ECE 445
SENIOR DESIGN LABORATORY
FINAL REPORT

Sleep Position Trainer

Group #48

BRIAN PARK (brianp7@illinois.edu)
KYLE LEE (klee281@illinois.edu)
NICK TSE (nstse2@illinois.edu)

TA: Ko, Po-Jen

May 2026

Abstract

This project presents the design and implementation of a sleep position trainer that monitors user posture during sleep and provides real-time feedback when improper positions are detected. The system utilizes an inertial measurement unit (IMU) to track orientation and a vibration motor to alert the user. The goal is to promote better sleeping habits and reduce the risk of sleep-related conditions such as back and neck pain. Testing results demonstrate that the system reliably detects posture changes and provides effective feedback.

Contents

1 Introduction

Poor sleeping posture can negatively impact overall health, leading to discomfort, reduced sleep quality, and long-term musculoskeletal issues. Many individuals are unaware of their posture during sleep and lack an effective method to correct it.

The objective of this project is to design a wearable sleep position trainer that detects improper sleeping positions and provides real-time feedback to encourage correction. The system is designed to be low-cost, comfortable, and reliable for overnight use.

The device consists of an IMU sensor for orientation tracking, a microcontroller for data processing, and a vibration motor for feedback. The system continuously monitors the user's position and activates the motor when an undesirable posture is detected.

Figure 1: Final Sleep Position Trainer Build

2 Design

2.1 Design Procedure

Several approaches were considered for detecting sleep posture, including pressure sensors and inertial sensing. An IMU based approach was selected due to its compact size, low cost, and ability to provide continuous orientation data.

The system processes IMU data through the microcontroller to determine the user's position. Threshold based logic is used to classify acceptable and unacceptable sleep positions.

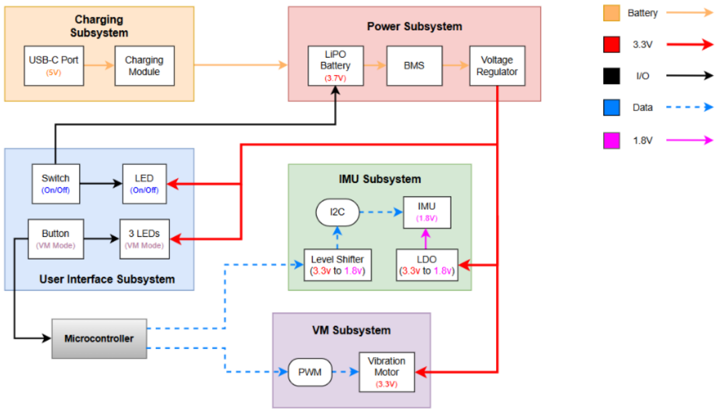

Figure 2: Block Diagram

2.2 Hardware Design Details

The Hardware Design includes 5 Subsystems:

2.2.1 Position Sensing

The position sensing subsystem is responsible for determining the user's orientation using an inertial measurement unit (IMU). The IMU measures position from the acceleration and angular velocity across 6 Axis axes, allowing the system to estimate the user's posture in real time. This data is communicated to the microcontroller via an I2C interface, where it is processed to determine the current sleep position.

To ensure reliable detection, the raw sensor data is filtered and compared against predefined threshold values that distinguish between acceptable and undesirable positions. These thresholds are selected based on expected body orientations during sleep and are tuned through testing to minimize unnecessary alerts.

Additionally, the design prioritizes stability and accuracy by ensuring proper sensor calibration and minimizing noise in the measurements. This allows the system to maintain consistent performance across different users and sleeping conditions.

