UiT The Arctic University of Norway

Faculty of Science and Technology
Department of Computer Science

System for measuring, storing, and visualizing pressure of patients dependent on a tracheostomy tube

Fredrik Mørstad

INF-3981 Master's Thesis in Computer Science – July 2023

“Det blåser nordavind fra alle kanter i dag, Solan.”
–Ludvig

Abstract

Wrongful removal of tracheostomy tubes is often associated with high drama for all parties involved. With children's repository system being under-documented, there is no consensus on when these tubes should be removed for children at different ages. Measuring the pressure generated by these children can help doctors make better decisions and give the tools for helping medical researchers reach a consensus on the correct time to remove the tracheostomy tubes. This thesis presents a system for measuring, storing, and visualizing measurements of children dependent on a tracheostomy tube.

Creating a wearable IOT device integratable with tracheostomy tubes allows pressure to be sampled over longer periods of time. These continuous measurements provide new and insightful measurements to be analyzed by doctors via a website. These measurements can then permanently be stored following the GDPR directives providing a database of measurements helping research in this area of medicine.

Acknowledgements

The project was presented by Consultant Bård Forsdahl (Pediatric department at University Hospital of North Norway), and the medical information used in Chapter 1 and 2 is based on his experience and medical expertise.

Contents

List of Figures

List of Tables

List of Abbreviations

I^{2}C Inter-Integrated Circuit

API Application Programming Interface

CAD Computer Aided Design

cm Centimeter

CRUD Create, Read, Update and Delete

CS Chip Select

EMI Electromagnetic Interference

EU Europian Union

GDPR General Data Protection Regulation

GHZ Gigahertz

GND Ground

HTTP Hypertext Transfer Protocol

HTTPS Hypertext Transfer Protocol Secure

HZ Hertz

ID Identifier

IOT Internet of Things

JSON Javascript Object Notation

JWT JSON Web Tokens

KB Kilobyte

kPa Kilopascal

mA Milliampere

mAh Milliampere hours

MISO Master Input Slave Output

mm Millimeter

MOSI Master Output Slave Input

ms Millisecond

OS Operating System

Pa pascal

PCB Printed Circuit Board

PETG Polyethylene Terephthalate Glycol

PLA Polylactic Acid

POC Proof Of Concept

psi Pounds per square inch

RAM Random Access Memory

REST Representational State Transfer

SCL Serial Clock

SCLK Serial Clock

SD Secure Digital

SDA Serial Data

SDK Software Development Kit

SMTP Simple Mail Transfer Protocol

SPI Serial Peripheral Interface

SS Slave Select

SSL Secure Sockets Layer

TLS Transport Layer Security

UART Universal Asynchronous Receiver-Transmitter

URI Uniform Resource Identifier

USB Universal Serial Bus

USB-C Universal Serial Bus type C

XXS attack Cross-Site Scripting attack

1 Introduction

Over the last decade, IOT has seen vast development and IOT devices are being used for new applications at a rapid pace [1]. The main reason for this growth in popularity is the device's high performance compared to its small size. Over time the manufacturers of these devices have created small and high-performing devices at a very low price allowing almost everyone to create an IOT system. These characteristics make it applicable for medical research, weather monitoring, and pollution detection, to name a few. The IOT devices, often called microcontrollers, also provide easy communication between a vast majority of hardware devices allowing complex systems to be built from small independent devices. This modular approach is the key to why IOT devices can be used in many different fields. One of the fields that have seen the growing usage of IOT devices is medical research [2]. From monitoring patients' health to creating safer homes for elderly patients are all applications where IOT solutions have had a powerful impact [3]. This thesis sets out to create a fully stand-alone solution for monitoring lung pressure for children who have undergone a tracheostomy. All from creating an IOT device measuring pressure to giving the physicians the tools necessary for understanding the data gathered.

As mentioned, the target group for this thesis is children dependent on a tracheostomy tube who have undergone a procedure to fit this tube into the front of the neck. This surgical procedure creates an alternative from the windpipe and through the newly created hole in front of the neck. This allows air to flow in and out of the new path instead of going up the windpipe and

throughout the mouth. There are numerous reasons why this is necessary; however, fundamentally, they solve the same problem, allowing the patient to breathe with less difficulty. After the surgery, the patient is fitted with a tracheostomy tube; this tube is inserted through the new opening, connecting the new hole with the windpipe. With this tube, all air flows in and out before entering the voice box, meaning patients usually have problems speaking because no air flows through the voice box. Therefore the tracheostomy tube can be fitted with a one-way valve allowing air to flow in through the tube, but when breathing out, the air passes beside the tracheostomy tube, over the vocal cords, and out of the mouth, at the same time giving the patient a voice. With this system, the patient breathes in regularly but must breathe through their nose. Children able to use the one-way valve are the main focus of this thesis, as their respiratory development is under-documented. Creating a better understanding of the development of the respiratory system can be useful in medical research and in creating better day-to-day care.

To better understand why gathering data on the development of children's respiratory system can improve day-to-day care, we first need to understand today's medical practice. Currently, whether a patient is dependent on further use of a tracheostomy is based on doctor-patient communication, some lung capacity measurements during checkups, and advanced and expensive medical equipment such as fiberoptic scopes. The idea of measuring the pressures in the trachea via the speaking valve is new and can therefore provide new information about the children's development. While the new measurement can provide a unique insight into the patient's development, the physician's experience combined with the relation to the patients is still the most vital part of the progression. The physician also uses samples taken during consultations to better understand the patient's state; however, these are small amounts of samples and do not give the full picture of the development. With a more detailed description of lung pressure at different stages of child development, the physician may better understand the current state of the child. Furthermore, for the children using the device, the physician can see the pressure readings for the entire duration of the day with high precision. This provides a unique insight into the pressure created in the lungs, especially during low and high-stress periods. Seeing how the child handles sleep as well as a relatively high activity allows the physician to get a deeper understanding of how the child is developing and progressing.

The ability for the device to be used during regular activities without limiting the child allows for a unique insight into the patient's current development. By having the child wear the device over multiple days, the device captures multiple key periods, which are hard to replicate in a controlled environment. For this to be possible, the device must be small enough to be wearable by a child and have enough battery to run for multiple days.

Creating a device capable of measuring pressure while mounted on the tracheostomy tube raises many challenges. The main challenges can be divided into two areas: size and power consumption. With the tracheostomy tube being a cylinder measuring only $37 \times 20 mmMillimeter(mm)$, the IOT devices have to be small enough not to limit the patient's everyday life. This restricts the viable solutions in terms of the size of the hardware, which affects battery life and performance. The device should measure the pressure produced by the users at a high sample rate. The conflicting objective of a small size while having high performance and long battery time forces the solution to find a balance between the size and the overall performance.

While this thesis mainly focuses on creating a device capable of measuring pressure, the system also needs a solution for storing and representing data. With the device ideally running for multiple days, having a volatile storage system would not be an acceptable solution as many things can create an unexpected failure causing loss of all the data. This is why persistent device storage is a vital part of a reliable solution. With the amount of data stored every day, the device needs to be able to offload the data for long-term storage. This will allow a physician to have a central location to see and understand the measurements. This requires the overall solution to provide a platform where the physician can get the visual representation of data. Since this device handles medical data, the platform should either have a high level of security, which usually means building on top of an existing system already used by hospitals, or anonymize the user completely.

1.1 Problem Definition and Requirements

This thesis presents a solution for monitoring, storing, and displaying pressure measurements of children dependent on a tracheostomy tube. The measurements are gathered by an IOT device created to be used with the tracheostomy tube, and the devices should sample at a desired rate. The device should also provide reliable storage while simultaneously measuring at the desired sampling rate. The data gathered from the device should be stored on a system securing the patient's privacy and ensuring secure health data storage. Finally, the data is presented to the physician on the visualization platform allowing the physician to access and interpret the readings easily.

1.2 Scope and Limitations

Creating a fully stand-alone solution to the abovementioned problem makes for a broad and complex task. It is, therefore, necessary to limit the scope of the different aspects of the problem. Both the monitorization and storage system should follow the requirements described above. The website, however, has some limitations as the website is mostly created as a POC to showcase a fully stand-alone solution. The following vital subjects for the website and the related backend logic are not in the scope of this solution:

- Scalability: How the system handles scaling to more traffic in different areas.
- Availability: How the system can minimize and recover from eventual downtime.
- Reliability: How the system copes with failure at different segments.

While the device is the main focus of this thesis, creating a device capable of running for multiple days must handle many different aspects. As this thesis is focused on creating a device usable for physicians there are some aspects that are necessary at a later stage but are not relevant at this point. The following features for a wearable device are not in the scope f this thesis:

- Water resistance: Creating water resistance or providing a high degree of protection against water is a long and difficult process. While it is relevant to provide some protection it is not important to guarantee this attribute in this thesis
- Handling large temperature fluctuations: Sensors are affected by temperature differences and this thesis is not focused on providing high accuracy in every condition.

2 Background

This chapter outlines the motivation behind the thesis as well as relevant concepts of which the solution is based upon. This section also describes the different relevant domains and the challenges the implementation must handle.

2.1 Motivation

Children with a tracheostomy tube can outgrow their need for the same tracheostomy tube. There is no consensus in the medical profession regarding the time when to remove the tracheostomy tube. If the tube is removed too early, it will cause the patient to collapse and requires the need for placing a new tracheostomy tube. This is often associated with high drama and wants to be avoided by all involved. The patient is experiencing suffocation when not enough air enters the lungs.

The fear of removing the tracheostomy tube too early leads to the other opposite, removing it later than necessary. Having a tracheostomy tube increases the risk of lung infections many folds, often it will compromise swallowing, and the voice is often weaker than necessary even when using a speaking valve [4].

Timing the right moment for removing the tracheostomy tube is, therefore in the patient's interest. At the moment, there is a study taking place at

Children Hospital Stanford in the US and at Akershus Uiversitetssykehus and Universitetssykehuset Nord Norge, both in Norway, measuring the tracheal pressures in children with a tracheostomy tube. Up until now we have only been able to make measurements at the outpatient clinic when the patients come for a check-up. The manometer that is used is handheld and does not store the measurements. The patients can not move during measurements. A new small device that can be worn by the patient during activity and store the data will gather data no one has been able to get before.

This device will also make some of the expensive and sophisticated medical equipment redundant regarding timing the right moment for decannulation. Making this device open source will include the possibility for underresourced areas to get it.

2.2 Internet of Things

As stated in Chapter 1, an IOT is a key part of this thesis. While the term Internet of Things was first introduced in 1999 [5], a formal definition of IOT is to this day not commonly agreed on. In general, IOT is an area of modern technology that encapsulates how smaller electronic devices and different peripheral components ("things") using interoperable communication protocols to create a more complex system ("device"). While a general definition is still not agreed upon, the adaption and development of IOT continue to increase. While there are many contributing factors, one of the main reasons for the increase in adaption is the low cost and high performance of microcontrollers.

Microcontroller is a generalized term for a small computer implemented in a single integrated circuit. These controllers vary in complexity, and some are created for simple peripheral communication and data processing, while others can support complex tasks such as supporting WIFI or using machine learning. While microcontrollers are not required for devices to be categorized as an IOT device, more and more systems use these controllers as microcontrollers can easily integrate other devices. This is possible as microcontrollers provide integrated pins that can connect to an arbitrary sensor.

2.2.1 Peripheral Communication

As mentioned, one of the main reasons IOT devices have emerged as a viable solution in multiple areas in the last decade is their modular approach. For microcontrollers to communicate with various hardware components, the microcontroller often supports analog and digital communication, with Serial

Peripheral Interface (SPI) and Inter-Integrated Circuit (I^2C) being the most widely supported digital communication protocols.

Serial Peripheral Interface

The serial peripheral interface is a communication protocol allowing synchronous communication between components or devices [6]. The protocol splits the devices into two categories master and slave, where there is only one master with one or more slaves. The master component is often a more complex component, such as microcontrollers, while the slaves are simpler peripheral devices, such as sensors or Secure Digital (SD) cars. The interface bus consists of four pins:

- Serial Clock (SCLK)
- Master Output Slave Input (MOSI)
- Master Input Slave Output (MISO)
- Chip Select (CS) or Slave Select (SS)

and each line or connection has a specific purpose. The SCLK line or the clock signal is responsible for synchronization between the two devices. The MISO and MOSI are the data lines, and it is used for sending data to the slave (MOSI) or sending data from the slave (MISO). This allows SPI to send and retrieve data simultaneously(full-duplex) [7]. The last line is the slave selector line, and it is responsible for indicating which slave is the recipient. The SS line combined with multiple data lines is the reason SPI can handle multiple slaves and fast data speed compared to other protocols such as I^2C and Universal Asynchronous Receiver-Transmitter (UART) [8, 9].

Inter-Integrated Circuit

Similarly to SPI, I^2C allows for multiple slaves, and it also allows for multiple masters using only one data channel. The interface uses one clock line Serial Clock (SCL) and one data line Serial Data (SDA). To support multi-master/slave, the I^2C protocol uses a message protocol. The message protocol consists of three data transfer sections and three synchronization stages, as seen in Figure 2.1. The first synchronization stage is the start signal, and then it is the not-/acknowledged stage, where an ack or nack is sent between each data frame. Finally, a stop signal is sent, signaling that the massage is completed. The

data transfer is done in three stages; the first stage is the address stage which dictates which device is the recipient. This is done as I^{2}C does not have a slave select line as SPI, and therefore each message must contain the desired recipient. After the address package, the master has to set a read/write bit indicating if the master wants to send or read data from the slave. When all these header packages are set, the data can be sent or received via 8-bit data frames. The protocol supports an arbitrary amount of data frames, meaning data will be sent until the stop signal is sent.

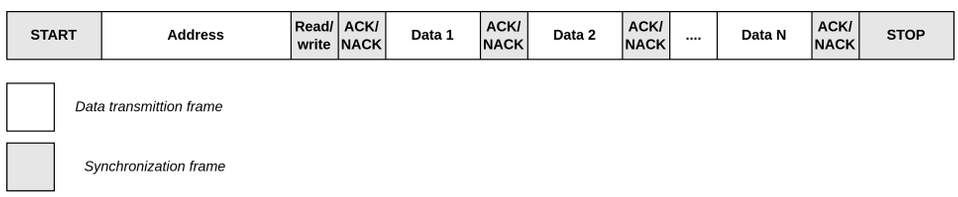

Figure 2.1: I^{2}C message structure

2.3 Capstone

This thesis builds on the Capstone thesis, which built a proof-of-concept for measuring device testing if key functionality was possible [10]. The device created in the Capstone thesis was only created to test if a small IOT device can measure and store pressure readings at 50Hertz (HZ). The methods for measuring the pressure uses many of the techniques used in the Capstone project. The next section describes the different methods for measuring the pressure using a Honeywell sensor as well as calibrating the sensor.

2.3.1 Piezoresistive Pressure Sensors

The sensor used is a Piezoresistive pressure sensor, and it works based on the Piezoresistive effect. The Piezoresistive effect detects the change in resistivity with applied stress, and based on that change in resistance; one can convert it to applied pressure [11]. This is done using a transfer function which defines the relation between change in resistance and applied pressure.

2.3.2 Transfer function

The relation between the output, change in resistance, and the different device-specific hardware for a Honeywell sensor is defined by Honeywell [12] as:

$$Output = \frac{Output_{max} - Output_{min}}{P_{max} - P_{min}} + (Pressure - P_{min}) + Output_{min} \quad (2.1)$$

where:

$Output_{max}$ = Ideal output at maximum pressure % of 2^{14} counts
$Outpu_{min}$ = Ideal output at minimum pressure % of 2^{14} counts
$Output$ = digital pressure output in counts
P_{max} = Maximum pressure output
P_{min} = Minimum pressure output

With this definition, the pressure applied can be extracted, giving the equation: [13]:

$$Presure = \frac{(Output - Output_{min}) \cdot (Pressure_{max} - Pressure_{min})}{(Output_{max} - Output_{min})} + Output_{min} \quad (2.2)$$

2.3.3 Auto-Zero Calibration

A problem all pressure sensors have to combat is minimizing the error margin caused by external factors such as thermal effects. One may suspect that these effects may interfere with the ideal transfer function and the slope of the function. As stated by Honeywell [14] these residual errors only manifest as offset errors and do not cause a change in the slope itself. This is why Auto-Zero calibration can be used to realign the outputs closer to the ideal transfer function.

$$Offset_{average} = \frac{1}{n} \sum_{n} (Pressure_{measured} - Pressure_{expected}) \quad (2.3)$$

$$Pressure = Pressure_{measured} - Offset_{average} \quad (2.4)$$

2.3.4 Pressure Measurement

When measuring pressure, there are three types of pressure: Absolute, Gauge, and Differential.

- Absolute pressure: is defined as the pressure measured with respect to absolute zero pressure i.e. perfect vacuum.

- Gauge pressure: is the pressure measured relative to the atmospheric pressure.

- Differential pressure: is the difference in pressure between two points.

All three types are widely used, and they are often chosen based on the purpose of the measurement. Absolute pressure is applicable when the measurements should not be influenced by local atmospheric pressure. This can be used in weather stations to measure the difference in atmospheric pressure. In contrast to absolute pressure applications, gauge pressure is used when the local atmospheric pressure should be a relevant factor. These measurements are often used in medical devices, for instance, measuring blood pressure. Differential pressure is often applied in industrial systems to determine pipe leakage or clogs by detecting increase or decrease between two points.

