CardiO2 - Final Report

A Report
Presented to
The Department of Electrical & Computer Engineering
Concordia University

In Partial Fulfillment
of the Requirements
Of ELEC/COEN 390

By
Team #6

Ali Turkman	40111059
Ian Stellema	40066301
Jake Dos Santos	40018271
Roger Mugisho Namukama	40070820
Talon Saintsbury	40031122

Professor: Dr. Wahab Hamou-Lhadj

Concordia University
December 4th 2021

Table of Contents:

1. Introduction

The CardiO$_2$, is a wristband that monitors accurate physical health information using many sensors to determine heart rate and blood oxygen levels thanks to the MAX3010x sensor.

The wristband alongside a companion app would allow for specialists to monitor the user's information and assist in health recommendations. The app shoulds the heart rate data in a graph or table form, and is used to monitor either stress during studying or heart rate data during physical activities.

The device is intended for students and adults who need to monitor stress level through heart rate data. The goal is to make people aware of their heart rate information to help them make important decisions regarding their health.

2. Mission Statement

2.1 Product Description

The purpose of the application is to retrieve sensor data through Bluetooth and then upload it to a Firebase database through the Internet. The main device is a sensor that is connected to an Arduino circuit powered by a rechargeable battery, and its purpose is to measure the heart rate and blood oxygen levels. With the data uploaded to the firebase, the app can plot said data on a graph.

2.2 Target Market

Our product caters to 3 types of customers. These are Athletes, Fitness enthusiasts and Health specialists. These three customers have a common desire to monitor cardiovascular data and record this information for review for their own specific reasons. Given the fact that over 20% of Americans use a fitness tracker according to the Pew Research center[1], there is clearly a strong market demand for this type of product.

2.3 Benefit Proposition

Most fitness trackers on the market can only measure BPM data without the blood oxygen monitoring functionality. Our device does both while maintaining portability, simplicity, and cloud functionality. Our app is focused on being easy to use and intuitive, and viewing the cardio information during a session is easily done with a graph.

Monitoring health metrics such as heart rate and blood oxygen levels is extremely important as it can provide an overview of an individual’s cardiovascular health. For example, an individual with a blood oxygen level of around 90% at rest is likely suffering from a disease or

sleep apnea, as the normal level is usually around and above 95%. Meanwhile, having a high heart rate at rest, meaning over 90 BPM, could also indicate a cardiovascular disease or extremely poor health.

2.4 Key business goals

The wristband will be made as cheap as possible, so that a subscription service later on can subsidize the cost of the wristband itself. There will be a paid and subscription-based version of the app. The business model of selling hardware for as near cost as possible while later making a profit through software subscriptions is becoming increasingly popular.

The wristbands on average will cost (estimate) around $30 (which includes manufacturing, distribution, and all other costs), this price will include the heart rate sensor, the wristband, the microcontroller, and the battery.

We will sell the wristband itself for $40.00, which will yield a $10.00 profit per unit. Our team will work a combined 300 hours on the project (both the app and the hardware), at a salary cost $35/h, totalling at $10,500 for labor costs. We would need to sell 1050 units to cover the labor cost, as that would generate us $10,500 in profit. 1050 units overall would cost us $42,000 in material cost (including distribution and other costs). Therefore the total cost, if our aim is to produce 1050 units, would be $52,500. All prices in CAD.

2.5 Assumptions & Constraints

2.5.1 Assumption:

1. We will be able to construct a band that is aesthetically pleasing and is comfortable to wear.
2. We will find a small bluetooth transmitter with good connectivity
3. Users will be willing to wear a device and open an app while working.

2.5.2 Constraints:

1. Product software must be compatible with Android
2. Product must be compatible with bluetooth
3. Wristbands must be small and lightweight.
4. Product must be completed by November 29th.

2.6 Stakeholders

2.6.1 Stakeholders of the product:

1. Manufacturers (Wristband, sensor suppliers)
2. Health Conscious Users
3. Google Playstore
4. Competition (Smart watch-manufacturers: Apple, samsung, Huawei)
5. Storefronts (Online availability only, or Storefronts)
6. Possibly advertisers and other affiliated companies

2.6.2 Stakeholders interviewed:

1. David - Interviewed October 4 2021, 15:00

   David, a sports enthusiast who does cardio on a daily basis and notes down his heartbeat (through his neck pulse) before and after each exercise session, was interviewed to learn more about how our product and service would be useful to him, and what additional features that we can add to improve our product and service.

2. Joe - Interviewed October 6 2021, 13:00

   A physiotherapist specializing in rehabilitating patients from long-term surgical recovery. After intensive surgery some patients need to remain off their feet for an extended period of time leading to muscle deterioration and reduced cardio health. Joe can use our product when rehabilitating these patients to help him monitor his patient's recovery.

3. Rebecca - Interviewed October 8 2021, 11:00

   Rebecca is a sales worker at an athletic apparel store who is interested in the health benefits associated with deep breath meditation and practicing yoga. In our interview she mentioned how our product would serve the useful function of keeping track of her calmness throughout her daily yoga routines. This would help her clearly see any progress or noticeable hiccups in her breathing form on both a daily and weekly basis.

3. Product and Sprint Backlogs

3.1 Complete Product Backlog

Story ID	Story Title	Card	Story Points	Sprint	Status	Conversation	Confirmation
UI-1	Simple User Interface	*As a typical user, I want to use the application and navigate easily through activities and data summaries*	5	Sprint 1	Completed	User should be given priority information first (current heart-rate), and then have clear UI paths to access anything else on the app.	*1. Make sure the app can be opened* *2. Menus can be navigated*
COM-1	Establish bluetooth connection	*As a typical user I want the sensor and control board to connect via bluetooth to the phone to display sensor data*	3	Sprint 1	Completed	The user should have a wireless connection between the wristband and the app.	*1. The app can communicate with the device via bluetooth*
COM-2	Easy bluetooth	*As a typical user I want to be able to connect to the device easily*	5	Sprint 3	Completed	The user should have a connect button in the app	*1. The app has a connect button for bluetooth that works*
ADV-1	Read heart-rate	As a typical user I want the app to display accurate sensor data	8	Sprint 1	Completed	The app will display the data acquired from the heart rate sensor through Bluetooth	*1. Heart-rate data is being transmitted to and received by the app*
ADV-2	Battery powered	*As a user, I want to have a portable device instead of a wired one*	5	Sprint 3	Completed	User should be able to use a portable device instead of relying on an external power source	*1. Can the device run on a battery and not any other power source*

ADV-3	Blood-Oxygen measuring	*As a user, I want to be able to see my blood-oxygen levels*	5	Sprint 2	Completed	User should have access to records data of Blood Oxygen	*1. Blood-Oxygen data is being transmitted to and received by the app*
ADV-4	Data precision	*As a user, I want the device to provide accurate information*	3	Sprint 3	Completed	User should not see large spikes in data caused by software problems	*1. Data shown to user is stable*
ADV-5	Login	*As a user, i want to be able to setup my own account*	3	Sprint 3	Completed	Each user should use their own account to login and access data	*1. User is presented with a login that is used.*
UI-2	Data Presentation	*As a user, I want the data to be displayed in an easy to comprehend graph*	5	Sprint 2	Completed	The app will present data in either graph mode	*1. Graph data is visible and accurate on the app (past 10 minutes, hour, day...)*
UI-3	Personalization	*As a typical user, I want to customize the app appearance*	3	Sprint 3	Completed	The app has to have some cosmetic customization	*1. The user can change their profile picture.*
BK-1	Data Storage	As a typical user, I want to store data in a reliable long-term database	13	Sprint 1	Completed	The app will require the users input for when to begin data measurement, with long term records and the potential option for exportability	*1. Data is being tracked and temporarily stored on the device* *2. Data is stored in a Firebase database*
DES-1	Create wristband	As a typical user, I want the wristband to be comfortable without limiting movement	8	Sprint 3	Completed	Wristband should occupy minimal space and be able to be worn like a simple comfortable bracelet	*1. Wristband can be worn without falling off or being too cumbersome* *2. Wristband is able to hold all components efficiently*

3.2 Sprint 1 Product Backlog

Sprint 1 Goal(s): Do simulation of app, establish Bluetooth connection with sensor, display heart-rate data, create database to store records					
Story ID	Task ID	Task Title	Task Description	Ideal Hours	Status
UI-1	UI-1-1	Format UI	Create a repo for developers, and then setup colour, text size, base layout of project.	8	Completed
	UI-1-2	Format Display activity page	Creating a layout for the user to see the data and interact with.	6	Completed
UI-2	UI-2-1	Display Data	Generating required activities to and material theme to show content	6	Completed
DES-1	DES-1-1	Get sensor and board	Borrow sensor and arduino from school	2	Completed
COM-1	COM-1-1	Arduino programming for bluetooth	Run example codes and find low power data transmission to android devices or serial out for testing purposes.	10	Completed
	COM-1-2	Android studio bluetooth connection	Create activities to handle Bluetooth permission and connection to app	10	Completed
BK-1	BK-1-1	Make Database	Create database to store heart rate sensor data	20	Completed
	BK-1-2	Get dummy data	Design array for parsing pseudo data to activities to be displayed.	4	Completed
ADV-1	ADV-1-1	Read heart-rate on arduino	Read raw heart-rate data from sensor	8	Completed
	ADV-1-2	Read heart-rate on app	Show the heart-rate data in the app	15	Completed
DOC-1	DOC-1-1	Documentation	Document and note down all processes and tests	4	Completed
TEST-1	TEST-1-1	Testing	Verifying the functionality of all app features and hardware processes that are implemented	8	Completed
PLAN-1	PLAN-1-1	Sprint Planning	Write down and planify tasks of Sprint 1	5	Completed

