GNG5140: Engineering Design

Deliverable H: Design Project User and Product Manual

Submitted by:

Team Opioid Overdose Monitor G2

Ayham AlAkhras 200207406

Min Ju Kim 7296534

Varsha Srinivasan 300157999

April 14, 2021

University of Ottawa

Table of Contents

List of Figures

List of Tables

List of Acronyms and Glossary

Table 1. Acronyms

Acronym	Definition
WHO	World Health Organization
OD	Opioid Overdose
RR	Breathing/Respiratory Rate
IDE	Integrated Development Environment
SpO2	Blood Oxygen Saturation
PPG	Photoplethysmography
LED	Light Emitting Diode
BLE	Bluetooth Low Energy
UI	User Interface

1 Introduction

This User and Product Manual (UPM) provides the information necessary for the effective use and/or development of the Opioid Overdose Monitoring System. Firstly, this UPM provides an overview of the product system: the needs of the user, and the key features of this product. It then goes through a sectioned step-by-step process to set up and disable the device. After which, detailed instructions are given on how use the product and its features. The manual then lists some common troubleshooting issues found by the team and methods to address them. The UPM covers the specifics of the prototype documentation, including a bill of materials (BOM), equipment list, and the process to develop each subsystem. Finally, the UPM concludes by summarizing the team’s work, the lessons learned, and possible future work.

2 Overview

The world is currently going through an opioid crisis. The WHO reported that approximately 115,000 people died of OD in 2017, and approximately 14,000 deaths in Canada between January 2016 and June 2019 [1, 2]. About 75% of the victims were in their home when they died from opioids [3]. The goal of this product is not to solve the opioid crisis, but to prevent people from dying from using opioids.

One of the problems when designing an OD monitor is the stigma and/or discrimination around the substance users that hinder their desire to access existing services such as safe injection sites. Another problem is the fact that harm reduction sites such as the Sandy Hill Community Health Centre's OASIS program are not distributed equally across Canada, aggravated by the current pandemic, that stops users in rural communities to access these services [2].

The key aspects that differentiate our product from others available on MakerRepo (https://makerepo.com/) is the design of a facemask – as shown in Figure 1 – to provide the user with privacy and discreetness, as well as the monitoring of the user's RR to provide a faster and more accurate OD detection.

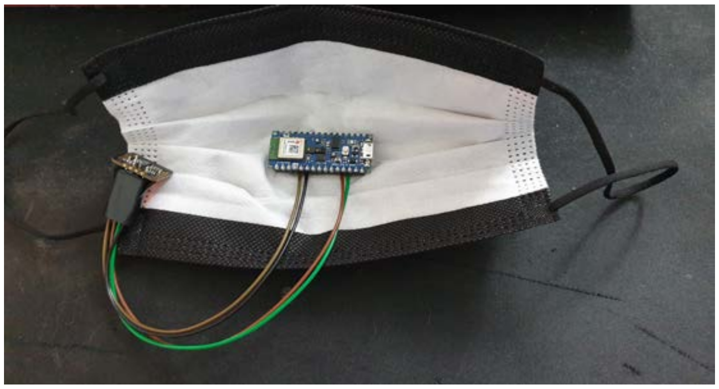


Figure 1. Final Prototype

The product’s key features include:

- the ability to measure the RR for a quicker and more accurate OD detection
- the ability to customize the emergency contact for user privacy
- addition of sliders in the companion app for preventing accidental cancellation of monitoring or alerting the emergency contact of an OD

The hardware system architecture is simple, which is as follows and shown in Figure 2:

1. The sensors measure the raw signals.
2. The raw signal is sent to the microcontroller.
3. The microcontroller processes the raw signals, calculating the vital signs.
4. The microcontroller sends the vital signs to the companion app.

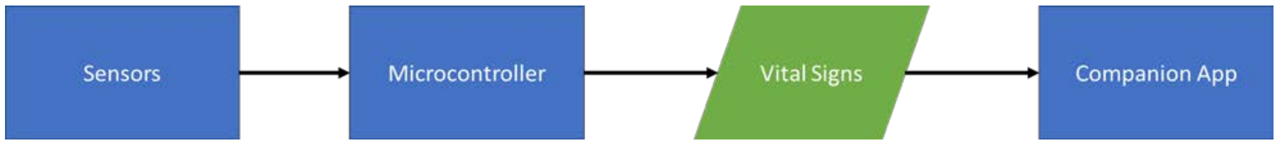


Figure 2. Hardware System Architecture

The companion app system architecture is more complex, which is as follows and shown in more detail in Figure 11:

1. The user starts the app.
2. The user starts the monitoring.
3. The app receives the vital signs from the microcontroller.
4. The app checks for an OD.
5. If an OD is detected, the app calls the emergency contact. The user can cancel this call if they want.
6. The user can stop the monitoring when they want.

2.1 Conventions

When an action is required on the part of the reader, the instructions are listed in a numbered list, and they are in second-person language (i.e., You should…)

2.2 Cautions & Warnings

Please note that the Bluetooth communication between the hardware and the companion app is not implemented.

Also, please note that the MAX30102 pulse oximeter is not functional. If recreating this product using a new MAX30102, please disregard this warning.

Finally, please note that when using or testing the device, the user should be in a dry environment and be in an upright position (i.e., standing or sitting straight).

3 Getting started

3.1 Set-up Considerations

3.1.1 Powering and running the Arduino

To power and run the Arduino, follow these steps:

1. Plug the Arduino into a computer with Arduino IDE installed. For an Arduino Nano 33 BLE Sense, use a data transferrable micro-USB cable. You should see a green LED turned on as shown in Figure 3. If not, you need a new board.

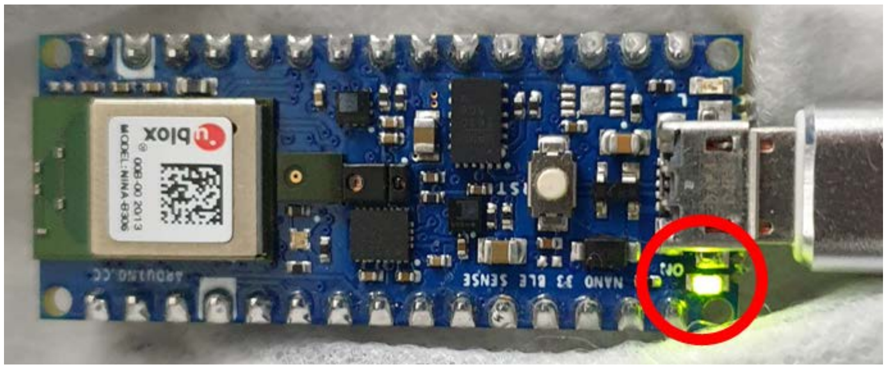
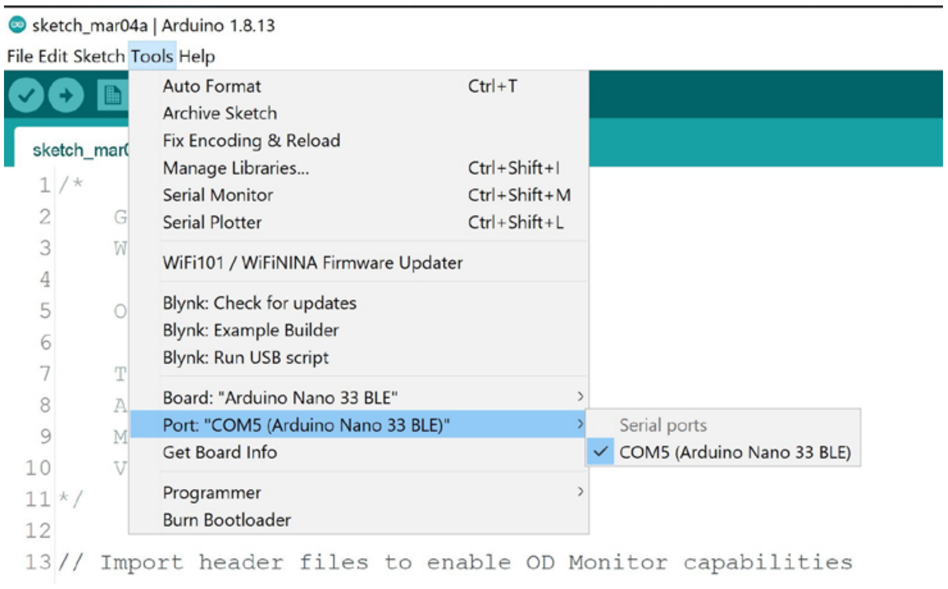


Figure 3. Green LED on Arduino Nano 33 BLE Sense

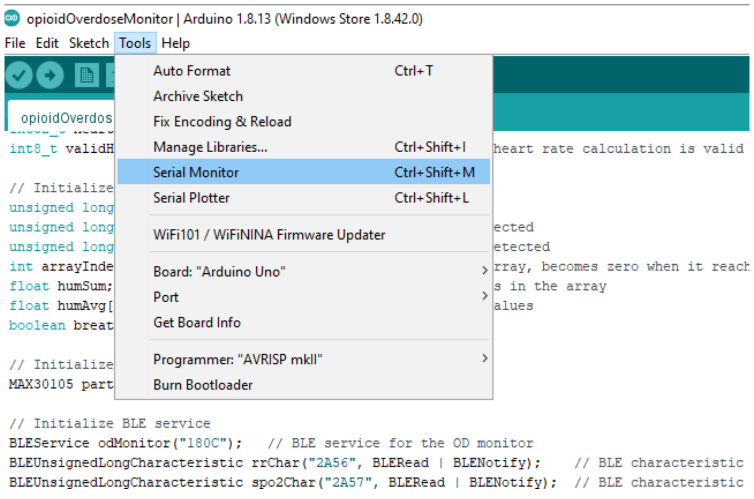
2. Select the port the Arduino is connected to. Usually, the IDE will select the port automatically, but you can manually select the port by clicking Tools → Port.


sketch_mar04a | Arduino 1.8.13
File Edit Sketch Tools Help
sketch_mar0
Auto Format Ctrl+T
Archive Sketch
Fix Encoding & Reload
Manage Libraries... Ctrl+Shift+I
Serial Monitor Ctrl+Shift+M
Serial Plotter Ctrl+Shift+L
WiFi101 / WiFiNINA Firmware Updater
Blynk: Check for updates
Blynk: Example Builder
Blynk: Run USB script
Board: "Arduino Nano 33 BLE"
Port: "COM5 (Arduino Nano 33 BLE)"
Get Board Info
Programmer
Burn Bootloader
Serial ports
COM5 (Arduino Nano 33 BLE)

```
/*
    G
    W

    O

    T
    A
    M
    V
*/

// Import header files to enable OD Monitor capabilities
```

Figure 4. Selecting the Port

3. To see the Arduino at work, open the serial monitor by clicking Tools → Serial Monitor. The system should open a separate window to display the results, i.e., the SpO2 and RR values.


opioidOverdoseMonitor | Arduino 1.8.13 (Windows Store 1.8.42.0)
File Edit Sketch Tools Help
opioidOverdos
Auto Format Ctrl+T
Archive Sketch
Fix Encoding & Reload
Manage Libraries... Ctrl+Shift+I
Serial Monitor Ctrl+Shift+M
Serial Plotter Ctrl+Shift+L
WiFi101 / WiFiNINA Firmware Updater
Board: "Arduino Uno"
Port
Get Board Info
Programmer: "AVRISP mkII"
Burn Bootloader

```
int8_t validH                                              heart rate calculation is valid

// Initialize
unsigned long
unsigned long                                              ected
unsigned long                                              etected
int arrayInde                                              rray, becomes zero when it reach
float humSum;                                              s in the array
float humAvg[                                              alues
boolean breat

// Initialize
MAX30105 part

// Initialize BLE service
BLEService odMonitor("180C");   // BLE service for the OD monitor
BLEUnsignedLongCharacteristic rrChar("2A56", BLERead | BLENotify);    // BLE characteristic
BLEUnsignedLongCharacteristic spo2Char("2A57", BLERead | BLENotify);  // BLE characteristic
```

Figure 5. Serial Monitor and Port in Arduino IDE

3.1.2 Installing the Companion App

To install the companion app, the user must run the attached .apk file titled “Opioid Overdose Monitor.apk” on an Android device. This can be done by connecting the Android device to a PC with the .apk file stored on it. The file can then be copied over to the mobile device. Once copied, simply tapping on the copied file in the file manager will start the installation process.

3.2 User Access Considerations

3.2.1 Age restrictions

This device is not permitted for users under 18 years of age. For exceptions like medical purposes, one must ensure to have written consent from a parent or legal guardian before going ahead. A parent or responsible adult (custodian) must be present during the substance consumption and monitoring using the device.

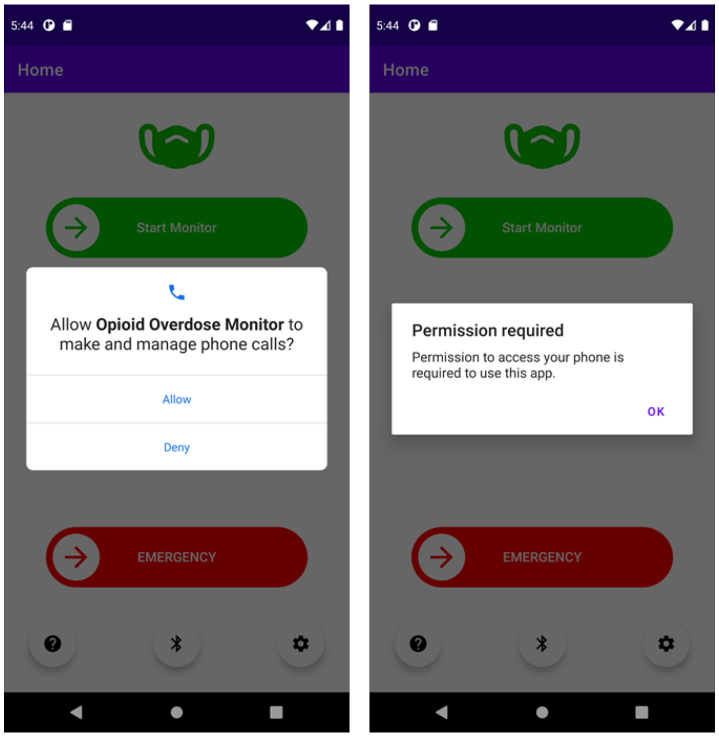
The guardian must protect and promote the welfare of under-18s and vulnerable adults. One must ensure treatments are safe and appropriate and that the individual wants to have the treatment and/or services. Please follow the instructions, protocols and guidelines provided in this user manual for safe usage.

3.2.2 Target audience

Middle-aged men who wish to be monitored discreetly while using substance are identified as the best recipients of this device. The users would be aware and willing to put the device on their body.

3.2.3 Allowing Permissions

The companion application has many features mentioned in section 4. Many of these features utilize special permissions that must be allowed by the user. Specifically features such as location reading, voice call initiation, and SMS sending require user permissions to maintain user privacy and control of their device. Thus, when using the system, the application will prompt the user to allow any needed permissions. If these permissions are rejected by the user, the application will inform the user of the permissions needed and will close.



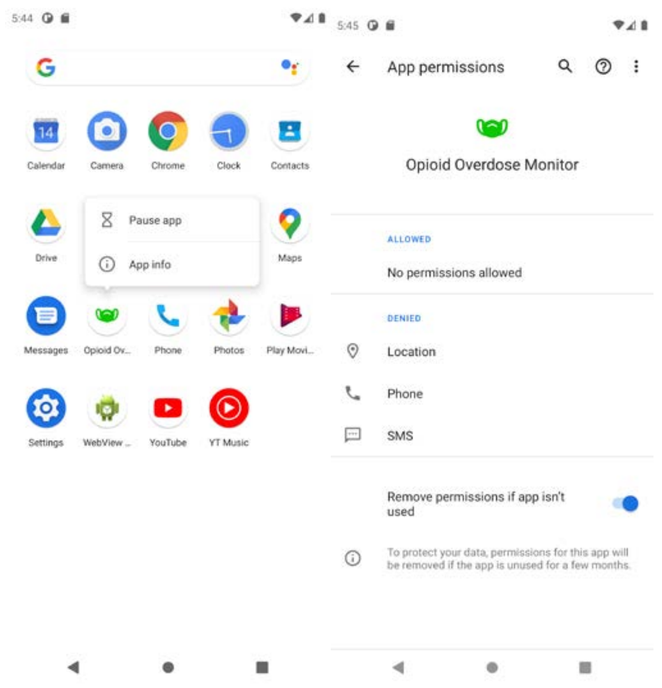
Allow **Opioid Overdose Monitor** to make and manage phone calls?

