# 3. DESIGN APPROACH

The SafePaws vest is a smarter way to track a dog's vitals remotely. The vest is integrated with a heart rate sensor and a body temperature thermometer to monitor one's dog while on-the-go. The SafePaws vest is designed to collect data and measurements autonomously, while also allowing for the user to access the sensor data wirelessly via a mobile application. This section explores the final design choices, including considerations of multiple design options, an overview of the system, and detailed descriptions of each subsystem. The project's design approach emphasizes data accuracy, a friendly user-experience, and pet safety, all of which is guided by the project requirements, constraints, and engineering standards.

## 3.1. Design Options

The objective of this section is to provide a detailed overview of the design choices considered. Two major design options were explored, each with its unique advantages and limitations. Background is provided for each option, explaining the decision-making process and design justifications. By providing an outline for the benefits and drawbacks of each design, further details are provided to justify how the chosen components provide optimal solutions for achieving the project's objectives. The design options were evaluated within the project requirements, constraints, and engineering standards to ensure they are practical and satisfy the required technical goals.

### 3.1.1. Design Option 1

A low-profile collar was considered in the original design approach. This design involves a collar with the sensors built into it, along with pockets for the controller and battery. A collar has the advantage of being smaller than a vest and can be secured to a dog to provide consistent conditions for sensors to make their readings.

The collar design has its advantages reduced when the other required components are factored into its design, particularly the microcontroller and battery. The degree of durability required for the components could not be properly addressed. As the battery, microcontroller, and sensors are not securely attached to the substrate (collar and pockets). As well as not being able to withstand the outdoor elements and normal wear-and-tear from dogs themselves.

### 3.1.2. Design Option 2

Another design option that was considered was an integrated vest. This approach incorporates a similar approach to Design Option 1, but instead uses a vest to store the technical components. Including the sensors, controller, and battery. A vest allows for the technical components to be better spaced throughout the design and allow the microcontroller and battery to be better secured to the substrate (inside of the vest) to minimize vibrations. The advantages to this design include pet comfort and more space for components to be easily integrated. A vest also allows for the component's weight to be distributed and be more comfortable for a dog to wear. This is an improvement over Design Option 1, as all components would be clustered around the dog's neck with a collar-centric design. The limitations are bounded by the vest's size, but incorporating straps can allow the vest to be adjusted to a secure fit for a broader range of sizes. For these reasons, the vest design was ultimately chosen.

## 3.2. System Overview

This section provides an overview of the SafePaws vest, exploring the system's functionality and how subsystems interact with one another.

