grEEEn:
An IoT Platform for Indoor Plant Monitoring

WYZ-4
Castor, John Danielle T
Gonzales, Airick Miguel R
Sabater, Zylm M

Introduction

Project Objectives

- Develop an Internet of Things (IoT) platform to monitor environmental variables
- Collect and analyze data from the assigned sensors: AM2320, BH1750, and SGP30
- Transmit data to cloud for real-time processing and analysis
- Gain insights into the impacts of plant introduction to a classroom

Motivation

- **“A Design of IoT-based Monitoring System for Intelligence Indoor Micro-Climate Horticulture Farming in Indonesia”**

- Monitor soil, water, and air conditions in horticulture cultivation using IoT device

- IoT sensor board components: CO_2, light, air humidity and temperature, soil moisture and temperature sensors, camera, WLAN module

Scope

- Design, development, and implementation of an IoT plant monitoring system
- Utilization of assigned sensors to monitor environmental variables
- Investigate the effectiveness of the system in monitoring indoor environment

Limitations

- Study did not evaluate or monitor soil quality and nutrition
- Findings have limited generalizability to different indoor settings and plant varieties
- Long-term effects were not fully observed

Methodology

- **Specifications Definition**
  - STM32F411RE MCU
  - ESP8266 WiFi Module
  - AM2320 Temperature and Humidity Sensor
  - BH1750 Light Sensor
  - SGP30 Air Quality Sensor
  - PC201 PCB, headers

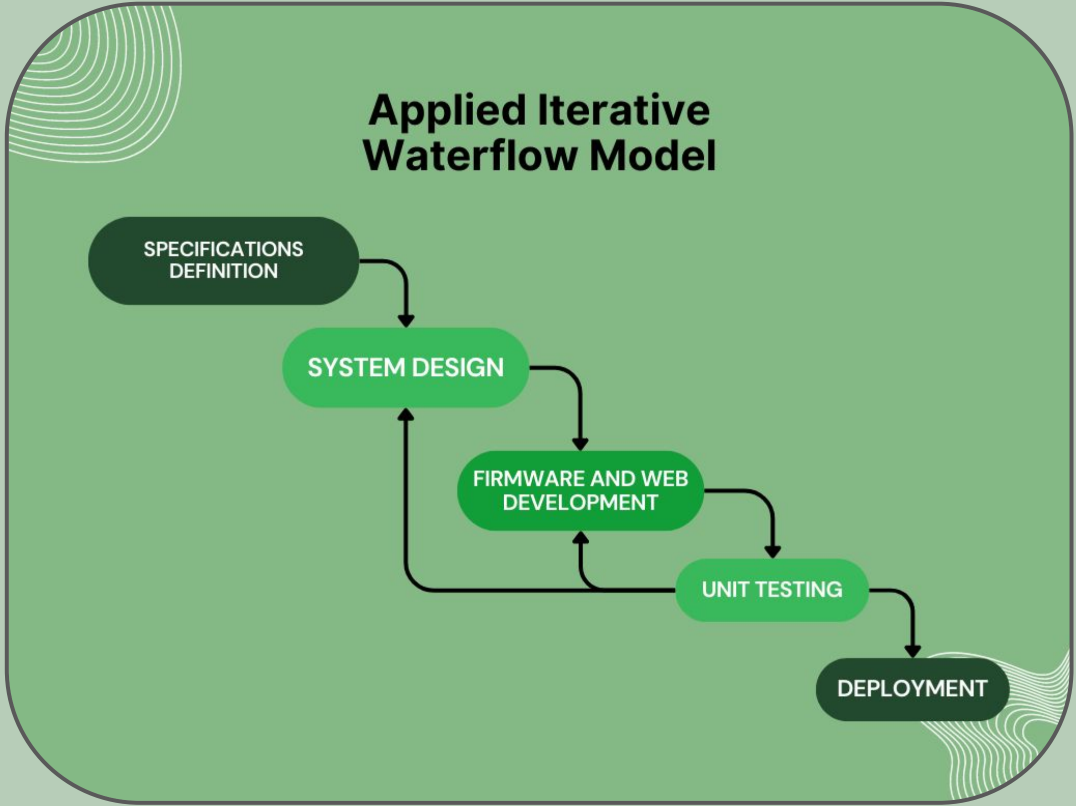
- Task- utilize the hardware to create an **IoT** platform for a UPD-EEEI Room (Room 321)

