# Lebanese American University

## School of Engineering

## Capstone Design Project II

## COE 596

**Project – Check My Heart**


Prepared by:

Guy Abi Hanna 201502542

Clara Akiki 201405063

Joseph Attieh 201501102

Supervised by:

Dr. Joe Tekli


June 10th, 2020

# Abstract

With the increasing hype in developing smart and portable health care solutions, there is a strong need for practical devices that perform prehospital electrocardiogram (ECG) capture and analysis, helping to detect heart anomalies at an early stage of the patient's life. The Lebanese Red Cross (LRC)'s first responders use a typical portable ECG device which provides accurate measurements yet does not allow any connectivity with other devices or platforms. In this context, the LRC has highlighted the need to connect their ECG devices with LRC headquarters or hospitals, so that the measured ECGs are displayed and analyzed on the fly by medical officers stationed at the headquarters/hospitals, providing support and helping the first responders make better decisions in the field. The project presented in this report aims at providing the Lebanese Red Cross with a solution that helps satisfy their ECG connectivity requirements. The suggested solution is a possible upgrade or replacement to the current ECG devices used in LRC ambulances. The proposed device detects the signal from the user's body using electrodes, amplifies the signal, and filters out the noise. Then, a microcontroller reads the signal and transmits it to a server for storage, solving the problem of transmission and storage. This server interacts with a dedicated Web Application allowing its user (the headquarters' medical officer, or the first responder) to visualize the ECG of the patient and various extracted features. The application will also display possible diagnosis of the patient by analyzing those features and feeding them into a dedicated machine learning (classifier) agent.

## Acknowledgments

We would like to express our deepest appreciation to all the people who helped us reach this stage of the project. A special gratitude to our advisor, Dr. Joe Tekli, who provided us with the constant support and motivation throughout the entire project. This product would not have been complete without his great effort and patience, and without his valuable and constructive guidance throughout our work.

We would also like to thank the Electrical and Computer Engineering department and the Laboratory Supervisors for providing us access to the labs and supplying us with the needed equipment and technical help. We would like to also acknowledge with much appreciation their help for entrusting us with the necessary material to complete the hardware aspect of the project.

We also want to thank Miss Grace Akiki, the medical student who dedicated her time to guide us and validate our results.

Last but not least, we would like to thank LRC officers Diana Krikorian and Shawky Amineeddine for sparing time out of their busy schedules to discuss the subject matter with us and share information and insight regarding LRC's needs in this context. We hope that the proposed work would be of use to the LRC and would help advance their noble cause.

# Table of Contents

# Tables of Figures

# Table of Tables

# 1 Introduction

Emergency Medical Service (EMS) is commonly faced with patients suffering from Heart Attack or Myocardial infarction. Following the common medical saying: “Time is muscle”; more of the patient’s heart muscle dies with every second that passes without proper medical attention, and the chances of the patient surviving diminish accordingly. That is why a better communication between the EMS and the hospital can save lives. By observing the health sector in Lebanon, one can realize that hospitals and emergency response teams like those of the Lebanese Red Cross (LRC) struggle from bad communication. Critical patients arrive to hospitals that are full or badly equipped to take care of them or cannot accept them for financial reasons, etc. All this time wasted can be avoided if the EMS sends the accurate patient information to the Emergency Officer with their respective Heart Monitor or electrocardiogram (ECG) before the patient arrives to the hospital. That way doctors are prepared in advance and can advise the EMS to give certain medications to the patient before arriving to hospital, so that proper medical care is started with minimal delays.

In Lebanon, the EMS lack adequate resources and coordination with hospitals. In 2014, the low out-of-hospital survival rate of heart failures in Beirut (i.e., around 5% [1], compared with 10.4% in the USA [2]) reveals the poor-quality of service of emergency medical teams and first responders. The weakness of health care measurements, especially when it comes to ECG analysis by medical heart experts, is creating serious obstacles for proper medical care of patients undergoing heart attacks, notably in rural areas where it takes a long time for the patient and first responder team to reach the hospital.

The aim of our final year project is to bridge that communication gap between the LRC first responder teams and LRC headquarters or hospitals, by building an electrocardiogram (ECG) reader from scratch, sending its measurements wirelessly to the hospital while the patient is still in the emergency vehicle, and storing the patient’s corresponding health records electronically for machine and human diagnosis.

In the following sections, we will discuss the background information of our system: connections, readings and analyses of the ECG. Then, we will review the existing solutions describing portable ECGs with transmission capabilities. Afterwards, we will go through the alternative approaches that could answer the needs of the LRC. We will continue our report with the proposed solution comprising the overall architecture as well as the hardware and software requirements of our apparatus. This will be followed by a detailed description of the two main parts of our implementation: ECG detection module and ECG storage and diagnosis module. Then, we will proceed with the experimental testing. Finally, we will complete the report with some necessary constraints and standards.

# 2 Background

## 2.1 Definition:

An electrocardiogram also known as EKG or ECG is a recording of the electrical activity of the heart. It is generated by placing electrodes on specific locations on the skin and performing several readings of the difference of potential [3]. The recording is represented through a graph of voltage versus time. The voltage detected by the electrodes represents the depolarization and repolarization of the heart muscle. The more electrodes or leads we use, the more we can visualize different electrical impulses across different areas of the heart. In particular, each electrode looks at the heart from a specific point of view.

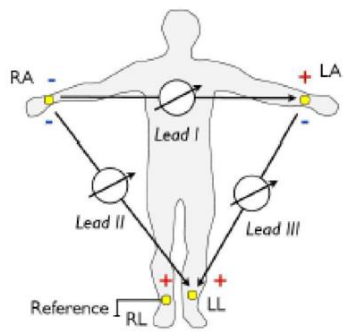



*Figure 1: Different views which correspond to different lead positions on the human body (3 lead ECG) [3]*

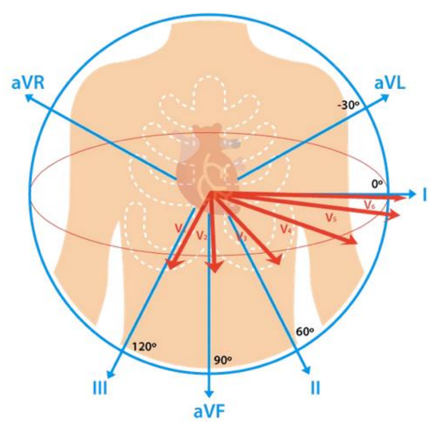



*Figure 2: Different views which correspond to different lead positions on the human body (12 lead ECG) [4]*

Several types of ECG exist and vary by the number of leads provided. Mainly two kinds of ECGs are used within medical applications:

- 3 Lead ECG

  Figure 1 depicts the three-lead configuration from the four electrodes attached to the body.

The aggregation of those three leads form the Einthoven triangle.

- Lead I represents looking at the heart from the right arm to the left arm
- Lead II represents looking at the heart from the left leg towards the right arm. This is the most common view
- Lead III represents looking at the heart from the left foot to the left arm

- 12 Lead ECG

Figure 2 depicts the 12-lead configuration from the 10 electrodes attached to the body: 6 electrodes on the chest, 3 electrodes on the limbs and 1 electrode on the Right Leg forming the ground (as shown in figure 3).

  - Lead I represents looking at the heart from the right arm to the left arm
  - Lead II represents looking at the heart from the left leg towards the right arm. This is the most common view
  - Lead III represents looking at the heart from the left foot to the left arm
  - aVR represents looking at the heart from the left arm and left leg to the right arm
  - aVL represents looking at the heart from the right arm and left leg to the left arm
  - aVF represents looking at the heart from the right arm and right leg to the left leg
  - V1, V2, V3, V4, V5 and V6 are precordial transverse leads that provide information of the heart's horizontal plane

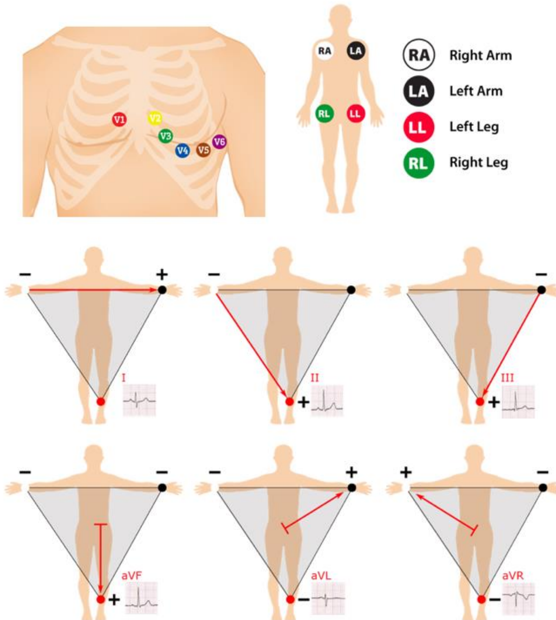



*Figure 3: Connections and leads of a 12 lead ECG [4]*

The difference between the two types of ECGs is the accuracy of the generated voltage graph since the 12 Lead device will look at the heart from different views and will be able to visualize all the leads readings. However, the compromise is vital when it comes to setting up the device and making the adequate connections. We'll discuss further our design choices in the upcoming sections.

We first define the term **depolarization** as the change during which the cell undergoes a shift in electric charge distribution, resulting in less negative charge inside it (it loses electrons, so it becomes positively charged) [5]. **In contrast, repolarization** refers to the change in potential that returns a cell to a negative value just after the depolarization. Those phenomena are showed in the figure below:

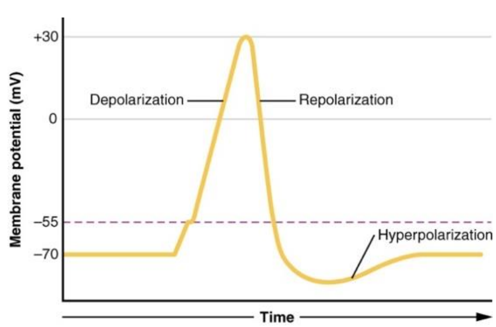



*Figure 4: Potential change in membrane during depolarization and repolarization [5]*

## 2.2 How to read the ECG?

Technically, the process of depolarization followed by a repolarization causes heart cells to contract and retract and hence the pumping of blood [6]. The electrical currents which are generated by the heart are spread throughout the body and hence are measured on the body surface. Figure 6 reveals the generic shape of an ECG. Each ECG cycles consists of 5 waves: P, Q, R, S, T corresponding to different phases of the heart activities [7]. The P wave represents the normal atrium (upper heart chambers) depolarization; the QRS complex (one single heart beat) corresponds to the depolarization of the right and left ventricles (lower heart chambers); the T wave represents the re-polarization (or recovery) of the ventricles. We will refer to Figure 5 to explain the different biological phenomenon behind each component of the ECG graph. We denote by atria the two upper chambers of the heart (right atrium and left atrium) and by ventricle the two lower chambers of the heart (right ventricle and left ventricle).

