Temperature Sensors for Veterans with Paralysis

Submitted To

Earl Swartzlander

Prepared By

Hao-Zhe Loo
Joonha Park
Kevin Tong
Tessa Urwin
Claudio Medina
Joon Lee
Paula Pijoan Gros

EE464 Senior Design Project
Electrical and Computer Engineering Department
University of Texas at Austin

2023-2024

CONTENTS

TABLES

FIGURES

EXECUTIVE SUMMARY

In this document, we will present an overview of the development process and assessment of a temperature monitoring application built for veterans with paralysis. The document addresses the challenges faced by the team, the solutions devised, and methods implemented in developing our device.

Firstly, in the Design Problem Segment, we'll delve into the challenges encountered while implementing temperature sensors for veterans, discussing both hardware and software obstacles. Individuals with paralysis encounter difficulties in regulating their temperature because their nerves are unable to detect temperature changes, making the individuals more prone to extreme temperature illnesses. This reduces individuals' quality of life, restricting their ability to engage in outdoor activities. The design problem of our project is to implement hardware peripherals that will be light and portable enough for individuals with paralysis to wear on outdoor activities, and software implementation that allows easy access of the temperature data, transmitted from the sensors.

Next, in Design Solutions, we'll explore the various solutions we've identified, ranging from different types of Bluetooth sensors to hardware structures and software implementations. For bluetooth sensors, we utilize Bluetooth Low Energy (BLE) technology which facilitates seamless communication between the temperature sensor and the smartphone application, ensuring efficient data transmission and connectivity. Among many sensors, we selected to use LM35DZ sensors due to their satisfying performance and economical pricing. The final hardware design features a wearable temperature sensing device that continuously monitors skin temperature. Utilizing custom produced shells to protect our device, we allow veterans to carry arduino boards and the sensors with ease. The board wirelessly transmits real-time data to a smartphone application, enabling users and caregivers to monitor temperature fluctuations and receive timely alerts. The software component provides a user-friendly interface for data visualization, alert management, and customization of temperature thresholds, ensuring ease of use and adaptability

to user preferences. Users can easily access this software by downloading an app to their mobile device via the app store, and connect it to the hardware sensors.

The Design Implementation section details the execution of hardware integration and software development. In the hardware, design decisions put performance, comfort, and pricing first while following project specifications and user requirements. We implemented the initial concept of our model in SolidWorks where we went through several cycles of refinements to finally output a compact shell we use today. This iterative approach facilitated the optimization of the design, prioritizing not only the device's functionality and efficiency but also user's comfort and convenience. Furthermore, there were modifications needed in our design because temperature sensing devices needed to be positioned on the user's body to ensure accurate readings and stable data transmission. During the iterative process, modifications on the device housing and strap were implemented to address this issue. In the software, many testings and iterative revisions were done to ensure functionality and reliability. In the front end, React Native framework was chosen for its ability to create native apps for both Android and iOS, allowing one codebase to support both biggest operating systems in mobile devices. With this feature, React Native significantly reduced the development time and resources used. In the backend framework of the software, Django library was used due to its widespread community and its flexibility. Along with its scalability and ease in rapid development, its mechanics in handling RESTful API requests allowed efficient development of the software and management of data-driven applications.

In the Test and Evaluation segment, we conducted tests in a structured manner to observe if our design implementations were working properly, culminating in the evaluation of the integrated system's functionality and performance. Insights gained from these evaluations informed improvements in the project to meet its goals effectively. The main representation of the testing can be found in the hardware implementation. In the process of choosing a temperature sensor, a small testing program was conducted for each of the temperature sensor candidates. Through the arduino program, a small code was written to observe if it was correctly reading the temperature. One, sensors' ability to read temperature was confirmed, testings such as ice bath method and

boiling water method were conducted to observe if sensors correctly read temperatures even in the extreme temperature environment.

Finally, Time and Cost Considerations, Safety and Ethical Aspects of the Design, and Recommendations sections outline the concerns on time and costs of the projects, safety and ethical problems, and recommendations for future direction of the project respectively. The budget constraints didn't cause big problems in completing this project, due to the large pool of initial budget of $1,000. By rigorously keeping track of our budget and minimizing our expenses, we managed to purchase all needed components with a plenty of surplus in the budget. There were some issues with the project due to delays of certain facets of the project, but we managed to work around these obstacles and resolved the problems as the project progressed. In ensuring safety of our device, we explored many ethical and safety boundaries concerning malfunction or misuse of our device. To avoid safety threats such as explosion of batteries, overheating wires, and toxic materials in our device, we carefully researched and selected components of our device to ensure their safety and resistance to extreme temperature conditions. In the recommendation section, we explore ways to enhance functionality of our device in which we centered around finding a more compact way to implement our case or changing the material in fabricating our device. With increased comfort and convenience that comes with a smaller device size, the users will have better experiences using the temperature sensor application.

This document offers a detailed description of the project's development, from identifying challenges to implementing solutions and conducting evaluations, aiming to provide valuable insights into the temperature monitoring systems for individuals with paralysis.

1.0 INTRODUCTION

This document thoroughly explores the intricate aspects of the design challenge, initiative specifications, and project management integral to the "Temperature Sensors for Veterans with Paralysis" project. Representing a collaborative venture between Adaptive Adventures and BAE Systems, the initiative is singularly focused on enhancing the quality of life for veterans grappling with traumatic injuries, particularly those dealing with limb injuries or paralysis and navigating the complexities of diminished sensitivity to temperature-related signals. The envisioned wearable temperature sensor aims to empower these veterans with comprehensive insights into their body status, facilitating their participation in extracurricular activities such as skiing.

The Design Problem segment meticulously examines the challenges inherent in hardware and software aspects, delving into the intricacies of implementation, providing insight into the rationale behind hardware component selection, and elucidating the hurdles encountered in software development. Furthermore, this section elaborates on project specifications, intricately detailing anticipated deliverables.

In the Design Solutions section, we explore potential remedies to our design problems, spanning a spectrum of Bluetooth sensors, hardware configurations, and software applications. Our mission to create a device with efficiency and seamless connectivity leads us to a meticulously engineered system to perpetually monitor skin temperature. Custom-designed shells not only safeguard our device but also facilitate its portability.

In the Design Implementation section, our focus centers on hardware integration and software finesse, where performance, comfort, and cost-efficiency is prioritized in the pursuit of project objectives and user satisfaction. SolidWorks became our tool for initial modeling, guiding us through iterative refinement cycles that built a compact and functional shell design.

In the Test and Evaluation arena, we utilized methodologies to ensure the robustness and efficacy of our design implementations. Rigorous testing, particularly in hardware integration, yields

invaluable insights, steering us ever closer to project goals. Notably, in selection of temperature sensors, we subjected sensors to rigorous small-scale testing programs.

The Conclusion section revisits each preceding segment, providing a retrospective overview of achieved milestones and insights gained throughout the process. This ensures a comprehensive understanding of the project's evolution and underscores the significant strides made in addressing the unique challenges faced by veterans with paralysis.

2.0 DESIGN PROBLEM

The problem in our mission stems from the inability of veterans with paralysis to properly sense their body conditions. Many people around the world seek excitement and adventure through outdoor activities, despite the threat of extreme temperature conditions such as hypothermia and hyperthermia. The nerves in our limbs and body are designed to monitor our internal temperature and alert us before these illnesses can occur. Unfortunately, individuals afflicted with paralysis experience many difficulties in temperature regulation, also referred to as thermoregulation [1]. Since their nerves cannot detect changes in temperature, their bodies will not perform autonomous functions such as sweating or shivering to regulate body temperature [1]. Since those with paralysis are more susceptible to extreme temperature illnesses [2], they are often advised to avoid extreme temperatures [1], limiting their ability and/or desire to engage in essential tasks and outdoor hobbies.

Adaptive Adventures, a non-profit organization that provides outdoor sports opportunities to improve quality of life for veterans with disabilities, has partnered with BAE Systems and the University of Texas to engineer a device that can monitor the temperature of the wearer in order to prevent conditions caused by extreme temperature and allow veterans with paralysis to enjoy spending time outdoors. During these activities, Adaptive Adventures will pair the veterans with an instructor that monitors their safety. The goal of the device is to alert the instructor and veteran when their limb or body temperature reaches an unsafe level.

Adaptive Adventures has commissioned this project to other companies in the past. A group of seniors from Iowa State University built a prototype of this device for their capstone design project [3]. While their product meets all the requirements, we plan to build upon theirs by implementing a design that is more accurate, resilient, and compact. Furthermore, our design will be operational in many more cases, such as camping to gardening, rather than just extreme cases.

2.1 Hardware Specifications

The user must be able to comfortably wear the device and still perform their activities unimpaired. Since this device will be used for recreational activities such as outdoor sports and events, the user must not be hindered by the worn device. For this we will implement some sort of band or bracelet that can be comfortably worn during outdoor activities. The chosen material for the prototype is elastic webbing, a common component used to wear headlights and heart rate monitors. Furthermore, we're making extensive research on the plausibility of using 3D printers in creating a shell for the temperature sensor granting the user an ability to wear the device..

