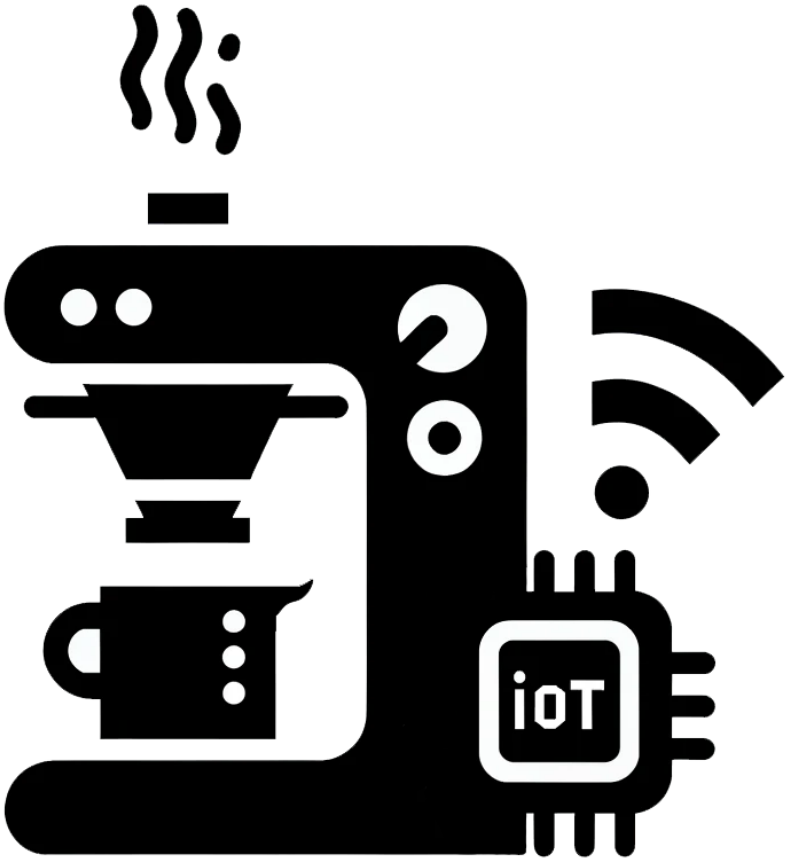
Proyecto Informática Industrial - Cafetera IOT

Roberto Carlos Barnhardt

10 de enero de 2024

En este proyecto se han integrado funcionalidades propias de los dispositivos IOT en una cafetera tradicional de filtro, incluyendo sensores y actuadores para controlar el estado y recibir datos remotamente. Como base se ha utilizado la cafetera *Melitta Easy* y el microcontrolador utilizado ha sido el *Espressif ESP32*.



Índice

1. Introducción.

Al comienzo del proyecto se plantearon varios objetivos, los cuales se han cumplido en distintos niveles.

- Capacidad de encender el dispositivo sin estar en contacto físico.
- Posibilidad de programar una hora de comienzo del ciclo.
- Colocación de sensores para conocer el estado de la máquina en remoto.
- Incluir funcionalidades del proyecto ejemplo del CV.
  - Comunicación vía MQTT.
  - Actualización FOTA.
  - Control mediante bot de Telegram.
  - Sistema SCADA en Dashboard de Node-RED.
  - Almacenaje y recuperación de datos de una base de datos MongoDB.

2. Esquema general.

A continuación se describirá el esquema global de la cafetera, microcontrolador, sensores y actuadores, así como el estado actual de construcción.

2.1. Electrónica.

Inicialmente no estaba del todo claro cuáles serían los componentes que compondrían el circuito final, lo mínimo indispensable es un microcontrolador, con capacidad para comunicarse con algún tipo de actuador para controlar la máquina, y el propio actuador. Durante el progreso del proyecto han surgido necesidades e ideas que han llevado a aumentar la cantidad de dispositivos, con ello ampliando las funcionalidades. El circuito final contiene los siguientes componentes:

