ECE 445

SENIOR DESIGN LABORATORY

FINAL REPORT

GainSense:
Exercise Repetition & TUT Counter

Team #31

Prithvi Patel
(prithvi7)

Arhan Goyal
(arhang2)

Vikrant Banerjee
(vikrant3)

TA: Sanjana Pingali
Professor: Yang Zhao
Spring 2025

May 4th, 2025

1. Introduction

1.1 Problem. Most people struggle to maintain high-quality workouts, especially without a gym trainer. Trainers are expensive and most trainers only correct basic form, count reps[1], and ensure reps are done slowly to reach the desired time-under-tension[2] (TUT). This problem gets exacerbated when progressively overloading or when muscles are tired at the end of the workout. Counting reps and reaching the desired TUT are the main metrics most gym-goers understand and struggle to hit.

1.2 Solution. Our wristwatch-style device counts the number of reps the user performs and measures TUT. It contains a vibration motor which buzzes once desired TUT is reached, prompting the user to bring the arm down to complete the rep.

1.3 Motivation. The modern fitness landscape is increasingly shifting toward personalized and self-guided workouts. However, one major challenge persists for users exercising without professional supervision: the inability to accurately track workout quality, particularly in terms of repetition count and time-under-tension (TUT). These metrics are critical for building strength and ensuring proper form but are often estimated imprecisely, especially in the absence of visual feedback or a trainer. While smartwatches and fitness trackers have made strides in heart-rate monitoring and activity tracking, most either offer limited rep counting functionality (~65% accuracy) or fail entirely to measure TUT.

Our entire team loves going to the gym and regularly faces this struggle. We decided to use our Senior Design project to solve this very real problem we face on a daily basis.

1.4 High-Level Requirements. The performance requirements for this project were as follows:

1. Adjustable TUT duration between 1 to 10 seconds in 1-second increments.
2. Minimum 90% accuracy in exercise rep detection.
3. Time-under-tension measurement error margin less than ±1 second per rep.

2. Design

2.1 Block Diagram

[1] Rep: Short for “repetition”, a single execution of an exercise
[2] Time-Under-Tension: Total amount of time a muscle is held under load or strain during a set of reps

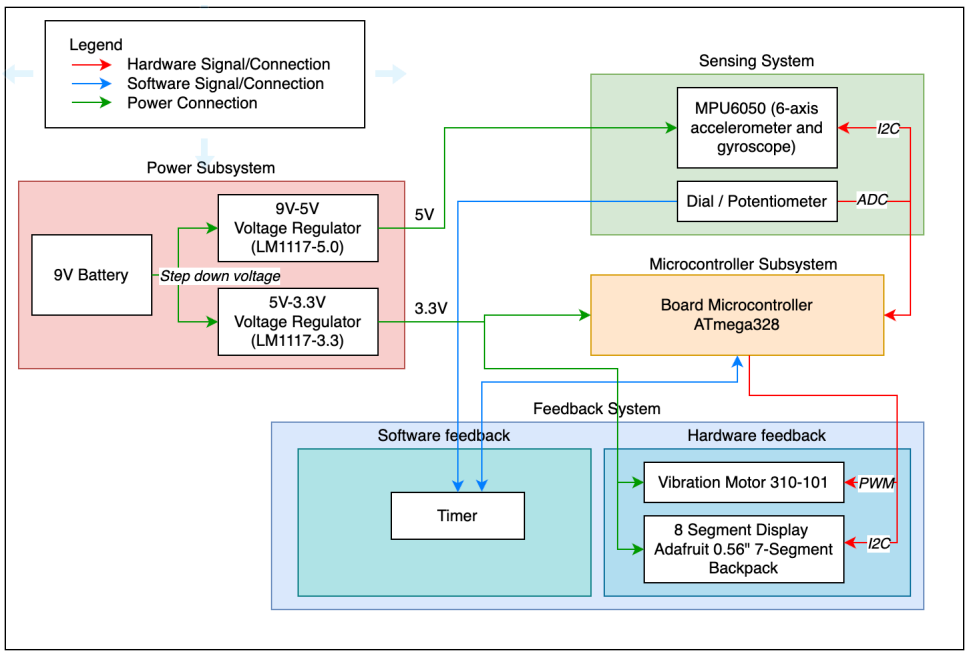
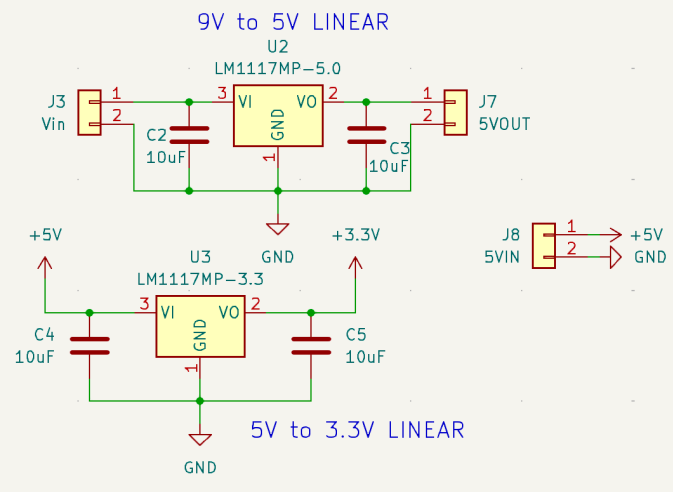