Permission required

Permission to access your phone is required to use this app.

Figure 6. Denying Permissions

The application cannot afterwards prompt the user to enable the permissions. The user must access the device settings and navigate to the Opioid Overdose Monitor app settings to enable the permissions and use the application with all its features. To do this, the user can first long press on the application in their app drawer then tap on App info. The user can then view the permissions for the app, allowing or denying the required permissions according to their needs by tapping each permission and allowing or denying it.



5:45

App permissions

Opioid Overdose Monitor

ALLOWED

No permissions allowed

DENIED

Location

Phone

SMS

Remove permissions if app isn't used [✓]

To protect your data, permissions for this app will be removed if the app is unused for a few months.

Figure 7. Permission configuration

The targeted market can be identified as people who wish to have a monitoring device on people who are prone to overdose while using the substance, for example, the OASIS program by Sandy Hill Community Health Centre is interested in providing a monitoring device to substance users.

3.3 Accessing the System

3.3.1 Turning on Arduino

Turing on the Arduino is very simple: just connect the board to a computer using a data transferrable micro-USB cable. To see the Arduino running, refer to the step-by-step instructions in *3.1.1 Powering and Running the Arduino*.

3.3.2 Running the App and Starting the Monitor

Starting the application in Android is also simple: once the app is installed as per section 3.1.2, the application can be found in the app drawer with the icon depicted in figure.

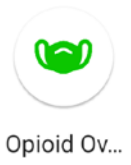


Figure 8. App caption

To initiate the monitoring screen within the application, the user simply slides the start monitor slider all the way to the right until a checkmark appears as shown:

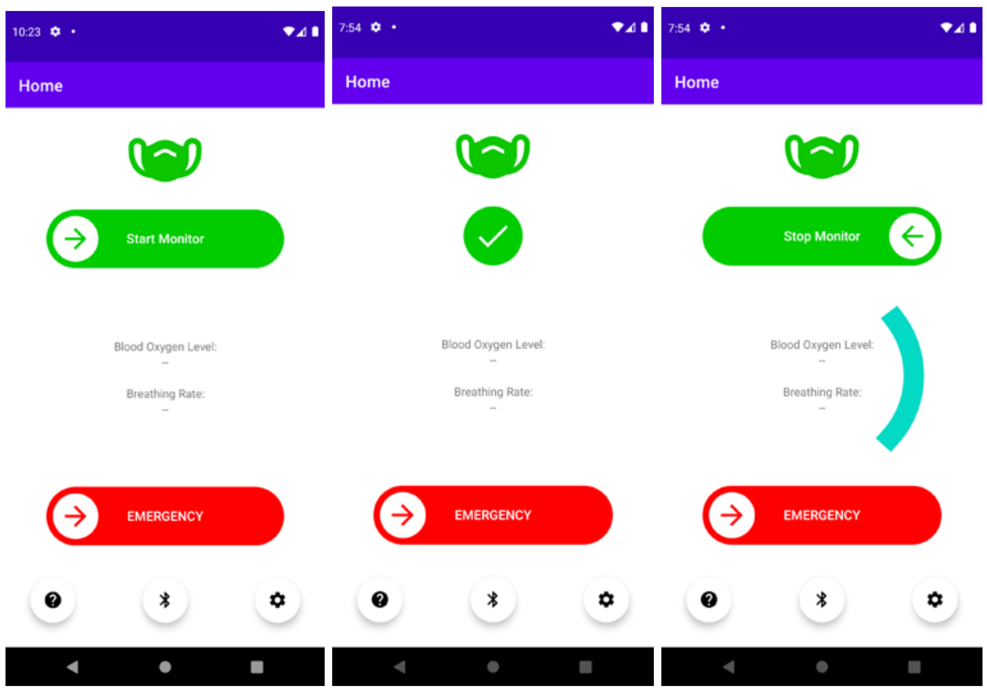


Figure 9. Starting the monitor

3.4 System Organization & Navigation

The companion application navigation is very intuitive. The user interface consists of four main screens that the user can interact with:

1. Home screen: the starting point for the user, navigates to other screens

2. Monitoring screen: displays an animation confirming that monitoring has begun, displays user’s vital signs, and monitors for overdose with the help of the signal processing unit

3. Emergency screen: triggered either manually by the user or automatically by the artificial intelligence onboard, sounds an alarm and calls/texts the user's emergency contacts

4. Settings: can be accessed through home or monitoring but not emergency screen, allows the user to customize their emergency contacts, select if the application calls or texts their emergency contact, and their alarm preferences such as sound and vibration

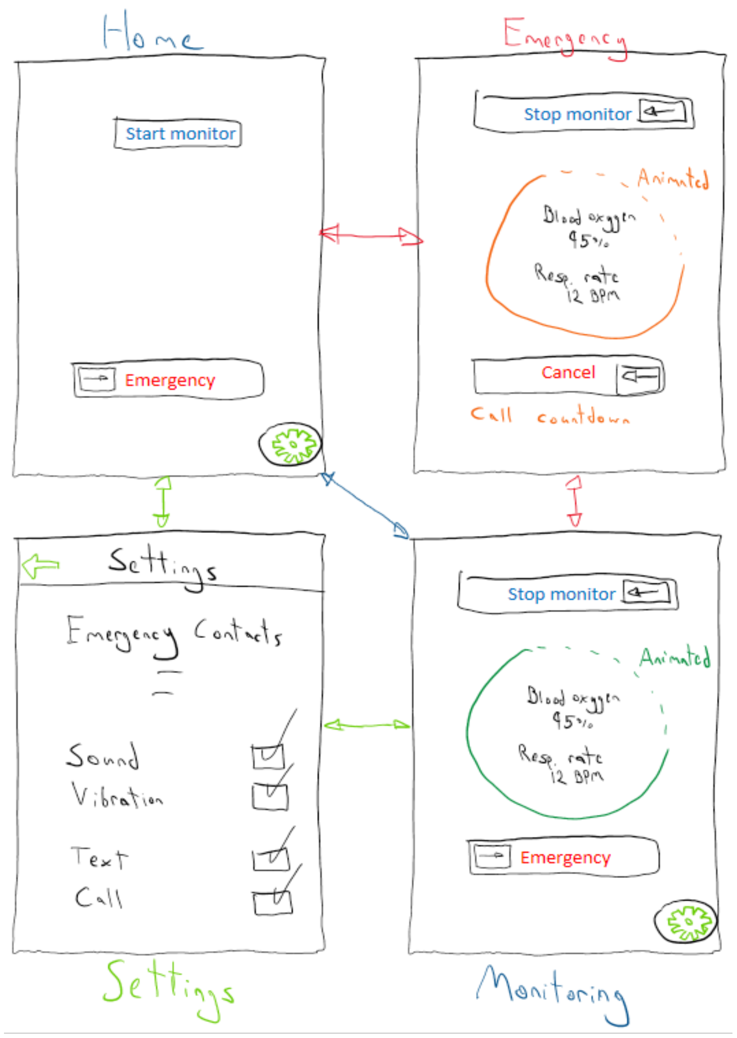


Figure 10. App navigation

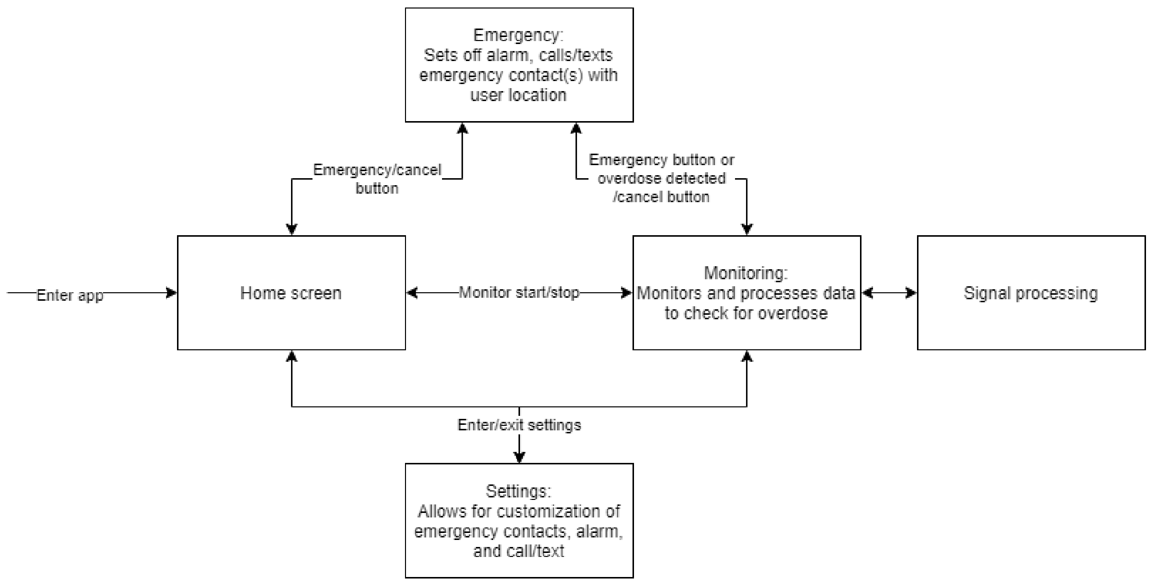

Emergency:
Sets off alarm, calls/texts emergency contact(s) with user location
Emergency/cancel button
Emergency button or overdose detected /cancel button
Enter app
Home screen
Monitor start/stop
Monitoring:
Monitors and processes data to check for overdose
Signal processing
Enter/exit settings
Settings:
Allows for customization of emergency contacts, alarm, and call/text

Figure 11. App flowchart

The application main screen allows the user to:

1. Start/stop the monitoring device

2. Trigger/cancel an emergency

3. Navigate to the application settings

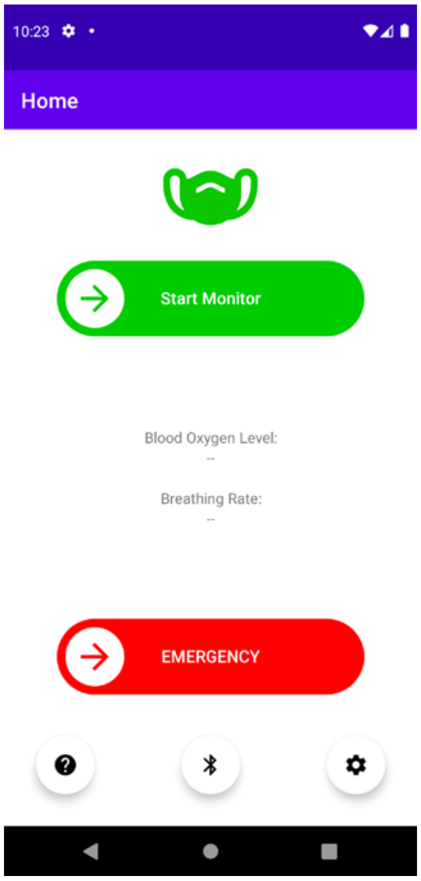
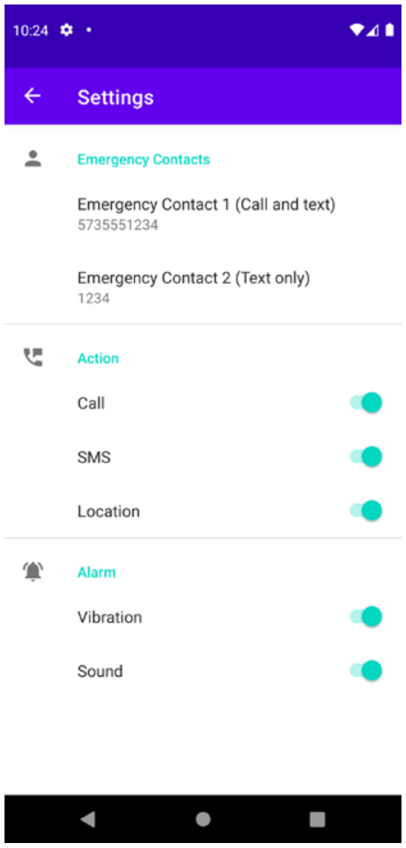


Figure 12. UI Main Screen

When accessed, the application settings screen allows the user to:

1. Customize emergency contacts
2. Customize emergency actions (location enabling was added for this deliverable)
3. Customize the emergency alarm



10:24

Settings

Emergency Contacts

Emergency Contact 1 (Call and text)
5735551234

Emergency Contact 2 (Text only)
1234

Action

Call ●

SMS ●

Location ●

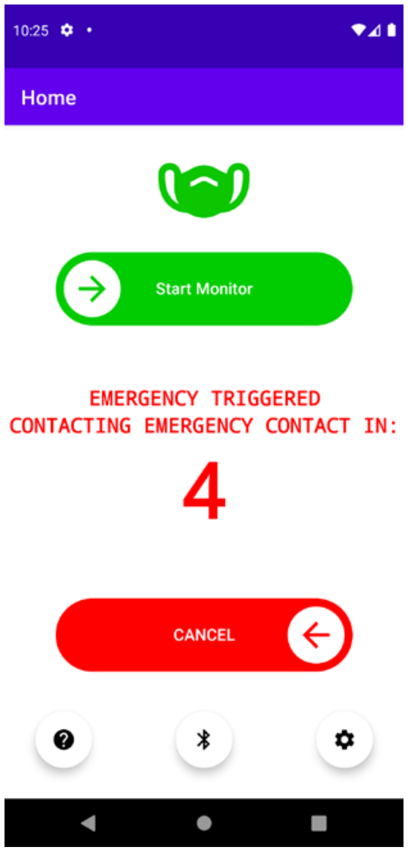
Alarm

Vibration ●

Sound ●

Figure 13. Settings Screen

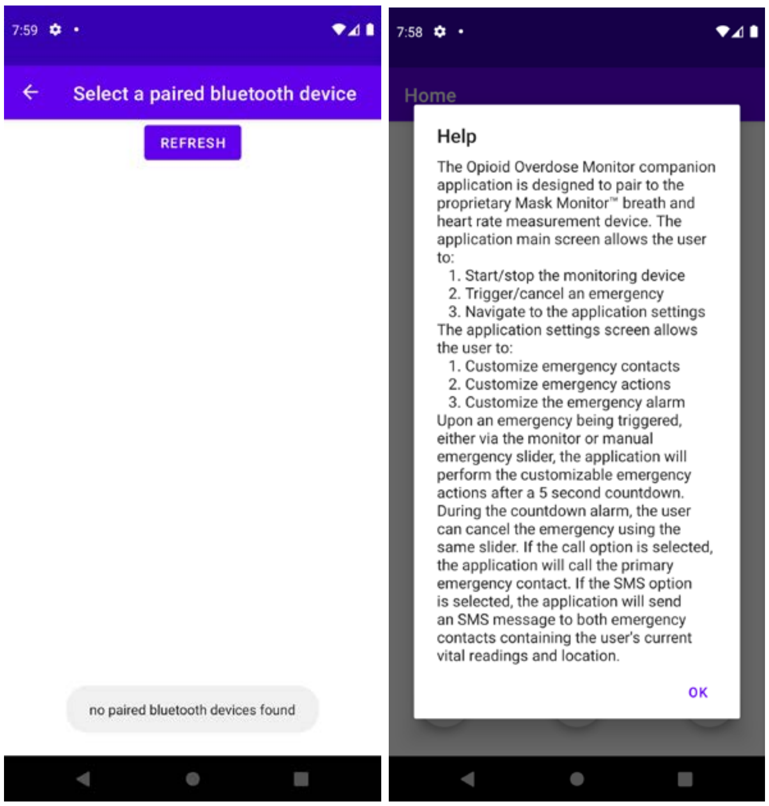
Upon an emergency being triggered, either via the monitor or manual emergency slider, the application will perform the customizable emergency actions after a 5 second countdown. During the countdown alarm, the user can cancel the emergency using the same slider. If the call option is selected, the application will call the primary emergency contact to continuously notify them and bring their attention to their device. If the SMS option is selected, the application will send an SMS message to both emergency contacts containing the user's current vital readings. Similarly, if location is selected, the device will include a current Google Maps location of the user. Lastly, if the options are selected in the settings menu, the application will play a beep and vibrate the device every second of the emergency countdown to indicate to the user that an emergency has been triggered.



