



UNIVERSIDAD NACIONAL DEL LITORAL

Facultad de Ingeniería y Ciencias Hídricas

Ingeniería en Inteligencia Artificial

Introducción a los Sistemas Ciber-físicos

Tutor/a: Juan Carrique

JTP1: Tomas Molas

JTP2: Bruno Zorzet

JTP3: Anibal Fernandez

Proyecto Final Compostera

Estudiante:

Pulvirenti, Gonzalo Iván

Pinto, Paulo

Favre, Mateo Agustin

Macchi Beltrocco, Nicolas

Bustaber, Matias

Messin, Joaquin Ignacio

OBJETIVOS

Diseñar un sistema de monitoreo para la compostera de la FICH que mida temperatura interna y humedad ambiente utilizando sensores digitales, asegurando registros confiables.

Implementar señales que permitan identificar estados del proceso, de manera que el usuario pueda interpretar el estado sin necesidad de acceder al sistema.

Simular el funcionamiento del sistema mediante un gemelo digital en Wokwi, garantizando que las condiciones de activación de alarmas se comporten según los rangos establecidos para el proceso de compostaje.

Documentar la selección de materiales, su justificación técnica y su disponibilidad en proveedores.

RESUMEN

El presente proyecto aborda la necesidad de gestionar los residuos orgánicos generados en la FICH mediante la optimización del proceso de compostaje. Para ello, se plantea el diseño y simulación de un sistema de monitoreo ciber-físico de bajo costo para la compostera de la facultad. El objetivo principal es desarrollar un prototipo funcional que mide variables críticas como la temperatura interna y la humedad ambiente, asegurando que el proceso alcance las fases termófilas necesarias para la higienización del compost.

El sistema propuesto se centra en un microcontrolador (ESP8266) como unidad central. Para el monitoreo, se seleccionó el sensor DS18B20 por su capacidad para medir la temperatura interna de forma impermeable y un DHT22 para la temperatura y humedad ambiente. Como actuadores, se utiliza un LED RGB y un buzzer que indican el estado del proceso (OK, Advertencia o Crítico) basándose en umbrales de temperatura predefinidos. Todo el funcionamiento, la lógica de control y las condiciones de alarma fueron validados mediante un gemelo digital desarrollado en la plataforma Wokwi.

Como resultado, la simulación en el gemelo digital validó el comportamiento esperado del sistema, demostrando su capacidad para monitorear las condiciones y activar las alarmas según los rangos establecidos. Este diseño no solo permite un control eficiente del compostaje y la eliminación de patógenos, sino que también sienta las bases para una futura implementación física de bajo costo y sirve como una herramienta educativa práctica para la institución.

INTRODUCCIÓN

El presente trabajo/proyecto aborda y plantea un diseño y una simulación de un sistema de monitoreo para la compostera de la FICH para abordar la creciente necesidad de gestionar los residuos orgánicos generados dentro de la facultad. Mediante la implementación de un sistema automatizado para el control y monitoreo del proceso de compostaje, buscamos optimizar la producción de compost y contribuir a la sostenibilidad ambiental de nuestra institución educativa. La generación diaria de residuos orgánicos en las cantinas y áreas verdes de la facultad representa una oportunidad significativa para implementar prácticas sostenibles. Si bien el compostaje tradicional es efectivo, requiere supervisión constante y ajustes manuales, lo que puede resultar ineficiente y demandar recursos humanos considerables.

Nuestro sistema automatizado no sólo optimizará este proceso, sino que también servirá como un ejemplo práctico y educativo para estudiantes de diversas carreras. El sistema está diseñado para integrar sensores, actuadores y un microcontrolador Arduino que gestiona las distintas etapas del proceso biológico del compostaje. Su tarea principal es monitorear y controlar variables físicas críticas, como la temperatura, tomando muestras periódicas para asegurar el cumplimiento de las condiciones óptimas de cada fase.

Por ejemplo, el sensor DS18B20 registra las temperaturas internas de la compostera, garantizando que superen los 50°C para eliminar bacterias anaeróbicas y patógenos. Una vez alcanzado este umbral en la etapa termófila, un LED rojo se activa como indicador del progreso del proceso. Una vez la temperatura comienza a bajar, se debe esperar que dicho descenso de la temperatura quede por debajo del umbral de los 20°C, estado en el cual se puede considerar que el compost llegó a la etapa de maduración. En este momento, un segundo actuador (LED verde) señala que el producto se encuentra en niveles de temperatura bajos. Además, también monitoreamos si la temperatura se encuentra entre los 50 y 70 grados centígrados se activa un tercer actuador (LED amarillo).

Materiales seleccionados, descripción y datasheets

Sensor de Temperatura: DS18B20

DataSheet:

<https://www.analog.com/media/en/technical-documentation/data-sheets/ds18b20.pdf>

El DS18B20 es un sensor de temperatura digital que se comunica mediante el protocolo One-Wire, lo que permite conectarlo utilizando un solo pin de datos. Su encapsulado y funcionamiento lo hacen apto para entornos con niveles elevados de humedad. Además, es un sensor ampliamente utilizado en proyectos educativos, lo que facilita el acceso a ejemplos, documentación y soporte.

