# From Bodily Sensors to Cloud and Back

# Design Document

Group 26

Client - Goce Trajcevski

Advisors - Goce Trajcevski, Liang Dong, Sung Min Kang

Isaac Zahau - Front-end/UI - isanga@iastate.edu

Justin Worley - Cloud Engineer - jmworley@iastate.edu

John Kivley - Electrical Engineer - jekivley@iastate.edu

Richa Patel - Database Engineer - rppatel@iastate.edu

Michael Lauderback - mlauderb@iastate.edu

http://sddec20-26.sd.ece.iastate.edu/

# Executive Summary

## Development Standards & Practices Used

- IEEE standards
- Waterfall Software Development
- Circuit, Block, and Use Case diagrams

## Summary of Requirements

- At least 2 sensor
- Data stored on the cloud
- Transmit and receive data securely
- User friendly user interface

## Applicable Courses from Iowa State University Curriculum

- EE 201 - Electric Circuits
- EE 230 - Electronic Circuits and Systems
- EE 285 - Problem Solving Methods and Tools for Electrical Engineering
- EE 321 - Communication Systems I
- CPRE 281 - Digital Logic
- CPRE 288 - Embedded Systems I: Introduction
- CPRE 388 - Embedded Systems II: Mobile Platforms
- COM S 227 - Object-oriented Programming
- COM S 228 - Introduction to Data Structures
- COM S 319 - Construction of User Interfaces
- COM S 309 - Software Development Practices
- COM S 363- Introduction to Database Management Systems
- COM S 474- Introduction to Machine Learning

## New Skills/Knowledge acquired that was not taught in courses

Gained knowledge of industry tools used for hardware design such as KiCad.

Gained knowledge and skill in designing electrical hardware starting from an idea to the final product.

Gained knowledge and honed skills in troubleshooting and debugging hardware designs and learning multiple methods to conduct troubleshooting.

Gained knowledge in project documentation practices.

Gained knowledge in Amazon DynamoDB

Gained knowledge on AWS infrastructure.

Gained knowledge on features of Spark and Cassandra

Gained knowledge in MVC

## Table of Contents

## List of figures/tables/symbols/definitions (This should be the similar to the project plan)

Will need to add the list of figures after we have all the diagrams up.

MCU - Master Control Unit

PCB - Printed Circuit Board

ETG - Electronics and Technology Group at Iowa State University of Science and Technology

# 1 Introduction

## 1.1 Acknowledgement

We would like to acknowledge Goce Trajcevski, Liang Dong, and Sung Min Kang. Goce Trahcevski has been there to support and encourage us. His advice and wisdom have greatly aided this project. Sung Min Kang and Liang Dong have been there to offer great advice in regards to sensors for our project.

## 1.2 Problem and Project Statement

General Problem:

For an athlete/someone who is looking to do an extreme training/workout, the details of that activity are very important. We are most interested in details like the user's heart rate, weight distribution, and the maximum body temperature. From these details, we are trying to give the user a good summary of their workout and how their body adapts over time.

General Solution:

To solve these and other possible needs, we have turned to the remote data storage, aka a cloud platform, and sensors placed in different locations on the human body. We will incorporate different sensors placed in key locations that all talk to a master control unit (MCU). This MCU will then process the data from all of the sensors. If the mobile application is open, it will transmit live statistics to the mobile application. Along with the mobile application the MCU will also send the data that was collected to the cloud platform. Once there, that data will be stored and analyzed to provide the user with multiple ways to view their statistics.

## 1.3 Operational Environment

The end product will be exposed to many different environments: essentially any environment that the human body is exposed to. It will have to endure sweat, heat, moisture, dust, repeated motion, mild detergents, and soft to abrasive material.

## 1.4 Requirements

-Use at least two sensors to create an IoT system from monitoring bodily functions.

-Sensors must have low power consumption and be wearable for reliability and for the device to be mobile-friendly.

-The sensors must be able to transmit data to an MCU (digital signals) where the MCU relays the data to the user's phone. The MCU relays digital signals directly, and it converts received analog signals to digital signals using an ADC before relaying the data.

-All electric hardware must be protected from outside elements when being worn on the body. The hardware cannot be disturbed by the rain or snow, and also sweat from the body.

