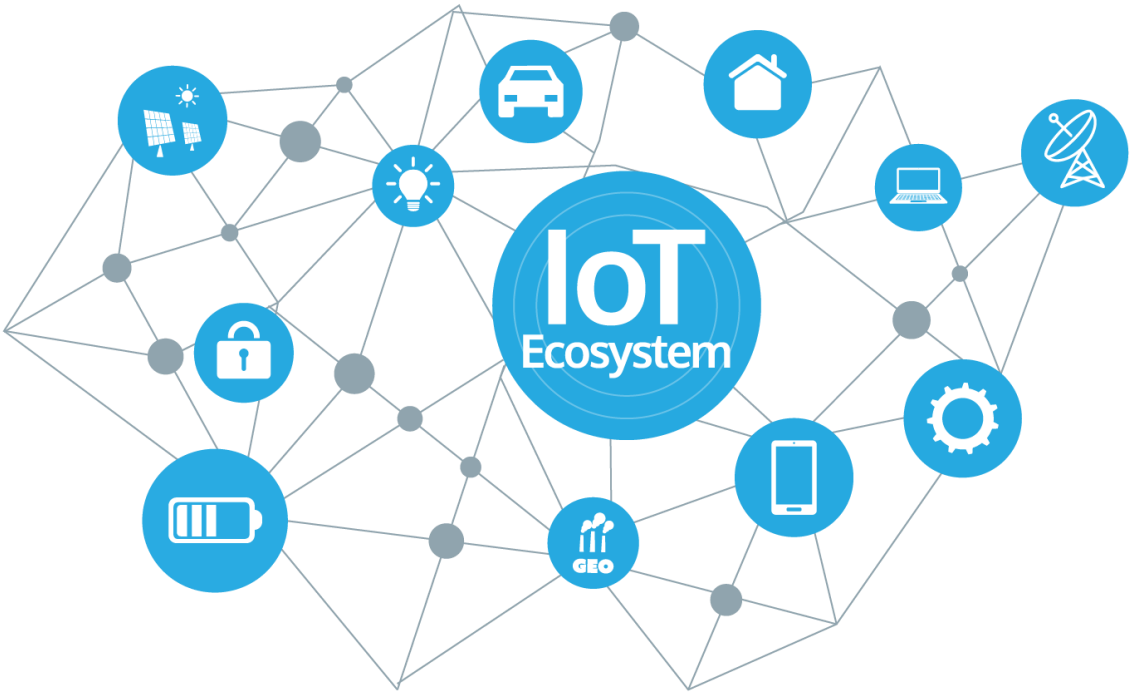


Proyecto Final IoT

DESARROLLO DE APLICACIONES EN LA NUBE

UNIVERSIDAD DE MÁLAGA

11/01/2021

García Rodríguez, Rafael Alejandro.

Martínez Rodríguez, María Teresa.

Pérez Jiménez, Pablo.

Ruiz Ruiz, Manuel.

Carlos Lobato Padilla no ha participado en este proyecto.

Índice

- Introducción y objetivos
- DIseño HW y esquema de conexionado
- Diseño SW
  - Diagrama de bloques/flujo del programa y descripción del funcionamiento
  - Técnicas para robustez del sistema
  - Librerías utilizadas
- Resultados y conclusiones
- Manual de usuario
- Lista y descripción de los ficheros (.ino y .json)

Introducción y objetivos

El objetivo principal de este proyecto es diseñar un sistema de monitorización y control multi-ubicación compuesto por varios nodos IoT basados en el módulo WIFI ESP8266 programados con el IDE de Arduino. Los nodos serán capaces de leer sensores, enviar datos y recibir órdenes para cambiar su configuración y el estado de los actuadores sobre los que tiene control mediante el protocolo MQTT. Además, se diseñará una aplicación que controlará cada uno de estos nodos, que se encontrarán distribuidos geográficamente formando una red de actuadores y sensores. Los datos generados por estos dispositivos serán recogidos por una aplicación (compuesta por un backend NodeRED, una base de datos MongoDB, y un broker MQTT) que los procesará y almacenará. La información de interés será accesible para los usuarios de una forma amigable utilizando una variedad de interfaces, al menos un interfaz gráfico web y otro conversacional Telegram.

Las funcionalidades concretas que se implementarán:

- Medir, almacenar de forma temporal y transmitir de forma inalámbrica parámetros fundamentales de cada ubicación (mediante mensajes con formato JSON).
- Controlar dos actuadores de forma inalámbrica (un LED controlado mediante PWM y un interruptor (switch)), y transmitir información sobre su estado actual.
- Permitir actualizar el firmware de todos los dispositivos inalámbricamente mediante FOTA. La comprobación de nuevas actualizaciones en el servidor se podrá realizar de 3 formas diferentes:
    - Al arrancar el dispositivo
    - Cuando se le ordene al dispositivo inalámbricamente o pulsando el botón flash del dispositivo
    - Periódicamente, cada X minutos (programable)
- Interactuar con los nodos a través del botón flash para modificar la intensidad del LED y comprobar si hay actualizaciones de firmware disponibles.
- Crear una interfaz web gráfica con un sistema SCADA (supervisión, control y adquisición de datos) en NodeRED, donde se podrá realizar:
    - Visionado de datos recogidos por los dispositivos, tanto de forma individual como conjunta, y con interfaz gráficos.
    - Consultas por rango de fechas, donde se mostrarán los datos clasificados y ordenados.
    - Análisis de los datos consultados (valores medios, desviaciones, máximos, etc.).
    - Descargar los datos almacenados en un fichero.
    - Almacenar un historial de los cambios y acciones realizadas en el sistema

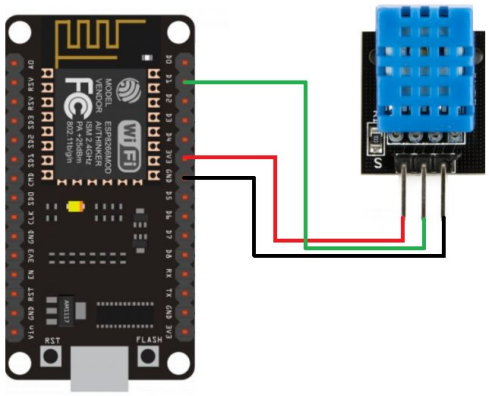
A parte de estas funcionalidades, también se han definido otra serie de requisitos:

- Corrección y robustez en los datos y en la interacción con el usuario
- Amigabilidad de las interfaces y las distintas partes del sistema
- Facilidad de mantenimiento y ampliación

Diseño HW y esquema de conexionado

El conexionado entre la placa ESP8266 y los sensores es muy sencillo, ya que solamente tenemos un sensor DHT11 conectado.

Para conectar este sensor a la placa solamente es necesario una conexión a **3V**, otra a **GND**, y otra para recibir los datos que, en este caso, hemos utilizado **D1** (**GPIO5**).

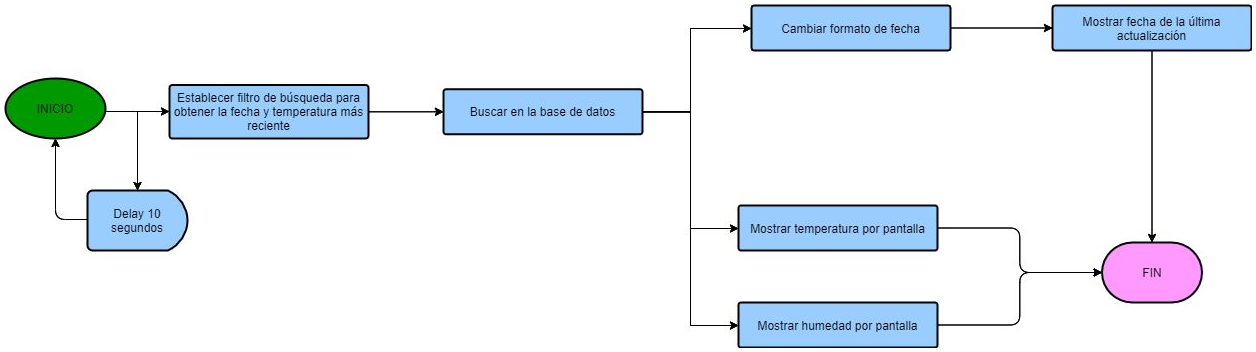


Cada integrante del equipo tendrá una placa y un sensor con el mismo conexionado mostrado anteriormente, y cada una de estas placas se conectará a un servidor MQTT donde enviará la información recogida del sensor, para luego visualizarla por una interfaz.