Sensor de humedad: DHT22

DataSheet: <https://cdn.sparkfun.com/assets/f/7/d/9/c/DHT22.pdf>

El DHT22 es un sensor digital de humedad relativa y temperatura que entrega lecturas por un único pin de datos (protocolo propietario, tiempo de respuesta lento comparado con sensores industriales). Es económico y fácil de usar en prototipos y proyectos educativos.

Actuador Luminoso: LED RGB (Ánodo Común)

DataSheet:

<https://www.sycelectronica.com.ar/catalogo/SYC%20-%20Catalogo%20de%20leds.pdf> (pág. 14)

Se seleccionó un LED RGB de ánodo común para representar visualmente el estado del sistema mediante colores. Esta elección permite utilizar un solo componente en lugar de tres LEDs individuales, lo que reduce el cableado, mejora la prolijidad del montaje y simplifica la instalación.

Placa de Control: ESP8266

DataSheet:

https://www.espressif.com/sites/default/files/documentation/0a-esp8266ex_datasheet_en.pdf

La placa ESP8266 se utiliza como unidad de procesamiento y control. Permite leer el sensor, tomar decisiones de acuerdo a las mediciones y accionar el LED RGB. Incluye conectividad Wi-Fi integrada, lo que posibilita en futuras extensiones el envío o registro de datos sin la necesidad de hardware extra

Materiales descartados:

Sensor: LM35

Actuador: 3 LEDs (rojo, amarillo, verde)

Actuador: Placa ESP32

Proveedores:

Mercado Libre: Utilizado para adquirir la mayoría de los componentes por su variedad, disponibilidad y precios accesibles.

Electrocomponentes S.A.: proveedor local confiable para componentes específicos y de mejor calidad.

Eneka (Uruguay): alternativa regional para asegurar disponibilidad en caso de faltantes locales.

SISTEMA DE MONITOREO PARA LA COMPOSTERA

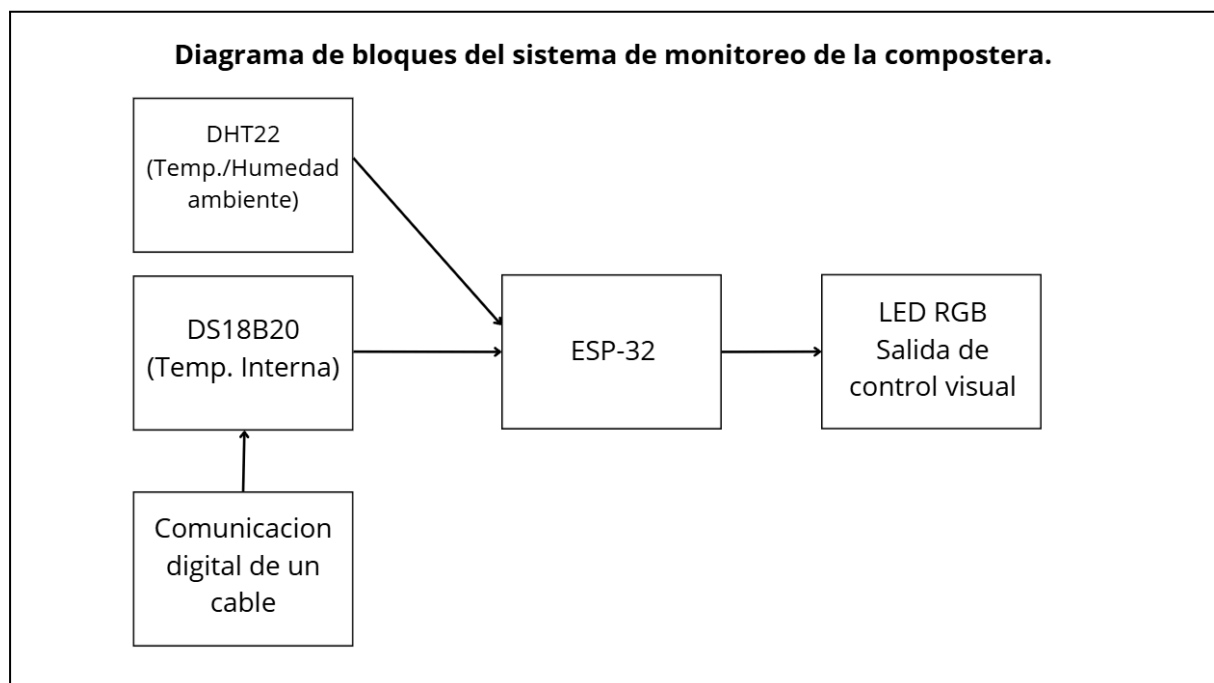
Diagrama del sistema de monitoreo:

El sistema de monitoreo de la compostera se diseñó utilizando la placa ESP32 (arduino) como unidad principal de control y actuador del sistema de compostaje, a la cual le vamos a agregar varios elementos para poder realizar el sistema:

- **Sensor DHT22:** encargado de medir la temperatura y humedad del ambiente de la compostera.
- **Sensor DS18B20:** utilizado para medir la temperatura interna de la mezcla orgánica.
- **LED RGB:** indicador visual de los distintos estados de funcionamiento (verde: normal, amarillo: advertencia, rojo: crítico).