EMERGENCY TRIGGERED
CONTACTING EMERGENCY CONTACT IN:
4

Figure 14. Emergency Alarm

Bluetooth and help screens were added in later in development as minor implementations. While the Bluetooth screen was coded in and selection of a device was successfully implemented and tested, the retrieval of data from the device was not possible. The help menu includes a brief description of the app and the ways in which the user can use it. This is a simple pop-up that does not inhibit any other aspects of the app.


7:59
Select a paired bluetooth device
REFRESH
no paired bluetooth devices found
7:58
Home
OK

Help

The Opioid Overdose Monitor companion application is designed to pair to the proprietary Mask Monitor™ breath and heart rate measurement device. The application main screen allows the user to:

1. Start/stop the monitoring device
2. Trigger/cancel an emergency
3. Navigate to the application settings

The application settings screen allows the user to:

1. Customize emergency contacts
2. Customize emergency actions
3. Customize the emergency alarm

Upon an emergency being triggered, either via the monitor or manual emergency slider, the application will perform the customizable emergency actions after a 5 second countdown. During the countdown alarm, the user can cancel the emergency using the same slider. If the call option is selected, the application will call the primary emergency contact. If the SMS option is selected, the application will send an SMS message to both emergency contacts containing the user's current vital readings and location.

Figure 15. Bluetooth and help screens

3.5 Exiting the System

3.5.1 Stop Monitor Slider

Once the monitoring is started, the user will get an option to stop the monitoring as shown in the figure 4. To stop the monitoring, the user must simply slide the button all the way to the left and then the app would go back to the initial state.

3.5.2 Turning off Arduino

To turn off the Arduino, simply plug out the data transferrable micro-USB cable from the computer to disconnect the board from the power source.

3.5.3 Closing the App

To close the app, just exit the app by clicking the back button on the smartphone.

To stop the app from running in the background, close the app from the overview screen, which shows all the open apps. Although this varies for different devices, usually swiping up the recent app or selecting the close all button closes the open apps running in the background.

Under rare circumstances, one may come across a situation where the app would not load properly or glitch no matter how many times it is closed and opened. When this happens, force close the app. To do so, tap and hold the app. Then, once the app starts bouncing and a little menu with options comes up, click on App info. This will take you directly to that app's page in your App Manager. The Force stop button would be on that screen.

Force closing an app can sometimes mess with the integrity or function of an app. If that happens, you may need to uninstall and reinstall the app.

4 Using the System

4.1 Measuring Vital Signs

4.1.1 Measuring SpO2

This product measures the SpO2 using the MAX30102 pulse oximeter. The sensor uses a method called reflective PPG where the light emitted from the LED on the sensor is reflected off the veins into the photodiode, also on the sensor. Figure 16 shows a visualization of reflective PPG.

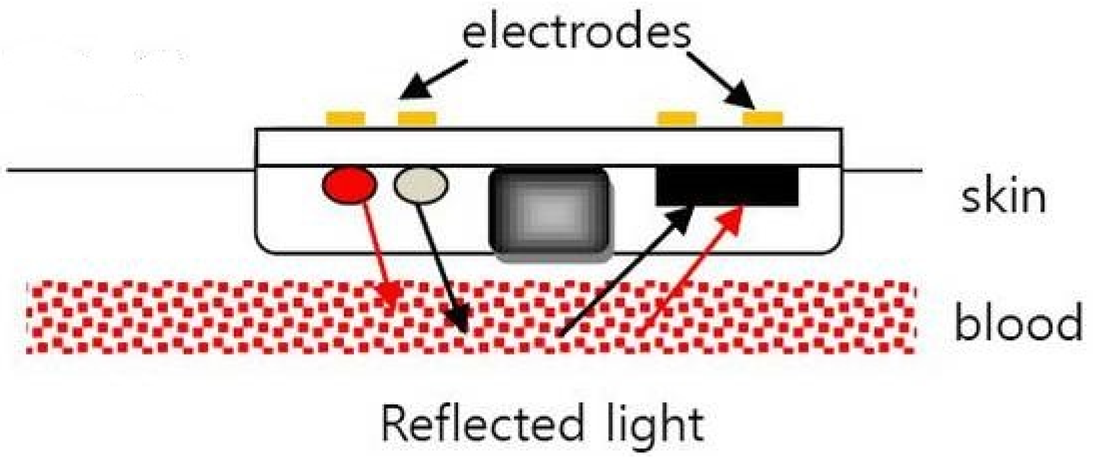


Figure 16. Principles of Reflective PPG [4]

The sensor stores the reflect light from the red and IR LEDs into two arrays redBuffer and irBuffer. The raw output from the red and IR LEDs are shown in Figure 17 [5].

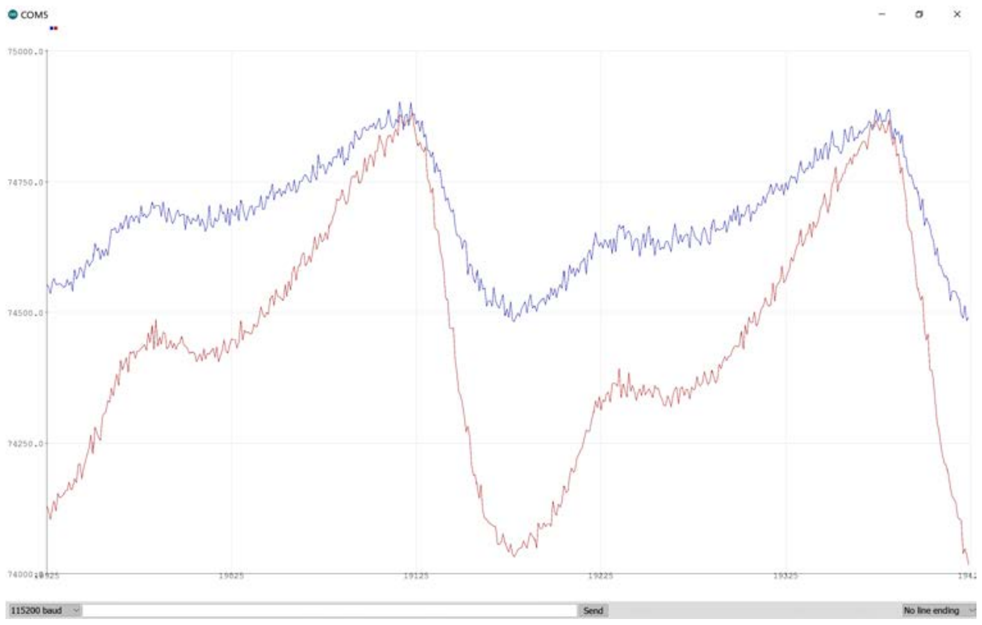


Figure 17. Raw PPG Signal [5]

These arrays are sent to the spo2_algorithm.h available on the SparkFun MAX301x Particle Sensor Library, which can be installed through the Arduino Library Manager as shown in *5.1.1 Arduino Errors* or available online at GitHub [6]. The SpO2 algorithm returns four values:

- spo2: the SpO2 value
- validSPO2: the indicator to show if the SpO2 calculation is valid (1 = valid, 0 = not valid)
- heartRate: the heart rate value
- validHeartRate: the indicator to show if the heart rate calculation is valid.

From these four values, heartRate and validHeartRate are discarded, and spo2 is transmitted via Bluetooth if validSPO2 indicates that the spo2 is valid.

4.1.2 Measuring RR

This product measures RR by first measuring the humidity level in the mask using the HTS221 humidity sensor embedded on the Arduino Nano 33 BLE Sense. As the user breathes, the humidity level fluctuates in sync with their breathing pattern, reaching a peak while exhaling and a valley while inhaling. The humidity signal is then processed with a moving average filter to “smoothen” the curve. Figure 18 shows the plot of the raw humidity signal in blue and the processed humidity signal in red.

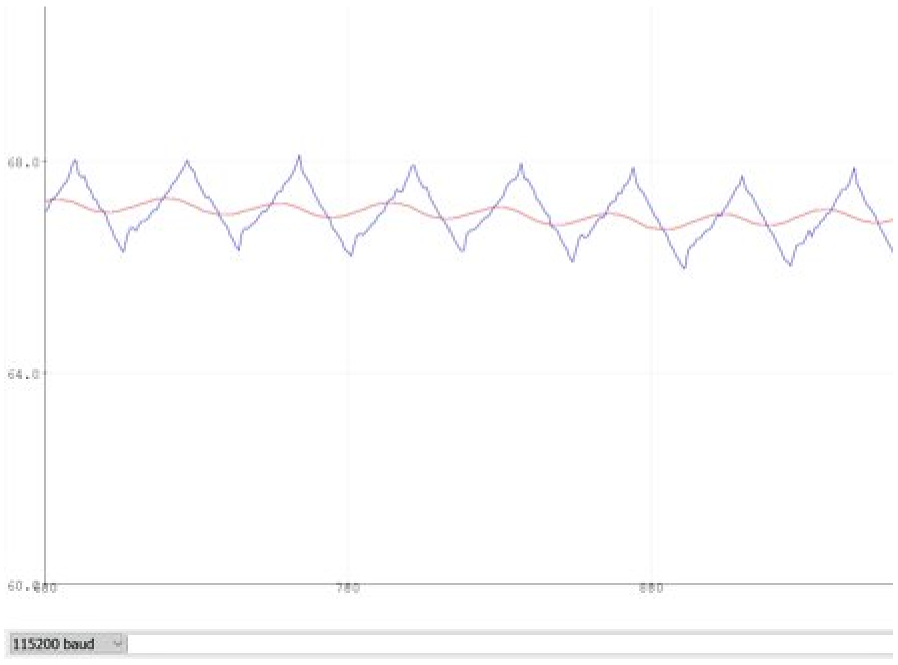


Figure 18. Raw and Processed Humidity Signals

Using the processed humidity signal, RR can be calculated by using the following equation [7]:

$$RR = \frac{30{,}000}{\Delta t} \tag{1}$$

where Δt is the time difference between two breaths (peaks). The RR value can then be transmitted to the companion app via Bluetooth.

4.2 Sending Bluetooth Data

Once the vital signs are measured and verified as shown in *4.1 Measuring Vital Signs*, they are sent to the companion app via Bluetooth using the integrated BLE module on the Arduino Nano 33 BLE Sense. However, as mentioned in *3.4*, the Bluetooth communication between the Arduino board and the companion app could not implemented.

4.3 Confirmation Sliders

The application features slider button functionality. These slider buttons were obtained from a GitHub file online [8]. By sliding these buttons to the right, the user can activate the monitor or initiate an emergency. By sliding the buttons to the left, the user is able to cancel the earlier performed function with the same button. Moreover, once the slide is completed, the button shows a checkmark application to indicate the system status.

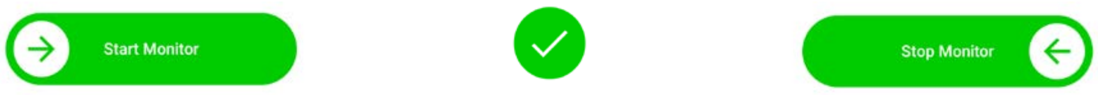


Figure 19. Slider Button Demonstration

4.4 Alarm

Upon an emergency being triggered via the emergency slider, the application will perform the customizable emergency actions after a 5 second countdown. During the countdown alarm, the user can cancel the emergency using the same slider. If the call option is selected, the application will call the primary emergency contact to continuously notify them and bring their attention to their device. If the options are selected in the settings menu, the application will play a beep and vibrate the device every second of the emergency countdown to indicate to the user that an emergency has been triggered.

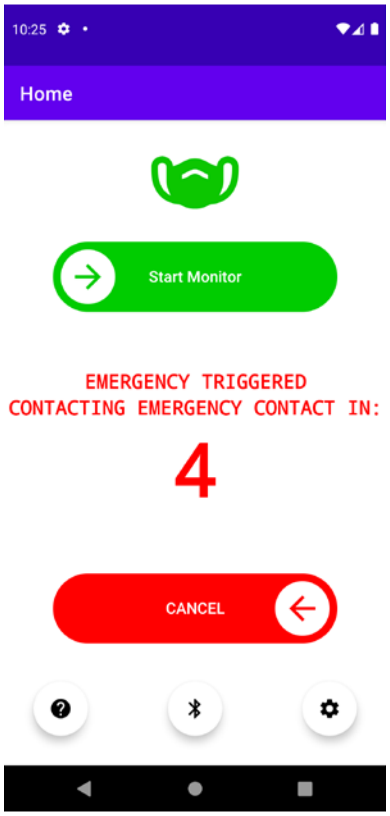


Figure 20. Emergency Alarm

4.5 Emergency Contact

4.5.1 Call

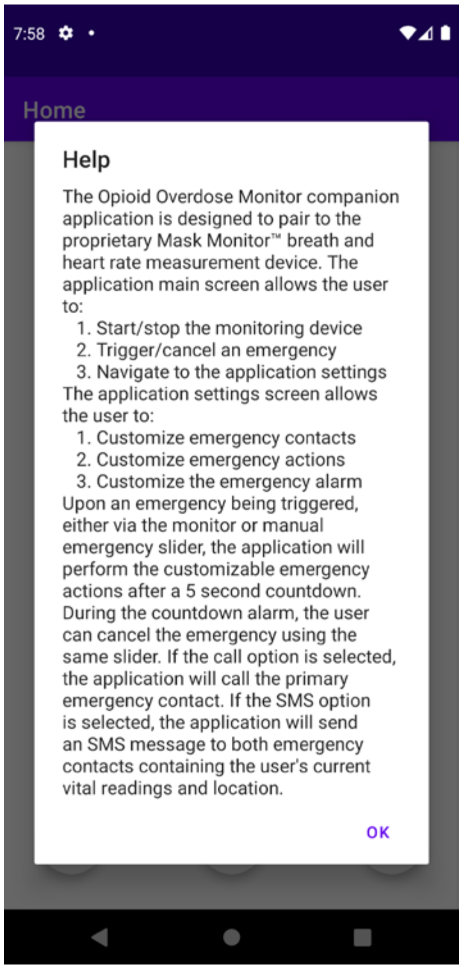
One of the most important features is the implementation of a call function, which automatically dials and calls a customizable number. Completing the emergency slider animation initiates a phone call to the first emergency contact if the user has enabled the option in the settings screen. This serves as a simple alarm for the person receiving the call to notify them of the overdose.

4.5.2 SMS and Location

The call functionality works best to notify the emergency contact to check their device, especially if SMS is enabled. If SMS is enabled, the application sends an SMS message to both emergency contacts with the user’s vital signs. It also informs them that the user’s opioid overdose monitor has been triggered. If location is enabled, another SMS is sent containing the user’s current location as a Google Maps link.

4.6 Help pop-up

The help menu simply includes a brief description of the app and the ways in which the user can use it. This is a simple pop-up that does not inhibit any other aspects of the app.


