CSCI 370 Final Report

Keeping Labor Safe 2

Brooke Bowcutt
Katherine Chartier
Megan McFeeters
Emma St. Laurent

Revised December 10, 2023

CSCI 370 Fall 2023

Mrs. Donna Bodeau

Table 1: Revision history

Revision	Date	Comments
New	9/3/2023	Added project description (introduction), functional & non-functional requirements, potential risks, and definition of done
Rev – 2	9/14/2023	Added System Architecture & Team Profile
Rev – 3	10/19/2023	Added Software Test and Quality & Project Ethical Considerations
Rev – 4	12/5/2023	Added Future Work, Lessons Learned, and Acknowledgement
Rev - 5	12/10/2023	Implemented suggested revisions & added Technical Design

Table of Contents

I. Introduction

Keeping Labor Safe, LLC is a startup company dedicated to helping at-risk pregnancies in Third World countries that is interested in developing a new tool for monitoring labor. While labor marks the beginning of life, it can be potentially life threatening for both the mother and the child. These risk factors are much more prevalent in areas where there is reduced access to medical technology. The company aims for a highly compact system inspired by a pre-existing medical device, the Novii Wireless Patch System.

The company is modeling their system off the Novii Wireless Patch System due to its ease of use by the mother when laboring. Unlike typical hospital labor monitoring systems, the Novii uses a Bluetooth connection. This allows the mother to be able to move freely during the labor process. At the end of the day, this product is intended to make the mother's life easier, and allowing the mother to be able to be comfortable is paramount for the client.

This new system will connect to their custom web application using a Raspberry Pi. Our goal for this project is to create a wireless connection between the monitoring device and a laptop. In these countries we are aiming to service, there may not be reliable access to the Internet, so the data would need to be transmitted locally and without network requirements. During labor, the mother's heartrate, the child's heartrate, and uterine activity would be monitored. Seeing as no one in the group is pregnant, there is no way to capture real data. Using the Raspberry Pi, we will simulate these three pieces of data to transmit via Bluetooth to a secondary device.

II. Functional Requirements

The overarching goal of this project is to establish a Bluetooth connection from the Raspberry Pi to the web interface. Therefore, our only functional requirement is to connect the Raspberry Pi to a secondary device using a Bluetooth connection that can be established and maintained without utilizing an internet connection. This Bluetooth connection will be used to transfer the data read from the sensor that is attached to the Raspberry Pi. The secondary device should have some sort of web application to show that the data can be altered to fit into a web application. The transmitted data should also be altered to be put into a csv file. This is because the FRI web application that Keeping Labor Safe group 1 is working on takes in a csv file as input. Eventually, the transmitted data would be put into the FRI web application, so it is important that the data can be transmitted into a csv.

III. Non-Functional Requirements

- Secure system for Bluetooth connection
- Not reliant on internet or Wi-Fi connection
- Able to be used in developing countries or areas
- Easy to use and quick to connect

IV. Risks

Creating a product that is intended to be used by women in labor comes with inherent risks. First, the product must be safe for both the mother and child. As our product uses electrical wires for the Raspberry Pi, we must ensure that the wiring is correct and will not harm the mother or destroy the Pi, although the Raspberry Pi is not meant to be used in the final version of the product. Additionally, we must ensure that the data we are transmitting is correct to the hardware. Our product takes in data from some piece of hardware and then transmits that data via Bluetooth to another device. The data that we transmit must be the same throughout the entire process. If some of the data is altered during transmission, this could be of great risk to mother and child, as this data would be processed by an algorithm that Keeping Labor Safe has created to tell if the mother is having issues during labor. If the data was altered during transmission, it is possible that when the algorithm is run, the incorrect data would not indicate that there is a risk during labor. This could cause significant harm to the mother and the child because the doctors would not know that the labor is unsafe, and the mother may miss out on some lifesaving procedure.

An additional concern that the team had was whether the product needed to be compliant with the Health Insurance Portability and Accountability Act (HIPAA). After discussions with the client, we came to the conclusion that HIPAA compliance would be left up to the client. The client plans on paying an outside agency to make all the products from the three teams HIPAA compliant, and thus this was no longer a direct requirement for our design. However, we still had to ensure that our product could be made HIPAA compliant in the future, as if we created a product that could not be made HIPAA compliant, we would run the risk of the product being scrapped in its entirety. This would involve making sure data is transmitted securely and is stored properly and privately.

