

# Leafi – Air quality, with attitude.

Leafi is a leaf... but smarter. by @htzforge

**Leafi** is a leaf-shaped air quality sensor designed to raise awareness about indoor pollution in a fun and educational way. Powered by an MQ135 sensor and a WS2812B LED strip, Leafi visualizes air quality in real time by transitioning **from green to red**—green when the air is clean, red when it's polluted. It also features multiple lighting modes, including ambient animations and a relaxing "lounge mode", with **adjustable brightness** to match any atmosphere. Fully open-source and 3D-printed, Leafi is eco-friendly, customizable, and perfect for makers, educators, and anyone who wants to make the **invisible visible**.



Leafi: RGB vibes for eco-nerds 🌈🌿

# Leafi – Quick Start Guide


## Power

 Plug Leafi into a 5V USB-C power source.


## Modes

Leafi features 4 lighting modes:

- MEASURE – Color changes based on air quality
- LOUNGE – Ambient color animation
- SPOT – Static light for focused use
- WAVE – Slow, flowing color transitions

 Hold the button to switch between modes.

## Brightness

 Short press the button to cycle through brightness levels.

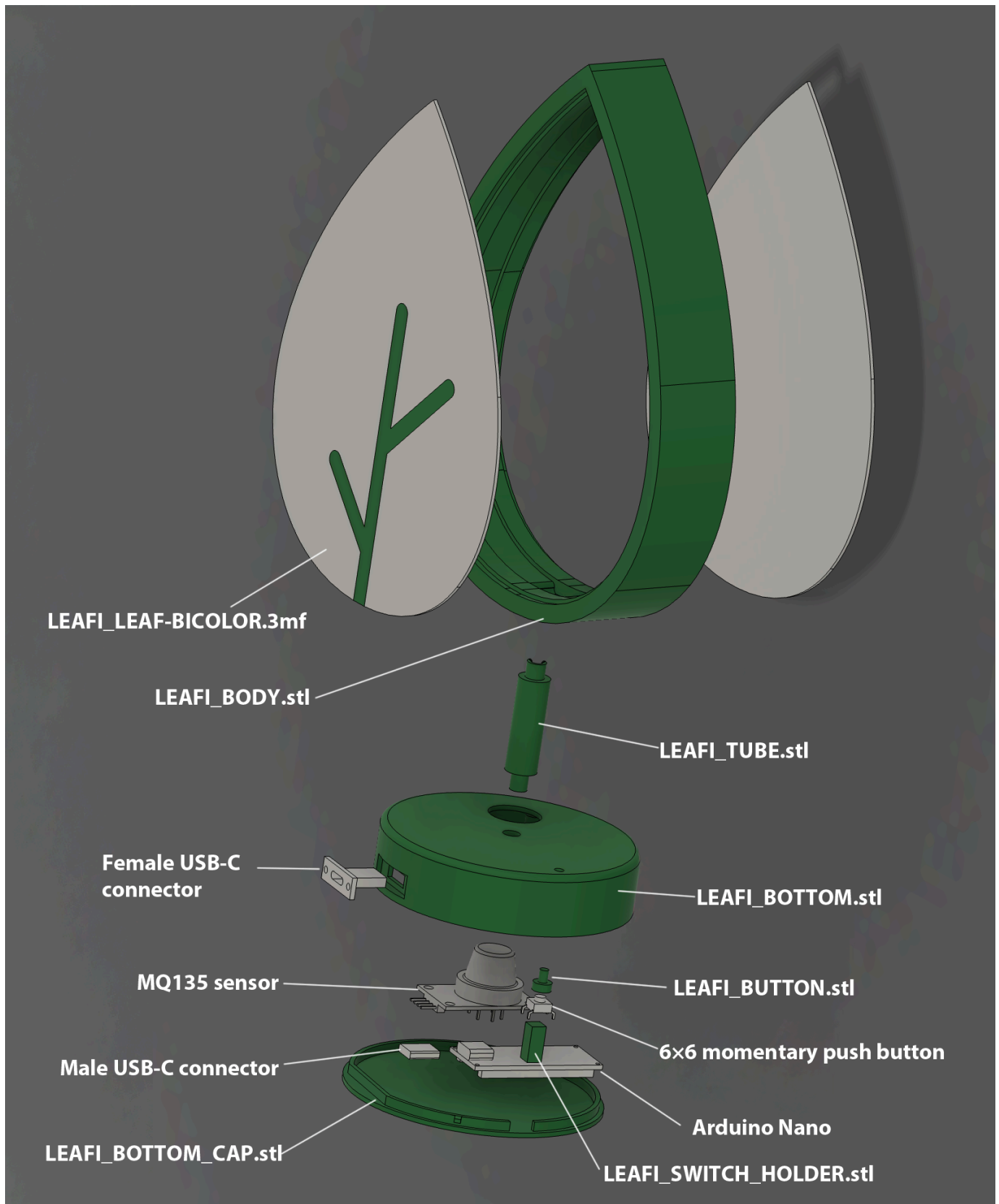
Enjoy the glow!

