IOT_PHASE 4: Development Part 2

SMART PUBLIC RESTROOM

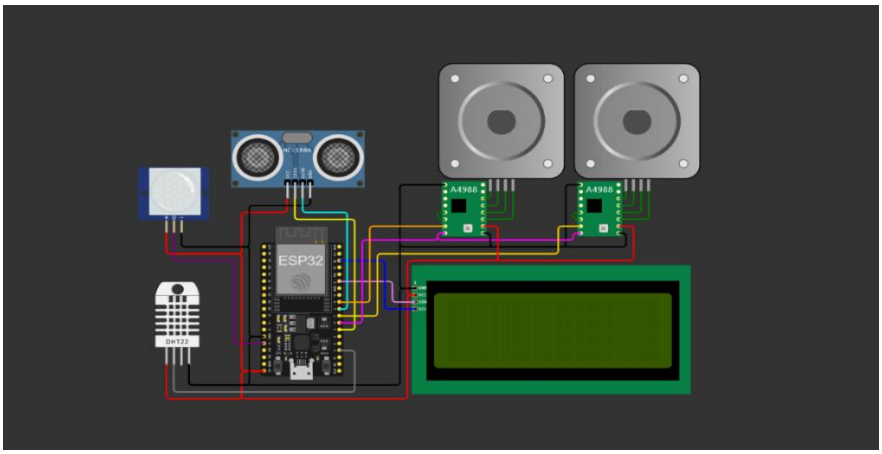
Hardware Components:

- ESP32 processor
- DHT22 temperature and humidity sensor
- Ultrasonic distance sensor
- Stepper motors (2)
- Liquid Crystal Display (LCD)
- Capacitive Sensor

Project Requirements:

- Monitor temperature and humidity in the restroom.
- Detect water levels in the restroom water tank.
- Automatically fill the water tank when it's low.
- Lower the room temperature if it exceeds a threshold.
- These requirements are essential for maintaining a comfortable and hygienic public restroom.
- To Detect Water Leakage in restroom.

Circuit diagram:



