Wearable Allergy Alarm G1

ECE/COSC 402 Design Report

Group Members: Breanna Slessman, Sandy Fata, Mak Mckinney, and Gracie Parham

TA Mentor: Imran Fahad
Technical Advisor: James Osborne

Executive Summary

Many people struggle with allergies in their daily lives and have trouble narrowing down what exactly causes their allergic reactions. Many people need a wearable device that can help monitor/track their allergic reactions and symptoms. For individuals that have severe allergic reactions, they might even need a device that can alert them that they are starting to exhibit signs of a severe allergic reaction.

To help users who need allergic monitoring, we have decided to create a wearable device that can help users who struggle with environmental allergies and health issues. Since we live in a tech-savvy world, we have also decided to incorporate a compatible app that takes the user's wearable device information and helps track potential allergen-causing environments/scenarios.

When designing this device, we had to consider power consumption, sensors, compactibility, comfortability, data transfer, data storage, and app development. To ensure that all hardware aspects were considered, we used a 3.7V LiPo battery, an environmental sensor, a health sensor, and an OLED display. As for the software, we needed to consider what server would best fit our needs as a means of communication between our software and hardware and a point to store our data. We also had to consider which technologies and coding languages we would use to create and deploy our app, along with how we would actually analyze the data taken in by our hardware device and through other resources.

We have thought through every deadline of this project and come up with realistic goals and checkpoints to keep ourselves on target. We believe that with our current visualization of the finished products and our deadlines, we can create a product that lives up to the expectations of professors and advisors.

Table of Contents

Problem Definition & Background

The health technology market of the modern day is vast, and overripe with interdisciplinary potential, profit, and necessity in our quickly advancing medical age, leaving producers with a certain ethical responsibility. Price gouging and useless scams are never-ending, miracle cures amounting to no more than snake oil, preying on the hopes of those in desperate need. With that responsibility comes a need to be both innovative and transparent about what services are both possible and provided by the technology created.

There is a gap in the field right now in terms of allergy care. There are so many technologies meant to be used day to day to care for things like heart conditions and pancreas malfunction, but next to no products available right now service the specific needs of those who encounter allergens regularly. This is most likely because currently, as we have found with the research available to us, any technology able to detect allergic reactions is still in testing–not in mass production, and certainly not FDA-approved. We are not promising a miracle. We are instead promoting preventative care, mindfulness, and tracking of information that may be useful to doctors.

We would be looking to aid two types of people. The first is people who encounter some sort of allergen on a day-to-day basis, but maybe have trouble identifying it, or who need additional support in tracking exactly when they're exposed to said allergen for accurate reporting back to their healthcare provider. The second customer of ours would be concerned parents of children with allergies. These parents would be looking to track information about what their kids interact with and provide a resource for their children who may not fully understand what to avoid to be safer without their parents constantly having to be helicopter parents to them. Our product seeks to provide independence, safety, and information in the face of these prevalent concerns.

The basic functionality of this device would include sensors to detect the user's current environmental factors and health status, as well as a small display to let the user know their current environment could be contributing to an allergic reaction. The associated mobile app would provide a history of the data the sensors have collected, weather and allergen data collected from online, a way to track symptoms potentially related to allergen exposure, and a way to store confirmed and potential allergens.

The watch will, in theory, be worn during waking hours, passively collecting data as the user goes about their day. The user will collect their information during the day, perhaps scanning a useful barcode or two, and then when they have a spare moment, that is when they will use the app. They will log any symptoms experienced throughout the day for our algorithms to better predict future reactions, and in the morning, they will check their allergy "forecast" to see how their day and week will likely go based on previous data collected and entered. Then, this data could be given to a healthcare provider, either orally or by handing the phone over with the data open, to gain a more accurate, objective idea of what the user has been experiencing and aid in diagnosis. Parents can also access this data if they set an account up for their child to monitor their child's health.

To understand this problem, one must understand the limited ways we, as outside observers, can detect allergic reactions. We must also have an understanding of how external factors influence allergies in order to make accurate statements and predictions about a user's experience.

The closest existing product we have found is the Allergy Amulet [10], though it is not FDA-approved and only detects allergens in food products, whereas our device is focused on the environment. The Allergy Amulet uses a chemical film that contains nanocavities that bind to target allergenic molecules [10]. Our device differs by looking at a holistic picture and focusing on symptoms of an allergic reaction rather than targeting any specific individual allergen. Our barcode scanning feature attempts to help users determine if food is safe while shopping before it's ever served to them on a plate.

A user could theoretically use a Fitbit, Apple Watch, Garmin watch, or similar health-tracking watch, a general symptom-tracking app, and a weather app in conjunction. This would provide all of the information our system does. However, there are a few shortcomings with this replacement. For one, this system would not alert to allergy-like symptoms or triggering conditions. There would also be no predictive work done to give the user an outlook; it would all be retrospective, and the user would have to make their own inferences. Not to mention, it would be a lot of work to connect all of those pieces in a concise and easy-to-read manner.