Methodology

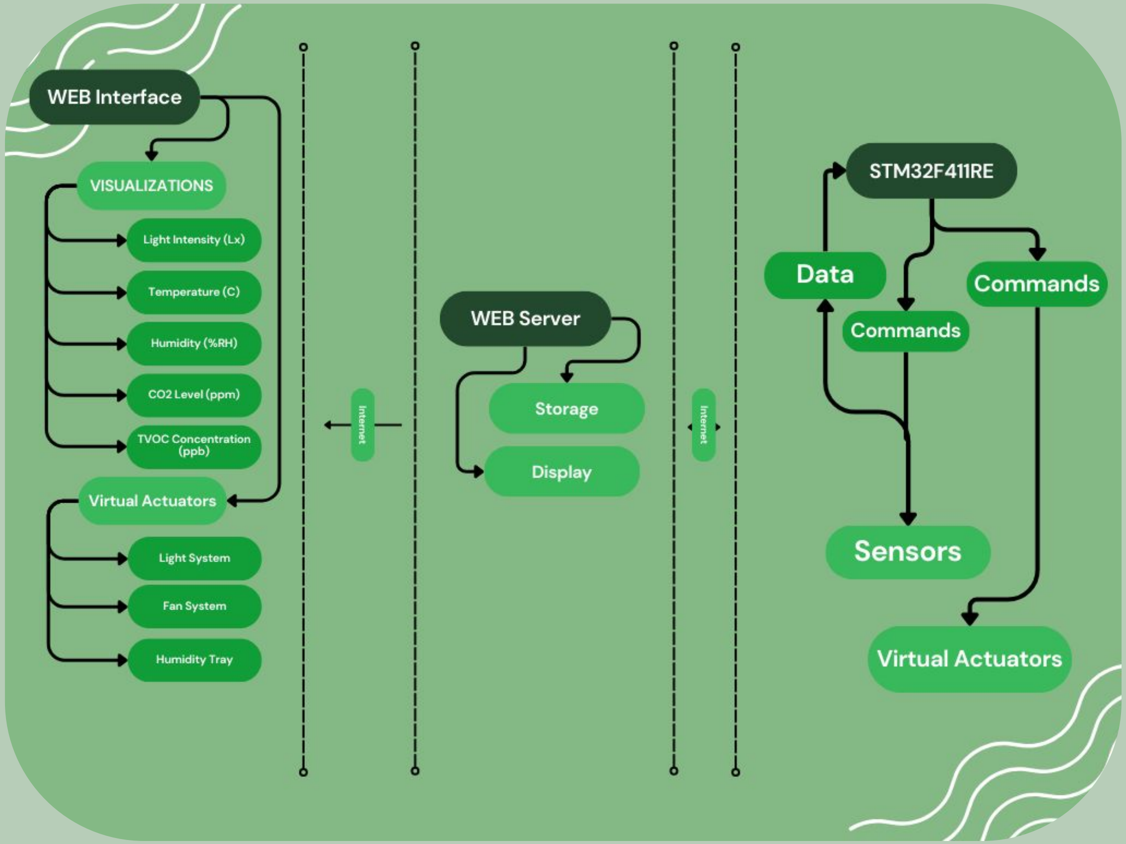
- **System Design and Planning**
  - IoT platform for **indoor plant monitoring**- sensors aligned with most plant requirements
    - Fixed soil quality (loam+garden+vermicast) and water frequency (every 1-2 weeks)

  - *Epipremnum aureum* (**Golden Pothos**)
    - Robust
    - Indoor effects

Methodology



Methodology: Overall System Software Design



Methodology: Daughterboard Development

- For **all sensors,**
  - Datasheet and manual reading
  - Paper Layouts- modified
  - Prototyping- breadboard
  - Drop and Continuity tests
  - Powered via power banks and charger adapters
    - ~5V/2A

Methodology: Daughterboard Development

- For **all sensors**,
    - **ESP8266**- reliable and integrated
        - 3.3V- VDD and Chip Enable
        - 115.2kHz baud rate
        - 2.4GHz
        - 802.11 b/g/n
        - **UART**- RX and TX wires
            - UART1
            - PA10 (RX) and PA9 (TX)

Methodology: Daughterboard Development

- **AM2320**- stable, fast and calibrated
  - -40 to 80 °C, 0.1°C resolution, <5s response
  - 0-99%RH, 0.1%RH resolution, <5s response
  - 5V
  - 4.7kΩ pull-up
  - **I2C**- SDA and SCL wires
    - I2C1- up to 50MHz
    - PB9 (SDA) and PB8 (SCL)

Methodology: Daughterboard Development

- **BH1750**- stable, fast and calibrated
    - 1-65535lx
    - 3.3V
    - 2.2kΩ pull-up
    - **I2C**- SDA and SCL wires
        - I2C1- up to 50MHz
        - PB9 (SDA) and PB8 (SCL)

Methodology: Daughterboard Development

- **SGP30**
  - TVOC: 0 ppb - 60000 ppb; 1, 6, 32 ppb resolution
  - CO2: 400 ppm - 60000 ppm; 1, 3, 9, 31 ppm resolution
  - 1.62-1.98V
  - **I2C**- SDA and SCL wires
    - I2C1- up to 50MHz
    - PB9 (SDA) and PB8 (SCL)

Methodology: Sensor-MCU Integration

- For **all sensors**,
    - **I2C**- Controllerstech repository
        - STM32F411RE as master
        - Sensors as slaves
        - **Configured** and **enabled** GPIO and I2C registers and clocks
        - Basic I2C **functions**-
            - Start and Stop conditions
            - Address Transmission
            - Master Transmission and Receiving
    - Bare-metal
    - Checked errors via **Debugging** and Live Expressions