Fig. Block Diagram

2.2 Subsystem Design

2.2.1 Power Subsystem:



Overview

The purpose of this subsystem is to provide a stable power supply to all components of the device. It consists of a 9V battery pack which serves as the primary energy source, which feeds into two voltage regulators: the AMS-1117-5V for a 5V line and the AMS-1117-3.3V for a 3.3V line. The 5V line powers the microcontroller, display, and vibration motor, while the 3.3V line powers the IMU (MPU6050).

Design Considerations

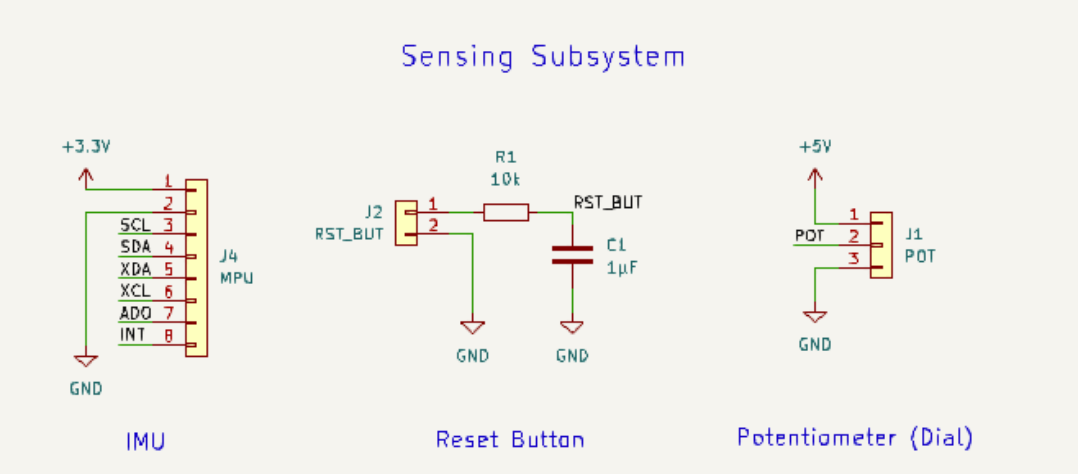
When designing the Power Subsystem, we weighed the pros and cons between switch mode DC-DC converters and classic linear regulators across various battery voltage values. Our original design used a 12V battery stepped down to 5V using a buck converter to avoid excess heat dissipation as buck converters efficiently step-down voltage rather than dissipating energy as heat. A buck converter was necessary due to the significant drop between 12V and 5V. This combination resulted in a heavier design because of the bulky weight of the 12V battery and more complex design due to the extra peripherals necessary for the buck converter circuit. Ultimately, we switched the buck converter to a linear regulator in combination with dropping the input voltage from a 12V battery to a much lighter 9V battery, which helped simplify the final circuit and drop the weight of the final product.