7:58

Home

Help

The Opioid Overdose Monitor companion application is designed to pair to the proprietary Mask Monitor™ breath and heart rate measurement device. The application main screen allows the user to:

1. Start/stop the monitoring device
2. Trigger/cancel an emergency
3. Navigate to the application settings

The application settings screen allows the user to:

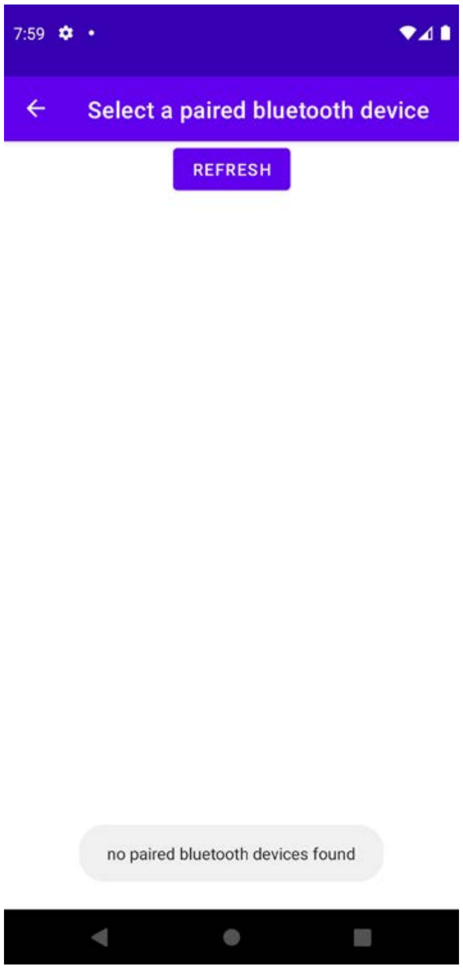
1. Customize emergency contacts
2. Customize emergency actions
3. Customize the emergency alarm

Upon an emergency being triggered, either via the monitor or manual emergency slider, the application will perform the customizable emergency actions after a 5 second countdown. During the countdown alarm, the user can cancel the emergency using the same slider. If the call option is selected, the application will call the primary emergency contact. If the SMS option is selected, the application will send an SMS message to both emergency contacts containing the user's current vital readings and location.

Figure 21. Help Pop-up

4.7 Bluetooth

While the Bluetooth screen was coded and selection of a paired device was successfully implemented and tested, the retrieval of data from the device was not implemented.

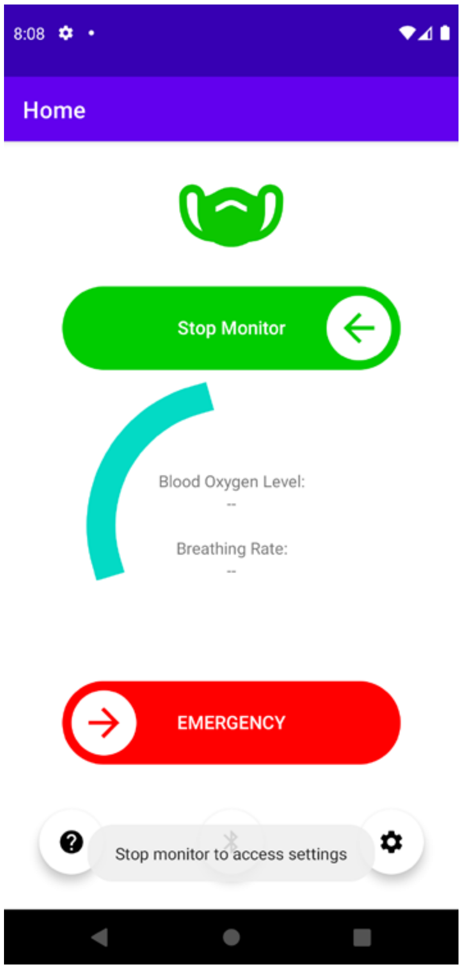


no paired bluetooth devices found

Figure 22: Bluetooth screen

4.8 Crash prevention

It is important to note that the settings menu is inaccessible while the monitor and/or emergency is running. This is to prevent crashes, as a glitch was found when switching screen fragments during slider causing the app to crash. Furthermore, switching fragments also stops the monitor. Therefore, as a safety precaution, the user must switch the monitor off manually. This could be improved by restructuring the whole program and navigation with the implementation of a monitor that can run in the background.



Stop monitor to access settings

Figure 23. Settings inaccessibility

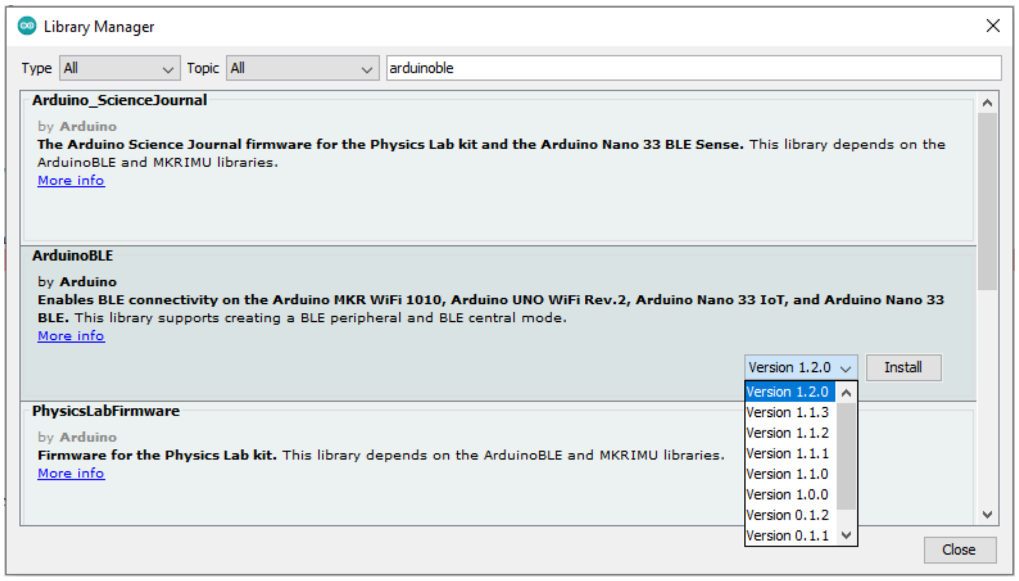
5 Troubleshooting & Support

5.1 Error Messages or Behaviors

5.1.1 Arduino Errors-jack

No such file or directory

1. Make sure that the file or library is installed on your computer. You can check by clicking Tools → Manage Libraries… and clicking on the Type and set it to Installed.
2. To install a library, search for the file or library in the search bar, select the version (usually the newest version is already selected) and click the Install button.


Library Manager

Type: All   Topic: All   arduinoble

Arduino_ScienceJournal
by Arduino
The Arduino Science Journal firmware for the Physics Lab kit and the Arduino Nano 33 BLE Sense. This library depends on the ArduinoBLE and MKRIMU libraries.
More info

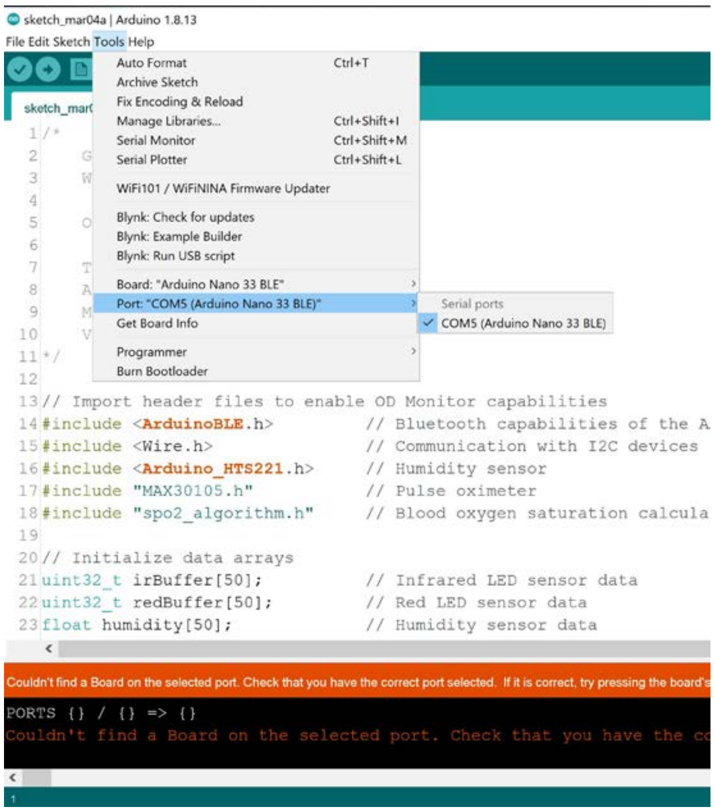
ArduinoBLE
by Arduino
Enables BLE connectivity on the Arduino MKR WiFi 1010, Arduino UNO WiFi Rev.2, Arduino Nano 33 IoT, and Arduino Nano 33 BLE. This library supports creating a BLE peripheral and BLE central mode.
More info

PhysicsLabFirmware
by Arduino
Firmware for the Physics Lab kit. This library depends on the ArduinoBLE and MKRIMU libraries.
More info

Figure 24. Installing Arduino Library

Couldn’t find a Board on the selected port….

1. Make sure that the Arduino is connected to your computer and you have the correct port selected. You can check by clicking Tools → Port and selecting the port where the Arduino is connected to.


sketch_mar04a | Arduino 1.8.13
File Edit Sketch Tools Help
sketch_mar0
Auto Format Ctrl+T
Archive Sketch
Fix Encoding & Reload
Manage Libraries... Ctrl+Shift+I
Serial Monitor Ctrl+Shift+M
Serial Plotter Ctrl+Shift+L
WiFi101 / WiFiNINA Firmware Updater
Blynk: Check for updates
Blynk: Example Builder
Blynk: Run USB script
Board: "Arduino Nano 33 BLE"
Port: "COM5 (Arduino Nano 33 BLE)"
Get Board Info
Programmer
Burn Bootloader
Serial ports
COM5 (Arduino Nano 33 BLE)

```
// Import header files to enable OD Monitor capabilities
#include <ArduinoBLE.h>          // Bluetooth capabilities of the A
#include <Wire.h>                // Communication with I2C devices
#include <Arduino_HTS221.h>      // Humidity sensor
#include "MAX30105.h"            // Pulse oximeter
#include "spo2_algorithm.h"      // Blood oxygen saturation calcula

// Initialize data arrays
uint32_t irBuffer[50];           // Infrared LED sensor data
uint32_t redBuffer[50];          // Red LED sensor data
float humidity[50];              // Humidity sensor data
```

Couldn't find a Board on the selected port. Check that you have the correct port selected. If it is correct, try pressing the board's

```
PORTS {} / {} => {}
Couldn't find a Board on the selected port. Check that you have the co
```

Figure 25. Checking the Arduino Port

5.2 Special Considerations

5.2.1 Environmental considerations

Due to the use of the Arduino NANO 33 BLE Sense board as the microcontroller, it is important to not use the device when the mask is wet or the region in contact with the board is at a risk of short circuit due to the moisture.

5.2.2 Positioning of the device

For effective monitoring, the user has to make sure that the mask is placed effectively. The sensor on the ear loop should be placed on the earlobe effectively and the mask should be worn to make sure that the microcontroller is in proximity with the nostrils to read the respiratory rate.

5.2.3 Slider Glitch

A glitch was found with the slider buttons used that crashes the application if the settings menu is accessed during an animation. This glitch was rectified in the code by preventing access to the settings menu during slider animation. Take consideration of this if adding more menus or more sliders.

5.3 Maintenance

To ensure a long lifecycle of this product, please ensure that the product is kept in a cool and dry environment.

Also, when placing the product in a new facemask, make sure that the jumper wires do not cross-contact onto other pins to prevent shoring the board.

5.4 Support

To get further clarifications about this device please contact the following people based on the roles described below:

User Interface: Ayham AlAkhras (aalak012@uottawa.ca)
Signal Processing: Min Ju Kim(jkim221@uottawa.ca)
Electrical: Varsha Srinivasan(varsha.srinivasan@uottawa.ca)
Components: Justine Lucie Boudreau <jboud030@uottawa.ca> (jboud030@uottawa.ca)

Documentation and design: Patrick Dumond (pdumond@uottawa.ca) and David Knox(dknox@uottawa.ca)

6 Product Documentation

6.1 Enclosure

6.1.1 BOM (Bill of Materials)

The device can be adapted to any mask currently in possession with the user. The cost of the mask is not included as a part of the device manufacturing.

6.1.2 Equipment list

Table 2. Equipment List for Enclosure

BOM Level	Part Name	Quantity	Unit of Measurement	Cost (CA$)	URL
1	Face Mask	1	Each	0	Already in Possession
Total				0	

6.1.3 Instructions

1. To place the mask for effective monitoring make sure that the microcontroller is on the inside and the sensor is placed on the earlobe.
2. Pick up the device and place it gently on the face.
3. Turn on the Arduino board by following the instructions on the next section.

4. Start the monitoring from the app by sliding the start monitoring button.

The most important non-performance feature of this device is its discreetness, comfortability, and freedom of the user’s hands. Due to the stigma and discrimination around opioid users, it prevents them from accessing services that reduce the risk of an OD such as a safe injection site. Moreover, if the device were uncomfortable to wear for extended periods of time, it could sway the user from using the device when using.

To hold the blood oxygen and breath rate measurement sensors and associated circuitry, multiple discrete enclosures were brainstormed by the team. Devices were brainstormed according to the measurement requirements. The blood oximeter chosen must be placed on a thin piece of skin to effectively evaluate oxygen saturation in the blood; usually, blood oximeters are placed on a patient’s finger, toe, or earlobe.

Breathing rate can be measured in one of three ways: via a sonar system, with a respiratory stretch sensor placed on the patient's chest or using a humidity/pressure sensor near the patient's mouth/nose. The sonar system poses many challenges and accuracy flaws in a variety of test cases; thus, it was eliminated from consideration. Furthermore, the stretch sensor was deemed too complex, inconvenient, and expensive to include in the device and was thus also eliminated. All proposed devices could discreetly measure blood oxygen level but not respiratory rate without the use of a separate measurement device.

The proposed designs included a pair of gloves, a shoe (seen in a previous project), a watch, and a pair of over-ear headphones. While these designs all offered discretion, none offered an effective respiratory rate measurement system. Thus, it was decided that this device will utilize a face mask to conceal all components and measurement systems. The face mask covers the microcontroller, a pulse oximeter is placed on the strap of the mask to be clipped on the user's earlobe, and a humidity sensor in the mask is used to measure the user's breathing pattern. Figure 26 shows the ergonomic concept of this prototype.