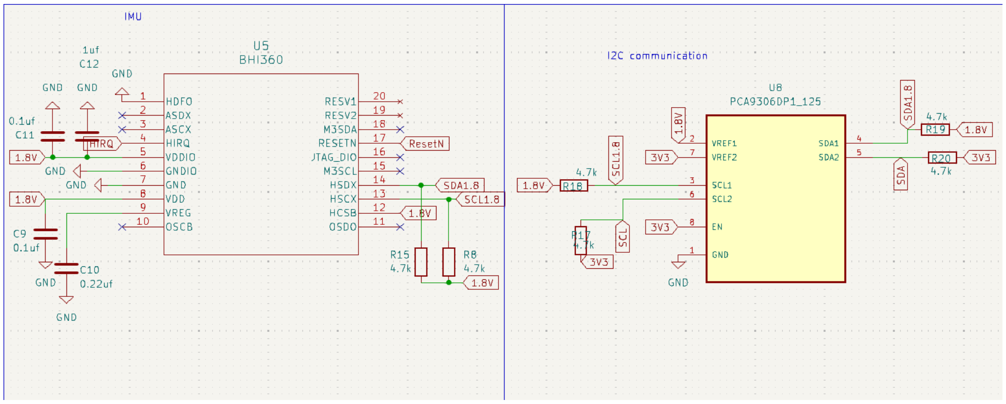

Figure 3: Position Sensing Schematic

2.2.2 User Alert System

This subsystem includes the eccentric rotating mass (ERM) vibration motor, which receives a PWM control signal from the microcontroller to determine when and how strongly the motor should activate. Its purpose is to alert the user when an unwanted sleeping position has been detected for a prolonged period.

The user alert subsystem provides physical feedback using an Adafruit 1201 ERM vibration motor rated for 5 V operation and approximately 11,000 RPM. When the microcontroller detects that the user has remained in an undesirable sleeping position beyond the programmed threshold, it activates the vibration motor to gently prompt the user to reposition. The vibration intensity and duration can be adjusted through the PWM signal, allowing the feedback to be noticeable while minimizing sleep disruption.

The motor driver circuit uses an N-channel MOSFET with a flyback diode to support safe and reliable operation of the ERM motor. The subsystem is controlled directly by the microcontroller and receives power from the power management subsystem. User-selectable vibration strength settings will be indicated through the LEDs.

This subsystem is critical to the overall function of the device. If the vibration motor fails to activate, the user will not receive feedback to correct their sleeping position. In contrast, if the motor remains active indefinitely, the device could become disruptive and uncomfortable rather than helpful.

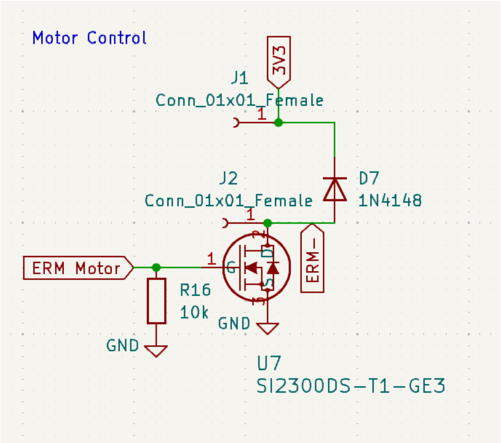

Figure 4: User Alert Schematic

2.2.3 Microcontroller

The microcontroller subsystem uses an ESP32-S3-WROOM-1.

The microcontroller classifies the user's position, it sends control signals to the user alert subsystem. If an improper position is detected, the system activates a vibration motor to prompt the user to adjust their posture. The subsystem operates continuously throughout the sleep period, providing real-time monitoring while maintaining low power consumption to support battery operation.

