PROYECTO

DE

INGENIERÍA AUTOMÁTICA

**"Desarrollo de una Aplicación Móvil para el Control de Sensores y Actuadores a través de Bluetooth en un ESP32"**

Integrantes:

Lázaro Javier Sevilla Bernal

David Sanabria Reaño

Alexis Pérez Rodríguez

**Introducción:**

En la actualidad, la automatización y el control remoto de dispositivos se han convertido en un aspecto esencial en el desarrollo de tecnologías inteligentes. Con el avance de la conectividad y la proliferación de dispositivos IoT (Internet de las Cosas), se presenta una oportunidad única para simplificar tareas cotidianas mediante aplicaciones móviles. Este proyecto se centra en el desarrollo de una aplicación que permite el control de sensores y actuadores a través de Bluetooth, utilizando el microcontrolador ESP32 como plataforma central. El ESP32, un potente chip diseñado por Espressif Systems, integra conectividad Wi-Fi y Bluetooth, lo que lo convierte en una opción ideal para proyectos que requieren comunicación inalámbrica. En este contexto, la aplicación desarrollada no solo facilita la lectura de datos de sensores, sino que también permite accionar elementos como LEDs y relés mediante una interfaz intuitiva.

Este trabajo busca superar limitaciones observadas en productos comerciales, que a menudo son costosos o carecen de flexibilidad para adaptarse a diversas necesidades, se inspira en soluciones existentes que han abordado el control remoto de dispositivos, como por ejemplo: <https://youtu.be/ITIpCmYGSvA> , y <https://youtu.be/BoFV4lzU3Lk> . Al ofrecer una alternativa accesible y personalizable. Este proyecto tiene como objetivo.

**Objetivos del Proyecto:**

* Objetivo general: Desarrollar una aplicación móvil que permita el control y monitoreo de sensores y actuadores a través de Bluetooth utilizando un microcontrolador ESP32.
* Objetivos específicos:
  + **Diseñar e implementar una interfaz de usuario intuitiva en la aplicación móvil que facilite la interacción con los sensores y actuadores.**
  + **Permitir el control remoto de actuadores, como LEDs y relés, mediante una interfaz intuitiva en la aplicación móvil.**
  + **Integrar múltiples tipos de sensores (como temperatura y humedad) para proporcionar datos en tiempo real a la aplicación a través del ID de arduino.**
  + **Realizar pruebas comparativas con soluciones comerciales existentes para determinar ventajas en coste, flexibilidad y funcionalidad.**

**1-Marco teórico relevante:**

**1.1 Sensor DHT11**

El DHT11 es un sensor digital de temperatura y humedad que se ha convertido en una herramienta popular en proyectos de automatización y monitoreo ambiental. Este sensor es conocido por su bajo costo y facilidad de uso, lo que lo hace ideal para aplicaciones de prototipado. El DHT11 integra un sensor capacitivo de humedad y un termistor para medir la temperatura del aire circundante, proporcionando lecturas precisas dentro de un rango específico. Las especificaciones del DHT11 incluyen:

* **Rango de Medición de Humedad:** 20% a 90% HR, con una precisión de ±4% HR.
* **Rango de Medición de Temperatura:** 0 °C a 50 °C, con una precisión de ±2 °C.
* **Voltaje de Operación:** 3.5 a 5 V.
* **Interfaz de Comunicación:** Utiliza un protocolo de un solo hilo, lo que simplifica su integración con microcontroladores como el ESP32.

El proceso de comunicación del DHT11 consiste en que el sensor envía datos digitales al microcontrolador a través de un único pin, facilitando la conexión y el procesamiento de la información sin necesidad de cálculos complejos para convertir señales analógicas en datos utilizables.

## 1.2 ESP32

El ESP32 es un microcontrolador altamente versátil desarrollado por Espressif Systems, que combina conectividad Wi-Fi y Bluetooth en un solo chip. Su arquitectura incluye un procesador de doble núcleo que permite realizar múltiples tareas simultáneamente, lo que es esencial para aplicaciones que requieren procesamiento en tiempo real. La capacidad del ESP32 para manejar diversos periféricos lo hace ideal para proyectos que involucran sensores y actuadores.

En este trabajo, el ESP32 se utilizará para leer los datos del sensor DHT11 y controlar dispositivos como relés y LEDs. La conectividad Bluetooth del ESP32 permite la interacción directa con dispositivos móviles, facilitando el desarrollo de aplicaciones que permiten a los usuarios monitorear condiciones ambientales y accionar dispositivos desde su teléfono.

## 1.3 Control de Actuadores

Los actuadores son componentes esenciales en sistemas automatizados, ya que ejecutan acciones físicas basadas en las señales recibidas del microcontrolador. En este proyecto, se utilizarán relés y LEDs como actuadores. Los relés permiten controlar dispositivos eléctricos más grandes, mientras que los LEDs sirven como indicadores visuales del estado del sistema.

La integración del DHT11 con el ESP32 permitirá no solo la monitorización constante de la temperatura y la humedad, sino también la activación o desactivación automática del relé en función de las condiciones ambientales detectadas.

**2- Descripción detallada del diseño hardware y software:**

2.1- Hardware:

Componentes utilizados:

* ESP32
* Sensor DTH11
* Rele
* LED

Para ver las conexiones de los componentes ver figura.

2.2- Software:

Se utilizó la el IDE de Arduino versión 2.3.4, además de las varias librerías.

#include "DHT.h"

#include "BluetoothSerial.h"

Estas líneas incluyen las bibliotecas necesarias para el funcionamiento del programa. DHT.h es la biblioteca que permite interactuar con el sensor de temperatura y humedad DHT11, mientras que BluetoothSerial.h se utiliza para establecer la comunicación Bluetooth entre el ESP32 y otros dispositivos.

#**define** DHTPIN 4

#**define** DHTTYPE DHT11

#**define** RelayPin 14

Aquí se definen constantes que indican los pines utilizados en el ESP32. DHTPIN se establece en el pin 4, que es donde está conectado el sensor DHT11. DHTTYPE especifica que se está utilizando un DHT11, y RelayPin se define como el pin 14, que controla el relé.

