SMART PUBLIC RESTROOM

Reg no :610821106062

**Hardware Components:**

1. **Arduino Board** e.g., Arduino Uno
2. **IoT Module** e.g., ESP8266
3. **Sensors** (PIR motion sensors, ultrasonic sensors, water flow sensors, air quality sensors, etc.)
4. **Actuators** (solenoid valves for water control, servos for doors, LED displays)
5. **Relays** (for controlling high-power devices)
6. **Feedback Mechanisms** (LED displays, buzzers, or LCD screens for user interaction)
7. **Power Supply** (batteries or external power sources)

ARDUINO WITH IoT PROGRAM :

#include <Arduino.h>

#include <WiFiClientSecure.h>

#include <ESP8266WiFi.h>

#include <Adafruit\_Sensor.h>

#include <Adafruit\_DHT.h>

// Define your Wi-Fi credentials and server details

const char\* ssid = "YourWiFiSSID";

const char\* password = "YourWiFiPassword";

const char\* server = "your-iot-server.com";

const int port = 443;

// Create a client instance

WiFiClientSecure client;

// Define sensor pins

const int motionSensorPin = 2; // PIR motion sensor pin

const int doorControlPin = 3; // Door servo control pin

const int waterValvePin = 4; // Solenoid valve control pin

const int feedbackLED = 5; // LED display or feedback indicator

// Define sensor variables

bool isRestroomOccupied = false;

// Door servo control variables

const int doorClosedPosition = 0;

const int doorOpenPosition = 90;

// Initialize the DHT sensor

#define DHTPIN 6

#define DHTTYPE DHT22

DHT dht(DHTPIN, DHTTYPE);

void setup() {

// Initialize sensors and actuators

pinMode(motionSensorPin, INPUT);

pinMode(doorControlPin, OUTPUT);

pinMode(waterValvePin, OUTPUT);

pinMode(feedbackLED, OUTPUT);

// Connect to Wi-Fi

WiFi.begin(ssid, password);

while (WiFi.status() != WL\_CONNECTED) {

delay(1000);

}

// Start DHT sensor

dht.begin();

// Connect to the IoT server

if (!client.connect(server, port)) {

// Handle connection error

Serial.println("Connection failed");

}

}

void loop() {

// Check occupancy status

int motionSensorValue = digitalRead(motionSensorPin);

if (motionSensorValue == HIGH) {

isRestroomOccupied = true;

digitalWrite(feedbackLED, HIGH);

} else {

isRestroomOccupied = false;

digitalWrite(feedbackLED, LOW);

}

// Control the door

if (isRestroomOccupied) {

openDoor();

} else {

closeDoor();

}

// Read temperature and humidity

float temperature = dht.readTemperature();

float humidity = dht.readHumidity();

// Send data to the IoT server

sendSensorData(temperature, humidity);

// Other actions and controls can be added here

// e.g., water flow control, air quality monitoring, etc.

delay(1000); // Add delay to avoid rapid sensor readings

}

void openDoor() {

// Control the door servo to open the restroom door

// You'll need to adjust this based on your servo and mechanism

// You may also want to implement safety checks

// (e.g., avoid opening when someone is near the door)

// and set a time limit for the door to remain open.

// Here, we assume a simple door servo control.

digitalWrite(doorControlPin, doorOpenPosition);

}

void closeDoor() {

// Control the door servo to close the restroom door

digitalWrite(doorControlPin, doorClosedPosition);

}

void sendSensorData(float temperature, float humidity) {

// Create a JSON payload with sensor data

String payload = "{\"temperature\":" + String(temperature) +

",\"humidity\":" + String(humidity) +

",\"occupied\":" + String(isRestroomOccupied) + "}";

// Send data to the IoT server

client.print(String("POST /your-api-endpoint HTTP/1.1\r\n") +

"Host: " + server + "\r\n" +

"Content-Type: application/json\r\n" +

"Content-Length: " + payload.length() + "\r\n" +

"\r\n" + payload);

// Wait for server response (optional)

while (client.connected()) {

if (client.available()) {

char c = client.read();

// Process server response if needed

}

}

// Disconnect from the server

client.stop();

}