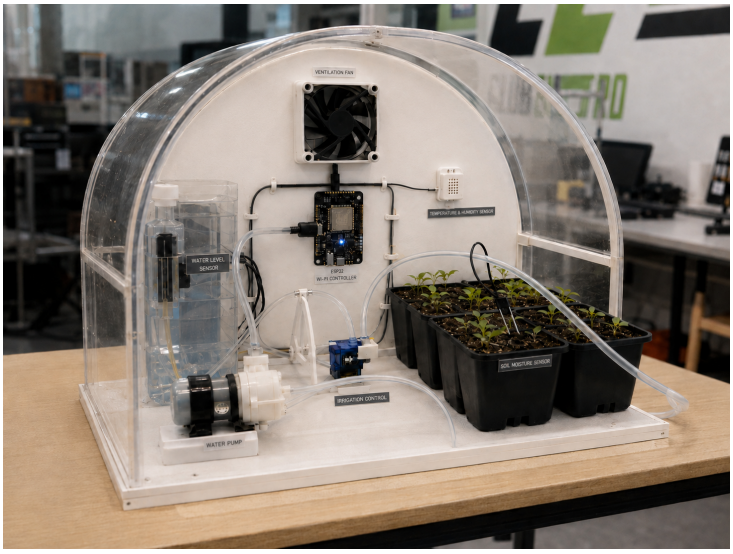
FEATURED PROJECT 02

IoT-Based Smart Farming Management System

ESP32 Environmental Monitoring and Rule-Based Greenhouse Automation

CONTEXT	ROLE	APPLICATION	PERIOD
Electro Scientific Club	Embedded Systems Developer	Smart Greenhouse	2022

Project Overview



System-layout visualization based on the original greenhouse prototype, clarifying the sensing, control, and irrigation arrangement.

Developed the embedded subsystem for a connected greenhouse prototype that combined environmental sensing, local automatic irrigation, and remote telemetry. An ESP32 acquired climate and soil data, executed rule-based actuator logic, and synchronized measurements with Firebase for display in an Android monitoring application.
The system was built within the Electro Scientific Club at M'Hamed Bougara University of Boumerdes as a team engineering demonstrator for more efficient greenhouse supervision and reduced manual intervention.

CORE OUTCOME

Integrated a physical greenhouse demonstrator with ESP32-based local control and Wi-Fi-connected mobile telemetry.

Engineering contributions

- Programmed the ESP32 firmware for DHT22 temperature and humidity acquisition, analog soil-moisture sensing, and water-level input.
- Implemented threshold-based pump actuation with a timed irrigation cycle using non-blocking `millis()` logic.
- Integrated PIR and LDR inputs with buzzer logic and implemented a humidity-driven status output.
- Configured Wi-Fi connectivity, Firebase authentication, and Realtime Database updates for greenhouse telemetry.
- Connected the embedded data layer to an Android dashboard for remote visualization of temperature, air humidity, and soil-moisture data.

System at a glance

ESP32 connected controller
Integrated Wi-Fi, ADC acquisition, GPIO control, and cloud communication.

Four telemetry variables
Temperature, air humidity, soil reading, and reservoir state.

Local irrigation logic
Soil threshold and timed pump cycle execute on the microcontroller.

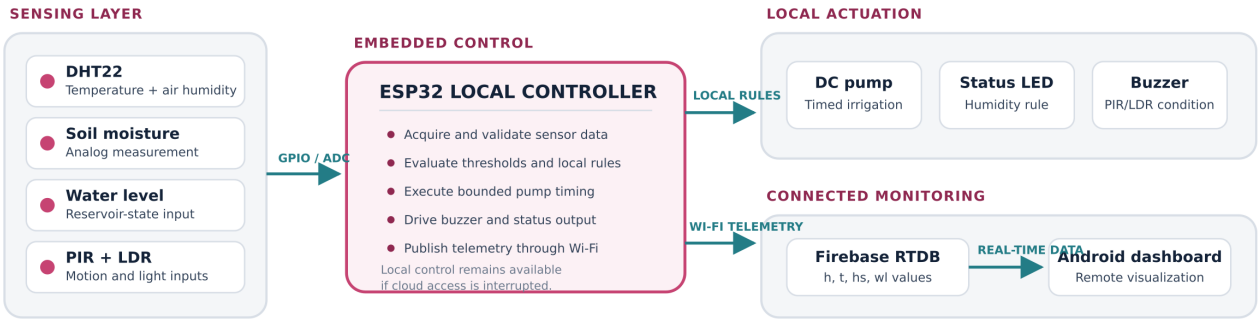
Mobile monitoring interface
Firebase-backed Android dashboard presents live greenhouse data.