The device must accurately measure the wearer's temperature and alert the user if the temperature exceeds or drops below a certain threshold. For this functionality, we will search various electronic component distributors such as Digikey or Mouser to find our ideal temperature sensor. There are a variety of potential temperature sensors such as the TMP114DIYMTR from Texas Instruments or the TSD3055 from Measurement Specialties. So far sensors LM35 and LM35H have shown successful results with the testing, although further testing will be conducted to test more vast temperature before finally building our prototype. We will attach the sensor to the band in a way that it will measure the body/limb temperature of the wearer through skin contact. The user will wear the device on their wrist or leg, and underneath their clothes so the sensor should be able to accurately measure the user's temperature.

Table 1: Hardware Input

Body temperature readings into the device sensor via skin contact

Data	Unit
Body temperature value	Fahrenheit

The device must be able to transmit temperature data in real time to a smartphone application via bluetooth. After extensive research we have decided to use the Arduino Nano 33 BLE for our bluetooth communication module. This component was chosen because it's small enough to wear and can connect with ideally both Android and IOS (prioritizing Android).

Table 2: Hardware Output

Real-time body temperature value sent from the device sensor to the application software

Data	Unit
Digitized body temperature value	Float

The device must be able to be cleaned properly so that it can be reused by the veterans. We will research what materials are easy to clean and wipe down and use it to build the band and the mold that will contain the electronics.

The device must be resilient to certain variables prevalent in outdoor activities so that it does not break during its use. Some of these variables include water (either from snow or sweat), light to moderate impact, and sudden/harsh movements. Therefore, the electronic components must be placed in a carefully designed enclosure resistant to impact, water, and other variables.

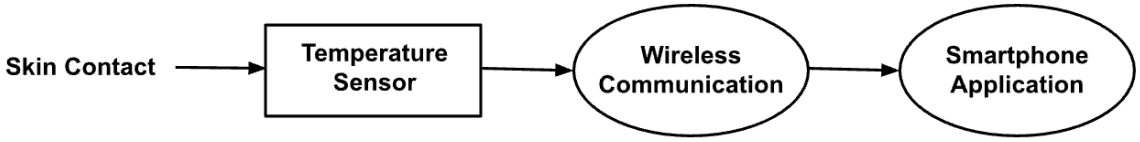

Figure 1: Hardware Data Flow

2.2 Software Specifications

The android application must connect with the Arduino UNO WiFi REV2 through bluetooth using the Arduino Nano 33 BLE to store and receive the real-time body temperature of the wearer

Table 3: Real-time Device Information

Real-time body temperature value received from the device sensor

Data	Unit
Digitized body temperature value	Float

The android application will have a user authentication page to maintain the security of clients and instructor information. The information will be stored in a PostgreSql database.

Table 4: Authentication

These information is required to login to and utilize the service

Input/description	Data Type
User ID	String
Password	String

The android application must store the wearer/client's personal health information to help set thresholds for alarms and detect abnormal temperatures

Table 5: Client Information

Client's personal health information that will help set threshold for alarms, and detect any abnormal temperatures

Input/description	Data Type
Client's Name	String
Client's DOB	Int
Body part of harm (where the device is equipped)	String

Threshold for normal temperature	Float
Client-Weight / Height	Float
Client’s Cell	String
Client’s Sex	String
Client’s allergies to medication	String
Client’s pre-existing health conditions	String
Emergency Contact	String

The android application must store the instructor information when required.

Table 6: Instructor Information

There is an instructor paired with each veteran, who will also be notified when abnormal temperatures are detected

Input/description	**Data Type**
Instructor’s Name	String
Instructor’s Email	String
Instructor’s Cell	String

The android application must alert the user when the temperature drops below or increases past a certain threshold. Once the application deems the temperature to be hazardous, it will alert both the wearer and the instructor through auditory and tactile alarms. Additionally, the application must store and display the wearer’s real-time body temperature to both the wearer and instructor.

Table 7: Instant High-temperature Alarm System

When the temperature goes over the preset threshold, it sends out alarms through the mobile app. These are the required outputs of the alarms.

Output/description	Data Type
Alert Message to Client	String
Alert Message to Instructor	String
Body Temperature	Float
Alert Time	datetime

The application will have a dashboard that displays important information that the device wearer and instructor can easily access.

Table 8: Dashboard

The dashboard of the mobile app contains the following data

Output/description	Data Type
Current Body Part Temperature	Float
List of recent alerts	String
Difference between current temperature and the avg. temperature of previous week	Float
Client's Name	String
Client's Age	Int
Client-Weight / Height	Float
Body part of harm (where the device is equipped)	String
Instructor's Name	String

Output/description	Data Type
Current Body Part Temperature	Float
Instructor's Cell	String

The android application will send out weekly reports to update the user and instructor on the user's health and activity over the past week.

Table 9: Weekly Report

The app will send out weekly reports based on the temperature history of the past week, which include the following information

Output/description	Data Type
Number of high-temp warning occurrences	Int
Highest temperature	Float
Lowest temperature	Float
Average temperature	Float
Temperature Line Graph	Img
Client's Name	String
Client's Age	Int
Client-Weight / Height	Float
Client's Sex	String
Body part of harm (where the device is equipped)	String

The application must have an intuitive user interface so that both the instructor and client can easily find and access the data they want to see.

Table 10: User Interface Specifications

Specifications	Description	Type of Element
Toolbar	Toolbar is the anchor of the app. The client and the instructor will use this to navigate through the different pages of the app and see all available pages at once.	Dropdown/Container Element
Specific Menu Items	These are the items that will be going into the toolbar. Users will select menu items to direct themselves to the page of interest.	Button Element
Page Header	The home page will contain some basic information regarding the user logged into the system such as name, cell phone, email, etc. This will be located at the top of the screen.	Text Element
Main Page (Dashboard)	The dashboard will consist of a collage of elements so that the user can see a summary of everything related to themselves on one page. For	Collage of Buttons/Text Elements

	clients this may include current temperature, message alerts, etc.	
High-temperature warning	The app will send notifications to alert the users in case of high-temperature (over the provided normal threshold) detection.	Notification/alarm/text
User Information Page	This page will allow the user to check and update their own information in case anything has changed. Things like new body parts needing to be monitored, date of birth, cell phone number, etc can all be changed on this page.	Collage of input text boxes + save button
User Preference Settings	This will be implemented as a button that will open up a preference window where users can edit app settings such as dark mode, font size, etc.	Button + Modal

The software operating system will be configured to run on android. Once the specifications are met, we will expand it to run on IOS as well.

Table 11:

Operating Environment Specifications

Specifications	Description
Python Django DRF	For operating the backend server of the app
React Native	For operating the client-side of the app
PostgreSql	For operating the database
AWS EC2	For hosting the backend server
AWS RDS	For hosting the database server

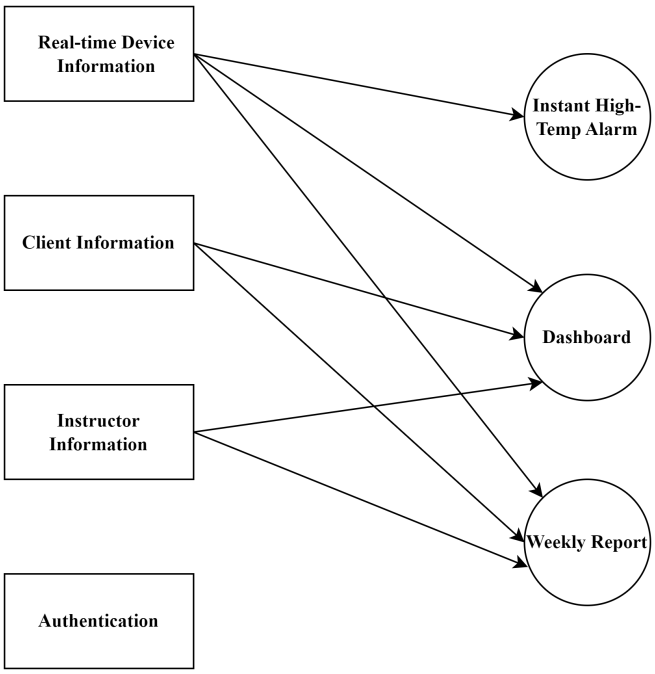

Figure 2: Data to App Functions

3.0 DESIGN SOLUTION

3.1 Hardware Design Solution

The final design solution for the hardware aspect of the temperature sensing device for veterans with paralysis provides an innovative and efficient means of monitoring skin temperature in real time. The device is intended to be worn comfortably by the end-user on their chosen location on the body, typically around extremities or other areas of concern.

The design features a temperature sensing device that makes direct contact with the user's skin to capture temperature readings continuously. These readings are then wirelessly transmitted to a smartphone application, providing real-time updates that can be accessed by the user and their instructor. The application allows for monitoring, analysis, and alerts based on predefined temperature thresholds, ensuring the user is aware of potential temperature-related hazards.

The system flow chart outlines the seamless process through which the temperature sensing device operates, from its initial contact with the user's skin to the delivery of data to a smartphone application. The process begins with the temperature sensing device making direct contact with the user's skin at the chosen location on the body, enabling continuous, real-time temperature monitoring. The device's temperature sensor, such as the LM35DZ, accurately measures the skin's temperature and sends the data to the microcontroller for further processing.

