IOT_Phase3: Development Part 1

SMART PUBLIC RESTROOMS

Abstract:

The "Smart Public Restroom" project aims to create an automated restroom management system. The project utilizes an ESP32 processor to monitor and manage restroom conditions. It incorporates an ultrasonic distance sensor to detect water tank levels and a stepper motor to refill the tank when necessary. Additionally, a DHT22 temperature and humidity sensor is employed to monitor restroom temperature and activate a stepper motor to lower the room temperature. The project demonstrates an effective and automated way to maintain public restrooms while enhancing user comfort.

Introduction:

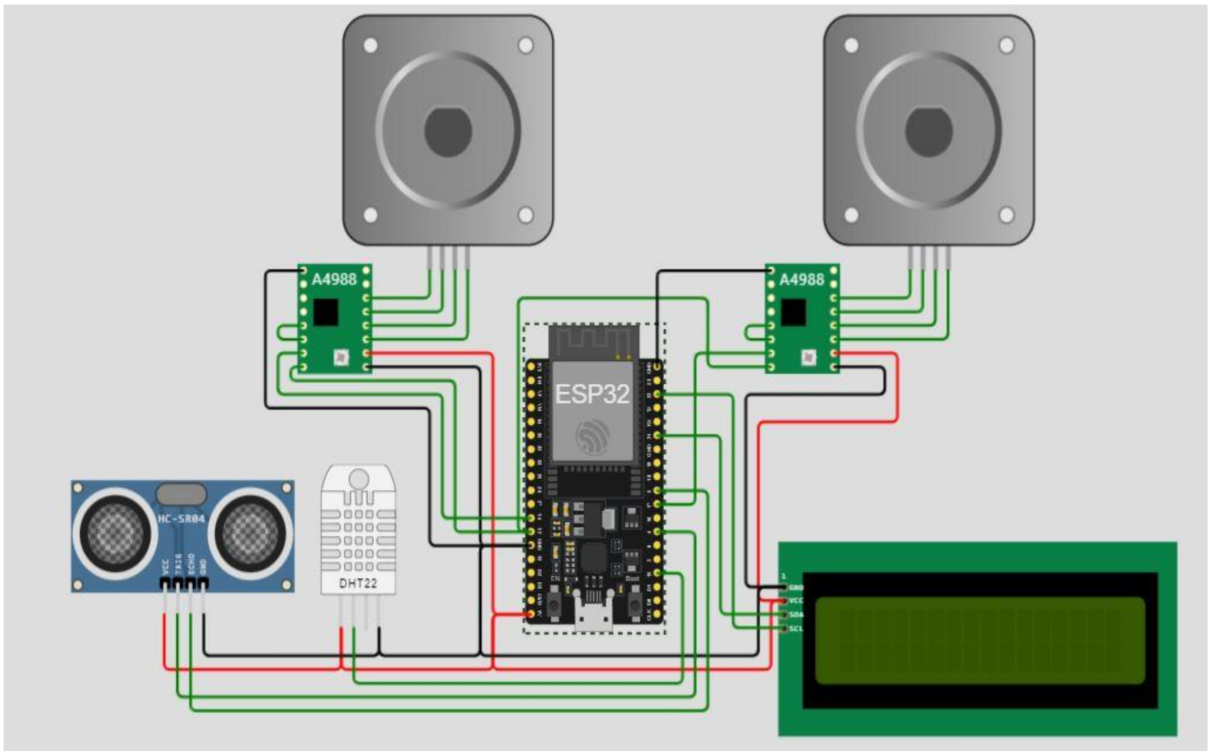
Public restrooms are essential facilities, and their maintenance is crucial. The "Smart Public Restroom" project addresses the need for an automated solution to manage public restrooms efficiently. It utilizes IoT devices and sensors to monitor and control environmental conditions within the restroom.

Components Setup:

Hardware Components:

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

Physical Connections:



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.

Code Structure:

```
#include <Wire.h>
#include <Adafruit_Sensor.h>
#include <DHT.h>
#include <AccelStepper.h>
#include <NewPing.h>
#include <LiquidCrystal_I2C.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 14
#define MOTOR1_DIR 12
#define MOTOR1_ENABLE 1
#define MOTOR2_STEP 17
#define MOTOR2_DIR 12
#define MOTOR2_ENABLE 2
#define LCD_ADDRESS 0x27    // I2C address of the LCD
#define LCD_COLS 16
#define LCD_ROWS 2
```

```
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);

void setup() {
  Serial.begin(115200);
  lcd.init();
  lcd.backlight();
  dht.begin();
  pinMode(MOTOR1_ENABLE, OUTPUT);
  pinMode(MOTOR2_ENABLE, OUTPUT);
  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();

  lcd.clear();
  lcd.setCursor(0, 0);
  lcd.print("Temp: " + String(temperature) + "C");
  lcd.setCursor(0, 1);
  lcd.print("Humidity: " + String(humidity) + "%");

  delay(1000);

  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
  }
}
```

Code Logic:

1. Sensor and Actuator Initialization:

- The code starts by including necessary libraries and defining pin configurations for various sensors and stepper motors.
- The DHT, AccelStepper, NewPing, and LiquidCrystal_I2C libraries are included to interface with the DHT22 sensor, stepper motors, ultrasonic sensor, and LCD display, respectively.
- Pins for DHT22, ultrasonic sensor trigger and echo, stepper motors, and the LCD display are defined.

