

Alumno: Joaquin Cortabarría

Carrera: TS Telecomunicaciones - Fecha: Noviembre 2023

## **Proyecto Integrador Programación: Evidencia 11 - Sistema completo**

Hasta aquí, a través de las evidencias 7, 8, 9 y 10 hemos completado el desarrollo de los principales componentes de este proyecto integrador. Ya contamos con la base de datos (NodoTemperatura.sql) en funcionamiento, el código (Termometro.py y conexion.py) necesario para el registro de los datos en la BD, y también contamos con el prototipo físico del dispositivo conformado por los componentes electrónicos y el código (para Arduino) correspondiente.

Ahora nos queda un paso más para unir todo el desarrollo, y básicamente es programar y poner a punto la conectividad de todos los componentes. Particularmente la conexión entre el dispositivo IoT y la PC donde corre la BD, para registrar los datos enviados por el sensor del dispositivo.

### **Actualización del código para la aplicación de la PC**

Primero vamos a necesitar instalar la librería Pyserial ejecutando el comando “pip install pyserial” en el terminal. Haciendo uso de esta librería podremos obtener los datos enviados desde el dispositivo e integrarlo en el código para su escritura en la base de datos.

Las siguientes líneas de código fueron incorporadas ([ver el código en repositorio anexo](#), Proyecto en VScode)

```
import serial
ser = serial.Serial('COM5', 9600, timeout=1)
temp_str1 = ser.readline()
```

De manera que junto a un bucle While logramos escribir el dato de temperatura (Float) recibido desde el dispositivo (Int). La frecuencia utilizada para esta prueba fue de una medición por segundo, que si bien es una frecuencia alta, nos permitirá comprobar el funcionamiento básico de nuestro proyecto de forma instantánea.

## Prueba piloto del proyecto.

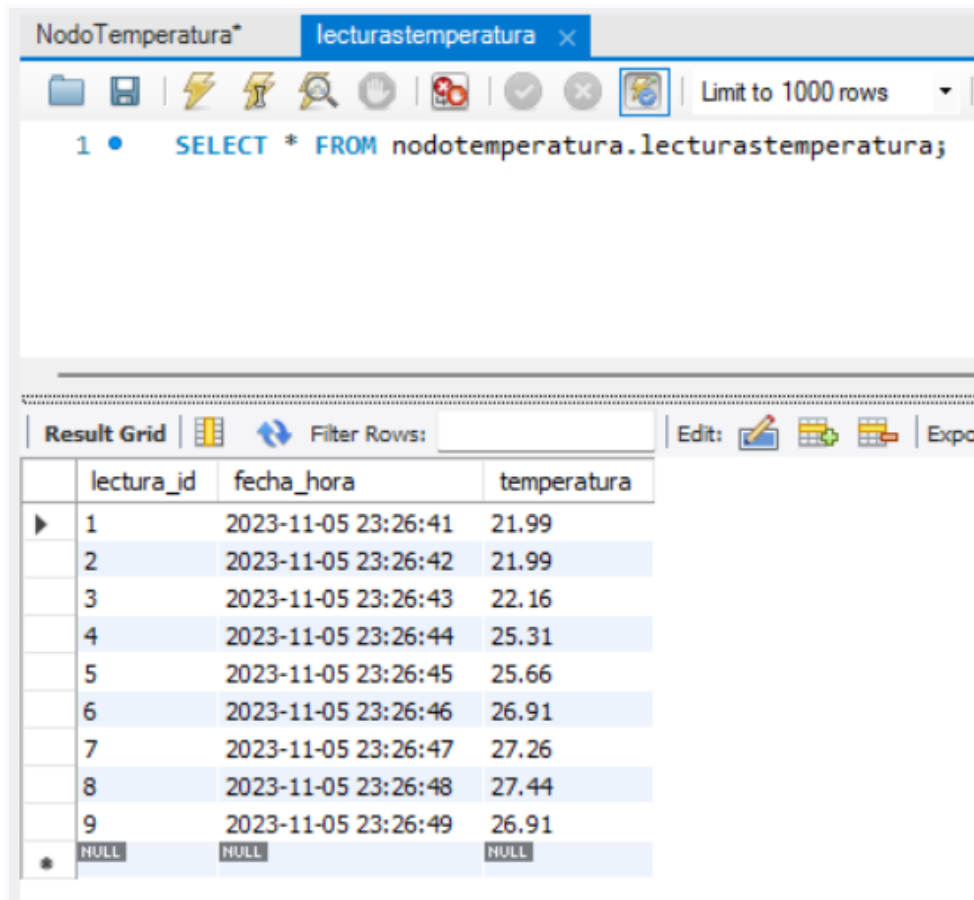
Realizamos la prueba siguiendo los siguientes pasos.