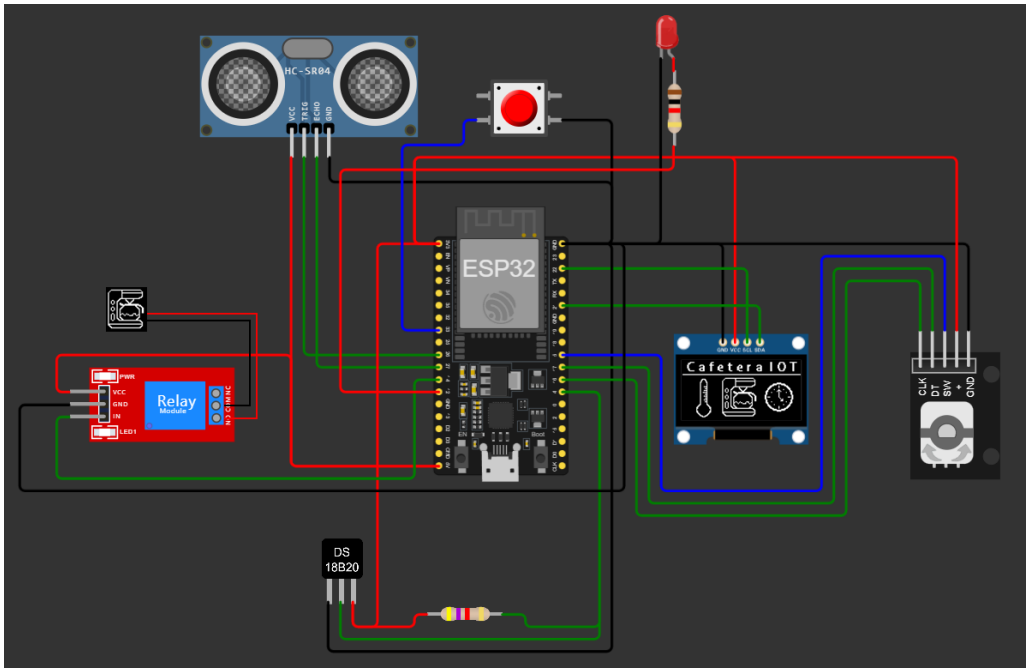


Figura 1: Diagrama circuito electrónico.

- **ESP 32:** Es un potente microcontrolador de la casa Espressif, cuenta con conectividad WiFi y Bluetooth, además de 25 pines GPIO, de los cuales todos pueden funcionar como pines de interrupción.

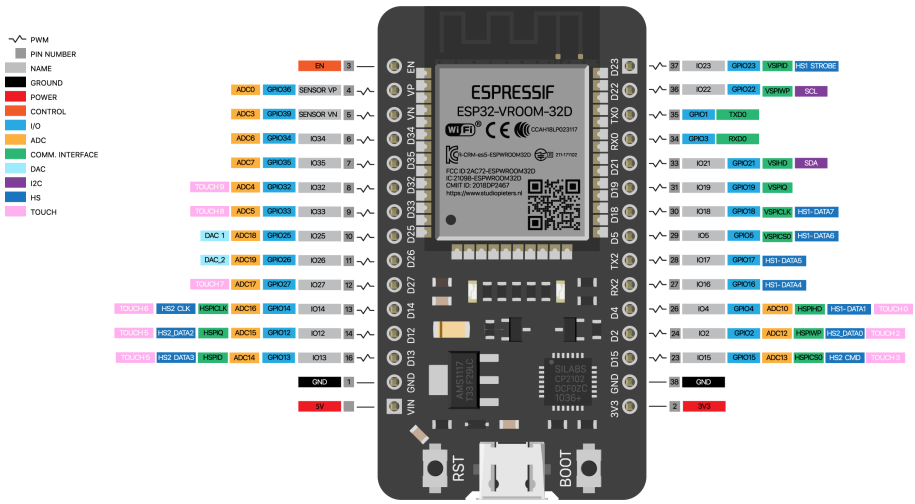


Figura 2: Esquema pines ESP32.

- **Modulo relé:** Gracias a ello podemos fácilmente conmutar una fuente de 230V AC utilizando solamente un pin GPIO de la ESP32. En este caso se ha utilizado un módulo doble.

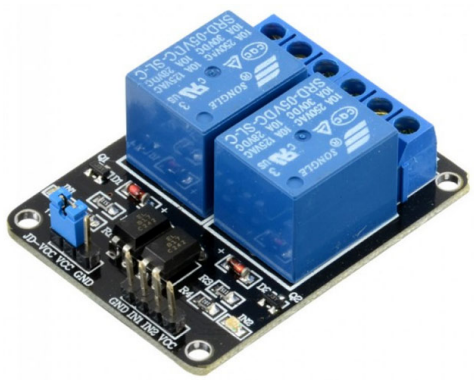


Figura 3: Modulo relé dos canales.

