UNIÓN DE ASOCIACIONES DE INGENIEROS TÉCNICOS INDUSTRIALES Y GRADUADOS EN LA INGENIERÍA DE LA RAMA INDUSTRIAL DE ESPAÑA

UNIÓN DE ASOCIACIONES DE INGENIEROS TÉCNICOS INDUSTRIALES Y GRADUADOS EN INGENIERÍA DE LA RAMA INDUSTRIAL DE ESPAÑA (UAITIE)

“CONVOCATORIA 2026”

XI PREMIO NACIONAL DE INICIACIÓN A LA INVESTIGACIÓN TECNOLÓGICA

Título del Trabajo: A.R.D.U. Casa

AUTOR/ES:
Galván García, Joaquín Norberto;Martínez-Cabrera Gutiérrez, Fernando
Rodríguez-Mourullo Montalban, Álvaro;
Serra Silvano, Nicolás

BLOQUE TEMÁTICO:
Urbanismo Inteligente

NIVEL EDUCATIVO:
4º ESO

COORDINADOR:
Luis Lago Espejo-Saavedra

Marzo 2026

Resumen

Nuestro proyecto “Casa A.R.D.U” es un sistema domótico construido con Arduino Uno R4 WiFi, diseñado para que la casa sea más segura, cómoda y automática. La idea principal es que la casa pueda cuidar ciertas cosas por sí misma, especialmente para personas que no siempre pueden estar pendientes de todo o que necesitan un poco de ayuda extra. La inspiración vino de problemas reales que todos podemos tener en casa: por ejemplo, que se quede una ventana abierta con lluvia o humedad, que alguien olvide cerrar la puerta, o que haya riesgo de incendios y nadie esté cerca para reaccionar. Queríamos un sistema que pudiera reaccionar automáticamente ante estas situaciones y que además fuera fácil de entender y de usar. Para lograr esto, el proyecto integra varios sensores y actuadores:

- Un sensor de humedad y temperatura (DHT11), que nos permite saber cuándo el ambiente está demasiado húmedo o frío.
- Un sensor ultrasónico, que detecta si alguien se acerca a la puerta, para abrirla o cerrarla automáticamente.
- Un sensor de llama, que detecta fuego o peligro de incendio.
- Actuadores como servos para abrir y cerrar ventanas y puertas, un buzzer como alarma sonora, una bomba de agua controlada por relé para apagar incendios y una pantalla LCD que muestra información en tiempo real sobre el estado de la casa.

Palabras Clave

Arduino
Domótica
Sensores
Programación

Colegio San Patricio Madrid

Casa A.R.D.U

Grupo 1 Sanpa
4º de la ESO
Colegio San Patricio el Soto
2026
Concurso UAITIE

Problema a resolver:

- Personas mayores en casa que les cuesta cuidarse solos y gestionar el hogar.
- Incendios que se propagan porque los bomberos tardan un poco en llegar.
- Casas que se inundan porque se dejan las ventanas abiertas.
- Incomodidad en tu propia casa.
- Dejarse la puerta de casa abierta.

Solución propuesta:

- Una casa domótica que se autorregula dependiendo de los valores de t°, humedad, movimiento y fuego. Si hay mucha humedad (riesgo de lluvia) se cierran las ventanas para evitar una inundación. Si la puerta detecta movimiento se abre, y si no se cierra. Además va a incluir un sistema de detección de fuego que activará el sistema anti incendios (tubos por el techo que liberan agua para apagar el fuego).

Necesidad que cubre:

Aunque ya hemos hablado antes del problema a resolver tenemos que analizar si es una verdadera necesidad que tiene el mundo o no. De los 5 puntos que he puesto anteriormente creo que hay cuatro que son los más importantes y uno que es más confort que otra cosa, por eso los iré explicando en orden de menor importancia o necesidad a más.

- El menos importante, o el menos necesitado es el de *"Incomodidad en tu propia casa."* Este es el menos necesario porque, al fin y al cabo, no tener comodidad en tu casa no va a afectar a la supervivencia ni al desarrollo humano básico, no es algo imprescindible ni universal, ni constituye una prioridad frente a necesidades básicas como la alimentación, la salud o la seguridad.

- El siguiente menos importante es el de *"Casas que se inundan porque se dejan las ventanas abiertas."* Esto también es poco necesario porque al fin y al cabo cerrar una ventana en días de lluvia es una cosa que ya hacemos de por sí por, tanto por el frío como por la lluvia. Sin embargo, para ocasiones en las que no estés en casa puede venir bien.

