Informe Técnico Sistema de Monitoreo de Sala de Computación

Introducción

Contexto del problema

Las salas de computación son espacios críticos en instituciones educativas y laborales, ya que concentran un gran número de equipos electrónicos sensibles, como computadoras, servidores, impresoras y dispositivos de red. Estos ambientes presentan diversos riesgos que pueden afectar tanto a los equipos como a las personas que los utilizan. Entre los más relevantes se encuentran:

• Calor excesivo, que puede dañar componentes internos y reducir la vida útil de los equipos.

• Niveles inadecuados de humedad, que favorecen la corrosión o problemas de estática.

• Posibles incendios, originados por sobrecalentamiento, fallas eléctricas o descuidos.

• Accesos indebidos fuera de horario, que comprometen la seguridad del espacio y su equipamiento.

La falta de un monitoreo constante en estos ambientes aumenta la probabilidad de fallas técnicas, pérdidas económicas y riesgos para la seguridad institucional.

Justificación del proyecto

Frente a esta problemática, se hace necesario contar con un sistema económico, confiable y de fácil implementación que permita monitorear de manera continua las condiciones ambientales y detectar situaciones de riesgo en la sala de computación. La elección de un sistema basado en Arduino Uno responde a su bajo costo, accesibilidad y facilidad de programación, lo que lo convierte en una herramienta ideal para proyectos académicos y prototipos funcionales.

El uso de sensores como el DHT11 (para temperatura y humedad), el LDR (para niveles de luz), el RTC (para control horario) y el sensor de fuego (para detección temprana de incendios), permite cubrir los principales factores de riesgo en este tipo de ambientes. Además, se incorporan una pantalla LCD para la visualización en tiempo real de los datos

y un módulo de almacenamiento SD para el registro histórico, facilitando un seguimiento detallado de las condiciones.

Como valor agregado, el sistema se complementa con un módulo Bluetooth y una aplicación desarrollada en MIT App Inventor, que posibilita a los usuarios consultar la información en sus teléfonos celulares, aumentando la capacidad de control y supervisión remota.

Breve explicación de la solución propuesta

El proyecto consiste en el diseño e implementación de un Sistema de Monitoreo y Seguridad para una Sala de Computación que funcione de manera automática, principalmente durante horarios nocturnos o de inactividad. El sistema deberá poder:

• Medir y registrar la temperatura y humedad del ambiente mediante el sensor DHT11, garantizando que los equipos informáticos se encuentren en condiciones adecuadas de operación.

• Monitorear la intensidad lumínica con el sensor LDR, detectando encendidos de luces o variaciones anormales en la iluminación durante horarios de inactividad.

• Gestionar la activación del sistema a través del módulo RTC, de manera que funcione principalmente durante la noche o en períodos en que la sala de cómputo no esté en uso.

• Detectar focos de incendio mediante el sensor de fuego, activando una alarma sonora y visual que permita una rápida respuesta ante emergencias.

• Mostrar la información en tiempo real en una pantalla LCD, facilitando la supervisión directa de las condiciones en la sala.

• Registrar datos históricos en una tarjeta SD, con el fin de llevar un control continuo y poder analizar variaciones ambientales a lo largo del tiempo.

• Transmitir los datos de manera inalámbrica utilizando un módulo Bluetooth, permitiendo que los usuarios puedan consultar la información desde una aplicación móvil desarrollada en MIT App Inventor.

• Validar el sistema en condiciones reales, realizando pruebas nocturnas y ajustando los parámetros de sensibilidad de los sensores y los umbrales de alarma según los resultados obtenidos.

En conjunto, esta solución busca proteger los equipos informáticos, optimizar el mantenimiento de la sala y mejorar la seguridad institucional mediante un sistema preventivo, accesible y adaptable a distintos entornos educativos o laborales.

Marco Teórico

Arduino Uno

El Arduino Uno es una placa de desarrollo basada en el microcontrolador ATmega328P. Se ha convertido en una de las plataformas más utilizadas en la enseñanza, prototipado rápido y proyectos de electrónica gracias a su bajo costo, accesibilidad y gran comunidad de usuarios.