V. Definition of Done

Minimal useful feature set

- At a minimum, the integration must allow for data to wirelessly transfer from the sensor patch on the mother's belly to the laptop, and from there, to the downloaded app or website application.
- Ideally, we would find a way to make this transfer HIPAA compliant; but that may be beyond the scope of our abilities and time on this project and will be left to outside sources. Eventually, HIPAA compliance will be a requirement for the final version of the project.

Tests the client will run before accepting the software

- The client did not specify any tests in particular they will run before accepting the software. We will be running our own unit tests and general tests to ensure correct functionality. These tests will include testing that data is being transferred at the correct rate, and that the data is being transferred accurately without corruption.

How the product is delivered

- The actual hardware components will be shipped back to the client directly. In creating the product, we had to use additional pieces of hardware supplied by Professor Phil Romig. These pieces were returned to Professor Romig, so the team put together a list of supplies for the client so that they could purchase these parts for themselves. In addition to sending the two Raspberry Pis back, we have also sent a wiring diagram (shown in the next section) so that they are able to use the product themselves. We have also agreed to have additional meetings with the client to show them how to run the code and wire the breadboard if they are unable to get the product to work with just the written documentation.
- For the software, the team has created our own individual GitHub repository. We have given the clients access to this GitHub repository, and there are instructions on how to run the code and what dependencies are needed in the repository.

VI. System Architecture

Below are four images describing the system architecture of the project. The first image was the flow diagram of the Monica Novii system. This includes the Novii Pod and Patch and how they interact with their own systems. The second image describes the original proposed idea given to the team by the clients, including the Novii Pod and Patch and how they could interact with the FRI Web Application. This is followed by the diagram describing the actual flow of our implementation, which matches the original proposed flow. The final image is the full system architecture, including both our project and the other project components, and what technologies are needed to complete the full project.