Program:

```
#include <Wire.h>
#include <Adafruit_Sensor.h>
#include <DHT.h>
#include <AccelStepper.h>
#include <NewPing.h>
#include <LiquidCrystal_I2C.h>
#include <WiFi.h>
#include <HTTPClient.h>

#define DHT_PIN 15     // GPIO connected to DHT22
#define DHT_TYPE DHT22
#define ULTRASONIC_TRIGGER 4 // GPIO connected to ultrasonic sensor trigger
#define ULTRASONIC_ECHO 5    // GPIO connected to ultrasonic sensor echo
#define MOTOR1_STEP 18
#define MOTOR1_DIR 16
#define MOTOR1_ENABLE 1
#define MOTOR2_STEP 17
#define MOTOR2_DIR 16
#define MOTOR2_ENABLE 2
#define IR_SENSOR_PIN 13    // GPIO connected to IR sensor
#define CAPACITIVE_SENSOR_PIN 14 // GPIO connected to the capacitive sensor
#define LCD_ADDRESS 39    // I2C address of the LCD
#define LCD_COLS 20
#define LCD_ROWS 6

DHT dht(DHT_PIN, DHT_TYPE);
AccelStepper stepper1(10000, MOTOR1_STEP, MOTOR1_DIR);
AccelStepper stepper2(1, MOTOR2_STEP, MOTOR2_DIR);
NewPing sonar(ULTRASONIC_TRIGGER, ULTRASONIC_ECHO);
```

```
LiquidCrystal_I2C lcd(LCD_ADDRESS, LCD_COLS, LCD_ROWS);

int count = 0;
int waterLeakage = 0; // Initialize the water leakage variable

const char* ssid = "Wokwi-GUEST";
const char* password = "";
const char* thingSpeakUrl = "https://api.thingspeak.com/update";
const String apiKey = "F9IE7GGUTH545VG5"; // Replace with your ThingSpeak API Key

void setup() {
  Serial.begin(9600);
  Serial.print("Connecting to WiFi");
  WiFi.begin(ssid, password);
  while (WiFi.status() != WL_CONNECTED) {
    delay(100);
    Serial.print(".");
  }
  Serial.println(" Connected!");

  Serial.begin(115200);
  lcd.init();
  lcd.backlight();
  dht.begin();
  pinMode(MOTOR1_ENABLE, OUTPUT);
  pinMode(MOTOR2_ENABLE, OUTPUT);
  pinMode(IR_SENSOR_PIN, INPUT); // IR sensor as input
  pinMode(CAPACITIVE_SENSOR_PIN, INPUT); // Capacitive sensor as input
  digitalWrite(MOTOR1_ENABLE, LOW);
  digitalWrite(MOTOR2_ENABLE, LOW);
```

```
  stepper1.setMaxSpeed(10000);
  stepper1.setSpeed(2000);
  stepper2.setMaxSpeed(1000);
  stepper2.setSpeed(200);
}

void loop() {
  float temperature = dht.readTemperature();
  float humidity = dht.readHumidity();

  unsigned int cm = sonar.ping_cm();

  if (cm < 170) {
    // Rotate motor 1 clockwise by a certain number of steps
    digitalWrite(MOTOR1_ENABLE, HIGH);  // Enable the motor
    digitalWrite(MOTOR1_DIR, HIGH);     // Set direction
    for (int i = 0; i <= 170; i++) {
      digitalWrite(MOTOR1_STEP, HIGH);
      delayMicroseconds(50);
      digitalWrite(MOTOR1_STEP, LOW);
      delayMicroseconds(50);
    }
    digitalWrite(MOTOR1_ENABLE, LOW);  // Disable the motor
  } else {
    digitalWrite(MOTOR1_ENABLE, LOW);  // Disable the motor
  }

  if (temperature > 25.0) {
    // Rotate motor 2 clockwise by a certain number of steps
    digitalWrite(MOTOR2_ENABLE, HIGH);  // Enable the motor
```

```
  digitalWrite(MOTOR2_DIR, HIGH);     // Set direction
  for (int i = 0; i < 200; i++) {
    digitalWrite(MOTOR2_STEP, HIGH);
    delayMicroseconds(50);
    digitalWrite(MOTOR2_STEP, LOW);
    delayMicroseconds(50);
  }
  digitalWrite(MOTOR2_ENABLE, LOW);  // Disable the motor
} else {
  digitalWrite(MOTOR2_ENABLE, LOW);  // Disable the motor
}

// IR sensor count detection
if (digitalRead(IR_SENSOR_PIN) == HIGH) {
  // Increment the count when IR sensor detects an object
  count++;
}

// Read the capacitive sensor value
waterLeakage = digitalRead(CAPACITIVE_SENSOR_PIN);

// Create a query string with all the data you want to send
String queryString = "api_key=" + apiKey +
                     "&field1=" + String(temperature) +
                     "&field2=" + String(humidity) +
                     "&field3=" + String(count) +
                     "&field4=" + String(waterLeakage) +
                     "&field5=" + String(cm); // Add ultrasonic distance as field5

// Send the data to ThingSpeak
```

```
  HTTPClient http;
  http.begin(thingSpeakUrl);
  http.addHeader("Content-Type", "application/x-www-form-urlencoded");
  int httpResponseCode = http.POST(queryString);

  if (httpResponseCode == 200) {
    Serial.println("Data sent to ThingSpeak successfully");
  } else {
    Serial.println("Failed to send data to ThingSpeak");
  }

  http.end();

  // Delay before the next iteration
  delay(1000);

  lcd.clear();
  lcd.setCursor(0, 0);
  lcd.print("Temp: " + String(temperature) + "C");
  lcd.setCursor(0, 1);
  lcd.print("Humidity: " + String(humidity) + "%");
  lcd.setCursor(0, 2);
  lcd.print("Count: " + String(count));
  lcd.setCursor(0, 3);
  lcd.print("Leakage: " + String(waterLeakage));
  lcd.setCursor(0, 4);
  lcd.print("Dis: " + String(cm) + "cm"); // Display ultrasonic distance
  delay(1000);
}
```

Website:

https://sites.google.com/view/smartrestrooms/home

1.Include Libraries:

- Several libraries are included at the beginning of the code to use specific sensors and components, as well as to enable WiFi and make HTTP requests.

2.Global Variables:

- The program defines various global constants and variables to store sensor pin configurations, sensor data, and WiFi credentials.

3.Setup Function:

- The setup function initializes the program. It performs the following tasks.
- Connects to a WiFi network using the specified SSID and password.
- Initializes the LCD display and DHT sensor.
- Sets up pins for motor control, IR sensor, and capacitive sensor.
- Configures and sets the maximum speed and speed for two stepper motors.
- **Note:** The code should have only one Serial.begin statement at a single baud rate, so it's recommended to remove the redundant Serial.begin(9600).

4.Loop Function:

- The loop function runs repeatedly and performs the following tasks:
- Reads temperature and humidity data from the DHT22 sensor.
- Measures the distance in centimeters using an ultrasonic sensor.
- Controls the rotation of two stepper motors based on the sensor values (distance and temperature).
- Detects objects using an IR sensor and increments the count.
- Reads a capacitive sensor to check for water leakage.
- Creates a query string with the sensor data and sends it to ThingSpeak.
- Displays sensor data, including temperature, humidity, count, and leakage, on an LCD screen.
- Displays the ultrasonic sensor's distance on the LCD.
- Waits for a delay of 1 second before the next iteration.

5.Sending Data to ThingSpeak:

- The program constructs a query string with the data in the format required by ThingSpeak, including the API key and individual field values. It then sends an HTTP POST request to the ThingSpeak server to update the data fields.