Diagrama de bloques y descripción del funcionamiento

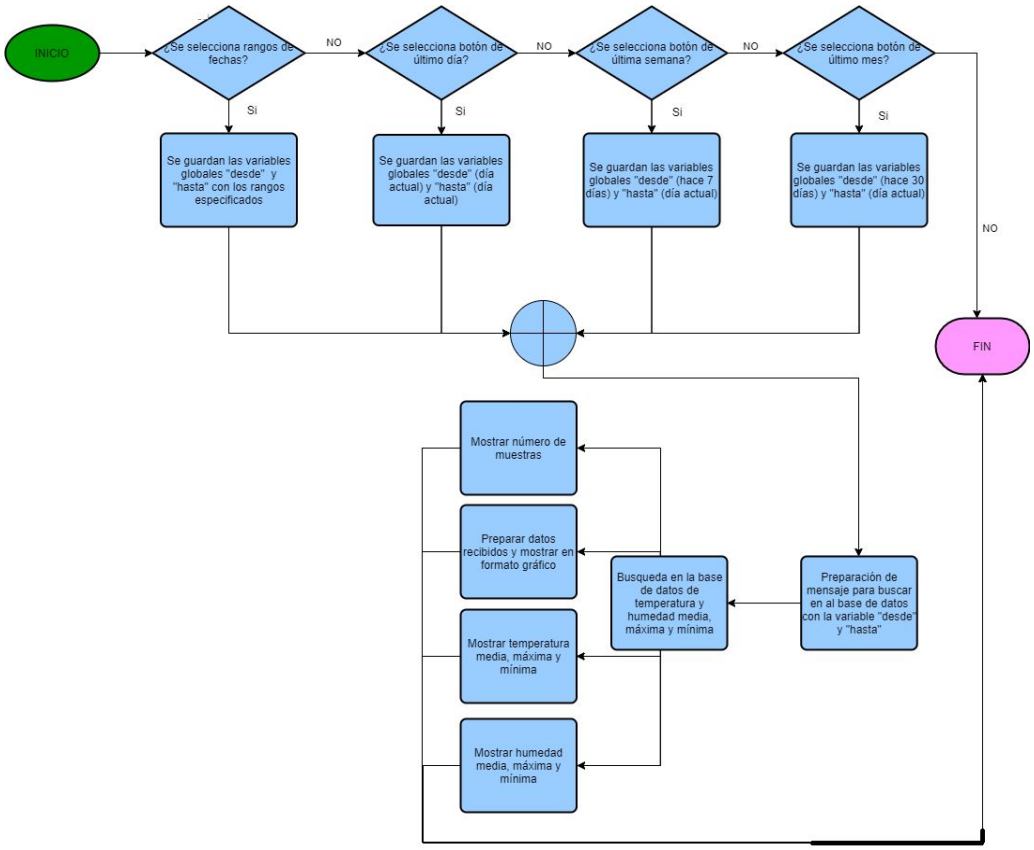
NodeRED Dashboard

Último valor recibido



Cada diez segundos se va a establecer un filtro de búsqueda para obtener los datos recientes, se va a buscar en la base de datos y se va a preparar la salida de esta para mostrar la humedad, temperatura y fecha de medición.

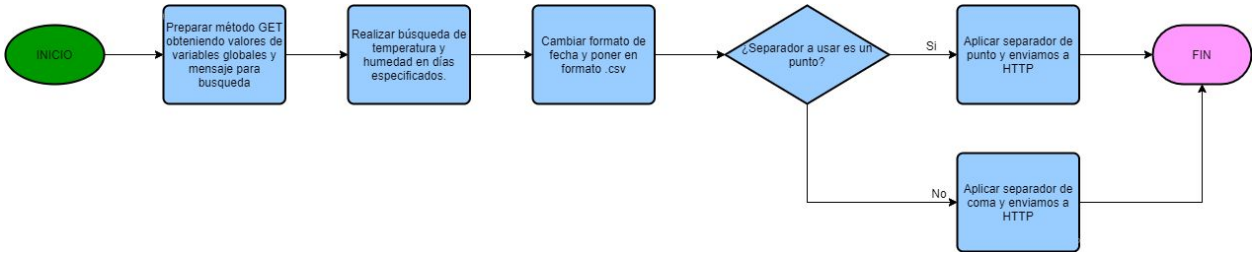
Temperatura y humedad por rango de fechas



Dependiendo de si se usa un rango de fechas o un acceso rápido por botón, se van a preparar las variables “desde” y “hasta”, con las se realizará una búsqueda en la base de datos para que esta nos devuelva la temperatura y humedad máxima, mínima y media siguiendo el rango de fechas especificado.

También nos mostrará el número de muestras que hay y una gráfica en la que se verá los valores máximos, medios y mínimos de temperatura y humedad por día.

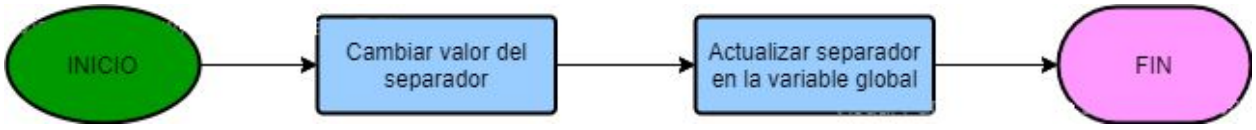
Descarga de CSV



Al pulsar el botón “**Descargar registros en CSV**” en el dashboard, se va a hacer una búsqueda en la base de datos para obtener la información de la temperatura y humedad por días, donde la fecha de inicio y fin vienen definidas por el último valor de las variables globales “desde” y “hasta”.

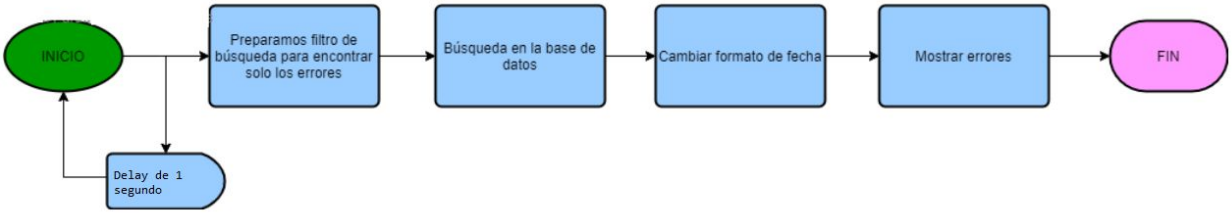
Al obtener los datos, se prepara el .csv con toda la información obtenida y con el separador que se haya elegido, finalmente se devuelve al HTTP.

Cambiar separador



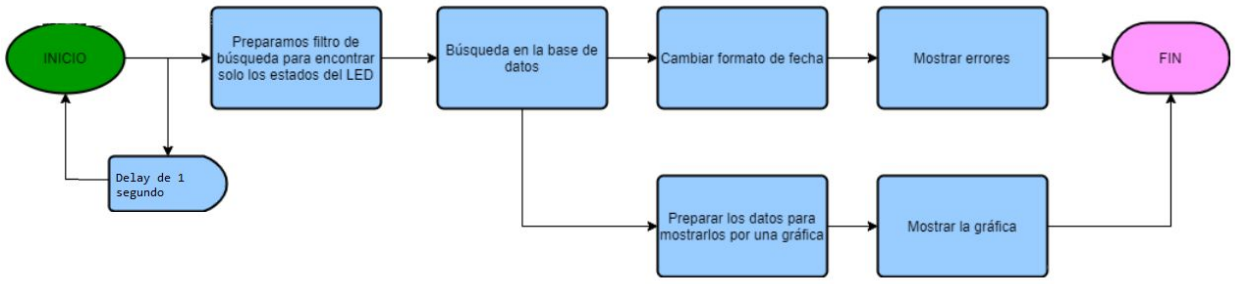
Si se cambia el valor del separador, se actualiza este cambio en la variable global para que cuando hagamos uso del flujo anterior a este, podamos usar un separador u otro.

Log de errores



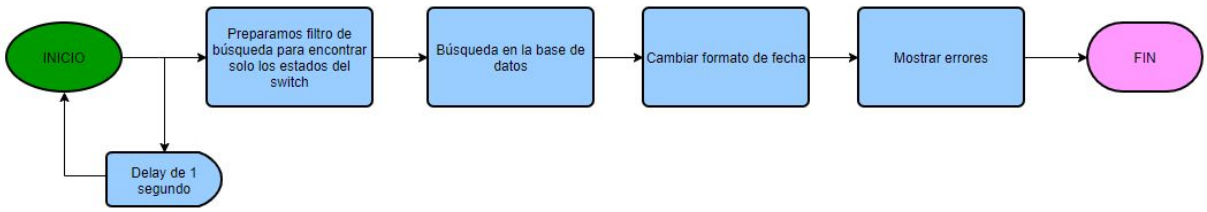
Cada segundo se va a hacer una búsqueda en la base de datos de todos los errores que se han generado y se van a mostrar en el dashboard.

Log de estado del LED



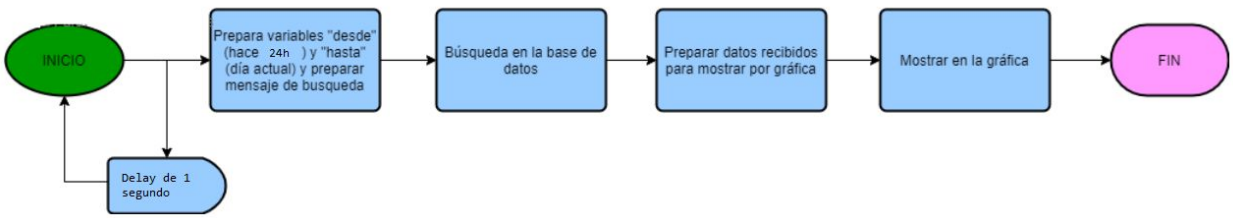
Cada segundo se va a hacer una búsqueda en la base de datos de todos los estados del LED que se han generado y se van a mostrar en el dashboard, además de una gráfica con estos cambios.