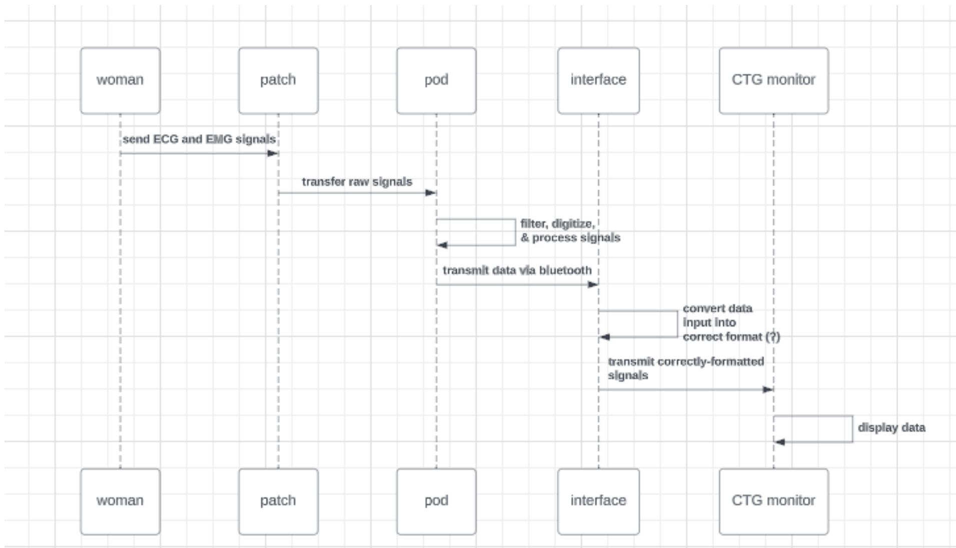

Figure 1: Sequence Diagram of Current Flow

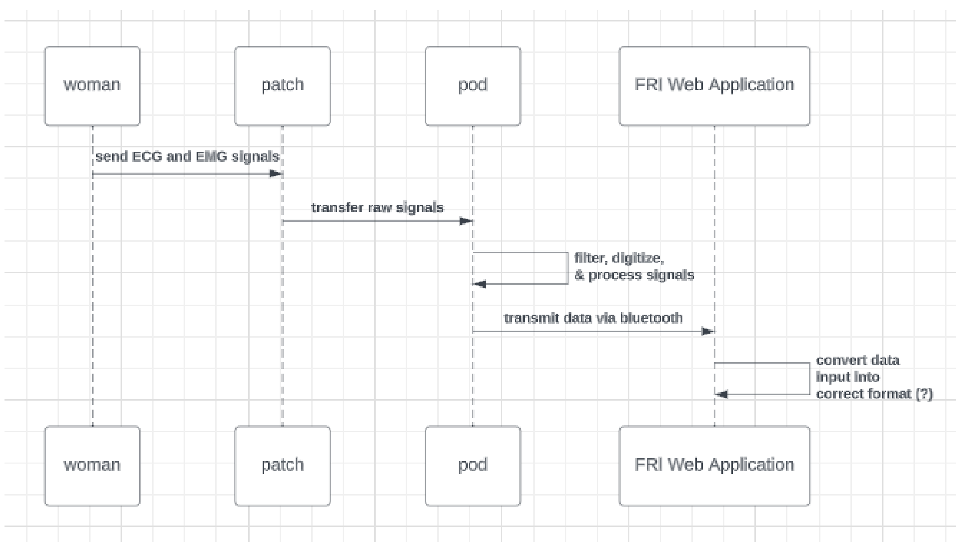

Figure 2: Sequence Diagram of Proposed Flow

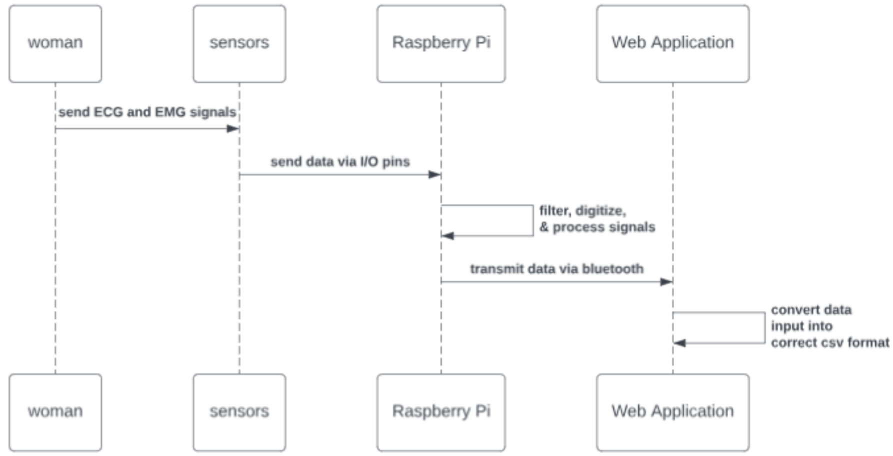

Figure 3: Completed Project Sequence Diagram

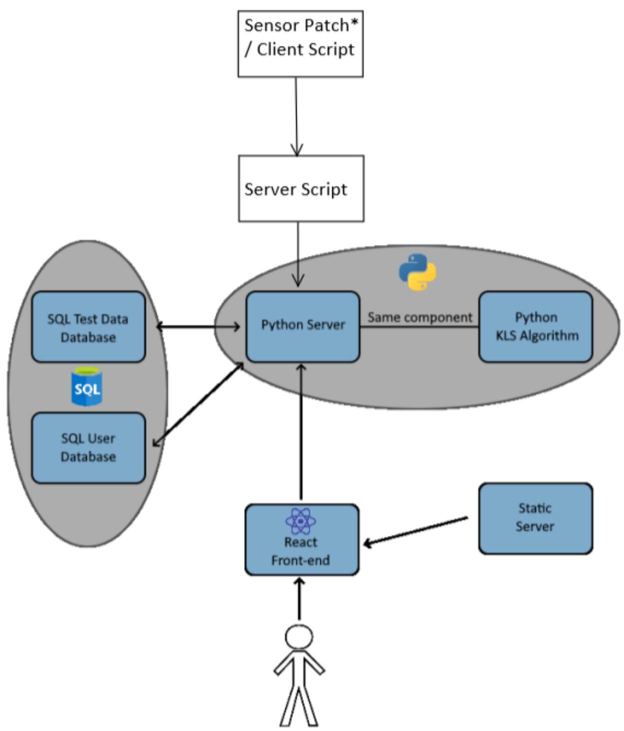

Figure 4: Full System Architecture

*Note: Sensor patch currently approximated by Raspberry Pi

VII. Technical Design

The Bluetooth element of our project was implemented using the PyBluez library. This is an adaptation of the Linux Bluetooth protocol stack Bluez for use in Python code. We chose this library because our group's Raspberry Pi coding experience was primarily in Python, so it would have less of a learning curve than other Bluetooth libraries.

