

SYSC 4907 Engineering Project

Dr. Bruce Wallace, PhD

Dr. Fateme Rajabiyazdi, PhD

Medical Advisor Dr. Andrew Frank

Final Report

Emergency GPS Tracker

Christopher Semaan, 101140813

Anthony Massaad, 101150282

Nicolas Tanouchev, 101143947

Cory Helm, 101171699

April 10th, 2024

Abstract

It is shown that almost 40% of people over the age of 65 experience some form of memory loss [1]. Unfortunately, dementia can even develop in people in their 40s and 50s [1]. This demonstrates how common dementia is and shows how important it is to have adequate measures in caring for people with dementia. A major concern for a person living with dementia (PLWD) is wandering, a case in which a person living with dementia becomes confused about their location that may lead them to becoming lost [2]. While various tracking applications exist such as phones or wearable devices to monitor individuals, they all lack the crucial requirement of battery life. A PLWD would unlikely consistently charge such device, rendering it useless and defeating the purpose of having a trackable device. This proposed project targets this problem as it will operate with minimal power consumption while utilizing various modules including GPS, Bluetooth, and an LTE. Overall, the minimal battery life goal will be approximately 6 months. In addition to the hardware, caretakers can leverage the mobile application to remotely monitor their loved ones wearing the emergency tracker device.

Table of Contents

Abstract	2
Table of Figures	5
Table of Tables	7
1.0 Introduction	8
1.1 Motivation, Background and Problem Statement	8
1.2 Current Market Solutions	10
1.3 Proposed Solution	11
1.4 Accomplishments	13
1.5 Overview of Report	13
2.0 The Engineering Project	15
2.1 Health and Safety	15
2.2 Engineering Professionalism	15
2.3 Project Management	16
2.4 Justification of Suitability for Degree Program	18
2.5 Individual Contributions	19
2.5.1 Project Contributions	19
2.5.2 Report Contributions	19
3.0 Requirements	20
3.1 Emergency Tracker Embedded System Requirements	20
3.2.1 Microcontroller Requirements	22
3.2.2 Bluetooth Requirements	22
3.2.3 GPS Requirements	23
3.2.4 LTE Requirements	23
3.2 Mobile App Requirements	23
Functional Requirements	25
Non-Functional Requirements	25
4.0 Design Overview	26
4.1 Mobile App Overview	26
4.1.1 Framework Decision	26
4.1.2 Codebase Organization	27
4.1.3 Front-End App Design	28

4.1.4 Back-End App Design.....	31
4.2 Embedded System Overview	33
4.2.1 Overall Design	33
4.2.2 Microcontroller Embedded System Design.....	35
4.2.3 GPS Embedded System Design.....	36
4.2.4 Bluetooth Embedded System Design	37
4.2.5 LTE Embedded System Design.....	38
4.3 Integration Overview of Hardware/Software with the Mobile App	38
5.0 Project Implementation	40
5.1 Embedded Systems Implementation	40
5.1.1 Bluetooth Location Tracking.....	40
5.1.2 GPS Location Tracking	41
5.1.3 MCU UART Communication Setup	42
5.1.4 MCU UART GPS Location Tracking.....	44
5.2 Mobile App Implementation.....	46
5.2.1 Front-End.....	46
5.2.2 Back-End.....	53
6.0 Testing	59
6.1 PCB Testing	59
6.1.1 GPIO Issues.....	60
6.1.2 Power Issues.....	63
6.2 Potential PCB Fixes	64
6.3 Embedded System Testing	65
7.0 Conclusion	66
References.....	69
Appendix	73

Table of Figures

Figure 1 Communication Paths for the Tracker	12
Figure 2 GitHub Repository layout	17
Figure 3 Discord Server	18
Figure 4 Overview of Tier of Power Consumption	20
Figure 5 File structure	27
Figure 6 User Interface for the Sign in Page	28
Figure 7 User Interface for Registering	28
Figure 8 Home page displaying the list of caregivers PLWD	29
Figure 9 Home page display the removal of the list of caregivers PLWD	29
Figure 10 Example of the Settings page	30
Figure 11 PLWD information in the app	31
Figure 12 The project database structure	32
Figure 13 Simplified Overview diagram of communication of modules on the PCB	34
Figure 14 Picture of the Printed Circuit Board	35
Figure 15 Apple AirTag communication path	37
Figure 16 ESP32 Location tracking setup	40
Figure 17 Find My App showing location of ESP32	41
Figure 18 Successful location fix in Serial Monitor using CAM M8C and Arduino Uno	42
Figure 19 UART example setup on the STM32CubeIDE	43
Figure 20 UART example setup on the STM32CubeIDE	43
Figure 21 Example Transmit and Receive from UART4	44
Figure 22 UART establishment between the GPS and MCU	44
Figure 23 STM32 code for retrieving GPS location	45

Figure 24 Firebase configuration variables.....	46
Figure 25 The Login page	47
Figure 26 The Registration page.....	47
Figure 27 The home page.....	48
Figure 28 The home page when delete button is pressed	48
Figure 29 Adding a new PLWD prompt for the user	49
Figure 30 Adding input to the new PLWD prompt.....	49
Figure 31 The settings page	50
Figure 32 The PLWD individual page.....	51
Figure 33 Safe-Zone page for configuring the home location for a PLWD	52
Figure 34 Configure radius dialog in the safe-zone page	53
Figure 35 Firebase Authentication service.....	54
Figure 36 The collection of PLWD that links from the tracker hardware	55
Figure 37 The collection of Coordinates for each tracker hardware.....	56
Figure 38 The collection of safe-zone per user	57
Figure 39 The collection of users associated with an account.....	58
Figure 40 Picture of the blank version of the PCB	59
Figure 41 MCU header schematics for the PCB	60
Figure 42 Exposed GPIO pins on the PCB.....	61
Figure 43 Schematic of GPS IO pin connections to MCU	62
Figure 44 Schematic of power switch for the GPS module on the PCB	63
Figure 45 OpenHaystack example of multiple devices connected and being tracked on the map [26]	75

Table of Tables

Table 1 Project Tasks Contributions	19
Table 2 Final Report Contributions.....	19
Table 3 Unit Tests for each Embedded System Module	65

1.0 Introduction

1.1 Motivation, Background and Problem Statement

The motivation of the project is to present a solution in caring for a person living with dementia.

Dementia is a neurological disorder that is progressive and mainly effects adults. Statistically, almost 40% of people over the age of 65 experience some form of memory loss [1].

Unfortunately, dementia can even develop in people in their 40s and 50s [1]. This demonstrates how common dementia is and shows how important it is to have adequate measures in caring for people living with dementia. A 2011 study conducted in Canada revealed that 340,000 people were living with dementia in Canada alone, with the number projected to be about double by 2031 [3]. It was also projected that of the doubled amount, older people would make up of 1.55-fold increase in the dementia population growth [3]. With these statistics in mind, it becomes apparent that the prevalence of dementia will inevitably increase and place a large strain on families and the health care system to aid PLWD. People living with dementia become unable to live independently due to the negative physical and mental symptoms that arise and require increasing levels of aid and care [4]. Furthermore, due to the lack of available treatments for dementia and the increasing number of people living with dementia, the need for palliative care will only grow [5]. This only increases the need for new healthcare innovations and technology to help the negative impact of dementia.

The negative impacts of dementia include cognitive decline, memory loss, and various behavioral changes [1]. Among these symptoms come a severe risk of wandering. Wandering is a case in which a person living with dementia becomes confused about their location and

begins to travel resulting in them becoming lost [2]. This disoriented behavior can lead to dangerous situations, resulting in injury or, worst case, even loss of life. This can occur both in areas they have lived in for years, or places they visit for only a couple minutes. Unfortunately, there is no clear reason behind a PLWD's wandering, and it is quite difficult to prevent these occurrences entirely [6]. Family members can attempt to contain their loved one indoors using locks or through other means. They may also use security systems such as cameras or sensors to stay alert while caretaking for their loved one. However, due to the unpredictable behavior, the safety of the PLWD is never guaranteed.

Due to these symptoms of dementia increasing with age, the load and negative effects on the people close to those living with dementia also increases. Care extends beyond hands-on care to include the following: monitoring and supervising, preserving the individual's sense of self, anticipating future support needs, and helping the individual develop new and valued "jobs" to do [7]. These extra factors can have a harsh and lasting effect on caregivers. In a study done to understand the effects dementia and the healthcare system has on caregivers, 23 caregivers participated in 3 focus groups [7]. The results of this analyses revealed two thematic areas. First, the caregiver feels forgotten or abandoned to care for the PLWD alone indefinitely, and second, that they face unrealistic expectations.

As mentioned, caregivers are often feeling left and forgotten. They frequently made examples of the way the system was too overburdened to help. Family caregivers stated that nobody would ever call them back or update them, with many caregivers feeling that they must fight to get the help they need [7]. Taking care of another person is no easy task, with many feeling that the unrealistic expectation of self-care is an added responsibility that affects their own well-being.

As mentioned above, the condition of a PLWD worsens as the person ages, resulting in the need for increasing care from a dedicated caretaker. Because of this, the amount of unpredictability and lack of modern solutions results in more time and finance in caretaking at home, leading to depression and other means in protecting the PLWD. In fact, it is expected that the caretaker's burden is related to the severity of the care recipient's health, making it more likely for the caregiver to rely on long-term care facilities [8]. Families taking care of a PLWD need support in any area possible, exemplifying the need for technology that relieves that burden.

1.2 Current Market Solutions

Given the number of people affected by dementia, there are a number of solutions that exist on the market currently to track a PLWD. Some tracking devices employ a GPS, using satellite signals to determine the location of the wearer [9]. This includes devices such as the AngelSense tracker [10] and the Jibit Location Monitor by Life360 [11]. These GPS tracking devices can be quite effective for locating users outdoors. Unfortunately, the GPS is demanding on the system's battery life. As a result, these systems need to be charged often. Other tracking devices on the market utilize a Bluetooth signal to locate the PLWD. Devices such as the Apple AirTag [12], while not designed explicitly for those living with dementia, can be used to track the location of the user. Unlike the GPS devices, Bluetooth low energy (BLE) devices are very energy efficient, with batteries being able to last for a full year if not longer. Unfortunately, the range on these devices is limited and is not well suited for PLWD who may travel far [9]. The final type of location trackers that currently exist make use of radio frequency (FR). These devices, such as Project Lifesaver, use radio signals to locate the device [12]. The FR devices offer a medium

between the GPS and BLE systems, as their range is larger than the BLE trackers, but shorter than the GPS trackers [9].

While it is great that so many solutions currently exist on the market, they each offer up shortcomings that can prove to be costly when caring for a PLWD.

1.3 Proposed Solution

The solution we propose is to create the proof of concept for a wearable emergency tracker for people living with dementia. The purpose of the tracker, that will be worn by the PLWD, is to ensure the safety of the PLWD by allowing the caretaker to check on their location status. The main focus of the proof of concept will be the tracker hardware, which is represented by individual development boards. The goal for the tracker hardware is to have a complete embedded system that will be able to collect location information while consuming as little power as possible. The other focus of the proof of concept is the accompanying mobile app. The mobile app will have full communication with the tracker hardware that will be able to inform the caretakers of the status of their loved one in real time. This proof of concept will help caretakers with the stress of caring for a PLWD by providing them ease of mind that they can find their loved one.

The overarching goal with the project is that the tracker would be deemed useless if the battery usage is not optimal. Therefore, our unique feature for the tracker hardware is that the battery will last long term. The goal is to minimize the need to remove the tracker, as this will create situations where the tracker is forgotten and not being worn as needed. A long-term battery life for the tracker is 3 to 6 months before requiring replacement.

The tracker's hardware system will include multiple communication and location tracking modules. Having multiple communication options will allow for optimal power-consumption based on the availability of each module in every situation. The modules that will be available for communication are Bluetooth (BLE), and Cellular (LTE). The modules that will be available for location tracking are Bluetooth, using Apple's Find My network and Global Positioning System (GPS). As the tracker establishes its communication and network type, it will be able to relay its data through a cloud database that the mobile app will track and update in real time. This will ensure the caretaker is kept up to date on the safety of their loved one. Figure 1 below represents the communication paths between the mobile app and the tracker hardware. It demonstrates how the tracker hardware obtains its location data, either via GPS or Bluetooth, and how both the tracker and mobile app establish communication with each other through the cloud database.

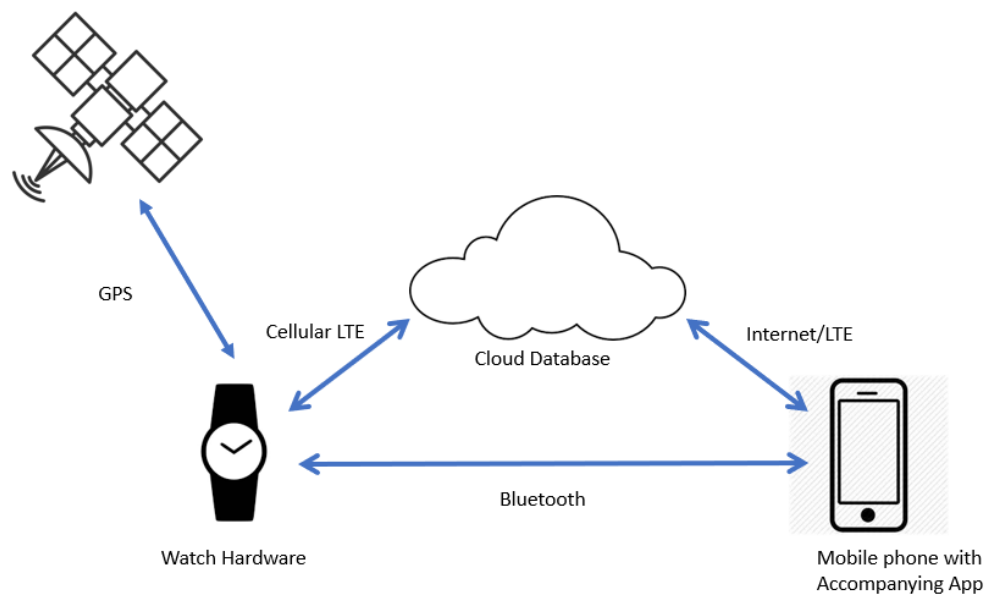


Figure 1 Communication Paths for the Tracker

1.4 Accomplishments

In terms of embedded systems development for the tracker hardware, the team was able to create functionality for the following modules using the development boards: Microcontroller (MCU), Bluetooth and GPS. For the GPS and Bluetooth modules, we created location tracking capabilities that work with the development boards connected to a host computer for debugging. For the MCU, we set up communication capabilities to all the other modules. This set up allows for the microcontroller to be used as a central unit to send commands to retrieve location data from the other modules.

The team spent a large portion of the term testing and debugging the PCB that was designed prior to this term. The team was able to find and fix multiple design flaws. One type of flaw includes issues with access to GPIO ports on the MCU. The other type of flaw includes power issues such as unwanted short circuits and overcurrents.

For the app development, we were able to create a complete mobile app that works on Apple and Android devices for use by caretakers of a PLWD. The necessary functionality for this app includes account registration and login for the caretaker, adding a loved one living with dementia to the application and a map feature to view last known location. The mobile app is accompanied by a complete backend that includes a cloud database to store user data.

1.5 Overview of Report

The overall report will contain information about the technical aspect of the project, which includes the privacy risks and requirements necessary to complete the project. It will also go

into detail about the group's progress over the year, demonstrating the work that has been accomplished and illustrate any problems that needed to be solved along the way.

The report will start with critical information on the impact of health and safety, and some of the precautions taken to ensure efficiency of work throughout the term. Deep diving further in the report includes the project requirements describing the functional and non-functional aspect of project, the design process of what is expected of the final product, as well as the implementation of what we were able accomplish, testing done throughout implementation and the overall milestones of what was expected to be done. At the end, the report will conclude with a reflection from the team which includes a summary of problems that were faced and the potential next steps of the project.

2.0 The Engineering Project

2.1 Health and Safety

While there were no serious and obvious health and safety concerns, the team still took all necessary precautions to ensure that the project could progress as smoothly as possible. With the work being done on the software side of the project being safe and harmless, we really focused on considering any and all health and safety concerns on the hardware portion of the tasks. We made sure to follow all general health and safety principles [13], like keeping food and beverages away from the development areas, to both protect ourselves and the equipment we were working with. In the same vein, we worked with unpowered development boards and PCBs when possible and made sure to check the boards for potential shorts before powering them on. This way we could protect the boards from potential damage and ourselves from the fallout of the potential damage. Our last major safety precaution was to utilize the knowledge and expertise of our lab technician when making modifications to the PCBs. We had detailed discussions with the lab technician about what changes we wanted to make and how they could be done safely, and had the lab technician do the soldering on the boards, as none of the members of the team had previous experience doing so.