Requirement Specification

Table #1: Design/Customer Requirements

1. A wearable device that can monitor environmental/health factors that signify an allergic reaction and can alert the customer of potential allergens a. The device has an environmental sensor b. The device has a health sensor c. The device is battery-powered d. The display can show a warning message to users
2. A compatible app that has symptom tracking/logging and weather information. a. Capable of presenting current data (to the second) b. Symptom tracking (user-inputted data)

c. Analytics (ex. predictions) easily found here d. Runs smoothly with little noticeable lag/load times (less than five seconds) e. Aesthetically pleasing and nice to interact with
3. Communication between hardware and software a. The hardware uses an I2C bus b. Hardware uses Bluetooth Protocol to sync with the App
4. A database to store customer data taken from wearable device a. Stores data from both hardware and software b. Stores data by user c. All parts of the software have the ability to interact with the database

When creating a wearable allergy alarm device, ensuring that a large range of potential allergens can be monitored is essential. To this end, we have incorporated an environmental sensor that tracks temperature and atmospheric pressure. We also incorporated the use of a health sensor that monitors heart rate (BPM) and oxygen levels(SpO2). Lastly, to provide this information to users in a timely manner, this information is be displayed on an OLED screen.

We decided to use an environmental sensor (BME180) that monitors temperature and atmospheric pressure. These two environmental factors are often big indicators that an environment could cause a potential allergic reaction. In more recent years, with climate changes becoming more evident, this all-around temperature increase can have a huge effect on the production and release of pollen, causing allergy season to begin earlier [1]. In the Northern US, the allergy season typically begins around springtime (March- May), meaning temperatures are starting to rise to be around 50-60° F [2]. High temperatures often indicate the production of allergen causing plants such as ragweed and certain trees [11].High humidity can also be a potential indication that an environment might be suitable for causing an allergic reaction because areas with high humidity are often associated with areas that also have heavy rainfall/thunderstorms which can spread pollen farther and break pollen down into smaller particles leading to “Thunderstorm Asthma” which causes these smaller pollen particles easier to ingest [8]. For similar reasons, areas that commonly have thunderstorms are known to be in areas with higher atmospheric pressure levels. While having high temperatures, high humidity, and high pressure alone wouldn’t signify a potential allergic reaction, a combination of at least two would be enough to set off our alarm system. Our alarm system will be set off when individuals encounter an environment with spring-like temperature levels (60° F / 16° C) or higherand high atmospheric pressure levels (1030 hPa) or higher.

Many health-conscious individuals who would like to monitor their allergies would also like to monitor certain aspects of their health. Due to this reason, we’ve decided to use a health sensor (MAX30102) that monitors oxygen levels and heart rate. When an individual is experiencing a severe allergic reaction and is going into anaphylactic shock, for example, their heart rate often rises due to the stress of the situation, and their oxygen levels decrease because breathing often becomes difficult so monitoring these two health aspects would be essential for users with food allergies [9]. Once again, while a user only experiencing a higher reading of heart rate or low oxygen levels alone couldn't be enough to signify a potential allergic reaction, a combination of both could signify that something is wrong. To account for this, our warning system will be set off if the user experiences high heart rate levels (120 bpm) or higher and low oxygen levels (90%) or lower.

As previously mentioned, the final result will include a wearable device, so it is vital that this project has some source of long-lasting charge. To combat this, we have decided to incorporate the use of a 3.7V LiPo battery that is specifically recommended for the Adafruit Feather nRF5252832. When using a small battery like the LiPo 3.7V, it is vital to consider power consumption and how this can affect the overall project. Bluetooth Low Energy protocol was also used to help mitigate power consumption.

There are many reasons we need a functional companion app to our system, primarily being that without the app, users cannot access data beyond the statistics present on the watch face. This app also provides us with the ability to present data the user would be otherwise unable to access with just the watch and gives us, as the creators, a convenient avenue by which we can accept user-reported data such as symptoms and allergens to watch out for. With an optimized backend and efficient use of power, we should be able to create an app that serves as a hub for meaningful information that the user will find pleasant to interact with in a way that encourages consistent use of the watch.

The hardware will connect to the app and database using RESTFul APIs. The hardware will have a hardware ID (assigned when registered to a user, changed if registered to a new user) as an approved API caller. The device will then POST sensor data or FETCH user information to display on the watch, including allergy warnings from the app.

Since our project includes both a physical hardware device and a software application, there needs to be a way for both parties to get access to the same data. Along with that, users should be able to create accounts that store only their data. In order for this to occur, there has to be a database that stores all collected data and separates it by each user's account. Since our wearable device will be collecting data that data needs to be stored in the database in order for our other code modules to use this data. For our software to actually make use of this data, we need to have it organized accurately in the database and ensure that all our code modules have a way of efficiently connecting with the server to gain access to the precise data needed.

Standards

IEEE Standards

The Adafruit Feather nRF52832 uses IEEE 802.15.1, the Bluetooth Low Energy (BLE) standard, allowing energy-efficient communication with paired devices, such as smartphones. The use of IEEE 802.11 and 802.15.1 standards imposes constraints on module selection and communication protocols, requiring robust interference management and energy-efficient operation to ensure reliable data transfer.