-The mobile application must be able to pull data from the database

-The UI must be easy to navigate while providing details from every sensor. The visual aids should be easy to view and manipulate.

-All transmissions to and from the cloud need to be encrypted.

## 1.5 Intended Users and Uses

-A properly designed end product that is wearable and easy to use will provide information regarding heart rate, temperature, and body weight all in realtime that is easy for the user to read and analyze.

## 1.6 Assumptions and Limitations

- Assumptions
  - The max number or user will greatly vary over any given time.
  - The product will, currently, be limited to the United States of America.
  - The users will agree to a information disclosure allowing bodily readings to be stored in a remote location.
  - There will be a monthly subscription charge to access and store bodily data.
  - The optimal body sensors would monitor temperature, heart rate, and weight.
  - The end product will be multiple wearable items each embedded with sensors for monitoring bodily functions.
  - Best approach is to design and build our own microcontroller board to receive and relay the data from the sensors.
  - The best method would be for the microcontroller to relay the sensor data to the users phone which then the phone relays the data to the cloud.
- Limitations
  - The cost of the final physical product should not exceed $100.
  - Due to funding and limited time less than 10 products will be available for testing.
  - Majority of testing will be with simulated users and devices.
  - Due to the limited amount of users, scalability will not be able to be tested as effectively.
  - The system must have a power supply consisting of batteries that will power the device for at least the majority of the day.
  - The system must operate on 5.5VDC or less.
  - Due to the nature of the data being collected, all data transferred must be heavily secured.

## 1.7 Expected End Product and Deliverables

This product is a wearable device that includes multiple body sensors. The sensors will measure temperature, heart rate, and body weight of the user. This product will be portable enough for the users to take it outside, away from home to be able to use it during workouts. The product is also able to connect to an Android phone to display live data when WiFi is not available. The device will use the bluetooth functionality to connect to the Android phone and display the live data in a clean user interface. If WiFi is available, the Android app will be able to pull in processed data from the cloud and display the data onto the app.

# 2. Specifications and Analysis

## 2.1 Proposed Approach

We did research on the databases which would be used for the project.

Setup development environment for the Android app.

We have researched sensors that are potentially viable for mobile friendly devices.

We have ordered a sensor to be tested and are researching bluetooth boards to connect to the sensors for communicating with the MCU.

We have discussed packaging designs for the final product

We have discussed the top-level architecture for the project.

## 2.2 Design Analysis

We brainstormed ideas and researched databases, ordered sensors to be tested, and discussed packaging designs. We decided to focus on getting one sensor to work at a time so we ordered a pulse sensor to get that functioning first. We also have researched designing an MCU and will have designs written up soon.

We have narrowed down the  database to AWS DynamoDB. We have decided to use the AWS cloud platform and its services.  We narrowed down the concept of our end product and will begin hardware testing soon. We will also have MCU circuit designs drawn up soon.

We have decided to display processed data from the database so we will set up a test app to make sure the backend and frontend can communicate.

If this idea doesn't work, then we will research other ideas, sensors and make modifications for the project.

## 2.3 Development Process

We are using the Waterfall development process. It was highly recommended by our advisor/client.

## 2.4 Conceptual Sketch

Use a system-level conceptual sketch diagram to describe the concept for your approach/design. Describe the modules in your design (dependency/concurrency of modules, interfaces, architectural overview, etc.), and module constraints tied to the requirements.