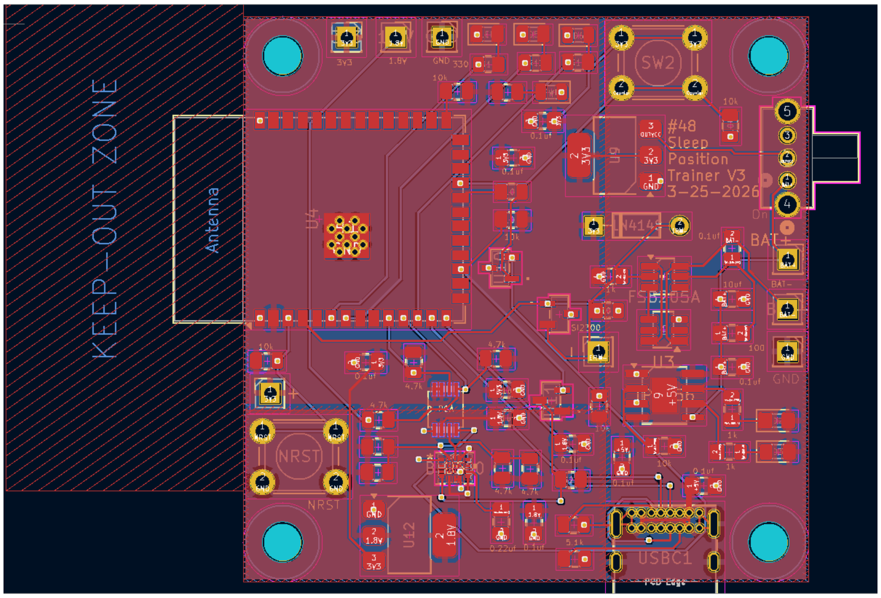

Figure 5: PCB Layout (Microcontroller Positioned on the Top-Left of the PCB)

2.2.4 Physical Build

The physical enclosure is designed using 3D-printed material to house the PCB, LiPo battery, and ERM vibration motor in a compact and lightweight form. The design keeps all components securely integrated while maintaining a small profile suitable for wearable use. Additionally, the material is safe to the user as well as reduces the possibility of injury or damage. The device is worn across the user's chest using an adjustable strap.

The enclosure also provides basic protection for internal components and allows easy access for charging through the USB-C port and switch to turn the device on or off. Overall, the design prioritizes simplicity, comfort, and functionality for continuous use during sleep.

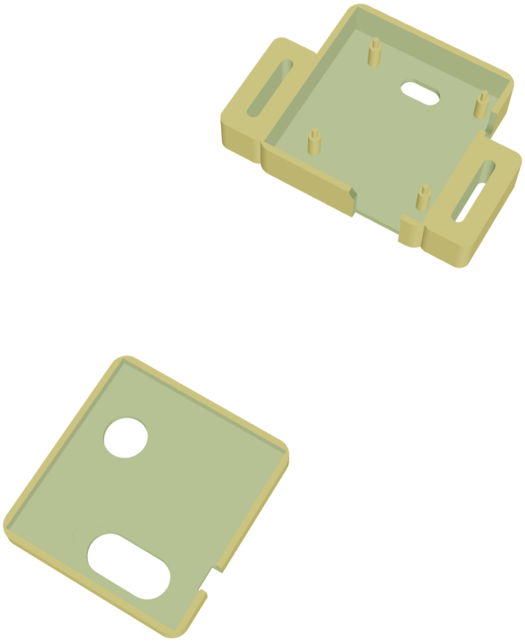

Figure 6: 3D-Model Enclosure

2.2.5 Power Management

The power subsystem is designed to support fully portable operation, allowing the device to function without a direct wired connection and ensuring user comfort during sleep. To achieve this, the system is powered by a 3.7 V lithium-polymer (LiPo) battery with a capacity of 500 mAh. This battery provides sufficient energy for overnight operation while maintaining a compact and lightweight form factor suitable for wearable use.

The LiPo battery is integrated with an on-board charging circuit, enabling convenient recharging through a dedicated charging module on the PCB. This charging system regulates the battery safely and efficiently, ensuring proper voltage and current control in accordance with lithium battery safety requirements.

Additionally, a USB-C port is incorporated into the design to serve dual functionality. It provides power input for the battery charging circuit and also enables programming and debugging of the ESP32 microcontroller. This eliminates the need for separate connectors and simplifies both user interaction and development.

Voltage regulation is implemented to ensure stable operation of all components, particularly the microcontroller and IMU, which require consistent supply levels. The overall power design prioritizes efficiency, safety, and ease of use, making the system suitable for continuous overnight operation while maintaining reliability and user comfort.