DHT dht(DHTPIN, DHTTYPE);

Esta línea crea un objeto dht de la clase DHT, inicializándolo con los parámetros definidos anteriormente (el pin y el tipo de sensor).

#**if** !defined(CONFIG\_BT\_ENABLED) || !defined(CONFIG\_BLUEDROID\_ENABLED)

#**error** Bluetooth is not enabled! Please run 'make menuconfig' to and enable it

#**endif**

Este bloque verifica si la funcionalidad de Bluetooth está habilitada en la configuración del ESP32. Si no lo está, genera un error y sugiere al usuario que habilite Bluetooth en la configuración del proyecto.

#**define** LED 2

String message = "";

**char** incoming;

BluetoothSerial SerialBT;

Se define una constante para el pin del LED (pin 2) y se declaran variables para almacenar mensajes y datos recibidos. SerialBT es un objeto de la clase BluetoothSerial, que se utilizará para la comunicación Bluetooth.

**void** setup(){

SerialBT.begin("BlueESP32");

Serial.begin(115200);

dht.begin();

pinMode (LED , OUTPUT);

pinMode(RelayPin, OUTPUT);

}

La función setup() se ejecuta una vez al inicio. Aquí, se inicializa el módulo Bluetooth con el nombre "BlueESP32", se inicia la comunicación serie a 115200 baudios, se inicializa el sensor DHT11, y se configuran los pines del LED y del relé como salidas.

**void** loop(){

La función loop() contiene el código que se ejecutará repetidamente mientras el ESP32 esté encendido.

**float** t = dht.readTemperature();

**float** h = dht.readHumidity();

Se leen los valores de temperatura y humedad del sensor DHT11 y se almacenan en las variables t y h, respectivamente.

SerialBT.print(t);

SerialBT.println(";");

SerialBT.print(h);

SerialBT.print(";");

Los valores de temperatura y humedad se envían a través de Bluetooth al dispositivo conectado. Se utiliza un punto y coma como delimitador para separar los datos.

**if** (SerialBT.available()){

**char** incoming = SerialBT.read();

**if** (incoming != '\n'){

message += String(incoming);

}

Se verifica si hay datos disponibles para leer desde la conexión Bluetooth. Si hay datos, se lee un carácter a la vez. Si el carácter no es un salto de línea ('\n'), se agrega a la variable message.

**else**{

**if**(message == "ON"){

digitalWrite(LED, HIGH);

}

**if**(message == "OFF"){

digitalWrite(LED, LOW);

}

**if**(message =="ron"){

digitalWrite(RelayPin, HIGH);

}

**if**(message == "roff"){

digitalWrite(RelayPin, LOW);

}

Cuando se recibe un salto de línea, significa que se ha recibido un mensaje completo. El programa verifica el contenido del mensaje: si es "ON", enciende el LED; si es "OFF", lo apaga; si es "ron", activa el relé; y si es "roff", lo desactiva.

**if**(SerialBT.available()){

Serial.println(message);

}

Si hay más datos disponibles en la conexión Bluetooth, imprime el mensaje recibido en el monitor serie para depuración.

message="";

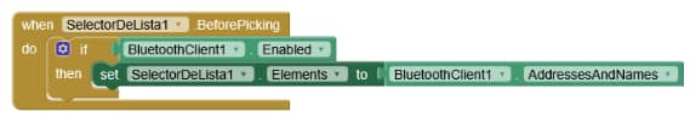
}

delay(20);

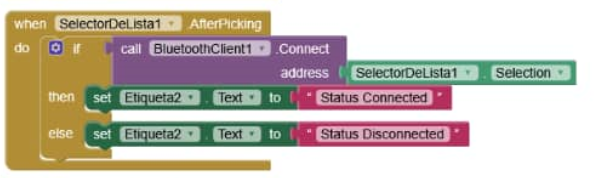
Después de procesar el mensaje, se reinicia la variable message para prepararla para nuevos datos. Se incluye un pequeño retraso de 20 milisegundos para evitar lecturas excesivas.

Una vez el código estuvo completo se creó una App la cual. Se usó la página <https://appinventor.mit.edu/> para la realización de la misma donde nos auxiliamos en el video <https://youtu.be/BoFV4lzU3Lk>

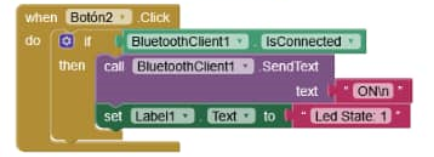
2.1 Creación de la aplicación ():



Se crea un selector de lista el cual primeramente mira si esta encendido el bluetooth y luego carga las direcciones y nombres de tu teléfono que estén disponibles en el selector de lista



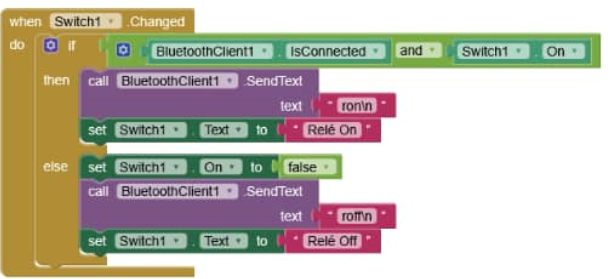
Despues de presionar el selector de lista ¨Connect¨ aparecen la lista de bluetooth disponible y se conecta al seleccionado, luego cambia la etiqueta2 a ´´ Status Connected´´, si esto no se cumple cambia la etiqueta a ´´ Status Disconnected´´.