2.2 Engineering Professionalism

To adhere to proper engineering professionalism, the team followed a complete engineering process [14] throughout the project. The process began with a project proposal, detailing the objectives, tasks, and methodologies the team set out to achieve, summarized into a timetable with distinct milestones. A progress report was then written halfway through the project,

explaining the state of the project, the work that had been completed, and the issues that arose. To wrap up the project, at least for the current team, this final report has been produced, building on the project proposal and progress report. The final report aims to report on all completed progress and the current state of the project, the issues the team has encountered throughout the year, and what work still needs to be done in the future, in the hope of having a smooth handoff to the succeeding team.

On a day-to-day basis, the team employed the ideas of an agile work environment to encourage a steady stream of progress and communication within the team. Weekly standup team meetings were held every Wednesday, allowing the team to discuss the current state of affairs, including both what had been accomplished in the prior week and what problems arose. These weekly updates allowed the team to determine whether the timeline for the milestones set in the project proposal was being met or if actions were needed to address the current situation. The frequent meetings allowed the team to react when there were problems and efforts could therefore be focused towards finding a solution so that the project could continue to progress.

2.3 Project Management

For code management and storage, the group was using GitHub, which is a version control application that allows for code branching and code history storage as a repository. We also used GitHub as our project task management tool to keep track of task scheduling and assignments. The GitHub Projects tool allows for linking tasks directly to repositories and opened code issues. The Figure 2 shows the overview of the organization of our GitHub repository.

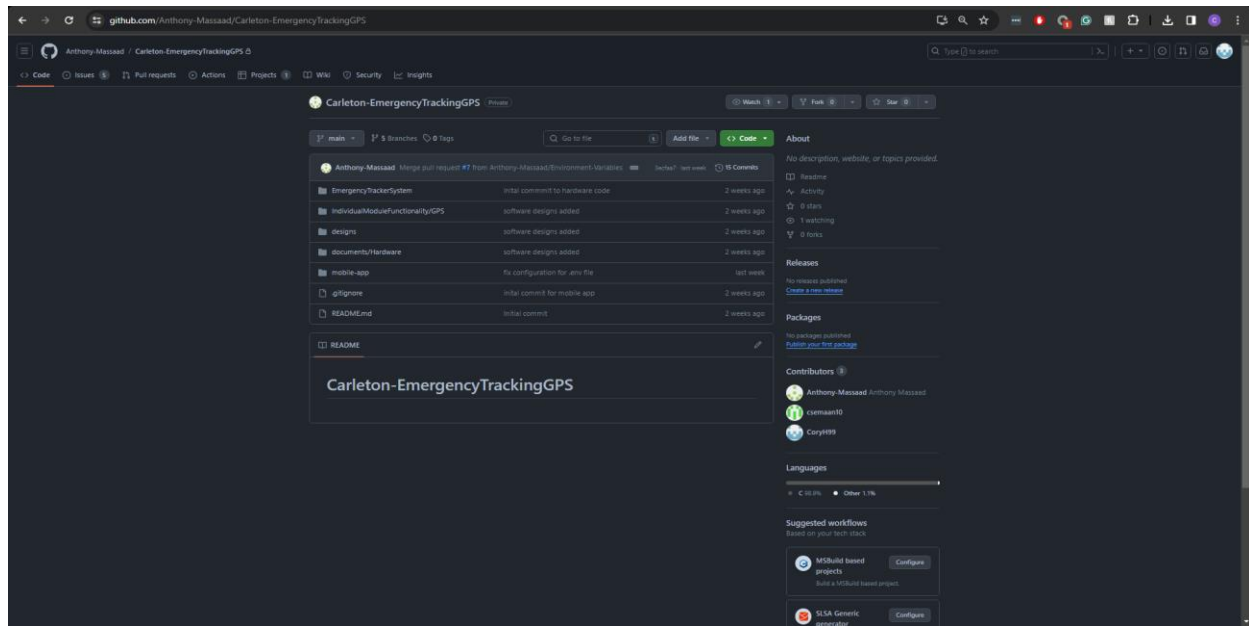


Figure 2 GitHub Repository layout

For communication, the team used Discord for text communication and voice meetings. It's been organized to keep history of each milestone and meeting minutes, which is shown below in Figure 3. The team met once a week on Discord for group programming time or other project management tasks. The team also met once a week in person for tasks related to tracker hardware.

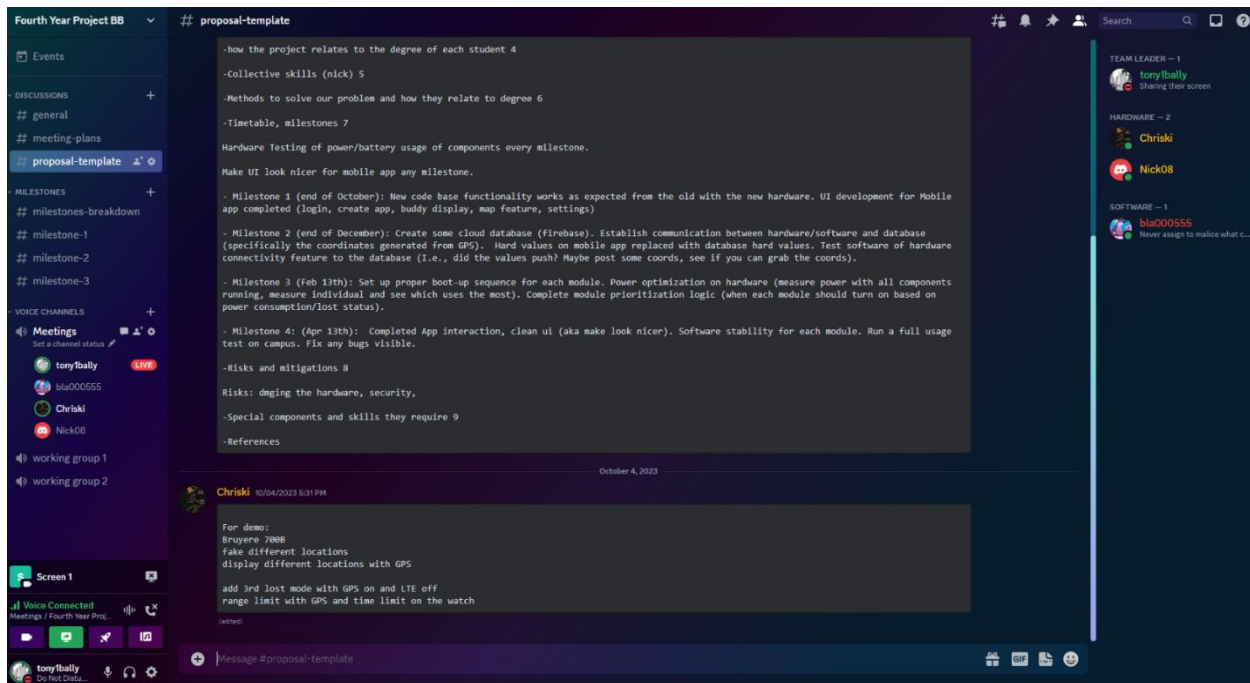


Figure 3 Discord Server

2.4 Justification of Suitability for Degree Program

This project relates to the knowledge learned in each group member's degree. The team consists of two Software engineers, Tony and Cory, as well as two Computer Systems engineers, Chris and Nicolas. This provided the group with a wide variety of skills. Both engineering disciplines allowed for the team to build strong problem-solving, structuring and organization skills. Because of the range of skills learned in class, our group has the resources and knowledge necessary, for example to identify and establish how the communication between the hardware and software will work. Our software engineering team members were also able to design and develop an in-depth mobile app that ties into the hardware and software. Finally, our team has also had plentiful experience in group and teamwork environments improving the team's ability to communicate and work together towards a common goal.

2.5 Individual Contributions

2.5.1 Project Contributions

Table 1 Project Tasks Contributions

Name	Tasks
Chris	GPS Location Tracking, PCB Testing and Debugging, Bluetooth Location Tracking, Microcontroller Communication
Cory	App Development, Back-End Development
Nick	PCB Testing and Debugging, GPS Location Tracking
Tony	App Development, Back-End Development, Bluetooth Location Tracking, Microcontroller Communication, PCB Testing

2.5.2 Report Contributions

Table 2 Final Report Contributions

Name	Section
Chris	1.4, 2.3, 3.2, 4.2, 4.3, 5.1, 6.1, 6.2, 6.3
Cory	1.1, 2.3, 2.4
Nick	1.1, 1.2, 2.1, 2.2, 6.1
Tony	Abstract, 1.5, 3.1, 4.1, 5.1, 5.2, 7.0

3.0 Requirements

3.1 Emergency Tracker Embedded System Requirements

The overall solution for the emergency tracker requires a low power-consumption embedded system to track location of a person living with dementia. To create a power efficient system, our design requires multiple location and communication options. Having multiple options allows the system to decide which module to use based on availability of location data and power efficiency of the modules.

Based on the project's previous contributors, the design requires separate tiers of power usage to decide which module to use in which situation. The tiers are separated based on their levels of power efficiency and location information availability. The default state of the embedded system would be in an idle mode. Idle mode would mean that the system is in sleep to conserve energy. In all other modes, the MCU module is on to coordinate all other modules. Figure 4 shows the overview for the tiers below.

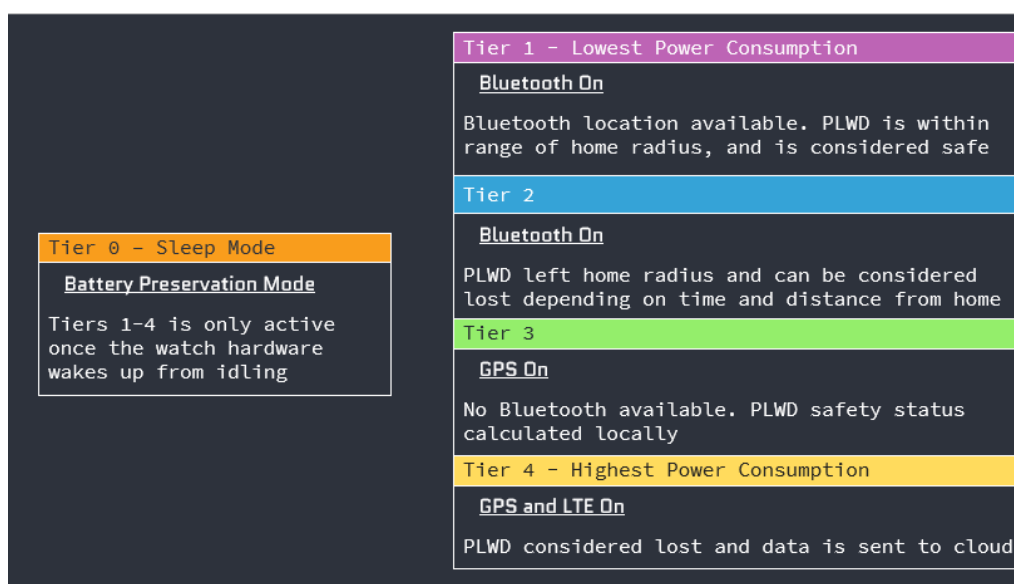


Figure 4 Overview of Tier of Power Consumption

The first tier is when the Bluetooth module is on and there is Bluetooth location information available. The tracker is found to be within the range of home or a trusted device. This tier is the lowest power consumption mode, and the PLWD will be considered safe while in this tier. The Bluetooth location information comes from the Apple Find My network, which utilizes nearby Apple devices to get the location data.

Tier two is triggered when the tracker is outside the selected home or trusted person's range. However, the tracker is continuing to use only the Bluetooth module to track the PLWD's current location. In this tier, the PLWD can be considered lost if they are outside the range of their safe zone or if the time spent outside the home exceeds a certain amount of time.

Tier three of power consumption is when there is no Bluetooth information available to the tracker hardware, so the GPS module is activated to confirm the location of the PLWD. Like tier two, the PLWD will be considered safe in this tier if they are within a set range from home or have been away from home for less than a set time limit. The lost status of the PLWD will be calculated locally on the tracker in this tier.

Tier four is the least power efficient mode and uses both the GPS and LTE modules. This tier is triggered on the device if the GPS information from the tracker is far from home and the set time limit outside their home has elapsed. This tier is activated when the PLWD is considered lost, and the tracker must transmit GPS data to the caretaker.

These separate tiers outline the requirements for each module's development discussed in the following subsections.

3.2.1 Microcontroller Requirements

The microcontroller module has the large responsibility of coordinating all the modules by selecting which modules should be powered to minimize power consumption. The MCU needs to be able to send commands to the modules to retrieve location data and module status information. To do both of these tasks, the MCU needs to have multiple communication protocols available to coordinate with multiple modules at the same time.

The MCU also requires data processing capabilities for handling all data from the other modules. The MCU is required to handle serial communication to handle the data received. The MCU also is required to be able to parse the data to a usable format to be read by other parts of the embedded system. The MCU should be able to compare location data to the PLWD's safe location to confirm their safety status.

3.2.2 Bluetooth Requirements

As mentioned, the Bluetooth module is required to have both communication and location tracking capabilities, which is enabled by the transmission of Bluetooth pings to nearby devices. Utilizing the Apple's find My Network, the tracker will send Bluetooth pings to Apple devices nearby through their Bluetooth and those devices will send the location data to the cloud. The communication between the Bluetooth module and Apple Find My network is required to be secure by using public key encryption.

The Bluetooth module needs to have serial communication capabilities to connect to the MCU. This is necessary for the module to communicate its power status and to receive commands from the microcontroller.

3.2.3 GPS Requirements

The GPS module is the backup location tracking method when Bluetooth is not available. The GPS needs to be more reliable when no Apple or Bluetooth device is nearby. The module should take advantage of the international GNSS positioning system to be available to as many satellites as possible.

The GPS module also needs serial communication capabilities to communicate location data to the MCU. Similar to the Bluetooth module, it needs to be able to communicate its power status and receive other commands from the MCU.

3.2.4 LTE Requirements

The LTE module needs to be able to connect to the cellular network. This means it needs a SIM card slot that is accessible to Canadian SIMs. It needs to be able to receive data from the MCU to send over the cellular network to our cloud database. To receive the data from the MCU, the LTE requires serial communication protocols available.

3.2 Mobile App Requirements

The mobile application's responsibility is to complement the tracker hardware and communicate to the caretaker details about their loved ones. It will have features that will allow caretakers to have extensive information regarding the PLWD, given from the tracking hardware. Key features that the mobile app will encompass includes viewing all the caretaker's loved one's and their safety status, GPS map integration that will show a live feed if the PLWD is lost, and a safe-zone location. Overall, the mobile application will be compatible with all types of operating systems including Android and IOS.

Starting with the display of the caretaker's loved ones and their safety status, this feature will be displayed as the home screen as the first information shown to the caretaker. It will have a list of PLWD associated with the caretaker and for each will show their safety status. In regards of the safety status, it will be a badge on the app signifying if the PLWD is "lost" or "safe", which is determined on the data received from the tracker hardware.

The GPS map integrated feature will belong onto another page that the caretaker can access if they select their loved ones from the home page. It will display number of information including the PLWD's name, last known location, safety status and a map that will have a marker of their last known location. While the PLWD is "lost" and the tracker hardware is active collecting location data, the mobile application will have a live feed and update as new information is communicated to it. In addition to the live feed, the caretaker will be able to open their device map application with the coordinates from the app.

The last key feature of the app is the ability to set a safe radius where the caretaker deems their loved one to be a safe distance from home. It is a designated safe location defined by the caretaker that can be set from the app for the tracker hardware to recognize from and determine if the PLWD is indeed "lost". An example of a safe location is a block radius around the household of the caretaker. If the loved one wishes to walk around the block, all aspects of the tracker will not deem the PLWD to be lost until they are outside of the safe location defined by the caretaker.

Overall, the mobile app must enable caretakers to verify the safety status of their loved ones by confirming if they are within the designated safe locations defined by the caretaker. The app will

also be capable of GPS tracking in real time, as well as be available on multiple devices such as Android and IOS. Lastly, additional enhancements such as notifications and app personalization will be considered to ensure a more robust and user-tailored experience.