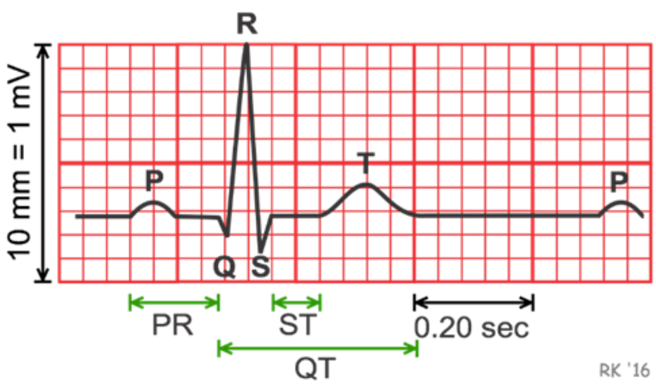



*Figure 6: Sample ECG Graph [7]*

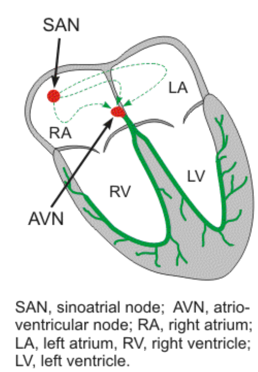



*Figure 5: Cross section of the heart [23]*

### 2.2.1 P Wave:

#### 2.2.1.1 Observations

The P wave occurs when both left and right atria are full of blood and the sinoatrial node fires. The signal causes both atria to contract and pump blood to the ventricles (lower chambers). It is also associated with an ***atrial depolarization*** since it involves a flow of charge from the sinoatrial node (SAN) through the atria. The separation between the P wave and the QRS complex is called the PR interval. This zero-voltage period represents the impulse traveling through the atrioventricular node (AVN). The PR interval usually ranges from 0.12 to 0.2 seconds.

#### 2.2.1.2 Diagnostics

Based on the P-wave, we could detect different heart anomalies:

| Abnormal patterns in P-wave | Possible causes | Consequences |
|---|---|---|
| Inverted | Sinoatrial block: the electrical impulse is delayed or blocked on the way to the atria | Fainting, altered mental status, chest pain, hypoperfusion, and signs of shock, ... |
| | Dextrocardia: The heart is in an abnormal location within the chest | Heart failure, infection, repeated pneumonias, … |
| Greater than 2.5mm | Right atrial enlargement | Heart failure, pneumonias, … |
| Longer than 100ms | Right atrial enlargement | |
| Longer than 200ms | AV block between atria and ventricles | First-degree heart attack |
| Invisible | Sinoatrial block: the electrical impulse is delayed or blocked on the way to the atria | Fainting, altered mental status, chest pain, hypoperfusion, and signs of shock, ... |

*Table 1: Anomalies detected from the P wave*

### 2.2.2 QRS Complex:

#### 2.2.2.1 Observations

The Q, R and S waves together are referred to as QRS complex (even if some of its components are missing). The QRS complex represents **ventricular depolarization**, which represents the lower part of the heart.

The ideal QRS complex is shown within Figure 2, however several other readings can be made depending on the lead’s positions. Figure 4 demonstrates different possible shapes that the QRS complex can have. Abnormal conduction of electrical impulses within ventricles could lead to changes in the QRS shape. Hence, different names are assigned as shown in Figure 7.

We should note that:

- Q wave is the first negative wave of the complex
- R wave is the first positive wave of the complex
- S wave is the first negative deflection after an R wave

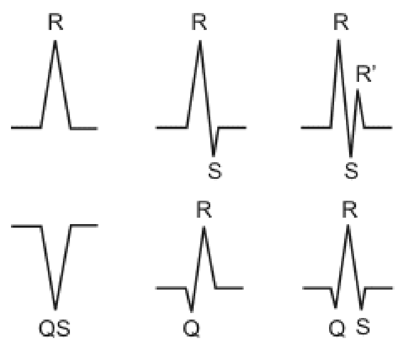



*Figure 7 - QRS Complex Shapes [7]*

### 2.2.2.2 *Diagnostics*

The most common abnormality is a longer QRS complex which means that conductions is weakened through the ventricles. The possible causes are showed in the table below:

| Abnormal patterns in QRS complex | Possible causes | Consequences |
|---|---|---|
| Duration greater than 100ms | Bundle branch blocks: Defect of the heart's electrical conduction system | Fainting and cardiac arrest |
| | Wolff-Parkinson-White Syndrome: Pre-excitation of the ventricles due to an accessory pathway | |

*Table 2: Anomalies detected from the QRS complex*

### 2.2.3 ST segment:

#### 2.2.3.1 *Observations*

The ST segments is the isoelectric period that follows the QRS complex. It represents the complete depolarization of the ventricles and marks the time for the ventricles to pump the blood to the lung and body [7]. In normal situations, it serves as the base line from which to measure the amplitudes of the other waveforms.

#### 2.2.3.2 *Diagnostics*

The anomalies that are diagnosed using the ST segment are showed in the table and figure below.

| Abnormal patterns in ST-segment | Possible causes | Consequences |
|---|---|---|
| Depression: Abnormally low below the baseline | Demand Ischemia: restricted supply of oxygen for body tissues | Chest pain, irregular rhythm, heart attack, … |
| Elevation: Abnormally high above the baseline | Supply Ischemia: restricted supply of blood for body tissues | |

*Table 3: Anomalies detected from the ST segment*


*Figure 8: ST segment elevation and depression [8]*

ST elevation is the hallmark of STEMI or ST elevation myocardial infarction which is a medical emergency requiring a reperfusion therapy to de-obstruct the stenotic coronary artery supplying the heart. Three broad categories of the areas of the heart are supplied by different coronary artery branches which can be mapped to ECG leads. This mapping is shown in the table below (it allows us to determine from the ECG graph which region of the heart needs reperfusion).

| Region of the heart | ECG leads involved | Arterial supply |
|---|---|---|
| Anterior | V1, V2, V3, V4 | Left anterior descending artery |
| Inferior | II, III, aVF | Right coronary artery |
| Lateral | I, aVL, V5, V6 | Circumflex artery |

*Table 4: Mapping between the region of the heart, its arterial supply and the ECG leads involved*

### 2.2.4 T wave:

#### 2.2.4.1 *Observations*

The T wave happens after the contraction empties the blood in the ventricles, so they begin to relax. It represents **ventricular repolarization**; hence ventricles' tissues are repolarized, and the heart is ready to pump at the end of this phase. Briefly, the main reason for the formation of this wave is that the last cells to depolarize are the same one that repolarize. In some cases, a U wave is introduced and may be seen following the T wave, which has no exact source and hence its formation remains unclear.

#### 2.2.4.2 Diagnostics

A failure of formation of the T wave signifies that no repolarization is happening. Hence, this will lead to a ***heart attack***. Usually, T wave malformation is linked with ST depression and hence causes ischemia.

We should note that the QT interval represents the whole time it takes for the ventricles' depolarization and repolarization. The interval of this happening ranges from 0.2 to 0.4 seconds depending on the heart rate. At high heart rates, ventricular actions durations will shorten.

| ECG component | Event | Sign in Leads |
|---|---|---|
| P-wave | Atrial depolarization. | **Positive** in leads I, II, aVF and V1 through V6 |
| QRS complex | Ventricular depolarization (atrial repolarization) | **Positive** in leads I, II, III, aVF, aVL, V5 and V6 |
| T wave | Ventricular repolarization | **Positive** in leads I, II, and V2 -V6 (follows QRS complex) |

*Table 5: Summary of the components of the ECG signal [9]*

## 2.3 ECG Characteristics:

Many heart characteristics can be deduced from the ECG graph. Hence, of the upmost used and referred to within medical context are the following:

- Rate: Heart Rate measured in beats per minutes (**bpm**). This can be determined as either:
        - Numerical: Count the number of QRS waves that happened in 60 seconds.
        - Graphical: One Square difference represents 300 bpm, Two Squares represent 150 bpm, Three Squares represent 100 bpm, Four Squares represent 75 bpm and Five Squares represent 60 bpm.
- Rhythm: Represented by the P-wave.
    - If the P wave exists and the heart rate < 60 bpm
        - The patient is suffering from bradycardia.
    - If the P wave exists and the heart rate > 100 bpm
        - The patient is suffering from tachycardia.
- Axis: The major direction of the overall electrical activity of the heart.
    - Different leads placement will cause different deviation in the PQRST points and waves.
- Conduction abnormalities: Represents a normal variation of heart Rhythm, that does not affect the patient's health usually.
- Hypertrophy: Examining the R maximum values, if it's more than 12 mm in amplitude then Left Ventricular Hypertrophy (LVH) exists.

# 3 Existing Solutions

Prehospital electrocardiography solutions can assess the cardiac rhythm in a pre-hospital environment, ensure early diagnosis, and accelerate the treatment time of heart abnormalities by enhancing the prehospital/hospital communication. Multiple systems are built worldwide in order to capture the ECG and transmit it.

## 3.1 STAT-MI network (Lifepak-12 Defibrillator and Bluetooth Wireless Communication):

The STAT-MI network is a system that uses ECG measurements inside a defibrillator and sends them wirelessly using the Bluetooth capability of that defibrillator. This approach was described in a paper and implemented in the University of Medicine and Dentistry, New Jersey with the support of Medtronic representatives and the assistance of Verizon Wireless. It uses the Lifepak-12 defibrillator/monitor to successfully transmit data to the Emergency department and offsite cardiologists. It leverages the Bluetooth feature introduced in 2004 that could be used to send the 12 lead ECG graph wirelessly [10]. The STAT-MI network transmits ECG data from the defibrillator device to the Bluetooth enabled Motorola phones held by the Emergency Medical Service (EMS) wirelessly using Verizon Wireless, then from these EMS phones to Lifenet (who sends ECG reports to hospitals or appropriate facilities) servers who sends ECG reports to the University Hospital's servers and eventually to the cardiologists for the purpose of diagnosing [11].