El sistema realiza una lectura constante de los sensores y, según los valores obtenidos, activa las salidas correspondientes.

El diagrama general muestra la interacción entre los sensores, la placa ESP32, idéntico al de la placa ESP8266 y los actuadores, estableciendo el flujo de información desde la medición hasta la respuesta visual..



Selección de sensores

Para la selección de sensores tuvimos en cuenta los siguientes criterios: rango de temperatura, costo, facilidad de conexión y resistencia al entorno.

Sensor	Rango de temperatura	costo	facilidad de conexión	resistencia al entorno
DHT22	-40°C a 80°C	~\$5.000	Simple (1 pin de datos)	Puede dañarse frente a la humedad
LM35	-55°C a 150°C	~\$3.500	Salida analógica, requiere ADC estable	No impermeable
DS18B20	-55°C a 125°C	Depende la medida: ~\$3.000- ~\$15.000	Digital, con protocolo One- Wire	Impermeable, viene sellado herméticamente, no entra humedad ni vapor

Se seleccionaron:

- **DHT22:** se seleccionó por ofrecer un equilibrio adecuado entre precisión, costo y facilidad de uso. Además su salida digital simplifica la conexión con el microcontrolador y evita errores asociados a lecturas analógicas ruidosas, asegurando mediciones confiables con una configuración mínima
- **DS18B20:** El sensor LM35 requiere un trabajo adicional de sellado para protegerse, se seleccionó como mejor alternativa el DS18B20, que no solo trabaja dentro del rango de temperatura adecuado para el proceso de compostaje, sino que además utiliza un protocolo de comunicación digital simple y confiable.

Estos sensores permiten monitorear las magnitudes físicas más relevantes para el control del proceso de compostaje: temperatura y humedad.

Selección de actuadores

Tomamos como una idea práctica el incorporar un led, el cual reflejaba el estado del material orgánico sin un alto consumo, manteniendo los costos y consumo eléctrico bajos

Para la selección del mismo buscamos algo práctico y sencillo, que simplifique la interpretación del estado del sistema a través de 3 colores. Consideramos Leds individuales por color, y un led RGB

Led	Ventajas	Desventajas
Led RGB	reduce el cableado, la cantidad de pines necesarios y los puntos de conexión	Si se rompe, se pierde toda la señalización. Precio levemente mayor
3 Led separados	Muy fácil de interpretar. Fácil de reemplazar si uno se quema. Costo bajo	Más cables y más espacio en el montaje. Usa más pines de salida.

Si bien el led RGB tiene un costo mayor, creemos que vale la pena dado a una diferencia despreciable frente a otros componentes. Permitiendo la simplicidad que buscamos, manteniéndonos conservadores frente a los gastos

Con respecto a la placa, Si bien contamos con una, creemos que es óptimo considerar la utilizada en nuestro gemelo digital en wokwi

Placa	Ventajas	Desventajas
Esp32	Mayor provecho del gemelo digital con una mejor simulación; Más estable si el sistema crece o se vuelve más exigente.	Consideramos que tiene un costo elevado, puede consumir más.
Esp 8266	Más familiarizados con la placa, no tuvimos que costearla, consideramos suficiente para la aplicación a dar.	Capacidad limitada: Si en el futuro se agregan más funciones, puede ser insuficiente Menor estabilidad en conexiones Wi-Fi cuando la señal no es buena. Puede volverse lento si intentamos procesar varias cosas al mismo tiempo.

Aunque la idea de una placa más confiable y estable es llamativa lo tomamos como un costo innecesario dado a que contamos con la Esp 8266 y no buscamos escalar el proyecto

Condiciones de alarma

Se establecieron umbrales de referencia para determinar el estado del proceso de compostaje:

Estado	Condición	Indicador visual/sonoro
OK (Termofilia normal)	$50\text{ }^{\circ}\text{C} \leq T \leq 70\text{ }^{\circ}\text{C}$	LED verde
ADVERTENCIA	$T < 50\text{ }^{\circ}\text{C}$ o humedad fuera del rango 30–90 %	LED amarillo + 2 beeps
CRÍTICO	$T > 70\text{ }^{\circ}\text{C}$	LED rojo + 5 beeps

Estos umbrales permiten detectar si la actividad biológica es insuficiente (temperatura baja) o excesiva (sobrecalentamiento), garantizando un control adecuado del proceso de degradación.

Gemelo digital en Wokwi

El gemelo digital fue implementado en Wokwi, utilizando una simulación basada en la placa ESP32, que es muy similar a la ESP8266, adecuada para el código, y los componentes DHT22, DS18B20, LED RGB y un buzzer.

El código fue desarrollado en micro python, e incluye un mecanismo de simulación de valores ya que el sensor DS18B20 no entrega lecturas reales (por razones las cuales desconocemos).

El programa:

- Lee las magnitudes físicas.
- Evalúa las condiciones predefinidas.
- Activa los actuadores de forma correspondiente.
- Muestra por consola los valores y el estado del sistema.