Requirements and Verification

Requirement	Validation Process
The subsystem must output 5.0V ±0.2V (for the display and motor) and 3.3V ±0.2V (for the microcontroller and MPU6050)	The battery was connected to the 5V linear regulator and the 5V linear regulation passed voltage into the 3.3V regulator. We used an oscilloscope to measure the output of each regulator and found the 5V regulator output 4.99V and the 3.3V regulator output 3.29V.
The battery must support ≥2 hours of continuous operation under peak load (300mA @ 5V, 100mA @ 3.3V).	The device was left running for 2 hours, interacting with it every 10 minutes to ensure it is still working as intended. The device properly worked for 2 hours
Includes overcurrent protection on both output rails. Interfaces: Input: 12V LiPo battery. Outputs: 5V (to	Mitigated overcurrent issues by ensuring the Linear Regulator used (LM1117) has overcurrent protection embedded within it

microcontroller, MPU6050) and 5V (to display, motor).	

2.2.2 Sensing Subsystem



Overview

The purpose of the Sensing Subsystem is to track the users arm movement and monitor their TUT during an exercise. This subsystem consists of an MPU6050 accelerometer and gyroscope, which are used to detect changes in motion and orientation to accurately sense the beginning and end of a repetition using *I2C communication protocol*. Additionally, a potentiometer will be used to allow the user to adjust the TUT criteria according to their own fitness goals. We will need to use an *analog to digital converter (ADC)* for the ATmega328 to be able to process analog data from the dial. This subsystem is necessary to gather motion data and send it to the board's microcontroller, which will process the information to accurately count repetitions and track TUT.

Implementation

1. The MPU6050 is configured during setup to use an 8G accelerometer range and 21Hz low-pass filtering for noise reduction. Communication is established via I2C.

```
1. // In setup():
2. mpu.setAccelerometerRange(MPU6050_RANGE_8_G);
3. mpu.setFilterBandwidth(MPU6050_BAND_21_HZ);
4. if (!mpu.begin()) {
5.  Serial.println("Failed to find MPU6050 chip");
6.  while (1) { delay(10); }
7. }
8.
```

2. Z-axis acceleration is sampled every loop iteration to detect vertical motion. A dead zone filter ignores minor movements below MIN_MOVEMENT.

This equation was used in which raw acceleration a_z (Z-axis) data is filtered to ignore small movements below a threshold Δa_{min}:

$$a_{\text{filtered}} = \begin{cases} 0 & \text{if } |a_z - a_{\text{rest}}| < \Delta a_{\min} \\ a_z & \text{otherwise} \end{cases}$$

where $a_{rest} = 11.5$ m/s^2 (resting Z-axis acceleration) and $\Delta a_{min} = 3$ m/s^2.

```
// In loop():
sensors_event_t a, g, temp;
mpu.getEvent(&a, &g, &temp);
float accelZ = a.acceleration.z;
if (abs(accelZ - 11.5) < MIN_MOVEMENT) return;
```

3. The potentiometer on pin A1 adjusts the Time-Under-Tension (TUT) threshold from 1–10 seconds. Changes trigger immediate updates to timing thresholds and display feedback.

