GNG 1103

Team 5

Final Report & User Manual

OPIOID OVERDOSE DETECTOR

By

GNG1103, SECTION C00, GROUP 5

Yi Ru Loh, 300134586
Mohamad Ahmad, 300108630
Dhvani Patel, 300121365
Sheetal Harakh, 300123274

Table of Contents

Abstract

This document demonstrates and visualizes all of the work completed by Group 5 of the GNG1103 course project; Opioid Overdose Detector. The report begins with a list of figures as well as tables and acronyms. This allows for the reader to have a better understanding of the technical terms used in the document, as well as easily navigate to the important images and tables. Next the introduction describes the process the group went through while developing their product. First, a general idea was determined and a client meeting was held. There, the students learned more about the conditions the device was required to meet and gained insight on methods they could follow in order to achieve their set goals. Early concepts were then made in order to begin proper planning and brainstorming. Next, scheduling and cost was determined to keep them on track throughout the semester. Multiple prototypes were made before the final product was presented to the class and a status update. This part of the document was included due to the COVID-19 pandemic that arose during the end of the semester. Here, the final status of the prototype and exact steps the students would have taken to finish the project is presented, concluding the document. The end of the document contains all instructions and resources required for you, the reader, to build the same device.

List of Figures

List of Tables

List of Acronyms

Acronym	Definition
SpO_2	Oxygen Saturation Percentage in Blood
LCD	Liquid Crystal Display
GPS	Global Positioning System
CAD($)	Canadian Dollar
GUI	Graphical User Interface

Introduction

All teams in GNG 1103, the Engineering Design section C class were given the task of designing and building an opioid overdose detecting device. This was done using the five step design process taught throughout the semester, which led us to a near perfect working prototype that was presented to the class in our final presentation.

Near the beginning of the semester we were given a general idea of what we needed to design, however, as the semester progressed, more information was introduced. A client meeting was held with members of the Sandy Hill Community Center, where we were introduced to the basic idea of how the device should work and the features it should include. We then started to work on ideas with the legitastics in mind, as specified by the client.

During our weekly meetings and on our own time, ideas were drawn by everyone and explained to each other in the group. After all the ideas were carefully considered, we decided to use key concepts from the top 3 designs and integrate them into our final design that satisfied the clients needs. We used Gantt charts to stay on track, split responsibility and plan out an estimated timeline of each job.

There were a total of 3 prototypes built, the first one was very simply done and the purpose of it was to give a general idea of what the device would do and how it would look. For the second prototype we focused mainly on the hardware and putting it together along with the computer code to ensure that it would work within the device. As for the third prototype, we mainly put the first two prototypes together and made sure that the hardware fit into the final design and worked when used by one of us to test the result. However, we did not fully complete prototype III because of an unexpected situation that was not foreseen.

Need Identification and Product Specification Process

Problem Statement

A need exists for people who use opioids to safely have their blood oxygen saturation monitored and location quickly sent to emergency services in case of overdose, with a device that is discrete and cost-effective.

Needs Identification

Category	Customer Need Statement	Interpretation	Priority
Functionality	I want the device to measure blood oxygen saturation level.	The device can measure blood oxygen saturation levels	**3**
	Device can send location(via GPS) to emergency services	Can send location (via GPS) to emergency services	**1**
	The user will be able to call for help in a short time.	Can call for help in under 3 minutes	**1**
	A single battery charge lasts up to one full day.	A single battery charge lasts up to one full day	**6**
Affordability	I need the device to be affordable/cheap.	Device is low cost (<$100)	**4**
User-friendly	I want the device to be hands-free/ handy to carry.	The device is hands-free	**5**
	Durable and can be worn for long periods of time	Device is durable and can be worn for long periods of time (comfortable)	**6**
	Device is discrete and does not disturb one's daily life	The device is discrete and does not disturb one's daily life	**7**
Safety	Device is accurate	The device is accurate	**2**

Table 1: Needs Identification

Justification and Reasoning:
Our list of needs had been determined based on information shared during our first client meeting. The data was then written in terms of what the product has to do and expressed specifically as the raw data implied. The needs were then expressed as an attribute of the product and phrased positively.

The needs were then categorized based on aspects determined by the raw data acquired from the initial client meeting. Those categories were functionality, affordability, ability to be user-friendly, and safety. After deciding how each need fit into our determined categories, they were then numbered based on priority levels.

The priority levels had also been based on the raw data acquired from the first meeting, clearly making the major functions and requirements of the device the main priority. Our first priority had been decided to be the device's ability to send the user's location to an emergency service and its ability to do so in under 3 minutes. Next we had decided that the device being accurate was another high priority for safety and practical reasons. Following this, the device's ability to measure blood oxygen saturation levels had been another high priority due to it being the determining factor for identifying an overdose. In addition, the device's cost was another priority since it being affordable was a major condition that the client specified. Moreover, the device being hands-free, durable, worn for long periods and discrete were listed as lower priority levels since they were not as crucial.

Benchmarking

Criteria Category	Mibest OLED Finger Pulse Oximeter	Digital LCD Automatic Oximeter Health Monitor	AK1980 Fitness Tracker
Picture of Product	OLED DISPLAY Large Bright Digital Numbers Accurate Pulse Rate & Oxygen Saturation Readings Adjustable Brightness Battery Indicator 4 Directional Display mibest		
Physical Product	This is a pre-existing device that uses a SpO$_2$ sensor (saturation percentage of	This is a wristband look alike device that comes with the function of monitoring blood	This device is similar to a fitbit, it tracks the heart rate, along with distance walked,

	oxygen in the blood) to check and display real-time levels as well as pulse strength. The design is simple and fairly compact. While it does require to be worn on the finger, the device provides accurate measurements and is quite compact.	pressure, pulse, and heartbeat rate. This design can also act as a watch as it has time displayed on the screen.	sleep cycle, and much more. The tracked numbers are then displayed on the screen for the user to refer back to at any time. It is also very easy to use and fairly handy for any person. In order to keep track, the device must simply be worn on the user's wrist.
UI	The device comes with batteries, a neck lanyard, and a storing pouch. The user interface itself is easy to follow and requires little effort to use since it uses a One Button Operation; a single button is used to start, stop, control brightness and other settings.	The device comes with an LCD display to display blood pressure, pulse, date, time, time and low battery icon, and memory record number. The device is user friendly as it has a large backlight LCD for easy reading and it is easy to operate.	The device is equipped with an LED screen to display the visuals for the user. A charging cord that charges the device for daily use. The touch screen lets the user also change certain settings to preference, such as brightness and sounds.
Safety	This is a portable and user friendly device that can easily accommodate all ages. It also requires minimal setup effort and previous knowledge regarding the device.	This is a portable design that allows users to monitor blood pressure easily at any time. The device is also able to memorize the measuring data and time of the most recent 60 times.	This easy to use and small device is very user friendly and safe for all ages. Very few steps are required to set up and get the device working.
Perception	The luxury design of dual color oximeter displays reading in 4 directions, 4 display modes and 10 levels of adjustable brightness. Besides that, it also comes with a battery-low indicator.	This device is good for sports and outdoor activities as it can be worn on the wrist at all times and does not hinder the movement of the user. The information that is measured by the device provides the user the vital measurements that the user needs to know to keep track of their health.	The easy to assemble wrist watch is also waterproof, making it very durable. It also has a long battery life and is made of flexible rubber material for all wrist sizes. Along with warnings/ notifications for unusual activities or abnormalities in the users' daily routines or body system. Such as an abnormally high heart rate.
Cost	$24- $30	$65-$70	$35-$40

Table 2: Benchmarking

Metrics

Metric	Importance	Units
Cost	3	CAD ($)
Dimensions	4	Millimeters(mm)
SpO_2 level	1	Percentage (%)
Heart Rate	4	Beats per minute (BPM)
Time	2	minutes (mins)

Table 3: Metrics

Target Specifications

	Design Specifications	Relation (=,< or >)	Value	Units	Verification Method
	Functional requirements				
1	Real time monitoring of blood oxygen saturation levels.	=	yes	mins	Test
2	Accurate location tracking services able to send location to ER	=	yes	N/A	Test
3	More than 24 hours of battery life per charge	>	24	hrs	Test
	Non-functional requirements				
4	Discrete and small	=	yes	cm^3	Test
5	Comfortable for user	=	yes	N/A	Test
6	Not obtrusive in user's daily life (i.e device can't go on fingers)	=	yes	N/A	Test
7	Affordable	<	100	CAD$	Analysis
8	Eye catching/ aesthetically pleasing	=	yes	N/A	Test
	Constraints				
9	Cost	<	100	CAD$	Estimate, Final check
10	Time (deadlines)	<	12	hrs	Estimate
11	Portability	=	yes	N/A	Test
12	Hardware: capabilities of devices being used (Arduino/Raspberry Pi)	=	yes	N/A	Analysis, Test
13	Software: capabilities of each programming language (Python, C, etc.)	=	yes	N/A	Analysis, Test

Table 4: Target Specifications

Conceptual Designs

Each team member was to come up with 3 possible designs, the ideas were then discussed as a group and evaluated according to their functionality, affordability, safety, and how user friendly the device would be. After the evaluation, 3 of the most functional solutions were decided to be used upon and discussed to come to a final conclusion of one single design. Along with deciding potential designs, we had to keep in mind that they would also have to be realistic and could be designed by us with the restrictions given to us. After considering all these factors, the 3 design solutions below were carefully chosen as a group.

Determining Which Aspects to Include in Our Final Design

Solution 1: (Sheetal's Concept #2)

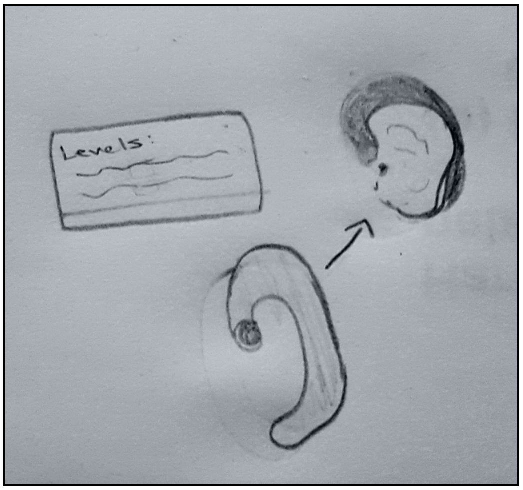


Figure 1 :Sheetal's Concept #2

Sheetal's second concept used the ear as the site of oximeter data collection (like a few other concepts). However, this concept used a frame which would wrap around the ear rather than a clip-on piece. The frame would thus likely be far more **comfortable** for the user than the clip-on. In addition, instead of using a watch or other wrapped device to transmit oximeter data to, this concept used a rectangular digital display. This display device would be cheaper to build than buying a watch. Furthermore, the Arduino experience we have gained from the lab sessions makes this device seem **very feasible to build**.

Features integrated from brainstorming sessions:

- The distance of the two devices should be near each other to ensure the effectiveness of the Bluetooth signal transmission (Ru). Users could get a visual or auditory alert as a reminder to keep components on their person.

Possible downsides:

- Bulky box for the user to carry around.
- System components are dependent on each other (must have both components for the system to work).
- Larger battery is likely required to power the screen.
- Have to recharge 2 devices rather than one.

Things to think about:

- Waterproof?
- What if the oximeter falls off the ear lobe?
- What if the user rolls over? Will components break?
- How do we protect the internals? What protective material should we use?
- What size constraints should we set?
- What if the user doesn't have the data reception device on their person? Will a signal from the oximeter still be received by the display device?
- How far should the bluetooth range be?