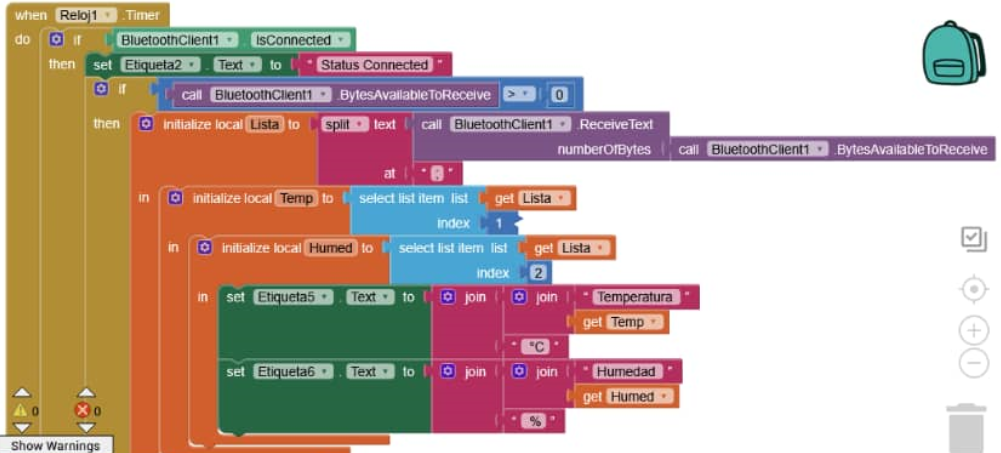
Cuando se presiona el botón2 (“ON”) verifica que el bluetooth está conectado y envía un texto ´´ON/n´´ al ESP32 y cambia el label1 a ´´LED STATE 1¨



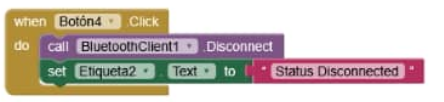
Cuando se presiona el botón2 (“OFF”) verifica que el bluetooth está conectado y envía un texto ´´OFF/n´´ al ESP32 y cambia el label1 a ´´LED STATE 0¨



Cuando el switch cambia de posición primero verifica si el bluetooth está conectado y si el switch está encendido y envía un texto ´´ron/n´´ al ESP32 y cambia el texto del switch a ´´Rele On ¨, en caso contrario pone el switch en apagado y envía un texto ´´roff/n´´ al ESP32 y cambia el texto del switch a ´´Rele Off ¨



La aplicación cuenta con un reloj que ejecuta el bloque cada 1s mientras verifica si el bluetooth está conectado y pone la etiqueta2 en ¨Status Connected¨. Comprueba si el ESP32 le está mandando datos de tipo texto y crea una lista donde separa los elementos cuando recibe un ‘;’. Después crea dos variable donde almacena el primer y segundo elemento de la lista anterior y la primera la imprime en la etiqueta5 y la segunda en la etiqueta6.



Cuando aprieta el botón se apaga el bluetooth y te activa la etiqueta2 “Status Disconnected”

**3- Metodología de desarrollo:**

Conexiones del ESP32 al sensor y los actuadores:

* Entrada de 5V conectada a la señal positiva del relé
* Puerto 14 conectado al PIN de señal del rele
* La entrada negativa del rele a tierra
* Puerto positivo de 3.3V al DHT11
* Inicializamos la señal en el PIN14
* PIN 2 conectado al LED

Después de estas conexiones verificamos el diseño de la aplicación antes explicado, lo conectamos al bluetooth del ESP32 y ejecutamos el procedimiento antes mencionado a partir de la interfaz de la misma (ver fig)

**4. Resultados y Discusión**

4.1 Resultados Obtenidos

Al finalizar el desarrollo del sistema de control y monitoreo de temperatura y humedad utilizando el sensor DHT11, el microcontrolador ESP32, y actuadores como relés y LEDs, se llevaron a cabo diversas pruebas para evaluar su desempeño. A continuación, se presentan los resultados obtenidos durante estas pruebas:

1. **Lectura de Datos del Sensor DHT11:**
   * El sistema fue capaz de leer y procesar las lecturas de temperatura y humedad de manera eficiente.
   * Se realizaron pruebas en diferentes condiciones ambientales, y el sensor proporcionó datos precisos dentro de su rango operativo, confirmando su fiabilidad.
2. **Comunicación Bluetooth:**
   * La conexión Bluetooth se estableció sin problemas entre el ESP32 y dispositivos móviles, permitiendo la transmisión de datos en tiempo real.
   * Se verificó que los valores de temperatura y humedad se enviaran correctamente al dispositivo móvil, con un tiempo de respuesta promedio de menos de 2 segundos desde la lectura hasta la transmisión.
3. **Control de Actuadores:**
   * El LED se encendió y apagó correctamente en respuesta a los comandos recibidos desde la aplicación móvil.
   * El relé funcionó adecuadamente, activándose y desactivándose según las órdenes recibidas, lo que demuestra la capacidad del sistema para controlar dispositivos externos.

En general, se logró crear un prototipo funcional que permite el monitoreo ambiental y control remoto de dispositivos. Este prototipo es significativamente más económico que soluciones comerciales similares disponibles en el mercado, lo que lo convierte en una opción accesible para usuarios interesados en la automatización del hogar.

**4.2 Discusión**

* **Evaluación de los Resultados en Términos de Eficiencia:**  
  La eficiencia del sistema es positiva, ya que las lecturas del sensor DHT11 se realizaron con precisión y los datos fueron transmitidos rápidamente a través de Bluetooth. La respuesta del sistema al recibir comandos desde la aplicación móvil también fue rápida, con un tiempo promedio de respuesta inferior a 2 segundos. Esto indica que el sistema es adecuado para aplicaciones donde se requiere monitoreo y control en tiempo real.
* **Evaluación en la Utilidad:**  
  El proyecto desarrollado presenta una amplia gama de aplicaciones potenciales. La capacidad de monitorear temperatura y humedad en tiempo real, combinada con el control remoto de dispositivos como relés, lo convierte en una solución versátil para entornos domésticos e industriales. Esto es especialmente útil para sistemas de climatización o deshumidificación, mejorando así el confort en espacios cerrados.
* **Identificación de Problemas o Limitaciones Encontradas Durante el Proceso:**  
  Durante las pruebas iniciales, se identificó que las interferencias electromagnéticas provenientes de otros dispositivos podían afectar la comunicación Bluetooth. Este problema se solucionó al reubicar el ESP32 y asegurarse de que no hubiera obstrucciones físicas entre el microcontrolador y el dispositivo móvil. Además, se observó que el sensor DHT11 tiene limitaciones en su rango operativo; por lo tanto, se recomienda considerar sensores más avanzados si se requieren mediciones fuera del rango especificado.