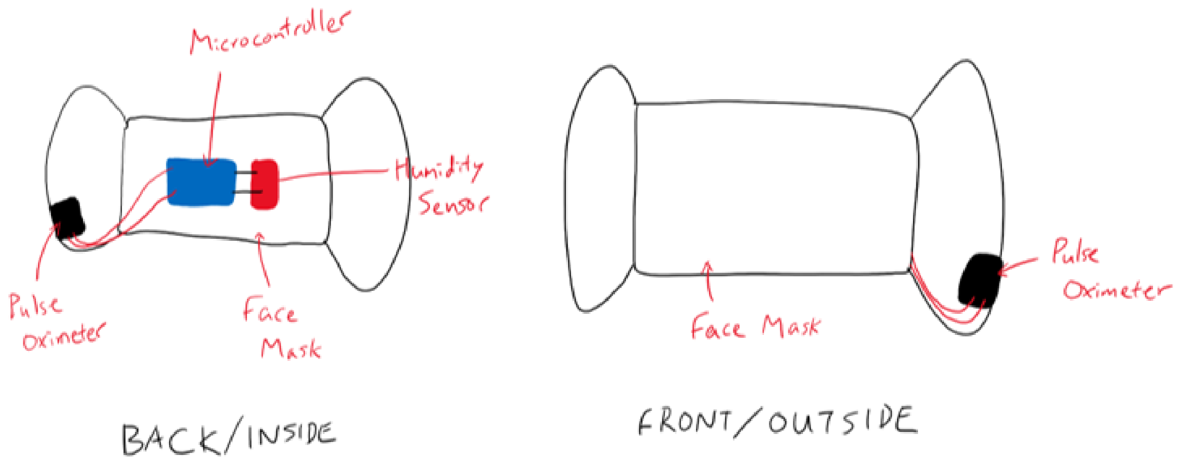


Figure 26: Ergonomic design concept

6.2 Electrical System

6.2.1 BOM (Bill of Materials)

Table 3. Electrical BOM

BOM Level	Part Name	Quantity	Unit of Measurement	Cost (CA $)	URL
1	Arduino Nano 33 BLE Sense	1	Each	57.05	https://www.amazon.ca/Arduino-Nano-Sense-headers-Mounted/dp/B07WXKDVTL
2	MAX30102	1	Each	17.42	https://www.amazon.ca/MakerFocus-Heart-Rate-MAX30102-Compatible-Arduino/dp/B07GGSM8SP
3	Mini Breadboard	1	Each	0	Provided from Makerspace
4	Jumper Wire	4	Each	0	Provided from Makerspace
6	Arduino BLE Library	1	Each	0	https://github.com/arduino-libraries/ArduinoBLE
7	Arduino Wire Library	1	Each	0	https://github.com/arduino/ArduinoCore-avr/tree/master/libraries/Wire
8	Arduino HTS221 Library	1	Each	0	https://github.com/arduino-libraries/Arduino_HTS221
9	SparkFun MAX30101 0x Sensor Library	1	Each	0	https://github.com/sparkfun/SparkFun_MAX3010x_Sensor_Library

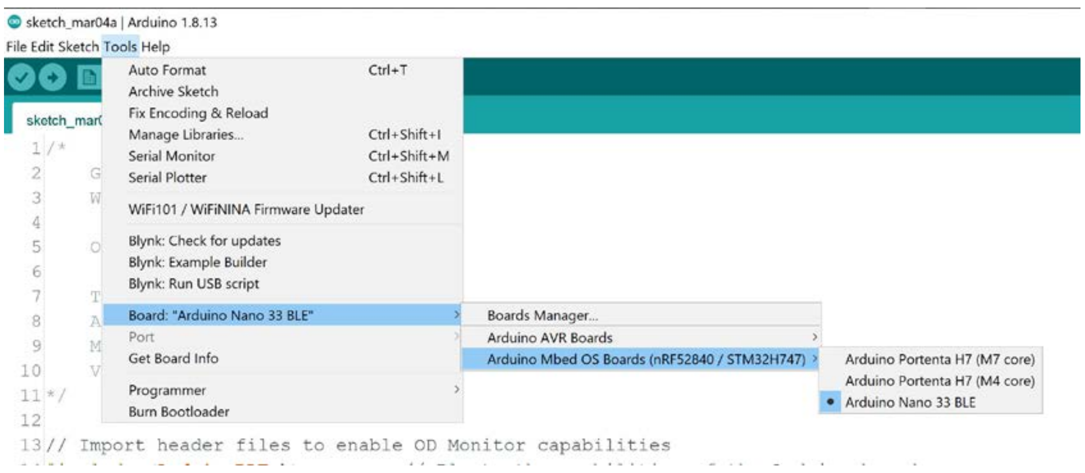
Total				74.47	

6.2.2 Equipment list

BOM Level	Equipment Name	Quantity	Unit of Measurement	Cost (CA$)	URL
1	Personal Computer	1	Each	0	Already in possession
2	Arduino IDE	1	Each	0	https://www.arduino.cc/en/software
3	Data transferrable micro-USB cable	1	Each	0	Already in possession
Total				0	

6.2.3 Instructions

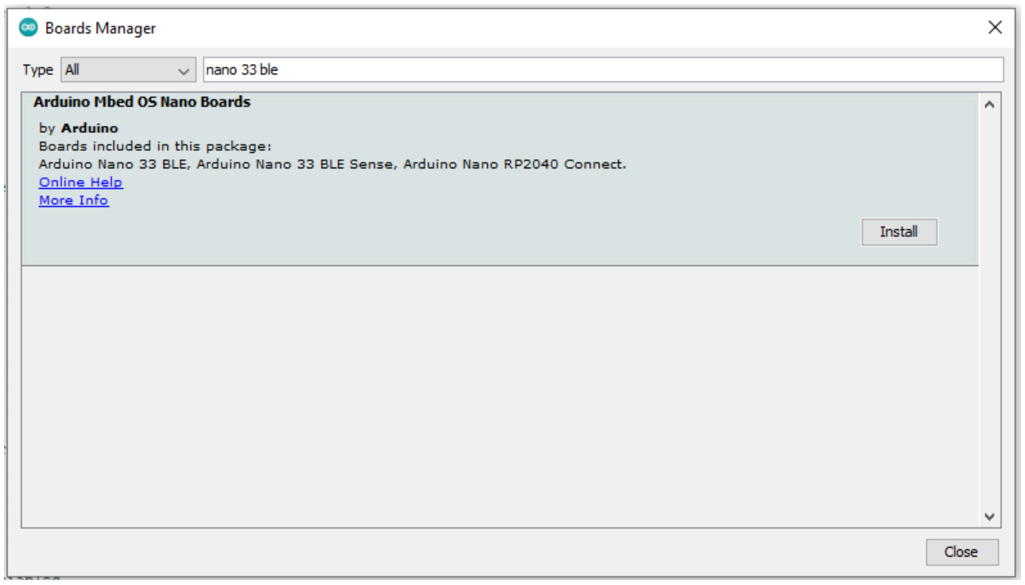
1. Using the data transferrable micro-USB cable, plug in the Arduino Nano 33 BLE Sense to a PC with the Arduino IDE installed. Make sure that the Arduino board is powered on by checking if the green LED is turned on.
2. Check that the port with the Arduino is selected by following the steps in *5.1.1 Arduino Errors*.
3. Check that all the libraries required for this product is installed on your computer by following the steps in *5.1.1 Arduino Errors*.
4. Set the board as Arduino Nano 33 BLE by clicking Tools → Board → Arduino Mbed OS Boards (nRF52840 / STM32h747) → Arduino Nano 33 BLE.


sketch_mar04a | Arduino 1.8.13
File Edit Sketch Tools Help
Auto Format Ctrl+T
Archive Sketch
Fix Encoding & Reload
Manage Libraries... Ctrl+Shift+I
Serial Monitor Ctrl+Shift+M
Serial Plotter Ctrl+Shift+L
WiFi101 / WiFiNINA Firmware Updater
Blynk: Check for updates
Blynk: Example Builder
Blynk: Run USB script
Board: "Arduino Nano 33 BLE"
Port
Get Board Info
Programmer
Burn Bootloader
Boards Manager...
Arduino AVR Boards
Arduino Mbed OS Boards (nRF52840 / STM32H747)
Arduino Portenta H7 (M7 core)
Arduino Portenta H7 (M4 core)
● Arduino Nano 33 BLE

```
// Import header files to enable OD Monitor capabilities
```

Figure 27: Setting up Arduino Board

If the dropdown menu for Arduino Mbed OS Boards (nRF52840 / STM32h747) is not visible, install the drivers by clicking Tools → Board → Boards Manager, searching for “nano 33 ble”, and installing the Arduino Mbed OS Nano Boards driver.


Boards Manager

Type: All — nano 33 ble

Arduino Mbed OS Nano Boards

by **Arduino**
Boards included in this package:
Arduino Nano 33 BLE, Arduino Nano 33 BLE Sense, Arduino Nano RP2040 Connect.
Online Help
More Info

Figure 28. Installing the Arduino Nano 33 BLE Sense

5. Upload the opioidOverdoseMonitor.ino file to the Arduino board.

6.3 User Interface

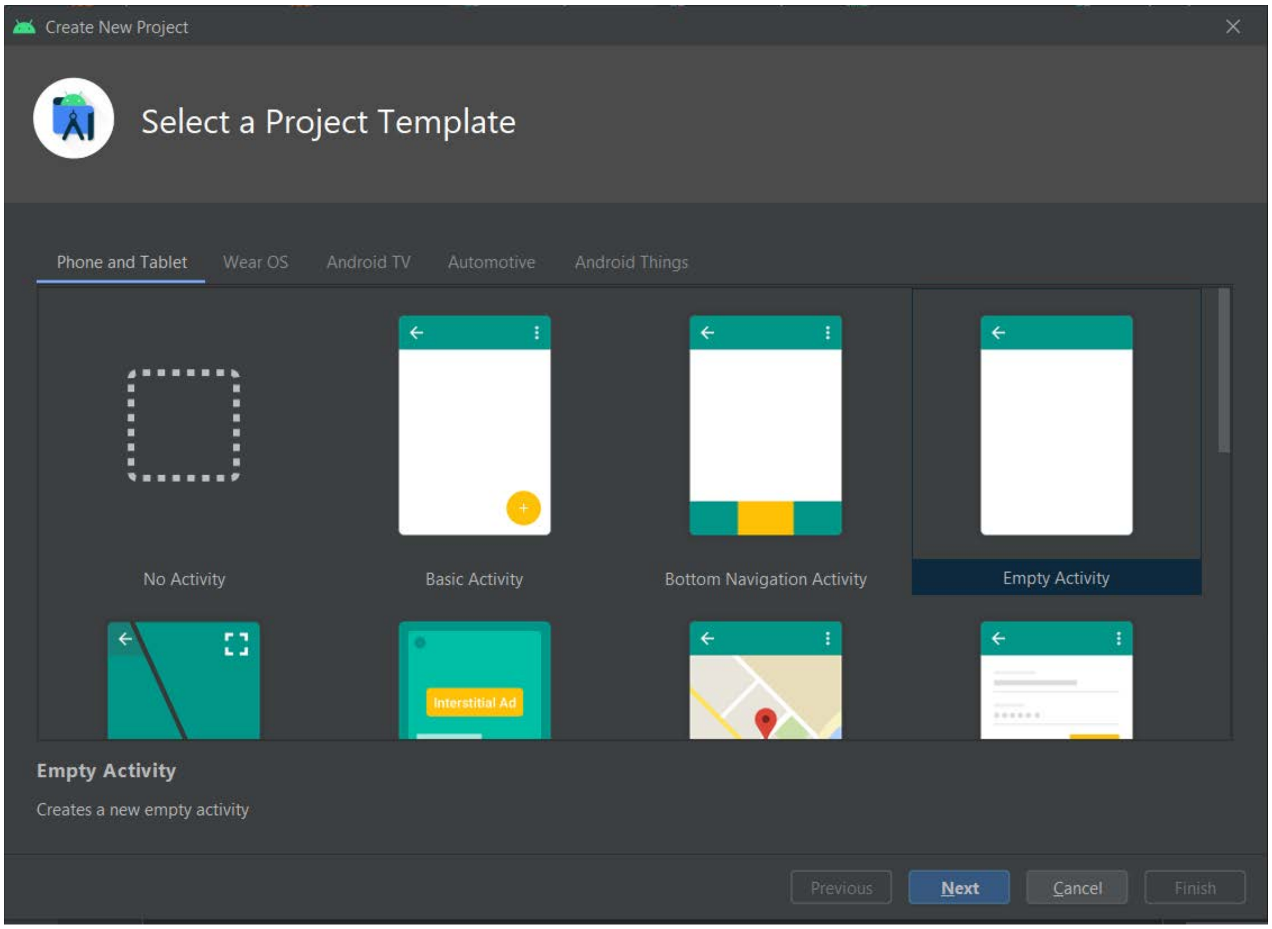
6.3.1 Equipment list

BOM Level	Equipment Name	Quantity	Unit of Measurement	Cost (CA$)	URL
1	Personal Computer	1	Each	0	Already in possession
2	Android Studio	1	Each	0	https://developer.android.com/studio
Total				0	

6.3.2 Instructions

The user interface is the main form of communication between the system and the user. For this UI in particular, simplicity and reliability are key. The application does not need an extremely wide range of functionality, it just has to do the required tasks and do them effectively. The design also must be minimalistic to ensure that users are not confused by its use in critical situations.

To build this system Android Studio was selected for its popularity, functionality, and available tutorials. Once installed, Android Studio allows the user to create a new project. Android Studio allows the user to utilize many pre-built activities.

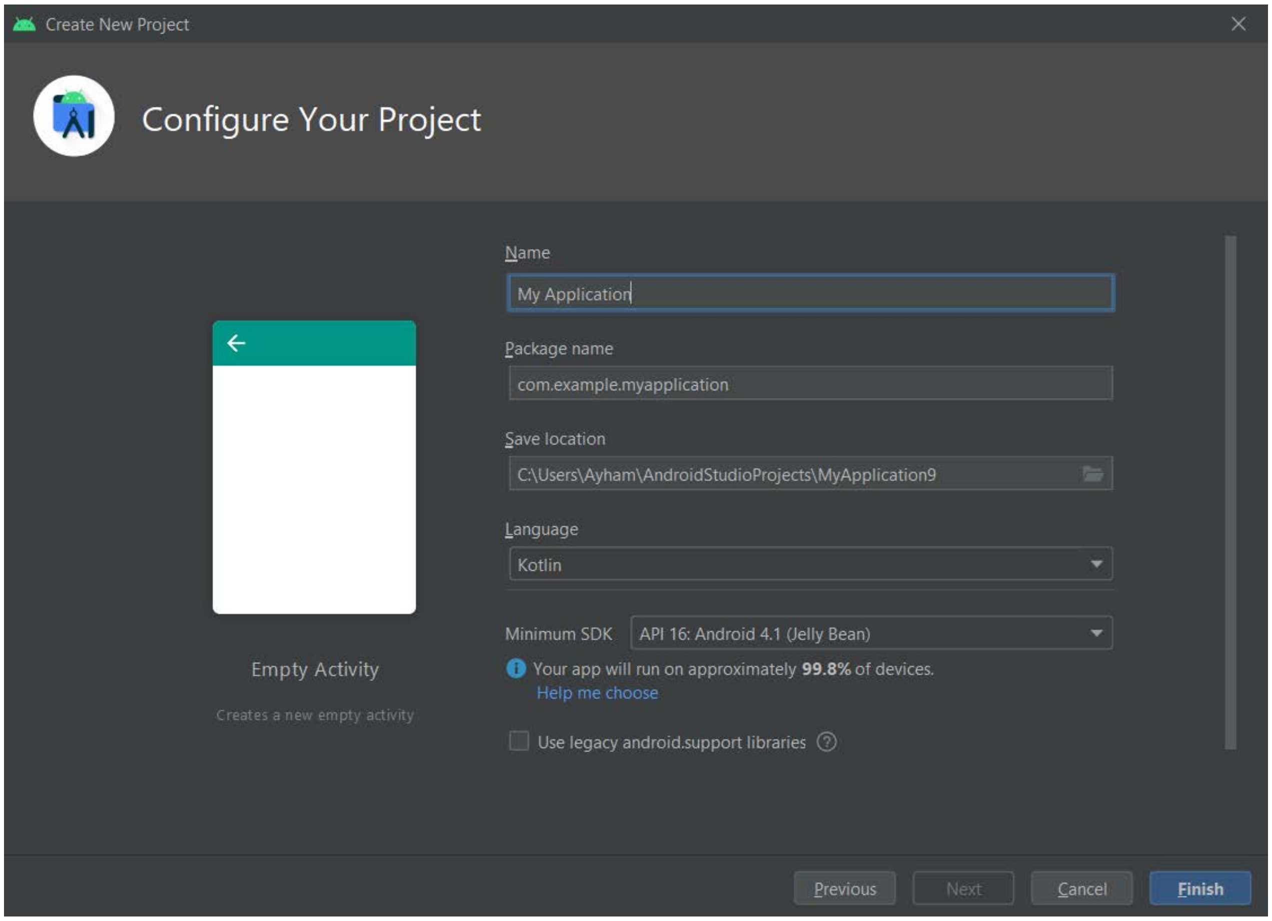


Empty Activity

Creates a new empty activity

Figure 29: Project templates

For this sub-system, an empty activity was used. The language was also selected as Kotlin and the minimum SDK, the minimum version of Android that can run the app, was selected as the oldest SDK available. Kotlin was selected as the preferred programming language as it is the current industry standard and is widely used for Android applications.



