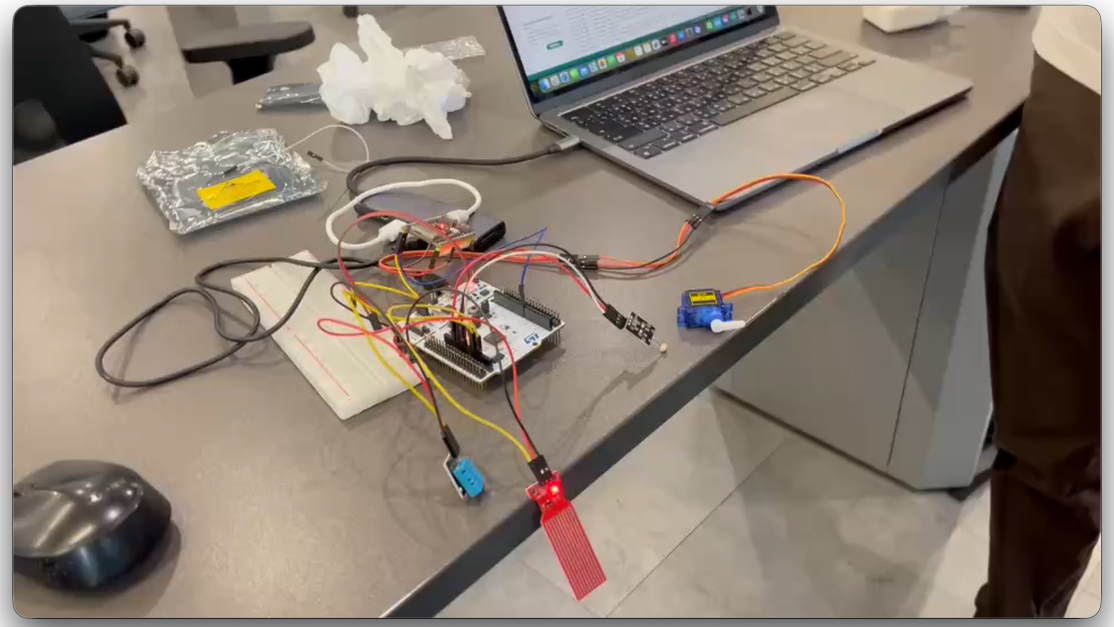
Rain Detection & Clothesline System

Final Project Report



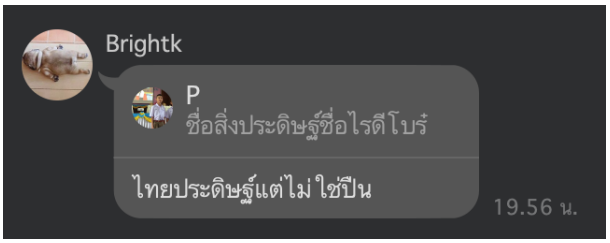
Group 28: 2P

6631343121 Metus Kerdpum
6630180021 Pakornkiat Opaprakasit
6630196021 Piyapanchanok Meeboonsalang
6630054621 Chayutphong Soisri

2025/1
2110366 Embedded System Laboratory I

Department of Computer Engineering
Faculty of Engineering
Chulalongkorn University

Contents



Git Repository:
https://github.com/Pantipkub/embedded-system-final-project.git

1 Introduction

1.1 Project Overview

Unexpected rainfall is a common inconvenience for households, often ruining laundry left to dry outdoors. To address this, our group has developed the **Smart Rain Detection & Automated Clothesline System**. This IoT solution autonomously monitors environmental conditions—specifically rain (via water level/moisture), light intensity, temperature, and humidity—to automatically retract a clothesline when unfavorable conditions are detected.

The system utilizes an **STM32 Nucleo F411-RE** for precision sensor data acquisition and an **ESP8266/ESP32** module for Wi-Fi connectivity, servo motor control, and cloud synchronization with Firebase. Users can monitor the system status in real-time through a modern React-based web dashboard.

1.2 Objectives

- To design a sensor array capable of detecting rain (via water sensor) and day/night cycles (via LDR).
- To implement a robust UART communication protocol between the STM32 (Sensing Unit) and the ESP Module (Control Unit).
- To automate a Servo Motor mechanism that extends or retracts the clothesline based on real-time sensor logic.
- To visualize system health and environmental metrics on a web application.

2 System Architecture

The system architecture is divided into three distinct layers: Sensing, Control/Connectivity, and Application.