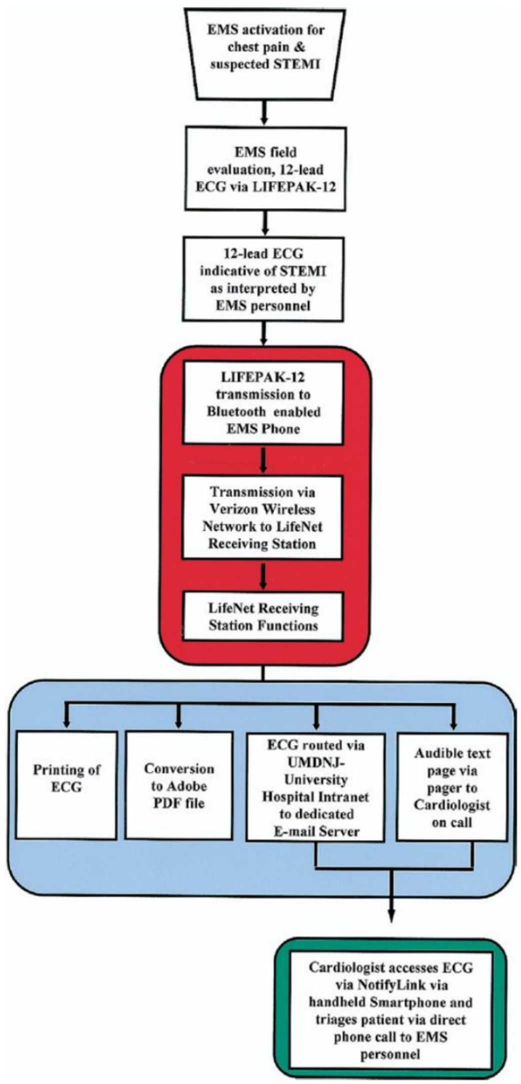



*Figure 9: Block diagram depicting the network of components of STAT-MI system extracted from [11]*

## 3.2 Zigbee and GSM Based Patient Health Monitoring System

The authors in [12] designed and implemented a patient monitoring system in the Army Institute of Technology Pune, India. This system builds a 3 lead ECG as well as other sensors in order to monitor the health of the patient. In fact, it collects real time data about the patient's health using multiple sensors, checks if he/she has normal health conditions by comparing the data to standard thresholds, updates the health records in the database on the doctor's PC, continuously monitors the health parameters by wirelessly transmitting the output of the sensors using the Zigbee technology and sends an SMS alert to the doctor in case of emergency using the GSM modem. Furthermore, four sensors are attached to the human's body: ***ECG sensor*** that detects the electrical activity of the heart, ***Heart sensor*** that measures the beats per minute (BPM) rate, ***Temperature sensor*** (LM35 series IC) that measures the temperature of the body in Celsius and ***Accelerometer sensor*** that measures the position of the patient (good position: standing or sitting; fallen down: sudden vertical change of position). Then a 32-bit ARM microcontroller collects the outputs of these sensors: ECG graph, heart rate, temperature and position, analyses them and sends them to the doctor's PC through Zigbee wireless transmitter (IEEE 802.15.4 standard wireless network that sends a small amount of data over short distances while consuming low power) and the doctor's mobile through the GSM modem (if the microcontroller detects any abnormalities). The block diagram describing this system is shown in the figure below.

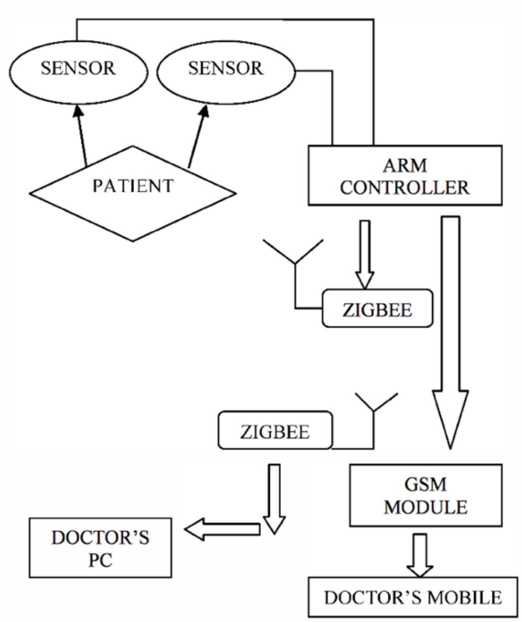



*Figure 10: Block diagram visualizing the interconnections between the components of "Zigbee and GSM based patient health monitoring system" extracted from [12]*

## 3.3 Handheld computer EKG device

The handheld computer EKG device is a patent system that uses a programmable Personal Digital Assistant (PDA) (e.g. com3 division of Palm Computing) which is a small portable PC containing a Wifi module. This system records a 12-lead ECG and transmits the measurements using a wireless communication network. In addition, the system optionally allows to measure the oxygen saturation. The patent information article describes the design of the circuitry used inside the interface of the PDA that connects the electrodes on the patient's body to the PDA device. Besides the oxygen sensor, ten electrodes (RA, RL, LA, LL, V1, V2, V3, V4, V5 and V6) are attached to the patient's body as shown in Figures 12 and 13. The output of these 10 electrodes are successively sent to 10 Analog to Digital converter channels through a sequence of instructions that are stored on the PDA. The latter channels are connected to a control circuit. The control circuit could be a microprocessor, microcontroller, or digital signal processor. It is the heart of the interface system since it takes the measurements from the A/D converter channels and relay the information to a serial transceiver that is electrically connected to the PDA (e.g. RS-232C serial connection, USB Serial connection). A general representation of the machinery of the interface of the personal digital assistant could be seen in Figure 11. Then the data could be wirelessly transmitted from the PDA to a remote location using the Wi-Fi module [13].

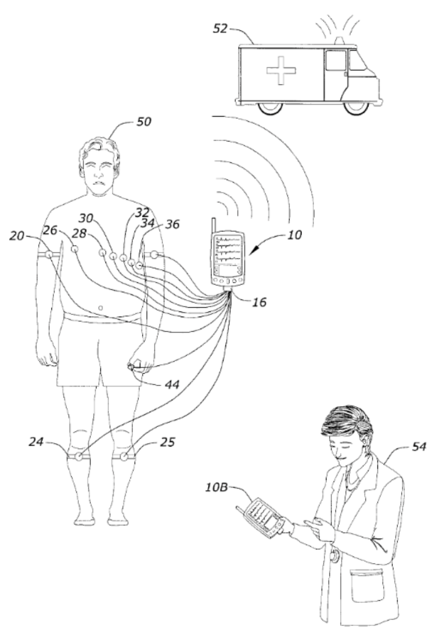


*Figure 13: Diagram showing the system of the invented technology [13]*

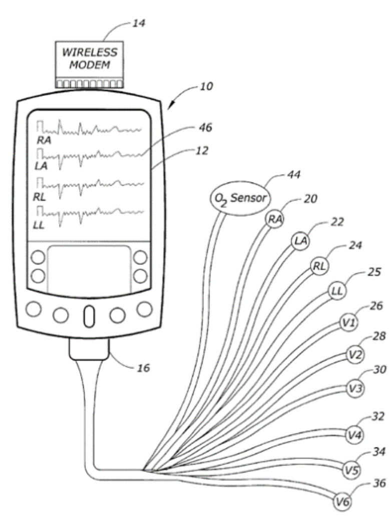



*Figure 12: Diagram showing the connections to the handheld ECG device [13]*

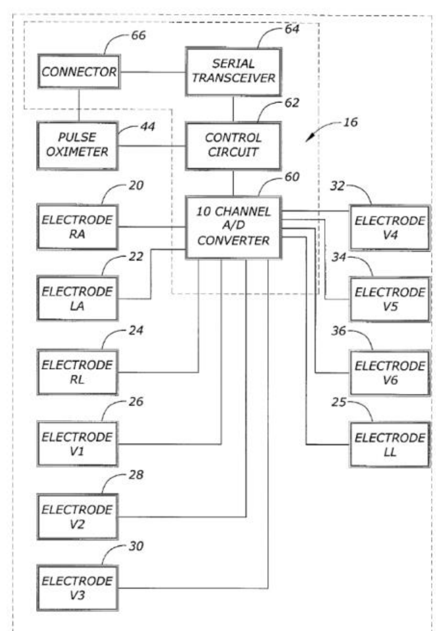



*Figure 11: General visualization of the circuitry of the personal digital assistant [13]*

# 4 Constraints and Standards

## 4.1 Constraints

### 4.1.1 Hardware constraints

| Constraint | Description | Course of Action |
|---|---|---|
| Availability Constraint | Availability of hardware components was limited due to the COVID-12 situation and due to the financial crisis. | • We opted out for a cheaper design with the components that we own (i.e. reduced the op-amp design to a 1 op-amp design). Refer to *Section 6.1*.<br>• The testing duration is limited by the lifetime of the disposable patches that we had (limited number of electrodes). |
| Compatibility Constraint | Data outputted from the ECG detection module must be compatible with the data stored in the database and methodology used for diagnosing. | The methodology proposed to label the ECG signal is unsupervised and does not rely on any dataset specific knowledge. Refer to *Sections 6.2.3.5* and *Appendix 1*. |
| Environment Constraint | Environmental conditions such as noisy rooms or dynamic setting could limit the accuracy of the detection and transmission done by the Hardware circuit | Testing the output of the product in the Laboratories which are noisy due to the presence of several electrical equipment. The filters in noisy rooms need to be narrower to filter out the noise. Refer to *Section 7.3*. |

