

```

/*
 * File: ActuatorModule.ino
 * Author: Varad Chaskar
 * Description: Code for Actuator Module using ESP8266 and ThingSpeak.
 * Date: January 28, 2024
 */

#include <WiFiManager.h> // Include the WiFiManager library for easy WiFi configuration
#include <ESP8266WiFi.h> // Include the ESP8266WiFi library for WiFi functionality
#include "ThingSpeak.h" // Include the ThingSpeak library for communication with ThingSpeak cloud

WiFiClient client; // Create a WiFi client object for connecting to the internet

unsigned long myChannelNumber = 2384399; // Channel number taken from cloud
const char *myWriteAPIKey = "ELW2NF5Q83OGB39G"; // Write API key of cloud
const char *myCounterReadAPIKey = "3D8NH4JCI0EDYMIU"; // Read API key from cloud

bool LED = false; // Variable to track the state of LED
bool LEDValue = false; // Variable to track the state of LEDValue

void setup() {
  pinMode(LED_BUILTIN, OUTPUT); // Set the built-in LED pin as an output
  pinMode(0, OUTPUT); // Set pin 0 as an output
  pinMode(D7, OUTPUT); // Set pin D7 as an output
  pinMode(D8, OUTPUT); // Set pin D8 as an output
  pinMode(4, OUTPUT); // Set pin 4 as an output
  pinMode(5, OUTPUT); // Set pin 5 as an output

  Serial.begin(115200); // Start serial communication with baud rate 115200

  WiFiManager wm; // Create a WiFiManager instance
  bool res;
  res = wm.autoConnect("ASAA"); // Attempt to connect to WiFi with SSID "ASAA"
  if (!res) {
    // Failed to connect to WiFi
    Serial.println("Failed to connect");
    digitalWrite(LED_BUILTIN, HIGH); // Turn on the built-in LED
    delay(1000);
    digitalWrite(LED_BUILTIN, LOW); // Turn off the built-in LED
    delay(1000);
    digitalWrite(LED_BUILTIN, HIGH); // Turn on the built-in LED
    ESP.restart(); // Restart the ESP8266
  } else {
    // Successfully connected to WiFi
    ThingSpeak.begin(client); // Initialize ThingSpeak with the WiFi client
    digitalWrite(LED_BUILTIN, LOW); // Turn off the built-in LED
    Serial.println("Connected... Yeey :)"); // Print message to serial monitor
  }
}

void loop() {
  digitalWrite(LED_BUILTIN, LOW); // Turn off the built-in LED

  int A = ThingSpeak.readLongField(myChannelNumber, 5, myCounterReadAPIKey); // Read data from ThingSpeak channel field 5

  analogWrite(5, A); // Set the analog output on pin 5 based on the read value

```

```

int B = ThingSpeak.readLongField(myChannelNumber, 6, myCounterReadAPIKey); // Read data from ThingSpeak channel field 6

analogWrite(4, B); // Set the analog output on pin 4 based on the read value

int C = ThingSpeak.readLongField(myChannelNumber, 7, myCounterReadAPIKey); // Read data from ThingSpeak channel field 7

analogWrite(0, C); // Set the analog output on pin 0 based on the read value

int D = ThingSpeak.readLongField(myChannelNumber, 8, myCounterReadAPIKey); // Read data from ThingSpeak channel field 8

if (D == 100 && LED == false) {
    // Toggle LED and indicate with built-in LED
    digitalWrite(D7, HIGH);
    delay(1000);
    digitalWrite(D7, LOW);
    LED = true;
    digitalWrite(LED_BUILTIN, HIGH);
    delay(500);
    digitalWrite(LED_BUILTIN, LOW);
}

if (D == 0 && LED == true) {
    // Toggle LED and indicate with built-in LED
    digitalWrite(D7, HIGH);
    delay(1000);
    digitalWrite(D7, LOW);
    digitalWrite(LED_BUILTIN, HIGH);
    delay(500);
    digitalWrite(LED_BUILTIN, LOW);
    LED = false;
}

if (D == 200 && LEDValue == false) {
    // Toggle LEDValue and indicate with built-in LED
    digitalWrite(D8, HIGH);
    delay(1000);
    digitalWrite(D8, LOW);
    LEDValue = true;
    digitalWrite(LED_BUILTIN, HIGH);
    delay(500);
    digitalWrite(LED_BUILTIN, LOW);
}

if (D == 300 && LEDValue == true) {
    // Toggle LEDValue and indicate with built-in LED
    digitalWrite(D8, HIGH);
    delay(1000);
    digitalWrite(D8, LOW);
    LEDValue = false;
    digitalWrite(LED_BUILTIN, HIGH);
    delay(500);
    digitalWrite(LED_BUILTIN, LOW);
}
}

```