**PROJECT SUBMISSION DOCUMENT**

**Course**: CS G523 **Date**: 04th May 2025

**Work division among team members –**

**Naggender Singh** **(2024H1120216P)**

a.  Responsible for Raspberry Pi setup and Mosquitto MQTT broker integration  
b.  Responsible for ESP-01 UART interface and MQTT client protocol stack

c .MQTT firmware logic on STM32 for structured sensor data publishing and actuator command handling.

d. Designed and tested AT-command-based Wi-Fi communication layer

**Shiven Keshav** **(2024H1120268P)**

a.  Responsible for ESP-01 UART interface and MQTT client protocol stack  
b.  Servo motor control via PWM and GPIO relay driver integration.

c.  Developed and tested MQTT publishing and subscribing modules

d. Conducted end-to-end system testing after firmware updates.

**Vaibhav Tripathi** **(2024H1030079P)**

a.  Responsible for Android app control interface

b.  Designed Android-side MQTT subscribe/publish mechanisms

c. Assisted in wiring setup and power distribution planning for components.

d. implemented manual control options for relay and servo

**Abhishek Bhosale** **(2024H11202 17P)**

a.  Developed Android app screen for real-time sensor data visualization

b.  Implemented soil moisture and light sensing with ADC and data conversion.

c. Developed servo and relay actuation logic based on thresholds.

d.  Collaborated in mobile testing and cross-platform MQTT validation

**Title:** PlantWise: Intelligent Hydration &Environmental Management System

**Proposal**:

Proposal to develop a PlantWise:smart watering and environmental monitoring system designed to automate plant irrigation and improve crop health. It continuously tracks key environmental parameters soil moisture, ambient light, air temperature and humidity—and automatically waters plants when needed. A dedicated Android application provides real-time remote monitoring and control, enabling users to view sensor readings and switch between automatic and manual irrigation modes. By integrating sensors, actuators, and network connectivity, PlantWise aims to optimize water usage and ensure plant wellbeing without constant human intervention.

**Learning outcomes expected:**

1. **Interfacing Sensors & Managing Hardware Challenges**

* Soil Moisture & Light Sensing  
  • Configure ADC channels (PA0, PA1) to sample analog voltages.  
  • Design and calibrate voltage-divider circuits for accurate LDR readings.  
  • Handle channel multiplexing and conversion timing to avoid cross-talk.
* Temperature & Humidity (DHT11)  
  • Implement 1-wire timing using microsecond delays (TIM1) for reliable reads.  
  • Manage sensor read errors and retries in bare-metal code.
* Actuators (Relay & Servo)  
  • Drive a 5 V relay via GPIO while isolating coil noise.  
  • Generate 50 Hz PWM on TIM4 (PB9) for precise servo control.

1. **Programming & Wireless Communication**

* ESP-01 Wi-Fi Module  
  • Initialize UART2 (PA2/PA3) at 115200 baud for AT-command interface.  
  • Manage AT-command sequences to connect to Wi-Fi (SSID/Password).  
  • Implement non-blocking, DMA-assisted UART for simultaneous sensor polling.

1. **MQTT Protocol**  
   • Publish JSON-formatted sensor data (plant/sensors) with QoS 0/1.  
   • Subscribe to control topics (plant/control) and handle incoming commands.  
   • Use mqtt\_process\_uart() to parse asynchronous MQTT notifications.