*Table 6: Hardware Constraints*

### 4.1.2 Software constraints

| | Constraint | Goal | Verification | Course of Action |
|---|---|---|---|---|
| 1 | Reliability | The software shall provide a diagnostic decision with high accuracy | High accuracy (above 80%) and validated by a medical practitioner. | The graphs are visualized, checked and validated by a medical professional. Refer to *Section 7.2*. |
| 2 | Efficiency | The software shall quickly interact with the server and generate an intelligent diagnosis | Very short response time not exceeding 5 seconds and decision-making time not exceeding 1 minute, (considering reliable connection). | Testing shows that interaction with the server deployed on Heroku, took on average 2.5 seconds for generating diagnosis and reading ECG data. Front-end operations take 1 second to view all the data upon the reception of the request. |
| 3 | Availability | The software shall always be available. | No downtime exceeding 5 minutes. | The server that we are using to host the application (i.e. Heroku) is very reliable and always available. |
| 4 | Operational Cost | The software shall have a reasonable cost. | Affordable server service should not exceed $20. | The server used is free of charge (we used the Flask API). We also made use of the Heroku free tier plan which can handle up to 100 concurrent requests. |
| 5 | Portability | The software shall be accessible by users via the famous Web Browsers. | Use a technology that can deploy the code as a website or as a regular application. | The technology used is Unity (deployable on multiple platforms). As for the hardware, the setup uses a standalone architecture, where a 9V battery powers the overall circuit. Furthermore, the diagnosis tool can be used as a standalone software application. Refer to *Section 6*. |
| 6 | Maintainability | The software shall be easy to update and extend based on the changing requirements of the LRC. | The software shall have a modular and efficient code (implementing common models and design pattern) | The codes are optimized (as small as possible) and followed multiple design patterns (Microservices, Webserver, MVC, etc.) |
| 7 | User constraints | - The user must be connected to the Internet in order to be able to have access to the application.<br>- The user must be accredited, by receiving login credentials from the administrator, in order to use the system | | |

*Table 7: Software Constraints*

### 4.1.3 General constraints

The project has several technical, economic and social constraints associated with it. These general constraints regarding the overall system are summarized in the table below:

| Constraint | Param. | Description | Course of Action |
|---|---|---|---|
| **General constraints** | Time | A working prototype need to be submitted by the end of Spring 2020. | The working prototype with a full-fledged software product is built successfully. The prototype uses 3 electrodes to provide one view of the heart (depending on the location of the electrodes). |
| | Scope | The final prototype is expected to include electrodes (3 to 10), different type of sensors and a web application. | |
| **Economic constraints** | Cost | The proposed solution should meet all the requirements while keeping the lowest cost possible (refer to the cost analysis below). | The cost is successfully met (especially after replacing the 3-opamp circuit with a one op-amp design). |
| **Functional constraints** | Power needed | Power supply should be enough to enable the device to perform its tasks properly. | The battery used is rechargeable and is able to power the circuit. |
| **Safety constraints** | Human | The electrodes need to be connected to the user. No electronic device should be left exposed to protect him/her. Furthermore, many components should be added to the circuit to ensure the safety of both the patient and the expert. | Resistors are added to make sure that the currents inside the circuit never exceeds 1mA. No additional components are added due to the lack of resources (however, this does not compromise the safety of the team, as the additional components are usually added for prevention) |
| | Privacy | Sensitive data collected from the patient should remain safe since the patient might not be awake during the ECG (emergency cases). | The database is only accessible by the flask server (password protected). Furthermore, the IP address of the server connected to the MangoDB is only whitelisted. |

*Table 8: Illustration of constraints*

### 4.1.4 Cost Analysis

The components of this project are subdivided into the following groups:

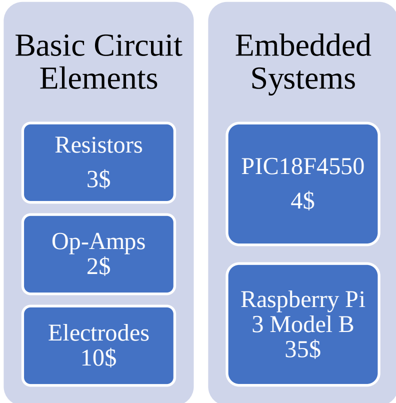



*Figure 14: Cost Analysis*

Therefore, the total cost for setting up the application with the associated hardware is 54$. We should note that the price of electrodes could go even lower (depending on the quality).

## 4.2 Standards

| Standard | Reference | Description |
|---|---|---|
| Wi-Fi | IEEE 802.11 | The project needs internet access (to send the ECG to the server and to be able to retrieve it to the mobile/web app) |
| UART | IEEE 1355 | A universal asynchronous receiver-transmitter is a computer hardware device for asynchronous serial communication in which the data format and transmission speeds are configurable. |
| RESTful API (HTTP, JSON) | RFC 2616 – RFC 7159 | The application uses HTTP requests to GET, PUT, POST and DELETE data. The JSON format is used for fast transmission of data between subordinate units. |
| ECG Processing Stages | ISO 11073-91064:2009 | ECG computer processing is reduced to three principal stages:<br>1. Data Acquisition<br>2. Pattern Recognition<br>3. Diagnostic Classification |
| American Heart Association (AHA) | AHA/ACC/HRS Circulation Journal. 2007;115:1306–1324 | Explains How ECG is derived and displayed. The journal establishes standards that will improve the accuracy and usefulness of the ECG in practice. |
| Analog to digital converter | IEEE 1241 | Provides common terminology and test methods for analog-to-digital converters whose output values have discrete values at discrete times, i.e. are quantized and sampled. |

*Table 9: Standards used*

# 5 Proposed Solution

The proposed approach consists of implementing our own ECG detection, transmission, storage and diagnosis device from scratch. The device proposed should be able to safely extract the data from the electrode patches attached to the patient’s body, process the data digitally, transmit it wirelessly, analyze it in order to detect heart abnormalities, store it in a database and diagnose possible ECG abnormalities. Since the approach implements the device from scratch, it provides a full flexible device where components could be upgradable and updatable following the user’s request. This section will present an overview of the whole system.

## 5.1 Overall architecture

The product to be developed should be able to capture the ECG signal from the patient's body, diagnose it and add it to his/her records. The overall architecture (context model) is shown below:

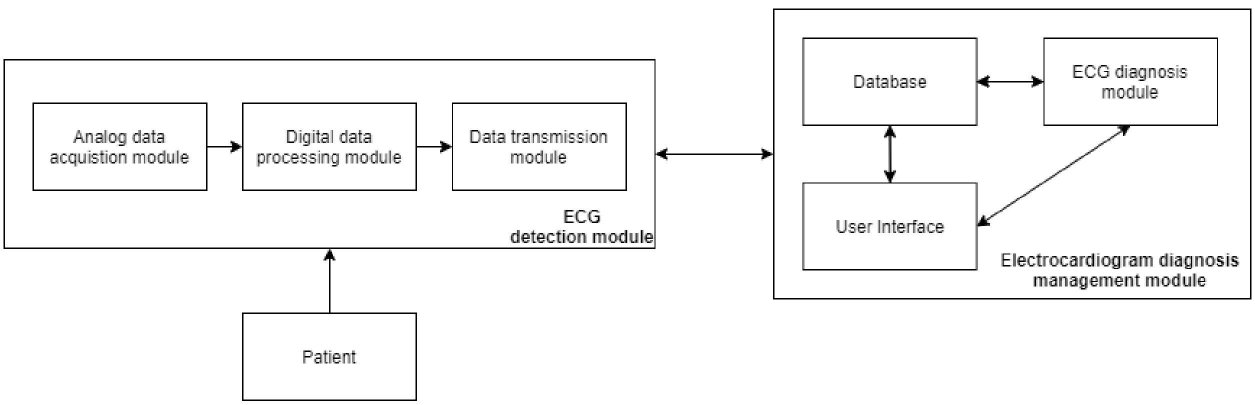



*Figure 15: Overall System Architecture*

Let's describe the various entities in the figure:

1. ECG detection module: The ECG detection module consists of all the hardware components that provide i) data acquisition from the patient, ii) digital processing of the data, and iii) transmission to the database. In other words, this system measures the ECG signal from the patient's body and sends this reading to the Database.
2. Electrocardiogram Diagnosis Management module: This module represents the set of subcomponents that allow the user to manage the ECG records stored in the database and request their diagnosis. Furthermore, it provides the user with an interface that could be accessed from any device connected to the internet. Let’s describe the submodules:
    a. Database: The Database will be able to communicate with the User Interface to allow the authentication of the user as well as the management of the various

records stored. It will also be able to store the data acquired from both the ECG detection module and the ECG diagnosis module.

b. ECG Diagnosis module: The ECG diagnosis module will be able to receive an ECG signal and return the corresponding diagnosis.

c. User Interface: The User Interface will be able to capture the requests of the user and handle it accordingly: if the request is a storage, retrieval, modification or deletion of a record, the request is communicated to the Database. If the request is to generate a diagnosis for a certain ECG signal, the request is communicated to both the ECG diagnosis module and the database.

## 5.2 Functional requirements

In this section, we will highlight the functional requirements for the different parts of the proposed system. We will distinguish between the functional requirements of the hardware components and the functional requirements of the software components.

### 5.2.1 Hardware requirements

The hardware components of the proposed system shall allow the user to capture the ECG signal of the patient's body and transmit the signal to the database of the *Electrocardiogram Diagnosis Management module*. Since most of the hardware components are located in the ECG detection module, the hardware requirements should be met by this module.

#### 5.2.1.1 *Overall architecture*

Thus, we propose the following subsystems in the diagram below:

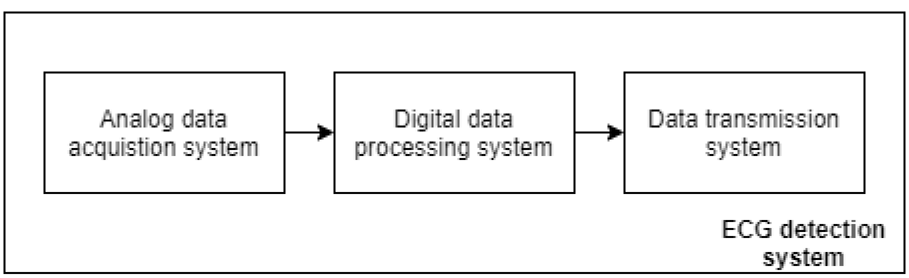



*Figure 16: Overall architecture of the ECG detection system*

From the following diagram, we could define the role of each of those subsystems:

a. Analog data acquisition system: the analog data acquisition system shall be able to capture the ECG signal from the patient's body.

b. Digital data processing system: the digital data processing system shall be able to process the captured ECG signal.
c. Data transmission system: the data transmission system shall be able to transmit the ECG signal to the database of the *Electrocardiogram Diagnosis Management System*.