Solution 2: (Ru's Concept #3 & Sheetal's Concept #1)

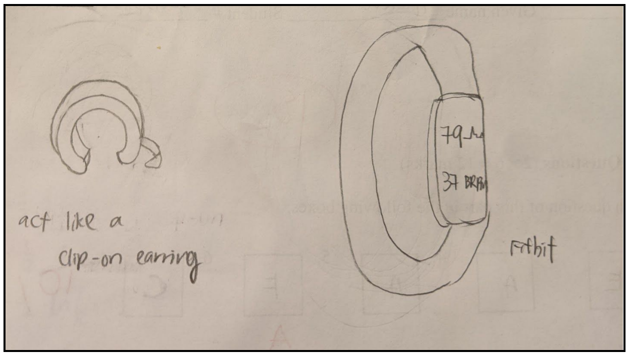


Figure 2 :Ru's Concept #3

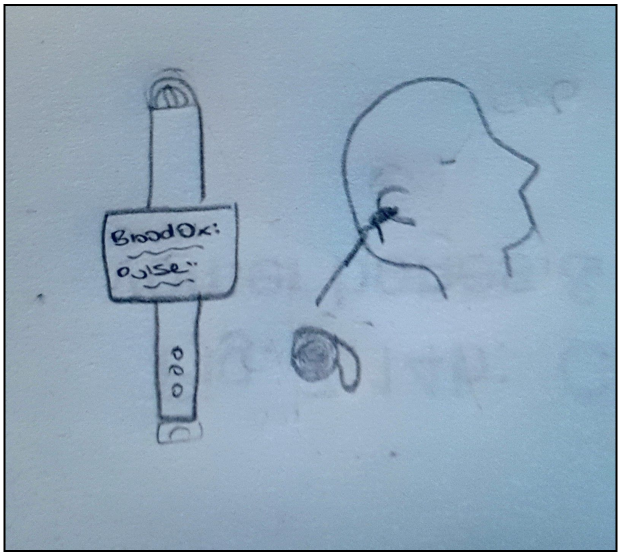


Figure 3 :Sheetal's Concept #1

Both of these concepts are similar as it consists of two parts: an accessory that would be placed at the ear area and also a fit bit band/watch. The ear accessories would act as a pulse oximeter to measure the blood oxygen saturation level as well as heart rate while the readings that are measured will be displayed on the fitbit watch via Bluetooth. The watch will be installed with a battery that powers the device and recharged using solar panels, Bluetooth receptors, and GPS detectors that can send emergency signals to emergency services in case of an emergency. The screen of the watch will display the heart beat rate and blood oxygen saturation level. The watch is durable and water-resistant.

Features integrated from brainstorming sessions:

- The distance of the two devices should be near each other to ensure the effectiveness of the Bluetooth signal transmission. Users will be advised to wear the ear accessory and the watch of the same side, for an example, if the watch is worn on the left wrist, the ear accessory should be worn on the left ear too.

Possible downsides:

- The size of the ear accessory has to be considered as the bluetooth modules may be too big to fit the ear piece which might affect the device's discreteness and user's comfort.
- The possibility of various software knowledge needed to make the devices work, for example, wireless Bluetooth connection from the ear accessory to the watch, GPS technology that can send signals to emergency services in case of emergency, and also the measurement of heart beat rate and blood oxygen saturation level.
- The accuracy and efficiency of the Bluetooth signal transmission at a certain distance.

- The ear accessory might be too heavy for the support of the ear as it will consist of small parts of chips and board inside of the structure to make measurement of blood oxygen saturation level works.

Things to think about:

- How to make it affordable and cost-effective?
- How accurate will the measurements of heart beat rate and blood oxygen saturation level be?
- What are the software needed to use to enable the device to send signals to emergency services in case of emergency?
- What is the approximate standard rate of blood oxygen saturation level that the device should be alerted to know that the user is near overdose?
- What is the technology needed to create a pulse oximeter?

Solution 3: (Sheetal's Concept #3)

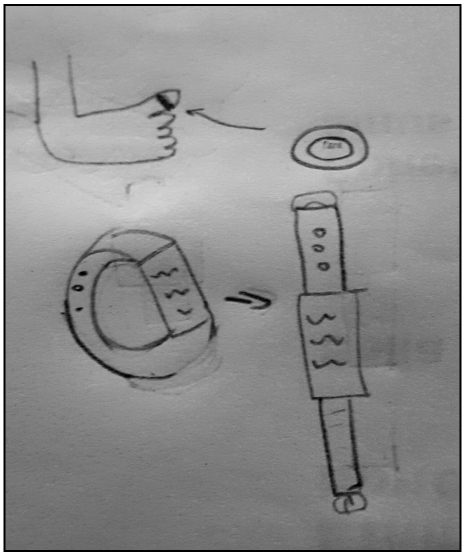


Figure 4 :Sheetal's Concept #3

The third concept selected had been Sheetal's third concept. The concept had been a two-piece device composed of a toe ring and a fitbit style watch. The toe ring acts as a pulse oxygen saturation device with the watch being the display and monitor. The watch will be powered using a rechargeable battery that can be recharged using solar energy. The toe ring will measure the pulse oxygen levels and send the levels to the thin wristband through Bluetooth receptors. THe watch will also be equipped with a GPS system that is capable of sending the user's location to emergency services in case of an overdose. As oxygen saturation levels drop, the watch will begin to vibrate and flash in order to warn the user. The watch itself is water-resistant but not waterproof and made to be durable.

Features integrated from brainstorming sessions:

- Toe ring could be used as a finger ring instead to avoid damaging

Possible downsides:

- The size of the watch will have to be able to house the bluetooth receptor and software
- Ring can be easily lost or misplaced
- Ensuring that the bluetooth connection between the ring and watch isn't lost can be difficult and possibly require multiple softwares
- Could end up being very expensive

Things to think about:

- How to make the ring more durable but not bulky?
- What materials will the ring and watch be made of to ensure it is water resistant
- What is the maximum distance between devices before bluetooth connection is lost
- How will the ring be charged?
- Size and weight constraints

Final Design

The final design of the device had been determined by combining the best aspects of each concept. From the first design we decided to follow the same type of safety and user-friendly aspects. This is due to the first concept having the highest rating from our evaluation in those given criteria. From this, we have decided to ensure that the device is comfortable for the user, as well as poses no potential threat or danger. From the second concept we had decided to follow the same functionality aspects due to it having the highest rating in that category from our evaluation as well. We decided that this will be done by ensuring that the device's ability to measure oxygen saturation levels is the highest priority in terms of functionality. Moreover, from the third concept we had decided to use the design as our foundation for the device. Our final device has been decided to be a pulse oxygen saturation ring, which is connected to a watch through bluetooth. The ring and watch would be battery powered and rechargeable via solar energy; as well as have a GPS system within it. From the third concept there had been some alternation in changing the toe ring into a finger ring, as well as ensuring that the ring is worn on the same hand as the watch to ensure a strong and stable connection. With this final design, most of our concerns regarding each solution individually had been solved and or modified.

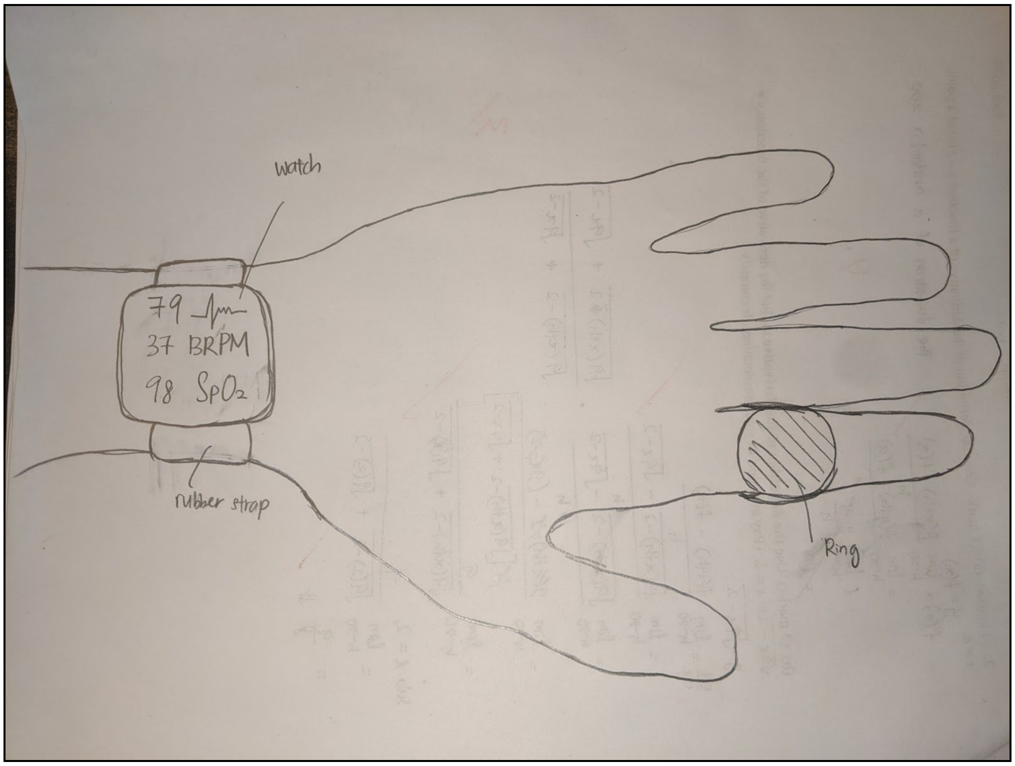
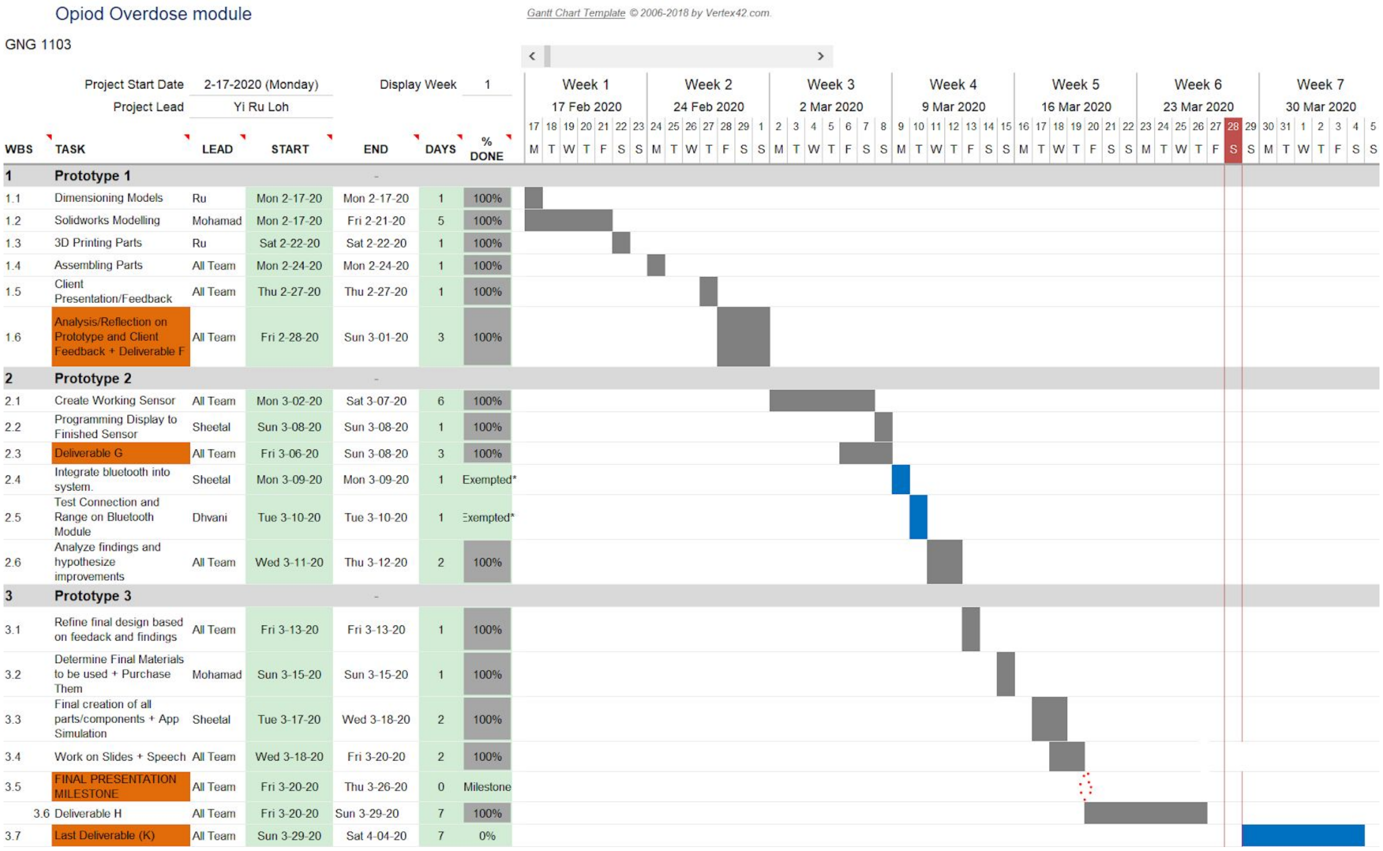


Figure 5 :A rough sketch of how the final design would be.

Project Plan, Execution, Tracking & Bill of Materials

Project Planning

A theme constantly picked up on in class was that project planning may seem to take a lot of time, but the time and stress it saves you makes it well worth it. This lesson was definitely picked up on. Over the course of the entire project, all group members referred to a Gantt chart *(Figure 6)* to orient themselves on what the next few weeks would consist of.



Opiod Overdose module

GNG 1103

Gantt Chart Template © 2006-2018 by Vertex42.com.