2.4 General Data Protection Regulation

A key factor when creating the visualization platform containing user information and a system storing health data is GDPR. GDPR is a privacy and security regulation intended to protect EU citizens, focusing on people's right to control their data. GDPR introduces seven key principles and requirements that organizations must adhere to when handling personal data. As stated on their official website[1], the key principles are:

- Lawfulness, fairness, and transparency: Processing must be lawful, fair, and transparent to the data subject.

- Purpose limitation: You must process data for the legitimate purposes specified explicitly to the data subject when you collected it.

- Data minimization: You should collect and process only as much data as absolutely necessary for the purposes specified.

- Accuracy: You must keep personal data accurate and up to date.

- Storage limitation: You may only store personally identifying data for as long as necessary for the specified purpose.

- Integrity and confidentiality: Processing must be done in such a way as to ensure appropriate security, integrity, and confidentiality (e.g. by

1. GDPR [website] https://gdpr.eu/what-is-gdpr/ (accessed 30. June 2023)

using encryption).

- Accountability: The data controller is responsible for being able to demonstrate GDPR compliance with all of these principles.

2.4.1 Health Data

With the system handling pressure generated from patients, the system must handle storing and processing sensitive data. These types of data have special rules when it comes to how and why the data is processed [15]. As stated by the Europian Union (EU) [16] "Processing of personal data revealing racial or ethnic origin, political opinions, religious or philosophical beliefs, or trade union membership, and the processing of genetic data, biometric data for the purpose of uniquely identifying a natural person, data concerning health or data concerning a natural person's sex life or sexual orientation shall be prohibited.". Processing health data i.e. pressure generated from patients, is prohibited unless it follows one of the ten exceptions [16]. While data processing is allowed, the data and the system processing them require higher protection than regular personal data due to their sensitive nature. This makes handling health data challenging, as there are many different factors to consider.

2.5 Summary

This chapter outlined the motivation for creating this system and why it is necessary. It also describes the current state of the research in this area and how this system can help further the research. This chapter gives an overview of the relevant terms and terminology and how they work. As this thesis also builds on findings done in my Capstone project, this chapter also presents the relevant information from the project.

3

Requirements Analysis

This chapter discusses the overall project specifications and outlines the different requirements for the different parts of the system. Since the patient monitorization device is vastly different from the storage and visualization segments, the requirements are divided into monitoring requirements and platform requirements.

The project idea was forwarded from senior physician Bård Anders Forsdahl with the goal of improving the data foundation and the overall care for child tracheostomy patients. One of the main points of interest was if an IOT device could sample at 50HZ as this was a good sample rate to catch small discrepancies in breathing patterns. This was proven in my Capstone project and this thesis uses the findings there to build a system for gathering, storing, and visualizing data.

The main goal for this project is to create a system where the device can be used 24 hours a day while reliably storing data and have a user-friendly system for physicians to use. A key feature here is to create the system in a way that the GDPR laws are followed and the data is not directly linked to a patient. As these tasks are overall goals for the end solution, this thesis's main focus is to create the device and store the data following GDPR laws. To create a fully stand-alone system, the visualization also needs to be in place, and therefore the thesis focus on creating a POC solution for the website.

3.1 Monitor Requirements

As mentioned above, the device must measure at 50HZ, however, there are several different aspects that the device should aspire to accomplish. The main design goal for the devices is to have as minimal intrusion to the patients as possible. The key reason for prioritizing usability is the fact that to get realistic data, the patients must be in a natural setting. This allows the patient to behave as normally as possible as the device presents minimal limitations. In regards to the device functionality, there are several requirements that are more strict. Battery life and reliable storage are two functional requirements that are crucial for gathering data. Since the project goal is to have up to 24 hours of observation, the battery must provide sufficient power over longer periods of time while still ensuring that data is stored. The logic behind a minimum 24-hour battery life is two folded. Firstly having the device running for 24 hours allows for a " battery pair" solution, where two rechargeable batteries can be used in tandem, where one is in use while the other is recharging. The second motivation is to provide value for the physicians from the start, as 24 hours of monitoring still gives data usable for the physicians.

Given that the objective of this thesis is to gather pressure measurements from tracheostomy patients, it is vital that the pressure sensor used can handle the maximum pressure generated by humans. This usually ranges around ±8 $Kilopascal(k\text{PA})$ with some coughs maybe surpassing this. Therefore in this project, the sensor must handle at least ±10 $k\text{PA}$.

Both the usability and functionality requirements are vital for a good solution however, the two requirements have conflicting properties. Since more battery life and persistent storage often require more or bigger components that usually increases the overall size. Therefore some properties have higher prioritization during development. In this thesis, the functional requirements are prioritized over usability requirements as data is the foundation for the whole system and the size can be reduced over time while still getting results.

3.2 Platform Requirements

During the development of the Capstone thesis, a very simple system was created to display the results. This was only meant for internal use; however, it became clear that a viable solution should give the physicians a platform to see the results in an intuitive way. To create such as system, there is one vital problem that must be handled, and that is how can the system store the patient's health data while still upholding the GDPR standards. For the website, the main goal is to create a platform where physicians can see

the measurement, as well as provide calculation and filtration on the data. An important parameter is the repertory rate which measures the length of breathing in and out. Another important factor is average pressure, which is a simple task to calculate; however, the data not be affected by dirty data. Dirty data in this context is data that is not actually pressure readings, such as when the tube is changed or cleaned. This data will indicate that the patient had zero pressure generated, which obviously should not be considered.

3.3 Spesifications

The aspects presented in this chapter are important when considering the design and implementation of the different parts of the system. When considering the monitor device, the following aspects are vital for a device usable for research and patient care:

- Sample rate and pressure range: The device must measure at 50HZ and support a pressure range of ± 10 kPa.
- Battery life: The device must be able to run for over 24 hours
- Representative data: The device must be small enough to avoid limiting the patient to get the most realistic data.
- Usability: The device must also be integrated with the tracheostomy tube and it should be easy to equip and remove.

All these factors allow the device to achieve the desired attribute allowing physicians to get a further understanding of the patient's development. With the device not creating limitations for the user and running for longer periods of time, the device presents a unique insight to the patients development. These measurements need to be stored to provide access for the physicians. As the health data must follow GDPR laws, the storage system must:

- store the measurements while upholding the GDPR laws.
- provides access for the visualization platform to the measurements stored on the system
- provides authentication of the user, ensuring only authorized parties can access the data.

With the visualization platform being POC, the requirements are less strict, and

the overall goal is to show the potential of a visualization platform. Therefore the visualization platform must:

- visualize the data in a meaningful way
- provide authentication for the users on the platform
- provide tools for physicians to better analyze the data

3.4 Summary

This chapter presents the requirements for the different components of the system. The monitor device is split into two requirements groups, functional and non-functional requirements and they are based on the needs of the different users of the system. The storage system is intended to store the measurements and provide controlled access while still upholding the GDPR directives and the requirements reflect that. The visualization platform is a POC and the requirements are focused on providing functionality showing the potential of this platform.

4

Design

As stated in Section 1.1, the objective of this thesis is to create a solution for measuring, storing, and visualizing pressure readings from tracheostomy patients. Before going into details about each component and its design it is important to get an understanding of the context in which the system is intended to be used. The system is designed to be integrated with the existing routine for doctor-patient care of tracheostomy patients. Meaning the device and its platform is designed to be used as a natural part of this routine. This routine consists of many checkups with variable distances between one another and the overall goal is to give an easy-to-use solution for physicians to collect and visualize measurements.

When a patient is equipped with the device the idea is that the physician should be able to easily dismount the device and transfer and visualize the measurements. After the data is analyzed the device can be mounted back onto the patient for further data collection. When creating such a system there are important factors to consider, as stated in Chapter 3. Based on these specifications, this chapter outlines the design of the system covering the monitor device, storage system, and visualization platform.

4.1 System Overview

The system design consists of three main components: the measuring device, the storage system, and the visualization platform. All of these components work together to create a system that monitors the patient and allows physicians to see the measurements. There are two vastly different users of the service, and the technology required to create these services is in different domains. This makes the design and implementation of the services differ in many aspects, and the two services must also consider the user's needs. This is why the system objectives are different for the platform and the monitor device. When it comes to the monitor device, there are two categories of requirements targeted at different users. The functional requirements are mainly focused on the needs of the physicians, while the usability requirements are focused on the patient's needs. Therefore the monitor device must

- be able to measure samples at a minimum of 50HZ to capture the breathing pattern in detail.
- support persistent storage, allowing data to be retrieved even if the device fails or runs out of power.
- be capable of running for a minimum of 24 hours.

To also fulfill the usability requirements, the device must:

- have a size that is small enough that a child should be able to wear it.
- be created so that the patient can move as they normally would do, making minimal impact on their behavior.

While the project goal is to have a device measuring for longer periods of time, this thesis focuses on creating a functional system on which the project can build on top. With 24 hours of measurements, the device should provide a good starting point for the physicians to get useful data. The sample rate is a requirement set by the physician as it is a desired rate to get a detailed understanding of the patient state. Regarding the usability requirements, the size requirement is self-explanatory as the patient group is children. The user limitations are focused on movement as the user will probably be in a controlled environment. This is why the device is not focused on handling different weather conditions, especially in regard to water resistance and increasing calibration errors in colder environments.

The goals for the storage system are mostly based on providing a service for the physicians while complying with GDPR laws. As the visualization platform and

the storage system has to work together to make the system work as intended, the storage system also has requirements for both storage capabilities and features related to serving the visualization platform. Therefore the storage system must:

- Store data in acordance with the GDPR directives.
- Only allow the correct physician to access the measurements.
- Delete data when the patient is done with the device.[1]
- Only authenticated users should be able to retrieve data.

As seen in Figure 6.1, the Application Programming Interface (API) is a part of the storage system, and therefore all the requirements above are directed to both the API and how the data is stored. By following these requirements, the storage system should follow the laws and guidelines for storing health data to ensure users' rights are respected.

The website is the key factor for creating easy access to the measurements, and its requirements are based on highlighting the potential of a visualization platform. To showcase this, the features are focused on providing the core functionality for the platform:

- Display the measurements of a given patient in a simple-to-use interface.
- Support uploading measurements via the website(onsite data transfer).

As seen in Figure 6.1, the overall design for the system is composed of four components, who directly or indirectly communicate with each other. Each component has a specific purpose:

- **Monitor device:** The monitor device is responsible for gathering and storing the pressure generated by the patient. There are two versions of the monitor device, and each version transfers the data slightly differently:

    1. *Version 1:* This version supports onsite data transfer using the website interface.

    2. *Version 2:* Supports both transferring data directly to the API and onsite data transfer.

1. If the data should be used for purposes other than doctor-patient care, the patient must be informed of how the data is stored and the purpose of the research.

- **Storage system:** Consists of both the API and the physical storage, and its objective is to support longtime storage of different measurements gathered by the different devices. The API also have basic authentication to ensure that only authorized users/devices can read and write measurements to the file system.

- **Website:** The website is the interface for physicians, and it must handle both fetching and uploading data. When a patient is at a consultation, the physician can use the device via the website to see and transfer the latest measurements. In addition, the device can be used as an authentication device, meaning the physician now has the ability to see the entire history of this particular patient.

With the design described above, we can see that the system by itself does not store any personal information, yet it provides the correct readings to the physician. The main idea here is to build on the doctor-patient confidentiality, using the physicians as an identifier. Meaning only when a physician is with a patient the data can be retrieved. This allows the system to be fully anonymous while giving the physicians the information necessary.

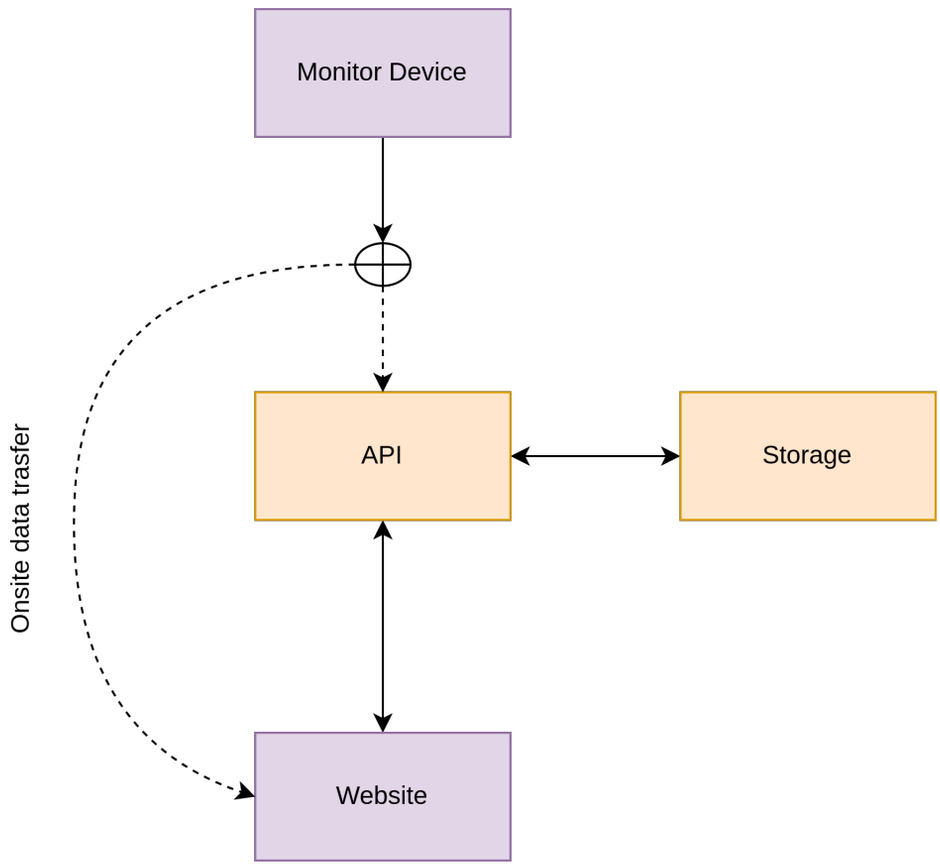

Figure 4.1: System architecture. The arrows represent communication between the components. The dotted arrows with the or symbol is signaling that the data can be transferred in either path. The storage system(colored in yellow) is one unit but is represented as two to show the data flow.

4.2 Monitor Device

As described above, there are two users with different needs, which are in focus when designing the device. To solve the functional requirements of the physician, the device must be able to measure and store pressure samples reliably. To accomplish these desired features, the device is designed as seen in Figure 4.2. The device itself consists of three parts: an SD card adapter, a microcontroller, and a pressure sensor. This simple structure allows the device to measure data from the pressure sensor and store it on the SD card via the microcontroller. There are several reasons for transferring data via the microcontroller. The main reason is that both the SD card reader and the pressure sensors are peripheral devices meaning they are not designed to

handle complex tasks such as orchestrating communication protocols. Another benefit is avoiding post-processing. Since the data transmitted from most pressure sensors are not the pressure itself but the resistance measured by the device. At some point, the data must be transformed using the transfer function (2.1) to get output pressure. In this way, the data is stored as the converted pressure in the desired form, saving space and avoiding post-processing. As modern computer process calculation at the Gigahertz (GHZ) level having the microcontroller perform the conversion should not affect the battery life noticeably.

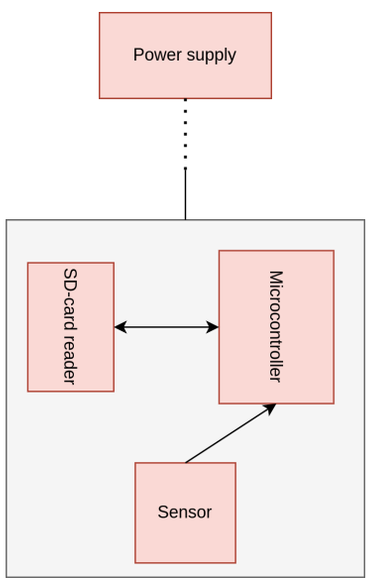

Figure 4.2: The design for the device capable of measuring and storing data. The device consists of three components needing to communicate with each other and an external power supply. The sensor transfer readings to the microcontroller, which stores the readings on the SD card. This design is used for both versions

As previously described, the monitor device comes in two versions, and they support different ways of transferring data to the remote storage. While they support different transfer protocols, the overall design is not different. The main difference is the components used and the software running on the microcontroller. Version 1 uses a simpler microcontroller which is designed for peripheral communication, while version 2 uses a more complex controller, which supports WI-Fi. The motivation for having a device that can transfer data via the internet is two folded. First, data is stored in multiple ways, meaning if the SD card is corrupted or damaged, the data is not completely lost. This is not an unlikely situation as the overall goal is for the patients to use this in everyday activity. Secondly, by having measurements sent continuously, the

physician can see and process the data before meeting the patient. This allows the physician to form an opinion before meeting the patient, allowing the appointment to be used more efficiently.

4.2.1 Transferring Data

Both versions have to be able to offload their measurements to the storage system at some point. Most likely will, the majority or all of the data be stored on the SD card for both versions. This is due to the fact that data is sampled at 50HZ, generating 200 bytes of data each second. The device is not designed to send this amount of data remotely, so much of the transfer is done onsite(via the website). When transferring data onsite, the SD card is removed from the device and inserted into the physician's computer. From there, the physician can upload the data; however, to be able to store the data at the correct location, the SD card must contain metadata that can be used by the website.