Our chosen protocol for communication was RFCOMM (radio frequency communication). RFCOMM provides many features that other Bluetooth protocols don't, such as reliable connections, error detection, retransmission, and flow control. It was extremely important for us to ensure minimal data loss during transit, since the data we are working with has potentially life-threatening information.

This project was built upon running Python scripts on a server and client machine. The client in this case is the Raspberry Pi, which handles all data sensing and digitizing. The server is any computer with Bluetooth functionality, which can additionally run the FRI web application to display the data in real time. In order to facilitate this communication, we used Bluetooth sockets. Bluetooth socketing is very similar to TCP socketing, where we establish a reliable point-to-point communication channel between 2 devices. Upon establishing a connection, we can send a continuous stream of data from the Raspberry Pi to a computer. This data can be sent endlessly, so we indicate the end of communication through sending an End of Transmission character.

Bluetooth only allows for 20 bytes of data to be sent per packet. PyBluez handles fragmentation, so we didn't have to worry about the 20 bytes maximum transmission unit. However, the send() function in PyBluez can only take up to 1024 bytes, so we did have to develop code that would split our data into chunks of 1024 characters. We do this by using numpy arrays to store data, which can be easily converted into string format with numpy's array2string() function. We chose to convert into ASCII characters before transmission because it allowed us to easily recompile the packets into a single string upon receiving them and minimize issues with splitting arrays. This made it easier to put everything into a CSV file, which we focused on so that our code would fit in with the existing codebase.