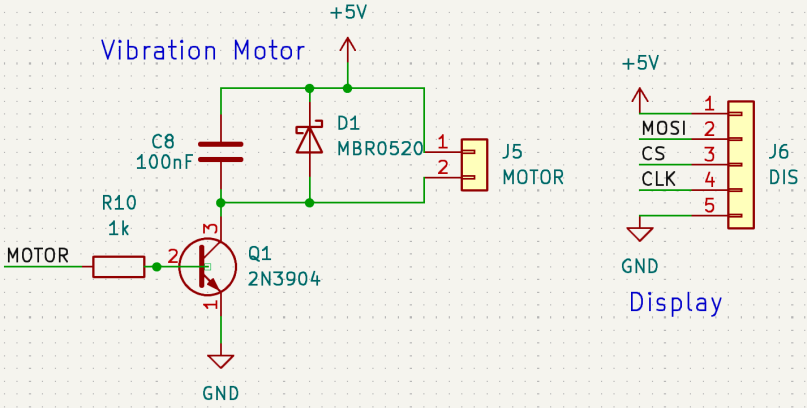
```
// In loop():
int raw = analogRead(POT_PIN);
unsigned long secs_setting = map(raw, 0, 1023, SEC_MIN, SEC_MAX);
if (BUZZ_THRESHOLD != millis_setting) {
    BUZZ_THRESHOLD = millis_setting;
    MIN_REP_TIME = millis_setting;
    display.print(secs_setting, DEC);
}
```

Requirements and Verification

Requirement	Verification
The MPU6050 must sample acceleration/gyro data at ≥ 10Hz	Connect the MPU6050 to the microcontroller and upload firmware that streams raw sensor data via a serial port. Print time stamps when sampling the data. Confirm that we are getting ≥ 10 samples per second.
The potentiometer must adjust TUT in 1–10s increments using a dial.	To test the dial, we turned the potentiometer and printed the dial, ensuring the value displayed from 1-10. The display correctly showed values 1-10 100% of the time.

Reset button calibrates gyro sensor to zero and sets the rep count to zero.	Place the device on a stable surface to ensure there is no motion and then click on the reset button. Pre-calibration data must show near-zero values (±0.05g for accelerometer, ±0.5°/s for gyro). Post-calibration motion data must reflect actual movement (e.g., >0.5g change during lifting).

2.2.3 Feedback Subsystem



Overview

The purpose of this subsystem is to provide real-time feedback to the user through both software and hardware mechanisms. The timer is a software feedback component that is preset by the user via a potentiometer/dial that will indicate a set amount of seconds that the user would like to be under tension during their repetition. Once the preset time is reached, the board's microcontroller will trigger the vibration motor, which will provide a haptic feedback to the user to signal that they can now complete their range of motion. The motor will be controlled using Pulse-Width Modulation (PWM). Furthermore, the 8-segment display will display the current count of repetitions completed, incrementing each time the user completes a repetition. This display will be controlled using the I2C communication protocol can be reset to the value 0 by using the "reset" button.

Implementation

The display update routine follows various different parts.

1. On startup, the device writes to the which exercise mode you are in betwerrn forward-backward movement displayed as "FBMd" or vertical movement displayed as "UPMd".

```
1.   if (showMode && (now-modeStart<1000)) {
2.     display.writeDigitAscii(0, modeFB ? 'F' : 'U');
3.     display.writeDigitAscii(1, modeFB ? 'B' : 'P');
4.     display.writeDigitAscii(3, 'M');  display.writeDigitAscii(4, 'd');
5.   }
```

```
6.
```

2. When the time under tension threshold has been met, as a secondary indicator we write the word “dOnE” in ascii on the display

```
1.   else if (showDone && (now-doneStart<1000)) {
2.     display.writeDigitAscii(0,'d'); display.writeDigitAscii(1,'O');
3.     display.writeDigitAscii(3,'n'); display.writeDigitAscii(4,'E');
4.   }
5.
```

3. If the user changes their Time Under Tension threshold using the dial, the value they choose will be written on the display

```
1.   else if (showSec && (now-secStart<1000)) {
2.     display.print(secSet,DEC);
3.   }
4.
```

4. When the user completes a repetition, the count of their repetition is updated if they meet the TUT threshold they set for themselves, if not then their repetition count is not incremented. The time is continuously being updated as well.