- El que voy a comentar ahora ya empieza a ser más necesario que los otros dos porque este sí que afecta a la seguridad de tu casa. Este es *"Dejarse la puerta de casa abierta."* Esto es una de las grandes preocupaciones de una madre de familia numerosa, que al ir con

todos los hijos y eso siempre tiene el miedo de haberse dejado la puerta de casa abierta. Esta solución lo que propone es que cuando detecte que has salido por la puerta la cierra y la bloquea hasta que la vuelves a abrir.

- Los siguientes dos que voy a mencionar creo que son igual de importantes, pero primero voy a decir uno y luego el otro. Del que voy a hablar ahora es del que dice *"Personas mayores en casa que les cuesta cuidarse solos y gestionar el hogar."* Esta es una necesidad muy importante porque las personas que se encuentran en casa solas normalmente tienen problemas de movimiento y tener una casa que se autogestione les soluciona el tener que estar ajustando ellos. Además así se pueden sentir más seguros.

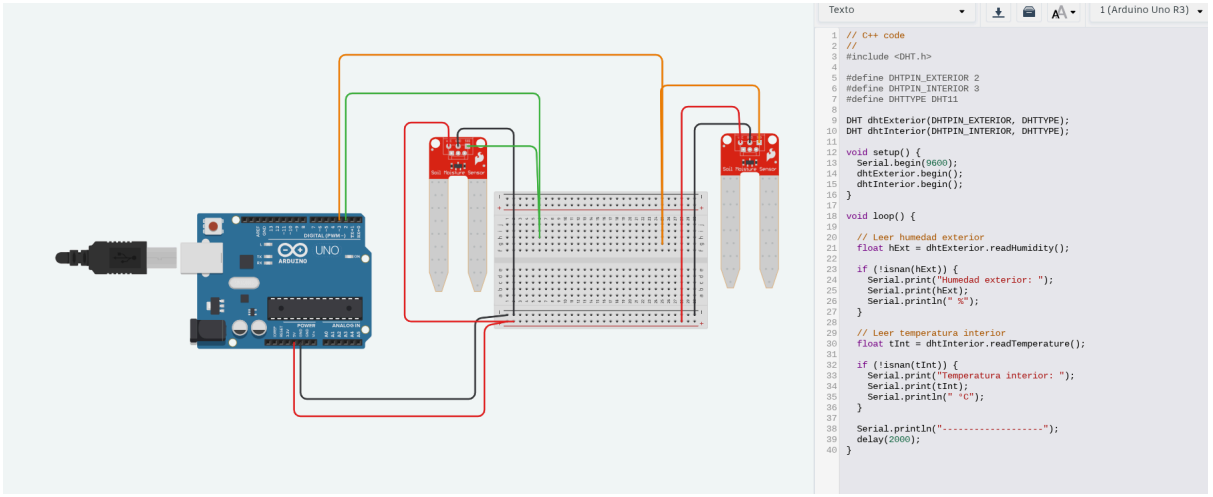
- El último problema que voy a mencionar es *"Incendios que se propagan porque los bomberos tardan un poco en llegar."* Este es de vital importancia porque muchas veces, aunque el servicio de bomberos sea muy rápido, tardan un poco en llegar lo que puede hacer que para cuando ya hayan llegado ya haya muerto alguien o que se haya quemado todo. Por eso un sistema de detección de fuego y de apagado es una solución a esta necesidad muy efectiva.

Diseño técnico del sistema: →placa de arduino uno R4 wifi.

- DTH11 exterior: pin 2 → pin de entrada
- TRIG PIN (ultrasónico): pin 4 → pin de salida
- ECHO PIN (ultrasónico): pin 5 → pin de entrada
- FLAME PIN: pin A1 → pin de entrada
- BUZZER PIN: pin 11 → pin de salida
- SERVO VENTANA 2: pin 7 → pin de salida
- SERVO PUERTA: pin 8 → pin de salida
- RELÉ BOMBA: pin 10 → pin de salida