Log de estado del switch



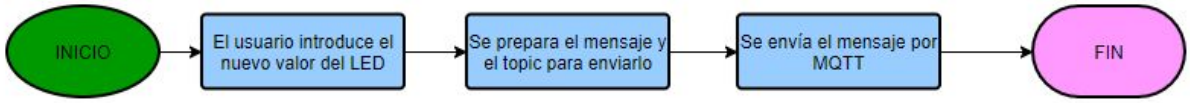
Cada segundo se va a hacer una búsqueda en la base de datos de todos los estados del switch que se han generado y se van a mostrar en el dashboard.

Evolución de temperatura y humedad



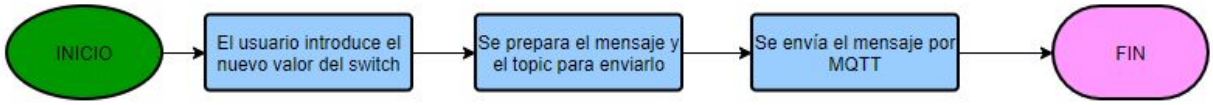
Cada segundo se va a hacer una búsqueda en la base de datos con un rango de fechas de las últimas 24 horas. Con los datos recibidos que son las temperaturas y humedades por día, se va a mostrar en una gráfica en el dashboard.

Cambiar intensidad del LED.



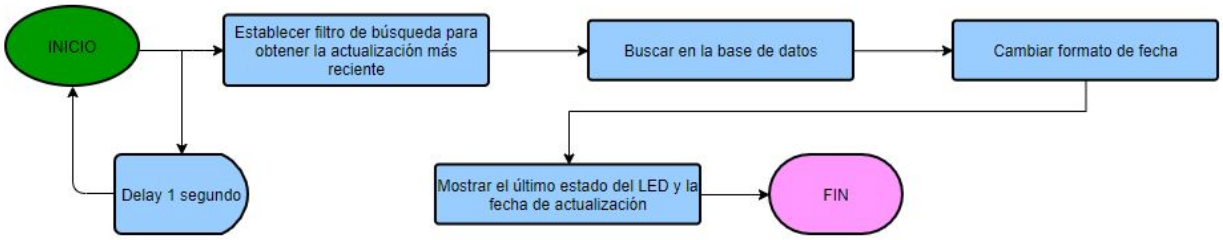
El usuario introduce un valor por el dashboard y se envía a la placa por MQTT poniendo el nuevo valor en un mensaje con el topic correspondiente.

Cambiar estado del switch.



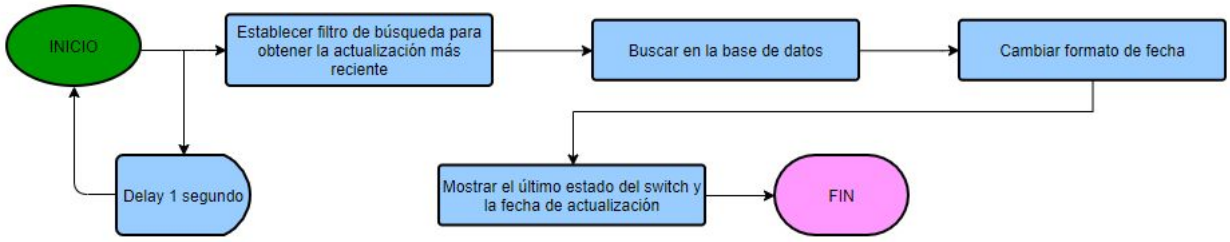
El usuario introduce un valor por el dashboard y se envía a la placa por MQTT poniendo el nuevo valor en un mensaje con el topic correspondiente.

Mostrar último estado del LED y fecha de actualización.



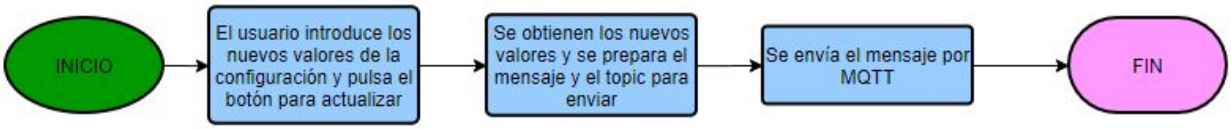
Cada segundo va comprobando el último valor recibido para el estado del LED y lo muestra en el dashboard junto a su última fecha de actualización.

Mostrar último estado del switch y fecha de actualización.



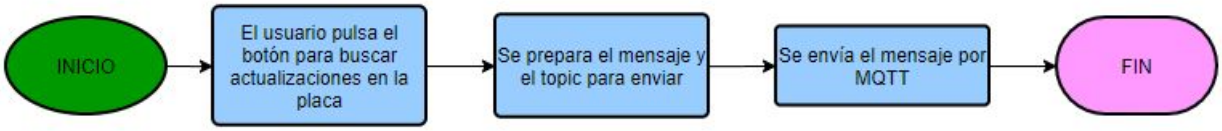
Cada segundo va comprobando el último valor recibido para el estado del switch y lo muestra en el dashboard junto a su última fecha de actualización.

Actualizar configuración.



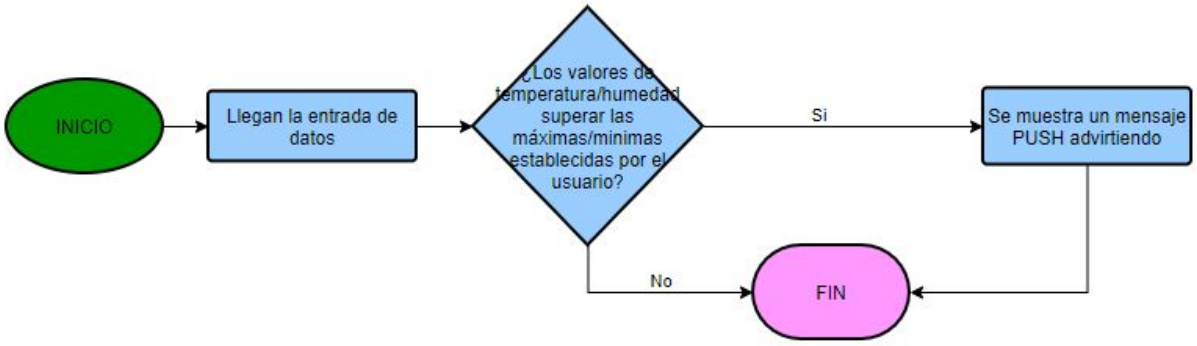
El usuario tiene la posibilidad de cambiar valores como el tiempo entre envío y envío de datos, o la frecuencia de comprobación de actualizaciones. Al cambiarlas y pulsar el botón de actualizar, lo que se va a hacer es enviar un mensaje MQTT con todos estos valores para que la placa los actualice.

Comprobar actualización OTA.



El usuario tiene también la posibilidad de comprobar si hay alguna actualización disponible para el programa Arduino cargado en la placa. Pulsando el botón correspondiente del dashboard, envía un mensaje a la placa para que compruebe si hay alguna actualización y actualizarse en caso de existir.

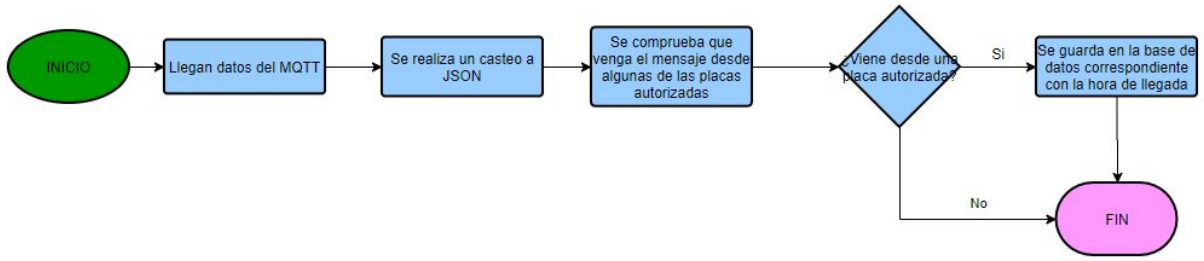
Mostrar avisos de máximos y mínimos.



También podemos elegir unos valores de temperatura máxima/mínima y humedad máxima/mínima, si se da el caso de que la temperatura/humedad superan estos valores se nos notificará en el dashboard con un mensaje push.

NodeRED Base de datos

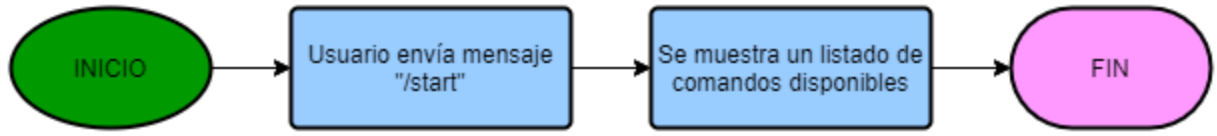
Base de datos.



Cuando llegan los datos vía MQTT, se castean a JSON, se comprueba que vengan de alguna de las placas que están en la whitelist y se meten los datos en la base de datos añadiendo la fecha de llegada.