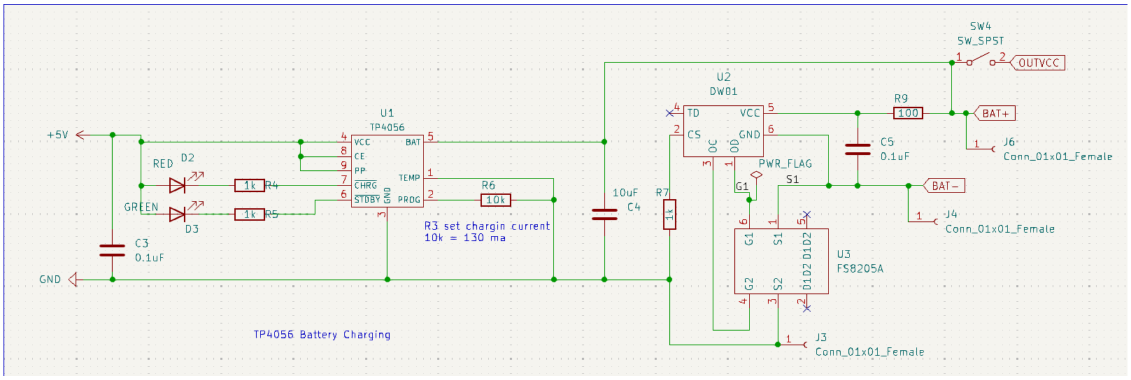

Figure 7: Power Management Schematic

2.3 Software Design Details

The Software performs the following tasks:

- I2C communication with the IMU
- Orientation data processing
- Threshold-based posture detection
- PWM control of the vibration motor
- Recognition of User Inputs and Outputs

We have met these requirements by defining the general behavior of our device using an FSM and then implementing said behavior by programming our MCU and sending control signals to the proper devices on our PCB.

2.3.1 Software Behavior

Our device uses the following FSM to determine its behavior.

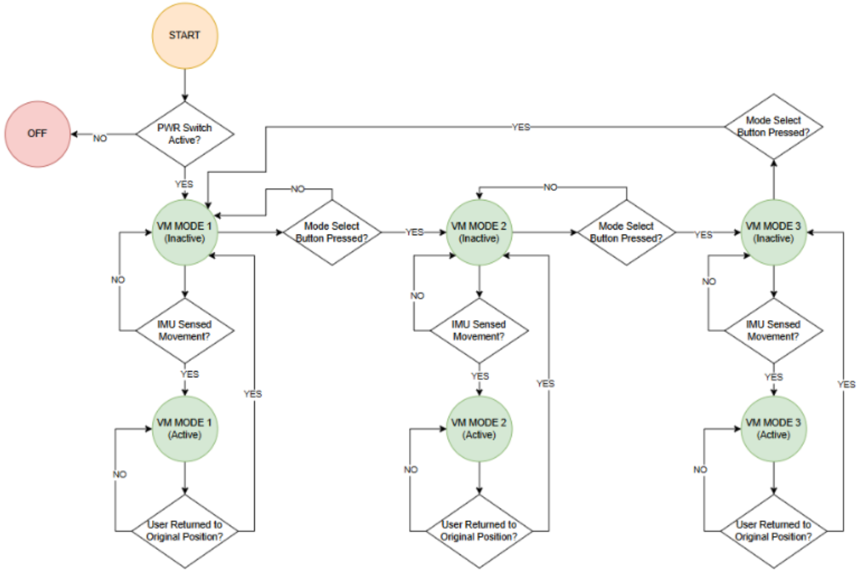

Figure 8: Finite State Machine

This FSM shows that the Power Switch must be active for our device to run. It then shows the 3 different Vibration Motor Modes, which can be cycled through using the Mode Select button. The primary comparison used to trigger the vibration is whether or not the user has changed their position. Doing so will cause the FSM to transition from an ”Inactive” to an ”Active” state.

2.3.2 Software Development

The Software aspect of this project involves developing a program that will implement the behavior described by our FSM. To test and verify these programs, we used Arduino IDE to upload our sketches to our MCU, which would then communicate with all the relevant devices, inputs, and outputs.

We started with a breadboard set-up to help develop our programs, since it would take some time for the PCBs to deliver. We used buttons, switched, and LEDs to mimic the IO on the sleep position trainer. We also used an ECE 110 motor in place of the Vibration Motor for testing. For the MCU, we used an ESP32-S3-WROOM-1, development board (same as our MCU on the PCB), and utilized the same GPIO ports as on our PCB to make sure our program would work for both the breadboard set-up and PCB set-up seamlessly.

2.3.3 IMU Communication

The IMU is the most complex aspect of our project. Not only did it require a Level Shifter Chip to help convert the MCU’s 3.3v to 1.8v, but it also required driver code to initialize itself before we could start reading data.

