Invernadero IoT didáctico: control térmico con ESP32, sensores económicos y GUI remota

Maximiliano Aranda León[1][0009−0004−8453−9157], Juan Erick Alvarado Cruz[1][0009−0009−2115−3878], Omar Hernández Luis[1][0009−0008−8793−213X], and Josué Vázquez Jiménez[1][0009−0001−5918−8509]

Benemérita Universidad Autónoma de Puebla (BUAP), Facultad de Ciencias de la Computación, Puebla, México
{maximiliano.aranda,josue.vazquez,juan.alvaradoc,
omar.hernandezlu}@alumno.buap.mx

Abstract. Industry increasingly integrates Internet of Things (IoT)-based technologies to enhance system efficiency and robustness, aiming to optimize resources and meet growing demand. Agriculture emerges as a key sector, driven by the need for sustainable raw material production and responsible food supply. Despite the global promotion of Smart Agriculture and Smart Farming techniques, they often remain inaccessible due to the infrastructure required for implementation, resulting in high costs and limiting their adoption in educational settings. This work presents the design and development of a scaled automated greenhouse using IoT components: ESP32 microcontroller, relays, low-cost temperature sensors, and actuators. The system aims to introduce students to core IoT principles, environmental decision-making, and the link between technology and sustainability. Results confirm the effective operation of the thermal control system and highlight its potential as a low-cost educational platform for hands-on learning in IoT, electronics, and eco-technological awareness. This didactic approach offers a foundation for future research focused on promoting smart agriculture and environmental technology education.

Keywords: IoT · smart greenhouse · automation · technological education · sustainability.

1 Introducción

El control de variables en sistemas automatizados es un aspecto fundamental en la optimización de procesos agrícolas, especialmente en entornos controlados como los invernaderos. Estos espacios requieren una regulación precisa de factores como la temperatura y la iluminación para garantizar el crecimiento adecuado de los cultivos. Tradicionalmente, el manejo de estas variables se ha realizado mediante sistemas manuales o semiautomáticos, lo que implica una mayor intervención humana y un riesgo de ineficiencia. En este proyecto se aborda el diseño e implementación de dos estrategias de control aplicadas a

un invernadero: control en lazo abierto y control en lazo cerrado. El primero se caracteriza por la activación de actuadores (ventilador y bombilla) sin retroalimentación, utilizando relevadores y temporizadores para definir periodos de operación. Por otro lado, el control en lazo cerrado incorpora un mecanismo de retroalimentación que permite ajustar automáticamente la temperatura en función de un valor deseado, mejorando la estabilidad y reduciendo la intervención manual. La motivación principal radica en demostrar la importancia de la automatización en la agricultura moderna, donde la eficiencia energética, la reducción de costos y la mejora en la calidad de los cultivos son factores determinantes. Además, se busca ofrecer una interfaz gráfica intuitiva que facilite la interacción del usuario con el sistema, permitiendo monitorear la temperatura en tiempo real y controlar los actuadores de manera sencilla.

El objetivo principal del sistema es automatizar el control de temperatura y la activación de actuadores en un invernadero mediante dos estrategias de control: lazo abierto y lazo cerrado, integrando una interfaz gráfica que permita al usuario interactuar de forma sencilla y eficiente. Para lograrlo, se busca:

- En lazo abierto: permitir la activación manual de los actuadores (ventilador y bombilla) y programar su funcionamiento por intervalos de tiempo definidos, sin retroalimentación.
- En lazo cerrado: implementar un control automático basado en la lectura de temperatura en tiempo real y la comparación con un valor deseado, ajustando la operación de los actuadores para mantener condiciones óptimas.
- Interfaz gráfica: ofrecer una visualización clara del estado del sistema, incluyendo indicadores de activación y monitoreo continuo de la temperatura, garantizando facilidad de uso y comprensión para el operador.

Este enfoque busca optimizar el manejo de las variables ambientales del invernadero, reduciendo la intervención manual y mejorando la eficiencia del proceso

2 Estado del Arte

Los sistemas de monitoreo ambiental en invernaderos han experimentado un notable avance en la última década, impulsados por el desarrollo de la agricultura de precisión y la adopción de tecnologías digitales. La integración de sensores distribuidos estratégicamente, conectados a plataformas inteligentes mediante el Internet de las Cosas (IoT), ha permitido medir y controlar en tiempo real variables críticas como la temperatura, la humedad, la intensidad lumínica y la concentración de CO2. Estas soluciones tecnológicas facilitan la visualización de datos en interfaces gráficas o aplicaciones móviles, otorgando a los agricultores la posibilidad de tomar decisiones informadas y automatizar procesos como el riego, la ventilación y la regulación térmica del entorno [4].