Name: My Application

Package name: com.example.myapplication

Save location: C:\Users\Ayham\AndroidStudioProjects\MyApplication9

Language: Kotlin

Minimum SDK: API 16: Android 4.1 (Jelly Bean)

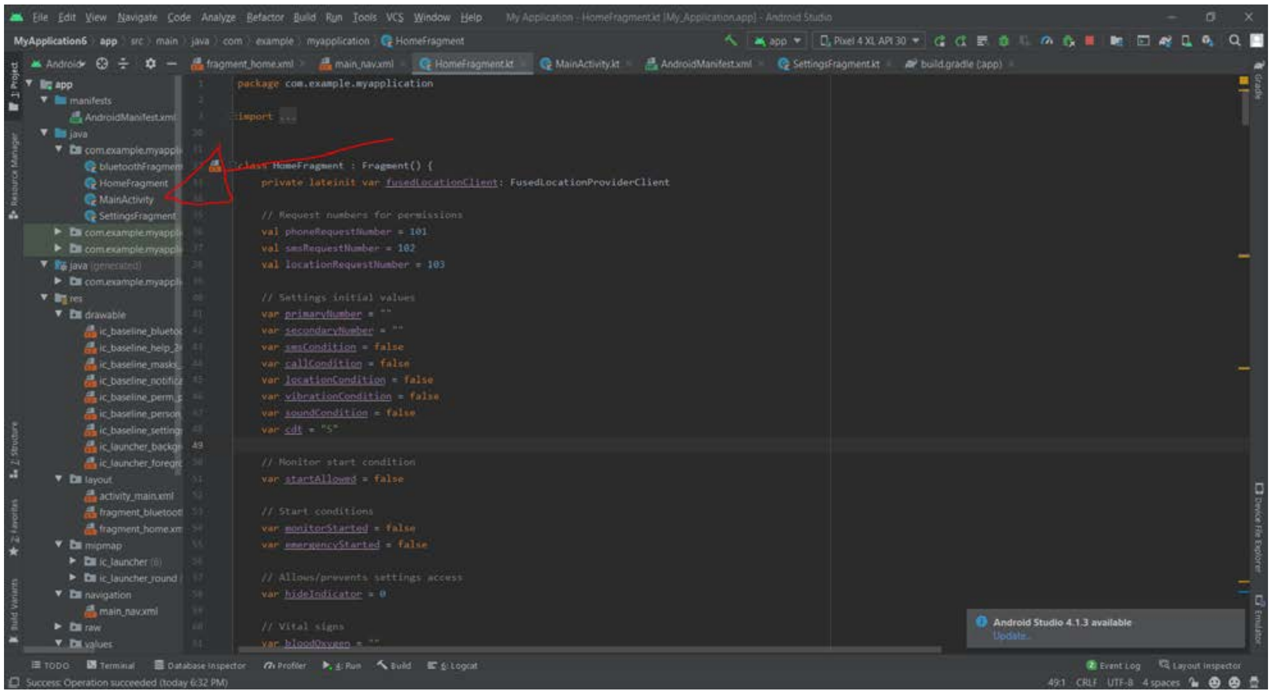
Your app will run on approximately **99.8%** of devices.
Help me choose

Use legacy android.support libraries

Figure 30: File creation

If modifying the attached application attached, the Android Studio can be extracted from the .rar file and placed at another location. This location can then be accessed by utilizing the top toolbar in Android Studio and navigating to file then open.

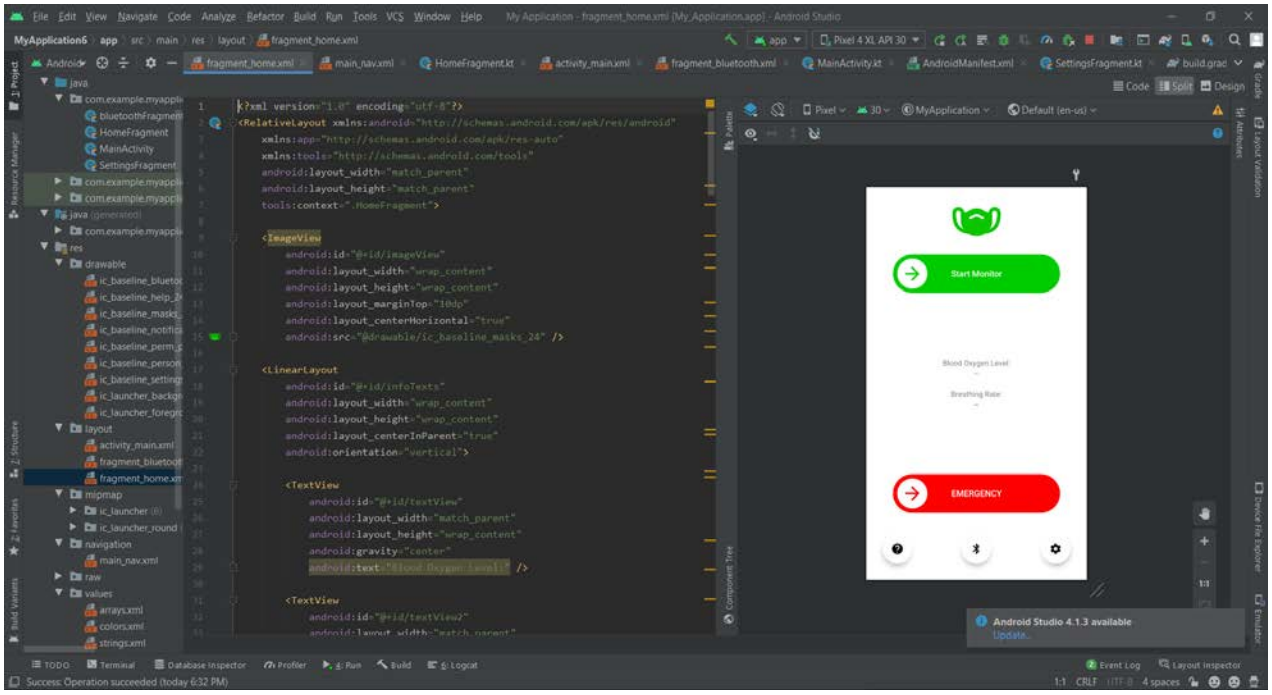
Once opened, the user can navigate around the Android Studio file using the navigation menu on the left.



```
package com.example.myapplication

import ...

class HomeFragment : Fragment() {
    private lateinit var fusedLocationClient: FusedLocationProviderClient

    // Request numbers for permissions
    val phoneRequestNumber = 101
    val smsRequestNumber = 102
    val locationRequestNumber = 103

    // Settings initial values
    var primaryNumber = ""
    var secondaryNumber = ""
    var smsCondition = false
    var callCondition = false
    var locationCondition = false
    var vibrationCondition = false
    var soundCondition = false
    var cdt = "5"

    // Monitor start condition
    var startAllowed = false

    // Start conditions
    var monitorStarted = false
    var emergencyStarted = false

    // Allows/prevents settings access
    var hideIndicator = 0

    // Vital signs
    var bloodOxygen = ""
```

Figure 31: Main layout

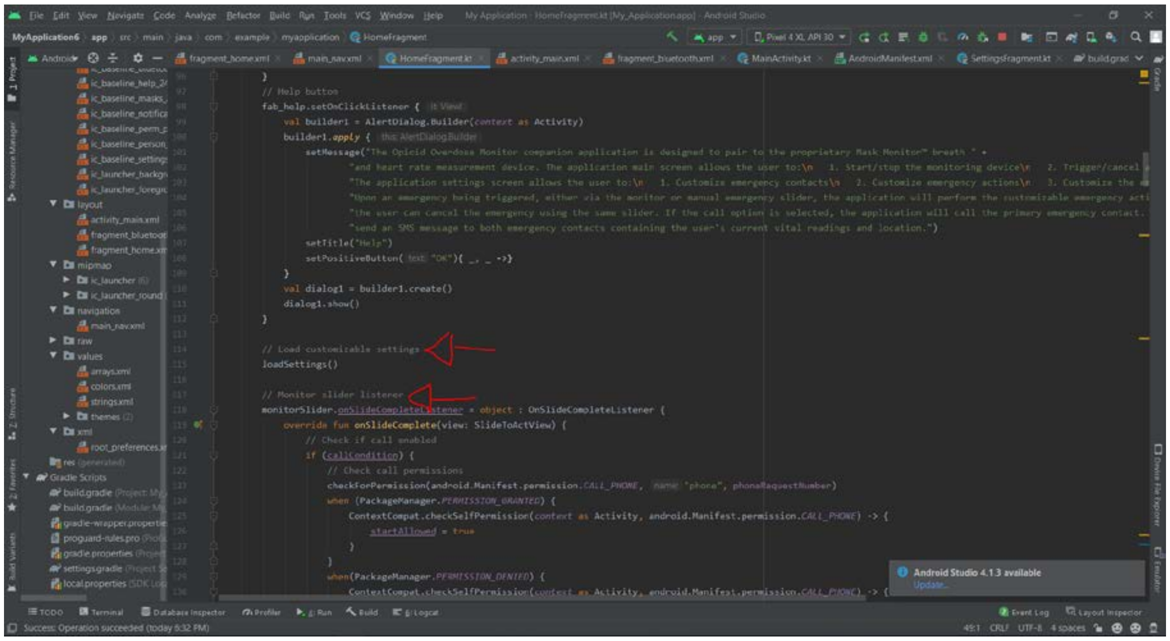
The main files of the android application are .xml and .kt files. These files form the main structure of the application and are where the developer spends most of their time. The .xml files, which are found in the layout folder, define the layout and visible elements. It is important to note that the application uses one activity with several fragments built in to switch between screens. There are also two other extremely important files labelled AndroidManifests.xml and build.gradle (project). The manifests file mainly allows the developer to include permissions, whereas the gradle file allows for the inclusion of plugins, such as the slider buttons. However, the GitHub slider buttons were coded in Java and had to be adapted for Kotlin to fit the opioid overdose monitor application.



```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".HomeFragment">

    <ImageView
        android:id="@+id/imageView"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginTop="10dp"
        android:layout_centerHorizontal="true"
        android:src="@drawable/ic_baseline_masks_24" />

    <LinearLayout
        android:id="@+id/infoTexts"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_centerInParent="true"
        android:orientation="vertical">

        <TextView
            android:id="@+id/textView"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:gravity="center"
            android:text="Blood Oxygen Level:" />

        <TextView
            android:id="@+id/textView2"
            android:layout_width="match_parent"
```

Figure 32: Main screen XML file

Within the application code, many comments are placed to indicate what specific sections of the code do.



```kotlin
}
// Help button
fab_help.setOnClickListener {
    val builder1 = AlertDialog.Builder(context as Activity)
    builder1.apply {
        setMessage("The Opioid Overdose Monitor companion application is designed to pair to the proprietary Mask Monitor™ breath " +
                "and heart rate measurement device. The application main screen allows the user to:\n   1. Start/stop the monitoring device\n   2. Trigger/cancel
                "The application settings screen allows the user to:\n   1. Customize emergency contacts\n   2. Customize emergency actions\n   3. Customize the
                "Upon an emergency being triggered, either via the monitor or manual emergency slider, the application will perform the customizable emergency acti
                "the user can cancel the emergency using the same slider. If the call option is selected, the application will call the primary emergency contact.
                "send an SMS message to both emergency contacts containing the user's current vital readings and location.")
        setTitle("Help")
        setPositiveButton("OK"){ _, _ ->}
    }
    val dialog1 = builder1.create()
    dialog1.show()
}

// Load customizable settings
loadSettings()

// Monitor slider listener
monitorSlider.onSlideCompleteListener = object : OnSlideCompleteListener {
    override fun onSlideComplete(view: SlideToActView) {
        // Check if call enabled
        if (callCondition) {
            // Check call permissions
            checkForPermission(android.Manifest.permission.CALL_PHONE, "phone", phoneRequestNumber)
            when (PackageManager.PERMISSION_GRANTED) {
                ContextCompat.checkSelfPermission(context as Activity, android.Manifest.permission.CALL_PHONE) -> {
                    startAllowed = true
                }
            }
            when(PackageManager.PERMISSION_DENIED) {
                ContextCompat.checkSelfPermission(context as Activity, android.Manifest.permission.CALL_PHONE) -> {
```

Figure 33: Comments in Kotlin file

6.4 Testing & Validation

6.4.1 Circuit Testing

To test the validity of the circuit, a simulation was carried out on Tinkercad with the available substitutions. A circuit prototype as shown in the figure was developed on Tinkercad and it represents the design of a circuit integrated with a pulse oximeter and a respiratory rate measurement sensor which is achieved by using a humidity sensor from the Arduino Nano 33 BLE Sense.

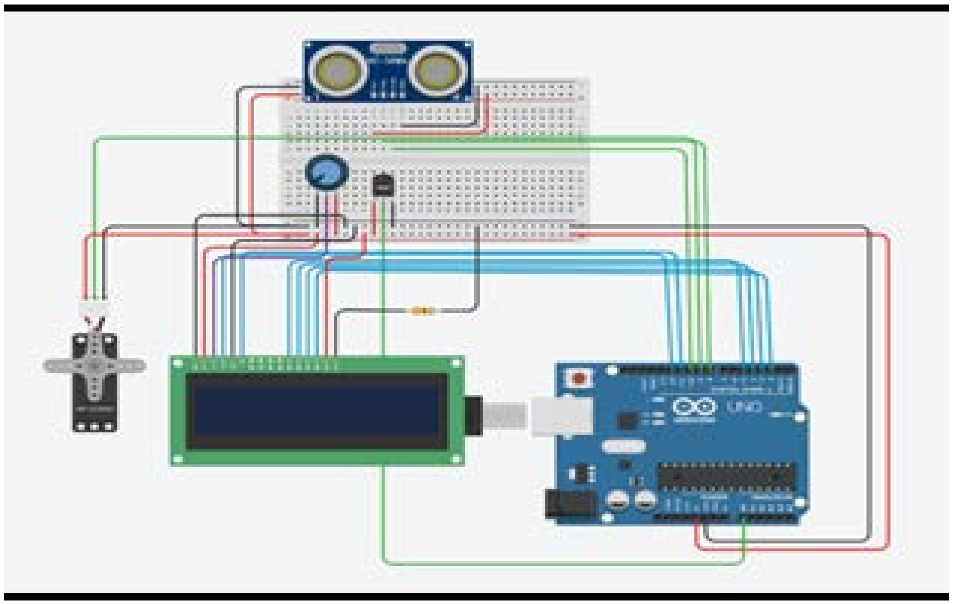
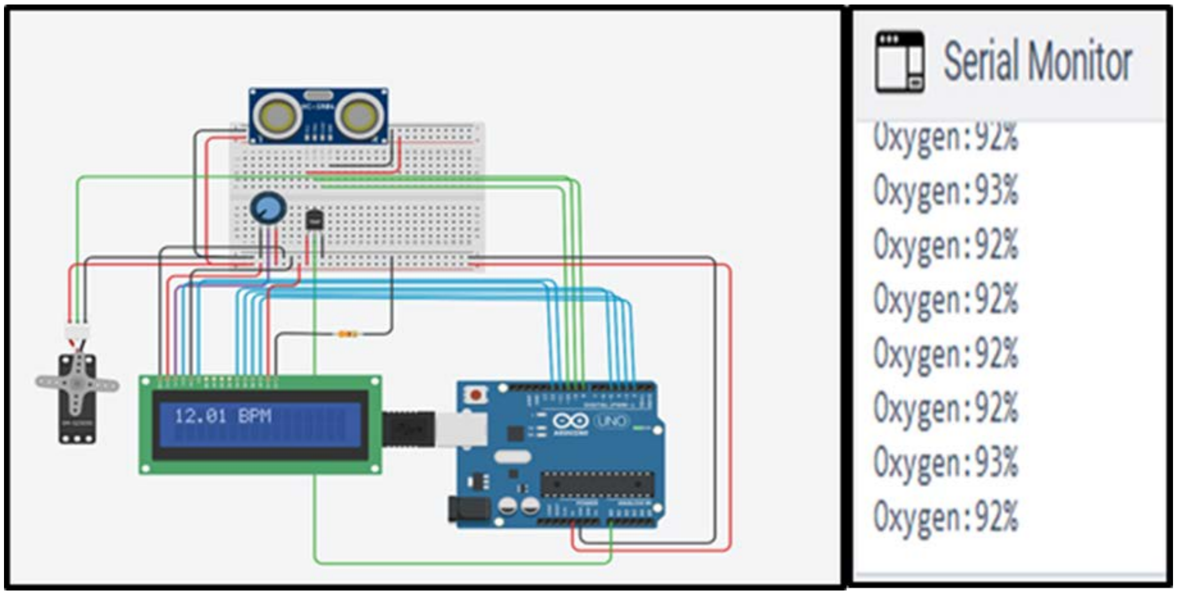


Figure 34 : Circuit diagram of initial prototype

In this prototype, the module is programmed to work in a way when there is an anomaly in the value of either the oxygen level or the breaths per minute, an alert is raised in the app which triggers the user interface via Bluetooth module present on the Arduino Nano 33 BLE Sense. Various images depicting the variation of output for various data combination is shown in the analysis section for better understanding of the test results.