Functional Requirements

Functional requirements are the aspects of the system that defines the specific behavior or functions of a system. It's an aspect that defines how well the system should perform to satisfy the users' needs and achieve its intended purpose. Overall, it describes what the system should do, rather than how it should it [15].

In terms of functional requirements for the mobile app, users should be able to register, log in, view the PLWD they are caretaking, access detailed information and location coordinates of the PLWD, set the safe zone for the PLWD, and lastly configure settings for customization and accessibility. The user interface should consist of six pages overall including login, registration, view all associated PLWDs to the caretaker, viewing individual PLWD information, configuring the safe zone for the PLWD, and settings configuration.

Non-Functional Requirements

Non-functional requirements are the aspects of the system that describes the general properties of the system, rather than how it should operate. It's the quality attributes that establishes constrains and behavior [15].

In regards for non-functional requirements, the app must prioritize security, scalability, reliability, maintainability, and performance. Given the sensitive nature of the data involved in the project, including passwords and people's coordinates, Firebase, a cloud database

supported by google, will be utilized for authentication and data storage. Firebase's built-in authentication service also includes password hashing ensuring data security. The app is expected to perform quickly and reliably while accommodating any number of users. Additionally, it is expected for the app to provide instant feedback to further ensure its reliability for the caretaker. For developers, the codebase will be required to be easily maintainable with consideration given to folder and file structuring for future developer comprehension.

4.0 Design Overview

4.1 Mobile App Overview

4.1.1 Framework Decision

The first step in designing the mobile app was establishing what framework will be used, for which the team has decided it would be using react-native [16]. React-native is a popular open-source framework for building cross-platform mobile applications, allowing the team to create the app for both Apple and Android. Additionally, we will be complementing react-native with the Expo ecosystem [17], which provides a set of tools and services to simplify development with react-native. For example, Expo's folder structure allows for automatic routing between pages without defining the routes ourselves. Another example is that Expo provides a map feature that the team has deemed necessary feature of the app. Another key reason for using Expo is its ability in bundling the apps for us, which removes the extra steps needed for react-native to run the app.

4.1.2 Codebase Organization

Overall, the codebase will be designed and structured for readability and maintainability. We have the app folder, defined by Expo, which contains the page routes for the mobile application. The assets folder will contain any images, fonts, and icons used in the app. The components folder will be used to encapsulate components, like the home or setting page, used throughout the app. The constants folder is designed to store variable references for all the assets so it can be used in the app. The contexts and globals folders are to globalize variables and styling in the application for simplicity and consistency. Lastly, we will have a .env file that stores the app secrets such as the firebase references, which includes the API token that is sensitive. Overall, the app was structured to make development straight forward. It is also crucial to note that each file will be named according to its purpose. The following figure is an example of what the codebase will look like.

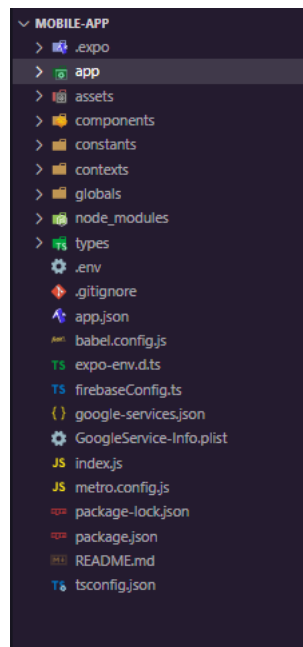
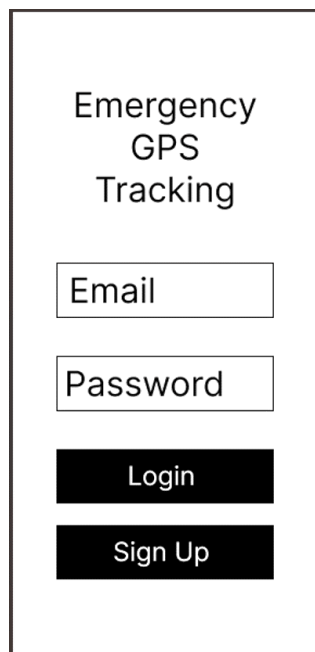


Figure 5 File structure

4.1.3 Front-End App Design

Each of the following page designs was done through Figma. Figma is a collaborative visual design application for web and mobile applications [18].

The user interface is expected to be easy to use and follow. Starting with the login page, it is expected that the user will be able to input their username and password as shown in Figure 6 below. Similarly, for registering a user shown in Figure 7, it is expected that the user is to input their name – username that is displayed on the app, email – for authentication, and password.



Emergency
GPS
Tracking

Email

Password

Login

Sign Up

Figure 6 User Interface for the Sign in Page



Register

Name

Email

Password

Sign Up

Login

Figure 7 User Interface for Registering

The input fields for both designs will clearly dictate the information needed to continue.

Additionally, both designs will allow for navigation to each other. For example, the Sign in page will have a link to navigate to the register page, while the register page will have a link to navigate back to the sign in page.

The home page will be responsible for displaying the list of PLWD for the caretaker, as well as options for adding and removing the people they are caregiving. This can be seen in Figure 8 and 9 respectively.

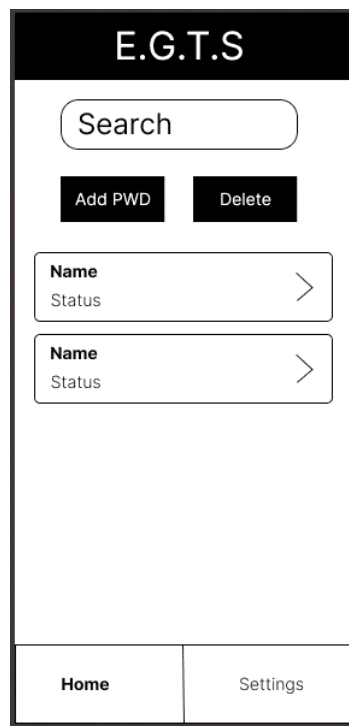


Figure 8 Home page displaying the list of caregivers PLWD

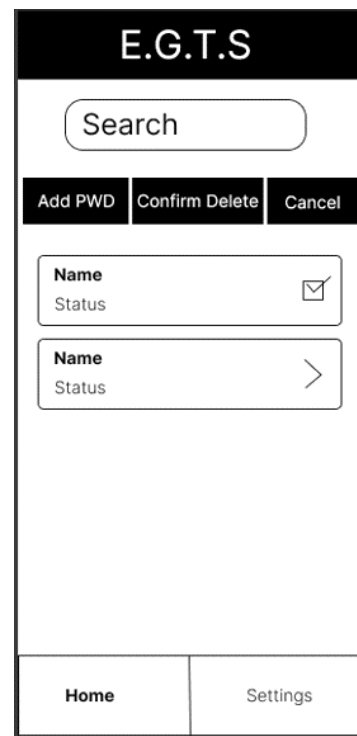


Figure 9 Home page display the removal of the list of caregivers PLWD

It is expected that each loved one has a status to tell the caregiver what the severity of that individual is. So far, we have established “Safe” and “Lost” per individual, but more can be added as the app gets developed. To delete the PLWD tracker status from the app, simply pressing the delete button, select the individuals to delete and confirm.

Furthermore, we plan to make navigating from page to page seamless. Hence, a tab layout, like most modern apps, is ideal to easily switch over between content. So far, the team has only decided to have two tabs. One tab will default to the homepage, while the other tab will contain

settings to apply additional features to the app later. Figure 10 below is an example of what the settings could have. The one major feature the settings page that has been decided for certain will be the ability to log out of the application.

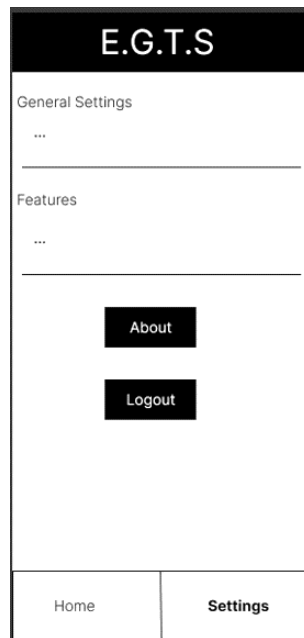


Figure 10 Example of the Settings page

Finally, when selecting any PLWD in the home page, the app will display further information about that loved one as shown in Figure 11 below.

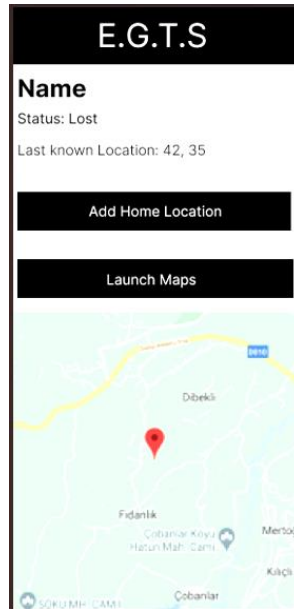


Figure 11 PLWD information in the app

For far, the team has established that the individual's name, severity status, and last known coordinates are essential for the app. It is also expected that the user can launch their mobile map application with the last known coordinates of their loved ones if needed. Additionally, an "Add Home Location" button will be available for every PLWD to specify a range that defines their safe zone.

4.1.4 Back-End App Design

As mentioned above, the app will be communicating with a cloud database to collect the up-to-date information from the hardware. As such, the team has designed the database as shown in Figure 10 with the minimum requirements needed to create the proof of concept.

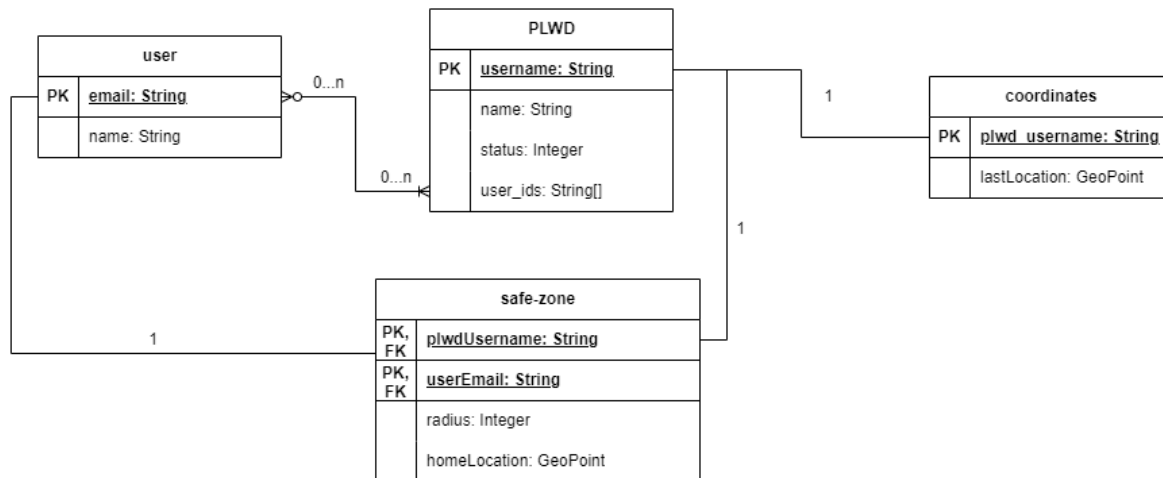


Figure 12 The project database structure

As for now, the user/caregiver is expected to have a username and a password. The username will be unique for every user. For security reasons, the password is required to be hashed, which the Firebase authentication database will provide for us. As for each PLWD, they will have some form of username that is identified from the hardware. This way, the hardware’s association with the app and each user is a unique identifier. Additionally, each PLWD’s data will include their name, their status (Lost or Safe), their Home Location (safe zone), and the user identifier (a user’s username) they are associated with. Finally, each PLWD will have the coordinates of their last know location stored, which will be updated when new location data is found for the individual.

Overall, the team has designed the app to contain the minimum functionality needed to create a working proof of concept with the tracker hardware. However, the mobile app is also designed to scale for newer features and additional data collection, while maintaining the security and accessibility of the app, as the project progresses.

4.2 Embedded System Overview

4.2.1 Overall Design

Our embedded system for our tracker hardware has two versions of hardware. All of development was done on using the individual development boards for all the modules including microcontroller, Bluetooth, cellular and GPS. The other version of our embedded system was the PCB that contains all the modules installed on it. No development was done on the PCB, since the debugging and testing of this board was done throughout these terms.

The software written for the development boards function independently of each other. These individual software modules were written with the future unification of code in mind.

The design goal for the communication between modules was for our microcontroller unit to act as central processing unit as well as our central unit for communication between the other modules. Our final communication design has a complete set up for each module to the MCU using UART communication. Each communication and location tracking module is set up on separate serial ports. The LTE and BLE modules are set to asynchronous communication mode to allow for asynchronous requests. The GPS module is set to synchronous mode on a USART port. The following diagram in Figure 13 shows the communication paths and protocols between each module as well as the chip used for each module.

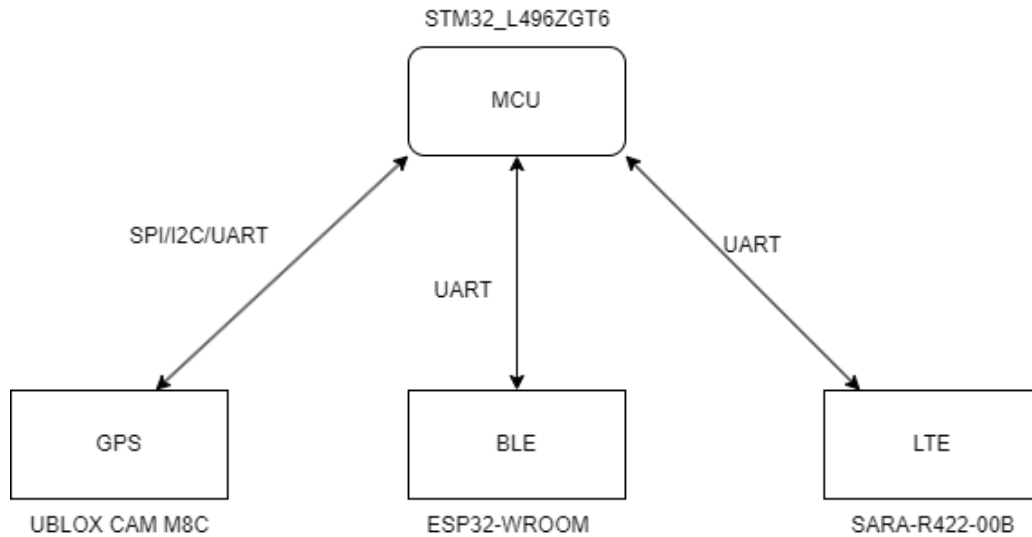


Figure 13 Simplified Overview diagram of communication of modules on the PCB

The above diagram in Figure 13 shows that the MCU is at the centre of all communication.

The other modules do not communicate directly to each other. The set up for this communication using UART is flashed on the MCU itself.

On the PCB itself, these modules are labelled, and all of the connections are available through the traces on the board as seen in Figure 14.

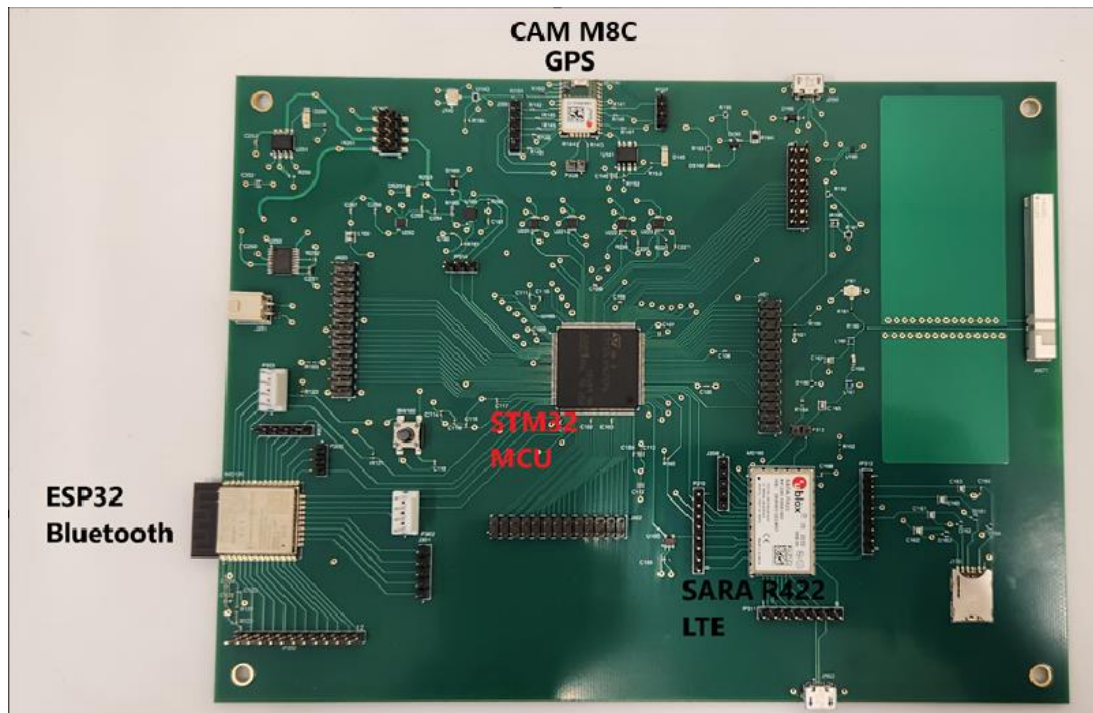


Figure 14 Picture of the Printed Circuit Board

The PCB also has power management units to track battery level and to limit unnecessary power consumption. These units are necessary to preserve the long term battery life requirement. The power management units require no software implementation.