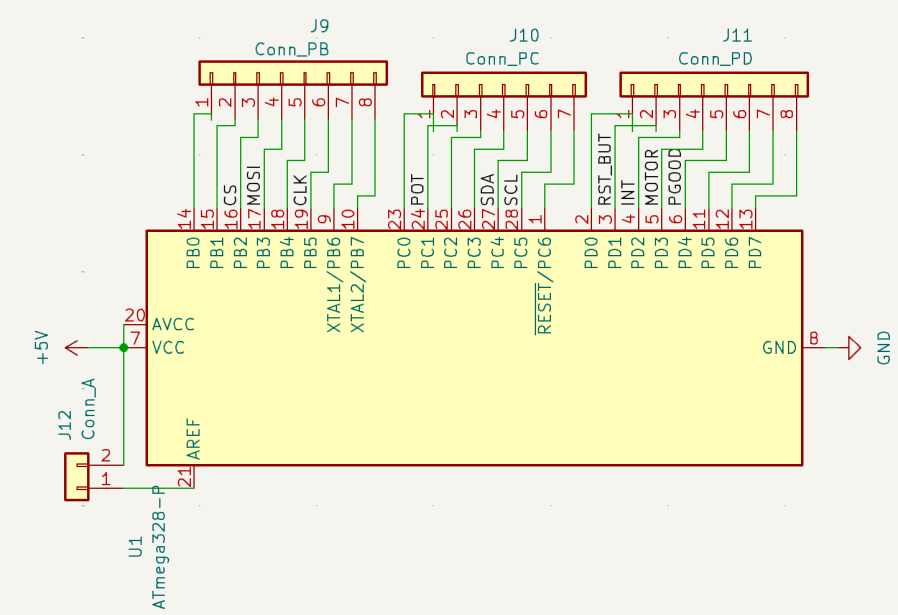
```
 1.   else {
 2.     showDone = showSec = showMode = false;
 3.     uint8_t tut = 0;
 4.     uint16_t repsShown = 0;
 5.     if (!modeFB) {
 6.       tut = lifting ? (now-repStart)/1000 : 0;
 7.       repsShown = repCount;
 8.     } else {
 9.       tut = outward ? (now-repStartFB)/1000 : 0;
10.       repsShown = repCountFB;
11.     }
12.     display.writeDigitNum(0,(tut/10)%10);
13.     display.writeDigitNum(1, tut%10);
14.     display.drawColon(true);
15.     display.writeDigitNum(3,(repsShown/10)%10);
16.     display.writeDigitNum(4, repsShown%10);
17.   }
18.
```

Requirements and Verification

Requirement	Verification
The 8-segment display must update within 1s of a detected rep.	Using the Arduino serial monitor, we took time stamps, marking the time of detected repetition and the time the repetition counter was incremented. This resulted in a 0.6s difference between detecting and displaying

The vibration motor must vibrate strong enough such that it is felt by the user	Each team member tested various vibration intensities and ranked them based on the criteria of how well the vibration is felt, and how little it impairs their movement. Using this we decided on a duty value of 150
The display must support 2-digit output for the timer and 2-digit output for repetitions	To test the timer, we left the device in a “mid repetition” state making sure the timer continues to increment past 10 seconds, it did. To test the repetition counter, we completed 11 repetitions and carefully made sure it tracked each one and updated the counter up to 11, it did.

2.2.4 Microcontroller Subsystem



Overview

The purpose of the Microcontroller (AtMEGA328-P) Subsystem is to be the central processing unit of the device, handling software and hardware data it receives and delivers between each subsystem. It receives motion data from the MPU6050 and analyzes it to detect and count repetitions. Furthermore, it reads the input from the potentiometer to determine the user’s expected TUT. Once processed, the microcontroller communicates with the Feedback System to

provide both visual and haptic cures. Finally, the microcontroller also ensures power is distributed properly between each of the subsystems.