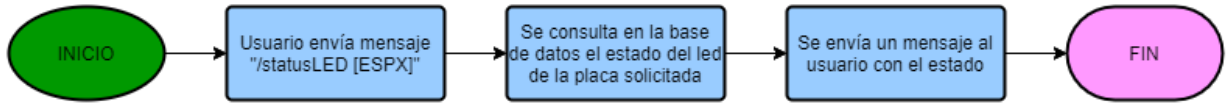
NodeRED Telegram

Mensaje /start



Cuando el bot recibe el comando */start*, este responderá con un listado de los comandos disponibles.

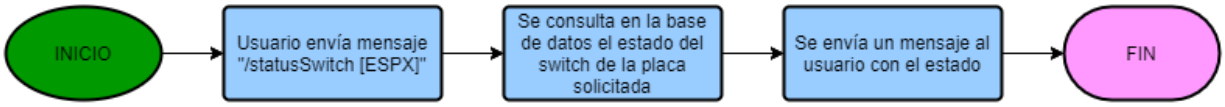
Mensaje /statusLED [ESPX]



Cuando el bot recibe el comando */statusLED [ESPX]*, este realizará una consulta en la base de datos para obtener el estado del LED de la placa solicitada. Si no hay valor *[ESPX]*, se

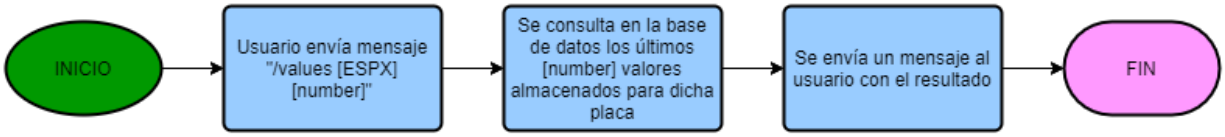
consulta el estado de todas las placas. Después, responderá con un mensaje mostrando el estado al usuario.

Mensaje /statusSwitch [ESPX]



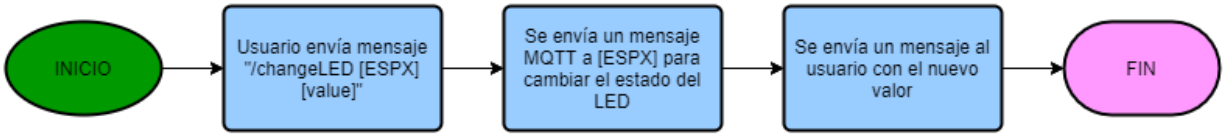
Cuando el bot recibe el comando */statusSwitch [ESPX]*, este realizará una consulta a la base de datos para obtener el estado del switch de la placa solicitada. Si no hay valor *[ESPX]*, se consulta el estado de todas las placas. Después, responderá con un mensaje mostrando el estado al usuario.

Mensaje /values [ESPX] [number]



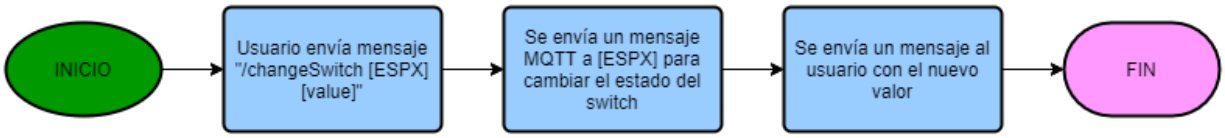
Cuando el bot recibe el comando */values [ESPX] [number]*, buscará en la base de datos los últimos *[number]* valores almacenados. Si no hay valor *[ESPX]*, se consultan los datos de todas las placas. Después, mostrará dichos datos al usuario.

Mensaje /chagneLED [ESPX] [value]



Cuando el bot recibe el comando */changeLED [ESPX] [value]*, enviará un mensaje MQTT al topic *infind/GRUPO2/ESPX/led/cmd* con el valor proporcionado por el usuario. Si no hay valor *[ESPX]*, se enviará el mensaje a todas las placas. Después, responderá al usuario con un mensaje con el valor (los valores) introducido(s).

Mensaje /changeSwitch [ESPX] [value]

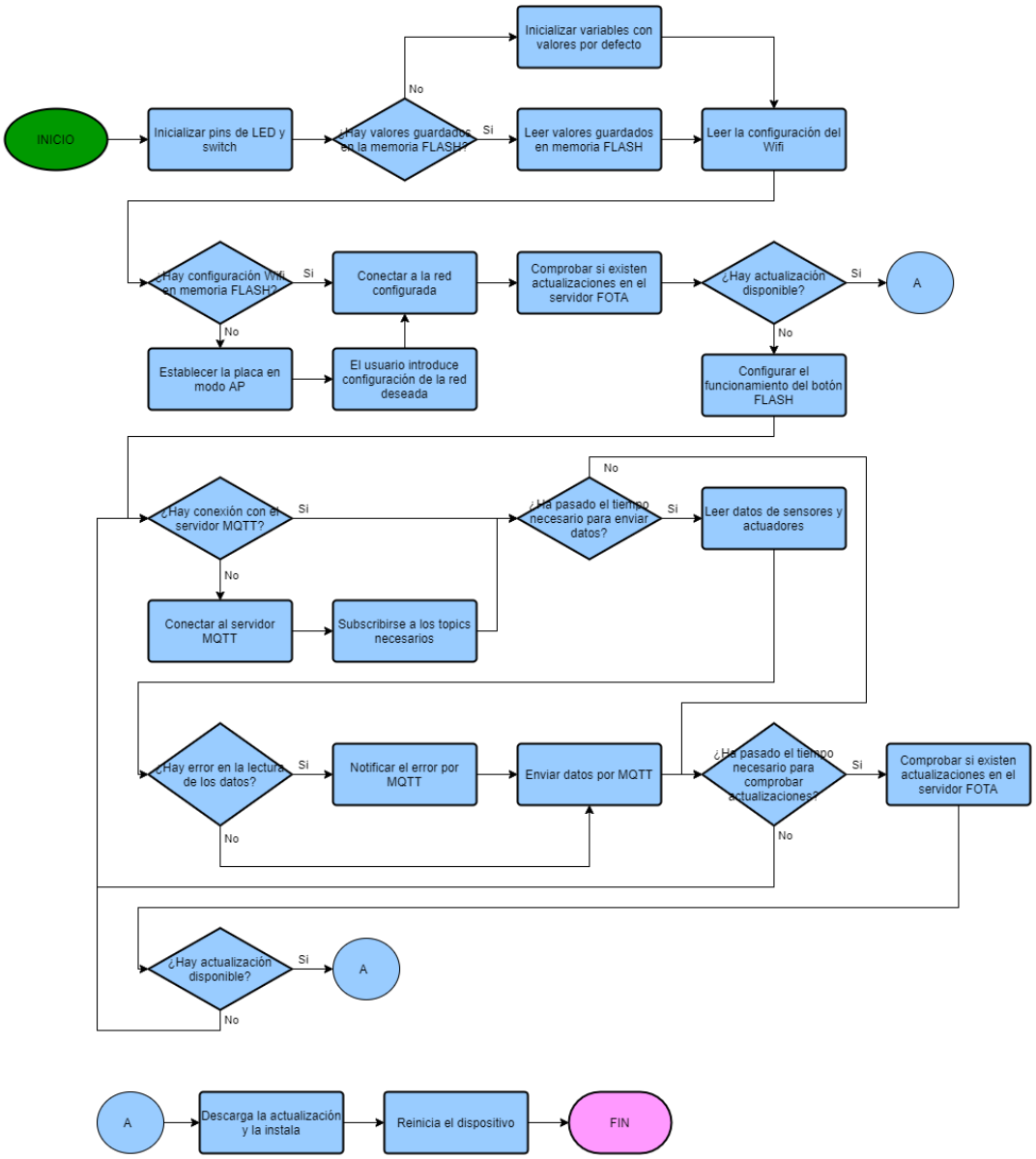


Cuando el bot recibe el comando */changeSwitch [ESPX] [value]*, este enviará un mensaje MQTT al topic *infind/GRUPO2/ESPX/switch/cmd* con el valor proporcionado por el usuario. Si no hay valor *[ESPX]*, se enviará el mensaje a todas las placas. Después, responderá al usuario con un mensaje con el valor introducido.