Project Start Date: 2-17-2020 (Monday)
Display Week: 1
Project Lead: Yi Ru Loh

WBS	TASK	LEAD	START	END	DAYS	% DONE
1	**Prototype 1**			-		
1.1	Dimensioning Models	Ru	Mon 2-17-20	Mon 2-17-20	1	100%
1.2	Solidworks Modelling	Mohamad	Mon 2-17-20	Fri 2-21-20	5	100%
1.3	3D Printing Parts	Ru	Sat 2-22-20	Sat 2-22-20	1	100%
1.4	Assembling Parts	All Team	Mon 2-24-20	Mon 2-24-20	1	100%
1.5	Client Presentation/Feedback	All Team	Thu 2-27-20	Thu 2-27-20	1	100%
1.6	Analysis/Reflection on Prototype and Client Feedback + Deliverable F	All Team	Fri 2-28-20	Sun 3-01-20	3	100%
2	**Prototype 2**			-		
2.1	Create Working Sensor	All Team	Mon 3-02-20	Sat 3-07-20	6	100%
2.2	Programming Display to Finished Sensor	Sheetal	Sun 3-08-20	Sun 3-08-20	1	100%
2.3	Deliverable G	All Team	Fri 3-06-20	Sun 3-08-20	3	100%
2.4	Integrate bluetooth into system.	Sheetal	Mon 3-09-20	Mon 3-09-20	1	Exempted*
2.5	Test Connection and Range on Bluetooth Module	Dhvani	Tue 3-10-20	Tue 3-10-20	1	Exempted*
2.6	Analyze findings and hypothesize improvements	All Team	Wed 3-11-20	Thu 3-12-20	2	100%
3	**Prototype 3**			-		
3.1	Refine final design based on feedack and findings	All Team	Fri 3-13-20	Fri 3-13-20	1	100%
3.2	Determine Final Materials to be used + Purchase Them	Mohamad	Sun 3-15-20	Sun 3-15-20	1	100%
3.3	Final creation of all parts/components + App Simulation	Sheetal	Tue 3-17-20	Wed 3-18-20	2	100%
3.4	Work on Slides + Speech	All Team	Wed 3-18-20	Fri 3-20-20	2	100%
3.5	FINAL PRESENTATION MILESTONE	All Team	Fri 3-20-20	Thu 3-26-20	0	Milestone
3.6	Deliverable H	All Team	Fri 3-20-20	Sun 3-29-20	7	100%
3.7	Last Deliverable (K)	All Team	Sun 3-29-20	Sat 4-04-20	7	0%

Figure 6 - Gantt Chart.

Communication

Although the Gantt chart provided a great schematic for what to expect and the deadlines ahead of us - copious communication was still required. The group used Skype almost daily. This would allow group members to share ideas, hold one another accountable, and most importantly - communicate via voice/video call when we are working on group deliverables.
If it weren't for the several hundred Skype messages that went back and forth between group members, we would've been in a far worse spot - likely not yet being very comfortable and open with each other and definitely not as organized.

Tracking

As aforementioned, all group members used Skype to hold each other accountable - asking things like "How's the solidworks modelling going? Will you have it ready by tomorrow? Do you need any help with it?" This very quickly established a mutual understanding that each and every single one of us are responsible for the overall success of the group - we count on each other and will openly express concerns with deadlines if a group member seems to be making the group fall behind.

In addition, most (if not all) Thursdays, the group would meet with the lab TA's. The instructors would ask us what's next up, what troubles we were currently having, and they would offer some great advice for what we should be doing.

Cost Tracking & Bill of Materials

Cost was constantly on our minds throughout the project. Before any purchases were made, group members performed hours worth of benchmarking and product comparisons. Once all group members agreed upon the purchase, the item would be bought.

Although searching up the tiniest hardware components for several hours seems a bit excessive, oftentimes through this action, we were able to find cheaper, and more reliable options. This proved to be extremely helpful since we stayed under our CAD$100 budget - *Table 5* displays our final bill of materials.

Date	Action	Product	Cost (+/- $CAD)
Feb 28, 2020	Buy	Max 30100 Oximeter Sensor	-11.50
Feb 28, 2020	Buy	Bluno Nano	-47.03
March 7, 2020	*Buy*	*Arduino Nano*	*-7.99*
March 10,2020	*Buy*	*HC-05 Bluetooth Module*	*-13.00*
March 13, 2020	Return	Bluno Nano	+37.48
March 13, 2020	Buy	Shipping labels/packaging for the return	-18.20
March 14, 2020	Buy	Max 30100 Oximeter Sensor	-11.50
March 16, 2020	Buy	Sun Founder 16x2 LCD Display	-15.99
TOTAL REMAINING	-	-	**=12.27**

Table 5: Bill of materials and the remaining amount from the $CAD 100 budget. Purchases made from the Makerspace are in italics, all other purchases are made through Amazon or RobotShop.

By reading about components on technical hardware forums, the group was able to come across a tremendous amount of open-source resources that we could apply to our project (ex: source code, arduino libraries, Android app codes, wiring and soldering schematics, etc.). The significance of these resources will be further analyzed in “Ideal Final Solution, Failures, and Intended Tasks” on page 26.

Analysis

As talked about in the introduction, there were a total of 3 prototypes that were conducted and each having their own purpose. After the list of materials and hardware components we would incorporate into our device were bought, we started to work on the prototypes. With keeping in mind the 5 step design process and the key concepts that the client wanted in the final device.

Prototype I:

The main purpose of this prototype was to give a basic idea of what the device would visually look like and how we planned on making it work. Here we 3D printed a watch and a ring, the 2 components of our device. The design of the watch was very basic and made using solidworks to give us a visual of how big it would need to be so that it is non-intrusive and can hold all the hardware. As for the ring, we printed it with PLA, the same material as the watch, however the predetermined size would not fit everyone. Therefore, we later changed the material so that it was more flexible and could fit fingers of all sizes.

Prototype II:

As for this prototype, we mainly focused on incorporating the hardware and software components together. Making sure that the raw hardware components that we ordered were in working condition and then connected them to computer code that would help detect information for the device to give us the working product. During this step we faced many challenges, such as our sensor being damaged due to the way we tried soldering the hardware component and this resulted in a lot of lost time and confusion as to if the problem was with the hardware or the software. We also faced a problem during the process of trying to figure out how the coordinates of the user would be sent to the emergency services during an overdose, however, after a lot of hard work and research we were back on track and had it all figured out.

Prototype III:

The objective of this prototype was to create a final working product that consisted of both the previous prototypes. Here we had the combined hardware and software components working together and displaying the data, as well as the physical object where all of this would be incorporated into one. Our final device layout was finalized and we had a plan laid out on how we would go about doing things, however, because of COVID-19 we were unable to have a visually appealing final product. Although the device did its job that it was meant to.

Final Solution and Status Update

Hardware of Actual Final Design

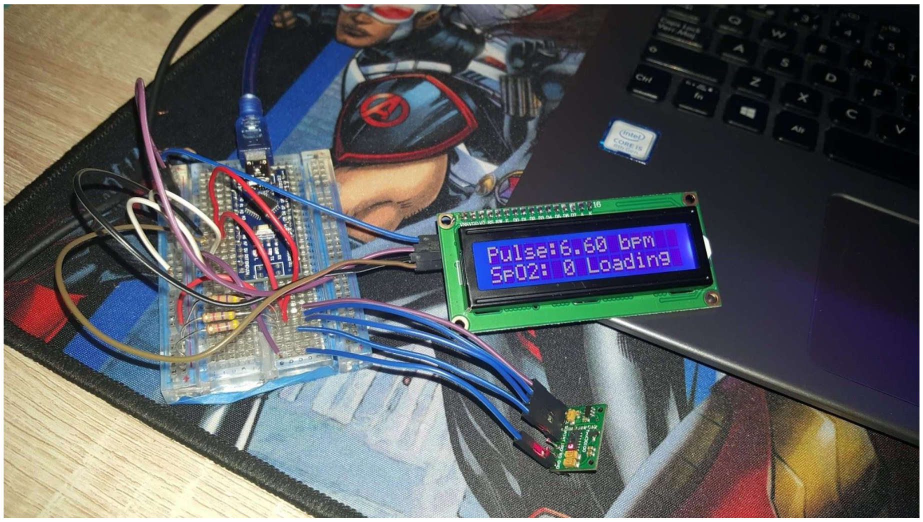


Figure 7 - Final Physical Device.

The physical assembly was not as pretty as we expected it to be, however, it was functional. Essentially, in order to receive oxygen saturation readings as well as pulse, a MAX30100 oximeter chip was used (*Figure 8*).

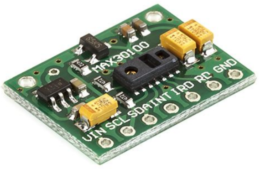


Figure 8 - Max 30100 Oximeter Sensor.

This sensor is then connected to the breadboard through some thin wires and attached to the arduino nano. From there, the arduino would be connected to an LCD display (*Figure 9*). This display is responsible for displaying the oximeter values to the user.

Figure 9 - LCD Display.

The arduino nano microcontroller (*Figure 10*) is responsible for making sure that all the aforementioned components are fully functional in a system of code which homogenizes all subsystems into one.

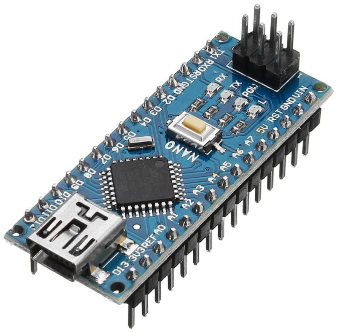


Figure 10 - Arduino Nano.

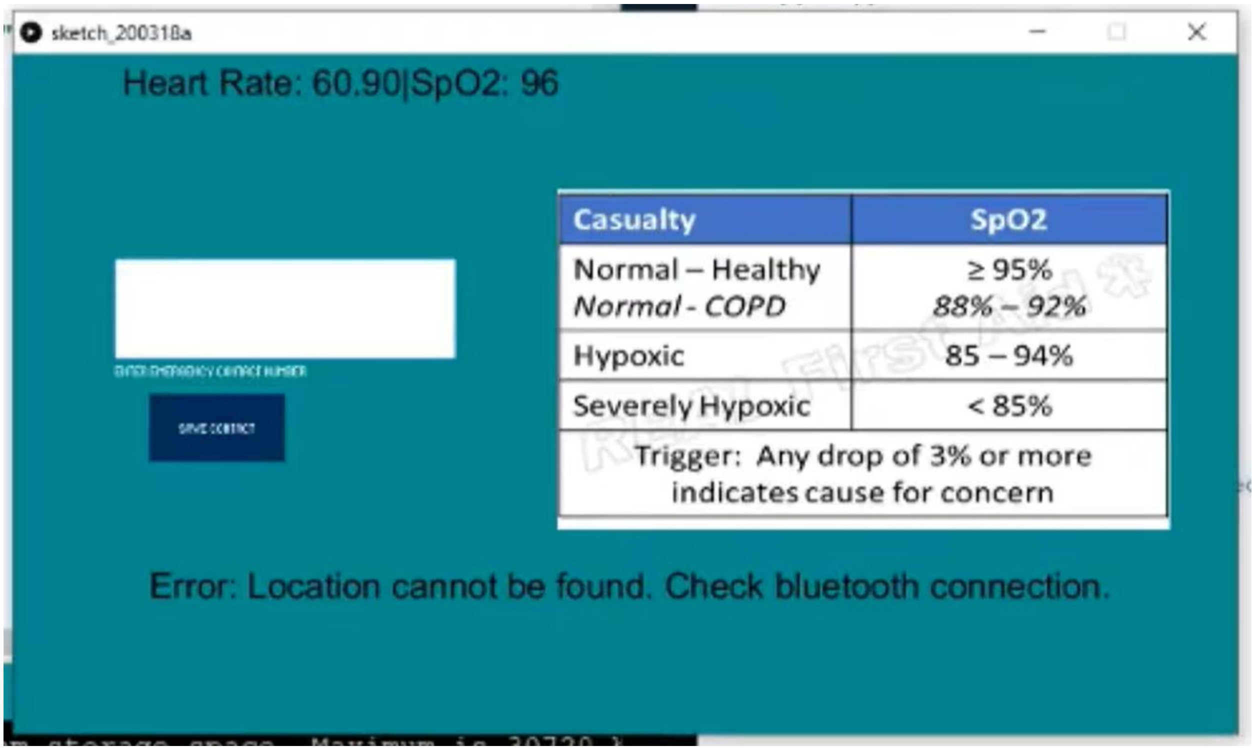
Software (All Code Used is in Appendix I)

The final design’s software is primarily arduino code. As aforementioned, by benchmarking hardware components, the group ran into several forums which mentioned the parts. These forums often had user-posted projects involving the parts we searched. The significance of coming across these forums was that before we were even shipped our parts, we had tremendous amounts of code which we could adapt to our project.