## Part list

To buy	To Print
1X MQ-135 Sensor <a href="https://tinyurl.com/yc4zb3pc">https://tinyurl.com/yc4zb3pc</a>	LEAFI_BODY
1X Arduino nano <a href="https://tinyurl.com/2wdw9s6p">https://tinyurl.com/2wdw9s6p</a>	LEAFI_BOTTOM_CAP
1X Female USB-C Connector <a href="https://tinyurl.com/54vz655f">https://tinyurl.com/54vz655f</a>	LEAFI_BOTTOM
1X 330Ω resistor <a href="https://tinyurl.com/2fp34p7y">https://tinyurl.com/2fp34p7y</a>	LEAFI_BUTTON
1X Male USB-C connector <a href="https://tinyurl.com/tsd4apfx">https://tinyurl.com/tsd4apfx</a>	2X LEAFI_LEAF-BICOLOR
WS2812B 5V Led strip (144 Leds/m) <a href="https://tinyurl.com/56exza8c">https://tinyurl.com/56exza8c</a>	LEAFI_SWITCH_HOLDER
1X 6×6 momentary push button <a href="https://tinyurl.com/yxjv8f8m">https://tinyurl.com/yxjv8f8m</a>	LEAFI_TUBE

## Assembly

No screws required for the printed parts– it's all snap-fit  
Female USB-C connector is glued to LEAFI\_BOTTOM.stl