3.2 Sprint 2 Product Backlog

Sprint 2 Goal(s): Do simulation of app, display heart-rate and blood oxygen data, read and write to database, get the device working off battery power					
Story ID	Task ID	Task Title	Task Description	Ideal Hours	Status
ADV-1	ADV-1-1	Read heart-rate on arduino	Read raw heart-rate data from sensor	4	Completed
	ADV-1-2	Read heart-rate on app	Show the heart-rate data in the app	5	Completed
	ADV-1-3	Troubleshoot Sensor	Sensor breakout board is non-functional	20	Completed
UI-1	UI-1-1	Format UI	Setup colour, text size, base layout of project.	5	Completed
	UI-1-2	Format Display activity page for heart rate	Creating layout for the user to see the data and interact with heart rate logs	4	Completed
	UI-1-3	Format Display activity page for blood oxygen	Creating layout for the user to see the data and interact with blood oxygen levels	4	Completed
BK-1	BK-1-3	Read from Database	Create database to store heart rate sensor data	15	Completed
	BK-1-4	Read Heart-rate from database	Automate storing and reading of sensor data	12	Completed
	BK-1-2	Get dummy data	Design array for parsing pseudo data to activities to be displayed.	4	Completed
ADV-3	ADV-3-1	Read blood oxygen on arduino	Read raw blood oxygen data from sensor	8	Completed
	ADV-3-2	Read blood oxygen on app	Show the blood oxygen data in the app	6	Completed
ADV-2	ADV-2-1	Design battery circuit	Research battery options and decide on a design	4	Completed
	ADV-2-2	Build battery circuit	Build an on-device battery circuit	5	Completed
DOC-1	DOC-1-2	Documentation	Document and note down all processes and tests	4	Completed
TEST-1	TEST-1-2	Testing hardware	Verifying the functionality of hardware processes that are implemented	4	Completed
	TEST-1-3	Testing software	Verify the functionality of app features	5	Completed
PLAN-1	PLAN-1-2	Sprint Meeting	Sprint 2 meeting with scrum master	5	Completed

3.4 Sprint 3 Product Backlog

Sprint 3 Goal(s): Display blood oxygen data, finish settings page and data display, get the device working off battery power and assembled					
Story ID	Task ID	Task Title	Task Description	Ideal Hours	Status
ADV-4	ADV-4-1	Optimize heart-rate transmission	Add communication protocol to provide clean data. Sometimes results are garbled and will crossover in the app.	5	Completed
	ADV-4-2	Test our heart-rate optimization	Test the solution found in ADV-4-1	3	Completed
COM-2	COM-2-1	Bluetooth on/off feature	Allow the app to turn on or off Bluetooth conveniently from a button in the action bar	5	Completed
ADV-3	ADV-3-1	Read blood oxygen on arduino	Read raw blood oxygen data from sensor	8	Completed
	ADV-3-2	Read blood oxygen on app	Show the blood oxygen data in the app	6	Completed
ADV-2	ADV-2-1	Design battery circuit	Research battery options and decide on a design	5	Completed
	ADV-2-2	Build battery circuit	Build an on-device battery circuit	5	Completed
UI-1	UI-1-2	Format Display activity page for heart rate	Creating layout for the user to see the data and interact with heart rate logs	6	Completed
	UI-1-3	Display heart-rate data from database in graph	Allow user to see heart-rate data in a graph	6	Completed
	UI-1-4	Format Display activity page for blood oxygen	Creating layout for the user to see the data and interact with blood oxygen levels	4	Completed
	UI-1-5	Display blood oxygen data from database in graph	Allow user to see blood oxygen data in a graph	4	Completed
ADV-5	ADV-5-1	Add Login Feature	Allow user to login with an email and password	5	Completed
UI-3	UI-3-1	Add Profile Picture Feature	Allow user to upload an image that appears on Main Activity as profile picture	6	Completed
	UI-3-2	Add Splash Screen	Add Splash screen of logo that appears on app start for a few seconds while the app loads	5	Completed

DES-1	DES-1-1	Create wristband design	Create design for the wristband including the location of all electronics	5	Completed
	DES-1-2	Assemble product	Assemble the presentation version of the product	5	Completed
DOC-1	DOC-1-2	Documentation	Document and note down all processes and tests	5	Completed
TEST-1	TEST-1-4	Testing hardware	Verifying the functionality of hardware processes that are implemented	5	Completed
	TEST-1-5	Testing software	Verify the functionality of app features	5	Completed
PLAN-1	PLAN-1-3	Sprint Meeting	Sprint 3 meeting with scrum master	5	Completed

4. Design Document

4.1 Abstract

Most fitness trackers on the market can only measure BPM data without the blood oxygen monitoring functionality. Our device does both while maintaining portability, simplicity, and cloud functionality. Our app is focused on being easy to use and intuitive, and viewing the cardio information during a session is easily done with a graph. We took a simplistic approach because stakeholder feedback was really focused on the real-time information first, and then optional charts secondary. Monitoring health metrics such as heart rate and blood oxygen levels is extremely important as it can provide an overview of an individual's cardiovascular disease.

For example, an individual with a blood oxygen level of around 90% at rest is likely suffering from a disease or sleep apnea, as the normal level is usually around and above 95%. Meanwhile, having a high heart rate at rest, meaning over 90 BPM, could also indicate a cardiovascular disease or extremely poor health. Seeing as these metrics affect virtually all adults, we strongly believe most people could benefit from having a device like ours at home for daily use.

Cardiovascular diseases often go unnoticed as they're generally symptomless, and by the time an adult can notice them they can be harmful, meaning that being aware of health issues before they can develop into diseases is extremely important for long term longevity.

4.2 Introduction

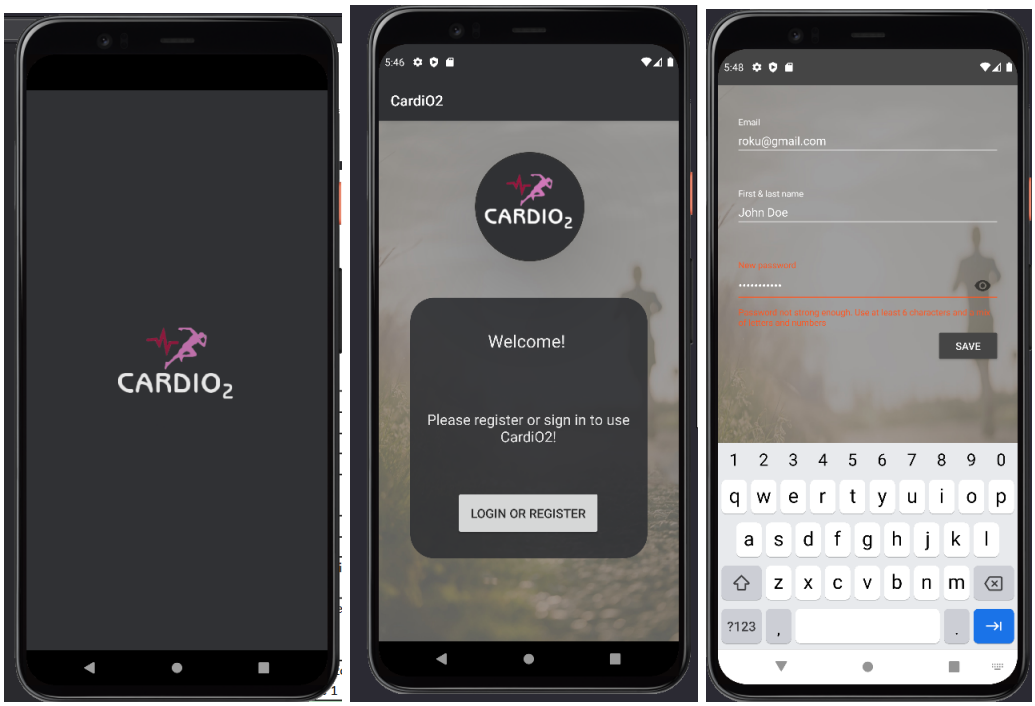
The primary purpose of CardiO2 is to retrieve sensor data and upload it to a database from which the data can then be plotted into a graph form in the application. The way this works is that while the MAX30102/05 sensor is turned on, the app constantly shows and updates live onto the screen the values calculated from the sensor, even without an internet connection. However this way nothing will be saved to Firebase. One of the benefits of our project is that the saved data is hosted on Google's very own Firebase, meaning that it is safe and secure with little risk of user data being leaked out or hacked.

The user can monitor both their heart rate data and blood oxygen levels simultaneously which is also useful compared to other health trackers that can only do heart rate tracking. After being saved to firebase once the toggle button is pressed again, any of the data sets saved until this point, that are all sorted and named by date/time, can be selected and shown instantly on a graph by using the data graph activity.