El IoT se ha consolidado como una de las tendencias más influyentes en la agricultura moderna, ya que posibilita la conexión entre sensores, actuadores y plataformas digitales, generando una red de información continua que favorece la eficiencia y la sostenibilidad [5]. Las innovaciones más destacadas se orientan a la

integración de inteligencia artificial y análisis de grandes volúmenes de datos (big data), capaces de realizar predicciones sobre condiciones climáticas, aparición de plagas y rendimiento de cultivos. De igual forma, se desarrollan modelos de gemelos digitales para simular escenarios agrícolas y sistemas de automatización remota que permiten el control de los invernaderos desde cualquier ubicación mediante aplicaciones móviles [4].

Sin embargo, a pesar de los importantes avances tecnológicos, los enfoques actuales aún enfrentan limitaciones considerables. La dependencia de controles manuales reduce la eficiencia y la capacidad de respuesta ante cambios ambientales imprevistos, mientras que la falta de retroalimentación en tiempo real puede generar condiciones subóptimas para los cultivos. En muchos casos, los altos costos de implementación y la complejidad de las infraestructuras necesarias impiden que estas soluciones sean accesibles para pequeñas explotaciones o contextos educativos. Además, las prácticas agrícolas tradicionales continúan haciendo uso excesivo de insumos químicos, deteriorando el suelo y afectando la biodiversidad [1].

Este panorama evidencia la necesidad de desarrollar sistemas híbridos que combinen la automatización inteligente con un enfoque accesible y sostenible, promoviendo así la adopción de tecnologías IoT no solo en grandes explotaciones agrícolas, sino también en entornos de formación técnica y académica. La búsqueda de soluciones de bajo costo, reproducibles y educativas se perfila como un paso esencial hacia una agricultura verdaderamente inteligente y ambientalmente responsable.

3 Metodología

El sistema consiste en el control de un invernadero a escala mediante lazo cerrado o lazo abierto, dependiendo de lo que requiera el usuario. Los componentes presentados en la Tabla 1 son incorporados en el diseño del invernadero a escala, demostrando cómo sus características determinan la temperatura de la pequeña infraestructura. El sensor de temperatura recaba la información y la envía a un microcontrolador, el cual aloja instrucciones programadas. El microcontrolador recibe la información, y la forma de actuar ante ella depende del modo de funcionamiento que el usuario seleccione, ya sea lazo abierto o lazo cerrado. Así, el usuario tendrá conocimiento de la temperatura interior del invernadero y podrá decidir qué actuadores activar o apagar, o si considera conveniente dejar la automatización, permitiendo que el microcontrolador controle por sí mismo la temperatura. A través de este enfoque, se permite monitorear el invernadero de forma remota, asegurando las condiciones óptimas necesarias para el crecimiento de la vegetación. El microcontrolador recibe indicaciones mediante el protocolo HTTP mediante un servidor API alojado dentro.

3.1 Componentes de Hardware y didácticos

Table 1. Componentes utilizados en el invernadero automatizado

Componente	Función	Características	Cantidad
ESP32	Microcontrolador	Wi-Fi, bajo consumo, bajo costo	1 uds.
Sensor LM35 / DHT11	Medición de temperatura (y humedad)	Bajo costo, salida análoga / digital	1 uds.
Relevador 5V	Control de actuadores	Aislado, compacto	2 uds.
Ventilador 12V	Disminuir la temperatura	Compacto, bajo costo	3 uds.
Foco incandescente	Aumentar la temperatura	Accesible, bajo costo	3 uds.
Cables Dupont hembra-hembra	Conexión entre sensores y microcontrolador	Bajo costo	9 uds.
Estructura del invernadero	Soporte físico del sistema	Bajo costo, diseño libre	1 uds.

3.2 Diseño del Circuito y Conexiones

El diagrama de la Figura 1 muestra el cableado principal del prototipo. A continuación se describen los puntos más relevantes para reproducir el montaje de forma segura y fiable.