**Conclusiones:**

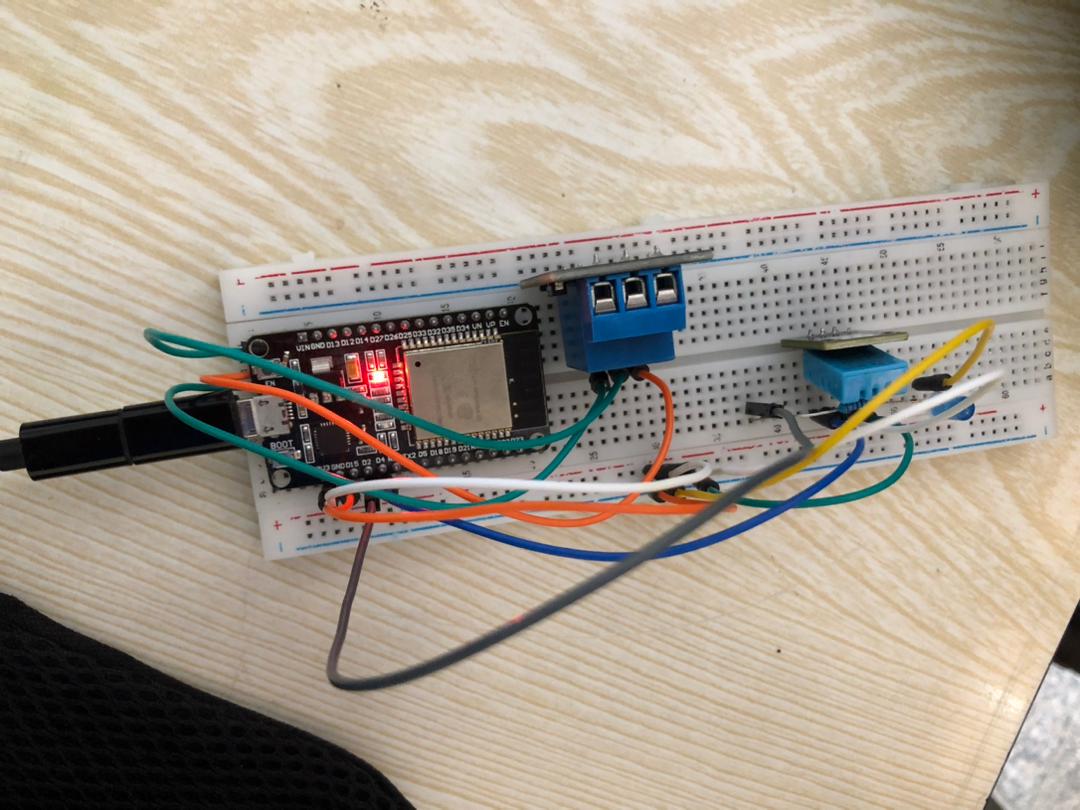
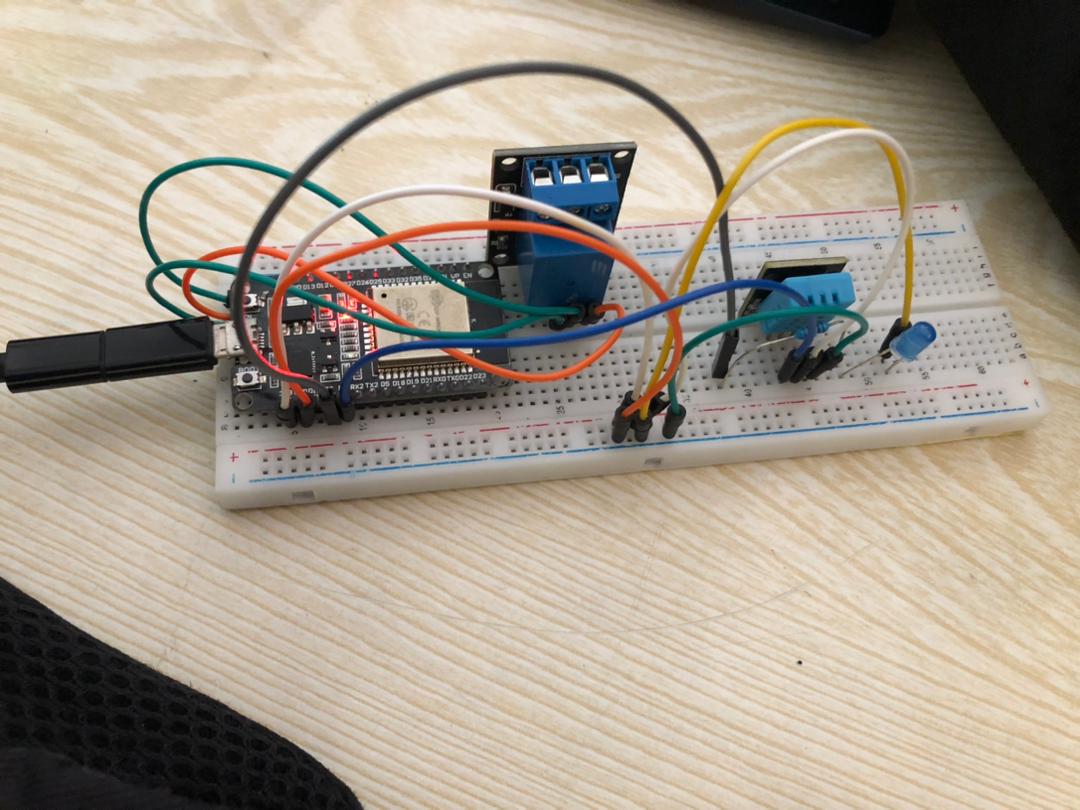
El sistema desarrollado para el control y monitoreo de temperatura y humedad utilizando el sensor DHT11, el microcontrolador ESP32, y actuadores como relés y LEDs presenta varias ventajas significativas. En primer lugar, la capacidad de obtener datos en tiempo real sobre las condiciones ambientales permite una respuesta rápida ante cualquier anomalía, lo que mejora la eficiencia en la gestión del entorno. Además, la automatización del control de dispositivos a través de la aplicación móvil no solo facilita la interacción del usuario, sino que también optimiza el consumo energético al activar o desactivar los actuadores según las lecturas del sensor. Otro aspecto relevante es la accesibilidad económica del sistema, ya que tanto el ESP32 como el DHT11 son componentes de bajo costo que permiten a los usuarios implementar soluciones de monitoreo ambiental sin necesidad de realizar grandes inversiones. Esto democratiza el acceso a tecnologías avanzadas, fomentando su adopción en entornos domésticos y educativos.