The microcontroller, in this case the Arduino NANO 33 BLE, acts as the central processing unit, preparing the data for wireless transmission. The processed data is then transmitted wirelessly to the user's smartphone application using Bluetooth Low Energy (BLE) technology facilitated by the Arduino Nano 33 BLE. The smartphone application receives the data in real time, displaying the temperature readings and providing customizable options for setting temperature thresholds. If the temperature exceeds the set thresholds, the application sends alerts and notifications to the user, ensuring their safety by informing them of potential temperature-related hazards. This system flow chart illustrates the device's efficient operation and its capability to deliver timely alerts to the user, enhancing the overall user experience and ensuring the well-being of veterans with paralysis.

To illustrate this idea, the hardware component block diagram provides a visual representation of the temperature sensing device's configuration, with the key hardware components and their interconnections.

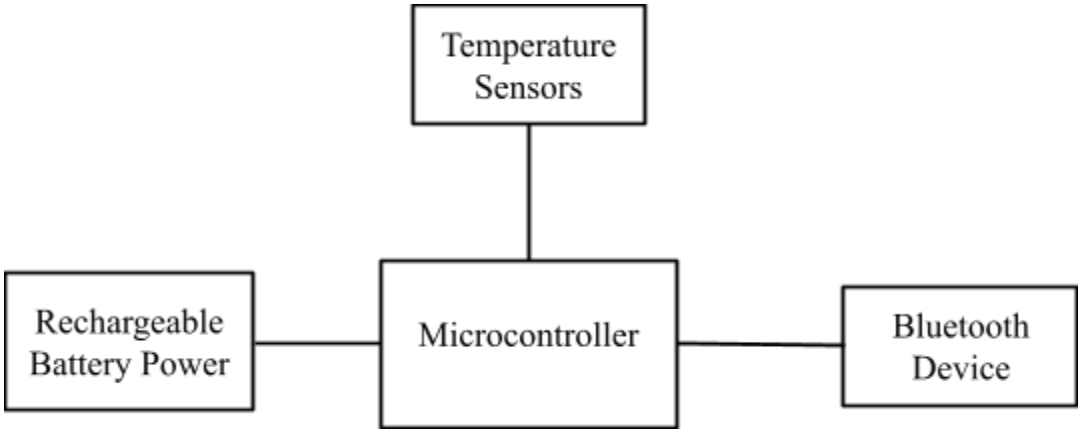

Figure 3: Hardware Components Block Diagram

Therefore, the final hardware design aims to deliver reliable performance and comfort for the user. The device is contained within a wearable, durable enclosure that integrates a temperature sensor, microcontroller, wireless communication module, and power supply.

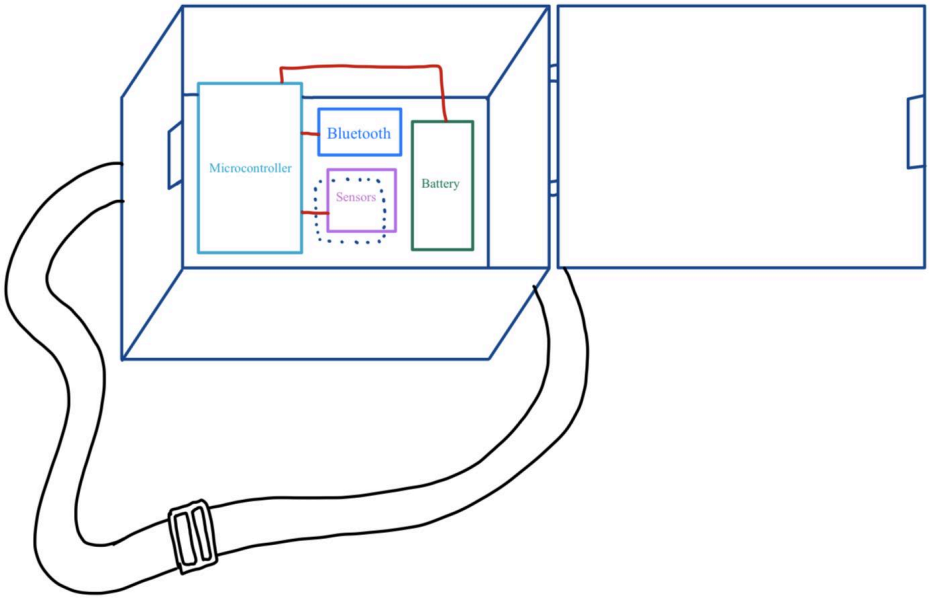

Figure 4: Drawing of Prototype Hardware Unit in Casing

3.1.1 Device Operation

The operation of the temperature sensing device is a carefully orchestrated process that integrates various subsystems to provide a seamless user experience and ensure accurate temperature monitoring. This section outlines the workflow of the device, detailing how it measures skin temperature and communicates the data to a smartphone application for further analysis and alerts. Through direct skin contact, real-time temperature sensing, wireless data transmission, and user-friendly interface integration, the device functions effectively to serve its primary purpose: assisting veterans with paralysis in safely monitoring their body temperature.

- ***Temperature Monitoring:*** The temperature sensor measures the skin temperature of the user in real time. This data is processed by the microcontroller and transmitted wirelessly to the smartphone application.
- ***Wireless Communication:*** Utilizing Bluetooth Low Energy (BLE) technology, the device communicates with the user's smartphone application, allowing for seamless and energy-efficient data transmission.
- ***Alerts and Notifications:*** The smartphone application can provide notifications and alerts based on customizable temperature thresholds, keeping the user informed of any potential dangers.
- ***User Comfort:*** The device is designed for comfort, with an adjustable strap for customizable positioning and an ergonomic enclosure to prevent discomfort during prolonged use.

3.1.2 Hardware Components

The hardware components of the temperature sensing device play a crucial role in the overall functionality and performance of the system. Hence, this section provides an in-depth examination of the key hardware elements, including their specifications, roles, and how they interact to deliver the desired results. The hardware components have been carefully selected and integrated to ensure reliable temperature monitoring, efficient power usage, and effective wireless communication. By understanding the specific hardware components and their contributions to

the system, we can gain insight into how the device operates and how it fulfills its intended purpose.

Microcontroller and Bluetooth Module

The Arduino NANO 33 BLE serves as the central processing unit for the device, handling data computation and communication with the smartphone application. The Arduino Nano 33 BLE was chosen for this project over other options, such as the standard Arduino, due to its advanced Bluetooth Low Energy (BLE) capabilities and compact form factor. The BLE functionality allows for efficient and low-powered wireless communication between the wearable temperature sensor and the smartphone application, which is a critical aspect of the project's requirements. By leveraging BLE technology, the device can establish reliable connections while minimizing energy consumption, thereby prolonging battery life.

Additionally, the Arduino Nano 33 BLE's small size makes it an ideal choice for wearable applications, as it can be discreetly integrated into the device without causing discomfort to the user. This compact design aligns with the project's emphasis on user comfort and ensures that the device remains non-intrusive while worn on the body. The combination of its BLE capabilities and small form factor allows the Arduino Nano 33 BLE to meet the specifications required by veterans with paralysis, providing them with a comfortable and efficient temperature sensing solution.

Temperature Sensor

The selection of the appropriate temperature sensor is a critical aspect of the project, as it directly impacts the accuracy and reliability of temperature measurements for veterans with paralysis. A broad array of temperature sensor options is available on the market, each offering distinct specifications and performance attributes. The project team will conduct rigorous testing and evaluation of a range of sensors, examining their performance under controlled and varying environmental conditions, including factors such as precision, response time, and compatibility with the device. This systematic approach is intended to identify the most suitable sensor that

aligns with the specified requirements and standards for the final design. By selecting an optimal sensor, the project ensures the delivery of precise temperature data for real-time monitoring and timely alerts, thereby contributing to the safety and well-being of the veterans served.

Power Supply

A rechargeable battery powers the device, supporting extended usage and facilitating user convenience. We will select a battery that adequately powers the Arduino microcontroller and other connected components, ensuring consistent and reliable performance across various operating conditions. This selection process involves evaluating batteries based on criteria such as capacity, voltage, and current output to meet the energy requirements of the device, as well as assessing factors like battery lifespan and safety features. After continuous research and analysis, we determined the best option is a 3.7 V LiPo (lithium polymer) battery, which offers the optimal balance of energy density, safety, and longevity for our application. By choosing this battery, the device will function effectively, offering uninterrupted monitoring and operation for the user throughout the day.

The hardware design solution achieves a balance between functionality, efficiency, and user comfort. The final product provides a reliable, easy-to-use, and comfortable means of monitoring and managing temperature for veterans with paralysis, addressing their unique needs and enhancing their safety and well-being.

3.2 Bluetooth Design Solution

We chose to use Bluetooth Low Energy, or BLE, as our communication method between the temperature sensor and the software application. There are two components to our bluetooth design solution: the arduino code in the microcontroller/bluetooth module and the reactNative code for the software application, which when both implemented properly, allowed the module to communicate with the software application.