Methodology: Sensor-MCU Integration

- **AM2320**- derived from Electrohobby repository
  - Configured 7-bit addressing mode
  - **Wake**- passive mode
    - START + 0xB8 (slave address) + 1ms + STOP
  - Send **Read Command**-
    - START + 0xB8 + 0x03 (function code [read register]) + 0x00 (start address) + 0x04 (register length) + STOP + 1ms
  - **Read** Sensor-
    - START + 0xB9 (0xB8 + 1 [R/read]) + <read sensor via buffers> + STOP

Methodology: Sensor-MCU Integration

- **AM2320-**
  - Data received consists 8 bytes
    - 3rd and 4th bytes- humidity reading in hex
    - 5th and 6th- temperature reading in hex
      - Converted them to **decimal** then **/10**
    - 7th and 8th- **CRC checksum**
      - Segregates correct information
      - Code from datasheet
  - Overall communication- <3s else dormant

Methodology: Sensor-MCU Integration

- **AM2320**-
  - **Virtual Actuators**- C **time** library
    - Desk fan
    - Ceiling fan- CW (circulate warm air) and CCW (push cool air)
    - Pebble Humidity Tray- drains (humidity too low)
    - Baking Soda Moisture Absorber Tray- opens (humidity too high)
  - **Calibration**-
    - Crowded room
    - Cloudy and cool outside
    - Spraying, fanning, blowing

Methodology: Sensor-MCU Integration

- **BH1750**- derived from BH1750FVI data sheet
  - **Wake**- passive mode
    - START + 0x23 (slave address) + 1ms + 0x01 (power on) + STOP
  - To begin data collection-
    - START + 0x23 + ack+ 0x10 (continuous high resolution mode) + STOP
  - **Read** Sensor-
    - START + 0x24 (0x23 + 1 [R/read]) + <read sensor via buffers> + STOP

Methodology: Sensor-MCU Integration

- **BH1750-**
  - Data received consists 2 bytes
    - Both are values of lux

Methodology: Sensor-MCU Integration

- **SGP30-**
  - The first 8 bits of the communication hold the I2C address, which is the slave address (0x58).
  - Following the address, there are 16 bits that can be either a memory address or a command.
  - If write operation, the MCU sends 16 bits of data (partitioned into two 8-bit parts) and a CRC.
  - If read operation, the MCU sends the slave address, and the sensor responds with a 16-bit data and CRC.

Methodology: Sensor-MCU Integration

- **SGP30-**
  - The "SGP30 - Write" function writes a 16-bit command to the sensor by initiating I2C communication, sending the slave address, and writing the upper and lower 8 bits of the command.
  - The "SGP30 - Read" function reads data from the sensor by initiating I2C communication, waiting for the start bit, sending the slave address, and reading data in a loop.
  - The "SGP30 - Init" function initializes the SGP30 sensor by calling "SGP30 - Write" with the command for "sgp30 iaq init" (0x2003).
  - The "SGP30 - Measure" function is invoked to initialize air quality measurement by using "SGP30 - Write" with the "sgp30 measure iaq" command (0x2008).

Methodology: Sensor-MCU Integration

- **SGP30-**
  - CO_2 and TVOC values are extracted from the read data by combining specific bytes.

Methodology: WiFi Module-MCU Integration

- **Similar** across all units- Controllerstech repository
  - HAL library
  - UARTRingBuffer.c library
    - UART methods
    - Ring buffer- reliable and efficient data handling, fixed data size
  - ESPDataLogger.c library
    - ESP8266 Initialization and Send methods
    - UARTRingBuffer.c module
    - AT commands

Methodology: WiFi Module-MCU Integration

- .ioc- GPIO, UART and clock (max) configuration
- Main function-
  - Initialization calls
  - ESP8266 Initialization
    - SSID
    - Password
- While(1)-
  - Read Data
  - Virtual actuators
  - Data readings to buffer, sent to **Thingspeak** via **API** and **TCP**
  - 15s delay (Thingspeak)

Methodology: WiFi Module-MCU Integration

- Auto-reconnect
  - Void SysTick_Handler(void)
  - HAL_GetTick();
  - Every 30-60s

Methodology: Web Interface Design

- **Simple monitoring** and indications, Golden Pothos requirements list
    - Environmental conditions
    - Actuator states

- Advantageous-
    - Internet
    - No installation
    - Data on cloud

Methodology: Web Interface Design

- **Development**- notepad
    - HTML
    - CSS
    - JavaScript

- Thingspeak accesses-
    - **Visualizations**- <iframe> provided
    - Latest readings- fetch() and data.feeds[0]

Methodology: Testing and Deployment

- **Preliminaries**-
  - 12-hour end-to-end communication test
    - Calibration
    - Relatively stable fiber-optic

- **Pretest**- control
  - Measure current room conditions for 1 day
  - **Setup**- merged armchairs and boxes, extension cord, tapes, units w/o BH1750

Methodology: Testing and Deployment

- **Posttest-**
  - Compared w/ pretest readings to assess improvements and recommendations
  - 1 day
  - Similar setup w/ pretest + BH1750 + Golden Pothos
  - Units nearby plants