**Recomendaciones**

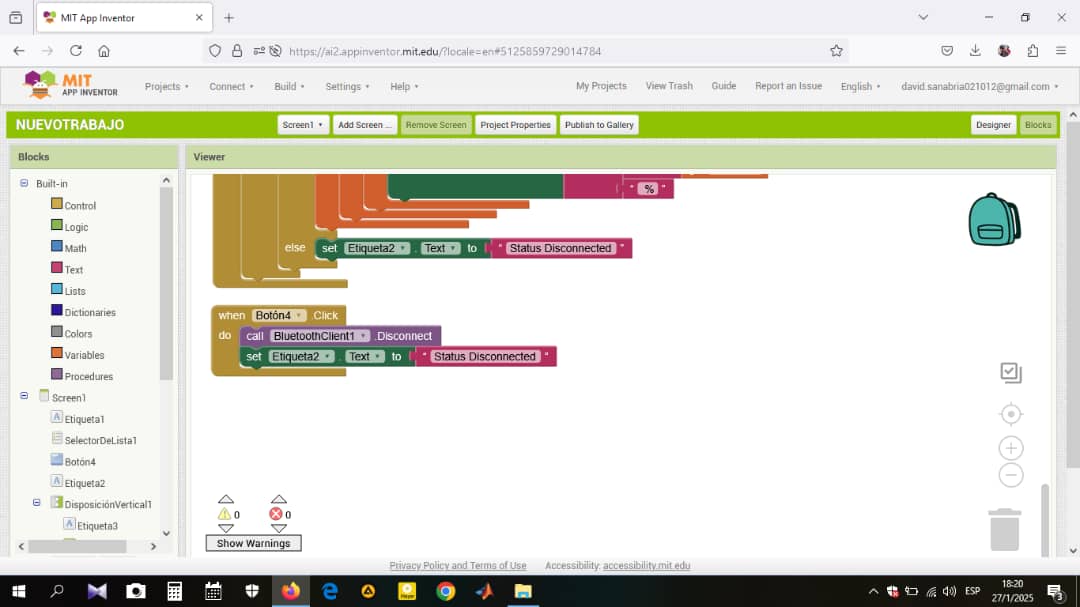
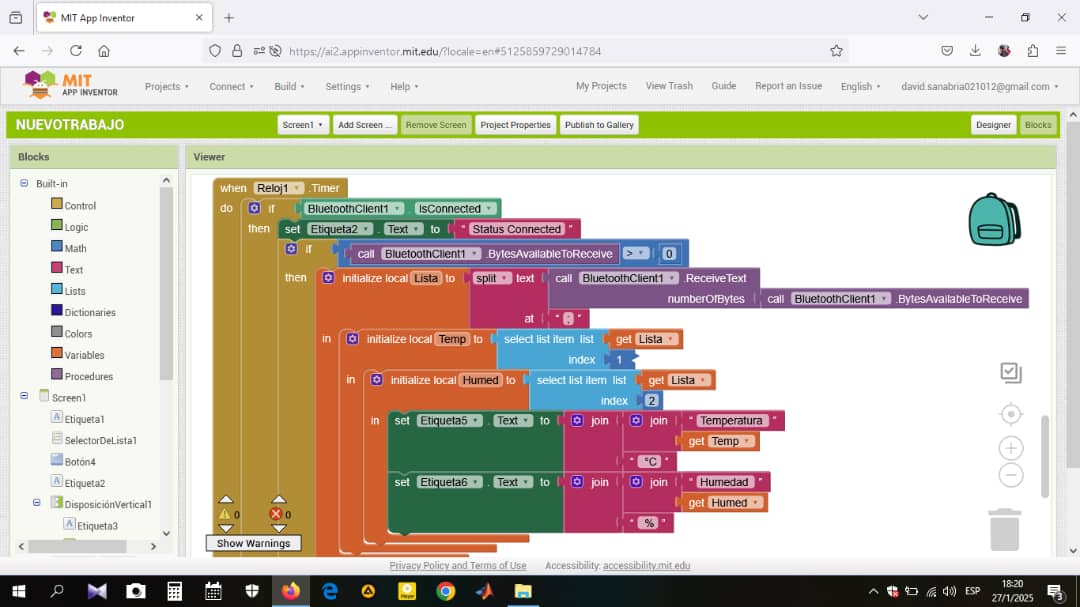
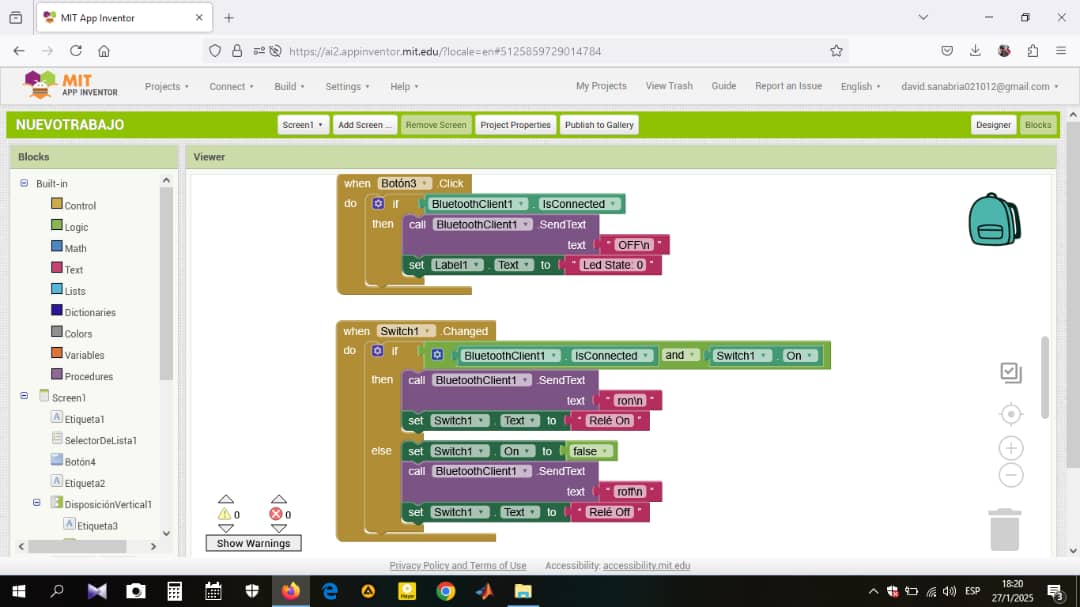
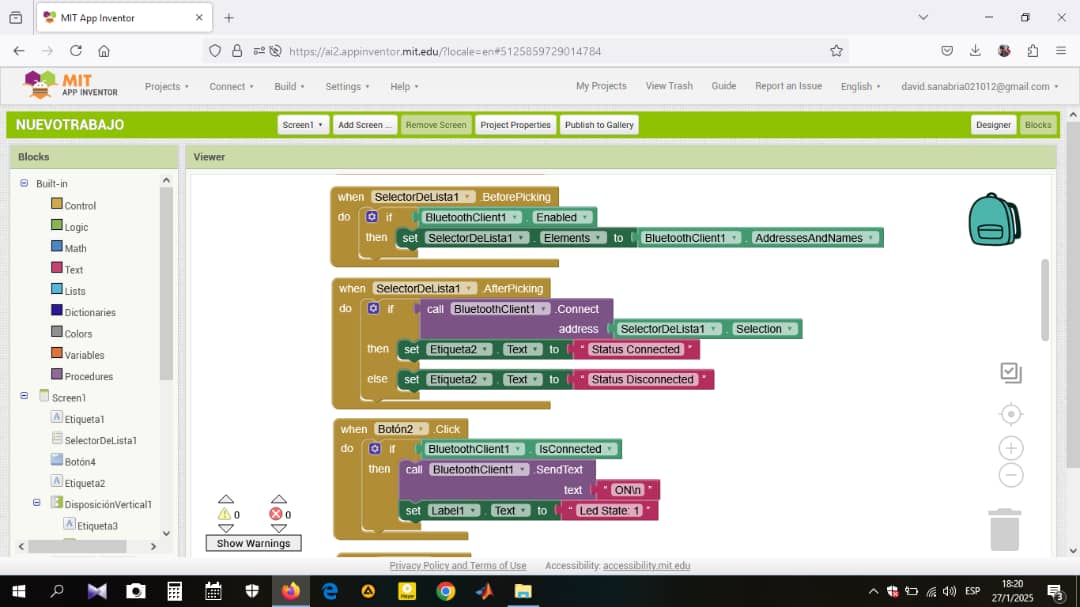
Se recomienda implementar un sistema de registro que almacene las lecturas de temperatura y humedad a lo largo del tiempo, lo que permitirá a los usuarios analizar tendencias y realizar ajustes proactivos en sus entornos. Además, sería beneficioso incluir notificaciones automáticas en la aplicación móvil que alerten a los usuarios cuando las condiciones ambientales superen umbrales predefinidos, como niveles de humedad excesiva o temperaturas extremas.