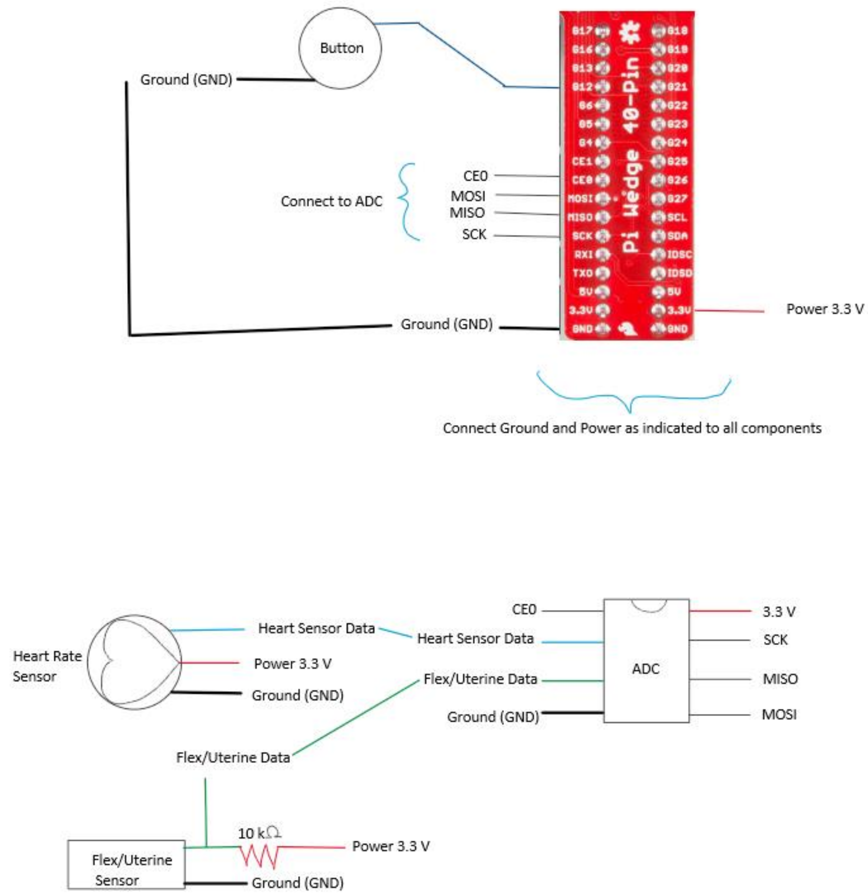

Figure 5: Wiring Diagram

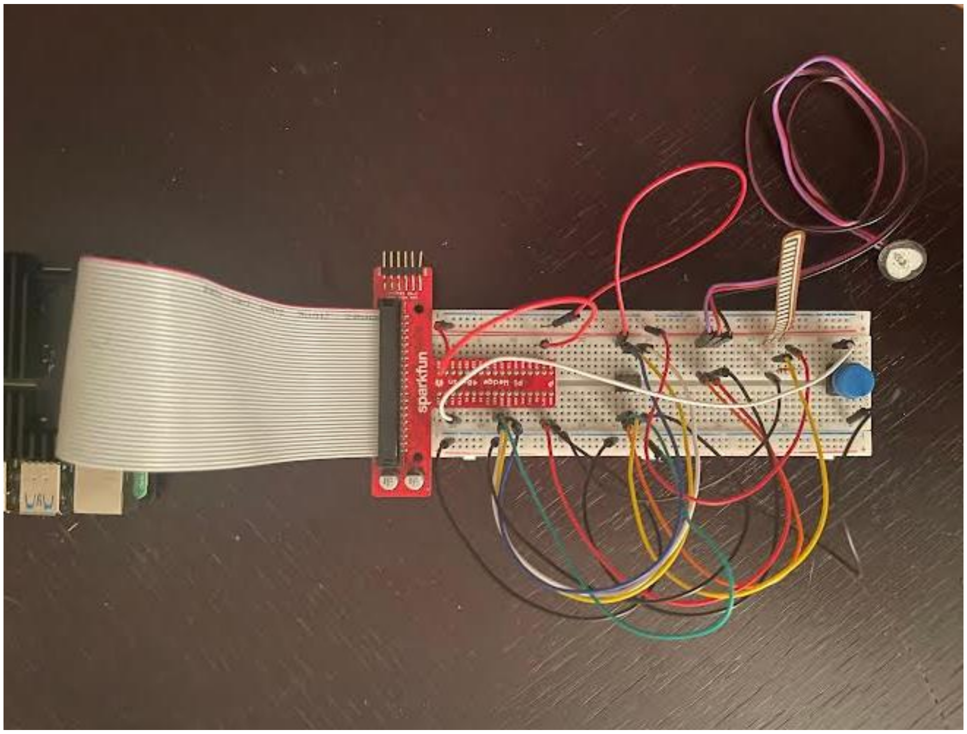

Figure 6: Photograph of Raspberry Pi (left) and breadboard (right), including the full wiring

The figure above displays the full Raspberry Pi setup, including both sensors. The Raspberry Pi is connected via ribbon cable to a Sparkfun Pi Wedge, which is the small red board in the middle that connects the Raspberry Pi's GPIO pins to the breadboard. The heart rate sensor is resting on the table above the breadboard and is beside the flex sensor, which we used to simulate the uterine activity measurement. The blue button on the right is used to begin the Bluetooth connection with the server computer. The final component shown on the breadboard is an ADC (analog-to-digital converter), which we connect the heartrate and flex sensors to in order to digitize their measurements and send them off. The heart rate sensor was used for both the mother and fetal heart rates.

Technical Design Issues

- The Monica Novii Pod which we were initially going to be working with could not be used any further. It was determined that we lacked the knowledge necessary to use a medical device in unintended ways. There were also significant ethical concerns associated with altering a medical device without any contact with the company that developed it. Instead, we compromised with the client and chose to complete the project using a Raspberry Pi to simulate using a sensor patch. We believe that this is a sufficient proof of concept for the client's needs.
- The team was never provided access to all the resources/repositories that would have been of use. This didn't stunt development too much, but it became an issue when trying to incorporate our code into the existing web application. We were not whitelisted for most Keeping Labor Safe Azure resources, which we were not made aware of until the end of the project.

VIII. Software Test and Quality

We used the PyTest framework to create tests for the code. We primarily tested our CSV formatting functionality, which is the final step in our code after data has been transmitted from the Raspberry Pi. The tests are structured to ensure the edge cases are covered, which means missing data can be handled appropriately. This is done by creating sample strings sent from the client that are missing data to show that our application can handle missing sensor readings efficiently and correctly without erroring or breaking, and that the strings sent from the client are being formatted correctly. The test results are used to verify that our code is properly working to manipulate our data for future use. These tests can cover all functional and non-functional requirements.

The feature of having a secure system for Bluetooth connection cannot be tested as it is a requirement from the hardware components which we are using from a third party. Not being reliant on the internet or Wi-Fi connections does not need to be tested either as it is being developed without reliance on the internet and will never rely on it. Additionally, the requirement of usability in developing countries or areas does not need to be tested because it is not a software requirement. Connectivity does not need to be tested because that is not related to our software and is instead a hardware concern. Lastly, ease of use does not need to be tested since our group is not creating any type of final user interface.