Additionally, the wearable allergy device follows:

- ISO/IEEE 11073-104xx for Health Informatics – Personal Health Device Communication – Device Specialization.
- IEEE 11073-10417-2023 for health informatics, ensuring interoperability and secure data handling.
- IEEE 11073-20601-2019 for interoperability between different device types and vendors.
- ANSI C63.27-2017 for the evaluation of wireless coexistence to manage interference between wireless communication protocols.

Safety Standards

The project employs the safety standard ISO 26262 to ensure electronic system safety. The microcontroller, sensors, and battery are designed and organized to prevent malfunction and overheating, both of which could be dangerous to the watch wearer. IEC 60950-1 is used to ensure that devices are powered by batteries and includes protection against overcurrent, short-circuiting, and excessive heat.

Additional safety and medical device standards include:

- ISO 14971:2019, which mandates risk management for medical devices throughout their life cycle.
- ISO 13485:2016, ensuring a quality management system for medical devices.
- IEC 62304:2006+A1:2015, covering medical device software lifecycle processes.
- ISO 10993-1:2018, which governs the biological evaluation of medical devices for biocompatibility.
- IEC 60601-1-2:2014, ensuring electromagnetic compatibility for medical electrical equipment.
- IEC 60601-1-11:2015, addressing medical electrical equipment in home healthcare environments.

Voltage Standards

We will be using USB 3.0 Voltage Specifications to power the microcontroller safely using common power sources. Our design practices align with IEC 62133, which governs the safe operation of LiPo batteries, including overcharge and discharge protection. The project must ensure that the battery provides sufficient power for continuous operation while adhering to voltage and current safety standards. This affects the selection of components and the design of power management systems.

Hazardous Materials

This project complies with the Restriction of Hazardous Substances (RoHS) Directive, ensuring that all electrical components are free of lead, mercury, cadmium, and other harmful materials so that the device is environmentally friendly and safe for prolonged skin contact. Compliance with RoHS limits supplier options and imposes additional checks during procurement.

Data Security & Labeling Standards

- ISO/IEC 27001:2013, which ensures information security management, protecting user data and device integrity.
- ISO 15223-1:2021, which specifies medical device symbols for labeling.
- NXP Semiconductors: I2C Bus Specification, which ensures reliable inter-chip communication for sensor and peripheral integration.
- Bluetooth Special Interest Group (SIG) Bluetooth Core Specification, ensuring interoperability and low-power operation of Bluetooth communication.

Technical Approach

As previously mentioned, we will be constructing a rechargeable battery-powered wearable device and a compatible app for users. This device will transfer data through Bluetooth Low Energy (BLE) to our server and compatible app either in real-time or every few seconds. Since this app is extremely interdisciplinary, it covers a wide range of computer science, computer engineering, and electrical engineering design principles.

When considering a wearable allergy warning system, the hardware team made sure to consider environmental sensors, health sensors, data display, and data transfer. There is a wide range of available environmental and health sensors on the market, so we mainly just had to ensure that we chose sensors that operated at a voltage compatible with our microcontroller (3.3V), had a communication protocol compatible with our microcontroller (I2C), and compact enough to keep the device wearable.

When considering the software, there was a focus mainly on data transfer, data storage, and app development. Data transfer and storage are both considerations when dealing with the backend and database, while the app is a matter of efficient and smart coding. All aspects will have to be optimized so there is minimal lag, loading times, and maximized data transfer due to the constant stream of input from the watch.

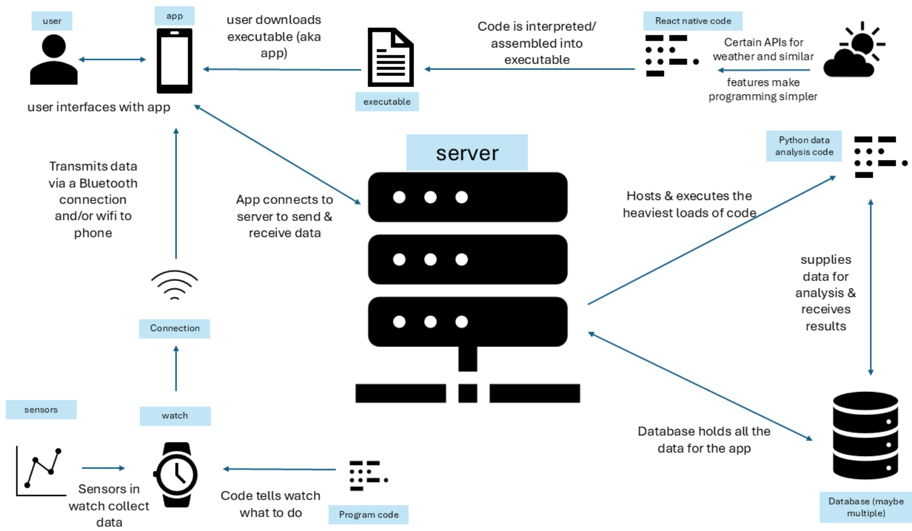

Figure #1: Flow of Data Diagram

The software for our project consists of about three parts: the frontend React Native, Node.js application, the Python data analysis and backend code modules, and the Google Firebase database, which will contain all of our collected data. The backend of the app will also be hosted on Railway.app in order to interact with the database and the frontend. All three sections of our software will need a way to interact back and forth with each other and mainly with our database, as that is where the data from the hardware will be stored. Each technology and coding language that was picked for our software was a decision made through strategic research that proves that these technologies can serve the purpose we intend for them and can also interact with each other through methods such as APIs.