**Anexos:**

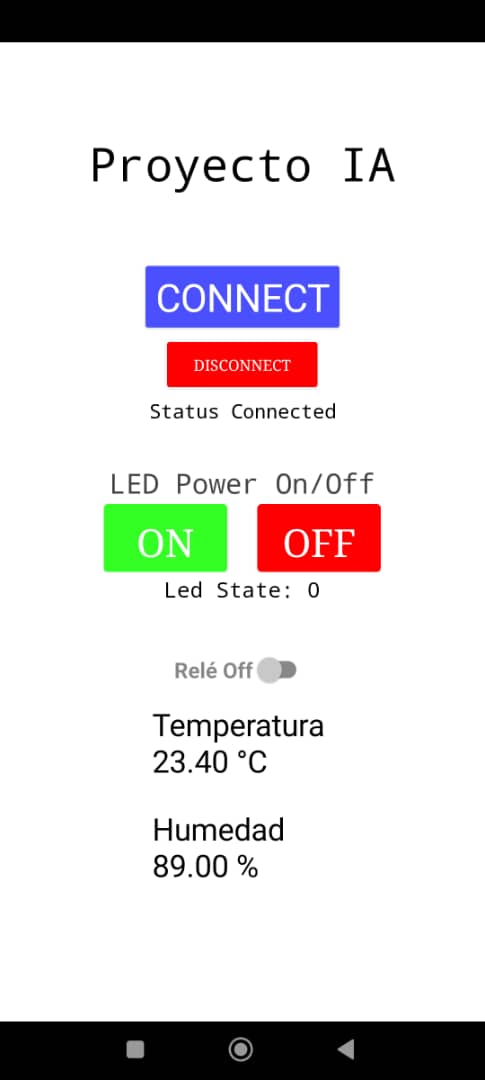
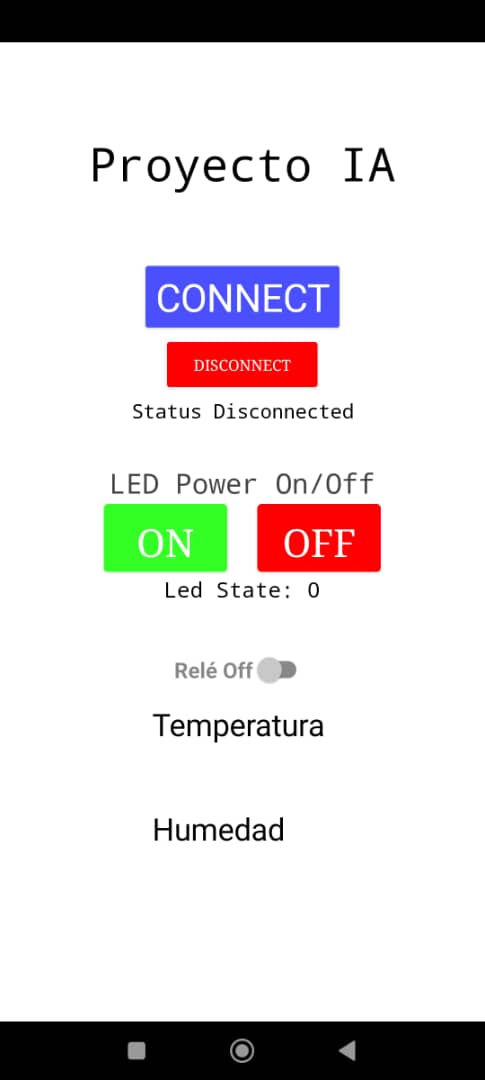
Conexiones del ESP32, el sensor y sus componentes