Implementation

1. The microcontroller configures I²C for sensor/display communication, PWM for motor control, and GPIO for user inputs.

```
void setup() {
    Wire.begin(); // I²C for MPU6050 and display
    pinMode(MOTOR_PIN, OUTPUT); // PWM motor control
    pinMode(POT_PIN, INPUT); // Potentiometer input
    mpu.begin(); // Initialize IMU
    display.begin(0x70); // I²C display
}

```

2. The loop() function runs at 10Hz (100ms intervals):
    - Read potentiometer → adjust TUT threshold
    - Poll MPU6050 → process acceleration data
    - Update rep state machine
    - Refresh display

```
void loop() {
    // 1. Read potentiometer
    unsigned long secs_setting = map(analogRead(POT_PIN), 0, 1023, 1, 10);

    // 2. Get MPU6050 data
    sensors_event_t a, g, temp;
    mpu.getEvent(&a, &g, &temp);

    // 3. Rep detection logic
    if (accelZ < LIFT_THRESHOLD && !lifting) { /* Start rep */ }

    // 4. Update display
    display.writeDigitNum(0, (liveTUT / 10) % 10);
    display.writeDisplay();
}

```

3. Hardware timers and millis() provide non-blocking timing for:
    - TUT duration tracking
    - Motor PWM control
    - Display refresh intervals

```
analogWrite(MOTOR_PIN, 150); // 59% duty cycle (490Hz default)
delay(500); // Fixed 500ms buzz duration
analogWrite(MOTOR_PIN, 0);

```

Requirements and Verification

Native I2C ports must interface with MPU6050	Oscilloscope checks SCL/SDA lines during data transmission for stable clock and ACK signals.

The PWM pin must output proper frequency and duty cycle to feel the vibration motor.	Oscilloscope measures PWM frequency/duty cycle during motor activation.

2.3 Final Product

Fig. Final Product

A picture of our final product is attached above. It features a handy USB-C charging port on the side for quick charging. The silver dial at the bottom can be used the adjust the target TUT. The RESET button can be used to recalibrate the device for a new exercise and reset current TUT and

rep count to zero. POWER button can be used to turn on/off the device. MODE button can be used to switch between upward-downward and forward-backward exercises.

Design Nuances

Potentiometer. The potentiometer is used to set the target TUT. The potentiometer's range of motion has been mapped to 1–10s with 1 second intervals. As the user adjusts the TUT, the live updated TUT value is shown on the display.

TUT Update Error. We wanted the TUT to be measured within 1s of actual value. However, "delay" in Arduino code would stop execution and after all components were interfaced with our microcontroller, the delays added up to increase error beyond 1 second. To fix the Time-Under-Tension (TUT) inaccuracy caused by the delay() calls, we removed those delays and now compute TUT using timestamps taken at the start and end of each interval.

Accelerometer + Gyroscope Calibration. We wanted to support several exercises. However, the orientation of the device changes from exercise to exercise. To account for this, we added a device re-calibration mechanism. The user brings their hands/device up to the start point of the exercise and presses the "RESET" button. This resets the rep count, live rep timer, and recalibrates the device. Recalibration is done by taking 200 accelerometer readings @ 200Hz (~1 second) when the device is stationary at the start point of exercise. The average of these readings forms the gravity vector. All subsequent readings are projected on to this vector to figure out which direction the watch/arm is moving in.

Vibration Motor. Our vibration motor features a rotating head, which vibrates the entire motor. However, the head stops moving when planted against a surface. We also did not want the entire unit on the wrist to vibrate very strongly, thereby endangering the user. We designed a special oversized box for the motor at the bottom of the wrist box unit. This way the motor stays propped up inside the box and vibrates freely. It makes a weak noise which only the user can hear and vibrates just enough to let the user know they can complete the rep in case they cannot view the "dOnE" prompt on the watch display. The vibration motor buzzes in the following scenarios:

- Reached target TUT
- Calibration completed and "rEdY" appears on screen (on switching on device or after pressing RESET)