3.2.1 Bluetooth Low Energy Explanation

Bluetooth Low Energy, also known as Bluetooth Smart or Bluetooth 4.0, is a wireless personal area network technology designed for low power and low cost communication between devices. While traditional bluetooth, now called Bluetooth Classic, is based on asynchronous serial communication, Bluetooth Low Energy operates through peripheral devices and central devices. Essentially, the peripheral device acts as the server while the central devices are the clients. The peripheral device will notify data and the central devices will read the data. In our case, our bluetooth module is the peripheral device because it is the one sending data (temperature) while the phone, which contains the software application, is the central device because it is receiving the data.

The data sent by the peripheral device is organized into services, which is further divided into characteristics. For our product, we will only need a single characteristic which is the temperature. Furthermore, these services and characteristics are identified by a unique 16-bit or 128-bit code called the UUID. These UUIDs can be manually assigned during the coding process.

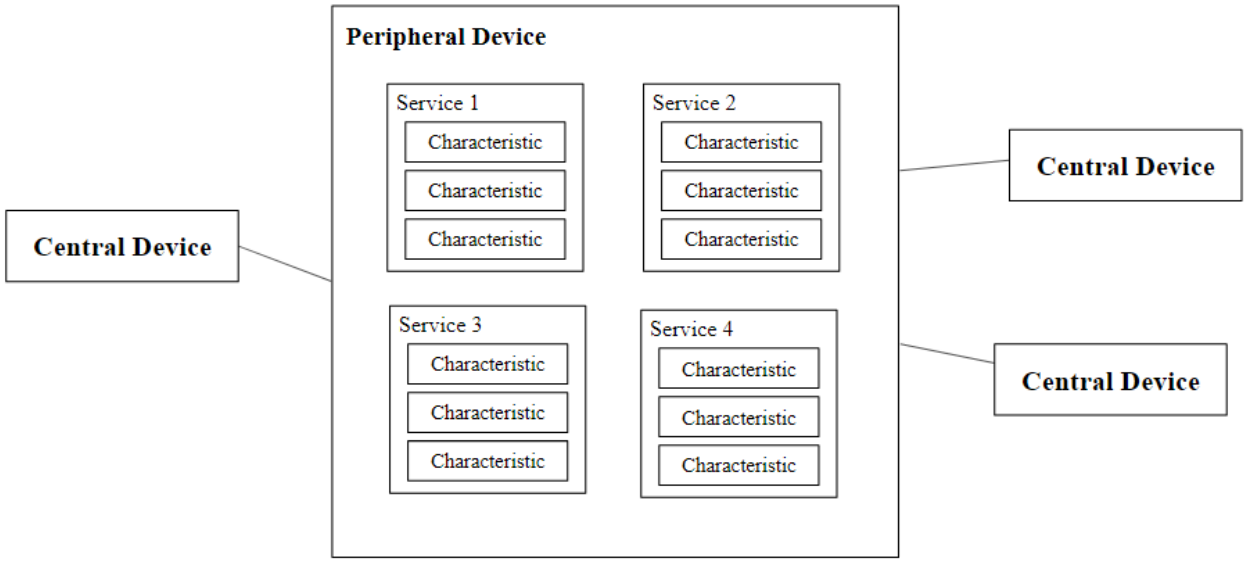

Figure 5: Diagram Showcasing BLE Structure

3.2.2 Microcontroller Bluetooth Design Solution

In order to use the Bluetooth Low Energy using the Arduino Nano, I utilized the given library, ArduinoBLE.h which contains a variety of classes and functions useful to our design solution. Using the classes BLEService and BLEIntCharacteristic and randomly generated UUIDs, I created a new service and an integer characteristic with read and notify properties. Then, using the provided functions, I added them to the data and started advertising. In the main function, and if statement checks if the central device is connected before starting the program. Once the central device is connected, the program will read the data from the A6 pin (which should be connected to the temperature sensors) on the microcontroller and update the characteristics. Whenever the characteristic is updated, it will notify the connected central device. The process of reading and updating the temperature value will be performed every 1000 milliseconds. The arduino code, along with comments explaining in more detail, is available in Appendix B.

3.2.3 Software Bluetooth Design Solution

We made the communication between Arduino and the React Native application possible by incorporating the react-native-ble-plx library in the React code. The library provides the functionality to bridge the communication between hardware and software. To achieve this we utilized the BLEManager class which is the mother class that handles all BLE operations such as scanning for devices, connecting to the devices, and transferring data between the mediums. To connect to the arduino, the scanAndConnect function handled all logic regarding connecting to a specific device, in this case, “tempSensor”. To read what the arduino is sending, we will need to read a characteristic value that is encoded in base64 in celsius. This value is originally read in hexadecimal. Therefore, to read the exact celsius temperature, we need to convert this value from hex to a float and ensure it reads enough bits to fit all ranges of our temperature number.

3.3 Software Design Solution

The software component of our temperature sensing system plays a critical role in processing and visualizing the data collected by the hardware. Designed to be both robust and user-friendly, it ensures that users can easily monitor their skin temperature, receive timely alerts, and adjust settings according to their specific needs.

3.3.1 Architecture

The software system is structured in a layered architecture to ensure scalability, maintainability, and robust functionality:

Presentation Layer

This layer includes the user interface and user experience components developed using React Native. It provides veterans with an intuitive interface to interact with the application on their smartphones.

Business Logic Layer

Business Logic Layer: Hosted on a Django backend server, this layer processes data, manages user sessions, and handles business rules such as alert conditions and data aggregation.

Data Access Layer

This includes the database management system where user data, temperature readings, and configuration settings are stored securely. It ensures data integrity and fast access to historical data.

Integration Layer

Facilitates communication between the frontend application and the backend server via RESTful APIs, ensuring that data is seamlessly transmitted to and from the user interface.

3.3.2 System Components

Mobile Application (Frontend)

- User Interface:
    - Dashboard: Displays real-time temperature data with graphical views for historical trends.
    - Alerts Module: Notifies users via the app if temperature thresholds are exceeded
- Data Handling:
    - Data Fetching: Retrieves temperature data from the backend in real time.
    - Data Submission: Sends configuration changes to the backend, such as updated thresholds.

Backend Server

- API Endpoints:

    - Authentication API: Manages user registration, login, and security.
    - Data API: Handles requests for current and historical temperature data.
    - Settings API: Processes changes to user settings and thresholds.
- Business Logic:
    - Temperature Monitoring: Analyzes incoming data to detect anomalies.
    - Alert System: Implements logic for sending alerts based on user-defined criteria.
- Database Management:
    - User Profiles: Stores user information and preferences.
    - Temperature Records: Keeps a log of all temperature readings for analysis and reporting.

3.3.3 Device Operation

Integration and Workflow

- Temperature Data Collection: The backend periodically receives data from the temperature sensor via Bluetooth, which is processed and stored.
- Real-Time Data Processing: Data is analyzed in real-time to detect any deviations beyond the set thresholds. Alerts are generated and pushed to the user's mobile device instantly.
- User Interaction: Users can view real-time data, receive notifications, and adjust settings through the mobile application, enhancing their control over the monitoring process.
- Alert Generation: If anomalies are detected, the server sends an alert signal to the mobile app, triggering a notification on the user's device.

3.3.4 Technologies Used

- React Native: Enables the development of the mobile application with native performance.
- Django DRF: Serves as the robust backend platform managing logic, user sessions, and API endpoints.
- PostgreSQL: Acts as the relational database system due to its performance and reliability.
- Bluetooth Low Energy (BLE): Used for efficient wireless communication between the hardware device and the mobile application.

Conclusion

The software design of the temperature monitoring system meets the critical needs of veterans with paralysis by providing a comprehensive, user-friendly, and reliable solution for temperature monitoring. The system's architecture and operational logic are grounded in robust theoretical principles, ensuring effective performance and user safety. This well-documented solution not only fulfills the project's specifications but also provides a scalable framework for future enhancements.

4.0 DESIGN IMPLEMENTATION

4.1 Hardware Design Implementation

In the implementation phase of the design solution, several important factors guided our approach to create a temperature sensing device for veterans with paralysis. Throughout the process, design decisions were influenced by the need to balance performance, comfort, and cost-effectiveness while meeting the project's specifications and the unique requirements of our target users.

4.1.1 Design Decisions and Changes

The design and development of the temperature sensing device demanded meticulous planning and numerous adjustments to achieve the target objectives and performance standards. Starting with an initial concept modeled in SolidWorks, we engaged in an iterative design process that included several cycles of refinement to accommodate the components within a compact shell. Our aim was to create a device that was both minimal in size and attuned to the requirements of veterans with paralysis. This iterative approach facilitated the optimization of the design, prioritizing not only the device's functionality and efficiency but also the comfort and convenience of the user. Through careful fine-tuning, we were able to produce a device that effectively balanced performance and user-centric design. This iterative process allowed us to optimize the design for functionality, efficiency, and user comfort.