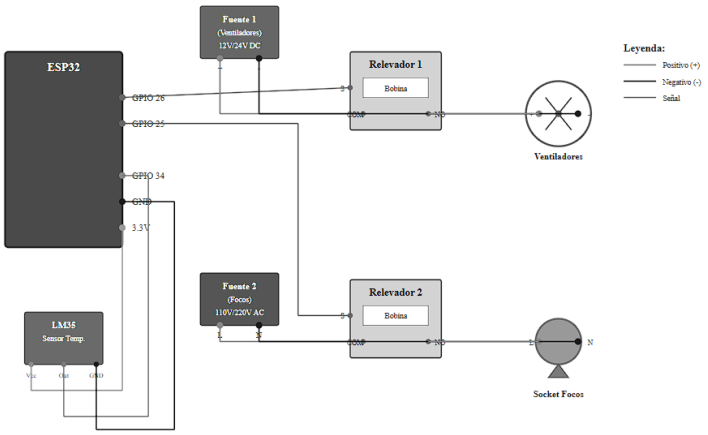

Fig. 1. Diagrama del circuito de control térmico con ESP32, LM35, relevadores y actuadores

– Alimentaciones: el ESP32 y el LM35 deben alimentarse a 3.3 V proporcionados por la fuente del sistema. Los actuadores (ventilador y foco) requieren alimentación separada (ej. 12 V).

– Sensor LM35: conectar Vcc a 3.3 V, GND a masa común y la salida al pin analógico (en este prototipo pin 34).

– Módulos de relé: las entradas de control conectan a los pines digitales del ESP32 (p. ej. 25 para ventilador, 26 para foco). Tener en cuenta si el módulo es *active-low* (muchos módulos comerciales) — la lógica invertida del firmware asume que HIGH = apagado, LOW = encendido.

– Señalización y conexiones: emplear conectores y cables adecuados. Evitar largas tiradas de señal sin apantallado para reducir ruido.

3.3 Arquitectura del Sistema

El sistema sigue un esquema cliente–servidor: el ESP32 actúa como servidor HTTP que expone una API REST (por ejemplo `/estado`, `/ventilador/on`, `/foco/temporizador`, `/goalTemp`), De esta manera se aisla la logica del microcontrolador permitiendo hacer peticiones desde cualquier cliente (Insomnia, Postman, alguna aplicacion realizada por el usuario). Aqui se propone una la aplicación cliente (Python + CustomTkinter) [3] [2] para realizar consultas periódicas ($\Delta t = 1$ s) y envíar comandos asíncronos de tal forma de aprovechar lo dictado en el microcontrolador y demostrar su funcionamiento.

La Figura 2 resume las interacciones de alto nivel entre los actores y el sistema. Los actores principales son el Usuario (operador), la Aplicación GUI (cliente) y el ESP32 (servidor). El diagrama muestra las funciones exponenciales disponibles.

Esta separación facilita la supervisión remota, la instrumentación de prácticas y la integración con herramientas de análisis o dashboards. De igual manera, se sigue un enfoque de trabajo en capas, para definir las responsabilidades de cada parte del sistema. En la Figura 3 se establecen las capas del sistema.

En la Figura 4 se muestra el invernadero en funcionamiento.