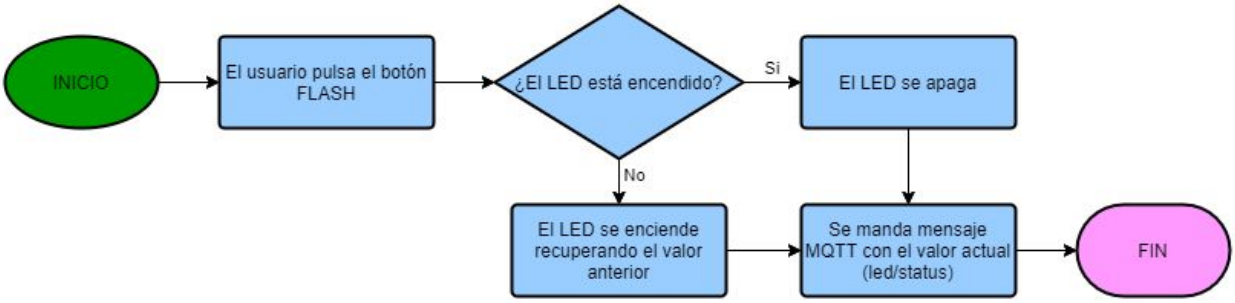
Código Arduino

Inicio/reinicio de la placa

Cuando la placa se enciende, suceden una serie de eventos. Primero se configuran los pines que corresponden al LED y al switch. Después se comprueba si hay variables guardadas en la memoria RAM. Si están guardadas, se recuperan esos valores. Si no, se inicializan las variables con unos valores por defecto. A continuación, se procede a leer la configuración del Wifi. Si no hay una configuración guardada en la memoria flash, la placa se inicia en modo AP, permitiendo que el usuario se conecte para configurar la conexión a una red Wifi. Tras este paso, o si la configuración ya existía en la memoria flash, se procede a realizar la conexión con la red Wifi configurada. Ahora, se comprueba si existen actualizaciones disponibles en el servidor FOTA. Si existen actualizaciones, se descarga el nuevo programa, y se reinicia el dispositivo. Si no, se continúa con el siguiente paso, que consiste en configurar el funcionamiento del botón FLASH. Después, se inicia el bucle. Primero se comprueba si hay conexión con el servidor MQTT. Si no hay conexión, se intenta conectar con el servidor MQTT, y se subscribe a los topics necesarios. Cuando ya hay conexión (o si ya la había desde un principio), se comprueba si hay que mandar los datos. En caso afirmativo, se leen los datos actuales de los sensores y actuadores. Si hay error en esos datos, se notifica el error mediante un mensaje MQTT al topic *infind/GRUPO2/ESPX/errores*. A continuación, y en el caso de que no hubiera error, se procede a enviar los datos a la base de datos mediante un mensaje MQTT al topic *infind/GRUPO2/ESPX/led/datos*. Después, (o si no era el momento de enviar los datos), se comprueba si ha pasado el tiempo necesario para conectarse con el servidor de actualizaciones. En el caso de que haya pasado el tiempo necesario, se procede a comprobar si existen actualizaciones disponibles. Si hay actualizaciones, se descarga la actualización, y se reinicia el dispositivo. Si no había actualizaciones, o no era el momento de comprobar si había actualizaciones, se vuelve a comenzar el bucle, comprobando la conexión con el servidor MQTT.

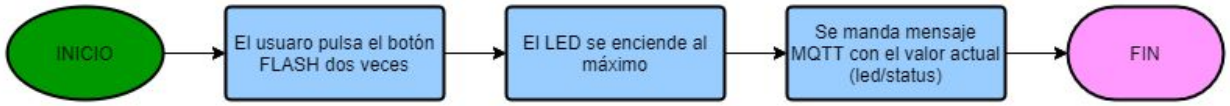


Pulsación simple - Encendido y apagado del LED



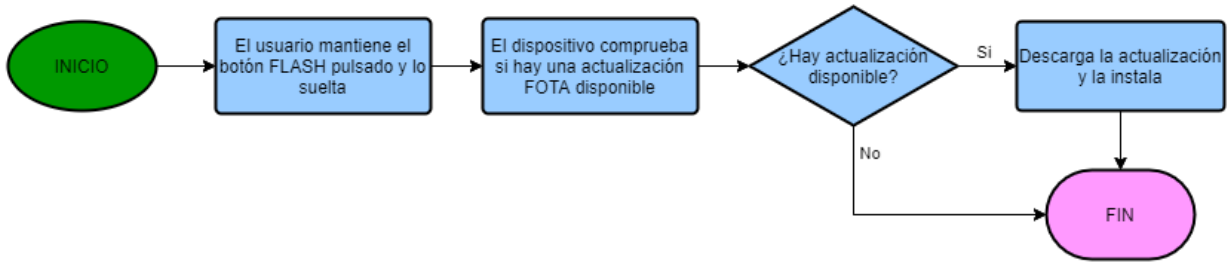
Cuando se pulsa el botón FLASH de la placa una vez, se comprueba si el LED está encendido. En el caso de que este encendido, se apaga, almacenando el valor que tenía. En el caso de que este apagado, se enciende al valor almacenado previamente. Cuando termina este proceso, se envía un mensaje MQTT al topic *infind/GRUPO2/ESPX/led/status* indicando el valor actual del LED

Pulsación doble - Encendido del LED al máximo



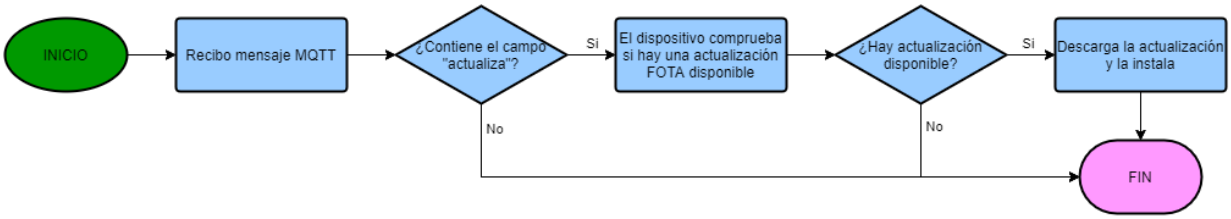
Cuando se pulsa el botón FLASH de la placa dos veces seguidas, el LED se enciende al nivel máximo, y se envía un mensaje MQTT al topic *infind/GRUPO2/ESPX/config* con el valor actual del LED.

Pulsación larga - Comprobación de servidor FOTA



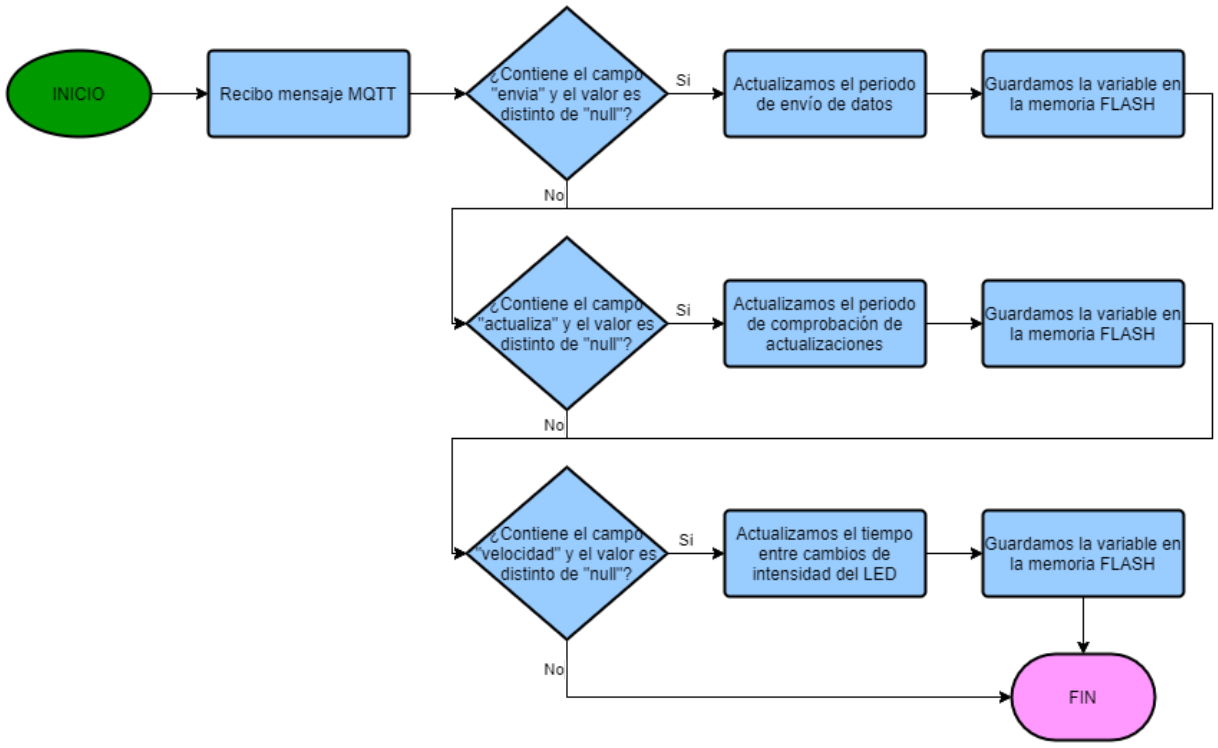
Cuando se mantiene pulsado el botón FLASH de la placa y se suelta, el dispositivo comprobará el servidor de FOTA en busca de actualizaciones disponibles. Si existe alguna actualización disponible, la descarga y la instala.

Mensaje MQTT - infind/GRUPO2/ESPX/FOTA



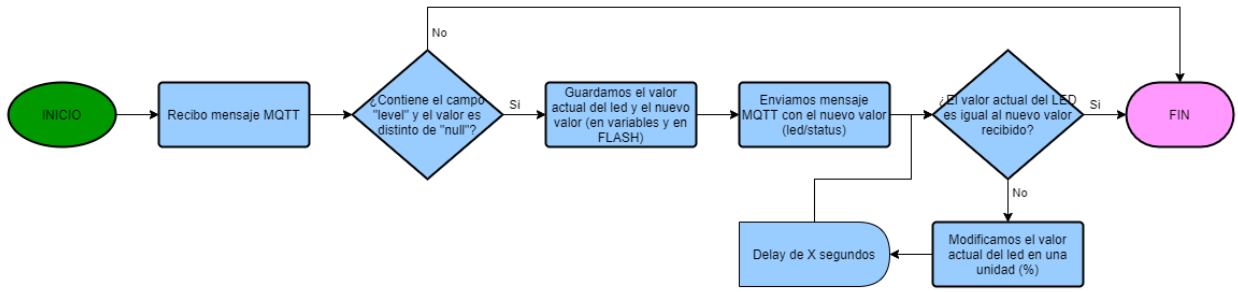
Cuando se recibe un mensaje MQTT en el topic *infind/GRUPO2/ESPX/FOTA* con el campo *"actualiza"*, el dispositivo comprobará el servidor de FOTA en busca de actualizaciones disponibles. Si existe alguna actualización disponible, la descarga y la instala.