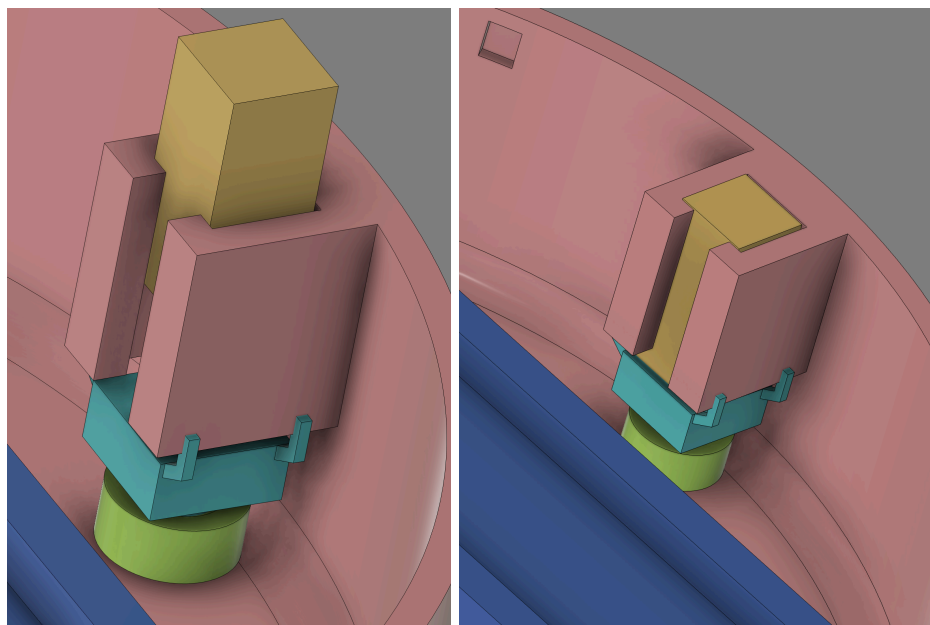
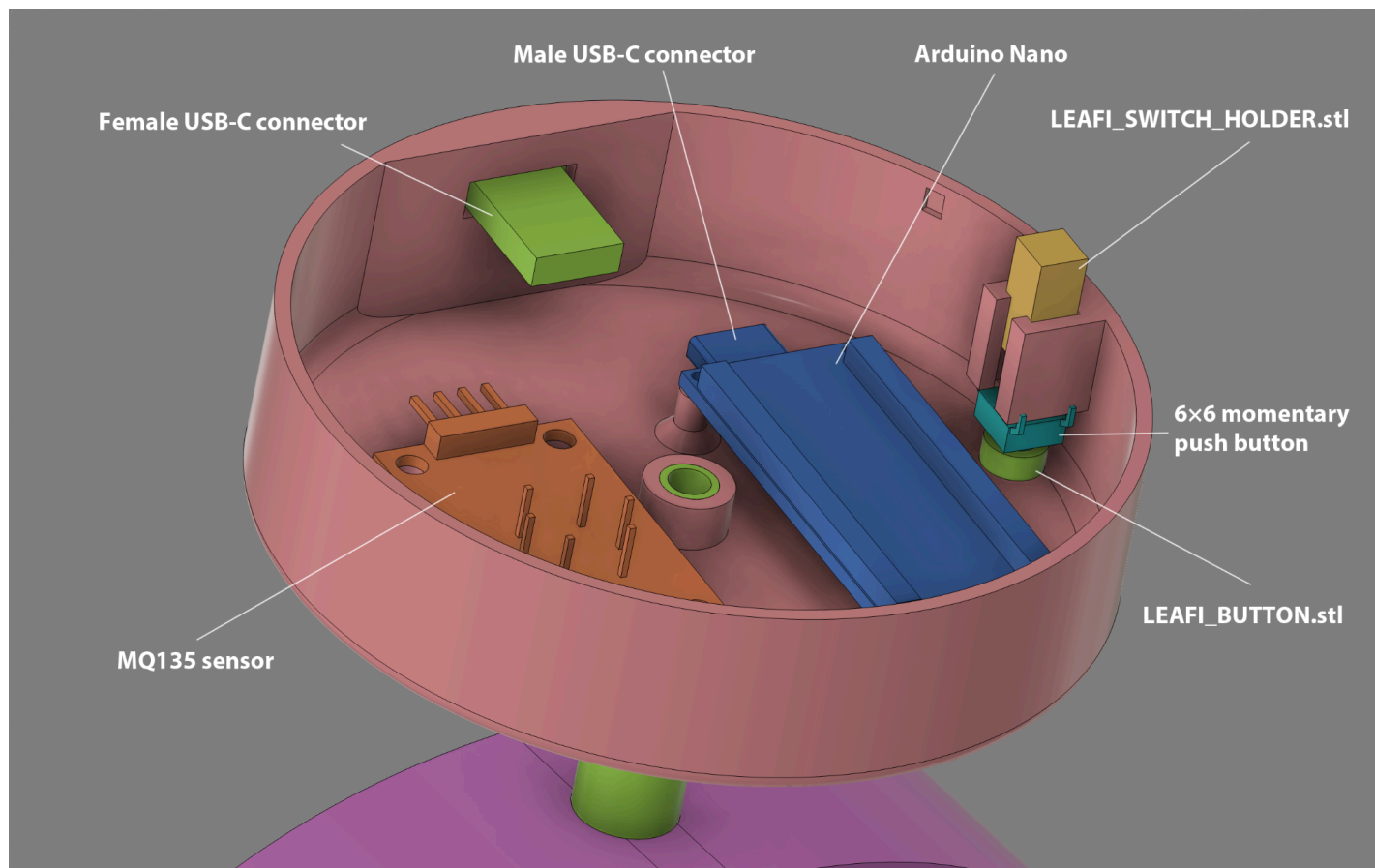
## 🕒 Installing the Button

Insert the LEAFI\_BUTTON.stl in the hole, then place the **6×6 mm momentary push button** on top of the **LEAFI\_BUTTON.stl**. Then, carefully place the **LEAFI\_SWITCH\_HOLDER.stl** on top to hold it in place.