4.2.2 Microcontroller Embedded System Design

The microcontroller in our embedded system is the STM32 [19]. It has the capability for multiple important functions including the coordination of modules and data processing. Our design focus was to set up serial communication that can be configured for all communication and location tracking modules. To program the STM32, we used the STMCubeIDE to allow for simple flashing and debugging on the board. We also used the STMCubeProgrammer application to configure the ST-Link debugger that is attached to the STM32. The final design of the software written for communication on the MCU uses the HAL UART library. This library allows for simple,

small-sized communication to other modules. The library allows for simple configuration for asynchronous transmit and receive messages.

The STM32 has many technical specifications and features that make it a good choice for our embedded system. For communication, it has up to 6 U(S)ART slots available which allow it to communicate to all our other modules at the same time [20]. It also includes SPI and I²C communication interfaces. Our STM32 also comes with an on-board debugger attached (ST-Link) and multiple user LEDs to simplify debugging.

4.2.3 GPS Embedded System Design

The GPS module in our embedded system is the UBLOX CAM M8C. The GPS is one of the location tracking modules available on our design. The GPS has higher power consumption than the Bluetooth module but will be more reliable in getting the location data as location tracking is done locally on the chip. Our final GPS software design utilizes an Arduino Uno in the place of our microcontroller to receive the location data. We used the Arduino to receive the location data as it allows for simple monitoring of the serial information (including our location data).

The software utilizes an NMEA library to parse location data received from the GPS module.

The CAM M8C board allows for different types of communication interfaces (UART, SPI, I²C). Our implementation utilizes I²C to an Arduino Uno [21]. The CAM M8C has a 2.5m accuracy with the GPS satellites and a short 29s time for first location fix. This accuracy and time for location fix are favourable for a tracker that would be used in emergency location tracking.

4.2.4 Bluetooth Embedded System Design

The Bluetooth module is the ESP32, which is our least power consuming module in our design. It has both communication and location tracking capabilities. As mentioned in section 3.1, the Bluetooth module, ESP32, in our design mimics an Apple AirTag that can ping Apple devices, which will then send location information using the Apple Find My network. This method is low on power consumption since the location data is obtained on the Apple device instead of on the Bluetooth chip. The following figure shows the communication path of the location data.

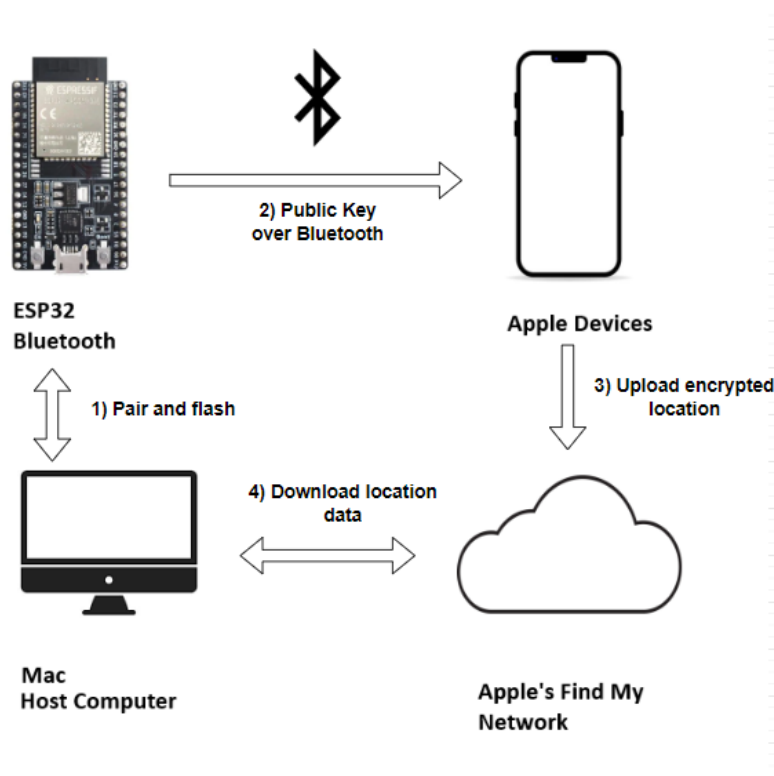


Figure 15 Apple AirTag communication path

The final design uses an open-source library called OpenHayStack to allow for the AirTag mimic behaviour. The library mainly uses C and bash languages for execution.

Since the tracker hardware would rely on apple devices being within 30-40 ft [22], the accuracy of the location data is tied to the Apple device that is pinged. An example of the accuracy of this method is an iPhone 14 or later would have L5 GPS bands available which can have up to sub-meter accuracy [23].

4.2.5 LTE Embedded System Design

The LTE module is the SparkFun Sara R4. The LTE module is our backup communication module as it also has higher power consumption compared to Bluetooth. The LTE module allows for communication to the cellular network. The LTE board has UART communication available, which would allow it to receive commands from a microcontroller board. The Sara R4 board contains one SIM card slot that accepts Canadian and American SIM cards to be able to communicate to the cellular network.

4.3 Integration Overview of Hardware/Software with the Mobile App

The conclusive decision by the team emphasizes the critical necessity of achieving full implementation of both the tracker hardware and the mobile app before initiating communication between the two products. This implementation not only serves as a prerequisite, but also the foundation of a proof of concept. Hence, the integration process between the tracker hardware and the mobile app will be the final step of the proof of concept. At this stage, we expect the tracker hardware to achieve complete synchronous communication with the cloud database. The mobile app will leverage this connection, enabling it to seamlessly collect and process the data sent from the tracker hardware. We expect this connection to allow

the app to not only retrieve information from the cloud database but also respond dynamically to the user's needs.

Additionally, as mentioned in the overview design of the mobile app, we expect to have a specific feature to link the user with their loved one's tracker hardware by using their identification. While this identification is securely stored on the hardware and linked to the database, we will enable the app to recognize the user's loved one based on this distinct identifier.

5.0 Project Implementation

5.1 Embedded Systems Implementation

5.1.1 Bluetooth Location Tracking

The team was able to continue the development of the embedded system using the individual development boards. On the ESP32 Bluetooth development board, we were able to expand on the work that was done before this year to create a more reliable location tracking using the OpenHaystack framework as mentioned in section 4.2.4. The set up of the ESP32 works while plugged into a Mac through USB as seen in the following figure.

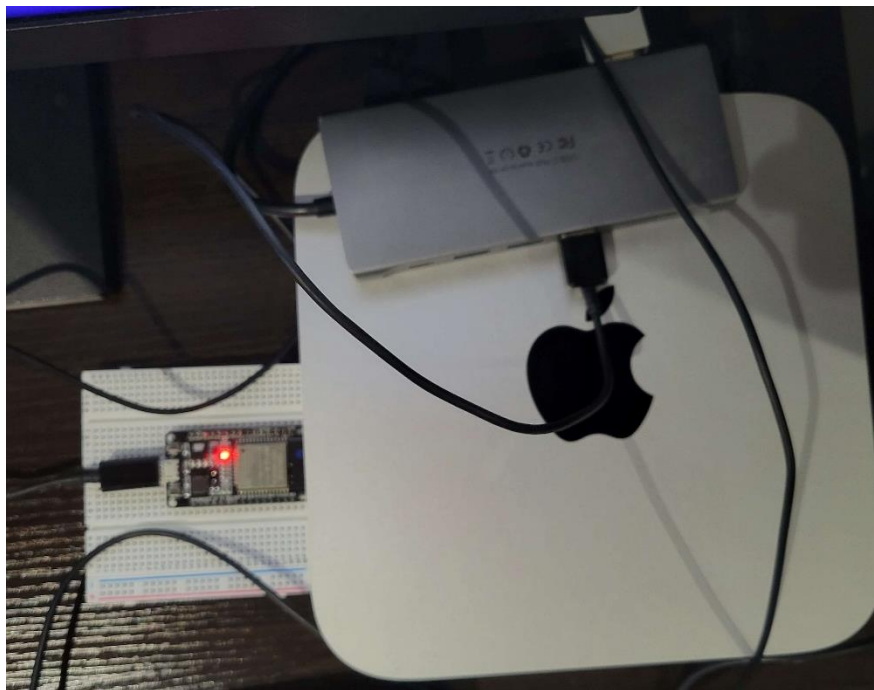


Figure 16 ESP32 Location tracking setup

The OpenHaystack script that is flashed onto the ESP32 allows it to act as a part of the Apple Find My network. In the following figure, the Find My app on the Mac shows the location ping from the ESP32 that was sent to the Apple Find My network.

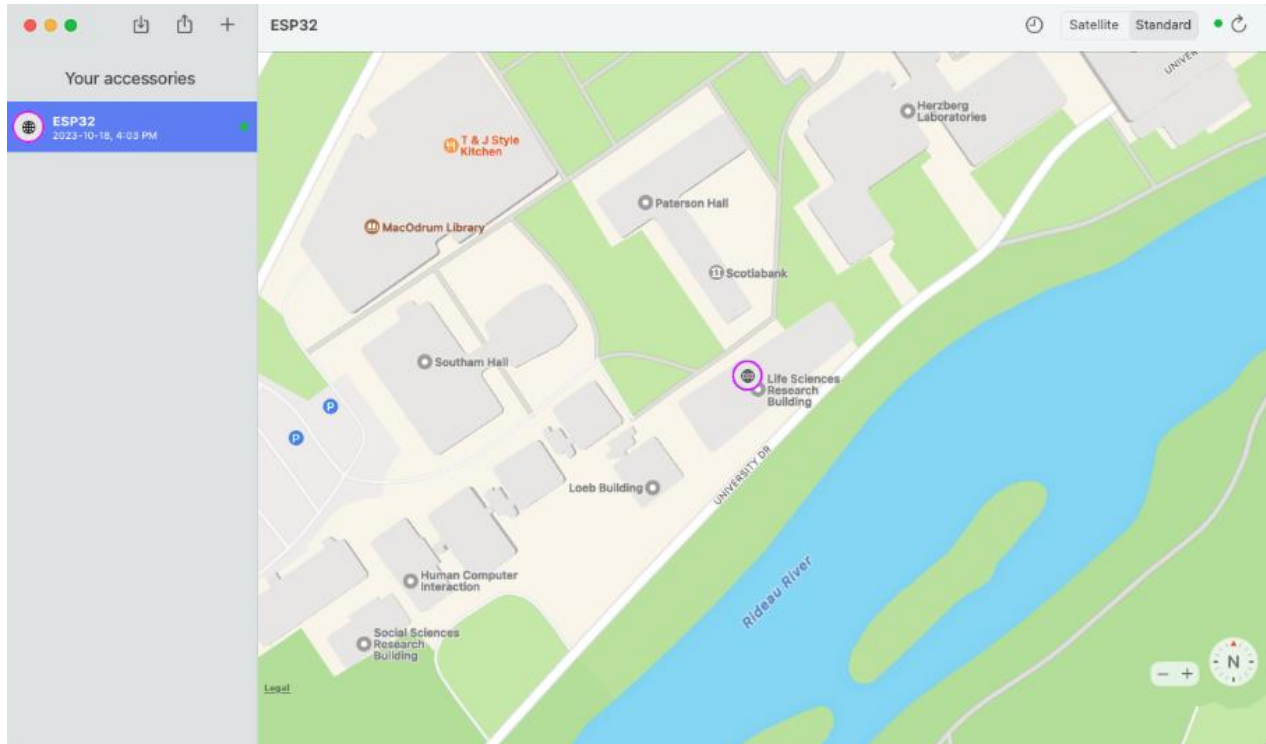


Figure 17 Find My App showing location of ESP32

The green circle icon beside the ESP32 label shows active tracking status. On the map, the purple circle icon shows the last location ping, which is in the ARISE building.

5.1.2 GPS Location Tracking

As mentioned in section 4.2.3, the final implementation of the GPS tracking software uses an Arduino Uno to request the location data from the CAM M8C board. The following figure shows the I²C connection setup between the Arduino Uno and CAM M8C.

The software works on a loop with a 0.25 second delay before requesting more location data. If a location fix is found, the data is parsed using NMEA formatting to get latitude and longitude. If no fix is found, the software prints the number of satellites available to the CAM M8C board. The following figure shows an example of a successful location fix using the Arduino Serial Monitor.

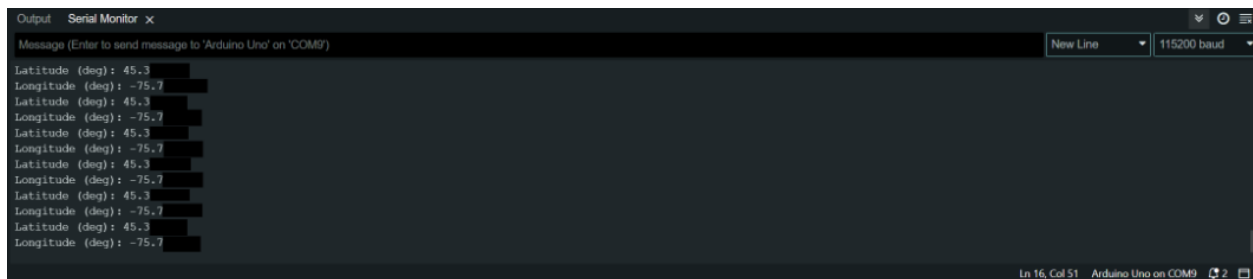


Figure 18 Successful location fix in Serial Monitor using CAM M8C and Arduino Uno

5.1.3 MCU UART Communication Setup

The first step in implementing the UART communication between each of the modules was to properly configure the STM32 Microcontroller through the STM32CubeIDE. This includes enabling UART1, UART4, and UART5 to use for each of the individual modules. The UART pins were established through careful review of the documentation of the Nucleo board as mentioned in 4.2.2. This ensured that the UART pins are available on the ST-Morpho pins, which are the inner pins of CN7 and CN8. Each UART communication default settings were altered based on the module of use described in section 4.2, which required in depth reading on what each module needed. For example, the Bluetooth module - ESP32 requires a 9600 baud rate.