#### 5.2.1.2 *Design choices*

In this section, we will discuss the possible design decisions to consider concerning the hardware components of the project (i.e. covering the different parts of the ECG detection system).

*1. Analog data acquisition system:*

This component is responsible for detecting the ECG signal from the patient's body. The signal to be detected ranges from 1 to 5 mV with a frequency bandwidth ranging from 0.05 to 100Hz [16]. Thus, the analog data acquisition system should i) capture the signal then ii) amplify its amplitude in order for the signal to be detected by the Digital data processing system.

i) To capture the signal caused by the activity of the cardiac muscles, there is a need for ***electrodes***.
   1) The first design decision lies in choosing the number of electrodes used by the system. As mentioned before, the most common configurations of the ECG are the 12-lead and the 3-lead configurations. The 12-lead ECG uses six chest electrodes (V1 to V6) and four limbs lead electrodes (left arm, right arm, left leg and right leg) while the 3-lead ECG uses only 3 electrodes (left arm, right arm and left leg). A higher number of electrodes used will lead to a better diagnosis of heart disorders. However, we will need more hardware components to be able to handle the different signals (i.e. multiplexers and wires) and minimize interference (i.e. shielded cables). The multiplexers work as lead selectors controlled by the brain of the system (the microcontroller in our case) which iterate between the 12 channels each 15 seconds.
   2) The type and quality of the electrode has a great impact on the accuracy of the ECG detected. Electrodes exist in different shapes and sizes. It is recommended that all electrodes in the same system should be of the same brand to help minimize noise. We distinguish several types of electrodes [17]:

- <u>Wet electrodes vs Dry electrodes:</u>

  Wet electrodes are generally made of silver/silver chloride material (Ag / AgCl). They use an electrolytic gel material as a conductor between the skin and the electrode. In contrast, dry electrodes consist of a single metal that acts as a conductor between the skin and the electrode. Usually, dry electrodes have a wider variation of skin-to-electrode impedances due to contact quality. As a result, amplifier input impedances and bias currents might result in higher noise levels.

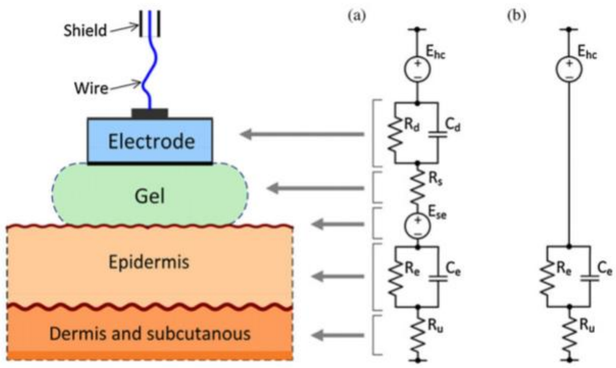



*Figure 17: Electrical model for electrode-to-skin interface (a) gel electrode and (b) dry electrode [24]*

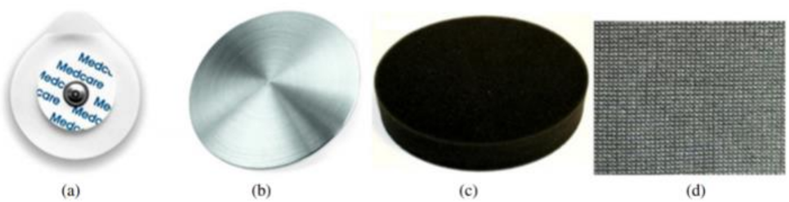



*Figure 18: Electrode categories [24]*

- <u>Active electrodes vs Passive electrodes:</u>

  The active electrodes have a pre-amplification module immediately after the conductive material between the skin and the electrode while passive electrodes do not have this pre-amplification module. This allows the signal to be amplified before additional noise is added between the electrode and the system that would capture, process or amplify the signal. It is usually recommended to use active electrodes in case of motion of the individuals or when working in areas with considerable electromagnetic noise. However, active electrodes generally have a higher price and are heavier than passive ones.

ii) To amplify the signal that was captured, there is a need for ***amplifiers***.

As we already discussed, amplifiers are needed in order for the ECG signal to be detected by the Digital data processing system. There exist many configurations of operational amplifiers than can be used for this task. Since we are measuring the voltage difference between both electrodes, we should use the ***differential*** configurations of the ***amplifiers***. It is recommended to use the ***instrumental amplifier***. It is a differential amplifier with additional input buffer stages which makes it easy for impedance matching with the previous stage. This type of amplifiers is commonly used in industrial test and measurement applications. The instrumentation amplifier is also characterized by a low offset voltage, high common mode rejection ratio (CMRR), high input resistance and high gain.

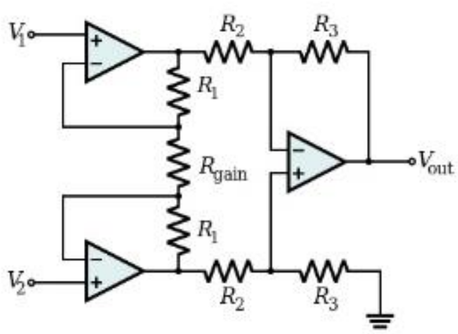



*Figure 20: Circuit of an instrumental amplifier*

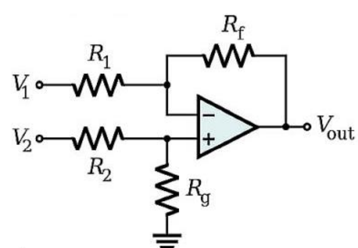



*Figure 19: Circuit of a differential amplifier*

2. *Digital data processing system:*

This component should be able to i) digitize the captured analog signal and ii) remove noise present in the signal.

i) There is a need to digitize the signal captured in order for it to be processed by the system. Thus, there is a need for an analog to digital converter. We could either use a microprocessor equipped with an A/D module or buy our own module.

ii) The amplification of the ECG signal will also result in the amplification of unwanted noise. If the frequency of interference is known in advance, it can be removed using a filter. One source of interference is the one associated with power lines. This type of noise is abundant indoors, since electrical systems in buildings uses AC power delivered at a certain frequency. To remove this noise, a 50-Hz to 60-Hz ***Notch filter*** could be used. Notch filters combine both high and low pass filters to create a small region of frequencies to be removed (in our case 50-60Hz noise). Furthermore, according to standards recommended by the AHA, the signal path requires a bandwidth of about 0.5 to 150 Hz for the ECG signal to be visualized clearly [18]. To achieve this, we could use a ***High Pass Filter*** at 0.5Hz (single pole/first order) and a ***Low Pass Filter*** at 150Hz (double pole/second order) (or a ***Bandpass Filter*** from 0.5Hz to 150Hz). Low pass filters on the ECG are used to remove high frequency muscle artifact and external interference while high-pass filters remove low-frequency components such as motion artifact, respiratory variation, and baseline wander. This minimizes noise content and anti-aliasing (signal attenuated outside of this range). One design decision is whether to use software filters (i.e. filters that are applied on the digital signal) or hardware filters (i.e. filters that are applied on the analog signal). The most common configurations of the hardware filters are showed below:

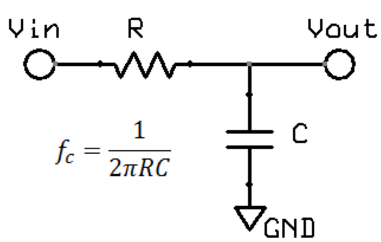



*Figure 21: Low Pass Filter [25]*

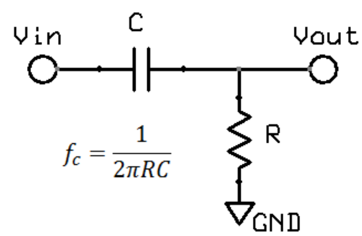



*Figure 22: High pass filter [25]*

*3. Data transmission system:*

This component should be able to transmit the acquired digital signal to the *Electrocardiogram Diagnosis Management System.* Thus, this system shall have a component that supports wireless transmission of the data.

### 5.2.2 Software requirements

The software requirements mainly cover the requirements that should be met by the *Electrocardiogram Diagnosis Management System*. In this section, we will first go over the roles of the users that can use this system. Then, we will state the functional requirements that need to be met.

#### 5.2.2.1 *User characteristics*

The Electrocardiogram Diagnosis Management System has mainly 3 types of users: ECG operator, medical practitioner and system administrator. In this section, we will list the functions provided by the program to each type of user.

*1. ECG Operator:*

ECG Operator has the most basic user role. In other words, any user of the system under any role will be able to perform the functionalities attributed to the ECG Operator. This is expressed by the arrows in the use case diagram (Figure 22) highlighting the inheritance relationship between the ECG Operator, the Medical Practitioner and the System Administrator. The ECG operator will be able to either login directly to the application using the credentials or request credentials to be approved by the system administrator. The user must be logged in in order to have access to the functionalities of the system (shown in the use case diagram with an includes relationship). The user will also be able to logout from the application. Furthermore, the user will be able to update the records of a patient if needed. If the patient is not in the database, the user should be able to add a patient. Moreover, the user will be able to retrieve the ECG signals that are measured by the ECG detection system and stored in the database, select an ECG signal to request its diagnosis from the ECG diagnosis system and attribute both diagnosis and ECG to a certain patient. At this stage, since the diagnosis is only generated and is not yet confirmed by a medical expert, it will be associated to a dummy patient and saved under Unconfirmed diagnosis. In addition to that, the user will be able to view the history of the commands he performed, as well as update his/her profile and access statistics related to the data in the database. Any user is able to request a change in the role request (i.e. to become either an operator, a medical practitioner or a system administrator). However, only the system administrator is able to approve such requests.

*2. Medical Practitioner:*

The medical practitioner will have access to the same functionalities available to the ECG operator. Additionally, the user having this role will be able to fetch the list of ECG signals and their diagnosis generated from the ECG diagnosis system that are still listed as unconfirmed. Then, he/she will be able to approve/disapprove machine diagnosis and add a textual human diagnosis in order to save it in the database.