Once we were shipped the parts, it was just a matter of making some changes to the open-source code (such as changing pin identification numbers, delays, print statements, etc.) and doing some standard debugging.

The arduino code is responsible for the following tasks:
- Making sure that the oximeter sensor is successfully receiving and interpreting data.
- Ensuring that information is properly relayed to the LCD screen
- Ensuring that the oximeter readings are properly sent to the desktop application.

Another subsystem of code was used to create a desktop application (Figure 11). This app was made to compensate for the Android application we intended to have (Page 26).



Heart Rate: 60.90|SpO2: 96

Casualty	SpO2
Normal – Healthy *Normal - COPD*	≥ 95% *88% – 92%*
Hypoxic	85 – 94%
Severely Hypoxic	< 85%
Trigger: Any drop of 3% or more indicates cause for concern	

Error: Location cannot be found. Check bluetooth connection.

Figure 11 - Desktop application to simulate some of the Android app's capabilities (displaying oximeter data, displaying GPS coordinates, detecting critically low SpO2 levels, etc.).

The application was created in Processing - an IDE typically used for its strong ability to create GUI's. It was coded for in Java, a language which no group members were familiar with but very quickly learned.

Ideal Final Solution, its Failures, and Intended Tasks

The group established an ideal design for the final product over a month ago (*Figure 12*).

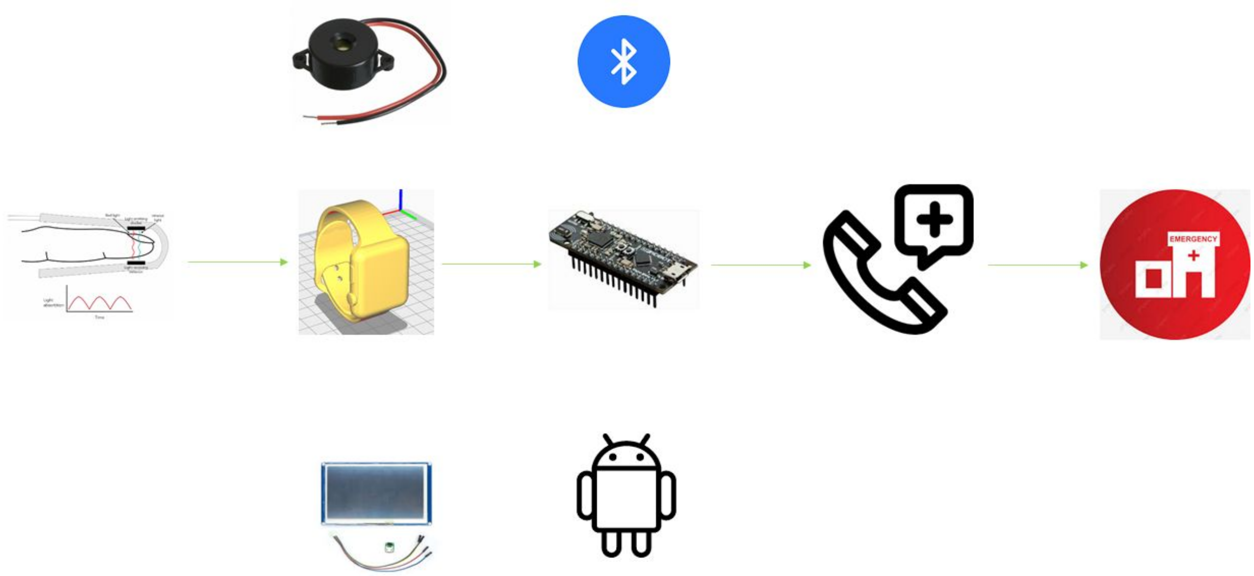


Figure 12 - An ideal flow chart of the final device with the following sequence:
Oximeter data is recorded at the finger
Data sent to a display on the watch
Data interpreted/regulated by an arduino nano

If the user's vitals are indicative of an overdose (i.e SpO_2<90%):

- A speaker tries to keep them awake through alerts
- Information that is sent to an Android app via bluetooth will be sent as an SMS to an emergency contact.
- The emergency contact will call provide the SMS information they got to EMS.

In reality, a lot of these subsystems were omitted due to reasons that will be discussed later. The group's actual final design is displayed in *Figure 13*.

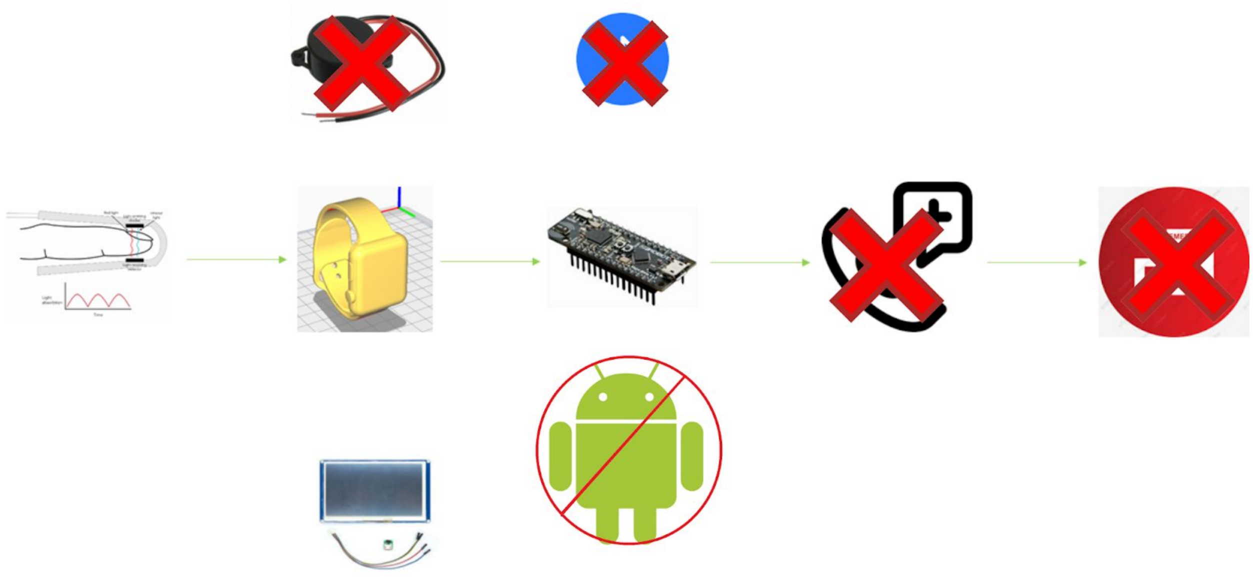


Figure 13 - Actual final design’s components. The components/subsystems crossed out have their failures further analyzed in *Table 6.*

Hardware left out of Final Design

We intended to use an HC-05 bluetooth module (*Figure 14*) which would transmit sensor data to an Android application (discussed further on page 28). However, a complication arose which prevented this subsystem’s completion. One group member, Ru, has the HC-05 module, and is also the only one with an Android phone. Therefore, we could not get a full bluetooth connection between the app and hardware since Sheetal (the group member which has the arduino/hardware systems) and Ru are on different ends of the province.

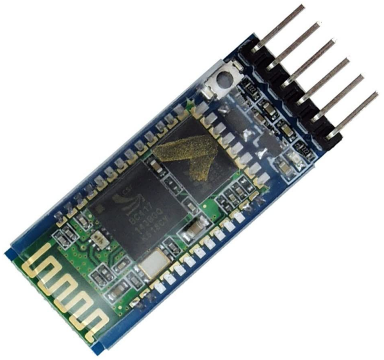


Figure 14 - HC-05 Bluetooth Module.

We also intended to use a small arduino speaker (*Figure 15*). The purpose of this was to try to keep the overdosing user awake longer (by omitting alarming audio cues), this functionality was picked up on during a client meeting.

Figure 15 - Arduino Micro Speaker.

The reason for this failure is that it busted after some testing done very shortly after we got it. The group planned on purchasing another speaker, but at the moment there were more important (and potentially costly) priorities.

Software left out of Final Design

The group intended on using an Android application as part of the device. The phone app was created using MIT app inventor *(code is shown in Appendix I)*, one week before campus got shut down and group members were separated. It was essentially supposed to tie up the oximeter data and gps subsystem into one program. **It's real purpose is just to execute one if-statement** - that is, if the SpO2 levels are below 90%, send the phone's coordinates, last recorded pulse, and SpO2 levels to a pre-user-saved emergency number. A video demo is shown in *Appendix III: Working Video Demo* of what it was supposed to be. *Figure 16* shows the user interface.

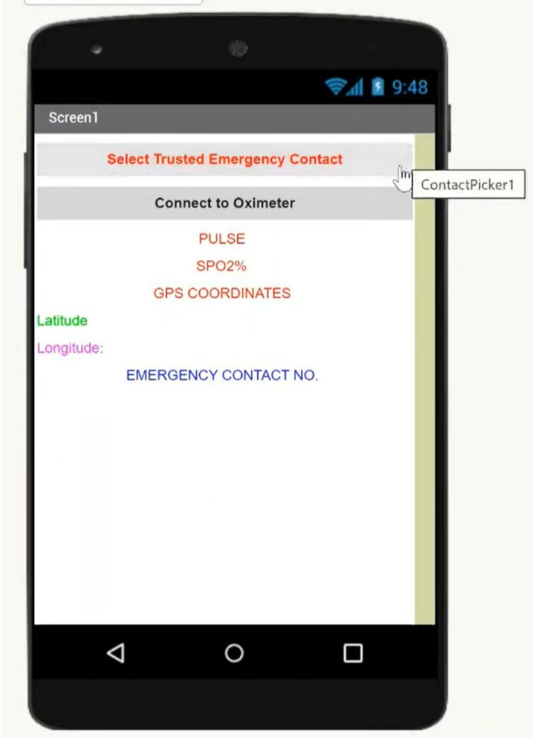


Figure 16 - User interface of the final application.

Aesthetics left out of Final Design

Had we more time to worry about the visual appeal of the system and less to worry about the actual functionality of it (which was our priority), we would have tried to encase everything in a watch and ring (*Figure 17*) and started benchmarking comfortable watch straps.

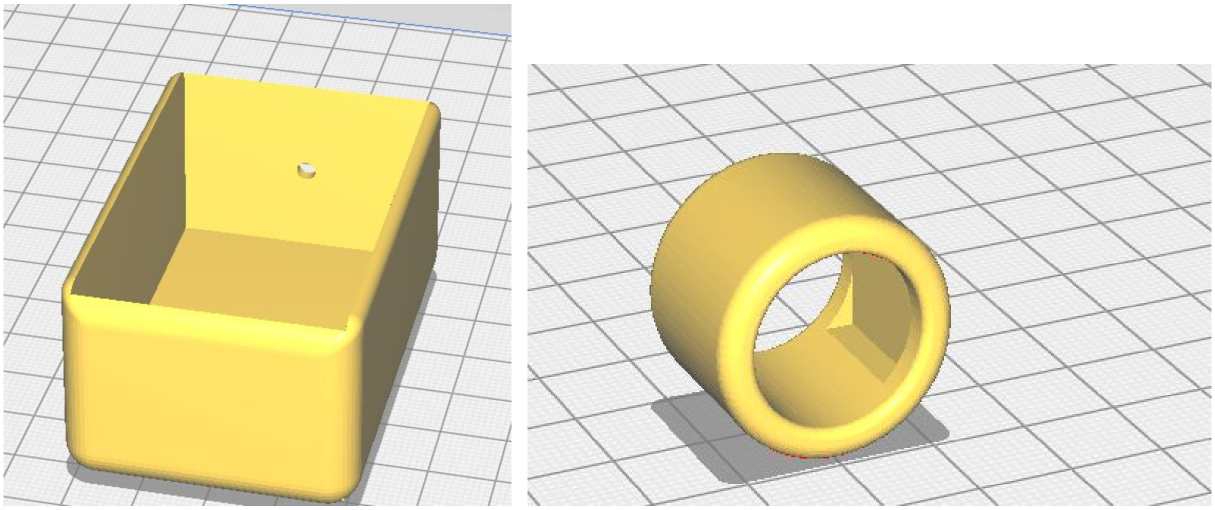


Figure 17 - Models of a watch case (left) and ring (right) ready to be 3D printed and encase our components. All modelling and dimensioning were done in solidworks to ensure that the hardware would sit flush (**watch case**: 69mm x 44mm x 32mm, **ring**: 20mm x 25mm x 25 mm). Ideally, the final assembly of the product would have looked like *Figure 18*.