⚠️ It might take a bit of gentle persuasion — don't worry, Leafi won't bite! 😊

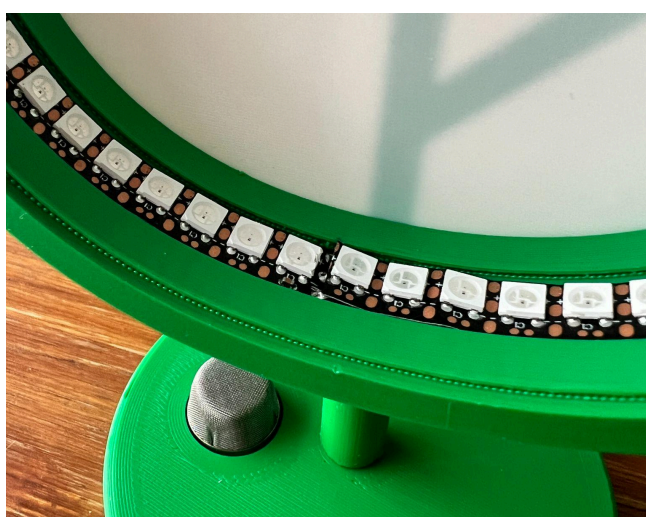
Once everything is aligned, I recommend adding a small dab of glue to keep the button nice and snug.