The current structure of the database and web application is quite limiting for the team. It requires CSV files for data processing, which must be formatted to contain 240 data points per row. This was put in place because the clients expect the sensor to read 4 times a second, so the web application was built upon that dependency. With the current Raspberry Pi, the team is only able to transmit 20 bytes of data at a time, so we developed code that could send a continuous stream of data and reassemble packets upon arrival. Our tests are able to ensure that the connection will not end until our chosen End-of-Transmission character is received. We cannot test the transmission itself, but we are able to make sure our transmission starts and ends at the correct time.

The tests included the following cases:

- Ensuring the client-server can handle the correct case
  - It is important to ensure our current structure is able to correctly handle the proper use case for data transmission. This is tested through having a CSV file created with correct data formatting and all data present.
- Ensuring the client-server can handle when there is data missing
  - With the Bluetooth transmission being out of our responsibilities, it is important to create a structure that is able to handle data transmission if Bluetooth transmitted packets are lost. Our code is able to save data to a CSV file regardless of any missing data points or missing sensors, so we can guarantee that a loss of data will not throw any errors. The web application informs users if a CSV file is missing data, so the full flow project can handle data loss without stopping transmission.

- Ensuring the client-server can handle when the data is out of order
  - It is also important to note that the data may be received out of order on the server side. As a result, there are dummy data CSV files that have the three data arrays out-of-order to ensure the data can still be filtered and saved correctly.

IX. Project Ethical Considerations

Principle	Reasoning
IEEE 1.01 - Accept full responsibility for their own work.	To always ensure we were holding ourselves accountable for our work, we followed this principle. This is done through code reviews and proper documentation for the users of the project to reference when confused.
IEEE 1.02 - Moderate the interests of the software engineer, the employer, the client and the users with the public good.	Since the project is focused on medical information, it is important that all users are not exposed to any harmful measures from the software. As a result, this product is developed with the intention of being a prototype to ensure that in the future, it can be transitioned into a HIPAA compliant device that has been certified by a certified HIPAA professional.
ACM 3.12 - Work to develop software and related documents that respect the privacy of those who will be affected by that software	Since we are developing a product that will be used in a medical setting, it is important to always have in mind the privacy of the user of the sensor. The sensor will be transmitting personal medical data, including heart rates and uterine activity, which needs to be properly protected to ensure there is no violation of privacy for all users involved.
ACM 3.13 - Be careful to use only accurate data derived by ethical and lawful means, and use it only in ways properly authorized.	This principle was important while considering the initial objective of our project, with utilizing General Electric's Monica Novii device. We were unable to get proper authorization to use the Bluetooth transmitted data that was being gathered by the Monica Novii system. As a result, we transitioned to creating our own data

	transmission device with the Raspberry Pi to ensure all ethical considerations would be met.

X. Project Completion Status

The project is complete with all updated requirements from our client met. Our initial requirements could not be met due to complications with the Monica Novii Pod, so our completion status is based upon our success with the Raspberry Pi. We were able to create a client-server model with a Raspberry Pi. This software is able to communicate between the Raspberry Pi and the computer with the goal of demonstrating that a connection can be wirelessly made to transmit the three data points of mother's heart rate, fetal heart rate, and uterine activity.

We were able to achieve the expected functionality using a computer running Ubuntu. The Raspberry Pi is able to collect data from a heartrate sensor and flex sensor connected via breadboard. The Raspberry Pi and computer each run a Bluetooth script to exchange data, which is then displayed on a web application. However, we were not able to test with other operating systems since security and firewall software on our personal computers restricted Bluetooth communication. In theory, the Python script written should work the same on other operating systems, since Python is platform independent. We were also unable to incorporate our code into the existing web application since we were not properly whitelisted for Azure resources. As a result, our mockup web application is in place to demonstrate the ease of incorporating our Bluetooth functionality into the FRI web application. Since we are handling CSV creation in accordance with the web application's expected format, our project should be able to easily plug into the existing code.

Overall, the definition of done is met for this project, allowing the client to move to the next step in the process.

XI. Future Work

There are three overarching goals for the future work of this project. The primary next step would be showing our portion of the project to receive a grant for Keeping Labor Safe. The purpose of our project was to create a prototype that could demonstrate transmitting data to be plugged into the web application that was created by another Keeping Labor Safe field session group. This grant would allow Keeping Labor Safe to develop its own hardware in place of the prototype we created, since the Raspberry Pi setup is not intended for use in the field. The timeline for this is dependent on the grant acceptance timing from our client.