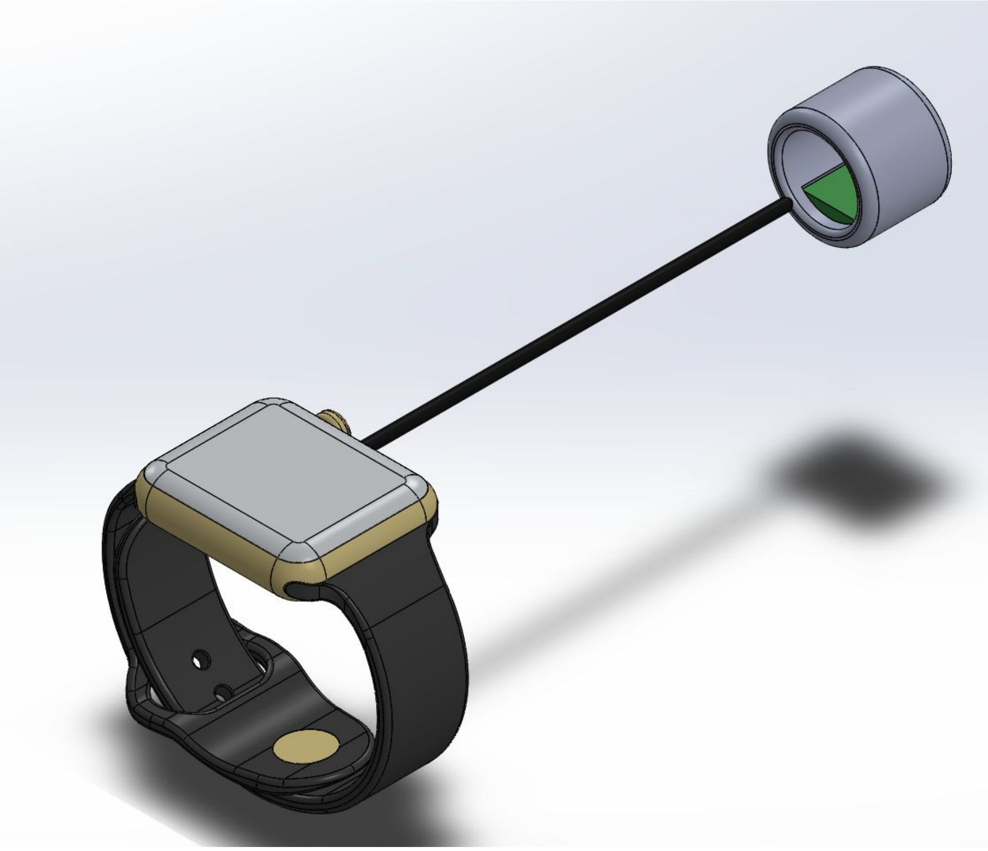


Figure 18 - An ideal visualization of the final assembly we intended to have (Oximeter in a ring connected to hardware in a wrist watch via a wire). Crafted in Solidworks.

Causes of Failures

The main limiting factor was ultimately realized to be the bluetooth subsystem. There were so many features that were partially omitted as they were dependent on having a bluetooth connection *(Figure 19*).

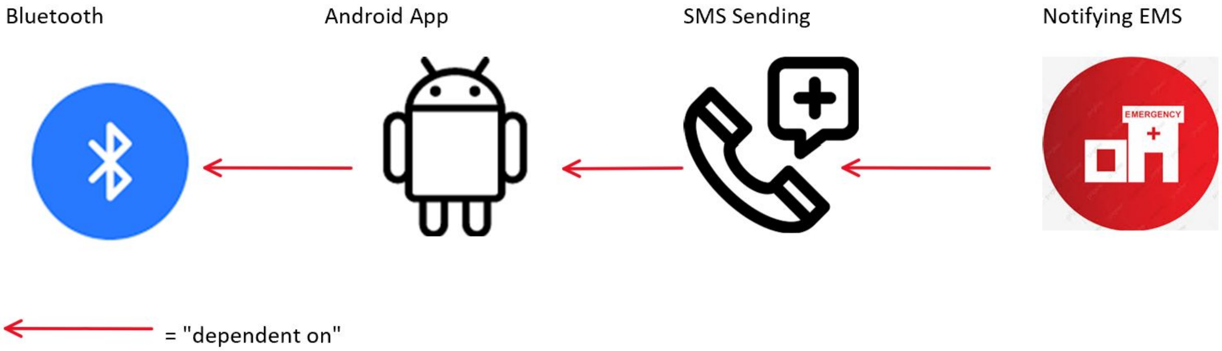


Figure 19 - An illustration (meant to be read from right to left) that shows the implications that the bluetooth failure had on other dependent subsystems.

Summary of Failures

Component/ Subsystem	Purpose	Reason for Failure	Testing done/intended
Speaker	Alert overdosing users via audible cues in efforts of keeping them conscious.	The system failed spontaneously - assumed to have busted.	We intended to test if the speaker went off at the right time (when oxygen levels are below 90%). We also intended to see where we could place the speaker in the watch without it protruding in an obtrusive manner.
Bluetooth	Send oximeter data to the Android app.	Oximeter/hardware was hundreds of kilometres away from the bluetooth module.	We would have tested if the bluetooth module sent reliable data without failure (all the time). This would have been done by keeping the system running for hours and seeing if at any point no data was being sent to the phone.
Android app	Detect phone GPS coordinates and have them ready to be sent via SMS, along with oximeter data, if an overdose is detected.	Dependency on Bluetooth.	GPS subsystem has been successfully tested (refer to “App Subsystem”).
SMS Sending	Send GPS coordinates, pulse, SpO2, to a pre-set emergency contact.	Dependency on Android app.	Although we just tested the sending of GPS coordinates, we assume that we could have sent the other information via SMS. We tested this system by repeatedly sending data via text and seeing if it was done in a timely manner.
Notifying EMS	If an SMS is successfully sent, then EMS can successfully be sent the information through the emergency contact.	Dependency on SMS sending.	The testing for this system is just ensuring that SMS messages are successfully received. This was tested by seeing how long it took for someone to receive an SMS. It was found to be successful.

Table 6 - All subsystems/components that failed or weren’t included in our final design, what their purpose was, why they failed, and the testing we did or intended to do on them (with colours indicative of our success doing so).

Intended Actions if COVID-19 CEED Closures Didn't Occur

1. Printing our final SolidWorks design (**1 day)**
    a. This includes: Watch and ring components (*Figure 17*) + additional add-ons
2. Testing comfortability of design (**1 day)**
    a. This includes: Determining which materials would be better suited for our application
3. Soldering the circuitry to a metal circuit board (PCB) for permanence (**1 day)**
    a. This includes: Ensuring all aspects of the circuit are fully functioning and switching to the use of a battery rather than USB power input.
4. Making the circuit fit inside of the 3D printed pieces (**1 day)**
    a. This includes: Ensuring everything is secure and does not fall apart during use
5. Testing with device (**2 days)**
    a. This includes: Wearing the device for long periods of time and testing its capabilities
        i. Such as: bluetooth connection distance, battery life, waterproofing, etc
6. Making improvements for any discovered issues **(1-2 days)**
    a. This includes: Remodelling design if needed and any adjustments
7. Aesthetics (**<1 day)**
    a. This includes: Ensuring the product is appealing and meets our criteria
8. Ensuring all software is still functioning after compiling (**1-2 days)**
    a. This includes: Testing the app and ensuring the user is able to connect through bluetooth seamlessly, accurate readings are being made, and is able to send out emergency information in case of overdose.
9. Preparing for presentation and Design Day (**2 days)**
    a. This includes: Creating a helpful presentation and practicing to ensure everything goes smoothly during the event.

Outline for the Manufacturing, Troubleshooting, and Maintenance of the Device

The hardest parts of building the device were by far finding which components to buy, and figuring out why things were failing (troubleshooting). Throughout this document, the answers on how to make this device can be interpolated. However, this section condenses the information for a user which wishes to build and maintain their own device.

Manufacturing the Device

This will cover how to get to the point to which our device is currently at. **If at any point you are running into problems, refer to *"*Troubleshooting the Device*"* on the page below**. For the remaining work we intended to do, refer to page 28, "*Intended Actions if COVID-19 CEED Closures Didn't Occur."*

1. Purchase the following materials
   i) Arduino Nano
   ii) Max30100 Oximeter Sensor
   iii) Sun Founder 16x2 LCD
   iv) HC-05 Bluetooth Module
   v) Connection wires + diverse resistors
2. Using a breadboard, follow the wiring schematics shown in *Appendix I*.
3. Using the Arduino IDE, use the code provided in *Appendix I*. Note that some minor changes may need to be done since you may not have made connections to the exact pins we did.
4. After a stable working condition has been observed in the system, use the code in *Appendix I* to create the Android app. The program you will use is MIT App Inventor 2.
5. Download MIT App Companion on an Android phone and follow the instructions on how to get the app from your computer to your phone.
6. Ensure that all components (including the app) are fully functional before any permanent connections are made.
7. To complete the device, follow the instruction on page 32, "*Intended Actions if COVID-19 CEED Closures Didn't Occur."*

Troubleshooting the Device

When creating this device, the possibility of issues arising is fairly large. In this section, how to overcome and solve common issues with the Opioid Overdose Sensor will be explained.

Hardware Issues:

Common hardware issues that can arise while compiling the following device mainly involve the Max30100 sensor. The sensor itself is fairly durable, however can be easily damaged if not careful. When connecting the sensor to the device, the correct resistance is required in order to have optimal results. In the case of this project, we required approximately 30kΩ. In order to achieve this, it was required of us to remove the pre-existing resistors found on the sensor, refer to *Figure 8.* Following this, if done correctly, the sensor should update the arduino serial monitor accurately and omit a bright red light. If done incorrectly the following may occur:

1. The sensor lights up, but is hot to the touch
2. The sensor does not light up correctly
3. The sensor lights up correctly, is not hot to the touch, but does not display data properly

In order to solve the first issue, it is important to ensure the correct resistors are used, as well as are properly connected to the circuit. Troubleshooting can be done by only connecting a ground and power to the sensor with a resistor in order to test the resistance.

For the second issue, in order to troubleshoot it is important to ensure the Arduino sketch is running before making judgment. In some cases, the sensor light does not activate until the code is running. If the code is running and the light is still not working, try taking apart the circuit and rebuilding it, or as described above, make a connection of only power and ground with a resistor to see if the light is functional. Lastly, check the sensor chip itself for any potential damage from soldering.

Moreover, for the third issue the most likely culprit is damage from soldering. If the sensor is incapable of receiving and transmitting data but appears to be functional and has no other obvious issues, it is almost certain there is an issue with the chip itself and may require replacing.

Any additional hardware issues that may occur are most likely related to parts being soldering incorrectly or faulty wiring, This issues can be easily tested by removing each component individually and testing them using a voltmeter or simply based on whether the part turns on.

Software Issues:

Common software issues that may occur when creating this device are generally very simple to fix. The following issues regarding the software of the device are the most common:

1. No data appears in the Arduino Serial Monitor
2. "________ does not name a type/program will not compile"
3. Data is not printing to LCD
4. LCD does not clear data after each reading
5. Processing sketch does not run
6. Processing sketch is not displaying data

In order to solve the first issue, it is important to ensure that the sensor is correctly connected and the Arduino sketch is uploaded to the device. Troubleshooting this issue is fairly easy, and just requires double checking connections and uploading the sketch

In addition, the second issue is due to improper importing of libraries. In order to see how to properly do this, refer to *Appendix I*. Troubleshooting this issues requires that the code is typed in properly and all libraries are spelt correctly

Following this, the third software issue is generally caused by the missing of the code lcd.print. In order for the code to print onto the display rather than the serial monitor, lcd,print must be used. Solving the fourth issue can also be solved in the same manner by using the code lcd.clear(); in order to clear the displayed data each time it is updated.

In order to troubleshoot the fifth and sixth issue, it is important to ensure that Arduino sketch is not running while trying to start the desktop application on Processing. In addition, making sure there are no evident issues in the code indicated by a red underline is greatly important in making sure everything runs smoothly. If the sketch is not displaying data, check all connections, specifically those involving the sensor. Finally, in order to double check the connections are correct, opening the serial monitor within the Arduino sketch can determine the issue. If values are being displayed there, then everything should be functioning properly.

IMPORTANT: Ensure all required libraries are downloaded and properly integrated into the Arduino application. If unsure on how to do this, follow a tutorial either on YouTube or Google.