Mensaje MQTT - infind/GRUPO2/ESPX/config



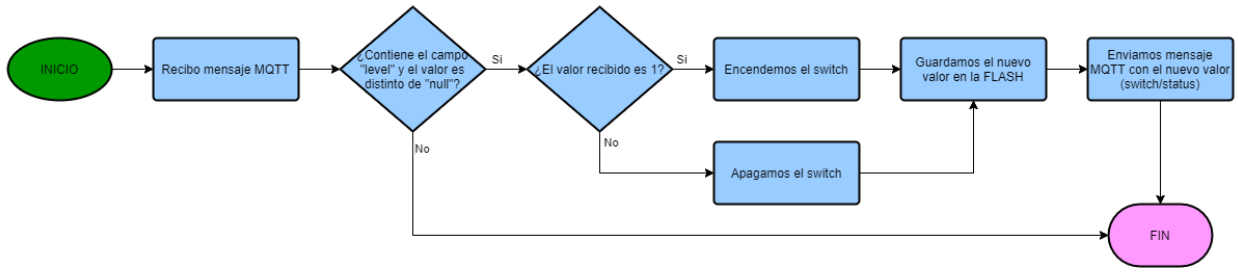
Cuando se recibe un mensaje MQTT en el topic *infind/GRUPO2/ESPX/config*, se comprueban varios campos. Si el campo *"envia"* existe y es distinto de *"null"*, se actualiza el período de envío de datos con el valor recibido, y se guarda en la memoria flash. Si no, se pasa a comprobar el siguiente campo. Si el campo *"actualiza"* existe y es distinto de *"null"*, se actualiza el período de comprobación de actualizaciones en el servidor FOTA con el valor recibido, y se guarda en la memoria flash. Si no, se pasa a comprobar el siguiente campo. Si el campo *"velocidad"* existe y es distinto de *"null"*, se actualiza el tiempo entre cambios de intensidad del LED con el valor recibido, y se guarda en la memoria flash. Si no, se termina el flujo.

Mensaje MQTT - infind/GRUPO2/ESPX/led/cmd



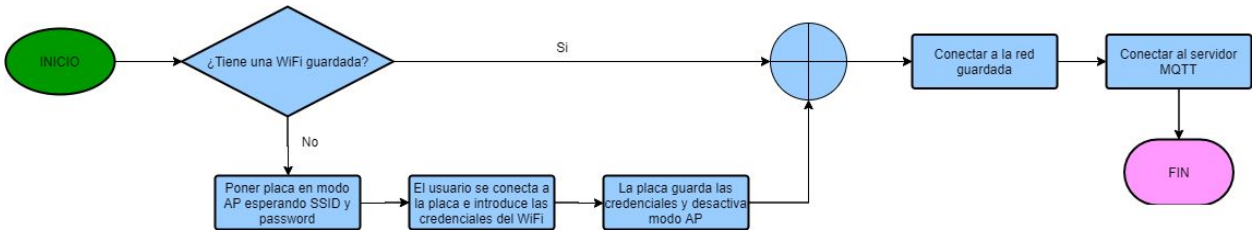
Cuando se recibe un mensaje MQTT en el topic *infind/GRUPO2/ESPX/led/cmd*, se comprueba que existe el campo *"level"*, y que tiene un valor no nulo. Si no se cumplen dichas condiciones, el flujo se termina. Si se cumplen, se almacena el valor actual del LED y el valor que se quiere conseguir en dos variables. Después, se envía un mensaje MQTT al topic *infind/GRUPO2/ESPX/led/status*, indicando el nuevo valor recibido. Después, se realiza un bucle donde se comprueba si el valor actual del LED es igual al nuevo valor recibido, y si no, se modifica un 1% de la luminosidad total del LED en la dirección deseada, y se espera un tiempo determinado. Cuando el LED alcanza el valor deseado, se termina el flujo.

Mensaje MQTT - infind/GRUPO2/ESPX/switch/cmd



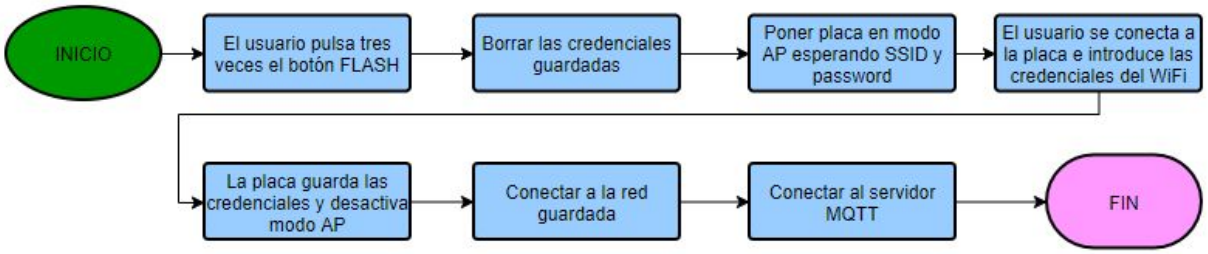
Cuando se recibe un mensaje MQTT en el topic *infind/GRUPO2/ESPX/switch/cmd*, se comprueba que existe el campo *"level"*, y que tiene un valor no nulo. Si no se cumplen dichas condiciones, se termina el flujo. Si se cumplen, se comprueba el valor recibido. Si es uno, se enciende el switch. Si es distinto de uno, se apaga el switch. Después de ambos casos, se envía un mensaje MQTT al topic *infind/GRUPO2/ESPX/led/status*, indicando el nuevo valor recibido.

Ampliación: Conexionado del WiFi cuando se enciende la placa por primera vez o se resetea.



Al resetear la placa o encenderla por primera vez, si tiene un WiFi guardado se va a conectar automáticamente. Si por el contrario no tiene nada guardado, va a estar en modo AP permitiendo que un usuario pueda conectarse a él desde una interfaz, donde podrá introducir las credenciales del WiFi para que la placa las guarde, quite el modo AP y se conecte con las credenciales introducidas.

Ampliación: Conexionado del WiFi cuando se pulsa el botón FLASH tres veces seguidas.



Si en algún momento de la ejecución de la placa se aprieta tres veces seguidas el botón FLASH, la placa borra las credenciales del WiFi y entra en modo AP permitiendo que un usuario se conecte a él para introducir las credenciales del WiFi desde una interfaz para que la placa los guarde. Una vez guardada, se quita el modo AP y se conecta con las credenciales introducidas.

Técnicas para robustez del sistema

Se han hecho diferentes acciones para hacer este sistema lo más robusto posible:

- Se ha creado una whitelist en la que solo se permite introducir datos por MQTT a las placas con el ChipID que nosotros especifiquemos en la whitelist. Si otra persona intenta introducir algo, no se le permitirá.

- Se tiene un registro de errores que se muestra por el dashboard. Por ejemplo, si el sensor envía mensajes null, o el LED tiene valores negativos, algo no anda bien y se registra este fallo para tenerlo en cuenta.
- Se comprueban los valores de los mensajes recibidos vía MQTT para ver si son nulos. En ese caso, no se modifica el valor actual.
- Se ha implementado una ampliación por medio de la cual todos los valores configurables del sistema como son: el estado del LED, el estado del switch, la velocidad de actualización, la velocidad con la que cambian la intensidad del LED, la velocidad del envío de datos y la configuración del WiFi son almacenados en la memoria flash de modo que si se produce una desconexión o un reseteo del sistema estos valores no se pierden.
- Se ha configurado el mecanismo de última voluntad que proporciona MQTT. Cuando la placa se desconecta de manera abrupta (sin mandar un MQTT disconnect), el broker avisará con un mensaje.
- Para el flujo de Telegram, antes de devolver un dato por mensaje se comprueba que el dato se ha obtenido correctamente, y no es nulo.

Librerías utilizadas

- **EEPROM**: La librería EEPROM nos permite leer y escribir bytes de la EEPROM de Arduino. Va a ser usada para escribir un número en la memoria y que se mantenga cuando esta se reinicie. Con este número podremos saber si el WiFi está configurado con anterioridad o es necesario usar una interfaz para que el usuario introduzca las credenciales de esta. También se ha utilizado esta librería para guardar los parámetros configurables del sistema: el estado del LED, el estado del switch, la velocidad de actualización, la velocidad con la que cambian la intensidad del LED y la velocidad del envío de datos.
- **WifiManager**: Sirve para configurar los parámetros de una red WiFi desde un dispositivo externo conectándose a la placa en modo AP. Todo esto sin tener que tocar ni una línea de código, simplemente rellenando un formulario en una página web. Se ha usado para la ampliación del proyecto, en el apartado de configuración WiFi.
- **ESP8266WiFi:** La librería permite conectar el dispositivo a una red WiFi para enviar y recibir datos. ha sido desarrollada basándose en el SDK de ESP8266, usando nombres convencionales y la filosofía de funcionalidades generales de la librería WiFi de Arduino.
- **ESP8266httpUpdate:** Esta librería nos permite realizar las actualizaciones FOTA. Permite al dispositivo conectarse a un servidor externo, comprobar si existe alguna actualización de software, descargarla e instalarla.