We chose to use the BHI360, which is a low-power IMU that has multiple measurement sensors and its own programmable MCU. We decided to communicate with this device using I2C. It took some time to figure out how to initialize the BHI360 in our Arduino sketch, but fortunately there were some existing repositories and libraries (Referencing SensorLib [7]) that we used to help communicate with our IMU. After that, we were able to pull data from the IMU, and successfully read it's Euler Angles, which described the orientation of our device and allowed us to determine when to trigger the vibration motor.

3 Verification

3.1 Testing Procedure

To efficiently debug the system and verify that each subsystem functioned properly, we implemented a unit testing approach. We began by testing the ESP32 development board using simple components such as LEDs and a motor to ensure basic functionality. Once the microcontroller and ERM vibration motor subsystems were operating correctly, we progressed to integrating and testing the IMU.

The IMU proved to be significantly more complex than the other subsystems, requiring extensive debugging and calibration. A large portion of development time was spent ensuring proper communication, interpreting sensor data, and obtaining accurate orientation values. Through iterative testing and refinement, we were able to achieve reliable measurements that could be used to determine the user's position.

The system was tested to verify functionality and performance. Testing included:

- Testing vibration motor activation under different conditions
- Have LEDs light up for different vibration modes
- Test battery charging module
- Verifying I2C communication between the microcontroller and IMU
- Measuring orientation accuracy

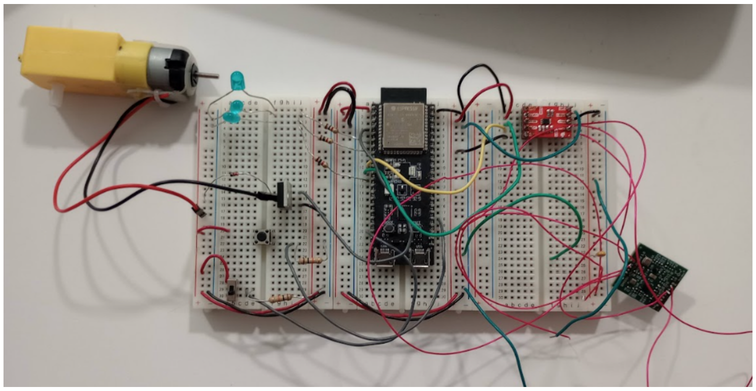

Figure 9: Breadboard Setup

3.2 Results

The system successfully detected improper sleep positions and activated the vibration motor accordingly. The response time was consistent, and the system operated reliably over extended testing periods.

Minor issues related to sensor noise and hardware connections were identified and resolved during debugging.

3.3 System Integration and Hardware Validation

Following successful subsystem testing, the complete sleep position trainer was assembled and validated through both breadboard prototyping and PCB integration. Initial development was performed using a breadboard setup to verify communication between the ESP32-S3 microcontroller, BHI360 IMU, LEDs, buttons, and ERM vibration motor before fabrication of the custom PCB. This allowed rapid debugging of firmware functionality, GPIO behavior, PWM motor control, and I2C communication reliability.

After verification of the prototype circuitry, a custom PCB was designed and manufactured to integrate all subsystems into a compact wearable platform. The PCB included the ESP32-S3 microcontroller, power regulation circuitry, IMU communication interface, battery charging circuitry, and motor driver subsystem. PCB validation included continuity testing, voltage rail verification, component solder inspection, and functional firmware upload testing.

The completed hardware system successfully demonstrated:

- Reliable I2C communication with the BHI360 IMU
- Stable 3.3 V and 1.8 V regulated power delivery
- Correct PWM control of the ERM vibration motor
- Proper LED indication for vibration mode selection
- Successful battery charging through the USB-C interface