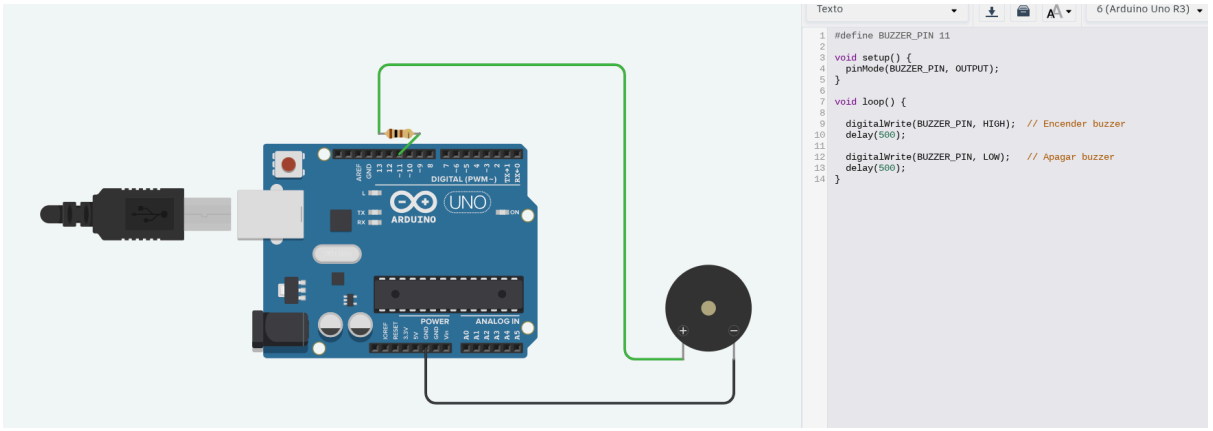
Diagramas del circuito por sensores:

- DTH11s: Al no haber en tinkercad el DTH11 he utilizado sensores de humedad de lluvia.



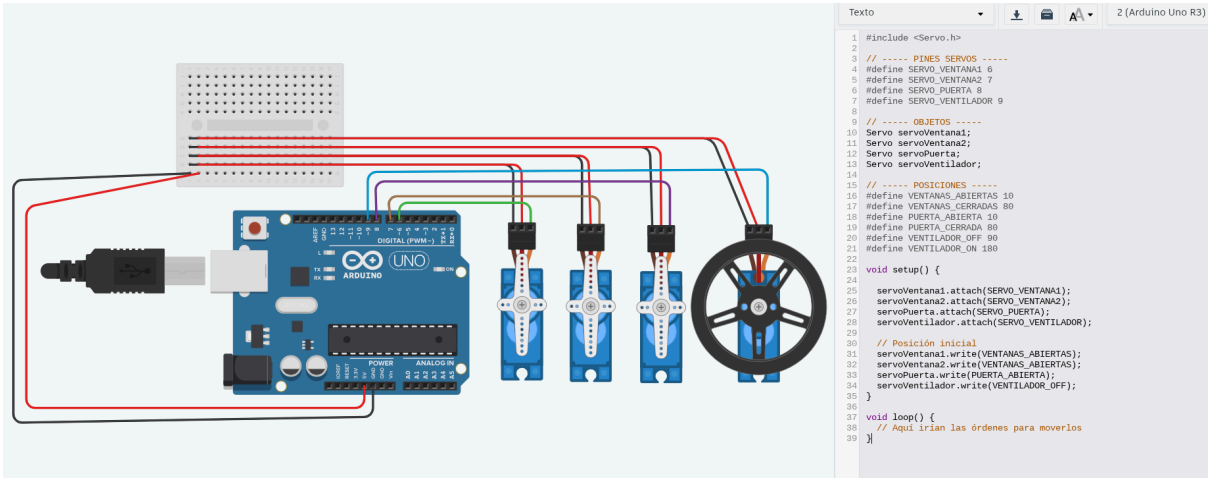
```
// C++ code
//
#include <DHT.h>

#define DHTPIN_EXTERIOR 2
#define DHTPIN_INTERIOR 3
#define DHTTYPE DHT11

DHT dhtExterior(DHTPIN_EXTERIOR, DHTTYPE);
DHT dhtInterior(DHTPIN_INTERIOR, DHTTYPE);

void setup() {
  Serial.begin(9600);
  dhtExterior.begin();
  dhtInterior.begin();
}

void loop() {

  // Leer humedad exterior
  float hExt = dhtExterior.readHumidity();

  if (!isnan(hExt)) {
    Serial.print("Humedad exterior: ");
    Serial.print(hExt);
    Serial.println(" %");
  }

  // Leer temperatura interior
  float tInt = dhtInterior.readTemperature();

  if (!isnan(tInt)) {
    Serial.print("Temperatura interior: ");
    Serial.print(tInt);
    Serial.println(" °C");
  }

  Serial.println("------------------");
  delay(2000);
}
```

- El buzzer:



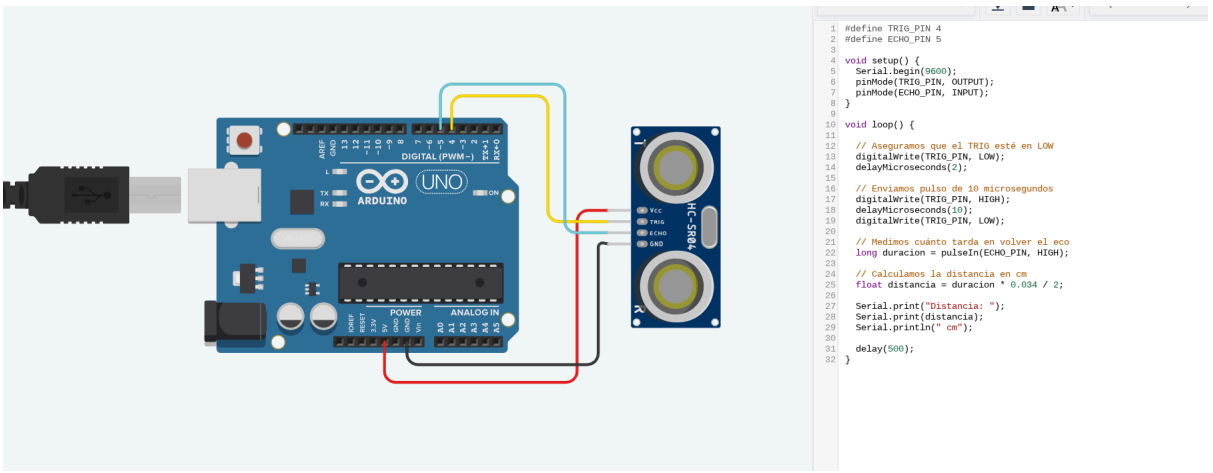
```
#define BUZZER_PIN 11

void setup() {
  pinMode(BUZZER_PIN, OUTPUT);
}

void loop() {

  digitalWrite(BUZZER_PIN, HIGH);  // Encender buzzer
  delay(500);

  digitalWrite(BUZZER_PIN, LOW);   // Apagar buzzer
  delay(500);
}
```

- Los servos: el que es como una rueda es de rotación continua



```
#include <Servo.h>

// ----- PINES SERVOS -----
#define SERVO_VENTANA1 6
#define SERVO_VENTANA2 7
#define SERVO_PUERTA 8
#define SERVO_VENTILADOR 9

// ----- OBJETOS -----
Servo servoVentana1;
Servo servoVentana2;
Servo servoPuerta;
Servo servoVentilador;

// ----- POSICIONES -----
#define VENTANAS_ABIERTAS 10
#define VENTANAS_CERRADAS 80
#define PUERTA_ABIERTA 10
#define PUERTA_CERRADA 80
#define VENTILADOR_OFF 90
#define VENTILADOR_ON 180

void setup() {

  servoVentana1.attach(SERVO_VENTANA1);
  servoVentana2.attach(SERVO_VENTANA2);
  servoPuerta.attach(SERVO_PUERTA);
  servoVentilador.attach(SERVO_VENTILADOR);

  // Posición inicial
  servoVentana1.write(VENTANAS_ABIERTAS);
  servoVentana2.write(VENTANAS_ABIERTAS);
  servoPuerta.write(PUERTA_ABIERTA);
  servoVentilador.write(VENTILADOR_OFF);
}

void loop() {
  // Aquí irían las órdenes para moverlos
}
```

- Sensor ultrasónico:




```
#define TRIG_PIN 4
#define ECHO_PIN 5

void setup() {
  Serial.begin(9600);
  pinMode(TRIG_PIN, OUTPUT);
  pinMode(ECHO_PIN, INPUT);
}

void loop() {

  // Aseguramos que el TRIG esté en LOW
  digitalWrite(TRIG_PIN, LOW);
  delayMicroseconds(2);

  // Enviamos pulso de 10 microsegundos
  digitalWrite(TRIG_PIN, HIGH);
  delayMicroseconds(10);
  digitalWrite(TRIG_PIN, LOW);

  // Medimos cuánto tarda en volver el eco
  long duracion = pulseIn(ECHO_PIN, HIGH);

  // Calculamos la distancia en cm
  float distancia = duracion * 0.034 / 2;

  Serial.print("Distancia: ");
  Serial.print(distancia);
  Serial.println(" cm");

  delay(500);
}
```

El resto de sensores que faltan es porque en tinkercad no están.