PROTOTYPE DEMONSTRATION

grEEEn

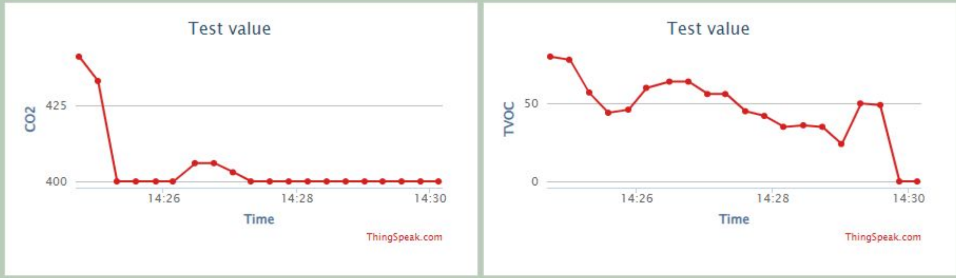
Results

Testing of Web Application

zylmmsabater.github.io/grEEEn/

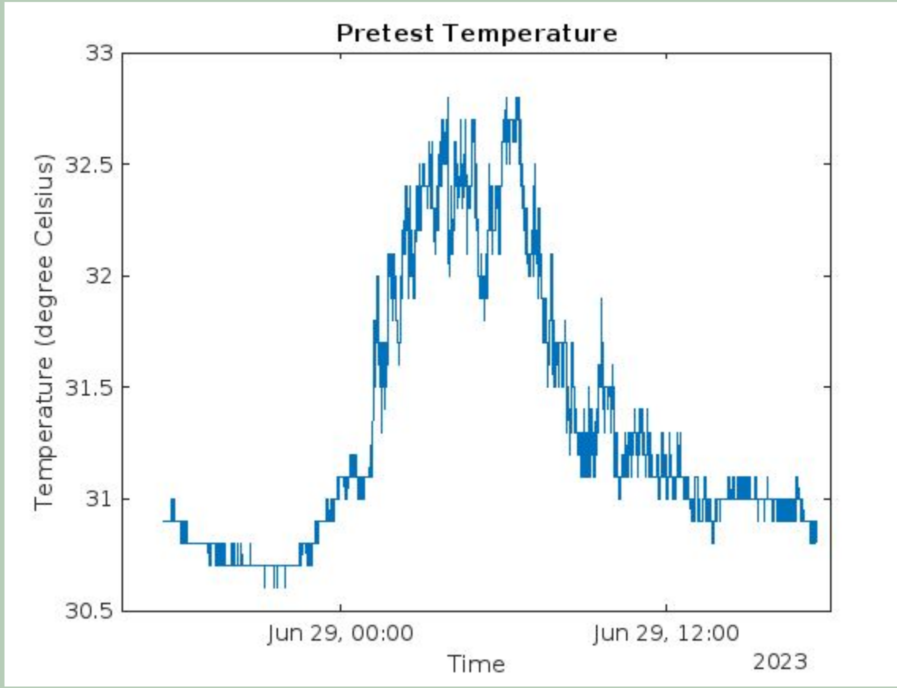
***Epipremnum aureum* (Golden Pothos) Requirements:**

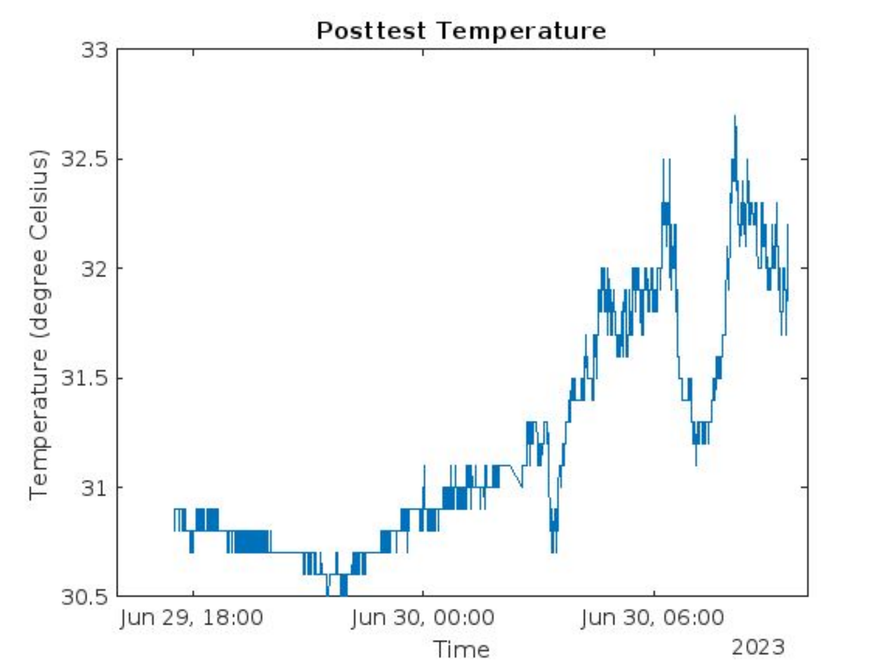
1. Light: >200ft-c (high-light areas), 75-200ft-c (med.-light), 25-75ft-c (low-light)
2. Temperature: 65degF (night), 75degF (day)
3. Relative Humidity: 25-49%
4. Water: when surface is dry (every 1-2 weeks)