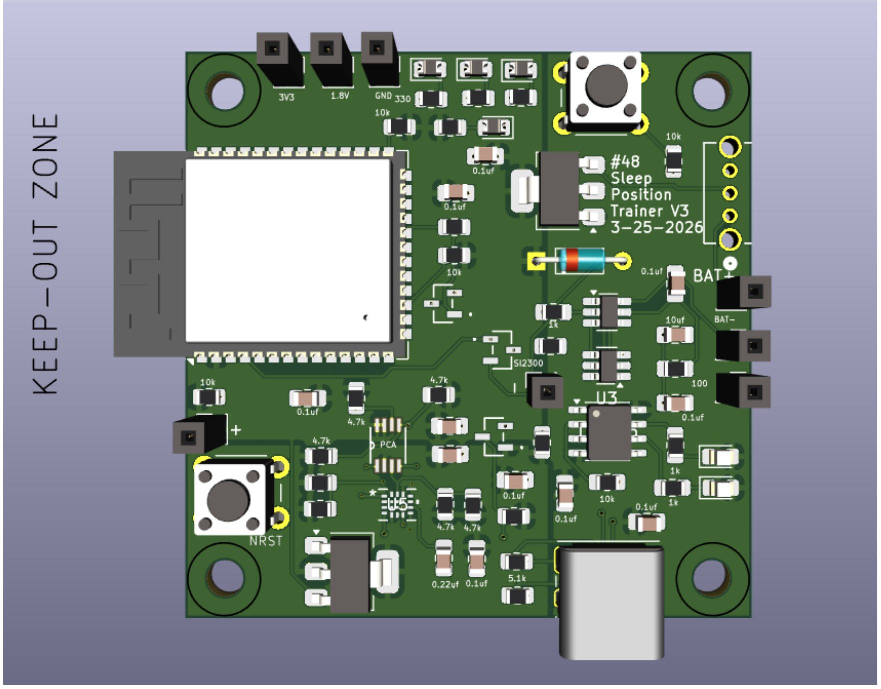

Figure 10: 3D view of the PCB

4 Cost Analysis

4.1 Consolidated Parts List

Component	**Total Quantity**	**Total Cost ($)**
PCA9306 Evaluation Board	1	5.25
LD1117-1.8 Voltage Regulator	6	0.96
BHI360 IMU	7	38.92
BHI360 Shuttle Board	1	15.88
PCA9306 Level Shifter	9	6.30
FS8205A MOSFET	6	2.04
TPB4056 Battery Charger	6	2.82
Slide Switch (SPDT)	6	3.90
100Ω Resistor	6	0.60
Vibration Motor	6	11.70
DW01 Battery Protection IC	6	0.60
Li-Po Battery (500mAh)	4	26.16
MIC5365 Voltage Regulator	3	0.51
SI2300 MOSFET	18	11.88
1N4148 Diode	3	0.30
Total		**127.82**

Table 1: Parts List and Cost from Digikey

4.2 Additional Passive Components Cost

Component	**Quantity**	**Total Cost ($)**
0.22 μF Capacitor (0603)	10	0.60
0.1 μF Capacitor (0805)	20	1.00
1 μF Capacitor (0805)	10	0.70
10 μF Capacitor (0805)	5	0.75
4.7 kΩ Resistor (0603)	30	0.60
10 kΩ Resistor (0805)	30	0.75
1 kΩ Resistor (0805)	30	0.60
330 Ω Resistor (0805)	15	0.45
LD1117-3.3 Voltage Regulator	10	3.50
ESP32-S3-WROOM Microcontroller	3	18.00
Total		**26.95**

Table 2: Additional E-Shop Components

4.3 Labor Cost

Labor cost is estimated using:

$$\text{Cost} = \text{Hourly Rate} \times \text{Hours Worked} \times 2.5$$

Assume each team member's hourly rate is $35/hour

Assume each team member works 7 hours a week, for the 15 working weeks in a semester (105 hours total)

Per member, our salary would be $35/hour x 105 hours x 2.5 = $9,187

Multiplied by 3 for each member makes $27,561 total

Therefore, our total labor cost is $27,561

4.4 Total Cost

The total cost of the parts obtained on Digikey is $127.82 (Table 1)

We will also add $26.95 as the cost of the E-Shop parts (Table 2), bringing our total to $154.77

In addition, PCB ordering on JLCPCB will cost $5, which brings our total cost to $159.77 pre-tax

*Although the ECE shop parts are free, it would be easiest to apply the tax and shipping cost to all parts to find a safe upper bound for our total cost

Assuming a 10% Sales Tax and a 5% Shipping Cost, makes our total parts cost sum to:

($159.77 x 1.15) = $183.74

Lastly, the sum of our labor cost ($27,561) and our parts cost ($183.74) which brings the total expenses needed to develop our project to:

Total Cost: $27,744.74

It is challenging to estimate the cost of mass-production by listing bulk-purchase costs because not every component from our parts list was used exclusively on the final build. We spent a good portion of our budget on development boards and replacements parts help us develop our final product.