While transferring 200 bytes per second is a trivial task for modern computers, this is not the case for small microcontrollers, as several factors affect these microcontrollers that do not apply to modern computers. Microcontrollers are often bounded by a limited power supply meaning the controller cannot uncritically use resources like a computer connected to a power supply. The device is also minuscule compared to modern computers and thus supports fewer features, especially when it comes to multiprocessing. This is why having the device sending data at this rate is a poor use of limited resources, as using WI-FI demands a lot of power compared to peripheral communication. Even if the data is sent in a few large requests, the amount of data will still be so large that it will demand a lot of power. So the main goal of version 2 is to send a portion of the data to the storage system to indicate the patient's current state. Another factor is that the device does not have internet access at any given time, so it is designed using an opportunistic transfer model. The reason that the device does not have continuous internet access is based on several factors. The first and most important is the fact that continuous data transfer gives little reward for the physicians compared to the costs. To support an internet connection at all times, the device must utilize some sort of cellular network, which requires the device to be fitted with a sim card, and it will cost money to use the network. Adding these extra expenses is not ideal, and the physicians will realistically not notice if the data is transferred continuously or in bulks. This is why the devices use WI-FI as it is not adding any continuous expenses, and the price difference between a controller supporting WI-FI is negligible compared to a controller only supporting peripheral communication. With the controller using WI-FI, the device can be paired with the patient's home network allowing for opportunistic transfer, i.e., when the patient is home.

4.3 Backend

The backend system is a vital part of the solution, and its objective is to provide the necessary services for both the website and remote data transfer from devices. Since the main traffic will come from the website, most features are directly related to serving the website. For a website to be able to have users and serve data, it must communicate with the backend. How these services exchange data through the internet varies. For this system, a Representational State Transfer (REST) API is intended to be the connection between the backend and the visualization platform and devices. The API exposes different predefined operations available for different resources accessed via a Uniform Resource Identifier (URI) or endpoint. The endpoints allow different methods to get, create, update, and delete resources via GET, POST, PUT, or DELETE requests. Different features, such as creating or updating a user, can be done with these operations. Since different resources should only be accessible for some users, the API must handle authentication and access control. This ensures that only the intended users should be able to access the resource.

4.3.1 Database Models

As mentioned above, the backend needs two database models to provide our desired features. First is the user model, encapsulating the physicians using the website. Secondly is the device model, which is where all the measurements are stored. Figure 4.3 shows that the device model contains a unique identifier and the measurements. As seen in Figure 4.3, the device does not store any identifiable information about the patient, ensuring that the system omits the GDPR laws. The reason is the fact that when the storage is fully anonymous, the GDPR law stated in 2.4.1 does not apply [17].

A natural problem that occurs when not storing personal identifiers is how to provide access for the physicians to the correct measurements. This is why the user model stores a field called *device_ids*, which dictates which device the physician has access to. This allows the measurement device to be used as an authentication mechanism, granting access to the measurements. This ensures that the physicians see the measurements of the patient having the device as well as the entire history. The database model also stores identifiable information about the physicians, as we want some sort of authorization on the website.

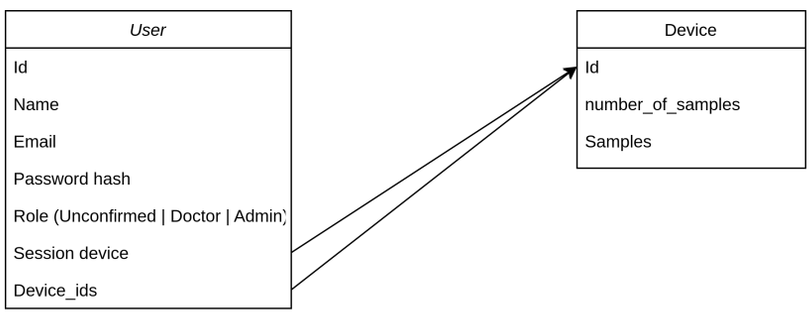

Figure 4.3: The database models and their relation, with the user model having two fields (*session_device and device_ids*) containing device Identifier (ID) (s)

4.3.2 Data access

For long-term storage and accessibility for users, the system has a database storing all the various data. Many of these resources must be accessible for both the device and the visualization platform. To accomplish this, an API serves as a middleman, allowing and controlling access and modification to different resources. Since data should only be accessible to specific users, the API must also provide authentication. As seen in Figure 4.3 all users must have an email and password associated with them, allowing password-based authentication to be utilized. The API must also provide logic allowing the data access design seen in Figure 4.3 to be possible. This means that the API must have support for allowing a link between a device and a user to be possible.

Another case that the API must handle is how the device can remotely transfer data. Since the service must know the device id to store it in the database, all devices must be paired before use. This device pairing depends on all components to work together to pair a new device successfully. As stated, the backend must know the device id to store the measurements in the database. Therefore the device must store its id before running. The physician does this via the website, where the new id is inserted into the controller. The website generates a new unique id, and after the device has received the new id, the backend can be notified that a new device is registered. Since the device should be able to use this id in multiple sessions, the id must be stored on persistent storage. When the device has access to its id, it can provide the identification to the API, allowing the data to be stored correctly in the database.

User Verification

Another vital part of the system is validating the users. This is done through email verification, and once the user is confirmed, the user gains access to the different features. There are several reasons for demanding email verification. The first is to guarantee that the system can always contact the user with important information or services, e.g., resting password. The second is that at a later stage, it would be natural part to have a deeper verification through different health institutions. For example, could a hospital use this system, and all emails belonging to this institution are automatically confirmed. Having an existing verification system would improve the transition to a stricter verification policy.

4.4 Visualization Platform

The visualization platform is a web-based solution that allows physicians to gain easy access to the measurements. The main reason for choosing a website is the fact that it does not require extra software. By creating a website, the physician can access the data from a computer anywhere in the world. This provides a flexible solution without introducing any dependencies, such as requiring a specific program. With the device described above, all the recent measurements are stored on an SD card, making it possible to visualize the data using a simple Python program. The motivation for choosing a website over providing a simple program running locally on the user's computer is based on user experience and security. Considering all the different environments and computer knowledge expected to download and run a Python program can create many problems. All from having the correct dependencies to Operating System (OS)-specific details. An obvious solution to these kinds of problems is Docker providing a predefined environment for the program to run in. This again expects too much computer experience from the end user. Another aspect is security and the process required to allow an executable to run on a hospital computer. All these problems make a website a more desired solution; however, it also creates more features that the backend must provide, which is explained in more detail in section 4.3

4.4.1 Website Design

With a web-based solution, the idea is to provide a minimal website where usability is in focus. To avoid a too complex interface, the website focuses on the main feature, which is visualization. This is achieved by having the website dynamically loading based on the state of the user following the logic stated

in Figure 4.4. As described in the previous chapter, the data should only be available to specific users, and the website must ensure that this property is upheld. By having the website dynamically provide the correct content, the user avoids having to use the time to find and understand which steps are necessary for gaining access to the data. It is worth noting that this design works as the website is simple and single purpose, and it only has two main states to consider. With a more complex website handling more features, this design may not be optimal.

As seen in Figure 4.4, the website must load or redirect the user to the correct content or page based on their state. Before the data can be accessed, the user must be authenticated. This page builds on the logic described in the previous section, whereas the physician can only gain access to the data by showing ownership of the device. This page should provide all steps in an intuitive way so that the physician can easily see and upload the new measurements.

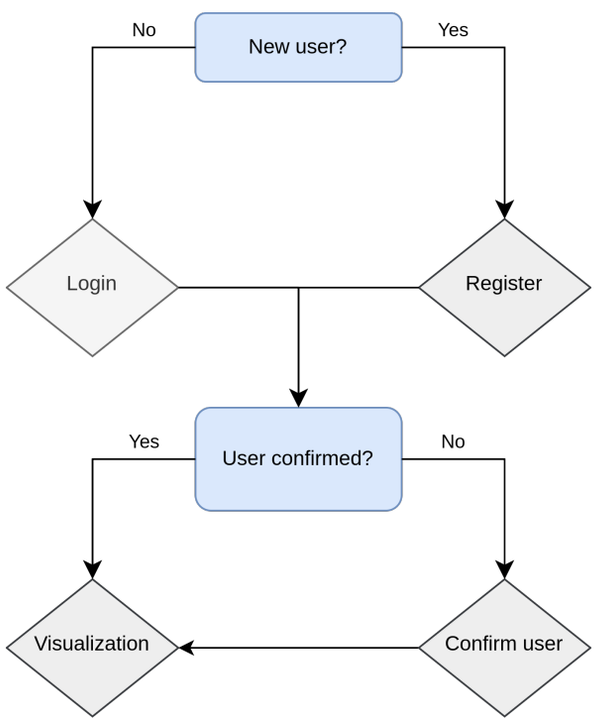

Figure 4.4: Shows the intended design of the home page and what content should be displayed in different states.

4.5 System Design and Integration

As mentioned in the introduction of this chapter, the design of the system is to be a part of the routine between the patient and physician. With all these individual components, it's important to get an overview of the usage of the different components and how they are designed to be used. When a physician decides to use this device on a patient, he logs into the website and pairs the device to his account. When this is done, the device is turned on, and data is collected. After a period, the patient returns, and the physician verifies that he is, in fact, having the device through the website. When the verification is complete, the physician can see the results. With results displayed on the website, the physician can analyze the newest measurements and see the patient's history. When the physician is finished analyzing, the data can be uploaded to the permanent storage.

4.6 Summary

This chapter presents the design of the system based on the requirements stated in Chapter 3. The design presented shows how each part of the system is designed to work and how they work together to create the system. Lastly, a description of how the system is intended to be used in the existing doctor-patient routine is provided, to show how the system can work in practice.

5

Implementation

This chapter describes the implementation of the system based on the design outlined in Chapter 4. Three components solve different aspects of the problem and span various domains of computer science. The monitor device is an IOT device who are meant to be small and efficient. These controllers are focused on low power usage while providing simple peripheral communication protocols to complex tasks such as providing WIFI. IOT devices usually consist of multiple single-purpose devices which need to communicate with a more complex device, often microcontrollers. Creating a larger device unifying all the different components into a single circuit requires a basic understanding of electrical engineering. With the development of an IOT device spanning multiple disciplines of engineering, the implementation and development are vastly different than the other two components.

While the monitor device is focused on limiting the size and power usage, the visualization platform with the subsequent backend is on the other side of the spectrum. These components are focused on serving the frontend and backend service as fast and to as many users as possible. Proving such services ideally available 24 hours a day requires a completely different approach than for IOT devices. While all computer system tries to be as efficient as possible, thereby limiting power usage, the overall objective is where the frontend and backend differ from the monitor device. The frontend and backend want to be available all the time and trade off power usage to always be ready to serve a request as fast as possible. The monitor device, on the other hand, runs on a limited power source and has to make compromises between performance and power

usage.

5.1 Monitor Device

As described in Section 3.1, the device should be able to measure reliably and store pressure readings while focusing on creating a small and power-efficient solution. The reason the device needs to be as small and efficient as possible is the fact that this device is meant to be integrated with a tracheostomy tube. For a patient having a device around the neck is something that can be unpleasant, and the device should strive to minimize this effect.

5.1.1 Hardware Specification

An important part of the development of the monitor device consists of finding suitable device components. As seen in Figure 4.2 the device consists of three components: the SD card adapter, the pressure sensor, and the microcontroller. These components have to have specific capabilities to fulfill all the different requirements stated in Section 3.3. As these requirements only cover how the end product should behave, there are multiple implementation-specific requirements needed to make the component work together. Both the functional and implementation requirements are as follows:

- **Pressure Sensor:**
    - Sample rate of 50HZ
    - Pressure range greater than ± 10 kPa
    - Should run on a power supply between 3 and 4 volts
- **SD card reader**
    - Enough storage space to store all the data gathered during one run
    - Should run on a power supply between 3 and 4 volts
- **Microcontroller:**
    - The microcontroller must support and have enough pins for the transfer protocol used by the peripheral devices

- Should run on a power supply between 3 and 4 volts

The reason that all components should be able to run on less than 4 volts is the fact that most common and affordable batteries are usually around 1.5 volts. With two 1.5 connected in series, the output voltage will be around 3-3.3 volts making the device compatible with two AA or AAA alkaline batteries. Having the components also withstanding between 3 and 4 volts allows for different battery composites, such as lithium, to be utilized, allowing higher voltage over time. With between 3 and 4 volts, rechargeable batteries also become usable without any modification, allowing batteries to be used in pair as described in Section 3.1

5.1.2 Proof of Concept

As previously described, the monitor device consists of two versions. The first version is designed to store all the data locally on the SD card. The second version extends this feature and adds the possibility to send data remotely. Since all functionality of version 1 can be seen as a subset of all functionality in version 2, creating a POC for version 2 will show the functionality for both versions. Therefore the development and the hardware testing focused on hardware supporting both peripheral communication and WI-FI.

The goal of the first device was to show that the device could perform measurements at 50HZ while sending data to the API. The size of the components was not as important at this stage as the functionality of the different devices was the focal point. Therefore the device consists of the components seen in Table 5.1

Device	
Microcontroller	ESP32-WROOM-32U
Pressure sensor	ABPMRRV015PD2A3
SD card reader	SPI Micro SD card adapter

Table 5.1: List of the components used in the POC device. The SD card reader's exact model was not identifiable; however, any 3-5 volts breakout micro card adapter is usable in this phase

The ESP32-WROOM-32U microcontroller provides support for both the WI-FI and the SPI communication protocols, which are used by the other devices to transfer data. The device is created by Espressif System and is a part of the ESP-32 series. To build a program capable of running on a microcontroller, a special

framework or library is required. The reason for this is that the microcontroller is an embedded system specialized to run a single program. This program or firmware must handle everything from OS-specific tasks such as booting scheduling and so on. By using a IOT development framework, all the tasks of making the controller run properly are handled, and the user-defined program can simply be uploaded as a part of the system. These frameworks provide a set of tools or Software Development Kit (SDK) allowing different functionalities to be accessible for the developer and more complex programs to be built on top of the firmware.

A natural choice for developing on the ESP controllers is the ESP-IDF development[1] framework, as this is an open-source project created by Espressif. This framework is also created in C, and the program has to be written in C as well. This allows for an efficient program with a high degree of control. As stated, C provides the opportunity to write a high-performing program; however, it also requires longer development time compared to other high-level programming languages. Since the main goal is to have a device with low power consumption, the complexity of the development is worth the potential benefits.

Looking at the other peripheral devices, the SD card adapter is a component allowing Micro SD cards to be accessible for the microcontroller. The device uses communication via the SPI protocol allowing the controller to access and modify content on the SD card. As stated, this device is a POC device, and the components have some properties which are not optimal in a final version. The SD card reader expects 5 volts power supply which is more than necessary as SD cards use a 3.3 power supply. The component regulates the 5-volt supply down to 3.3 volts; however, this is only necessary when a 5-volt power supply is used. Since the intended voltage on the battery should be between 3 and 4 volts, this component is only relevant in this development stage.

The pressure sensor, on the other hand, has all the attributes needed in a final version. It is a small pressure sensor with a dimension of $12.75 \times 11.80 \times 8.25\ mm$ after removing access material on the mounting legs. It has a pressure range of $\pm 103 kPa$, and it uses SPI to transfer data. The sensor can also provide a sample rate of 50HZ, as this was shown in my Capstone thesis.

When developing the first iteration, a breadboard was used as this allows components to be easily added and removed from the circuit. When all components were connected, the device could easily be tested and modified to prototype different possibilities similar to Figure 5.8. With all the components connected, the software must handle measuring and storing data on the SD card while transferring data to the API. Since my Capstone thesis tested that sending data

1. ESP-IDF[Github] https://github.com/espressif/esp-idf

while measuring at 50HZ was possible, the software had to extend this feature to read and store to and from the SD card. The system works by having a producer gather the data, and when it wants the data to be stored remotely, it sends a message via the OS to another process to start sending the data. This was the starting point for this thesis; however, the design was changed to optimize for power usage and handle variable internet connection.

The new version starts two processes at startup, one running the measurements and storage, while the other is dedicated to sending data to the API. The difference here is that in the new design, the second process always tries to connect to the WI-FI and subsequently send measurements. The two processes have a new location to share the measurements since the SD card is added to the device. The new design now allows the two processes to perform their task independently by having the process responsible for sending remote data probing for WI-FI at a scaling interval. When a connection is established, the process retrieves the data from the SD card and sends it to API. The data is sent to a dummy endpoint during development, and it was only used to show that the measurements could be sent to the remote API.

To ensure the two processes read and write to different files, all files are stored with timestamps as filenames. This allows the SD card to be sorted based on the name, and the oldest file can be read and sent to the API. When the data is sent, the file is deleted, and the process checks for more data to be stored on the API. This is done by checking if there is more than one file on the SD card, as only one file indicates that the measuring process is still writing to that file. The reason the process should wait for the measuring process to be done is that it creates unnecessary work as it would be multiple smaller partial updates instead of one large. As stated in 4.2.1, the device does send bulk updates; however waiting for enough measurements while keeping the WI-FI module on is not optimal.

When the process has no more work or the internet connection is lost, the process goes into a probing mode waiting for available measurements or a WI-Fi connection. This probing works with a scaling timeout based on the number of failed probes. When the process unsuccessfully probes for either work or an internet connection, the timeout scales with the number of failed attempts capping at 30 minutes. To further optimize the power usage, the device also turns all the WI-Fi modules completely off when entering a timeout, as this component uses a large amount of power compared to the other tasks.