Maintenance of the Device

From our testing, we found that the device should not be overcharged to avoid overheating. It is also important to follow the guideline given on purchasing the battery as a battery that has higher voltage than the suggested voltage will burn the components of the device. Besides that, it is important to always keep the oximeter sensors clean whenever possible to avoid error or obstruction in reading the blood oxygen saturation levels and heart rate. In addition, the device has to stay away from water as it is not waterproof and if it contacts water, it may cause an accident where the user will be short-circuited. When designing the device, a clearance fit is designed to ensure easier access to replace or remove the components in the device for maintenance purposes in the future. If the device malfunctioned, there are steps that the user can take, such as,

1. Remove the casing of the device and carry out individual tests on each component using voltmeter, ammeter and other testing instruments that are suitable.
2. Replace the faulty components and resolder.
3. If all components are in good condition and the device is still not working, try resoldering all the components as the problem may be that the soldering has loosen over a long period.
4. Lastly, if steps above are all taken and the device is not working, it may not be a hardware problem but software issue. Thus, refer to the Troubleshooting steps as recommended in the section above.

Conclusions and Recommendations for Future Work

Our final device is in working condition, however, it is not aesthetically pleasing because we could not 3D print our final prototype due to the makerspace closure. Although we would've had a working and well put together final design if we were to not run into the COVID-19 complication. This product was still made by university students in the end and could use improvements from professionals and more knowledgeable individuals. We put in our best efforts and to our capabilities made this device that met the client needs specified at the beginning of the semester.

Bibliography

Knox, D. (2020). GNG 1103 Lecture Notes. Retrieved from https://uottawa.brightspace.com/d2l/le/content/140895/Home

Appendices

Appendix I: Code and Wiring Schematics Used

I.a - Arduino Code:

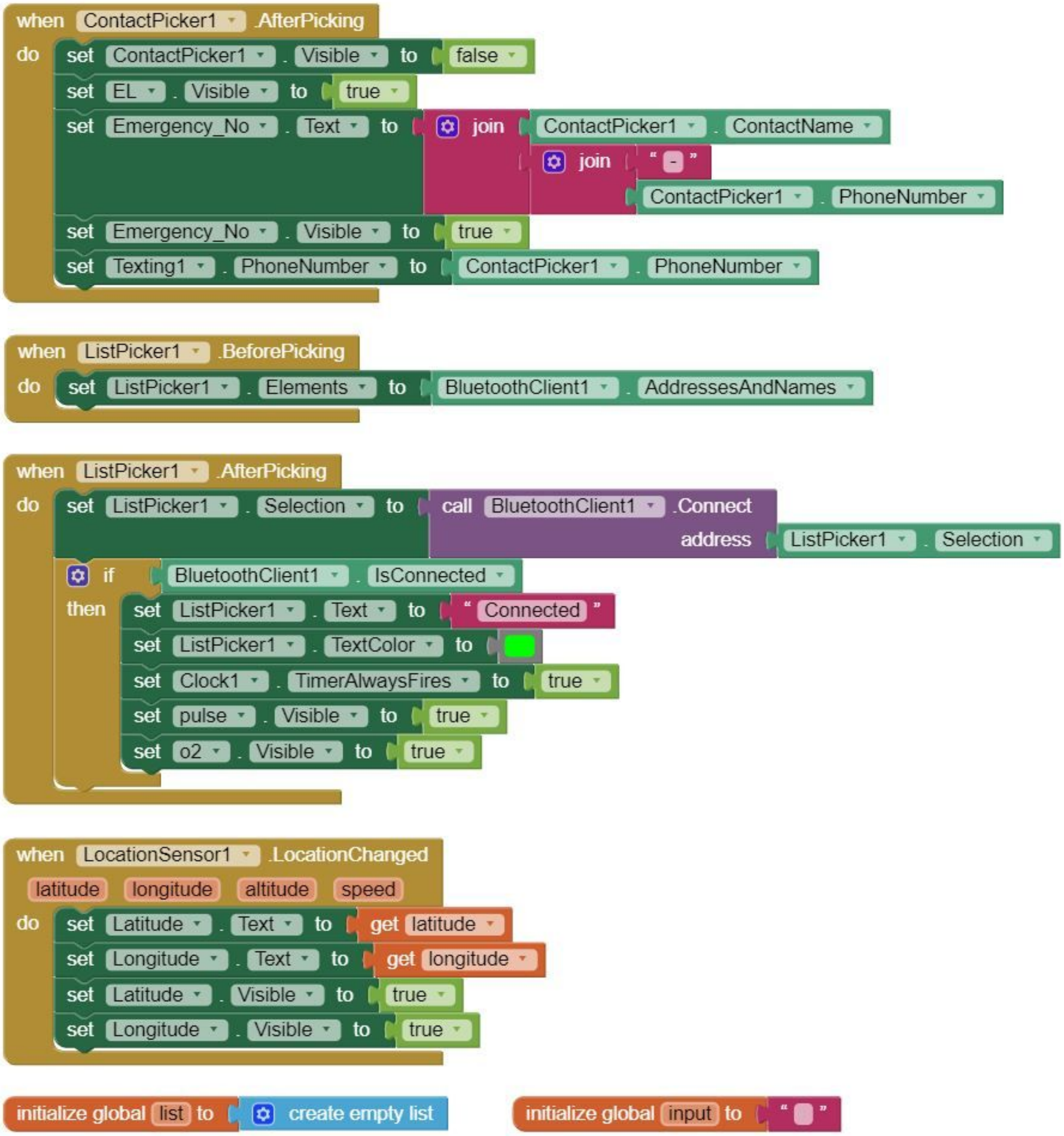
```
#include <Wire.h>
#include "MAX30100_PulseOximeter.h"
#include <LiquidCrystal_I2C.h>
//#include <SoftwareSerial.h>

#define TxD 1
#define RxD 0

//SoftwareSerial bluetoothSerial(TxD, RxD);
LiquidCrystal_I2C lcd(0x27, 16, 2);
String a = "";
#define REPORTING_PERIOD_MS     1000

// PulseOximeter is the higher level interface to the sensor
// it offers:
//  * beat detection reporting
//  * heart rate calculation
//  * SpO2 (oxidation level) calculation
PulseOximeter pox;

uint32_t tsLastReport = 0;

// Callback (registered below) fired when a pulse is detected
void onBeatDetected()
{
   //Serial.println("Beat!");
}

void setup()
{
//    bluetoothSerial.begin(9600);
   Serial.begin(9600);
   lcd.init();
```

```
    lcd.backlight();
    //Serial.print("Initializing pulse oximeter..");
    lcd.print("Initialzing pulse oximeter...");


    // Initialize the PulseOximeter instance
    // Failures are generally due to an improper I2C wiring, missing power supply
    // or wrong target chip
    if (!pox.begin()) {
        //Serial.println("FAILED");
        for(;;);
    } else {
        //Serial.println("SUCCESS");
    }


    // The default current for the IR LED is 50mA and it could be changed
    //   by uncommenting the following line. Check MAX30100_Registers.h for all the
    //   available options.
    // pox.setIRLedCurrent(MAX30100_LED_CURR_7_6MA);

    // Register a callback for the beat detection
    pox.setOnBeatDetectedCallback(onBeatDetected);
}

void loop()
{
    // Make sure to call update as fast as possible
    pox.update();

    // Asynchronously dump heart rate and oxidation levels to the serial
    // For both, a value of 0 means "invalid"
    if (millis() - tsLastReport > REPORTING_PERIOD_MS) {
        lcd.clear();
        lcd.print("Pulse:");
        lcd.print(pox.getHeartRate());
        lcd.print(" bpm");
        lcd.setCursor(0,1);
        lcd.print("SpO2: ");
        lcd.print(pox.getSpO2());
        Serial.print("Heart Rate: ");
        Serial.print(pox.getHeartRate());
        Serial.print("|");
        Serial.print("SpO2: ");
        Serial.print(pox.getSpO2());
        Serial.print("\n");

        if (pox.getSpO2() <= 0){
         Serial.print("INITALIZING...");
         lcd.print(" Loading");
        }

        if (pox.getSpO2() <= 100){
         Serial.print("");
         lcd.print(" Loading");
        }


        tsLastReport = millis();

    }
}
```

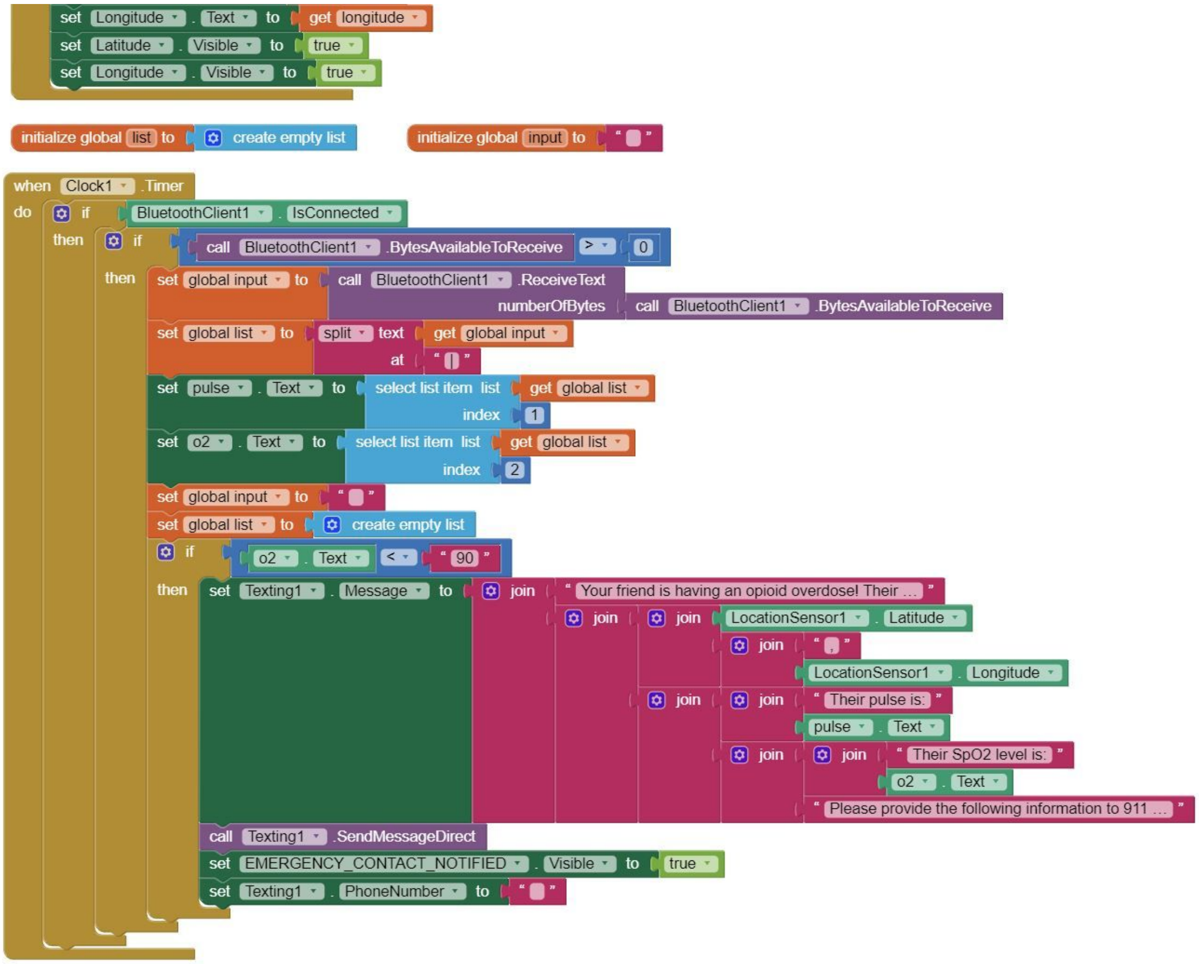
I.b - Processing Code:

```
import controlP5.*;
import processing.serial.*;

ControlP5 cp5;
Serial mySerial;
int x = 0;

//shit for the button
PFont f;
PImage img;

float x1 = 100;
float y1 = 50;
float w1 = 150;
float h1 = 80;

Textfield number;
String value = "";

public void settings(){
  size(900,500);
}

void setup(){

  img = loadImage("heart.png");

  background(255);
  stroke(0);
  noFill();


  f = createFont("Arial",42,true);




  mySerial = new Serial(this, "COM8", 9600);
  println(" ");

  cp5 = new ControlP5(this);

  //adding buttons
  //cp5.addButton("Contact")
  //.setValue(0)
  //.setPosition(600,170)
  //.setSize(100,50)
  //;

    //adding buttons
  cp5.addButton("Save Contact")
  .setValue(0)
  .setPosition(100,250)
  .setSize(100,50)
  ;

  //adding textfields

 // number = cp5.addTextfield("Enter Emergency Contact Name")
  //.setPosition(100,100)
  //.setSize(250,50)
  //.setAutoClear(false)
  //.setColorBackground(color(255,255,255))
  //.setColorValueLabel(color(0,0,0))
  //;

  number = cp5.addTextfield("Enter Emergency Contact Number")
  .setPosition(75,150)
  .setSize(250,75)
  .setAutoClear(false)
  .setColorBackground(color(255,255,255))
  .setColorValueLabel(color(0,0,0))
  ;



}

void draw(){

 if(x == 0){
  background(0,143,143);
  textSize(28);
  text("ahAHHA...",100,0);
  delay(1000);
  x=1;
  }

  if(x ==1){
  background(0,143,143);
  if (mySerial.available() > 0){
    value = mySerial.readString();

  }

  text(value,80,30);
  delay(700);

  }

  textFont(f,26);
  fill(0);
  text("Error: Location cannot be found. Check bluetooth connection.", 100,400);


  image(img, 400, 100, width/2, height/2);
}
```