**Components List**

1. Controller & Broker
   * STM32F407IG-DISC1 (Cortex-M4 @168 MHz, 1 MB Flash, 192 KB RAM)
   * ESP-01 Wi-Fi Module (UART ↔ STM32, MQTT client)
   * Raspberry Pi 3 as MQTT broker (Mosquitto)
2. Sensors
   * Soil-Moisture Module (YL-69/Module)
   * Ambient-Light (LDR) Module (KY-018)
   * DHT11 Temperature & Humidity Module
3. Actuators
   * 1-Channel Relay Module (5 V) driving mini water pump
   * REES52 Futaba S3003 Servo Motor via L298N driver (PWM)

**Android App**

* PlantWise App: real-time dashboard,
* manual/auto mode,
* crop selection, alerts & logs

**Description of the system:**

PlantWise continuously monitors soil moisture, ambient light, and air temperature/humidity, and automatically waters plants based on user‐selected thresholds. The STM32 node samples sensors every 500 ms (UI updates) and evaluates irrigation decisions hourly. It publishes JSON payloads via MQTT over ESP-01 to the Raspberry Pi broker, which relays data to the Android app. Users can switch between Automatic and Manual modes, select crop types (preloaded thresholds), and receive alerts within 1 s..

**Sensor Modules & Implementation**

## Module 1: Sensor Acquisition & Processing (STM32F407)

* SysTick\_Handler
  + ms / hr counters → s500\_flag, h\_flag
* ADC\_IRQHandler
  + m\_flag = 1 (soil ADC ready)
* I²C Read
  + fetch DHT11 data → dht\_flag ● do ConvertAndScale()
  + raw ADC & DHT readings → engineering units
* do mqtt\_publish("plant/sensors", JSON{soil,light,hum})

## Module 2: Irrigation Control & Local Logic (STM32F407)

* USART2\_IRQHandler
  + parseTopic(msg):
    - "plant/control/app" → manual\_flag=1
    - "plant/config" → threshold=payload; config\_flag=1

* do EvaluateWatering()
  + if (m\_raw < threshold) ∨ manual\_flag:
    - Pump\_ON(); Servo\_Sweep()
    - Fork-wait on {moisture≥threshold; manual\_stop; timeout(60 s)}
    - Pump\_OFF(); Servo\_OFF()

* do mqtt\_publish("plant/notification", “watering started|stopped”)

## Module 3: Broker & Automation Engine (Raspberry Pi)

● Mosquitto MQTT

● Python Service

* subscribe: plant/sensors, plant/notification, plant/control/app, plant/config
* do store incoming data/events in SQLite
* do on config update → publish to plant/config → await ack → publish plant/config/ack
* do in Automatic mode → evaluate latest soil vs. config → publish “start”/“stop” to plant/control/app
* do respond to app’s data-fetch requests by republishing latest sensor JSON

## Module 4: Mobile UI & Configuration (Android App)

● MQTT Client (Paho)

* subscribe: plant/sensors, plant/notification, plant/config/ack
* do Display real-time dashboard (soil, light, hum, pump status)
* do Crop-selection → mqtt\_publish("plant/config", {cropId})
* do Manual controls → mqtt\_publish("plant/control/app","start" / "stop")
* do History view → request via plant/sensors/request → display past readings
* do Surface alerts: low-light, threshold breach, watering events

**Task Structure for PlantWise (Bare-Metal STM32)**

**1. Main Routine: Sensor Data Acquisition**

* + This routine is implemented using timer-based polling or SysTick for scheduling, since no RTOS is used.
  + Soil Moisture Reading (every 5 seconds)
  + Light Intensity Reading (every 5 seconds)
  + Temperature & Humidity (DHT11) Reading (every 10 seconds)
  + Actuator Status Check (Relay, Servo positions)

**2. Communication with MQTT**

* + Executed every 1 minute:
  + Publish sensor values to Broker
  + Fetch actuator control thresholds (e.g., moisture threshold) or override commands from Firebase

**Application Requirements (Android)**

**Subscriptions:**

- plant/sensors/updates (live data)

- plant/sensors/status (periodic heartbeat)

- plant/config/ack (config acknowledgements)

**Displays:**

- Real-time soil moisture, light, temperature & humidity

- Pump ON/OFF indicator, current operating mode

**Controls:**

- Toggle Automatic ↔ Manual mode

- Manual Start / Stop watering buttons

- Crop-Selection Dropdown (e.g. “Tomato,” “Lettuce,” “Rose”) that applies preset

moisture thresholds under the hood