5.1.3 Design Itteration

During the development of the device, the remote data storage system became an increasingly larger part of the device, and it started taking more and more resources to become viable. By performing a simple test using Keweisi KWSV20, a simple device measuring the power usage of a Universal Serial Bus (USB)-connected device, it became clear that using WI-FI would become a very expensive operation to have. The power consumption increased by eight times by just performing a simple operation using WI-FI compared to peripheral communication. Based on this observation and the fact that updates during usage are not a crucial feature, the prototyping for a usable device drops the remote update feature. With WI-FI not being a part of the prototyping, the device can be created smaller and use less electricity to run. These are the most important features; the device still provides the necessary attributes.

5.2 Version 1

As the remote data feature was removed from the device, the first prototype can be created. The prototyping process for creating the device consists of four stages: finding suitable components, creating the schematics, creating the circuit board, and creating the casing.

5.2.1 Components

As the components used in the POC stages are mostly chosen based on their flexibility and are, for the most part, not intended to be used in a prototype, new components have to be used. The overall goal of the device is to be as small as possible; therefore, a smaller and more specified microcontroller is necessary. Since the controller must support peripheral communication with both devices, the microcontroller must support SPI with two slave select lines. The reason is that almost all SD card readers use SPI over I^2C as SPI is faster, and the pressure sensor already uses SPI [8, 9]. The ESP32-S2-mini microcontroller has all the attributes with the dimension of 25.4×34.2 *mm* and SPI support for multiple devices.

Due to unexpected problems with the delivery of the part, another microcontroller had to be used. The QT Py SAMD21 from Adafruit is another microcontroller that is suitable as it provides SPI with multiple devices and is smaller than the ESP32-S2-mini. The only problem with using the controller is that it does not support the esp-idf framework making the existing codebase unusable. Since this controller supports either Arduion or CircuitPython the new program

could be rewritten faster as both frameworks are written in a more high-level programming language. Since this microcontroller also supports I^{2}C, another slightly different pressure sensor using I^{2}C was preferred over the existing one using SPI. The theory behind this change was that it would simplify the circuit and software. This is due to the fact that SPI uses four lines compared to the two used in I^{2}C. Another factor is that the data and clock lines in multi-slave SPI communication are shared, and this creates more complex wiring on the circuit board. The software would also be more complex as the SPI bus has to be shared between the SD card and the pressure sensor making synchronization more complex compared to having the sensor using I^{2}C.

5.2.2 Schematic

With the new microcontroller and the peripheral devices using different communication protocols, the circuit must be created following the specifications of the different devices. All the components are created to be used in a predefined matter which is described in a datasheet provided by the manufacturer. This technical documentation specifies multiple important factors, including the designed responsibilities of the different pins. If the documentation is not followed, the consequences can range from unexpected behavior to breaking the device entirely.

By looking at the datasheets for the different components, a schematic can be constructed. Figure 5.1 show how all the components are connected following the specification of the different devices. It is worth noting that the schematics are different from how the circuit board will look, as the goal of the schematic is to highlight the connection between the components. This means that the optimal path for the wiring is not the essence of the schematics, it is to make the position and connection between the devices easy to see and understand.

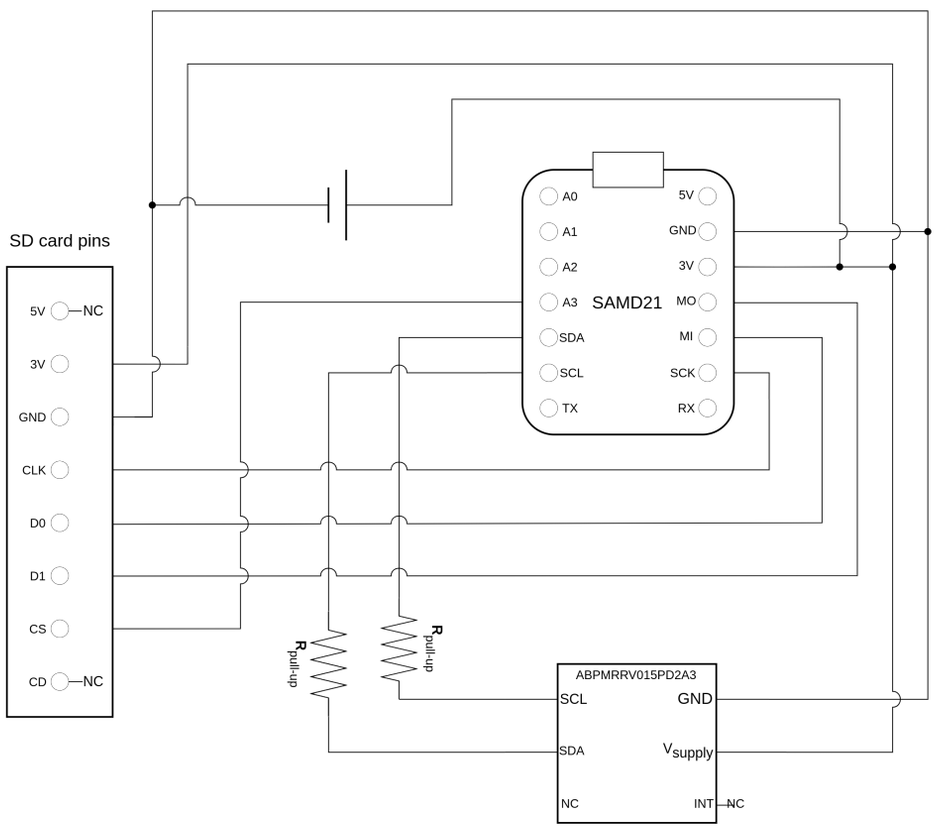

Figure 5.1: Schematics for the device prototype connected to a battery. The schematic follows the standard convention for indicating different attributes on the circuit. The dots are joints i.e. the wires are connected, The arcs are wire jumps indicating that they are not connected. The DC power supply i.e. the battery, is indicated with two parallel lines where the longest is power and the shortest is ground.

Starting with the pressure sensor, the sensor has six pins provided by the manufacturer. Based on the datasheet [13], the manufacturer defines five pins enabling the device to measure and transfer data using I^2C. Pin 1,2,3,5 and 6 are responsible for connecting to the Ground (GND), voltage supply, interrupt, SDA, and SCL source as seen in Figure 5.1. As the microcontroller is the master device, all the data lines SCL and SDA are connected to their respective I^2C pins. The interrupt line is not used in this circuit as it is not intended to be used; as the master-slave communication used in this device, does not need the ability for the slave to send an interrupt to the master. This can easily be added without affecting the existing circuit, however, for this to be relevant, the software must handle the interrupt and its source in a meaningful way. The interrupt flag signals that there is something wrong with the slave and that the master must start a process to handle the problem. Handling this situation

is not important for prototyping as it is very rare, and it is a time-consuming process to make it work properly.

An important part of the communication between the microcontroller and the pressure sensor is the two pull-up resistors. These are integral to the I^2C protocol for several reasons. First, it ensures that only intended 1 and 0's are sent between the components called logical high and low. The reason this is necessary is that the I^2C is an open-drain architecture, meaning that the intended behavior is that the line is "high" in the idle state [18, 19]. This logical high is defined by the supply voltage powering the I^2C communication and different voltage supply gives different logical highs. However, the I^2C is created to work with different power supplies so moving different batteries i.e. changing the voltage of the power supply, does not affect the communication. The reason for having the idle state too high is to ensure that components can only pull the resistor down, avoiding bus contention. A contention can occur when one slave creates a high while another creates a low, effectively canceling each other out.

Another key attribute provided by the resistors is that it avoids noise affecting communication. All pins are affected by Electromagnetic Interference (EMI), causing small fluctuations between the low and high states resulting in unexpected behavior. Since the resistors always pull the line high small fluctuation would not be misinterpreted by other devices as a logical 0 can only be created by providing a short to the ground. While these fluctuations are mostly not a problem as most microcontrollers provide internal pull-up resistors, these are usually not strong enough to be used in I^2C. Therefore the circuit provides external pull-up resistors with a high enough resistance for the sensor to send and interpret data. As seen in the technical documentation [13], the sensor expects a resistor with a minimum 1 $k\Omega$ of resistance. Therefore, the circuit has a 2.5$k\Omega$ resistor on the SDA and SCL line as the resistance has to exceed 1 $k\Omega$. While the resistor has to be high enough it also cannot be too high as the I^2C protocol uses rise time, the time used by the signal to move from low to high or high to low. A too-high resistance will make the rise time too slow, resulting in communication problems. Consider the rising edge using too long time to reach the acceptable level i.e. 70% of the voltage supply, and then it is pulled low, the logical-high will never be read [18].

The maximum and minimum resistance can be calculated [20], however, for circuits using microcontrollers with a 3.3-volt power supply, a resistor between 2 and 10 $k\Omega$ usually work. As 1$k\Omega$ is the minimum value stated in the datasheet, the maximum value is the only uncertain value. As these values usually range between 2 and 10 $k\Omega$, a 2.5 $k\Omega$ was chosen as a similar experiment showed that a 3.6 $k\Omega$ resistor worked with 3.6-volt power supply [21]. When testing the circuit with 2.5$k\Omega$ resistance, the pressure data was sent correctly, showing

that the resistance was not too high.

The key factor for this device is size, and to optimize the space, the SD card reader is laying over the microcontroller and sensor. This is not reflected in the schematics due to the fact that it makes the schema very unreadable. With the SD card reader being the largest component, having the components underneath it gives a more compact device. The device becomes slightly larger in the z-direction however, it becomes more compact in the x and y direction. This is due to the fact that the microcontroller and pressure sensor almost fit entirely under the SD card, essentially making the device almost as large as the SD card adapter in the x and y direction.

5.2.3 Circuit Board

As previously stated, the schematics are used to show how the components are placed and electrically connected; however, to optimize the size, the wiring on the board is somewhat different. As seen in Figure 5.2, it is a clear difference in how the circuit is made vs the schematics.

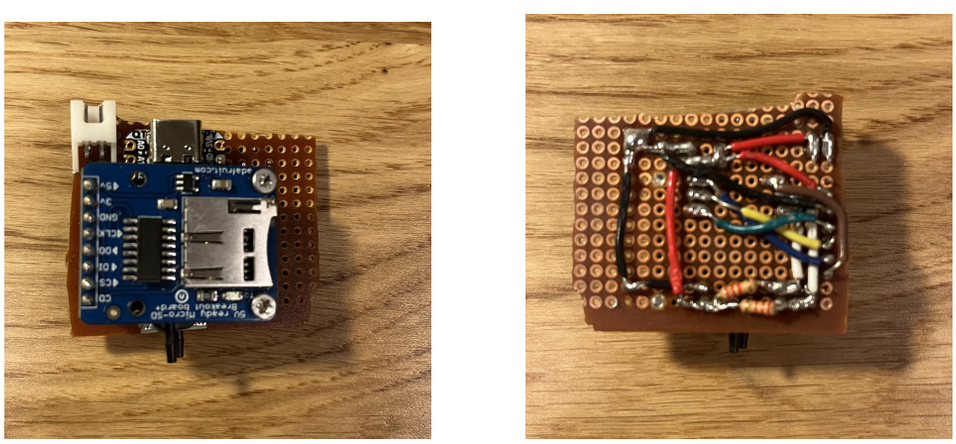

Figure 5.2: The circuit board based on the schematic 5.1 for version 1. The leftmost image shows the device from a top-down view with the SD card laying over the microcontroller and pressure sensor. The rightmost image shows a bottom-up view showing the wiring of the device.

The circuit board is mainly focused on minimizing the space, and therefore wires have to cross each other. While this is shown in the schematics there are some areas where multiple wires stack over each other as there is no room to go around. Another practical difference between the schematics and the circuit board is the resistors seen in the bottom left in the bottom-up view of Figure 5.2. These resistors have to connect to the voltage supply as well as the

pins. While this is obvious with knowledge of the behavior of pull-up resistors it is something that requires understanding of the component to implement correctly. As previously stated the SD card is intended to be laying over the microcontroller and the pressure sensor and to accomplish this the circuit is fitted with female header pins. These pins are used to allow components to be a part of the circuit without the component needing to be soldered to the circuit board. It also gives the SD card adapter a small elevation allowing the pressure sensor to fit under it. The pins are not seen in the schematic however they are placed under all the SD card pins as seen in the schematic.

The circuit is also fitted with an adapter for the power supply as it should be a simple process to remove the battery. As a starting point, the battery is in a separate 2×AAA battery box with a power switch extended to have the male connector to the circuit's female adapter. While the wires are approximately 10 cm in length there is no problem extending this to allow the battery pack to be further.

5.2.4 Case

As the device is intended to be used by a patient in an open environment, the device cannot be exposed without any protection. Therefore the device needs a casing as it protects the electronics from wear and tear. The casing also works as a protective layer between the patient and the electronics minimizing the risk of direct contact with the electronics. With the casing, the device can also be fastened on the Velcro strap around the patient's neck allowing the device to be integrated with the tracheostomy tube.

The casing was manufactured using 3D printing, which offers several advantages for both rapid prototyping of customized objects and the creation of relatively robust products. 3D printing is a manufacturing process that transforms a digital model into a physical object. It achieves this by using a 3D printer, which constructs the object layer by layer. To build each layer, the printer heats a thermoplastic filament and extrudes it through a nozzle. The printer precisely moves the nozzle to create each layer, gradually forming the object.

To create the casing for the device the model was created using OnShape a platform for creating Computer Aided Design (CAD) models. These 3D models can then be sliced i.e. converting it into a set of instructions that can then be read and executed by the 3D printer. Another factor when 3D printing is choosing the filament as there are different types providing different attributes. For this case, Polyethylene Terephthalate Glycol (PETG) filament was used as it is stronger and more durable than the more commonly used Polylactic

Acid (PLA). PETG also provides good layer adhesion which is good as this will improve the case's protective ability i.e. hold unwanted substances away from the circuit board.

As seen in Figure 5.3 the casing is specialized to fit the circuit board. The front is created with one tube intended to be inserted into the tube from the tracheostomy device. This tube is created similarly to the tube on the tracheostomy device and a tube fitting the tracheostomy device will also fit the tube in the case. The front wall is also created with a small opening next to the tube intended to be used by the second port on the pressure sensor. As the sensor is a differential pressure sensor it is important that it has a clear opening as this is used as a reference point. The back wall is created with a square opening, allowing the battery to be connected to the device. As the wiring is not perfectly flat, the case is also constructed with a square hole in the middle to avoid tearing on the wiring. The elevated side inside is also used as a mounting point intended to be fastened by screws. This can be seen behind the right wall two holes are created inside the case allowing the screws to be placed. This ensures that the device does not move when placed on the patient's neck.

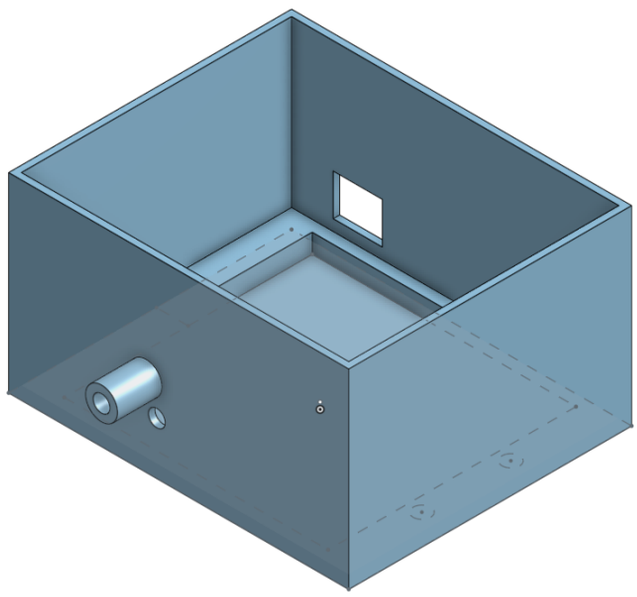

Figure 5.3: 3D model for the first case, designed based on the device seen in Figure 5.2. The case provides an opening in the back to allow the device to be connected to the power supply. It is also equipped with a tube with the same dimension as the tracheostomy tube. Lastly it is fitter with holes on the left most side, where the device can be mounted.

5.3 Version 2

When creating the device there are many different factors to consider and finding the most suitable solution is an iterative process. During the implementation of the first prototype, the experience of creating it gave the idea for optimizing different parts of the device. While version 2 does not introduce new components it optimizes the circuit and the layout of the different components. The reason for creating a new version is that after creating the first device it was discovered that the shape of the device was not optimal. Why the shape is sub-optimal it is important to understand in which direction an object presents the most resistance. Defining the x-direction as the direction around the neck and the y-direction as the direction upwards of the neck; the device is preferably larger in the x-direction than the y-direction. This is due to the fact that the device does not block any movement in the x-axis as the closes thing to blocking in this direction is the chin bone. In the y-direction, however, the device becomes more of an obstruction as our entire head can

collide with the device. Therefore the first circuit board was not optimal as it was more of a quadratic shape. With the dimension $41,5 \times 39,0 \times 30$ *mm* it became noticeable when using.