El código:

- 1er código: Este código solo era para el primer sensor que teníamos pensado programar, el DTH11. Para este primero teníamos que incluir su librería (#include “DHT.h”). También definimos el pin y el tipo de sensor (#define). Los dos begin siguientes sirven para iniciar la comunicación con el PC y para activar el DTH11. Donde pone los dos float es que le estamos diciendo que lea en bucle los valores de humedad y t°. El if lo que está diciendo es que si la lectura falla imprima “Error leyendo el DTH11”. Los serial print están diciéndo a arduino que enseñe en el serial monitor los valores de t° y humedad cada 2000 milisegundos o 2 segundos.

```cpp
#include "DHT.h"

#define DHTPIN 2
#define DHTTYPE DHT11

DHT dht(DHTPIN, DHTTYPE);

void setup() {
  Serial.begin(9600);
  dht.begin();
}

void loop() {
  float humedad = dht.readHumidity();
  float temperatura = dht.readTemperature();

  if (isnan(humedad) || isnan(temperatura)) {
    Serial.println("Error leyendo el DHT11");
    return;
  }

  Serial.print("Humedad: ");
  Serial.print(humedad);
  Serial.print("%  Temperatura: ");
  Serial.print(temperatura);
  Serial.println("°C");

  delay(2000);
}
```

- 2nd código: En este segundo código incluimos el sensor de sonido y el sensor pir. Además aprovechando que la placa arduino tiene bluetooth le hemos añadido el bluetooth ble. Ahora voy a explicar el código por partes.

```cpp
#include <DHT.h>
#include <ArduinoBLE.h>
```

Aquí le decimos a arduino que vamos a usar las bibliotecas de el dth11 y de el bluetooth ble.

```cpp
#define DHTPIN 2
#define DHTTYPE DHT11
#define PIR_PIN 3
#define SOUND_PIN A0
```

Aquí definimos los pines receptores o de entrada.

```
#define MOTOR_VENTANAS 4
#define MOTOR_PUERTA 5
#define VENTILADOR 6
#define ALARMA 7
```

En esta parte del código definimos los pines actuadores o de salida.

```
#define UMBRAL_HUMEDAD 80
#define UMBRAL_TEMP 30
#define UMBRAL_SONIDO 500
```

Aquí definimos los umbrales que no se pueden traspasar o si no se activan los motores.

```
BLEService casaService("180A");
BLEFloatCharacteristic tempChar("2A6E", BLERead | BLENotify);
BLEFloatCharacteristic humChar("2A6F", BLERead | BLENotify);
BLEByteCharacteristic movChar("2A56", BLERead | BLENotify);
BLEByteCharacteristic ventiladorChar("2A57", BLERead | BLEWrite);
BLEByteCharacteristic alarmaChar("2A58", BLERead | BLEWrite);
```

En este fragmento del código le decimos a arduino que cree un servicio de bluetooth BLE llamado casaService. También se crean algunas características para que a la hora de enviar información de los sensores a un móvil.

```
Serial.begin(9600);
dht.begin();
pinMode(...);
```

Esta ya la expliqué antes, pero quiere decir que es el momento de iniciar la comunicación con el PC y que se inicien los sensores.

```
if (!BLE.begin()) { ... }
BLE.setLocalName("Casa_Inteligente_Nico");
BLE.setAdvertisedService(casaService);
...
BLE.advertise();
```

Con esta parte del código arduino entiende que tiene que iniciar BLE habilitado para que un dispositivo se conecte. Además también define el nombre “Casa_Inteligente_Nico”.

```
BLEDevice central = BLE.central();
if (central) { ... }
```

Aqui lo que dice es que tiene que esperar a que un móvil se conecte. Mientras cada 2 segundos hace lo siguiente.

```
float h = dht.readHumidity();
float t = dht.readTemperature();
if (!isnan(h) && !isnan(t)) { ... }
```

Lee humedad y t° en bucle y actualiza los datos de BLE para que un móvil los vea. También activa los motores según los umbrales que habíamos definido antes.

```
int movimiento = digitalRead(PIR_PIN);
movChar.writeValue(movimiento);
digitalWrite(MOTOR_PUERTA, movimiento ? HIGH : LOW);
```

Detecta movimiento y abre la puerta.

```
int sonido = analogRead(SOUND_PIN);
digitalWrite(ALARMA, sonido >= UMBRAL_SONIDO ? HIGH : LOW);
```

Activa una alarma si el umbral que definimos antes se ve traspasado.

```
if (ventiladorChar.written()) { ... }
if (alarmaChar.written()) { ... }
```

Con estas dos líneas se permite activar el ventilador y la alarma a través de la aplicación de móvil y de BLE.