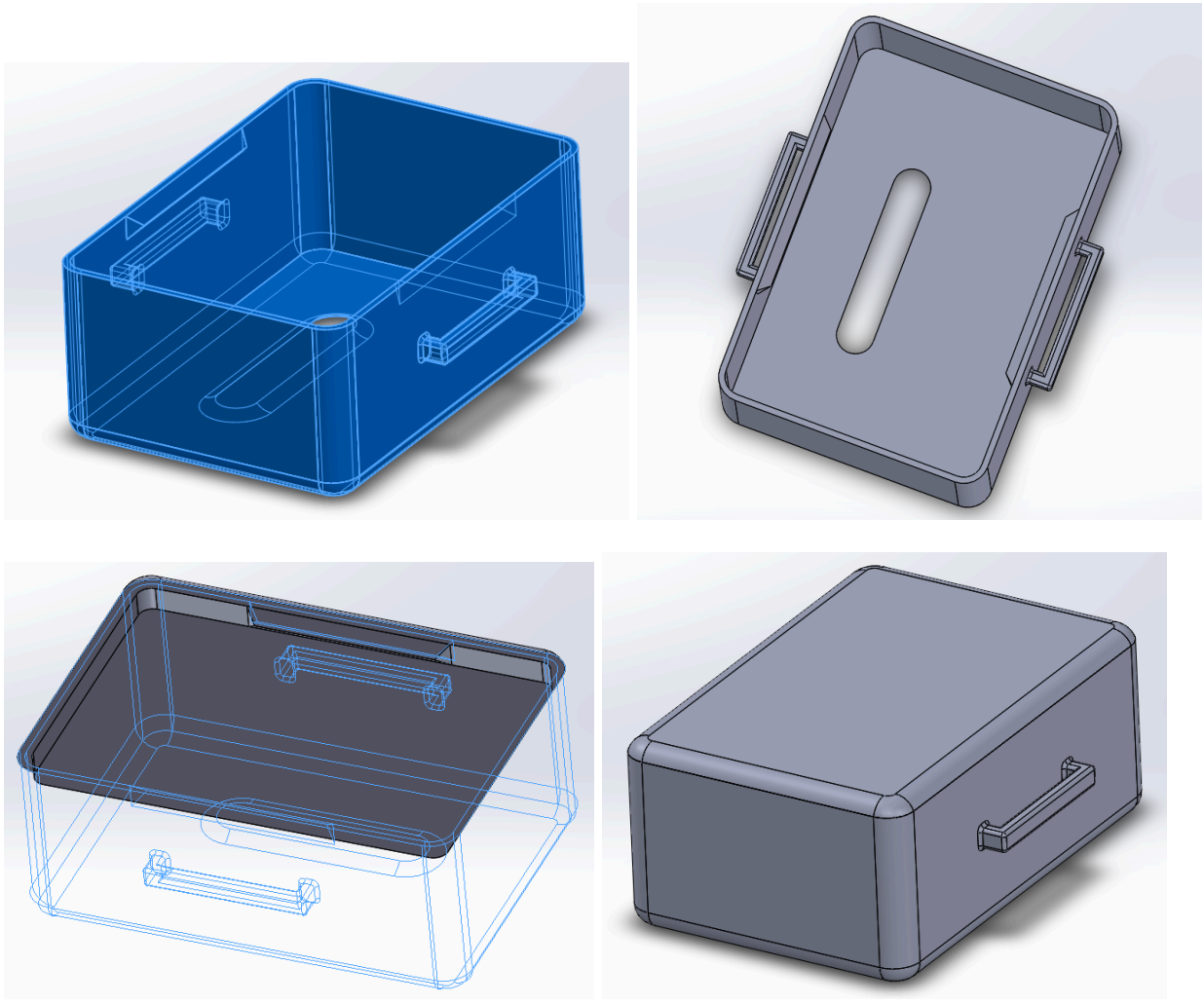

Figure 6: 3D Shell to Encompass Design

Throughout the development process, a pivotal aspect involved crafting a modular housing, or "shell," to encapsulate the electronic temperature sensing components. This shell was meticulously designed to ensure comfort and wearability for users, particularly veterans with paralysis. Iterative refinement cycles were employed to perfect the shape and size of the shell, aiming for a balance between compactness and functionality. Central to this design was the accommodation of essential components such as the rechargeable battery, Arduino, and PCB of temperature sensors within the confines of the shell. Moreover, the shell featured a strategic opening to allow the temperature sensors to sit flush against the skin, enabling accurate temperature measurements during use. To materialize these designs, we experimented with different 3D printing filaments, namely PLA and PETG. After careful evaluation, PETG emerged

as the preferred choice due to its sturdiness and ability to produce a smoother surface finish, ensuring optimal comfort for the user. These iterative design efforts, coupled with material experimentation, ultimately culminated in a wearable device seen in Figure 16 that seamlessly integrated functionality, efficiency, and user comfort for our target demographic.

4.1.2 Overcoming Obstacles

Achieving a seamless integration of hardware and software components presented a significant challenge throughout the design and development process. The primary hurdle was ensuring compatibility between the microcontroller, temperature sensors, and wireless communication module. This required careful adjustments to the original design to establish a stable and reliable communication link while ensuring the accurate transmission of temperature data to the smartphone application. Iterative testing and debugging were crucial to fine-tune the system's performance and address any unforeseen issues that arose during integration.

Another complex obstacle was maintaining user comfort while ensuring the device's functionality. The temperature sensing device needed to be securely positioned on the user's body to ensure accurate readings and stable data transmission. Modifications to the device housing and strap were implemented to achieve an optimal balance between a snug, secure fit and ease of use. These adjustments included refining the design of the strap for better adjustability and comfort, as well as improving the housing to minimize any potential discomfort or irritation to the user's skin. Through this careful process of trial and error, we were able to optimize the device for both functionality and user experience.

4.1.3 Economic Analysis and Cost/Benefit Considerations

To manage project expenses while still achieving the desired accuracy and precision, we strategically employed an approach that involved initial testing and experimentation with lower-cost temperature sensors. We selected the LM35DZ sensor, priced at $1.95 per unit, as the most suitable option from the range of sensors tested. This economical choice enabled us to validate the system's design and functionality in a cost-effective manner, facilitating early-stage prototyping and refinement.

After confirming successful operation of the device using the initial set of sensors, we progressed to integrate higher-cost temperature sensors, such as the LM35AH, which are priced at approximately $40 per unit. These advanced sensors offer enhanced precision and accuracy, aligning with the project's stringent design specifications. This shift to higher-quality sensors allowed us to significantly improve the system's performance while maintaining a careful balance with budgetary constraints. Through this calculated approach, we optimized the project for superior accuracy and precision without compromising cost-effectiveness.

4.1.4 Incorporating Veteran FeedBack

Despite the challenges of time constraints, which prevented us from physically delivering the device to the veterans in time to gather direct feedback, we established a consistent line of communication through weekly Zoom calls every Thursday. These calls provided us with valuable opportunities to engage with the veterans, enabling us to collect their input and suggestions on design decisions and options. Their insights significantly influenced the evolution of our final design and solution, allowing us to tailor the device to better align with the specific needs and expectations of the end-users. This continuous collaboration ensured that the final product not only met but exceeded the anticipated requirements, providing a more user-friendly and effective solution for the veterans.

4.1.5 Performance Testing and Evaluation

The performance testing and evaluation process played a critical role in ensuring that the temperature sensing device met the required specifications and operated reliably in a variety of scenarios. This process involved rigorous testing of the device under different conditions to confirm its accuracy, response time, and overall functionality.

Initially, we conducted testing with inexpensive temperature sensors to validate the core functionality of the system at a lower cost. The primary focus was on verifying the accuracy and precision of the temperature measurements, as well as the reliability of wireless communication between the device and the smartphone application.

As the design progressed and demonstrated successful operation, we introduced higher-cost temperature sensors to improve the accuracy and precision of the system. These sensors were subjected to a series of tests, including accuracy testing, response time testing and reliability testing, all of which are further explored in the test and evaluation section.

In addition to sensor-specific tests, we conducted integration testing to assess the seamless communication between the temperature sensing device and the smartphone application. This involved verifying data transmission, compatibility, and real-time updates on the app.

Throughout the testing process, we documented the results and analyzed the data to identify any areas of improvement. This iterative approach allowed us to refine the device and make necessary adjustments to achieve optimal performance.

Overall, by combining rigorous performance testing and evaluation with feedback from veterans during weekly Zoom calls, we ensured that the final design met the required standards and provided a reliable, accurate, and user-friendly solution for veterans with paralysis.

4.2 Bluetooth Configuration

We chose to use bluetooth as our communication method between the temperature sensor and the software application because it is stable, reliable, and independent of location. Since our client will use our product for extreme sports such as skiing and camping, we had to consider that they would be in areas that lacked service and internet connectivity. Therefore, bluetooth would be the best possible solution to the communication problem. We specifically decided to go with Bluetooth Low Energy over Bluetooth Classic due to its lower power consumption and smaller data transfer rates. We wanted the temperature to be updated at most once every two seconds which is way slower than the transfer rate limit of BLE. Additionally, the lower power consumption is another important factor in our decision because we wanted the device to be operable for a long period of time.