5.3.1 Schematic

Electrically the new schematic is equal i.e. there are not introduced any new wires or resistors. The difference is how and where the components are placed. As seen in Figure 5.4 the SD card reader is moved and rotated. As this is the largest component previously extending the size in the y-direction, it now extends the device in the x-direction. This simple movement gives the desired characteristics as the device becomes longer in the x-direction and shorter in the y-direction, giving the device a more rectangular shape. The microcontroller is rotated upwards for two reasons, first, the Universal Serial Bus type C (USB-C) port cannot face the SD card mounting point or the pressure sensor as this will effectively remove access to the device. Secondly, it allows for easier wiring as the voltage supply and ground can be easily implemented and it also minimizes the number of wire jumps.

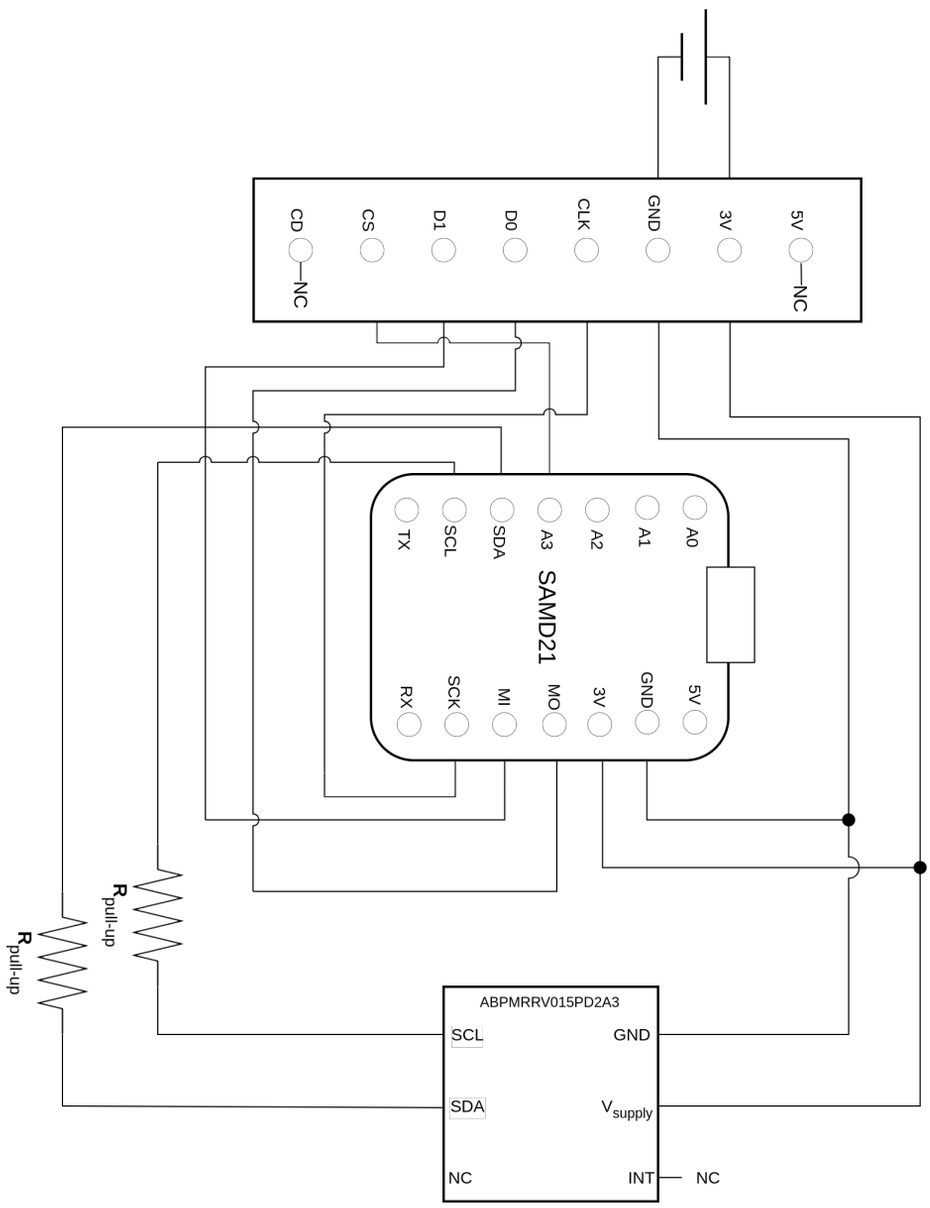

Figure 5.4: Schematic for version 2

5.3.2 Circuit Board

With the new schematic, the circuit board can be created, similarly to version 1. The new circuit is more compact and utilizes the space more effectively compared to version 1. This is a combination of both improving the design and implementation. With experience from creating the first circuit, better decisions

were made during implementation allowing for a more space-efficient circuit. With less space, the implementation also becomes more complex and the soldering does become more of a challenge.

As seen in Figure 5.5 the device is effectively the same size as the SD card in the x and y direction, improving the size significantly. With the new device having a dimension of $47.5 \times 29.75 \times 27$ *mm* the new version has minimized the size in y-direction by almost 10 millimeters. This is a size decrease of almost 10 percent, giving more flexibility to the end user.

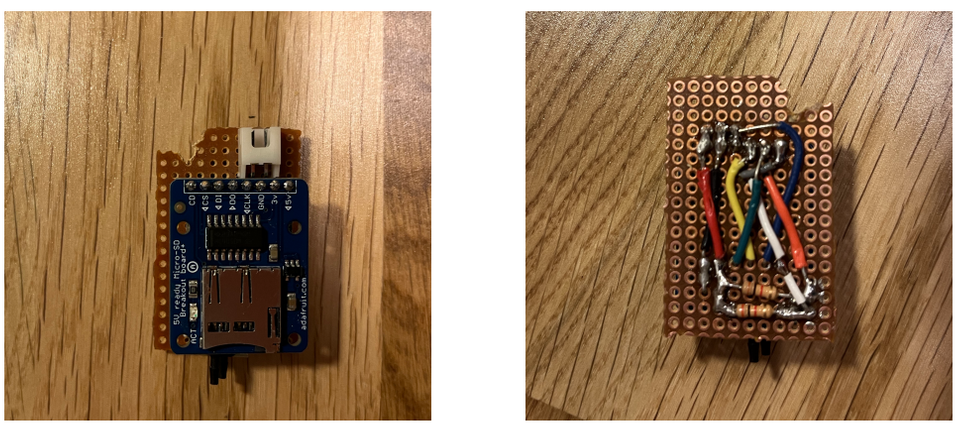

Figure 5.5: New device circuit board, with the dimension: $47.5 \times 29.75 \times 27$ *mm*

5.3.3 Case

With the new shape of the circuit board, the case needs to be redesigned. The case utilizes the same features as the previous version, with some small modifications. The height of the case is lowered to fit the device better, and the case only uses one mounting point to screw the board in place. The new case is also equipped with a lid to finish to make the case completely encapsulated. This lid is based on a snap-fit design allowing the lid to be fastened to the case without any moving parts. This design utilizes the flexibility of the plastic filament for the lid to be snapped in place by adding some pressure.

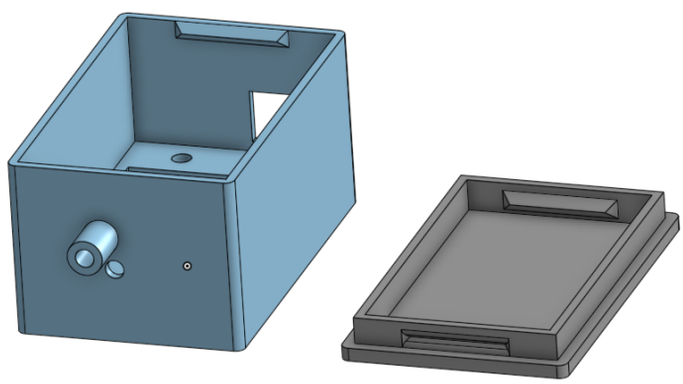

Figure 5.6: The new design for the case with the snap-fit lid.

A vital part of the device design is access to the micro SD card inside the SD card adapter. As seen in Figure 5.6 this is not possible as it is not enough space between the reader and the case wall. This layout is done as the intended usage is to extract the entire adapter when the micro SD card is needed. This is done for two reasons, it minimizes the size and it improves the usability. With the case being 3 Centimeter (cm) wide on the inside, it is not an easy process to insert the micro SD card into the adapter. In contrast, lifting the adapter is relatively easy, as the circuit board was already fitted with female header pins.

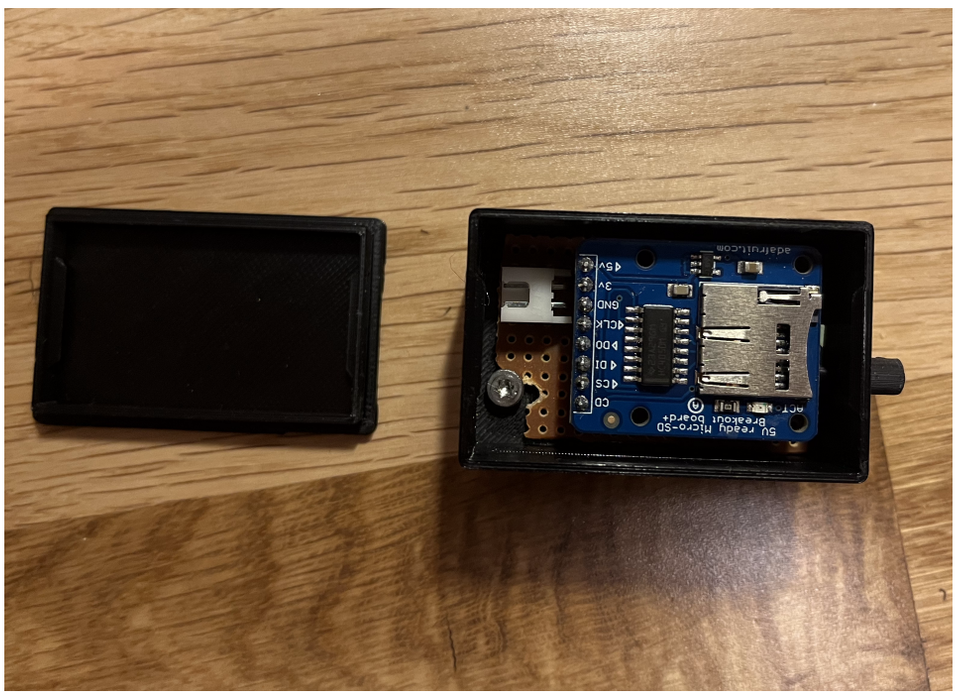

Figure 5.7: Device in the 3D printed case based on the design seen in Figure 5.6

5.4 Monitor Device Software

As Section 5.2 and 5.3 describe the hardware implementation of the device, the software was mostly developed using a breadboard setup seen in Figure 5.8. This allows the schematics to be easily tested with the software as the breadboard allows components to be easily added and removed from the circuit. Using the breadboard during development also helps to minimize potential sources of errors. The reason is that creating a circuit board for testing can create many problems, from poor connection when soldering to unintentional shorts due to the wire isolation melting and more. Additionally, using a breadboard allows different components to be removed when debugging, helping test one configuration isolated. For instance, the pressure sensor can be removed to isolate the SD card and test the configuration and the software responsible for communicating with the SD card. Having the software work on the breadboard also allows an easy transition to the circuit board, as if the device does not work when used on the custom circuit board, something went wrong in the soldering process.

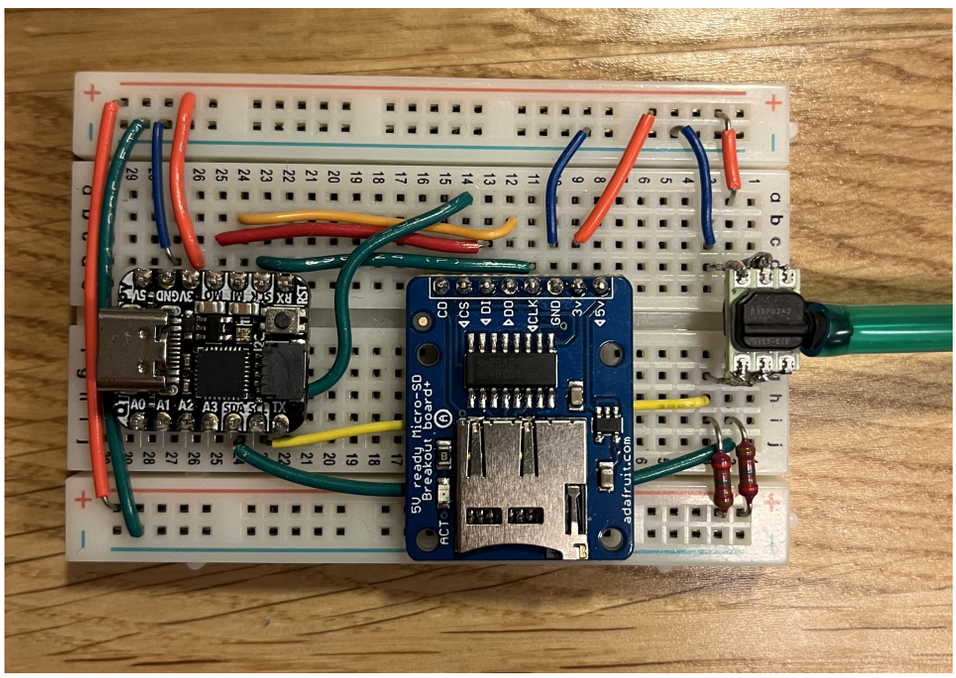

Figure 5.8: Development setup using breadboard

5.4.1 Memory Management

With the SAMD21 microcontroller being a small microcontroller, it is also equipped with very limited memory. The device has 32 Kilobyte (KB) of Random Access Memory (RAM), and by modern standards, this is very little, even for microcontrollers. On top of the limited amount of RAM, the Python-based development framework CircuitPython[2] is also used. While this framework provides fast development time and flexibility, Python is not a language designed for low-level memory control. While the framework is specialized in running on microcontrollers, the Python language provides little control of the memory usage for developers. This is because Python uses its internal memory allocator and garbage collector to handle the memory allocation and cleanup. The garbage collector is part of a programming language that continuously monitors the memory and looks to reclaim memory that is no longer referenced by the program. While this abstraction from memory management is the reason for making Python flexible and fast to develop, it also becomes noticeable when running on 32KB of RAM.

During the development, the limited RAM became a problem, and different

2. CircuitPython [website] https://circuitpython.org/ (accessed 08 May 2023)

techniques had to be utilized to enable all the necessary programs to run simultaneously. The CircuitPython project has addressed these problems when developing on a small amount of memory[3]. When files are uploaded to the controller, the files are read and interpreted by the Python interpreter increasing the RAM usage as these files are human readable, containing extra information that is unnecessary for running the program. To optimize the file size, the file can be pre-compiled to a mpy file. This compilation compiles the Python code into byte code and removes unnecessary parts for running the program, such as doc strings, extra whitespace, and comments. While this improved the memory usage, it was not enough for the program to run. Therefore internal frozen modules were introduced to the system. When a Python module is imported compiled or not, the module is loaded into RAM. To reduce this constant loading into RAM, frozen modules can be integrated into the image itself[4]. By having the necessary library integrated into the firmware itself, the libraries are executed from the flash memory instead of RAM. As the SAMD21 controller has 256 KB of flash memory, this method allows memory usage to be distributed away from RAM into the more spacious flash memory. It is important to note that a lot of the flash memory is used for making the microcontroller operate and providing the functionality allowing the custom program to run on the controller. However, CircuitPython equips the default firmware with modules that are most likely going to be used by the majority of users. By identifying and removing the modules not essential for CircuitPython internal use and not used by the custom program, a portion of flash memory can be freed up to fit our desired libraries.

As the remote data storage was discontinued, the program became much simpler due to the program being able to run sequentially. The program does not need parallel programs to run, as all the tasks can be done in a sequential matter. The program only consists of the measuring done via I^2C and storing the results on the SD card via SPI. With the usage of the development framework, common tasks such as communicating with an SD card are already created, and it is a simple process to implement. The communication with the pressure sensor, however is a more specialized task as Honeywell has a custom protocol to interpret the information received via I^2C.

3. "Dan Halbert", Library File Types and Frozen Libraries, https://learn.adafruit.com/welcome-to-circuitpython/library-file-types-and-frozen-libraries
4. "Dan Halbert", Adding Frozen Modules, https://learn.adafruit.com/building-circuitpython/adding-frozen-modules

5.4.2 Calibration

As described in 2.3.3 all sensors are affected by environmental conditions affecting their accuracy. Therefore sensors need to be calibrated to minimize this effect, increasing their accuracy. For this device, the calibration is performed at startup and is done using the Auto-Zero technique seen in equation 2.3. The key idea for this calibration is to compare the measurements against an expected output. The simplest method for the Auto-Zero calibration is to don't apply any pressure when calibrating, resulting in the expected value being 0 pascal (Pa). The sensor performs 1000 measurements at startup and finds the average offset from the correct output. With the calibration done, the pressure can be calculated using equation 2.4

5.4.3 Reading Pressure Data

While the communication with the pressure sensor is very similar to the SPI version described in my Capstone thesis, there is a key difference. The difference is that the master, in this case, the microcontroller, must request each measurement from the slave device. This is done by sending an 8-bit message from the master to the slave, as seen in Figure 5.9. This initial message contains the slave address and a read bit. The sensor then replies with a two-byte response containing the compensated pressure readings. As the documentation describes, the two most significant bits are used as status bits indicating if the reading was successful. As this protocol is used for all the different sensors in the Honeywell ABP series, it covers three different readouts. Two-byte readout, which is used by the sensor in this project, three and four-byte readout. The three and four-byte readout contains additional temperature information. As a standard, all sensors send four bytes of information regardless of being able to measure the temperature. As a result, the sensor has to retrieve all four bytes of data and neglect the last two bytes as these contain garbage values.