6.LCD Display:

- The LiquidCrystal_I2C library is used to control the LCD display. The sensor data (temperature, humidity, count, leakage, and distance) is displayed on the LCD screen for monitoring.

7.Retrieve Data from ThingSpeak:

- Use the ThingSpeak API to retrieve data from your channel.

8.Host Your Webpage:

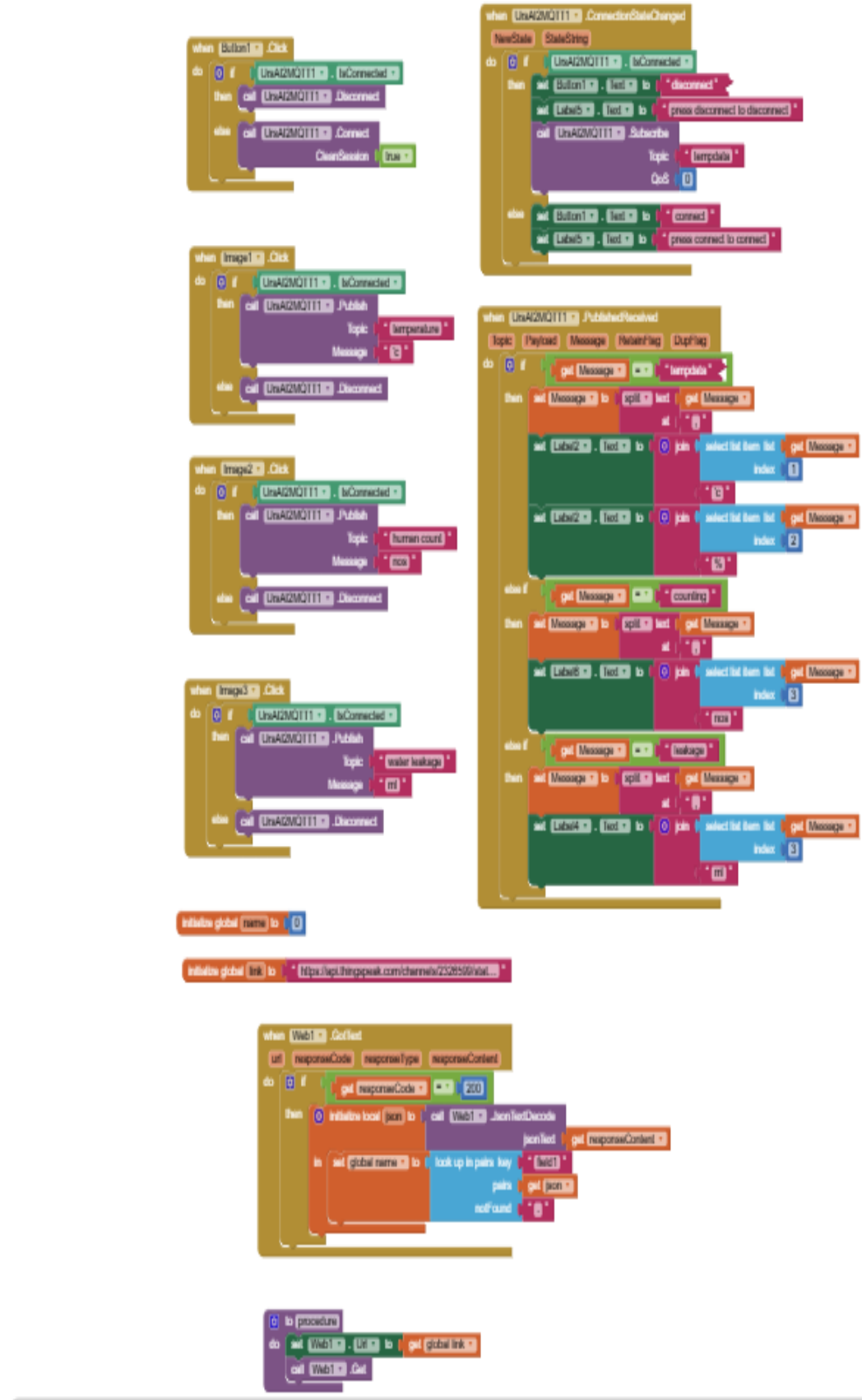
- Upload your HTML webpage to a web server or use a hosting service that supports HTML, CSS, and JavaScript files.

9.Access Your Webpage:

- Open a web browser and navigate to the URL of your hosted webpage to view the ThingSpeak data displayed.

Mobile application development:

Codeblock for mobile application:



Future Enhancements:

- Adding remote monitoring and control capabilities.
- Implementing water quality monitoring.
- Enhancing energy efficiency.

Conclusion:

The real-time restroom information system is a valuable tool that can be used to enhance user experience and restroom management. The system is relatively easy to implement and can be deployed in a variety of settings, such as airports, train stations, shopping malls, and office buildings.