TECHNOLOGIES

ESP32 · Embedded C++ · DHT22 · Soil-Moisture Sensor · Water-Level Input · DC Water Pump · PIR · LDR · Wi-Fi · Firebase RTDB · Android Application

EMBEDDED AND IOT ARCHITECTURE

From Environmental Sensing to Local Control and Mobile Data



Verified functional architecture derived from the supplied firmware and Firebase data paths. Local actuation remains independent of mobile-dashboard rendering.

Embedded implementation

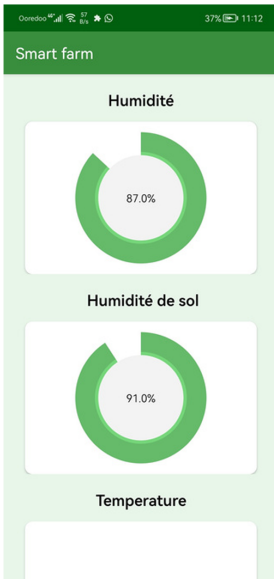
The ESP32 formed the local acquisition and control layer. The DHT22 supplied air temperature and relative humidity, while ADC/GPIO inputs captured soil, reservoir, motion, and light conditions. Local rules controlled the DC pump, buzzer, and status output independently of dashboard rendering.

Control and communication flow

1. Sample the greenhouse sensors and reject invalid DHT22 readings.
2. Compare the soil signal with the configured irrigation threshold.
3. Run the pump for a bounded interval, stop it, and reassess the soil.
4. Evaluate the PIR/LDR alarm condition and the humidity status rule.
5. Publish temperature, humidity, soil, and water-level values to Firebase for display in the Android application.

Engineering decisions

- Kept time-critical actuation on the ESP32 rather than depending on a continuous cloud connection.
- Used the ESP32's integrated Wi-Fi and mixed ADC/GPIO resources to consolidate sensing, actuation, and telemetry on one controller.
- Separated periodic cloud updates from the pump timing logic so the irrigation sequence did not depend on database response time.



Android monitoring dashboard displaying air humidity, soil moisture, and temperature.

RESULTS AND VALIDATION BOUNDARY

The supplied material documents an assembled greenhouse prototype, embedded firmware, Firebase data paths, and an Android monitoring interface. It supports prototype-level validation of sensor acquisition, local pump/output logic, and remote telemetry. No calibration procedure, long-duration crop trial, water-use comparison, control accuracy, or quantified reliability result was provided.

PROPOSED 2026 ARCHITECTURE EVOLUTION — NOT YET IMPLEMENTED

- Move to ESP32-S3 with ESP-IDF/FreeRTOS tasks, watchdogs, persistent setpoints, and fault-state handling while retaining deterministic local control.
- Add RS-485/Modbus sensor and actuator nodes, calibrated probes, pump-current or flow feedback, and reservoir interlocks for multi-zone expansion.
- Replace direct database coupling with MQTT v5 over TLS, device identities, buffered telemetry, and signed OTA updates with rollback.
- Train an irrigation or anomaly model only after collecting labelled time-series data; keep pump limits and fail-safe decisions on the microcontroller.

AI would recommend schedules or flag anomalies at the data layer. The ESP32 would continue to enforce thresholds, maximum pump time, reservoir protection, and safe fallback behaviour.