The main choke point in terms of efficiency will be in the connection between the backend and the app, as that is where the most information has to travel from the quickest, and it must go both to and from the server while the app is open if we want it to run properly. We also need to make sure our analysis code isn't too heavy and that it only runs at certain times, that way it won't cause a lag spike as soon as the app opens.

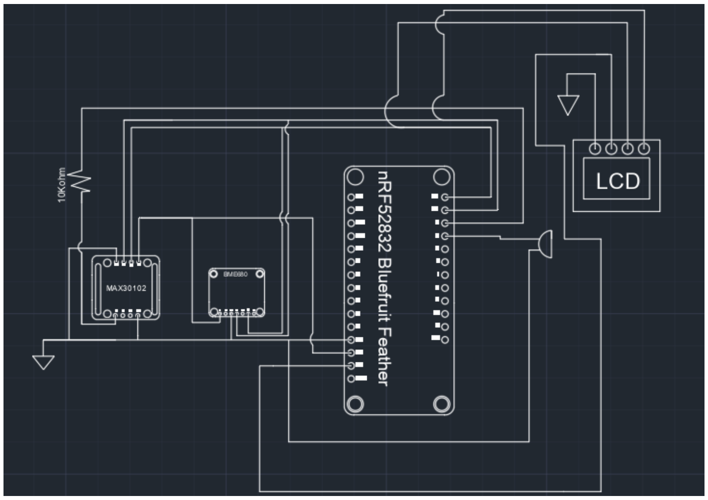

Figure #2: Hardware Wiring Diagram

Components:	***Average Current Draw***
OLED Display	*15 mA*
MAX30102	*(sampling rate: 100 Hz, 5 mA)*
BME180	*2 mA*
nRF52 Feather	*15 mA (using BLE)*
Total:	*37 mA*

Figure #3: Battery Management

Each component in the system contributes to the overall performance by influencing the processing load, power consumption, and functionality. The microcontroller's processing power determines the system's ability to handle data from sensors, manage communication protocols, and update the display in real time. Current and voltage regulations are critical for ensuring stable operation, particularly for high-demand components like the OLED display. The choice of pins for connecting components to the microcontroller affects signal integrity and efficiency, while the logic implemented for data processing optimizes the system's responsiveness and minimizes delays. These design considerations ensure that the wearable device operates efficiently while maintaining a compact form factor and meeting performance requirements.

Design Decisions

Table #2: Hardware Design Decisions

Design Decision	Risk	Likelihood	Impact	Mitigation Strategy
Bluetooth Module (nRF52832)	Interference with WiFi signals (2.4 GHz Band)	Low	Medium	Use channel management to avoid interference
OLED Display	Data refresh delays during real-time updates	Medium	Medium	Use optimized libraries to improve refresh rates
Sensor Calibration	Inaccurate readings due to poor calibration	High	High	Perform Calibration in Controlled Environments
I2C Protocol	Bus Congestion with Multiple Devices	Medium	High	Assign Unique Addresses and Prioritize Data Flow

The Adafruit Feather nRF52832 integrates BLE for short-range communication, which will enable real-time notifications and alerts without draining the battery significantly. The ESP32 BLE was considered, but it would complicate the software stack and increase the processing load. Classic Bluetooth modules offer higher data transfer but at the cost of greater consumption. The nRF52832 is an ARM Cortex M4 device, which will allow it to serve as a driver of the entire project. However, its wearability limits the GPIO pins, so I2C is the correct protocol to increase pin efficiency. SPI is faster but needs more pins, and UART is not ideal for addressing multiple devices.

When looking at the app, there are many areas where decisive action needs to be taken. Before we even begin programming, we need to decide what operating systems this app should be available on, and adjust our programming language and content accordingly. The programming language we choose will also affect our compatibility with languages that deal with other aspects of the project, such as data analysis and server interaction. If the languages compile into different machine languages, that will make us liable to translate their actions to each other, which would only serve to complicate the project. Then, once the programming language is decided on, there are many decisions about how and when the app runs in the background that will impact things such as our phone's battery life, our accuracy as an app, and lag times. This is especially important to take note of, as a theoretical user would not want to use our app if it impacts their quality of life so severely or if it's outdated information to the point of being incorrect. The smoothness of interaction with the app will be our second priority, just after fully debugged app functionality.