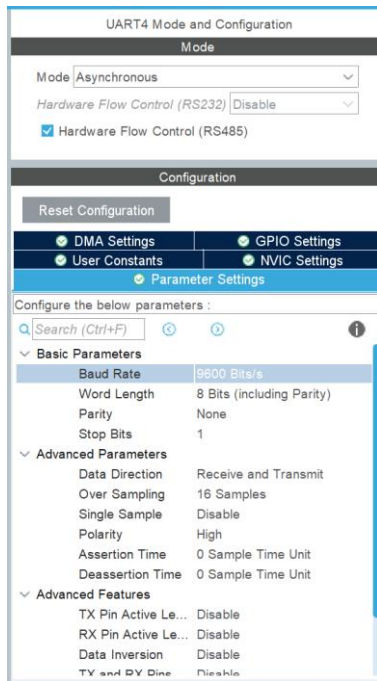


Figure 19 UART example setup on the STM32CubeIDE

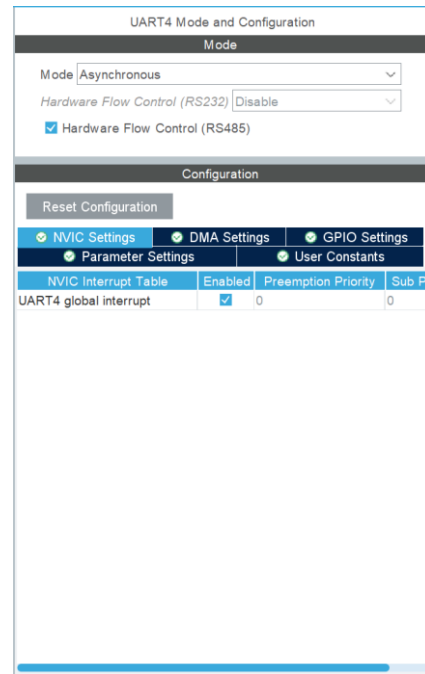


Figure 20 UART example setup on the STM32CubeIDE

The next step was to ensure each UART connection was properly configured. As mentioned in 4.2.2, we used the HAL UART library to implement a simple transmission of a message and retrieve of the same message from a UART port. Once retrieval of the message has been established between each UART communication, it was concluded that the UART communication was fully integrated and could begin development on the separate modules. The following figure shows a simple transmission using the HAL UART library followed by a message receive.

```

char txBuffer[] = "Hello, UART!\r\n";
char rxBuffer[20]; // Assuming a buffer size for receiving data

// Transmission
HAL_UART_Transmit(&huart4, (uint8_t*)txBuffer, strlen(txBuffer), HAL_MAX_DELAY);
printf("Send to uart4: %s", txBuffer);

// Reception
HAL_UART_Receive(&huart4, (uint8_t*)rxBuffer, strlen(txBuffer), HAL_MAX_DELAY);
printf("Received from uart4: %s", rxBuffer);

```

Figure 21 Example Transmit and Receive from UART4

5.1.4 MCU UART GPS Location Tracking

After confirming the GPS functionality using Arduino and its NMEA library in section 5.1.2 and establishing the MCU UART communication described in Section 5.1.3, we established a connection between the MCU and the GPS module through UART4 to continue with the development of a prototype.

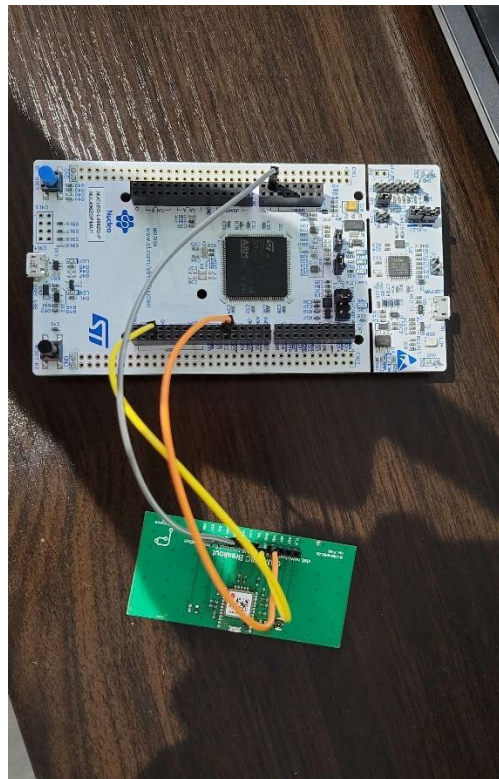


Figure 22 UART establishment between the GPS and MCU

The first step was to implement logic to receive from the GPS through the UART connection and parse the data that is collected to the MCU. Figure 23 below demonstrates the logic needed in creating the seamless transmission between the MCU and GPS for the prototype. It defines a callback function, which gets triggered every time a receive happens on a UART port. In this case, it's checking for a receive from UART4. After an initial receive is initialized, it continues to collect GPS data one byte at a time until 225 bytes is obtained. Once that logic is satisfied, we begin the process of parsing the GPS data. Specifically, we aim to obtain the latitude, longitude, and their respective direction.

```
void parseGPSData(char *data) {
    char sentenceType[6]; // Buffer to store the sentence type (e.g., GPGGA)
    float latitudeValue, longitudeValue;
    char latDirection, lonDirection;

    // Parse the sentence type
    if (sscanf(data, "%S[^,],", sentenceType) == 1) {
        if (strcmp(sentenceType, "GPGGA") == 0) {
            // Parse GPGGA sentence
            if (sscanf(data, "$GPGGA,%f,%f,%c,%f,%f,%c", &latitudeValue, &longitudeValue, &latDirection, &longitudeValue, &lonDirection) == 5) {
                // Print the parsed data
                printf("Latitude: %.4f %c\n", latitudeValue, latDirection);
                printf("Longitude: %.4f %c\n", longitudeValue, lonDirection);
            } else {
                printf("Failed to parse GPGGA sentence\n");
            }
        } else {
            printf("Unsupported sentence type: %s\n", sentenceType);
        }
    } else {
        printf("Failed to extract sentence type\n");
    }
}

void HAL_UART_RxCpltCallback(UART_HandleTypeDef *huart) {
    if (huart->Instance == UART4) {
        if (huart->ErrorCode == HAL_UART_ERROR_ORE) {
            printf("[ERROR] something went wrong\n");
            __HAL_UART_CLEAR_OREFLAG(&huart4);
            HAL_UART_Receive_IT(&huart4, (uint8_t *)&huart4.Instance->RDR, 1);
            return;
        }
        static uint8_t dataIndex = 0;
        if (dataIndex < sizeof(gpsBuffer) - 1) {
            gpsBuffer[dataIndex++] = huart->Instance->RDR;
            if (huart->Instance->RDR == '\n') {
                gpsBuffer[dataIndex] = '\0';
                parseGPSData(gpsBuffer);
                dataIndex = 0;
            }
        }
    }

    HAL_StatusTypeDef res = HAL_UART_Receive_IT(&huart4, (uint8_t *)&huart4.Instance->RDR, 1);
    printf("Hello: %d\n", dataIndex);
    printf("res in second callback uart4 receive: %d\n", res);
}
```

Figure 23 STM32 code for retrieving GPS location

Through logging, it did show signs of data collection from the GPS module. However, because of the inconsistency and unreliability of the hardware, it was difficult to see noticeable consistency of data collection. Deep diving into the issue through several location tests, the team concluded that this was a hardware issue where the signal of the GPS was too weak.

5.2 Mobile App Implementation

While the team worked on the embedded system side of things, we also progressed the mobile app to a state in which it's capable of communicating with Firebase cloud database and function with filler data. As mentioned in Section 4, we configured Firebase to support authentication and a database to store the necessary data for a proof on concept.

5.2.1 Front-End

The first step in developing the front end was getting the communication established with the back-end. There are 3 specific files: firebaseConfig.ts, google-services.json and GoogleService-info.plist. These are the configuration files to fully integrate the app, for both IOS and Android, and communicate with the Firebase API. Inside the firebaseConfig.ts file has 3 specific variables as shown in Figure 23 below.

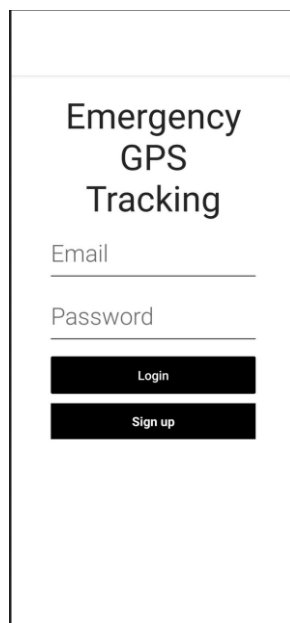
```
export const FIREBASE_APP = initializeApp(firebaseConfig);  
export const FIREBASE_AUTH = getAuth(FIREBASE_APP);  
export const FIREBASE_DB = getFirestore(FIREBASE_APP);
```

Figure 24 Firebase configuration variables

The FIREBASE_APP variable is simply the reference to the firebase app, which defines whether we successfully initialized the app to the firebase API. The FIREBASE_AUTH variable is used to

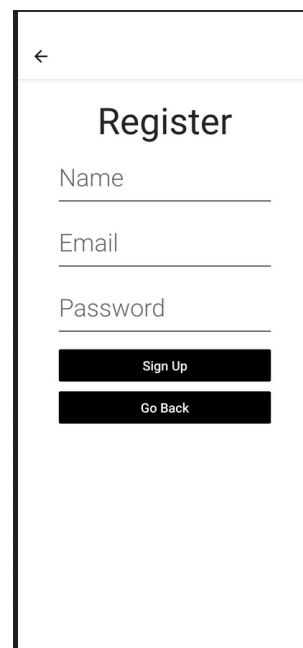
access the authentication service shown in Figure 24 above. Finally, the FIREBASE_DB variable is used to access all the collections shown later in section 5.2.2.

Figure 25, representing the login page, is the main screen upon launching the application for the first time. It will prompt the user to login if they have an account already. Otherwise, the user is free to create an account through the register page as shown in Figure 24. The login page verifies the credentials entered with the authentication database. If the credentials do not pass, the app will not proceed further. The registration page will ensure the email entered is unique and add the new user to the authentication service and Firestore database. The password entered will be automatically encrypted by the firebase authentication service, ensuring full security to the user account.



The login page features a title 'Emergency GPS Tracking' at the top. Below the title are two input fields labeled 'Email' and 'Password'. At the bottom, there are two black buttons: 'Login' and 'Sign up'.

Figure 25 The Login page



The registration page features a title 'Register' at the top, preceded by a back arrow. Below the title are three input fields labeled 'Name', 'Email', and 'Password'. At the bottom, there are two black buttons: 'Sign Up' and 'Go Back'.

Figure 26 The Registration page

The scenario shown below in figure 26 is the home page in which the user is redirected to once they successfully sign into the app. It displays all the PLWDs that they are caretaking, showing

their status and name. The user also has the option to delete their association with their PLWDs if they wish to, as shown Figure 27 below, by clicking the delete button, selecting the individuals they are caretaking, and confirm delete. By confirming deletion, it will remove the caretaker's association with the PLWD tracker hardware and delete the information on the app as well.

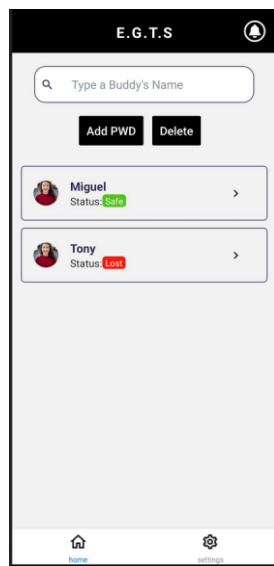


Figure 27 The home page

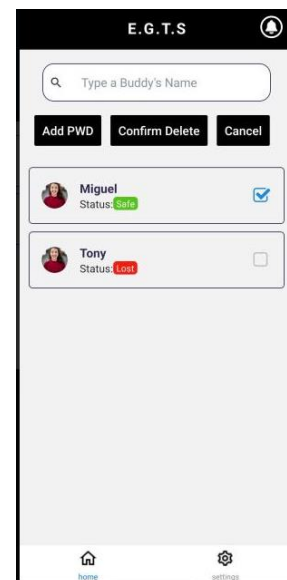


Figure 28 The home page when delete button is pressed

In addition to deleting individual(s), there is also adding a user's association with a PLWD tracker hardware through the click action of the "Add PWD" button at the top of the page. This scenario can be shown below in Figure 30, where a pop up be visible to the user, allowing the user to input the username associated with the tracker hardware as highlighted in Figure 31.

Attempting to add the new individual to the user first checks if that username exists in the database and, if so, will associate the user with the PLWD and return to the home page.

Otherwise, the app ensures no new tracker hardware association is added if the unique identifier is incorrect. If the user changes their mind in adding a new PLWD, they can click anywhere on the screen to close the dialogue, making it a very intuitive feature to use.



Figure 29 Adding a new PLWD prompt for the user

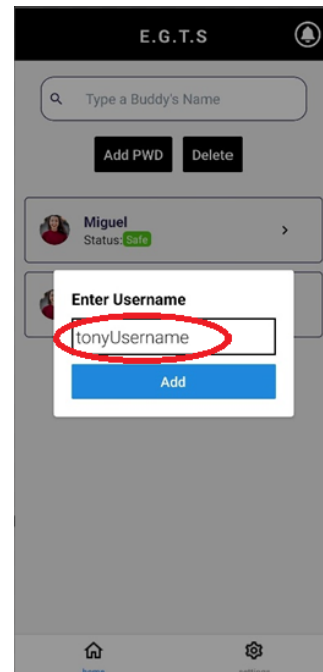


Figure 30 Adding input to the new PLWD prompt

Additionally, a tabbing layout as described in the mobile app design was also implemented for easy navigation between essential pages, which currently allows for navigation between the home and settings pages. The tabs are at the bottom of the screen, which also includes respective icons, like the home and gear icon to indicate home and settings page respectively, for users to easily recognize the meaning without necessarily reading the text. Overall, the user can switch between the two pages, where Figure 31 below represents the settings page.

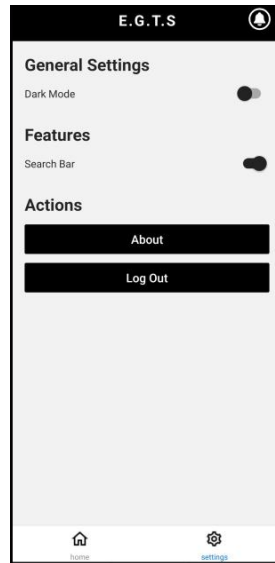


Figure 31 The settings page

The settings page currently has the log out feature implemented, allowing users to gracefully exit the app.

We also implemented the individual PLWD page showcasing their name, status, the configuration for adding a home location, a button to launch the phone mobile maps app locally with the last known location, and an integrated map to visually see the last known location found from the database and shown on the map through a location marker. This scenario is active when a user clicks on a person, they are caretaking from home page shown in Figure 30 above.

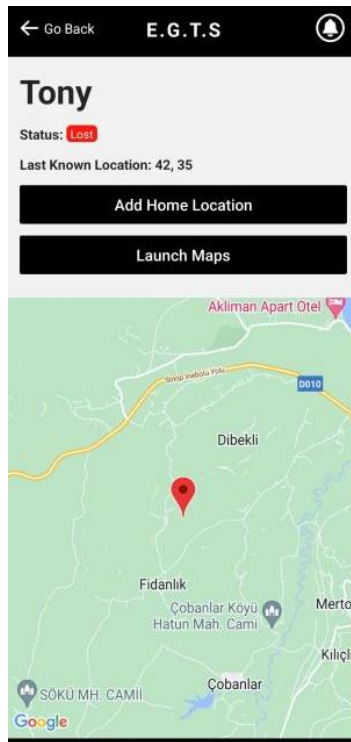


Figure 32 The PLWD individual page

Finally, clicking the Add Home Location opens the scenario of relocating the user to the configuring of the safe-zone for that respective individual they are caretaking. This can be seen in Figure 33 where it will allow the user to set the radius in the unit of Kilometre's and the set the location that defines the home location. This can be through the button of using the user's current location, or by clicking around the map itself. Clicking save will apply the configurations in the database.

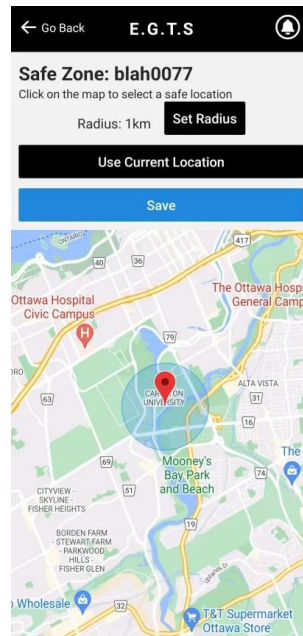


Figure 33 Safe-Zone page for configuring the home location for a PLWD

The “Set Radius” button will open a dialog box allowing you to input and save a new radius as shown in Figure 34.

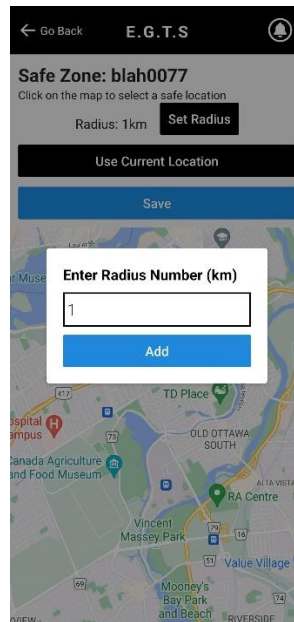


Figure 34 Configure radius dialog in the safe-zone page

Overall, the app has been fully integrated with the database along with real time features. On any page, any changes applied to the database will reflect on the app providing full reliably and visible feedback. It is also completely scalable for new features. At the same time, everything in the codebase is well documented and named accordingly to ensure code reviewers, or new developers, can adapt to the codebase without much effort.