A continuación se comparte el link del Gemelo Digital para ingresar haga click en el siguiente [Enlace](#)

Componente (Tipo/Modelo)	Variable en Código	Línea/Pin (GPIO)	Protocol o	Alimentaci ón	Convertor de Nivel
CPU					
Placa ESP32 (WROOM-32)	No aplica	No aplica	No aplica	3.3V (Lógica)	No aplica
Sensores					
Temp./Hum. Ambiente (DHT22)	dht_sensor	15	Digital (Prop.)	3.3V	No
Temp. Interna (DS18B20)	ds, ow	13	1-Wire	3.3V	No
Actuadores					
LED Rojo (parte de RGB)	led_r	12	Digital (PWM)	3.3V	No
LED Verde (parte de RGB)	led_g	14	Digital (PWM)	3.3V	No
LED Azul (parte de RGB)	led_b	27	Digital (PWM)	3.3V	No

https://docs.google.com/spreadsheets/d/1o2yncFVxxH35GUXKBddsc0WY8aFmcbB728Bz7_U9wG4/edit?usp=sharing

Aclaración de Alimentación (3.3v ó 5v)

Todo el sistema de monitoreo está diseñado para operar con una lógica de 3.3V, que es el voltaje propio de la placa ESP32.

El ESP32 opera a 3.3V, el sensor DHT22 es compatible con 3.3V, el sensor DS18B20 opera en un rango de 3.0V a 5.5V, por lo que no hay problemas con ellos, y los actuadores (LED RGB y Buzzer) son controlados directamente por los pines GPIO del ESP32, que entregan una señal de 3.3V.

Aclaración de Conversión de Niveles: No se necesita ningún tipo de convertor de niveles lógicos. Todos los componentes seleccionados (sensores y actuadores) son compatibles con el nivel de voltaje de 3.3V de los pines GPIO del ESP32. Esto simplifica el circuito, reduce costos y minimiza puntos de falla.

Diagrama de flujo del algoritmo principal

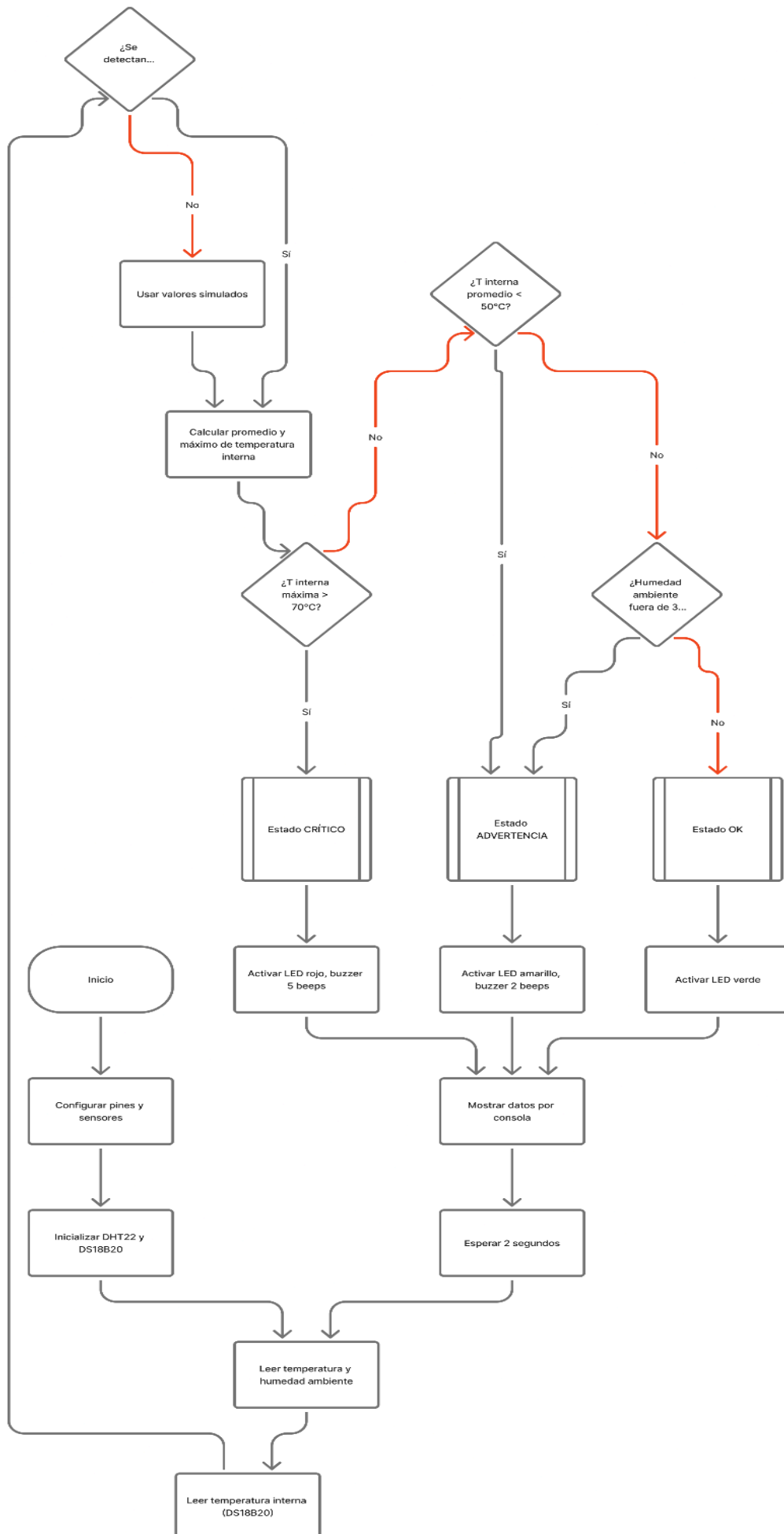
El siguiente diagrama describe el proceso lógico del sistema, siguiendo un flujo descendente:

Inicio -> Configurar pines y sensores -> Inicializar DS18B20 -> Lectura de temperatura y humedad -> Lectura de temperatura interna -> Verificar sensores -> Calcular promedio y máximo -> Evaluar condiciones:

- Si $T_{\text{interna}} > 70\text{ }^{\circ}\text{C}$ significa que está en estado crítico.
- Si $70 > T_{\text{interna}} > 50\text{ }^{\circ}\text{C}$ o HR fuera de rango significa que está en advertencia
- En caso contrario todo el estado está OK

Luego el sistema actualiza los indicadores (LED y buzzer), muestra los datos por consola y repite el ciclo cada 2 s.

En la siguiente página se encuentra el diagrama de flujo.



Código de MicroPython

```
from machine import Pin
from time import sleep
import dht
import onewire, ds18x20

# pines
PIN_DHT      = 15          # DHT22
PIN_1WIRE     = 13          # DS18B20 (DQ)
PIN_LED_R     = 12          # LED rojo
PIN_LED_G     = 14          # LED verde
PIN_LED_B     = 27          # LED azul
PIN_BUZZ      = 5           # buzzer

COMMON_ANODE = True

# umbrales del proceso
TEMP_OK_MIN = 50.0         # °C
TEMP_OK_MAX = 70.0         # °C
HUM_AMB_MIN = 30.0         # %HR
HUM_AMB_MAX = 90.0         # %HR

# simulación de respaldo
# como el sensor digital esta en beta no tira valores, por lo tanto uso
valores de respaldo
FORCE_SIM_IF_ZERO = True
SIM_VALUES = [55.0, 57.5, 53.0]

print("BOOT | DHT@{}, 1-Wire@{}, LED R/G/B {} {} {}, BUZZ {}".format(
    PIN_DHT, PIN_1WIRE, PIN_LED_R, PIN_LED_G, PIN_LED_B, PIN_BUZZ))

# sensores
dht_sensor = dht.DHT22(Pin(PIN_DHT))
ow = onewire.OneWire(Pin(PIN_1WIRE))
ds = ds18x20.DS18X20(ow)
roms = ds.scan()
print("ROMs iniciales:", roms if roms else "ninguna (se reintenta)")

# actuadores
```

```

led_r = Pin(PIN_LED_R, Pin.OUT); led_r.off()
led_g = Pin(PIN_LED_G, Pin.OUT); led_g.off()
led_b = Pin(PIN_LED_B, Pin.OUT); led_b.off()
buzzer = Pin(PIN_BUZZ, Pin.OUT); buzzer.off()

def led_color(r: bool, g: bool, b: bool):
    if COMMON_ANODE:
        led_r.value(0 if r else 1)
        led_g.value(0 if g else 1)
        led_b.value(0 if b else 1)
    else:
        led_r.value(1 if r else 0)
        led_g.value(1 if g else 0)
        led_b.value(1 if b else 0)

def beep(times=1, t_on=0.1, t_off=0.1):
    for _ in range(times):
        buzzer.on(); sleep(t_on)
        buzzer.off(); sleep(t_off)

def estado_ok():
    led_color(False, True, False)    # verde

def estado_advertencia():
    led_color(True, True, False)    # amarillo
    beep(times=2, t_on=0.08, t_off=0.12)

def estado_critico():
    led_color(True, False, False)    # rojo
    beep(times=5, t_on=0.1, t_off=0.1)

# loop para que arroje lecturas cada 15ms
SCAN_EVERY = 15
loops = 0

while True:
    # de DHT22
    try:
        dht_sensor.measure()
        t_amb = float(dht_sensor.temperature())
        h_amb = float(dht_sensor.humidity())
    except Exception as e:

```

```

        print("Error DHT22:", e)
        t_amb, h_amb = float("nan"), float("nan")

# de DS18B20
if (not roms) or (loops % SCAN_EVERY == 0):
    roms = ds.scan()

t_internas = []
have_real = False
if roms:
    try:
        ds.convert_temp()
        sleep(0.75)
        for r in roms:
            val = ds.read_temp(r)
            if val is not None:
                v = float(val)
                t_internas.append(v)
        # ¿hay alguna lectura distinta de 0.0?
        have_real = any(abs(v) > 0.01 for v in t_internas)
    except Exception as e:
        print("Error DS18B20:", e)

if FORCE_SIM_IF_ZERO and (not roms or not have_real):
    t_internas = SIM_VALUES[:]

# salidas
t_prom = sum(t_internas)/len(t_internas) if t_internas else float("nan")
t_max = max(t_internas) if t_internas else float("nan")

print("---- Lecturas ----")
print("T internas:", ["{:.1f}".format(x) for x in t_internas] if t_internas else "N/D")
print("T int prom: {:.1f} °C | T int máx: {:.1f} °C".format(t_prom, t_max) if t_internas else "Sin DS18B20")
print("T amb: {:.1f} °C | HR amb: {:.1f} %".format(t_amb, h_amb))

critico = (t_internas and t_max > TEMP_OK_MAX)
advert_temp_baja = (t_internas and t_prom < TEMP_OK_MIN)
advert_hum_amb = (h_amb == h_amb) and not (HUM_AMB_MIN <= h_amb <= HUM_AMB_MAX)