- **Sensor de temperatura DS18B20:** Este sensor integrado nos proporciona una señal digital con hasta 12 bits de resolución, con ello evitamos los problemas clásicos de lectura de sensores analógicos debido a la mala calidad de los convertidores analógico-digital del microcontrolador. Esta unidad cuenta además con un recubrimiento impermeable, gracias a ello es apto para mediciones sumergidas.

Figura 4: Sensor DS18B20.

- **Display OLED SD1306:** Este pequeño display hace posible mostrar al usuario todo tipo de datos y opciones. Este modelo en particular cuenta con una resolución de 128x64px. Está

dividido en dos colores, una banda superior amarilla de 128x16px, y una zona inferior azul de 128x48px.

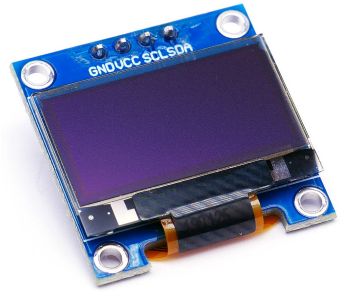


Figura 5: Display SD1306.

- **Rotary encoder:** Este encoder tiene 3 pines de salida, dos para identificar rotación y dirección, y uno que funciona como botón. Es una opción compacta que introduce mucha flexibilidad de inputs para el usuario.

Figura 6: Rotary encoder.

- **Sensor ultrasonidos HC-SR04:** Este sensor nos permite medir distancias con una precision de unos 2-3mm. En este proyecto se ha utilizado para medir la altura del agua en el depósito.

Figura 7: Sensor ultrasonidos.

- **Microswitch:** Este interruptor tiene una palanca que permite incrementar el rango de funcionamiento. Tras un pequeño accidente realizando pruebas fue necesario poder determinar en un momento dado si la taza está correctamente colocada o si está ausente.

Figura 8: Microswitch.

2.2. Construcción.

El desarrollo comenzó con el menú en el display y el encoder, para conseguir primero un control local antes de incorporar comunicación inalámbrica y control remoto. En un primer instante se trabajaba sobre una protoboard pero este estaba bastante desgastado provocando mal contacto con el encoder. Para lidiar con esto se soldó el encoder y headers para el display a una placa de desarrollo.

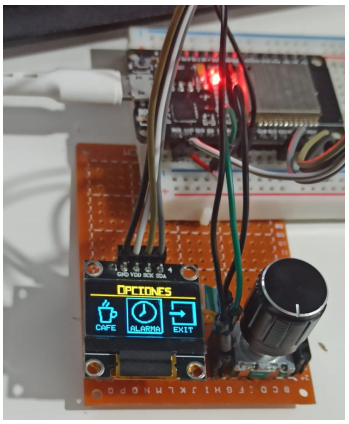


Figura 9: Primer prototipo.

Tras soldar también headers para la ESP32 en la placa, resultó ser bastante inflexible para hacer modificaciones por lo que para la segunda versión se volvió a protoboard, esta vez nuevas para asegurar el buen contacto.

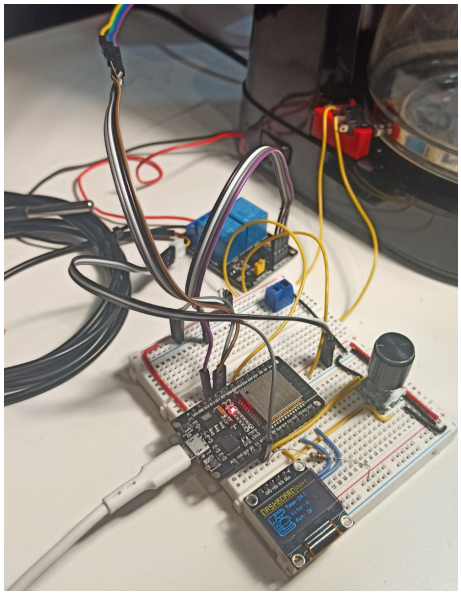
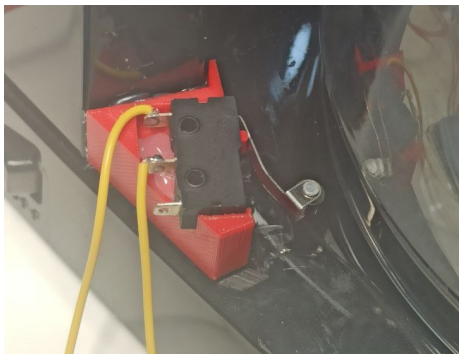