*3. System Administrator:*

The system administrator will have access to the same functionalities available to the ECG operator. Additionally, the user having this role will be able to search for the users in the system and update their user role if requested. Furthermore, he/she will be able to approve the account requests of the other users. The list of functions provided are showed in the use case diagram below:

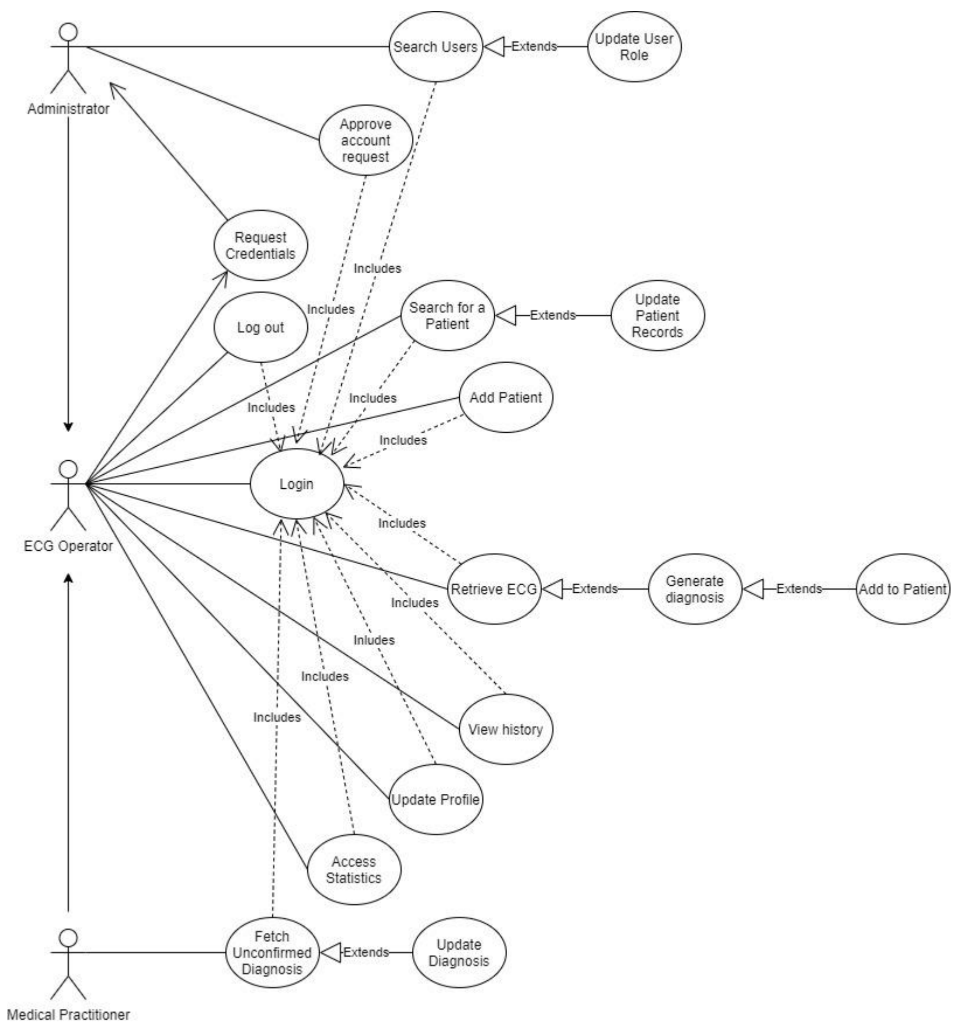



*Figure 23: Use case diagram*

#### 5.2.2.2 *Functional requirements*

<u>All Users:</u>

1. The system shall allow users who have existing accounts to log in using their credentials (the user shall not be able to use the application without an account).
2. The user shall be able to logout from the system.
3. The user shall be able to use his/her credentials to login.
    i. If the user's credentials exist in the database, then access is granted
    ii. If the credentials do not exist in the database, the user shall request credentials approval. Once the approval is granted, the user shall be able to access the application upon login.

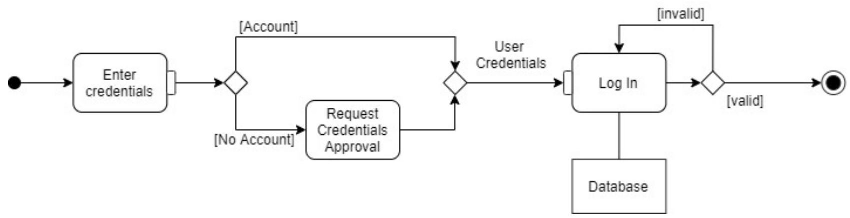



*Figure 24: Process Model of the login/request credentials activity*

4. The user shall be able to provide search metrics to the database, then select a single record of a patient to visualize.
5. The user shall be able to modify or delete the entry visualized after a search request.

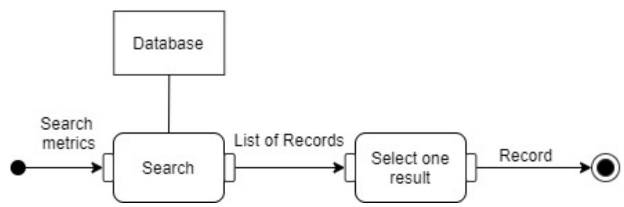



*Figure 25: Process model of the search patient activity*

6. The user shall be able to add a patient to the database.
7. The user shall be able to view his/her history.
8. The user shall be able to update the information in his/her profile.
9. The user shall be able to access statistics related to the system.
10. The system shall be able to retrieve an ECG signal from the database, request a diagnosis of the ECG and associate it with a certain patient. This requires that:
    i. The user shall be able to retrieve all ECG recordings not associated with a certain patient from the database then select one ECG record.
    ii. The user shall be able to request a diagnosis of the selected ECG signal.
    iii. The user shall be able to select a patient or add a patient corresponding to the ECG signal selected (could use a dummy patient for fast addition).
    iv. The user shall not be able to save a diagnosis in the database (i.e. associate diagnosis and ECG to a patient) without selecting a patient (visualized as a fork in the process model below).

v. The system shall be able to add the ECG signal and the diagnosis to the patient's records as being *unconfirmed*.

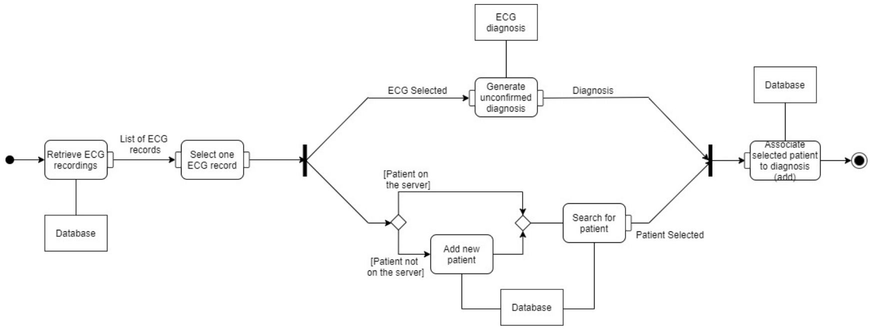



*Figure 26: Process model of the "associate diagnosis to patient" activity*

<u>Medical Practitioner:</u>

11. The medical practitioner shall be able to retrieve a list of ECG signals and their unconfirmed diagnosis from the database.
12. The medical practitioner shall be able to select a single diagnosis to confirm and update it accordingly.

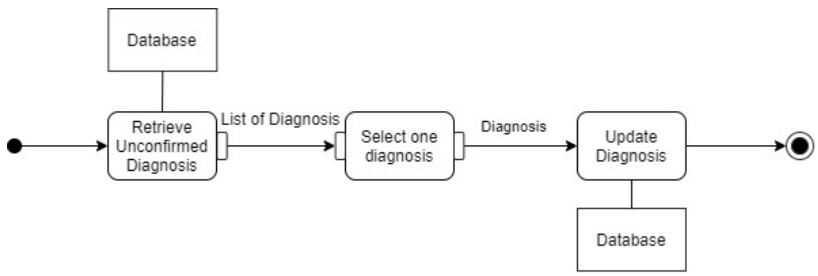



*Figure 27: Process model of the "confirming and updating the ECG diagnosis" activity*

<u>System Administrator:</u>

13. The administrator shall be able to search for the users and update their role.
14. The administrator shall be able to retrieve a list of credential requests from the database.
15. The administrator shall be able to select a single request and then shall either:
    i. Reject the creation of a new user (deleting the request)
    ii. Accept the request and add the credentials to the database.

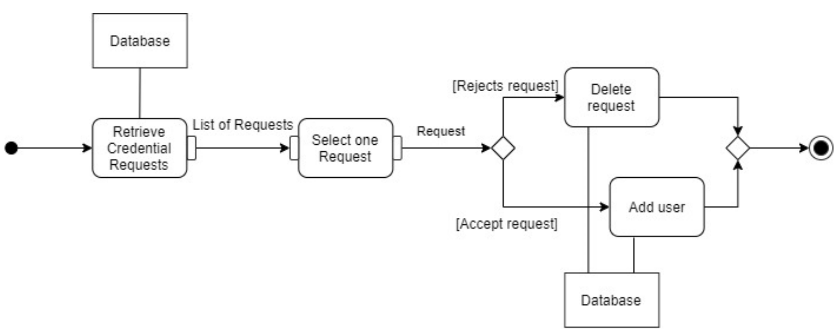



*Figure 28: Process model of the "approve/reject credentials request" activity*

# 6 Implementation

In this section, we will talk about the implementation details of the system proposed. We will be going over the detailed implementation of the ECG detection system as well as the Electrocardiogram Diagnosis Management System.

## 6.1 ECG detection system

As we have previously proposed, the ECG detection system consists of the analog data acquisition system, digital data processing system as well as the data transmission system.