- 3er código: Este código es igual al anterior, lo único que cambia es que los motores son servos que necesitan una programación especial. Además en este 3er código incluimos una fotoresistencia. Al ser tan parecido al anterior solo voy a explicar las partes nuevas.

```
#include <DHT.h>
#include <ArduinoBLE.h>
#include <Servo.h>
```

En la parte de librerías incluimos la de los servos.

```
#define DHTPIN 2
#define PIR_PIN 3
#define SOUND_PIN A0
#define LDR_PIN A1
```

En la parte de los pines y los sensores incluimos el LDR PIN que es la fotoresistencia.

```
#define SERVO_VENTANAS 4
#define SERVO_PUERTA 5
#define SERVO_VENTILADOR 6
#define ALARMA 7
```

También definimos nuestros nuevos servos.

```
DHT dht(DHTPIN, DHTTYPE);
Servo servoVentanas;
Servo servoPuerta;
Servo servoVentilador;
```

Aquí creamos los objetos para manejar el DTH11 y los servos.

```
#define VENTANAS_ABIERTAS 10
#define VENTANAS_CERRADAS 80
#define PUERTA_ABIERTA 10
#define PUERTA_CERRADA 80
#define VENTILADOR_OFF 10
#define VENTILADOR_ON 160
```

Aquí definimos los ángulos de los servos según la acción deseada.

```
#define UMBRAL_HUMEDAD 80
#define UMBRAL_TEMP 30
#define UMBRAL_SONIDO 800
#define UMBRAL_LUZ 700
```

Definimos el umbral de luz justo al resto.

```
int contadorRuido = 0;
#define VECES_RUIDO 3
```

Para evitar falsas alarmas por ruido hemos metido que se tiene que superar el umbral 3 veces seguidas para activar la alarma.

```
servoVentanas.attach(SERVO_VENTANAS);
servoPuerta.attach(SERVO_PUERTA);
servoVentilador.attach(SERVO_VENTILADOR);

servoVentanas.write(VENTANAS_ABIERTAS);
servoPuerta.write(PUERTA_ABIERTA);
servoVentilador.write(VENTILADOR_OFF);
```

Aquí lo que le decimos a arduino es que inicie los servos en su posición segura.

```
float h = dht.readHumidity();
float t = dht.readTemperature();

if (h >= UMBRAL_HUMEDAD) servoVentanas.write(VENTANAS_CERRADAS);
else servoVentanas.write(VENTANAS_ABIERTAS);

if (t >= UMBRAL_TEMP) servoVentilador.write(VENTILADOR_ON);
else servoVentilador.write(VENTILADOR_OFF);
```

Aquí se ajustan las ventanas y el ventilador según los umbrales. También envía los valores via BLE y los imprime en el serial monitor.

```
int movimiento = digitalRead(PIR_PIN);
movChar.writeValue(movimiento);
servoPuerta.write(movimiento ? PUERTA_CERRADA : PUERTA_ABIERTA);
```

Abrir o cerrar puerta dependiendo del movimiento.

```
int luz = analogRead(LDR_PIN);
lightChar.writeValue(luz);
```

Envía los valores de luz al móvil a través de BLE.

- 4to y último código hasta el momento: en este código lo único que cambia es que en vez del pir utilizamos un sensor ultrasónico e incluimos un sensor de llama que detecta fuego, activa el relé y este

activa una bomba de agua que apagará el fuego. Además hemos eliminado el sensor de sonido que no servía para mucho. Solo pondré los cambios.

```
#define DHTPIN_EXTERIOR 2
#define DHTPIN_INTERIOR 3
#define TRIG_PIN 4
#define ECHO_PIN 5
#define LDR_PIN A0
#define FLAME_PIN A1
#define BUZZER_PIN 11
```

TRIG y ECHO son los pines del sensor ultrasónico. Flame y buzzer son los nuevos.

```
#define SERVO_VENTANA1 6
#define SERVO_VENTANA2 7
#define SERVO_PUERTA 8
#define SERVO_VENTILADOR 9
#define RELE_BOMBA 10
```

Añadimos el relé.

```
#define UMBRAL_HUMEDAD 80
#define TEMP_MAX 25
#define TEMP_MIN 15
#define UMBRAL_FUEGO 500
bool ventanasCerradas = false;
bool incendioActivo = false;
```