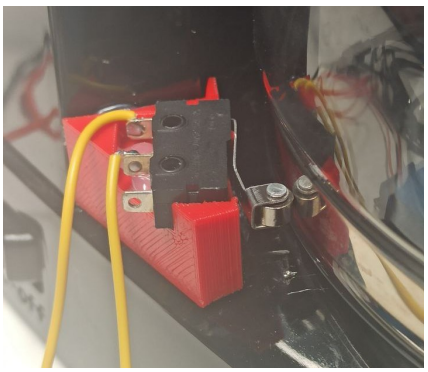
Figura 10: Segundo prototipo.

En cuanto al sensor térmico DS18B20, se experimentó colocándolo en distintas posiciones, dentro y fuera de la máquina, pero finalmente el único sitio que proporcionaba una lectura fiel y útil de la temperatura del café, era colocándolo directamente dentro de la taza. Esto es posible gracias a que es impermeable pero en el futuro se buscará una solución más elegante.

Se ha utilizado una pequeña pieza impresa en 3D para colocar correctamente el microswitch para detectar la presencia de la taza.

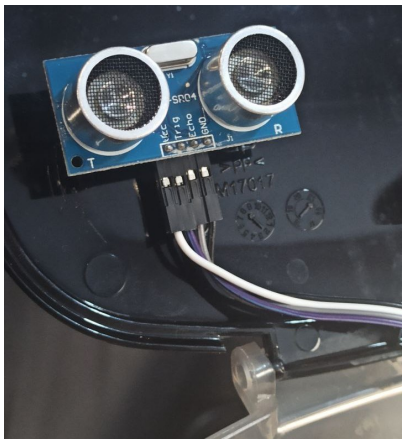


(a) Microswitch posicion abierto.



(b) Microswitch posición cerrado.

En cuanto al sensor ultrasónico, se encuentra unido con adhesivo a la tapa del depósito de agua, tal que en posición cerrada apunta hacia el fondo del depósito.

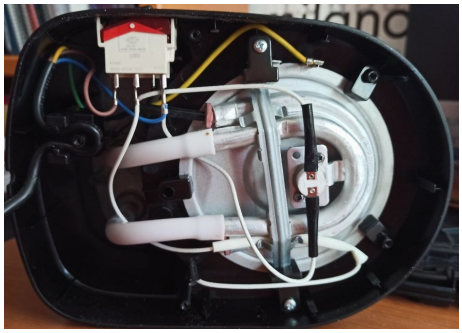


(a) Ultrasonidos con tapa abierta.

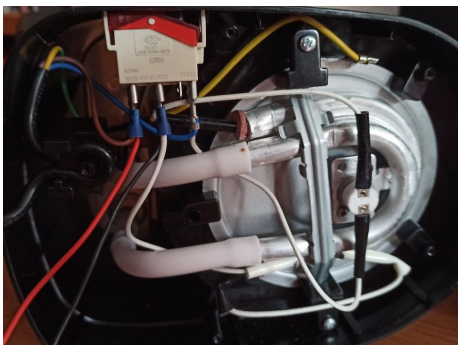
(b) Ultrasonidos con tapa cerrada.

En la práctica, este sensor ha demostrado no ser la mejor solución a largo plazo para desempeñar esta función, ya que enseguida el agua del deposito se evapora y se condensa en la malla que protege los actuadores ultrasónicos, tapándolos e inhibiendo su correcto funcionamiento. Además, el entorno húmedo podría llegar a oxidar rápidamente los conductores descubiertos de la PCB. No obstante, esta instalación ha sido más que suficiente para probar el funcionamiento dentro del sistema global.

Respecto al circuito de 230V, es bastante simple, la máquina tiene un elemento resistivo calefactor en serie con un termostato y el interruptor. Al ponerse en marcha se calienta a máxima potencia hasta alcanzar la temperatura del termostato, momento en el que se abre el circuito y deja de circular corriente hasta que el termostato vuelve a estabilizarse.
Para conseguir control sobre este proceso se ha instalado el relé en paralelo con el interruptor ya existente de fábrica, por ello, si se desea, se puede utilizar con normalidad la cafetera como si no se hubiese realizado ninguna modificación.