Fig. Vibration Motor

3. Cost & Schedule

3.1 Cost Analysis

Cost 1: Labor Cost

$$Labor\ Cost = Ideal\ Hourly\ Salary\ *\ Actual\ Hours\ Spent\ *\ 2.5$$

$$Average\ Ideal\ Hourly\ Salary\ for\ 3\ Members = 60\frac{USD}{h}$$

$$Actual\ Hours\ Spent\ by\ 3\ Members = 500h$$

$$Labor\ Cost = 60\frac{USD}{h}\ *\ 500h\ = \mathbf{30{,}000\ USD}$$

Cost 2: Machine Shop Cost

$$Machine\ Shop\ Cost = Hourly\ Salary\ for\ Greg * Hours\ Spent + Cost\ of\ Materials\ for\ Box$$

$$Machine\ Shop\ Cost = \ 150\frac{USD}{h}\ * 3h = \mathbf{450\ USD}$$

Cost 3: Cost of Parts

ATmega328P	$1.71
Potentiometer	$1.42
Push buttons x3	$3.75

Vibration motor	$4.10
MPU6050 board	$1.60
Adafruit display	$3.95
Encloser	$6.99
Total	$23.52

Total Cost

$30473.52 to get the first product done.

3.2 Schedule

March 10th - March 17th	• Build and demo the breadboard prototype. • Finish ordering all the parts that were not used for the breadboard demo. • Make sure the breadboard prototype works with the buck converter and linear regulator.
March 18th - March 24th	• Design the CAD model for the machine shop. • Solder the PCB to check for working/not working components. • Make edits to the PCB as needed for the order.
March 25th - March 31st	• Program ATTmega328 with the MPU attached to the PCB. • Make the display work with a microcontroller. • Work on condensing the PCB design so it could fit in a smaller case.
April 1st - April 7th	• Install the product into the case. • Test for the sensor requirements being fulfilled. • Test for the power requirements being fulfilled.
April 8th - April 14th	• Work on the final demo presentation.

4. Conclusion

4.1 Accomplishments

There were plenty of accomplishments throughout this project. To start, each of our high level requirements were met. We can adjust TUT duration between 1 to 10 seconds in 1-second increments, when performing exercises, we have a 98% accuracy of repetition detection for exercises with vertical movements, and the error margin of time under tension measurement is les that 1 second per repetition. Furthermore, we added multiple other features to enhance our product for the user. Our battery is a rechargeable battery with its charging port exposed which

allows the user to charge the battery rather than replace it. We added a mode button to toggle interchange between vertical and horizontal movements, allowing us to track more exercises. And finally, we enhanced the reset button to not only reset the rep counter but also recalibrate the device so the user can perform a new exercise upon reset.

4.2 Uncertainties

The main uncertainty with our product is its ability to detect exercises that move in a forward and backwards motion. We managed to detect vertical motion with a near 100% accuracy by creating a reference point to the direction of gravity, which will always read a constant value. Using this information we could create a stable reference point for tracking upward movement and downward movement. Because gravity only acts vertically, we cant use the same trick to detect horizontal movement. Instead we attempted to force read along the y-axis since the MPU is in a set position within the device, by taking note the base_y_accel. We then calculate curr_y_accel – base_y_accel in an attempt to detect forwards and backwards movement. This method had 2 glaring issues. First, it is not nearly as dynamic since it forces you to always have the device strapped in a way that the y-axis is horizontal, and second, it generally did not detect our reps as we intended.

4.3 Future Work

There are a few things that need to be fine tuned in future work. First, we would like to use a more robust accelerometer/gyroscope. The MPU6050 has already reached its end of life, and newer, more accurate devices have replaced it such as the MPU6500. This change would help to create more accurate readings and help finetune the detections of reps by being less prone to noise. After finetuning our exercise logic with the new accelerometer, we would like to compress the PCB design to make it more watch like. Finally, we would like to integrate it into a compression sleeve which would keep it more stabilized and allow it to be work on both the leg and arm.

4.4 Ethical Considerations

Our team adhered to the IEEE Code of Ethics and ACM Code of Ethics and Professional Conduct throughout the project lifecycle to ensure an ethical and rigorous workflow. Central to this commitment was implementing a structured revision process, where every component of the project underwent thorough testing and peer review, with at least one team member independently verifying all work. Weekly meetings with our TA provided additional oversight, fostering accountability and aligning with ACM Code 2.1's emphasis on high-quality processes and outcomes. To further refine our product, we actively solicited and integrated feedback addressing technical shortcomings and correcting errors. This approach directly honored IEEE Code 1.5, which prioritizes honest criticism, error correction, and transparent attribution of contributions.

