de datos de la interfaz gráfica, evitando bloqueos.
4. **Gestión de Datos:** Los datos se almacenan temporalmente en memoria (buffer circular) para su visualización en gráficas y se permite su persistencia mediante exportación a CSV.

# Resultados del proyecto

## Plan de pruebas

### Alcance:

Definición de las funcionalidades, requisitos y/o características que serán probadas:

* Precisión y lectura correcta de los sensores BME280 y LTR390.
* Conectividad WiFi y estabilidad de la conexión MQTT.
* Visualización correcta de las 4 variables (Temp, Hum, UVI, Presión) en el Dashboard.
* Funcionamiento de la lógica de alertas del LED RGB.
* Capacidad de filtrado de datos ajenos (simuladores externos).

### Estrategia de Pruebas.

Métodos y enfoques que se utilizarán para realizar las pruebas:

* **Pruebas de Integración:** Verificar que el ESP32 envía datos y el Dashboard los recibe.
* **Pruebas Funcionales (Caja Negra):** Someter los sensores a estímulos (luz directa, calor corporal) y verificar si el LED cambia de color y la gráfica reacciona.
* **Pruebas de Estrés:** Dejar el sistema funcionando durante 1 hora continua para verificar fugas de memoria o desconexiones.
* **Prueba de Concurrencia:** Ejecutar el simulador de datos (ID: B2) simultáneamente para verificar que el sistema no mezcla los datos en las gráficas.

### Criterios de Aceptación.

Condiciones que deben cumplirse para considerar que una prueba ha sido exitosa:

* La latencia entre el evento físico y la visualización en pantalla debe ser inferior a 3 segundos.
* El LED debe cambiar a ROJO inmediatamente si la temperatura supera los 35°C o el UVI supera 8.0.
* El archivo CSV descargado debe contener todos los registros sin corrupción de datos.
* No deben aparecer valores "NaN" o vacíos en el log de mensajes.

### Recursos:

Detalle de los recursos necesarios:

* **Hardware:** Kit ESP32-S3, sensores, cableado, PC Portátil.
* **Software:** IDE de Arduino, Python 3.8+, Librerías Paho-MQTT y Streamlit.
* **Infraestructura:** Conexión a Internet (WiFi o punto de acceso móvil) sin bloqueo de puerto 1883.

## Resultados de las pruebas

Captura de pantalla de un celular

El contenido generado por IA puede ser incorrecto.Pantalla de computadora

El contenido generado por IA puede ser incorrecto.Gráfico

El contenido generado por IA puede ser incorrecto.

Las pruebas realizadas arrojaron los siguientes resultados:

1. **Lectura de Sensores:** Inicialmente, el sensor UV mostraba valores bajos (0.15). Tras ajustar la ganancia a 18x en el firmware y realizar pruebas en exterior (terraza), los valores de UVI subieron a rangos esperados (0.8 - 1.5), validando el funcionamiento.
2. **Transmisión de Datos:** Se detectó un error MQTT FAIL al incluir la presión atmosférica debido al tamaño del paquete JSON. Se corrigió aumentando el buffer de transmisión a 512 bytes, logrando una transmisión estable de todas las variables.
3. **Visualización:** El Dashboard muestra correctamente las 4 gráficas diferenciadas por colores. El log de tráfico discrimina correctamente los mensajes propios (verde) de los ajenos (gris).
4. **Alerta Visual:** El LED responde correctamente a los cambios de temperatura y luz, sirviendo como indicador fiable del estado ambiental.

# Coste del proyecto

El coste material del prototipo (BOM - Bill of Materials) se detalla a continuación. Los precios son estimados de mercado para componentes unitarios:

| **Componente** | **Descripción** | **Coste Estimado (€)** |
| --- | --- | --- |
| **ESP32-S3 DevKit** | Microcontrolador con WiFi y BLE | 12.50 € |
| **Adafruit BME280** | Sensor Temp/Hum/Presión (I2C) | 8.00 € |
| **Adafruit LTR390** | Sensor UV/Luz (I2C) | 7.50 € |
| **Cableado y Prototipado** | Cables Dupont y Breadboard | 4.00 € |
| **Fuente de Alimentación** | Power Bank USB o Adaptador 5V | 5.00 € |
| **Software** | Python, Arduino IDE, Streamlit (Open Source) | 0.00 € |
| **TOTAL** | **Hardware Físico** | **37.00 €** |

# Trabajo futuro

Para evolucionar este prototipo hacia un producto final desplegable en un entorno operativo real, se proponen las siguientes líneas de trabajo:

1. **Independencia Energética:**
   * *Mejora:* Integración de una batería LiPo con módulo de carga solar.
   * *Estimación:* +15€ en material, 2 semanas de desarrollo (gestión de energía deep-sleep).
2. **Base de Datos Persistente:**
   * *Mejora:* Sustituir el almacenamiento temporal en RAM/CSV por una base de datos local (SQLite) o de series temporales (InfluxDB) para mantener históricos de meses/años.
   * *Estimación:* 0€ (Software libre), 1 semana de desarrollo de software.
3. **Encapsulado Robusto:**
   * *Mejora:* Diseño e impresión 3D de una carcasa ventilada que proteja la electrónica del polvo pero permita el flujo de aire a los sensores.
   * *Estimación:* 5€ material (PLA/PETG), 1 semana de diseño CAD e impresión.
4. **Control Bidireccional:**
   * *Mejora:* Añadir capacidad para activar un relé (ventilador/riego) desde el Dashboard de Streamlit enviando comandos MQTT de vuelta al ESP32.
   * *Estimación:* 3€ (Módulo Relé), 3 días de programación.