- **ArduinoJson y Arduino_JSON:** Son dos librerías utilizadas para la construcción de objetos JSON en Arduino (C/C++). Arduino_JSON es la librería oficial, pero ArduinoJson proporciona muchas más funcionalidades y es más sencilla de utilizar.
- **PubSubClient.h:** Es una librería de cliente para el protocolo MQTT. Este protocolo es ideal para entornos IoT, ya que es muy ligero. Está basado en un servicio de mensajería centralizado, con una estructura de publicador/subscriptor. Permite utilizar tanto MQTT 3.1.1, como el 3.1, lo que significa que dispone de las últimas funcionalidades.
- **DHTesp.h**: Es una librería de Arduino que permite leer los datos proporcionados por los sensores de la familia DHT.
- **Button2.h:** Esta librería permite simplificar el trabajo con botones. Permite utilizar funciones *callback* para pulsaciones simples, dobles, triples y de larga duración. También se ocupa de controlar el rebote de los botones.

Resultados y conclusiones

Como resultado se ha obtenido una solución completa en el ámbito de "Internet de las cosas" en la que se ha diseñado un prototipo a partir del módulo WiFi ESP8266, un sensor DHT11 y un programa arduino. Este dispositivo es capaz de leer datos recopilados por sensores, enviar información a una base de datos en MongoDB vía MQTT y recibir órdenes por medio de las cuales se permite cambiar su configuración y el estado de los actuadores conectados utilizando nuevamente el protocolo de comunicación MQTT. La aplicación diseñada permite por tanto controlar a varios dispositivos asociados que se encuentran distribuidos geográficamente (Málaga - Jaén - Almería), lo que forma una red de sensores y actuadores. Por otro lado la información será accesible por los usuarios a través de un dashboard diseñado en Node-RED, desde donde podrán interaccionar con los dispositivos conectados a la red, estas interacciones también están disponibles a través de un bot en Telegram.

Manual de usuario

En este proyecto se ha llevado a cabo la realización de un sistema IoT. La aplicación controlará una serie de dispositivos asociados (los integrantes de proyecto) que se encontrarán distribuidos geográficamente formando una red de sensores y actuadores de forma que cada sistema será capaz de leer datos de los sensores conectados, enviar información y recibir órdenes para cambiar su configuración y el estado de los actuadores que tiene conectados. Por otro lado la información de interés será accesible para los usuarios de una forma amigable utilizando: Una interfaz gráfica web (Dashboard) y otra conversacional en Telegram, además de accesos rápidos. Ambas interfaces junto con el

conexionado del sistema y los accesos rápidos van a ser explicados en este manual de usuario.

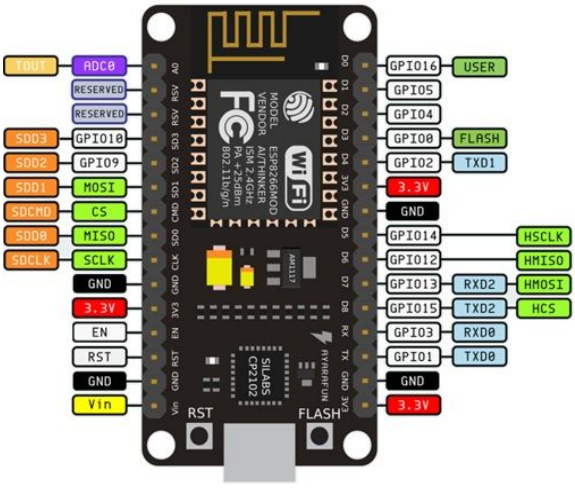
Tecnologías y conexionado:

Para el montaje del sistema IoT se requiere de:

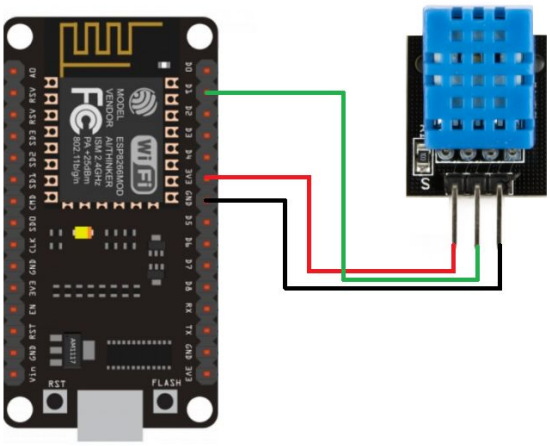
- Placa NodeMCU ESP8266
- Cable microusb para la conexión del dispositivo a la computadora
- Sensor DHT11
- Cable dupont hembra-hembra

Este sistema se encargará de medir, almacenar de forma temporal y transmitir de forma inalámbrica, los parámetros fundamentales a medir en cada estancia/ubicación (temperatura, humedad, etc.).

También debe funcionar como controlador de dos actuadores a través de dos GPIOs. Uno de ellos controlará una luminaria mediante PWM (GPIO2) y el otro se comportará como un interruptor, apagándose o encendiéndose (GPIO16). También será posible actuar localmente sobre el dispositivo pulsando el botón conectado al GPIO0.

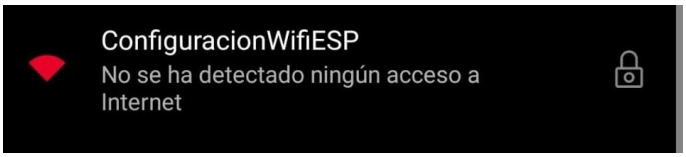


La conexión entre el sensor DHT11 y el microcontrolador se debe de realizar siguiendo las indicaciones de color de la imagen, es muy importante respetar estas conexiones:



Por último es importante saber que cuando el sistema se conecta a una alimentación eléctrica o se resetea, puede que tenga una red WiFi guardada o no. Esto es crucial ya que para el envío de datos a la aplicación del dashboard o de Telegram, se debe de tener acceso a internet.

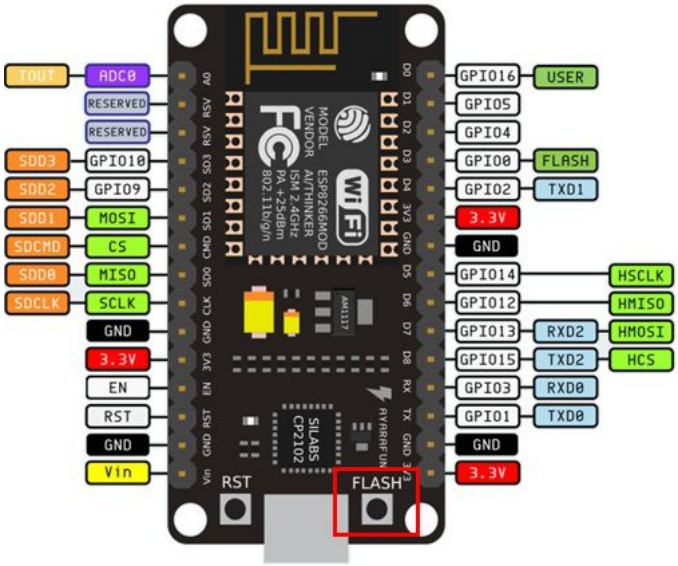
De este modo, si la configuración ya existe, el dispositivo va a conectarse a esa red. En caso contrario, va a crear un WiFi llamado "**ConfiguraciónWifiESP**" con clave "**password**" tal y como se puede apreciar en la figura X:



Cuando el usuario se conecte, le aparecerá una interfaz donde se le pedirán las credenciales del WiFi.

Al introducirlas, se guardan en el sistema y el dispositivo ya puede conectarse al WiFi de manera exitosa.

Existe una manera rápida con la que el usuario puede borrar la configuración del WiFi, de modo que deba introducir de nuevo las credenciales, tal cual se ha explicado anteriormente, para ello solo se tiene que pulsar tres veces seguidas el botón FLASH de la placa, situado en la esquina inferior derecha.



Accesos rápidos:

1. Una pulsación:

Si el usuario pulsa una única vez el botón flash señalado en la figura, el LED se apagará o encenderá con el nivel de intensidad establecido previamente.

2. Doble pulsación:

Si el usuario realiza una pulsación doble sobre el botón flash señalado en la figura, el LED se encenderá con el nivel máximo de intensidad (100).

3. Triple pulsación:

Si el usuario realiza una pulsación triple sobre el botón flash señalado en la figura, se reseteara la configuración WiFi guardada en el dispositivo, de modo que el usuario deberá de entrar en modo configuración para volver a especificar: SSID y PASSWORD.

4. Pulsación prolongada:

Si el usuario mantiene una pulsación prolongada sobre el botón flash señalado en la figura, el dispositivo comprobará si hay alguna actualización disponible (FOTA), para así actualizarse a la última versión.