(a) Circuito antes de modificar.



(b) Circuito después de modificar.

3. Software.

Todo el material del proyecto está incluido en un repositorio de Github que se encuentra en la bibliografía.

3.1. Arduino.

El programa .ino instalado en la ESP32 tiene una estructura de tipo máquina de estados, cada posible función que puede desempeñarse tiene un estado asignado. Resumidamente, se pueden categorizar las funciones del programa de la siguiente manera:

3.1.1. Interrupciones.

En la ESP32 se están utilizando 4 pines de interrupción externa por hardware:

- Dos para la rotación del encoder, utilizando dos pines podemos saber la dirección en la que ha rotado. Gracias a ello podemos utilizarlo para incrementar o decrementar un contador de posición para navegar por el menú.
- El tercer pin de interrupción también es para el rotary encoder, pero este para el botón, con él se seleccionan los elementos dentro del menú.
- Finalmente, el cuarto pin de interrupción es para el microswitch de la taza, cuando este se activa, en la siguiente iteración del bucle principal se comprueba el estado.

3.1.2. Display.

Para mostrar información por el display y construir el menú se han utilizado las librerías *Adafruit_GFX.h* y *Adafruit_SSD1306.h* de **Adafruit Industries**.

El menú se ha construido incluyendo una combinación de funciones de las librerías, como mostrar texto, lineas o cuadrados; y bitmaps customizados, como fondos de pantalla o iconos. El procedimiento para crear estos bitmaps consiste en primero crear el dibujo, en este caso se ha utilizado **Adobe Photoshop 2021**, partiendo de una plantilla del tamaño del display (128x64px).

Figura 14: Ejemplo de bitmap.

Una vez dibujado lo que sea desea se puede exportar en formato PNG y mediante una herramienta online, en este caso, esta herramienta creado por Marlin, un firmware Open Source para impresoras 3D, se puede convertir el archivo PNG a bitmap, el cual la librería de Adafruit puede interpretar y mostrar por pantalla. Se han creado funciones para gestionar las distintas pantallas del menú para facilitar la lectura del bucle principal.

Figura 15: Fondo con iconos incluidos.

3.1.3. MQTT.

Se han establecido varios Topics para llevar a cabo distintas funcionalidades del programa:

- **"cafeteraiot/alarm/status"** - Formato: {"status": "on", "time": "23:30"}. Este Topic representa el estado actual de la alarma, o hora de arranque programado.
- **"cafeteraiot/coffee/status"** - Formato: {"status": "off"}. Este topic guarda el estado de encendido de la cafetera.
- **"cafeteraiot/coffee/temp"** - Formato: {"temp": "65"}. Aquí se publica la temperatura actual medida por el sensor DS18B20
- **"cafeteraiot/water_level"** - Formato:{"level": "90"}. Se mapea la distancia medida por el sensor ultrasonidos a un porcentaje [0-100 %] de llenado del depósito.
- **"cafeteraiot/pot"** - Formato:{"status": "ok"} o {"status": "missing"}. Se actualiza este topic con el estado actual de posición de la taza.
- **"cafeteraiot/FOTA"** - Formato:{"update": "true"}. Cuando "update" tiene el valor "true", se comprueba la existencia de actualizaciones FOTA en la próxima entrada al bucle principal. Una vez comprobado, se publica "false".

3.1.4. FOTA.

Se ha implementado funcionalidad FOTA *(Firmware Over The Air)* en el proyecto. En el arranque se comprueba si la última versión del firmware es la más reciente, si hubiese una más nueva se actualizaría. Además, el cliente MQTT está suscrito a un Topic que cuando se recibe un mensaje con el campo {*"update": "true"*} en la próxima ejecución del bucle se comprueba la existencia de actualización.

3.2. Node-RED.

3.2.1. Flujos.

Para este proyecto existen dos flujos de Node-RED operativos simultáneamente. Por un lado el flujo principal, que incluye la gestión de MQTT, MongoDB, Dashboard y Telegram. Y por otro lado el flujo para FOTA, tomado de una práctica anterior.

En el flujo principal existen varios subflujos agrupados por funciones:

- **Subscripciones MQTT + introducción en base de datos.** En este grupo se reciben datos de todos los Topics utilizados en el proyecto, posteriormente se añade la fecha actual y se almacenan, en algunos casos se asignan a variables globales porque pueden ser requeridos en otros nodos.

