

I

Example of OpenAI's, ChatGPT assist; on this project.

William Lucid

It seems like the issue is that the modified code is not continuously streaming the URLs. Currently, the code sends a single HTTP request to the WLED IP address when the touch sensor is touched and releases the touch sensor only after the HTTP request has been made.

To continuously stream the URLs, you can move the code that sends the HTTP request and waits for the touch sensor to be released outside of the if (isRGBRunning) block. This way, the code will continuously loop through the loop() function and send HTTP requests based on the values obtained from the MPU6050 sensor.

Here's an example of what the modified code could look like:

scss

Copy code

```
void loop() {  
    // read the touch sensor value  
    int touchValue = touchRead(TOUCH_PIN);  
  
    // if the touch sensor is being touched and is not already touched  
    if (touchValue < 50 && !isTouched) {  
        isToggled = !isToggled; // toggle the state  
        isTouched = true; // mark the touch state as touched  
        isRGBRunning = !isRGBRunning; // toggle the RGB effect state  
  
        // update the LED based on the toggle state  
        digitalWrite(LED_PIN, isToggled ? HIGH : LOW);  
    } else if (touchValue >= 50 && !isTouched) {  
        isTouched = false; // mark the touch state as not touched
```

```

    ledsON();
    noMotion(); //Random generater effects and color palettes
}

if (isRGBRunning) {
    // Get effects, intensity, and color palette from Accelerometer/Gyroscope (MPU6050)
    sensors_event_t accel;
    sensors_event_t gyro;
    sensors_event_t temp;

    mpu_temp->getEvent(&temp);
    mpu_accel->getEvent(&accel);
    mpu_gyro->getEvent(&gyro);

    //float scaled = 0.07782219916379; // Scaling factor 255/32767
    float scaled = 0.299782219916379; // Scaling factor to get closer to goal ranges

    float scaled_x = accel.acceleration.x * scaled * 12000; // 25000, 25000, and 6000 act as "feedback
gain."
    float scaled_y = accel.acceleration.y * scaled * 9000;
    float scaled_z = accel.acceleration.z * scaled * 1600;

    //Effect
    int effects = map(scaled_x, 0, 32767, 0, 255);

    //Intensity
    int intensity = map(scaled_y, 0, 32767, 0, 255);

    //Color Palette

```

```

int colors = map(scaled_z, 0, 32767, 0, 255);

//Modify Effect, Intensity, and Color Palette from MPU6050
String url = WLED_IP_ADDRESS + "/win"
    + "&A=" + String(128)
    + "&CL=" + "hFFA000"
    + "&C2=" + "h000000"
    + "&FX=" + abs(effects)
    + "&SX=" + String(128)
    + "&IX=" + abs(intensity)
    + "&C1=" + String(128)
    + "&C2=" + String(128)
    + "&C3=" + String(128)
    + "&FP=" + abs(colors);

```

```

HTTPClient http;
http.begin(url);
int httpCode = http.GET();
if (httpCode > 0) {
    Serial.println(httpCode);
}

```

Is that complete?

Based on the code you provided, the `isRGBRunning` variable is being toggled when the touch sensor is touched. When `isRGBRunning` is true, the code reads values from the MPU6050 sensor and maps those values to effect, intensity, and color palette values. It then constructs a URL with those values and sends an HTTP GET request to the specified WLED IP address.

The code then waits for the touch sensor to be released before looping back and checking the touch sensor value again. If the touch sensor is no longer being touched, the `ledsON()` and `noMotion()` functions are called, which randomly generate effect and color palette values.

It's worth noting that the code snippet you provided doesn't include the `ledsON()`, `ledsOFF()`, or `noMotion()` functions, so it's difficult to say exactly what those functions are doing. However, based on their names, it seems likely that `ledsON()` turns on the LED strip, `ledsOFF()` turns off the LED strip, and `noMotion()` generates random effect and color palette values when there is no motion detected by the MPU6050 sensor.

Sums it up well; thanks.

You're welcome! Let me know if you have any other questions or if there's anything else I can help you with.