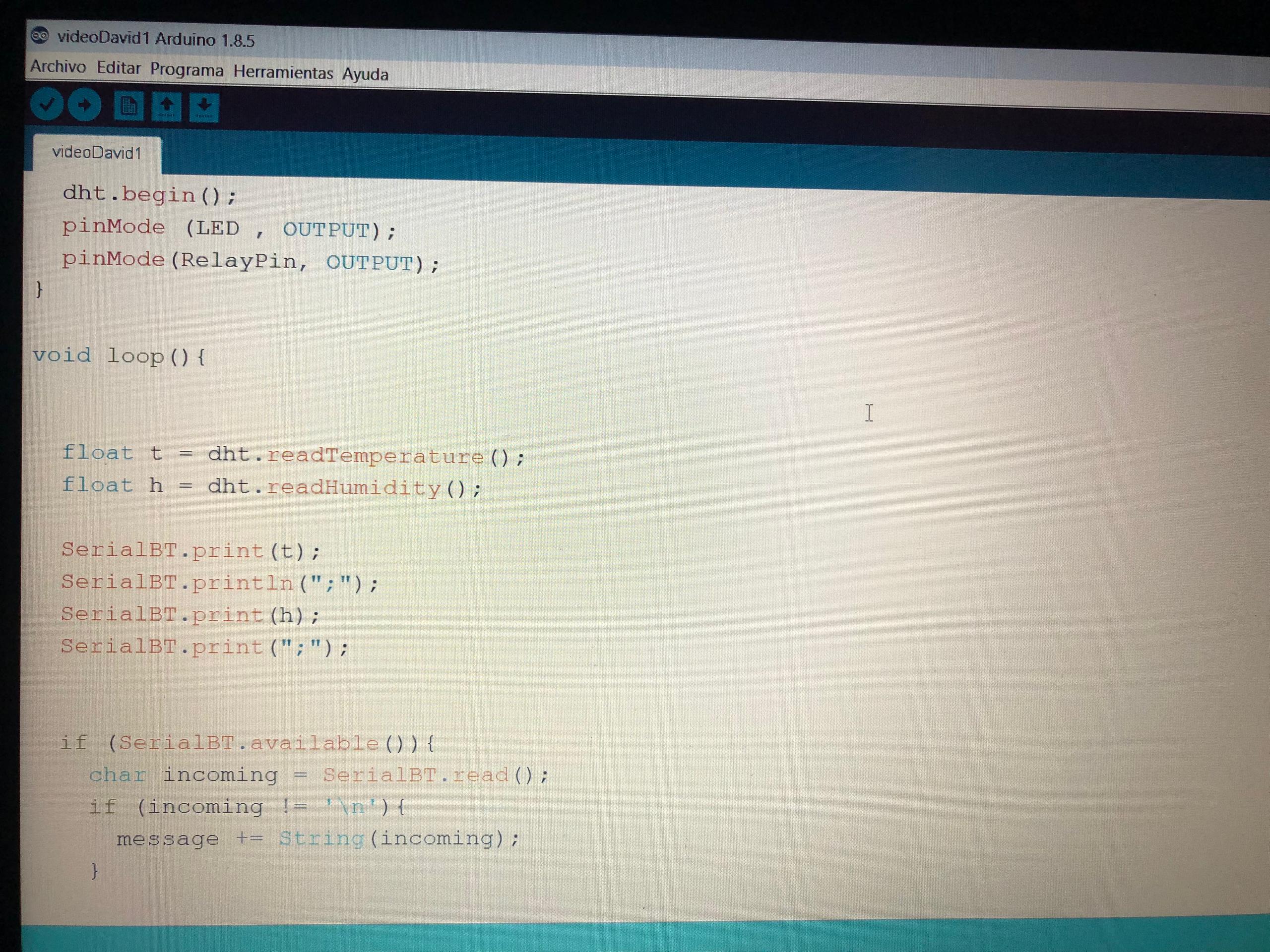
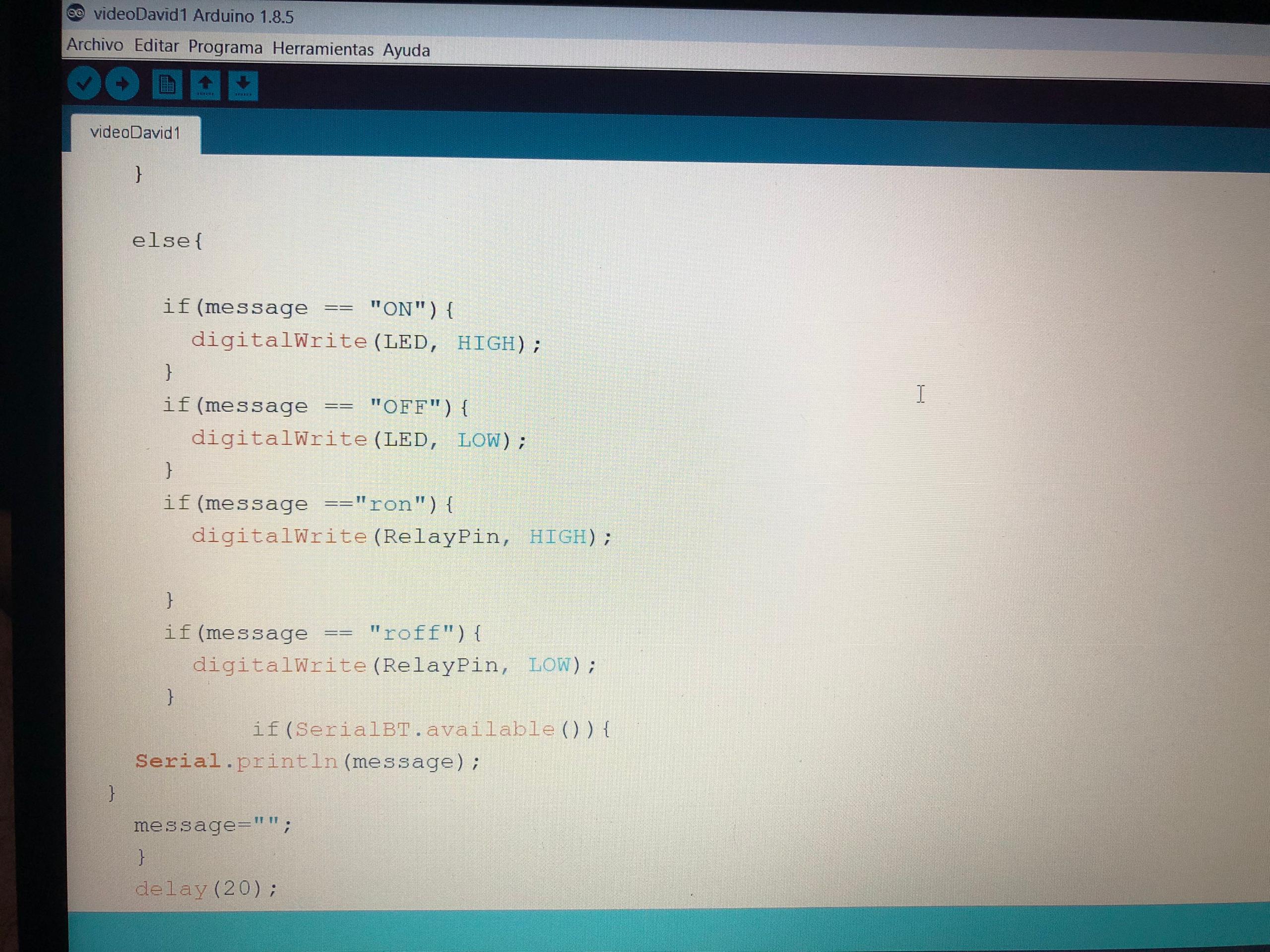
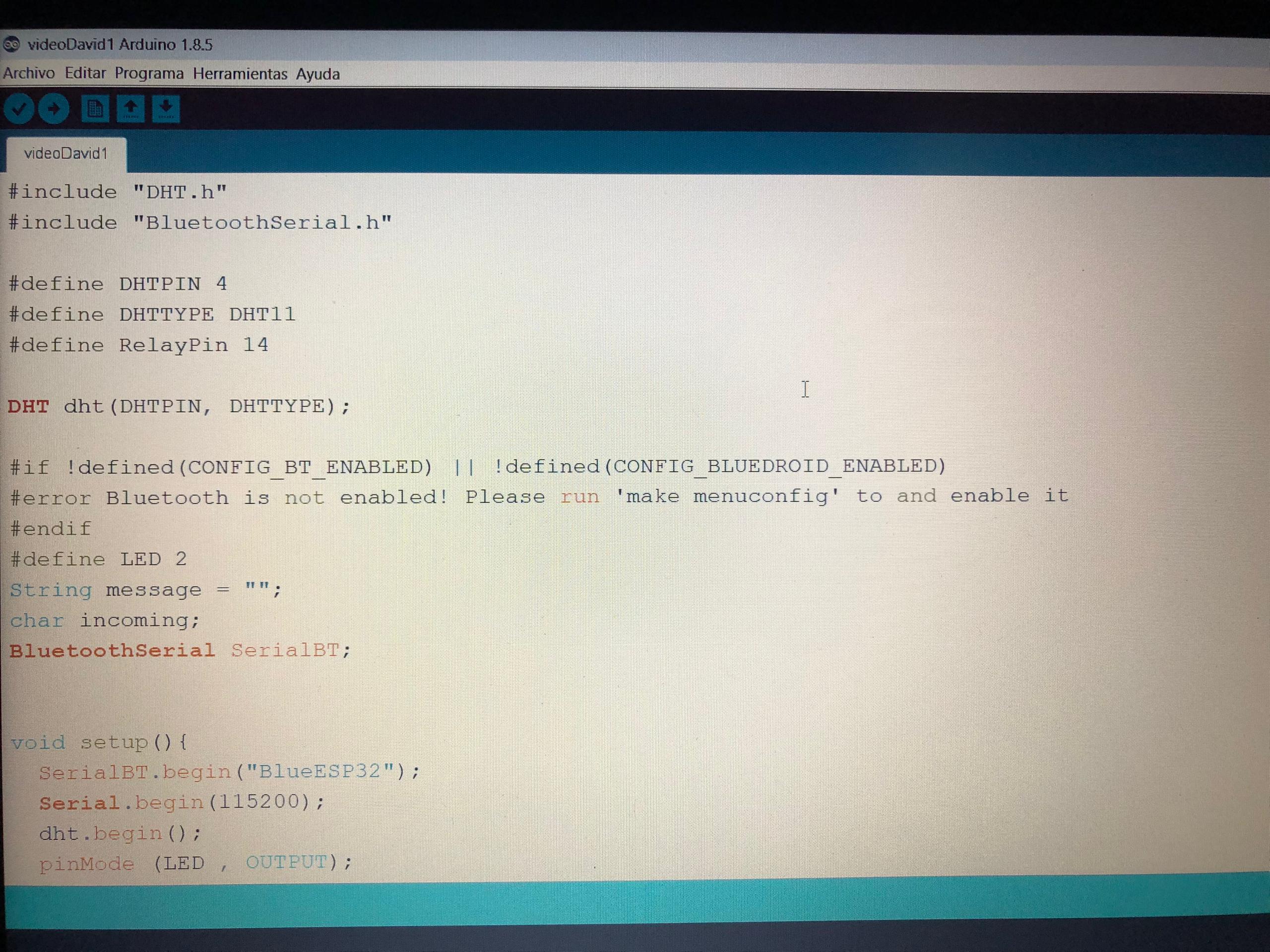
Creación de la aplicación



Interfaz terminada de la aplicacion



Código en el ID arduino:



Componentes utilizados para la creación del trabajo (ESp32,protoboard, resistencia,sensor DTH11 ,LED y un rele)a