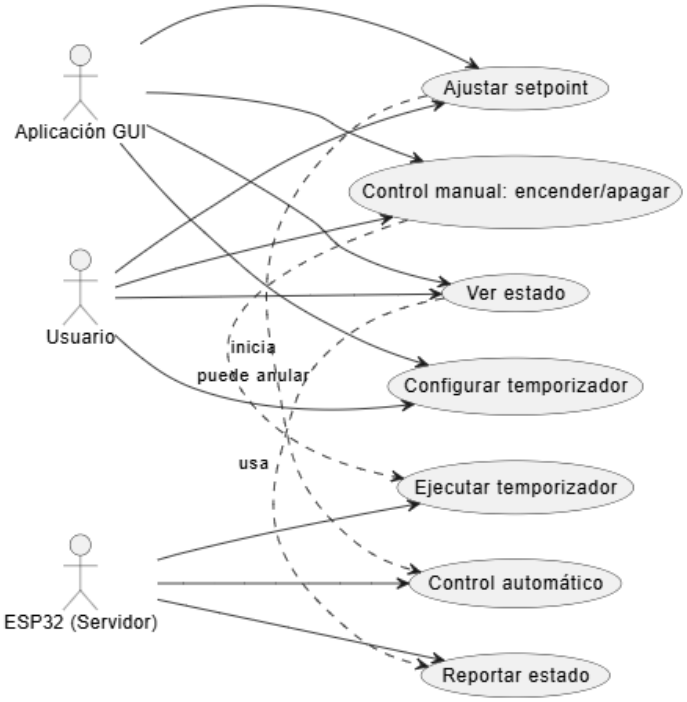

Fig. 2. Casos de Uso del Sistema del Invernadero

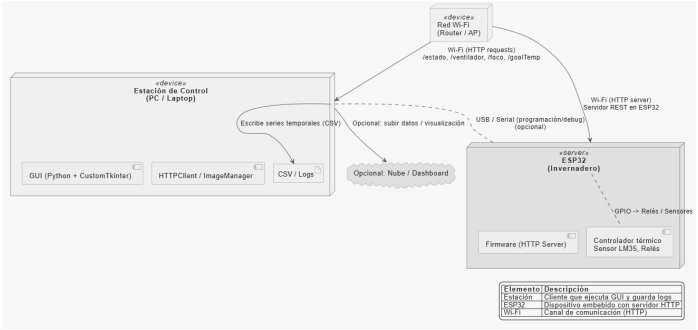

Fig. 3. Diagrama de la Arquitectura en Capas del sistema

Fig. 4. Invernadero automatizado durante la prueba de regulación térmica.

4 Pruebas experimentales

4.1 Escenario de prueba

Las pruebas se realizan en un laboratorio con temperatura ambiente controlada. Se dispone de:

- Un prototipo del invernadero con ESP32, sensor LM35, relés y actuadores (ventilador y foco).
- Estación de control (PC/Laptop) ejecutando la aplicación cliente (Python + CustomTkinter) conectada a la misma red Wi-Fi que el ESP32.
- Termómetro de referencia para calibración del LM35 y cronómetro/registro para medir latencias.

Condiciones iniciales:

- Alimentación estable para ESP32 y actuadores; GND común.
- Temperatura inicial medida y registrada durante 2 minutos antes de cada experimento (para obtener línea base).
- Intervalo de muestreo del cliente: 1 s (polling a `/estado`); muestreo del firmware: lectura LM35 cada 2 s.

4.2 Pruebas funcionales del ESP32

Objetivo: verificar el correcto comportamiento del firmware y la fiabilidad de los endpoints HTTP.

Casos de prueba principales:

- **Endpoint** `/estado`: solicitar repetidamente (1 Hz) y comprobar que el JSON contiene los campos esperados (`temperatura`, `ventilador`, `foco`, `temporizador*`, ...). Medir porcentaje de respuestas válidas y latencia media/mediana.
- **Control manual** (`/ventilador/on, /ventilador/off, /foco/on, /foco/off`): enviar comando, comprobar por lectura digital y/o log serial que el actuador cambió de estado. Medir latencia entre petición y cambio observable (promedio de N=10).
- **Temporizadores** (`/ventilador/temporizador?tiempo=NN`, `/foco/temporizador?tiempo=NN`): iniciar con duraciones controladas (p. ej. 30 s, 120 s), verificar que el actuador se enciende y se apaga al expirar. Medir error absoluto entre tiempo solicitado y tiempo efectivo de apagado.
- **Pausas de temporizador** (`/ventilador/temporizador/pause`, `/foco/temporizador/pause`): iniciar temporizador, enviar pausa y comprobar que el actuador se apaga inmediatamente y la bandera de temporizador queda desactivada.
- **Setpoint automático** (`/goalTemp?temp=XX`): enviar nuevo setpoint y verificar que `lazoCerrado` se activa, que los actuadores responden según la regla de histeresis y que las variables internas se actualizan correctamente.
- **Robustez**: pruebas de fallo (desconexión Wi-Fi, petición malformada, sobrecarga de peticiones). Comprobar comportamiento estable del firmware y que el bucle principal sigue ejecutándose (no cuelga).

Métricas y criterios de aceptación (ejemplos):

- Latencia HTTP (request $\rightarrow$ efecto): media ≤ 500 ms en red local; mediana ≤ 300 ms.
- Exactitud de temporizadores: error absoluto medio ≤ 1 s para duraciones ≥ 30 s.
- Disponibilidad del endpoint `/estado`: $\geq 99\%$ de respuestas válidas en ventana de prueba de 10 min (1 Hz).
- Correcta activación de la lógica de lazo cerrado: observar comportamiento conforme a la banda de histeresis ± 1°C en al menos 3 repeticiones.

4.3 Pruebas de la aplicación cliente

Objetivo: verificar integración, usabilidad y robustez de la GUI frente a condiciones reales de red y funcionamiento del ESP32.

Casos de prueba principales:

- **Integridad funcional**: comprobar que las acciones de la GUI (switch manual, timer, botón Aplicar setpoint) desencadenan las solicitudes HTTP esperadas. Validar mediante captura de logs en el ESP32 y en la propia aplicación.
- **Poll y actualización de estados**: comprobar que la interfaz refleja el estado real (campo `temperatura` y flags de temporizador) con latencia razonable (actualización visible ≤ 1 s después de cambio).