When looking at a database to use for this project, there were many things we needed to take into consideration. Since our overall project includes multiple code modules for the software and a hardware device that collects and displays data, one of the most important considerations is a database that can interact with all of these different parts in an efficient manner. Another consideration is how we will actually get the data from whichever part of the project to the database and how quickly this can happen. We have the option of either a database that we can directly connect our hardware device to using Bluetooth or the option of going through the app and then into the database. This, though, depends on the database we end up selecting.

Design Concepts, Evaluation, & Selection

When creating a project, it is vital to consider design decisions and how to narrow down what fits your project best. Below, we go further in-depth about user functionality, software and hardware decisions, considered alternatives, and how we can create test cases to ensure that our project is working properly from all angles.

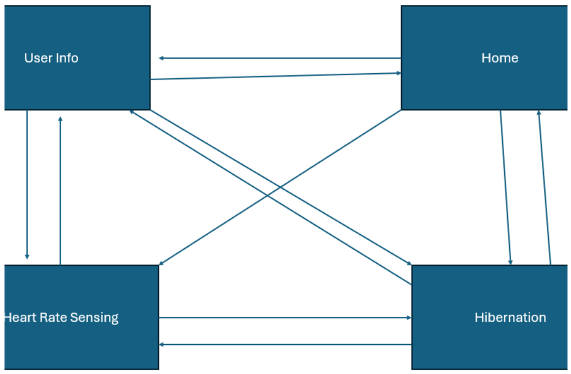

Figure #3: State Map

The state map for the wearable allergy alarm demonstrates the system's functionality and user interaction flow. The device operates across four primary states: Hibernation, Home, Heart Rate Detection, and User Info. The Hibernation state is a low-power mode designed to conserve battery life, and it is accessible from any other state to maximize energy efficiency. The Home state serves as the central hub, allowing the user to navigate to other states or view general information. From the Home state, users can transition to the Heart Rate Detection mode to log heart-related data to the app or the User Info state to access personalized data such as associated account email or settings.

The system supports intuitive navigation through two methods: a carousel mechanism and direct selection buttons. The carousel allows users to cycle sequentially through all states in a circular fashion, providing a simple and predictable navigation experience. Direct selection buttons offer an alternative for users to jump to a desired state immediately, enhancing accessibility and usability. Additionally, the Hibernation state can be entered manually by the user at any time or automatically after a period of inactivity, ensuring effective power management without disrupting the user experience. This flexible design focuses on energy efficiency and user convenience, with seamless transitions between states to support the device's intended functionality.

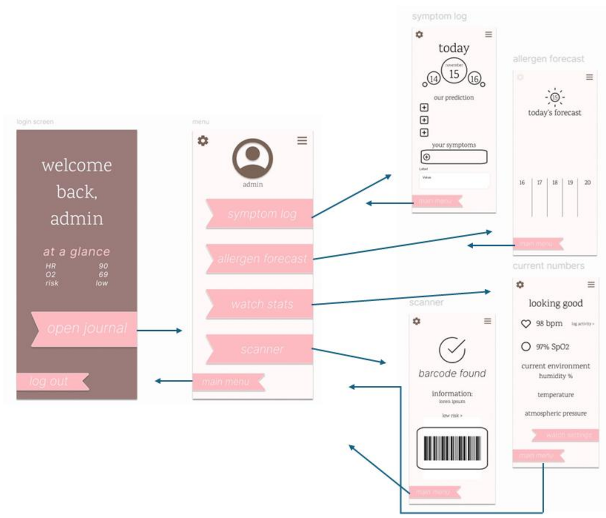

Figure #4: Figma Prototype

The user is intended to open the ‘journal’ section and be able to access any of the features they may want or need; for example, if they would like to log their symptoms, as the requirement states, they may go to their symptom log. There isn’t a particular flow necessary with this app, as the user would, in theory, come for one feature at a time, although they may exit from the symptom log to check their forecast when they first open their app in the morning. Each of these features (barring the scanner, which simply provides data pulled from a database) has an element of personalized data analysis built into it, from the low vs. high-risk calculation to the symptom prediction in the log. The ‘watch stats’ page also has a button to take the user to their watch settings to adjust things like frequency of checks and watch calibration, although that was not featured in the design as it will be heavily dependent on how the device gets built.

The typical use day for this app goes as follows: the user logs in and opens their forecast to see what’s in store for them for the day. They act appropriately, taking preventative medications if needed and going about their day. When they notice their ears feel plugged up, they check their watch stats and see that the atmospheric pressure is high today. They go in and log their symptoms, adding a note that they didn’t get much sleep last night, so that might impact things.

When we are building the app, there are a few evaluation metrics we can measure to adjust major design decisions accordingly. One metric is load time. If we notice a significant amount of time is taken to load the information into the app, we may want to consider either a different database, or different timing around loading the information – for example, fetching the data only when that tab is opened, or conversely, loading all the information upfront, so the app has a seamless feel to it once loaded. If it takes so long that neither of these situations work, we may want to consider switching databases or changing how the data is presented – for example, only presenting a few minutes' worth or a formulated average value by default.

