## Documento de Pruebas

## Objetivos de la Prueba

* Verificar la correcta conectividad del ESP32 con AWS IoT, asegurando una comunicación estable y continua.
* Evaluar la precisión y confiabilidad de las mediciones de los sensores de temperatura, humedad e iluminación bajo diferentes condiciones ambientales.
* Validar la respuesta del sistema ante condiciones predefinidas, asegurando que los LEDs y el motor de ventilación se activen o desactiven de manera adecuada en los modos automático y manual.
* Garantizar que los comandos enviados desde AWS IoT sean correctamente interpretados y ejecutados por el sistema.
* Comprobar la correcta publicación y recepción de datos en AWS IoT para su análisis y monitoreo en tiempo real.

## Entorno de Pruebas

Para la realización de las pruebas, se ha configurado el siguiente entorno:

Hardware:

* Módulo ESP32 con conectividad Wi-Fi.
* Sensor de temperatura y humedad.
* Sensor de iluminación.
* Actuadores: LEDs de diferentes colores y motor de ventilación.
* Fuente de alimentación estable para el ESP32 y los sensores.

Software:

* Plataforma de desarrollo Arduino IDE con bibliotecas necesarias para la comunicación con AWS IoT.
* AWS IoT Core configurado con tópicos MQTT para la recepción y envío de datos.
* Monitor serie para visualizar los datos en tiempo real.
* Node-RED está configurado para suscribirse a los tópicos MQTT de AWS IoT y visualizar datos en un dashboard interactivo.
* Dashboard en Node-RED con gráficos en tiempo real para temperatura, humedad e iluminación.

Condiciones de Prueba:

* Pruebas en un ambiente controlado con variaciones de temperatura, humedad e iluminación simuladas.
* Conectividad estable y pruebas con desconexión de red para evaluar la reconexión automática.
* Evaluación del tiempo de respuesta del sistema ante cambios de estado.
* Comprobación de la correcta representación de los datos en el dashboard de Node-RED.

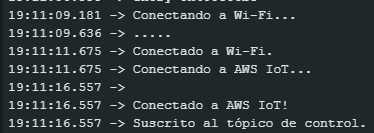
## Casos de Prueba

Conectividad Wi-Fi y AWS IoT

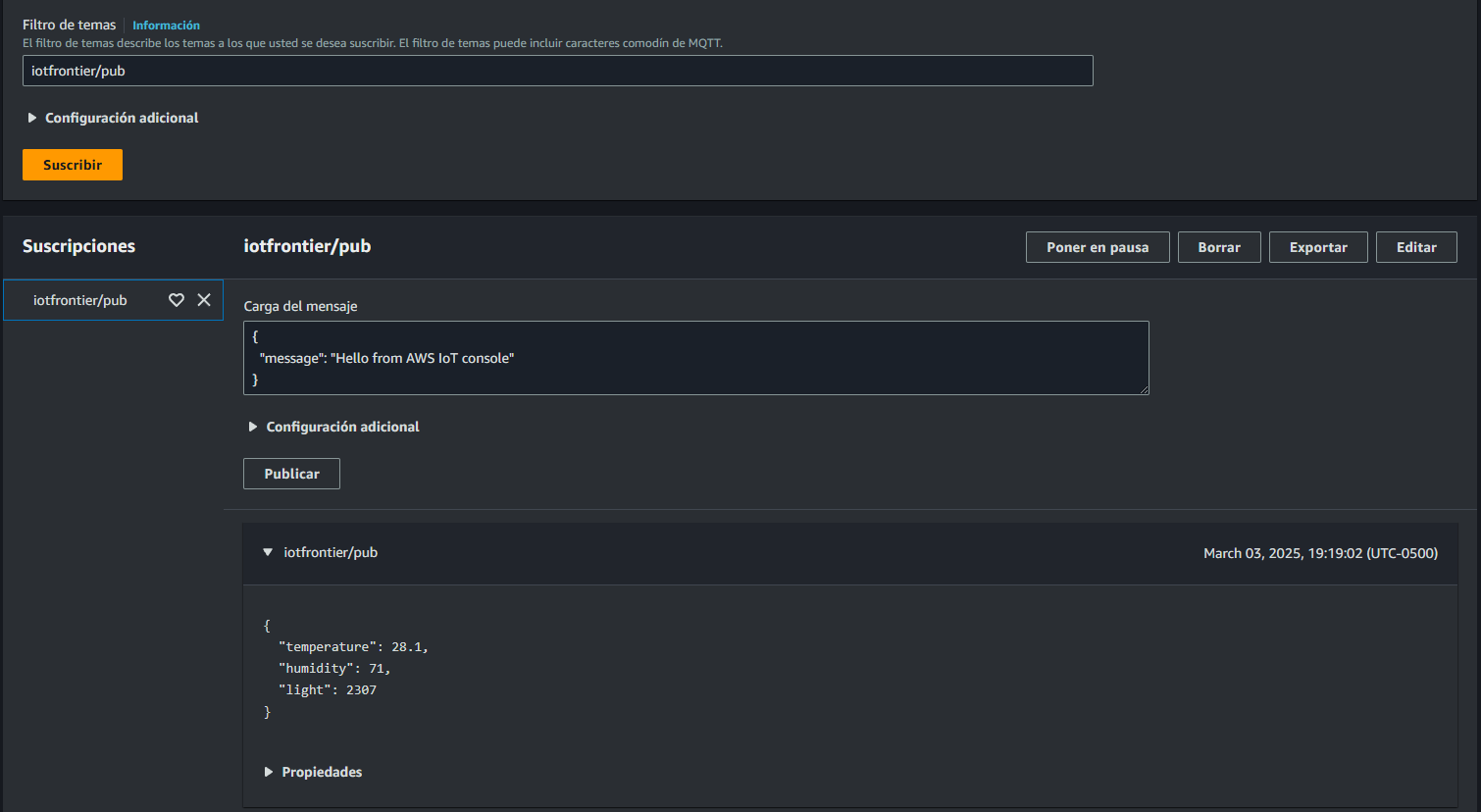
Descripción: Verificar que el ESP32 se conecte correctamente a la red Wi-Fi y a AWS IoT.

