#include <WiFi.h>

#include <WiFiClient.h>

#include <Adafruit\_MPU6050.h>

#include <Adafruit\_Sensor.h>

#include <Wire.h>

#include "FlexLibrary.h"

// WiFi Settings

const char\* ssid = "Amna";

const char\* password = "123456789";

const char\* serverIP = "192.168.65.219";  // Your PC/Server IP

const int serverPort = 3333;

WiFiClient client;

Flex flex[5] = {Flex(32), Flex(35), Flex(34), Flex(39), Flex(36)};

Adafruit\_MPU6050 mpu;

const char\* HAND\_ID = "LH\_";

// Calibration Offsets for Gyro & Accel

float gyroOffsets[4][3] = {

    {-0.57, 0.94, 1.20}, {-0.54, 0.91, 1.17}, {-0.58, 0.99, 1.17}, {-0.60, 0.93, 1.16}

};

float accelOffsets[4][3] = {

    {0.06, -0.16, 0.97}, {0.06, -0.15, 0.97}, {0.06, -0.15, 0.97}, {0.05, -0.15, 0.97}

};

// Flex Sensor Calibration

int sensorMins[5] = {816, 585, 653, 662, 805};

int sensorMaxs[5] = {1374, 1296, 1392, 1383, 1350};

void setup() {

    Serial.begin(115200);

    // Connect to WiFi

    WiFi.mode(WIFI\_STA);

    WiFi.begin(ssid, password);

    Serial.print("Connecting to WiFi");

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

        Serial.print(".");

        delay(1000);

    }

    Serial.println("\nConnected to WiFi!");

    Serial.print("ESP32 IP Address: ");

    Serial.println(WiFi.localIP());

    // Connect to TCP Server

    Serial.print("Connecting to server...");

    while (!client.connect(serverIP, serverPort)) {

        Serial.print(".");

        delay(1000);

    }

    Serial.println("\nConnected to Server!");

    // Initialize MPU6050

    if (!mpu.begin()) {

        Serial.println("Failed to find MPU6050!");

        while (1) delay(10);

    }

    Serial.println("MPU6050 Initialized!");

}

void loop() {

    sensors\_event\_t a, g, temp;

    mpu.getEvent(&a, &g, &temp);

    // Apply Calibration

    float gyroX = g.gyro.x - gyroOffsets[0][0];

    float gyroY = g.gyro.y - gyroOffsets[0][1];

    float gyroZ = g.gyro.z - gyroOffsets[0][2];

    float accelX = a.acceleration.x - accelOffsets[0][0];

    float accelY = a.acceleration.y - accelOffsets[0][1];

    float accelZ = a.acceleration.z - accelOffsets[0][2];

    // Read and Normalize Flex Sensor Data

    int flexValues[5];

    for (int i = 0; i < 5; i++) {

        flex[i].updateVal();

        int rawValue = flex[i].getSensorValue();

        flexValues[i] = map(rawValue, sensorMins[i], sensorMaxs[i], 0, 255);

        flexValues[i] = constrain(flexValues[i], 0, 255);

    }

    // Create Data Message

    String message = String(HAND\_ID) + "Flex:" + String(flexValues[0]) + "," +

                     String(flexValues[1]) + "," + String(flexValues[2]) + "," +

                     String(flexValues[3]) + "," + String(flexValues[4]) +

                     " | " + String(HAND\_ID) + "Gyro:" + String(gyroX) + "," +

                     String(gyroY) + "," + String(gyroZ) +

                     " | " + String(HAND\_ID) + "Accel:" + String(accelX) + "," +

                     String(accelY) + "," + String(accelZ) + "\n";

    // Send Data via TCP

    if (client.connected()) {

        client.print(message);

        Serial.println("Data sent to PC: " + message);

    } else {

        Serial.println("Lost connection to server. Reconnecting...");

        while (!client.connect(serverIP, serverPort)) {

            Serial.print(".");

            delay(1000);

        }

        Serial.println("\nReconnected!");

    }

    delay(150);

}