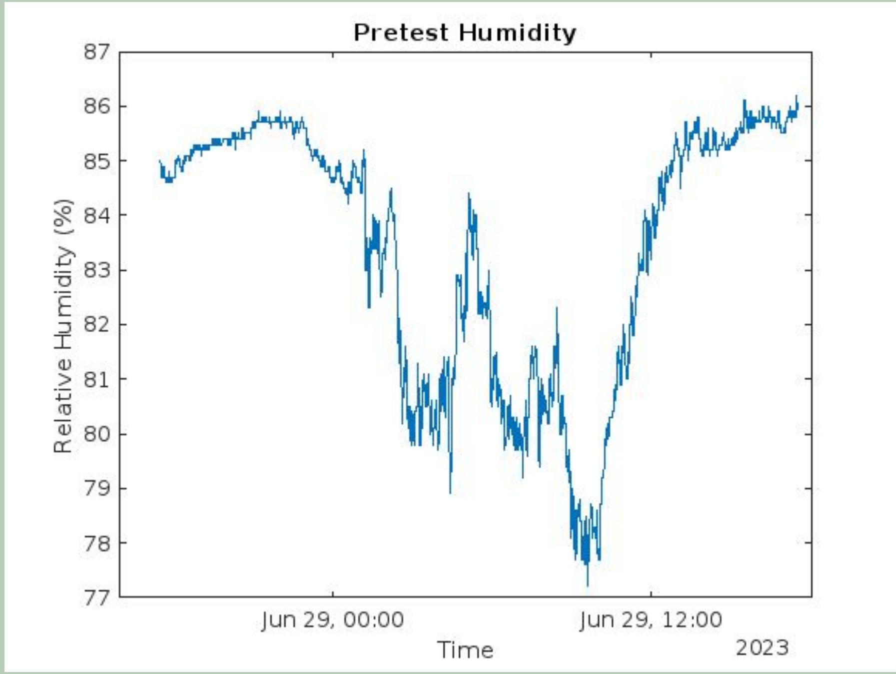
RESULTS

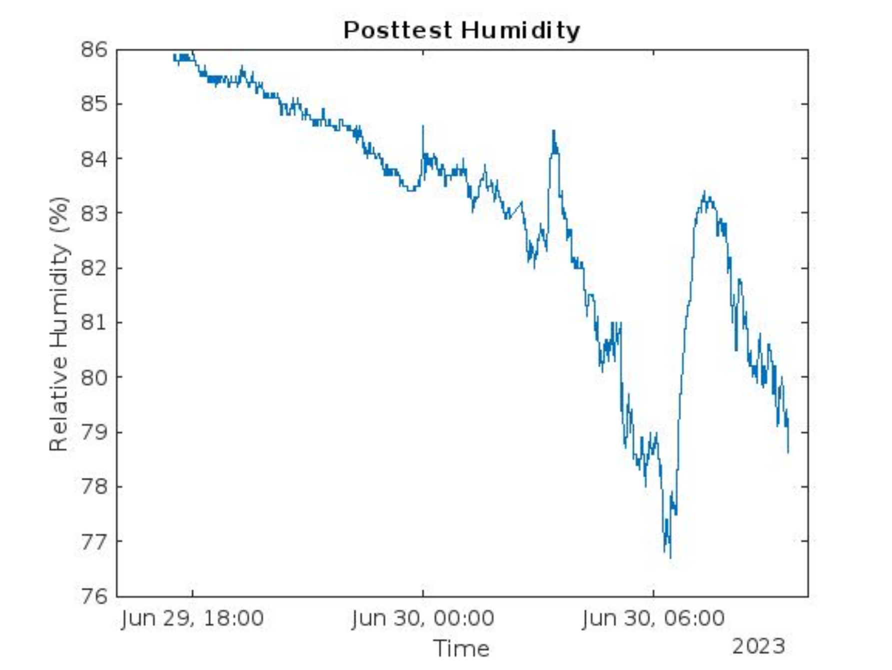
Temperature



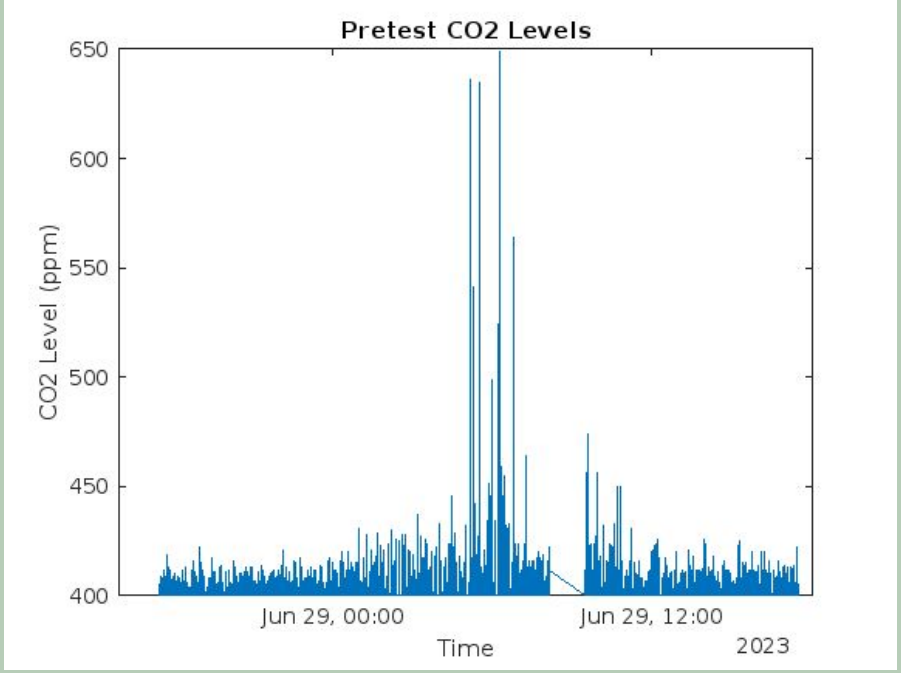


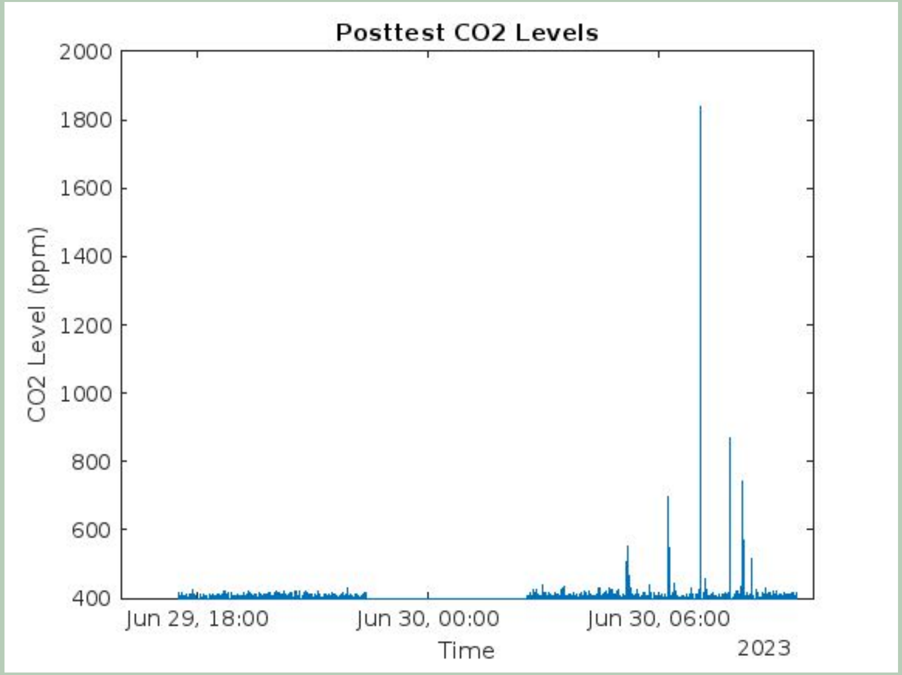
Relative Humidity