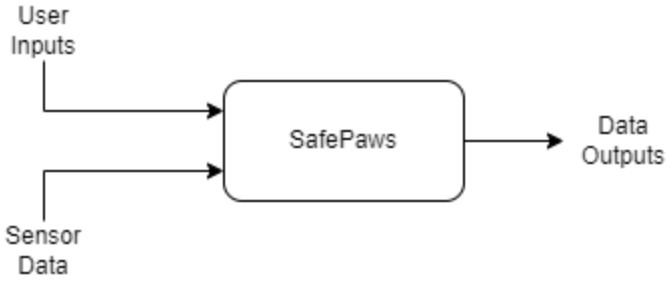


**Figure 3-1: SafePaws System at a Glance (Level 0)**

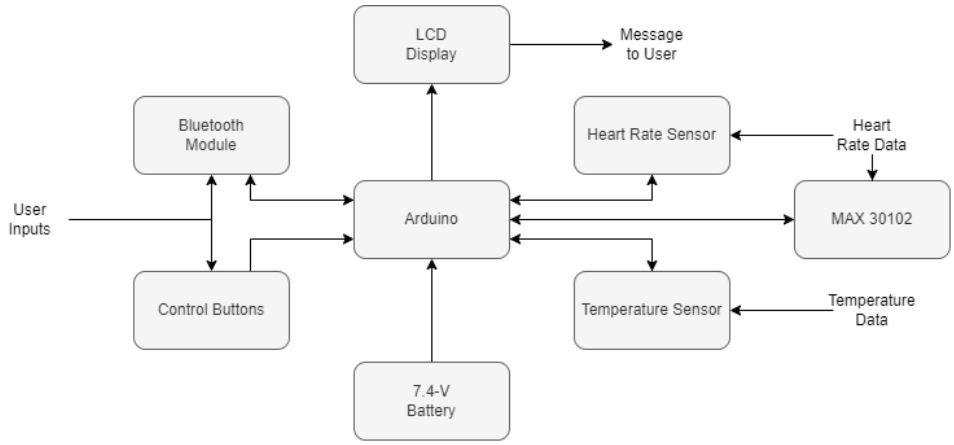


**Fig. 3-2: SafePaws Functionality (Level 1)**

In this section, we list the components that will make up the primary system and each of its subsystems. Details for each subsystem will include the functions that they will provide for the project and their most important technical specifications. In section 3.2.1, the other microcontrollers considered for the project are also listed.

### 3.2.1. Microcontroller

The microcontroller chosen for the project was the Arduino Uno REV3. As shown in Figure 3-2, it is at the center of all I/O functions and is where the code for the sensors and peripherals will run. It has several features that allow for integration of sensors and peripherals, as well as a well-supported software library. It comes with several connectivity options for plug-and-play sensors, including support for both analog and digital devices. Off-the-shelf peripherals are readily available and allow for easy implementation of 802.11-based devices for Bluetooth and/or Wi-Fi connectivity. A myriad of software libraries are available for implementing the peripherals and allow for customization to suit the needs of the project. In Table 3-1, devices considered for the project are listed, along with the advantages and drawbacks of each.

**Table 3–1: Microcontroller Considerations**

| Device | Advantages | Drawbacks |
|---|---|---|
| Arduino Uno REV3 | • Provides support for both analog and digital sensors, which allows for more sensors to be considered<br>• C++/C-based, native code can be run directly on the hardware (no pre-compiled executables are required)<br>• Wide range of hardware peripherals are available<br>• Large, well-supported software library is available for use by its developer community<br>• Low power consumption | • Slightly larger profile versus other microcontroller options<br>• Low storage available; requires additional SD card peripheral to store files<br>• Low memory (32 KB) [1] |
| Basys Board | • Lowest power consumption of all microcontrollers considered<br>• HDL-based assembly language used to program the Basys Board requires fewer hardware resources to run [2] | • HDL-based programming (more difficult and time-consuming to write and debug)<br>• Hardware sensor support (fewer options available when compared to Arduino)<br>• Less support for its software library when compared to other options<br>• Best suited as a field programmable gate array (FPGA) rather than as a microcontroller [3] |
| Raspberry Pi | • Most powerful processor out of all the microcontrollers considered (1.4 GHz), large built-in memory (1 GB+) [4]<br>• Native SD-card slot.<br>• Multiple programming languages are supported via compiled executables (C, C++, Python, Java, etc.).<br>• Large, well-supported software libraries available by its developer community.<br>• Native USB, Bluetooth, Wi-Fi, and network support [4].<br>• Low-power consumption. | • Software development-focused device. Thus, fewer hardware/sensor options are available [5].<br>• Does not include built-in support for analog sensors. |

Based on Table 3-1, the Arduino Uno REV3 was selected as the microcontroller for its overall advantages. It provides better flexibility for the types of sensors and peripheral hardware that can be used versus the Raspberry Pi and has a more robust software library for implementing subsystems when compared to the Basys Board.

### 3.3. Subsystems

The prototype for SafePaws includes three subsystems: sensors, power systems, and software. The sensors are used to measure heart rate and body temperature, which will be stored on the device to view as historical data. Additionally, live data from the sensors will be viewable via the LCD screen during development and companion app in the final design. Power systems include a lithium-ion battery which allows for the system to operate autonomously and remotely, and a charging component to ensure proper charging of said battery. The software includes programming of both the controller and the companion app. The code of the controller directs the sensors, peripherals, and subsystems while the companion app will be used on mobile devices.

#### 3.3.1. Subsystems

A photoplethysmography (PPG) sensor was chosen for the heart rate sensor. Other types of sensors that were considered were piezoelectric and electrocardiogram. Piezoelectric sensors use vibrations generated by heartbeats to determine heart rate, while electrocardiogram sensors have electrodes that rely on skin contact to detect heart rate. A PPG sensor utilizes a light-emitting diode as its light source producing light at precise wavelengths to detect changes in blood volume within the body. Pulse Sensor and MAX30102 PPG sensors were chosen for this project as they allow for heart rate to be monitored through fur, which would interfere with electrocardiogram and piezoelectric sensors. The team decided that two sensors are needed for certain wavelengths of light are better suited in capturing heart rate through lighter versus thicker amounts of pet fur. The Pulse Sensor is used on areas of the animal with less fur while the MAX30102 is used for thicker areas of fur.

**Table 3-2: Heart Rate Sensor Options**

| Sensor | Price | Dimensions | Voltage (V) | Amp (mA) | LED Output Wavelength (nm) |
|---|---|---|---|---|---|
| Project Requirements | ≤ $1000 | ≤ 10.98 x 10.94 x 1.97 in | ≤ 7.4 V | 4 – 20 mA | 450 – 1200 nm |
| Pulse Sensor (PPG) [7] | $24.99 | 4 x 0.5 x 6 in | 3.0 – 5.5 V | 4 mA | 565 nm |
| MAX30102 (PPG) [8] | $12.99 | 5.6 x 3.3 x 1.55 mm | 3.3 – 5.5 V | 20 mA | 660/880 nm |
| AST1041 [9] | $19.95 | 5.6 x 3.3 x 1.55 mm | 1.8 – 5.0 V | 20 mA | 870/900 nm |


**Fig. 3-3: Pulse Sensor**

**Fig. 3-4: MAX30102**

For the body temperature sensor, DS18B20 was chosen. The DFRobot I2C Temperature and Humidity Sensor, while a less expensive option, are outperformed by the DS18B20. The DS18B20 has a feature that allows for multiple uses of the same sensor. This feature is called the one-wire interface which uses only one digital pin for communication. The DS18B20 has a temperature range of -55°C – 125°C.

**Table 3-3: Body Temperature Sensor Options**

| **Sensor** | **Price** | **Dimensions (mm)** | **Voltage (V)** | **Amp (A)** | **Temperature Range (**°C) |
|---|---|---|---|---|---|
| Project Requirements | ≤ $1000 | ≤ 10.98 x 10.94 x 1.97 in | ≤ 7.4 V | 1 – 1.5 mA | -55°C – 125°C |
| DS18B20 (Digital thermometer) [10] | $13.66 | 91 x 4 mm | 3.0 – 5.0 V | 1 – 1.5 mA | -55°C – 125°C |
| DFRobot I2C Temperature & Humidity Sensor [11] | $7.43 | 5cm x 1.5m | 2.5 – 5.5 V | 3.0 µA | -40°C – 125°C |

**Fig 3–4: DS18B20**

The DS18B20 was chosen for the body temperature sensor due to its versatile, 1-wire interface, allowing the use of multiple sensors with just one digital pin for communication. The sensors were chosen for their accurate and reliable data collection, as per the project requirements.

### 3.3.2. Power Systems

The design team compared the various battery types, and the most practical option was the lithium-based batteries. In research, it was found that lithium-ion and lithium-polymer batteries can both provide adequate power and sustain battery life in the required space, and both are reasonably priced within the budget. However, the lithium-ion battery offers greater power density within the cells which led to this battery type being chosen.

**Table 3-4: Battery Type Comparisons**

| Battery Type | Advantages | Disadvantages |
|---|---|---|
| Nickel-Metal Hydride | • Inexpensive<br>• Medium weight<br>• Less charge memory compared to nickel-cadmium | • High loss of charge over time<br>• Lifespan of 500 cycles<br>• Low cell voltage |
| Nickel-Cadmium | • Inexpensive<br>• Lifespan of 800 cycles | • Heavy weight<br>• High loss of charge over time<br>• Charge memory<br>• Low cell voltage |
| Lithium-Polymer | • No charge memory<br>• Lightweight<br>• Low loss of charge over time | • Less power density than most lithium-ion<br>• More costly than nickel-based battery types |
| Lithium-Ion | • No charge memory<br>• Lightweight<br>• Low loss of charge over time<br>• Higher power density than lithium-polymer | • More costly than nickel-based battery types |

The power requirements of each component are small enough that the system can operate continuously. Removing the need for power management (low-power mode) and still achieve the battery life goal.

**Table 3-5: Power Draw of Each Component**

| Component Name | Power Draw (mA) |
|---|---|
| Arduino Uno | 42 mA |
| Temperature Sensor | 2 mA |
| Pulse Sensor | 4 mA |
| Max 3012 Pulse Sensor | 20 mA |
| LCD Screen | 1.5 mA |
| **Total power required** | 69.5 mA |

SafePaws uses Arduino's built-in voltage regulator to convert the battery's input voltage of 7.4 volts to the required 5 volts. A voltage of 7.4 was chosen because it is the closest commonly available battery voltage to 5 volts that could be found. All calculations are done with respect to Arduino's efficiency of 67.6%. By multiplying the voltage at which the components will run by the amperage that they draw, the wattage of the system can be found. In the case of SafePaws, it requires 0.3475 watts (Eq. 1) of power. This value can then be used to calculate the required battery size.

$$5\,V \cdot 0.0695\,Ah = 0.3475\,W \quad \text{(Eq. 1)}$$

$$\left(\frac{24\,hours}{0.676}\right) \cdot 0.3475\,W = 12.34\,Wh \quad \text{(Eq. 2)}$$

$$\frac{12.34\,Wh}{7.4\,V} = 1.67\,Ah \quad \text{(Eq. 3)}$$

To meet the goal of 24-hour battery life a battery with 12.34-watt hours (Eq. 2), or 1.67-amp hours (Eq. 3), of battery life is required. The battery chosen has an amp hour rating of 2.6, which exceeds the required battery life by 13.43 hours (Eq. 4 and 5).

$$7.4\,V \cdot 2.6\,Ah = 19.24\,Wh \quad \text{(Eq. 4)}$$

$$\frac{19.24\,Wh}{0.3475\,W} \cdot 0.676 = 37.43\,h \quad \text{(Eq. 5)}$$

This battery was chosen as it will provide a higher margin of usage that the project will experience if the components real-world amperage draws do not match the specification sheets. It is also well within the project's budget, thus providing additional benefits.

### 3.3.3. Software

The software design can be broken into two different systems: controller software and app software. A high-level software design can be seen in Figures 3-5 and 3-6.

#### 3.3.3.1. Controller Software

Figure 3-5 shows the general flow of the controller software once the device is turned on and the processes initialized.

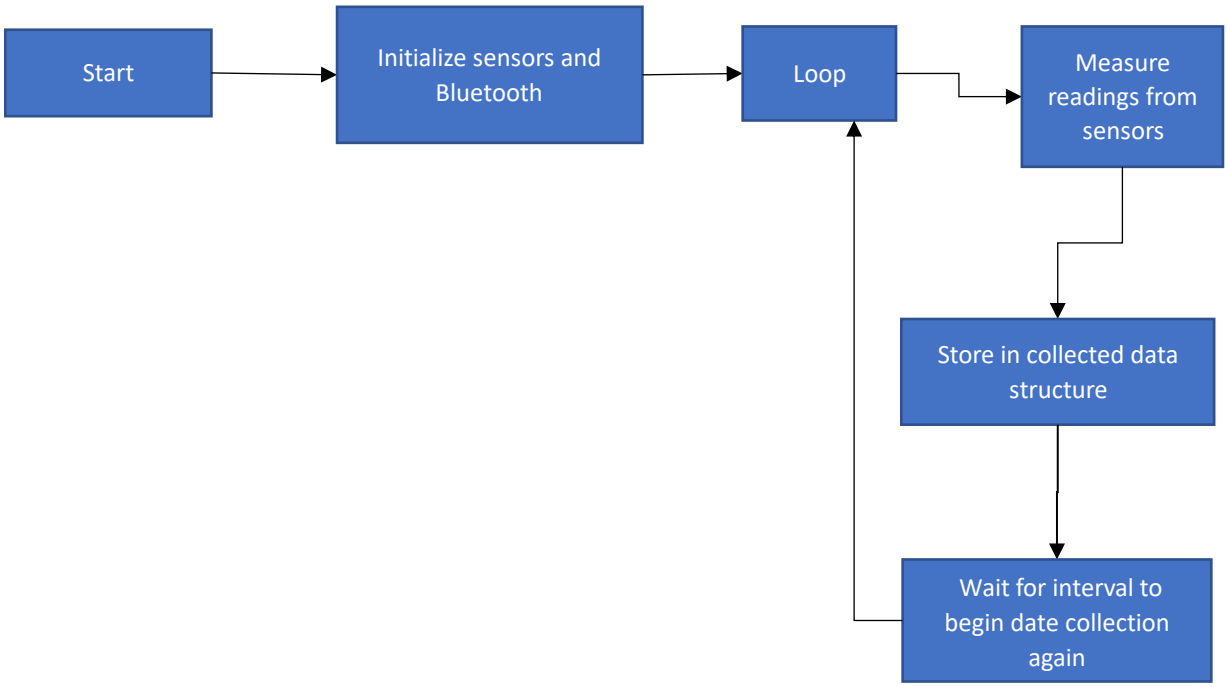

**Fig. 3-5: SafePaws controller flowchart**

The flowchart in Figure 3-5 shows how the components and controller board work. The sensor components are initialized along with the Bluetooth communication module. The components are then set to operate autonomously. The readings are gathered from the sensors, stored into a data structure, and set to be obtained from the user when requested.

### 3.3.3.2. Mobile Application Software

The app software serves as the user interface for the vest. The software program is designed to allow the user to set base parameters and view data readings. The app also triggers user alerts if the sensor data exceeds those parameters at any moment. The design itself ensures user-friendliness and data accuracy.

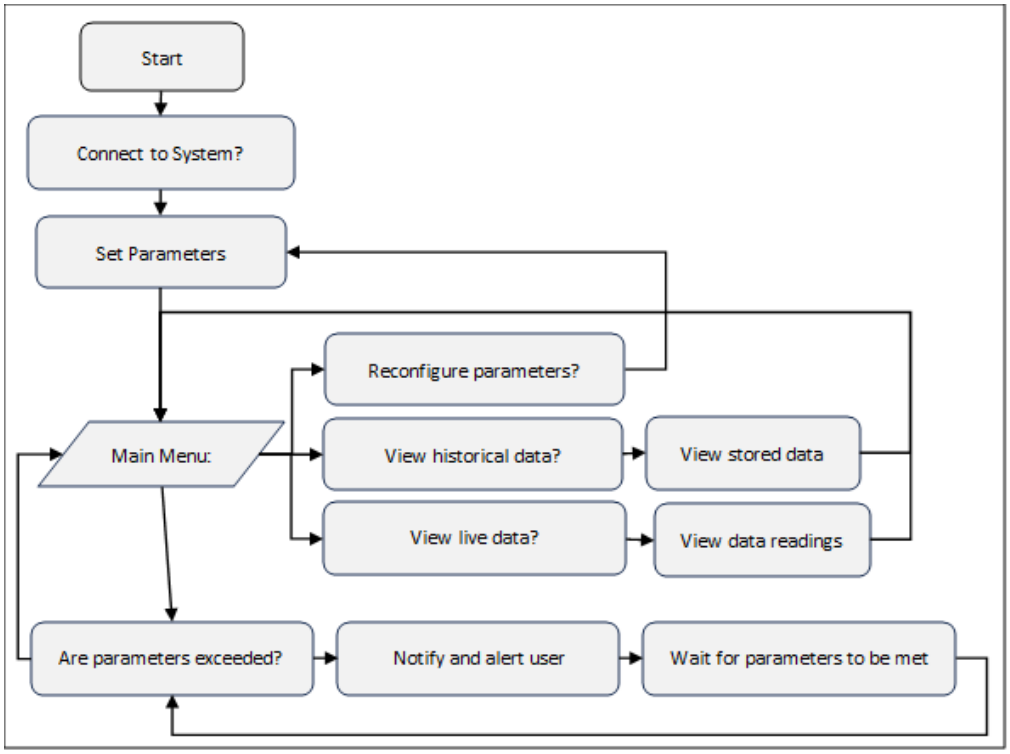


**Fig. 3-6: SafePaws mobile application flowchart**

The application software subsystem of SafePaws is designed to offer a user-friendly experience while ensuring strong data management. In Figure 3-6, an overview of the app flow and user experience branch is shown. The interface, developed using Flutter programming, focuses on a user-friendly design with a graphical user interface (GUI) for mobile devices. Flutter was chosen due to its cross-platform compatibility with Android and iOS applications from a single codebase. The GUI enables users to seamlessly connect to the monitoring system, set parameters, access live and historical data, receive notifications, and configure system settings. Data exchange occurs wirelessly in JSON format, promoting clarity and interpretation [12]. This format includes fields for sensor data, timestamps, and system status. To ensure the system's responsiveness and error-free operation, Flutter's protothread approach manages simultaneous events efficiently. For example, one thread manages data collection and analysis, while another handles user commands and GUI updates. This parallel processing approach effectively prevents deadlock and livelock situations, ensuring that the system remains responsive to user input while efficiently collecting and processing data. This software design ensures a balance between user interaction and data processing.

### 3.4. Level 2 Prototype

In the Design II semester, the primary objective is to test and improve upon a fully functional prototype of the SafePaws vest and to integrate all components and subsystems. The prototype should represent the design values of data accuracy, user-friendliness, and safety. The goal of finalizing the prototype is to have it be available to a user of a random testing subject group and allow that user to utilize the prototype based on their own understanding. The prototype should operate remotely upon activation, enabling the user to access the mobile application via the Bluetooth communication module. Within the mobile application, the user should have authority to set parameters of sensor readings and access real-time or historical data readings. Most importantly, the system will remain watchful, promptly notifying users if sensor parameters are ever exceeded through alerts and notifications. Design II for the design team centers around the process of testing, evaluation, improvement, and redesign.

#### 3.4.1 Level 2 Diagram

In Figure 3-7, the Level 2 diagram outlines the low-level components of the finalized design for the SafePaws vest.

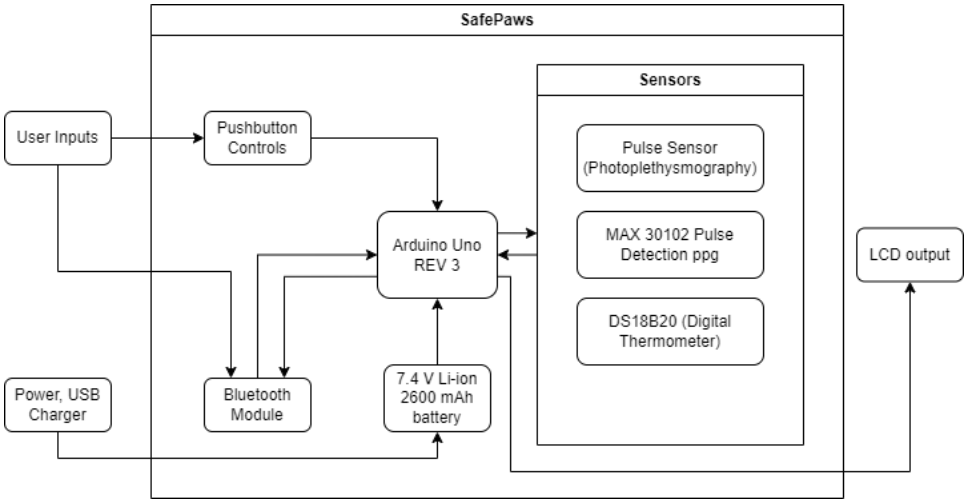


**Fig. 3-7: Diagram for SafePaws (Level 2)**

Figure 3-7 describes how the software and hardware components connect to become the SafePaws prototype.

The SafePaws vest represents the solution for remote health monitoring of dogs, as the prototype aims to integrate a heart rate sensor and body temperature thermometer in a simple, and a wearable design for on-the-go tracking. With autonomous data collection and wireless control through a mobile app, the design choices prioritize the data is recorded, stored, and displayed accurately, the user experience is simple and intuitive, and monitor a pet's safety. The final design considerations, from exploring multiple options to subsystem descriptions, bring the team into its next phases of the design process: testing, evaluation, and improvement.

### 3.5. References

[1] "Arduino Uno R3 Specification Sheet." Arduino Documentation. https://docs.arduino.cc/hardware/uno-rev3 (accessed Oct. 20, 2023).

[2] “Basys 3 Reference Manual,” Basys 3 - Digilent Reference, https://digilent.com/reference/programmable-logic/basys-3/reference-manual (accessed Oct. 20, 2023).

[3] Y. Tan, “Basys MX3 Trainer Board - Review,” Element14, https://community.element14.com/products/roadtest/rv/roadtest_reviews/756/basys_mx3_trainer_bo_3 (accessed Oct. 20, 2023).

[4] “Raspberry Pi 4 Model B Specifications,” RaspberryPi.com. https://www.raspberrypi.com/products/raspberry-pi-4-model-b/specifications/ (accessed Oct. 20, 2023).

[5] “Arduino vs. Raspberry Pi: Which Board is Best?,” Circuito.io Blog, https://www.circuito.io/blog/arduino-vs-raspberry-pi/ (accessed Oct. 20, 2023).

[6] “MC33269 - Voltage Regulator Adjustable Output, Low Dropout,” On-Semi Data Sheet, https://www.onsemi.com/pdf/datasheet/mc33269-d.pdf (accessed Oct. 21, 2023).

[7] “In-Depth: Detect, Measure & Plot Heart Rate Using Pulse Sensor & Arduino,” LastMinuteEngineers.com. https://lastminuteengineers.com/pulse-sensor-arduino-tutorial/ (accessed Oct. 23, 2023)

[8] “High-Sensitivity Pulse Oximeter and Heart-Rate Sensor for Wearable Health.” Analog.com. https://www.analog.com/media/en/technical-documentation/data-sheets/max30102.pdf (accessed Oct. 23, 2023).

[9] “AST1041.” DigiKey.com. https://www.digikey.com/en/products/detail/tinycircuits/AST1041/11594531 (accessed Oct. 23, 2023)

[10] “Waterproof 1-Wire DS18B20 Digital Temperature Sensor,” Adafruit.com. https://www.adafruit.com/product/381?gclid=CjwKCAjws9ipBhB1EiwAccEi1I_Cpjkz6gxzXgaTveIUXcOG1HRZNsbkSlXBfyg8ZbmgEaWvQ2WrZRoCuzIQAvD_BwE (accessed Oct. 23, 2023).

[11] “DFRobot I2C Temperature & Humidity Sensor (Stainless Steel Shell),” RobotShop.com. https://www.robotshop.com/products/dfrobot-i2c-temperature-humidity-sensor-stainless-steel-shell?gclid=CjwKCAjws9ipBhB1EiwAccEi1CuoByaq1hzph8IVKYTeXjcKXWlgK6sr-yW5Cu4VAhkbwjlkPy68TBoCAgoQAvD_BwE (accessed Oct. 23, 2023).

[12] L. Miller, “How to Parse JSON Data with Arduino (ArduinoJSON Tutorial),” LearnRobotics.org, https://www.learnrobotics.org/blog/parse-json-data-arduino/#:~:text=To%20parse%20JSON%20files%20using%20Arduino%2C%20you%E2%80%99ll%20need,set%20up%20the%20sketch%20as%20you%20would%20normally. (accessed Oct. 23, 2023).