Dashboard:

1. Selección del dispositivo:

Antes de comenzar el usuario debe de elegir sobre que dispositivo quiere realizar las consultas o las modificaciones, para ello existe un desplegable desde donde podrá elegir el ChipID de su dispositivo. También existe la opción de actuar sobre todos los dispositivos conectados.

2. DHT11 Sensores:

En esta sección del Dashboard, el usuario puede visualizar los últimos valores de humedad (%) y temperatura (ºC) que han sido captados por el sensor. Estas mediciones están nuevamente disponibles para todos los sistemas conectados, solo hace falta seleccionar el sistema del cual se quieren visualizar. Al final también se muestra la fecha de la última actualización y un histórico de 24 horas de todos los dispositivos conectados.

2.1. Modificar la frecuencia de envíos:

Como funcionalidad adicional, el usuario también puede modificar la frecuencia de envío de datos por parte del sensor conectado al dispositivo, incrementándola o reduciéndola, así como también el tiempo entre actualizaciones.

Por otro lado, el usuario también puede modificar la velocidad a la que quiere que se cambie la intensidad del LED.

2.2. Histórico:

Finalmente el usuario podrá visualizar un histórico de los valores de humedad y temperatura de todos los dispositivos conectados al sistema.

3. Consultas:

Desde esta sección del Dashboard, el usuario puede realizar consultas temporales a la base de datos, de este modo puede ver cómo ha evolucionado la temperatura o la humedad en un determinado período de tiempo. Nuevamente se puede elegir entre ver los datos que corresponden a un único dispositivo asociado a un ChipID o por el contrario

visualizar la media obtenida si se tienen en cuenta los valores de humedad y temperatura de todos los dispositivos conectados. Toda esta información es representada gráficamente.

3.1. Consultar por rangos:

Utilizando los selectores de fecha de inicio y de fin. El usuario puede visualizar: la temperatura y humedad media, máxima y mínima dentro de ese rango de fechas, además de una gráfica mostrando esos valores por días. También tenemos accesos directos a los rangos de fechas teniendo un botón para el último mes, para la última semana o para el último día.

3.2. Archivo Excel (csv):

Como funcionalidad adicional el usuario también podrá descargarse esos valores obtenidos tras filtrar por un período de tiempo en un archivo excel con formato .CSV, pudiendo elegir si quiere que los separadores sean comas o puntos. Para esto solo hay que pulsar en el botón "**Descargar registros en CSV**".

3.3. Gráfica comparativa:

Finalmente el usuario podrá visualizar los valores de humedad y temperatura en una gráfica comparativa.

4. LOGS:

En esta sección se muestra al usuario un registro de los cambios realizados y las alertas de error enviadas por el dispositivo:

4.1. Alertas de error:

En esta sección se muestran los posibles errores que puedan ocasionarse en las lecturas del dispositivo, se han contemplado:

- Lecturas nulas, por parte del sensor DHT11.
- Lecturas con valores de intensidad negativos del LED.
- Lecturas con valores diferentes de 0 o 1 para el switch.

4.2. Notificaciones de LED:

En esta sección se muestran notificaciones cada vez que se producen cambios en la configuración del LED.

4.3. Notificaciones de switch:

En esta sección se muestran notificaciones cada vez que se producen cambios en la configuración del switch

4.4. Gráfica comparativa:

Finalmente el usuario podrá visualizar los valores de intensidad del LED en una gráfica comparativa.

5. Configuración: LED/ Switch:

Desde esta sección del Dashboard, el usuario puede modificar la intensidad del LED (GPIO2) o apagar/encender el interruptor Switch (GPIO16). Estas dos acciones pueden hacerse para el propio dispositivo del usuario, seleccionado en el desplegable su ChipID correspondiente o por el contrario se pueden aplicar a todos los dispositivos.

5.1. Configuración del LED:

Para jugar con los niveles de intensidad lo único que hay que hacer es modificar el botón del slider, de este modo el LED se pondrá a la intensidad deseada, como información orientativa también se muestra la fecha y hora del último envío realizado para el estado del LED.

- Apagado. (LED = 0)
- Valores de intensidad intermedios. (0 < LED< 100)
- Máximo valor de intensidad. (LED = 100)

5.2. Configuración del Switch:

De la misma forma que podemos modificar el LED, también podemos controlar el interruptor del switch apagándolo o encendiéndolo, nuevamente esta acción puede realizarse sobre el propio dispositivo (Seleccionando nuestro ChipID) o sobre todos, como información orientativa también se muestra la fecha y hora del último envío realizado para el estado del switch.

En esta ocasión tenemos dos botones y una luminaria que nos informará del estado del Switch:

- Apagar Switch: Luminaria Gris.
- Encender Switch: Luminaria Azul.

5.3. Configuración de datos anómalos:

El usuario puede crear alertas bajo determinadas condiciones de temperatura o humedad, esto es muy útil si el usuario quiere centrarse en unos rangos determinados de valores. Cuando las condiciones se cumplan aparecerá un mensaje emergente al usuario.

Telegram:

El usuario también podrá interactuar con los dispositivos, así como visualizar los registros de cada uno de ellos, haciendo uso de un bot de Telegram propio "@grupo2iotbot".

Lo primero que se debe hacer es localizar el bot desde el buscador de Telegram:

Una vez encontrado, iniciamos una conversación con él utilizando el comando /start, de esta forma aparecerá una lista con las posibles funcionalidades que se enumeran a continuación:

- **/statusLED:** Este comando acompañado del nombre de usuario de la placa correspondiente, devolverá al usuario el nivel de intensidad del LED. ej: /statusLED Pablo. Si no se especifica ningún nombre devolverá el estado del LED en todos los dispositivos.
- **/statusSwitch:** Este comando acompañado del nombre de usuario de la placa correspondiente, devolverá al usuario el estado del interruptor Switch (encendido/apagado). ej: /statusSwitch Pablo. Si no se especifica ningún nombre devolverá el estado del switch en todos los dispositivos.
- **/values:** Este comando acompañado del nombre de usuario de la placa correspondiente, devolverá los últimos valores de temperatura y humedad recopilados por el sensor, pudiendo elegir cuántos valores se quieren visualizar. Si no se especifica ningún nombre devolverá la información contenida en todos los dispositivos.
- **/changeLED:** Este comando acompañado del nombre de usuario de la placa correspondiente, permitirá al usuario modificar la intensidad del LED, siempre dentro del rango de valores [0 - 100]. ej: /changeLED Pablo 75. Si no se especifica ningún nombre cambiará el estado del LED en todos los dispositivos.
- **/changeSwitch:** Este comando acompañado del nombre de usuario de la placa correspondiente, permitirá al usuario encender/apagar el interruptor Switch[0 - 1]. ej: /changeSwitch Pablo 1. Si no se especifica ningún nombre cambiará el estado del interruptor en todos los dispositivos.

El usuario también podrá unirse a una lista de difusión en la que se notificará de los cambios realizados sobre los dispositivos tanto por su parte como por la de otros usuarios.

Lista y descripción de los ficheros (.ino y .json)

Para terminar, haremos una lista de los ficheros utilizados y los describiremos brevemente.

El proyecto se compone básicamente de dos partes: una parte hecha con Node-RED, de los que se obtienen ficheros de tipo .json que recogen cada uno de los flujos creados, y una parte hecha en Arduino, de la que se obtiene un único fichero .ino.

Los flujos de Node-RED utilizados han sido los siguientes:

- **Telegram.json:** Este flujo recoge todo el funcionamiento de nuestro bot de Telegram. En él, se detalla cómo el usuario debe comunicarse correctamente con el bot y qué funcionalidades puede explotar, como por ejemplo: cambiar el valor de los actuadores, consultar sus valores o toda la información reciente recogida con los sensores de las distintas placas.
- **Dashboard.json:** En él se recoge todo el funcionamiento del dashboard. Es decir, controla el interfaz gráfico con el que se comunicará el usuario, así como las respuestas asociadas a este. Este flujo debe procesar cuidadosamente la información recibida mediante mqtt para mostrarla al usuario de forma agradable, a la vez que debe estar atento a las órdenes que pueda enviarle el usuario para transmitirlas a la placa correspondiente mediante mqtt nuevamente.
- **BBDD.json:** Con este flujo únicamente se recogen los valores interesantes de mqtt y se almacenan en la base de datos para su posterior uso en otros ficheros como los dos mencionados anteriormente.

La ventaja de este tipo de ficheros es que al importarlos a Node-RED, se ve detalladamente el esquema utilizado y cómo funciona cada una de las diferentes partes que lo componen de una forma bastante sencilla y agradable, por lo que no es necesario comentar exhaustivamente dichos flujos.

En cuanto a Arduino, el fichero .ino controla todo el funcionamiento relacionado con las distintas placas y sensores. En él se detalla cómo las placas reciben la información, cómo la procesan y cómo la envían. Además de tener en cuenta la información procedente de los sensores y los actuadores de la propia placa, debe poder comunicarse correctamente con el usuario, tanto desde Telegram como desde el dashboard, por lo que debe implementar todo el código necesario para trabajar con mqtt.

En este caso, este fichero está programado en el lenguaje C, por lo que para entender cómo funciona cada proceso es necesario explicarlo detalladamente además de ordenar el código para hacerlo más legible para aquellos que no tienen conocimientos profundos en programación, incluso para los que sí lo tienen.