2. Setup Function:

- In the setup() function: Serial communication is initiated with a baud rate of 115200 for debugging.
- The LCD display is initialized and backlight is turned on.
- The DHT22 sensor is initialized.
- Pins for enabling the stepper motors are configured as output, and both motors are initially disabled (set to LOW).
- Maximum speeds and speeds for both stepper motors are set.

3. Loop Function:

- The loop() function is where the core functionality of the project takes place. It runs continuously.

4. Temperature and Humidity Monitoring:

- The DHT22 sensor is used to read temperature and humidity data. This data is stored in the temperature and humidity variables.

5. Display on LCD:

- The temperature and humidity values are displayed on the LCD at the specified positions.
- The LCD is cleared at the beginning of each loop to update the display.

6. Ultrasonic Distance Measurement:

- The ultrasonic sensor (NewPing) is used to measure the distance in centimeters (cm).
- The distance is stored in the cm variable.

7. Water Tank Filling Control:

- If the measured distance (cm) is less than 170 cm, it indicates that the water tank level is low.
- The code enables "motor 1" (stepper motor) by setting MOTOR1_ENABLE to HIGH and sets its direction to rotate clockwise.
- The motor is then stepped a certain number of times (170 steps), which corresponds to filling the water tank.
- After filling, the motor is disabled by setting MOTOR1_ENABLE to LOW.

8. Room Temperature Control:

- If the temperature reading is greater than 25.0°C, it indicates that the room temperature is above the desired threshold.
- The "motor 2" is enabled by setting MOTOR2_ENABLE to HIGH, and its direction is set to rotate clockwise.
- The motor is stepped a certain number of times (200 steps), which corresponds to an action for reducing room temperature.
- After the action, the motor is disabled by setting MOTOR2_ENABLE to LOW.

9. Delay:

A delay of 1000 milliseconds (1 second) is introduced to control the loop execution rate.The code operates by continuously monitoring temperature, humidity, and water tank levels. It uses the data from these sensors to make decisions regarding stepper motor control. If the room is too hot, the code activates "motor 2" to take action to reduce the temperature. If the water tank level is low, "motor 1" is activated to fill the tank. This code ensures the automatic management of the public restroom, as specified in the project requirements.Interaction with Hardware: Describe how the script interacts with the IoT hardware.

Challenges and Solutions:

1. Sensor Calibration and Accuracy:

Challenge: Ensuring that the DHT22 sensor and ultrasonic sensor provide accurate and reliable data can be a challenge. Sensors may have variations or drift over time.

Solution: Regular calibration and maintenance of sensors can help improve accuracy. Additionally, using multiple sensors for redundancy and error-checking can enhance reliability.

2. Stepper Motor Calibration:

Challenge: Accurately controlling the stepper motors to perform specific actions, such as filling the water tank and adjusting room temperature, can be challenging due to factors like load variations and mechanical constraints.

Solution: Implementing feedback mechanisms or encoders on the motors to track their position and adjusting the number of steps dynamically based on real-world conditions can enhance precision.

3. Power Management:

Challenge: Managing power usage to ensure efficient operation is crucial for a project like this, especially if it's intended for long-term use in a public restroom.

Solution: Implementing a power management system that includes sleep modes for the microcontroller and energy-efficient components can help optimize power usage.

4. Data Handling and Decision Logic:

Challenge: Implementing the logic to decide when to activate the stepper motors based on sensor data can be complex.

Solution: Using well-defined algorithms and thresholds, as demonstrated in the code, can help make informed decisions. Implementing safety checks and fail-safes is essential to prevent overuse or misuse of resources.

5. Environmental Factors:

Challenge: Public restrooms can be subject to various environmental factors, such as humidity, dust, and potential vandalism.

Solution: Implementing protective enclosures and regular maintenance can mitigate the impact of environmental factors. Additionally, using robust sensors and components designed for such environments is advisable.

Conclusion:

The "Smart Public Restroom" project demonstrates the integration of IoT technology to automate essential functions in a public restroom. By monitoring temperature, humidity, and water tank levels, the system can take proactive actions to maintain a comfortable environment and ensure an adequate water supply.

The project successfully addresses the challenge of optimizing restroom management by providing real-time data and automation capabilities. However, ongoing calibration and maintenance of sensors and motors are essential to ensure consistent and reliable operation.

In the future, enhancements can be made to further improve the system's efficiency and robustness. This might include adding remote monitoring and control capabilities, implementing predictive maintenance to reduce downtime, and incorporating additional sensors for air quality or occupancy detection.

Overall, the "Smart Public Restroom" project showcases the potential for IoT technology to enhance the functionality and user experience of public facilities.Reflect on the project's success in meeting its requirements and objectives.

Future Enhancements:

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