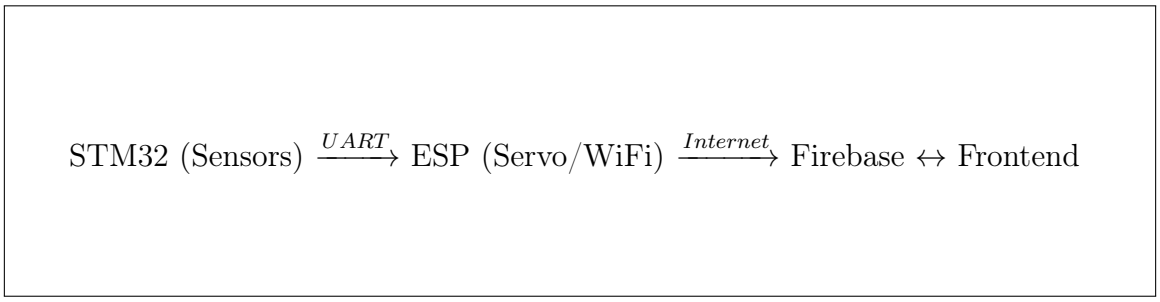


Figure 1: High-Level System Architecture

2.1 Hardware Components

1. **STM32 Nucleo F411-RE:** Acts as the primary Data Acquisition Unit (DAQ). It interfaces with analog and digital sensors.
2. **ESP8266 / ESP32 NodeMCU:** Functions as the IoT Gateway and Actuator Controller. It receives processed data from the STM32, controls the servo motor, and syncs data to Firebase.
3. **DHT11 Sensor:** Measures ambient Temperature and Humidity.
4. **LDR (Light Dependent Resistor):** Monitors light intensity to determine day/night cycles.
5. **Water Level Sensor:** Used as a rain detector; high values indicate rainfall.
6. **Servo Motor:** The actuator responsible for physically extending or retracting the clothesline mechanism.

3 Software Implementation

3.1 STM32 Firmware (Sensing Layer)

The STM32 is responsible for high-speed sensor polling. It utilizes a microsecond delay timer ('DWT') to communicate with the DHT11 sensor and standard ADC polling for the LDR and Water sensors.

Key Logic:

- **Sampling:** LDR and Water sensors are sampled every 10ms and averaged to reduce noise.
- **DHT11:** Read once every 1000ms.
- **Data Formatting:** All sensor data is packed into a JSON string and transmitted via UART1/UART2.

```
// From main.c
if ((now - lastJSONTime) >= 1000) {
    // Calculate Averages
    ldr_avg = ldrSum / ldrCount;
    water_avg = waterSum / waterCount;

    // Send JSON to ESP
    // Format: {"temperature":28,"humidity":65,"ldr":1234,"water":3000}
    Send_JSON(g_temp, g_hum, ldr_avg, water_avg);
}
```

Listing 1: STM32 Main Loop Logic

3.2 ESP Controller (Connectivity & Actuation)

The ESP module parses the JSON received via UART. It contains the *Autonomic Logic* to control the Servo motor. It connects to the Firebase Realtime Database (RTDB) to upload status.

Automated Control Logic: The clothesline retracts if:

$$(\text{Humidity} > 70) \ \wedge \ (\text{Temp} < 25^{\circ}\text{C}) \ \wedge \ (\text{LDR} > 800) \ \wedge \ (\text{WaterLevel} > 1600)$$

```
// From final_project_arduino.ino
String controlServo(float temperature, int humidity, int ldr, int water
    ) {
  // Thresholds: HUMID_HIGH 70, TEMP_LOW 25, LDR_DARK 800, WATER 1600
  if (humidity > HUMID_HIGH && temperature < TEMP_LOW &&
      ldr > LDR_DARK && water > WATER_LEVEL) {

    Serial.println("Condition met: RETRACT clothesline");
    servoMotor.write(180);  // Retract
    return "RETRACTED";

  } else {
    Serial.println("Normal condition: EXTEND clothesline");
    servoMotor.write(0);    // Extend
    return "EXTENDED";
  }
}
```

Listing 2: ESP Servo Control Logic

3.3 Frontend Application (User Interface)

The dashboard is built using **React** (Next.js) and **Tailwind CSS**. It subscribes to the Firebase RTDB to update the UI instantly when sensor values change.

