



# **POSTURE CORRECTING BELT**

# **MEET THE TEAM**

**AARIF NAZAR**

**ABHINAV K**

**NAFIH JIDAN K M**

**SHIVSHANKAR S**


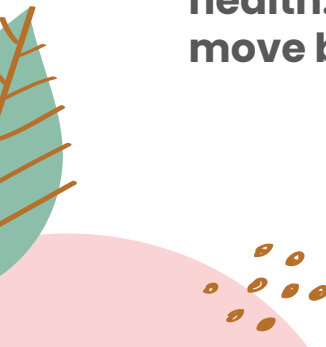


# INTRODUCTION

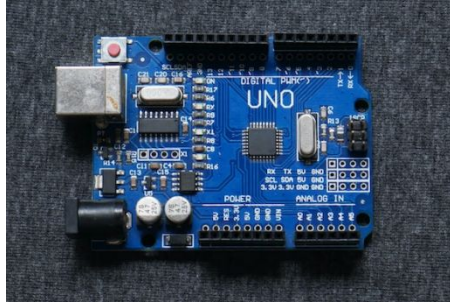


**Our posture correction belt uses cutting-edge motion sensing (MPU6050 accelerometer and gyroscope) technology to monitor your upper body posture in real-time. When it detects slouching or abnormal spine tilt for over a few seconds, it gently alerts you with a quiet vibration, helping you make instant adjustments. No loud alarms, no discomfort — just a subtle nudge to sit or stand tall.**

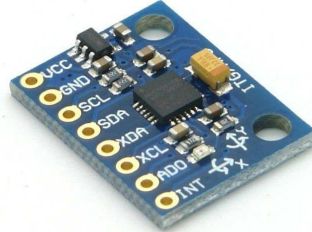
**Lightweight, discreet, and easy to wear, is ideal for office workers, students, fitness enthusiasts, and anyone who wants to take control of their posture health. It blends seamlessly into your day, empowering you to feel better, move better, and live stronger.**



# COMPONENTS



**ARDUINO UNO**



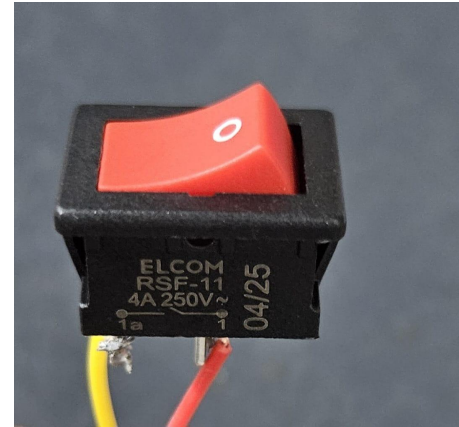
**MPU6050**



**VIBRATION MOTOR**



**BATTERY 9V**



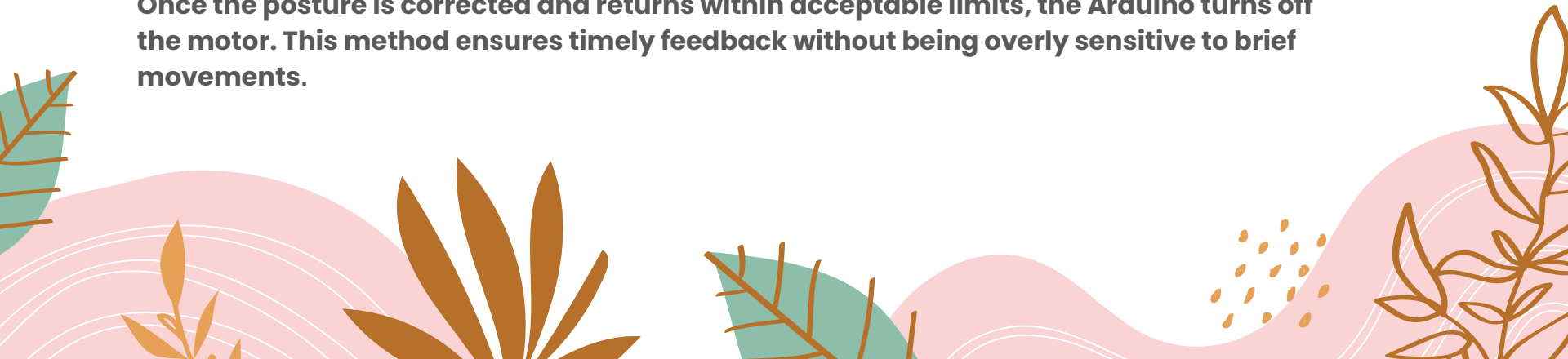
**SWITCH**

# WORKING

The system begins by calibrating the user's current posture during the initial startup. This baseline angle serves as the reference for detecting posture deviations. As the user moves, the MPU6050 continually sends orientation data to the Arduino.

If the orientation deviates beyond a set angular threshold (e.g.,  $10^\circ$ ), and this deviation is maintained for longer than four seconds, the Arduino interprets this as bad posture. In response, it activates the vibration motor using a transistor switch. The motor vibrates intermittently (1s on, 1s off) to alert the user.

Once the posture is corrected and returns within acceptable limits, the Arduino turns off the motor. This method ensures timely feedback without being overly sensitive to brief movements.