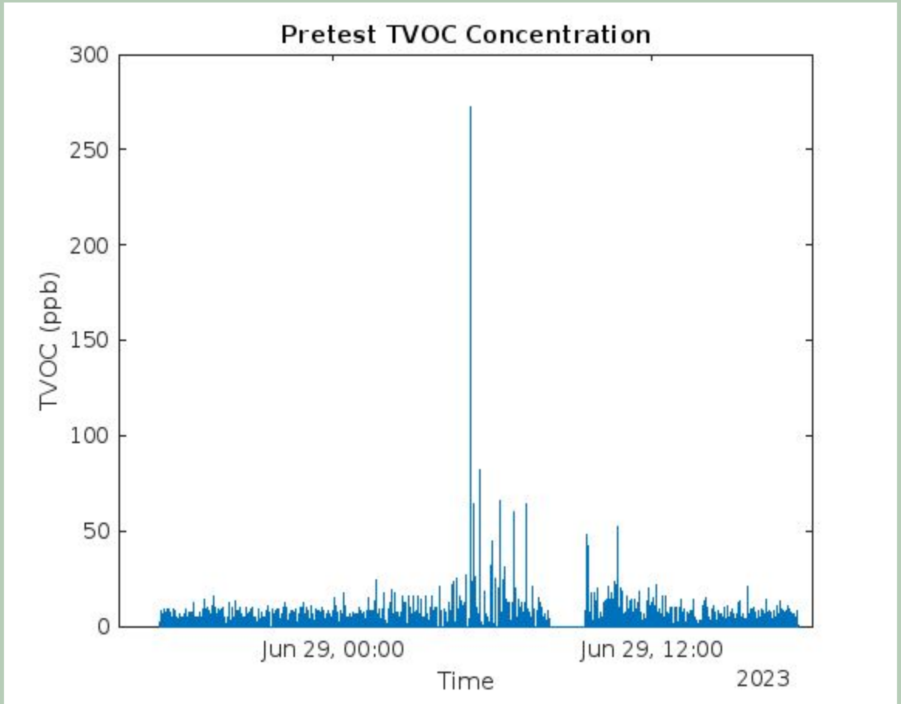


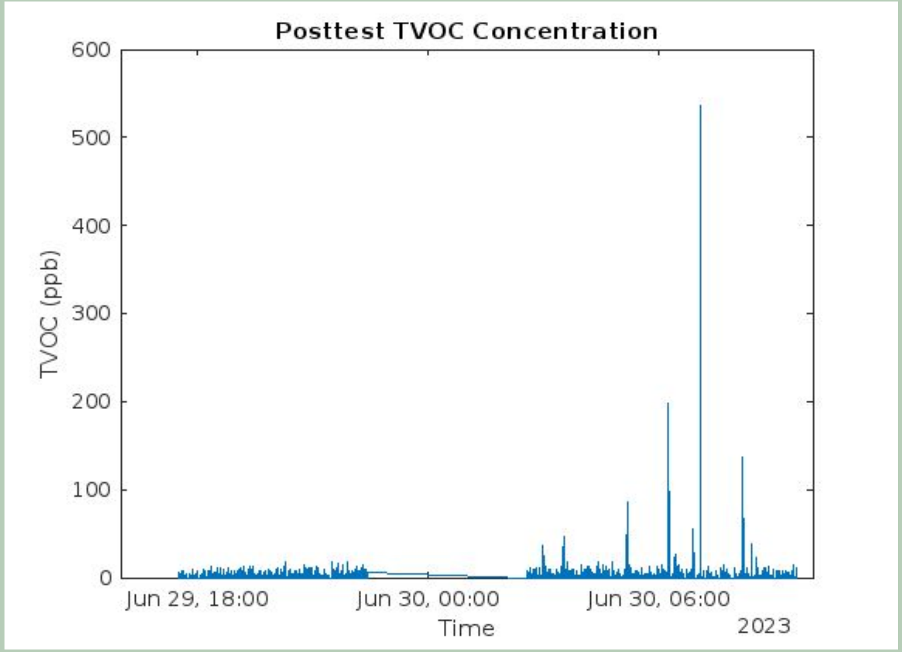
CO2 Level


Pretest CO2 Levels
650
600
550
500
450
400
CO2 Level (ppm)
Jun 29, 00:00
Jun 29, 12:00
Time
2023