Features:

- **Status Indicators:** Displays whether the system is "Extended" (Sunny) or "Retracted" (Raining).
- **Live Gauges:** Visual bars for Temperature, Humidity, Water Level, and Light.
- **Motor State:** Indicates if the motor is currently running.

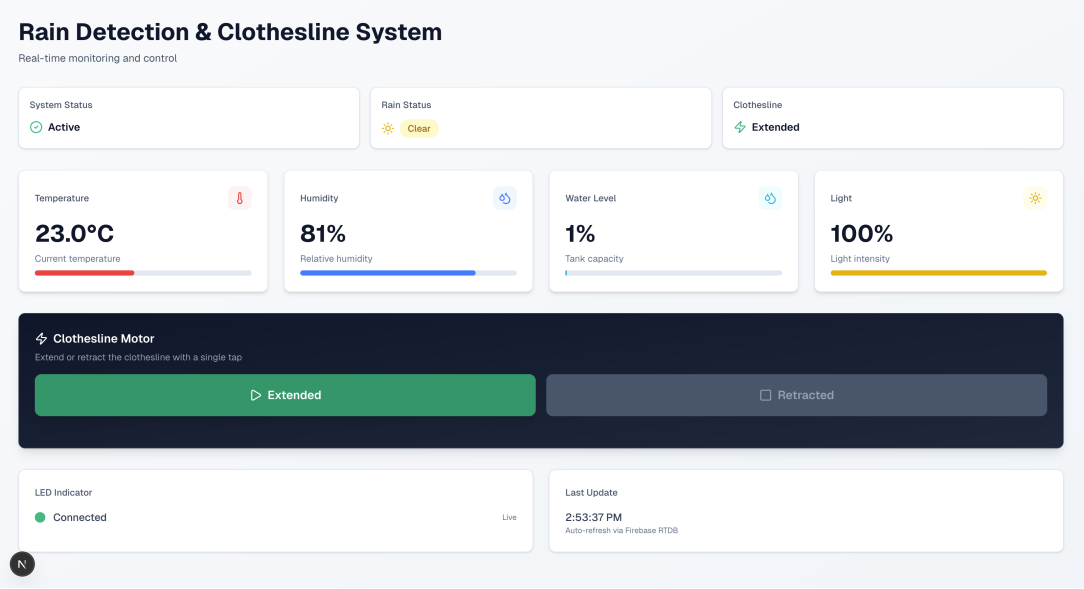


Figure 2: React Web Dashboard Interface

4 Logic Flow & Data Pipeline

1. **Acquisition:** STM32 polls sensors.
   - `DHT11`: Digital protocol (One-wire).
   - `LDR`: Analog (PA0).
   - `Water`: Analog (PA4).
2. **Transmission:** STM32 formats data into a JSON string:

   ```
   {"temperature":28, "humidity":60, "ldr":2048, "water":100}
   ```

3. **Processing:** ESP receives JSON via UART. It checks the "Rain Condition" thresholds.
4. **Actuation:**
   - If Rain/Dark detected → **Servo moves to 180** (Retract).
   - If Clear/Sunny → **Servo moves to 0** (Extend).
5. **Synchronization:** ESP updates Firebase path `/clothesline/status`.
6. **Visualization:** The User's browser receives the update and changes the UI from "Sunny" to "Raining".

5 Group Roles and Responsibilities

6631343121 Metus Kerdpum (Connectivity)

- Configured the Firebase Realtime Database (RTDB) structure for syncing sensor data.
- Implemented the Wi-Fi connection and auto-reconnect logic on the ESP module.
- Managed API authentication and secure data transmission between the hardware and the cloud.

6630180021 Pakornkiat Opaprakasit (Embedded Dev - STM32)

- Developed the core firmware in ‘main.c‘ for the STM32 Nucleo F411-RE.
- Configured ADC peripherals to sample data from the LDR and Water Level sensors.
- Implemented UART serial communication to package sensor readings into JSON format for transmission.

6630196021 Piyapanchanok Meeboonsalang (Frontend & UX/UI)

- Built the interactive web dashboard using React (Next.js) and Tailwind CSS.
- Designed the user interface to visualize real-time temperature, humidity, and rain status.
- Implemented the frontend logic to listen for Firebase updates and reflect motor states instantly.

6630054621 Chayutphong Soisri (Documentations and Team Mgt.)

- Managed the project timeline and coordinated tasks among team members.
- Compiled all the final report and documentations.
- Prepared presentation materials and organized the final project demonstration.