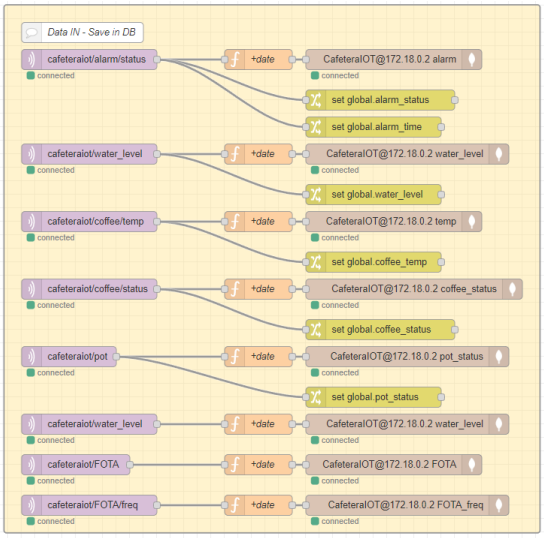


Figura 16: Flow MQTT + MongoDB

- **Comprobación de alarma.** Aquí se comprueba si el último mensaje almacenado en "cafeteraiot/alarm/status" contenía {"status":"on"}, y si es así, se compara ["time"] con la hora actual para determinar si es hora de encender la cafetera o no.

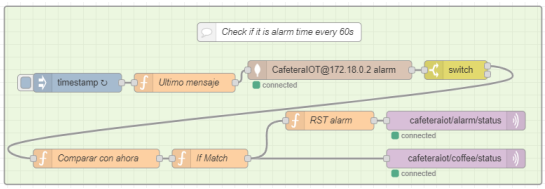


Figura 17: Flow comprobación alarma

- **Comprobación de FOTA.** Se mira cuándo fue la última vez que se comprobó la existencia de actualizaciones. Luego se calcula la diferencia temporal entre el momento actual y la última vez que se comprobó, si es mayor que el periodo establecido en "cafeteraiot/FOTA/freq", se vuelve a comprobar.

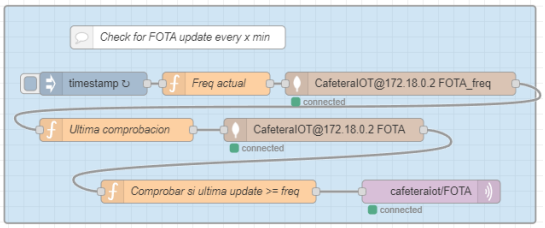


Figura 18: Flow comprobación FOTA

- **Comprobación de timeout**. El interruptor de fábrica de la cafetera tiene un temporizador interno de 40 minutos, tras 40 minutos encendido se apaga automáticamente. El funcionamiento de este flujo es igual, si hace más de (tiempo máximo) minutos desde que se encendió la máquina, se apaga.

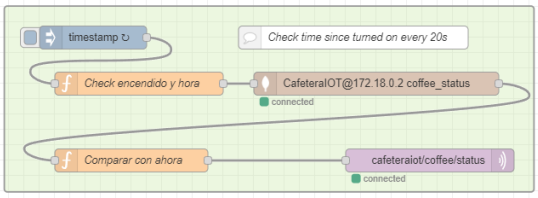


Figura 19: Flow timeout

- **Gestión Bot de Telegram.** Este flujo maneja los comandos del Bot de Telegram que permite encender y apagar la máquina, modificar la alarma y solicitar datos remotamente.

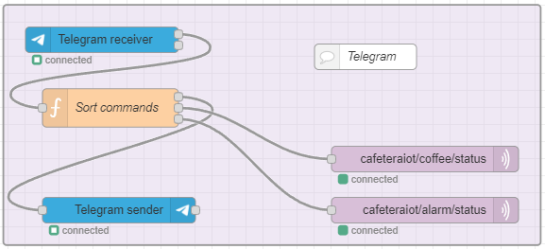


Figura 20: Flow Telegram

3.2.2. Dashboard.

El Dashboard de Node-RED finalmente es bastante simple. Nos ofrece información imporante como el nivel del agua, el estado actual de encendido de la máquina, una gráfica histórica de temperatura, el estado de la taza y un switch para encender y apagar la máquina.