# IMPLEMENTATION CODE

```
1  #include <Wire.h>
2  #include <MPU6050.h>
3
4  MPU6050 mpu;
5
6  const int motorPin = 8;
7  float baselineAngleX = 0.0;
8  float baselineAngleY = 0.0;
9  const float angleThreshold = 10.0; // Degrees of tilt allowed
10 const unsigned long badPostureDelay = 2000; // 4 seconds
11 const unsigned long vibrationCycle = 1000; // 1 second ON/OFF
12
13 unsigned long postureStartTime = 0;
14 unsigned long lastVibrationToggle = 0;
15 bool postureIsBad = false;
16 bool vibrating = false;
17 bool motorState = false;
18
19 void setup() {
20   Serial.begin(9600);
21   Wire.begin();
22   mpu.initialize();
23
24   pinMode(motorPin, OUTPUT);
25   digitalWrite(motorPin, LOW);
26
27   if (!mpu.testConnection()) {
28     Serial.println("MPU6050 connection failed");
29     while (1);
30   }
31
32   Serial.println("Calibrating... Keep good posture.");
```



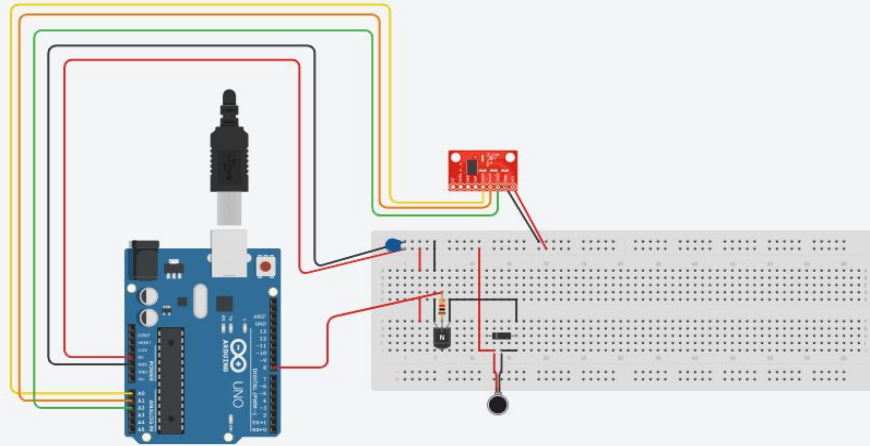
```
33 delay(3000); // Hold still
34
35 int16_t ax, ay, az, gx, gy, gz;
36 mpu.getMotion6(&ax, &ay, &az, &gx, &gy, &gz);
37 baselineAngleX = atan2(ay, az) * 180 / PI;
38 baselineAngleY = atan2(ax, az) * 180 / PI;
39 Serial.println("Calibration complete.");
40 }
41
42 void loop() {
43   int16_t ax, ay, az, gx, gy, gz;
44   mpu.getMotion6(&ax, &ay, &az, &gx, &gy, &gz);
45
46   float angleX = atan2(ay, az) * 180 / PI;
47   float angleY = atan2(ax, az) * 180 / PI;
48
49   float deviationX = abs(angleX - baselineAngleX);
50   float deviationY = abs(angleY - baselineAngleY);
51
52   // Print continuous data
53   Serial.print("Angle X: "); Serial.print(angleX);
54   Serial.print(" | Angle Y: "); Serial.print(angleY);
55   Serial.print(" | Dev X: "); Serial.print(deviationX);
56   Serial.print(" | Dev Y: "); Serial.println(deviationY);
57
58   bool badPosture = (deviationX > angleThreshold || deviationY > angleThreshold);
59   unsigned long currentTime = millis();
60
61   if (badPosture) {
62     if (!postureIsBad) {
63       postureIsBad = true;
64       postureStartTime = currentTime;
```





```
65 } else if ((currentTime - postureStartTime >= badPostureDelay)) {
66     vibrating = true;
67
68     // Toggle vibration every 1 second
69     if (currentTime - lastVibrationToggle >= vibrationCycle) {
70         motorState = !motorState;
71         digitalWrite(motorPin, motorState ? HIGH : LOW);
72         lastVibrationToggle = currentTime;
73
74         Serial.println(motorState ? ">> VIBRATION ON" : ">> VIBRATION OFF");
75     }
76 }
77 } else {
78     postureIsBad = false;
79     vibrating = false;
80     digitalWrite(motorPin, LOW);
81     motorState = false;
82     Serial.println(">> Posture OK. Vibration stopped.");
83 }
84
85 delay(100);
86 }
```

# CIRCUIT DIAGRAM



# KEY FEATURES

## 1. Real-Time Posture Monitoring

Utilizes the MPU6050 accelerometer and gyroscope to detect upper body tilt and posture changes accurately in real-time.

## 2. Smooth Feedback System

Gently alerts the user with a vibration when poor posture is detected and sustained for more than 4 seconds.

## 3. Non-Intrusive Alert Mechanism

The vibration motor operates in a 1-second on/off pattern, giving a subtle and effective reminder without distraction.

## 4. Custom Calibration

Automatically calibrates the user's "good posture" when the device is powered on, ensuring personalized accuracy.



# CONCLUSION

**This posture correcting belt provides an innovative and affordable solution for real-time posture monitoring. It combines sensors, microcontrollers, and feedback mechanisms in a wearable format that encourages better habits and reduces the risk of postural injuries.**

**Through intelligent detection and minimal disruption, users are guided to maintain good posture consistently. The prototype can be further improved and commercialized for wide-scale use.**





# ACKNOWLEDGMENT ●

Farhan – Taught us soldering



**THANK  
YOU**