I.c - MIT App Inventor code for Android app

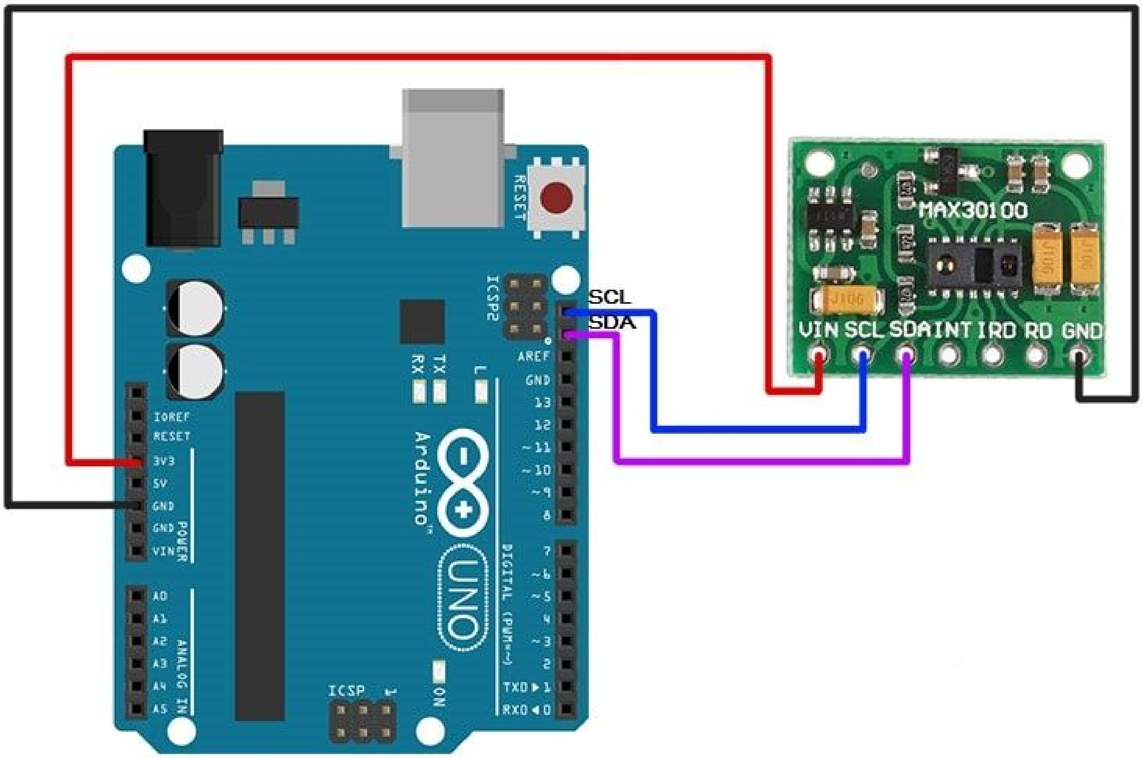


```
when ContactPicker1 .AfterPicking
do  set ContactPicker1 . Visible to false
    set EL . Visible to true
    set Emergency_No . Text to join ContactPicker1 . ContactName
                                    join " - "
                                         ContactPicker1 . PhoneNumber
    set Emergency_No . Visible to true
    set Texting1 . PhoneNumber to ContactPicker1 . PhoneNumber

when ListPicker1 .BeforePicking
do  set ListPicker1 . Elements to BluetoothClient1 . AddressesAndNames

when ListPicker1 .AfterPicking
do  set ListPicker1 . Selection to call BluetoothClient1 .Connect
                                                  address ListPicker1 . Selection
    if   BluetoothClient1 . IsConnected
    then set ListPicker1 . Text to " Connected "
         set ListPicker1 . TextColor to 🟩
         set Clock1 . TimerAlwaysFires to true
         set pulse . Visible to true
         set o2 . Visible to true

when LocationSensor1 .LocationChanged
 latitude  longitude  altitude  speed
do  set Latitude . Text to get latitude
    set Longitude . Text to get longitude
    set Latitude . Visible to true
    set Longitude . Visible to true

initialize global list to create empty list        initialize global input to " "
```

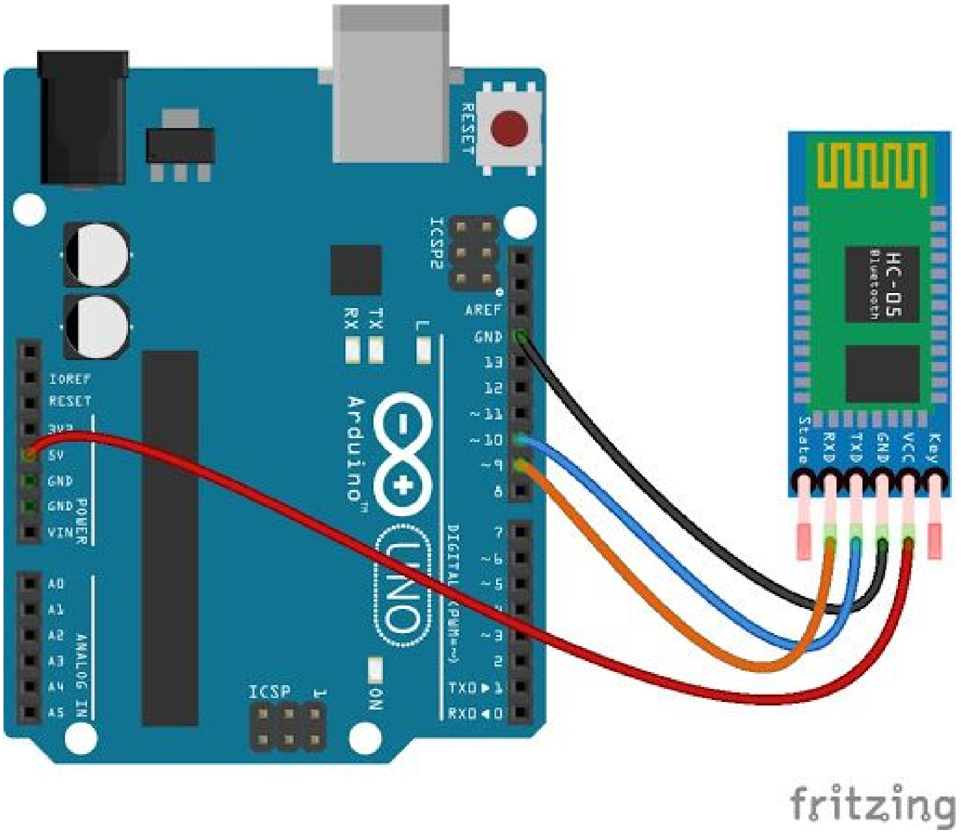


```
    set Longitude . Text to get longitude
    set Latitude . Visible to true
    set Longitude . Visible to true

initialize global list to create empty list        initialize global input to " "

when Clock1 .Timer
do  if BluetoothClient1 . IsConnected
    then  if call BluetoothClient1 .BytesAvailableToReceive > 0
          then  set global input to call BluetoothClient1 .ReceiveText
                                          numberOfBytes call BluetoothClient1 .BytesAvailableToReceive
                set global list to split text get global input
                                         at " | "
                set pulse . Text to select list item list get global list
                                                     index 1
                set o2 . Text to select list item list get global list
                                                  index 2
                set global input to " "
                set global list to create empty list
                if o2 . Text < " 90 "
                then  set Texting1 . Message to join " Your friend is having an opioid overdose! Their ... "
                                                     join join LocationSensor1 . Latitude
                                                               join " , "
                                                                    LocationSensor1 . Longitude
                                                          join join " Their pulse is: "
                                                                    pulse . Text
                                                               join join " Their SpO2 level is: "
                                                                         o2 . Text
                                                                    " Please provide the following information to 911 ... "
                      call Texting1 .SendMessageDirect
                      set EMERGENCY_CONTACT_NOTIFIED . Visible to true
                      set Texting1 . PhoneNumber to " "
```

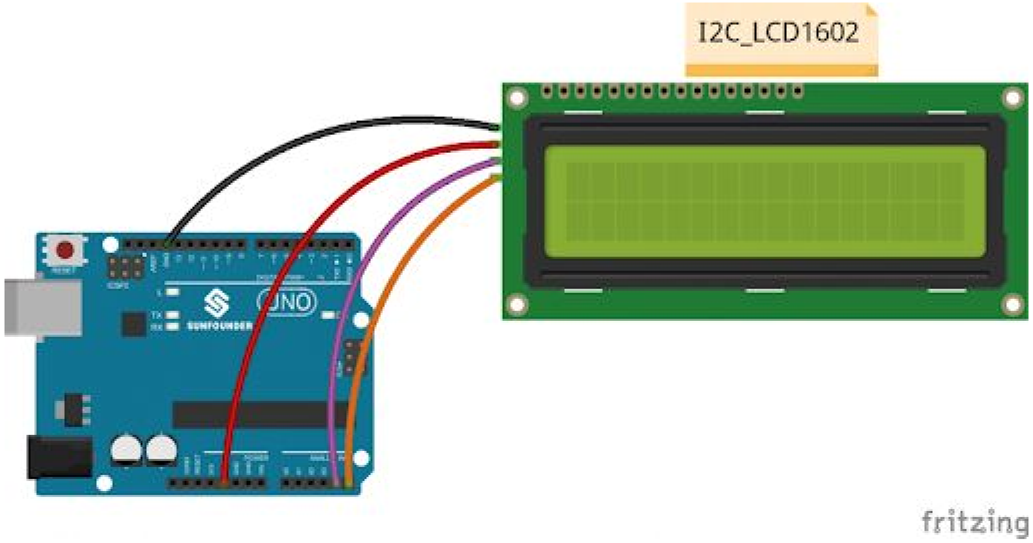
I.d - Max30100 Oximeter Wiring Schematic



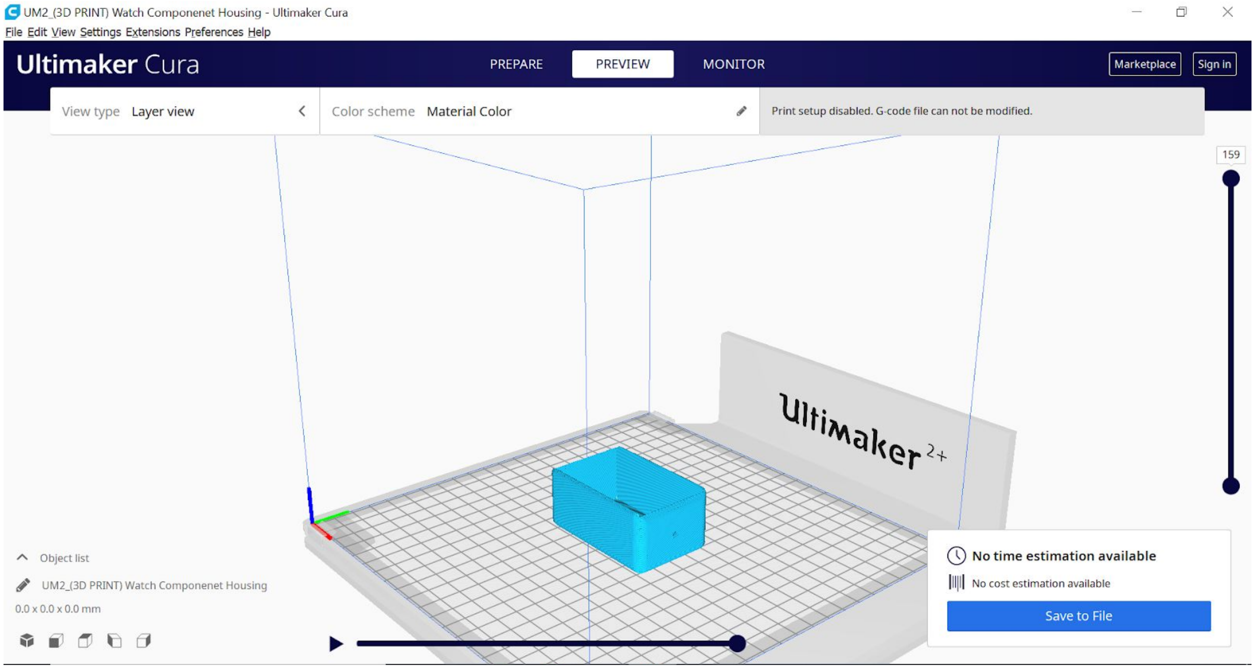
I.e - HC-05 Bluetooth Module Wiring Schematic