5.2.2 Back-End

The Firebase authentication not only supports the hashing of user passwords for security purposes, but also supports methods to reset passwords and disable and delete an account.

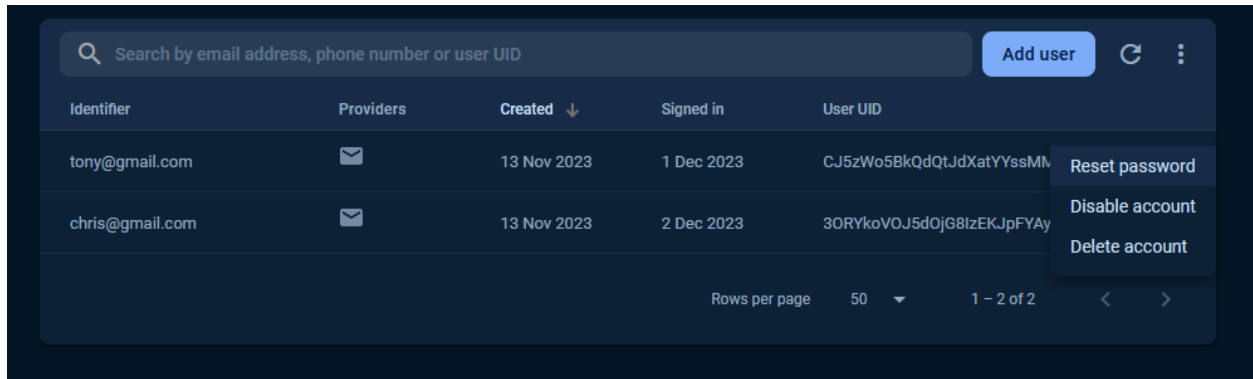


Figure 35 Firebase Authentication service.

As seen in Figure 35, the authentication database uses email providers to identify users. This way, we can use email providers for the app and all its features. For example, resetting a password will email the user to reset the password. This can be done by sending an API request to the firebase authentication “Reset Password” service.

In addition to the authentication, a basic database was created for the app to collect data based on the user signed in. This can be seen in the following firebase figures, showing the collection of PLWD from the tracker hardware, the collection of coordinates that links with the PLWD, a collection of users authenticated by the app, and a collection of safe-zones per PLWD with a associated user.

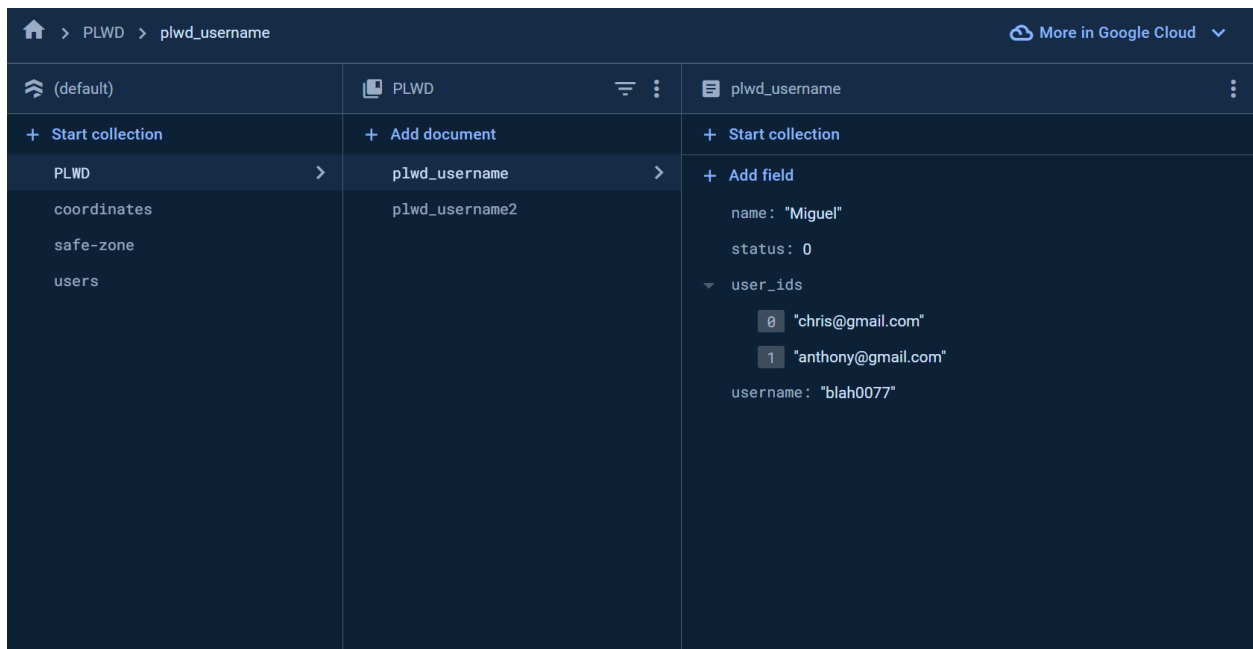


Figure 36 The collection of PLWD that links from the tracker hardware

As described in the design overview of the mobile app, each PLWD using the tracker hardware will have a name and a status. Status will describe the severity of the PLWD, in which 0 is defined as safe while 1 is defined as lost. Each PLWD will also have a list of users (caretakers) they are associated with, and a username that represents their unique identification from the tracker hardware.

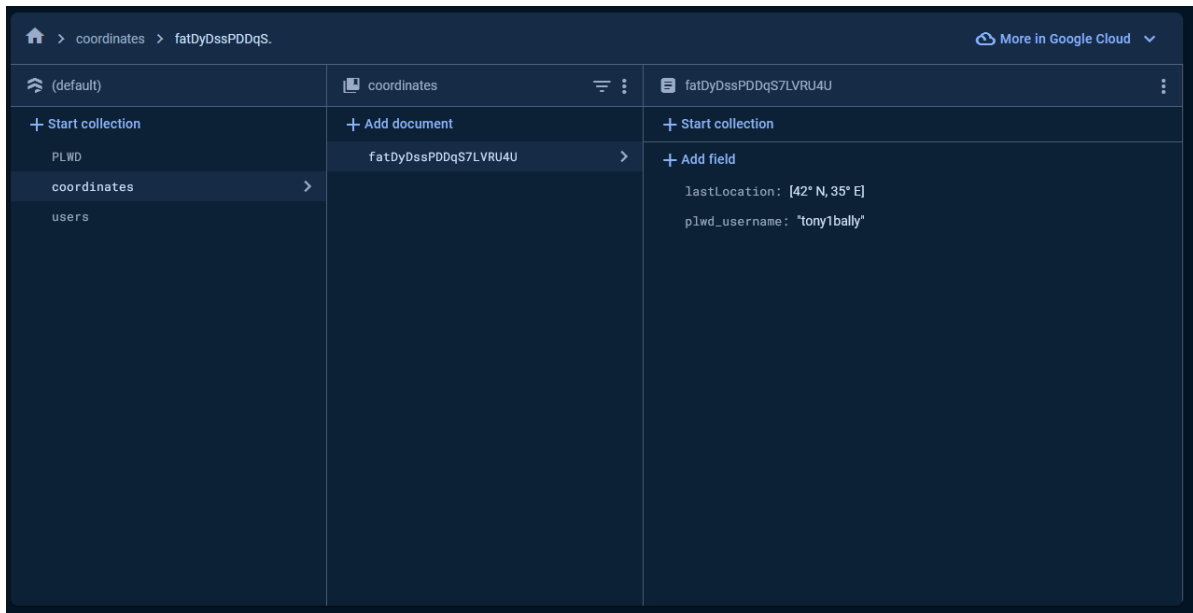


Figure 37 The collection of Coordinates for each tracker hardware

The coordinate collection shown in Figure 37 represents the last known location of the PLWD. It will have a pair of coordinates representing latitude and longitude, and the identifier of the tracker hardware the coordinates is associated with. If the tracker hardware did not log its last known location associated with the PLWD, the status will result as N/A until the PLWD is considered lost.

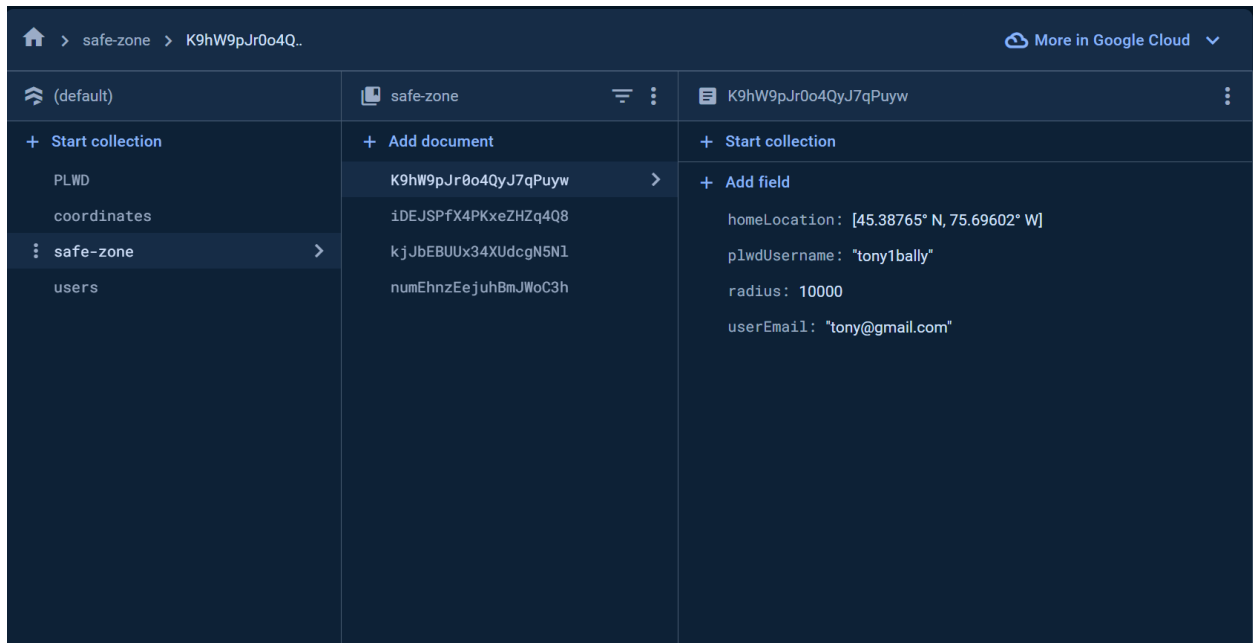


Figure 38 The collection of safe-zone per user

The safe-zone collection defined in Figure 38 represents the data in which the PLWD is considered safe or loss depending on its home location and the respective radius distance. This collection was designed to specifically allow multiple users caretaking the same PLWD to have unique home locations and set radius on what is considered safe for the PLWD. This level of scalability creates an open environment in handling multiple caretakers. Overall, each collection includes the coordinates of the home location, the set radius in kilometers, the identification of tracker hardware and the associated user of the app through the email authentication.

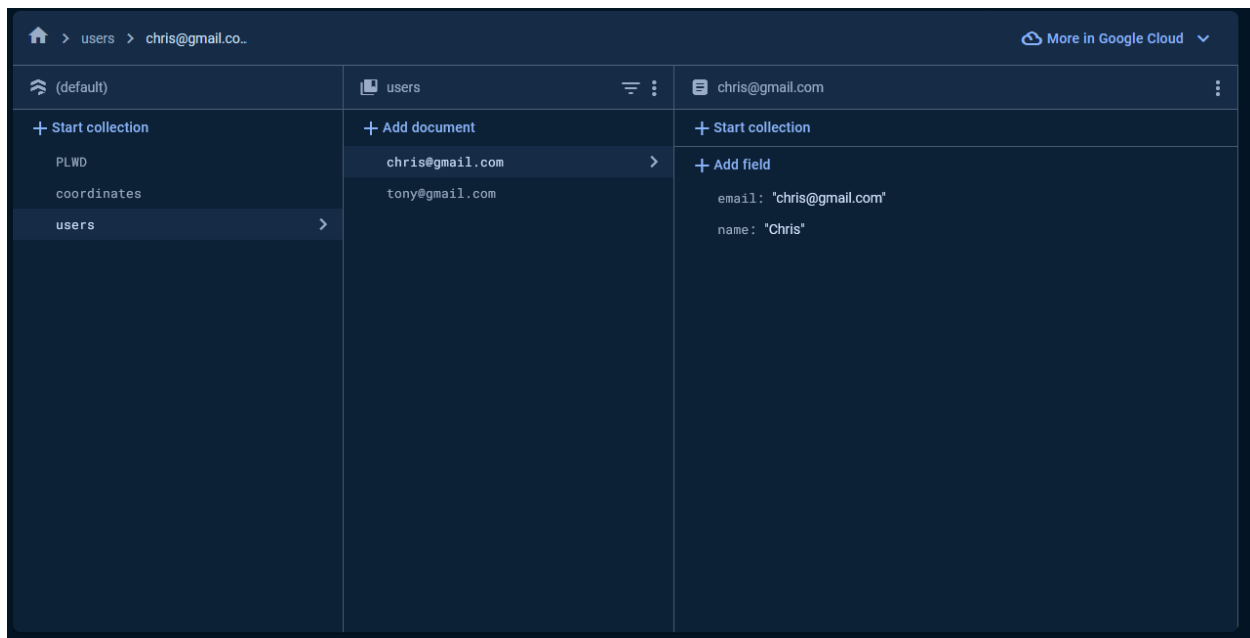


Figure 39 The collection of users associated with an account

Finally, as users get authenticated, they are also stored within the database to be referenced by other collections in the database. This completes the full back-end experience that the mobile application will communicate with. Through the preset values in the database as shown in the figures above, we can communicate, collect data, and provide a full user experience. Additionally, as values change in the database, real-time updates are applied to the app.

6.0 Testing

6.1 PCB Testing

A large portion of the term was spent on testing and debugging the PCB. The first test we completed is a power test of the printed circuit board. To ensure proper connections, we tested the wiring on the board by using an ohmmeter to look for resistance between connection points. To ensure proper power, we tested all the power and ground lines to find any possible short circuits. The connections are compared to the CAD schematics of the PCB to ensure that the connections were printed according to the initial design. To test these connections more easily, we used a blank version of the PCB picture in the following figure. We used the blank to get easy access to all pins as all the pins are exposed.

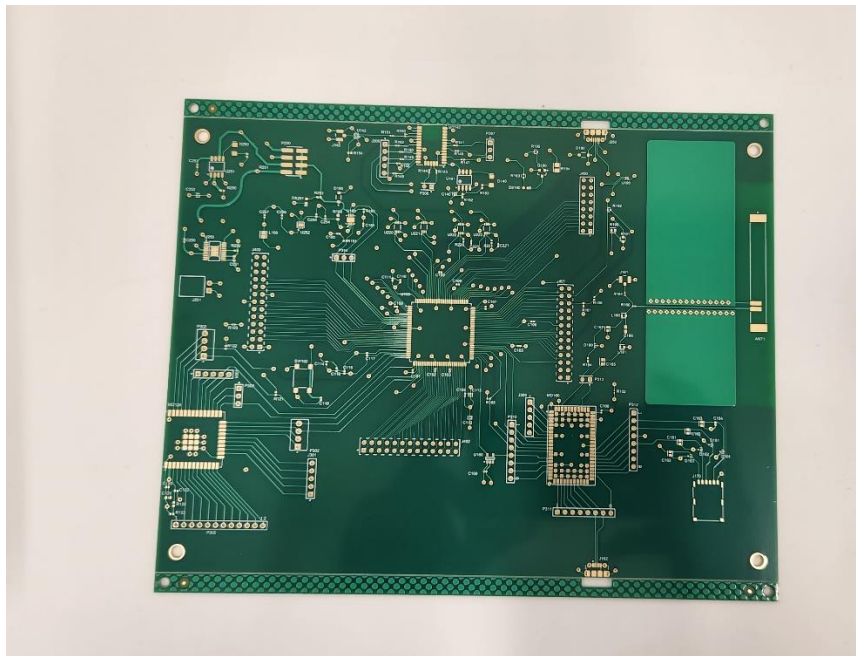


Figure 40 Picture of the blank version of the PCB

After testing just the power lines, we found that there were no shorts along our power lines. We were also successfully getting the recommended voltage to our microcontroller. This allowed us to attempt to begin development. However, after attempting to begin development, we found multiple issues that did not allow us to continue development.