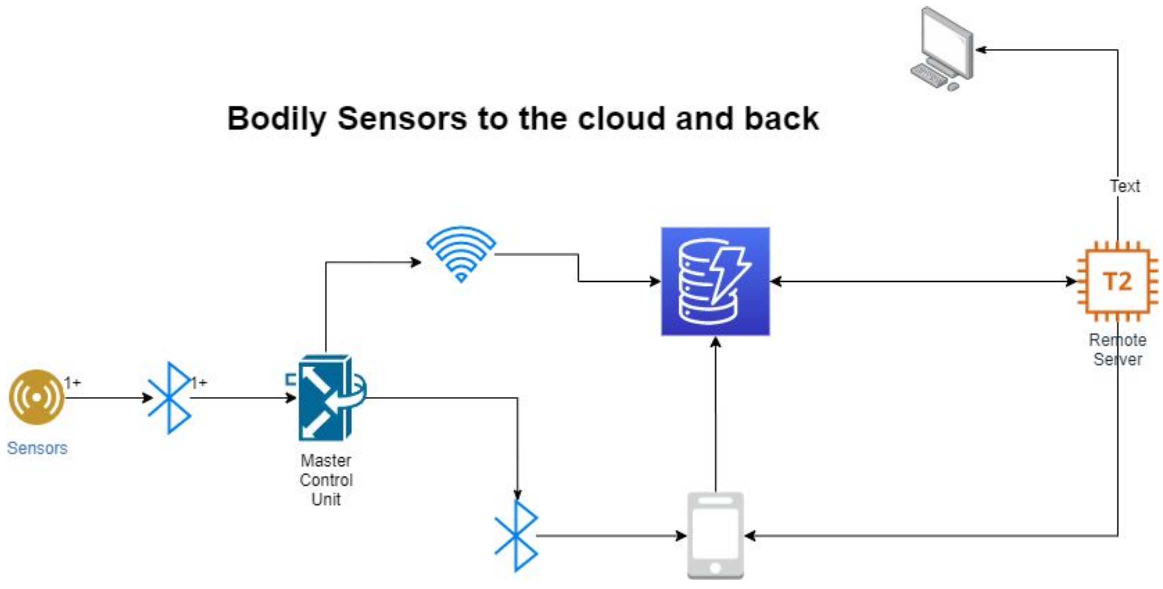


Figure 2.1

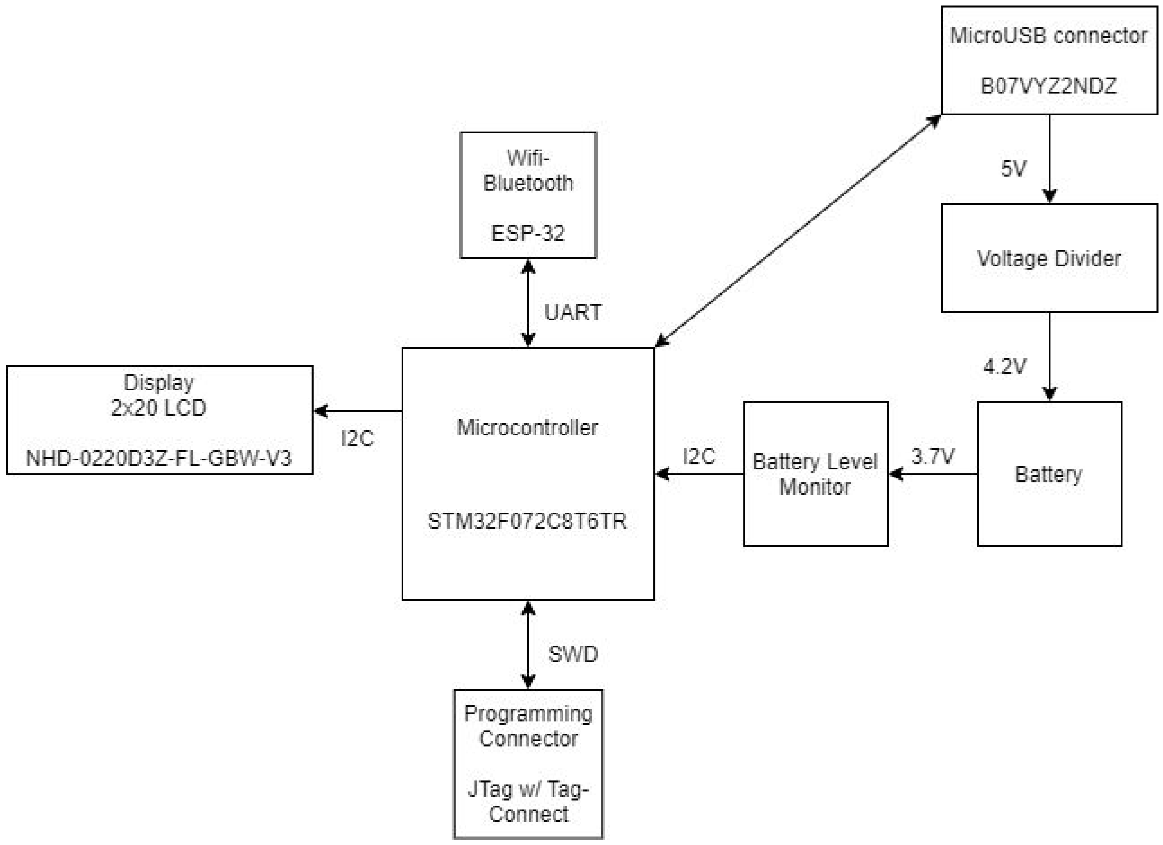


Figure 2.2

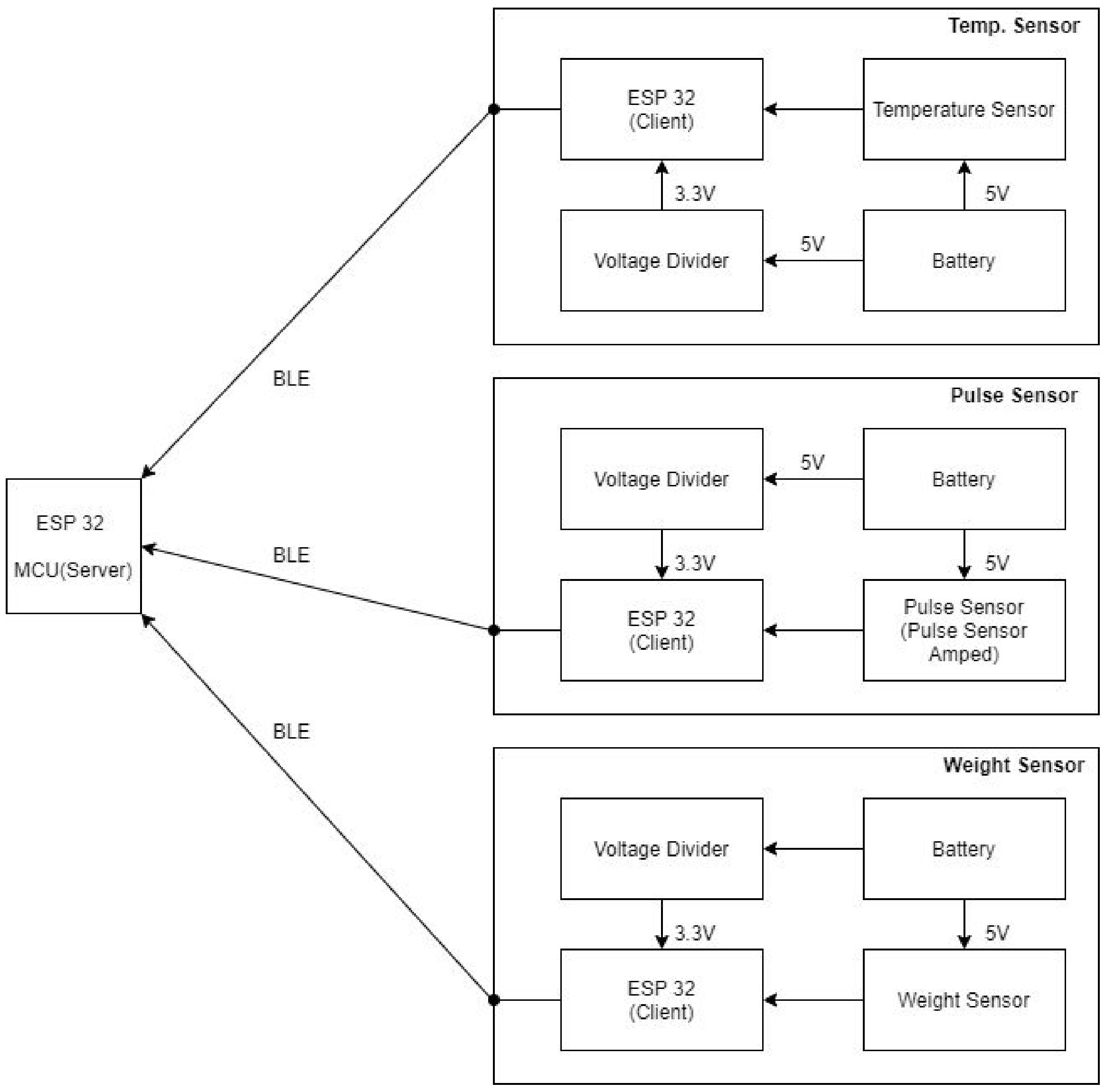

Temp. Sensor
ESP 32 (Client)
Temperature Sensor
3.3V
5V
Voltage Divider
5V
Battery
BLE
Pulse Sensor
Voltage Divider
5V
Battery
ESP 32
MCU(Server)
BLE
3.3V
5V
ESP 32 (Client)
Pulse Sensor (Pulse Sensor Amped)
BLE
Weight Sensor
Voltage Divider
Battery
3.3V
5V
ESP 32 (Client)
Weight Sensor