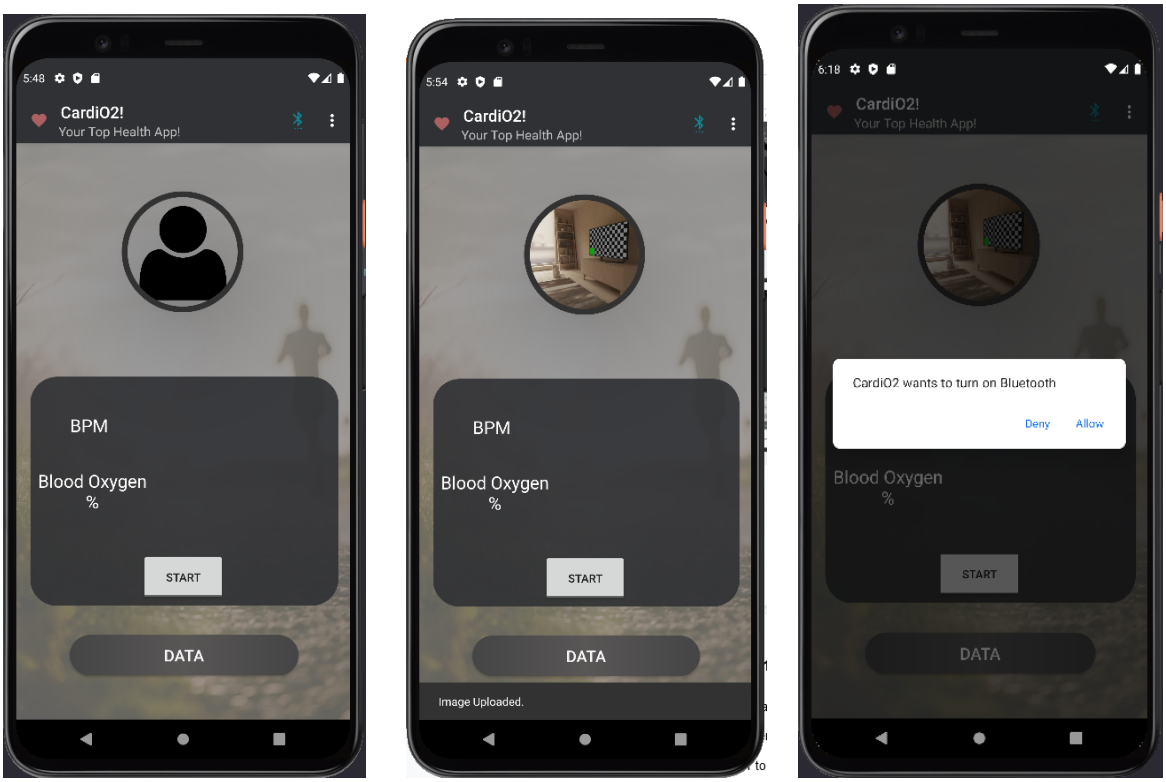
This line graph displays both the user's bpm in red, and blood oxygen% in green, all scaled automatically, and a list underneath the graph allows the switching between any of the other saved datasets. This allows the user to go back and see the difference in progress between different sessions with exact date and times naming the session to keep it clear for the user.

4.3 Design

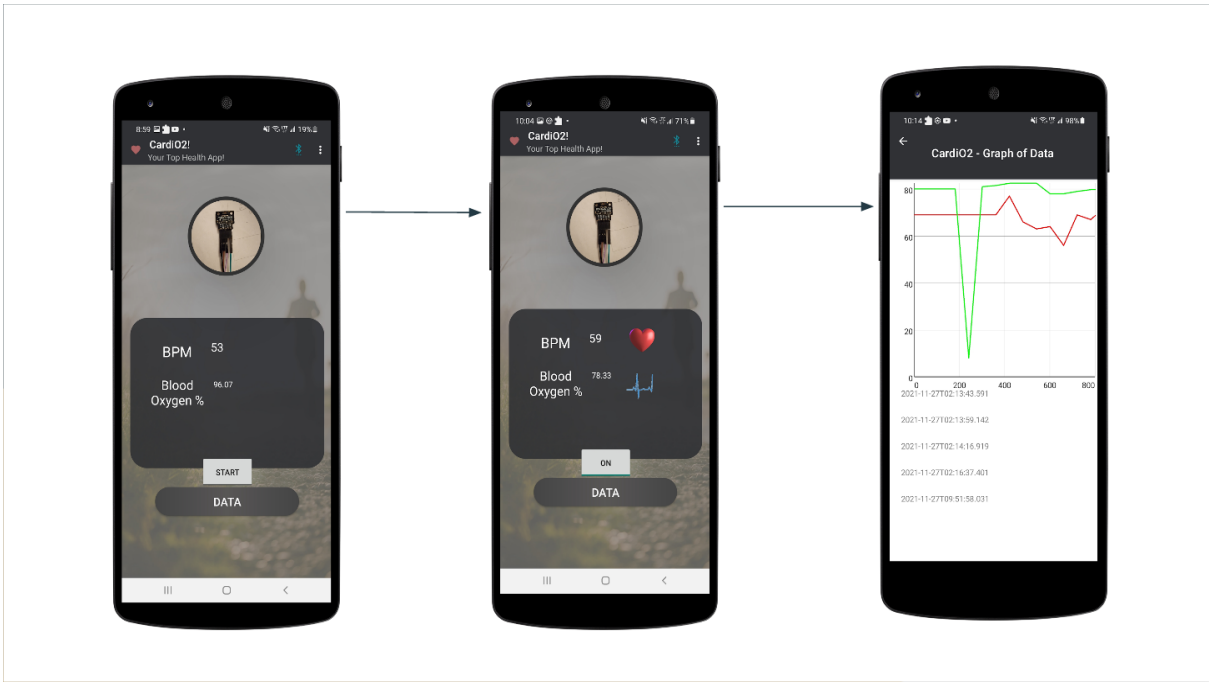
4.3.1 Android Application Wireframes



As shown above, when the app is first launched, the user will first see a splash screen that displays the app's logo for a second before fading out, which is very common in new and developed apps as it gives time for the rest of the application to load while displaying the splash screen. Following that, the user is then greeted by a login screen on the first usage, giving the user the option to register or login depending on whether they've signed up previously. Assuming that this is the user's first experience with the app, they will be directed to the registration page where they will input their email, name and password, needing only the email and password to log in following that. This information keeps the user logged in until they delete all app data or uninstall it. By doing so, the user will be greeted by the login page again after starting the app, but inputting the registered email after pressing the login button will allow the user to login without registering, simply directing them to the password page. Following that, the user is greeted to the main home page of the app.



Following the login process, the user will then be greeted by the homepage (the main activity), at which the user will be able to choose a profile picture by pressing the profile picture. After pressing the profile picture, the user will then be redirected to their phone's gallery or storage, and from it they will be able to choose a profile picture for CardiO2. After choosing the picture, it will then be uploaded to a Firebase database where it will be stored. It will take around a second for the image to upload, and once it does a snackbar appears letting the user know the picture has been uploaded. Furthermore, as can be noticed, there is a Bluetooth button that can be pressed to turn Bluetooth on or off. If Bluetooth is already on and the button is pressed, then the Bluetooth is turned off immediately. If Bluetooth is off and the user presses the button, then the app will ask permission to turn the Bluetooth on and as shown in the third picture above. Near the Bluetooth button is the action bar menu in which there is a menu item titled Log Out, which is used to log the user out and back to the registration page which appears after the splash screen.



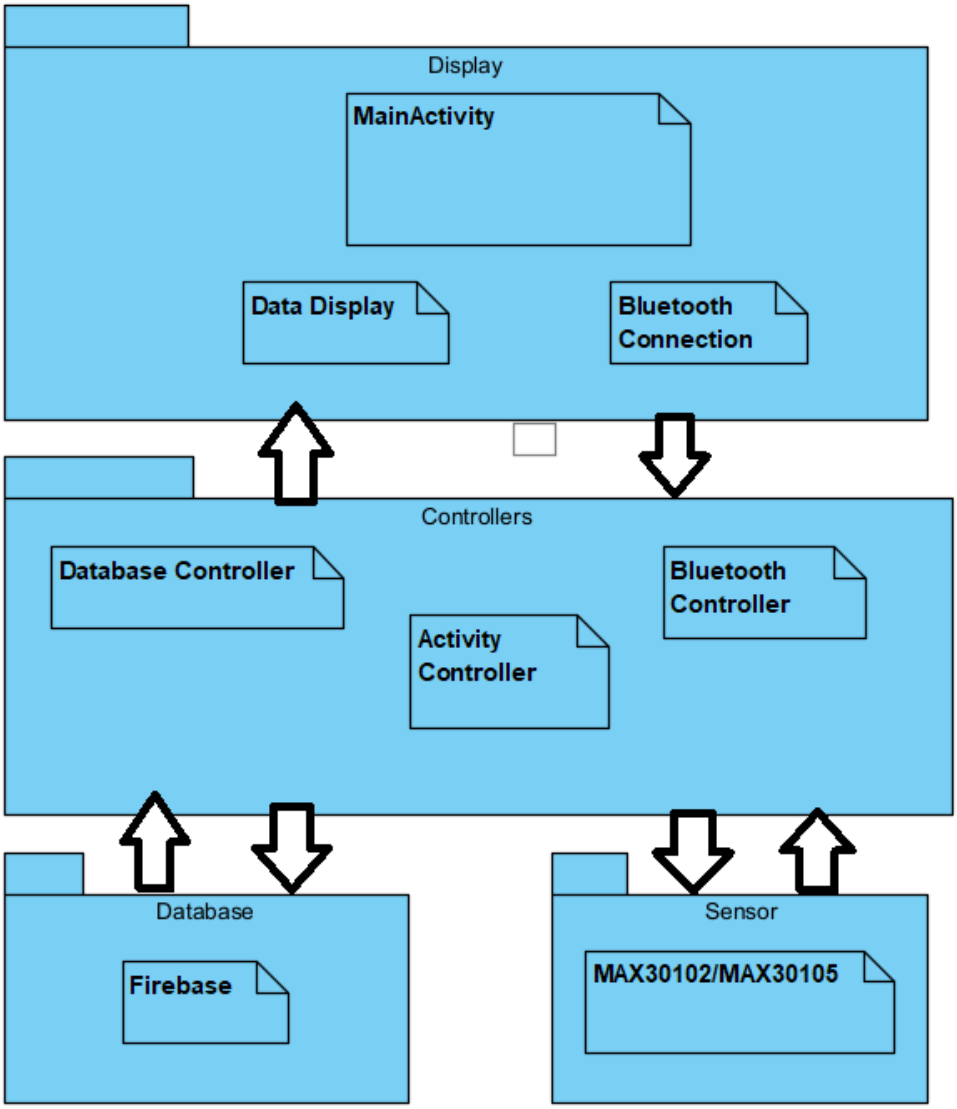
With everything setup, including the connection to Bluetooth through the phone's BT settings, the user can now use the application for its intended purpose. As soon as the user properly sets up the device, has the sensor attached to the finger, and turns on the sensor, the app's homepage will show the heart rate figure as shown above in the first image to the left.