Another way we can evaluate the app is by conducting a Wizard of Oz session early on or running a blind test for a consumer once the app is progressing. A Wizard of Oz session is where an experiment conductor runs a mock interface for a user to interact with. The goal is to find where users get stuck before actual code gets written. Alternatively, we can simply test users' experiences in a similar way once the app is in development, as long as we don't expect any major changes to be necessary.

While testing, we will be operating on a control dataset with normal information and potentially multiple test datasets with data that should trigger a specific reaction in the app. This dataset will be compiled randomly using code and should allow the software to develop independently of hardware at the beginning phases of the process.

At first, we thought we would use something commonly used for our database like Amazon Web Services, because we thought there would be a lot of readily available information on how to use it and how to incorporate it into our project. After conducting some research and consulting with a teacher assistant about this topic, we were advised to look into Google Firebase as a better alternative. Our initial issues were with understanding what kind of AWS server we should use and how we could make a prediction of the cost of that server. When considering Google Firebase as an alternative, we first had to research pricing and compatibility with the rest of our project, and it seemed straightforward. [7] We consulted with our technical advisor about the switch, and he approved, so at that point we felt that we had enough reason to choose Google Firebase as the database to use for this project.

Commented [1]: double check IEEE in text citations format later

Chosen Hardware	**Considered Alternative**	**Analysis of Chosen Hardware**
Adafruit Feather nRF52832 Bluefruit Microcontroller	*TM4C123G*	*Less functionality but more compact and realistically wearable. Bluetooth on board*
Adafruit BME180 (Environmental Sensor)	*DHT22 (AM2302)*	*Monitors temperature and pressure as opposed to temperature and humidity.*
MAX30102 (Health Sensor)	*DF Robot Heart Rate Sensor*	*Monitors heart rate and SpO2 levels, more compact*
Adafruit 0.96" 128x64 OLED Display (SSD1306)	*Nokia 5110 LCD Display*	*Uses I2C, better resolution, and a higher refresh rate*
LiPo 3.7 V battery (350 mAh) / (500 mAh)	*Solar Panels*	*Easier integration with microcontroller, less complex*

Table #3: Hardware Design Decisions and Contribution to Device Performance

Evaluation Plan

Test Case 1.1: Sensor Data Accuracy
Objective: Compare the accuracy of environmental readings (temperature, pressure).
Method: Place the sensor in a controlled environment with known levels. Record output over twenty-four hours at different conditions. Compare readings against calibrated measurement tools.
Metrics: Accuracy percentage for each parameter, consistency of readings over time.
Pass Criteria: +/- 5% accuracy for levels, Bell Curve for Deviation.

Test Case 1.2: Power Consumption
Objective: Evaluate power consumption under active and standby conditions.
Test Method 1: Measure the current draw during sensor operation and in idle mode, varying sample rates.
Metrics for Test 1: Average current consumption in mA, total power consumption in mWh over a standard use cycle.
Pass Criteria 1: Minimum power draw that does not compromise accuracy.
Test Method 2: Test maximum use duration and maximum idle duration until battery death.
Metrics for Test 2: Life hours

Pass Criteria 2: Minimum 3 hours heavy use duration, minimum 10 hours idle only duration, minimum 8 hours mixed use duration.

Test Case 3.1: Communication Protocol
Objective: Bluetooth data transfer efficiency
Test Method: Transmit different sized files to the server, record time taken for transfer
Metrics: Data transfer time and reliability of transfer
Pass Criteria: Bluetooth is fast and reliable for every size of file it will need to send regularly

Test Case 4.1: Navigation Method
Objective: Evaluate User Experience for Both Navigation Methods
Test Method: Conduct usability tests with random participants unfamiliar with the device. Ask participants to navigate to a specific state using both methods.
Metrics: Average task completion time and user satisfaction.
Pass Criteria: High Satisfaction Score and minimal task completion time.

Beyond hardware, we will run a series of tests on Google Firebase to ensure it can handle the strain of the computational load.

Test Case 5.1: Server Constant Data Transfer
Objective: Test if the server is capable of handling a constant stream of data.
Method: Feed the server data at faster intervals and for increasingly longer periods of time until it begins lagging.
Metrics: The time information logged compared to time sent (ms).
Pass Criteria: The server does not begin lagging until the rate of sending is so small it is imperceptible, even when it goes for long periods of time.

Test Case 5.2: Server Disconnected Catch-Up
Objective: Test how fast the server can process a large chunk of data
Method: Create five datasets that grow incrementally larger, equivalent to one hour of backlogged data, two hours, four hours, eight hours, and twenty-four hours, as well as a control of thirty seconds worth of data.
Metrics: Test time compared to control (ns).
Pass Criteria: Tests scale upwards from control less than or equal to the calculated proportional time based on data increase. Time should not exceed one minute.

If Google Firebase fails one of these tests significantly enough, we will revisit our server choice and discuss potentially purchasing an AWS server for a limited time to run similar tests and compare their performances. AWS will need to improve efficiency by more than 5% for us to stick to it, otherwise the cost does not justify the performance.

Commented [2]: Where did test case 2.1 go....