Principales características:

• Microcontrolador: ATmega328P de 8 bits.

• Velocidad de reloj: 16 MHz.

• Memoria flash: 32 KB para almacenamiento de programas.

• SRAM: 2 KB.

• EEPROM: 1 KB para guardar datos permanentes.

• Entradas/Salidas: 14 pines digitales (6 PWM) y 6 entradas analógicas.

• Alimentación: 5V por USB o fuente externa de 7–12V.

Su funcionamiento se basa en leer datos de sensores (entradas), procesarlos mediante el programa cargado en la memoria y generar acciones de salida (activar alarmas, mostrar datos en una pantalla, registrar información, etc.). El entorno de desarrollo utilizado es Arduino IDE, que emplea un lenguaje basado en C/C++.

Sensor DHT11 (Sensor de Humedad y Temperatura)

El DHT11 es un sensor digital que combina en un solo módulo un termistor y un sensor capacitivo de humedad, junto con un circuito integrado que digitaliza la señal para facilitar su lectura con microcontroladores como Arduino.

• Medición de temperatura: el termistor varía su resistencia en función de la temperatura. El circuito convierte esa variación en un valor digital.

• Medición de humedad relativa: utiliza un condensador cuya capacitancia cambia según el contenido de agua en el aire. Esta variación se transforma en datos digitales.

• Comunicación: se realiza a través de un pin único (protocolo de un solo hilo), lo que simplifica las conexiones.

Especificaciones técnicas aproximadas:

• Rango de temperatura: 0 a 50 °C.

• Precisión: ±2 °C.

• Rango de humedad: 20 a 80 % HR.

• Precisión: ±5 % HR.

• Frecuencia de muestreo: 1 Hz (una lectura por segundo).

Su simplicidad lo hace ideal para proyectos educativos, aunque su precisión es menor que la del DHT22.

Sensor LDR (Light Dependent Resistor)

Un LDR (resistor dependiente de la luz) es un dispositivo electrónico pasivo cuya resistencia eléctrica disminuye cuando aumenta la intensidad de la luz incidente sobre su superficie.

Principio de funcionamiento:

• Fabricado con materiales semiconductores como el sulfuro de cadmio (CdS).

• Cuando recibe fotones de luz, los electrones del material ganan energía y pasan a la banda de conducción, reduciendo la resistencia eléctrica.

• En la oscuridad, la resistencia puede alcanzar valores de varios megaohmios. Con luz intensa, desciende a solo unos cientos de ohmios.

Aplicaciones:

• Sensores de encendido automático de luces.

• Sistemas de seguridad (detección de linternas o intrusiones).

• Medición de niveles de iluminación ambiental.

En Arduino, el LDR suele conectarse en un divisor de tensión con una resistencia fija, y se mide el voltaje resultante en un pin analógico.

Módulo RTC DS1302

El DS1302 es un módulo de reloj en tiempo real (RTC) diseñado para mantener la hora y la fecha de manera precisa incluso cuando el sistema principal está apagado. Incorpora un cristal oscilador de 32.768 kHz y permite la conexión de una batería de respaldo (CR2032) que le otorga autonomía durante meses o años.

Principio de Funcionamiento:

• Utiliza un oscilador de cuarzo para contar el tiempo con alta precisión.

• Divide la frecuencia del cristal para obtener segundos, minutos, horas, día, mes y año.

• Gracias a la batería de respaldo, el reloj sigue funcionando, aunque Arduino esté sin alimentación.

Comunicación:

• Utiliza un protocolo serial de tres hilos (SCLK, I/O y CE), lo que lo diferencia de otros RTC como el DS1307 o DS3231 que usan I2C.

• Es compatible con librerías de Arduino (ej. DS1302.h) que simplifican su programación.

Características Técnicas:

• Voltaje de operación: 2.0V a 5.5V.

• Consumo de corriente: extremadamente bajo (< 1 µA en modo standby).

• Formato de hora: 24 horas o 12 horas con indicador AM/PM.

• Memoria RAM integrada: 31 bytes de almacenamiento adicional para datos temporales.