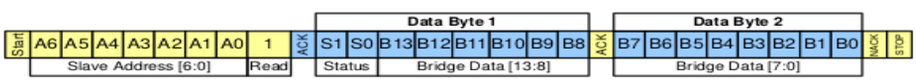

Figure 5.9: I^{2}C two-byte readout from the Honeywell I^{2}C documentation [18]

As mentioned in Section 4.2 the data received from the device is not the pressure itself, but the resistance in the silicon-based membrane. This resistance referred to as counts or output, can then be used in equation 2.2 to calculate the pressure. As the other variables are sensor-specific details defined in the datasheet [13], the function is only dependent on the output to calculate the pressure.

5.5 Backend

The backend's main goal is to support access control and serve the Website following the design and specifications described in Chapter 4.1. This is done using an API serving both the devices and the website. The following section will describe the implementation of these different components making the service possible.

5.5.1 RESTful API

The API main objective is to provide access control to the various resources. Since the backend is also a vital part of making the visualization platform, it also provides many features that are not directly linked to only storing measurements. To serve the website and its features, the API exposes various endpoints for different features. These endpoints can be grouped into different groups with different objectives:

- **Users:** Endpoints for creating, updating or deleting users.
- **Device:** Endpoints for creating and pairing devices and storing measurements.
- **Auth:** Group of endpoints related to authenticating users.

These endpoints follow the same URI structure, where the path is divided into different sections. All requests follows the prefix *api* followed up by their group. For example, creating a user would be a POST request to *api/user/*, and getting the user would be a GET request to */api/user/user_id* where user_id is the ID stored in the database on creation.

The REST API is built using FastApi[5], a high-performance web framework with native integration with Pydantic[6]. Pydantic is a Python library for data validation and parsing, allowing strong typing for Python programs. With FastApi and Pydantic, the API can provide a higher level of type validation than a regular Python program. This type validation also extends to different data structures as well as classes allowing complex objects to be parsed. Pydantic also allows for strong typing to be used, ensuring that data must be in the expected form. With this system, one can ensure that the API only performs tasks when requests/data are sent in the correct form.

5. FastAPI [website] https://fastapi.tiangolo.com/ (accessed 29.April 2023)
6. Pydantic [website] https://docs.pydantic.dev/latest/ (accessed 29.April 2023)

5.5.2 Users

The user is a vital part of the website, and, the API must provide Create, Read, Update and Delete (CRUD) functionality. The CRUD acronym describes four essential functionalities in a database, which can also be extended to API. This extension means that the API must provide the same functionality, only applying custom logic before modifying the database. This logic can extend from formatting the data to the expected syntax for the database query to complex tasks such as authentication. These endpoints allow modifications while following the database models and their typing as described in Section 4.3.1. This is fully handled by the Pydantic library, and the API only returns an appropriate response when Pydantic throws a validation error.

Since the user must provide a name, email, and password upon creation, some fields have to be validated on more complex parameters than just type and existence. The email goes through a soft validation, only checking that the format is of an email, while the password must fulfill several requirements. The password must:

- Contain minimum eight characters
- Contain one lover and upper case letter
- Contain a special character

When the password provided is sufficient, the password is hashed using the sha-256 algorithm with salting to ensure that the data password is secure. Salting is the process of adding a random nonce to the original string before hashing, ensuring a unique hash each time. This adds additional security to the password as the salting removes the determinism from the hashing process, as two equal passwords will get a different hash. Given that the database gets compromised, one cannot find two users using the same password as the salt will be unique for each password, creating two different unique password hash with the same input. This mechanism also helps protect the passwords against dictionary or rainbow attacks [22]

Email Verification

As described in Chapter 4, the user should only gain access after verifying their email. To enforce this, the backend stores all new users as unconfirmed. By then using authentication on the different endpoint related to the device group, the new user will never have the ability to access device-specific data before confirming their user. As seen in Figure 5.10, conforming a user is a multi-stage

process involving many different components. The process starts when either a user is created, or a user is trying to confirm their user via the website. When the user is created, a confirmation code is generated and stored with the ID in the database. After the database changes are complete, the backend sends a request to a remote Simple Mail Transfer Protocol (SMTP) server, sending an email with the confirmation code to the user's email. The user then types the code in the browser, and a request with the code is sent to the API. If the code matches the one stored in the database, the field is removed, and the user role is updated. As email can easily be lost in various ways, users can also request a new confirmation code. This process follows the same logic as seen in Figure 5.10; only the first request is a request for a new code.

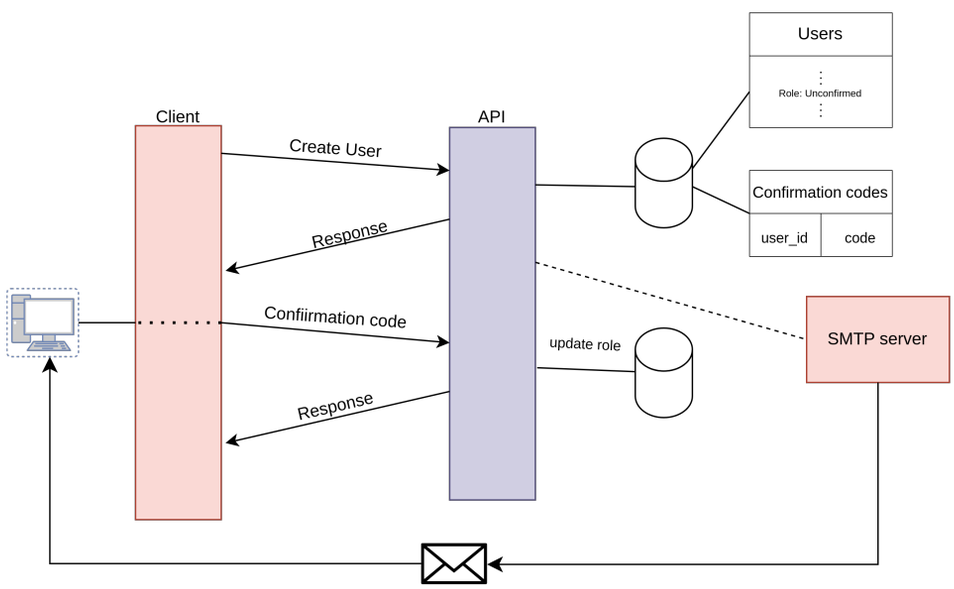

Figure 5.10: Logic for confirming users via email

5.5.3 Security

A vital part of any website handling personal data is the ability to enable secure communication. To ensure that physicians using the platform can transfer personal information over the internet without risking the data being read or manipulated by unwanted third parties, Hypertext Transfer Protocol Secure (HTTPS) is used. The protocol is an extension of the Hypertext Transfer Protocol (HTTP) protocol, the fundamental data protocol used on the world wide web. The HTTPS protocol adds a security layer to ensure that all parties can communicate without the content being read or manipulated. This is achieved by using Transport Layer Security (TLS) or Secure Sockets Layer (SSL) certificate, which utilizes public/private-key encryption.

HttpOnly Cookie

A key attribute for many modern applications is storing information about the user in the browser. To accomplish this, local storage, session storage, and cookies are mechanisms allowing user information to be stored in the browser. As these storage systems can hold user information, attackers may use different methods to gain access to the information stored in the browser. For instance, an Cross-Site Scripting attack (XXS attack) can be performed on a user gaining access to access or refresh a token stored in the browser. As these tokens can grants access to potentially sensitive information as further described in Section 5.5.4. By utilizing HttpOnly cookies using their secure attribute, the data inside the cookie become unavailable from non-HTTP APIs such as the web browser [23], protecting the data from a potential attacker.

5.5.4 Authentication

A vital part of the backend is the ability to authenticate and protect private resources such as device measurements or user information. To solve this, the backend uses JSON Web Tokens (JSON) tokens to validate if a user is signed in. JWT tokens are compact tokens containing information meant to be shared between parties. JWT works on the premise that the content of the token cannot be altered without the recipient noticing. JWT tokens usually do not focus on keeping the data private, i.e., encrypting the data, as it is created to guarantee the data integrity of the content [24]. Given that the website is running over HTTPS and the tokens are transferred using HTTP-Only cookies, encrypting the JWT token is not really a necessity. The HTTPS protocol guarantees that the requested content is encrypted and not accessible to other parties. This allows the JWT token only to encapsulate the validation aspect. This is archived as JWT tokens are issued with a limited validation period, allowing the recipient to easily verify if the token is expired and, thereby, invalid. When a token expires, the user must authorize themselves again to access the system resources. To avoid unnecessary email password authentication, the system provides a *access, refresh token-pair* on the successful authorization. The access token is a short-lived token and is used for accessing resources. In contrast, the refresh token is a long-lived token and is used for generating new access tokens without requiring new authorization. With this set of tokens, a client can be authorized for longer periods of time without requiring the user to authorize themself while still providing a high level of security. As described in the previous section, these tokens should only be obtainable for the specific user, and it is important to protect these tokens. Since the token is transferred using HttpOnly cookies, the tokens are not exposed to XXS attack.

As seen in Figure 5.11 the API can easily authenticate the different requests

based on the JWT tokens. Since the token content can contain arbitrary data, one can also extend the validation to validate the content itself. Some endpoints should only be accessible to users with a specific role, such as admins or physicians. The tokens can be extended to hold information about a user's role and give a 403-Forbidden response if a user with insufficient privileges tries to access a resource requiring higher privileges.

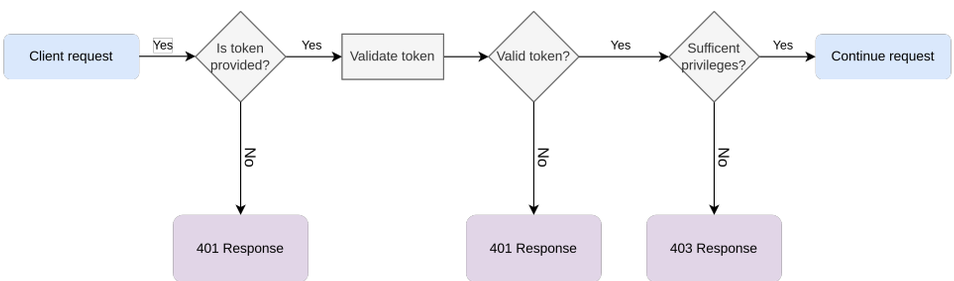

Figure 5.11: Verification process for JWT tokens and the appropriate responses

5.5.5 Database

For the system to provide storage of the various data collected via the website and devices, a MongoDB[7] database is used. MongoDB is a document-oriented database program that provides flexible and scalable data storage. The reason for using MongoDB is its flexibility while still providing an efficient solution [25]. Since this is the initial work on the project, it is difficult to create a solution which easy to expand. With MongoDB, this is simpler compared to a relational database with stricter types who are usually more fitting when the database models are known.

As stated at the beginning of this section, the API is written using FastApi, which is a Python library. Since the API must access the database, Pymongo is used[8]. Pymongo is a package providing a driver for MongoDB, allowing the backend to use the MongoDB API to perform the necessary operations on the database.

5.5.6 Device Pairing

The idea behind having the storage system is to use the device to identify the correct measurements. To accomplish this, the device must be paired against

7. Mongodb [website] https://www.mongodb.com/ accessed 25 April 2023
8. Pymongo [Documentation] https://pymongo.readthedocs.io/en/stable/ (accessed 25 April 2023)

the physician's profile. This is done via the website, where a unique ID is generated and stored on the SD card. This ID is then used by physicians to get and upload the measurements for the device. This is only necessary for the first time using the device as if the ID is already generated, the physician can use this in the verification process. As the website handles private data, it is important that physicians do not have access to the data for a long time after having the device. For instance, if the physician is no longer responsible for the patient, they should not have access to the data. To accomplish the physician device ID is stored in the refresh token, and when this expires, the relation is removed from the database. Ensuring that physicians who are not regularly meeting with the patient do not access the data indefinitely.

5.6 Website

To develop the website React[9], a modern Javascript library was used. The React library provides facilities to create interactive user interfaces. On top of this library, the Chakra-Ui[10] package is used to provide faster development with their component library.

As the website is the only access point between the physicians and the devices, the website works as a gateway between the physicians and the system. It is, therefore, imperative that it is easy to access the necessary features. As described in Chapter 4, the website relay on a dynamic home page providing the necessary steps to get access to the data. Following the chart in Figure 4.4, the home page must handle Registration/authentication, confirmation, and visualization.

5.6.1 User Authentication

When the user first comes to the website, the user is met with a login form, as seen in Figure 5.12. An existing user can log in and access the next part of the website. The form also contains a redirect link to the register so that new users can register. When the user is authenticated, they get redirected to the home page, where the user can see and upload the measurements.

9. React [website] https://react.dev/ (accessed 2 May 2023)
10. Chakra [Website] https://chakra-ui.com/ (accessed 2 May 2023)

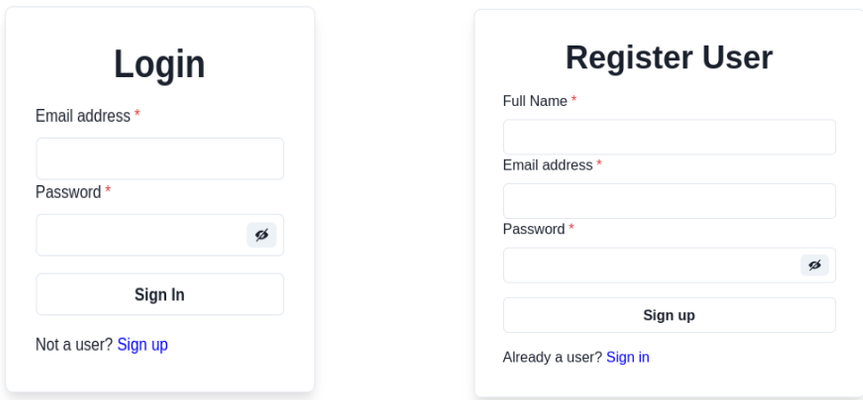

Figure 5.12: Registration and login components

5.6.2 Device Verification

As mentioned, the idea behind the system is to have the physician use the device to get access to the measurements. As described in Section 5.5.6, the device must be linked via a unique ID stored on the SD card. This process is done on the home page, as seen in Figure 5.13. Here the physicians can upload the device ID and then the newest measurements. As described in Section 4.3.1, the device itself is a separate entry in the database to allow multiple physicians to access the measurements and avoid linking one device to one physician.

When a device is first created, it needs to be paired, where an ID file is generated by the website, and the user uploads it to the device's SD card. This creates a new entry in the database accessible via the device ID. The ID is stored in plain text and accessible for everyone having access to the SD card. While this is not optimal, it is mostly to show the potential of a website.

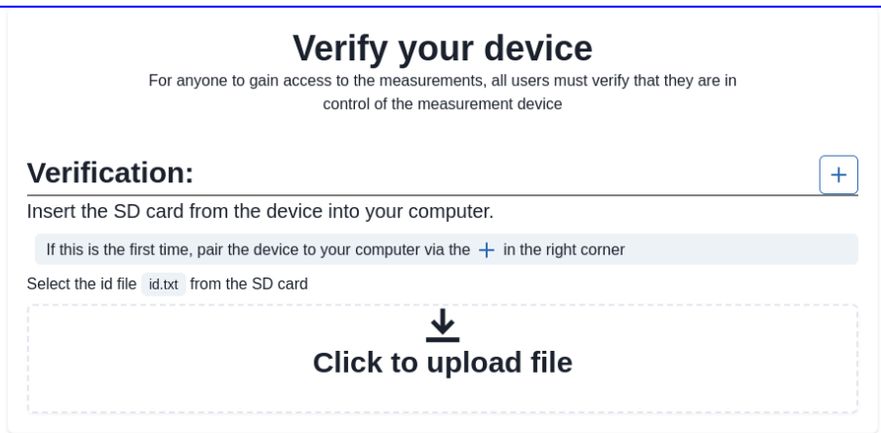

Figure 5.13: Device verification

5.6.3 Measurements

After the file containing the measurements, the data is uploaded and displayed as seen in Figure 5.14. This is done via the ignite-UI's[11] react library for graphing high detailed graphs. This gives the physicians the possibility to see and study the measurements. After the physician has analyzed the data, it can be permanently stored on the server for later retrieval.