Additionally, the future work of this project includes creating a full flow product implementing the work of all three Keeping Labor Safe groups from this semester. This would involve plugging our product into the web application that is deployed through Microsoft Azure. Our code is designed to work around existing limitations and dependencies built into the web application, so it would theoretically already work with some minor adjustments. This will take minimal time as the Bluetooth functionality we wrote will just need to be called in the code of the FRI web application owned by Keeping Labor Safe.

Lastly, we would like to remove all CSV dependencies from the full flow project. As the web application is currently set up, all data must be read and written into a CSV with correct formatting. As a result, the code we have written puts the data into a CSV file before handing it off to the application. To eliminate the stutter step of writing and reading to a file to ensure real time data transmission, the team would like to remove all CSV dependencies from the code. This would mainly involve editing the web application itself, since our code does not rely on a specific file format. This will take a substantial amount of time due to all previous and current groups building their code upon the CSV dependencies.

XII. Lessons Learned

- Clear communication with the client is vital to the success of a project; continuously shifting requirements and goals from the client provided significant roadblocks for planning what needed to be done in advance. A clear definition of the requirements and a general plan for the duration of the semester should be clearly discussed at the beginning of the project, so both parties are very clear on the expectations for the end goal of the project.
- Good documentation is very important within complicated codebases, especially when it can be expected for those codebases to be transferred to other teams in the future, as with these projects. Codebases that are rushed together often do not have the clearest logic, and this problem can be exacerbated without documentation, which makes them difficult to work with.
- Code should be designed with future expansions to the code in mind. Otherwise, future work can be significantly constricted by attempting to work within the narrow and suboptimal frameworks put in place by the previous code.
- A plan that can be easily modified or changed should be established. We quickly learned how to adjust goals throughout this project as we had to maneuver from using pre-existing products, like the Monica Novii, to developing our own to transmit the data. This is important to ensure a quality product is still delivered.
- How to work with non-technical clients to ensure all goals are met and understood clearly. We now understand how to explain goals and demonstrate progress in a sense that eliminates the communication gap between technical terms and the client’s goals.

XIII. Acknowledgments

We would like to thank our clients, Mark Evans and Gregory Ryan, for their support throughout this project. Without them, we would not be able to complete such a successful project.

We would also like to thank our advisor, Donna Bodeau, for her support. Throughout this project, Donna was able to provide support through client meetings and brainstorming to ensure our project was successful and meaningful.

We would also like to thank Dr. Phil Romig for his support with the Raspberry Pi for this project. Dr. Phil Romig was able to provide support with hardware while we were learning how to implement a Raspberry Pi in this aspect of software engineering.

Lastly, we would like to thank Keeping Labor Safe teams 1 and 2. The collaboration was helpful to start the development of a full-flow product.

XIII. References

[1] Evans MI, Britt DW, Evans SM, Devoe LD. Improving the interpretation of electronic fetal monitoring: the fetal reserve index. Am J Obstet Gynecol. 2023 May;228(5S):S1129-S1143. doi: 10.1016/j.ajog.2022.11.1275. Epub 2023 Mar 17. PMID: 37164491.

XIV. Team Profile

Katherine Chartier

Class: Senior

Major: Computer Science

Hometown: Colorado Springs, Colorado

Experience: Transamerica Internship, ICR Internship, CS 101 and 128 TA

Hobbies: Hiking, playing board games, watching movies

Emma St. Laurent

Class: Senior

Major: Computer Science

Hometown: Castle Rock, Colorado

Experience: Software Engineer Intern @ Charter Communications, CSCI250 TA

Hobbies: Playing music, skiing, baking

Megan McFeeters

Class: Senior

Major: Computer Science, Space focus

Minors: Digital Systems, Robotics and Intelligent Systems

Hometown: Littleton, Colorado

Experience: Lockheed Martin Internship, CSCI 128 TA, CSCI 101 TA

Hobbies: Theater, Singing, Reading, Writing

Brooke Bowcutt

Class: Senior

Major: Computer Science, Business focus

Hometown: Cheyenne, Wyoming

Experience: Embedded Software Test Engineer @ Gogo Business Aviation, CSCI 200 TA

Hobbies: Crocheting, reading, sewing, running