UNIVERSITY OF WISCONSIN – MADISON

POGO

Pulse oximetry, on the go

Department of Biomedical Engineering

Advisor: Dr. Amit Nimunkar
Client: Dr. Fred Robertson

Chris Fernandez (Team Leader)
Olivia Rice (Team Communicator)
Rafi Sufi (BSAC)
Nick Glattard (BWIG/BPAG)

[12/11/2013]

Abstract

A pulse oximeter is a device that noninvasively measures oxygen saturation in arterial blood. This allows for the observation of basic respiratory function. Patients with chronic diseases such as high-risk congestive heart failure (CHF), chronic obstructive pulmonary disease (COPD), or severe asthma do not require continuous monitoring in a hospital, however, could greatly benefit from periodic oxygen saturation monitoring to determine their state of health. Effective monitoring is challenging in the home because current pulse oximetric technologies have a limited data collection range, require patient cooperation and do not automatically share vital data with physicians. To solve these types of problems in outpatient care, the POGO device automatically collects a patient's oxygen saturation reading and then transmits the data over the 2G/3G network directly to a web application. The physician and care team can access this web application and view the pulse oximetric data in real time. On a high level, the progression of data transmission is as follows: from the POGO device the data is sent to a Xively server. It is then pulled from this server, uploaded and stored on a Heroku cloud database where it can be accessed through the Django web framework and Twitter Bootstrap user interface. Physicians will be able to access this website from their cell phone, personal computer, tablet or other configured device. Ultimately this device could improve a patient's quality of life and provide insight to trends regarding the decline of health in patients with various chronic diseases.

Table of Contents

1.0 Problem Statement

Many patients affected by chronic diseases such as congestive heart failure (CHF), chronic obstructive pulmonary disease (COPD), or severe asthma would greatly benefit from frequent blood oxygen saturation monitoring by a physician. Monitoring patients outside of the hospital is difficult with current technologies due to their limited transmission range, requirement of high patient compliance and their inability to automatically transmit the data to a physician. This POGO device will automatically collect and transmit patient's blood oxygen saturation data from any location with cellular coverage, to one or more physicians simultaneously through the website application that is designed. Last semester the POGO device was designed and prototyped. Although the device met key client specifications, an application and user interface is needed for physicians to view patient oximetry data in real time. Physicians, nurses and even patients themselves can use this highly functional user interface, complete with adjustable alarm thresholds, to stay up to date with the patient's state of health. This will allow for a physician to make early intervention decisions to prevent hospital readmission. Additionally, a secure database that has the ability to store and analyze medical information will provide insight to trends regarding the decline of chronic patients' health. The device improves patient quality of life by providing freedom of mobility, a hands-free lifestyle and has the potential to reduce hospital readmissions and emergency room visits.

2.0 Background

Pulse oximeters measure arterial blood oxygen saturation. For adult patients, sensors are typically placed on the fingertip or the ear lobe. The oximeter sensor contains two LED lights on one side, which passes Red (660 nm) and infrared (940 nm) light through the patient's skin to a photo detector on the other side of the device. The photo detector collects the light that wasn't absorbed by hemoglobin in the bloodstream.[1] The LED lights typically flash in an alternating fashion so a single photo detector is able to measure each of the light intensity levels. With a known absorption of both infrared and red light, a ratio of oxygenated hemoglobin to deoxygenated hemoglobin can be calculated. This ratio is what is commonly referred to as a patient's oxygen saturation reading (SpO_2).[2] The raw data collected by the photo detector is integrated and can be displayed both graphically and numerically as a plethysmograph and numerical reading accordingly.

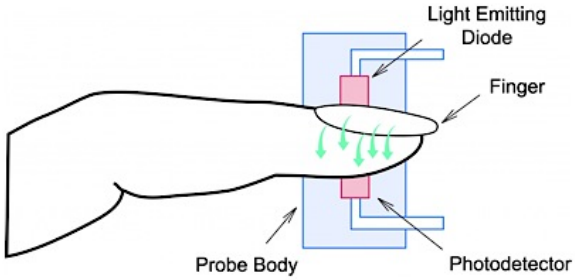

Figure 1. Depicts the basic functionality of a finger pulse oximeter.[7]

Figure 2. Nonin's Avant 9700 oximeter displays both graphical and numerical data.[8]

Pulse oximetry is an essential practice for physicians to assess a patient's health. Studies have shown that a patient with oxygen saturation levels below 90% for an extended amount of time can lead to an increased chance of mortality. When considering health and safety of the patient, it is highly recommended that they be admitted to the hospital when oxygen saturation levels have declined to 93%.[3] However, in an at home setting, patients without constant monitoring are often unaware that their oxygen saturation levels have declined past this threshold. Eventually the patient will become severely deprived of oxygen and in the case of CHF, will start to experience heart failure and be admitted to the emergency room or possibly die. This high-risk situation could be avoided if physicians were aware of their patient's SpO_2 levels and could make an early intervention decision.

2.1 Previous Work

Last semester, a thorough patent search was conducted to see if anyone had acquired space in automatic SpO_2 transmission over the cellular network. However, there was no such patent that indicated this form of communication. The oximeters currently on the market for a home care setting either don't transmit data, use Bluetooth for data transmission to a device nearby, or are directly connected to an alternate device that is used for the transmission of data such as a cellular phone or computer. The POGO is different from all current technologies, such as a Bluetooth pulse oximeter or one that plugs into a smart phone, because it is a stand-alone device with unlimited transmission range that requires minimal patient compliance. The patient simply has to wear the device and their SpO_2 readings will be automatically collected and transmitted to their physician. There is also no dependence on the functionality and battery life of other devices such as a cellular phone. In Figure 3 is a photo of the device we created last semester. This device has the capability to collect a signal from the oximeter sensor, and send it over the cellular network to a TCP server called Xively.

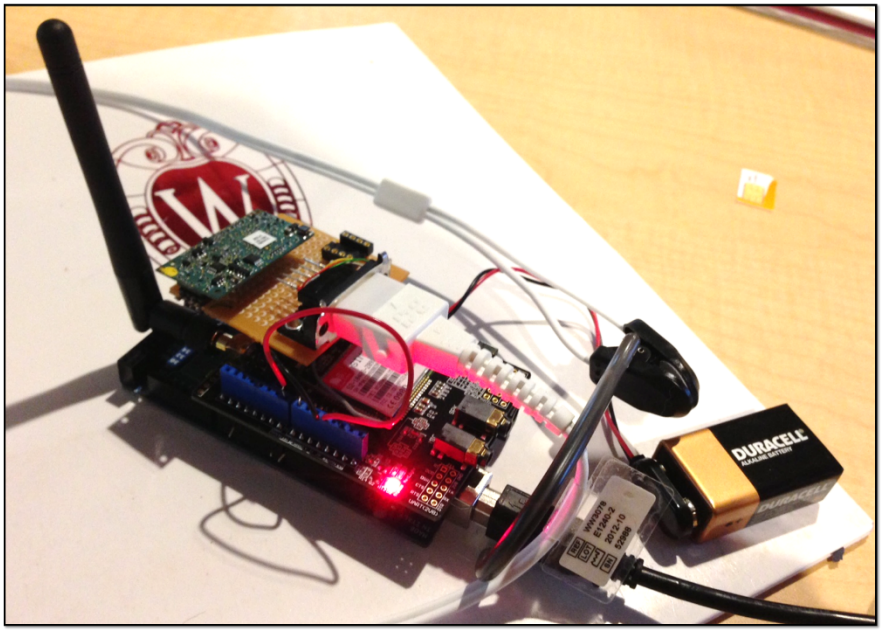

Figure 3: Pogo device complete with an oximeter sensor, BCI OEM board, Arduino microcontroller and GSM shield.

A power usage test was conducted as well as a test of time required for the device to change state from completely off to being able to process commands that initialize data transmission. Each test was performed 20 times. Figure 4 shows the results of the power consumption tests. It is shown that power does not exceed 630 mW for extended periods of time. This low power consumption level indicates safe temperatures for patients.

All prototype components connected at 4.5 Volts	Current (mA)	Power (mW)
GSM off and Arduino idle	69	310
GSM connecting/ sending	140	630
GSM module on and idle	120	540
Average if sending once every 15 mins with Arduino power on	75	340
Average if sending every 15 minutes with complete power down between transmissions	12	68

Figure 4: Shows a comparison of current in various modes of the prototype.

It was also shown that it took 16.1 seconds on average during the twenty tests performed for the device to change state with a standard deviation of 1.21 seconds. The time allotment before commands were sent was set to 19.7. This allowed for three standard deviations of the mean to be covered which will allow for 99.7% of the attempts to upload data to be received correctly by the GSM Shell if a normal distribution is assumed. Using this 19.7 seconds we see the average current draw is 135 mA over the course of the 80 seconds between power on and power off. If we average this over 15 minutes with the system powered off in between each packet of data sent, the average current over time is 12 mA. With alkaline AA batteries with a battery life of 2700 mAh, this leads to a conservative approximation of a weeklong battery life.

The team attempted to show that the SpO_2 and pulse rate values calculated by the OEM system (Smith's Medical BCI) were not corrupted during communication to the host device (Arduino). The Arduino was receiving a signal from the BCI processor however, the BCI was sending a consistent flag indication "No finger present in sensor" whether or not a finger was placed appropriately in the sensor and therefore no accurate data was collected. This issue is still prevalent and a new processing board is being acquired.

3.0 Design Motivation

The need for a real-time clinical decision making tool is increasing due to the aggressive policies aimed at improving the current state of healthcare in the U.S. The Affordable Care Act added a section to the Social Security Act establishing a Hospital Readmissions Reduction Program which requires Centers for Medicare and Medicaid to reduce payments to hospitals with excess readmissions beginning in 2012.[10]The healthcare quality of care model is seeing a rapid shift in ideology, due to these external influences. For example, the healthcare model in America used to encourage more patients to visit the hospital. This meant greater revenue for the hospital. However today healthcare model is shifting towards more preventative methods and keeping patients out of the hospital with better outpatient care.

Telehealth technologies are emerging in healthcare markets because of their ability to reduce healthcare costs as well as provide prevention techniques to enhance the quality of care. Telehealth technologies can be described as the use of electronic information and telecommunications technologies to support long distance clinical health care[4]. A study conducted in 2008 by the Department of Health in the UK focused on health care costs between patients who were treated with standardized health care equipment and patients who were treated with telehealth technologies. The study concluded that there were reductions across various cost categories as well as a 45% decrease in mortality rate, as seen in the figure below.[5]

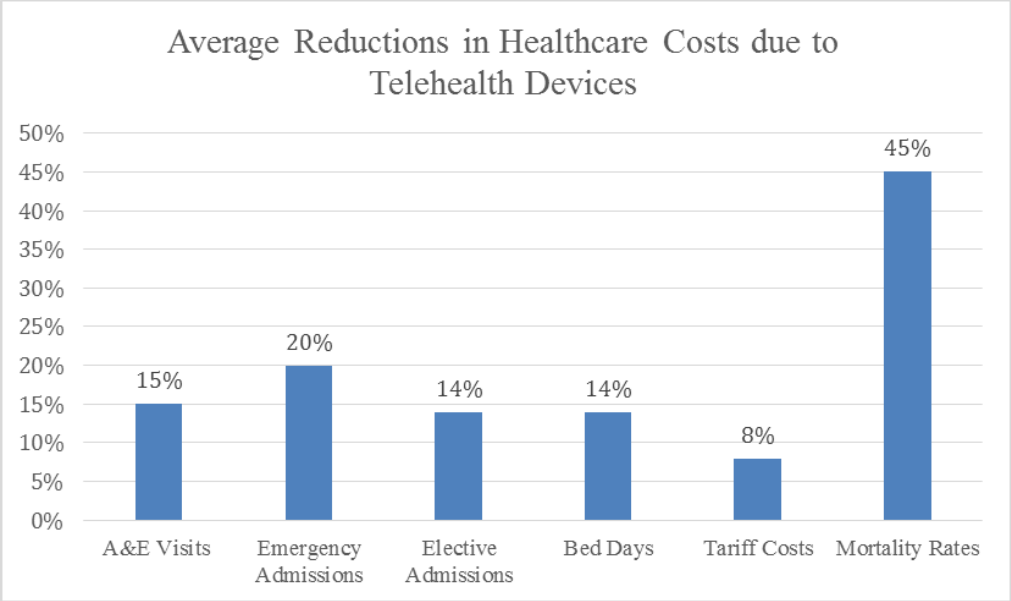

Figure 5: Department of Health study showing various reductions in healthcare costs. United Kingdom. Department of Health. Whole System Demonstrator Programme. London: 2011. Web.

In addition to proving cost reductions are attributed to utilizing telehealth devices, the study also states at least three million people who suffer from long-term conditions such as heart

failure, chronic obstructive pulmonary disorder (COPD) could benefit from using telehealth technology[5]. The Pogo, termed as a telehealth device, will be able to reduce costs and improve healthcare intervention techniques.

4.0 Client Requirements

The main client requirement described for this semester is to create a web development framework to be able to store and view real-time patient data that is being transferred from the pulse oximeter. This web development framework will need to feature several key components. First, a method of retrieving the data from the GSM shield is needed. Next, a method of storing the data is required in order to handle multiple patient data and view recent trends of biometrics. Finally, a method for viewing and interacting with patient data is needed, which will be user interface.

Another important requirement of this project is to be able to produce a fully functioning POGO, which transmits sp02 and heart rate from the sensor to a web client, where the data can be viewed in real-time. More specifically, a new pulse oximetry board needs to be ordered to accomplish this due to circuitry issues with the current BCI board.

In terms of content, the client specified several key components that a physician needs on the web application in order to accurately assess the patients health. Listed below are the initial criteria needed to be displayed.

- A method to retrieve the patient identifier number which allows the physician to access medical records
- The ability to view the patient's biometric trends within the past month
- Most recent waveforms of heart rate, oxygen saturation
- User adjusted alarm set points
    - Alarm notification to caregiver
- Storage for patient data
- Signal quality indicator to indicate accuracy of data

By performing contextual interviews with local physicians, ease of use and accessibility were among the top concerns for viewing a web client. The physicians emphasized the ability to quickly scan patient measurements, since some health care staff members are responsible for monitoring many patients at a time. Speed of transmission was also a suggestion brought up in interviews. Physicians were concerned with the data load times of patient data onto the website. While this may be attributable to the hospital Wi-Fi connectivity, it can also be effected by the number of files and code that is uploaded to the web client. Another important requirement derived from contextual interviews is the ability to categorize patients in terms of the severity of their vital signs. The at-risk patients who display sp02 readings of 87% or below, for instance, can be highlighted in red and placed at the top of the dashboard. In essence, physicians wanted:

- Quick overview of all patients
- Quick loading of data onto the web client
- Categorization of patients based on severity of sp02 reading

These additional suggestions will also be incorporated into our final design and testing procedures.

5.0 Design Constraints

Web Application – front end: The website being designed must display all of the relevant data collected by the BCI sensor (blood oxygen level, heart rate, plethysmograph, etc) for each patient in a user-friendly interface. The data should be grouped by patient and should be accompanied by additional patient information such as a picture and contact information. The client should have the ability to set the real-time measurement thresholds for when they should be alerted, who should be alerted (patient, significant other, doctor), and how they should be alerted (text message, email, pager). Patients should be prioritized on the main interface by their alarm statuses making the most important information actionable at a glance. Since the data being transmitted and received is private health information, it must be encrypted and clients must enter the site through a secure login page. If we make a patient interface, it must also have a secure login, cannot display other patient's information, and should have clear instructions on when to contact their healthcare provider.

Web Application – back end: At this point in the project, the current design sends information through the Xively and Heroku databases. These services charge once our device uses more than the allotted memory, which the team would like to avoid. This restricts the scalability of our project in the long run and avoiding these issues will be discussed further in the future work section.

POGO Device: Device must be small enough to be wearable. At this point we would like it small enough to be clipped to the belt or slipped in a pocket while the sensor remains clipped to the ear. The pulse oximeter must collect accurate data that satisfies FDA regulations and the microprocessor must be able to correctly parse the data being sent to it from the pulse oximeter. The GSM Shield component should be able to reliably connect to the GSM network. Furthermore, since the data being transmitted is private health information, it must be encrypted.

6.0 Design Alternatives

To address the clients design specifications, the potential effectiveness of the *POGO* patient data visualization program was evaluated on three different platforms; as a desktop

computer application, a mobile application, or a website application. Each alternative would require a different application workflow process, set of development tools, and testing protocols. Moreover, each alternative impacts important application usability factors including where, when, and how often physicians are able to use this tool to supplement clinical decisions.

6.1 Alternatives Defined

Desktop Application: The desktop application alternative is defined as a program that is native to a single laptop or desktop computer, and is reliant on WiFi connectivity to receive and update patient information.

Mobile Application: As a design alternative, the mobile application is an 'App' that is native to a single tablet or smartphone. Depending on the specific hardware used, this alternative can be compatible with WiFi and Cellular wireless networks to receive and update data.

Website Application: The final design alternative is the website application. This is defined as a program that is hosted on web servers independent from any end-user device. Therefore, this application could be accessed through a web browser on any web compatible device.

6.2 Evaluation Criteria

In order to objectively assess which of the three potential alternatives would best suit the client and end-users needs, each alternative's efficacy was analyzed across 7 criteria. The two criteria deemed most critical are ease of use by physicians and reliability of the application. Ease of use encompasses the factors that determine the convenience to use the app. In order for this tool to be used effectively, the design must promote frequent enough utilization that patient trends can be recognized. Reliability relates to the probability of errors occurring on the end-user hardware or earlier the process of transferring information.

The next highest tier of criteria includes cross-platform portability and the team's ability to continuously deploy software updates. The applications' cross-platform portability depends on the end-user devices native operating system and preferred web client. Furthermore, the team sees value in having the ability to deploy updates to the program functionality quickly and iteratively. The third and final tier of criteria encompasses the ease of application development, safety, and cost to the end-user. In this context, safety is defined as the likelihood that an end-user hardware error hinders a physician's ability to make a real-time intervention on a patient. Although this safety is a central part of the team's vision, the team is unable to take responsibility for patient care beyond creating the best possible tools for intervention.

7.0 Decision Matrix & Discussion

Alternatives & Criteria	Weight	Desktop Application		Mobile Application		Website Application	
Cost	10	5	10	5	10	3	6
Safty	10	2	4	4	8	5	10
Ease of Use	20	2	8	3	12	5	20
Continuous Deployment	15	3	9	4	12	5	15
Reliability	20	4	16	3	12	5	20
Portability (cross-platform)	15	1	3	1	3	4	12
Ease of Development	10	2	4	2	4	4	8
Total	**100**		**54**		**61**		**91**

Figure 6. The web application decision matrix showing the design alternatives as columns and evaluation criteria as rows. Category winners are shown in green, with second place shown in yellow, and third in red.

Following the analysis of each design alternative, the website application was found to be the best design by a significant margin. The web application was determined to be the leading design in the two most important evaluation criteria; ease of use and reliability. It was superior the desktop and mobile apps in terms of ease of use because it is interoperable across both of those platforms. In the case of the desktop, physicians must be beside their computer and have WiFi access, and in the mobile format they would be unable to view patient data on a desktop computer. Furthermore, in any cases where a physician's tablet, phone, or laptop was to die, their ability to monitor patients would also fail if they are using a native desktop or mobile app respectively. Because the web app could be accessed from any other device after authentication in the case of a failure, the web app was ranked first in the reliability category as well.

In addition, the web application was also shown to be the superior alternative in the second tier categories; cross-platform portability and continuous deployment. If the app was developed for a desktop or laptop computer, compatibility issues would arise unless versions of the app were created for each of the common operating systems; Windows, OSX, and Linux. Similarly, a mobile or tablet application would need to be compatible with numerous mobile operating systems, including Android, iOS, and Windows. In the case of the website application, the URL could be accessed from any operating system. Moreover, there are web development tools available which render a website compatible with all major browsers, including Google Chrome, Mozilla Firefox, Internet Explorer, and Safari. Next, the web application is the only of the three alternatives whose software updates capabilities rest fully in the hands of the design team. This is advantageous because it enables the most flexible, iterative software updating and improvement process possible, and simplifies the user experience for the physicians, as they do not have to engage in that process actively.

Finally, the web application was found to be the best alternative for two of the three third tier criteria; cost, ease of development, and safety. The desktop and mobile applications were both more cost efficient solutions than the web application because of the increased server

architecture that would be necessary to manage all of the computation, data, and users on a pure website structure. The web application, however, benefits from the greatest flexibility in terms of tools, tutorials, and open source code that is available to aide in accelerating the application development. The mobile application development would be restricted to the native devices development program, for example the iOS development center SDK. Furthermore, desktop applications face the greatest complexity in building a full program from scratch. Finally, the web application has the best safety features because of its flexibility and portability across numerous devices and operating systems. In cases where physicians receive critical health notifications from the POGO device, the response time can be expedited the most by this broad flexibility.

8.0 Final Design

8.1 Signal Acquisition and Processing (The POGO Device)

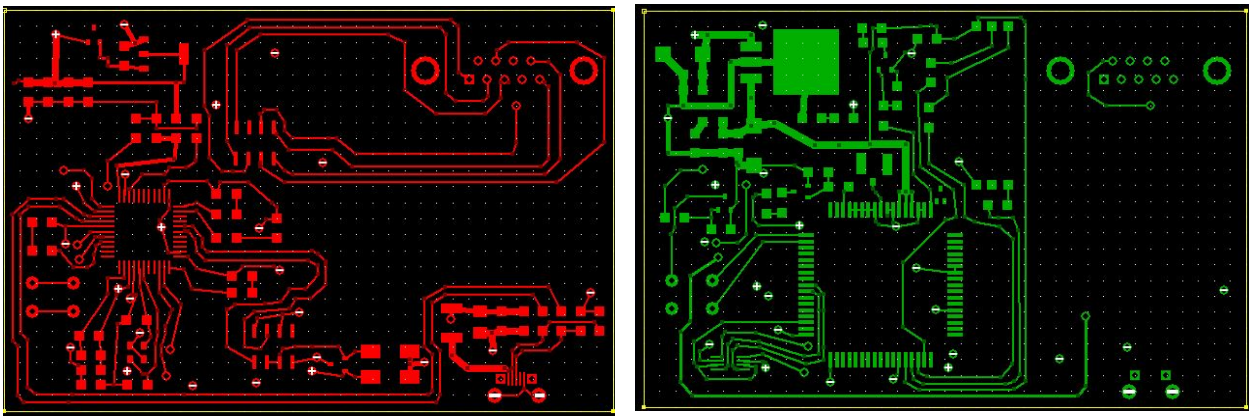

Figure 7. Circuit board layout of the second generation POGO device. These layouts were designed using Eagle PCB design software.

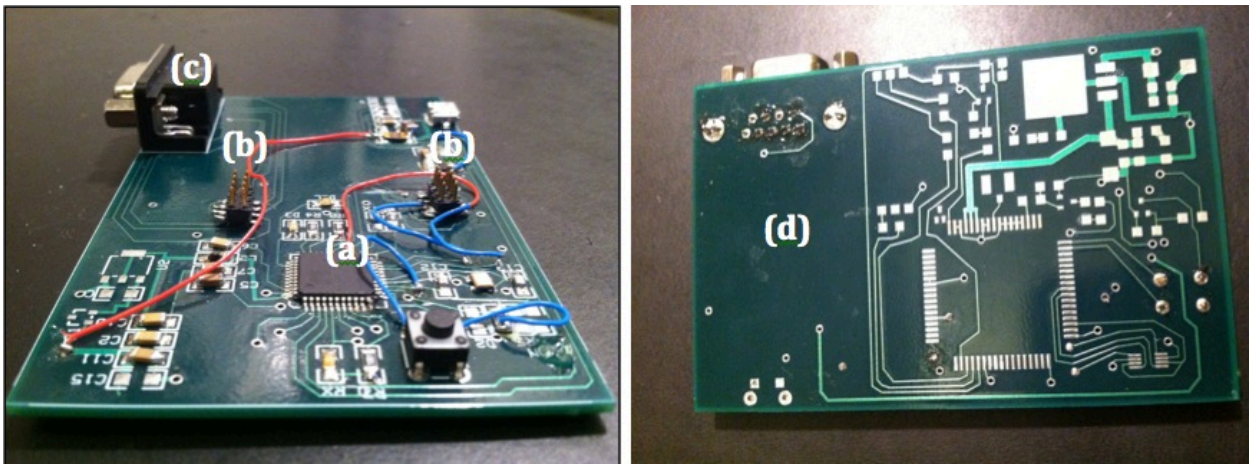

Figure 8. POGO 2.0 populated with ATMega32U4 microcontroller (a), BCI board connectors (b), and BCI sensor DB9 connector (c). Seeedstudio GSM shield will be populated on the opposing side (d).

The POGO 2.0 has started the miniaturization stage of its design for patient comfort/portability and power usage efficiency for battery life. The first iteration of the circuit board is comprised of the three main components of the device: a mounting area for the Smith's Medical Pulse Oximetry module, a wireless GSM transmission module, as well as the ATMega32U4 microcontroller. The base of the board was developed using the open hardware EAGLE files from Seeedstudio (GSM module) and Arduino. A simple USB interface is also included through which debug and programming of the ATMega32U4 is carried out. The ATMega32U4 serves to host the Smith's Medical module by determining averaging modes, plethysmogram syncing and gain factors, as well as receiving and storing data. The ATMega32U4 also serves to host the GSM module. At appropriate intervals, a TCP/IP connection is made with the Internet database to upload the relevant physiological data of that was retrieved from the pulse oximetry module.

Currently, the ATMega32U4 and its peripheral components as well as the Smith's Medical Pulse Oximetry module mounting components are populated on the board. The GSM components have not yet been received from shipping. This partially populated board runs like the currently used Arduino and BCI board but with the new USB interface.

In parallel, the team spent the semester trying to debug the issue with the BCI board and sensor, but have not been able to determine the problem. Attempts to solve the problem included code review (to ensure the problem did not lie in the code), raw data parsing (to ensure the data outputted from the BCI board was being parsed correctly), datasheet analysis (for deeper understanding of the board itself), and troubleshooting with BCI quality engineers. After multiple conversations with the BCI engineers, it was confirmed that the issue remains between the photodiode and the input to the BCI board. Further attempts to solve the problem will be outlined in the future work section.

8.2 Database Layer: Xively & Heroku

Patient data is sent from the POGO to the Xively cloud platform as a service (PAAS) through a GSM websocket embedded in the Arduino code. This transmission is secure for two reasons. First, no identifiable patient data is transmitted and received by Xively, only an anonymously generated patient ID. Next, the transmission is only receivable by Xively because a secure API access key provided by Xively must be verified to initiate the transfer of data.

A Python script written in Django asynchronously acquires data through the Xively API with the assistance of AJAX web development libraries. This allows common transactional processes between the web client and the database to execute normally while new datastreams are stored. More specifically, a 'subscribe' method executes any time there is an update to the Xively datastream from the POGO device. Once acquired by the 'subscription', a 'callback'

function is called to parse the new data, which is received in a JSON format, then it is subsequently stored in the Heroku database.

This patient data is archived in Heroku, a cloud database PAAS the Heroku database is implemented in PostgreSQL, a powerful system for managing object-relational databases. The data fields shown on the left of figure 6 are received by the Heroku PostgreSQL database when a Python script requests data updates from the open Xively API. Using the patient ID as an identifier, this data table is then related to a separately partitioned data table containing all of the relevant patient and physician contact information. This database partitioning is valuable because it enables the minimization of data redundancy, and correspondingly maximizes the possible efficiency for searching and querying the physiological data. Once data is available, it can be accessed for viewing and analysis through the Django framework.

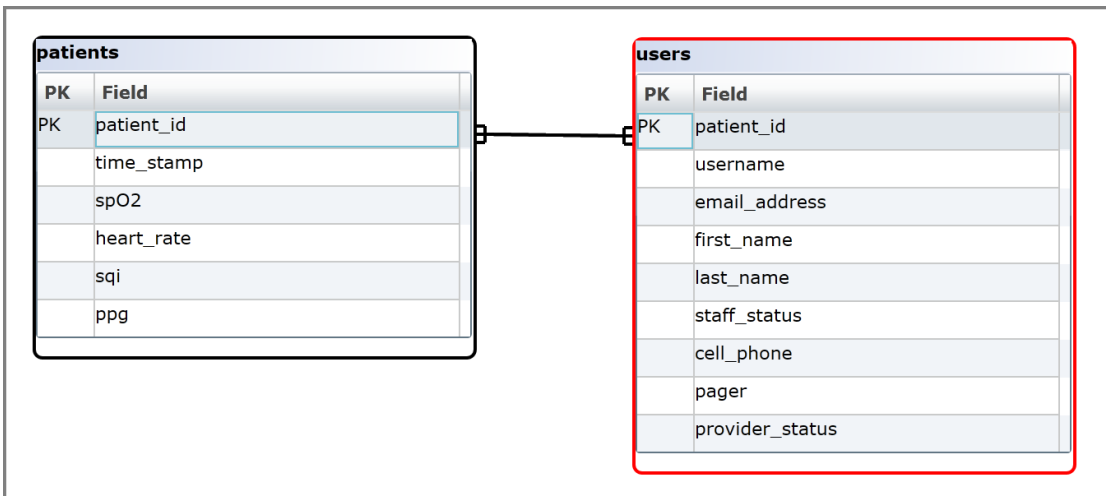

Figure 9. The left data table shows the data fields collected and transmitted from the POGO device. The data table on the right contains relational information and is partitioned separately from the physiological data.

8.3 Application Layer: Django, Python, & AJAX

Django is python-based web framework designed for developing database-driven websites. It acts as an intermediary between the web client and the database to perform data and http response and requests. Django acts as a purely transactional module, as it pulls data from the Heroku PostgreSQL Database and transfers it to the interface. Since it is built on python code, the framework is highly flexible and is supported by all platforms (Mac OS X, Linux, and Windows) and is used widely across the corporate atmosphere (Microsoft, Google, Sun).

The Django framework is built upon the Model-View-Controller (MVC) software architecture design. However it is slightly modified to a Model-View-Template design[6]. The theory behind this software design is to allow strict separation between each layer of the

framework (Model, View and Template), making it easier to make changes in one layer without affecting the others. Each layer has a specific function which contributes to the web frameworks overall purpose.

The model layer is the data access portion. This layer is responsible for the data of the project. The layer extracts the data from the database, validates it, and decides what are the behaviors and relationships between the data sets[6]. In terms of our design, the data layer will interface with the PostgreSQL database to extract patient information.

The template layer is the presentation layer. The data accessed from the data layer is in raw format and the template layer decides how the data showed be displayed and the format in which it appears. The layer possesses syntax similar to HTML in order to accommodate web designers[6]. This layer includes all the HTML files that are viewed on the interface such as the patient page and D3 virtualization.

The view layer is the business logic layer. View is responsible for accessing the data from the model layer and attributes the appropriate template for display. In other words, the view layers acts as an intermediary between the model and the template layer. The view layer receives an http request from the web client for example, the user clicking on a button to view a patient's heart rate, and sends the appropriate http response, in this case the heart rate.

What connects Django to the PostgreSQL database is a web socket using AJAX. AJAX is not a new programming language, but a combination between JavaScript and XML.[9] AJAX is an acronym for Asynchronous JavaScript and XML and is used to create asynchronous web applications. With AJAX, constant, uninterrupted data communication can occur from a server to a web client in the background without any external factors.[9] This means the data on the webpage can be constantly updating, without the user needing to keep pressing 'refresh' on the web browser. In regards to our project, AJAX is used as a web socket technology to asynchronously update our patient data (sp02, heart rate) from the database, through Django, to the web client. Currently, AJAX is deployed between our PostgreSQL database and Django application to allow for constant pulling of data from the database.

8.4 Interface Layer: Bootstrap, jQuery, & D3

Twitter Bootstrap is a framework that incorporates HTML and CSS-based templates for buttons, forms, typography, navigation and other various user interface components of a webpage. There are also JavaScript extensions that are compatible as well. Bootstrap is a completely free collection of tools for creating the user interface design of web applications and websites. So far a basic web layout for both a home and personal patient page has been designed. Seen in figure 10 below is the functional patient page. The patient page may be accessed at https://mht-c9-jbuckner.c9.io .

The Data-Drive Documents (D3) library was used to create interactive, asynchronously updating data visualizations of patient data (figure 10). In order to accomplish this, patient data was first pulled from the database layer using SQL statements triggered by the loading of a patient page. The data in this array was extracted, parsed for correct JSON formatting, and sent to the D3 graphing function. This function uses minimum and maximum time frame, SpO2 and heart rate values to scale the x and y axes respectively. Following this, data points were constructed as JSON objects containing an x value (timestamp) and y value (SpO2 or heart rate). Each of these JSON objects is represented by a SVG (scalar vector graphic), which enabled the color-coding of data points based on specified thresholds. As a result, data points in the 'severe' category were colored red, the 'cautionary' category yellow, and the 'well' category green. Following this process, a D3 'canvas' was constructed supporting the correct x and y scales, and the JSON points are overlaid in the correct locations. A 'linefunction' was then constructed and called to connect subsequent points to one another, allowing for the visualization of trends.

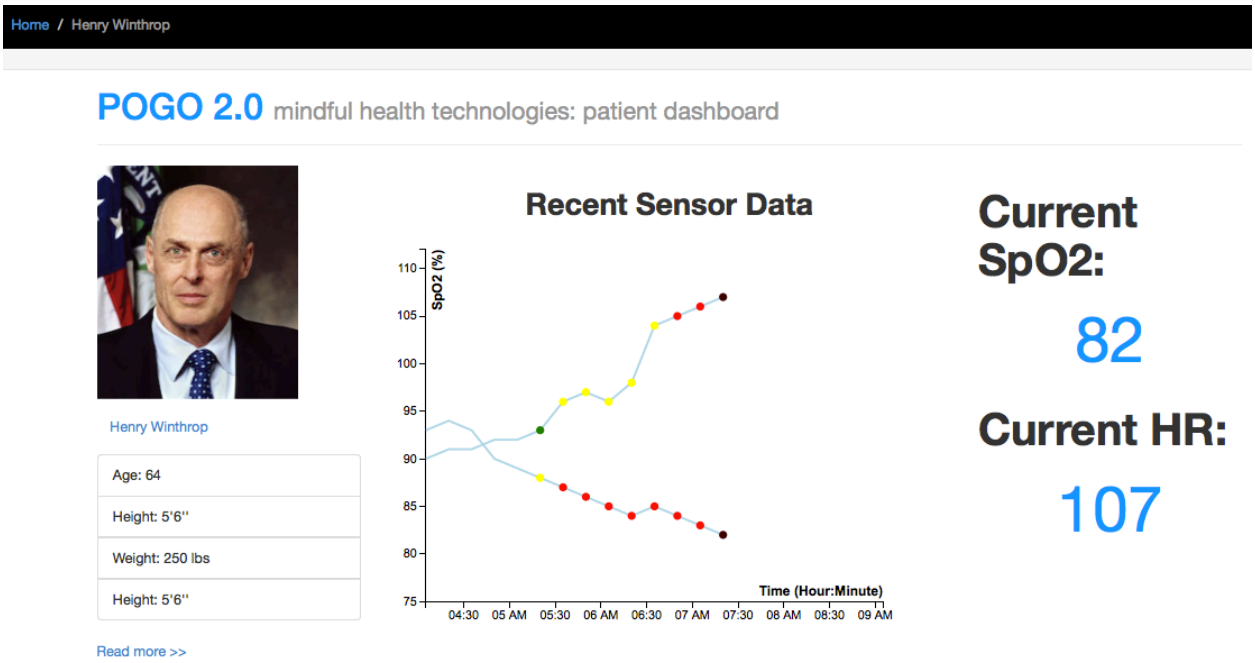

Figure 10: A personalized patient page with SpO_2 and heart updating at predefined time intervals.

The team then developed an algorithm to trigger an alert notification on the web browser. This algorithm counted the number of consecutive 'severe' (red) JSON data points and called the alarm using jQuery when 10 consecutive points met this criterion (figure 11). jQuery enabled the handling of this event by calling a hidden HTML5 script for the red alert notification and making it visible when called by this algorithm.

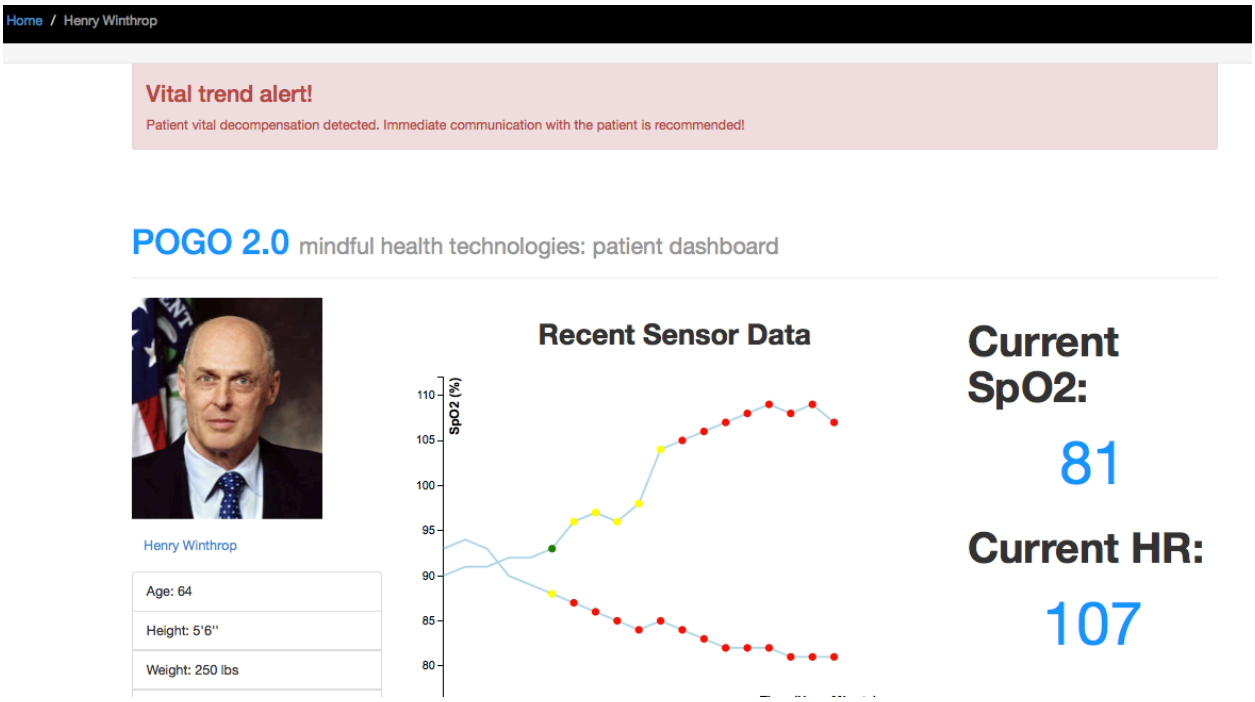

Figure 11: A personalized patient page with a real time alert notification.

The home page (Figure 12) is still on a personal computer server and thus cannot be accessed on the internet. After consulting with a few physicians, a home page was designed with a stop (critical-red), warning (yellow) and go (stable- green) mentality. The physicians expressed that they wanted to be primarily concerned with the critical patients so that they could make quick decisions regarding their state of health.

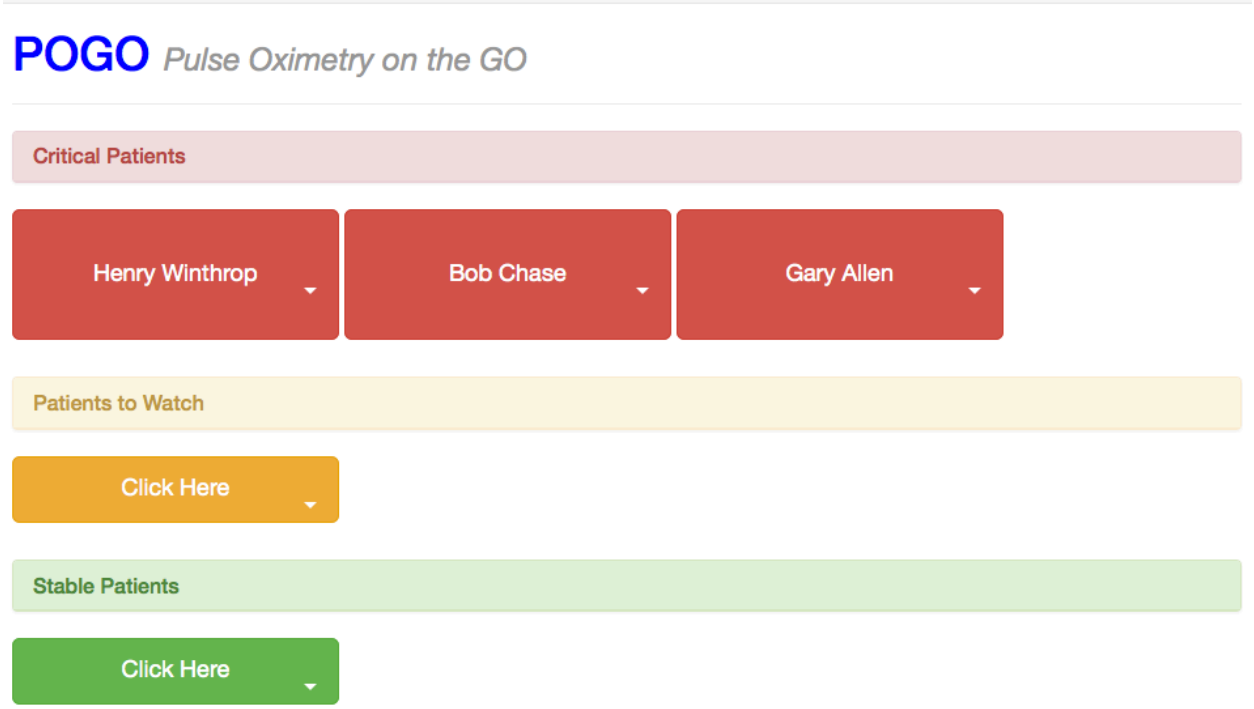

Figure 12: The home page user interface created with Bootstrap

9.0 Testing

9.1 POGO Device

The POGO tests must verify that the device can accurately measure the patient's blood oxygen data and send it via the GSM network to Xively. First, is to test that the BCI pulse oximetry processing chip is working and that the Arduino can parse the information being received. The procedure is as follows:

Pulse Oximetry Accuracy Test

1. Attach both the BCI pulse oximetry sensor and a separate pre-verified pulse oximeter to your body
2. Set Arduino to debug mode
3. Run program to retrieve data from pulse oximeter chip
4. Output values retrieved in the debug window
5. Record the values from the BCI pulse oximeter and the verified pulse oximeter at the same moment in time in the provided table:

Trial #	BCI Pulse Oximeter		Verified Pulse Oximeter	
	$\%SpO_2$	Pulse Rate (bpm)	$\%SpO_2$	Pulse Rate (bpm)
1				
2				
3				
4				
5				
6				
7				
8				
9				
10				
11				
12				
13				
14				
15				
16				
17				
18				
19				
20				

6. Compare values for SpO_2 and pulse rate with the secondary pulse oximeter output

7. Record if BCI values are within the guaranteed accuracy range of the verified pulse oximeter for each trial (ex. If verified pulse oximeter guaranteed accurate +/-5%, then BCI output must lie within that range
8. If all BCI data is within the verified pulse oximeter's accuracy range, then this test passes

As the BCI pulse oximeter is currently not working, this testing must be held off until the issue has been resolved.

The second test would be to send this pulse oximetry information through the GSM Shield, through the GSM network, to Xively. The procedure is as follows:

GSM Shield Test

1. Set Arduino to debug mode
2. Run program to retrieve data from pulse oximeter chip
3. Record data received in the table provided
4. Run program to send that data through the GSM Shield to Xively
5. Record data that was sent to Xively in the table provided

Trial #	BCI Pulse Oximeter		Xively Database	
	$\%SpO_2$	Pulse Rate (bpm)	$\%SpO_2$	Pulse Rate (bpm)
1	100	100	100	100
2	99	101	99	101
3	98	102	98	102
4	97	103	97	103
5	96	104	96	104
6	95	105	95	105
7	94	106	94	106
8	93	107	93	107
9	92	108	92	108
10	91	109	91	109
11	90	110	90	110
12	89	111	89	111
13	88	112	88	112
14	87	113	87	113
15	86	114	86	114
16	85	115	85	115
17	84	116	84	116
18	83	117	83	117
19	82	118	82	118
20	81	119	81	119

6. Confirm that all data received from the BCI board was transferred with 100% fidelity to the Xively database
7. If the data sent was the same as the data received, then passes the test

It is important to note that because the BCI pulse oximeter could not gather data, the test was run on simulated SpO_2 and pulse rate data. Since this is a separate process from the pulse oximeter, this test still passes with 100% fidelity of data from the GSM Shield, over the GSM network, to the Xively database.

9.2 Web Client Functionality

Testing is also needed to verify that the Django web framework is appropriating the right data structures to the specified parts of the web page. A simple latency test will be conducted in reference to each file that is uploaded to the web client through Django static template files.

File Latency Tests

The purpose of this test is to observe the latency period of each source file that is used on the interface. With the results of this test, coding can be improved or debugged to maximize efficiency and cut down on latency periods to load the web page faster. The files being tested are:

- Bootstrap.css – styling of the web interface
- Bootstrap.min.js – interactive functions on the interface (buttons)
- Jquery-latest.min.js – JavaScript library allowing for animation, event handling
- D3.v3.js – data visualization (real-time graph)
- /html page – actual web domain

1. Run the command prompt **python manage.py runserver $IP: $PORT** to deploy the website on the developmental server.
2. If there are any errors validating models, go back and check file directories within python and make sure **mysite.settings** exists.
3. If the command successfully executes, navigate to **http://mht-c9-jbuckner.c9.io** in web browser to validate server operation.
4. In the Chrome Debug toolbar, under network, record the latency times for each file as the page refreshes 5 times.

		Trial 1	Trial 2	Trial 3	Trial 4	Trial 5	
File	**File Size (KB)**	**Latency time (ms)**					**Average**
bootstrap.css	117.08	73	114	78	684	86	207
bootstrap.min.js	27.07	305	192	145	578	106	265.2
jquery-latest.min.js	90.92	210	103	142	1164	83	340.4
d3.v3.js	316.48	210	70	136	659	87	232.4
/html	27.23	207	205	229	754	353	349.6
Total load time:							**1394.6**

This test will first verify that all static files that are referenced in Django's template layer are correctly referenced and uploaded. The latency times are recorded to evaluate average latency per file. The file that took the most time to load (349.6) was the actual html web domain. It is also worth mentioning that trial 4 times were significantly larger than other trial times. This may have been due to the lag of internet connection where this test was conducted. The total time required to load the whole web page is, on average, 1394.6 milliseconds or **1.39 seconds**. This test verifies that the website loads relatively quickly, which is a criteria that was suggested in the physician interviews. However one thing to note is that the data being transmitted is for one patient, Henry Winthrop. A latency test with a larger data transaction will be conducted in the next semester, with an expectation for these latency periods to increase.

10.0 Future Work

The POGO team has reached many significant milestones throughout the lifecycle of product development, but many more lay ahead prior to real world implementation of the POGO solution. Specific to the POGO device and the web application, clear incremental product development steps are outlined in the subsections below.

10.1 POGO

The first objective of future work is to resolve issues with gathering accurate data using the current pulse oximetry setup. The board can tell if the sensor is plugged in and will respond to user input that can change modes, but it will not output any valid sensor readings. The processing chip outputs data packages, but the data packages are the default bit pattern sent when the board is not receiving any data from the sensor, even though the LEDs are switching normally. Since it is reasonable to assume that the board is working correctly, this leads to three hypotheses. One, the sensor is broken (photodiode is not responding to changes in light, or the connection between the photodiode and the DB9 output is broken for the photodiode but not the LEDs). Two, the connection between the DB9 and the input to the BCI board is broken. And three, there is an issue between the input and output of the BCI board. To test hypothesis one, a multimeter can be hooked up to the LED DB9 pins with the sensor on (flashing on and off) and the tester would look for changes in voltage. If there were no changes in voltage, then the sensor is broken and the team should request a free replacement or invest in a new one. To test hypothesis two, do the hypothesis one test, then if the sensor is working, check to see if the same change in voltage can be observed on the input pins of the BCI board. If there is a loss of signal, then the connection between the sensor and board should be redone. To test hypothesis three, the tester should read professor Webster's book on pulse oximetry and follow his testing procedures

on the board. Another option is to request a new board since they are guaranteed to work by BCI. This is the final hurdle to a fully functional POGO device.

Populating the board with the GSM shield is the final step of POGO 2.0, but once all of the connections are tested and proved correct, further miniaturization of the circuit will be done. In the end, miniaturizing our POGO processing system benefits the project by minimize energy consumption. The final product will be encapsulated in a durable and waterproof casing to protect the device from normal wear and tear.

Even farther down the line, once the real-time data visualization website up and running, it would be beneficial to look into adding different sensors that would then also send their data through the GSM network to be viewed by our clients. In theory, the user would be able to hook up any portable sensor as long as we had the correct processing chips and decoding algorithms.

10.2 Database Layer

The goal of future development on the database layer is to improve web application performance by further separating the background data aggregation process from the core transactional functionality needed to serve data and files to active end users. Therefore, a new database workflow is proposed to support the existing database schema, add flexibility and scalability to this application, and to provide high level insights to healthcare systems and hospitals.

At the core of this new database workflow is Hadoop; an open source Apache framework for storage and scale processing of data. Hadoop is built on HDFS (Hadoop distributed file system), a way to store files across multiple machines, and Mapreduce, a method of parallel processing by breaking and distributing data in small pieces. In order to utilize this system, data would first be collected from the POGO device using Flume, a Hadoop tool for aggregating large amounts of log data. Flume was designed with the capability to collect near real-time data from mobile devices like the POGO, and supports our current event-driven transmission protocol. After being collected into Flume, the data is serialized by a Hadoop distribution called Avro, where it can finally be stored in an unstructured, open source, column-oriented database called HBase. From HBase, data can be queried and analyzed using Hive, an SQL like data access language for Hadoop, and Pig, a Python like Hadoop data querying and scripting language. Finally, because the unstructured architecture of HBase and Hadoop are not immediately compatible with the structured schema of the relational PostgreSQL database, the Hadoop distribution Sqoop must be used to reformat data and pipeline it into the transactional database.

While the addition of this data architecture adds complexity to the current application design, the benefits of utilizing these tools are significant. First, this fully separates and parallelizes the POGO data collection process from the process of serving patient data to the web client. This allows the PostgreSQL database performance to be optimized by only serving client content, responding to client requests, and receiving updates when needed. In doing this, PostgreSQL would only need to store the most recent 24 hours to 1 week of data, while the rest

would remain in Hadoop. Furthermore, the Hadoop database is better designed to serve as a permanent patient data archive, and has many times the storage capacity of a relational database like PostgreSQL. In addition, as a result of the flexible and unstructured nature of Hadoop and HBase, new physiological signals and new devices could more easily connect and become compatible with our client facing application. This is because data schema would not need to be predefined as they do in SQL databases, as Hadoop can readily handle new types of data in new formats. This would allow for the generation of new SQL schema based on the structure based on the format of new types of data, rather than creating the need to plan the database schema ahead of time as we seek to collect new types of data from new devices.

This also creates features that the client strongly expressed interested in. Most importantly, the Hadoop archival system is more conducive to producing monthly, quarterly or annual reports that outline a hospitals performance in managing all of their network CHF and COPD patients as a population. Especially in light of the trend towards hospital system consolidation, the client sees great value in providing Hospitals with the capability to compare this performance between multiple locations, allowing the results-based financial incentives for their staff, and enabling administrators to more accurately track ‘best practices’ for outpatient management. In addition Physicians could query an anonymized version of the Hadoop POGO data archive as a whole, allowing for the data-driven testing of various preventative treatment hypotheses they may have.

10.3 Application Layer

As of now, the Django web application is connected to the PostgreSQL database via an AJAX web socket connection, which allows asynchronous data transmission. However, the other end of Django, the web client connection, does not have the same implementation. In the next semester, improvements to the application layer will focus on creating a web socket application between the web client and Django, to allow for total asynchronous data transmission. Additionally, the python script written in the data layer of Django needs to be improved and optimized to be able to parse different data parameters from the PostgreSQL database such as time stamp and patient ID.

10.4 Interface Layer

After meeting with a few physicians, a home page and personal patient page were created to meet their desired requirements. Additional physicians, nurses, physical assistants and nurse practitioners will be surveyed regarding their opinion on the interface and usability of webpage design. It was conveyed by the physicians that actual M.D.’s wouldn’t be viewing the website as much as their care team, consisting of the previously mentioned positions. Additionally, a personal page for patients to view themselves will be designed at the request of our client. Dr. Robertson expressed the value of patients becoming interested in their own health and well-

being. The ability for the patients to view their own data would allow them to be more engaged and involved with their state of health to hopefully promote wellness. There are many more portions of the website that still need to be incorporated into our website, such as an administration log in system and physician threshold settings, to name a few, which will all need to be wrapped with Bootstrap code in order to appear in an organized, well designed fashion for the user to see and interact with. I hypothesize that the final user interface will be changed and polished multiple times to create the most user friendly design that has the ability to be widely used and understood by the variety of people who will be using the application.

10.5 Conclusion

In the bigger picture, the POGO team has additional goals that can greatly maximize the impact of this device and solution. First, an institutional review board application must be compiled and filed prior to launching a pilot test in collaboration with the UW Department of Anesthesiology. Next, FDA 510K class II approval is needed in order to bring the device to market or facilitate any commercial distribution. This process involves completing a 24 page acceptance checklist which demonstrates substantial equivalence to predicate ear oximeter devices. Finally, the team is also working in partnership with the UW Law & Entrepreneurship clinic to secure intellectual property on the overall device and working process. Once this is in place, the team plans to participate in the Qualcomm Wireless Innovation Prize, the Burrell Business Plan, the Tong Design Awards, and the Perkins Coie innovative minds competitions.

In conclusion, the team is strongly motivated to finalize a working prototype, because the POGO device has the potential to positively impact the lives of thousands of chronically ill patients around the globe. By enabling continuous remote monitoring through unparalleled convenience, a detailed longitudinal data set can be aggregated on the aforementioned patients. With more quantitative information than ever, analysis can be performed to better understand the progression of these diseases, and even to uncover previously unknown indicators of oncoming cardiovascular decompensation.

11 Acknowledgments

Our team would like to acknowledge the following individuals for their encouragement, support, wisdom, and time spent helping this project progress. Without these individuals, the level of success we have achieved would not be possible:

- Dr. Fred Robertson
- Dr. Amit Nimunkar
- Brandon Jonen
- Jared Buckner
- Geoff Cohn

References

1. Oximetry.org. (2002, September 10). Principles of pulse oximetry technology. Retrieved from http://www.oximetry.org/pulseox/principles.htm
2. Bailey, J., Fecteau, M., & Pendleton, N. (2008). Wireless pulse oximeter. Informally published manuscript, Worcester Polytechnic Institute, Retrieved from http://www.wpi.edu/Pubs/E-project/Available/E-project-042408-101301/unrestricted/WPO_MQP-Final_04242008.pdf
3. Masip, J. (2012). Pulse Oximetry in the Diagnosis of Acute Heart Failure. ScienceDirect.com. Retrieved December 1, 2012, from http://www.sciencedirect.com/science/article/pii/S1885585712001570
4. United States. Department of Health and Human Services.Telehealth. 2012. Web. <http://www.hrsa.gov/ruralhealth/about/telehealth/>.
5. United Kingdom. Department of Health. Whole System Demonstrator Programme. London: 2011. Web.
6. Holovaty, A., & Kaplan-Moss, J. (2009). The Django Book. Retrieved from http://www.djangobook.com/en/2.0/chapter05.html
7. Haydon, T., The ABCs of Pulse Oximetry. HomeCare. Retrieved from http://homecaremag.com/senior-care-products/abcs-pulse-oximetry
8. Nonin Medical Inc. - AvantÂ® 9700 Tabletop Pulse Oximeter. (n.d.). *Nonin Medical Inc.*. Retrieved October 22, 2012
9. W3, S. (2012). *Ajax introduction*. Retrieved from http://www.w3schools.com/ajax/ajax_intro.asp
10. Center for Medicare & Medicaid, (2012). *Readmissions reduction program*. Retrieved from website: Readmissions Reduction Program

Appendix
A
Product Design Specifications

Function:

This pulse oxitelemetry device will automatically collect and transmit patient's blood oxygen saturation data from any location with 3G cellular coverage, to one or more physicians simultaneously. The device will be interfaced with a front end system, which will allow for the pulse oximeter data to be stored and viewed in real-time on a web interface. In doing so, the device improves patient quality of life by providing freedom of mobility, a hands-free lifestyle, and reducing hospital readmissions.

Client requirements:

Device:
- Wireless transmission from device to base station at predetermined intervals
- Transmission of real, pulse oximetric data from device to web client
- Comfortable design that will not burden day to day activities
- Battery life beyond 1 week for discontinuous monitoring
- Ability to customize data collection intervals and signal threshold notifications

Front End System:
- Ability to store large amount of patient biometric data
- Synchronized with patient medical records
- Real-time data visualization
- Intuitive user interface for ease of patient health assessment
- User adjusted alarm set points
- Alert notification at pre-determined sp02 points

Design requirements:

1. Physical and Operational Characteristics

a. *Performance requirements*: Primarily 24/7 monitoring, during day-to-day activities and while sleeping. Monitoring will consist of wireless signal transmission from the device to the cellular network and vice versa. Clinical and ambulatory settings would also be desirable.

b. *Safety*: The thermal state of the device cannot cause discomfort to the patient. Patients cannot be exposed to any harmful currents or voltages from the device. The RF exposure guidelines will be taken into account. Waterproofing the device to limit the likelihood of these events is strongly preferred. It needs to be thoroughly sterilized. Safety warnings will be included and Continua Healthcare Alliance standards will be considered.

c. *Accuracy and Reliability*: Precision and accuracy should very closely resemble the signal outputs of contemporary pulse oximetry devices. A specific signal tolerance from the

wireless output relative to the wired output will be determined.

d. *Life in Service*: Signals must be transmitted by the device at least every 15 minutes, 24 hours a day, 365 days per year. Battery life must last longer than one week supporting these transmission intervals.

e. *Shelf Life*: Shelf life and life cycle of usage should be a minimum of 1 year.

f. *Operating Environment*: The device should not be exposed to temperature ranges, pressure ranges, humidity, shock loading, dirt or dust, corrosion from fluids, noise levels, insects, or vibration beyond those of clinical outpatients. Therefore the device should be encased.

g. *Ergonomics*: The device usages will be restricted to the heights, reach, forces, and operation torques standard to clinical outpatients. The user interface should be easy to use and provide the most important data in an efficient web layout.

h. *Size*: Device size and weight will ideally be comparable to or smaller than standard hearing aids, in order to fit comfortably on the ear to allow for minimal lifestyle disruption.

i. *Materials*: Any materials used cannot irritate skin, or be functionally disrupted by bodily fluids and oils.

j. *Aesthetics*, *Appearance, and Finish*: The device should be as close to the patients skin color as possible, in shape that snugly fits behind the ear, with a smooth, comfortable, soft texture and finish. The user interface will be built with bootstrap, which provides modern template styling for an aesthetically pleasing design. The interface will also employ several human factors and ergonomics (HFE) principals in order to achieve an efficient, ergonomic design.

2. Production Characteristics

a. *Quantity*: 1.

b. *Target Product Cost*: Less than $100.00 to purchase, manufacture and distribute each device.

3. Miscellaneous

a. *Standards and Specifications*: FDA approval is required, IEEE wireless transmission certification is beneficial, and the Continua Healthcare Alliance certification is also beneficial. Before trials, IRB approval will be requested.

b. *Patient-related concerns*: Device will need to be sterilized on a monthly basis. Unprocessed patient pulse oximetry frequency responses must be transmitted over a secure network.

c. *Competition*: Masimo, Nonin, and Phillips pulse oximeter