Social Impact Evaluation

Within society, there is a huge disconnect between middle and lower class consumers and accessible, affordable healthcare. Many consumers have the desire to monitor and treat health related concerns, but no real way to do that. The health technology industry has flourished for that reason, but allergy care–a facet of individual health that is not only prominent, but sometimes fatal–still falls behind.

The innovation and progression of technology within society as a whole falls on the backs of engineers. It is engineers who must pave a path forward, through trial and tribulation, to greater access where medicine and research fall short due to the increasing focus on profit over prosperity. It is a responsibility that must not be taken lightly; as engineers, while we must maintain forward-thinking mindsets, we cannot extend our scope past the knowledge we have as a team, whether that team involves scientists and researchers or information only available to the public and us leading as researchers. We must also understand that health data is private, and it is our responsibility to handle it safely and responsibly once it is in our care.

When marketing and engineering a consumer-grade home medical device, it must be understood that some consumers, regardless of warning, will rely on the device in place of a medical expert. Professionally, the correct messaging, including symbol use, must explicitly communicate that the device is not as reliable as a licensed and educated medical professional. However, ethically, the engineer must understand that these messages are often ignored, and we must create a device that is as reliable as physically possible. The device must have negligible false positives/false negatives and must maintain functionality throughout expected wear/tear/stress on the device.

Deliverables

When we finish our project, we will have a physical device that a person can wear on their person. The device will have sensors that collect information about their environment, such as temperature and pressure, as well as a health sensor that will collect information about the user's BPM and SpO2 levels. The device will also have a screen that displays warning messages. There will also be a complete companion app, connected to the watch, that can be used on both iOS and Android platforms. The app will provide useful functionalities relating to tracking allergy symptoms, allergic reactions, and assisting with preventative care.

Project Management

Table #4: Project Milestones

* Red deadlines are hard deadlines imposed by the professor; black deadlines are personal deadlines to keep us on track

Deliverable	Description	Assigned Member	ETA
Video Presentation 1 min	Create a video describing each individual vision for the project	All	9/20/24
Video Presentation 3 min	Create a video describing the team's overall vision for the project	All	9/27/24
Video Presentation 7 min	Create a video describing design decisions, alternatives, and assumptions	All	10/4/24
Video Presentation 10 min	Create a video addressing the purpose of the project, technical scope, development plan, and key design choices	All	11/8/24
Figma Prototype Finish Hardware Diagrams	Finish frontend prototype for inclusion in the report. Create a hardware wiring diagram and data transfer diagram	Breanna Mak Gracie	11/24/24
Design Report	Finish final draft of the design report for approval by TA and submission by 12/3	All	12/3/24
Backend Understanding Begin Testing & Programming Sensors	Understand how to communicate with backend for our specific purposes (knowledge-based deliverable) Test receiving/sending sensor data to app/server	Sandy, Breanna Mak, Gracie	12/4/24
Frontend Prototype Begin Testing &	A semi-functional version of the frontend done using Node.js and react native. Implement weather API. No hard aesthetics implemented. Should accept basic user input for testing. Test sending barcode data to the	Sandy, Breanna Mak, Gracie	12/7/24

Programming the Barcode Scanner	main microcontroller/server		
Backend Prototype Program Display & Buttons	A semi-functional version of the backend done. The database should have space for account information and storing watch data. The database should be hosted on a server. Program buttons to interact with the barcode scanner and display	Sandy, Breanna Mak, Gracie	1/30/25
Software Prototype 1	Frontend and backend connected with basic usability, i.e. information sent from the front end is stored in database.	Sandy, Breanna	1/31/25
Hardware Prototype 1	Basic concept of hardware with a single sensor implemented and communication capabilities.	Gracie, Mak	2/10/25
Software Prototype 2	Symptom logging implemented, backend updated to handle the new data. Extremely simple algorithm implemented (proof of concept).	Sandy, Breanna	2/11/25
Project Presentation 1	Two-minute video about the design and early prototypes	All	2/12/25
Project Report 1	Submit report covering early design and prototype decisions	All	2/13/25
Overall Prototype 1	Connect software to hardware, limited functionality, minor bugs, documentation of changes to make for prototype 2	All	2/22/25
Hardware Prototype 2	Sensor data transfer complete, sensor, button, and display programming complete. Begin programming warning system	Gracie, Mak	2/22/25
Software Prototype 3	Use information about hardware to implement necessary functionality to get sensor data.	Sandy, Breanna	2/22/25