6.1.1 GPIO Issues

The first issue we found that blocked development was the fact that some GPIO pins from the MCU, required for debugging, were not available on the PCB as headers. The debugging pins were required to connect the ST-Link debugger to the PCB. The pins required for debugging were PA14 (SWCLK), PA13 (SWDIO) and the MCU reset pin. These pins were missing from the headers as seen in the following figure.

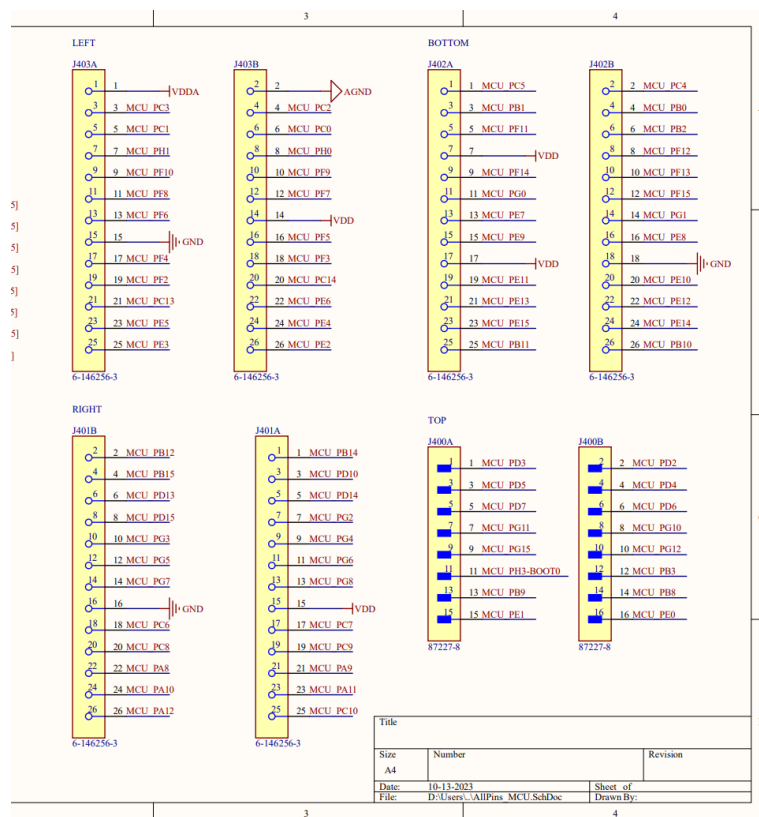


Figure 41 MCU header schematics for the PCB

The short-term fix to this issue was to expose these pins directly on the board by cutting the trace and placing headers on those traces. This fix is shown in the following figure.

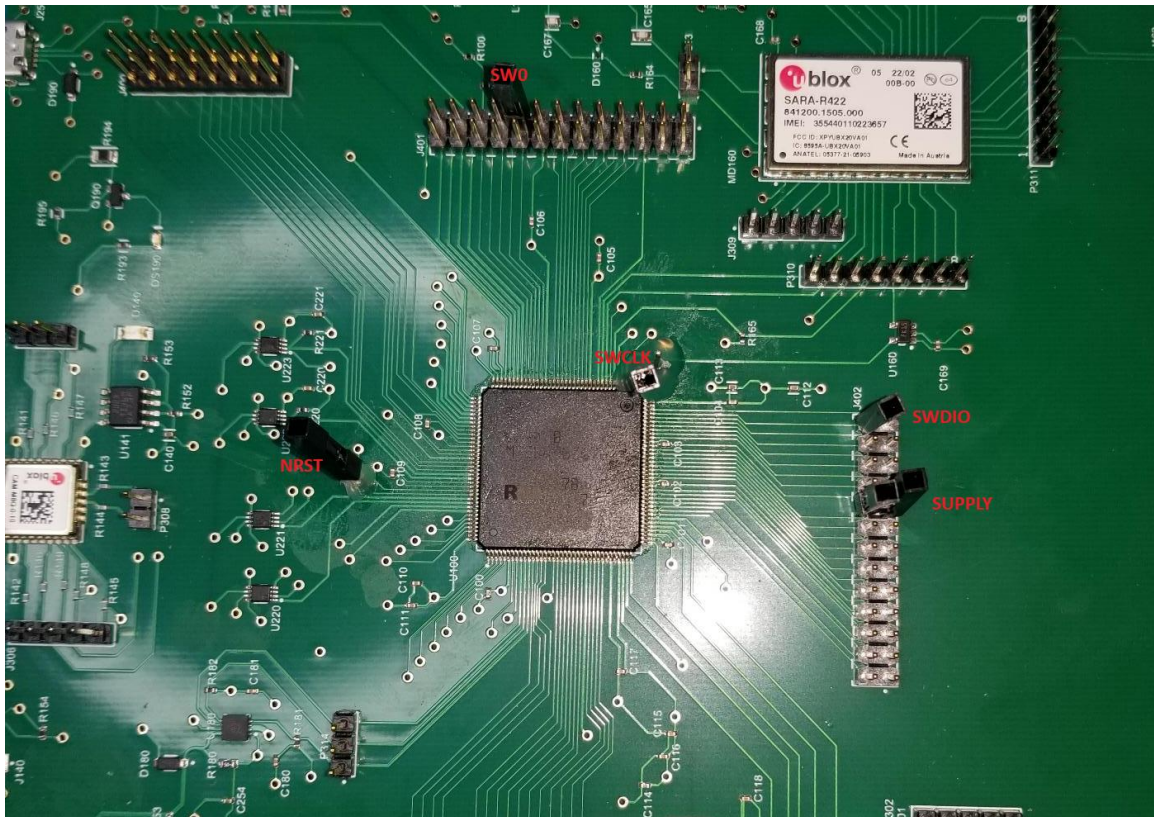


Figure 42 Exposed GPIO pins on the PCB

In this figure you can see the new headers for the NRST (MCU reset) and PA14 (SWCLK). This allowed us to attempt to connect the debugger and attempt development again.

After adding the new headers, we still had a potential issue where the new headers we exposed are connected to two modules. This means the signal could be occupied by another module instead of the MCU, where the pin is required for debugging. In the following figure, you can see the GPS' IO pins connected to the MCU.

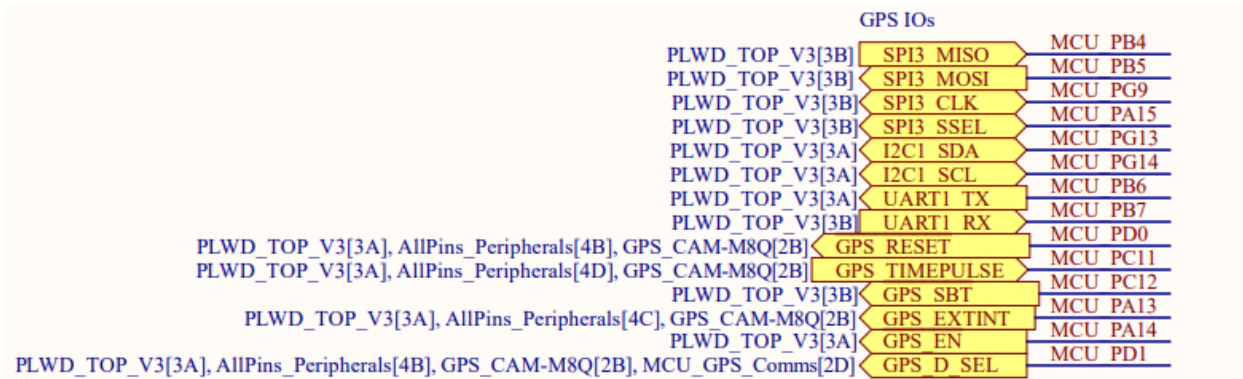


Figure 43 Schematic of GPS IO pin connections to MCU

This shows that both PA13 and PA14 could be occupied for communication between the MCU and GPS. Specifically, these pins were used as external interrupt and enable from the microcontroller to the GPS. Our short-term fix for this was to completely sever the PA13 and PA14 traces and adding headers on both sides of the trace.

The biggest issue we had encountered is that the MCU was fabricated in the incorrect orientation on the PCB. The MCU was installed 90 degrees clockwise from the orientation intended in the schematics for the PCB. This led to all the pins being placed on the wrong traces, which made all the pins unusable.

The short-term fix to this was simply to desolder the MCU from the PCB, rotate it 90 degrees counterclockwise and resolder. These fixes were done with the help of a researcher in the Bruyère research team.

6.1.2 Power Issues

When the MCU was in the wrong orientation, there were two power issues noted on the PCB. Even with the MCU receiving the proper voltage, there was power management unit that was receiving an overcurrent. The following figure shows the schematic of one of the power management units, a power switch, which has an LED (labelled D140) that was indicating an overcurrent on the PCB.

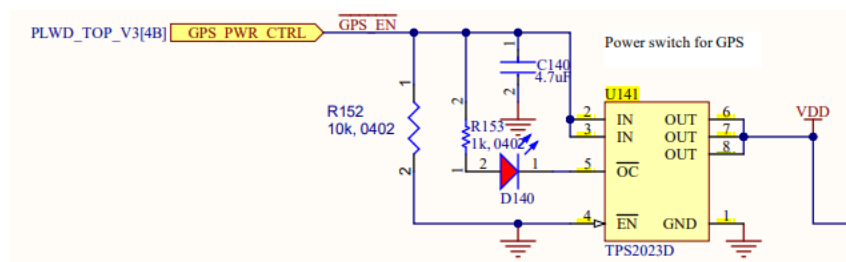


Figure 44 Schematic of power switch for the GPS module on the PCB

The last issue we found while debugging was found after the MCU rotation. The MCU was not receiving the operating voltage necessary. After some more rigorous debugging, we found that the PB13 pin of the MCU was shorted to ground. We went back to both the previous board and a blank board to see if the issue existed elsewhere and determine what was causing it. We found that the blank board did not contain this connection to ground, meaning that it was being caused once the MCU was placed. The PB13 pin, both before and after the rotation of the MCU was consistently found to be connected to ground, which was causing the power issue. As a

result, we were left to determine that there was an issue with the MCU itself that would need further debugging in the future.

6.2 Potential PCB Fixes

To address the GPIO pin conflicts on the PCB, the future design should free the MCU pins required for debugging from being used as communication to the GPS (pins PA13 and PA14). These pins should be available as headers on the board instead. Additionally, there is one other missing MCU pin missing from the headers required for debugging (NRST). This new design would also require replacing the pins PA13 and PA14 to connect to the GPS. Two possible pins currently available on the header are PD10 and PD11. PD10 and PD11 are general purpose pins whose alternate functions is for connecting external memory to the MCU. Since these pins are not required for debugging or other important functions, a future design can use these for communication instead.

To fix the MCU orientation, there is no redesign required. The original schematics have the MCU placed in the correct orientation.

To address the power issues, we believe the whole power management should be redesigned. One potential central issue is the modules along the VDD line. VDD is the main power supply to the location and communication modules, and it is outputted from the voltage regulator, U252. This VDD supply is most likely the central issue to the overcurrent in U141 and inconsistencies in power supply to the MCU.

6.3 Embedded System Testing

For each development board we developed on this term, we tested for correct functionality. For location tracking boards, we tested for boot-up time for first location fix and the location accuracy. To test boot-up time, we timed from module power on until the first location fix. We confirmed location accuracy by comparing the coordinates onto Google Maps. For the MCU, we confirmed proper communication functionality to the other boards.

Table 3 Unit Tests for each Embedded System Module

Module	Test Expectations
STM	<ul style="list-style-type: none">• Successfully transmit messages to modules over UART
Bluetooth	<ul style="list-style-type: none">• Establish connection to the Find My network within 60 seconds of module boot-up• Receive location data within 5 metre accuracy
GPS	<ul style="list-style-type: none">• Establish connection to GPS network within 60 seconds• Receive location data within 5 metre accuracy

7.0 Conclusion

The problem at hand that this project addresses is ensuring the safety and security of individuals living with dementia (PLWD) while providing peace of mind to their loved ones who are caregivers. A PLWD often face the risk of wandering and getting lost, this being one of the more severe symptoms, which poses significant concerns for their well-being and safety.

Additionally, most traditional methods of supervision for a PLWD, for example caretaking at home, is usually insufficient and can lead to heightened stress among caregivers. Additionally, despite existing technologies, such as most portable watches, providing tracking functionalities they also lack long battery life. Thus there is a major lack of solutions available in the modern world with devices that has long enough battery life to be considered sufficient for the problem at hand. Overall, there exists a pressing need for an innovative approach that integrates various technologies to address the challenges in assisting caretakers to aid for their loved ones.

To tackle the problem at hand, the project proposes the development of a device that will be compressed, in the future, into some form of watch hardware to be worn by the PLWD. This device will leverage a combination of hardware including a Microcontroller, Bluetooth board, GPS board and an LTE board all integrated together. It aims to provide real-time tracking and monitoring of the PLWD, easing the caretaker's burden, and enhancing the PLWD safety and security. Along side the hardware device will be a user-friendly application that enabled caregivers to track the whereabouts of their loved ones and receive alerts in case of emergencies, for example the PLWD is officially lost. Through this integrated approach, the project intends to alleviate the burdens faced by caregivers and enhance the quality of life for the PLWD and their families.

Amidst the challenges encountered, functionality for the development boards including the microcontroller, GPS and Bluetooth has been established, as well substantial progress in the mobile application. The GPS and Bluetooth component can function independently, both providing accurate location data that can be used for the project. At the same time, communication within the microcontroller has also been established, which provides a solid foundation for integrated communication between each module and the microcontroller.

Continuing with the software side, all the mobile app core features have been implemented along with their respective pages. This includes registration, login, home page, settings, and various functionalities such as viewing coordinates data and configurations of a safe location for the PLWD in respect to the user. Additionally, robust security measures have been integrated to safeguard user data and ensure a seamless experience. Overall, the app is now equipped with a fully connected backend containing all the necessary data to be used with the front-end, providing users with a comprehensive and intuitive experience. Through the integrated connection of the backend and front-end, the software works in real-time to fully secure its reliability for the user.

Next steps for this project will require the redesigning of the PCB made from the summer of 2023, which the team considered to be a crucial priority for a successful prototype. Despite the development boards being sufficient, a fully connected board is still necessary to later confirm its ability in compressing into a watch. As of now, there were a substantial number of faults throughout the PCB including needed pins that were missing, a mis-orientated MCU by 90 degrees, and a critical short connected to the internal system of the MCU. The team recommends that, as the redesign and fabrication is in progress, schematics should also be

thoroughly looked at as there were miscommunications with the fabrications company resulting in inconsistent outcomes that confused us while debugging.

However, the progress done thus far has shown to be promising, with significant strides made towards achieving the project's objectives. The successful establishment of functionality for the microcontroller, GPS and Bluetooth components, coupled with the completion of core features in the mobile application is a major step forward in trying to improve quality of life for PLWD and caretakers.