Now your Leafi is ready to click and glow! ✨



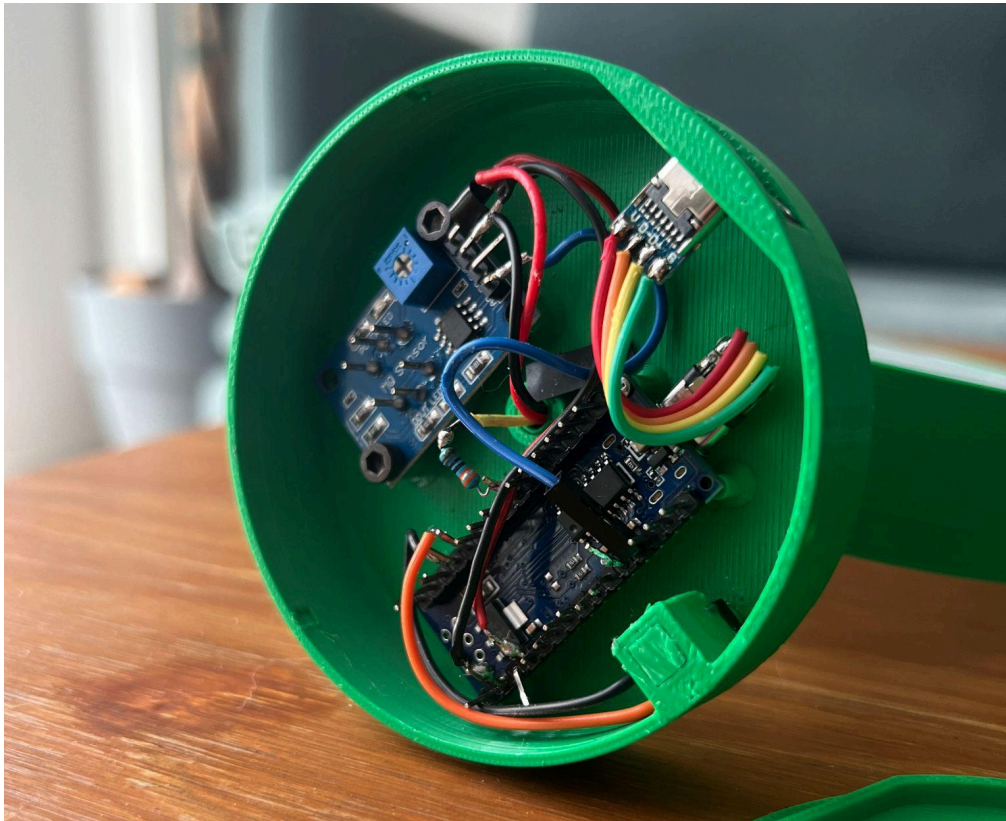


## + LED strip placement



## 🧠 Wiring Summary

Component	Connected to Arduino Nano	Notes
<b>MQ135 Sensor</b>	VCC → 5V GND → GND AOUT → A0	Power supply Ground Analog output to read air quality
<b>WS2812B LED Strip (61 LEDs)</b>	DIN → D6 +5V → 5V GND → GND	Data input (make sure you connect to the <b>input</b> side of the strip) Power supply — use external 5V if needed for stability Common ground
<b>Tactile Button (6×6mm)</b>	One leg → D2 Other leg → GND	Digital input for mode/brightness change Completes the circuit when pressed Pull-down resistor, not needed if using <code>INPUT_PULLUP</code>



And plug the male usb-c to the arduino and connect it to the female usb-c plug

(The Arduino Nano and the MQ135 sensor can be mounted using M3 and M1 screws, but you can also glue them in place instead of screwing.)

## Arduino Code

```
// LeafCheck v8 - (MESURE, LOUNGE, SPOT, WAVE)

#include <Adafruit_NeoPixel.h>

#define LED_PIN 6
#define NUM_LEDS 61
#define MQ135_PIN A0
#define BUTTON_PIN 2

Adafruit_NeoPixel strip(NUM_LEDS, LED_PIN, NEO_GRB + NEO_KHZ800);

const int airMin = 100;
const int airMax = 600;

const uint8_t brightnessSteps[] = {26, 77, 128, 192, 255};
uint8_t brightnessIndex = 2;

bool lastButtonState = HIGH;
bool buttonPressed = false;
bool longPressTriggered = false;
unsigned long buttonPressTime = 0;
const unsigned long longPressDuration = 800;

enum Mode { MESURE, LOUNGE, SPOT, WAVE };
Mode currentMode = MESURE;

uint16_t loungeHue = 0;
int spotPosition = 0;
const int spotSize = 8;
uint16_t spotHue = 0;

float pulseValue = 0;
bool pulseUp = true;

void setup() {
    delay(1500);
    Serial.begin(9600);
    delay(1000);
    pinMode(BUTTON_PIN, INPUT_PULLUP);

    strip.begin();
    strip.setBrightness(brightnessSteps[brightnessIndex]);
    strip.show();

    for (int i = 0; i < NUM_LEDS; i++) {
```

```

uint16_t hue = map(i, 0, NUM_LEDS - 1, 0, 65535);
strip.setPixelColor(i, strip.gamma32(strip.ColorHSV(hue)));
strip.show();
delay(25);
}

fadeToAirColor();
}

void loop() {
  handleButton();

  switch (currentMode) {
    case MESURE: updateAirQuality(); break;
    case LOUNGE: animateLounge(); break;
    case SPOT: animateSpot(); break;
    case WAVE: animateWave(); break;
  }

  delay(20);
}

void handleButton() {
  bool currentState = digitalRead(BUTTON_PIN);

  if (currentState == LOW && lastButtonState == HIGH) {
    buttonPressTime = millis();
    buttonPressed = true;
    longPressTriggered = false;
  }

  if (buttonPressed && currentState == LOW) {
    unsigned long pressDuration = millis() - buttonPressTime;
    if (pressDuration >= longPressDuration && !longPressTriggered) {
      currentMode = static_cast<Mode>((currentMode + 1) % 4);
      blinkTwice();
      Serial.print("MODE: ");
      Serial.println(currentMode);
      longPressTriggered = true;
      buttonPressed = false;
    }
  }

  if (currentState == HIGH && lastButtonState == LOW && buttonPressed && !longPressTriggered) {
    brightnessIndex = (brightnessIndex + 1) % (sizeof(brightnessSteps));
    Serial.print("Luminosité : ");
    Serial.println(brightnessSteps[brightnessIndex]);
    buttonPressed = false;
  }
}