### 6.1.1 Choice of technologies

a. Analog data acquisition system: the analog data acquisition system is responsible for capturing the signal from the patient's body then amplifying the signal for it to be detected by the next system.

   i) Electrodes:

   - The ECG device that we are developing is a one lead ECG device. Choosing to implement a 1-Lead ECG device rather than a 3-Lead device was done due to budget constraints. The depreciation of the Lebanese currency as well as the lack of access to the hardware components following the pandemic forced the team to reduce the scope of the device implemented. Building a 1-lead device would provide one view of the heart compared to the 3-lead device which provides 3 different views of the heart. Covering more views of the heart affect the confidence score of the anomaly detected within one view, since viewing it on other perspectives confirms its presence. We should note that the 1-lead ECG device proposed is a prototype that could be replicated 3 times to form a 3-lead ECG device where the location of the electrode differs between each 1-lead ECG device used.
   - The electrodes chosen are wet passive electrodes. Since the proposed solution intends to be budget friendly, the electrode chosen was passive (active electrodes are more expensive and less available). Furthermore, wet electrodes are the most commonly available electrodes used in hospital settings.

ii) Amplifiers: the amplifiers are needed in order to amplify the ECG signal, for it to be captured by the next system. Multiple configurations were considered. The simplest, smallest and cheapest configuration of the operational amplifiers is showed below, with the following resistor values $R_0 = 2k\Omega$, $R_1 = 2k\Omega, R_3 = 10M\Omega$ and $R_f = 100k\Omega$ :

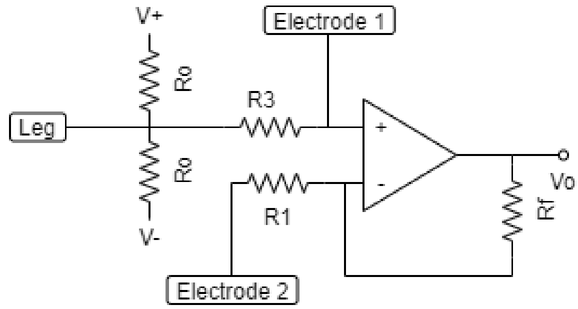



*Figure 29: Circuit of the amplifier used in order to amplify the ECG signal*

Let's explain the circuit above by deriving the proper formulas:

- Let's first examine the operational amplifier configuration used:

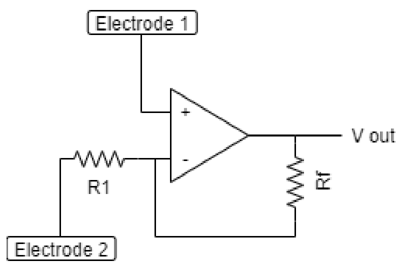



*Figure 31: Same circuit as above (ECG amplifier)*

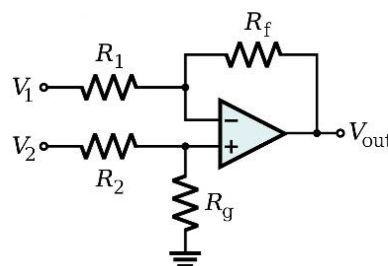



*Figure 30: Differential Amplifier*

The op-amp proposed has the same configuration as a differential amplifier. Using superposition, the gain of a differential operational amplifier can be derived to be: $V_{out} = \frac{(R_f+R_1)R_g}{(R_g+R_2)R_1} \times V_2 - \frac{R_f}{R_1} \times V_1$

By comparing both designs, we could clearly identify $R_f$ and $R_1$ and that $R_2 = 0$ (closed circuit, i.e. wire) and $R_g = \infty$ (open circuit). Thus, having $R_f = 100k\Omega$ and $R_1 = 2k\Omega$, the gain of the amplifier is equal to 50:

$$
\begin{aligned}
V_{out} &= \frac{(R_f+R_1)R_g}{(R_g+R_2)R_1} \times V_{electrode1} - \frac{R_f}{R_1} \times V_{electrode2} \\
&= \frac{(100+2)\cancel{R_g}}{(\cancel{R_g+0})2} \times V_{electrode1} - \frac{100}{2} \times V_{electrode2} \\
&\approx 50\ (V_{electrode1} - V_{electrode2})
\end{aligned}
$$

- Let's talk about the resistors $R_0$. Those resistors are connected to the positive and negative ports of the battery that powers the amplifier. They act as act a voltage divider that pulls the voltage in the body (i.e. leg) near the optimal input voltage for amplification by the op-amp (i.e. the middle).

  The voltage from the battery sets the range of voltages that can be amplified (i.e. sets the maximum and minimum outputs of the amplifier, making sure that the output of the amplifier cannot exceed the values set by the battery). We should make sure to choose $R_o$ to be small compared to $R_3$ (to draw minimum current in $R_3$). Furthermore, $R_3$ is chosen to be high in order to minimize the current entering the body of the patient from leg and electrode.

b. <u>Digital data processing system:</u> the digital data processing system shall be able to process the captured ECG signal. To perform the task, we need to use a microcontroller (pic18f4550) and a microprocessor (raspberry pi). We should note that the codes used are documented in Appendix 2. This section will only be covering the logic behind some of the choices taken.

   i) The first step consists of using the analog to digital module of the pic in order to capture the ECG signal. Due to the delays introduced by the transmission of the digital signal to the Raspberry Pi, there is a need to have as many consecutive samples as we can. Therefore, we proceeded by:

   - Connecting the analog signal that need to be digitized to 8 analog ports of the PIC18F4550: this will allow multiple consecutive samples to be acquired by the A/D module.
   - Storing as many samples before transmitting: this will allow us to skip the overhead related to the transmission to the pic.

   ii) The second step consists of transmitting the digital signal serially from the pic to the Raspberry Pi. PIC18F4550 has an in-built Universal Synchronous Asynchronous Receiver Transmitter (USART module) that allows asynchronous communication. At this stage, we are transmitting a batch of readings at a time. This is done by capturing readings and storing them in a buffer. Once the buffer is full, the data is transmitted to the Raspberry pi.

c. <u>Data transmission system:</u> the data transmission system shall be able to transmit the ECG signal to the Server. At the server, filtering techniques will apply to the signal in order to filter out the noise. Then, the filtered signal will be added to the database. This will allow the user to manage the ECG records using the *ECG diagnosis management system*.

   i) The transmission is done using Restful API. The data is transmitted as a json file using a POST request to the server.

   ii) The filtering of noise is done on the server side after the reception of data. After filtering the signals, they are added to the database. This task was originally designed to be included in the *Digital data processing system*, but was moved from the Raspberry Pi to the server to relieve the Raspberry Pi from the burden of filtering and to give us more flexibility in finetuning the code and the filters (easier to manipulate the code of the server). The filters are software-based filters rather than hardware-based filters since they provide an easy and more accurate filtering solution.

   The filters used are:

   - Band reject filters for frequencies ranging from 50 to 60Hz
   - Bandpass filters for frequencies ranging from 1Hz to 200Hz as instructed by the American Heart Association. However, the filters need to be narrower in environments where noise is found to be great (i.e. the bandpass filter need to be narrowed, since the ECG frequency components are suppressed by the huge noise).

For details concerning the code, kindly refer to the Testing section.

### 6.1.2 Circuit Diagram

After describing the components of the ECG detection system, we are going to give an overview of the flow of the signal from the patient to the database from a hardware perspective. After connecting the electrodes to the patient (the non-blue wires in the circuit), the user initiates the data detection system by pushing the button. The LED lights up to signal that the data acquisition started and turns off when the data acquisition ends. Doing so, the ECG signal travels through the operational amplifier to the Analog to Digital module of the PIC18F4550 chip. The PIC18F4550 transmits the digitized signal using the USART module (the transmitter pin of the PIC is connected to the receiver pin of the Raspberry Pi). The Raspberry Pi receives the data and transmits it to the server using the Wireless module. On the server side, filtering techniques are applied to the signal to clean it from the interference of the environment. The filtered signal is transmitted to the database.

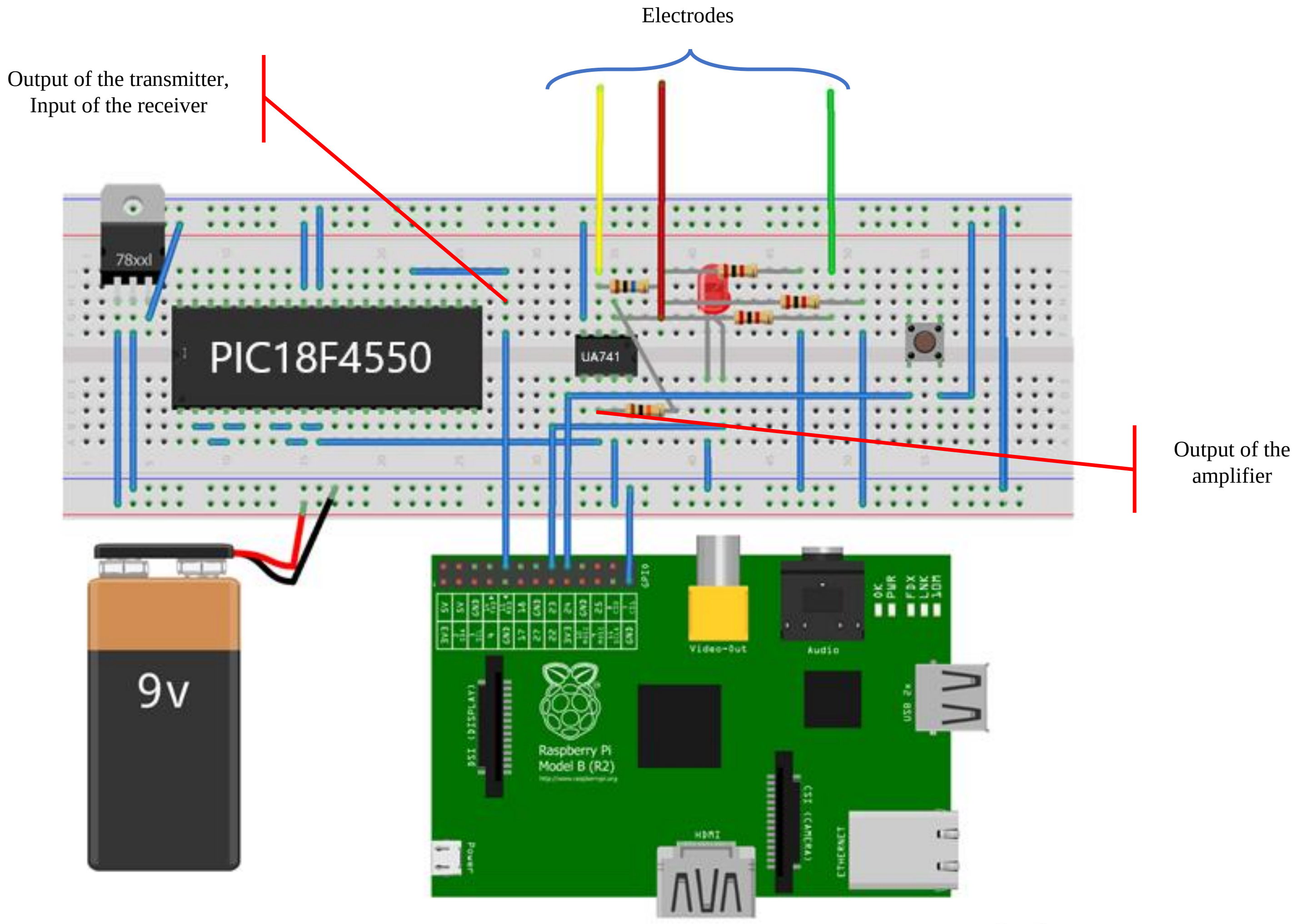



Figure 32: Circuit Diagram

The components showed in the circuit above are:

- 9 V battery
- Voltage regulator (used to regulates the voltage to 5V to power the pic)
- LED and Push button
- PIC18F4550
- Raspberry PI 3
- UA741 (the operational amplifier)

## 6.2 ECG Diagnosis Management System

In this section, we will talk about implementation details tackling the software part of the capstone project. The main components used in the implementation of the diagnosis management system are the webserver, the database, the diagnosis and the graphical user interface system.