Aplicaciones:

• Programación de tareas horarias (ej. activar el sistema solo en la noche).

• Registro con sello de tiempo en bitácoras de datos (logs).

• Sistemas de control de acceso y alarmas temporizadas.

En el proyecto, el DS1302 se utiliza para que el sistema solo funcione en los horarios nocturnos o de inactividad de la sala de cómputo, evitando falsas alarmas durante las clases.

Módulo Bluetooth HC-06 (Comunicación Serie Inalámbrica)

Los módulos **Bluetooth clásicos HC-06** permiten establecer comunicación serie inalámbrica mediante el perfil **SPP (Serial Port Profile)**. Son ampliamente utilizados en proyectos con **Arduino** para enviar y recibir datos hacia un teléfono móvil o una PC. Se conectan a través de la interfaz **UART**, por lo que pueden emplearse tanto en los pines serie hardware como mediante la librería SoftwareSerial en pines digitales del Arduino Uno.

Principio de funcionamiento:

- El módulo actúa como un puente entre la comunicación serie por cable (UART) y la comunicación inalámbrica Bluetooth.
- El **HC-06** funciona únicamente en modo **esclavo**.
- Una vez emparejado con el dispositivo remoto, los datos enviados desde Arduino se transmiten de forma inalámbrica como si se tratara de un puerto serie convencional.

Características relevantes:

• Velocidad de transmisión por defecto: **9600 baudios** (HC-05 y HC-06). • Voltaje de alimentación en las placas breakout: **5 V**; la línea **RX** del módulo trabaja a **3.3 V lógico**, por lo que se recomienda un divisor resistivo desde el pin TX de Arduino hacia RX del módulo.

• Emparejamiento con PIN típico: **1234** o **0000**, según el firmware del módulo.

• El **HC-05** dispone de un **modo AT** (configuración) accesible normalmente a **38400 baudios**, que permite cambiar parámetros como el nombre del dispositivo, el PIN de emparejamiento o la velocidad de transmisión. **HC-06:** solo funciona como **esclavo** y su conjunto de comandos AT es **más limitado**. Aun así, podés cambiar parámetros básicos como:

• Nombre del dispositivo

• PIN de emparejamiento

• Velocidad en baudios

Conexiones

El sistema se conectó de la siguiente manera:

La pantalla LCD 16x2 con adaptador I2C se alimentó con 5V y GND del Arduino, mientras que sus líneas de comunicación SDA y SCL se conectaron a los pines analógicos A4 y A5 respectivamente.

El módulo RTC DS1302 se conectó mediante tres hilos: el pin DAT (IO) al pin digital 5, el pin CLK (SCLK) al pin digital 6 y el pin RST (CE) al pin digital 4, además de VCC a 5V y GND a tierra.

El sensor DHT11 se conectó con su pin de señal al pin digital 7, y con alimentación a 5V y GND.

El sensor LDR se configuró como un divisor de tensión: un extremo del LDR al pin de 5V, el otro extremo al pin analógico A0, acompañado de una resistencia de 10kΩ entre el pin A0 y GND.

El sensor de fuego KY-026 se conectó con su salida digital (DO) al pin digital 8, la salida analógica (AO) al pin analógico A1, y sus pines de VCC y GND a 5V y tierra, respectivamente.

Finalmente, el módulo lector de tarjetas SD se conectó mediante el bus SPI: el pin CS a la digital 10, MOSI a la digital 11, MISO a la digital 12 y SCK a la digital 13, más la alimentación correspondiente de 5V y GND.

Conexión del módulo Bluetooth:

• TXD del HC-05/HC-06 → pin D0 (RX de SoftwareSerial en Arduino).

• RXD del HC-05/HC-06 ← pin D1 (TX de SoftwareSerial)

• VCC → 5V; GND → GND (masa común).

Objetivos del proyecto

General:

• Implementar un sistema embebido económico y confiable para monitorear condiciones ambientales y eventos de riesgo (fuego) en una sala de computación, con visualización local y registro histórico.

Específicos:

• Medir temperatura/humedad (DHT11), luminosidad (LDR) y fuego (KY-026).