6.4.2 Prototype Analysis

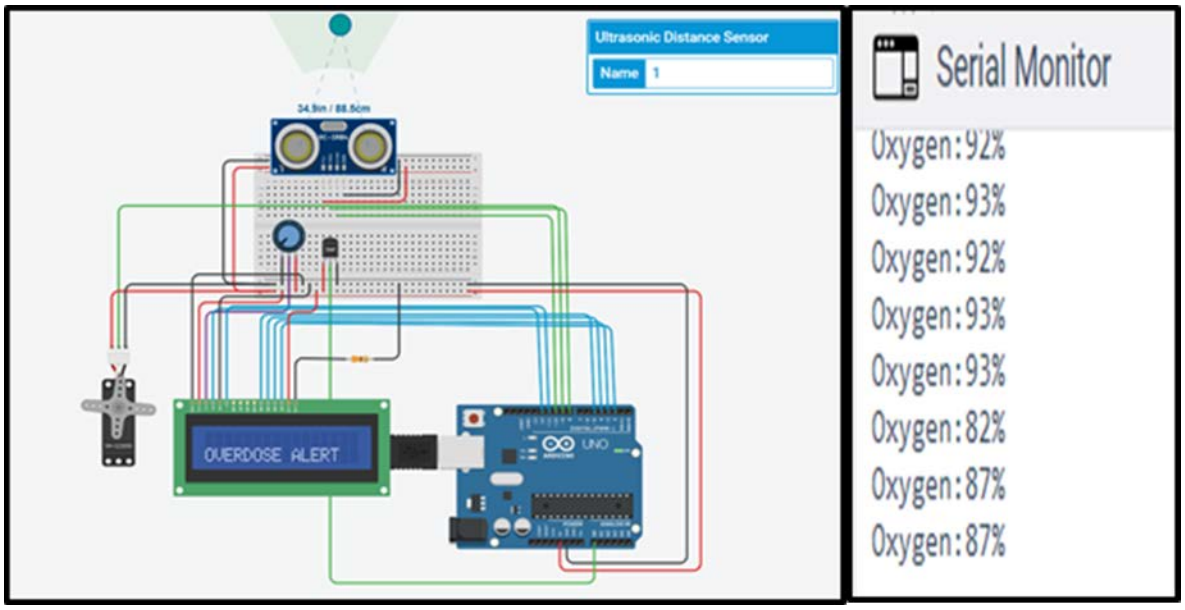
As it can be seen in Figure 35, constant monitoring of respiratory rate and oxygen level is being done.



```
Oxygen:92%
Oxygen:93%
Oxygen:92%
Oxygen:92%
Oxygen:92%
Oxygen:92%
Oxygen:93%
Oxygen:92%
```

Figure 35: Normal Condition BPM and SpO2

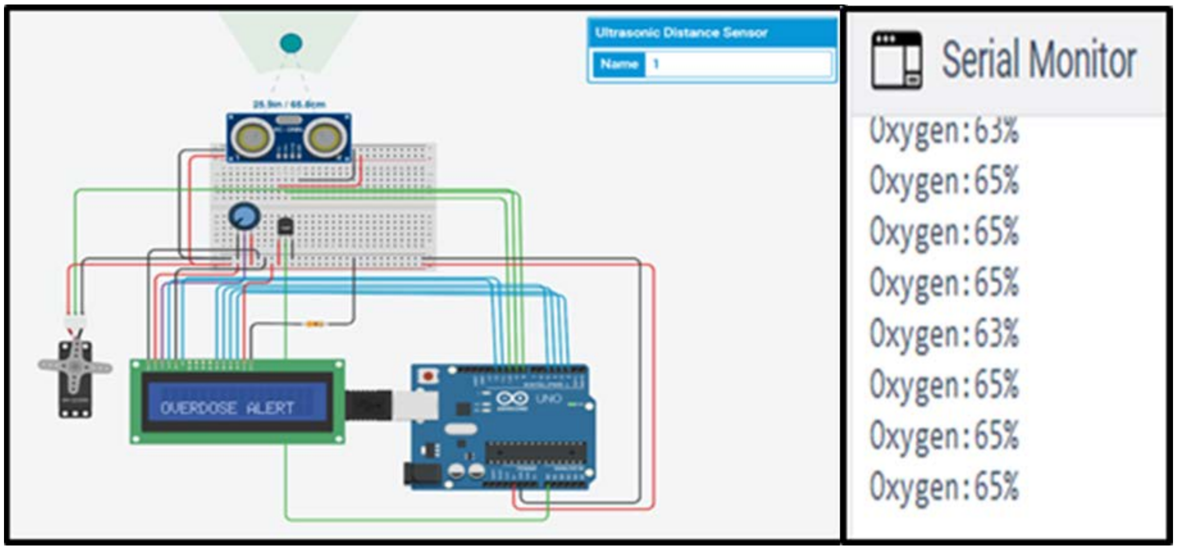
The test case in Figure 36 is depicted for a marginal drop in the oxygen level which is stated as a danger signal as stated by the client. Here the BPM is kept at 14 but even if one criterion is fulfilled, the app is activated.



```
Oxygen:92%
Oxygen:93%
Oxygen:92%
Oxygen:93%
Oxygen:93%
Oxygen:82%
Oxygen:87%
Oxygen:87%
```

Figure 36: OD due to Marginally Low SpO2

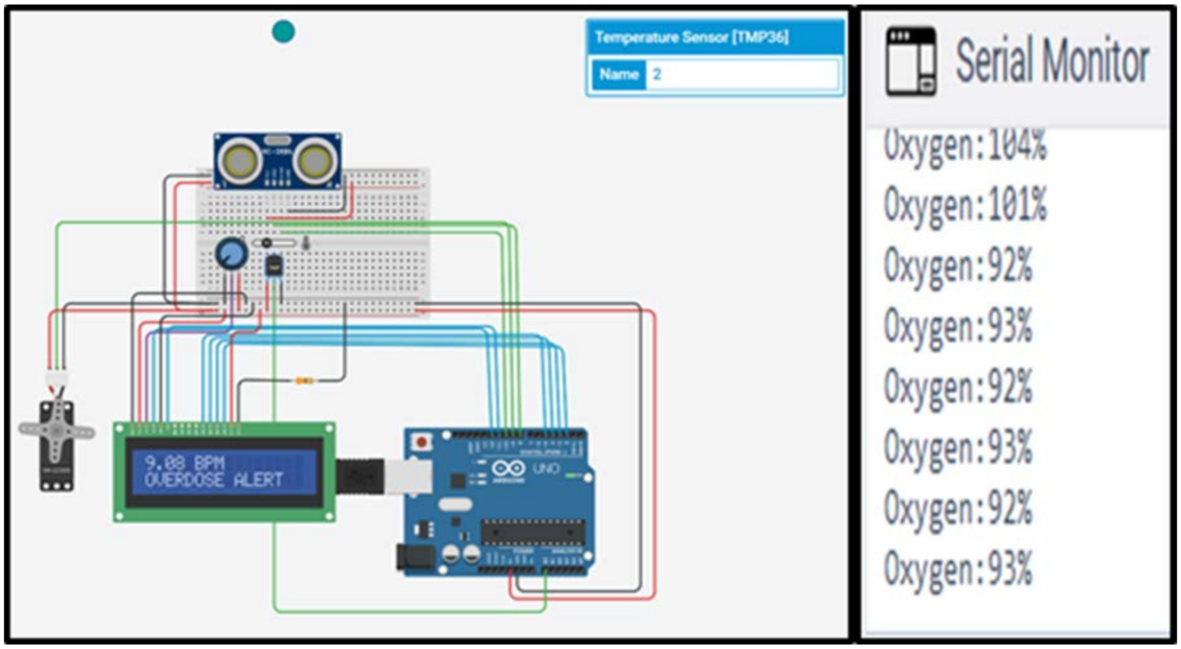
The test case in Figure 37 is an extreme scenario, to check the performance of circuit in boundary conditions.



```
Oxygen:63%
Oxygen:65%
Oxygen:65%
Oxygen:65%
Oxygen:63%
Oxygen:65%
Oxygen:65%
Oxygen:65%
```

Figure 37: OD Alert when SpO2 Significantly Below 90%

The test case in Figure 38 is depicted for a drop in the respiratory rate which is stated as a danger signal as stated by the client. Here the oxygen is kept at 93 but even if one criterion is fulfilled, the app is activated.


Temperature Sensor [TMP36]
Name 2

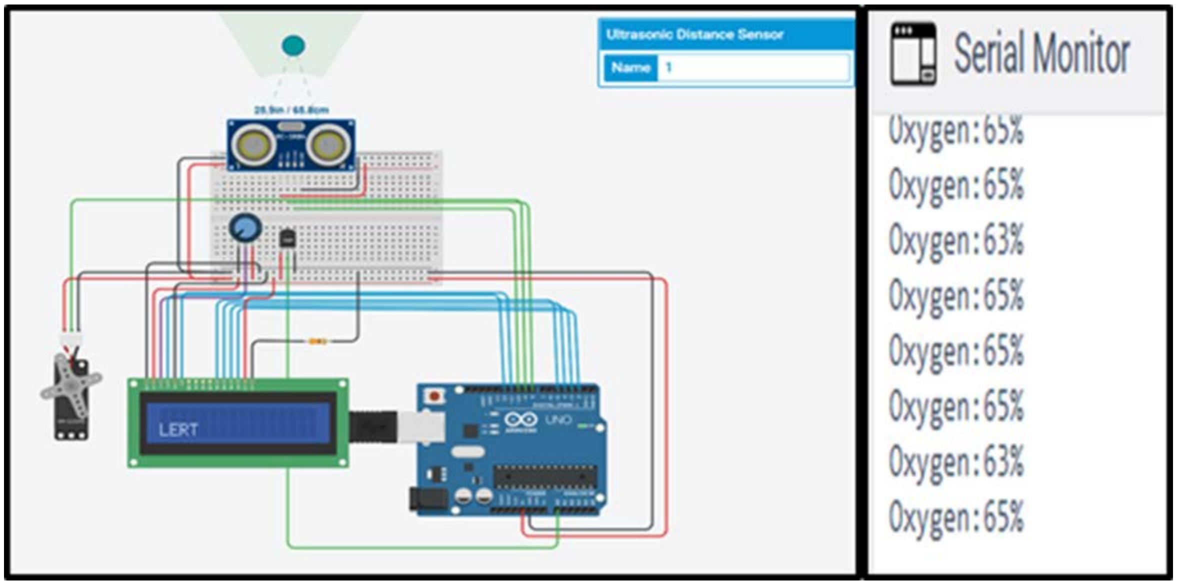
```
9.08 BPM
OVERDOSE ALERT
```

Serial Monitor

```
Oxygen:104%
Oxygen:101%
Oxygen:92%
Oxygen:93%
Oxygen:92%
Oxygen:93%
Oxygen:92%
Oxygen:93%
```

Figure 38: OD Alert due to Low BPM

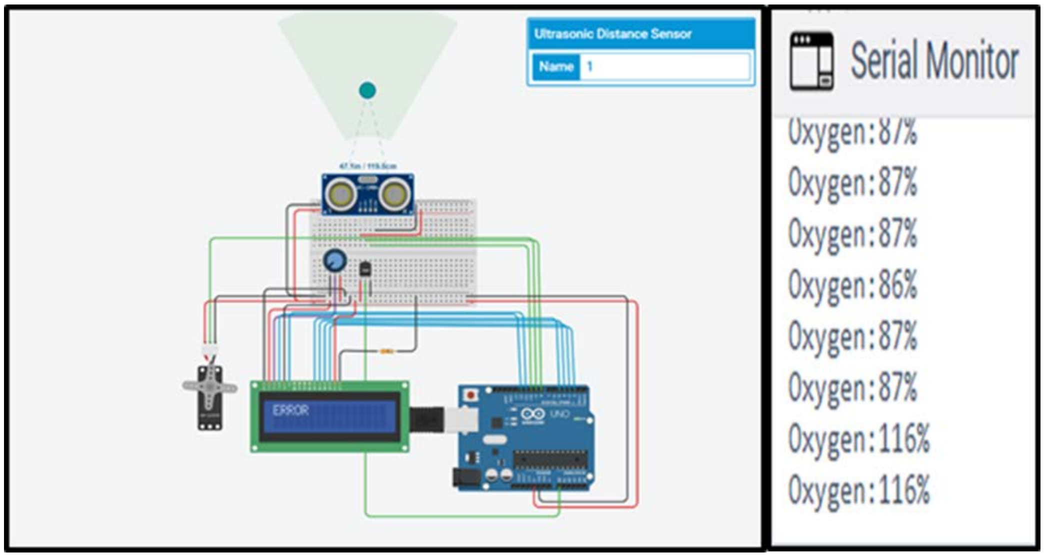
It can be seen in Figure 39 below that the servo motor is rotating as an indication of the app being activated due to overdose alert.



```
Oxygen:65%
Oxygen:65%
Oxygen:63%
Oxygen:65%
Oxygen:65%
Oxygen:65%
Oxygen:63%
Oxygen:65%
```

Figure 39: Servo Rotating due to Alert

If the sensor is not reading a predicted value, an error message is shown in the LCD as shown in Figure 40.



```
Oxygen:87%
Oxygen:87%
Oxygen:87%
Oxygen:86%
Oxygen:87%
Oxygen:87%
Oxygen:116%
Oxygen:116%
```

Figure 40: Error Border Values

6.4.3 Ergonomic Testing

To ensure that our product is comfortable to wear for long periods of time, we conducted two ergonomic tests: one using cardboard models which were superglued to a facemask shown in Figure 41 and another using the final prototype shown in Figure 1.

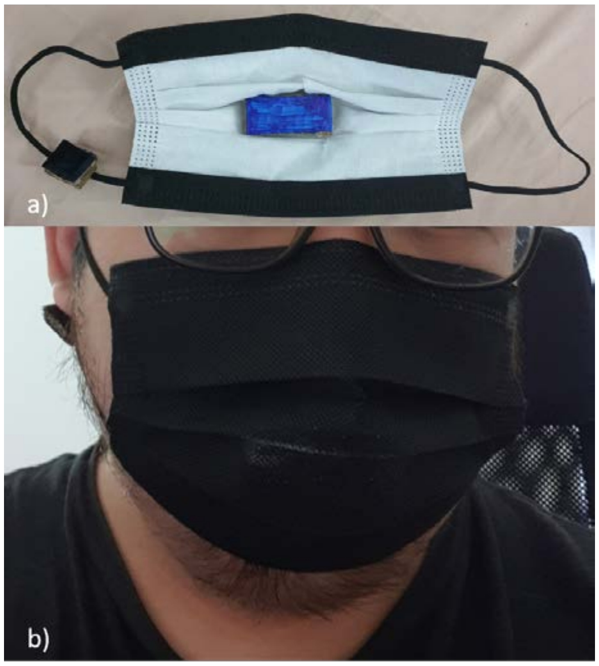


Figure 41. Initial Prototype with Cardboard showing a) the insides, b) a user wearing

We asked three participants to wear the product and asked them five questions regarding the comfortability that they gave a rated answer between 1 to 5 where 1 means strongly disagree and 5 means strongly agree. Table 5 shows the ergonomic survey results for both tests [7, 9].