Safety was paramount in both design and execution. We collaborated closely with the Machine Shop to ensure the physical build met safety standards for user interaction. Electrically, subsystems on the PCB were optimized to prevent overheating, and particular attention was given to the safe integration of the lithium-polymer (Li-Po) battery. Recognizing risks such as

thermal runaway, swelling, and combustion, we enforced strict protocols: charging was exclusively performed with certified Li-Po chargers to avoid voltage mismatches, and the enclosure was designed to be impact-resistant, sweat-proof (achieving IPX4 compliance), and adequately ventilated. Users were instructed to avoid puncturing, bending, or exposing the battery to moisture, and disposal guidelines emphasized full discharge and recycling through authorized facilities. These measures collectively minimized hazards while maintaining device reliability, ensuring compliance with industry safety benchmarks for Li-Po applications.

5. References

All references for the project are as follows. We would also like to thank our TA Sanjana Pingali for her constant support through the process.

1. MPM3550E. 36V, 5A, High-Efficiency, Fast Transient, Non-Isolated, DC/DC Power Module with Integrated Inductor | MPS. (n.d.). https://www.monolithicpower.com/en/mpm3550e.html
2. MPU6050 (gyroscope + accelerometer + temperature) interface with .. MPU6050 (Gyroscope + Accelerometer + Temperature) interface with .. (n.d.-a). https://www.electronicwings.com/avr-atmega/mpu6050-gyroscope-accelerometer-temperature-interface-with-atmega16
3. ATMEGA328P.(n.d.-a).https://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-7810-Automotive-Microcontrollers-ATmega328P_Datasheet.pdf
4. hk_jh. (2023, July 9). Controlling mini vibration motors with MOSFET instead of BJT.Arduino Forum.https://forum.arduino.cc/t/controling-mini-vibration-motors-with-mosfet-instead-of-bjt/1146260
5. IEEE - IEEE Code of Ethics.(n.d.).https://www.ieee.org/about/corporate/governance/p7-8.html
6. LM1117-5.0 Datasheet. Texas Instruments. (n.d.). https://www.ti.com/lit/ds/symlink/lm1117.pdf
7. LM1117-3.3 Datasheet. Texas Instruments. (n.d.). https://www.ti.com/lit/ds/symlink/lm1117.pdf
8. MPU6500 (6-Axis Gyroscope + Accelerometer) Datasheet. TDK InvenSense. (n.d.). https://invensense.tdk.com/wp-content/uploads/2015/02/MPU-6500-Datasheet1.pdf
9. Arduino Uno Rev3 | Arduino Documentation. (n.d.). https://docs.arduino.cc/hardware/uno-rev3
10. ESP32-WROOM-32 Datasheet. Espressif Systems. (n.d.). https://www.espressif.com/sites/default/files/documentation/esp32-wroom-32_datasheet_en.pdf
11. Buck Converter Inductor Selection Guide. Coilcraft. (n.d.). https://www.coilcraft.com/en-us/edu/inductor-selection-for-buck-converter/

12. General Inductor Finder & Selector Tool. Digi-Key Electronics. (n.d.). https://www.digikey.com/en/resources/conversion-calculators/conversion-calculator-inductor-finder
13. Adafruit 0.56" 4-Digit 7-Segment Display w/I2C Backpack. Adafruit. (n.d.). https://www.adafruit.com/product/878
14. Small Vibration Motor - ROB-08449. SparkFun Electronics. (n.d.). https://www.sparkfun.com/products/8449
15. Arduino Wire Library (I2C). Arduino Documentation. (n.d.). https://www.arduino.cc/en/reference/wire
16. Adafruit MPU6050 Arduino Library. GitHub. (n.d.). https://github.com/adafruit/Adafruit_MPU6050