• Marcar cada registro con fecha/hora del RTC DS1302.

• Mostrar en LCD 16×2 (I2C) y en el Monitor Serie.

• Registrar en microSD en modo append; tolerar extracción/inserción sin bloquear el sistema.

• Reinicializar la SD automáticamente cuando sea reinsertada (hot-swap).

• Transmitir la misma línea de datos por Bluetooth a una app móvil (MIT App Inventor) mediante perfil SPP.

• Documentar pinout, librerías y consideraciones de memoria.

Desarrollo del sistema (Software)

• Temporización a 1 s con `millis()`.

• Lecturas: DHT11 (D7), LDR (A0), fuego DO (D8 = LOW indica fuego), RTC DS1302 (IO=5, SCLK=6, CE=4).

• Visualización: línea compacta en Serie; LCD con hora/temperatura en la primera fila y humedad/luz/estado de fuego en la segunda.

• SD (SPI): CS=10; se fija pinMode(10, OUTPUT) antes de SD.begin() para asegurar al UNO como maestro.

• Registro en datalog.txt en modo append para agregar datos y no sobre escribirlos. Si SD.open() falla se marca sd_ok=false y se reintenta SD.begin() cada 2 s sin detener Serie/LCD.

• Bluetooth HC-06: se inicializa con Serial.begin(9600) y se envía la misma cadena de datos que va al monitor serie. El módulo trabaja como esclavo y transmite vía SPP (Serial Port Profile) hacia el dispositivo Android. La comunicación con Arduino se realiza mediante el puerto serie (pines **TX** y **RX**), transmitiendo datos en formato ASCII o binario.

• 'Guard' de año: si Year()<2024 o Year()>2099, se ajusta una vez con __DATE__/__TIME__ y se desactiva protección de escritura si aplica.

• La app en MIT App Inventor recibe esta cadena vía Bluetooth Client (SPP) y la muestra/almacena.

Formato de datos

En Monitor Serie (cada 1 s):

• <temp>C | <hum>% | <luz>% | <ON/OFF> | DD/MM/AAAA HH:MM:SS |

En microSD (datalog.txt, modo append):

• DD/MM/AAAA HH:MM:SS, Temp:<temp>, Hum:<hum>, Luz:<luz>, Fuego:<ON/OFF>

En Bluetooth (SPP, misma frecuencia):

• <temp>C | <hum>% | <luz>% | <ON/OFF> | DD/MM/AAAA HH:MM:SS |

Metodología de pruebas

• Arranque sin SD: verificar que Serie/LCD continúan y que se reintenta SD cada 2 s.

• Inserción en caliente: insertar SD con el sistema corriendo → detección, creación/append de datalog.txt.

• Extracción en caliente: retirar SD durante ejecución → error de apertura, sd_ok=false; al reinsertar, recuperación y continuidad del log.

• RTC (año): forzar año inválido y comprobar corrección automática al iniciar.

• Sensores: DHT11 a 1 Hz; variación del LDR con luz; fuego DO=LOW como evento.

• Bluetooth: emparejar HC-05/06 con el teléfono, abrir la app (MIT App Inventor) y verificar la recepción de la cadena `datos` a 9600 baudios sin pérdidas.

Resultados y observaciones

• El sistema muestra y registra datos cada 1 s de manera estable.

• La SD opera en modo append y tolera extracción/inserción sin bloquear Serie/LCD.

• El ‘guard’ del RTC evita años obsoletos; los sellos de tiempo son consistentes.

• La transmisión Bluetooth replica la línea de datos del Monitor Serie y puede visualizarse en la app móvil.

Limitaciones y mejoras futuras

• Mejorar precisión con DHT22/SHTxx y RTC DS3231 (I2C).

• Optimizar memoria reemplazando `String` por `snprintf` si se ampliara el sistema.

• Añadir buzzer/LED de alarma y umbrales configurables por usuario.

Conclusiones

El sistema cumple con los objetivos planteados: mide variables críticas del ambiente, visualiza en tiempo real, registra en microSD con tolerancia a hot-swap y sella cada entrada con fecha/hora del RTC. La integración por Bluetooth permite enviar la misma cadena de datos a una app móvil, ampliando las capacidades de monitoreo y registro sin modificar la lógica base.