We decided to use the Arduino Nano 33 BLE as both our microcontroller and bluetooth module due to its support for BLE, small size, and compatibility with Arduino coding language. Next, we had to figure out how to use BLE to communicate between the module and the android application. In order to properly implement our design solution, we researched how Bluetooth Low Energy worked and various Arduino and React libraries. For Arduino, they provided the ArduinoBLE.h library which contained all the classes and functions needed to write the code for the hardware device. After testing and evaluation confirmed that the module could connect and communicate with an iPhone, we had to conduct more research on integrating it with ReactNative, which is what the software application is written in. We used the react-native-ble-plx library to connect the arduino to the mobile application. This library has a class BLEManager that contains the functionality of scanning, connecting, and reading external low energy bluetooth devices. We decided to use react-native-ble-plx because of its simplicity and compatibility with connecting with BLE devices. After scanning and connecting the device we needed to decode the data we read using base64. This process includes using the react-native-base64 library to decode the data and convert our final result to a decimal from hexadecimal.

4.3 Software Design Implementation

The implementation phase of the software for the temperature monitoring system involved a series of strategic decisions, rigorous testing, and iterative revisions. This phase was critical in transforming theoretical designs into a functional and reliable application tailored to the needs of veterans with paralysis. The following sections detail the decision-making processes, design alterations, and economic considerations that shaped the final software product.

4.3.1 Decision-Making and Design Choices

Choice of Technology Stack:

- *Frontend (React Native):* Chosen for its ability to create native apps for both Android and iOS from a single codebase, significantly reducing development time and resources.

React Native's component-based architecture facilitates modular design and enhances maintainability.

- *Backend (Django with Django REST Framework):* Selected for its robustness, scalability, and ease of rapid development with a clean design for handling RESTful API requests. Django's built-in admin panel and mature ecosystem allow efficient management of data-driven applications.

Data Handling and Processing:

- *Real-time Data Streaming vs. Batch Processing:* Initially considered batch processing for temperature data; however, real-time processing was necessary for timely alerts. The data received from the temperature sensor is encoded in base64 which needs to be decoded by a base64 decoder to become readable to the human eye. We utilized react-native-base64 to decode the data.
- *Database (PostgreSQL):* Chosen for its strong consistency, reliability, and support for complex queries necessary for statistical analysis of temperature data.

4.3.2 Overcoming Obstacles and Making Modifications

- *Challenge:* Initially, the frequent data transmissions between the sensor devices and smartphones drained the device batteries quickly, which was impractical for daily use by veterans.
- *Solution:* Optimized the data transmission intervals and implemented a more efficient data aggregation algorithm on the devices. This reduced the frequency of transmissions without compromising data quality. Additionally, switched to Bluetooth Low Energy (BLE) for less power-intensive communication.

4.3.3 Economic Analysis

- Development Costs: Consider the costs of cross-platform development tools (React Native) versus native development for each platform. React Native reduced the need for separate iOS and Android teams, thereby lowering overall development costs.

- Operational Costs: Utilized cloud services (AWS) for backend operations, which provided scalability and reliability without upfront hardware investments. Costs varied with usage, allowing budget flexibility.
- Benefits: The system significantly reduces the risk of skin temperature-related health issues through continuous monitoring, potentially preventing costly medical interventions.

4.3.4 Design Innovations and Adjustments

- Integration with External Hardware:
    - *Original Design:* Initially planned to use standard Bluetooth for communication.
    - *Modification:* Switched to Bluetooth Low Energy (BLE) to enhance battery efficiency and data transfer rates.

4.3.5 Prototype Construction and Final Adjustments

- Software Prototype Development:
    - *Testing and Validation:* Conducted extensive unit and integration testing using Jest and Django's test framework to ensure every component functioned correctly under various scenarios.
- Final Design Implementation:
    - *System Security:* Implemented advanced encryption for data transmission and secure authentication protocols to protect user data.

Conclusion

The design implementation phase was pivotal in crafting a software solution that not only meets the technical specifications but also addresses the practical needs of veterans with paralysis. The economic considerations balanced cost-efficiency with high functionality, resulting in a product that is both affordable and highly effective. The iterative process of testing and subsequent revisions ensured that the final product was not only innovative but also user-centric and ready for real-world application.

5.0 TEST AND EVALUATION

The Test and Evaluation section of the Senior Design Project was conducted in a structured manner, with each subgroup performing independent testing and evaluation of their respective components and subsystems. This approach allowed for detailed assessment and fine-tuning of individual elements according to the specific needs and requirements of each component. Once each subgroup completed their testing and evaluations, all components and subsystems were integrated into a cohesive system. The final stage involved testing and evaluating the entire project as a whole to determine whether it met the initial requirements and set goals. This comprehensive evaluation provided insights into the overall functionality and performance of the integrated system, ensuring that it delivered the desired outcomes for the project.

5.1 Hardware Test and Evaluation

5.1.1 Methodology

Acceptance testing involved the evaluation of individual hardware components, including the temperature sensor to ensure that they met the predetermined specifications and requirements set out by the industry company sponsoring this Senior Design Project. Hence, a series of methodologies and tests were carried out to assess sensor accuracy, sensor response time, and reliability under controlled conditions. This means that each component was tested independently to identify any defects or deviations from the expected performance and make a final decision on design components. Testing procedures focused on verifying inter-component communication, functionality, and reliability under varying conditions. Special emphasis was placed on assessing the integration of individual components into cohesive subsystems and modules.

Initially the set of temperature sensors chosen for testing included: LM35DZ, LM35AH, LM35CZ/NOPB, LM35DT, DS, SEN TMP, SEN, BME.

For each temperature sensor, a small testing program was conducted, in which through the Arduino program, a small code was developed to see if it was able to read the temperature. This code was first checked off with room temperature, to ensure that both the code and component

were working, and given that the temperature of the room was known, we would be able to check off and ensure before further testing. Attached below, in Appendix A is the code that was used and implemented to test the LM35DZ temperature sensor.

Once all components were able to pass the test and measure room temperature correctly, we had to adapt the circumstances to be able to test our components in “more extreme conditions” and determine the accuracy of these sensors. Additionally, other key characteristics and specifications that were initially requested within the component were tested to ensure the objectives of the project were met. This testing included:

Ice Bath Method

In the Ice Bath Method, a small container is prepared by mixing water and ice to achieve a stable temperature of 0 degrees Celsius (32 degrees Fahrenheit). Once the desired temperature is reached, the temperature sensor is carefully immersed in the ice bath and allowed to stabilize for a few minutes. This stabilization period ensures that the sensor has time to acclimate to the cold environment, allowing for more accurate measurements. After the stabilization, the sensor's temperature reading is recorded, and the value is compared with the known temperature of the ice bath to evaluate the sensor's accuracy at low temperatures. This method provides a reliable and controlled environment to assess how well the sensor performs under cold conditions, ensuring its suitability for potential integration into the final design.

Boiling Water Method:

In the Boiling Water Method, water is boiled in a small container until it reaches its boiling point of approximately 100 degrees Celsius (212 degrees Fahrenheit) at sea level. Once the water has reached a steady boil, the temperature sensor is carefully submerged into the boiling water and allowed to stabilize for a few minutes. This period of stabilization helps the sensor adjust to the high temperature and provides a more accurate reading. After stabilization, the temperature reading displayed by the sensor is noted and compared with the known boiling point of water to evaluate the accuracy of the sensor at high temperatures. This method offers a reliable means of

assessing the sensor's performance in extreme heat and is essential for determining its suitability for applications where precision at high temperatures is critical.

After these two basic methods to test the accuracy of the components, the results were analyzed and the best components were selected for further testing. These new testing methods include:

Reference Thermometer Comparison

To assess the accuracy of the temperature sensor using a calibrated reference thermometer, both the sensor and the reference thermometer are placed in the same environment, such as a controlled chamber set to a specific temperature. This controlled setting ensures consistent conditions for precise measurement and comparison. Simultaneous readings are taken from both the reference thermometer and the temperature sensor. The variance between the measurements from the two devices is calculated to assess the accuracy of the temperature sensor relative to the highly accurate reference thermometer. This method provides a clear benchmark for evaluating the performance of the temperature sensor and its suitability for precise applications.

Response Time Testing

To measure the temperature sensor's response time, which is the time taken for the sensor to detect and report a change in temperature, a sudden shift in the environment is introduced. This is achieved by initially placing the sensor in an ice-cold bath and then quickly transferring it to hot boiling water. The time it takes for the sensor to register the change in temperature is recorded, providing an indication of the sensor's responsiveness to sudden temperature fluctuations.

Once the response times are documented, the results are analyzed by comparing the sensor's response time against specified requirements or against the performance of other sensors. This comparison helps assess the sensor's efficiency and suitability for real-world applications, where rapid and accurate detection of temperature changes is essential. This method provides a

comprehensive assessment of the sensor's ability to respond to abrupt changes in temperature, ensuring that the chosen sensor meets the desired standards for the project.

5.1.2 Results

The table below summarizes the results of comprehensive testing conducted on a variety of temperature sensors as part of the evaluation process. Each sensor was subject to multiple testing methodologies, including the Ice Bath Method, Boiling Water Method, Reference Thermometer Comparison, and Response Time Assessment.