```

```
    if critico:
        estado_critico()
        estado_txt = "CRITICO (T interna > {:.0f}
°C)".format(TEMP_OK_MAX)
    elif advert_temp_baja or advert_hum_amb:
        estado_advertencia()
        if advert_temp_baja and advert_hum_amb:
            estado_txt = "ADVERTENCIA (T baja y HR amb fuera de rango)"
        elif advert_temp_baja:
            estado_txt = "ADVERTENCIA (T interna < {:.0f}
°C)".format(TEMP_OK_MIN)
        else:
            estado_txt = "ADVERTENCIA (HR amb fuera de rango)"
    else:
        estado_ok()
        estado_txt = "OK (termofilia en rango)"

    print("Estado:", estado_txt)
    print("-----")
    loops += 1
    sleep(2)
```


APARTADO DE ENERGÍA

A continuación vamos a presentar el cálculo de la autonomía de nuestro sistema y criterios que adoptamos para hacer más eficiente y duradero al sistema.

-Tensión de trabajo del sistema: 3.3 V.

-Muestreo: 1 vez cada 12 h, entonces: 14 mediciones/semana.

-Duración de cada medición: 1s

-Corriente activa máxima según datasheets sacadas del fabricante (eso es cuando está midiendo):

-Cálculo de corriente durante una medición=ESP8266: 80 mA + DS18B20: 2 mA + DHT22: 80 mA

-Corriente total durante una medición = **162 mA**. Esta corriente estará funcionando durante una medición de 1 segundo aproximadamente

Ahora calculamos la potencia consumida por cada medición de 12 horas que es un sensado que tiene como duración 1s:

Potencia que ocupan al momento de sensar:

-Ps1 (Potencia del sensado)= $1s \cdot 1h/3600s \cdot 0,162A \cdot 3,3V = \mathbf{0,001485Wh}$

-Ps2 (Potencia de la placa prendida) = $1h \cdot 80mA \cdot 3,3V = \mathbf{0,264Wh}$

LEDs RGB:

Estos leds tienen una corriente de 20 mA cuando están encendidos, según nuestro criterio y para que no gaste demasiada energía con un solo led que está prendido las 24 horas del día durante semanas, adoptamos agregar un pulsador, el cual, al ser se prende durante 10 segundos. Para el cálculo adoptamos una cantidad de veces que se pulsa y en promedio serán unas 30 veces por periodo de 12 h.

-Períodos por día: $30 \times 2 = 60$ pulsos/día, cada pulso mantiene el LED 10 s, entonces por medición (1 vez cada 12 horas), está prendido el led durante 300s. Durante cada pulso asumimos que el ESP también está con una corriente máxima (por lo tanto consume 80 mA) + LED 20 mA durante cada evento de 10 s, y luego cuando no se está prendido el pulsador usamos una corriente mínima de **0.1 mA (100 μA)** a 3.3 V.

-Ps3 (Potencia cuando está prendido)= $300s \cdot 1h/3600s \cdot 0,02A \cdot 3,3V = \mathbf{0,055 Wh}$

-Tiempo en el que no está prendido el Led RGB: $(3600s \cdot 12) - 300s = 42.900s$

-Ps4 (Potencia cuando no está prendido)= $42900s * 1h/3600s * 0,0001A * 3,3V=0,0079325Wh$

-Potencia total utilizada durante un ciclo (12 Horas)= $Ps1 + Ps2 + Ps3 + PS4=0,3280875Wh$

-Potencia total utilizada durante una semana= $0,3280875Wh * 14= 4,593225Wh$

-Para elegir la batería usamos una batería de 12V con un valor de corriente que debe suministrar la batería por hora. Para ello primero optamos por un valor de eficiencia del 90%, es decir que debe suministrar más:

$Pt=4,593225Wh/0.90=5,1035Wh$

-Corriente de la batería como mínimo= $5,1035 Wh/12V=0,4252Ah$

-Bateria x hora= $0,4252Ah/168h=2,53mA$

Batería elegida para el sistema es: Batería de Litio recargable 12 V de 2000 mAh mucho más que los 425mAh porque pueden llegar a generarse picos en la corriente y puede afectar a los 12V de la batería, también por si no llega a respetar del todo el rendimiento del 90%

Especificación de estrategia de recarga de batería

Opciones de recarga:

Una opción es la recarga por USB / red eléctrica (más simple):

Cargar con un cargador USB-C 5V con circuito de carga Li-ion (MPU/charger IC).

Usar cargador con corriente de carga adecuada (p. ej. 1 A a 5 V para batería grande) y un controlador de carga que limite corriente y gestione protección.