Anexo A – Resumen de pines

Módulo	Señal	Arduino
DS1302	IO / SCLK / CE	D5 / D6 / D4
LCD I2C	SDA / SCL	A4 / A5 (0x27)

DHT11	DATA	D7
LDR	OUT (divisor)	A0
Fuego KY-026	DO (LOW=FUEGO)	D8
Fuego KY-026	AO (no usado)	A1
microSD	CS / MOSI / MISO / SCK	D10 / D11 / D12 / D13
Bluetooth HC-05/06	TXD / RXD	→ D0 (RX) / ← D1 (TX con divisor)

Nota SPI: configurar pin 10 como OUTPUT antes de SD.begin(SD_CS). Para Bluetooth, usar divisor resistivo en RX del módulo y 9600 baudios en modo datos.

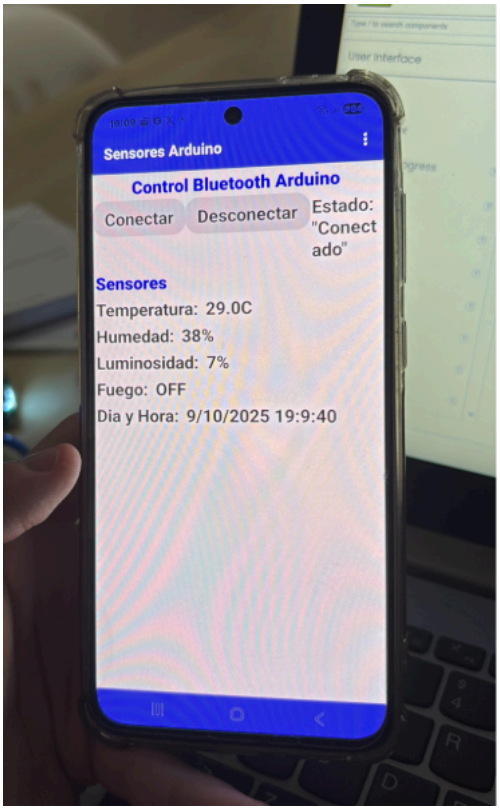
Aplicación Móvil:

En este proyecto se desarrolló una aplicación móvil utilizando MIT App Inventor con el objetivo de recibir y visualizar los datos provenientes de nuestros sensores. La comunicación entre ambos dispositivos se realizó a través de un módulo Bluetooth HC-06, lo que permite la transmisión inalámbrica de la información.

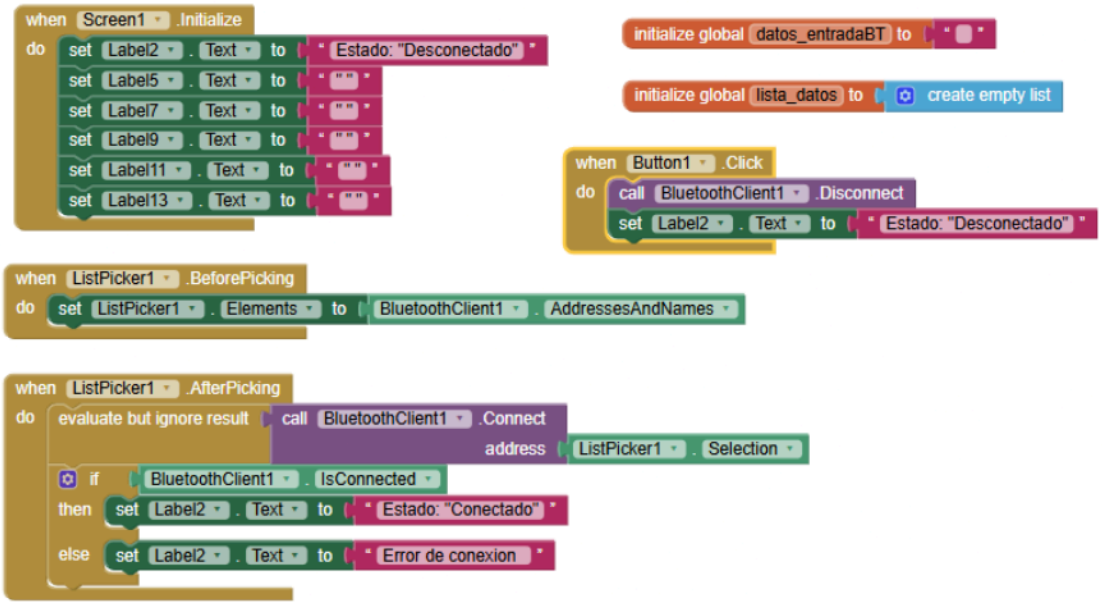
Nuestro objetivo es visualizar en tiempo real los valores obtenidos de los sensores para un correcto monitoreo de la sala de computación.