Figure 2.3

# 3. Statement of Work

## 3.1 Previous Work And Literature

Apple Watch is a wearable device that many athletes use. The watch has a heart rate sensor that can be used to measure and monitor heart rates. The watch also allows users to track their activities on their phones.

Apple Watch's main functionality is not its ability to measure body data whereas our product focuses solely on measuring data on the body. On top of the heart rate sensor, our product will be able to measure body temperature as well as body weight. In addition, our product will cost less than an apple watch.

## 3.2 Technology Considerations

**Hardware/Embedded**
MCU options

1. Teensy board
    a. Pros
        i. Easy to program
        ii. Small
        iii. 2 ADCs
        iv. Easy to test
    b. Cons
        i. No wifi (would need an addon)
        ii. No bluetooth (would need addon)
        iii. Would need to design battery tech
2. Pi zero w
    a. Pros
        i. Cheap
        ii. Battery pack already designed
        iii. Wifi built in
        iv. Linux from scratch (robust software framework)
        v. Easy to test
    b. Cons
        i. No ADCs (everything will have to be digital coming from the sensors)
        ii. No bluetooth
        iii. Linux from scratch takes time to set up
3. Custom design
    a. Pros
        i. Learning experience
        ii. Everything exactly the way we want it
        iii. Cleaner final result (no dongles/addons)
    b. Cons

i. Time consuming
ii. Bug prone
iii. More expensive to test
iv. Need to design *everything* from scratch

**Pulse Sensor options**

1. MAXREFDES117 Heart Rate monitor and pulse oximeter
    a. Pros
        i. Can be worn on finger or earlobe
        ii. Low Power
        iii. Small in size
        iv. Free Algorithm provided
        v. Works well with arduino and mbed platforms(example c code provided)
        vi. cheap
    b. Cons
        i. May require 5.5VDC
        ii. Requires arduino or specific mbed board(only 3 boards)
2. Pulse sensor Amped
    a. Pros
        i. Recommended by ETG
        ii. Works with multiple boards(including arduino)
        iii. Processing visualization software and arduino code provided
        iv. Only 3 connections (Power, common, data)
    b. Cons
        i. 5VDC power
        ii. 3-4 mA input
        iii. More expensive compared to MAXREFDES

**Front End - Web**

1. Angular
    a. Pros
        i. DOM manipulation
        ii. No extra packages needed
        iii. Two way data binding
        iv. Good server performance - Caching
    b. Cons
        i. Hard to use if inexperienced with MVC
        ii. Debugging is hard for beginners
2. Vue
    a. Pros
        i. Good readability
        ii. Beginner friendly
        iii. Two way data binding
    b. Cons
        i. Small community
3. Potentially useful libraries

   a. Charts
      i. Chart.js
      ii. D3.js
      iii. Plotly
      iv. Victory

**Front End - Mobile**

1. Android
2. IOS

**Front End - Priority**

1. Web-app
   a. Pros
      i. Anyone with a web browser will have access
      ii. No download required for users
      iii. Bigger user base
   b. Cons
      i. Inexperienced with web development
2. Mobile app
   a. Pros
      i. Experience with mobile apps
      ii. Easier access
      iii. Probably better UI
   b. Cons
      i. Having to choose between IOS vs Android
      ii. Not everyone will have access
      iii. Compatibility issues

**Cloud Services**

1. AWS
   a. Pros
      i. Scalability
      ii. Low Cost
      iii. Variety of Services
   b. Cons
      i. N/A (not many cons)

**Data Analysis/Databases**

1. Amazon DynamoDB
   a. Pros
      i. Highly scalable
      ii. Cost Effective
      iii. Flexible

   b. Cons
      i. N/A

In designing the overall functionality of our product, we decided that the MCU will be able to relay data to the users' phone via bluetooth to give the heart rate, temperature and weight data in realtime when wifi is not available. Then the users' phone can then relay the data to the database and cloud for the web application to view the data. When wifi is available, the MCU will relay the data to the database via wifi and to the phone via bluetooth. This way the users' phone won't require additional processing power to relay the data to the database.