Estrategia: mantener SOC (estado de carga) entre 20% y 90% para mayor vida útil. Evitar carga completa frecuente al 100% si el sistema está siempre enchufado.

Otra opción es recarga por panel solar (autonomía off-grid parcial):

Si quieres recarga solar, usar controlador MPPT (mejor rendimiento) y dimensionar panel para generar al menos la energía diaria esperada en promedio.

Ejemplo rápido: si consumo medio diario ≈ 5.71 Wh, un panel de 5–10 W con ~ 4 h SOL pico puede recargar suficiente energía. (Aquí hay muchas variables: irradiación local, inclinación, sombreado — si quieres lo calculo con datos de Córdoba).

Estrategia de recarga que utilizamos:

- Cargar la batería cada vez que esté por debajo del 30% si es posible. Si hay recargas limitadas, priorizar mantener por encima del 20% para evitar desgaste y apagados.
- Si hay fuente intermitente (solar/USB), usar prioridad a carga y bypass: cuando haya fuente, alimentar la placa directamente y simultáneamente cargar la batería (power path). Usar controlador con gestión de carga y protección (carga CC/CV para Li-ion).
- Registrar el estado de carga y programar alertas vía WiFi (si el sistema lo permite) cuando SOC < 25%.
- Si el LED consume mucho, programar el LED para modo ahorro cuando SOC < 40% (p. ej. notificaciones por parpadeo corto en vez de encendido continuo).

Videos

En este apartado se compartirán los links de los videos:

Video de 2:10 minutos - [Video con solución](#)

Video de 7:46 minutos - [Video detrás de escenas](#)

Aprendizajes y Evidencias del Proceso

Durante el desarrollo del proyecto, el grupo adquirió una serie de aprendizajes técnicos significativos, principalmente en el proceso de selección de componentes, la optimización del sistema y la superación de desafíos de simulación.

1. Aprendizajes en Selección de Componentes:

- **Importancia de la Resistencia al Entorno:** El aprendizaje clave fue la priorización de la robustez de los sensores. Se compararon el DHT22, LM35 y DS18B20. Se descartaron los dos primeros por su vulnerabilidad a la humedad, concluyendo que el sellado impermeable del DS18B20 era un requisito no negociable para el interior de la compostera, además de su conveniente protocolo de comunicación One-Wire.
- **Optimización de Variables (Descarte del Sensor de Humedad):** Inicialmente se consideró medir la humedad interna, pero se aprendió que esta variable podría **inferirse indirectamente a través de la temperatura**. Además, se constató que los sensores de humedad económicos (como el DHT22) no son fiables en entornos de humedad saturada constante, y uno especializado sería demasiado costoso, llevando a una optimización del diseño.
- **Balance entre Simplicidad y Costo (Actuadores):** Al comparar LEDs individuales con un LED RGB, se aprendió que, aunque el RGB tiene un costo levemente mayor, la ventaja de reducir el cableado, la cantidad de pines GPIO utilizados y los puntos de conexión justificaba su elección para mantener la simplicidad del montaje.
- **Balance entre Potencia y Disponibilidad (Placa):** Se aprendió a diferenciar entre la placa ideal para la simulación (ESP32) y la placa suficiente para la implementación (ESP8266). Se concluyó que, aunque la ESP32 es más potente, la ESP8266 era suficiente para la aplicación actual y, al ya contar con ella, se evitaba un costo innecesario.

2. Aprendizajes en Simulación y Desarrollo:

- **Validación con Gemelo Digital:** El uso de Wokwi fue fundamental para desarrollar y probar la lógica de control del sistema.
- **Superación de Limitaciones:** Se encontró una limitación en la simulación (el sensor DS18B20 no entregaba lecturas reales). El aprendizaje fue crear un mecanismo de simulación de valores dentro del propio código MicroPython. Esto permitió probar y validar el diagrama de flujo y las condiciones de alarma (OK, Advertencia, Crítico) sin depender del hardware físico.

3. Aprendizajes en Diseño y Eficiencia:

- **Optimización de Energía:** El análisis de consumo (Apartado de energía) reveló que los actuadores, como el LED, serían grandes consumidores. El aprendizaje fue implementar una estrategia de ahorro (ej. un pulsador que enciende el LED solo por 10 segundos) para reducir drásticamente el consumo y permitir una autonomía de batería viable.
- **Simplificación del Circuito:** Se diseñó todo el sistema para operar con una lógica de 3.3V, compatible con la placa ESP y los sensores seleccionados. El aprendizaje fue que este diseño elimina la necesidad de convertidores de niveles lógicos, lo que reduce costos, complejidad y potenciales puntos de falla en el circuito final.