Table 4. Ergonomic Survey Results [7, 9]

	Rated Answers (Scale of 1 to 5; 1 = Strongly disagree, 2 = Disagree, 3 = Neutral, 4 =Agree, 5 = Strongly agree)					
Questions	**Participant 1**		**Participant 2**		**Participant 3**	
	Initial	Final	Initial	Final	Initial	Final
Is the device comfortable to wear for long period of time?	4	3	3	3	3	3
Is the device small and discreet?	4	4	4	4	4	4
Is the device hands-free?	5	5	5	5	5	5
Does the pulse oximeter provide good contact with the earlobe?	4	3	1	3	4	3
Do you think you would use this device if you were a substance user?	3	3	3	3	3	3

6.4.4 SpO2 Measurement

We asked the same three participants that undertook the ergonomics testing to test our SpO2 measurements. The pulse oximeter was placed on their fingers for sixty seconds. The results are shown in Table 6.

Table 5. SpO2 Measurement Results

	PARTICIPANT 1	**PARTICIPANT 2**	**PARTICIPANT 3**
TIME (S)	SpO2 (%)	SpO2 (%)	SpO2 (%)
0	-999	-999	-999
2	-999	-999	-999

4	-999	-999	-999
6	-999	61	-999
8	62	61	64
10	62	70	64
12	67	70	72
14	67	78	72
16	97	96	98
18	97	96	98
20	69	67	98
22	44	45	61
24	44	45	41
26	47	43	41
28	56	43	41
30	57	58	55
32	59	61	55
34	59	93	61
36	59	93	78
38	94	70	93
40	77	65	93
42	75	65	74
44	75	41	63
46	45	41	63
48	45	68	48
50	72	73	48
52	72	73	65
54	95	94	65
56	95	94	96
58	68	94	96
60	68	69	96

6.4.5 RR Measurement

We once again asked the three participants that undertook the ergonomics and SpO2 testing to test our RR measurements. They wore our product and breathed into it under their usual breathing pattern for sixty seconds. To verify the results from our RR measurements is accurate, we also counted their number of breaths during testing and compared the results. The results from the RR measurement testing is shown in Table 7.

Table 6. RR Test Results

	RR (BrPM)		
Time (s)	**Participant 1**	**Participant 2**	**Participant 3**
0	15	12	13
4	14	15	17
8	17	13	16
12	13	14	14
16	16	14	15
20	18	14	15
24	12	14	15
28	15	14	15
32	13	13	15
36	13	17	13
40	13	15	16
44	14	14	15
48	16	14	15
52	12	14	14
56	17	13	15
60	14	16	14
Counted RR	16	14	15

6.4.6 UI

To assess the performance of the UI, three participants were asked to observe the initial UI prototype and interact with it. The participants were then surveyed, and a short focus group session was held to discuss issues with the design. The survey results were as follows:

Table 7: UI Survey Results

	Rated Answers (Scale of 1 to 5; 1 = Strongly disagree, 2 = Disagree, 3 = Neutral, 4 =Agree, 5 = Strongly agree)		
Questions	Participant 1	Participant 2	Participant 3
The UI is visually appealing	5	4	5
The UI is easy to navigate	5	5	5
The emergency alarm is effective in warning the user	2	3	1
The settings offer all the customization I need	4	5	5
I find it difficult to accidentally turn off the monitor or trigger an alarm	1	2	1

In the focus group session, the users indicated that the emergency alarm was not noticeable enough and should be a pop up rather than a small notification at the bottom. This was noted and additional visual cues will be added to the alarm. They also stated that it was too easy to trigger a false alarm or cancel a real one. This will be rectified with the addition of the slider buttons.

To assess the performance of the updated UI, the same three participants were asked to again observe the UI and interact with it. The participants were then surveyed using the same survey to compare the results and a short focus group session was held to discuss issues with the design. The survey results are shown in Table 7.

Table 8. UI Survey Results

	Rated Answers (Scale of 1 to 5; 1 = Strongly disagree, 2 = Disagree, 3 = Neutral, 4 =Agree, 5 = Strongly agree)		
Questions	Participant 1	Participant 2	Participant 3
The UI is visually appealing	5	5	5
The UI is easy to navigate	5	5	5
The emergency alarm is effective in warning the user	5	4	5
The settings offer all the customization I need	5	5	4
I find it difficult to accidentally turn off the monitor or trigger an alarm	2	1	2

In the focus group session, the users were impressed with the progress made in UI design. They noted that the added alarm was effective. They also recognized the slider buttons and said that the sliders were intuitive. Furthermore, users found the call and location SMS features to be reliable and effective at notifying emergency contacts.

To further determine the quality of the UI, another survey was done with a new focus group, consisting of participants that have not seen earlier prototypes of the software. This time, the users were asked to evaluate whether the UI met Jakob Nielsen's 10 general principles for interaction design.

Table 9. Nielsen's Principle Survey on UI

Rated Answers (Scale of 1 to 5; 1 = Strongly disagree, 2 = Disagree, 3 = Neutral, 4 =Agree, 5 = Strongly agree)

Questions	Participant A	Participant B	Participant C
Visibility of system status	5	5	5
Match between system and the real world	1	2	1
User control and freedom	4	4	3
Consistency and standards	5	5	5
Error prevention	5	5	5
Recognition rather than recall	5	4	4
Flexibility and efficiency of use	4	4	5
Aesthetic and minimalist design	5	5	5
Help users recognize, diagnose, and recover from errors	4	3	4
Help and documentation	5	4	4

Users found the system status to be clearly available at all times. With both the sliders and center of the screen indicating the condition of the application, the participants had no issue describing the system status while using the app.

Users commented that while the user interface seemed to be able to match the real world effectively, they were not able to effectively test this due to the nature of the pandemic and the lack of Bluetooth implementation.

User control and freedom was deemed sufficient with room for improvement. The focus group stated that background functionality would benefit the application.

Consistency and standards were deemed excellent as the application shows high quality throughout.

Error prevention was deemed perfect by all users as the focus group did not encounter any errors or crashes while using the app. To add to this, regression testing done by the programmer found a glitch with the slider buttons mentioned in section 5.2.3; this issue was rectified before showing the app to the participants.

Recognition rather than recall was not found to be highly applicable to this application as it only consists of three switchable screens. This means that the settings menu is easily at reach and any options set can be found with one tap. Users did however comment that it was inconvenient to stop monitoring to access the settings.

Flexibility and efficiency of use was found to be well addressed within the settings menu. However, users mentioned the benefits of adding a customizable countdown timer.

Aesthetic and minimalist design was highly praised by all participants. The use of Android Studio proved highly beneficial in this regard.

The ability for the application to help users recognize, diagnose, and recover from errors was deemed to be somewhat acceptable with the help menu. Participants suggested that a support number or help chat function could be helpful in aiding users. While an excellent suggestion for future work, the suggestion was deemed out of scope for this project.

Help and documentation was found to be effective in the form on the help pop-up because of the app’s simplicity. However, users indicated that if Bluetooth functionality was implemented, a separate Bluetooth help menu should be developed.

7 Conclusions and Recommendations for Future Work

The device was created to help the target audience mentioned in the report to be safe in case of an opioid overdose. The main aim of the opioid overdose monitor is to prevent deaths caused by substance use by contacting the editable emergency contacts fed into the app by the user while setting up the device.

While designing and implementing the system, the team kept this in mind to allow for the possibility to improve the design later. All improvements are well documented, and all sub-systems are well described by their responsible team members; making the device scalable for future modifications and added features. The device's simplistic design approach also keeps manufacturing and assembly scalability in mind. The device's quality was taken very seriously, and the device achieves much of the client's requirements reliably within the allotted budget. However, technology cannot stagnate and must be continuously improved.

The usability of the device is developed while keeping the state of the user in mind. The UI is very simple and intuitive, and the hardware is small, compact and discreet. The module is very cost effective, making it reachable to a large majority of the targeted market.

Future scope for this project is endless. The team sincerely hopes that this document and the previous documentations help the future teams to work towards this social problem and create a great solution to save precious lives.

If the team had more time to develop the project, it would have included a timer that can be incorporated in the device to count the number of seconds since the overdose has been detected so that the urgency can be understood and acted upon. A live feed of the user could also be transmitted to the emergency contact to effectively know the state of the user. These developments would greatly prevent the overdose related deaths and would help us win the battle.

8 Bibliography

[1] World Health Organization. (2020, August 28). Opioid overdose. Retrieved January 28,2021, from https://www.who.int/news-room/fact-sheets/detail/opioid-overdose

[2] Cahill, T. (2020). Canada's Overdose Crisis [PowerPoint slides]. Ottawa, ON.

[3] Ontario Agency for Health Protection and Promotion (Public Health Ontario), Office of the Chief Coroner, Ontario Forensic Pathology Service, Ontario Drug Policy Research Network. (2019). Opioid mortality surveillance report: analysis of opioid-related deaths in Ontario July2017-June 2018. Queen's Printer for Ontario

[4] EC21. (2008). Reflective PPG Integrated Sensor - APMKorea. Retrieved from https://hansangh2.en.ec21.com/Reflective-PPG-Integrated-Sensor--10605888.html

[5] AlAkhras, A., Kim, M. J., Srinivasan, V., (2021). *GNG5140: Engineering Design Deliverable E: Revised Prototype Analysis & Test Results*.

[6] Seidle, N., (2016). SparkFun_MAX3010x_Sensor_Library. Retrieved from https://github.com/sparkfun/SparkFun_MAX3010x_Sensor_Library

[7] AlAkhras, A., Kim, M. J., Srinivasan, V., (2021). *GNG5140: Engineering Design Deliverable G: Design Day Pitch & Final Prototype*.

[8] Corti, N., 2020. *slidetoact* v0.9.0, GitHub. Retrieved from https://github.com/cortinico/slidetoact

[9] AlAkhras, A., Kim, M. J., Srinivasan, V., (2021). *GNG5140: Engineering Design Deliverable D: Initial Prototype & Test Results*.

APPENDICES

9 APPENDIX I: Design Files

Table 10. Referenced Documents

Document Name	Document Location and/or URL	Issuance Date
Team Contract	https://teams.microsoft.com/l/file/7194b988-4c39-45cd-b47d-1c4ccc0d8e3c?tenantId=d41fdab1-7e15-4cfd-b5fa-7200e54deb6b&fileType=docx&objectUrl=https%3A%2F%2Fuottawa.sharepoint.com%2Fsites%2FGNG5140W00-Opioid_Overdose1%2FShared%2520Documents%2FOpioid_Overdose%25201%2FTeam_Contract_GNG5410.docx&baseUrl=https%3A%2F%2Fuottawa.sharepoint.com%2Fsites%2FGNG5140W00-Opioid_Overdose1&serviceName=recent	14/Jan/21
Deliverable B	https://teams.microsoft.com/l/file/25591915-503b-4107-a44f-66090a5de2ee?tenantId=d41fdab1-7e15-4cfd-b5fa-7200e54deb6b&fileType=docx&objectUrl=https%3A%2F%2Fuottawa.sharepoint.com%2Fsites%2FGNG5140W00-Opioid_Overdose1%2FShared%2520Documents%2FOpioid_Overdose%25201%2FDeliverable%2520B%2520-%2520Design%2520Research.docx&baseUrl=https%3A%2F%2Fuottawa.sharepoint.com%2Fsites%2FGNG5140W00-	31/Jan/2021

	Opioid_Overdose1&serviceName=recent	
Deliverable C	https://teams.microsoft.com/l/file/CACE5EB6-DE03-47DD-8306-7AE09C304894?tenantId=d41fdab1-7e15-4cfd-b5fa-7200e54deb6b&fileType=docx&objectUrl=https%3A%2F%2Fuottawa.sharepoint.com%2Fsites%2FGNG5140W00-Opioid_Overdose1%2FShared%20Documents%2FOpioid_Overdose%201%2FDeliverable%20C%20-%20Design%20Requirements%20and%20Project%20Plan.docx&baseUrl=https%3A%2F%2Fuottawa.sharepoint.com%2Fsites%2FGNG5140W00-Opioid_Overdose1&serviceName=teams&threadId=19:1726c71658ef42dfa61a50d9e2bdd576@thread.tacv2&groupId=f71c8592-94bf-49e7-877e-77aac667522a	14/Feb/2021
Deliverable D	https://teams.microsoft.com/l/file/17F51756-AF2C-4956-84B8-AC1D7F0C7DAB?tenantId=d41fdab1-7e15-4cfd-b5fa-7200e54deb6b&fileType=docx&objectUrl=https%3A%2F%2Fuottawa.sharepoint.com%2Fsites%2FGNG5140W00-Opioid_Overdose1%2FShared%20Documents%2FOpioid_Overdose%201%2FDeliverable%20D%20-%20Initial%20Prototype%20Analysis%20%26%20Test%20Results%20(1).docx&baseUrl=https%3A%2F%2Fuottawa.sharep	7/March/2021

	oint.com%2Fsites%2FGNG5140W00-Opioid_Overdose1&serviceName=teams&threadId=19:1726c71658ef42dfa61a50d9e2bdd576@thread.tacv2&groupId=f71c8592-94bf-49e7-877e-77aac667522a	
Deliverable E	https://teams.microsoft.com/l/file/C0E195E0-0A4E-4ED1-A720-D3372E51B376?tenantId=d41fdab1-7e15-4cfd-b5fa-7200e54deb6b&fileType=docx&objectUrl=https%3A%2F%2Fuottawa.sharepoint.com%2Fsites%2FGNG5140W00-Opioid_Overdose1%2FShared%20Documents%2FOpioid_Overdose%201%2FDeliverable%20E%20-%20Revised%20Prototype%20Analysis%20%26%20Test%20Results.docx&baseUrl=https%3A%2F%2Fuottawa.sharepoint.com%2Fsites%2FGNG5140W00-Opioid_Overdose1&serviceName=teams&threadId=19:1726c71658ef42dfa61a50d9e2bdd576@thread.tacv2&groupId=f71c8592-94bf-49e7-877e-77aac667522a	21/March/2021
Deliverable G	https://teams.microsoft.com/l/file/DA8BD121-3726-482F-BCBC-B0DF3AE18CE8?tenantId=d41fdab1-7e15-4cfd-b5fa-7200e54deb6b&fileType=docx&objectUrl=https%3A%2F%2Fuottawa.sharepoint.com%2Fsites%2FGNG5140W00-Opioid_Overdose1%2FShared%20Documents%2FOpioid_Ove	8/Apr/2021

	rdose%201%2FDeliverable%20G%20-%20Design%20Day%20Pitch%20%26%20Final%20Prototype.docx&baseUrl=https%3A%2F%2Fuottawa.sharepoint.com%2Fsites%2FGNG5140W00-Opioid_Overdose1&serviceName=teams&threadId=19:1726c71658ef42dfa61a50d9e2bdd576@thread.tacv2&groupId=f71c8592-94bf-49e7-877e-77aac667522a	
Arduino Code	https://uottawa.sharepoint.com/sites/GNG5140W00-Opioid_Overdose1/Shared%20Documents/Opioid_Overdose%201/opioidOverdoseMonitor.ino	
Application APK file	https://uottawa.sharepoint.com/sites/GNG5140W00-Opioid_Overdose1/Shared%20Documents/Opioid_Overdose%201/Opioid%20Overdose%20Monitor.apk	
Android Studio Files	https://uottawa.sharepoint.com/sites/GNG5140W00-Opioid_Overdose1/Shared%20Documents/Opioid_Overdose%201/UI%20Files%20Deliverable%20G.rar	