```



```

}

lastButtonState = currentState;
}

void blinkTwice() {
    for (int i = 0; i < 2; i++) {
        for (int j = 0; j < NUM_LEDS; j++) strip.setPixelColor(j, strip.Color(255, 255, 255));
        strip.setBrightness(brightnessSteps[brightnessIndex]);
        strip.show();
        delay(100);
        for (int j = 0; j < NUM_LEDS; j++) strip.setPixelColor(j, 0);
        strip.show();
        delay(100);
    }
}

void updateAirQuality() {
    static uint8_t lastRed = 0;
    static uint8_t lastGreen = 255;
    int mqValue = analogRead(MQ135_PIN);
    Serial.println(mqValue);

    mqValue = constrain(mqValue, airMin, airMax);
    int pollutionLevel = map(mqValue, airMin, airMax, 0, 255);
    pollutionLevel = constrain(pollutionLevel, 0, 255);

    uint8_t targetRed = pollutionLevel;
    uint8_t targetGreen = 255 - pollutionLevel;

    for (int s = 0; s <= 4; s++) {
        float t = s / 4.0;
        uint8_t r = lastRed + t * (targetRed - lastRed);
        uint8_t g = lastGreen + t * (targetGreen - lastGreen);
        for (int i = 0; i < NUM_LEDS; i++) strip.setPixelColor(i, strip.Color(r, g, 0));
        strip.setBrightness(brightnessSteps[brightnessIndex]);
        strip.show();
        delay(10);
    }

    lastRed = targetRed;
    lastGreen = targetGreen;
}

void fadeToAirColor() {
    int mqValue = analogRead(MQ135_PIN);
    mqValue = constrain(mqValue, airMin, airMax);
    int pollutionLevel = map(mqValue, airMin, airMax, 0, 255);

```

```

uint8_t targetR = pollutionLevel;
uint8_t targetG = 255 - pollutionLevel;
for (int fade = 0; fade <= 30; fade++) {
    float t = fade / 30.0;
    for (int i = 0; i < NUM_LEDS; i++) {
        uint32_t fromColor = strip.getPixelColor(i);
        uint8_t r0 = (fromColor >> 16) & 0xFF;
        uint8_t g0 = (fromColor >> 8) & 0xFF;
        uint8_t b0 = fromColor & 0xFF;
        uint8_t r = r0 + t * (targetR - r0);
        uint8_t g = g0 + t * (targetG - g0);
        uint8_t b = b0 + t * (0 - b0);
        strip.setPixelColor(i, strip.Color(r, g, b));
    }
    strip.setBrightness(brightnessSteps[brightnessIndex]);
    strip.show();
    delay(20);
}

void animateLounge() {
    loungeHue += 256;
    for (int i = 0; i < NUM_LEDS; i++) {
        uint16_t hue = loungeHue + i * 500;
        strip.setPixelColor(i, strip.gamma32(strip.ColorHSV(hue)));
    }
    strip.setBrightness(brightnessSteps[brightnessIndex]);
    strip.show();
}

void animateSpot() {
    strip.clear();
    for (int i = 0; i < spotSize; i++) {
        int index = (spotPosition + i) % NUM_LEDS;
        uint16_t hue = (spotHue + i * 1000) % 65535;
        strip.setPixelColor(index, strip.gamma32(strip.ColorHSV(hue)));
    }
    strip.setBrightness(brightnessSteps[brightnessIndex]);
    strip.show();
    spotPosition = (spotPosition + 1) % NUM_LEDS;
    spotHue += 256;
}

void animateWave() {
    static uint16_t waveHue = 0;
    static int waveCounter = 0;
    unsigned long time = millis();

```

```
waveCounter++;  
if (waveCounter >= 10) {  
    waveHue += 1;  
    waveCounter = 0;  
}  
  
for (int i = 0; i < NUM_LEDS; i++) {  
    float phase = (time / 600.0) + i * 0.2;  
    float intensity = (sin(phase) + 1.0) * 0.5;  
    uint8_t level = map(intensity * 255, 0, 255, 0, brightnessSteps[brightnessIndex]);  
    uint32_t color = strip.gamma32(strip.ColorHSV(waveHue * 256, 255, level));  
    strip.setPixelColor(i, color);  
}  
strip.show();  
}
```

Leafi – Nature’s way of saying: “Open a window.”