Ajustamos los umbrales.

```
servoVentana1.write(VENTANAS_ABIERTAS);
servoVentana2.write(VENTANAS_ABIERTAS);
servoPuerta.write(PUERTA_ABIERTA);
servoVentilador.write(VENTILADOR_OFF);
digitalWrite(RELE_BOMBA, LOW);
digitalWrite(BUZZER_PIN, LOW);
```

Iniciamos todo.

```
digitalWrite(TRIG_PIN, HIGH);
long duracion = pulseIn(ECHO_PIN, HIGH, 30000);
int distancia = duracion * 0.034 / 2;
if (distancia > 0 && distancia < 20) servoPuerta.write(PUERTA_CERRADA);
else servoPuerta.write(PUERTA_ABIERTA);
```

Aquí codificamos el sensor ultrasónico.

```
int fuego = analogRead(FLAME_PIN);
if (fuego < UMBRAL_FUEGO && !incendioActivo) {
  // Buzzer intermitente
  for (int i = 0; i < 10; i++) {
    digitalWrite(BUZZER_PIN, HIGH); delay(200);
    digitalWrite(BUZZER_PIN, LOW); delay(200);
  }

  digitalWrite(RELE_BOMBA, HIGH);
  delay(4000);
  digitalWrite(RELE_BOMBA, LOW);
  incendioActivo = true;
}
```

Aquí codificamos el flame sensor y como este activará el relé y la bomba.

```
if (fuego > UMBRAL_FUEGO + 100) incendioActivo = false;
```

Aquí le decimos que reinicie la detección para poder volver a activarse.

El siguiente código es el código final. En este hemos mejorado el sistema anti incendios y añadido un lcd. Hemos calibrado el sensor ultrasónico, los servos y los dth11. Hemos retirado el sensor de luz, una de las ventanas y el ventilador. También hemos retirado uno de los dth11 porque lo veíamos conveniente para la maqueta.

```
#include <DHT.h>
#include <ArduinoBLE.h>
#include <Servo.h>
#include <LiquidCrystal.h>

// ----- LCD -----
LiquidCrystal lcd(12, 13, A2, A3, A4, A5);

// ----- PINES SENSORES -----
```

```
#define DHTPIN_EXTERIOR 2
#define DHTTYPE DHT11

#define TRIG_PIN 4
#define ECHO_PIN 5

#define FLAME_PIN A1

#define BUZZER_PIN 11

// ----- PINES SERVOS -----
#define SERVO_VENTANA2 7
#define SERVO_PUERTA 8

// ----- PIN RELÉ -----
#define RELE_BOMBA 10

// ----- OBJETOS -----
DHT dhtExterior(DHTPIN_EXTERIOR, DHTTYPE);

Servo servoVentana2;
Servo servoPuerta;

// ----- POSICIONES -----
#define VENTANAS_ABIERTAS 10
#define VENTANAS_CERRADAS 80
#define PUERTA_ABIERTA 110
#define PUERTA_CERRADA 180

// ----- UMBRALES -----
#define UMBRAL_HUMEDAD 80
#define UMBRAL_FUEGO 500

// ----- ESTADO -----
bool ventanasCerradas = false;
bool incendioActivo = false;
bool fuegoDetectado = false;

// 🔥 CONTROL PUERTA INTELIGENTE
unsigned long ultimaDeteccion = 0;
const int tiempoEspera = 4000;
```

```
bool puertaAbierta = false;

// 🔊 CONTROL LCD Y BUZZER
unsigned long ultimaAlarma = 0;
unsigned long ultimaActualizacionLCD = 0;
const unsigned long intervaloLCD = 1000;

// ----- FUNCIONES -----
int medirDistancia() {
  long suma = 0;
  int validas = 0;

  for (int i = 0; i < 3; i++) {
    digitalWrite(TRIG_PIN, LOW);
    delayMicroseconds(2);

    digitalWrite(TRIG_PIN, HIGH);
    delayMicroseconds(10);
    digitalWrite(TRIG_PIN, LOW);

    long duracion = pulseIn(ECHO_PIN, HIGH, 20000);
    int d = duracion * 0.034 / 2;

    if (d > 2 && d < 200) {
      suma += d;
      validas++;
    }

    delay(10);
  }

  if (validas == 0) return 0;
  return suma / validas;
}

void setup() {
  Serial.begin(9600);

  dhtExterior.begin();

  pinMode(TRIG_PIN, OUTPUT);
```

```
  pinMode(ECHO_PIN, INPUT);
  pinMode(RELE_BOMBA, OUTPUT);
  pinMode(BUZZER_PIN, OUTPUT);

  servoVentana2.attach(SERVO_VENTANA2);
  servoPuerta.attach(SERVO_PUERTA);

  servoVentana2.write(VENTANAS_ABIERTAS);
  servoPuerta.write(PUERTA_CERRADA);

  digitalWrite(RELE_BOMBA, LOW);
  digitalWrite(BUZZER_PIN, LOW);

  BLE.begin();
  BLE.setLocalName("Casa_Nico");

  lcd.begin(16, 2);
  lcd.print("Casa Intelig.");
  lcd.setCursor(0,1);
  lcd.print("Iniciando...");
  delay(2000);
  lcd.clear();

  Serial.println("Sistema iniciado correctamente");
}

void loop() {
  // ----- HUMEDAD -----
  float hExt = dhtExterior.readHumidity();

  if (!isnan(hExt) && hExt >= UMBRAL_HUMEDAD && !ventanasCerradas) {
    servoVentana2.write(VENTANAS_CERRADAS);
    ventanasCerradas = true;
  }

  // ----- PUERTA INTELIGENTE -----
  int distancia = medirDistancia();
  if (distancia > 0 && distancia < 15) {
    servoPuerta.write(PUERTA_ABIERTA);
    ultimaDeteccion = millis();
    puertaAbierta = true;
```

```
}

if (puertaAbierta && (millis() - ultimaDeteccion > tiempoEspera)) {
  servoPuerta.write(PUERTA_CERRADA);
  puertaAbierta = false;
}

// ----- SENSOR DE LLAMA -----
int fuego = analogRead(FLAME_PIN);
fuegoDetectado = fuego < UMBRAL_FUEGO;

if (fuegoDetectado) {
  digitalWrite(RELE_BOMBA, HIGH);

  if (millis() - ultimaAlarma > 2000) {
    for (int i = 0; i < 5; i++) {
      tone(BUZZER_PIN, 2000);
      delay(100);
      noTone(BUZZER_PIN);
      delay(100);
    }
    ultimaAlarma = millis();
  }

  incendioActivo = true;
} else {
  digitalWrite(RELE_BOMBA, LOW);
  noTone(BUZZER_PIN);
  incendioActivo = false;
}

// ----- ACTUALIZACION LCD -----
if (millis() - ultimaActualizacionLCD > intervaloLCD) {
  lcd.clear();

  lcd.setCursor(0,0);
  lcd.print("H:");
  lcd.print(hExt);

  lcd.setCursor(0,1);
  lcd.print("D:");
```

```
    lcd.print(distancia);
    lcd.print(" F:");
    lcd.print(fuegoDetectado ? "SI" : "NO");

    ultimaActualizacionLCD = millis();
  }

  Serial.print("H:"); Serial.print(hExt);
  Serial.print(" D:"); Serial.print(distancia);
  Serial.print(" F:"); Serial.println(fuegoDetectado ? "SI" : "NO");

  delay(100);
}
```