11. Ignite-Ui [Website] https://www.infragistics.com/products/ignite-ui-react (accessed 30 April.2023)

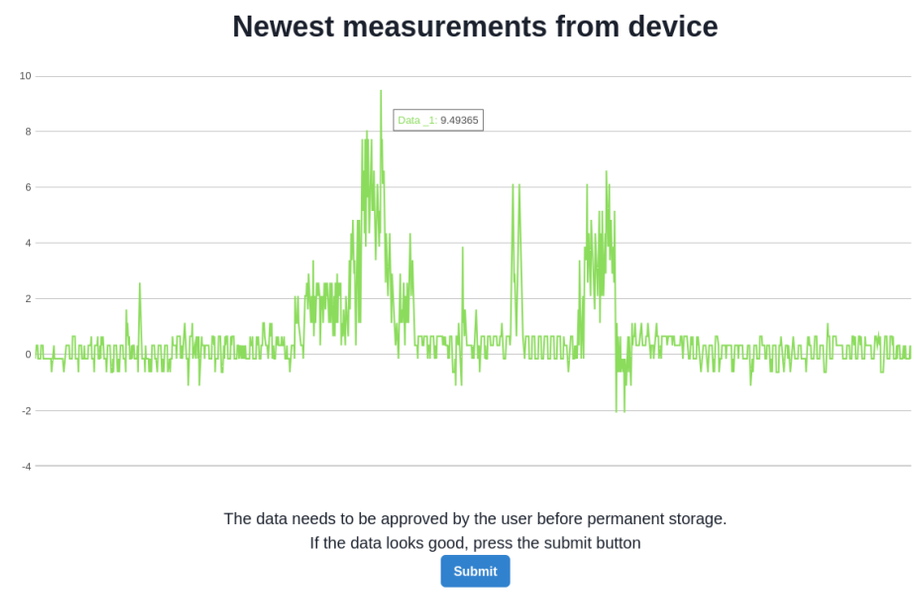

Figure 5.14: Showing the page where measurements form an SD card is uploaded to the website

With the high precision of the measurements performed by the device over 24 plus hours, the graph may become substantially larger than the one seen in Figure 5.14. Therefore the graph has the ability to zoom in on different sections providing more detailed views of the measurements. As seen in Figure 5.15 showing a zoomed-in version of the peak in Figure 5.14, the measurements become easier to analyze.

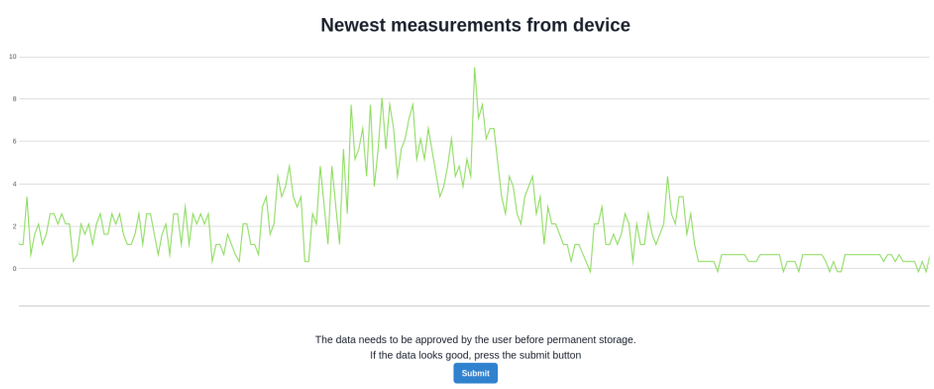

Figure 5.15: A zoomed-in section of the original graph seen in Figure 5.14

When the physician uploads the data to the website, minimal of the data processing is done in the browser. This is done for two main reasons; the first is the fact that the data must be formatted to follow the expected form for the graphing library to work, giving a relatively high workload for the PC. The second is that it was relevant to have the possibility to filter the received data, e.g., removing stale or dirty measurements. These measurements can occur when a tube is removed for cleaning, or the battery is changed. Therefore it was theorized that uploading them to the server and performing the filtering via a data processing program such as Numpy would be a more scalable solution. With a browser handling both formatting and processing of the measurements, it was thought that it would create an unnecessarily high workload for the browser. While this creates a higher load on the server, it is acceptable as this avoids the browser using a lot of resources on the user's computer.

5.7 Summary

This chapter describes the implementation of the different parts of the system based on the design presented in Chapter 4. With the device being the focal point of this thesis, the implementation goes through different iterations to create the final version. This chapter also describes the different aspects of development, such as choosing hardware, creating a development board, and prototyping. With the final device ending up removing remote storage, the storage system and visualization platform are created to focus on on-site transfer.

With the system transferring health data via the website, the storage platform provides authentication for the users to ensure secure transfer. The backend containing the storage system uses an API to allow the website to retrieve the measurements. This API provides mechanisms to ensure the correct users access the data and different user-related functionality for the website to use. Lastly, the website presents the measurements in a graphing component, allowing physicians to analyze the measurements.

6 Evaluation

The previous chapter describes the implementation of the system based on the design outlined in Chapter 4. This chapter provides the experiment for the device and evaluation of the system in context of the requirements defined in Section 3.3.

6.1 Monitor Device

Due to the delayed shipment, the prototype was finished later than expected, so the device was not tested on patients. While this is an important part, the end product can still be evaluated based on the goal of this thesis. With the requirements specified in Section 3.3, the system will be evaluated against the different requirements.

6.1.1 Battery Life

One of the most vital aspects of the device is the energy consumption and how long the device can run on a limited power source. The experiment was performed with two AAA batteries with different battery compositions. During the experiment, the device runs as close to normal as possible; the only difference is that it measures the run time. The device also measures at approximately 50HZ and stores the results to the SD card with the elapsed

time. This time is stored on the SD card every minute to minimize the effect of the test while still keeping a detailed log. During testing, it is not applied any pressure as this should not affect the power usage in a noteworthy manner keeping the test result close to the *true* run time.

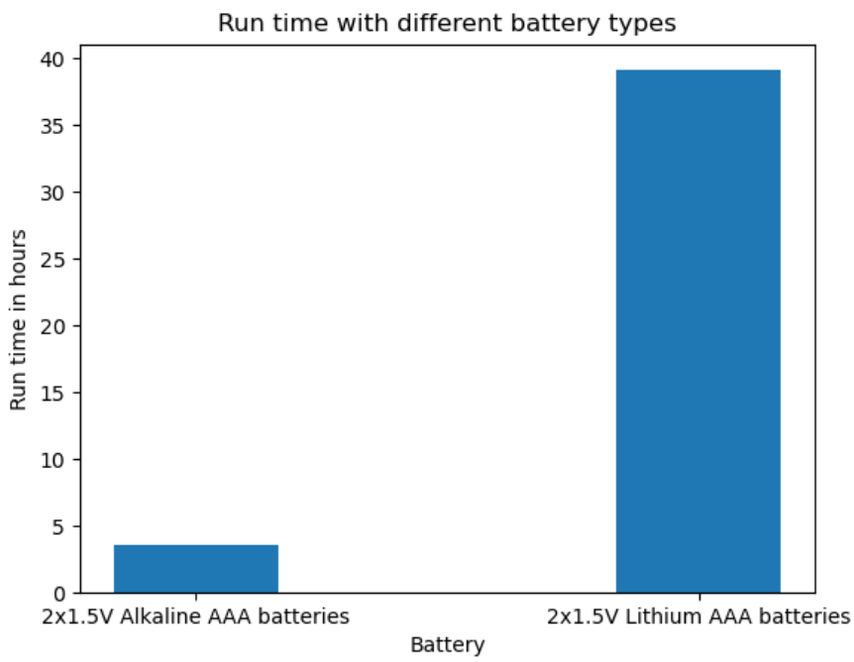

Figure 6.1: Results from the battery life experiment

As seen in Figure 6.2 it is a clear difference between the different batteries. The more widely used alkaline battery has only 3.6 hours of battery life, whereas the lithium battery has 39.05 hours of run time. The lithium battery provides over ten times the run time compared to the alkaline batteries, which is a significant difference. The reason for the large difference in run time is the property of lithium batteries, as these batteries provide the same voltage until there is little to no energy left. In contrast, the alkaline battery loses voltage over time, dropping the voltage below the minimum operating voltage before all of the energy is utilized. This explains the significant difference in operating time, as the alkaline battery cannot provide high enough voltage over the entire battery life.

Given that the alkaline batteries start around 3.3-3.4 volts, it is not a large percentage of voltage loss necessary to go below 3 volts which is the minimum operational voltage. This property was not specified in the datasheet; however,

the pinouts were specified to run on 3 volts, and the battery had about 2.9 volts left after testing. It can therefore be concluded that the circuit cannot run on a voltage below 3 volts which is common for these types for many microcontrollers. These results show that lithium batteries are more suitable for this application compared to regular alkaline batteries.

The original test was also intended to include two different coin cell batteries, CR2025 and CR2430, to see how they performed versus the larger AAA batteries [26, 27]. When trying to use the different coin cell batteries as a power source, the device would either not start or run for a couple of minutes before shutting down. Looking at their specifications, they both provide over 3 volts and when testing on a multimeter, the voltage was around 3.2. With the device not starting using both power sources, the main theory was that the voltage was not high enough as it was below 3.3 volts, the recommended voltage for the microcontroller. Therefore two CR2430 were connected in series to increase the voltage as seen in Figure 6.2. As this would increase the voltage to approximately 6 volts, the batteries were equipped with a regulator converting the voltage down so that the components are not damaged. With the higher voltage, the device started. However, it could only run for 2 minutes before shutting down. The reason for this is the fact that these batteries are designed for low currents over longer time periods. This characteristic does come at a cost, as these batteries cannot provide a high max current. The documentation provided by Omenergy states that these devices cannot provide a current over 10 Milliampere (mA) [28]. Looking at the results from the previous experiments, the current for the device can be calculated to verify that this theory is correct

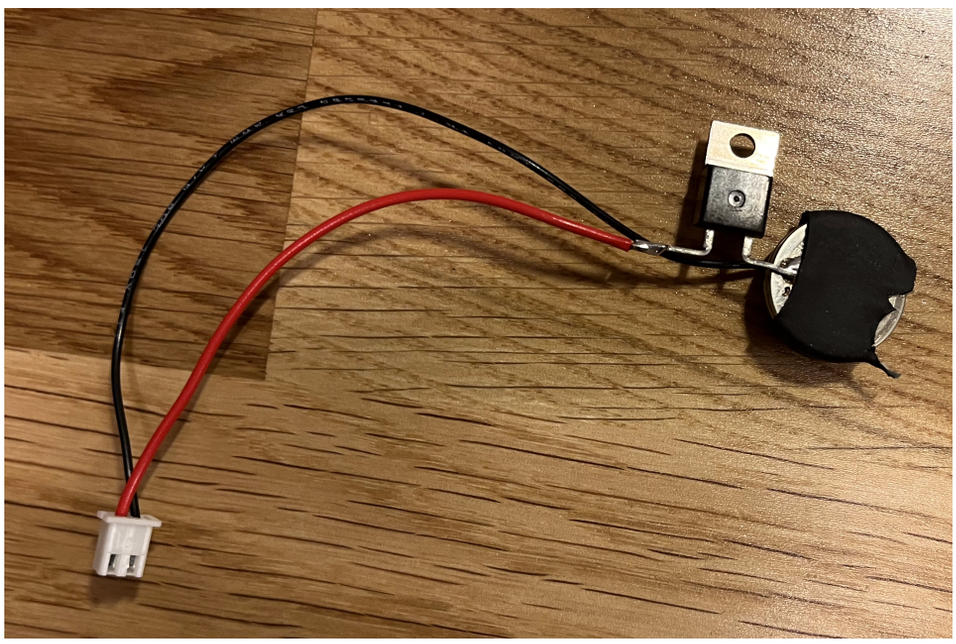

Figure 6.2: CR2430 in series with a voltage regulator, for testing coin cell batteries as a power supply

With the lithium AAA batteries having a capacitance of 1250 Milliampere hours (mAh), the average current for the circuit can be calculated. The circuit power consumption can be expressed as a relation between the capacitance and the operation time.

$$\frac{c}{x} = t \tag{6.1}$$

where:
c = c is the capacitance of the battery
x = is the unknown power consumption of the circuit
t = is the time the device can run on the power source
This can be reordered to find the unknown power consumption x:

$$\frac{c}{t} = x$$

$$\frac{1250\ mAh}{39.05\ h} = 32.01\ mA$$

As expected, the current for the device is over the maximum current for the battery, and therefore the button cell batteries are not able to power the circuit. If the current had been closer to 10 mA, a capacitor could be placed in the circuit to take over when during the peaks. Whether this would work in practice is hard to predict and may not be worth the effort as these batteries only provide

320 mAh, resulting in a theoretical run time of close to 10 hours, given that the voltage stays over 3 volts.

As the previous section described, the device can run for up to 39 hours with lithium batteries. While this is over 24 hours, it is not enough for being used between physician appointments. With a small change moving to rechargeable batteries in pairs, the device can withstand longer periods, however, as further described in Section 7.1, this version is not fully capable of this. While this is not optimal for the project goal, it is a good starting point as the current device can almost give two days of monitoring, allowing the measurement device to capture many of the aspects described in Chapter 2 and 3

6.1.2 Usability

Figure 6.3 shows that the device fits on the tracheostomy tube using the existing velcro and can simply be added and removed. Like the device, the power supply fastens with velcro on the bottom to allow easy integration with the tracheostomy tube. The battery pack uses a sliding lid so the battery can be changed without having to remove the entire device. While this allows for easy change of the batteries due to the placement of the power source, this would most likely be done by an external person.

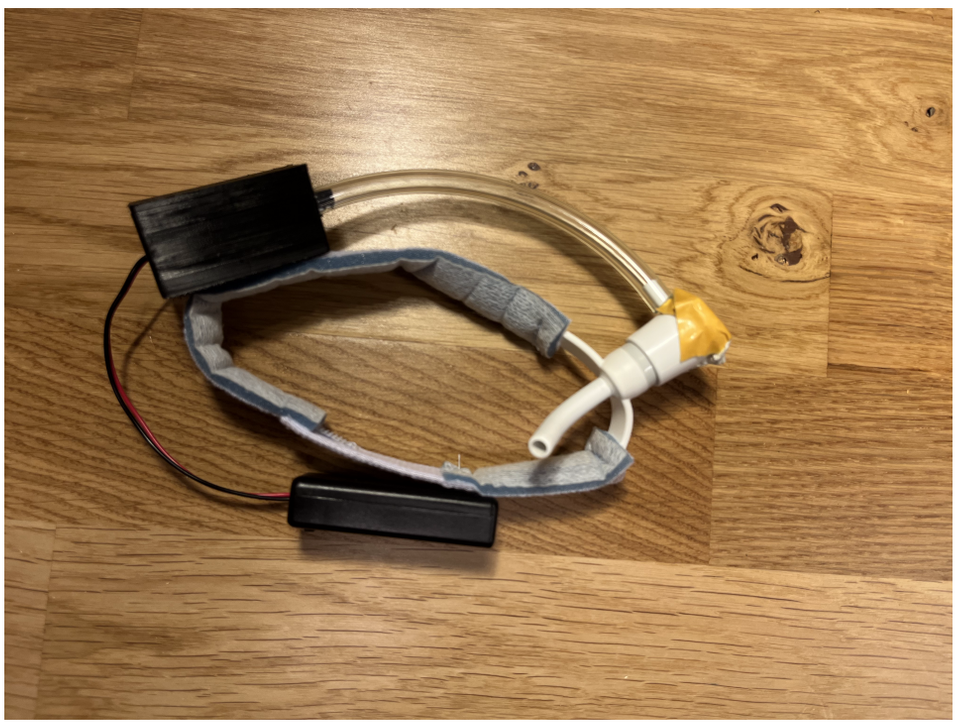

Figure 6.3: Device mounted on a tracheostomy tube

As mentioned, the device should be small enough to avoid limiting the patient, and with the device being $50.5 \times 32.3 \times 25.2$ *mm* the device is not much wider than the strap itself. With the y-direction causing the most problems during usage, having the device close to the same width would allow for easier adaption for children. The device is also relatively light as the device and battery pack weigh 69 grams in total. The device is also designed to use the same diameter as the tracheostomy tube, giving an easy integration with the existing device.

6.1.3 Measurement Attributes

With the usage of the ABPMRRV015PD2A3 pressure sensor, the pressure range is ±15 Pounds per square inch (psi) [13]. The device handles all the different pressure generated by a potential user. Since this is the same model as the one tested in my Capstone thesis, with the only difference being the communication protocol, the device has no problem handling 50Hz sampling rate. As tested in my Capstone project, the device was on average 81Pa away from the *true* output making the device close enough to be used for this research [10].

6.2 Visualization Platform & Storage

An important aspect of this thesis was showing the potential for an easy-to-use visualization platform working with the storage system. While this is a POC, the website provides the basic features to highlight the necessity for such a platform. The website provides an interface to analyze the data; however, the tools are limited. The only tools for the physician to use are zooming and getting the exact value of a specific point.

With these features, the physicians get the most basic needs for analyzing the measurements, which is the most important aspect. The website and storage system can access and display the measurements for a given device. It also provides authentication protecting against unwanted parties accessing the data. A key feature described in Chapter 3 is the possibility to look at previous measurements, the system does not support this as of now. This is an important feature as this gives physicians the possibility to compare and analyze the development of the patient.

6.2.1 Privacy

With the system not storing any personally identifiable data with the measurements, the storage system omits the GDPR directive. By having the system comply with the EU regulations the storage system fulfills this requirement as stated in Section 3.3. While the GDPR aspect is handled by the system not all of the seven principles are not fulfilled by the frontend and backend.