### 3.3 Task Decomposition

In order to solve the problem at hand, it helps to decompose it into multiple tasks and to understand interdependence among tasks.

- Design MCU
- Order parts for sensors and MCU
- Assemble MCU
- Debug/troubleshoot MCU build
- Test Sensors with generic microcontroller
- Test Sensors with MCU
- Assemble sensors and MCU
- Setup mobile application
- Setup AWS server
- Setup AWS DynamoDB
- Connect mobile application with database
- Run tests with simulated data within mobile application
- Connect MCU to mobile application
- Calibrate sensors and MCU
- Work on data analytics for short and long term overviews
- Setup web and mobile application
- Test system and refine project where needed.

### 3.4 Possible Risks And Risk Management

Database and frontend is currently limited in terms of progress as both depend on the hardware side to be functional. So far, we have not decided on how the data measured will be processed and displayed onto the client's device. Designing and building our own microcontroller is also limited as no member of our team as made one before, and they will be consulting with experts such as ETG who have assembled their own microcontrollers before. Additionally, our product will ideally be a wearable product similar to pieces of clothing like headbands, wristbands, and shoes. No member of the team has prior experience working with embedding clothing with technology.

### 3.5 Project Proposed Milestones and Evaluation Criteria

One key milestone is to have a functional sensor by the end of the semester. We will need to test the accuracy of the sensor and calibrate it based on the results. In addition, having the MCU designed will be a major step towards building and testing the final product.

Another milestone is to have decent security. We will also need to test the security of the data that is being sent to the cloud.

## 3.6 Project Tracking Procedures

We will be using Trello as a method for tracking progression on each task that needs to be accomplished in order to create the final product. This method of SCRUM also documents the progression with time.

We will also be using GitLab to keep track of changes in our codes.

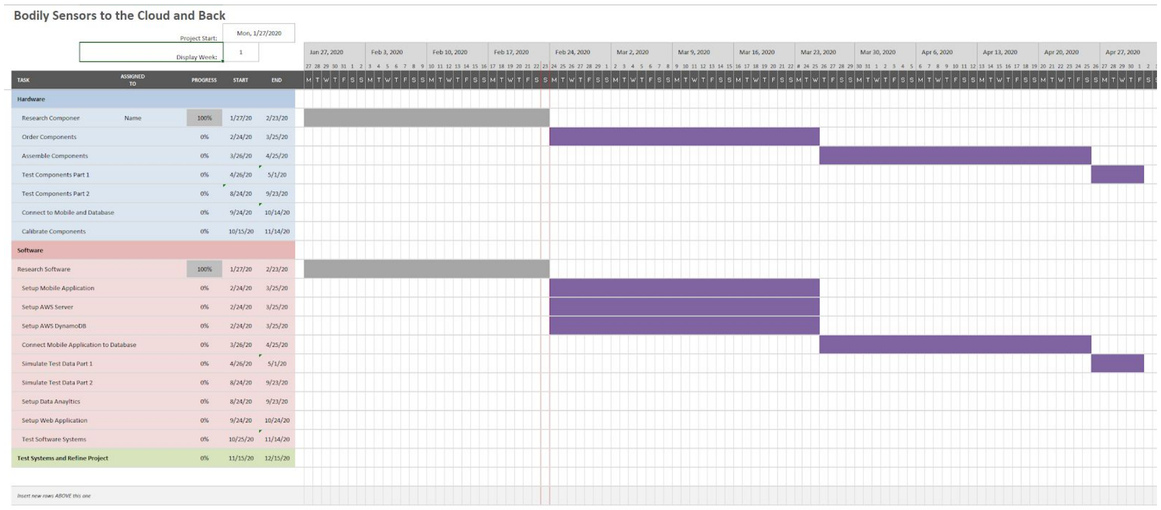
## 3.7 Expected Results and Validation

Our desired outcome is to have a product that is accurate and secure. The final product needs to be able to gain users' trust the moment it is being presented.