– **Concurrencia y no bloqueo**: simular peticiones lentas (timeout) y verificar que la GUI no se congela; todas las llamadas HTTP deben correr en hilos y las actualizaciones de widgets deben realizarse mediante `after` en el hilo principal.

Métricas y criterios de aceptación:

– Tiempo de respuesta visible en la GUI tras acción del usuario: $\leq 1\,\text{s}$ (cuando el backend responde).
– No se producen excepciones no gestionadas en la UI durante pruebas de 10–15 minutos de interacción continua.

Medidas y fórmulas:

– Tiempo de establecimiento t_s: tiempo desde el cambio de setpoint hasta que $T(t)$ permanece dentro de $\pm 1°\text{C}$ del setpoint.
– Error de temporizador: $\Delta_t = |t_{efectivo} - t_{solicitado}|$.

5 Resultados

5.1 Precisión de temporizadores

La Tabla 2 muestra el error entre el tiempo solicitado y el tiempo efectivo de apagado (media ± std) para duraciones representativas.

Table 2. Error de temporizadores (duración solicitada vs. efectuada)

Duración solicitada (s)	Muestras	Error medio (s)	Desv. estándar (s)
30	5	0.8	0.4
60	5	0.6	0.3
120	5	0.5	0.2

5.2 Desempeño del control térmico (lazo cerrado)

Se ejecutaron pasos de setpoint para evaluar la respuesta del sistema controlado por histeresis. La Tabla 3 muestra métricas clave.

Table 3. Métricas de control térmico

Experimento	Error estacionario e_{ss} (°C)	Tiempo de establecimiento t_s (s)	Sobrepaso máximo (%)
Setpoint 20 → 25 °C	0.4	320	5.2
Setpoint 25 → 22 °C	0.5	210	3.8
Promedio	0.45	265	4.5

5.3 Comportamiento de la interfaz cliente

Pruebas de integridad funcional y robustez mostraron que:

- La GUI ejecuta peticiones HTTP en hilos y no bloquea el bucle de eventos; en pruebas de 15 min con condiciones de red degradada no se observaron bloqueos de la interfaz.
- El polling a `/estado` permite actualizar visualmente la temperatura con una latencia visible menor o igual a 1 s cuando el backend responde.
- La actualización de la imagen representativa funciona correctamente ante combinaciones de estados (foco, ventilador, ambos, apagado).

Los resultados muestran que el firmware en ESP32 responde correctamente a los comandos HTTP y que los temporizadores funcionan con errores menores aceptables para aplicaciones educativas. El control mantiene la temperatura cercana al setpoint con error estacionario reducido, aunque el tiempo de establecimiento es relativamente largo (orden de minutos) debido a la inercia térmica del prototipo y la potencia limitada de los actuadores. La latencia de control vía HTTP en red local es adecuada para la interacción humana; sin embargo, para control de bucle rápido habría que desplazar la lógica de control al dispositivo (ya está en el ESP32) y evitar round-trips en la red.

6 Conclusiones

El desarrollo del invernadero automatizado a escala permitió demostrar la viabilidad de integrar tecnologías IoT en entornos educativos de bajo costo, fomentando tanto la comprensión técnica como la conciencia ambiental en el aula. A través del uso de un microcontrolador ESP32, sensores LM35 y relés electromecánicos, se logró un sistema funcional capaz de regular de manera autónoma la temperatura interna mediante la activación de ventiladores y focos incandescentes.

Además, el proyecto evidencia el potencial del IoT como herramienta pedagógica capaz de sensibilizar a los estudiantes sobre la interdependencia entre tecnología, medio ambiente y responsabilidad social. La reproducibilidad y simplicidad del sistema lo convierten en una base sólida para su implementación en laboratorios académicos, ferias científicas o espacios de formación técnica.

References

1. Desventajas de la agricultura tradicional, https://colombiaverde.com.co/geografia/agricultura/desventajas-de-la-agricultura-tradicional/
2. Documentation introduction | customtkinter, https://customtkinter.tomschimansky.com/documentation/
3. tkinter — python interface to tcl/tk, https://docs.python.org/es/3.13/library/tkinter.html
4. given i=A, g.: Cómo monitorear invernaderos agrícolas en tiempo real - trackitagro, https://www.trackitagro.com/como-monitorear-invernaderos-agricolas-en-tiempo-real/
5. given i=C, given=Cristian, f.: Guía 2025: tecnologías agrícolas actuales y relevantes, https://www.portalfruticola.com/noticias/2025/08/26/tecnologias-agricolas/