The table highlights key performance metrics, such as accuracy at 0 and 100 degrees, comparison to a reference thermometer, and response time in seconds. Favorable results are indicated to facilitate the identification of sensors that exhibit superior accuracy and responsiveness. These findings serve as invaluable insights for selecting the most suitable temperature sensor for integration into the final design, with particular emphasis on LM35DZ, LM35AH and LM35DT models which demonstrated notable performance across multiple testing criteria.

The detailed results of the testing conducted on the temperature sensors are presented below. Each sensor was evaluated, and the findings are summarized in the tables provided. The values in the tables represent averages obtained from multiple trials to ensure accuracy and reliability. These summarized results are presented in the following table, facilitating a comprehensive overview of the performance of each sensor across different testing methodologies. This approach ensures that the data presented accurately reflects the capabilities of each sensor and assists in informed decision-making regarding sensor selection for integration into the final design.

Sensor Model	Ice Bath Method (Accuracy at 0°C)	Boiling Water Method (Accuracy at 100°C)	Reference Thermometer Comparison (Accuracy)	Response Time (Seconds)
LM35DZ	**±0.5°C (Favorable)**	**±0.5°C (Favorable)**	**±0.3°C**	**5 seconds (Favorable)**
LM35AH	**±0.7°C**	**±0.6°C**	**±0.4°C**	**6 seconds**
LM35CZ/NOPB	**±0.6°C**	**±0.9°C**	**±0.2°C (Favorable)**	**7 seconds**
LM35DT	**±0.4°C (Favorable)**	**±0.4°C (Favorable)**	**±0.2°C (Favorable)**	**4 seconds (Favorable)**
DS	**±0.3°C**	**±0.3°C**	**±0.3°C**	**8 seconds**
SEN TMP	**±0.8°C**	**±0.7°C**	**±0.5°C**	**10 seconds**
SEN	**±0.6°C**	**±0.8°C**	**±0.6°C**	**9 seconds**
BME	**±0.5°C**	**±0.5°C**	**±0.4°C**	**11 seconds**

Table 12: Table Results Temperature Sensor Testing

5.1.2 Analysis of the Results

The primary goal of the device is to create a solution to address the needs of veterans with paralysis who are unable to perceive temperature extremes, enhancing the safety and well-being of these individuals by providing real-time monitoring of their body temperature. In our evaluation of temperature sensors for the device, we recognize the critical importance of accuracy, reliability, and responsiveness, considering the unique challenges of our users. The device must be capable of accurately detecting temperature variations and transmitting this information promptly for immediate access by the user.

Our testing methodology involved subjecting each sensor to rigorous evaluation under controlled conditions, including the Ice Bath Method, Boiling Water Method, Reference Thermometer Comparison, and Response Time Testing. By conducting these tests, we aimed to assess accuracy at extreme temperatures, compare sensor readings to a calibrated reference thermometer, and measure response time to temperature changes.

The results of our comprehensive testing revealed significant variations in the performance of different temperature sensors. Notably, the LM35DZ, LM35AH, and LM35DT models exhibited superior accuracy across multiple testing criteria. These sensors demonstrated minimal deviations from the expected temperature readings at both low and high temperatures, with response times within acceptable limits. This information allowed us to determine that these sensors objectively provide sufficient reliability and validity to perform according to our device design specifications.

Analysis of our testing methodology prioritized mapping performance compatibility with the user's skin, ensuring that the sensors could effectively measure skin temperature with direct contact. However, it's imperative to acknowledge that these conventional testing methods, such as the Ice Bath Method and Boiling Water Method, may not fully replicate the conditions experienced by individuals with paralysis as skin contact assumes a different interface for the temperature sensors to gather readings. Therefore, while our testing provided valuable insights into sensor performance under controlled conditions, further evaluation in collaboration with end-users is necessary to validate the device's effectiveness in real-world scenarios.

5.1.4 Recommendations and Actions Taken

Our testing procedures provided valuable insights into the performance of temperature sensors under controlled conditions. However, since the ultimate objective of the device is to measure skin temperature with precision, we recommend conducting additional testing specifically focused on simulating skin contact conditions. This involves incorporating materials that mimic the thermal properties of human skin into the testing environment, allowing us to evaluate sensor accuracy and response time under realistic settings. Such targeted testing will ensure that the device performs effectively in real-world applications.

As development of the skin temperature sensing device progresses, it is essential to integrate the chosen temperature sensors into the overall system and conduct thorough validation testing to ensure smooth and efficient functionality. This includes verifying inter-component communication, compatibility with the microcontroller, and seamless integration with the mobile application. A systematic approach to integration testing, encompassing both laboratory-based

assessments and field trials with end-users, is recommended to validate the performance of the entire system in practical settings.

The testing and evaluation process should be viewed as iterative, with ongoing opportunities for refinement and improvement. As the shell housing the temperature sensing device is designed, adjustments will be made to the shell's construction and the configuration of the sensors based on performance metrics such as accuracy and efficiency. Continuous updates to testing methodologies and criteria, informed by user feedback, will be crucial for maintaining the device's efficacy and relevance in serving the needs of veterans with paralysis. This approach will facilitate continuous improvement and optimize the device's impact on the end-users' quality of life.

5.2 Bluetooth Test and Evaluation

To ensure that we acquired the proper bluetooth module and have written the correct code we developed a process of evaluation which involved checking off a series of requirements and functionality that the module must have. Below is the list of requirements we identified that would best fit our system:

- Small size
- Fast transmission frequency
- Compatible with ReactNative
- Compatible with android
- Compatible with apple
- Coded in either C or Arduino
- Low Battery Requirement

These requirements helped us evaluate a variety of potential modules and components as they could be easily checked off through research. By looking over a components data sheet we would be able to identify if it passed our evaluation criteria. Ultimately through many weeks of research and evaluation, our team decided to select the Arduino Nano 33 BLE because it satisfied all of our conditions as well as acting as a microcontroller.

Although the component passed our initial evaluation, we still had to physically test it to see if we were able to effectively use this as part of the final product. To do this, we must test the components functionality as a microcontroller and bluetooth module. First, we wrote a simple program on the Arduino IDE in which the Arduino Nano increments a value once a second and displays it on the serial monitor. After confirming the device could function as a microcontroller, we moved on to test its bluetooth connection capabilities. This required us to do research in which we learned that the module used BLE, or Bluetooth Low Energy, and using the arduino BLE library, we wrote a program that initialized the module as a peripheral and transmitted the incrementing value through bluetooth to a connected device. This program can be referred to in Appendix A. Using an iPhone application called LightBlue, we were able to connect with the peripheral and display the transmitted incrementing values, confirming the module's bluetooth functionality and compatibility condition.

After confirming the devices functionality, we moved on to test the boundaries and properties of the connectivity. Initially, the value was incremented and then transmitted every second. While this worked without error and was sufficiently fast for our final product, we wanted to test the transmission frequency limit. We found that when coded to transmit 10 times per second, the iPhone display would skip values, indicating that the limit was 10 Hz.

The Arduino Nano 33 BLE passed our evaluation and testing so we have chosen to move forward with it as both our microcontroller and bluetooth module.

5.3 Software Test and Evaluation

The software component of the temperature monitoring system, consisting of a mobile application and a backend server, underwent extensive testing and evaluation to ensure its reliability, efficiency, and usability. This section details the methodologies employed, the results obtained, and how these findings align with the initial design specifications.

5.3.1 Testing Methodologies

The testing methodologies employed for the temperature monitoring system were crucial in ensuring the integrated functionality and user experience of the software. Integration testing was conducted using Django's test client for the backend and the React Native Testing Library for the frontend. This phase focused on the seamless interaction between API endpoints and the frontend, verifying that data flows correctly and that components integrate seamlessly across the system. Additionally, end-to-end (E2E) testing was carried out manually to simulate real-user scenarios thoroughly. This included user registration, login, temperature data viewing, and modifications to alert settings. The purpose was to test the complete flow of the application, from initial user interaction to final output, to ensure the system operated effectively and met all functional requirements.

5.3.2 Results and Data Presentation

The results from the integration testing were highly positive, with all defined routes and component integrations passing successfully. This confirmed that both the API and the mobile app components interacted correctly without significant data flow issues, as all endpoints responded within the acceptable time limits set by the project specifications. During the end-to-end testing, all user flows were verified without any critical failures; registration, login, data visualization, and alert settings workflows performed flawlessly. However, these tests did reveal some minor user interface delays during periods of data loading. These issues were subsequently addressed and optimized in the post-testing phase to enhance the overall user experience.

5.3.3 Comparison Against Design Specifications

When compared against the design specifications, the software performed admirably, meeting most of the outlined functional requirements. This included efficient real-time data processing, robust user authentication protocols, and comprehensive alert management systems. The software's performance also generally met the expected standards, handling all prescribed tasks effectively under normal to moderately high load conditions. However, it fell short under extreme

load conditions, indicating a need for further optimization and resource enhancement. These findings have highlighted areas for future improvement, particularly in scaling the backend infrastructure to better manage higher user concurrency and data throughput, ensuring the system's reliability and efficiency in all usage scenarios.