After doing some math, the cost of a single device reaches roughly $30, but this value is heavily estimated.

4.5 Schedule

The schedule we followed to develop this project is in Appendix A, section A.2.

Throughout the development of this project, we have followed this schedule closely. This planning has allowed us to stay on top of our tasks as we built the Sleep Position Trainer.

5 Conclusion

5.1 Accomplishments

This project successfully demonstrates a functional sleep position trainer, capable of monitoring posture and providing real-time feedback. By providing real-time posture correction, our device promotes better sleeping habits and reduces the effects of sleep-related conditions.

The sleep position trainer addresses sleep conditions that affect people of all demographics and economic situations. By offering an affordable alternative to typical sleep monitoring devices, we can provide help to those that cannot afford professional sleep therapy treatment. In addition, from an environmental standpoint our device prevents waste by using a rechargeable LiPo battery, which increases the effective use time of our trainer. To summarize, our device's simple design and low component cost imply the potential for widespread marketability and reduced manufacturing expenses.

5.2 Uncertainties and Future Improvements

Although we had some issues while developing the PCB, we solved the issues before the final deadline. These issues mainly related too components not behaving as expected, or not facilitating proper communication between the MCU and IMU. A major issue we had was our 3.3v to 1.8v LDO was not working, so we replaced it with a different footprint in later PCB version, which solved the issue.

To improve the performance of our device, we could add new sensors, such as a heart rate monitor, to track how our device affects sleep quality throughout the night. We could also integrate a mobile application into our trainer's interface to make it easier to switch modes or enable more advanced posture detection algorithms.

5.3 Ethical Considerations

Ethical Consideration Explanations:

- ACM 1.2 – This code discusses the safety of our design. We will test for hazardous scenarios (long runtime, repeated vibrations, charging behavior) and implement methods to prevent problems that could arise from these cases. [3]
- ACM 1.3 – We will not claim any guaranteed health or treatment outcomes for our device, unless extensive and justifiable evidence is found. [3]
- ACM 1.6 – If the system stores or transmits any sleep-position data (even just timestamps or posture labels), we will minimize data collection, store it locally when possible, and avoid collecting identifying information. [3]
- IEC 62133-2 – safety requirements and tests for portable sealed rechargeable lithium cells/batteries, including foreseeable misuse. [2]

- UN 38.3 transportation testing – relevant if shipping batteries or a finished product, and required for lithium batteries offered for transport. [4]

Safety Risk Explanations:

- Battery overheating, swelling, fire, or burns – Lithium-ion batteries can overheat and ignite if charging is not properly designed. [2]
- Skin irritation – This wearable can cause irritation if not properly cleaned or worn.
- Excessive vibration – Excessive vibrations could reduce sleep quality.
- Electrical hazards – Shorts or incorrect connections could create heat or shock risk.

6 References

[1] Bosch Sensortec, “BHI360 Programmable, low power, inertial measurement unit with integrated sensor fusion library: Datasheet,” Accessed: Feb. 28, 2026. [Online]. Available: https://www.bosch-sensortec.com/media/boschsensortec/downloads/datasheets/bst-bhi360-ds000.pdf

[2] IEC, “IEC 62133-2 Edition 1.1 2021-07: Secondary cells and batteries containing alkaline or other non-acid electrolytes—Safety requirements for portable sealed secondary cells and batteries (Lithium systems),” 2021. [Online]. Available: https://cdn.standards.iteh.ai/samples/21479/fc1425401c494ac88927998737150b9b/IEC-62133-2-2017.pdf

[3] Association for Computing Machinery, “ACM Code of Ethics and Professional Conduct,” Jun. 22, 2018. [Online]. Available: https://www.acm.org/code-of-ethics

[4] United Nations, “Lithium metal and lithium ion batteries,” in *Recommendations on the Transport of Dangerous Goods: Manual of Tests and Criteria*, 8th rev. ed., New York and Geneva, 2023, Sec. 38.3. [Online]. Available: https://unece.org

[5] Mayo Clinic, “Sleep apnea - Symptoms and causes,” Apr. 6, 2023. [Online]. Available: https://www.mayoclinic.org/diseases-conditions/sleep-apnea/symptoms-causes/syc-20377631

[6] “Does tossing and turning affect sleep quality?,” *Biology Insights*, Dec. 14, 2025. [Online]. Available: https://biologyinsights.com/does-tossing-and-turning-affect-sleep-quality/ (accessed Feb. 28, 2026).