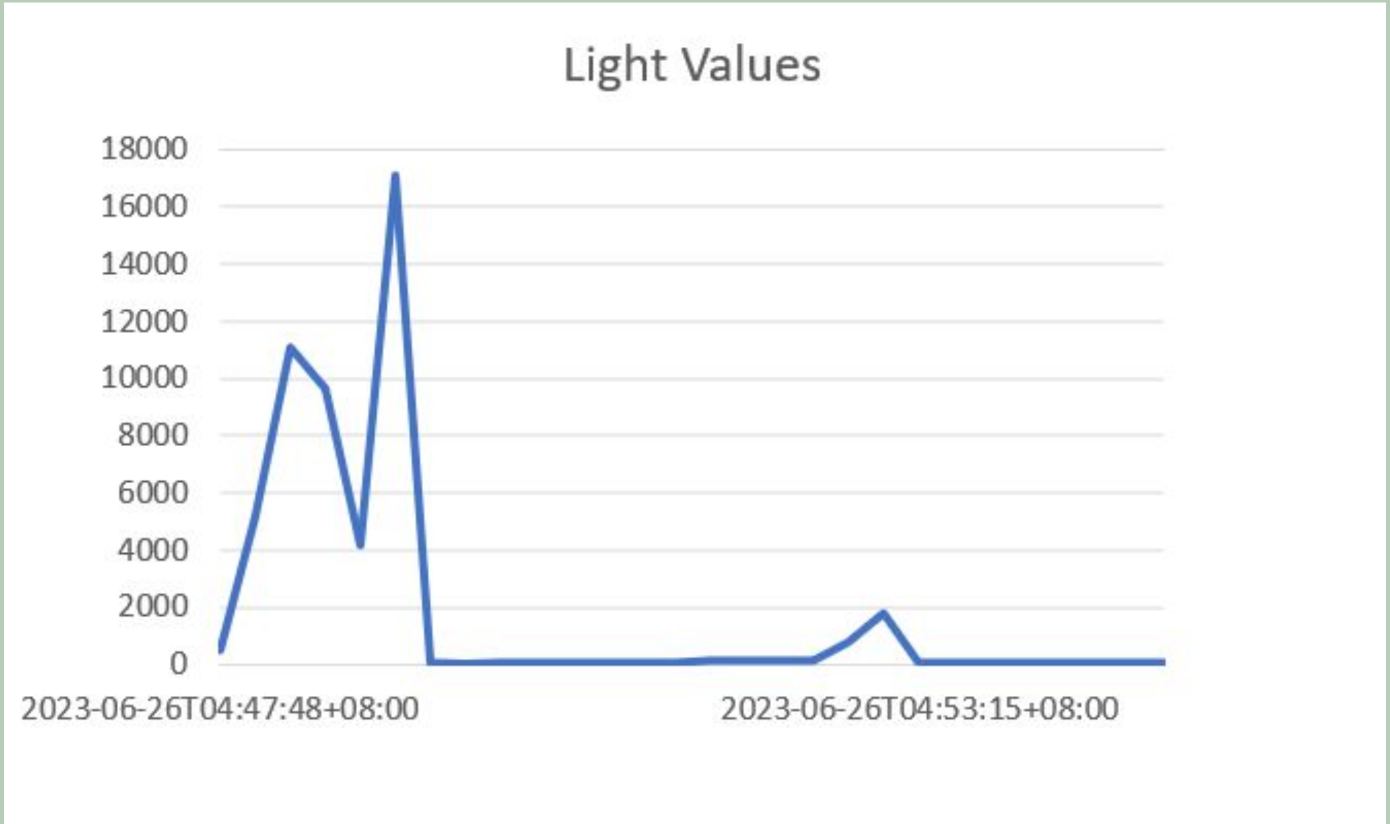

Posttest CO2 Levels
2000
1800
1600
1400
1200
1000
800
600
400
CO2 Level (ppm)
Jun 29, 18:00
Jun 30, 00:00
Jun 30, 06:00
Time
2023

TVOC Concentration

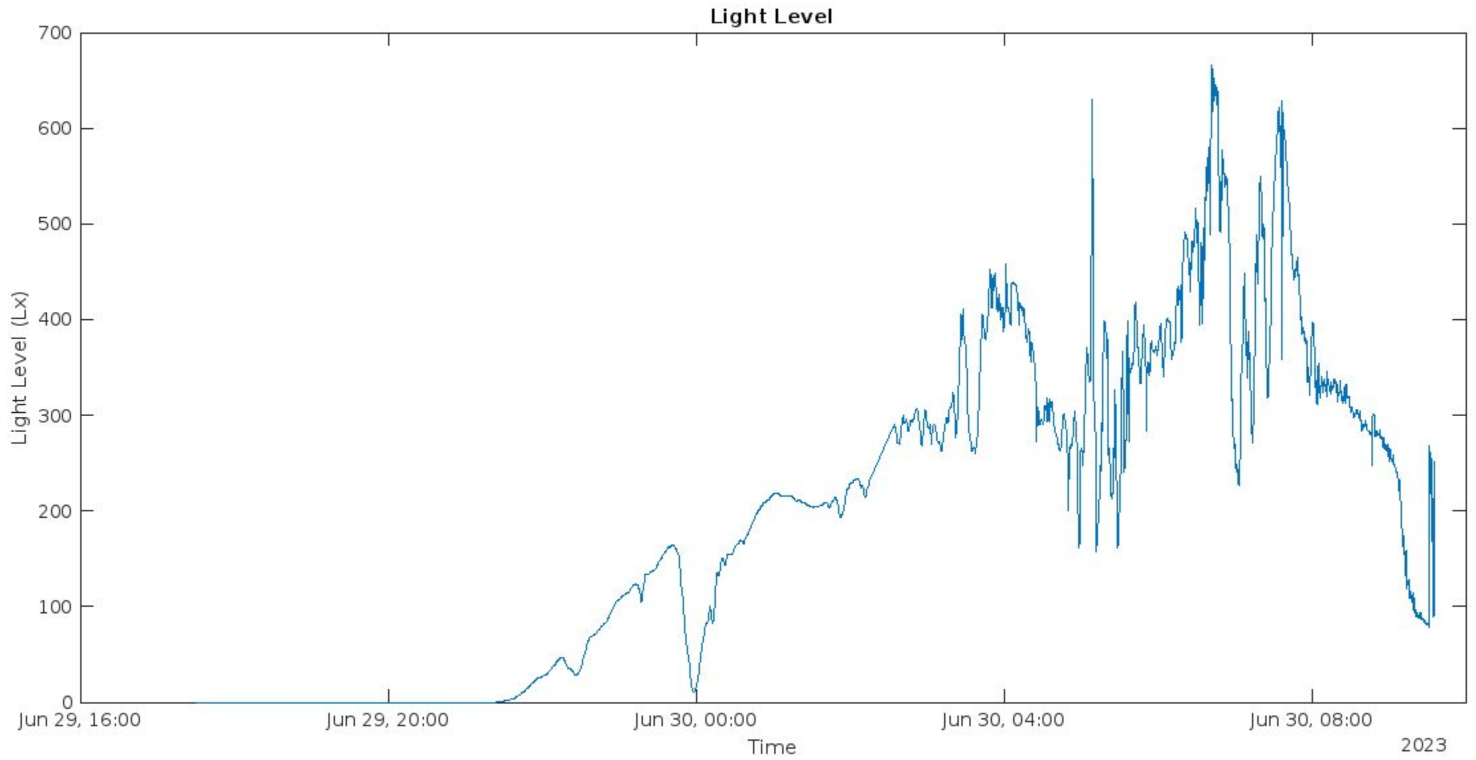




Light Level (Testing)



Light Level (Posttest)



Conclusion

- The group was able to successfully develop a reliable IoT indoor environment monitoring platform.

- Data was collected and analyzed from the assigned sensors: AM2320 and SGP30 for two days, and BH1750 for a day.

- There was no significant changes between a classroom setup with and without indoor plants.

- There were external factors that have affected the measurements.

Appendix: Final Setup

Q&A