- Se conectó el dispositivo a través del puerto serie COM5
- Se puso en servicio la base de datos en MySQL, con Xampp.
- Se ejecuto el software en la PC
- Se eligio la opcion de recolección de datos (En este caso la opción 2)

El resultado de las mediciones fue el siguiente según el terminal de ejecución:

```
▼ TERMINAL Python + ▾ 🗑️ ...
ams/Python/Python311/python.exe c:/Users/Usuario/Desktop/repositorios/Dispositivos_I
nteligentes/Evidencias_Programacion/11_SistemaCompleto/Codigos_Aplicacion/Termometro
.py
Conexión exitosa
Bienvenido, las opciones son:
- Leer la tabla de temperaturas, ingrese 1
- para comenzar a recolectar los datos de temperatura desde el dispositivo, ingrese
2
- Insertar nuevo dato de temperatura, ingrese 3
- Modificar una medicion de temperatura, ingrese 4
- Eliminar un registro de la tabla, ingrese 5
- Para desconectarse de la BD y Salir, ingrese 0 (cero)
Ingrese el numero correspondiente a la opcion deseada:2
Datos insertados correctamente
Temperatura a registrar en °C:
21.99
Datos insertados correctamente
Temperatura a registrar en °C:
21.99
Datos insertados correctamente
Temperatura a registrar en °C:
22.16
Datos insertados correctamente
Temperatura a registrar en °C:
25.31
Datos insertados correctamente
Temperatura a registrar en °C:
25.66
Datos insertados correctamente
Temperatura a registrar en °C:
26.91
Datos insertados correctamente
```

Luego comprobamos que los datos fueron registrados en la DB correctamente, y en el mismo orden:



The screenshot shows a database management interface with a query window titled 'NodoTemperatura\*' and a tab for 'lecturastemperatura'. The query is 'SELECT \* FROM nodotemperatura.lecturastemperatura;'. Below the query, a 'Result Grid' displays the data. The grid has three columns: 'lectura\_id', 'fecha\_hora', and 'temperatura'. It contains 10 rows of data, with the last row showing NULL values. The interface also includes a toolbar with various icons and a 'Limit to 1000 rows' dropdown.

	lectura_id	fecha_hora	temperatura
▶	1	2023-11-05 23:26:41	21.99
	2	2023-11-05 23:26:42	21.99
	3	2023-11-05 23:26:43	22.16
	4	2023-11-05 23:26:44	25.31
	5	2023-11-05 23:26:45	25.66
	6	2023-11-05 23:26:46	26.91
	7	2023-11-05 23:26:47	27.26
	8	2023-11-05 23:26:48	27.44
	9	2023-11-05 23:26:49	26.91
*	NULL	NULL	NULL

La prueba fue exitosa.

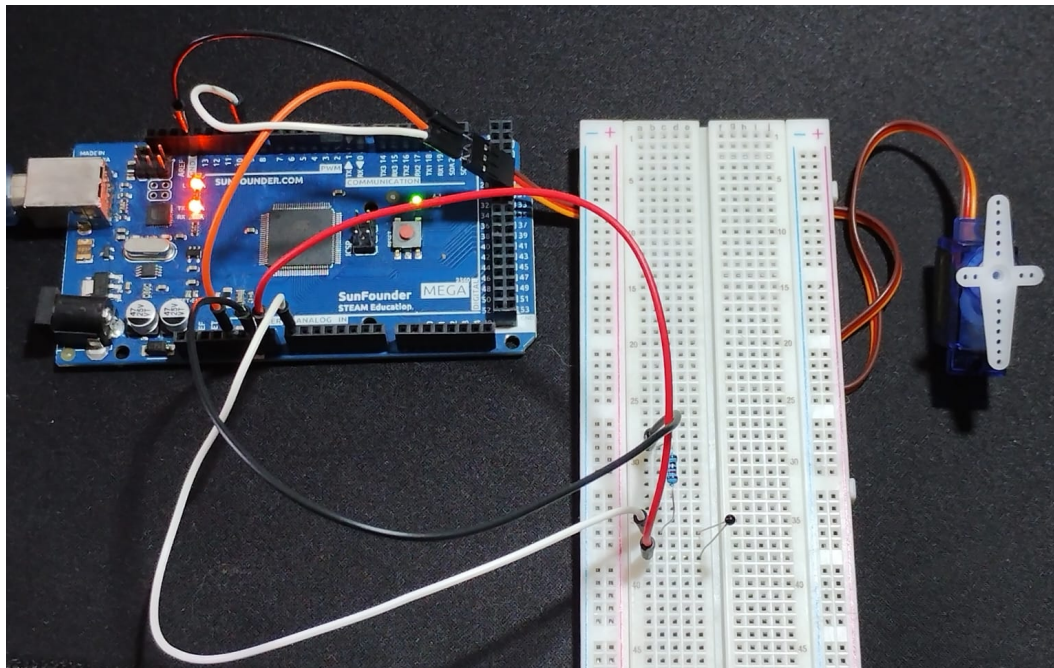
## Posibles mejoras

Se pueden proponer múltiples mejoras ya que el desarrollo del proyecto está en fase de prototipo. En esta primera etapa podríamos plantear algunas mejoras:

- Escalar la cantidad de datos que podemos registrar en la DB (ejemplo: batería, estado del servo)
- Agregar opciones para elegir un modo de trabajo del servo.
- Disponer de una interfaz gráfica para usuarios.
- Definir una frecuencia óptima para la toma de valores.

## Prototipo físico

Se puede ver el video de demostración ([en formato gif](#))



## ANEXOS:

### Repositorio:

Los siguientes archivos del proyecto se encuentran en el [repositorio](#)

Codigo\_Dispositivo

Codigo\_Dispositivo.ino

Wokwi\_termoActuador.zip

Codigos\_Aplicacion

conexion.py

NodoTemperatura.sql

Termometro.py

Video\_calibracion\_TermoActuador.gif

### Proyecto en Wokwi:

[Aqui esta](#) el enlace del proyecto en el simulador Wokwi.

<https://wokwi.com/projects/380111597344082945>

Link textual del repositorio:

[https://github.com/joacorta/Dispositivos\\_Inteligentes/tree/main/Evidencias\\_Programacion/11\\_SistemaCompleto](https://github.com/joacorta/Dispositivos_Inteligentes/tree/main/Evidencias_Programacion/11_SistemaCompleto)