ANEXO A: FÓRMULAS

Ps1: Consumo del Sensado (1 segundo)

$$Ps1 = (1 \text{ s} * (1 \text{ h} / 3600 \text{ s})) * I_{\text{sensado}} * V_{cc}$$

Ps2: Consumo de Placa Activa (1 hora)

$$Ps2 = t_{\text{activa}} * I_{\text{activa}} * V_{cc}$$

Ps3 y Ps4: Tiempos del LED RGB (Ciclo 12h)

$$t_{\text{led_on}} = 30 \text{ pulsos} * 10 \text{ s/pulso} = 300 \text{ s}$$

$$t_{\text{led_off}} = (12 \text{ h} * 3600 \text{ s/h}) - t_{\text{led_on}} = 42.900 \text{ s}$$

Ps3: Consumo con LED Prendido

$$Ps3 = (t_{\text{led_on}} * (1 \text{ h} / 3600 \text{ s})) * I_{\text{led_on}} * V_{cc}$$

Ps4: Consumo con LED Apagado (Reposo)

$$Ps4 = (t_{\text{led_off}} * (1 \text{ h} / 3600 \text{ s})) * I_{\text{reposo}} * V_{cc}$$

Consumo Total por Ciclo (12 Horas)

$$P_{\text{total_12h}} = Ps1 + Ps2 + Ps3 + Ps4$$

Consumo Total Semanal

$$(2 \text{ ciclos por día} * 7 \text{ días}) = 14$$

$$P_{\text{semanal}} = P_{\text{total_12h}} * 14 \text{ ciclos}$$

Energía Requerida (con Eficiencia n = 90%)

$$\text{Energía_requerida} = P_{\text{semanal}} / n$$

ANEXO B: AMPLIACIÓN DE APRENDIZAJES Y EVIDENCIAS DEL PROCESO

El resumen de aprendizajes presentado en el cuerpo principal del informe se basa en el siguiente registro de actividades, decisiones técnicas y evidencias recolectadas durante el desarrollo del proyecto.

B1. Cronograma de Actividades

A continuación, se presenta el cronograma de actividades que el grupo utilizó para la gestión del proyecto, detallando los responsables de cada tarea clave.

Día/Fecha	Actividad Realizada	Responsables
Día 2	Investigación y comparativa de sensores.	Favre, Mateo; Messin, Joaquin
Día 2	Selección de plataforma.	Pulvirenti, Gonzalo; Bustaber, Matias
Día 3	Creación del "Gemelo Digital".	Pinto, Paulo; Macchi Beltrocco, Nicolas
Día 4	Desarrollo de la lógica de control.	Favre, Mateo; Messin, Joaquin
Día 4	Desafío Técnico: Detección de limitación simulación DS18B20.	Pulvirenti, Gonzalo; Bustaber, Matias
Día 5	Solución: Creación de código para simular valores.	Pinto, Paulo; Macchi Beltrocco, Nicolas
Día 5	Análisis de consumo energético.	Favre, Mateo; Messin, Joaquin
Día 6	Montaje físico del prototipo (Planificación).	Pulvirenti, Gonzalo; Bustaber, Matias

B2. Memoria Técnica y Evidencias

Esta sección amplía las decisiones técnicas tomadas (resumidas en la sección "Aprendizajes") y presenta la evidencia que las respalda.

B2.1. Evidencia de Selección de Componentes

Las siguientes tablas, extraídas del análisis en el cuerpo del informe, justifican la selección de hardware.

Tabla B.1: Comparativa de Sensores de Temperatura

Sensor	Rango de temperatura	Costo	Facilidad de conexión	Resistencia al entorno
DHT22	-40°C a 80°C	~\$5.000	Simple (1 pin de datos)	Puede dañarse frente a la humedad
LM35	-55°C a 150°C	~\$3.500	Salida analógica, requiere ADC estable	No impermeable
DS18B20	-55°C a 125°C	~\$3.000-\$15.000	Digital, con protocolo One-Wire	Impermeable, viene sellado herméticamente

Tabla B.2: Comparativa de Actuadores

Led	Ventajas	Desventajas
Led RGB	reduce el cableado, la cantidad de pines necesarios y los puntos de conexión	Si se rompe, se pierde toda la señalización. Precio levemente mayor
3 Led separados	Muy fácil de interpretar. Fácil de reemplazar si uno se quema. Costo bajo	Más cables y más espacio en el montaje. Usa más pines de salida.

Tabla B.3: Comparativa de Placas

Placa	Ventajas	Desventajas
Esp32	Mayor provecho del gemelo digital con una mejor simulación, Más estable si el sistema crece	Consideramos que tiene un costo elevado, puede consumir más
Esp 8266	Más familiarizados con la placa, no tuvimos que costearla, consideramos suficiente para la aplicación	Capacidad limitada: Si en el futuro queremos agregar más sensores, puede quedarse corto.

REFERENCIAS

BBVA. ¿Qué es el compost y cuáles son sus fases y beneficios? (16 de Mayo 2025):
<https://www.bbva.com/es/sostenibilidad/que-es-el-compost-y-cuales-son-sus-fases-el-poder-del-suelo-vivo/>

Venturemfgco. Una breve guía para seleccionar actuadores para sus proyectos de robótica(8 de Junio 2021):
<https://www.venturemfgco.com/blog/a-brief-guide-to-select-actuators-for-your-robotics-projects/>

ALLDATASHEET. (2018, septiembre). Electronic Components Datasheet Search.
<https://www.alldatasheet.com/> Components101. (2024, noviembre). Datasheets.
<https://components101.com/>

OpenAI. (2024, noviembre 19). ChatGPT: Your AI assistant.
<https://chatgpt.com/>

MERCADO LIBRE: <https://www.mercadolibre.com.ar/>

[Repositorio en GitHub](#)