I.f - SunFounder 16x2 LCD Wiring Schematic



Appendix II: All CAD Models of Conceived Prototypes


UM2_(3D PRINT) Watch Componenet Housing - Ultimaker Cura
File Edit View Settings Extensions Preferences Help
Ultimaker Cura
PREPARE
PREVIEW
MONITOR
Marketplace
Sign in
159
Ultimaker 2+
Object list
Save to File

View type Layer view

Color scheme Material Color

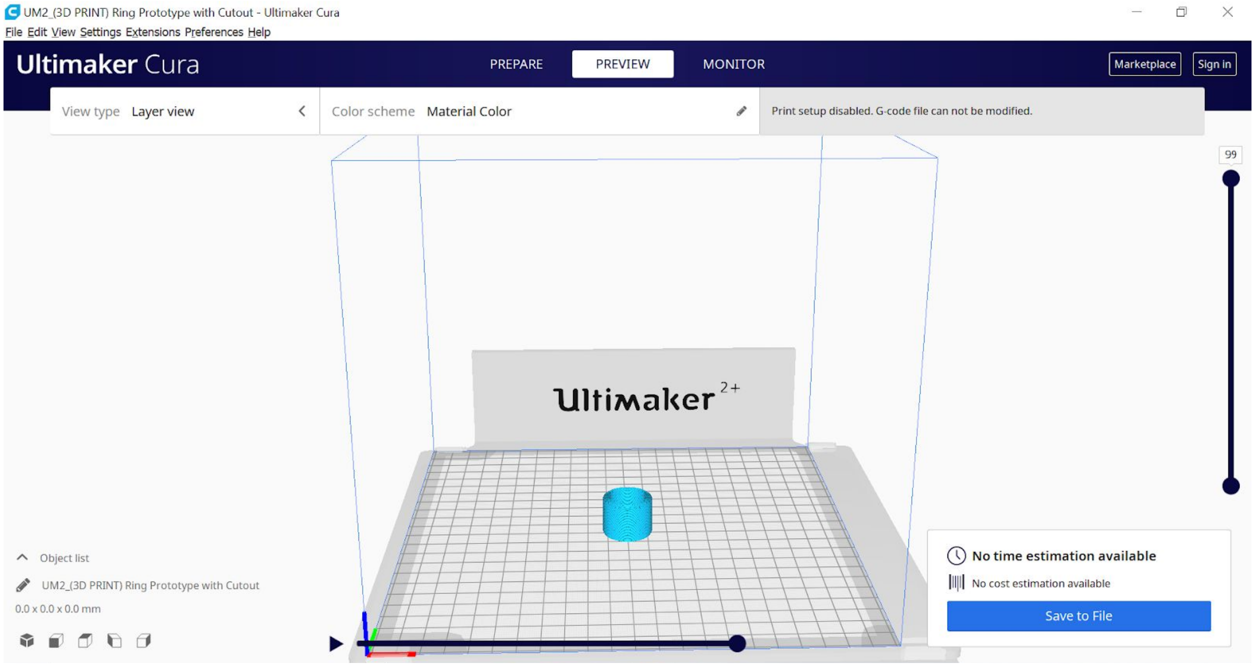
Print setup disabled. G-code file can not be modified.

UM2_(3D PRINT) Watch Componenet Housing

0.0 x 0.0 x 0.0 mm

No time estimation available

No cost estimation available


UM2_(3D PRINT) Ring Prototype with Cutout - Ultimaker Cura
File Edit View Settings Extensions Preferences Help
Ultimaker Cura
PREPARE
PREVIEW
MONITOR
Marketplace
Sign in
99
Ultimaker 2+
Object list
Save to File

View type Layer view

Color scheme Material Color

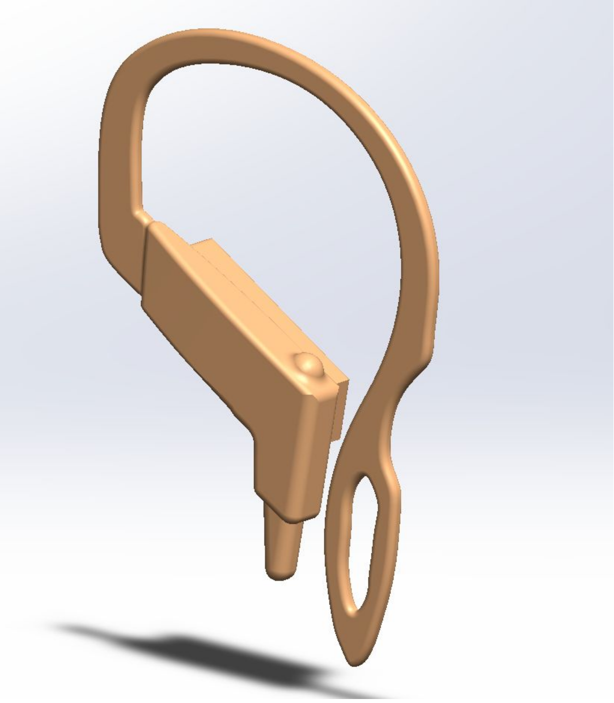
Print setup disabled. G-code file can not be modified.

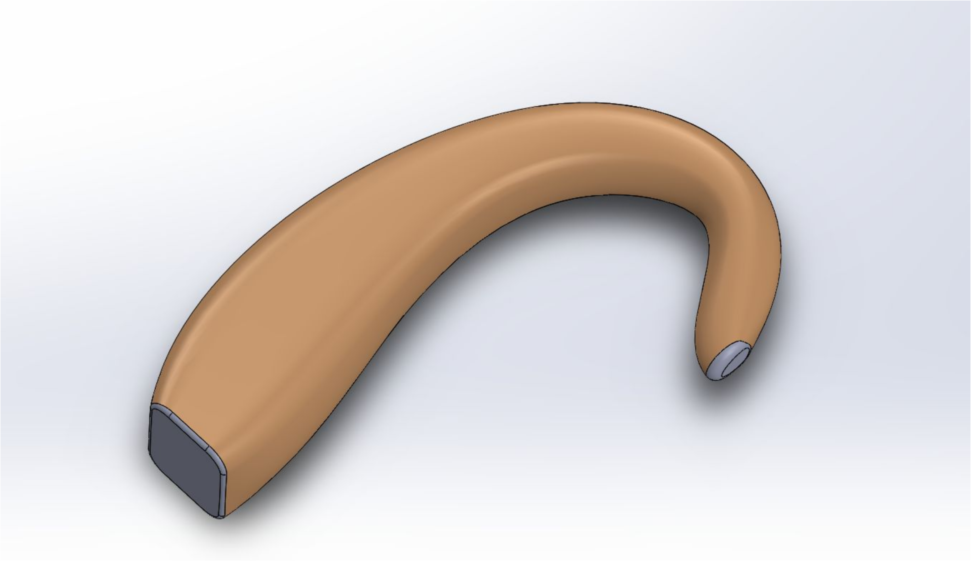
UM2_(3D PRINT) Ring Prototype with Cutout

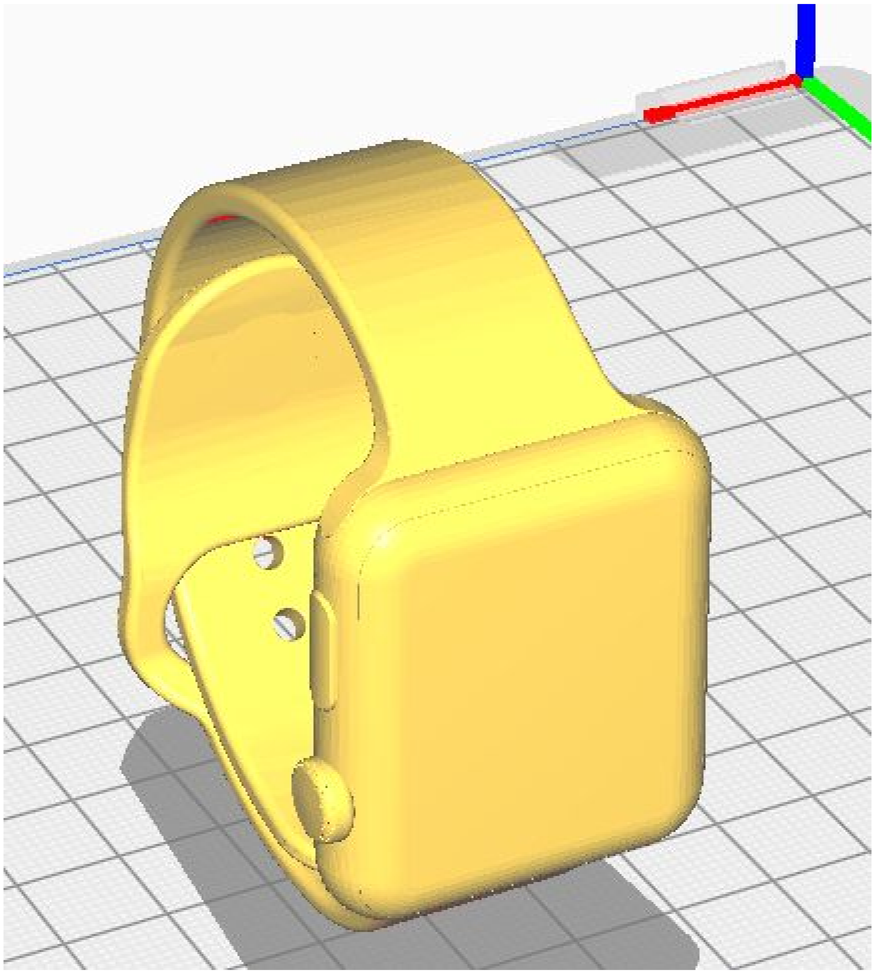
0.0 x 0.0 x 0.0 mm

No time estimation available

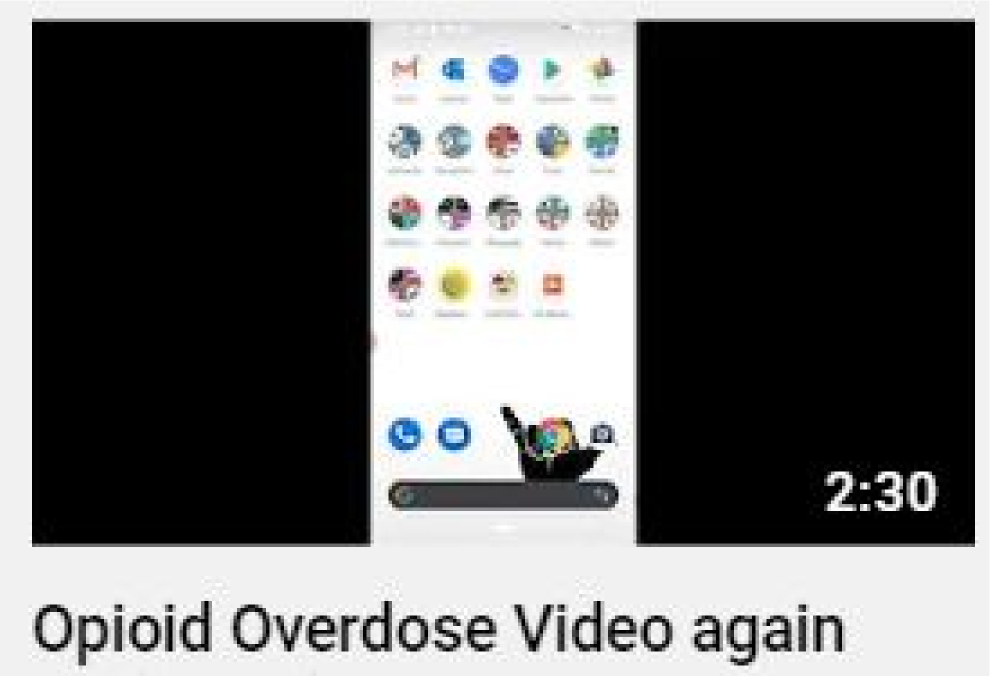
No cost estimation available







Appendix III: Working Video Demonstration



Link: https://www.youtube.com/watch?v=0pFZTh7h1uI