- Lawfulness, fairness, and transparency: are handled by the system as the physicians will inform the patient that the device will sample the data.
- Purpose limitation: the system only processes the data for the purpose of giving better care for the patients.
- Data minimization: is fulfilled as the system only collects data necessary for fulfilling the specific purpose.
- Accuracy: this principle is not fulfilled as the website does not have all the functionality for physicians to update their user.
- Storage limitation: The system will only store data as long as the user has an account.
- Integrity and confidentiality: The system utilizes HTTPS ensuring integrity and confidentiality.

- Accountability: This is not fulfilled as the accuracy section misses some key functionality.

The API has most of the features necessary for updating and deleting users, the website has to implement an interface for these functionalities. While this is a fairly simple process, it results in some work needing to be done before fully complying with GDPR directives. As stated in Section 3.3 the website is a POC and the thesis is not focused on delivering a fully functional visualization platform. Therefore missing these features do not affect the requirement directed towards GDPR laws as this is only focused on the storage system.

6.2.2 Access for Physicians

An important aspect of creating a system for measuring patients dependent on a tracheostomy tube is to gather a better understanding of the development of the children. With the current solution, the physicians would only be able to access the measurement while having the device itself, resulting in limited access to the data. While this is the result of having the patient being anonymous, the overall window for studying the measurements becomes limited. While this limitation is a drawback, the system provides functionality for the most important aspect, integrating it with the existing doctor-patient routine.

6.3 Summary

This chapter presented an evaluation of each component in the context of the requirements stated in Chapter 3. The monitor device is evaluated based on the functional and non-functional requirements with an experiment based on the battery life of the device. This chapter also evaluates the storage system and visualization platform based on privacy, and accessibility.

/7 Discussion

This Chapter discusses the system in the context of the requirements presented in Chapter 3. It also looks at the results found in the experiments and how this can be used to create an improved solution. This Chapter also looks at the systems limitations and how they can be improved.

7.1 Power Source

The main reason for choosing AAA batteries was based on the fact that triple AAA alkaline is a widely available battery type. As seen in the experiment testing run time with different battery types, it becomes clear that these batteries are not suitable for this application. The lithium AAA batteries were shown to provide a substantially longer run time for the device. As these lithium batteries are more expensive and not rechargeable, other batteries are more optimal for long-term use. The overall goal for the device is to have device used between doctor-patient checkups, and with 39 hours of battery life, this is not enough. The goal however was to exceed 24 hours, for two reasons. The first was to have rechargeable batteries work in pairs, i.e. one source is charging while the other is in use. Secondly, was to give the physicians something usable to start with and then iterate based on the feedback.

With the 39 hours of run time with a capacity of 1250 mAh the two devices can easily be used until the other is fully recharged essentially giving the device full

coverage between each physician visitation. With almost two days of battery life, the effort of continuously changing the battery is improved compared to the goal of 24 hours. Looking at the 18650 rechargeable battery, it has a capacity of 2950mAh giving approximately 92 hours of battery life [29].

7.1.1 Handling Power Loss

As this implementation is mainly focused on running on one continuous power source, some features necessary to make it work with the battery pair system described above are not implemented. The reason is that the goal of the thesis is to show the potential of the device as well as provide value from the first version and then iterate over time. Therefore implementing support for handling expected and unexpected power loss was not prioritized. Support for handling these scenarios requires minor modifications for the device and can be added with small modifications in the software. With a patient using the device over multiple days expected power loss i.e. changing the battery may and unexpected i.e. battery running out of power may occur and the device can handle this by always resuming measuring data. As of now the device always calibrates the sensor before usage, however, this value needs to be persistent between runs to allow the device to continue measuring after power loss. To handle this the calibration result can be stored on the SD card and loaded into memory at startup. If the value is non-existent existent, the device performs a calibration. In this version, the device can only be calibrated by deleting the calibration file and the physician has to delete the files from the SD card to ensure that data does not become mixed up.

7.2 Deep Sleep

A feature available in many microcontrollers is deep sleep, this mechanism provides low energy consumption for a defined duration. This sleep period can greatly improve the overall power consumption for the device increasing the battery life. The program does not utilize this functionality as the duration between each measurement is between 15 and 20 Millisecond (ms) and it is theorized that deep sleep would not have a major effect in so short duration. The library used for deep sleep in CircuitPython was also not included in the default build, and with the limited memory, it was not prioritized to make room for the firmware. In the current implementation light sleep is utilized so the system still saves power when not having a task to do, however, moving to deep sleep would also require some structural changes in the code and device. As CircuitPython restarts the code when coming back from a deep sleep, resulting in all initialization needing to be redone. The solution to handle this scenario

is the same as the one described above, and both cases can be handled using this approach.

7.3 Device Identifier

As mentioned, the visualization platform and the related backend is mostly a POC and many aspects of creating a scaleable solution are outside the scope of this thesis. However, there are some potential problems when it comes to the device ID and how they are stored. As of now if for some reason, the original ID for the device is deleted or changed, the connection between the device and the backend is lost, resulting in the physician not being able to find the measurements. A potential solution for this is to use the flash memory of the device to store the first device ID. By having the device store the ID it ensures that the ID file is always correct, and if a new ID is seen this could be erased. When the device starts it checks that the flash memory and SD card has the same ID and if a new ID is inserted it overwrites this. This would ensure that a user error could not result in the connection between the device and backend being lost. While this flash storage can be erased, this would not be an accident by a user which are the most likely event for a lost ID

7.4 Graphing

With the Ignite-UI graphing chart capable of handling millions of data points, the device would accumulate several million points during the first 24 hours. With a data size in the millions, the graphing library would probably have problems handling it without compromising on performance. Therefore the data should be partitioned into groups of millions, allowing the graphing library to run smoothly. As of now the entire content is uploaded and displayed on the website, however it would be a simple process to partition the data on the backend and server the different partion to the user on the website.

7.5 Measurement History

A goal for the visualization platform was to be able to see the entire history of the patient, enabling the physicians to see the development. With the current state of the backend this feature would be possible to implement without any major modifications. The API would need to expose an endpoint where a specific set of measurements can be retrieved. When this is achieved the frontend can

provide a user interface for requesting a previous set of measurements and the website can use the existing system to display these measurments.

8

Conclusion and Future Work

This chapter provides concluding remarks based on the thesis problem description outlined in Chapter 1. The conclusion also reflects how the presented system compares to the requirements stated in Section 3.3. Finally, the chapter provides future work encapsulating some potential aspects the system can be expanded on in the future.

The motivation behind this thesis was to create a system capable of giving physicians the opportunity to collect more information about children using tracheostomy tubes. With this information, physicians can better understand when children are no longer dependent on a tracheostomy, minimizing the chance of a tracheostomy tube being wrongly removed. To give the physicians the tools necessary to further their understanding, this thesis presents a system for monitoring, storing, and visualizing the pressure generated by children dependent on a tracheostomy tube.

As stated in Chapter 1, the main challenge for the monitor device was to be integrated with the tracheostomy tube. As the patient wants a wearable solution with minimal intrusion during use and the physicians wanting a device with a high sample rate and long battery life the different requirements for the device were as stated in Section 3.3. These requirements are the core of the device and the goal of this thesis was to create a device capable of all these

features.

Sample Rate and Pressure Range

The device must measure at 50Hz and support a pressure range of ± 10 *kPa*

The device uses a sensor with a pressure range of ± 15 $psi \approx 103$ kPa, more than capable of handling the pressure generated by humans. The device can also handle well over 50Hz, making the device fast enough to capture at the desired sample rate.

Battery life

The device must be able to run for over 24 hours

As shown in the experiment in Section 6.2, the device can run for 39 hours on 2 lithium AAA batteries. This shows that the device can be used for over 24 hours giving physicians the opportunity to start sampling data over longer periods of time. This also shows that the "battery pair" system described in Section 3.1 is possible with the modification described in Section 7.1.1. This allows the physicians to get data from the device while also providing a solution that can be extended to be used to fulfill the project goal of having the device used between checkups.

Representative data

The device must be small enough to avoid limiting the patient to get the most realistic data

While the device is not tested on children, the device is not much wider than the straps fastening the tracheostomy tubes and it wights 69 grams resulting in some extra weight for the patient. While this is not a high weight increase, this can be noticeable for a child. Without user testing for the device, the limitation is still unknown.

Usability

The device must also be integrated with the tracheostomy tube and it should be easy to equip and remove.

The device and the power supply can be fastened on the existing straps of the tracheostomy tube and can easily be equipped and removed. The device case is also designed to use the same tube size as the tracheostomy tube making the device simple to use for the physicians.

The presented device provides attributes handling the measurement, battery, and usability aspects of the requirements. With the device not being tested on children, it is not possible to conclude whether it limits their behavior or not. The device is relatively small and light so it is likely that it would not pose a limitation in their behavior.

This thesis also presents a system for storing the measurements gathered by the devices to allow physicians to analyze the data. As stated in Section 3.3, the storage has several requirements providing functionality on the platform as well as protecting the privacy of the users.

Storage

The storage system must store the measurements while upholding the GDPR laws.

As the storage system does not use any personally identifiable data when storing the health data, the system upholds the GDPR directives.

Access

The storage system must provide access for the visualization platform to the measurements stored on the system

The storage system or backend uses an API to allow the website to request data. While the physician can see the measurements on the device, it does not allow the users to see the entire history.

Authentication and Authorization

The storage system must provide authentication of the user, ensuring only authorized parties can access the data.

As previously stated the storage system uses an API to allow authorized and authenticated users to gain access to the measurements. With the physicians having to show ownership of the device to access measurements, the system

provides an authorization mechanism to guard the data.

With the storage system providing anonymous storage of the measurement and access via an API and authentication and authorization it fulfills the requirements sat at the start of this project.

The last component of this system is the visualization platform, and as stated in Chapter 1 this platform is a POC. Therefore the requirements defined in Section 3.3, is focused on highlighting the potential of the platform.

Visualization

The visualization platform must visualize the data in a meaningful way

The website presents the measurements in an interactive graph providing an intuitive way of visualizing the data.

Authentication

The visualization platform must provide authentication for the users on the platform

The website provides authentications by requiring the user to login and verify their email to access the features of the website. It also requires the user to show ownership of the device to upload and see the measurement.

Analytics

The visualization platform must provide tools for physicians to better analyze the data

The website uses an interactive graph to display the measurements and this has tools such as zooming in on a section and getting the exact value of the measurement. While this is not the most advanced tools they do allow the physicians to analyze the data.

8.1 Conclusion

As described above the system presents a system capable of measuring, storing, and visualizing pressure generated by children dependent on a tracheostomy tube. The measuring device provides the attributes desired functional attributes, by handling a high enough sample rate and pressure range. It also is integrable with the existing tracheostomy tubes, allowing it to be easily integrated with the current patients. With the device being 69 grams and $50.5 \times 32.3 \times 25.2$ *mm* it is a small and light device, however, it is not tested on children. With the width of the device being close to the strap width it is likely that the device is wearable for children.

The measurements captured by the device are stored for later retrieval and accessible to the physicians. By anonymously storing the measurements the system follows the GDPR directives and using the device as an authentication mechanism, the physician always gets the correct data. With the website working as a visualization platform the physicians can use the interactive graphing feature to analyze the measurements.

8.2 Future Work

With this thesis being the first project working on creating this system, many aspects still need further research and development to fully create a system capable of being integrated into the doctor-patient check-up routine.

8.2.1 Printed Circuit Board

As described in Section 5.2, the circuit board builds in the z-direction to minimize the size in the x and y direction. This proved to make the device more suitable for wearing around the neck, however, with a smaller SD card a Printed Circuit Board (PCB) may be able to optimize this size even more. With PCBs the circuit board can be made or printed by an industrial machine allowing circuits to be made smaller and more precise than a hand solder circuit board could be. This could essentially make a smaller component without stacking the SD card over the other sensors. In addition, when the diagram is made, a new PCB could just be ordered from a manufacturer in contrast to a person hand-wiring the circuit.

8.2.2 Analitical Tools

As stated in Section 3, the physicians would benefit from more advanced tools, such as automatic or manual removal of unwanted data and finding the repertory rate. With these tools, more advanced analyses can be performed by physicians. With the data foundation in place, these tools can range from simple filtering to complex data analysis.

8.2.3 Website

As the website is a POC, only the most vital features were prioritized to show the potential of a website. With the nature of POC solution, many key aspects are not implemented, and therefore, the website needs more features to be fully ready for use. Handling deployment, adding a user interface for updating user profile, and integrating a SMTP server are some of the features necessary for having the website ready for use.

8.2.4 Machine learning

An intriguing possibility for this system is to use machine learning to help physicians make a decision about the further need for the tracheostomy tube. By adding a label on the measurements when the tube was un-/successfuly removed, one can generate a data set with measurements and labels, allowing machine learning to be applied. This model can give a suggestion of the correct action to help the physicians in their decision.

Bibliography

[1] P. P. Ray, "A survey on internet of things architectures," *Journal of King Saud University-Computer and Information Sciences*, vol. 30, no. 3, pp. 291–319, 2018.

[2] B. Pradhan, S. Bhattacharyya, and K. Pal, "Iot-based applications in healthcare devices," *Journal of healthcare engineering*, vol. 2021, pp. 1–18, 2021.

[3] S. Deepika and K. Vijayakumar, "Iot based elderly monitoring system," in *2022 6th International Conference on Trends in Electronics and Informatics (ICOEI)*, pp. 573–579, IEEE, 2022.

[4] M. S. Niederman, R. D. Ferranti, A. Zeigler, W. W. Merrill, and H. Y. Reynolds, "Respiratory infection complicating long-term tracheostomy: the implication of persistent gram-negative tracheobronchial colonization," *Chest*, vol. 85, no. 1, pp. 39–44, 1984.

[5] Kevin Ashton, "That 'Internet of Things' Thing," *Rev. Sci. Instrum.*, June 2009.

[6] Texas Instruments, *Serial Peripheral Interface (SPI) for KeyStone Devices User's Guide* , March 2012. Rev. A.

[7] T. D. Shingare and R. Patil, "Spi implementation on fpga," *International Journal of Innovative Technology and Exploring Engineering (IJITEE)*, vol. 2, no. 2, pp. 7–9, 2013.

[8] J. Chen and S. Huang, "Analysis and comparison of uart, spi and i2c," in *2023 IEEE 2nd International Conference on Electrical Engineering, Big Data and Algorithms (EEBDA)*, pp. 272–276, 2023.

[9] K. Mikhaylov and J. Tervonen, "Evaluation of power efficiency for digital serial interfaces of microcontrollers," in *2012 5th International Conference on New Technologies, Mobility and Security (NTMS)*, pp. 1–5, IEEE, 2012.

[10] F. Mørstad, "Proof-of-concept for an iot device measuring the pressure of patients using tracheostomy tube." Capstone project, University of Tromsø, 2022.

[11] L.E. Hollander, G.L. Vick, & T.J. Diesel, "The Piezoresistive Effect and its Applications.," *Rev. Sci. Instrum.*, vol. 31, pp. 323–327, 1960.

[12] Honeywell International Inc., *SPI Communication with Honeywell Digital Output Pressure Sensors*, May. 2012.

[13] Honeywell International Inc., *"ABP SERIES: Basic Board Mount Pressure Sensor", 32305128 Issue H*, Sept. 2020.

[14] Honeywell International Inc., *Auto-Zero Calibration Technique for Pressure Sensors*, May. 2021.

[15] A. Nautsch, C. Jasserand, E. Kindt, M. Todisco, I. Trancoso, and N. Evans, "The gdpr & speech data: Reflections of legal and technology communities, first steps towards a common understanding," *arXiv preprint arXiv:1907.03458*, 2019.

[16] "Offial journal of the european union."

[17] european data protection supervisor, "10 misunderstandings related to anonymisation," Apr.

[18] NXP Semiconductors, *I^2C-bus specification and user manual*, Oct. 2021. Rev. 7.0.

[19] J. Valdez and J. Becker, "Understanding the i2c bus," *Texas instruments*, 2015.

[20] R. Arora, "I2c bus pullup resistor calculation," *Texas Instrum., Dallas, TX, USA, Appl. Rep. SLVA-689*, 2015.

[21] D. Friesel and O. Spinczyk, "I2c considered wasteful: Saving energy with host-controlled pull-up resistors: Poster abstract," in *Proceedings of the 18th International Conference on Information Processing in Sensor Networks*, IPSN '19, (New York, NY, USA), p. 315–316, Association for Computing Machinery, 2019.

[22] P. Gauravaram, "Security analysis of salt||password hashes," in *2012 International Conference on Advanced Computer Science Applications and Technologies (ACSAT)*, pp. 25–30, 2012.

[23] A. Barth, "HTTP State Management Mechanism." RFC 6265, Apr. 2011.

[24] M. B. Jones, J. Bradley, and N. Sakimura, "JSON Web Token (JWT)." RFC 7519, May 2015.

[25] B. Dipina Damodaran, S. Salim, and S. M. Vargese, "Performance evaluation of mysql and mongodb databases," *Int. J. Cybern. Inform.(IJCI)*, vol. 5, 2016.

[26] Panasonic Energy, *CR2025 CR Coin-Type Lithium Battery*, Unknown.

[27] Sony Corporation, *Lithium Manganese Dioxide Battery CR2430*, Unknown.

[28] Omenergy, *SPECIFICATION FOR LITHIUM BATTERY Model : CR 2032*, Unknown.

[29] Samsung SDI Co. ,Ltd., *SPECIFICATION OF PRODUCT for Lithium-ion Rechargeable Cell Model : ICR18650-30B*, Feb. 2010. Version 1.1.