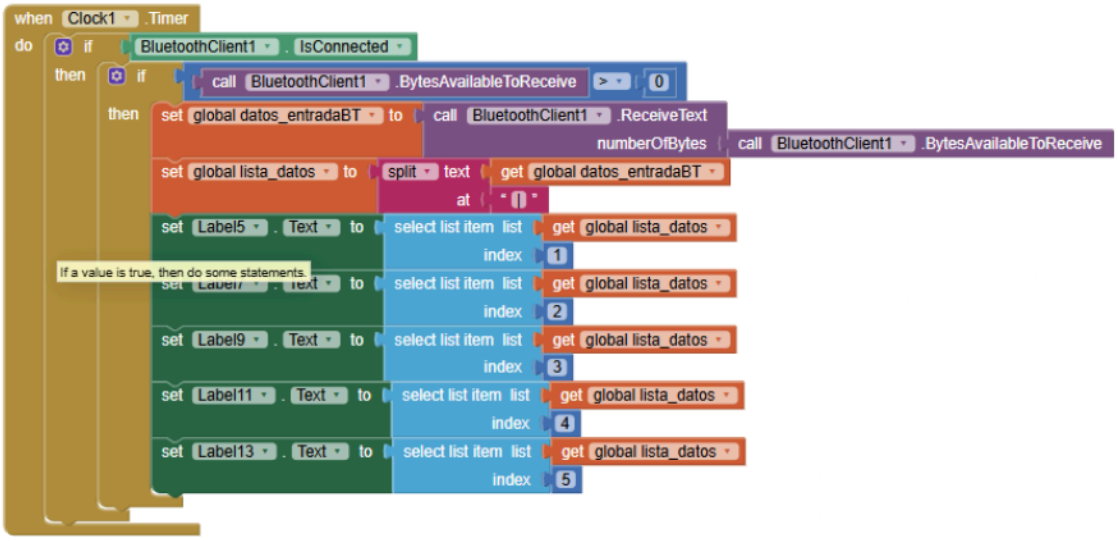
Diseño de la aplicación en MIT App Inventor:

Se creó una interfaz gráfica simple que permite al usuario conectarse al módulo HC-06 y visualizar los datos recibidos. Se añadieron botones para conectar/desconectar y un área de texto para mostrar los valores.



Programación con bloques: Se utilizaron bloques de MIT App Inventor para gestionar la conexión Bluetooth y la recepción de datos. Los datos recibidos se procesan y se muestran en pantalla en tiempo real.




when Clock1 .Timer
do if BluetoothClient1 . IsConnected
then if call BluetoothClient1 .BytesAvailableToReceive > 0
then set global datos_entradaBT to call BluetoothClient1 .ReceiveText
numberOfBytes call BluetoothClient1 .BytesAvailableToReceive
set global lista_datos to split text get global datos_entradaBT
at " [] "
set Label5 . Text to select list item list get global lista_datos
index 1
set Label7 . Text to select list item list get global lista_datos
index 2
set Label9 . Text to select list item list get global lista_datos
index 3
set Label11 . Text to select list item list get global lista_datos
index 4
set Label13 . Text to select list item list get global lista_datos
index 5
If a value is true, then do some statements.