We can measure temperature, heart rate, and body weight using external tools and compare the results to the ones we measured using our product.

# 4. Project Timeline, Estimated Resources, and Challenges

## 4.1 Project Timeline



Bodily Sensors to the Cloud and Back

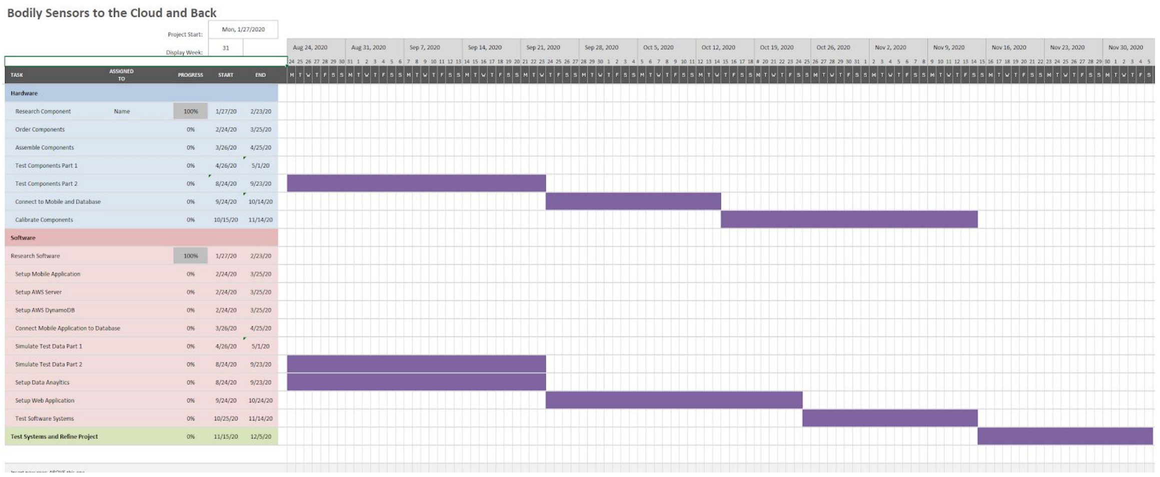
Project Start: Mon, 1/27/2020

Display Week: 1

| TASK | ASSIGNED TO | PROGRESS | START | END |
|---|---|---|---|---|
| **Hardware** | | | | |
| Research Componer | Name | 100% | 1/27/20 | 2/23/20 |
| Order Components | | 0% | 2/24/20 | 3/25/20 |
| Assemble Components | | 0% | 3/26/20 | 4/25/20 |
| Test Components Part 1 | | 0% | 4/26/20 | 5/1/20 |
| Test Components Part 2 | | 0% | 8/24/20 | 9/23/20 |
| Connect to Mobile and Database | | 0% | 9/24/20 | 10/14/20 |
| Calibrate Components | | 0% | 10/15/20 | 11/14/20 |
| **Software** | | | | |
| Research Software | | 100% | 1/27/20 | 2/23/20 |
| Setup Mobile Application | | 0% | 2/24/20 | 3/25/20 |
| Setup AWS Server | | 0% | 2/24/20 | 3/25/20 |
| Setup AWS DynamoDB | | 0% | 2/24/20 | 3/25/20 |
| Connect Mobile Application to Database | | 0% | 3/26/20 | 4/25/20 |
| Simulate Test Data Part 1 | | 0% | 4/26/20 | 5/1/20 |
| Simulate Test Data Part 2 | | 0% | 8/24/20 | 9/23/20 |
| Setup Data Anayltics | | 0% | 8/24/20 | 9/23/20 |
| Setup Web Application | | 0% | 9/24/20 | 10/24/20 |
| Test Software Systems | | 0% | 10/25/20 | 11/14/20 |
| **Test Systems and Refine Project** | | 0% | 11/15/20 | 12/15/20 |