* Entradas: Credenciales Wi-Fi y certificados AWS.
* Procedimiento:
  + Encender el ESP32.
  + Observar en el monitor serie los mensajes de conexión.
* Resultados esperados:
  + Mensaje en consola: "Conectado a Wi-Fi."
  + Mensaje en consola: "Conectado a AWS IoT"
  + Suscripción exitosa a los tópicos AWS IoT.

Log de microcontrolador:



Conexión con el AWS IOT



Control Automático y Manual del Sistema

Descripción: Verificar si la lógica del sistema reacciona correctamente a las instrucciones manuales y las configuradas.

Caso 1: Modo Automático (ControlMode = false)

| Entrada | LED Amarillo | LED Rojo | Motor |
| --- | --- | --- | --- |
| Temperatura < 25°C | Apagado | Apagado | Apagado |
| Temperatura >= 25°C y < 30°C | Encendido | Apagado | Apagado |
| Temperatura >= 30°C y < 35°C | Apagado | Encendido | Apagado |
| Temperatura >= 35°C | Apagado | Encendido | Encendido |

Procedimiento:

* Simular cambios en la temperatura.
* Observar la activación de LEDs y motor.

Resultado esperado:

Los LEDs y el motor deben activarse según los valores de temperatura establecidos.

Caso 2: Modo Manual (ControlMode = true)

| Escenario | Entrada (Comando MQTT) | LED Amarillo | LED Rojo | Motor | Resultado Esperado |
| --- | --- | --- | --- | --- | --- |
| Caso 1: Apagar todo | { "led\_amarillo": false, "led\_rojo": false, "motor": false } | Apagado | Apagado | Apagado | Ningún actuador encendido. |
| Caso 2: Encender solo LED Amarillo | { "led\_amarillo": true, "led\_rojo": false, "motor": false } | Encendido | Apagado | Apagado | Solo LED Amarillo encendido. |
| Caso 3: Encender solo LED Rojo | { "led\_amarillo": false, "led\_rojo": true, "motor": false } | Apagado | Encendido | Apagado | Solo LED Rojo encendido. |
| Caso 4: Encender solo el Motor | { "led\_amarillo": false, "led\_rojo": false, "motor": true } | Apagado | Apagado | Encendido | Solo el motor encendido. |
| Caso 5: Encender LED Amarillo y Motor | { "led\_amarillo": true, "led\_rojo": false, "motor": true } | Encendido | Apagado | Encendido | LED Amarillo y Motor encendidos. |
| Caso 6: Encender LED Rojo y Motor | { "led\_amarillo": false, "led\_rojo": true, "motor": true } | Apagado | Encendido | Encendido | LED Rojo y Motor encendidos. |
| Caso 7: Encender todo | { "led\_amarillo": true, "led\_rojo": true, "motor": true } | Encendido | Encendido | Encendido | Todos los actuadores encendidos. |

Proceso: Para la aplicación de los dos suscriptores para cada actuador, los cuales están configurados para ser utilizados independientemente, para objetivo del documento unimos la entrada de ambos comandos MQTT.

Procedimiento:

* Simular los distintos estados de los actuadores LEDS y el motor
* Observar la activación de LEDs y motor.

Resultado esperado:

Los LEDs y el motor actúan de acuerdo a los comandos o instrucciones proporcionados por las suscripciones MQTT.

Caso 3: Configuración de los parámetros del invernadero

Configuración inicial:

Entrada: Archivo de configuración válido enviado a iotfrontier/sub-config.

Procedimiento:

* Enviar configuración inicial con parámetros de umbrales y conexiones.

Resultado esperado:

* El sistema debe aplicar la configuración correctamente.
* No debe haber errores de validación.

Configuración incorrecta:

Entrada: Archivo de configuración con valores fuera de rango.

Procedimiento:

* Enviar configuración con valores inválidos.

Resultado esperado:

* La interfaz debe rechazar la configuración.
* La interfaz gráfica en Node-RED debe evitar la entrada de valores menores a 20°C y mayores a 60°C.

Cambio de configuración en tiempo real:

Entrada: Actualización de parámetros en iotfrontier/sub-config.

Procedimiento:

* Modificar la configuración mientras el sistema está en funcionamiento.
* Verificar si los nuevos parámetros se aplican correctamente.

Resultado esperado:

* Los cambios deben reflejarse en tiempo real.
* No debe haber interrupciones en el funcionamiento del sistema.

Interfaz: Los cambios se deben poder realizar mediante el uso de la interfaz gráfica

usando los valores para ambientes ya configurados o ingresar valores manuales.