References

- [1] "What Is Dementia?," What is Dementia? [Online]. Available:
<https://www.alz.org/alzheimers-dementia/what-is-dementia>
- [2] Alzheimer's Association, "Wandering," Wandering. [Online]. Available:
<https://www.alz.org/help-support/caregiving/stages-behaviors/wandering#:~:text=Alzheimer%27s%20disease%20causes%20people%20to,any%20stage%20of%20the%20disease>
- [3] D. G. Manuel *et al.*, "Alzheimer's and other dementias in Canada, 2011 to 2031: a microsimulation Population Health Modeling (POHEM) study of projected prevalence, health burden, health services, and caregiving use," *Popul. Health Metr.*, vol. 14, no. 1, Art. no. 1, Dec. 2016.
- [4] Sara Shu and Benjamin KP Woo, "Use of technology and social media in dementia care: Current and future directions," Use of technology and social media in dementia care: Current and future directions. [Online]. Available:
<https://www.ncbi.nlm.nih.gov/pmc/articles/PMC8040150/#:~:text=Technological%20advancements%20over%20the%20past,improve%20functioning%2C%20tracking%20and%20mobility>
- [5] "Alzheimer's Association 2016 Alzheimer's disease facts and figures," Alzheimer's & Dementia, 2016.
- [6] Alzheimer Society of B.C., "Wandering and Dementia," Wandering and Dementia. [Online]. Available: <https://alzheimer.ca/bc/en/help-support/programs-services/dementia-resources-bc/wandering-disorientation-resources/wandering-dementia>

- [7] M. B. Lilly, C. A. Robinson, S. Holtzman, and J. L. Bottorff, "Can we move beyond burden and burnout to support the health and wellness of family caregivers to persons with dementia? Evidence from British Columbia, Canada: Supporting family caregivers in Canada," *Health Soc. Care Community*, vol. 20, no. 1, pp. 103–112, Jan. 2012, doi: 10.1111/j.1365-2524.2011.01025.x.
- [8] E. Miller, R. Rosenheck, and L. Schneider, "Caregiver burden, health utilities, and institutional service use in Alzheimer's disease," *Int. J. Geriatr. Psychiatry*, vol. 27, pp. 382–393, 2012.
- [9] Alzheimer Society, "Tracking Devices," Tracking Devices. [Online]. Available: <https://alzheimer.ca/en/help-support/im-caring-person-living-dementia/ensuring-safety-security/tracking-devices>
- [10] AngelSense, "GPS Tracking For Kids," GPS Tracking for Kids. [Online]. Available: <https://www.angelsense.com/product-tour/>
- [11] Life360, "GPS Tracking for Elderly," GPS Tracking for Elderly. [Online]. Available: <https://www.jiobit.com/seniors>
- [12] Apple, "Apple Airtags," Apple Airtags. [Online]. Available: <https://www.apple.com/ca/airtag/>
- [13] GlobalEduTech, "Health and Safety." [Online]. Available: <https://www.computerengineeringconcepts.org/10>.
- [14] "Science and Technology (2022)." [Online]. Available: <https://www.dcp.edu.gov.on.ca/en/curriculum/science-technology/context/processes>

- [15] "Functional and Nonfunctional Requirements: Specification and Types." [Online]. Available: <https://www.altexsoft.com/blog/functional-and-non-functional-requirements-specification-and-types/>
- [16] React-Native, "React-Native Documentation," React-Native Documentatoin. [Online]. Available: <https://reactnative.dev/docs/getting-started>
- [17] Expo, "Expo Documentation," Expo Documentation. [Online]. Available: <https://docs.expo.dev/>
- [18] Figma inc., "Start designing with Figma." [Online]. Available: <https://www.figma.com>
- [19] STMicroelectronics, "STM32 32-bit Arm Cortex MCUs." [Online]. Available: <https://www.st.com/en/microcontrollers-microprocessors/stm32-32-bit-arm-cortex-mcus.html>
- [20] NUCLEO-L496ZG-P, "NUCLEO-L496ZG-P documentation." [Online]. Available: <https://os.mbed.com/platforms/ST-Nucleo-L496ZG-P/>
- [21] Ublox, "u-blox M8 Concurrent GNSS Antenna Modules." [Online]. Available: https://content.u-blox.com/sites/default/files/CAM-M8-FW3_DataSheet_%28UBX-15031574%29.pdf
- [22] Link Labs, "What Technology Does AirTag Use?" [Online]. Available: <https://www.link-labs.com/blog/how-does-airtag-share-location#:~:text=It%20is%20relatively%20energy-efficient,backpacks%2C%20purses%2C%20and%20laptops.>
- [23] GPS.Gov, "New Civil Signals." [Online]. Available: <https://www.gps.gov/systems/gps/modernization/civilsignals/>

[24] Robert Triggs, “Tested: How much does Bluetooth actually drain your phone battery?,”

Tested: How much does Bluetooth actually drain your phone battery? [Online]. Available:

<https://www.androidauthority.com/does-bluetooth-drain-battery-1145853/>

[25] “Impact of GPS Signal Strength on Power Consumption.”

[26] Alexander Heinrich, Milan Stute, “OpenHaystack Map Reference,” OpenHaystack map

reference. [Online]. Available: <https://github.com/seemoo->

[lab/openhaystack/blob/main/Resources/OpenHaystack-Screenshot.png](https://github.com/seemoo-lab/openhaystack/blob/main/Resources/OpenHaystack-Screenshot.png)

Appendix

1.0 Overview of Tasks

Throughout this capstone project, the team will be working with a printed circuit board (PCB) to implement a proof of concept for the tracker hardware to demonstrate tracking functionality.

This implementation will include four tiers of power-consumption using embedded systems programming. To create these tiers, we will complete the software functionality of all hardware modules that will be used for tracking and communication.

The first steps to creating the proof of concept is to guarantee that each hardware module functions as expected. This means each module are sufficiently powered and are only powered when directed. Each module can be turn on and off on demand. Lastly, each module needs to establish communication to the CPU and back, which is crucial to handle controlled commands. The team will be testing the following modules on the PCB: LTE, GPS, Bluetooth, and CPU, as well as optimizing power usage across the whole PCB.

While optimizing power usage across whole PCB, the team will consider the accuracy in location data between Bluetooth and GPS as they greatly vary, however, there is a major trade-off in power consumption between the two modules. Because of this, as mentioned before, Bluetooth will be prioritized in obtaining the PLWD location data in order to minimize battery consumption as much as possible. In fact, leaving Bluetooth on compared to off averagely drains an extra 1.6% over the span of 4 hours [24]. Compared to GPS, which also depends on signal strength, can drain from 13%-38% of battery [25]. Hence, the team will be focused on optimizing Bluetooth accuracy in attempt to reduce the usage of GPS for the device.

To create the hardware functionality, the following behaviours will need to be created: power status of the board and individual modules, required control signals between modules, and the core module functionalities. The GPS module will need to receive location data through satellite networks. The LTE module will need to be able to connect to cellular networks to communicate the PLWD's location data. The Bluetooth module will need to both receive location data and communicate the data through the Find My network. The CPU's functionality will be to coordinate communication between each module by using command signals.

To date, the handoff from the previous team includes a working demonstration of the Bluetooth and LTE development boards. The Bluetooth development boards uses OpenHaystack, an open-source software that allows the hardware to simulate an Apple Air Tag. It allows the hardware to connect to the Find My network and map its location onto the integrated map of OpenHaystack as shown in Figure 4. The LTE development board will consist of a SIM card to connect to a cellular provider such as Fido, Bell, or Rogers. Once the SIM card is connected, code written in embedded C onto the CPU will allow commands to launch data from the LTE to cellular networks. Lastly, the team has inherited the embedded C code for the CPU that consist of all the setup for the connectivity from the CPU to each module, as well as communication functionality to send commands.

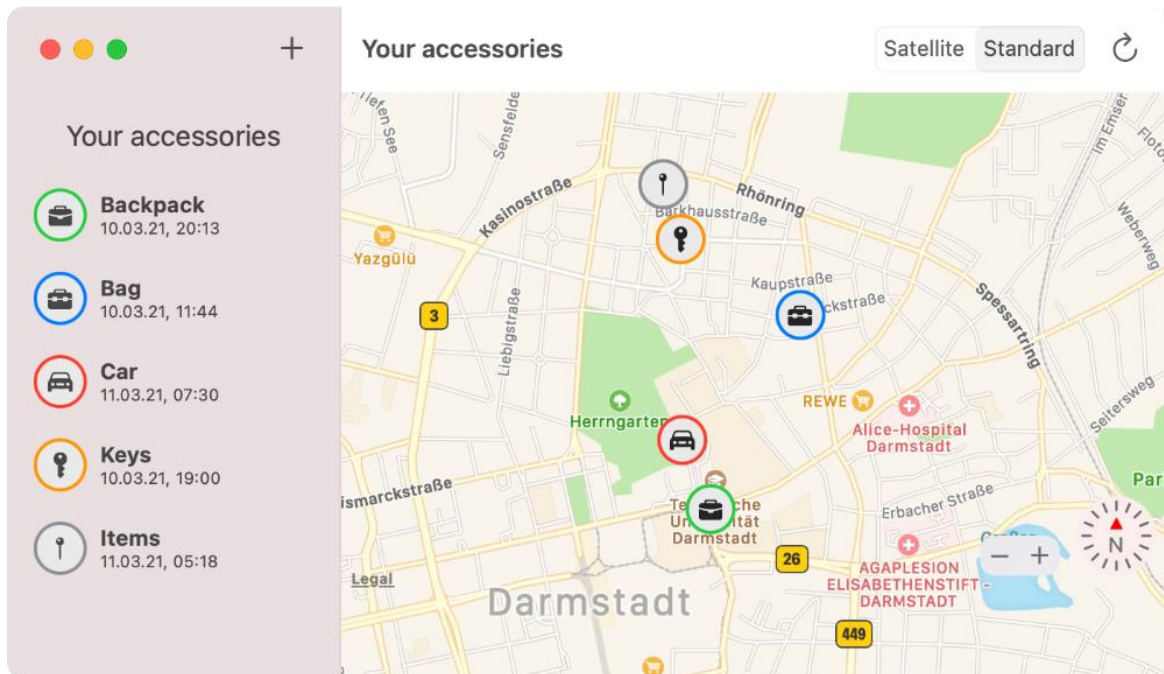


Figure 45 OpenHaystack example of multiple devices connected and being tracked on the map [26]

In addition to developing a PCB, we will also be implementing a mobile application that functions on both IOS and Android. We will also be hosting a cloud database to store data such as the wearers of the device, users of the app, and the GPS location of each device wearer. We will ensure that this set of data is only accessed by authorized entities through a secure token to keep confidentially. The app will contain the caretaker's loved one and for each PLWD there will be a map that will show their current location if they are lost. Overall, we will establish communication between the hardware and mobile app through the database integration.

2.0 Project Methods

Creating an emergency tracking device requires a multi-disciplinary approach to determine the appropriate methods to use to obtain the maximum results. In this case, we concluded the approach for creating a working proof of concept would be to use an STM32 microcontroller to establish communication between the software and the hardware modules using embedded C. At the same time, we are also determined in linking this product to a mobile app integrated with an application programming interface (API) database to showcase real time GPS updates with our product.

2.1 Embedded Systems Methods

For the embedded systems development, we will analyze and choose the appropriate hardware components to determine the most efficient approach in integrating the hardware and the software. We will test the components using simple electrical analysis methods, such as voltmeters. Additionally, working with circuit boards will be crucial as it is fundamental to establish a connection between each component. The core functionality of the hardware will be developed using embedded C and the STM32Cube IDE. The functionality for the Bluetooth location tracking will be implemented using an open-source version of Apple's Find My network called OpenHaystack.

2.2 Software Methods

For the app development, we will need to use a framework capable of integrating iOS and Android, as well as create a user-friendly interface for users to view real-time locations. We have

chosen React Native as the framework as it has Android and iOS support, as well as many user interface libraries. Additionally, we will build an API that interfaces with a database to store or retrieve sets of data, such as users and GPS coordinates. At the same time, we would need to investigate security measures for data transmission between the app and the API. The database that will be interfaced will be an open-source cloud database that will allow for fast communication to the mobile app. Lastly, real-time location displays as well as emergency alert handling would be a priority as it makes the app easier to use. Lastly, quality assurance is important to test the integrated system thoroughly to identify and resolve any issues. The quality assurance will be in the form of unit tests and integration tests. Unit tests are necessary to ensure expected behaviour from the different hardware modules and software modules. Integration tests will also be necessary to test connections between mobile app to database, and database to board hardware.

2.3 How the Methods Relate to our Degrees

All this relates to the knowledge learned in our degree to date as it involves problem solving, organizing, and structuring, all of which we have learned in our classes. For example, identifying and establishing how the communication works between the hardware and software, and how it ties to the mobile app. Also, team collaboration, in which we experienced throughout our years, will be crucial to ensure an efficient development as hardware engineers specialize in hardware and software specializes in software.

3.0 Relation to Degree

Two of our team members, Christopher, and Nicolas, are in the Bachelor of Computer Systems Engineering program. In Computer Systems, we have gained experience in both computer hardware and software, including embedded systems programming. Our role in the project is to expand the embedded system's software to complete the functionality of the tracker hardware. Some of the skills involved in this are embedded C programming, microcontroller hardware knowledge and hardware communication methods.

Our other two members, Anthony and Cory, are in the Bachelor of Software Engineering program. The Software stream has a greater focus on software development and design with less experience in hardware and embedded systems. The experience in software implementation will help greatly in the mobile app development and database development for the accompanying mobile app. Experience in software design will also help for structuring code for all the modules on our tracker hardware.

4.0 Group Skills

As mentioned previously, the group for this project contains two software engineering students and two computer systems engineering students. As such, the group has a diverse skill set in areas of both software and hardware engineering that can be utilized to complete the project. Through schooling and co-op work terms, the team has amassed and solidified strong programming skills in various languages, including but not limited to C, C++, and JavaScript. With experience in both software development and testing, the group is well equipped for the development of both embedded systems and mobile applications. All members of the team have also had previous experience working in a group setting together, and therefore understand how to operate in a team environment with one-another.

The software engineering students will be able to use the skills they have acquired in JavaScript, web development, and database management to design and develop a mobile application that is efficient, easy-to-use, well structured, and capable of running on both iOS and Android. The data stored by the application will be protected and kept secure using methods and practices that have been learned in cyber security courses. Meanwhile, the computer systems students will be able to apply their knowledge of embedded systems to develop the required software needed to turn the printed circuit board provided into a fully functioning emergency tracking device. A strong understanding of communication protocols, such as I2C, will be utilized to ensure that the various features of the emergency tracking device are able to work together to achieve the end goal.

5.0 Project Risks and Mitigation Strategies

1. Scope Creep

Risk	There is a possibility that the team continuously requests changes to the design and project scope, leading to delays and increased costs.
Mitigation Strategies	Define the project scope and objectives in the initial project documentation.
	Implement a process that requires formal approval to change the scope.

2. Security and Data Privacy

Risk	Sharing a user's location with unwanted parties or companies instead of only allowing authorized personal to see the user's location.
Mitigation Strategies	Establish database authentication and authorization.
	Allow only authorized parties to access to the PLWD information.

3. Budget

Risk	The project will have a budget set; it is very easy to exceed the budget when proper procedures are not followed.
Mitigation Strategies	Have a clear and defined budget for the project.
	Have team members take extra caution with project materials.

4. Technical Risks

Risk	There is a risk of encountering technical challenges and limitations that could affect the project timeline and success. There is also a risk that the PCB board could fail or not work as intended.
Mitigation Strategies	Assign sufficient time for prototyping and testing.
	Assemble a team with expertise in relevant technologies.
	Establish a timeline and contingency plans or alternative approaches.
	Establish a line of communication with board designer.
	Compare functionality with the individual development boards.

6.0 Timetable and Milestone

The group has planned to complete the following milestones as shown in Table 4 below. It consists of the new milestones set with the new hardware in place. Rows colored in light green indicates completed milestones, while those colored in light yellow indicates milestones currently in progress.

Table 4 Milestones to complete a working prototype

#	Milestone	Brief Description	Deadline
1	UI Development for Mobile App	A fully integrated user interface will be completed for the mobile application, including the following: login, create app, PLWD display, map feature, and settings.	November 15th
2	Create Cloud Database and Link with Mobile App	The cloud database will be fully designed and created, containing the necessary status and location information of each registered user. The	November 31st

		values displayed on the mobile application will now come from the values in the database.	
2	Debugging New Hardware	The new PCB will require testing and debugging to confirm functionality and power usage of the modules.	December 31st
3	Code Functionality for New Hardware	The code that has been passed on from the previous team will be compiled into a git repository without loss of functionality and run on the new hardware.	January 15th
4	Establish End-to-End Communication	Communication will be established between the hardware/software and the database to store the coordinates generated by the tracking device into the database.	February 1st
5	Power Efficiency	Optimize power use across the new PCB. Each module must be optimized in software and testing the power usage on hardware.	March 1st
6	Module Boot-up	Set up proper power on logic for each module. Complete module power prioritization logic (when each module is needed).	March 15th
7	Finalize Mobile App	All interactions on mobile application will be completed and a finalized user interface will be introduced.	March 22nd

6.0 Special Components

The current tracker hardware is made up of several different development boards. These boards include:

STM32: Microcontroller board

Used as central control unit to distribute tasks to all modules.

ESP32: Bluetooth module

Used for location data and communication for the Find My network.

U-BLOX M8C: GPS module

Used for supplying location data.

SparkFun Sara R4: LTE module

Used for communication with user's device.

The future tracker hardware will be a printed circuit board containing all the above modules and a battery management unit. This circuit board was designed in the summer.