	Test new algorithms for symptom prediction.		
Overall Prototype 2	Implement rudimentary versions of the four major functionalities. Have sensors on BLE & semi-functional. Document changes still needed and bugs that need to be fixed.	All	2/29/25
Hardware Prototype 3	Ssensor data transfers are fully functional. The alarm system is fully functional. Outside casing is complete.	Gracie, Mak	3/1/25
Software Prototype 4	Make final decisions about algorithm, finish features, and work on polishing aesthetic presentation.	Sandy, Breanna	3/1/25
Project Presentation 2	Five-minute video presentation about MVP	All	3/5/25
Project Report 2	Submit report covering current work and decisions	All	3/6/25
Overall Prototype 3	Finish basic prototype; either begin fixing bugs/small tweaks or implementing want features.	All	3/15/25
Project Presentation 3	Seven minute video presentation about project pre-release	All	4/2/25
Project Report 3	Submit report covering current work, prototype, and design choices.	All	4/3/25
FINAL PROTOTYPE OVERALL	Have the project fully complete and functional; execute stress tests and edge cases and fix anything too likely to occur.	All	4/12/25
Final Poster and Pitch	One minute video pitching our final prototype; a poster outlining our design and process	All	4/23/25
Project Presentation	Seven-minute product release video showing off our final	All	4/30/25

	prototype		
Final Report	Finishing report capturing the entire process and outcome of our project.	All	5/1/25

Budget

When creating a university-funded senior design project, it was important to consider budgeting. When budgeting project expenditures, we considered the quality, quantity, and price of the hardware chosen. We aimed for a budget of around $500, and we were able to stay below that in case of any hardware required in the future. We also made sure to account/estimate human hours on this project as if this were a real-life scenario and engineers on the team had to be paid for.

Table #5: Accounting of Project Expenditures

Hardware	Quantity	Cost
Adafruit Feather nRF52832 Bluefruit Microcontroller	3	$85.5
Adafruit BME680 (Environmental Sensor)	2	$38
MAX30102 (Health Sensor)	2	$13
Adafruit 0.96" 128x64 OLED Display (SSD1306)	2	$35
LiPo 3.7 V battery connector type: JST-PHR-02 (2000 mAh)	2	$26
LiPo 3.7 V battery connector type: PH2.0A (5000 mAh)	2	$52
ESP8266 WiFi Transceiver	2	$14

ESP32-CAM	2	$20
Total:	16	$283.50

Note: the hardware cost is $127.50 per watch unit.

Note: Replacement nRF52832 must be ordered due to short-circuit during testing.

Table #6: Estimated Human Hours of the Whole Project

Milestone	Date Done	Human Hours
Video Presentation 1 min	9/20/24	2 x 4 = 8
Video Presentation 3 min	9/27/24	2 x 4 = 8
Video Presentation 7 min	10/4/24	3 x 4 = 12
Video Presentation 10 min	11/8/24	3 x 4 = 12
Software Prototype 1	1/7/25	30 x 2 = 60
Hardware Prototype 1	1/14/25	30 x 2 = 60
Software Prototype 2	1/14/25	30 x 2 = 60
Hardware Prototype 2	2/1/25	30 x 2 = 60
Software Prototype 3	2/1/25	30 x 2 = 60
Project Presentation 1	2/12/25	10 x 4 = 40
Project Report 1	2/13/25	10 x 4 = 40
Hardware Prototype 3	3/1/25	30 x 2 = 60
Software Prototype 4	3/1/25	30 x 2 = 60
Project Presentation 2	3/5/25	10 x 4 = 40
Project Report 2	3/6/25	10 x 4 = 40
Prototype 2	3/6/25	15 x 4 = 60
Overall Prototype	3/15/25	15 x 4 = 60
Project Presentation 3	4/2/25	10 x 4 = 40
Project Report 3	4/3/25	10 x 4 = 40
Final Prototype	4/12/5	15 x 4 = 60
Final Poster & Pitch	4/23/25	10 x 4 = 40
Final Presentation	4/30/25	15 x 4 = 60
Final Report	5/1/25	10 x 4 = 40
Total Human Hours	-	1220

References

[1] “Climate Change & Allergies: Can Temperature Changes Worsen Allergies?”, allegra.com. Can Temperature Changes Worsen Allergies? | Allegra

[2] “Sudden Temperature Changes Can Trigger Your Allergies”, weather.com. Sudden Temperature Changes Can Trigger Your Allergies | The Weather Channel

[3] “Seasons in the USA”, seasonsyear.com. Seasons in the USA: Weather and Climate

[4] “Firebase Realtime Database”, firebase.google.com. Google Firebase

[5] “Accurate Weather Forecasts for Any Location”, open-meteo.com. Weather API

[6] “The Best and Worst Weather for Your Seasonal Allergies”, goodrx.com. The Best and Worst Weather for Your Seasonal Allergies

[7] “Weather conditions and climate change have an effect on allergies”, springer.com Weather conditions and climate change have an effect on allergies | Allergo Journal International

[8] “The role of environmental factors in allergy: A critical reappraisal”, onlinelibrary.wiley.com. The role of environmental factors in allergy: A critical reappraisal - Gilles - 2018 - Experimental Dermatology - Wiley Online Library

[9] “World Allergy Organization Guidelines for the Assessment and Management of Anaphylaxis”, sciencedirect.com. World Allergy Organization Guidelines for the Assessment and Management of Anaphylaxis - ScienceDirect

[10] “Allergy Amulet,”allergyamulet.com. Allergy Amulet

[11] “Understanding Weather Change and Allergies”, Enticare. Understanding Weather Change and Allergies