Conclusion

The software testing and evaluation phase was crucial in identifying and rectifying issues, optimizing performance, and validating the system against the defined specifications. While the design mostly met the anticipated standards, We anticipate the current server capabilities aren't sufficient for high loads. Moving forward, plans include upgrading server capabilities and optimizing database configurations to support greater user concurrency effectively.

6.0 TIME AND COST CONSIDERATIONS

The cost/budget constraints weren't an issue at all when completing this project. This is due to the fact that we started with a pretty large budget of $1,000. We also did a good job on keeping track of our budget by updating the bill of materials every time we purchased an item. Another thing we did is we made sure at the beginning of the project to make a list of necessary items needed for the completion of the project, this way we could allocate the necessary budget to purchase these items.

On the other hand there were some issues that were out of our hands that delayed some faces of the project. We did find ways to work around some of these obstacles and continue to make progress while the problem was being resolved. The main problem we encountered was the delay in the shipment/arrival of the android phone we ordered. This caused the testing of bluetooth communication with the actual android phone to be delayed. To work around this our team decided to go ahead and test the bluetooth communication using an android emulator. This way we would have a starting framework to run the bluetooth communication on the actual phone as soon as it arrived.

7.0 SAFETY AND ETHICAL ASPECTS OF DESIGN

Since we built an electronic device that will be worn and used there are some ethical and safety boundaries we took into consideration. First, we evaluate the safety of the device itself, making sure that the wearer will not be harmed in any way by some sort of malfunction or misuse. A couple safety threats would include the batteries exploding, wires/bondings overheating, and toxic materials. To avoid these threats, we have carefully researched the materials, batteries, and devices we selected to construct the device and made sure their thresholds and resistances will survive harsh outdoor conditions.

Furthermore, because we worked with health related data and information, we must protect the users personal information. We will not in any way share or distribute any information collected and stored inside of our application and device. The data is also protected from unauthorized access through a login system in order to safe-guard the users information.

To help streamline the process, our group adopted the following standards when building the device to make sure we stay within ethical and practical guidelines in Appendix C.

8.0 RECOMMENDATIONS

In striving to enhance the functionality of our bluetooth temperature sensor model, designed to assist veterans with paralysis, we direct our focus in reducing the size of our device. A smaller and more compact device not only promises greater portability, but increased comfort and convenience for veterans. They will be able to go through better experiences with the devices, and the device will be easier to wear with its smaller size and the lighter weight.

Looking ahead, an alternate fabrication method that transcends traditional 3D printing opens gates for opportunities to experiment with more diverse materials in developing our device

housing. By utilizing materials with varying textures and levels of protection, we can heighten the comfort and convenience of our device as well as its durability and resilience.

Efforts to refine temperature sensors tailored for veterans with paralysis center around the pursuit of compactness for heightened user satisfaction. By prioritizing the minimization of device size, we aim to alleviate potential burdens associated with device handling and integration into daily routines.

9.0 CONCLUSION

This document meticulously explores crucial aspects of the "Temperature Sensors for Veterans with Paralysis" project, encompassing the Design Problem, Design Solution, Design Implementation, and Test and Evaluation. In the Design Problem section, we delve deep into the intricacies of the design problem, outlining project specifications, and articulating expected deliverables. In the Design Solution section, we discuss a spectrum of potential solutions to our design problem, delving into solutions with bluetooth sensors, hardware configurations, and software applications. Transitioning to the Design Implementation, meticulous attention is paid to hardware integration and software development, prioritizing performance, comfort, and cost-effectiveness in alignment with project objectives and user needs. The iterative was used to build our design which underscores our commitment to optimizing functionality while enhancing user comfort and convenience. The Test and Evaluation stage adopts a structured approach to ensure the efficiency of our design implementations, culminating in a comprehensive assessment of the integrated system's performance and functionality.

The core purpose in our mission is to bring profound enhancement in the quality of life of veterans who are combatting the limb injuries or paralysis. Our overall goal is to grant them access to cutting-edge temperature monitors, fostering their safety and comfort. To manifest our vision, we developed a strategic plan and put it into action. The plan involved customizing sensors for veteran comfort, serializing and transmitting the data output, and ensuring convenient accessibility through the user-friendly smartphone interface. By paying a thorough attention to

design challenges, clear requirements, and individual roles and responsibilities, we successfully executed our meticulously devised deployment plan. This document serves as a comprehensive overview of our problems, plans, and executions of our project.

REFERENCES

[1] Shirley Ryan AbilityLab, “Spinal Cord Injury Complications: Temperature regulation,” May 2022; https://www.sralab.org/lifecenter/resources/spinal-cord-injury-complications-temperature-regulation

[2] S. Khan, M. Plummer, A. Martinez-Arizala, K. Banovac, “Hypothermia in Patients with Chronic Spinal Cord Injury” 2007; https://www.ncbi.nlm.nih.gov/pmc/articles/PMC2032005/

[3] Iowa State University, “sdmay23-07 • temperature sensors for veterans with paralysis,” May 2007; https://sdmay23-07.sd.ece.iastate.edu/

[4] A. Subham and S. Mahata, “Develop a Smart Android App for Temperature Alert System,” Jan. 2020; https://www.electronicsforu.com/electronics-projects/temperature-alert-system-smart-android-app.

[5] M. Zahner, T. Helbling, L. Durrer, and E. Schwyter, “Sensor Unit for a Portable Computer System and Integration of the Sensor Unit,” Sep. 19, 2023. [Online]. https://patentimages.storage.googleapis.com/a9/8e/30/82768ffcb696ab/US11759111.pdf

APPENDIX A – ARDUINO TEMPERATURE SENSOR TESTING CODE

```
const int lm35_pin = A6;  // Assuming the LM35 sensor is connected to
analog pin A6 on the Arduino Nano 33 BLE

void setup() {
  Serial.begin(9600);
}

void loop() {
  int temp_adc_val;
  float temp_val;
  temp_adc_val = analogRead(lm35_pin);    // Read temperature
  temp_val = (temp_adc_val * (3.3 / 1023) * 100);    // Convert ADC value
to equivalent voltage and then to temperature
  Serial.print("Temperature = ");
  Serial.print(temp_val);
  Serial.print(" Degree Celsius\n");
  delay(1000);
}
```

APPENDIX B – ARDUINO MICROCONTROLLER CODE

```
#include <ArduinoBLE.h>

BLEService testService("37b11d5e-ece1-4bb8-9e76-132521d99ef6");

// Bluetooth® Low Energy Battery Level Characteristic
BLEIntCharacteristic testLevelChar("33718771-a266-4194-9a74-3ea52862d7d0",
// standard 16-bit characteristic UUID
    BLERead | BLENotify); // remote clients will be able to get
notifications if this characteristic changes
const int lm35_pin = A6;
long previousMillis = 0;  // last time the battery level was checked, in
ms

void setup() {

  Serial.begin(9600);
  // begin initialization
  if (!BLE.begin()) {

    while (1);
  }


  BLE.setLocalName("tempSensor"); //set local name
  BLE.setAdvertisedService(testService); // add the service UUID
  testService.addCharacteristic(testLevelChar); // add the battery level
characteristic
  BLE.addService(testService); // Add the service
  testLevelChar.writeValue(0); // set initial value for this
characteristic

  /* Start advertising Bluetooth® Low Energy.  It will start continuously
transmitting Bluetooth® Low Energy
     advertising packets and will be visible to remote Bluetooth® Low
Energy central devices
     until it receives a new connection */

  // start advertising
  BLE.advertise();
```

```
}

void loop() {
  // wait for a Bluetooth® Low Energy central
  BLEDevice central = BLE.central();

  // if a central is connected to the peripheral:
  if (central) {
    // turn on the LED to indicate the connection:
    //digitalWrite(LED_BUILTIN, HIGH);

    // check the battery level every 200ms
    // while the central is connected:
    while (central.connected()) {
      long currentMillis = millis();
      // if 200ms have passed, check the battery level:
      if (currentMillis - previousMillis >= 1000) {
        previousMillis = currentMillis;
        updateValue();
      }
    }
    // when the central disconnects, turn off the LED:
    //digitalWrite(LED_BUILTIN, LOW);
  }
}

void updateValue() {
  /* Read the current voltage level on the A0 analog input pin.
     This is used here to simulate the charge level of a battery.
  */
  float temp_adc_val = analogRead(lm35_pin);
  float temp_val_serial = (temp_adc_val * (3.3 / 1023) * 100);
  int temp_val = (temp_adc_val * (3.3 / 1023) * 100) * 100;
  Serial.print("Temperature = ");
  Serial.print(temp_val_serial);
  Serial.print(" Degree Celsius\n");
  testLevelChar.writeValue(temp_val);
}
```

APPENDIX C – STANDARDS/GUIDELINES ADOPTED FOR BUILDING DEVICE

1. 360-2022 IEEE Standard for Wearable Consumer Electronic Devices which outlines different specifications, technical requirements, and testing methods for different types of wearable electronic devices.
2. P2888.1/D02, Aug 2023 IEEE Draft Specification for Sensor Interface for Cyber and Physical World which defines vocabulary, data formats, and APIs for collecting sensor data and communicating between the digital and physical world. This standard will be used to streamline the communication between our device which analyzes physical data and the smartphone app which formats and presents the data in digital form.