[7] lewisxhe, “GitHub - lewisxhe/SensorLib: Commonly used I2C , SPI device multi-platform libraries,” GitHub, Apr. 11, 2026. Available: https://github.com/lewisxhe/SensorLib/tree/master

Appendix A Additional Data

Additional schematics, testing data, and extended results can be included here.

A.1 Requirements and Verification

Subsystem	**Requirement**	**Verification**
Position Sensing	The system must obtain accurate position data from the IMU with a sufficient update rate for reliable motion tracking.	Verify proper I^2C communication by confirming successful data transfer between the IMU and microcontroller using the serial monitor. Monitor real-time position data and confirm responsive changes when the device orientation changes.
Position Sensing	The system must include initialization or calibration procedures to ensure consistent orientation measurements between uses.	Confirm successful initialization by observing orientation feedback through the Arduino IDE serial monitor after startup. Verify calibration by operating the device for an extended period and checking that measurements remain stable.
User Alert System	The ERM vibration motor must alert the user when a bad sleeping position is detected.	Confirm motor driver operation by measuring the motor driver output and observing that the ERM vibration motor activates only during intended alert conditions.
User Alert System	The ERM vibration motor must support adjustable vibration strength settings.	Use a counter from 1–3 in the microcontroller firmware to control vibration strength. Confirm that each selected mode produces a different vibration intensity and that LEDs indicate the selected mode.

Subsystem	**Requirement**	**Verification**
Microcontroller	The microcontroller must obtain motion and orientation data from the IMU.	Display real-time IMU data through the serial monitor and confirm that the values update correctly when the device orientation changes.
Microcontroller	The microcontroller must control ERM vibration motor operation.	Verify that the firmware generates a PWM signal to the motor driver and confirm that programmed vibration settings produce corresponding changes in motor intensity.
Physical Build	The enclosure must safely hold and protect the internal components.	Inspect the assembled device to confirm that the PCB, battery, IMU, and motor are secured inside the enclosure and protected from normal handling.
Physical Build	The enclosure must withstand normal sleeping pressure without damaging internal components.	Apply a representative load to the enclosure and confirm that it does not crack, collapse, or damage the internal hardware.
Power Management	The design must maintain regulated output voltage and include low-voltage cutoff protection.	Use a multimeter to confirm stable regulated voltage rails during operation. Test reduced input voltage conditions and confirm that the device shuts down safely before battery damage occurs.
Power Management	The USB-C input must charge the battery and supply power to the board.	Measure the battery voltage over time while connected to USB-C and confirm that the voltage increases. Also verify that required voltage rails are present on the board.

Table 3: Requirements and Verification Table

A.2 Project Schedule

Week	**Task**	**Person**
1–6	Choose a project, submit RFA, submit project proposal, start PCB design, and submit design document.	Everyone
7	Design review, assemble PCB V1, submit PCB, order parts, and test programming on the ESP32.	Everyone, Brian, Nick
8	Prepare for second round of PCB orders, troubleshoot PCB, write code to control the vibration motor, and troubleshoot code.	Everyone, Nick
Spring Break	Ensure components can be programmed and continue debugging as needed.	Everyone, Nick
9	Prepare for third round of PCB orders, troubleshoot PCB, update PCB, and test component functionality with power source.	Everyone, Brian, Kyle
10	Prepare for fourth round of PCB orders and design physical enclosure using 3D modeling.	Kyle
11	Solder fourth round PCB, program IMU with microcontroller, and stream IMU data through I2C to verify gyroscope measurements.	Brian, Nick, Kyle
12	3D print enclosure, ensure enclosure connects to belt, make belt adjustable, and create buckle for strap.	Brian, Nick, Kyle
13	Add PCB and components to physical enclosure, power the device, verify function, and complete mock demo.	Everyone
14	Finalize demo requirements and coordinate demo timeline.	Everyone
15	Prepare for showcase, demo functionality, and ensure final system operation.	Everyone

Table 4: Project Schedule