Pressing the START button will begin the data monitoring process which uploads the data on screen to the Firebase database.

Following this, the user will be able to press the DATA button to open the Graph of Data activity, which showcases the user's data in graph form. A list below is interactive and by pressing any of the dates it shows the previously saved data in graph mode.

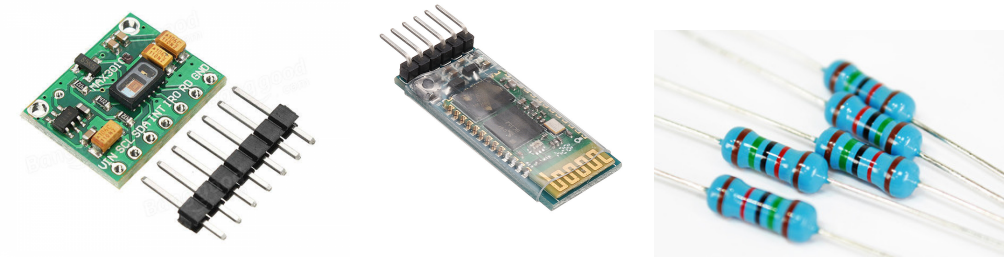
4.3.2 System Architecture

As we're using Android Studio to develop our app, our application is therefore set up with Activities and Fragments architecture with the Java language. The application uses a countdown timer that is turned on by the user via toggle button to determine the length of time to be saving measurements to the Firebase cloud server. Measurements are saved at a set rate periodically for the entire duration of the count down timer, or if the toggle button is turned back off. Once the set is saved to Firebase, it can be turned into graph representation and observed by the user.



4.3.3 Hardware Architecture

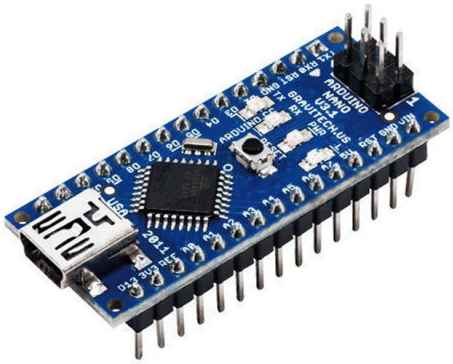
The project application will make use of a device with a single sensor, the MAX30102 heart rate and SpO2 sensor. This sensor is connected to an Arduino Nano board, which will then transmit heart rate and SpO2 sensor data to our mobile application through an HC-05 Bluetooth module chip. The hardware will be powered by a 5 volt rechargeable battery bank.



The MAX30105 sensor, the HC-05 Bluetooth module and resistors for voltage divider

For the purposes of testing and demonstrating the product, a soldered breadboard has been used to minimize the size of the hardware. This breadboard fits the arduino nano, the bluetooth board, and terminal connections to connect the sensor, which has been tested with different lengths of cable to determine what would be appropriate for a wrist-mounted device. Further improvements for production would include a printed circuit board, which would reduce the size of the device significantly.

The Arduino Nano uses a built-in power regulator, and also provides power lines for other devices to be powered from the source as the Nano. This allows us to connect an external 5 volt battery bank to the arduino, and it will power all the Arduino, the bluetooth module, and the MAX30102 sensor.



(Pictured above: the Arduino nano and the 5 volt battery bank)

Powering the device is a 2500 mAh rechargeable battery with a usb output to connect to the Arduino and a microusb output and a microusb output to recharge the battery. Considering its large capacity and the minimal power load of the Arduino board and sensor, the battery is able to keep the sensor turned on for many hours with a full charge before needing to be recharged again.

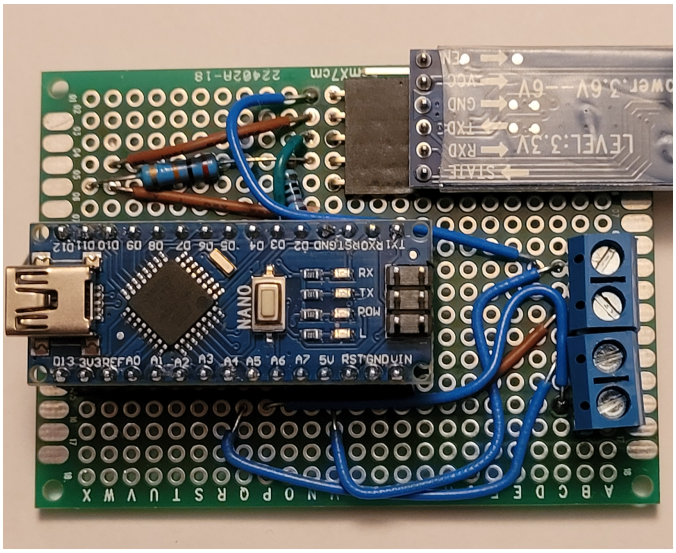


Image: The soldered breadboard with the bluetooth module and Arduino connected to the quick-connect terminals.

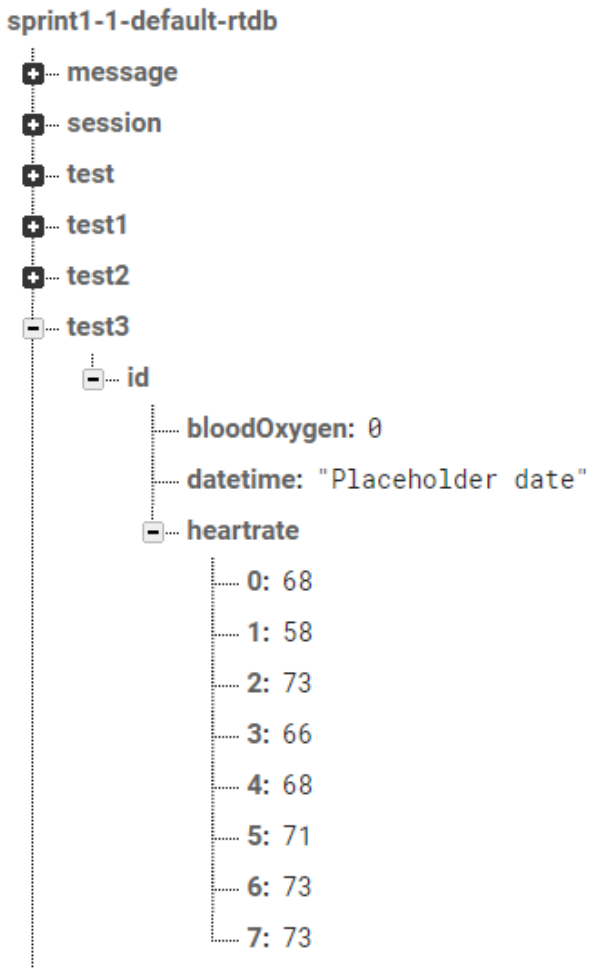
The sensor is connected to the soldered board using longer wires that are attached to the screw terminals (located on the right side of the above image) on the soldered board. The sensor will be held in place to the user's finger via a finger splint. The device was built on a PCB and not on a regular breadboard. The blue sponge attached to this sensor helps block out external light which interferes with sensor data and causes poor readings.

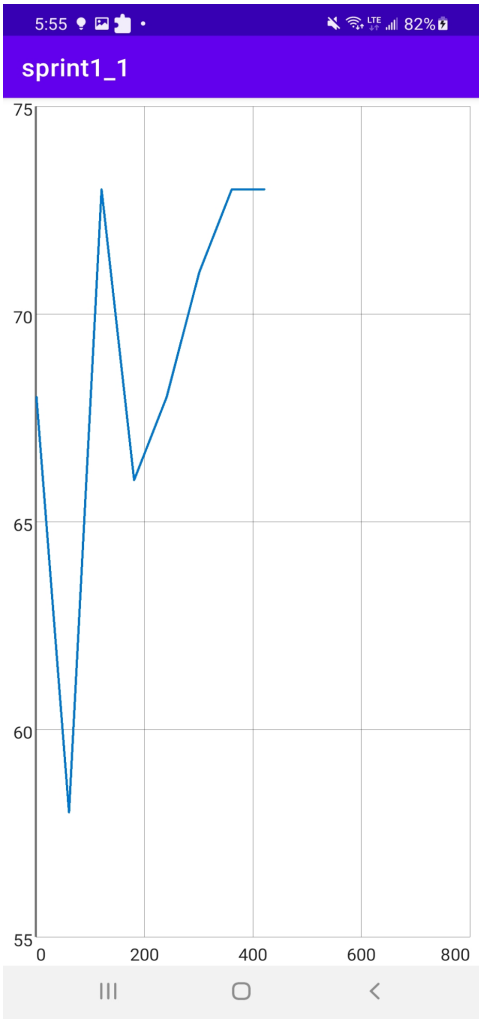
The assembled prototype, as shown in the demo

4.3.4 Software Architecture

In the CardiO2 application, the session class contains the heart rate, blood oxygen, and time information to track the user's information. This information will be filled with incoming data from the arduino once that is functional.

This data is then automatically stored in a cloud firebase database using the Database_Helper class. This data can later be retrieved by the database helper and displayed on a graph using the DataDisplay class. The DataDisplay class uses a GraphView class from another library to simplify the coding.



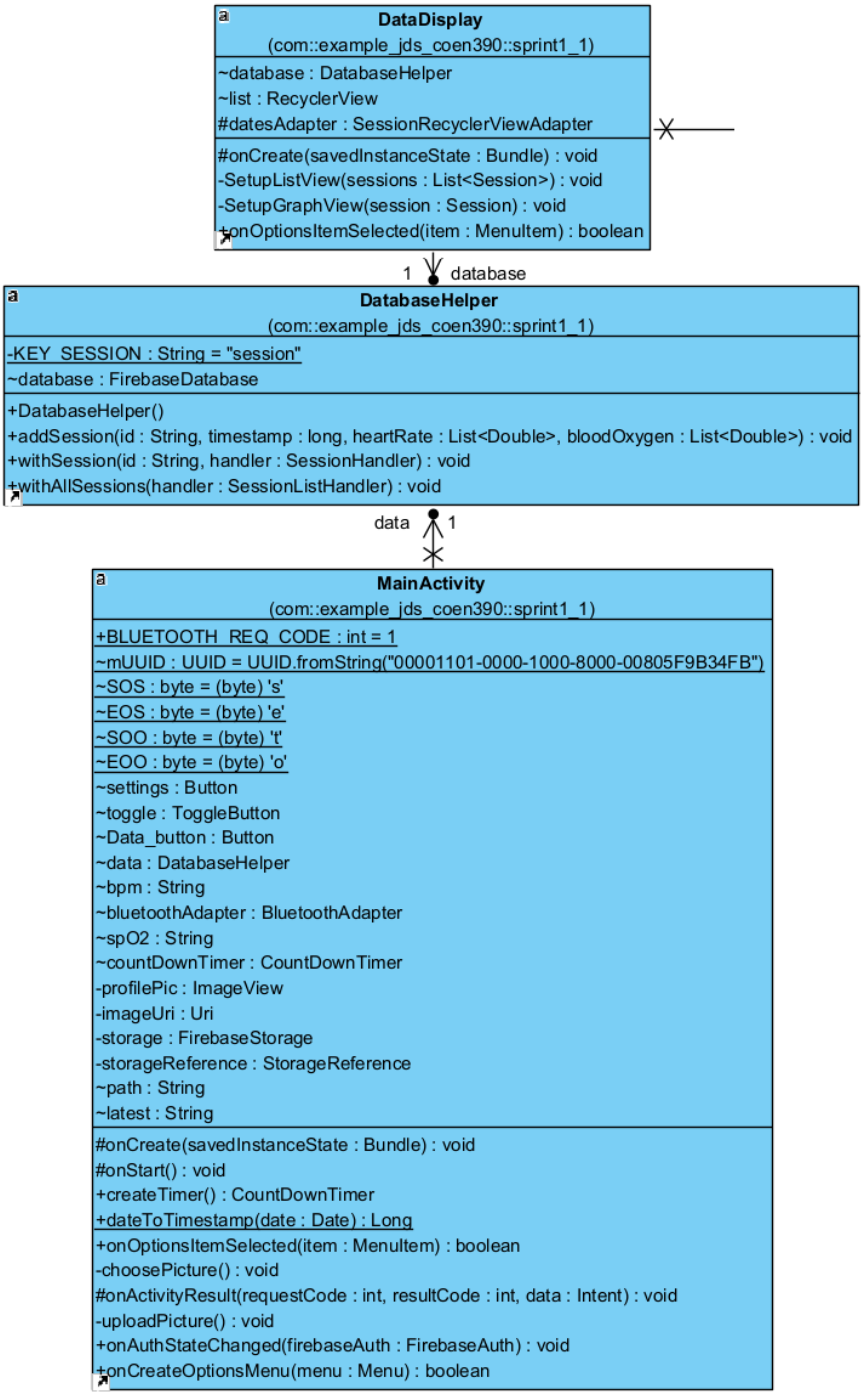


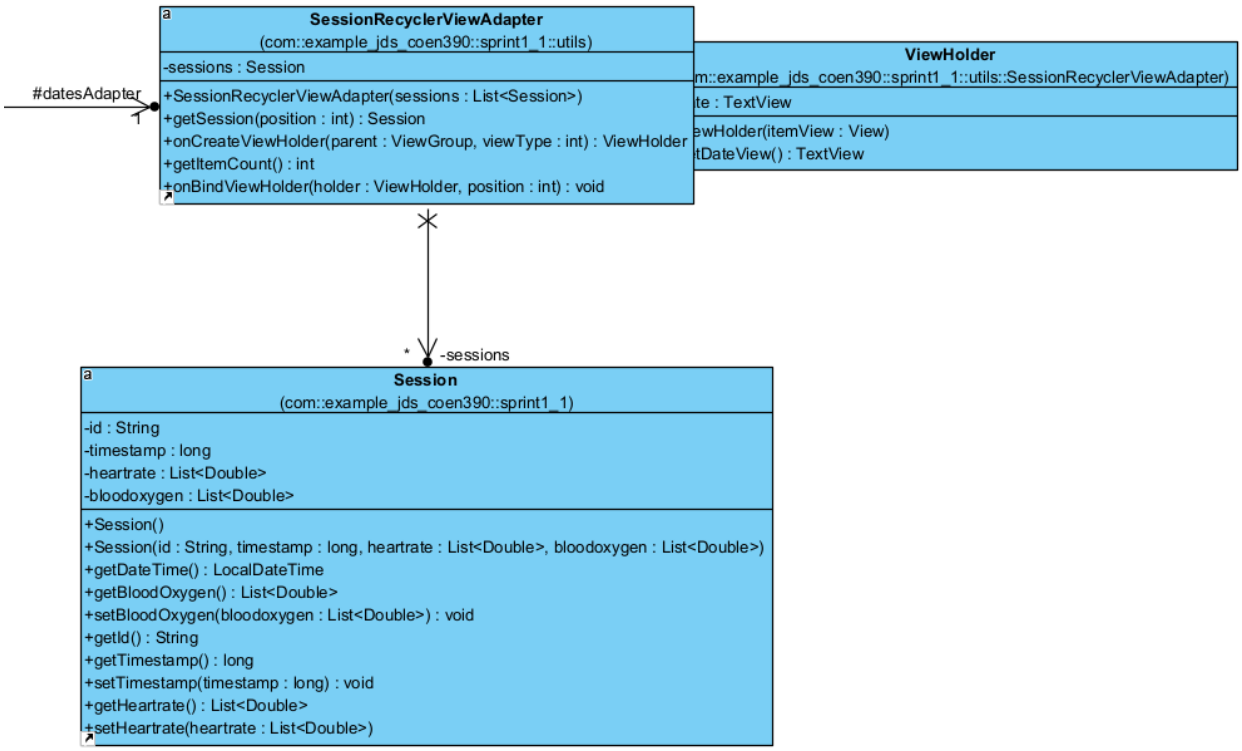
Class	Explanation	Relationship
MainActivity	This is where most of the app's features lie, allowing the user to begin monitoring heart rate and blood oxygen level data. This activity is the center of the application and	Receives data from arduino, has button leading to data page, uploads data to firebase through database helper and displays data received from arduino
LoginActivity	Page connects the user through firebaseUI using email address and first name. The user can also log off.	It is first page to open when run the app then connect the user to the main page
SplashActivity	Displays the app logo very briefly as the app loads and then fades out.	It connects to either the registration/login page or the Main Activity depending on whether the user is logged in or not.
DatabaseHelper	It sends and receives data from firebase to android	Called on by main activity to send data to firebase and called on by data activity to receive data from firebase
DataDisplay	It displays the graphs,the heartbeat and blood oxygen pressure	Calls on database helper to receive data from firebase
~~DatesRecyclerAdapter~~	Unused	
Session	Class for holding data from an individual session	Data is placed in a session post recording session and individual sessions are sent and received to/from firebase through database helper

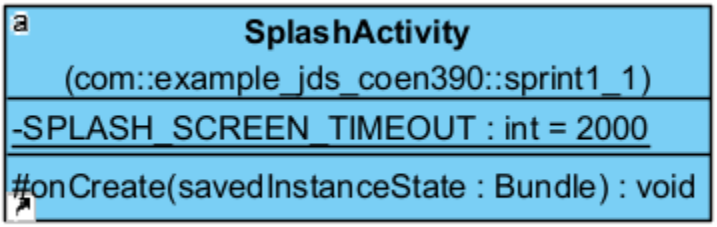
Package	Interface	Explanation	Relationship
Handlers	Session Handler	Handles a session when pulled from database	Used by database helper to handle data from firebase
	SessionListHandler	Handles a list of sessions when pulled from the database	Used by database helper to handle data from firebase

Package	Class	Explanation	Relationship
Utils	RecyclerItemClickListener	Makes list elements clickable	Retrieves session from firebase according to which date is clicked
	SessionRecyclerView Adapter	Adapts date list for recycler view	Binds the date list to the recycler view

Project Code UML

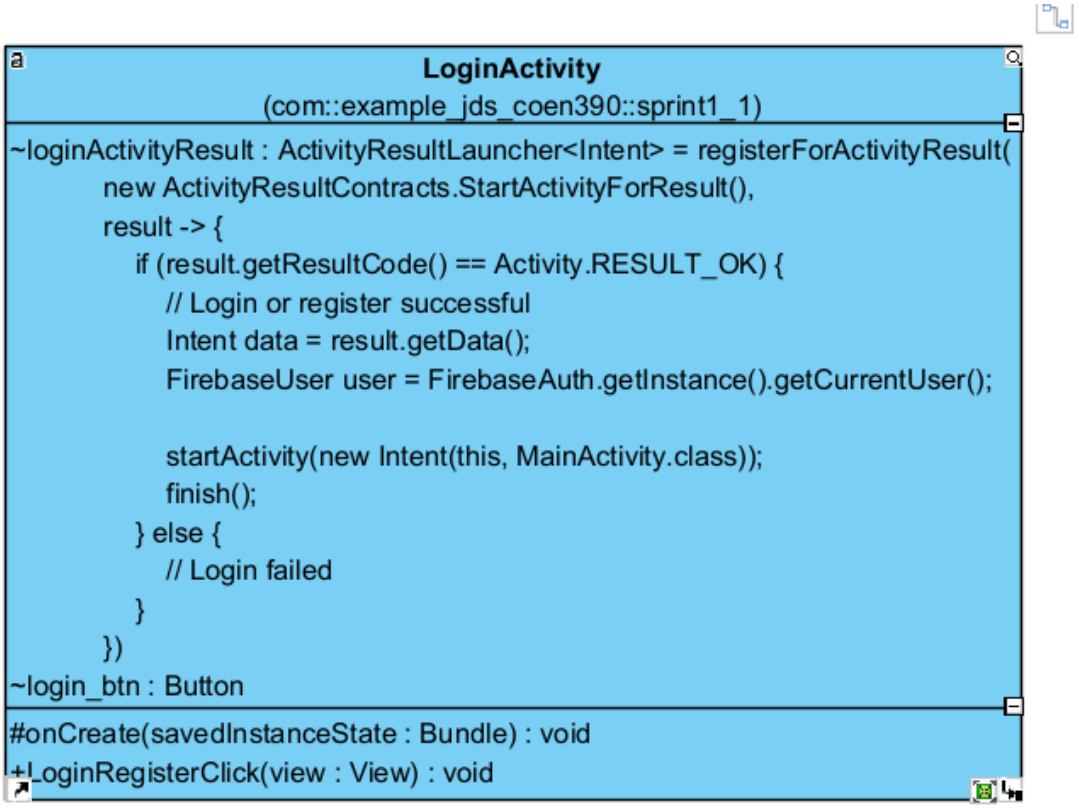


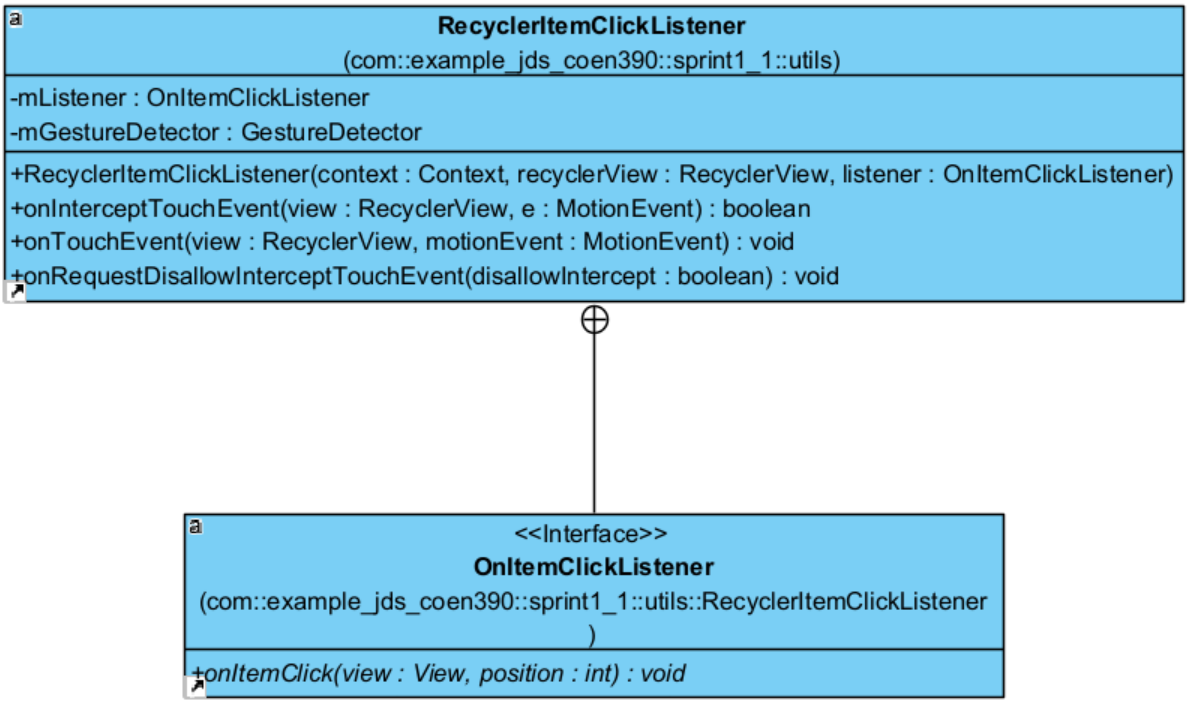

SessionRecyclerViewAdapter
(com::example_jds_coen390::sprint1_1::utils)
-sessions : Session
+SessionRecyclerViewAdapter(sessions : List<Session>)
+getSession(position : int) : Session
+onCreateViewHolder(parent : ViewGroup, viewType : int) : ViewHolder
+getItemCount() : int
+onBindViewHolder(holder : ViewHolder, position : int) : void
#datesAdapter
1
ViewHolder
m::example_jds_coen390::sprint1_1::utils::SessionRecyclerViewAdapter)
te : TextView
ewHolder(itemView : View)
tDateView() : TextView
*
-sessions
Session
(com::example_jds_coen390::sprint1_1)
-id : String
-timestamp : long
-heartrate : List<Double>
-bloodoxygen : List<Double>
+Session()
+Session(id : String, timestamp : long, heartrate : List<Double>, bloodoxygen : List<Double>)
+getDateTime() : LocalDateTime
+getBloodOxygen() : List<Double>
+setBloodOxygen(bloodoxygen : List<Double>) : void
+getId() : String
+getTimestamp() : long
+setTimestamp(timestamp : long) : void
+getHeartrate() : List<Double>
+setHeartrate(heartrate : List<Double>)

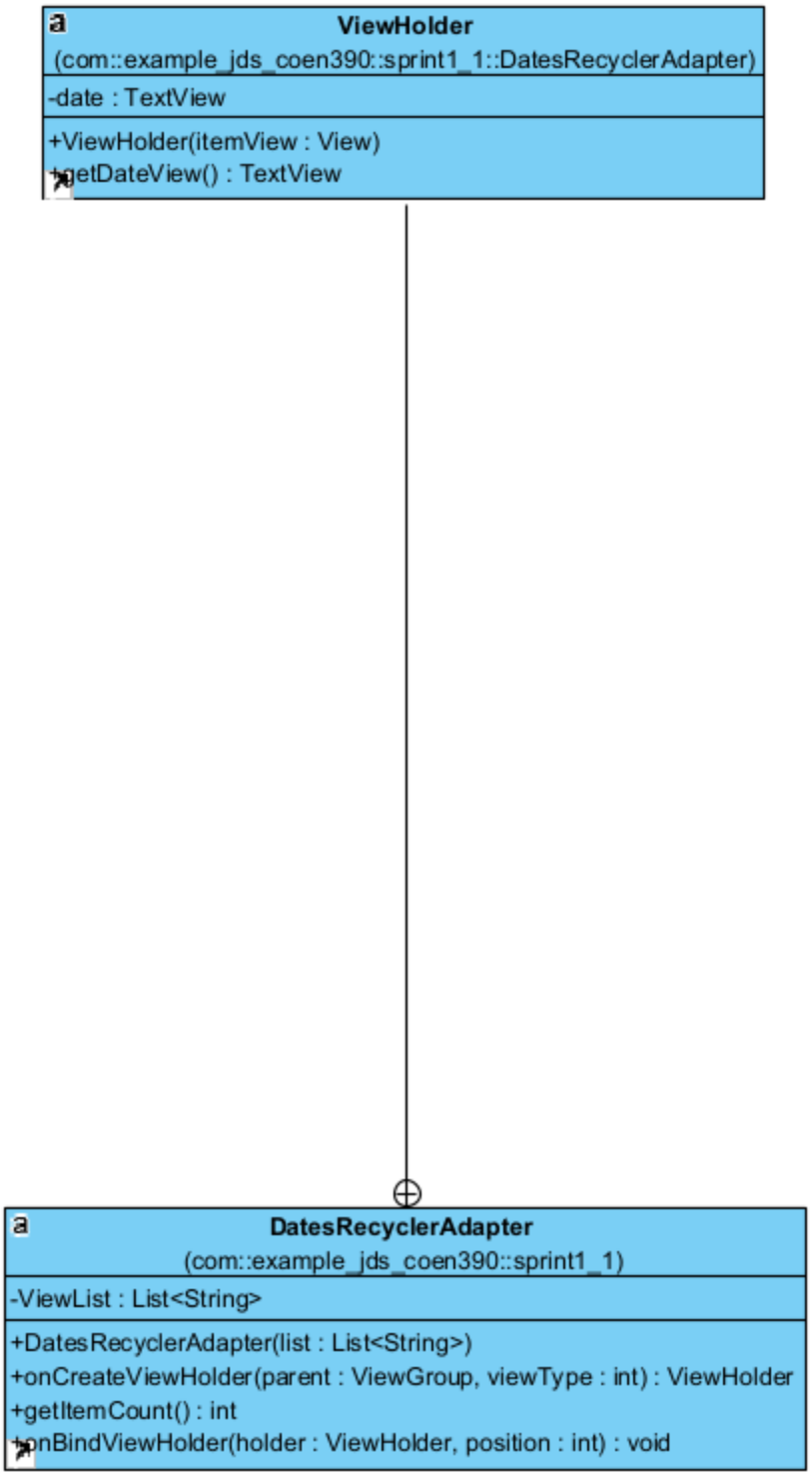

SplashActivity
(com::example_jds_coen390::sprint1_1)
-SPLASH_SCREEN_TIMEOUT : int = 2000
#onCreate(savedInstanceState : Bundle) : void


<<Interface>>
SessionHandler
(com::example_jds_coen390::sprint1_1::handlers)
+handle(session : Session) : void


<<Interface>>
SessionListHandler
(com::example_jds_coen390::sprint1_1::handlers)
+handle(sessions : List<Session>) : void


LoginActivity
(com::example_jds_coen390::sprint1_1)
~loginActivityResult : ActivityResultLauncher<Intent> = registerForActivityResult(
new ActivityResultContracts.StartActivityForResult(),
result -> {
if (result.getResultCode() == Activity.RESULT_OK) {
// Login or register successful
Intent data = result.getData();
FirebaseUser user = FirebaseAuth.getInstance().getCurrentUser();
startActivity(new Intent(this, MainActivity.class));
finish();
} else {
// Login failed
}
})
~login_btn : Button
#onCreate(savedInstanceState : Bundle) : void
+LoginRegisterClick(view : View) : void


RecyclerItemClickListener
(com::example_jds_coen390::sprint1_1::utils)
-mListener : OnItemClickListener
-mGestureDetector : GestureDetector
+RecyclerItemClickListener(context : Context, recyclerView : RecyclerView, listener : OnItemClickListener)
+onInterceptTouchEvent(view : RecyclerView, e : MotionEvent) : boolean
+onTouchEvent(view : RecyclerView, motionEvent : MotionEvent) : void
+onRequestDisallowInterceptTouchEvent(disallowIntercept : boolean) : void
<<Interface>>
OnItemClickListener
(com::example_jds_coen390::sprint1_1::utils::RecyclerItemClickListener
)
+onItemClick(view : View, position : int) : void


ViewHolder
(com::example_jds_coen390::sprint1_1::DatesRecyclerAdapter)
-date : TextView
+ViewHolder(itemView : View)
+getDateView() : TextView
DatesRecyclerAdapter
(com::example_jds_coen390::sprint1_1)
-ViewList : List<String>
+DatesRecyclerAdapter(list : List<String>)
+onCreateViewHolder(parent : ViewGroup, viewType : int) : ViewHolder
+getItemCount() : int
+onBindViewHolder(holder : ViewHolder, position : int) : void

5. Test Cases

Test Case UI1-1.1		
Pre-Condition: Application launched		
Steps:	Expected Results	Actual Result
1. Launch application	Home screen loads without crashing	As Expected
Result: Pass		
Test Case UI 1-1.2		
Pre-Condition: Settings page loads		
Steps:	Expected Results	Actual Result
1. Click settings button 2. Settings page loads	Settings page loads without crashing	As Expected
Result: Pass		
Test Case UI1-1.3		
Pre-Condition: Data Page loads		
Steps:	Expected Results	Actual Result
1. Click data button 2. Data page loads 3.Graph loads and plots dummy data	Data page loads without crashing	As Expected
Result: Pass		
Test Case ADV 1-2		
Pre-Condition: Heart rate data is transferred to the app		
Steps:	Expected Results	Actual Result
1. Run the app with a logging function to display heart rate values 2. Values are displayed in the logs	Roughly accurate heart rate data appears in the logs	As Expected
Result: Pass		

Test Case BK1-3		
Pre-Condition: Data stored on the database can be retrieved		
Steps:	Expected Results	Actual Result
1. Run the app 2. Check values of variables being retrieved from the database	Data matches with what is on the database	As Expected
Result: Pass		

Test Case BK 1-1.1		
Pre-Condition: Firebase stores data		
Steps:	Expected Results	Actual Result
1. Run App with dummy data	Data appears in database	As Expected
Result: Pass		
Test Case BK 1-1.2		
Pre-Condition: Data can be retrieved		
Steps:	Expected Results	Actual Result
1. Run code to initialize with values from database	Values are initialized	All values are null
Result: Fail		
Test Case BK 1-4		
Pre-Condition: Heart-rate data is retrieved from database		
Steps:	Expected Results	Actual Result
1. Run App with real data 2. Real data is retrieved (as opposed to the dummy data)	Data matches what is on the database	As Expected
Result: Pass		
Test Case BK 1-2		
Pre-Condition: Dummy data is created		
Steps:	Expected Results	Actual Result
1. Design and input raw data to be pushed and pulled	Values are initialized	As Expected

from the database		
Result: Pass		

6. Definition of Done

6.1 Definition of Done Criteria

Reviewed Design	
Completed Code	Code refactored
	Code peer reviewed
Unit Tested	
Zero known defects	

6.2 Definition of Done Validation

6.2.1 Sprint 1

Story ID: UI-1	
User Story: Simple User Interface	
DoD checklist for this PBI:	**Status**
1. Reviewed Design 2. Completed Code a. Code refactored b. Code in standard format 3. Unit Tested 4. Zero known defects	1. Done 2. Done 3. Done 4. Done

Story ID: COM-1	
User Story: Establish Bluetooth Connection To Sensor	
DoD checklist for this PBI:	**Status**
1. Reviewed Design 2. Completed Code a. Code refactored b. Code in standard format 3. Unit Tested 4. Zero known defects	1. Done 2. Done 3. Done 4. Done

Story ID: COM-2	
User Story: Easy Bluetooth	
DoD checklist for this PBI:	**Status**
1. Reviewed Design 2. Completed Code a. Code refactored b. Code in standard format 3. Unit Tested 4. Zero known defects	1. Done 2. Done 3. Done 4. Done

Story ID: UI-2	
User Story: Data Presentation	
DoD checklist for this PBI:	**Status**
1. Reviewed Design 2. Completed Code a. Code refactored b. Code in standard format 3. Unit Tested 4. Zero known defects	1. Done 2. Done 3. Done 4. Done

6.2.1 Sprint 2

Story ID: ADV-1	
User Story: Read heart rate	
DoD checklist for this PBI:	**Status**
1. Reviewed Design 2. Completed Code a. Code refactored b. Code in standard format 3. Unit Tested 4. Zero known defects	1. Done 2. Done 3. Done 4. Done

Story ID: ADV-4	
User Story: Data Precision	
DoD checklist for this PBI:	**Status**
1. Reviewed Design 2. Completed Code a. Code refactored b. Code in standard format 3. Unit Tested 4. Zero known defects	1. Done 2. Done 3. Done 4. Done

Story ID: BK-1	
User Story: Database storage (Firebase connection)	
DoD checklist for this PBI:	**Status**
1. Reviewed Design 2. Completed Code a. Code refactored b. Code in standard format 3. Unit Tested 4. Zero known defects	1. Done 2. Done 3. Done 4. Done

6.2.1 Sprint 3

Story ID: ADV-2	
User Story: Add Rechargeable Battery to Circuit	
DoD checklist for this PBI:	**Status**

1. Reviewed Design
2. Completed Code
  a. Code refactored
  b. Code in standard format
3. Unit Tested
4. Zero known defects
	1. Done
	2. Not applicable.
	3. Done
	4. Done

Story ID: ADV-3	
User Story: Blood-Oxygen measurement	
DoD checklist for this PBI:	**Status**

1. Reviewed Design
2. Completed Code
  a. Code refactored
  b. Code in standard format
3. Unit Tested
4. Zero known defects
	1. Done
	2. Done
	3. Done
	4. Done

Story ID: ADV-5	
User Story: Login page	
DoD checklist for this PBI:	**Status**
1. Reviewed Design 2. Completed Code a. Code refactored b. Code in standard format 3. Unit Tested 4. Zero known defects	1. Done 2. Done 3. Done 4. Done

Story ID: DES-1	
User Story: Wristband	
DoD checklist for this PBI:	**Status**
1. Reviewed Design 2. Assembled product 3. Zero known defects	1. Done 2. Done 3. Done

Story ID: UI-3	
User Story: Personalization	
DoD checklist for this PBI:	**Status**
1. Reviewed Design 2. Assembled product 3. Zero known defects	1. Done 2. Done 3. Done

6. Ethical Dimensions

6.1 Medical device disclosure

There are several significant concerns for the ethics of this product. The first is the legal requirements for an app with medical functionality. Any device that can measure more than just heart rate is subject to an approval process in the United States, where they are regulated by the FDA. Similar restrictions are used in Canada, and devices with their companion app have to be approved. Our device, due to the limited functionality, connectivity, and inability to interpret data (it only shows data), does not require FDA approval to be sold in the U.S..

There is also always a potential for devices making faulty readings that can lead to users inferring self-diagnoses that may lead to the seeking of unneeded treatment. If their heart-rate is falsely depicted as being unhealthily high, this could give them anxiety about that fact and genuinely degrade their health. Similarly, these diagnosis devices may also have the effect on users to think it ok to bypass seeking professional medical advice if the readings seem to be normal to them. Only a trained doctor can provide a proper diagnosis to a patient after monitoring their vitals.

6.2 Privacy, Marketing and User-Data Storage

Furthermore, there is also the privacy issue. The app logs heart rate and blood oxygen saturation data, and will upload this information to a database. One worry customers might have is their information being sold off to third party actors for our own personal profit. Our policy is such that the customer will have nothing to fear as we will not share their information with any

unauthorized parties, and most importantly, will delete all their data if customers ask for total data deletion.