Problemas encontrados: Al principio habíamos utilizado un sensor pir para detectar movimiento, pero ese nos dio demasiados problemas ya que detectaba movimiento cuando no lo había y no lo detectaba cuando había. Lo que hice fue usar el ultrasónico que detecta la distancia que hay entre él y un objeto y si es pequeña significa que tiene algo cerca, lo que fue una buena solución. También a la hora de buscar el flame sensor en mi cole no lo encontré, lo que encontre es un led negro que es un supuesto flame sensor, pero funcionaba muy mal. Esto atrasó un poco mi proyecto. Poco después buscando en unas cajas de sensores desorganizados tuve la suerte de encontrar un flame sensor bueno de verdad. Con el relé también tuve problemas porque encontré uno que yo creía que era un relé y cuando lo conecté me quedé estancado pensando que era un relé pero no funcionaba. Al final llegué a la conclusión de que eso no era un relé y busqué un relé de verdad. Lo del BLE lo puse porque estaba en proceso de crear una web a la que se conectara, pero eso fracasó porque no se conectaba bien. Cuando termine el proyecto lo volveré a intentar.
Por la parte física la casa la empezamos a construir creando una base de cemento que por problemas a la hora de dejarlo secando no funcionó por lo que empezamos a pegar los ladrillos con silicona. Los primeros pisos quedaron bien, pero los siguientes se fueron desviando un poco y eran débiles. La última complicación que nos hemos encontrado con esto es que hace poco se nos cayó el modelo y había que recontruirlp.

Fotos del proyecto y de los sensores: