ECE 441 Project Document
Team 10: Wearable Stress Detection Device

Hamad Alali, Kris Haro, Arthur Kichatov, Jimmy Parra

2021-2022

Contents

1 Overview

1.1 Executive Summary

The objective of this project is to create a device capable of detecting stress in its user. The wearable stress detection device will have 4 primary methods of determining if the user feels stressed: galvanic skin response, heart rate, temperature, and a switch. The switch can be flipped by the user if they feel stressed. All the sensors will be connected to a microcontroller housed in an enclosure the size of a person's wrist. The data collected by the microcontroller is then sent to a website where the information is graphed and can be exported as a csv.

Currently, the project is in its early stages of development, where members of the team are researching methods of how to implement different blocks of the system to meet the requirements. Over the next few months, this research will then be applied to realize small subsections and then implemented into one uniform device.

1.2 Team Communication Protocols and Standards

Group member contact information :

1. Hamad Alali
   Phone number: (541) 368-8136
   Email: alalih@oregonstate.edu
2. Kris Haro
   Phone: (971) 283-9774
   Email: harokr@oregonstate.edu
3. Arthur Kichatov
   Phone: (971) 295-9125
   Email: kichatoa@oregonstate.edu
4. Jimmy Parra
   Phone: (503) 858-4650
   Email: parraj@oregonstate.edu

Team standards	
Communication	For this group project we have all agreed to connect with each other using discord, this way we could either use chatting or have remote meetings. Having one set platform makes it easy because we will not have to switch between different platforms and all our work is kept in one place.
Group Meetings	We will have weekly group meetings as a group and to avoid any teammate missing a meeting, we decided on a day and time that fits all group members so that there isn't any time conflict between our classes or job's. That way, everyone could be present and contribute to the project. If any group member has a necessary matter, he could inform the group members that he won't be able to attend the meeting.
Issues	Whenever a group member is facing a technical issue or difficulty, he is encouraged to share it with the group so that we could all contribute to solve the issue together. But all issues should be shared with enough time so that the group does not miss any submission.
Behavior	All group members should respect each other regarding the ethnicity or color of one another. All group members should be honest with the group.
Group tasks	All the team's assignments are going to be shown through Asana, which is an online platform that is used for improving team collaboration. We will upload what tasks that are going to be due soon and assign each part of the task to a group member. All assignments that are due should be done before the submission time by at least 3-5 hours.

We have met with the group partner to discuss the project we will be working on. In our first meeting, we had multiple questions prepared to share the team's approach for the project and what parts of the project we should start to think about to achieve and accomplish this project. We agreed with the partner that we will send her weekly emails explaining what we have worked on during the week or we could ask her any questions or concerns we might have. There will be a meeting with the partner every other week so that the partner stays up to date.

1.3 Gap Analysis

In today's day and age, smartwatches with built-in heartbeat sensors are readily available, this would be how the average person monitors stress due to convenience [2]. The goal of this project is to do more than the bare minimum seen in consumer electronics. The hope for the future is that With this wearable stress detection device, users information will be provided to a machine learning algorithm in order to better understand, what causes a person stress and how often they feel stressed. From the results, the device will be able to help the user feel more relaxed and in control of their stress levels.

The device will come equipped with four sensors and a switch that can be pressed when a person feels stressed. This will provide the machine learning program with accurate information. Currently, most smartwatches monitor sudden elevation in heart rate to detect stress levels, but that is not sufficient [1]. That is why our device will come with four separate sensors. This product will be aimed towards users who have a hard time controlling their stress levels. It will mainly be used to detect acute stress. This product's intent is not to stop the user from experiencing stress all together, it is merely a coping mechanism. Giving users a way to track high stress level activities.

1.4 Timeline/Proposed Timeline

44X GANTT CHART

PROJECT TITLE	Stress Device Project
RED = Crit Path	Green = complete

WBS NUMBER	TASK TITLE	TASK OWNER	START DATE	DUE DATE
1	**Research and Design**			
1.1	Block Diagram Inital Layout	Everyone		10/22/21
1.1.1	Block Diagram Revisions	Everyone	10/24/21	10/29/21
1.2	Research	Everyone	10/31/21	11/5/21
1.2.1	Pcb Size and Microcontroller	Hamad A	11/7/21	11/11/21
1.2.2	Power source(expected output, size design)	Arthur K	11/11/21	11/18/21
1.2.3	Measuring Electrical Consumption	Kris H	11/14/21	11/18/21
1.2.4	Heart sensor	Kris H	11/21/21	11/25/21
1.2.5	GSR sensor	Jimmy P	11/28/21	12/2/21
1.2.6	Temperature Sensor	Hamad A	12/2/21	12/2/21
1.2.7	Final circuit diagram	Jimmy P	12/5/21	12/9/21
1.2.8	Simulations	Kris H	12/12/21	12/16/21
1.2.9	Bill of Materials	Arthur K	12/19/21	12/23/21
1.3	TBD			
1.4	TBD			
2	**Project Management**			
2.1	Scope and Goal Setting	Arthur K		
2.2	Group Meetings	Arthur K		
2.3	Communication Plan	Arthur K		
3	**System Integration**			
3.1	Ordering Parts	Everyone		
3.2	Order Testing(Arduino)	Everyone		12/23/21
3.2.1	Code temp senor	Everyone	12/26/21	12/31/21
3.2.2	Assemble/test Temperature sensor	Jimmy P	1/1/22	1/6/22
3.2.3	Code heart sensor	Everyone	1/6/22	1/6/22
3.2.4	Assemble/test heart sensor	Hamad A	1/9/22	1/13/22
3.3	Code GRS sensor	Everyone	1/16/22	1/20/22
3.4	Assemble/test GRS sensor	Arthur K	1/23/22	1/27/22
4	**Project Presentation**			
4.1	Project Objectives	Everyone	3/1/22	4/9/22
4.2	Video Recording /Editing	Kris H or Arthur K	3/1/22	4/9/22
4.3	Quality Deliverables	Everyone	3/1/22	4/9/22
4.4	Project Performance	Everyone	3/1/22	4/9/22

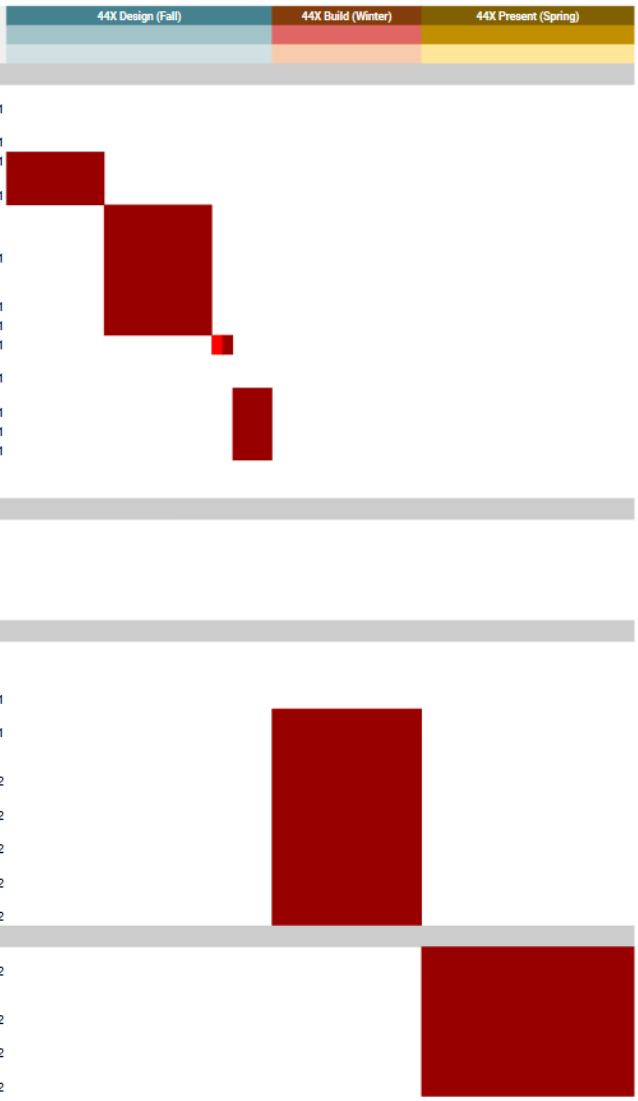


Figure 1: Project timeline

1.5 References and File Links

References

[1] J. Norreel. "How do smartwatches help manage stress?" digitalhealthcentral.com https://digitalhealthcentral.com/2020/10/11/how-to-manage-the-stress-with-a-smartwatch/ (accessed Oct. 16th, 2021).

[2] "Monitor your heart rate with Apple Watch," *Apple Support*, 14-Oct-2021. [Online]. Available: https://support.apple.com/en-us/HT204666. [Accessed: 20-Oct-2021].

1.5.1 File Links

1.6 Revision Table

10/22/2021	Everyone: First draft of section 1
11/10/21	Kris: Revised Executive Summary, Formatting on 1.5
11/10/2021	Hamad: Revised the team protocols and standards
11/12/2021	Kris: Revised formatting on team contact information.v
11/12/2021	Jimmy: Revised Gap Analysis and made Team Standards table bigger
11/12/2021	Arthur: Added visual clarity to timeline
11/19/2021	Kris: Revised visuals for timeline.
11/29/2021	Kris: Revised executive summary to mention a user switch rather than a button
12/3/2021	Kris: Updated timeline.
05/05/2022	Jimmy: Updated Gap analysis

2 Requirements, Impacts, and Risks

2.1 Requirements

2.1.1 PPR: The system must collect data every minute.
Requirement: The system will collect data from the sensors once every minute (2 second tolerance).

Verification Method:

1. Observe the timestamps on the data sent by the microcontroller to the destination webpage.

If the data sent out by the device has a one-minute difference with the previous item for at least 10 continuous minutes, then the requirement is met.

2.1.2 PPR: The sensors must accurately collect data.
Requirement: The margin of error from the sensors will vary based on sensor. For temperature, the margin cannot exceed 3 degrees C. For GSR, the margin cannot exceed 10% of the accepted value. For heart rate, the average heart rate collected in a 30-minute trial cannot exceed 5% error or 5 BPM (whichever is larger).

Verification Method:

1. For temperature, the user's temperature will be taken with the sensor, while at the same time by an external thermometer.
2. For heart rate, an external heart rate monitor will be used as the accepted value.
3. For galvanic skin response, the resistance readings from the sensor will be compared with those obtained with a pre-built module.
4. Every minute, the accepted values will be written down.
5. After 10 minutes, begin to compare sensor and accepted values.

If the sensor values are within the range of the tolerance relative to the accepted values, then the requirement is met.

2.1.3 PPR: The system must be able to sit on a person's wrist.
Requirement: The enclosure for the whole system cannot exceed two inches in width.

Verification Method:

1. Using a measuring tape or ruler, the width of the device will be measured.

If the measured value is 2 inches or less, then the requirement has been met.

2.1.4 PPR: The system must last at least five hours.
Requirement: The system must have a battery life of at least five hours.

Verification Method:

1. Ensure that the system is fully charged.
2. Begin timing on a stopwatch.
3. Continue to keep track of time until the device stops logging data on the webpage.

If the device logs data for five hours or more, then it can be assumed that the system has been powered for at least five hours and therefore, the requirement is met.

2.1.5 PPR: The system will send information wirelessly.
Requirement: User most be able to access information wirelessly, within a distance of 10 feet.

Verification Method:

1. Using a measuring tape, ensure that the system and the receiver are 10 feet apart.
2. Visit the webpage where sensor data will be displayed to
3. Inspect the timestamps on the data sent.

If the webpage is logging data at the correct intervals (1 minute) for 10 consecutive minutes, then the requirement is met.

2.1.6 PPR: If the system does not collect any data, it must turn off.
Requirement: The system will shut down if no data is collected for five consecutive minutes.

Verification Method:

1. The system will consider data not being collected based on the values of temperature.
2. As there is a large difference between room and body temperature, every minute where the logged temperature is less than 28 degrees Celsius the system will increment a counter.
3. For every consecutive minute logging room temperature, the count will continue to increase. Once the count reaches five, the device will enter sleep mode, and no longer transmit data.
4. Verify that no data is being transmitted by waiting 3 minutes and observe the last timestamp on the project website

If the system logs data at the required intervals after conducting the procedure, then the requirement is met.

2.1.7 PPR: The user must be able to tell the system when they are feeling stressed.
Requirement: The system should have a module that the user can interact with, to signify when they are feeling stressed. It should also display the input. Verification Method:

1. Over a 10-minute period, the built-in switch is going to be flipped between the 'stressed' side and 'non-stressed' side. We will record what side the switch is on and for how long.
2. After the 10-minute period, the logged data will be compared to the recorded data.

If the recorded data and the logged data match, that means the microcontroller was able to differentiate the different switch modes. Therefore, the requirement is met.

2.1.8 PPR: The system's battery will be rechargeable.
Requirement: The system's battery will be fully charged under 90 minutes.

Verification Method:

1. Assume that the system's battery is out of charge if it does not power on after 10 minutes of attempting to.
2. Connect the USB interface to power.
3. Using an onboard LED to display the battery status, observe the LED behavior (blinking for charging, solid for full).
4. Once the LED indicates a full charge, use the device until the battery is out of charge.

If at the end of the charge, the device battery life meets requirement 2.1.4, then this requirement is met.

2.2 Design Impact Statement

Everybody has felt stressed at one point in their lives. It can be due to an impending dead- line, or an unexpected event causing the stress. However, it is not always a straightforward process to determine if a person is stressed and what may be causing it. This is the problem that this project is aiming to solve.

With regard to the safety in the manufacturing process for this project, the ethical dilemma of sourcing parts cannot be ignored. Starting with batteries, Lithium-ion batteries are linked to human rights abuses, including child labor in the Democratic Republic of Congo [2]. This is not an ethical problem that applies to this project, but to the entire battery industry as the world works to transition away from fossil fuel power. Trading human rights for a smaller carbon footprint does not uphold the IEEE code of ethics. Currently, no country requires that a company reports their cobalt(key metal in batteries) supply chains, which hinders the ability for any enforcement [1]. Therefore, much of the materials for the device are not ethically sourced, adding a massive utilitarian cost to the project.

The Cultural and Social Impacts from this device vary on the demographics of users. As mentioned by the project partner, this device is intended to be used by researchers and professionals in industry. Assuming said experts share the same intentions as the device, they intend to use the device for better preemptive diagnosis of stress to begin helping a patient before it starts to become more harmful for their mental well-being.

A common impact in all electronic devices now is the materials used to make them and the impact they have on the environment. While the battery implemented into the project is rechargeable, allowing for the user to use the device for the lifetime of the battery, once the battery has completely degraded, disposing of it harnesses and environmental impact.

Many consumer products detect stress based off physical responses in the body, such as heart rate. But they are not necessarily the direct response of stress [3]. This is where the project can make an economic disruption due to the lack of accurate products that are currently available. A device developed by [4] is a rival to the project, as galvanic skin response is a novel sensor for stress detection when compared to another common sensor such as heart rate. With many of these devices still being in the development phase, the first of these to come out and function successfully is sure to make an impact.

It is important to remember the purpose of the device in spite of the harms it brings. Stress detection devices are a relatively new market, and a successful implementation can lead to aiding medical professionals in diagnosing physiological problems well in advance, allowing for patient treatment sooner than what was previously possible.

2.3 Risks

Risk ID	Risk description	Risk category	Risk probability	Risk impact	Performance indicator	Responsible party	Action plan
R1	Sensor gear delay	Timeline	25%	Medium	Timeline change	Jimmy	Retain
R2	One sensor fails	Timeline	25%	Medium	Error in simulations/data	Arthur	Prepare
R3	Enclosure discrepancy	Technical	50%	High	Dimensional errors in the enclosure	Hamad	Avoid
R4	PCB delay	Timeline	25%	Medium	Timeline change	Arthur	Prepare
R5	Teammate illness	Timeline	50%	High	COVID or other flu's	Hamad	Reduce
R6	Components out of stock	Timeline	70%	High	Components needed are not available	Jimmy	Prepare
R7	Burning the microcontroller	Technical	30%	High	Microcontroller burning due to short circuit or other reasons	Kris	Prepare
R8	PCB error	Technical	60%	High	System behavior is different than when previously tested	Kris	Avoid

Figure 2: Project risks

2.4 References and File Links

2.4.1 References

References

[1] "Amnesty challenges industry leaders to clean up their batteries," *Amnesty International*. [Online]. 21-Mar-2019. Available: https://www.amnesty.org/en/latest/press-release/2019/03/amnesty-challenges-industry-leaders-to-clean-up-their-batteries/. [Accessed 29-Oct-2021].

[2] "'This is what we die for': Human rights abuses in the Democratic Republic of the Congo power the global trade in cobalt" *Amnesty International*. 19-Jan-2016. Available https://www.amnesty.org/en/documents/afr62/3183/2016/en/. [Accessed 29-Oct-2021].

[3] Samson Cheyenne, Koh Ahyeon "Stress Monitoring and Recent Advancements in Wearable Biosensors," Frontiers in Bioengineering and Biotechnology. 2020. https://www.frontiersin.org/article/10.3389/fbioe.2020.01037

[4] Lee HB, Meeseepong M, Trung TQ, Kim BY, Lee NE. "A wearable lab-on-a-patch platform with stretchable nanostructured biosensor for non-invasive immunodetection of biomarker in sweat." *Biosens Bioelectron*. 2020 May 15;156:112133. doi: 10.1016/j.bios.2020.112133. Epub 2020 Mar 6. PMID: 32174559.

2.4.2 File Links

2.5 Revision Table

11/10/2021	Kris: Revised Requirements subsection
11/11/2021	Jimmy: Added in table to Revision Table section
11/11/2021	Arthur: Revised the risks table
11/11/2021	Hamad: Revised the risks table
11/19/2021	Kris: Changed formatting in requirements section
11/19/2021	Jimmy: Changed formatting in requirements section and added PPR.
12/3/2021	Hamad: Risk table revised.
05/05/2022	Kris: Revised testing methods, changed requirement 2.1.2 as approved by Don.

3 Top-level Architecture

3.1 Block Diagram

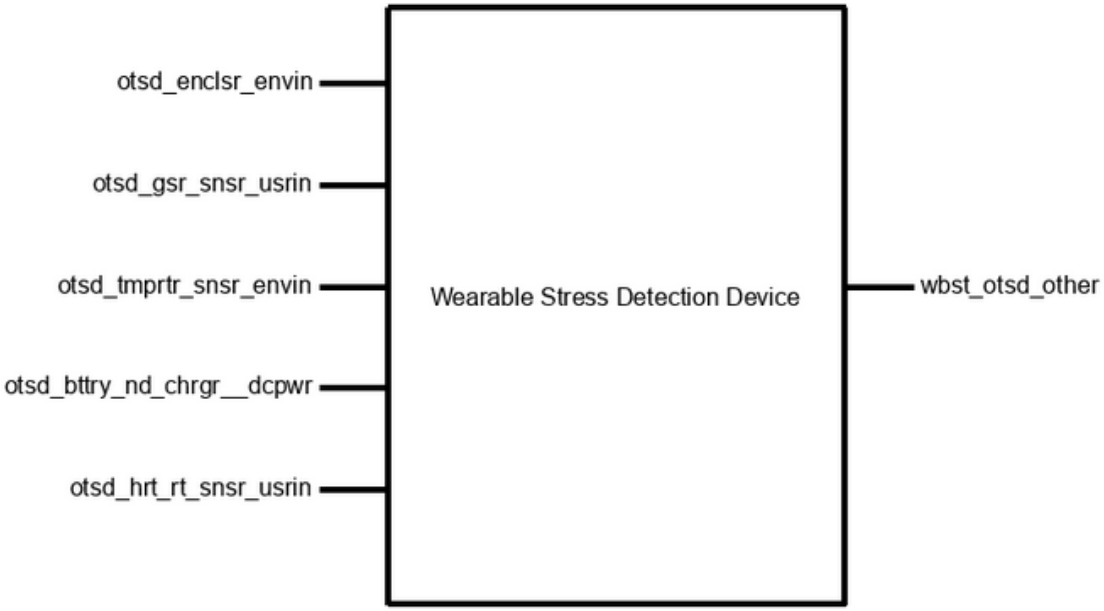


Figure 3: Black Box Diagram

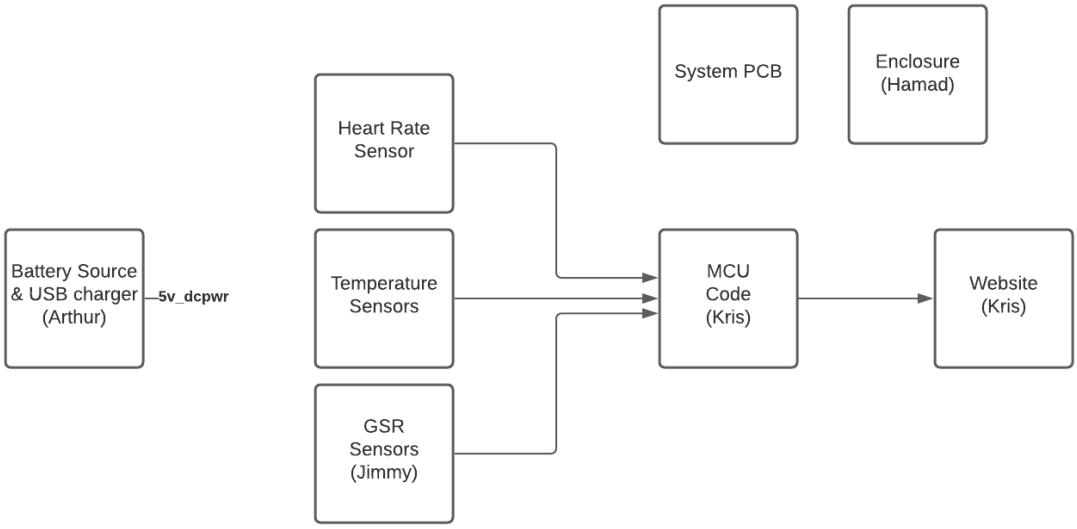


Figure 4: Block Diagram

3.2 Block Description

Name	Description
GSR Sensor Champion: Jimmy Parra	The purpose of the GSR sensor block is to create and implement a working GSR module. The module will detect changes in the electrical conductance of the skin. We are able to do this by taking advantage of the way that the resistivity of the skin changes based on sweat gland activity. Greater sweat gland activity decreases the resistivity of the skin and vice-versa. This data will then be sent to the microcontroller for processing.
Enclosure Champion: Hamad alali-Hamad	Alali is assigned this block, which will be the enclosure block which is going to be very essential for the system since it is going to be the protection of the system as well as its going to be designed in a way which will have the perfect size for it to be on a person's wrist. The enclosure will serve as a protection for all the PCBs and power supply as well as the systems sensors which will be enclosed in the enclosure. The system will be worn by the users on the wrist, by wrapping a strap around the user's hand, which will make the enclosure secure in the users' hand.
Temperature sensor Champion: Hamad alaliHamad	Alali is assigned this block, which will be a temperature sensor block which is going to be very essential for the system since it is going to be collecting temperature data from the user using a TMP 36 sensor, to determine the stress level of the user. The temperature sensor (TMP 36) is going to function by getting readings of the skin of the user and sending this data to the microcontroller through the system PCB. The process will be done by having the sensor in a bracelet which is worn by the user, and the sensor will have direct contact to the user so that the data retrieved is accurate. Then the data will be processed by the website to determine the stress level of the user.
Battery and Charger Champion: Arthur Kichatov	The battery and charger block will supply power to the entire system. This design will utilize DC power from a 9v battery which can be replaced once the battery is "dead". This module will utilize a voltage regulator rather than a buck converter for simplicity in housing this module as it will need to be small to fit all the other modules on a single wrist. This voltage regulator in specific is the L7805 Voltage Regulator which is a positively fixed linear voltage regulator that outputs through a usb 1.5 amps and 5 volts.
Website Champion: Kris Haro	This block will be a user interface for our client to see the relevant sensor data. It will retrieve the information from a MySQL database using PHP and display the data on graphs using JavaScript. Most importantly, the website will be able to export the data via CSV file to be processed using the project partner's machine learning program.
Heart Rate Sensor Champion: Jimmy Parra	The purpose of the Heart Rate block is to create and implement a working heart rate module. It is responsible for taking an individual's heart rate and outputting the data which will be processed by the microcontroller. The sensor measures heartbeat by shining a green light (500nm) on the skin and measuring the amount of reflected light using a photosensor. As the blood is pumped the amount of light reflected changes. A green light is chosen because the oxygenated hemoglobin can absorb green light.
System PCB Champion: Arthur Kichatov	This block will be the circuit board for the microcontroller as well as the circuitry required for the sensors. This block represents the final PCB of the whole system. Before adding to the system each block will have their own PCB design that will be tested before inclusion.
MCU Code Champion: Kris Haro	This block will be the code for controlling the microcontroller system. It will be responsible for reading the sensor inputs, processing them into readable values, and logging the data at the required intervals. It will also be responsible for shutting the system off when it has been inactive for less than 5 minutes.

3.3 Interface Definitions

Name	Properties
otsd_enclsr_envin	**Other:** the enclosure will be not wider than 2 inches(except for the place of the strap) **Other:** the enclosure will be wearable. **Other:** The enclosure will withstand vigorous user movement in all directions
otsd_gsr_snsr_usrin	**Other:** Skin contact **Other:** Two connection points **Type:** Resistance
otsd_tmprtr_snsr_envin	**Other:** Sensing range: within contact **Other:** Type: temperature **Temperature (Absolute):** (70F-109F) 21C-42C
otsd_bttry_nd_chrgr_dcpwr	**Inominal:** 80ma **Ipeak**: 0.5a **Vmax:** 9v **Vmin:** 7v
otsd_hrt_rt_snsr_usrin	**Other:** One connection point **Other:** Skin contact **Type:** Heart Rate
gsr_snsr_mc_cd_data	**Datarate:** Every 30s **Messages:** Resistance **Other:** Integer
tmprtr_snsr_mc_cd_data	**Datarate:** Every 30s **Messages:** Temperature **Other:** Integer
hrt_rt_snsr_mc_cd_data	**Datarate:** Every 30s **Messages:** Beats per Minute **Other:** Integer
“ mc_cd_wbst_data	**Datarate:** Transmission every 60 seconds **Messages:** User heart rate, user body temperature, user galvanic skin response, switch input, current time. **Other:** SQL table values
wbst_otsd_other	**Other:** Presented format: Temperature, Heart Rate, GSR, Switch on/of, Timestamp **Other:** Website Feature: Graphs the temperatures, heart rate, and GSR over time **Other:** CSV File

3.4 References and File Links

3.4.1 References

3.4.2 File Links

LM324
- MAX30102
- SEN0203
- TMP 36

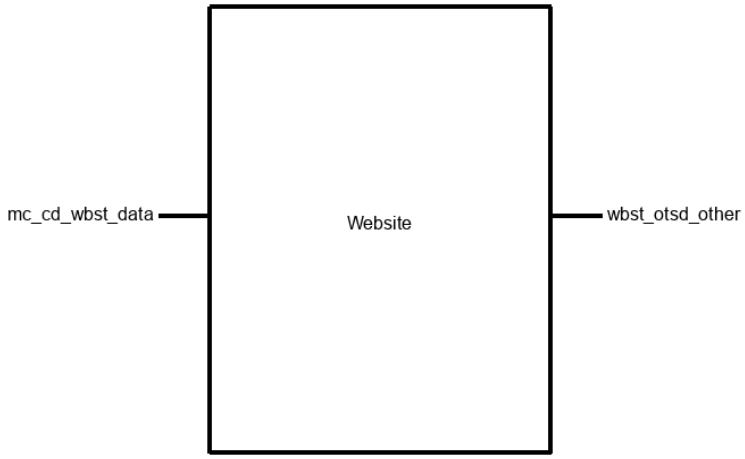


Figure 5: Website black box diagram

3.5 Revision Table

11/17/2021	Kris: Created Section 3
11/18/2021	Jimmy: Made table for interface descriptions and worked on block descriptions.
11/18/2021	Hamad: worked on table of interfaces and worked on block descriptions.
11/18/2021	Arthur: Worked on timeline block descriptions and risks.
11/19/2021	Kris: Added block descriptions to document.
11/29/2021	Jimmy: Updated block descriptions and added hyperlinks.
12/3/2021	Kris: Updated section 3 based on feedback received.
05/05/2022	Kris: Updates sections 3.1, 3.2, 3.3 to match what is on the student portal.
05/05/2022	Jimmy Parra: Updated sections 3.2 and 3.3

4 Block Validations

4.1 Website

4.1.1 Description

The block in question is the website, with Kris Haro being the block champion. This block will be a user interface for our client to see the relevant sensor data. It will retrieve the information from a MySQL database using PHP and display the data on graphs using JavaScript. Most importantly, the website will be able to export the data via CSV file to be processed using the project partner's machine learning program.

4.1.2 Design

Black Box Diagram
Pseudocode
Website HTML Structure

```
<body>
    PHP code to interact with database (see below)
    <chart div>
    Javascript code to graph the key data
    </chart div>
    <button>
    Press this button to export data.
    Javascript function to create CSV file.
    </button>
    <button>
    Press this button to manually enter new data.
    Javascript function to redirect to entry form.
```

```
    </button>
</body>
```

PHP Pseudocode

```
    variables for SQL database credentials


    create SQL connection
    check for successful connection
    select SQL variables from table
    execute SQL query
    while we have not reached the last row in the table
        $row_id = $row["id"]; // get the value from the id column at row i
        $row_bodyTemp = $row["bodyTemp"]; // repeat for all variables
        ..
        store the retrieved values in arrays
    free memory associated with result
    close SQL connection
```

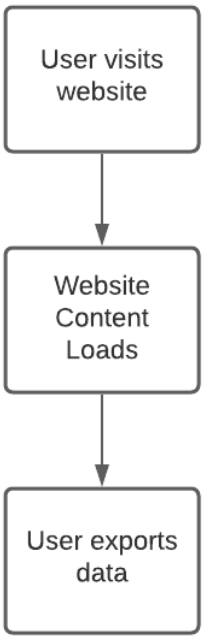
Javascript Pseudocode

```
    // function to graph data
    get div that will contain the chart
    create new Chart object
    // insert necessary data into chart object
        chart type: 'line';
        data:{
            name: <?php echo $bodyTemp?>
            dataset: [<?php echo $bodyTempArray?>]
        };
        set aspect ratios and colors
    // function end

    // function to export as CSV
    getElementByID(Export Button);
    bool clicked = false;
    Once the button is clicked:
        Insert the table variables into one array
        var csv
        while we have not reached the end of the array
            add arrray[i] to csv row
            create new row
        download the file named 'data.csv'

    //function to manually add a new form
```

Website Flowchart



4.1.3 General Validation

This block will aid in meeting the needs of the project, as it is the end user interface that will be used by the project partner. It directly aids to meet requirements listed in the project document. The user interface consisting of no hardware or proprietary software helps in keeping the overall cost of the project low (while not an explicit requirement, it is a guideline). Having created a website of similar structure, tuning it to meet the needs of the project can be done by the deadline.

4.1.4 Interface Validation

Interface Property	Why is this interface this value?	Why do you know that your design details for this block above meet or exceed each property?

mc_cd_wbst_data: Input

Datarate: Transmission every 60 seconds	This is a project requirement	This property will be met as it is programmed to do so.
Messages: User heart rate, user temperature, user galvanic skin response, switch input, current time.	These messages were chosen based on the needs of the project partner.	These messages will be sent as they are inputs collected by other blocks.
Other: SQL table values	An SQL table was chosen due to being an intermediate way to verify data transmission.	This property will be met as the website executes SQL queries.

wbst_otsd_other: Output

Other: Presented Format: Temperature, Heart Rate, GSR, Switch Input, Timestamp	This interface is in this format to meet the project partner's needs.	The website will have a function to export data in this format
Other: CSV file	This interface is a CSV file to meet the project partner's needs.	The website will have a function to export data as a CSV file.
Other: Website Feature: graphs the temperature, teart rate, and GSR, over time	This feature exists to give visualizations to the data.	The website will have these graphs as it is coded to do so.

4.1.5 Verification Process

1. Using this website, manually enter 10 test values into the database.
2. The previous step is done as inputs depend on other blocks.
3. Verify communication between microcontroller and database using the PHPmyAdmin interface.
4. This verifies that mc_cd_wbst_data works as intended.
5. Visit the website here to export as CSV.
6. Inspect the CSV file to verify that wbst_otsd_other meets the interface properties.

4.1.6 References and File Links

References
File Links

4.1.7 Revision Table

1/5/22	Kris: First draft.
1/21/22	Kris: Revised second draft Changed the interfaces to be more in line with what is being sent from the database Interface validation changed as a result Revised block testing process to not require a microcontroller Added flowcharts and pseudocode
3/2/22	Kris: Integration into project document. Updated interfaces that were outdated.

4.2 MCU Code

4.2.1 Description

The block in question is the microcontroller code. The block champion is Kris Haro. This block will consist of all the code used to control the ESP32 used in the project. Operations of the microcontroller include, but are not limited to: time tracking, reading analog inputs, and wirelessly transmitting data to a remote database.

4.2.2 Design

Black Box Diagram Pseudocode
Note that some areas of the pseudocode are direct C++ code, as needed to better emphasize a specific process.

```
// declare neccessary functions
void readNLog(); // used to insert into SQL table
void valueRead(); // used to collect analog values

// create ticker objects for time tracking
```

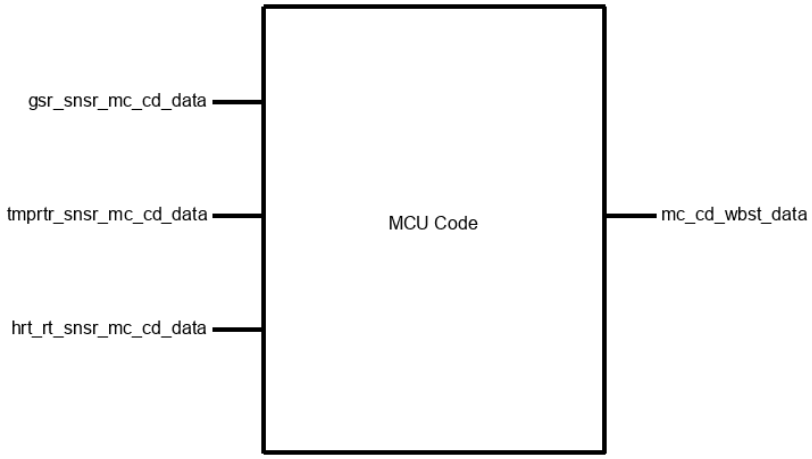


Figure 6: Black box diagram for firmware.

```
TickTwo timer1(readNLog, 1000*60); // once, every minute
every minute, the readNLog function will execute
TickTwo timer2(valueRead, 5); // repeat every 5 ms
every 5ms, valueRead executes

// variables for data collection

void setup(){
    timer1.start(); // initialize timers
    timer2.start();

    // connect to Wi-Fi network
    Wifi.begin(ssid, password);
    ensure connection
}

void loop(){ // int main
    timer2.update();
    timer1.update();
    // nothing should really happen in main
    // all the work in scheduled around ticks
    // for efficiency
}

void valueRead(){ // read analog values
    read GSR, temperature, and heart rate
    get the average after 15 iterations // abritrarily chosen
    reset iteration count
    discard the average if it is bad
}

void readNLog(){ // insert into database
    ensure wifi connection
    begin http request

    // prepare http post request data
    condense all sensor values into one string
    send HTTP post request
    check for response
    free resources using http.end();
}
```

4.2.3 General Validation

The design of the block fits the needs of the system, as it is the firmware that dictates the operations of the device. This is critical as this block will ensure many project requirements are met. The clock configuration shown in the pseudocode ensures that data is logged once a minute, fulfilling one requirement. The code for wireless transmission fulfills another requirement. The periodic checks for good data aid in meeting a third requirement.

The code is not entirely responsible for meeting the three requirements. As the firmware can only process based on the inputs it has been given from other blocks.

4.2.4 Interface Validation

Note that many of the interfaces are going to be reworked and will may not reflect what is on the student portal when this is graded.

mc_cd_wbst_data: Output

Interface Property	Why is this interface this value?	Why do you know that your design details for this block above meet or exceed each property?
Datarate: Transmission every 60 seconds	This is a project requirement	This property will be met as it is programmed to do so.
Messages: User heart rate, user temperature, user galvanic skin response, switch input, current time.	These messages were chosen based on the needs of the project partner.	These messages will be sent as they are inputs collected by other blocks.
Other: SQL table values	An SQL table was chosen due to being an intermediate way to verify data transmission.	This property will be met as the website executes SQL queries.

gsr_snsr_mc_cd_asig: Input

Datarate: Every 30s	This property was chosen based on project requirements.	This property will be met as it is programmed to do so.
Other: Integer	This interface is an integer to show software communication	The interface will be an integer as it is programmed that way.
Messages: Resistance	Galavanic Skin Response is a person's resistance.	The code used to process the signal will pass on the calculated resistance.

tmprtr_snsr_mc_cd_asig: Input

Datarate: Every 30s	This property was chosen based on project requirements.	This property will be met as it is programmed to do so.
Other: Integer	This interface is an integer to show software communication	The interface will be an integer as it is programmed that way.
Messages: Temperature	The block outputting this interface measures temperature	The code used to process the signal will pass on the measured temperature.

hrt_rt_snsr_mc_cd_asig: Input

Datarate: Every 30s	This property was chosen based on project requirements.	This property will be met as it is programmed to do so.
Other: Integer	This interface is an integer to show software communication	The interface will be an integer as it is programmed that way.
Messages: Heart Rate	The block outputting this interface detects heartbeats	The code used to process the signal will pass on the calculated heart rate.

4.2.5 Verification Process

1. Since the inputs are sensor values from other blocks, placeholder values will be used to emulate those inputs.
2. The code file will have three arrays of size five for temperature, heart rate, and galvanic skin response.
3. The values of the array will be substituted in locations where one would expect to read from sensors.
4. Once the necessary preparations have been made, ensure Wi-Fi connection by inspecting the serial monitor.
5. Data should be getting logged to the database every minute, this can be verified by inspecting the web interface for the database.
6. Ensure continuous data transmission over five minutes.

4.2.6 References and File Links

References
File Links
ESP32 datasheet
TMP36 temperature sensor
Grove GSR Sensor

4.2.7 Revision Table

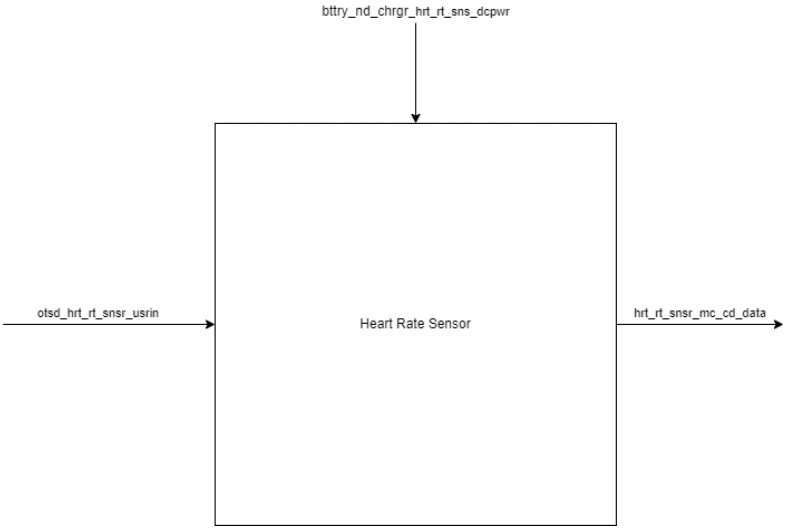
2/4/22	Kris: First draft.
2/17/22	Kris: Second draft with revisions.
Feedback Addressed from Noah	Deleted the flowchart as it added confusion Removed inconsistencies in reference to which microcontroller was being used. Revised testing process to no longer use test cases.
Feedback Addressed From Yuhao	Explained what the ESP32 is. Made the pseudocode easier to understand.
3/2/22	Kris: integration into project document Updated interfaces and design sections that were outdated/changed

4.3 Heart Rate Sensor

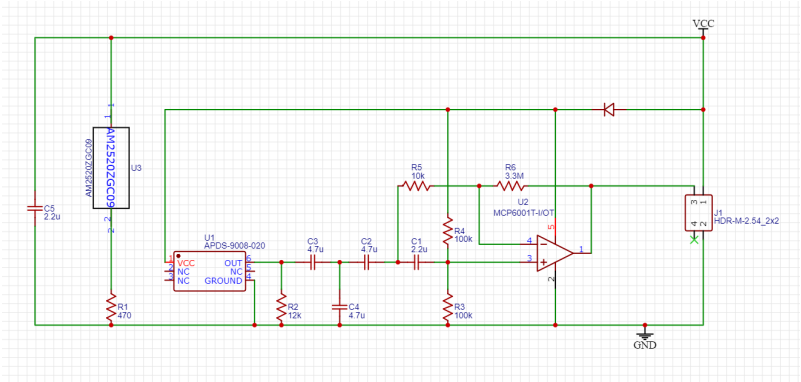
4.3.1 Description

The purpose of the Heart Rate block is to create and implement a working heart rate module. It is responsible for taking an individual's heart rate and outputting the data which will be processed by the microcontroller. The sensor measures heartbeat by shining a green light (500nm) on the skin and measuring the amount of reflected light using a photosensor. As the blood is pumped the amount of light reflected changes. A green light is chosen because the oxygenated hemoglobin can absorb green light.

4.3.2 Design



Black Box Diagram



Schematic

4.3.3 General Validation

This design will meet the block requirements because it will have the ability to detect the heartbeat of a user, it will send this information to the micro controller by using the analog input. This also satisfies the goal that our project partner had given us. We are currently going to use the pulse sensor design. This design was chosen because it is open sourced and is extremely well documented. It is not expensive and we should not run into any parts availability issues. A possible alternate solution to this block is the Gravity: Heart Rate Monitor Sensor. While this design is also open sourced it is not as well documented as the pulse sensor. However, it also seems like it will be reliable and it is not expensive either.

4.3.4 Interface Validation

Interface Property	Why is this interface this value?	Why do you know that your design details for this block above meet or exceed each property?
otsd_hrt_rt_snsr_usrin: Input		

Other: One connection point	This was chosen because the sensor requires at most one solid connection point.	We know that this design will meet this property because in the data sheet it is advertised as being able to measure heart rate from a single point.
Other: Skin contact	This was chosen because the sensor will be connected to a finger.	We know that this design will meet this property because in order to measure heart beat it must be on the ski. This way it can measure the reflected light.
Type: Heart Rate	This was chosen because it is what the sensor will be recording.	In the provided data sheet under the general description. It states that this sensor is used to measure heart rate.

hrt_rt_snsr_mc_cd_asig : Output

Datarate: Every 30s	This was chosen because the sensor will only have to output data every 30 seconds	We know that this will be met because the sensor is running at a baud rate of 9600, allowing us to gather enough information withing the 30 seconds.
Message: Beats per minute	This value was chosen because it is the data we want to collect	If we collect the number of beats within 30 seconds, multiplying the value by two will give us the total beats in one minute.
Other: Integer	This was chosen because the micro controller will need to send an integer to the website block.	The detected voltages will have a "peak" each peak will be counted as one beat. Allowing us to send the number of beats as an integer.

4.3.5 Verification Plan

1. Hook up the heart rate sensor to the micro controller and plug it into the computer.
2. Run the code and open the serial monitor.
3. Show that no data is read when there is not a connection point.
4. Place one connection point on the sensor and verify information is being received.
5. Manually check heart rate by placing fingers on wrist and counting the number of beats in one minute.
6. Check the calculated value by the sensor.
7. Compare both values.
8. Verify that heart beat is the message being outputted and verify that an integer is being received every 30 seconds.

4.3.6 References and File Links

Monitor the Heart Rate using Pulse Sensor and Arduino.
Data sheet

4.3.7 Revision Table

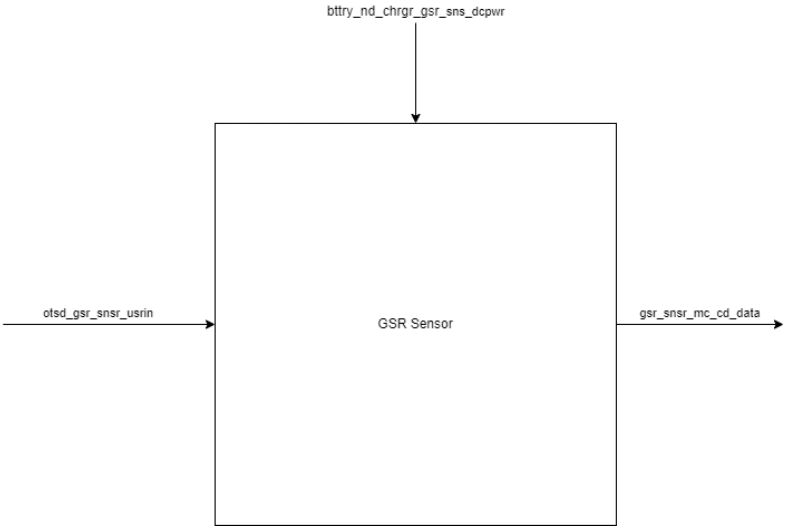
01/31/2022	Document Created
01/31/2022	Sections 1 through 7 composed and drafted
02/14/2022	Schematic and black box revised
02/16/2022	Sections 1 and 3 revised
02/23/2022	Updated Interface validation and verification plan based on changes

4.4 GSR Sensor

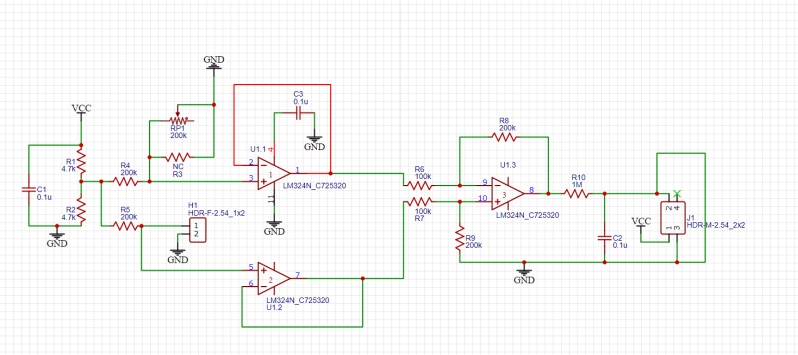
4.4.1 Description

The purpose of the GSR sensor block is to create and implement a working GSR module. The module will detect changes in the electrical conductance of the skin. We are able to do this by taking advantage of the way that the resistivity of the skin changes based on sweat gland activity. Greater sweat gland activity decreases the resistivity of the skin and vice-versa. This data will then be sent to the micro controller for processing.

4.4.2 Design



Black Box Diagram



Schematic

4.4.3 General Validation

This design will meet the needs of the system and the block because it has the capability of measuring resistivity of the skin and that is what the project partner has asked us to do. Placing two electrodes on two separate fingers and applying a constant voltage (typically 0.5 v) allows the system to measure the electrical conductance of the skin. The model that we are using is connected to the fingers because it is a bare surface and there is less skin thickness, as shown in figure 3, allowing us to get a more accurate reading. By plotting the information gathered visual spikes will be present, these spikes in the data correspond to changes in emotion. This system is only responsible for gathering the information and sending it to the micro controller for processing and it has the capability of doing that because it is connected via a voltage port on the micro controller. To create this system we will be using the LM324 chip, this chip has been mass produced so there

should not be an issue acquiring it. Besides that we will be using common resistor and capacitor values so there shouldn't be any issues with those either. However an alternative chip could be the LM6134.

4.4.4 Interface Validation

Interface Property	Why is this interface this value?	Why do you know that your design details for this block above meet or exceed each property?
otsd_gsr_snsr_usrin: Input		
Other: Skin Contact	This interface is added because in order for the electrodes to work it must have skin contact	In the datasheet it states that the electrodes must have skin contact. Preferably fingers.
Other: Two connection points	There must be two connection points in order to 'ground' the device. It should not work if there is only one.	This is what is measured when the two sensors are placed on the fingers
Type: Resistance	This is what is measured when the two sensors are placed on the fingers	From the datasheet under the 'specification' sections it states that resistance is being measured.
gsr_snsr_mc_cd_data: Output		
Datarate: Every 30s	This was chosen because the sensor will only have to output data every 30 seconds	We know that this will be met because the sensor is running at a baud rate of 9600, allowing us to gather enough information with-ing the 30 seconds.
Message: Resistance	This value was chosen because it is the data we want to collect	We know that this sensor is measuring the resistance of the skin because it is in the data sheet.
Other: Integer	This was chosen because the micro controller will need to send an integer to the website block.	The detected voltages will have be averaged over the 30 second interval. This in turn will create an integer that can be sent by the micro controller

4.4.5 Verification Plan

1. Hook up the GSR sensor to the micro controller and plug it into the computer.
2. Run the code and open the serial monitor.
3. Place one connection point on the sensor and verify no information is being received.
4. Place two connection points on the sensor and verify that information is being received
5. Using two electrodes that are placed on each finger we can measure the resistance of the skin. Using one resistance, in series with the skin resistance, to form a voltage divider [2].
6. Using the equation $V_0 = V_s \frac{R_2}{R_2+R_s}$ the resistance of the skin will drop the voltage value.
7. Using a multimeter verify that there was a voltage drop, proving that resistance is being detected.
8. Verify that resistivity is the message being outputted and verify that an integer is being received every 30 seconds.

4.4.6 References and File Links

Grove GSR Sensor
A Stress Sensor Based on Galvanic Skin Response (GSR) Controlled by ZigBee
Data sheet

4.4.7 Revision Table

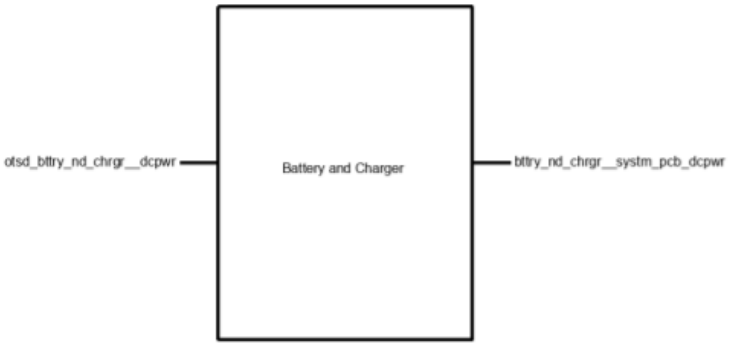
01/03/2022	Document created
01/03/2022	Sections 1-2 updated
01/05/2022	Sections 4-5 started
01/07/2022	Changed some of the interfaces and their verification's
01/19/2021	Document revised based on feedback
02/23/2022	Updated Interface validation and verification plan based on changes

4.5 Power Block

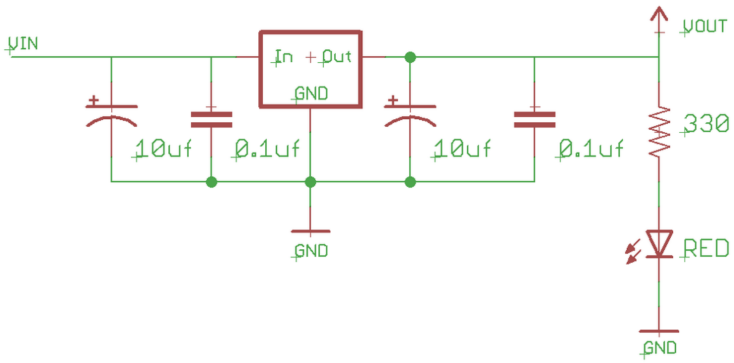
4.5.1 The battery and charger block will supply power to the entire system. This design will utilize DC power from a 9v battery which can be replaced once the battery is "dead". This module will utilize a voltage regulator rather than a buck converter for simplicity in housing this module as it will need to be small to fit all the other modules on a single wrist. This voltage regulator in specific is the L7805 Voltage Regulator which is a positively fixed linear voltage regulator that outputs through a USB 750 ma and 5 volts.

4.5.2 Design

4.5.3 Black Box Diagram



Schematic



Battery Schematic

4.5.4 General Validation

This block works because it relied on the L7805 Voltage Regulator data sheet which will convert 9v provided by the battery to 5v output for the microcontroller. The components, the voltage regulator, battery clips, 26 gauge wires, and USB female pin-out are relatively cheap and accessible through OSU sources like Texbots and Resistore or could be outsourced through online vendors.

For testing purposes, a breadboard will be used in order to measure current and voltage, and for housing purposes, it is easier to not use a PCB so that multiple molds can be discussed for the final design

4.5.5 Interface Validation

Interface Property	Why is this interface this value?	Why do you know that your design details for this block above meet or exceed each property?

Otsd_bttry_nd_chrgr_dcpwr input

Battery:	9-12 volts	From the description L7805 Voltage Regulator datasheet
Efficiency	60%	This is what is under the specification, saying All power supplies in the system must be at least 65 percent efficient

bttry_nd_chrgr_systm_pcb_dcpwr

Vout	5 Volts which is the required voltage to power the microcontroller.	From the microcontroller datasheet, which can take a max of 5v
Inominal	750ma which is the current needed to power the microcontroller.	From the microcontroller datasheet, which needs 750 ma
Vmin	3.3 Volts which is the minimum voltage requirement.	From the microcontroller datasheet, which can take a min of 3.3v

Verification Plan

1. hook up to power block and measure vout and vmin
2. measure current
3. calculate efficiency

References and File Links Voltage regulator
Microcontroller

Revision Table

01/03/2022	Document created
01/03/2022	Sections 1-2 updated
01/06/2022	Sections 4-5 started
01/25/2021	Document revised based on feedback
02/30/2022	Updated Interface validation and verification

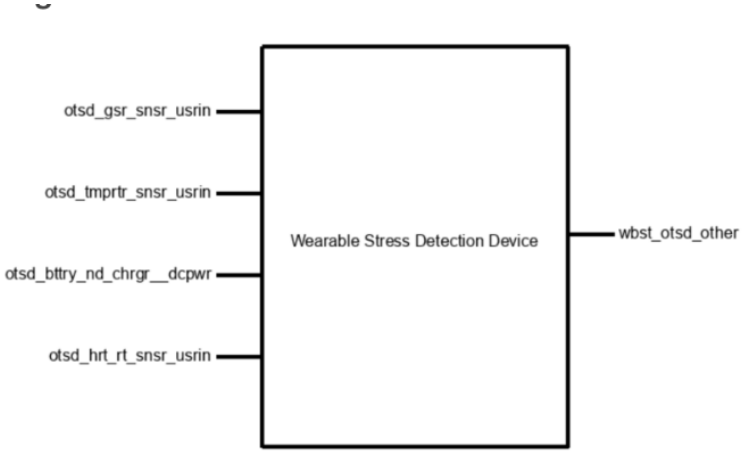
4.6 Temperature Sensor

4.6.1 Description

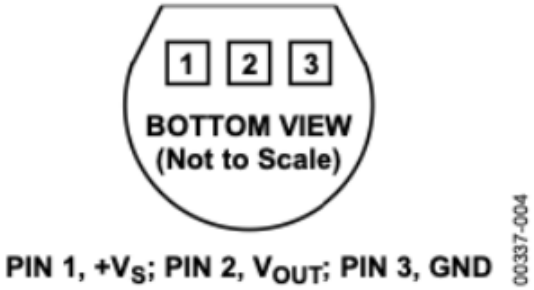
Hamad Alali is assigned this block, which will be a temperature sensor block which is going to be very essential for the system since it is going to be collecting temperature data from the user using a TMP 36 sensor, to determine the stress level of the user. The temperature sensor (TMP

36) is going to function by getting readings of the skin of the user and sending this data to the microcontroller through the system PCB. The process will be done by having the sensor in a bracelet which is worn by the user, and the sensor will have direct contact to the user so that the data retrieved is accurate. Then the data will be processed by the website to determine the stress level of the user.

4.6.2 Design



Black Box Diagram



Sensor diagram

4.6.3 General Validation

The TMP 36 sensor as chosen by the group members with the help of the project partner will fit the system that we are working on since it has an accuracy rate of about 2 percent and the size of this temperature sensor is perfect to fit in a bracelet shaped device, and the price of 5 TMP 36 sensor is 13 dollars which is considered affordable for our project, as well as the sensor is available in amazon. Since this sensor will meets all the requirements that are needed by both the project partner and the group, we as a group decided to use the TMP36 sensor as our temperature sensor.

4.6.4 Interface Validation

otsd_tmprtr_snsr_envin: Input

Interface Property	Why is this interface this value?	Why do you know that your design details for this block above meet or exceed each property?
Other: Sensing range: within contact	The range when the temperature sensor is going to accuratley read data has been experimented.	since we are going to have the sensor directly attached to users hand, which will gice the most accurate temperature reading.
Other: Type: temperature	The temperature sensor will be collecting data from both the room and user to distinguish whether there.	Since both the temperatures of the room and user are retrieved, the system will distinguish if the room temperature is affecting the user.
Temperature (Absolute): (70F-109F) 21C-42C.	since out remperature sensor will be used to determine the users temperature, and normal people temperatures are between 21C to 42C.	based on the datasheet of the temperature sensor, the temperature senor can read from -30C to 125C so out chosen values are within the range of the sensor.

tmprtr_snsr_mc_cd_data: Output

Datarate: Every 30s	This was chosen because the sensor will only have to output data every 30 seconds	We know that this will be met because the sensor is running at a baud rate of 9600, allowing us to gather enough information with-ing the 30 seconds.
Messages: Temperature	Temperature was chosen because it is what we want collect, and for the value to be processed correctly.	Wthe sensor will be mesauring the temperature based on the data sheet of the senosr.
Other: Integer	our system works with integers, and this is what is being proccesed through our code.	our system code has been created to read integers which will make everything run correclty.

4.6.5 Verification Plan

1. Buying the sensor from Amazon
2. The right leg of the sensor will be connected to the 5V pin in the Arduino.
3. The middle leg of the TMP 36 sensor will be connected to the A0 pin in the Arduino. received.
4. The left leg will be connected to the GND pin in the Arduino.
5. Connect the Arduino to the laptop by using a USB.
6. Downloading the code to operate the sensor, which will give readings every 10 seconds.
7. Using a commercial thermometer to measure my hands temperate and the data that the sensor is reading.
8. Using a commercial thermometer to measure the room temperature, then comparing it with the temperature the sensor reads.
9. To verify maximum range of the sensor, which is 125C, we will measure the temperature with a lighter from a moderate distance by a commercial thermometer and the sensor.

4.6.6 References and File Links

"Amazon.com: Customer reviews: TMP36 - temperature sensor." [Online]. Available: https://www.amazon.com/Analo
Devices-TMP36-Temperature-Sensor/product- reviews/B00JYQAIBM?pageNumber=2. [Accessed:
08-Jan-2022].
about temperature Sensors
Temperature sensor data sheet

4.6.7 Revision Table

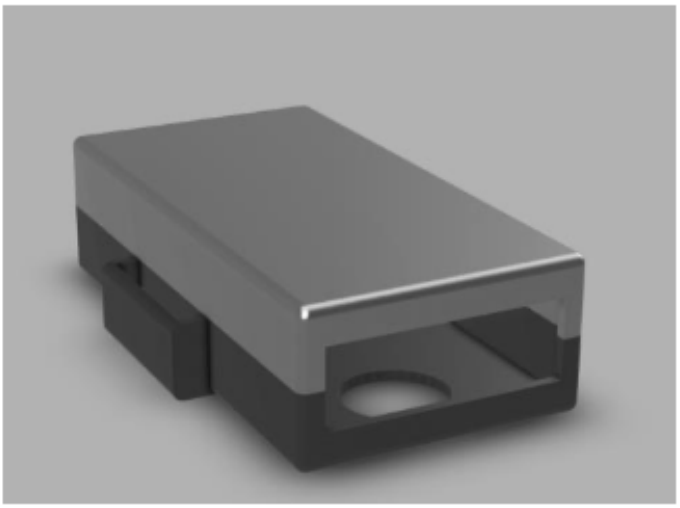
01/07/2022	block 1 validation draft created
01/21/2022	After feedback, I rewrote the block testing Process, and added the black box diagram, as well as I added more on the description

4.7 Enclosure

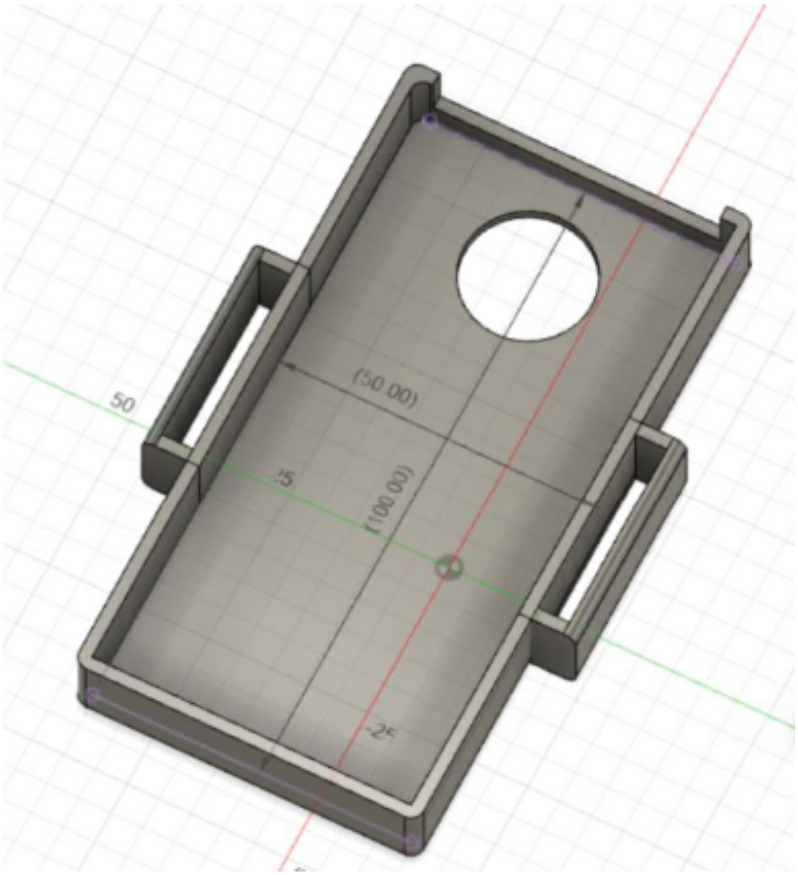
4.7.1 Description

Hamad Alali is assigned this block, which will be the enclosure block which is going to be very essential for the system since it is going to be the protection of the system as well as its going to be designed in a way which will have the perfect size for it to be on a person's wrist. The enclosure will serve as a protection for all the PCBs and power supply as well as the systems sensors which will be enclosed in the enclosure. The system will be worn by the users on the wrist, by wrapping a strap around the user's hand, which will make the enclosure secure in the users' hand.

4.7.2 Design



System enclosure



System Ecnlosure

4.7.3 General Validation

The enclosure of this system is going to be 3D printed by PLA and will have connectors connected to the enclosure wall for the sensor connections. We as group chose to 3D prints the enclosure since the main use of our system will be medical use so it does not need to be fall resistant, and it needs to be withing the range of 2 inches by width as requested by the project partner, so that it can fit in anyone's hand. The enclosure will be adjusted to be wearable by attaching straps to each side of the enclosure. As well as the enclosure will have the ability to be opened and closed to change the batteries whenever needed.

4.7.4 Interface Validation

otsd_enclsr_envin: Input

Interface Property	Why is this interface this value?	Why do you know that your design details for this block above meet or exceed each property?
Other: the enclosure will be not wider than 2 inches(except for the place of the strap)	it is this value because 2 inches will fit the majority of users.	We designed the enclosure to be 2 inches and, we checked using a ruler
Other: The enclosure will withstand vigorous user movement in all directions.	since the enclosure will be wearable it must withstand any sudden hand movement from the user.	The enclosure has been designed to have components mounted and secured inside it.
Other: the enclosure will be not wider than 2 inches(except for the place of the strap)	it is this value because 2 inches will fit the majority of users.	We designed the enclosure to be 2 inches and, we checked using a ruler

4.7.5 Verification Plan

1. For testing the first interface, the size of the enclosure could be proven by the online schematics of the enclosure or by a ruler which will show its size, and the enclosure will be worn by a group member to show that it will fit in a person's hand.

2. to prove the second block interface, a group member will shake the enclosure and show that everything is still in its place without any movement in the inside components of the enclosure.

3. To prove that the enclosure is wearable a group member will wear the enclosure.

4.7.6 References and File Links

"Enclosure Design with Eagle and Fusion 360." Autodesk University, https://www.autodesk.com/autodesk-university/class/Enclosure-Design-Eagle-and- Fusion360-2018.
System used to create enclosure.

Instructions on how to use fusion 360.

4.7.7 Revision Table

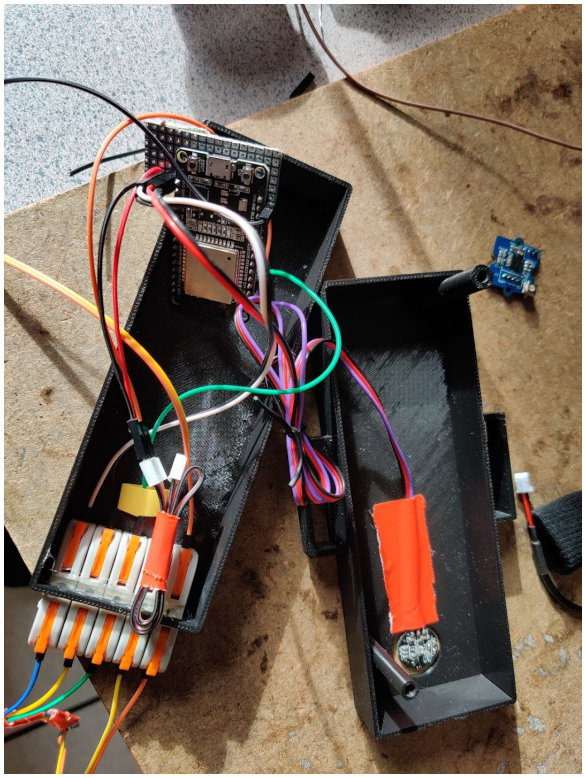
02/04/2022	block 2 validation draft created
02/18/2022	Changed pictures in the block design, included block interfaces, some minor improvements on the block general validation.

5 System Verification Evidence

5.1 Universal Constraints

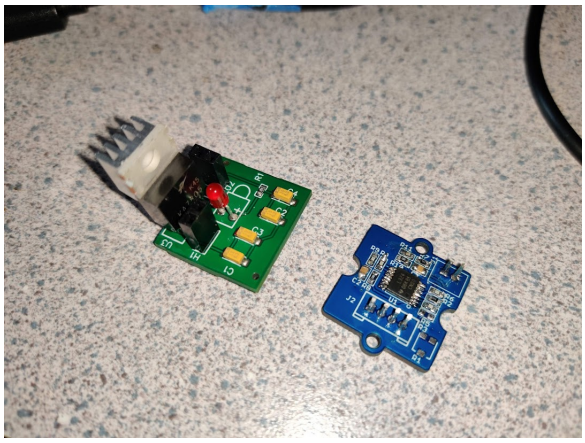
5.1.1 The system may not include a breadboard

The image below shows the final system. We can observe that no breadboard is being used.



5.1.2 The final system must contain both of the following: a student designed PCB and a custom Android/PC/Cloud application

The image below shows the PCBs created for the system. The green one is used to supply power and the blue is a GSR sensor. Here is the link to the website used as a UI for the project.



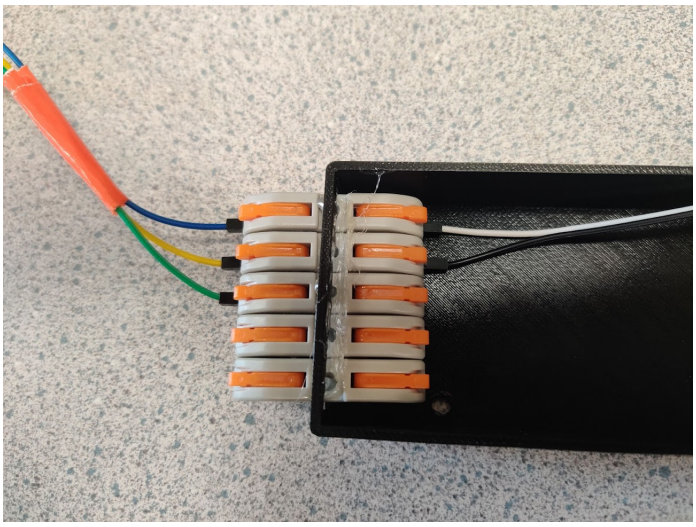
5.1.3 If an enclosure is present, the contents must be ruggedly enclosed/mounted as evaluated by the course instructor

The system enclosure will fit in all the PCB's of the system, power supply, and the microcontroller, it has been designed to be 2 inches wide and wearable since the sensors will be attached to the user's fingers and wrist. Here is a link to a short clip showing how the interior contents are tightly enclosed.

5.1.4 If present, all wire connections to PCBs and going through an enclosure (entering or leaving) must use connectors

This will be proven by showing all wire connections through the system, and showing that all wires entering and leaving the enclosure go through connectors. The image below shows the connectors

used for wires leaving/entering the enclosure. Refer to the image for the previous constraint, and it is observed that the header pins are used as connectors to the rest of the system.



5.1.5 All power supplies in the system must be at least 65 percent efficient

link Efficiency Test

5.1.6 The system may be no more than 50 percent built from purchased modules

This constraint can be met by observing the interior of the enclosure and listing the amount of purchased modules relative to the amount of total key components. Referring to the image in 5.1.1, the only purchased module is a heart rate sensor module seen on the bottom right of the image.

5.2 Requirement: Enclosure Size

5.2.1 Requirement

PPR: The system must be able to sit on a person's wrist.
Requirement: The enclosure for the whole system cannot exceed two inches in width.

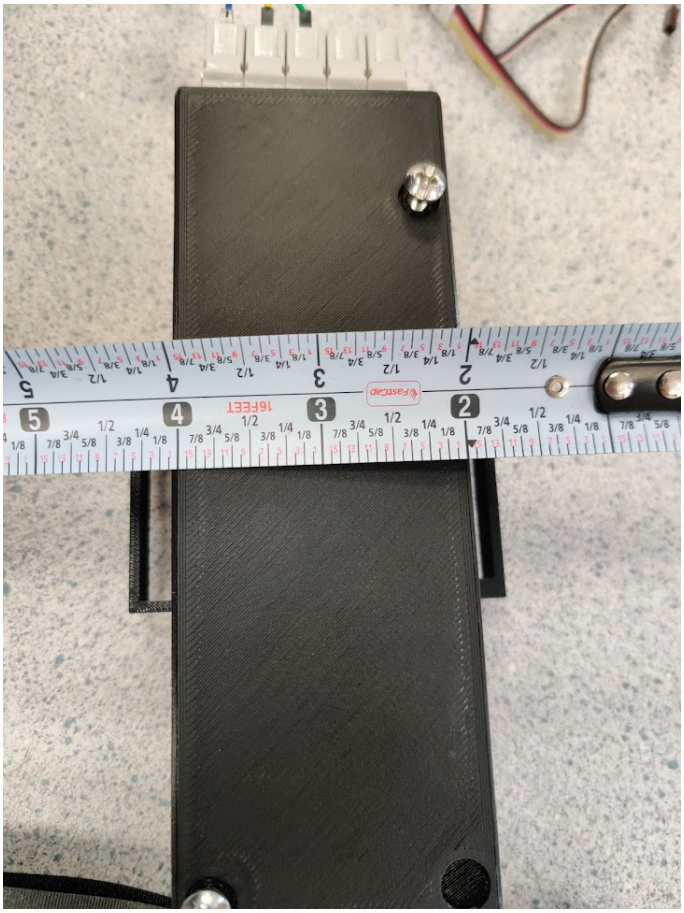
5.2.2 Testing Process

1. Using a measuring tape or ruler, the width of the device will be measured.

If the measured value is 2 inches or less, then the requirement has been met.

5.2.3 Testing Evidence

The image below shows the testing process. We can observe that the measured width is within the requirements.



5.3 Requirement: Data Collection Timing

5.3.1 Requirement

PPR: The system must collect data every minute.
Requirement: The system will collect data from the sensors once every minute (2 second tolerance).

5.3.2 Testing Process

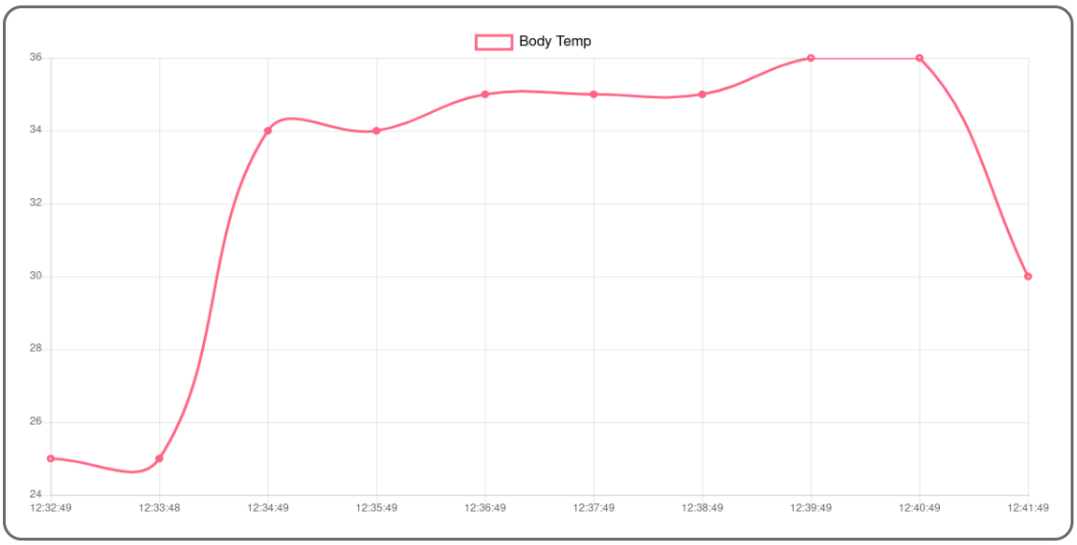
Verification Method:

1. Observe the timestamps on the data sent by the microcontroller to the destination webpage.

If the data sent out by the device has a one minute difference with the previous item for at least 10 continuous minutes, then the requirement is met.

5.3.3 Testing Evidence

Here are some screen captures of data entries to the destination webpage, showing 60-second intervals of logging data.



5.4 Requirement: Battery Life

5.4.1 Requirement

PPR: The system must last at least five hours.
Requirement: The system must have a battery life of at least five hours.

5.4.2 Testing Process

Verification Method:

1. Ensure that the system is fully charged.
2. Begin timing on a stopwatch.
3. Continue to keep track of time until the device stops logging data on the webpage.

If the device logs data for five hours or more, then it can be assumed that the system has been powered for at least five hours and therefore, the requirement is met.

5.4.3 Testing Evidence

Link to a time-lapse showing usage for five hours. The video shows when data was starting to be logged, and when five hours have passed with continuous transmission.

5.5 Requirement: Accuracy

5.5.1 Requirement

PPR: The sensors must accurately collect data.
Requirement: The margin of error from the sensors will vary based on sensor. For temperature, the margin cannot exceed 3 degrees C. For GSR, the margin cannot exceed 10% of the accepted value. For heart rate, the average heart rate collected in a 30-minute trial cannot exceed 5% error or 5 BPM (whichever is larger).

5.5.2 Testing Process

Verification Method:

1. For temperature, the user's temperature will be taken with the sensor, while at the same time by an external thermometer.
2. For heart rate, an external heart rate monitor will be used as the accepted value.
3. For galvanic skin response, the resistance readings from the sensor will be compared with those obtained with a pre-built module.
4. Every minute, the accepted values will be written down.
5. After 10 minutes, begin to compare sensor and accepted values.

If the sensor values are within the range of the tolerance relative to the accepted values, then the requirement is met.

5.5.3 Testing Evidence

We tested all three sensors individually due to compound error of our GSR module, we cannot replicate accurate results. The other two sensors have not met the 10 percent accuracy benchmark, however, are close and more testing is required to tune them to the correct accurate reading.

5.6 Requirement: Stress Indicator

5.6.1 Requirement

PPR: The user must be able to tell the system when they are feeling stressed.
Requirement: The system should have a module that the user can interact with, to signify when they are feeling stressed. It should also display the input.

5.6.2 Testing Process

Verification Method:

1. Over a 10 minute period, the built-in switch is going to be flipped between the 'stressed' side and 'non-stressed' side. We will record what side the switch is on.
2. After the 10 minute period the logged data will be compared to the recorded data.

If the recorded data and the logged data match, that means the microcontroller was able to differentiate the different switch modes. Therefore, the requirement is met.

5.6.3 Testing Evidence

Here is a screenshot of the interface used to identify the state of the switch. It alternates between zero and one.

	id ▲ 1	bTemp	hr	gsr	switch	timeS
Edit Copy Delete	517	30	66	22489	0	14:53:46
Edit Copy Delete	518	34	98	22755	0	14:54:46
Edit Copy Delete	519	33	118	23389	0	14:55:46
Edit Copy Delete	520	32	94	23481	0	14:56:46
Edit Copy Delete	521	32	106	22666	0	14:57:46
Edit Copy Delete	522	32	100	22139	0	14:58:47
Edit Copy Delete	523	31	88	21967	0	14:59:46
Edit Copy Delete	524	31	106	22577	0	15:00:46
Edit Copy Delete	525	30	106	22226	0	15:01:46
Edit Copy Delete	526	30	108	23025	1	15:02:46
Edit Copy Delete	527	29	78	22053	1	15:03:46
Edit Copy Delete	528	26	102	23025	1	15:04:46
Edit Copy Delete	529	30	118	24233	1	15:05:46
Edit Copy Delete	530	30	100	25209	1	15:06:46

5.7 Requirement: Shut Down Feature

5.7.1 Requirement

PPR: If the system does not collect any data it must turn off.
Requirement: The system will shut down if no data is collected for five consecutive minutes.

5.7.2 Testing Process

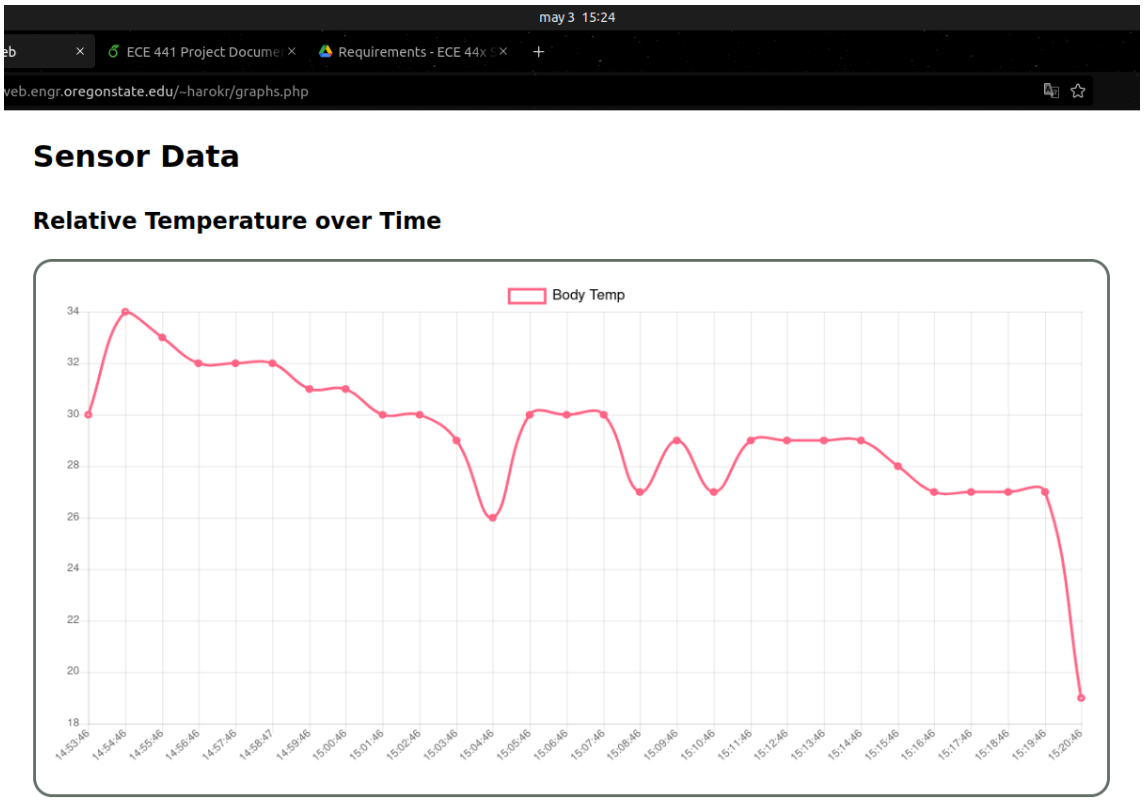
Verification Method:

1. The system will consider data not being collected based on the values of temperature.
2. As there is a large difference between room and body temperature, every minute where the logged temperature is less than 28 degrees Celsius will increment a counter.
3. For every consecutive minute at room temperature, the count will increase. Once the count reaches five the device will enter sleep mode and no longer transmit data.
4. Verify that no data is being transmitted by waiting 3 minutes and observe the last timestamp on this website.

If the system logs data at the required intervals after conducting the procedure, then the requirement is met

5.7.3 Testing Evidence

Here is a screenshot of the temperature graph with the current timestamp at 3:25pm. We can see that data stopped being transmitted due to continuous readings of room temperature values.



5.8 Requirement: Information Access

5.8.1 Requirement

PPR: User most be able to access information either wirelessly or through Bluetooth, within a distance of 10 feet.
Requirement: User most be able to access information wirelessly, within a distance of 10 feet.

5.8.2 Testing Process

Verification Method:

1. Using a measuring tape, ensure that the system and the receiver are 10 feet apart.
2. Visit the webpage where sensor data will be displayed to.
3. Inspect the timestamps on the data sent.

If the webpage is logging data at the correct intervals (1 minute) for 10 consecutive minutes, then the requirement is met.

5.8.3 Testing Evidence

Link to part 1. This video shows that the device and receiver are ten feet apart.
Link to part 2. This cuts to a clip after ten minutes have passed showing that data has been transmitted as required.

5.9 Requirement: Rechargeable

5.9.1 Requirement

PPR: The system's battery will be rechargeable.
Requirement: The system's battery will be fully charged under 90 minutes.

5.9.2 Testing Process

Verification Method:

1. Assume that the system's battery is out of charge if it does not power on after 10 minutes of attempting to.
2. Connect the USB interface to power.
3. Using an onboard LED to display the battery status, observe the LED behavior (blinking for charging, solid for full).
4. Once the LED indicates a full charge, use the device until the battery is out of charge.

If at the end of the charge, the device battery life meets requirement 2.1.4, then this requirement is met.

5.9.3 Testing Evidence

link Test demo of charging a variety of rechargeable batteries(3 minutes into the video). Due to miss communication we concluded that video evidence of a battery test would suffice, and did not consider testing it in scope of our design(which introduces a different environment.

5.10 References and File Links

5.11 Revision Table

03/04/2022	Hamad: Added universal constraints descriptions
03/04/2022	Jimmy: Worked on section 5
03/04/2022	Kris: Added Universal constraints.
03/04/2022	Arthur: Added section 4 interfaces.
04/22/2022	Kris: Added more requirements to section 5.
04/22/2022	Jimmy: Updated requirements of section 5.

6 Project Closing

6.1 Future Recommendations

6.1.1 Technical recommendations

1. The first technical recommendation is to create in-house sensor modules from scratch. This is more aimed at collecting more reliable values for heart rate and GSR, as the requirement for accuracy is still a work-in-progress.

2. The next technical recommendation is to reduce the size of the device. This ties into the first recommendation, as developing a device with less pre-made items allows for optimization of space. For a device that is meant to be on a person's wrist for hours at a time, comfort should be considered.

3. Another recommendation is to make sure that all the components fit inside the enclosure. Not only do you need to account for PCB size but also for the battery any jumper wires that may be used. If this isn't accounted for, the enclosure will not close or everything will be crammed inside.

4. The last technical recommendation would be to use a temperature sensor that could easily fit on person's finger without having to read room temperature to get the exact reading of a person's body temperature.

6.1.2 Global impact recommendations

1. Alex Pisarev, CEO of battery supplier OneCharge claims that around 80% of lithium-ion cells are still usable at the end of the battery's life cycle, and they can be repurposed to less-demanding jobs [1]. Therefore, to minimize the impact on the environment, a recommendation would be to attempt to source a battery that can be repurposed.

2. A global impact recommendation would be to use PLA for 3D printing, for the enclosure of the project, instead of using aluminum which creates many dangerous gases when being recycled [2]. Since PLA is made up of starch and sugar which made to a poly lactic acid, which means that using PLA is safe for the environment.

6.1.3 Teamwork recommendations

1. One recommendation for teamwork is to have regular in-person meetings. The team communicated primarily on Discord which was fine in the early stages of the project, however as tasks and updates started to become more important in-person updates as well as having dedicated times to work the project would have made better progress.

2. A second recommendation is to make sure you are updating the timeline. The terms go by fast and many unpredicted things can happen. Updating the timeline can help make sure your team does not fall behind.

3. A third recommendation is if you are having difficulty with a task or teammate, go and talk to the instructors. They are here to help, and they can help you overcome your problem or at the very least point you in the right direction.

4. Lastly, is to always be prepared for any timeline delay, or shipment delays that could happen by starting on the task early in order to have extra time for any sudden delays.

6.2 Project Artifact Summary with Links

The artifacts included in this section consist of the code and PCB layouts used in the creation of this project. The first piece of code is the firmware. Then there is the PHP code used for the project website. This the design of our enclosure. Figure 7 is a bar graph of hours spent on the project by each member.

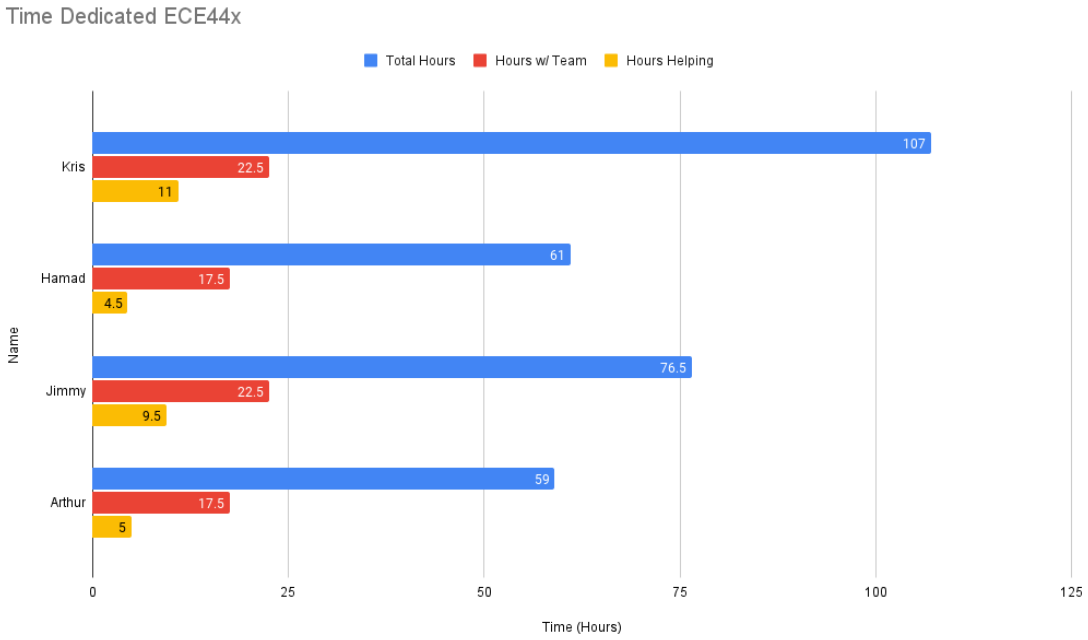


Figure 7: Time spent on the project, by member.

6.3 Presentation Materials

Engineering Expo Poster

References

[1] R. Rapier, "Environmental implications of lead-acid and lithium-ion batteries," *Forbes*, 20-Jan-2020. https://www.forbes.com/sites/rrapier/2020/01/19/environmental-implications-of-lead-acid-and-lithium-ion-batteries.

[2] N. Kokemuller, "What type of household products would have copper sulfate in it?," Education, 29-Sep-2016. [Online]. Available: https://education.seattlepi.com/type-household-products-would-copper-sulfate-it-4474.html. [Accessed: 06-May-2022].

A Code Used

A.1 Firmware

A.1.1 Main.cpp

```
#include <WiFi.h>
#include <WiFiClientSecure.h>
#include <TickTwo.h>
#include <string.h>
#include "refs.h" // contains ca certificate for https post

// ------- MCU CONNECTIONS ----------
// HR sensor          GPIO36/VP
// Gsr sensor        GPIO39/VN
// Temp sensor         GPIO34/D34
// User switch        GPIO21/D2

// function declarations
void readNLog();
void valueRead();

// network & API credentials
```

```
const char* ssid     = "OnePlus 7T";
const char* password = "d22f3117dc9e";
const char*  server = "web.engr.oregonstate.edu";  // Server URL
const int port = 443;
String apiKeyValue = "tPmAT5Ab3j7F9"; // API key for communicating with serverName


// These timers execute the respective function at given intervals,
// readNLog is for https transmission, valueRead is for sensor readings
TickTwo timer1(readNLog, 1000*60); // once, every 40s
TickTwo timer2(valueRead, 250); // once every 250ms
WiFiClientSecure client;


void setup() {
  pinMode(SWITCH_PIN, INPUT_PULLUP); // initialize the pushbutton pin as an pull-up
     input
  timer1.start(); // initialize timers
  timer2.start();
  Serial.begin(115200); // set baud rate

  //Print the wakeup reason for ESP32
  print_wakeup_reason();

   // Setup wifi connection
   WiFi.mode(WIFI_STA); //The WiFi is in station mode.
   WiFi.begin(ssid, password); // connect to wifi network
   while (WiFi.status() != WL_CONNECTED) {
      delay(500);
      Serial.print(".");
   }
   // successful connection
   Serial.println("");  Serial.print("WiFi connected to: "); Serial.println(ssid);
   Serial.println("IP address: ");  Serial.println(WiFi.localIP());
   client.setCACert(ca_cert); //Only communicate with the server if the CA
    certificates match

   delay(1000);
}

void loop() {
 //timer1.update();
 timer2.update();
 // update timers, nothing else should happen here since
 // all operations are based on the ticker functions
}

// this funciton will collect sensor data
// It will be ran every 500ms to detect a heartbeat
// Every second, gsr and temperature are also collected
// after 30 seconds, or 60 iterations, averages and bpm will be
// calculated
void valueRead(){
   // collect analog values
   if(count < 120){
      if(count % 4 == 0){ // enter here every second
         int tempVal = analogRead(tPin);
         float volts = tempVal/1023.0;             // normalize by the maximum
    temperature raw reading range
         float temp = (volts - 0.5) * 100 ;         //calculate temperature celsius
     from voltage as per the equation found on the
         int gsr = 0;
         tempArr[count / 4] = (int)temp;
         gsrArr[count / 4] = gsr;
         currentState = digitalRead(SWITCH_PIN);
         lastState = currentState;
      }
      int hr = analogRead(hrPin); // read hr every 250ms
      if(hr > 1995){
         hrArr[count] = 1; // log if beat is detected
      }
      else{hrArr[count] = 0;}
   }
```

```
87
88    else if(count >= 120){
89       //Serial.println("data is ready for transmission");
90       int sumT = 0, sumG = 0;
91       for(int i = 0; i < 30; i++){
92          sumT += tempArr[i];
93          sumG += gsrArr[i];
94       }
95       tAvg = sumT / 30;
96       gAvg = sumG / 30;
97       int beatCount = 0;
98       for(int i = 0; i < 120; i++){
99          if(hrArr[i] == 1){
100            beatCount++; // count beats detected
101         }
102      }
103      hAvg = (beatCount * 2) - 10; // number of beats in 30s * 2 = bpm
104      if(hAvg < 35){ // filter out heart rates that are too low as they are
105      // most likely wrong
106         hAvg = 0;
107      }
108   }
109   count++;
110}
111
112void readNLog(){
113  int conn;
114  conn = client.connect(server, port);
115  String body = "api_key=" + apiKeyValue + "&btemp=" + String(tAvg)
116                           + "&hr=" + String(hAvg)
117                           + "&gsr=" + String(gAvg) + "&switch=" + String(
     currentState);
118  int body_len = body.length();
119  if (conn == 1) {
120      Serial.println(); Serial.print("Sending Parameters...");
121      //Request
122      client.println("POST /~harokr/post.php HTTP/1.1");
123      //Headers
124      client.print("Host: "); client.println(server);
125      client.println("Content-Type: application/x-www-form-urlencoded");
126      client.print("Content-Length: "); client.println(body_len);
127      client.println("Connection: Close");
128      client.println();
129      count =0; // reset count
130      for(int i = 0; i < 120; i++){
131         hrArr[i] = 0; // reset beat array
132      }
133      //Body
134      client.println(body);
135      client.println();
136
137      //Wait for server response
138      while (client.available() == 0);
139
140      //Print Server Response
141      while (client.available()) {
142         char c = client.read();
143         Serial.write(c);
144      }
145   } else {
146      client.stop();
147      Serial.println("Connection Failed");
148   }
149
150}
```

A.1.2 Refs.h

```
1 /**
2  * @file refs.h
3  * @author Kris Haro
4  * @brief Header file for other variables to not clutter main.cpp
5  * @version 0.2
```

```
 * @date 2022-04-30
 *
 * @copyright Copyright (c) 2022
 *
 */
#define uS_TO_S_FACTOR 1000000  /* Conversion factor for micro seconds to seconds */
#define TIME_TO_SLEEP 60 *5       /* Time ESP32 will go to sleep (in seconds) */
RTC_DATA_ATTR int bootCount = 0;

/*
Method to print the reason by which ESP32
has been awaken from sleep
*/
void print_wakeup_reason(){
  esp_sleep_wakeup_cause_t wakeup_reason;

  wakeup_reason = esp_sleep_get_wakeup_cause();

  switch(wakeup_reason)
  {
    case ESP_SLEEP_WAKEUP_EXT0 : Serial.println("Wakeup caused by external signal using RTC_IO"); break;
    case ESP_SLEEP_WAKEUP_EXT1 : Serial.println("Wakeup caused by external signal using RTC_CNTL"); break;
    case ESP_SLEEP_WAKEUP_TIMER : Serial.println("Wakeup caused by timer"); break;
    case ESP_SLEEP_WAKEUP_TOUCHPAD : Serial.println("Wakeup caused by touchpad"); break;
    case ESP_SLEEP_WAKEUP_ULP : Serial.println("Wakeup caused by ULP program"); break;
    default : Serial.printf("Wakeup was not caused by deep sleep: %d\n", wakeup_reason); break;
  }
}

#define SWITCH_PIN 21
int lastState = HIGH; // the previous state from the input pin
int currentState;     // the current reading from the input pin

// ca certificate for https
const char* ca_cert= \
"-----BEGIN CERTIFICATE-----\n" \
"MIIF3jCCA8agAwIBAgIQAf1tMPyjylGoG7xkDjUDLTANBgkqhkiG9w0BAQwFADCB\n" \
"iDELMAkGA1UEBhMCVVMxEzARBgNVBAgTCk5ldyBKZXJzZXkxFDASBgNVBAcTC0pl\n" \
"cnNleSBDaXR5MR4wHAYDVQQKExVUaGUgVVNFUlRSVVNUIE5ldHdvcmsxLjAsBgNV\n" \
"BAMTJVVTRVJUcnVzdCBSU0EgQ2VydGlmaWNhdGlvbiBBdXRob3JpdHkwHhcNMTAw\n" \
"MjAxMDAwMDAwWhcNMzgwMTE4MjM1OTU5WjCBiDELMAkGA1UEBhMCVVMxEzARBgNV\n" \
"BAgTCk5ldyBKZXJzZXkxFDASBgNVBAcTC0plcnNleSBDaXR5MR4wHAYDVQQKExVU\n" \
"aGUgVVNFUlRSVVNUIE5ldHdvcmsxLjAsBgNVBAMTJVVTRVJUcnVzdCBSU0EgQ2Vy\n" \
"dGlmaWNhdGlvbiBBdXRob3JpdHkwggIiMA0GCSqGSIb3DQEBAQUAA4ICDwAwggIK\n" \
"AoICAQCAEmUXNg7D2wiz0KxXDXbtzSfTTK1Qg2HiqiBNCS1kCdzOiZ/MPans9s/B\n" \
"3PHTsdZ7NygRK0faOca8Ohm0X6a9fZ2jY0K2dvKpOyuR+OJv0OwWIJAJPuLodMkY\n" \
"tJHUYmTbf6MG8YgYapAiPLz+E/CHFHv25B+O1ORRxhFnRghRy4YUVD+8M/5+bJz/\n" \
"Fp0YvVGONaanZshyZ9shZrHUm3gDwFA66Mzw3LyeTP6vBZY1H1dat//O+T23LLb2\n" \
"VN3I5xI6Ta5MirdcmrS3ID3KfyI0rn47aGYBROcBTkZTmzNg95S+UzeQc0PzMsNT\n" \
"79uq/nROacdrjGCT3sTHDN/hMq7MkztReJVni+49Vv4M0GkPGw/zJSZrM233bkf6\n" \
"c0Plfg6lZrEpfDKEY1WJxA3Bk1QwGROs0303p+tdOmw1XNtB1xLaqUkL39iAigmT\n" \
"Yo61Zs8liM2EuLE/pDkP2QKe6xJMlXzzawWpXhaDzLhn4ugTncxbgtNMs+1b/97l\n" \
"c6wjOy0AvzVVdAlJ2ElYGn+SNuZRkg7zJn0cTRe8yexDJtC/QV9AqURE9JnnV4ee\n" \
"UB9XVKg+/XRjL7FQZQnmWEIuQxpMtPAlR1n6BB6T1CZGSlCBst6+eLf8ZxhyVeE\n" \
"Hg9j1uliutZfVS7qXMYoCAQlObgOK6nyTJccBz8NUvXt7y+CDwIDAQABo0IwQDAd\n" \
"BgNVHQ4EFgQUU3m/WqorSs9UgOHYm8Cd8rIDZsswDgYDVR0PAQH/BAQDAgEGMA8G\n" \
"A1UdEwEB/wQFMAMBAf8wDQYJKoZIhvcNAQEMBQADggIBAFzUfA3P9wF9QZllDHPF\n" \
"Up/L+M+ZBn8b2kMVn54CVVeWFPFSPCeHlCjtHzoBN6J2/FNQwISbxmtOuowhT6KO\n" \
"VWKR82kV2LyI48SqC/3vqOlLVSoGIG1VeCkZ7l8wXEskEVX/JJpuXior7gtNn3/3\n" \
"ATiUFJVDBwn7YKnuHKsSjKCaXqeYalltiz8I+8jRRa8YFWSQEg9zKC7F4iRO/Fjs\n" \
"8PRF/iKz6y+O0tlFYQXBl2+odnKPi4w2r78NBc5xjeambx9spnFixdjQg3IM8WcR\n" \
"iQycE0xyNN+81XHfqnHd4blsjDwSXWXavVcStkNr/+XeTWYRUc+ZruwXtuhxkYze\n" \
"Sf7dNXGiFSeUHM9h4ya7b6NnJSFd5t0dCy5oGzuCr+yDZ4XUmFF0sbmZgIn/f3gZ\n" \
"XHlKYC6SQK5MNyosycdiyA5d9zZbyuAlJQG03RoHnHcAP9Dc1ew91Pq7P8yF1m9/\n" \
"qS3fuQL39ZeatTXaw2ewh0qpKJ4jjv9cJ2vhsE/zB+4ALtRZh8tSQZXq9EfX7mRB\n" \
"VXyNWQKV3WKdwrnuWih0hKWbt5DHDAff9Yk2dDLWKMGwsAvgnEzDHNb842m1R0aB\n" \
"L6KCq9NjRHDEjf8tM7qtj3u1cIiuPhnPQCjY/MiQu12ZIvVS5ljFH4gxQ+6IHdfG\n" \
```

```
"jjxDah2nGN59PRbxYvnKkKj9\n" \
"-----END CERTIFICATE-----\n";

// variables for data collection
const int hrPin = 36; // vp pin
const int gPin = 39; // vn pin
const int tPin = 34; // d34
int count = 0; // used for array counting

// arrays for storing sensor values
int tempArr[30];
int gsrArr[30];
int hrArr[120];

// averages that will be transmistted
int tAvg = 1;
int gAvg = 2;
int hAvg = 3;
```

A.2 Website Code

A.2.1 graphs.php

```
<!DOCTYPE html>
<html>
<head>
  <meta charset="utf-8">
  <meta name="author" content="Kris Haro">
  <meta name="description" content="Device Sensor Data">
  <title>Tech Demo Web</title>
  <link rel="stylesheet" href="css/styling.css"
</head>
<body>
  <script type="text/javascript" src="https://cdnjs.cloudflare.com/ajax/libs/Chart.
    js/2.7.2/Chart.bundle.min.js"></script>
  <script src="path/to/chartjs/dist/chart.min.js"></script>
  <script src="path/to/chartjs-plugin-annotation/dist/chartjs-plugin-annotation.min
    .js"></script>
  <h1>Sensor Data</h1>
<?php
/*
  Original author Rui Santos
  Complete project details at https://RandomNerdTutorials.com/esp32-esp8266-mysql-
    database-php/

  Permission is hereby granted, free of charge, to any person obtaining a copy
  of this software and associated documentation files.

  The above copyright notice and this permission notice shall be included in all
  copies or substantial portions of the Software.
*/

// Modifications done by Kris Haro for ECE44x project
// Will plot the sql data in different graphs

$dbhost = 'classmysql.engr.oregonstate.edu';
$dbname = 'capstone_2021_harokr';
$dbuser = 'capstone_2021_harokr';
$dbpass = 'VxbYGMg1qJi!h.2Z';

// Create connection
$conn = new mysqli($dbhost, $dbuser, $dbpass, $dbname);
// Check connection
if ($conn->connect_error) {
    die("Connection failed: " . $conn->connect_error);
}
// get stuff from table
$sql = "SELECT id, bTemp, hr, gsr, switch, timeS FROM StressDeviceEntry ORDER BY id
     ASC";

if ($result = $conn->query($sql)) { // execute query
    while ($row = $result->fetch_assoc()) {
        $row_id = $row["id"];
```

```
        $row_bTemp = $row["bTemp"];
        $row_heartRate = $row["hr"];
        $row_skinResponse = $row["gsr"];
        $row_switch = $row["switch"];
        $row_timeS = $row["timeS"];

        $data1 = $data1 . '"'. $row['bTemp']. '",'; // store query values into arrays
        $dataH = $dataH . ''. $row['hr']. ',';
        $dataG = $dataG . '"'. $row['gsr']. '",';
        $dataR = $dataR . '"'. $row['switch']. '",';
        $dataT = $dataT . '"'. $row['timeS']. '",';
    }
    $data1 = trim($data1,",");
    $dataH = trim($dataH,",");
    $dataG = trim($dataG,",");
    $dataR = trim($dataR,",");
    $dataT = trim($dataT,",");
    $result->free(); // free memory associated with result
}

$conn->close(); // close sql connection
?>
</table>
<!-- create graphs -->
<h2>Relative Temperature over Time </h2>
<div class="container">
  <canvas id="tempChart">
  </canvas>
  <script>
  // javascript code for one graph, using chart.js library
  var ctx = document.getElementById("tempChart").getContext('2d');
    var myChart = new Chart(ctx, {
      type: 'line',
      data: {
          labels:[<?php echo $dataT; ?>],
          datasets:
          [{
              label: 'Body Temp',
              data: [<?php echo $data1; ?>],
              backgroundColor: 'transparent',
              borderColor:'rgba(255,99,132)',
              borderWidth: 3
          }]
      },

      options: {
        responsive: true,
        maintainAspectRatio: true,
        scales: {scales:{yAxes: [{beginAtZero: false}], xAxes: [{autoskip: true, maxTicketsLimit: 20}]}},
        tooltips:{mode: 'index'},
        legend:{display: true, position: 'top', labels: {fontColor: 'rgb(0,0,0)', fontSize: 16}}
      } // options

  });
  </script>
</div>
<!-- another graph -->
<h2>Heart Rate </h2>
<div class="container">
  <canvas id="hrChart">
    <script>
    var ctx = document.getElementById("hrChart").getContext('2d');
    const average = 3;
    const hr = [<?php echo $dataH; ?>];
    const time = [<?php echo $dataT; ?>];
    const movingAverage = [];
    for(i = 0; i < hr.length -2; i++){
      const threedatapoints = hr.slice(i, average + i);
      threedatapoints.reduce((total, num) => total + num) / 3;
      movingAverage.push(threedatapoints.reduce((total, num) => total + num) /
```

```
    threedatapoints.length);
      console.log(movingAverage)
    }
    var myChart = new Chart(ctx, {
      type: 'line',
      data: {
          labels: time,
          datasets:
          [{
            label: 'Heart Rate',
            data: hr,
            backgroundColor: 'transparent',
            borderColor:'rgba(100,200,255)',
            borderWidth: 3
          },{
            label: 'Moving Average',
            backgroundColor: 'transparent',
            borderColor: 'rgb(0,99,0)',
            data:movingAverage
          }]
      },

      options: {
        responsive: true,
        maintainAspectRatio: true,
        scales: {scales:{yAxes: [{beginAtZero: false}], xAxes: [{autoskip: true,
    maxTicketsLimit: 20}]}},
        tooltips:{mode: 'index'},
        legend:{display: true, position: 'top', labels: {fontColor: 'rgb(0,0,0)',
    fontSize: 16}}
      } // options
    }); // myChart
    </script>

  </canvas>
</div>

<h2>Galvanic Skin Respone</h2>
<div class="container">
  <canvas id="gsrChart">
    <script>
    var ctx = document.getElementById("gsrChart").getContext('2d');
      var myChart = new Chart(ctx, {
        type: 'line',
        data: {
            labels:[<?php echo $dataT; ?>],
            datasets:
            [{
              label: 'Skin Response',
              data: [<?php echo $dataG; ?>],
              backgroundColor: 'transparent',
              borderColor:'rgba(100,100,230)',
              borderWidth: 3
            }]
        },

        options: {
          responsive: true,
          maintainAspectRatio: true,
          scales: {scales:{yAxes: [{beginAtZero: false}], xAxes: [{autoskip: true,
    maxTicketsLimit: 20}]}},
          tooltips:{mode: 'index'},
          legend:{display: true, position: 'top', labels: {fontColor: 'rgb(0,0,0)',
     fontSize: 16}}
        } // options
    });
    </script>
  </canvas>
</div>
<button id="new entry"> Create new entry </button>
<button id="export"> Export data </button>
</body>
<script src="js/dataPage.js"></script>
```

```
<script type="text/javascript">
    // assume each element of $arrayWithVars has already been json_encoded
    toCSV([<?php echo $data1 ?>], [<?php echo $data2 ?>], [<?php echo $dataH ?>],
    [<?php echo $dataG ?>], [<?php echo $dataR ?>], [<?php echo $dataT ?>]);
</script>

</html>
```