Since the device and app will keep track of aspects of one's cardiovascular health, there is also a potential for insurance companies to weaponize that data if acquired to raise premiums of those they see as unfit. Many companies offer their services for free and make revenue by selling user data to advertisers and other third party actors. We do not agree with this business model and promise to not share any user info with any other party, with the exception of the cloud storage being located on google servers.

6.3 Price Cost And Accessibility

The cost of the device is chosen for several reasons. The first is because there are already many expensive solutions for monitoring heart-rate and blood oxygen saturation, creating a price gap near the bottom end. This factors into the utility of the device, being prices cheap enough to be accessible to a larger number of individuals. While professional athletes and others have access to more expensive products, many individuals who desire the ability to track their own information and progress do not. Cardio$_2$ is priced to increase the number of people who have access to these tools.

6.4 Ecological Concern

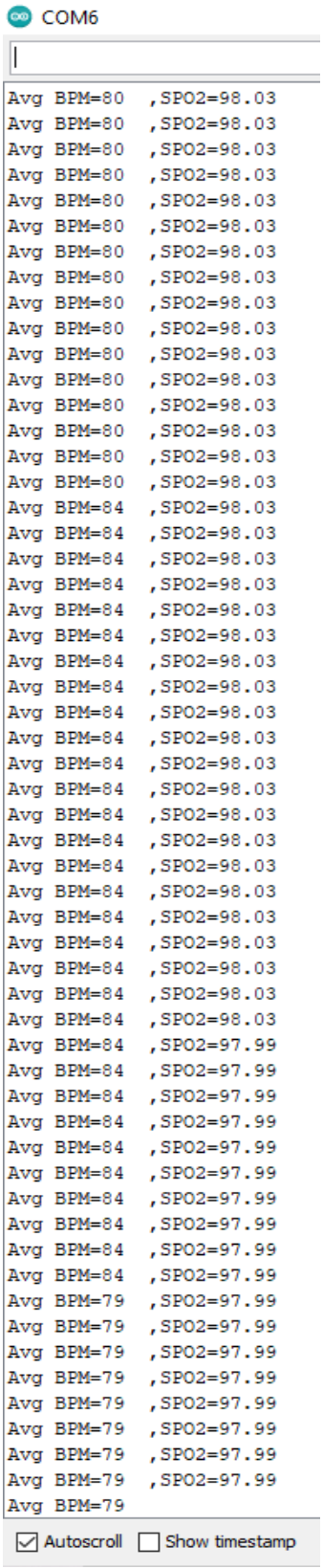
Ecological concerns related to the product life-cycle are also significant. Like most devices, this one does present a solution to the increasing e-waste issue created by increasing electronics production. The use of heavy metals, plastics, and Li-Po batteries, similar to most commercial devices, does not improve our environmental impact. Further effort would be required to incorporate electronics and battery recycling into the product life-cycle.

7. Computer Simulation Summary

Our simulations were divided into two categories, hardware and software, the former which included testing the Arduino code on the sensor to get readings and the latter to upload said data to a Firebase database. As for the database testing, all tests were done using hard coded data to pass back and forth with firebase. The initial tests were done using just strings and we learned how firebase organizes the data it stores and how this relates to the functions that need to be called for sending data. Our app utilizes a class called session to keep a heart rate, timestamp and blood oxygen reading together in one session, so we learned that we can store one session on firebase under an automatically generated ID.

Eventually we updated the session class to use lists of doubles and a Long for the timestamp and this did not change how it was stored on firebase. We initially had a bit more difficulty trying to retrieve data from firebase. In the tests for data retrieval, we realized that getting the data back from firebase tended to cause a lot of scoping issues that left the data in an unusable state. Our initial solution for this was very messy and involved putting the graphing setup and data retrieval all together in the main function of the page the graphs were being displayed on. Through some repeated testing and research we realized these issues could be solved using handlers and this allowed us to clean it up to adhere more closely to an MVC structure.

Testing the MAX30102 sensor with the Arduino was impossible at first, having to do with a manufacturing defect with the sensor, but a solution was found as the sensor had some of its components desoldered and moved around. After having fixed the sensor, we were able to run our Arduino code with it to acquire sensory data displayed in the serial monitor as shown in the picture below.



Serial Monitor Output Displaying Sensor Values

8. Team Blog

Team Members		
First Name	Last Name	Nickname (max 6 characters)
Ali	Turkman	AL
Ian	Stellema	IN
Jake	Dos Santos	JK
Talon	Saintsbury	TL
Roger	Mugisho Namukama	RG

Milestone 2															
3-Oct-21	1	1	1	1	1	meeting	1	Create draft interview script	Interview script prepared	1	1	1	1	1	
9-Oct-21	1	1	1	1	1	Finalize Repo	4	Finalize report and backlog	Finished report	5.5	4	4	4	3.5	
									Total hours	6.5	5	5	5	4.5	
									Total team hours						26
Sprint 1															
25-Oct-21	1	1	1	1	1	Meeting	4	Plan and work Sprint 1	Sprint 1 goal and backlog	4	4	4	4	4	
										0	0	0	0	0	
27-Oct-21			1			Coding	2	Initialize Repo/Empty Activities + App layout planning	Git Repo + App activities	0	0	2	0	0	

29-Oct-21			1			Research	2	Familiarizing with Firebase	N/A	0	0	2	0	0	
30-Oct-21			1			Coding	7	Integrating Firebase	Basic Firebase Functionality + Setup test data sets	0	0	7	0	0	
31-Oct-21			1	1		Coding	3	Add graphing	Graphing/formatting of Data page done	0	0	3	3	0	
	1	1		1	1	Research	2	Familiarize with MAX30102	N/A	2	2	0	2	2	
	1	1		1	1	Coding	1	Setup code for sensor	Get heart-rate data	1	1	0	1	1	
	1	1		1	1	Testing	6	Debug sensor	Found reason the sensor did not work heart-rate data	6	6	0	6	6	
31-Oct-21	1	1	1	1	1	Meeting	4	Plan and work Sprint 2	Assembled circuit, sensor can upload and download data to database	4	4	4	4	4	
									Total hours	17	17	22	20	17	
									Total team hours						93
Sprint 2															
2-Nov-21		1		1		Testing	6	Debug sensor	Get heart-rate data	0	6	0	6	0	
4-Nov-21			1			Coding	4	Attempt to make settings page interact with other pages	N/A	0	0	4	0	0	
7-Nov-21			1			Coding	4	Attempt to retrieve data sent to Firebase	N/A	0	0	4	0	0	
11-Nov-21			1			Coding	3	Database retrieval and graphing of data added	Data page can now graph data from firebase	0	0	3	0	2	

12-Nov-21		0.5	1	1		Coding	4	Integrated timing with incoming bluetooth data	Sessions can now be created using the data received via bluetooth	0	2	4	4	0	
12-Nov-21	1	1	1	1	1	Meeting	5	Decide what new features to add and which to abandon	Decided to add several new features later added to product and worked on code, decided to buy MAX30105 sensor and circuit layout	5	5	5	5	5	
13-Nov-21	1					Coding	8	Dedicated Bluetooth Sensor Connection Activity	Unable to make BT discovery auto-connect to sensor work, activity scrapped and BT turn ON/OFF button kept	8	0	0	0	0	
13-Nov-21				1		Coding	6	Optimized/fixed timer	Sessions can now be created using the data received via bluetooth	0	0	0	6	0	
14-Nov-21	1	1	1	1	1	Meeting	4	Plan and work Sprint 3	Goal: Fully functional product	5	4	4	4	4	
										0	0	0	0	0	
									Total hours	18	17	24	25	11	
									Total team hours						95
Sprint 3															
										0	0	0	0	0	
		1		1		Research	1	Research implementations of blood-oxygen setup	N/A	0	1	0	1	0	

		0.5		1		Coding	2	Get Blood oxygen Readings on arduino alongside heart-rate	Read Blood-Oxygen saturation on arduino	0	1	0	2	0	
		0.5		1		Testing	4	Test arduino code with added blood oxygen readings	Functional Arduino blood oxygen readings	0	2	0	4	0	
		1				Assembly	3	Soldered breadboard assembly	Soldered board that takes up less space, more reliable.	0	3	0	0	0	
19-Nov-21			1	1		Coding	8	Refactoring	Much cleaner code, selectable session list for graphing added	0	0	8	8	0	
20-Nov-21	1					Coding	8	Design UI	App UI designed, activity schemes and logo defined	8	0	0	0	5	
22-Nov-21	1					Coding	5	User experience features	Profile picture, additional user-interactive elements, splash screen, and login page visuals	5	0	0	0	0	
25-Nov-21	1	1	1	1	1	Meeting	4	Team meeting	Planned future presentation and finalized some aspects of the app	4	4	4	4	4	
27-Nov-21	1	1	1	1	1	Testing	5	Testing/debugging	Less buggy app	5	5	5	5	5	
28-Nov-21	1	1	1	1	1	Presentation	3	Presentation preparation	Presentation	3	3	3	3	3	
30-Nov-21	1		1	1	1	Testing	2.5	Emergency bug fixes	Fixed new bugs in app	2.5	0	2.5	2.5	2.5	
2-Dec-21	1	1	1	1	1	Meeting	5	Final Report Work	Finished Final Report	7	7	5	5	5	
									Total hours	34.5	26	27.5	34.5	24.5	

9. References

[1] Pew Research. “About one-in-five Americans use a smart watch or fitness tracker.” *About one-in-five Americans use a smart watch or fitness tracker*, Pew Research Center, 9 January 2020, https://www.pewresearch.org/fact-tank/2020/01/09/about-one-in-five-americans-use-a-smart-watch-or-fitness-tracker/. Accessed 2nd December 2021.