Faculty of Engineering and Applied Science

ENGR 4941U Capstone Systems Design for ECSE II

Remote and Mobile Healthcare System for Home Care

R5: Final Engineering Report

Team Members:
Mamun Hossain, 100553073
Gobikah Balaruban, 100742539
Manreet Kaur, 100766207
Aaditya Rajput, 100622434

Faculty Advisor: Dr. Mikael Eklund
Capstone Coordinator (Fall 2022 and Winter 2023): Dr. Vijay Sood

Executive Summary

This report is intended to summarize the engineering process and events of the overall timeline of our fourth year Capstone project as Group 12. Titled “Remote Mobile Healthcare System”, our project involves creating a device that would accurately take medical diagnostics and then store that information in a database that would then be able to be accessed remotely via either a webpage or a mobile application. The way this will be done is through the implementation of using Arduino hardware as well as initializing a cloud hosting service via Digital Ocean, and storing the readings in a backend database using MySQL and accessing any of the information required with the creation of an Android application using Android Studio.

The purpose of this capstone project is to develop a platform that offers remote medical services, especially for people living in remote or underprivileged areas. Remote Patient Monitoring (RPM) is a system that has gained importance, as it can provide valuable health insights to patients at their homes, reduce emergency visits, and saves hospital stay costs.

The contents of this report will include an in-depth overview of all the technical and general aspects that pertained to the creation of this project. This includes all of the various problem identifications, problem statements, background research, engineering requirements, design components, and several more factors. We will also be highlighting the various issues that were encountered throughout the development and testing of this project, alongside the troubleshooting that was required in order to alleviate the issues. The project’s timeline ranged from the beginning of Fall 2022 semester to the end of Winter 2023 semester. Deliverables for the final semester includes 5 engineering reports, with this report being the fifth report, a video demo presentation of our project, 5 biweekly activity logs pertaining to our process in regards to the project, as well as a poster to visually demonstrate the contents of our project.

Table of Contents

List of Figures

List of Tables

1. R1: Project Identification, Research and Requirements Specification

1.1 Problem Identification

Over the past two years , COVID-19 has increased the burden on healthcare facilities globally not only because of its deadly nature but also because of the overwhelming number of patients[1]. With the focus being shifted on the COVID patients , non-COVID patients were delayed or denied healthcare during the pandemic either due to the limited available resources or staff[2]. Many chronically ill patients avoided hospitals during pandemic because of fear of getting in contact with the COVID- 19 virus [3]. As a result, many chronically ill patients were mostly left with two options, either risk themselves to covid or neglect their regular healthcare . In some cases patients did not even have the first option either because of the mandatory isolation or because of the hospital restrictions.

The population anxiety and isolation as a result of the pandemic eventually changed people's dietary and physical habits . This further aggravated chronic disease and psychological stress[4]. Also most of the population screening programmers including cancer screening were either deferred or neglected during pandemic which resulted in avoidable deaths [5].

While the remote healthcare system has been in use since 1960'[6], its benefits started gaining limelight during the pandemic. The remote healthcare monitoring system also known as the remote patient monitoring(RPM) is beneficial for health care during pandemics as it can provide treatment and care to patients at their home. Also , remote patient monitoring will limit the time a patient stays at hospital , hence helps in managing the healthcare limited resources[7]. Considering the COVID 19 undetermined length of stay, the use of remote monitoring healthcare systems will eventually be beneficial in upcoming days .

Beyond pandemic , remote patient monitoring reduces the risk of any disease transmission, reduces the chronically ill patients regular emergency departments visits . RPM also helps in saving the hospital stay cost. With the real time monitoring of the patient , doctors can access the patients data easily and the chances of the data accuracy increases[8]. Sometimes patients forget to convey some important information to their doctor due to several factors such as dementia . In these situations, RPM can be very beneficial as it improves the clinical insights on the condition of patients which will eventually result in providing better care to the patient[9].

1.2 Background and Research Review

In recent years , the use of various technologies including sensors , GPS to measure the quality of life and environment has become very prevalent . The fusion approach to connect various devices together through sensors has gained more importance recently among the other technologies[10]. The Internet of things(IOT) allows physical devices like cars , watches , kitchen appliances etc to connect . communication and share data among each other with the help of sensors and other technologies like cloud computing[11].

Remote patient monitoring is an emerging new healthcare delivery empowered by IOT which monitors the patient's real time conditions outside the hospital with the use of technology, sensors and software [12]. Remote patient monitoring targets the patients with chronic disease, mobility issues , elderly patients and patients with limited health care, for example rural people. It is better to monitor all of these patients' conditions continuously [13]. RPM uses specific technology and physical sensors to read , collect and transmit patients data to a central board like a cloud connectivity which can be accessed by the healthcare practitioners .

RPM components are similar to various modules of IOT setup. Personal Monitoring Device, Patient-side mobile application , Cloud Database and Hospital side wide Application are some major components of an RPM setup. Personal Monitoring devices are usually equipped with bluetooth modules and these are responsible for collecting relevant data from the patient all the time. These devices track all kinds of health factors, for example blood pressure , electrocardiogram(ECG) etc . These devices then further send the collected data to the patient's healthcare provider. Patient-side mobile application , another component of RPM, is a mobile app which must be compatible with the Bluetooth Low Energy (BLE) data exchange networks . This mobile app is responsible for sending patient’s data from sensors to the healthcare clinic . These apps must be user friendly and can be equipped with resources like video calling, messaging facility , booking appointments , setting medical reminders etc . The third major component of RPM is the cloud database which is responsible for holding the raw data transferred from the patient mobile and distinguishing this data into labeled, manageable clusters . Hospital side wide Application is another major component of RPM which is a web app that is responsible for displaying the collected patients data. [14]

Remote Patients monitoring system has many advantages especially in this fast growing population with increased health complications. Enhanced accessibility is one of the critical advantages of RPM as doctors will access the patient’s data quickly . Patients no longer have to miss their routine checkup , wait for long to get their doctor’s appointment and less travel cost will eventually help to cut their medical cost . Also healthcare workers will now longer have to settle with less patient data and communication with patients will become easy . Apart from this, healthcare workers now have precise and accurate data which helps them in medication management . The response time for an emergency will now be less as the healthcare workers will be able to know the cause of the problem quickly , hence can better handle the situation.[15]

Blood Pressure monitor, weight monitor , Blood glucose monitor , Spirometer, cardiac monitors, apnea monitors,audiometers,breathing frequency monitors, electroencephalographs (EEGs) , electrocardiographs (ECGs), electronic thermometers, and electronic stethoscopes are some of the remote patient monitoring devices that are available in today's market [13].

1.3 Design Process

The main objective that we will be accomplish is to be able to create a portable device that will have as many sensors as possible to retrieve information from the user that will store that information to the backend server of the potential web server or mobile application that has a clean and accessible UI to those who have access to it. The user will have access to basic information such as heartbeat BPM and more listed below in the stakeholder requirements, but will need to access their ECG readings from the professional that has overseen the medical records of the user.

In order to create a solution to the problem identified in the problem identification, our team must view the problem from different factors and to determine the best system that should be built, while ensuring that the device has a high degree of scalability, reliability, affordability, and adaptability and ease of access for those that require a healthcare system for Home Care.

This report will focus on the components required to achieve a functioning remote and mobile healthcare system for home care, and the proper design steps to easily manage and create the system. The proposed solution will include various sensors, different types of microcontrollers, display units, and a backend system that will act as a server to store the information that will be retrieved from the aforementioned components. This report will describe the interactions between the components and how they will achieve the goal.

Due to both the time and budget constraint and the availability of the hardware components that would be required to build the system, we will be designing a device that would allow for the hardware to wirelessly connect to a backend server to send and collect electrocardiogram signals via the sensors to be sent over a server and allow for various interactions between the hardware and software components in the overall system. This requires the utilization of a strong understanding of electronic components and software adequacy and will require the usage of understanding how embedded boards can be capable of running a server to retrieve and store data over a network, and how the low level components such as the display unit and the sensor will be used for the end user.

To design this system, the approach that we will need to take for the design process is an iterative process with our group finding multiple test subjects and testing to refine our model as required. To add on, our design methodology will include the identification of the individual components and subsystems and how they will be incorporated in the design as a whole. In doing so, we will be supplementing a thorough identification of the required interfaces, a high standard of documentation to understand how the data is being sent over the various data transferring capabilities, and the degree of abstraction to understand each layer of the system architecture.

Four our design process, we will partake in an agile approach to the project. Because of the scalability and flexibility during the agile process, we will be able to successfully complete our tasks with the provided architecture below, as well as be able to make predictions based on how much time it will take to implement a certain feature. The agile process will also have a high level of reliability as it will have the ability to fit all the needs that will be required upon us.

The aforementioned iterative approach will explain the development of the model and the design process, which in turn will indicate that we will need to constantly test while making adjustments as they are needed during the development process. During this time, ensuring each subsystem is working at the top level of functionality as per the specs before the data is able to be sent to the server to hold the patient information. This in turn will allow the project to progress without any issues and delays.

The design process will begin with understanding and the identification of the multiple components and interactions of the components in the subsystem. In the tables below, we have identified two separate projects to propose as preliminary designs (Table 3-1), (Table 3-2).

Table 1.3.1 Component Design

Component Name	Description
Server Unit (Raspberry Pi Model B)	The Raspberry Pi Model B will be the Server computer that will be the brain of the system. It will be connected to the entire system via the provided GPIO pins. It will be accessible as a User Interface (UI) server system that will have sensors connected to it via the GPIO pins and using a MQTT process will provide information receiving the patient's electrocargiongram signals and the end user's heartbeat. It is also responsible to store the patients information that can be later accessed by the administrator and to the health care professionals that

	can override the administrator commands to view the stored data.
Display Unit	The display unit is responsible for showing that the device is functioning and will display the time and battery life to the end user.
ECG Unit (Polar H10 (Heart rate sensor))	The heart rate monitor will act as a way to monitor and to receive the information from the end user, by accessing the Raspberry Pi's GPIO pins. It will keep track of the heart rate fluctuation.
End User Unit (BPM Reader)	The user will be able to access their BPM
Monitoring System Unit	The Monitoring System Unit will be where all the information and the server will have all its main access to. It will be connected to every component in the system. It will be responsible to store all the end user information as required by the individual.
PPG Unit (Pulse Sensor + MCP3008 ADC IC)	The PPG unit will retrieve the data from the pulse sensor and pass it along to the MCP3008 via an analog signal and then will be able to transfer that information of the ECG to the number of Hz that is being transmitted.
Oximeter Unit (Masimo RAD-8 Oximeter)	The oximeter unit will be able to monitor oxygen level and heart rate that will have its information stored and assessed by the healthcare professional to check for any abnormalities in the reading that gets stored into the back end server.

1.3.2 Component Design 2

Component Name	**Description**
Server Unit (Arduino Mega*)*	The Arduino will act as a high powered computer that will be able to connect to the server via SSH protocol and send the data retrieved from the device to the cloud server. It will be accessible via a User Interface (UI) server system that will have the cloud script be running within the Arduino IDE to send information from the device to the allocated cloud server receiving the users ECG reading, PPG, BPM, and blood oxygen readings. It will also be responsible for withholding the user information that can later be accessed by a healthcare professional that will have administrator rights to the system to view the stored data.

Display Unit	The display unit is responsible for showing that the device is functioning and will display the time and battery life to the end user.
ECG Unit (ECG Module AD8232)	The ECG module will act as a way to access the ECG requirements as deemed necessary by the user to retrieve the information from the device and then send that information over to the cloud server, disallowing the user to have direct access to the readings.
End User Unit (BPM Reader)	The user will be able to access their BPM
Monitoring System Unit	The Monitoring System Unit will be where all the information and the server will have all its main access to. It will be connected to every component in the system. It will be responsible to store all the end user information as required by the individual.
Photo Sensor (Photodiode LPT80A)	The Photo Sensor will allow for the display unit to interact with the user's PPG reading via a waveform pattern that will be displayed moving through an average filter.
Pulse Detector Unit (Pulse Oximeter MAX30102)	The Pulse Detector Unit will provide a high sensitivity pulse oximeter that the patient will be able to receive and acknowledge alongside a hate rate sensor.
Math Unit	The Math Unit will calculate the ECG waveform to calculate the Hz of the ECG waveform.

The software design process involves being able to identify and develop the software architecture in cohesion with the hardware design using the components required for the prototype. Below in **Figure 1.3.1,** a multi layered architecture diagram is being proposed as the main system architecture.

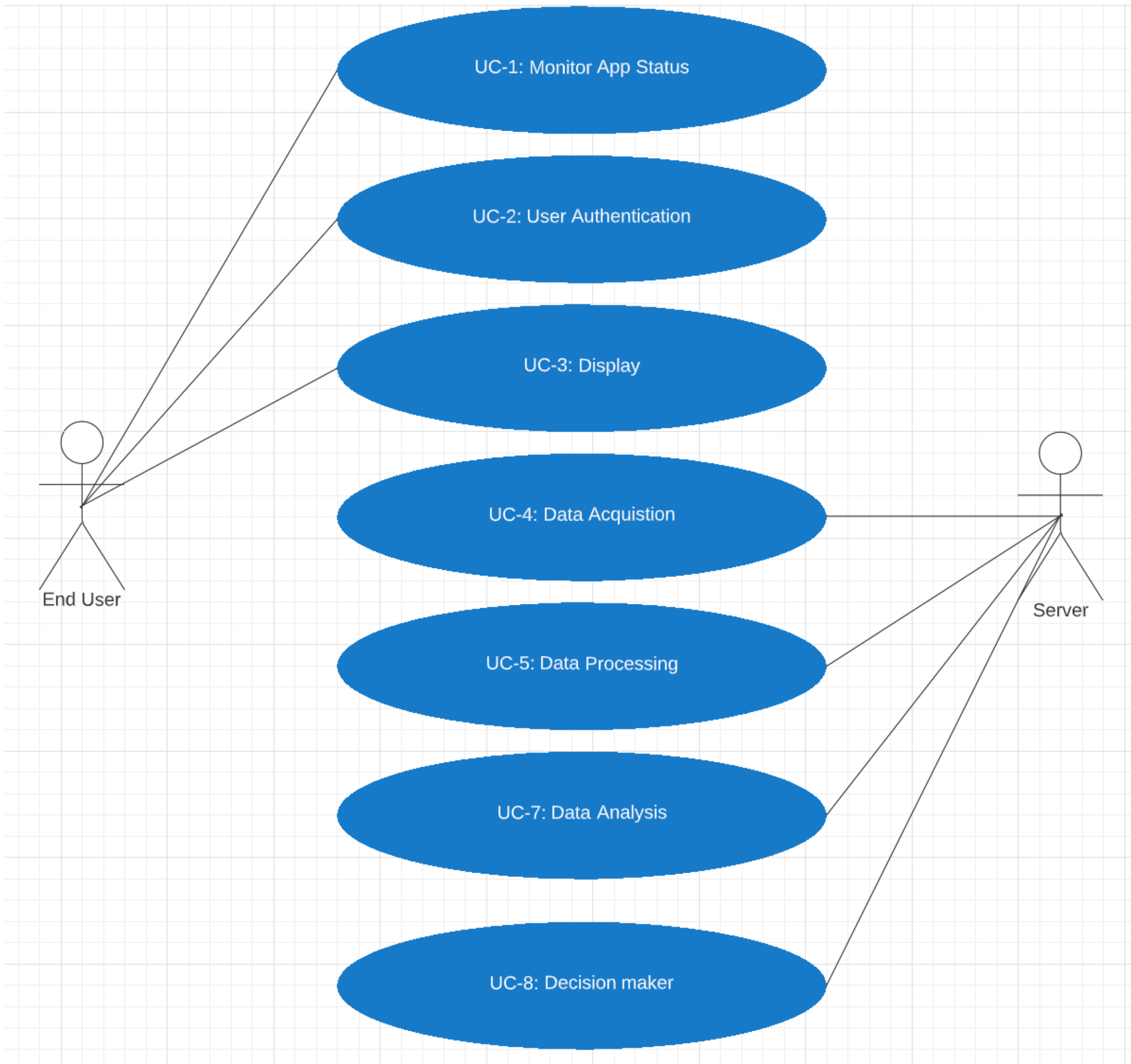


Figure 1.3.1 Software architecture in cohesion with the hardware for the Mobile and Remote HealthCare monitoring system

The progress of the project will be tracked in accordance with the following:

1. Each members' individual weekly activity report
2. Consistent testing and evaluating of our prototype in accordance with the milestones set with our group found in Table 7-1 below.
3. Weekly project progress meeting with faculty advisor Dr. Mikael Eklund
4. Weekly project progress reports with ***the*** team members.

With the constant prototyping and monitoring of our project by the team as well as the faculty advisor alongside the activity logs of each student and the project reports, this will ensure an appropriate outcome for the stakeholders for the project. THe team is responsible for coming to a unanimous decision when it comes to any measures throughout the progress of the project. Furthermore, each individual of the team will also be responsible for each of the given tasks that they are responsible for, that being but not limited to: development of the server, handling the electronics, integration of the server to the website.

1.4 User Stories/Scenarios:

1.4.1 Scenario One: Pulse Detection

User has come back after a long marathon run and notices themselves breathing faster and heavier. User presses the buttons on the device to check his heart rate after a burst of intense activity. The device displays the user's heart rate.

1.4.2 Scenario Two: Continuously Monitor your Heart Rate

User is facing irregular heartbeat patterns and is feeling a noticeable deterioration in health, the device detects the irregularity and gives them a notification notifying them of unusual/irregular heart rate trends informs them to see a doctor as soon as possible.

1.4.3 Scenario Three: Data Storage to Cloud

User visits their doctor for a regular check-up. Whilst the user was using the device, the data and recorded averages are being stored and saved on a cloud server. Health care professionals can access this information when required to determine fluctuations and pinpoint when irregularities/problems began to form, allowing them to get to the root cause quicker and diagnose problems faster.

1.4.4 Scenario Four: Internet/BT compatibility with Cell-Phone/Computer

User wants more in depth analysis of the recorded data which is not accessible from the medical monitoring system's interface. To do this the user is required to use an external device such as a cell phone or a computer.

1.4.5 Scenario Five: Data Storage to Cloud

Enduser has started living a healthier lifestyle in the past year, they are eating clean, working out daily and have quit smoking. To track their progress and make sure it is improving their cardiovascular system the user wants to compare their prior resting heart rate/ active heart rates with their new found heart rates. The user will need their health professional to retrieve their older data and compare their progress with the new ones. This information is not directly accessible to the user to prevent anxiety and stress from self-diagnosis.

1.4.6 Use Cases

Below are the use cases that will be considered for during the design process of the Medical Monitoring System (MMS). The intended design of the system, system goals, major features and system complexity will all be addressed in this section.

Table 1.4.6.1 – Heart Rate Check on Medical Monitoring System

Use Case ID	**UC-MMS-HRC-1**
Use Case	User wants to check their current Heart Rate
Description	User presses a button on the System's Interface which will then take the user to a screen where there heart rate is displayed in Beats per minute.
Actors	1. User 2. Heart Rate Sensor 3. MMS Display 4. Database System
Pre-Condition	System must be turned on and running.
Post-Condition	After the Heart Rate is requested, the display will show the Heart Rate BPM and allow the user to go back to the home page.

Use Case Component	Step	Steps
Heart Rate Sensor	1	Heart Rate is continuously being measured when on (if applicable, averages are recorded, minutely/hourly)
MMS Display	2	Button is clicked to request the Heart Rate to be shown on screen.
MMS Display	3	Fetch the current heart rate and print it on the Display.
MMS Display	4	Allow user to either re-attempt the testing or let the System go back in to resting mode.
Extensions		

Heart Rate not Detected	1a	Notify user via user interface to try again as sensor did not detect a BMP.

Table 1.4.6.2 – Heart Rate Irregular trend e-mail.

Use Case ID	**UC-MMS-HRI-2**
Use Case	Doctor is notified of the patients health status.
Description	User's heart rate has become too fast or too slow which than their regular historical data. This is detected by the system and an email will be sent to the user's doctor.
Actors	1. User (Doctor) 2. User (Patient) 3. Database System
Pre-Condition	System must be turned on and doctor's email is already saved.
Post-Condition	-

Use Case Component	**Step**	**Steps**
Heart Rate Sensor	1	Heart Rate is continuously being measured when on (if applicable, averages are recorded, minutely/hourly)
MMS Display	2	Irregularity detected in the program.
Server	3	Sends an email to the Doctor.
Extensions		
Faulty Readings recorded, I.e lose connection with skin and shows a terrible inaccurate result.	1a	Backend algorithm has a certain threshold and criteria where results are discarded from the Databases. For example, during high intensity training the system recorded 5 BMP for a minute in between minutes showing 100 BPM. Clearly the 5BMP minute in that scenario is an error and should not be accounted.
Email to the Doctor failed.	1b	Send an email to the user that communications with the doctor is compromised and to input a correct email address into the system.

Table 1.4.6.3 – Internet Connectivity with Mobile Application using WIFI

Use Case ID	UC-MMS-ICM3
Use Case	IoT compatibility, system has to have ability to communicate with other devices
Description	User's wants to check the analysis done by the mobile application. The data that has been recorded by the MMS is stored and analyzed on the Mobile app.
Actors	1.User 2. Mobile Application Interface 3. Database System 4. Hosting Server
Pre-Condition	Device has to be turned on as well as the mobile application. Server has to be running and routing messages from the MMS to the Mobile Application.
Post-Condition	-

Use Case Component	Step	Steps
Heart Rate Sensor	1	Heart rate sensor records data.
Heart Rate Sensor Server Mobile App	2	Recorded data that was detected from the Sensor is routed to the Mobile application where the data is processed.
Mobile Application Interface	3	The Processed data is then displayed in a manner that is understandable to the user.
Extensions:		
Server fails	1a	The data from the Medical Monitoring System Cannot be communicated to the Mobile app due to failure on the server cause message routing problems. Returns error on the App UI.

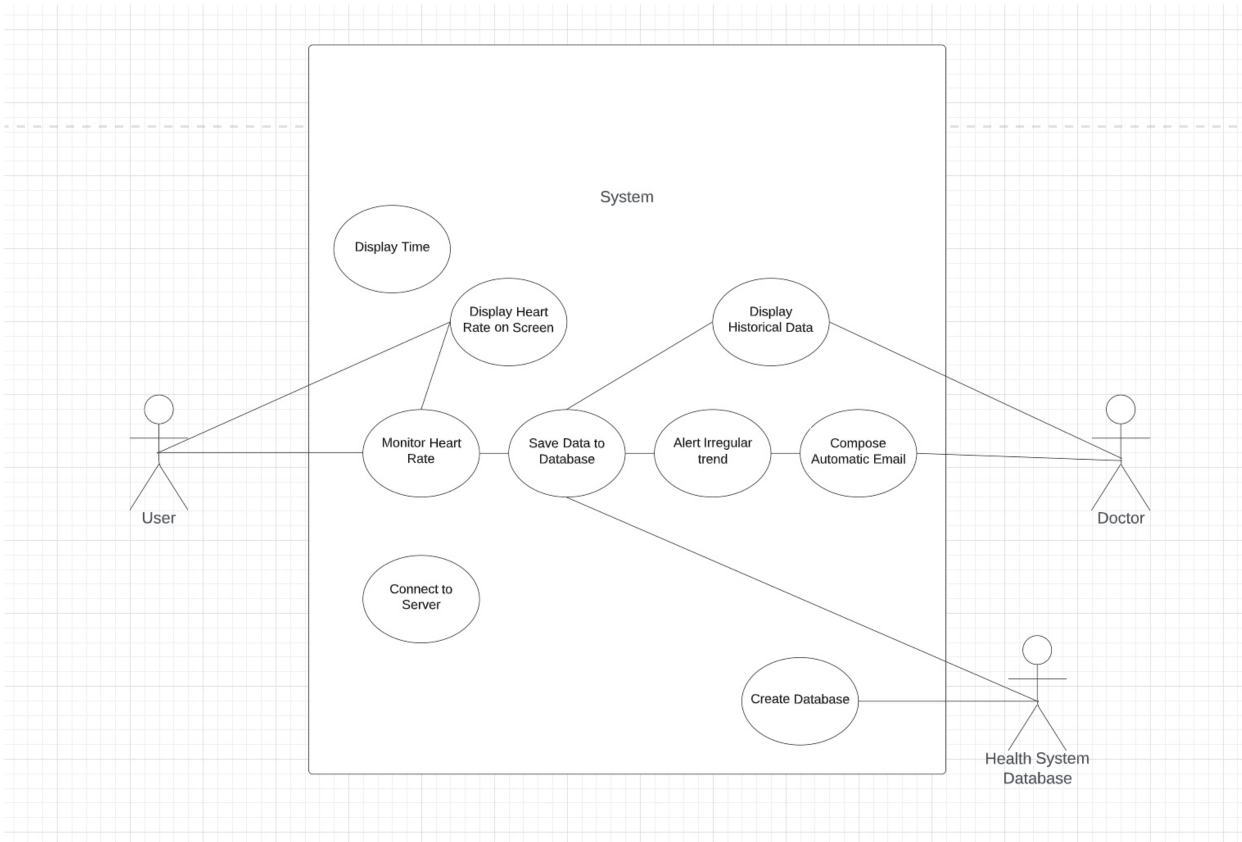


Figure 1.4.4.1 UML Chart of Use Case Scenarios

1.5 Stakeholder Requirements and Traceability Matrix

1.5.1 Stakeholder Requirements

Table 1.5.1 Stakeholder Requirements

Table 1.5.1 below includes all the stakeholder requirements that have been identified. These requirements will be traced throughout the duration of the project until completion.

Requirement ID	Requirement Description
[REQ-DIAG-1]	The mobile medical monitoring system must be able to perform diagnostics and collect

	data for measures like electrocardiogram (ECG), heart rate (in BPM), photoplethysmography (PPG), blood oxygen levels and calculating the frequency of the ECG (Hz) from users.
[REQ-STOR-1]	The application must be able to store the data from the checkups and provide a real-time analysis of the data accumulated.
[REQ-DISP-1]	The heart rate monitor system must display the results of a checkup within moments.
[REQ-DISP-2]	If any of the monitors detect any abnormalities with the user, it should notify the user immediately and advise them to visit a physician or depending on the severity of the situation to inform the user to contact emergency services.
[REQ-DISP-3]	If the audiometer detects a high decibel (dB) level, it should notify the user that the level of noise could damage their hearing.
[REQ-DATA-1]	While the data is being collected, the system should filter out any bad noise that may result in inaccurate data, such as when performing an ECG.
[REQ-DATA-2]	The user should not be able to see all the data collected from the diagnostics. Some data should only be seen by physicians.
Optional Requirement [REQ-DISP-3]	The information displayed should be in an understandable user interface that untrained users can comprehend.
Optional Requirement [REQ-STOR-2]	The collected information can be sent to a third-party for analysis, with the user's permission.
Optional Requirement [REQ-DES-1]	The device is designed to be used by a single user.

1.5.2 Traceability Matrix

The traceability matrix, Table 1.5.2, was created. It includes the requirement ID, the use case ID and acceptance ID(s) associated with that.

Table 1.5.2 Traceability Matrix

Requirement ID	Use Case ID	Acceptance ID
[REQ-DIAG-1]	[UC-MMS-HRC-1]	[AT-DIAG-1] [REQ-DISP-1] [AT-DET-1]
[REQ-DISP-2]	[UC-MMS-HRI-2]	[AT-ALM-1]
[REQ-STOR-1]	[UC-MMS-ICM3]	[AT-STOR-1]

1.6 Definition of Acceptance Tests

The acceptance test table, Table 1.6.1, identifies all the acceptance tests.

Table 1.6.1 Acceptance Tests

Acceptance Test ID	Description
[AT-DET-1]	System detects if a user is wearing the device.
[AT-DIAG-1]	System performs diagnostics on request by the user.
[AT-DISP-1]	System displays the information/results of the diagnostics after it has been performed.
[AT-ALM-1]	System alerts the user if there are any abnormalities or issues with the results of the diagnostics and informs the user to seek medical assistance.

[AT-STOR-1]	System stores the collected data and sends it to a cloud-based system.
[AT-STOR-2]	System stores the data for users to review afterwards on the mobile app. (For example: the data collected within the past six months)

1.7 Project Plan

1.7.1 Project Milestones

The table below represents the project milestones and anticipated start and end dates (**Table 1.7.1).**

Task Number	Task	Start Date (MM/DD/YYYY)	End Date (MM/DD/YYYY)
1	**Idea Research**	**09/18/2022**	**09/22/2022**
2	**Project Research**	**09/23/2022**	**10/02/2022**
3	**Design Identification**	**10/06/2022**	**10/10/2022**
4	**Capstone Report 1 Deliverable**	**10/09/2022**	**10/16/2022**
5	**Hardware acquisition**	**10/26/2022**	**11/02/2022**
6	**Hardware compilation and Prototyping**	**11/04/2022**	**11/10/2022**
7	**Software Component design and development**	**11/03/2022**	**11/07/2022**
8	**Capstone Report 2 Deliverable**	**11/02/2022**	**11/08/2022**
9	**Software Integration with Hardware**	**11/12/2022**	**11/19/2022**
10	**Integration Testing**	**11/19/2022**	**11/21/2022**
11	**Acceptance Testing**	**11/22/2022**	**11/29/2022**

12	**Project Demo**	**12/01/2022**	**12/02/2022**
13	**Project Poster**	**12/01/2022**	**12/02/2022**
14	**Final Engineering Report**	**11/25/2022**	**12/06/2022**
15	**Team Retrospective Report**	**12/01/2022**	**12/07/2022**
16	**Individual Activity Long**	**11/03/2022**	**12/12/2022**

1.7.2 Project Gantt Chart

Figure 1.7.2.1 describes the project Gantt Chart for the project's projected timeline.

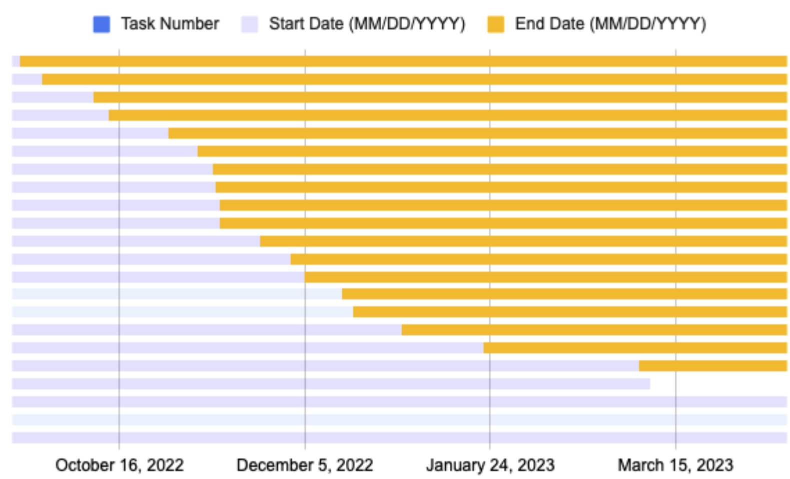


Figure 1.7.2.1 Project Gantt Chart

2. R2: Concept Generation, Conceptual Design and Prototype

2.1 Concept Generation and Analysis

2.1.1 Understanding the Problem

In our Capstone Project Report 1, the problem to be solved was identified as: The development of a hardware and software system that allowed for a remote patient monitoring service that will be created to aid chronically ill patients to have the ability to have their health monitored to decrease health risks by monitoring different health conditions (eg. heart beats per minute, blood volume changes, blood oxygen levels, blood pressure waveform, etc).

To further clarify this problem, due to the previous burden on healthcare facilities globally, which as a result ended up in population anxiety and isolation due to the COVID-19 pandemic, many dietary habits and physical habits were altered. Many people were unable to make visits to their doctor in person due to the pandemic and were unable to monitor their health status as a result. In order to prevent this lack of health monitoring data in the future, we have conducted the required research into different existing solutions to understand and verify the requirements on how to approach the project requirements definition and the development of a well-formed prototype.

To develop a hardware system that interacts with the software based system, we have identified the important requirements to allow for the development of our mobile health care device. Certain requirements will be implemented (i.e., storing the different data from the sensors appropriately), and the calculation of the data (i.e., calculating waveforms, beats per minute, blood oxygen levels) and the potential real-time monitoring of these events.

The main intent of this project is to allow the user to use various health monitoring sensors, such as ECG sensor, a heart rate sensor, a blood oxygen level sensor, and a way to collect the data from those sensors and display them in either a numerical or graphical format. The information would then be stored in a server that is passed through a cloud server.

This section will define and analyze the various components that would be required for a Remote and Mobile Healthcare System for Home Care on a conceptual basis and how they come together to form a cohesive system.

2.1.2 Existing Solutions

Remote and Mobile Healthcare Systems have been deployed by many industries amongst both the commercial industry and the healthcare industry for several different purposes. From a basic level purpose to an enterprise level requirement, Remote and Mobile Healthcare systems have been continuously evolving. Industries such as the telecommunications industry and the healthcare industry are using remote and mobile healthcare systems as a way to monitor both lifestyle habits as well as the different health risks that can be imposed on an individual. A typical remote healthcare system will consist of several different electrical components, which in turn will connect to the backend server via the cloud network that will communicate with the system as well. The typical aspects of a system of this size would involve the sensor actuation reading the inputs, sending that data to the system and then the distribution of that data to the server. Listed below are some alternative solutions to the aforementioned product at a systems level**.**

2.1.2.1 Chronisense Medical

A medical company in charge of the development of medical-grade devices that allow the monitoring of vital signs and other health factors which allow the patients and healthcare professionals to manage chronic disease without interfering with the patient's everyday activity in or outside their home [1].

The way this works is with the usage of their toolset that measures medical data, they are able to take in different output values, such as ECG and blood oxygen levels, and it can communicate with the system in real time. They use a specific sensor called the "Radial Artery Sensor" which gives live time accurate vital sign data in the format they find is needed for the reading, be it waveform or in numerical form. It is a patient friendly device that is wearable on the wrist, like a watch that presses up against the radial artery to retrieve the patient data and send it over the server.

2.1.2.2 Telehealth

With the requirement of digital information and communication technology, Telehealth has created a way to allow the access of healthcare services and being able to remotely manage the patient's healthcare from a remote distance. The way this works is that a provider will issue the patient with devices that wirelessly send data, such as blood pressure, blood sugar, and oxygen levels. They are able to do this by using web-based or mobile-based applications that can either update the data from the specific monitor in real-time or have the patient manually input the data, such as dietary logs throughout the day. They also have devices that can record data such as heart rate or even data that detects activities such as falls for dementia patients [2].

Unlike Chronisense however, the application is not as small as a wearable watch-like device, but a larger physical device that will act on behalf of the provider’s instructions, such as inputting blood pressure readings into the app manually, as Telehealth is catered to those with limited access to resources.

2.1.2.3 Wearable Devices (eg. Apple watch, Fitbit devices, Samsung watches)

With wearable devices taking the world by storm in the past decade, the applications of wearable technology have significantly evolved as well. From the initial phases of simply taking calls to being able to monitor blood oxygen levels, taking ECGs, as well as fall detection, wearable tech has become a viable form of a mobile healthcare system. With several sensors in place that sends data from the end user, these can be blood pressure monitors that send numerical data and waveform data, ECG data that sends waveform data, or different types of biosensors, which collect data that will be sent as user data over to the mobile device that is connected to the cloud for easy and in depth viewing of the data. This provides both a hardware and software interface for the required needs and monitoring purposes [3].

2.1.2.4 Remote Mobile Healthcare Concept Analysis

The remote mobile healthcare system will have two major components that will need to follow suit to be on par with the aforementioned components. In contrast to the other components, where the end user will have constant access to their own data, we will have major data be sent via a cloud server that can be accessed by the user to reduce anxiety for the patient, and they will be able to view the data as much as they deem is necessary.

2.1.3 Comparisons:

We had discussed several different systems, beginning with using a system that incorporated an Arduino system that can interact with a portal based device with an IoT aspect. The way that this would work is that an android application would be created that would have an implementation of a WiFi module that will connect to an ECG module that will be available to access wirelessly to the user using the Arduino cloud program within the IDE. With this system in place, the user will be able to access their information on a waveform every time that it is updated to the application. It will use the analog pins on the Arduino while having the ECG waveforms be sent over via Bluetooth using a serial interface. The data that is accessed here will be able to be viewed by anyone using a phone or web based application with the URL provided from the system.

A similar application that we had conducted research upon was to create a system using the Raspberry pi module and create a web based application that will allow the user to send the required input such as ECG signals whilst using a 3-lead ECG sensor. This in turn will allow for an ECG monitoring system that will be able to convert an analog signal that will be converted to a digital signal that can be sent to the embedded Raspberry Pi. The way this will work is that using an online plotting tool (ie. Plot.ly), the signal can be viewed if anyone has the programmers username and password for the mentioned online tool. The real time data will be monitored and the user will be able to see if there is any abnormality reached in the ECG signals.

A third system that we had researched about would be the incorporation of both the Arduino module as well as the Raspberry pi module. The way that this would work is that beginning with the Raspberry Pi, there will be an interface with the 3-lead ECG sensor which will be able to sample the data at the given rate we desire in our web based application. From there, the analog to digital conversions will occur for the sample rate in Hz. After that, we will be able to view the peak detection of the ECG signal in our web based application that will display the information and it will upload the analysis to the cloud. Now, to display the actual peak detection, this is where we will be incorporating the Arduino that will be able to chart the single data and give us the waveform chart, and with the waveform chart, we will also be able to create a square waveform chart as well, allowing us to have multiple readings of different requirements.

The application will be linked from the sensor devices to the IoT devices and will then relay the detailed information with the appropriate units of measurement that will wirelessly send the data over a cloud server and store the information there. In addition to this, the user will have access to some minimal data, such as beats per minute. That will give some sense of stability to both the user to ensure that the device is functioning and to the server that is collecting the data to confirm that the product is being used correctly.

With the applications in mind and the systems to narrow down, we will be using the third system as we feel that when needing to interact with the cloud server, it is ideal to use the Linux based kernel to allow access to that server seamlessly, and allow us to connect different sensors directly to the board, without needing to have any extra modules such as a Bluetooth or WiFi module as they are already integrated into the board, whilst the Arduino also will be able to assist with plotting the data into our own web server rather than needing to use a general web based application such as plotGraph.oy for example so that way the information that we record will remain confidential to the user that is using and there is no risk of any third party application taking that information.

The overall intent when creating the concept generation of the device with the cloud server is to prioritize the user friendliness and make the device with as many sensors as possible to relay as much data as possible to monitor the patients' health and ensure that they are living a healthy lifestyle, whilst monitoring the progression of their health statistics as it is being stored to the cloud server statically. This will be done through the implementation of the server that will be created through the IoT devices, such as the Raspberry Pi and Arduino Mega unit that will connect to many different sensors via the GPIO pins on the circuit board. The more sensors that the system will contain, the more types of data the system will be able to display and store. The application will consist of different screens displaying different data sources and the static data accumulated over a period of time. It should also be noted that all the connections will be done wirelessly over an unspecified cloud server. The table below displays a comprehensive list of the various sensors and devices that the application will contain. The server will be using a Raspberry Pi and Arduino system to control the data and send it back to the server via a currently undefined cloud platform. However, we have gone down the route of integrating both the Arduino unit and the Raspberry Pi because with the Arduino unit that will be connecting the sensors, and the Raspberry Pi that will be hosting the cloud server, it will leave our system to be very scalable in its design whilst having a reliable cloud server. This can be compared to an existing solution such as they have their reliable cloud servers that will store their data and be able to look at the information very clearly and not have any discrepancies in the data collection, however not only will be collecting the main data, but unlike the other solutions we will be continuously updating the data the second the user wears it rather than having to enable a certain setting to use it (eg. Apple watch "Start workout" method). Doing this will ensure that we have an accurate real time response to the data.

Table 2.1.3.1: Component Design

Component Name	Description	Justification
Server Unit (Raspberry Pi Model B)	The Raspberry Pi Model B will be the Server computer that will be the brain of the system. It will be connected to the entire system via the provided GPIO pins. It will be accessible as a User Interface (UI) server system that will have sensors connected to it via the GPIO pins and using a MQTT process will provide information receiving the patient's electrocardiogram signals and the end user's heartbeat. It is also responsible for storing the patient's information that can be later accessed by the administrator and the health care professionals who can override the administrator commands to view the stored data.	Will be able to send the information via the cloud server accessing the Raspberry Pi by enabling SSH features and connecting to IoT hub with Node.js and using the GPIO pins in unison with the various sensors
Sensor Unit (Arduino Mega)	The Arduino will act as a high powered computer that will be able to connect to the server via SSH protocol and send the data retrieved from the device to the cloud server. It will be accessible via a User Interface (UI) server system that will have the cloud script be running within the Arduino IDE to send information from the device to the allocated cloud server receiving the users ECG reading, PPG, BPM, and blood oxygen readings. It will also be responsible for withholding the user information that can later be accessed by a healthcare professional that will have administrator rights to the system to view the stored data.	Will be connecting all the sensors to the Arduino Mega unit to seamlessly power the sensors and retrieve the user data that sends it back to the Server Unit.

Display Unit	The display unit is responsible for showing that the device is functioning. The time and battery life will be displayed to the end user.	Screen will serve as a main menu to display basic statistics and ensure functionality.
ECG Unit (Pulse sensor (Heart rate sensor))	The heart rate monitor will act as a way to monitor and receive the information from the end user, by accessing the Raspberry Pi's GPIO pins. It will keep track of the heart rate fluctuation.	The sensor is able to send the beats per minute data information to show the user on the display unit and to store in the server.
End User Unit (BPM Reader)	The user will be able to access their BPM.	Connects to the display unit to ease anxiety.
Monitoring System Unit	The Monitoring System Unit will be where all the information and the server will have all its main access to. It will be connected to every component in the system. It will be responsible to store all the end user information as required by the individual.	Will allow for ease of access monitoring of the system data to ensure things are progressing and processing smoothly in a timely manner.
PPG Unit (Pulse Sensor + MCP3008 ADC IC)	The PPG unit will retrieve the data from the pulse sensor and pass it along to the MCP3008 via an analog signal and then be able to transfer that information of the PPG to the number of Hz that is being transmitted.	The sensor sends pulse data to the cloud server and not shared to the user, displaying the data via a waveform chart.
Oximeter Unit (MAX30100)	The oximeter unit will be able to monitor oxygen level and heart rate that will have its information stored and assessed by the healthcare professional to check for any abnormalities in the reading that gets stored into the back end server.	Sensor to monitor blood oxygen levels to increase functionality of hardware systems.

Using the table above, with detailed planning and taking in account the time and resources we have available to complete the project, we have decided to take select the usage of the Raspberry Pi to be able to connect into the cloud server much more efficiently whilst also use lower power consumption that will in turn increase the battery life of the system and result in longer data retrieval so there is more data that can be analyzed for the lifestyle and health standards of the patient.

2.1.4 Stakeholder Analysis

For the proposed system design, the various stakeholders and associated requirements must be considered as shown in the table below.

Table 2.1.4.1: Stakeholder Analysis

Stakeholder	Stakeholder Requirements (Non-functional and functional requirements)
Project Team Members	- The project must be finished within the specified time constraint from September 2022 to March 2023 - The project must be completed within the allocated budget specified by the coordinator, which has been budgeted to $800 - The project must have a unified approach where all the team members are able to identify the development and be confident in the final integration - The project must be completed with a high level of solution for the task at hand with an adequate grasp of the technologies used
Faculty Advisor and Course Coordinator	- The project will provide a prototype during the development period (December 2022) as well as a completed prototype by the end of the development period (March 2023) - The project must be completed within the allocated time and budget as mentioned previously
Client/Patient	- The completed prototype must be flexible to allow the user to be able to comfortably use the system - The completed prototype must be able to consume the allocated real time data that will be displayed in correspondence with the

	performance requirements, which includes being able to use the display unit to view the beats per minute as it gets updated at the specified time

2.2 Conceptual System Design

In this section, the conceptual system design covers the several concepts generated in the previous section, concept generation and analysis. The individual components will be further specified, including their responsibilities and interfaces. The behavior, structure, parameters of the components are specified in this section. The operations of the components and how they interact with each other will be illustrated with the diagrams below.

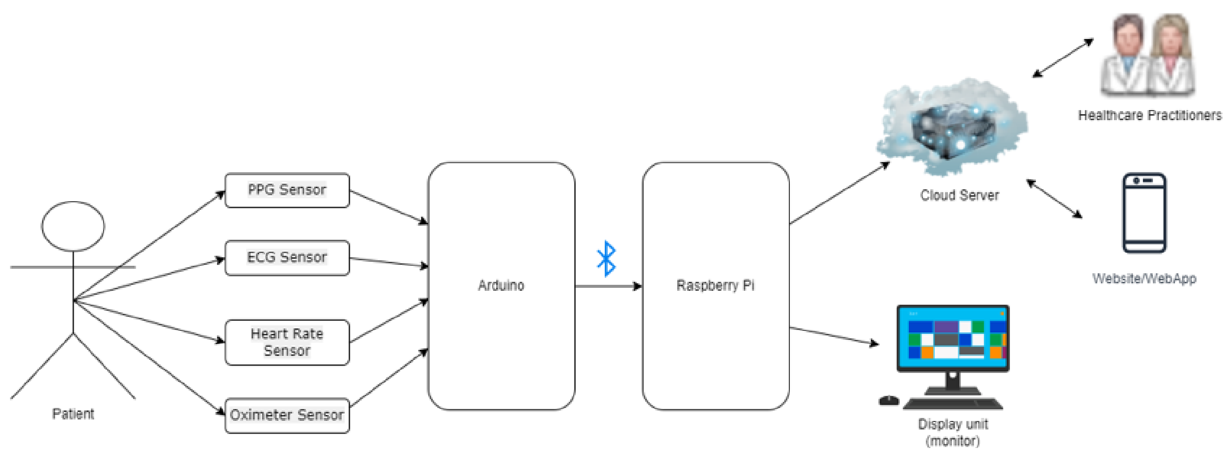


Figure 2.2.1: Overall System Design

In the above figure, it illustrates the overall system design and how the components interact with each other. Starting from the left, the user has their vitals inputted by the sensors, which is connected to the Arduino. For the Arduino to connect to the Raspberry Pi, bluetooth is used. From there, the display of the diagnostics is shown on the Display Unit, while the data is sent to the cloud server. From the cloud server, the data can be accessed and retrieved by healthcare practitioners and/or by the user accessing it from the website/mobile application.

2.2.1 Component Level Design

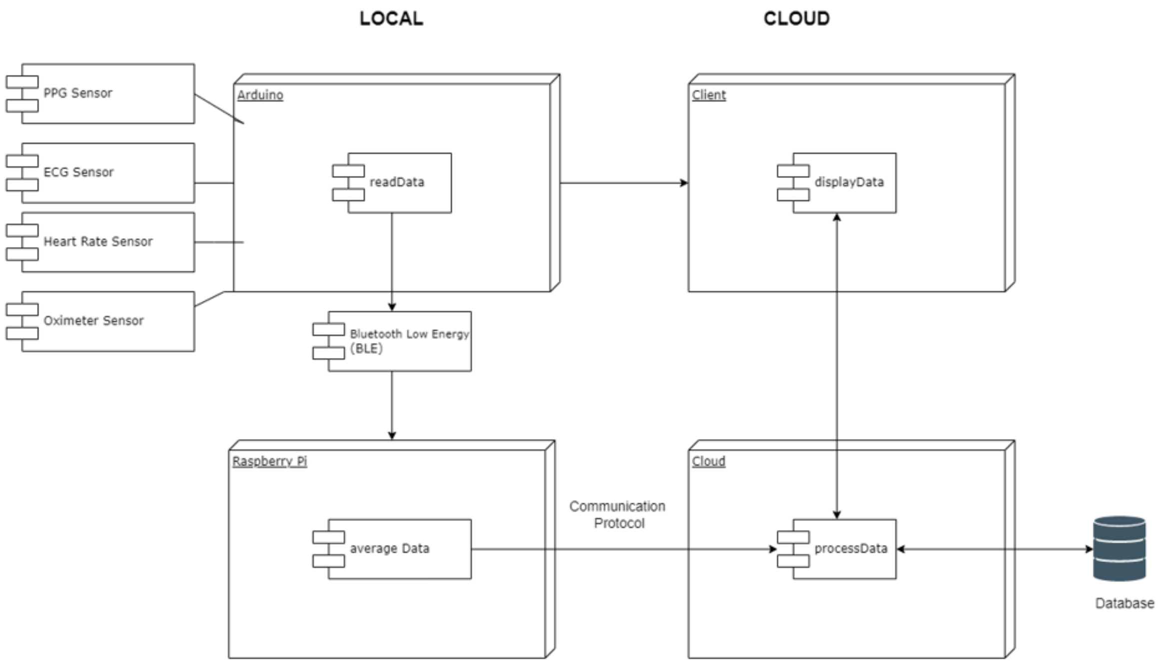


Figure 2.2.2: Component Diagram of the Overall System

The diagram in Figure 2-2 above continues with the overall system components and displays the interfaces of the components. It also shows how the various components connect with each other too. For instance, in the Arduino block, there is a readData module that reads the data from the sensors that are linked to the block on the diagram. As this diagram shows interactions between components, it also shows how the local components are connected to the cloud components. The Raspberry Pi is able to connect to the cloud via a Communication Protocol. The modules are further explained in the table below:

Table 2.2.1: Modules

Module	**Description**

readData	Arduino reads the data from the sensors periodically(for example in every 2 mins) and then concatenates them into strings . These strings are then encoded into a char array and sent to the Raspberry Pi using a bluetooth module(BLE).
averageData	Raspberry Pi then reads the data from the bluetooth module by decoding the array . The data is then added into an average function which gives the avg data (or average readings of the data). Then this averageData will be transmitted to the cloud server using REST/MQTT communication protocol.
processData	This module will be responsible for storing all the data received from the raspberry pi to the database. This module will then send data to the displayData in every 10s. Apart from this processData will be responsible for sending the requested data to the displayData(for example if the client requested the data for a particular day or time.
displayData	This will be a client facing app (like a webApp) which will display the data to the user. Patients and health practitioners will have their own login id's . There will be graphs of each sensor readings(for example ECG graph) and the current data readings .

2.2.2 Overall System Design

The Block Diagram shows how the components work together as a whole unified system. The diagram below illustrates how and where they are connected, starting from the inputs coming in from the left and the outputs on the right. Starting from the left hand size, the Arduino receives the data collected from the various sensors. The Arduino then transfers this data to the Raspberry Pi, which acts as a central hub for the data. The data can be sent to the website, which can be viewed by the user and/or healthcare professionals. It could also be sent to the digital display, for the user to view immediately. The digital display for Raspberry Pi may or may not be implemented into the system and is currently under review.

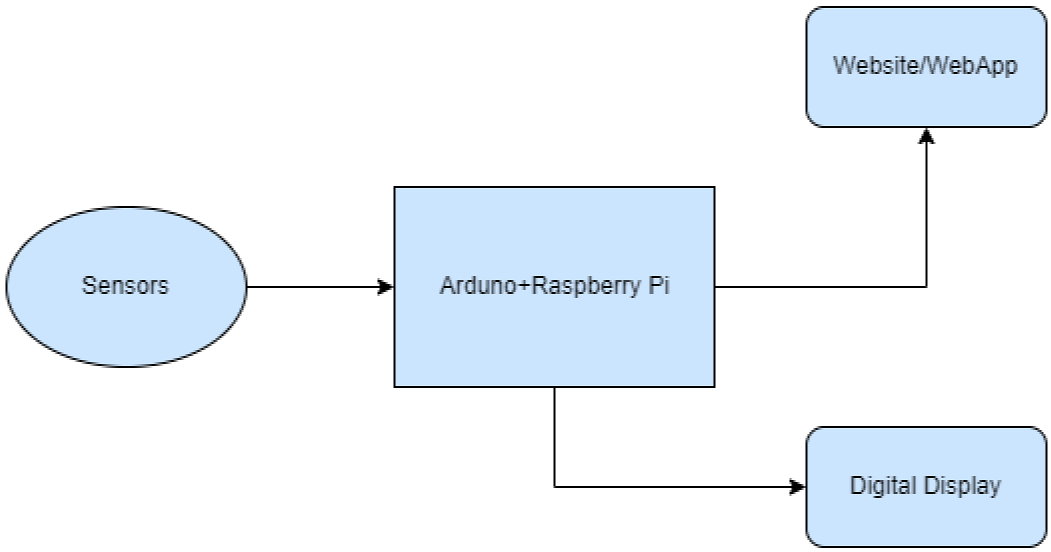


Figure 2.2.3: Block Diagram of the System

2.2.3 Activity Diagram

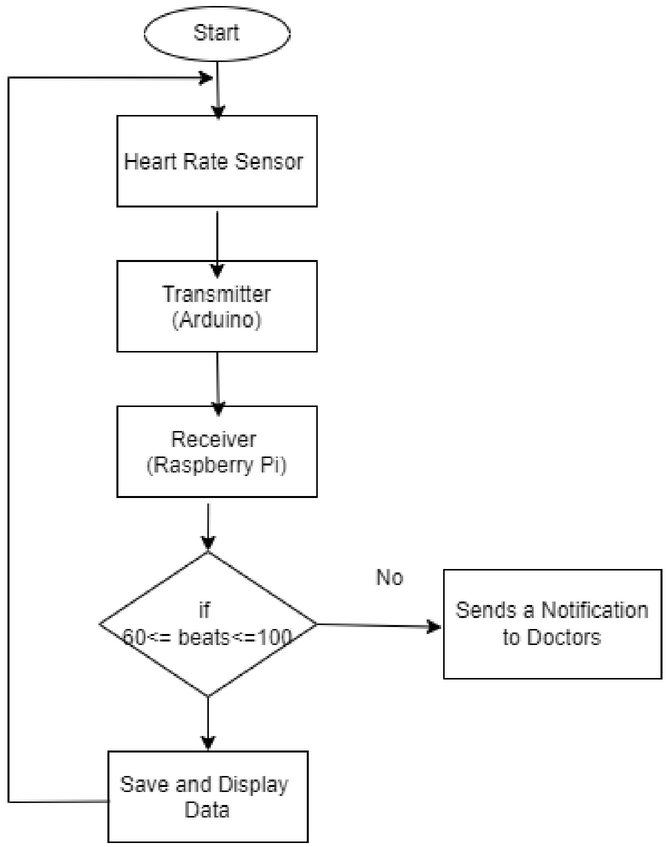


Figure 2.2.4: Activity Diagram for Heart Rate Sensor

The activity diagram above showcases one of the sensors in the monitoring system: the heart rate sensor. It begins with the user requesting a heart rate diagnostic and the diagram shows the steps that proceed with the request. One of the steps includes a condition placed with a range of an adult's normal heart rate. If the condition is not satisfied, a notification/alert is sent to doctors/users. If the heart rate goes beyond a certain limit, such as surpassing 120 BPM, a more serious alert is sent to the user and to healthcare authorities. The same case applies for an unusually low heart rate, which is abnormal.

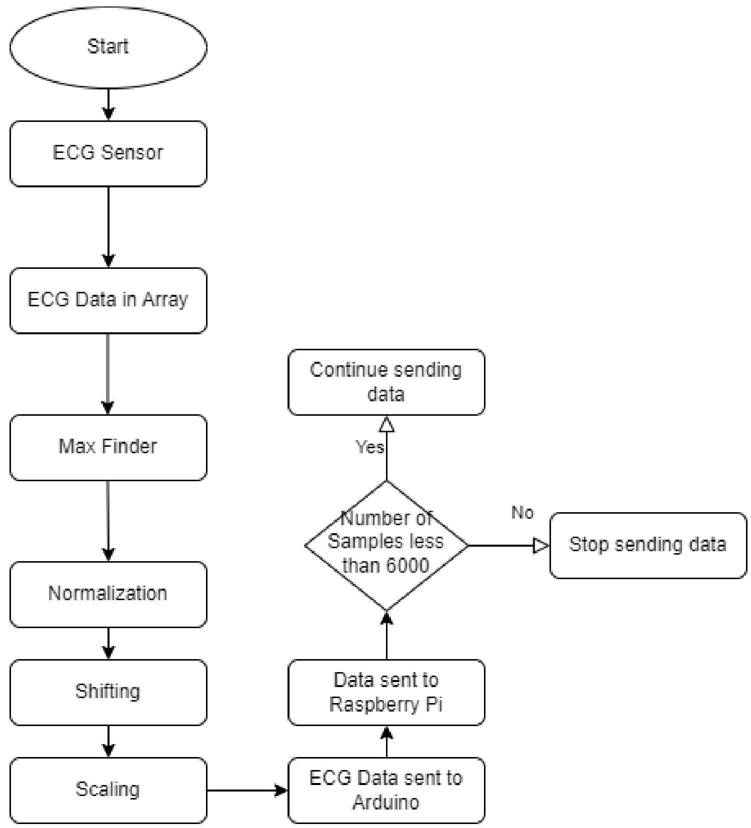


Figure 2.2.5: Activity Diagram for Electrocardiogram (ECG) Sensor

The activity diagram above shows the activity for the Electrocardiogram (ECG) Sensor. It starts off with the user starting the ECG diagnostic and the sensor collects the data. The collected data gets put into an array. Then, more data processing occurs with max finder, normalization, shifting, and scaling. The signal data and patterns are processed and the data is sent to the Arduino. It is then sent to the Raspberry Pi. If there are less than 6000 samples, the process will continue until it satisfies the condition.

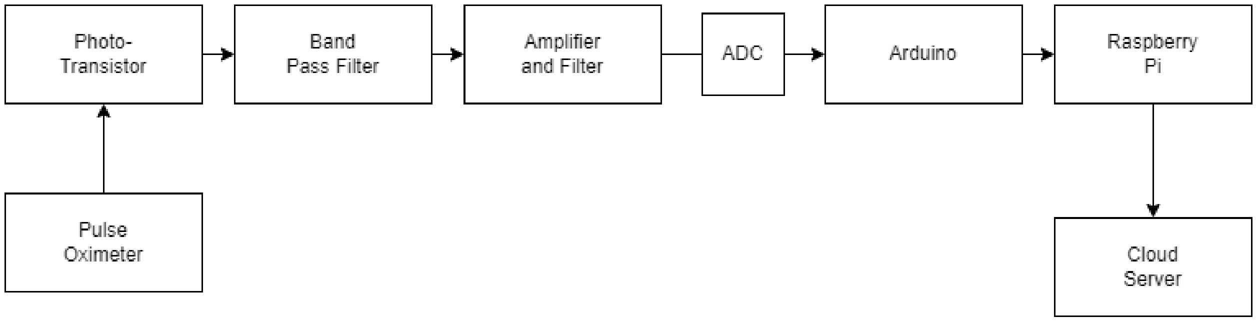


Figure 2.2.6: Block Diagram for Pulse Oximeter

The block diagram above shows the various components for the pulse oximeter. It consists of the Pulse Oximeter sensor and the various filters that are used. The data is then converted into digital from analog and then sent to the Arduino. The Arduino sends the filtered data to the Raspberry Pi and is finally sent to the Server.

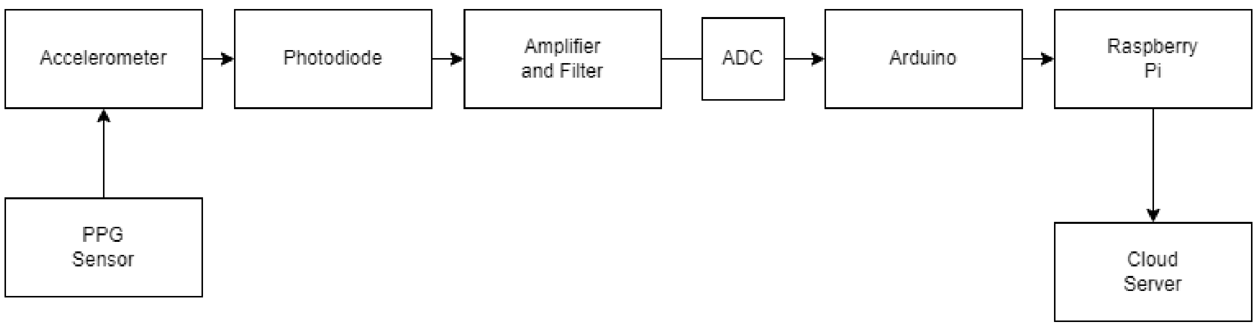


Figure 2.2.7: Block Diagram for PPG

Similarly to Pulse Oximeter, PPG follows the same process with similar components. It has the sensors, including the accelerometer and photodiode. The signals are then processed and then sent to Arduino, Raspberry Pi, and end up in the Cloud Server.

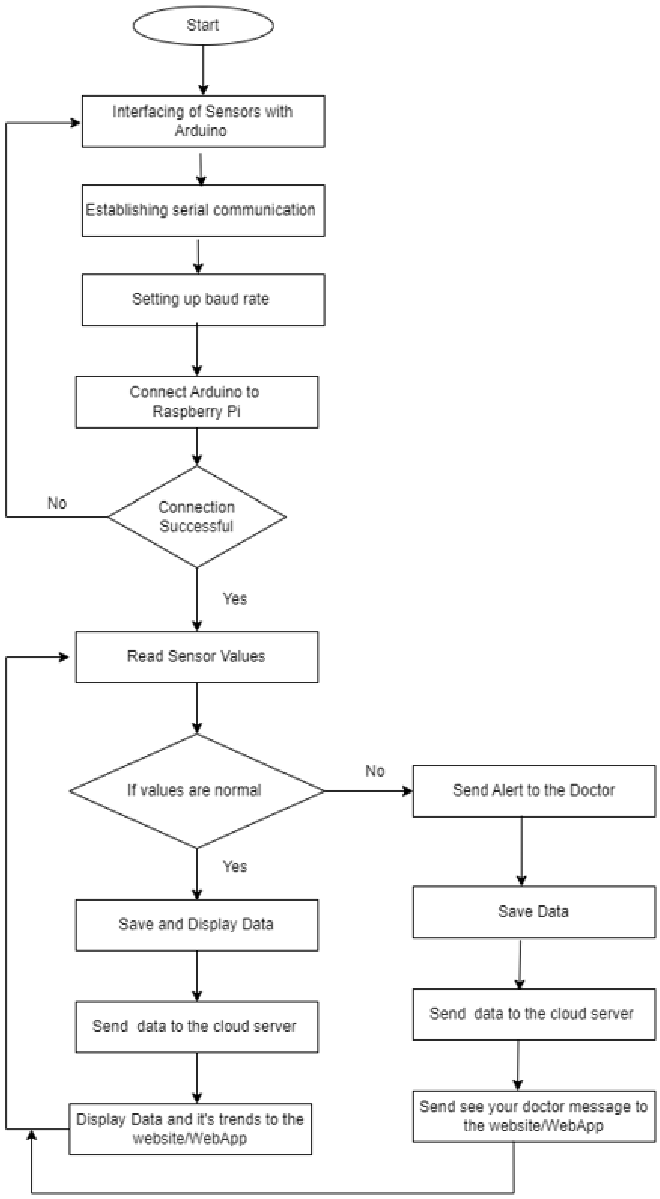


Figure 2.2.8: Activity Diagram for Overall System

The activity diagram above shows the overall system. It shows the steps and includes all the component units and the various situations that occur too. This includes the case where values are abnormal and the proceeding steps that occur. It also accounts for unsuccessful connections.

2.3 Engineering Requirements and Traceability Matrix

Table 2.4.1: Engineering Requirements

Req-ID	**Requirement**	**Category**	**Quality Attribute**
REQ-01	The mobile medical monitoring system must be able to perform diagnostics and collect heart rate, blood volume levels, blood oxygen levels, and blood pressure waveforms at 60Hz.	Functional Requirement	
REQ-02	The backend should handle all calculations of data (waveforms, conversions, etc).	Functional Requirement	
REQ-03	The application must be able to store the data from the sensors and provide a real-time analysis of the data accumulated within 3 seconds.	Non-Functional Requirement	Performance, Usability
REQ-04	If the heart rate monitor detects abnormalities ranging 40% higher or lower in heart rate, it should notify the user/doctor via a notification/email immediately and advise them to visit a physician.	Non-functional Requirement	Reliability

REQ-05	The system should run on a server which has a good performing GPU, and at least 16 GB of RAM to allow training and testing of the model done quickly.	Non-Functional Requirement	Efficiency, Stability
REQ-06	The system should handle multiple requests (multiple diagnostics like heart rate, blood volumes) from the user at the same time.	Non-Functional Requirement	Scalability, Efficiency
REQ-07	The monitor system must display the results of the diagnostics in an easy to understand way.	Functional Requirement	
REQ-08	The system should store previous data up on a cloud database server which allows storage capacities up to 10GB.	Non-Functional Requirement	Capacity

2.3.1 Test Cases

Table 2.4.2: Test Cases

Test Case ID	Test Case Description	Test Case Steps

TC1	Run the calculations with an input data set to test backend calculations and the Display.	1. Input a test case where the parameters are similar to what the sensors read and insert them into the backend (Raspberry Pi) of the system. 2. Invoke the methods that run the calculations which measure for average heart rate, blood oxygen, and pulse. 3. Retrieve the output from the system on the Display.
TC2	Verify that sensors transmit to the Raspberry PI Server Unit.	1. User will have their vitals read by the sensors. 2. Data will be sent to Arduino locally and will be read via the readData module. The strings then are encoded into a char array and sent to the Raspberry Pi using bluetooth. 3. Data is then Decoded in the Raspberry PI. 4. Test method to display the decoded raw data that the sensors recorded.
TC3	Verify that the data communication to the cloud server is active.	1. The data that has been resulted from the calculations in the averageData Module is transmitted to the cloud server using REST/MQTT communication protocol. 2. The processData module will store this data to the Database. 3. Retrieve data from the database by querying for the data in the tables and validating the output.
TC4	Verify the stability of the system.	1. Send multiple requests to the system to detect how well it performs under heavy traffic. 2. Overload the database with a large dataset to test for the capacity as well as the reliability of the system when parsing a large database.
TC5	Verify that the System sends an alert if the values are varying by 40%.	1. Input a test data set and have a test database composed of average data in the past. 2. The test dataset will have data that is

		highly varying to the average data in the database. 3. The input of the new “abnormal” dataset will validate the functionality of the notifying feature of the system.

Table 2-7: Traceability Matrix

REQ	**01**	**02**	**03**	**04**	**05**	**06**	**07**	**08**
TC1	**X**	**X**					**X**	
TC2			**X**		**X**			
TC3	**X**		**X**		**X**			
TC4					**X**	**X**		**X**
TC5				**X**				**X**

2.4. Definition of Integration Tests

The system has multiple components: Server Unit, Display Unit, ECG Unit, End User Unit, Monitoring System Unit, PPG Unit, Oximeter Unit, Photo Sensor Unit, Pulse Detector Unit and Math Unit.
A set of Integration Tests have been defined below, in Table 3-1, to test the components if they are working properly and if they are working together.

Table 2.3.1: Definition of Integration Tests

Integration Test Case ID	**Integration Test Case Description**	**Expected Result**
[IT-DATA-1]	The Server Unit is running and handling the data that is being stored and retrieved.	Users successfully log their data and are able to retrieve data.
[IT-DISP-1]	After performing diagnostics, the data is displayed to the user with real-time results.	Once the diagnostic is successfully completed, the results should be shown on display for the user to view.

[IT-DATA-2]	After a diagnostic has been completed, the data should be logged/stored to the server.	The connection to the server is successful. The data is sent to the server to be stored and can be accessed/retrieved later.
[IT-ALERT-1]	If the data collected from the diagnostic indicates any abnormalities with the user, it should alert the user immediately.	After a performed diagnostic that indicates any irregularities, the system should be able to detect that the data is abnormal and create an alert to be displayed for the user.
[IT-ANLYS-1]	The accumulated data collected can be used to create an analysis to provide the user insight on what the results indicate.	A real-time analysis of the diagnostic should be shown upon request by the user.
[IT-DATA-3]	Some of the data logged should not be accessible by the user, as it is meant for a physician to view.	This data is stored into the server but is hidden, so the user cannot access/retrieve the data.
[IT-DATA-4]	The “bad noise/inaccurate” data is collected by the sensor, providing inaccurate information.	The data is filtered out during or after the diagnostic to prevent inaccurate data from being recorded.
[IT-DISP-1]	The time and battery life is displayed to the user.	The device accurately shows the current time and battery life on the Display Unit.
[IT-CALC-1]	The Math Unit calculates the ECG waveform.	The Math Unit accurately performs a calculation to get the frequency (Hz) of the ECG waveform.

2.5. Revised Project Plan

The estimated costs will include the materials needed for the system but will not include labor and build costs at this time. All the prices below are shown in CAD and will correspond in the pricing of the area we will be purchasing from within Canada. The prices listed below are estimates and do not reflect the final price.

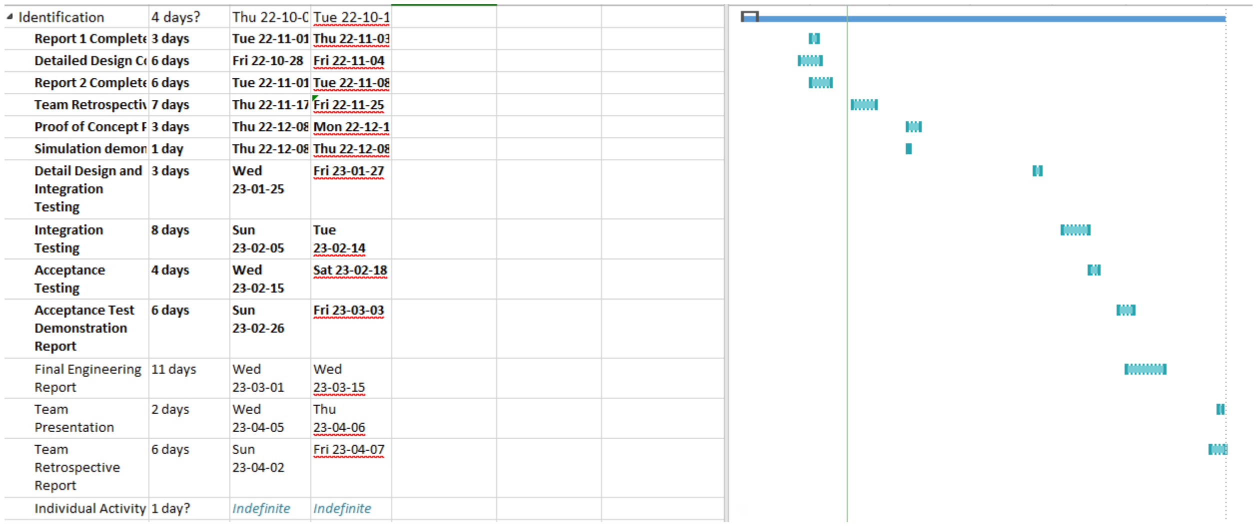
Table 2.6.1: Estimated Cost of the Project

Component	**Component Description**	**Cost per unit ($ - CAD)**	**Physical Quantity**	**Total Cost (cost/unit * quantity) ($ - CAD)**
Raspberry Pi	The Raspberry Pi will be used to run and host a server and power the system and communicate with the display and cloud server for data collection and distribution	240	1	240
Arduino Mega	The Arduino will host all the sensors and connect to the Raspberry Pi and give us accurate readings	68	1	68
Display Unit (240x240, General 1.28inch Round LCD Display Module, 65K RGB)	The LCD Display Module will display the data the end user will have available to them	20.45		20.45

PPG Unit (Pulse Sensor + MCP3008 ADC IC)	The pulse sensor that will send the heartbeat via a waveform pattern and the MCP2008 ADC that will be sending the ECG data in Hz	33 + 6	1 + 1	33 + 6 = 39
Oximeter Unit (MAX30100)	The sensor that will determine the blood oxygen levels and transfer that data to the cloud	18	1	18
Developer man hours	The developer man hours for developing the project	Not added to the project cost	N/A	N/A
Total Estimated Cost of all Components (CAD)				$385

2.6 Updated Project Plan

An updated Microsoft Project Schedule has been revised for the phases of the project, including the revisited timelines for all the activities. This can be seen below, the schedule will show the entirety of the first semester deadlines and schedule with detailed activities, however for the second semester activities the schedule completion will remain undefined as we are not aware of the complete requirements of the second semester at this time.



Identification	4 days?	Thu 22-10-0	Tue 22-10-1
Report 1 Complete	**3 days**	**Tue 22-11-01**	**Thu 22-11-03**
Detailed Design C	**6 days**	**Fri 22-10-28**	**Fri 22-11-04**
Report 2 Complete	**6 days**	**Tue 22-11-01**	**Tue 22-11-08**
Team Retrospectiv	**7 days**	**Thu 22-11-17**	**Fri 22-11-25**
Proof of Concept F	**3 days**	**Thu 22-12-08**	**Mon 22-12-1**
Simulation demon	**1 day**	**Thu 22-12-08**	**Thu 22-12-08**
Detail Design and Integration Testing	**3 days**	**Wed 23-01-25**	**Fri 23-01-27**
Integration Testing	**8 days**	**Sun 23-02-05**	**Tue 23-02-14**
Acceptance Testing	**4 days**	**Wed 23-02-15**	**Sat 23-02-18**
Acceptance Test Demonstration Report	**6 days**	**Sun 23-02-26**	**Fri 23-03-03**
Final Engineering Report	11 days	Wed 23-03-01	Wed 23-03-15
Team Presentation	2 days	Wed 23-04-05	Thu 23-04-06
Team Retrospective Report	6 days	Sun 23-04-02	Fri 23-04-07
Individual Activity	1 day?	*Indefinite*	*Indefinite*

Figure 2.7.1: Updated Project Plan

3.Revised Design Report

This section will consist of our revised design content for this project. All the design modifications we have made and detailed design aspects for our project will be outlined. Please refer to the sections below for the corresponding information.

3.1 Detailed Design

As mentioned in the previous reports, this project is composed of three components with the second component interlinking the other two components to create the system, which include the Arduino component acting as the hardware component, the cloud hosting service acting as the middleware component and the mobile application acting as the software component. In terms of the application, the proposed plan for the application is to contain many functionalities, such as retrieving vital signs data from the arduino and having a way for it to be displayed accurately, and then viewing the data whenever the user would like to see, whilst ensuring that the data is kept secure. The detailed design process of this project revolves around the fundamental design basis of several different inputs and outputs of the overall system, as well as the design basis of the system in its entirety and how all the inputs and outputs correspond with each other. These will be discussed in the sections below. To add on, an overview of the overall system will also be seen in the sections below.

3.1.1 System Prototype

The prototype is developed using concepts such as separation of concerns, modularity, and loose coupling. The prototype is built with ideas like separation of concerns, modularity, and loose coupling. Each sub-system is made up of hardware and software components that are all integrated into the prototype to allow it to function as a whole. The diagram below depicts a high-level overview of the project system.

System Block Diagram

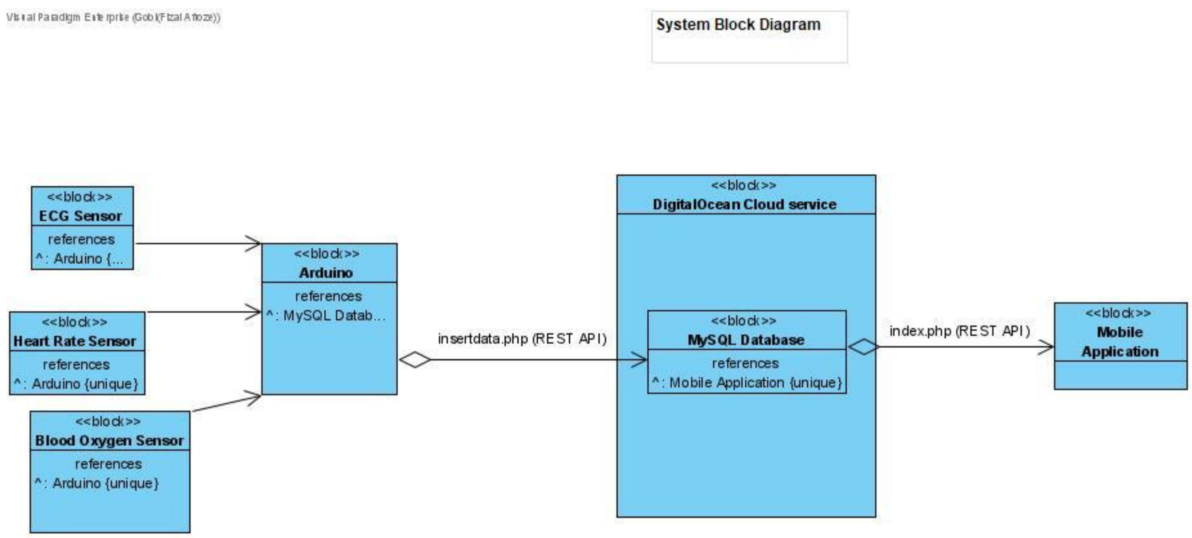


Figure 3.1.1.1: System Block Diagram

This component diagram showcases the various components of the system, as well as how they interact with each other. The Arduino is connected to the Cloud Host (DigitalOcean) via the RESTful API (REST API). The data collected from the sensors will be sent to the cloud, which is later stored in the MySQL Database.

DigitalOcean (Cloud)

Digital Ocean is the cloud service provider for the system that is responsible for handling the data being stored and retrieved from the Arduino and the mobile application. The DigitalOcean Cloud droplet has PHP, MySQL, and Apache set up on it. The PHP component receives the data from the Arduino Mega through HTTP requests and passes it on to the MySQL database. MySQL is a relational database management system used to store the data collected from the Arduino. It has various parameters including the User's ID, the measurements of data and the name of the sensor used for that data, and a time stamp for the time the measurement occurred.

Mobile Application

The mobile application is connected to the cloud and requests data from the database. The data is sent from the cloud to the mobile application, where it is able to be viewed by the user in a readable UI. It can be viewed by the patient and/or the doctor.

Deployment Diagram

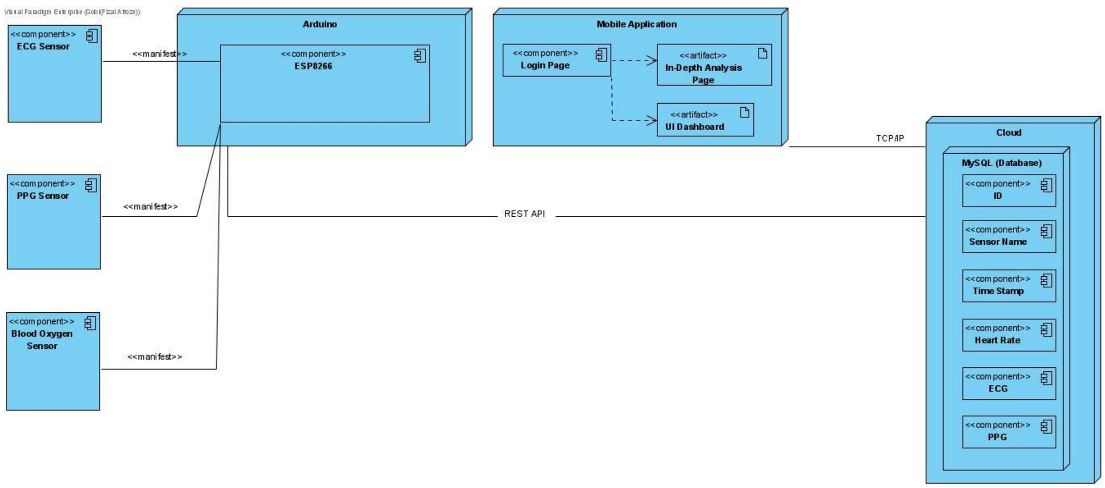


Figure 3.1.1.2: Deployment Diagram

The data flow diagram illustrates the flow of information and processes in a our system that collects and monitors health data from sensors using an Arduino Mega 2560 microcontroller, stores the data in a MySQL database on a DigitalOcean server, and displays the data on a mobile application created in Android Studio. The data flow begins with the sensor devices, which collect health data such as PPG data, blood oxygen level, and ECG waveform data. The Arduino Mega 2560 is then used to communicate with the sensors and retrieve this data, which is then stored in the MySQL database on the DigitalOcean server. The mobile application is able to access this data by requesting it from the server, and is able to display the PPG graphs, blood oxygen level, and ECG waveform graphs in a readable format. This data can be accessed by doctors and patients either through cloud or through

Data Flow Diagram

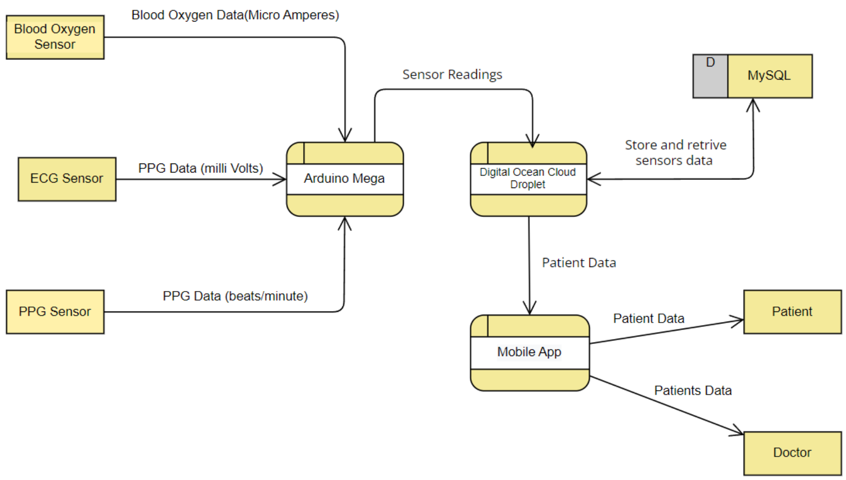


Figure 3.1.1.3: Data Flow Diagram

The main functions in this sequence diagram is:

1. The ability to turn the device on
2. The ability to send real time data to the cloud and then to the mobile application
3. The ability to store offline data and then upload to the cloud when possible
4. To view data as deemed necessary
5. View the timestamps of the data

Sequence Diagram

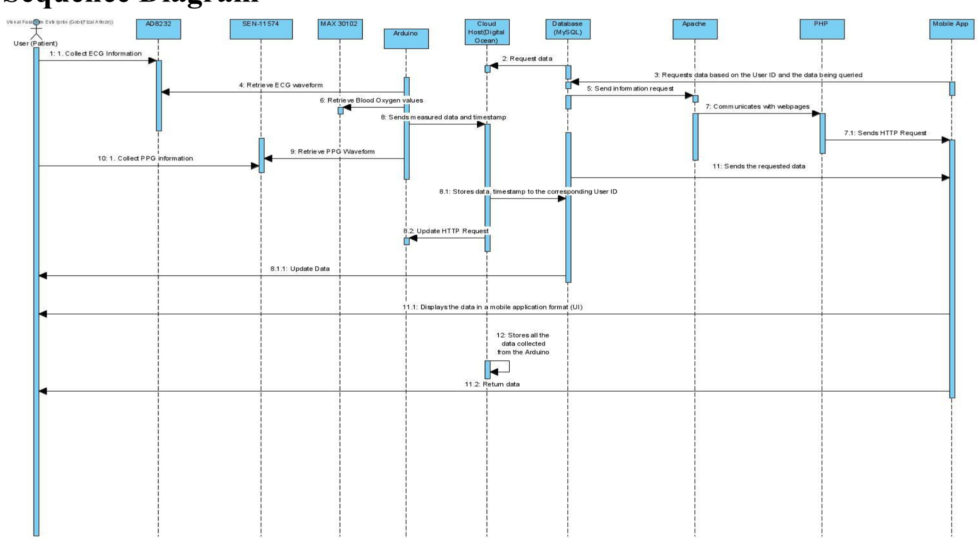


Figure 3.1.1.4: Sequence Diagram

As depicted above, the Patient and the Doctor have many different attributes and interactions with each other and the other entities in the Mobile Application. The main players in this application are in fact, the Patient and Doctor, and what information they can access from their Dashboards/UI. Authentication is a big part of the security and confidentiality of Health Monitoring systems so in order to access any information on the application, both the user and doctor must be logged in. Doctor and Patient can access the Digital Ocean Database via a connection port on the application that will allow Android Studio to read from our MySQL

Entity Relationship Model

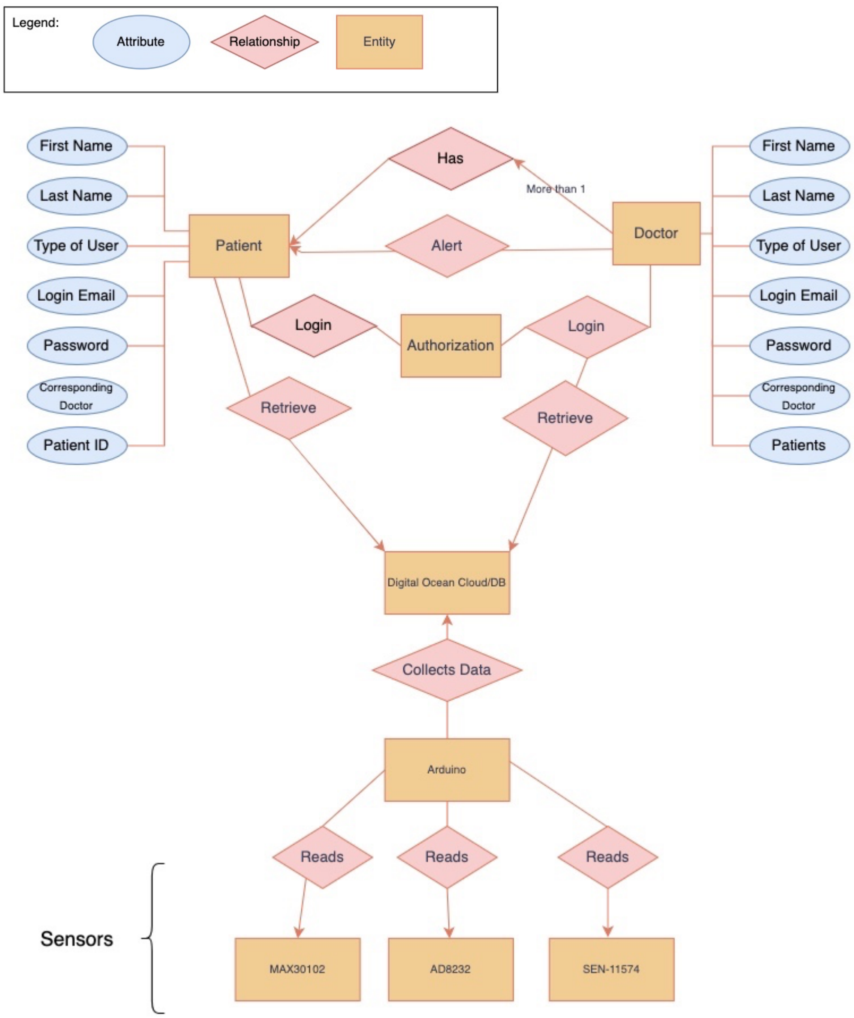


Figure 3.1.1.5: Entity Relationship Model

3.2 Unit and Integration Testing

Definition of Unit Tests

The unit tests specified are for testing that functions return the correct data and the behavior of the system is as expected. Most of the testing will take place on the Arduino Mega 2560 which will connect to the Digital Ocean server and the mobile application where a large portion of the main functionality will occur.

ID	**Test Case Objective**	**Test Case Description**
1	To test whether the connections between the Arduino Mega 2560 and the sensors are properly established and working as expected.	Test if the Arduino Mega 2560 is able to communicate with the sensors and retrieve accurate data. For the SEN-11574, measuring the PPG data on the Mega and comparing it with the data obtained from the SEN-11574. For the MAX30102 Pulse Oximeter, measure blood oxygen level from the Mega and compare it with the data obtained from the sensor. For the 3 lead ECG AD8232, measure the ECG waveform data obtained from the Mega, and then compare the data that we obtain from the sensor.
2	To test data is properly stored in the SQL database and in the expected format	Test that the data is stored in the SQL database and ensure that it contains all the required information, such as the PPG sensors red values and ir values, the numerical blood oxygen levels, as well as the ECG values with their corresponding timestamps. Ensure that the blood oxygen level is stored as a numerical value.
3	To test if the DigitalOcean server is properly receiving and storing the data	Test if the data from the Arduino Mega 2560 is being sent to the DigitalOcean server properly and ensure that it is being received and stored. Ensure that the server corresponds with the correct HTTP protocol and that the data is being stored in the server database that has been created with MySQL with the correct required formats and is accessible to the mobile application.
4	To test the mobile application created in Android Studio is able to receive the data from DigitalOcean.	Test that the mobile application is receiving the data properly from the DigitalOcean server that is set up with PHP and MySQL, and make sure that the received data contains all the required information to calculate and plot PPG graphs, blood oxygen level, and the ECG waveform graphs in the app .

5	To test the mobile application is able to display the data	Test that the mobile application created in Android Studio is able to display the PPG graphs, the blood oxygen level in numerical value, and the ECG waveform graphs in a readable format.
6	To test the system being the ability to handle invalid requests	Test that system will be able to handle incorrect inputs, such as invalid data or data outside the scope of the sensors ie. incorrect timestamps. Ensuring that the system is responding to the error messages. An example of this would be if there is a PPG reading being made outside of the constraints of the sensor, the system should be able to recognize it and allow the user to know that there is an error occurring.
7	To test the system's performance	Test that the system has an appropriate level of performance and reliability by running it over prolonged periods of time and ensure that the system does not have any major issues in doing so, whilst the data is able to store information in real-time and have a storage option as well when going offline.

Table 3.2.1: Definition of Unit Tests

Definition of Integration Tests

Integration testing is necessary to ensure that the required components are not coupled to a point where there will be any issues. We desire our system to be efficient in its performance and as loosely coupled as necessary to ensure that if a component fails, the system can still continue to function.

ID	Test Case Objective	Test Case Description
1	Check if there as been a successful transfer of data of PPG data from SEN-11574	This will check the communication between the Arduino and the SEN-11574s by simulating the PPG data in real-time from the SEN-11574 to the server via the Arduino Mega and it is being stored in the MySQL database in the correct format. The PPG signal will be sampled at a rate of 100 Hz.
2	Check if there has been a successful transfer of data of blood oxygen data from the MAX30102	This will check if the communication from the Arduino and the MAX30102 is working by sending blood oxygen levels in numerical format to the server via the serial connections in the Arduino Mega to the MySQL database in the correct format. The bpm values will be

		sampled at a rate of 100 Hz.
3	Check if there has been a successful transfer of data of ECG data from AD8232 to the server	This will check if the communication from the Arduino and if the AD8232 is working by sending the simulated data from the system to the MySQL database and ensuring it is received and stored in the correct format. The parameters for this will be measured in mV with a range of +/- 1.5mV. The waveform should capture a frequency range of 0.5 Hz to 150 Hz, and have a sampling rate of at 500 Hz to be able to accurately capture the waveform.
4	Check if the retrieval of data is successful of the PPG, blood oxygen, and ECG data by the mobile application	This will check if the HTTP request from the mobile application to the DigitalOcean server is being retrieved to store PPG, blood oxygen, and ECG data and verifying that the data that is being received is being sent to the appropriate display in the mobile application and in the correct format.
5	Check if the storage of PPG, blood oxygen, and ECG data is successful in the MySQL server	This will check if the data from the sensors from the Arduino Mega 2560 is being sent to the server and is stored in the MySQL database in the correct format.
6	Check if validation of incoming data is successful with PHP scripts	This will check if the simulated data that is sent to the server will verify that the data is being processed in an appropriate manner and being stored in the MySQL database within the correct parameters, whilst also rejecting data that is outside the parameters of the data.
7	Check if the integration of DigitalOcean, Apache, PHP, and MySQL is successful	This will check if the data is sent to the server and will verify that the data that is sent over is stored and processed in the database, therefore allowing the system to ensure that all the individual components are working together in cohesion.
8	Check if data is being transferred accurately to the server in real-time	This will check that the data that is being sent over will be stored in the MySQL database in real-time, to confirm that the system can handle a high volume of incoming data.
9	Check if the system can handle multiple data HTTP requests from mobile applications	This will check the system's response to sending several HTTP requests within a short time period to retrieve the allocated data (ie. PPG, blood oxygen, ECG) and ensure that the server will return the correct data under the time load.

10	Check if the system can handle multiple streams of data from Arduino Mega 2560	This will check the data from the sensors and ensure that the server can receive and process the data in the correct format.
11	Check if the storage is successful and data is transferred in the correct format in MySQL	This will check the data that is sent to the server and ensure that the data that is being stored in the database is in the correct format.
12	Check if the security protocols in place are successful	By sending the data over to the server via an unrecognized source, the system will ensure that the data is rejected and will display an error message.
13	Check if the mobile application can handle errors	Forcefully input an error in the mobile application and ensure that the error message is displayed in accordance with the data parameters.
14	Check if the server can handle errors	Forcefully input an error in the server and ensure that the error message is displayed in accordance with the data parameters.
15	Check if the HTTP protocol is successful	This will check if the data that is being processed and sent over to the MySQL server is using an incorrect HTTP protocol and will reject the data if so and will have an error message.
16	Check if the PHP script data processing can handle errors	By forcing an error in the PHP script for any incoming data outside of the correct parameters, we will verify that it will be able to handle it as so and prompt an error message.
17	Check if we have the correct data analytics and visualization tools	Query the MySQL database using the data analytics tool that can be used for test cases to ensure that it meets the needs for the end-users.

Table: 3.2.2 Definition of Integration Tests

ID	Title	Steps	Inputs	Outputs	Acceptance Criteria
1	Take health	1: Login to Mobile	Sensor inputs	Sensor	Once the user is

	readings from sensor	Application 2: Go to selected sensor 3. For required sensor, place either finger on device or apply 3 lead	(be able to use the sensor correctly)	reading based off sensor being used	taking the reading, it should be able to update in real time with timestamps with the desired reading
2	Login to application with user specific information	1: Have a unique user ID and password 2: Have information be saved in database	Correct ID being supplied	ID being initialized by the database	Once the user has attempted to login, the login should either be successful or unsuccessful based off the user credentials
3	View specific sensor data	1: Have separate databases for different data 2: Have the correct parameters for each specified data	Data is accurate with the correct parameters	Data is being sent to the user	Once the user accesses the data logs based off the timestamps, it should allow for accurate and consistent data
4	Suggestions for different data	1: Login on mobile application 2: Go to suggestions dashboard	Any suggestions the user has	Suggestions will be displayed	Once the user signs in and uploads data about different sensors they want to have in the system, the system will acknowledge it
5	Data from a specific timestamp	1: Login to mobile application 2: Go to desired sensor dashboard for the data that wants to be viewed	Health data from the user	The correct data parameters for the data that the user is seeking for in that	The mobile application should display the correct data within the correct parameters for the health section that they are seeking data from

				timestam p	with respect to the required timestamp
6	Database is retrieving and sending data properly	1: Data is being sent to cloud hosting service and stored 2: Data is being retrieved from cloud hosting service and being displayed	Data storage	Data is being stored and retrieved with the necessary POST and GET api calls	The cloud hosting service should be able to use GET api calls to retrieve the sensor data from the Arduino MEGA 2560 and should store that data, and then when the application demands the information based on the unique ID of the user, should be able to retrieve any data using a POST api call.
7	Mobile application works	Non-functional requirement	N/A	N/A	There should be no issues launching the android application
8	Easy to use	Non-functional requirement	N/A	N/A	User should be able to navigate throughout the application easily
9	Backend is functioning	Non-functional requirement	N/A	N/A	There should be no major errors in the application

Table 3.2.3: Functional and Non-Functional Acceptance Tests

3.3 Test Results

ID	Results	Status	Comments
1	Data is being sent within desired parameters	**Pass**	The test for transmitting data from the medical sensors has passed. Data is being transmitted in accordance with the Hz values of the sensors, allowing for accurate readings.
2	Users were able to login with unique IDs and passwords with some minor issues	**Pass**	Test passed successfully as intended
3	Currently working on the GET requests to allow for connection to Arduino sensors and POST requests to view on mobile application with cloud hosting service	**Pass**	Data is being sent to the backend via the Cloud hosting service and then being retrieved by the mobile application
4	Suggestion box has been implemented in the mobile application	**Pass**	Test passed successfully
5	Data successfully having real time data being sent and checked at specific timestamps	**Pass**	Data successfully accessed at time data was registered
6	A suggestion has been made to change Arduino script with some more unique IDs in the script to enable PHP cohesion to retrieve unique user information	**Pass**	Data has been configured that it will only be registered to the user using the system
7	Mobile application	**Pass**	Has been

	contains multiple functioning sections that operate successfully, such as the login page, patient view and doctor view.		implemented successfully with no apparent glitches
8	Easy to navigate through the various sections of the mobile application, accessible buttons, arrows, etc.	**Pass**	Has been implemented successfully
9	Backend is functioning properly.	**Pass**	Sensor data is setup properly with the corresponding tables

Table 3.3.1: Acceptance tests Results and Status

4. Ethical Considerations

It is of utmost importance to outline all of the ethical related considerations for this project as well. In terms of utilizing a remote mobile healthcare system and its devices, many ethical concerns can be considered. These may include informed consent of the user using the remote mobile healthcare system. In order for a remote mobile healthcare system to maintain a high level of integrity and offer any type of informed consent, it must be ensured that the rights and responsibilities of the system are relayed clearly to the consumer so that they would be fully aware of any of the purpose, risks, and benefits the system may consist of. The users should have the right to also be able to make informed decisions in regard to their healthcare, and transparency in how the information is obtained to respect the user's autonomy. Many companies follow the trend by having a term of conditions that the user may accept in-depth, which makes it difficult for the user to know exactly what they are consenting to. With our system, we aim to offer as much transparency as possible for users in regard to any informed consent. These may include direct customer service lines to ask any questions, or a section in our application that is highlighted and details what the user is specifically signing up to use.

Another crucial ethical consideration is privacy. The main ethical concern in regard to privacy is to ensure that any sensitive medical data is not disclosed to any unauthorized parties. Due to our application being on the Android Platform and having several medical sensors storing data, we want to ensure to have the highest degree of privacy possible. That being said, although any medical data for a patient is often protected by the law, any breaches of privacy can lead to both legal and ethical implications. What we have been able to do to respect the patients' privacy rights is that we have used a REST API that will securely transfer data from the system to the user without needing to go through any extra hurdles as it will be tied directly to the user's login information.

When it comes to the ethical considerations of accountability in regard to the remote healthcare system, we need to be able to prioritize the system's performance against any adverse effects that may occur. When providing any type of medical-grade sensor to someone who is not well-versed in the technology, it is possible for the user to become flustered and be unable to use the system. Due to the level of the system, it is of utmost importance that we will be transparent about the system's capabilities, limitations, and potential risks. The Remote Mobile Healthcare System will have an in-depth guide provided with the final product either physically or created online for the user to easily access and learn about the product as deemed necessary.

5. Safety Considerations

There are several safety concerns that need to be addressed while developing a Remote Mobile Healthcare System. The system relies on both hardware and software concerns, so there were several safety concerns that needed to be addressed. To start off with, a major safety concern is the risk of inaccurate readings due to hardware malfunction or incorrect placement of the sensors. As the sensors are purchased off-the-shelf, we have made sure that there is thorough testing and validation for the system as it is necessary to do so to ensure that the system can accurately collect and transmit data. Failure to do so could end up resulting in incorrect diagnosis, delayed treatment, or even potential harm to the patient.

Given the nature of this project, data confidentiality and data privacy is a major safety consideration that needs to be thought of. Data is highly sensitive, and as data is constantly being added peruse of the system, the risk increases each time. We had ensured that there was a secure way to have data transmitted and we also made sure to sign up for the application. We did not require a significant amount of information required to create an account so the user will not feel pressured or obligated to provide any personal information. We also only save the user information strictly to the user that is logged in and can be accessed only by the user's specific ID and password that they set upon themselves.

It is important to conduct regular updates for the application to ensure optimal function and the testing methods themselves also carry a high degree of safety that should be held at all times. Protecting the users from any malfunctions or data breaches requires adherence to safe work practices as well as cybersecurity practices and ensuring the system can remain dynamic to resolve any issues. For example, having a guide on how to use the equipment or updating the application software in accordance with any cyber threats that the application may come under.

In summary, developing a remote mobile healthcare system addresses different safety concerns such as sensor malfunctions or data confidentiality, and will ensure adequate user training and education on how to use the system and protect user data through robust security measures. By being able to address these safety concerns, the system will ultimately be able to improve patient care while ensuring patient safety and privacy.

6. Conclusions

In conclusion, our project was successful in developing an Arduino system that connects to WiFi and transmits data, along with the Digital Ocean cloud hosting service that connects to the mobile application. The mobile application is able to successfully send and receive data to the cloud. Our mobile application was able to retrieve data without being directly connected. Through this project, we learned more about different medical sensors and the growing need for such technology. Additionally, we gained valuable experience in reading datasheets efficiently and learned important engineering design and development skills. In this report, we included the revised design report, which showcases our final design with some changes. The ethical and safety considerations were also addressed, such as the risk of inaccurate readings due to hardware malfunction, data privacy, etc. Overall, the project achieved the main functionality of providing remote healthcare, ensuring users have access to these services in real time.

7. Acknowledgements

We would like to express our sincere gratitude to our Capstone Coordinator and Capstone Professor for their unwavering support, guidance, and encouragement throughout our project. Their expertise and feedback have been invaluable to our success.

Our Capstone Professor, Dr. Mikael Eklund, provided us with insightful advice, and expert knowledge, and helped us to remain on track with our objectives. His mentorship and regular feedback have been invaluable to the success of our project.

We also extend our heartfelt appreciation to our Capstone Coordinator, Dr. Vijay Sood, for his unwavering support and guidance throughout the project. He is always available to offer assistance and support in all areas of the project, and we are grateful for his dedication and commitment to our success.

We would also like to extend our thanks to our colleagues and fellow students who provided valuable feedback and support during the development of our project. Their contributions and insights helped us to refine our ideas and develop a more robust and effective system.

Finally, we would like to thank our families and loved ones for their unwavering support and encouragement throughout our academic journey. Their love and support have been a constant source of strength and inspiration, and we are deeply grateful for everything that they have done for us.

8. References

[1]. S. Aziz, Y. M. Arabi, W. Alhazzani, L. Evans, G. Citerio, K. Fischkoff, J. Salluh, G. Meyfroidt, F. Alshamsi, S. Oczkowski, E. Azoulay, A. Price, L. Burry, A. Dzierba, A. Benintende, J. Morgan, G. Grasselli, A. Rhodes, M. H. Møller, L. Chu, S. Schwedhelm, J. J. Lowe, D. Bin, and M. D. Christian, "Managing ICU surge during the COVID-19 crisis: Rapid guidelines - intensive care medicine," *SpringerLink*, 08-Jun-2020. [Online]. Available: https://link.springer.com/article/10.1007/s00134-020-06092-5. [Accessed: 14-Oct-2022].

[2]. "Delay or avoidance of medical care because of COVID-19–related concerns - United States, June 2020," *Centers for Disease Control and Prevention*, 10-Sep-2020. [Online]. Available: https://www.cdc.gov/mmwr/volumes/69/wr/mm6936a4.htm. [Accessed: 14-Oct-2022].

[3]. A. M. Clerk, "Beware of neglect of Non-COVID patients in COVID ERA," *Indian journal of critical care medicine : peer-reviewed, official publication of Indian Society of Critical Care Medicine*, Aug-2021. [Online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC8559735/. [Accessed: 14-Oct-2022].

[4]. A. T. Farooqi, F. J. Snoek, and K. Khunti, "Management of chronic cardiometabolic conditions and mental health during COVID-19," *Primary Care Diabetes*, 08-Sep-2020. [Online]. Available:

https://www.primary-care-diabetes.com/article/S1751-9918(20)30252-7/fulltext [Accessed: 14-Oct-2022].

[5] . A. Gheorghe, C. Maringe, J. Spice, A. Purushotham, K. Chalkidou, B. Rachet, R. Sullivan, and A. Aggarwal, “Economic impact of avoidable cancer deaths caused by diagnostic delay during the COVID-19 pandemic: A national population-based modelling study in England, UK,” *European Journal of Cancer*, 05-May-2021. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0959804921002501. [Accessed: 14-Oct-2022].

[6]. Neoteryx, “A brief history on the Remote Patient Monitoring System,” *Neoteryx*, 15-Jul-2020. [Online]. Available: https://www.neoteryx.com/microsampling-blog/a-brief-history-of-remote-patient-monitoring#:~:text=In%201967%20physicians%20started%20transmitting,were%20Kaiser%20Foundation%20and%20Lockheed. [Accessed: 14-Oct-2022].

[7].“How remote patient monitoring is moving into the mainstream,” *Healthcare IT News*, 09-Mar-2022. [Online]. Available: https://www.healthcareitnews.com/news/how-remote-patient-monitoring-moving-mainstream. [Accessed: 14-Oct-2022].

[8]. Kore, “5 benefits of Remote Patient Monitoring,” *Home*. [Online]. Available: https://www.korewireless.com/news/5-benefits-of-remote-patient-monitoring#:~:text=Faster%20Access%20to%20Better%20Patient,cuffs%2C%20glucometers%2C%20and%20more. [Accessed: 14-Oct-2022].

[9]. M. H. A. F. H. E. L. A. Florence Kariuki RN, “The top 13 benefits of Remote Patient Monitoring,” *HRS*, 05-Oct-2022. [Online]. Available: https://www.healthrecoverysolutions.com/blog/the-top-13-benefits-of-remote-patient-monitoring. [Accessed: 14-Oct-2022].

[10]. S. A. Lowe and G. ÓLaighin, “Monitoring human health behaviour in one's living environment: A technological review,” *Medical Engineering & Physics*, 30-Dec-2013. [Online]. Available: https://www.sciencedirect.com/science/article/abs/pii/S1350453313002567?via%3Dihub. [Accessed: 14-Oct-2022].

[11].“What is the internet of things (IOT)?,” *What Is the Internet of Things (IoT)? | Oracle India*. [Online]. Available: https://www.oracle.com/in/internet-of-things/what-is-iot/ . [Accessed: 14-Oct-2022].

[12].S. Dolan, “Remote Patient Monitoring Trends & Health Devices in 2022,” *Insider Intelligence*, 16-Jan-2022. [Online]. Available:

https://www.insiderintelligence.com/insights/remote-patient-monitoring-industry-explained/. [Accessed: 14-Oct-2022].

[13]. Editor, “Remote patient monitoring systems: Components, types, vendors, and implementation steps,” *AltexSoft*, 03-Nov-2020. [Online]. Available: https://www.altexsoft.com/blog/remote-patient-monitoring-systems/. [Accessed: 14-Oct-2022].

[14] “11 Remote Patient Monitoring Companies You should know about,” Plug and Play Tech Center. [Online]. Available: https://www.plugandplaytechcenter.com/resources/10-remote-patient-monitoring-companies-you-should-know-about/. [Accessed: 08-Nov-2022].

[15] “Managing your health in the age of wi-fi,” Mayo Clinic, 18-Jun-2022. [Online]. Available: https://www.mayoclinic.org/healthy-lifestyle/consumer-health/in-depth/telehealth/art-20044878. [Accessed: 08-Nov-2022].

[16] S. M. Lee and D. H. Lee, “Healthcare wearable devices: An analysis of key factors for continuous use intention - service business,” *SpringerLink*, 30-Oct-2020. [Online]. Available: https://link.springer.com/article/10.1007/s11628-020-00428-3#:~:text=Wearable%20devices%20in%20healthcare%20include,and%20biosensors%20(Phaneuf%202020). [Accessed: 08-Nov-2022].

[17] “Understating ECG sensors and how to program one to diagnose various medical conditions,” *Understating ECG Sensors and How to Program AD8232 ECG Sensor with Arduino to Diagnose Various Medical Conditions*. [Online]. Available: https://circuitdigest.com/microcontroller-projects/understanding-ecg-sensor-and-program-ad8232-ecg-sensor-with-arduino-to-diagnose-various-medical-conditions. [Accessed: 18-Feb-2023].

[18] “Technical reference - espressif.” [Online]. Available: https://www.espressif.com/sites/default/files/documentation/esp8266-technical_reference_en.pdf. [Accessed: 18-Feb-2023].

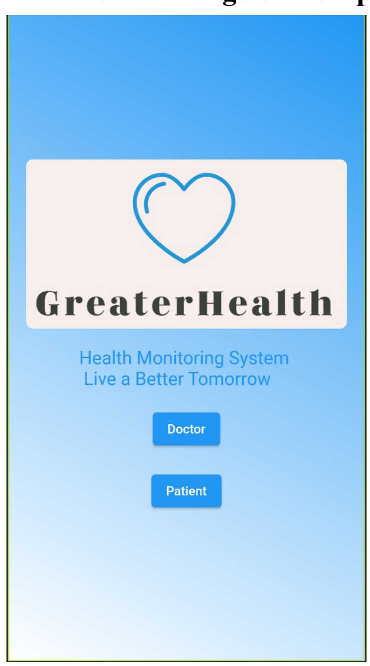
[19]“Single-lead, Heart Rate Monitor Front End Data sheet AD8232.” [Online]. Available: https://www.analog.com/media/en/technical-documentation/data-sheets/AD8232.pdf. [Accessed: 18-Feb-2023].

[20]“Sen-11574 Datasheet(PDF) - sparkfun electronics,” *Allicdata electronics*. [Online]. Available: https://www.allicdata.com/pdf/sen-11574-datasheet/2718390.html. [Accessed: 18-Feb-2023].

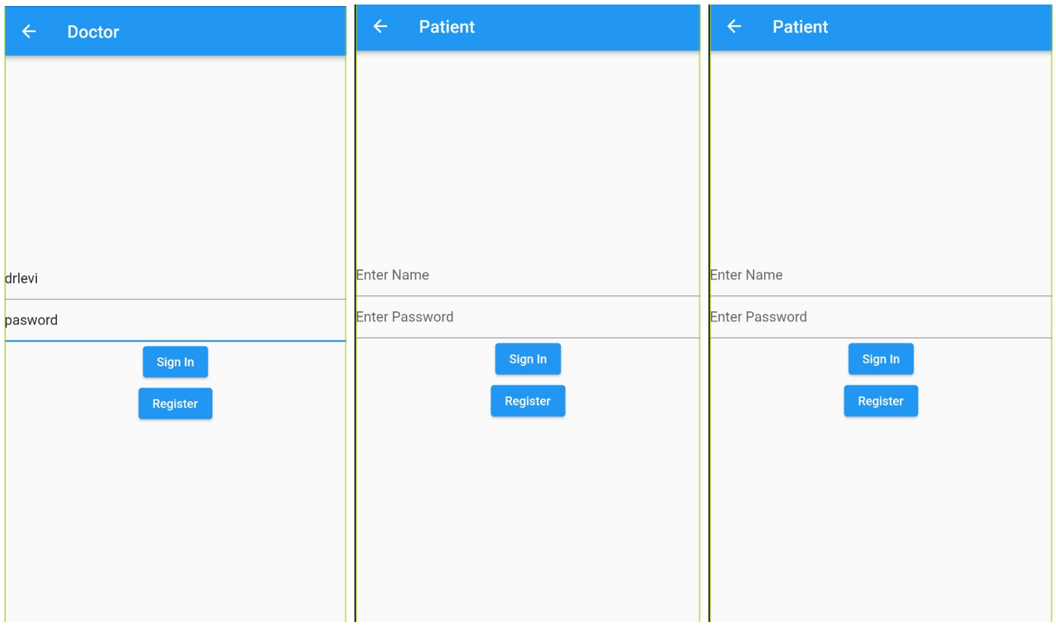
[21]Last Minute Engineers, “In-depth: Detect, Measure & Plot Heart Rate Using Pulse Sensor & Arduino,” *Last Minute Engineers*, 25-Nov-2022. [Online]. Available: https://lastminuteengineers.com/pulse-sensor-arduino-tutorial/. [Accessed: 18-Feb-2023].
[22]“Errors when uploading a sketch – arduino help center.” [Online]. Available: https://support.arduino.cc/hc/en-us/articles/4403365313810-Errors-when-uploading-a-sketch. [Accessed: 18-Feb-2023].
[23]“Click MAX30102 higsensitivity plse oimeter an heartrate sensor or ...” [Online]. Available: https://www.analog.com/media/en/technical-documentation/data-sheets/MAX30102.pdf. [Accessed: 18-Feb-2023].
[24]“Android - PHP/mysql,” *Tutorials Point*. [Online]. Available: https://www.tutorialspoint.com/android/android_php_mysql.htm. [Accessed: 18-Feb-2023].

9. Appendices

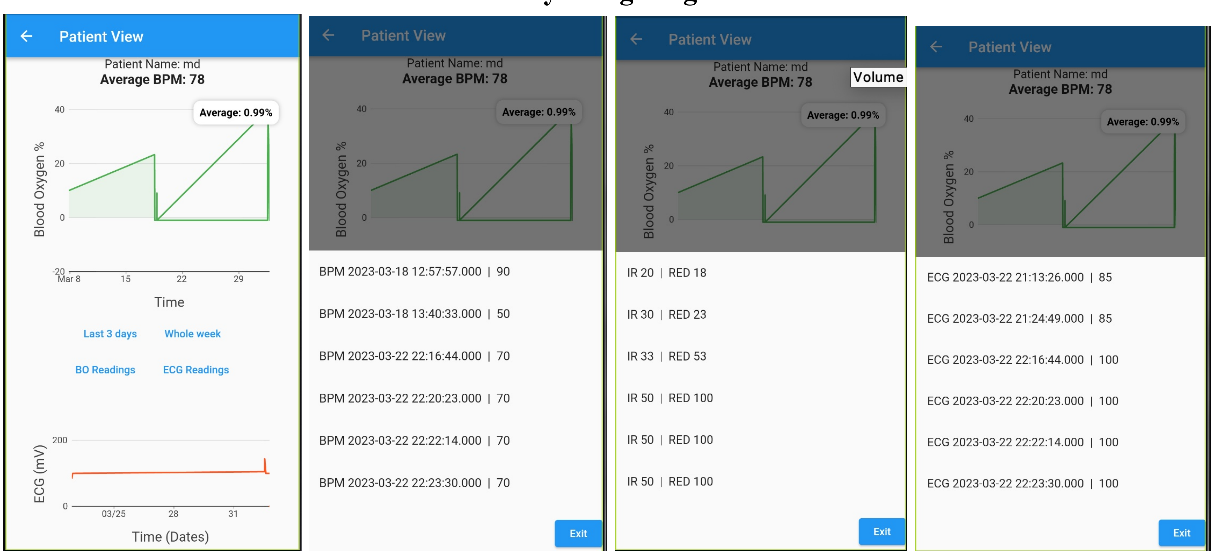
Appendix A: The Main Page of the Application



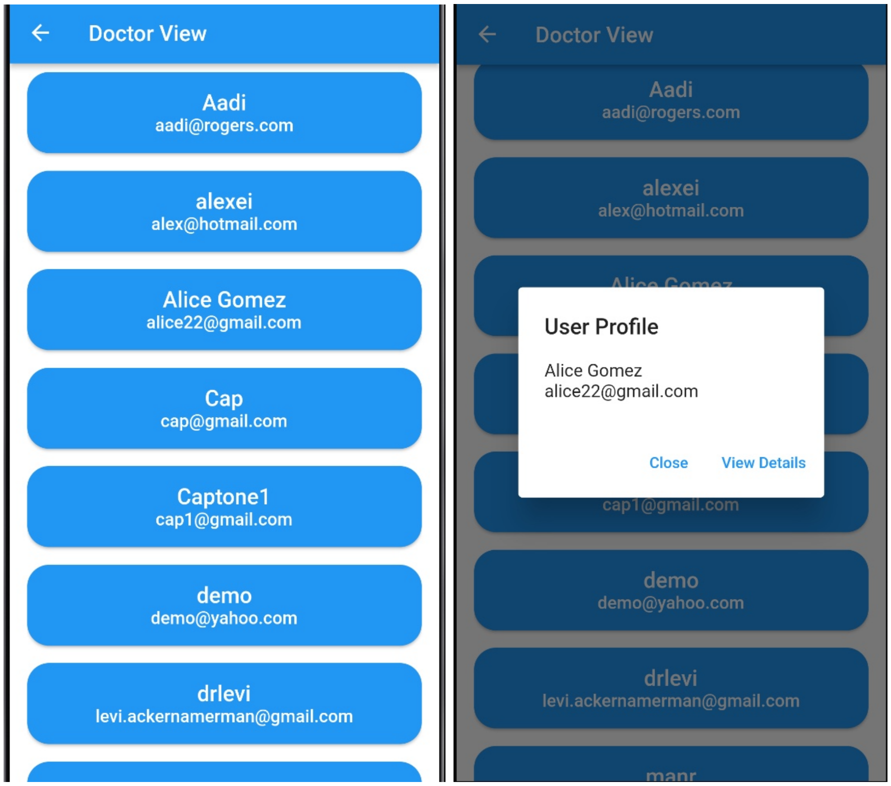
Appendix B: The login page for the Doctor, Patient, and The registration, respectively



Appendix C: Once a patient logs in they can see information and charts regarding their Heart Rate, ECG and BO Level by navigating the User Interface.



Appendix D: Once a Doctor logs in they can see information about their patients



Code Snippets and implementation:

Server Side Scripts:

Appendix E: Showing the configuration of the Server and how the script is mounting to the database

```php
<?php
    define("DB_HOST", "localhost");
    define("DB_USER", "root");
    define("DB_PASSWORD", "CapMysql");
    define("DB_NAME", "health");
?>
```

```php
<?php

    include_once 'config.php';

    class DbConnect{

        private $connect;

        public function __construct(){
        $this->connect = mysqli_connect(DB_HOST, DB_USER, DB_PASSWORD, DB_NAME);

            if (mysqli_connect_errno()){
                echo "Unable to connect to MySQL Database: " . mysqli_connect_error();
            }
        }

        public function getDb(){
            return $this->connect;
        }
    }
?>
```

Appendix F:shows how we query the database to obtain the required data for the selected User, by matching their username and password to the ones in the database.

```php
    $json = array();

        $query1 =  "SELECT email, username, password, doctor FROM ".$this->db_table." WHERE username = '$username' AND password = '$password' LIMIT 1";
        $result1 = mysqli_query($this->db->getDB(), $query1);

        $query2 = "SELECT U.username, U.email, E.value AS ecg_data, E.timestamp AS ecg_timestamp
           FROM ".$this->db_table." U
           LEFT JOIN ECG_data E ON U.username = E.username
           WHERE U.username = '$username' AND U.password = '$password'";
        $result2 = mysqli_query($this->db->getDB(), $query2);

        $ecg_data = array();
        $ecg_timestamp = array();

        $query3 = "SELECT U.username, U.email, B.value AS bpm_data, B.timestamp AS bpm_timestamp
           FROM ".$this->db_table." U
           LEFT JOIN BPM_data B ON U.username = B.username
           WHERE U.username = '$username' AND U.password = '$password'";
        $result3 = mysqli_query($this->db->getDB(), $query3);

        $bpm_data = array();
        $bpm_timestamp = array();

        $query4 = "SELECT U.username, U.email, P.ir_value, P.red_value, P.timestamp AS ppg_timestamp
           FROM ".$this->db_table." U
           LEFT JOIN PPG_data P ON U.username = P.username
           WHERE U.username = '$username' AND U.password = '$password'";
        $result4 = mysqli_query($this->db->getDB(), $query4);

        $ir_value = array();
        $red_value = array();
        $ppg_timestamp = array();
        $query5 = "SELECT username AS users, email AS emails, password as passwords FROM ".$this->db_table."";
        $result5 = mysqli_query($this->db->getDB(), $query5);

        $users  = array();
        $emails = array();
        $passwords = array();
```

Appendix G: These are the POST requests.

```php
<?php

    require_once 'user.php';
    $username = "";

    $password = "";

    $email = "";

    if(isset($_POST['username'])){

        $username = $_POST['username'];

    }

    if(isset($_POST['password'])){

        $password = $_POST['password'];

    }

    if(isset($_POST['email'])){

        $email = $_POST['email'];

    }
            $userObject = new User();

    // Registration

    if(!empty($username) && !empty($password) && !empty($email)){

        $hashed_password = md5($password);

        $json_registration = $userObject->createNewRegisterUser($username, $hashed_password, $email);

        echo json_encode($json_registration);

    }

    // Login

    if(!empty($username) && !empty($password) && empty($email)){

        $hashed_password = md5($password);

        $json_array = $userObject->loginUsers($username, $hashed_password);
        header('Content-type: application/json');
        echo json_encode($json_array);
    }
?>
```

Appendix H: Showing how the message output will be stored in the $json variable depending on the case which will later be sent to the mobile application where it gets analyzed and utilized for plotting.

```php
    if(mysqli_num_rows($result1) > 0){
        $row1 = mysqli_fetch_assoc($result1);
        $email = $row1['email'];
  $doctor = $row1['doctor'];
        $username = $row1['username'];
        $password = $row1['password'];

while ($row2 = mysqli_fetch_assoc($result2)) {
    $ecg_data[] = $row2['ecg_data'];
    $ecg_timestamp[] = $row2['ecg_timestamp']; // Add the ECG timestamp from each row to the array
}
while ($row3 = mysqli_fetch_assoc($result3)) {
    $bpm_data[] = $row3['bpm_data'];
    $bpm_timestamp[] = $row3['bpm_timestamp']; // Add the ECG timestamp from each row to the array
}

while ($row4 = mysqli_fetch_assoc($result4)) {
    $ir_value[] = $row4['ir_value'];
    $red_value[] = $row4['red_value'];
    $ppg_timestamp[] = $row4['ppg_timestamp'];
}

while ($row5 = mysqli_fetch_assoc($result5)) {
  $users[] = $row5['users'];
  $emails[] = $row5['emails'];
  $passwords[] = $row5['passwords'];
}
        $json['success'] = 1;
        $json['message'] = "Successfully logged in";
        $json['email'] = $email;
        $json['username'] = $username;
        $json['password'] = $password;
  $json['doctor'] = $doctor;
        $json['ecg_data'] = $ecg_data;
        $json['ecg_timestamp'] = $ecg_timestamp;
  $json['usernames']= $users;
        $json['emails'] = $emails;
  $json['passwords'] = $passwords;
  $json['bpm_data'] = $bpm_data;
        $json['bpm_timestamp'] = $bpm_timestamp;
        $json['ir_value'] = $ir_value;
        $json['red_value'] = $red_value;
        $json['ppg_timestamp'] = $ppg_timestamp;

    }else{
        $json['success'] = 0;
        $json['message'] = "Incorrect details";
    }
    mysqli_close($this->db->getDb());
    return $json;
```

.

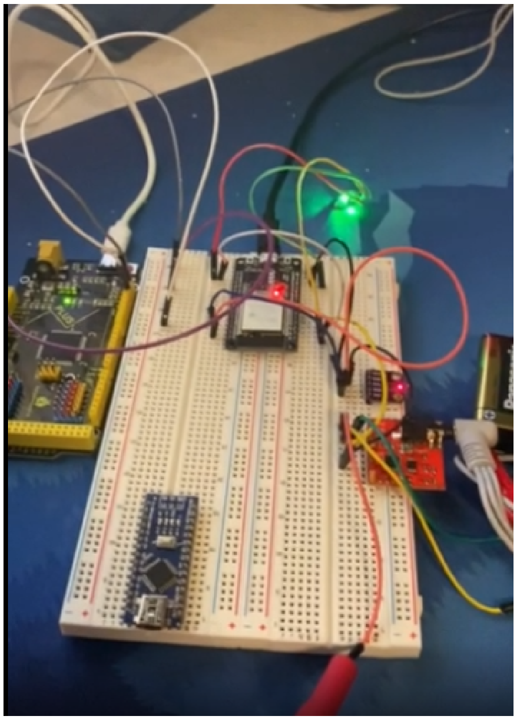
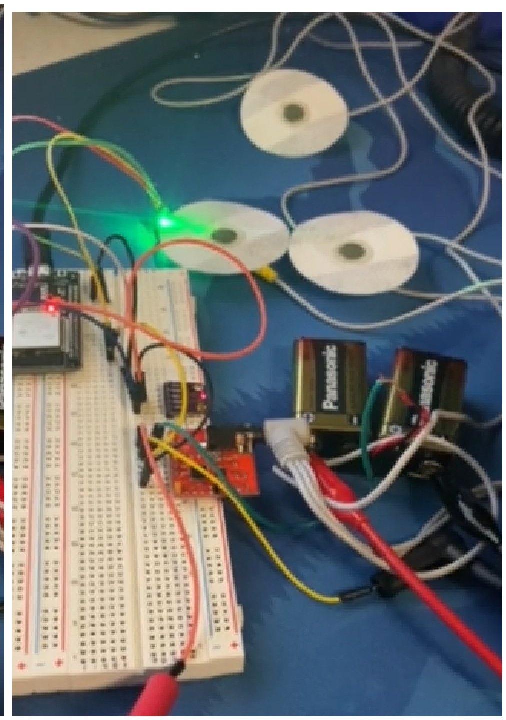
Appendix I: How sensor data is inserted into the database.

```php
// Check if the request method is POST
if ($_SERVER['REQUEST_METHOD'] === 'POST') {
        $json = file_get_contents('php://input');
    $data = json_decode($json, true);

    // Retrieve the user's credentials from the database
    $username = isset($data['username']) ? $data['username'] : null;
    $password = isset($data['password']) ? $data['password'] : null;
    $db = new DbConnect();
    $result = $db->getDb()->query("SELECT * FROM Users WHERE username='$username'");
    $user = $result->fetch_assoc();


    // Verify the password
        if ($user && md5($password)== $user['password']) {
        // If the password is correct, insert the sensor data into the database
        $ecgValue = isset($data['ecg_value']) ? $data['ecg_value'] : null;
        $irValue = isset($data['ir_value']) ? $data['ir_value'] : null;
        $redValue = isset($data['red_value']) ? $data['red_value'] : null;
        $bpm = isset($data['bpm']) ? $data['bpm'] : null;
        $timestamp = isset($data['timestamp']) ? $data['timestamp'] : null;;
    $sensors_data = new Sensors_data();
        if ($ecgValue !== null) {
            $sensors_data->insertECGData($ecgValue, $timestamp, $username);
            echo "ECG data inserted successfully.";
        }
        if ($irValue !== null && $redValue !== null) {
            $sensors_data->insertPPGData($irValue, $redValue, $timestamp, $username);
            echo "PPG data inserted successfully.";
        }
        if ($bpm !== null) {
            $sensors_data->insertBPMData($bpm, $timestamp, $username);
            echo "BPM data inserted successfully.";
        }
    } else {
        echo "Incorrect username or password.";
    }
} else {
    echo "Invalid request method.";
}
?>
```

Appendix J: Sensor Data Processed from Arduino

```cpp
// database
String url = "http://68.183.205.184/insertdata.php";
// last measurement time variables
unsigned long last_bpm_millis = 0;
unsigned long last_ecg_millis = 0;
unsigned long last_ppg_millis = 0;

void setup() {

  Serial.begin(9600);

  delay(1000);

  // setup time
  setTime(19,0,0,1,4,2023);
```

Appendix K: Arduino Interface



Panasonic
Panasonic

10. Contribution Matrix

The following contribution matrix provides details on how each member contributed to the content of the report both in writing and material. The list of tasks and contributors are shown in Table 10.1.

Table 10.1. Contribution matrix

Task	People			
	Mamun Hossain	Aaditya Rajput	Gobikah Balaruban	Manreet Kaur
Executive Summary	40%	5%	15%	40%
Report 1	25%	15%	25%	35%
Report 2	25%	45%	15%	15%
Revised Design Report	15%	15%	35%	35%
Ethical Considerations	40%	15%	15%	35%
Safety Considerations	40%	30%	20%	10%
Conclusions	30%	30%	20%	25%
Acknowledgements	30%	25%	20%	20%
References	15%	15%	35%	35%
Appendices	15%	40%	35%	20%
Contribution Matrix	25%	25%	25%	25%