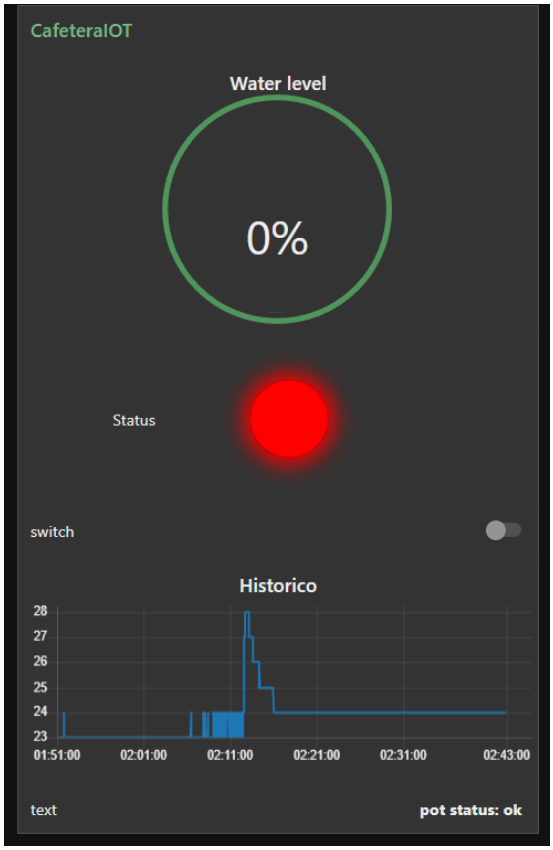


Figura 21: Node-RED dashboard

3.3. Telegram.

Se ha creado un Bot de Telegram con ID **CafeteraIOT_bot**. Desde este bot es posible encender y apagar, programar la alarma y obtener datos acerca del estado de la máquina. Se tiene en cuenta el nivel del agua, el estado de la taza y si la máquina está o no encendida a la hora de solicitar el encendido desde el Bot.

Figura 22: Ejemplo de chat con el Bot

4. Puntos a mejorar.

Ya acabado el proyecto, debido a limitaciones de recursos y tiempo, hay una serie de cosas en las que se podría trabajar en el futuro para mejorar.

- En cuanto al sensor de nivel de agua, ya se ha mencionado que el tipo utilizado no parece ser la mejor opción para este sistema. Una mejora podría ser utilizar un sensor ultrasónico mejor encapsulado y resistente a la humedad, del tipo que se utilizan en los coches para la ayuda al aparcamiento. Otra opción podría ser usar un sensor de tipo flotador.
- Crear una PCB final, ya sea de tipo prototipo creado manualmente o diseñado y pedido a un fabricante.
- Caja para contener la PCB, ordenar mejor el cableado y posicionar óptimamente el display y el encoder. Fabricado por ejemplo con impresión 3D.
- Interfaz más permanente entre en sensor de temperatura y el café. La situación actual de sumergir el sensor no es idóneo, sería preferible que estuviera conectado al exterior de la taza, pero con un buen contacto térmico. Además debe ser facil de quitar y poner para lavar la taza, por ejemplo con imanes.
- Expandir posibles funcionalidades con Dashboard de Node-RED e integrar funcionamiento con Amazon Alexa.

5. Bibliografía.

- BenoitBlanchon. “ArduinoJson: Efficient JSON Serialization for Embedded C++.” ArduinoJson, arduinojson.org/.
- MarlinFirmware. “Bitmap to C/C++ Converter.” Marlin Firmware, marlinfw.org/tools/u8glib/converter.html.
- Adafruit. “Adafruit/Adafruit_SSD1306.” GitHub, 11 Jan. 2020, github.com/adafruit/Adafruit_SSD1306.
- Arduino. “Arduino Reference.” Arduino.cc, 2019, www.arduino.cc/reference/en/.
- “Wokwi - Online Arduino and ESP32 Simulator.” Wokwi.com, wokwi.com/.
- Hansen, Mathias Munk. “DS18B20.” GitHub, 21 June 2023, github.com/matmunk/DS18B20.
- Harrington, Kevin. “ESP32Encoder.” GitHub, 24 Apr. 2023, github.com/madhephaestus/ESP32Encoder.
- Roberto-Barnhardt. “Roberto-Barnhardt/CafeteraIOT.” GitHub, 10 Jan. 2024, github.com/Roberto-Barnhardt/CafeteraIOT/. Accessed 10 Jan. 2024.