*Insert new rows ABOVE this one*

Figure 4.1



Bodily Sensors to the Cloud and Back

Project Start: Mon, 1/27/2020

Display Week: 31

| TASK | ASSIGNED TO | PROGRESS | START | END |
|---|---|---|---|---|
| **Hardware** | | | | |
| Research Component | Name | 100% | 1/27/20 | 2/23/20 |
| Order Components | | 0% | 2/24/20 | 3/25/20 |
| Assemble Components | | 0% | 3/26/20 | 4/25/20 |
| Test Components Part 1 | | 0% | 4/26/20 | 5/1/20 |
| Test Components Part 2 | | 0% | 8/24/20 | 9/23/20 |
| Connect to Mobile and Database | | 0% | 9/24/20 | 10/14/20 |
| Calibrate Components | | 0% | 10/15/20 | 11/14/20 |
| **Software** | | | | |
| Research Software | | 100% | 1/27/20 | 2/23/20 |
| Setup Mobile Application | | 0% | 2/24/20 | 3/25/20 |
| Setup AWS Server | | 0% | 2/24/20 | 3/25/20 |
| Setup AWS DynamoDB | | 0% | 2/24/20 | 3/25/20 |
| Connect Mobile Application to Database | | 0% | 3/26/20 | 4/25/20 |
| Simulate Test Data Part 1 | | 0% | 4/26/20 | 5/1/20 |
| Simulate Test Data Part 2 | | 0% | 8/24/20 | 9/23/20 |
| Setup Data Anayltics | | 0% | 8/24/20 | 9/23/20 |
| Setup Web Application | | 0% | 9/24/20 | 10/24/20 |
| Test Software Systems | | 0% | 10/25/20 | 11/14/20 |
| **Test Systems and Refine Project** | | 0% | 11/15/20 | 12/5/20 |

Figure 4.2

For our timeline we have split the team into software and hardware. Software is furthermore split into 3 sub-groups: user interface, server, and data.

## 4.2 Feasibility Assessment

One big concern we have is data security. We need to be able to implement a secure way of sending data between applications and the cloud. This is important since user data protection is required. Furthermore, designing a microcontroller requires a lot of troubleshooting and debugging which can lead to unforeseen delays.  Our approach to work around this is to test all sensor functionality with a generic microcontroller to ensure the IoT sensor network is functioning properly when we have to test our custom MCU.

## 4.3 Personnel Effort Requirements

| Task | Estimated Time Per Week |
|---|---|
| Cloud application | 20 hours |
| Hardware application | 20 hours |
| Web Service | 20 hours |
| Web application | 20 hours |
| Mobile application | 15 hours |
| Testing | 20 hours |

## 4.4 Other Resource Requirements

Research components, order components, mounting equipment, packaging, wiring, installation laying, mount distribution boards, and writing distribution boards.

## 4.5 Financial Requirements

This project will require an estimate of $500.

# 5. Testing and Implementation

N/A

## 5.1 Interface Specifications

We currently do not have any hardware/software interfacing for testing.

## 5.2 Hardware and software

- Jest
- Mocha
- JUnit
- Postman
- KiCad

Jest and Mocha are libraries that are used to test JavaScript codes. These libraries can be used to test the functionalities of Vue.

JUnit is used to test Java applications. This unit testing tool will be used to test the functionalities of the Android app.

Postman is a platform for API development. It will be used to test the backend server.

KiCad is a CAD design tool for printed circuit boards(PCBs). This tool will be used to design the IoT circuits as well as the MCU PCB.

## 5.3 Functional Testing

- Test to see if the mobile and web apps pull correct data from the database
- Test if measured data can be displayed on the mobile app live
- Test if the mobile app can push data to the database
- Test if new users can be added to the database
- Test if users can login
- Test if data is displayed to the correct user

### 5.4 Non-Functional Testing

- Test wearability and sensor connection distance
- Test how long it takes to pull data and display it onto the frontend
- Test how fast users can login
- Test the responsiveness of the web app
- Pen-testing for user data

### 5.5 Process

N/A

### 5.6 Results

**N/A**

## 6. Closing Material

### 6.1 Conclusion

So far, we have started designing the MCU board, and Sensors that we will be using. Our end goal is to interface the MCU with our sensors in an IoT environment and have the MCU send data from the sensors to the cloud and a mobile app where the user can then view the data and analytics from a mobile or web application.

To accomplish our goals in the time frame we have, our development groups need to work concurrently. While the hardware team designs the MCU, the cloud team will be setting up infrastructure with AWS and implementing stubs to ensure communication between the cloud and the web app is running smoothly. Developing in parallel reduces the time cost to developing this project and gives us more time for testing.

### 6.2 References

N/A

### 6.3 Appendices

N/A