### 6.2.1 Choice of technologies:

#### 6.2.1.1 *Storage of the data*

As mentioned previously, the application's data should be stored in a database that supports efficient insertion, update and removal of the records needed. The database chosen is MongoDB. MongoDB is a cross-platform document-oriented database that is classified as a Not Only SQL (i.e. NoSQL) database. It uses JSON-like documents in order to represent and store the data. We chose MongoDB as it is known for its efficiency, availability and scalability. Furthermore, it provides the MongoDB Atlas which allow the user to access/host the database on the cloud at minimal cost (for our project, we have experimented with the free version).

#### 6.2.1.2 *Backend of the application*

1. Microservices in C#

Microservices were developed in C# in order to provide functionalities to the frontend such as connection to the webserver as well as asynchronous feedback. This will be discussed in detail in later sections.

2. Python

The application's main code was written using python. Python is known for being one of the most dynamic languages used for programming. It is also extensively supported by the community of developers, especially that it allows the user to implement functionalities with the least effort possible. Since the audience/customers of our application are non-programmers audience, python is also suitable since it enhances the modularity of the code as well as its readability. Furthermore, python supports the **pymongo** library as well as the **flask** library which are both needed for the database and webserver.

#### 6.2.1.3 *Frontend of the application*

The system provided need to support multiple platforms. Thus, the frontend of the application was implemented using Unity. Even though Unity was not built for two-dimensional user interface, a lot of effort has been put to create and integrate new modules in the framework to provide components that range from simple checkboxes to more complex graphic visualization tools. This tough choice of technology allowed our application to be transparent and available on multiple platforms (such as Windows and MacBook devices).

#### 6.2.1.4 *Web Server*

The communication between the frontend and the backend has been established using the **Flask** webserver supported by python. Flask is a lightweight web frame that allows the user to refactor the code to run a server-side web API consisting of publicly exposed endpoints and a defined request–response message system where the data is expressed in JSON format. The choice of this technology allowed the frontend to establish Restful API requests (POST, GET) to communicate with both the database and diagnosis systems

### 6.2.2 Storage of the data:

As we have mentioned previously, we have stored the records/data in a MongoDB database. The MongoDB was setup locally as a first stage of the project. We have experimented with an online instance of the database on MongoAtlas (implemented in the demo).

We will be discussing the several collections of data as well as the format (i.e. schema) of each document stored in the MongoDB database. The database's name is ***LRC***. The table below shows the collection (table) names, the purpose of using the collection as well as the attributes of the Json files that will be inserted.

| **Database Collection** | **Usage** | **Attributes** |
|---|---|---|
| accountRequests | Stores the credential requests of the users | ['reason','status','username','password','email','name', 'gender','dateOfBirth',' role','roleDescription','picture'] |
| users | Stores information concerning the users | ['reason', 'status', 'username', 'password', 'email', 'name', 'isAuthenticated', ' dateOfBirth', 'gender', 'picture', 'role', 'roleDescription'] |
| roleChangeRequests | Stores the role change requests of the user | ['username', 'newRole', 'dateOfRequest', 'timeOfRequest’, 'reason', 'status', 'reasonRejected', 'by'] |
| patients | Stores information concerning the patients | ['patientId', 'name', 'gender', 'picture', 'bloodType', 'diseases'] |
| ecgFetched | Stores the ECG records that were captured (yet not associated to any patient) | ['ecgId', 'data', 'fs', 'dateOfAcquisition', 'timeOfAcquisition', 'picture', 'numberOfRRIntervals'] |
| ecgRecords | Stores the ECG records associated to the patients. | ['ecgId', 'data', 'fs', 'dateOfAcquisition', 'timeOfAcquisition', 'picture', 'numberOfRRIntervals', 'bpm', 'patientId', 'user', 'pExtracted', 'qExtracted', 'rExtracted', 'sExtracted', 'tExtracted', 'machineDiagnosisKey','machineDiagnosisProba', 'machineDiagnosisBoolean', 'humanDiagnosis'] |
| history | Stores the user's interactions with the functionalities of the application. | [‘user’, ‘day’, ‘time’,’message] |
| logindates | Stores the login history of the users. | [‘dates’] |
| statistics | Stores the statistics of the system. | [‘number’, ‘title’, ‘image_bytes’] |

*Table 10: Database collections, usages and attributes*

### 6.2.3 Backend of the application and Webserver

#### 6.2.3.1 *Microservices in C#*

The microservices in C# developed in the system include:

1. Connection to the web server (currently using the localhost endpoint, which will be hosted on any VPS or web hosting service such as MongoDB atlas in the next phase).
2. GET and POST requests to the server.
3. Asynchronous feedback from the server.

In order to achieve an asynchronous feedback from the server using Unity and C#, we had to make use of several Object-Oriented Programming notions as well as some C# - only notations. Particularly, there are multiple features (i.e. filling input fields, drawing a graph and even moving from a panel to another) that need to be performed whenever we receive a feedback from the server. Thus, there is a need to set-up listeners on different events. By using C# "IEnumerator" class, we were able to mitigate the functionality of asynchronous methods. Briefly, we would assign a delegate anonymous class as a parameter with a generic data type of the form of data received. Hence, after a successful response of the server feedback we would call the "IEnumerator" with a custom invokable function as its parameter, in order to reuse the same asynchronous method for several operations. This task allowed an efficient development through the application and only makes use of a Model and a Controller with the NotificationView associated for user feedback.

#### 6.2.3.2 *Python*

As mentioned previously, python was used in order to establish the services needed by the Graphical User Interface. In this section, we will be discussing specific details of the backend implementation. We will first go over the general class diagram, then look into features supported by our application that are worth highlighting.

##### 6.2.3.2.1 Class diagram

The general visualization of the class diagram is showed in Figure 32.

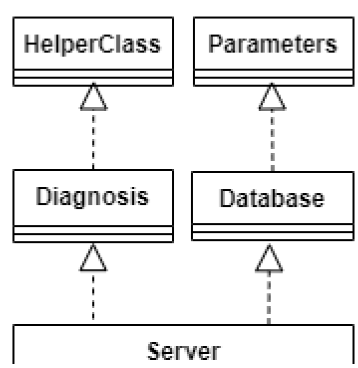


*Figure 33: General Class Diagram of the backend implementation*

We developed 5 python classes:

1- Database class: this class establishes a connection to the MongoDB server and implements all the functionalities needed by the user (i.e. retrieve the list of all patients records, insert a new user, etc.)

2- Diagnosis class: this class implements all the methods needed to retrieve, analyze, annotate and diagnose an ECG.

3- HelperClass class: this class implements static methods used by the diagnosis class (but not used by the Graphical User Interface).

4- Parameters class: his class contains static methods and variables that are used in the Database class. This class was done to centralize the parameters such as the database name, port number, error messages, etc.

5- Server class: this class uses Flask in order to establish API endpoints that provide the functionalities implemented in the Database and Diagnosis classes accessed from the User Interface.

#### 6.2.3.2.2 Credentials states:

As we have mentioned briefly, the User communicates with the Graphical user interface (implemented in Unity) that will call the API services from the Flask Server. There are multiple states of the user's credentials. We could clarify them using the state diagram below:

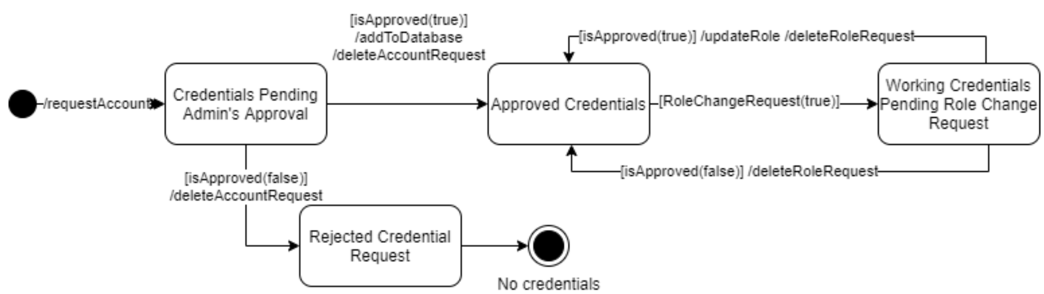



*Figure 34: State diagram showing the states of the credentials/user*

*1- Credentials pending admin's approval:* this is the initial state of any user who enters his/her details and request approval to login to the system (waiting to be approved by an administrator).

*2- Approved Credentials:* once approved, the credentials are added to the *users* database and the account request is deleted. The user is now able to login to the system.

*3- Rejected Credential Request:* once rejected, the credential request is deleted (the user cannot login with the current credentials).

*4- Working Credentials Pending Role Change Request:* the user's credentials are in this state if the user was allowed to login to the system with a certain role and decided to issue a changeRole request. Thus, a role change request is sent to the database and awaits the approval of an administrator.

a) If the role change request is accepted, the user enters the Approved Credentials state with an updated role.

b) If the role change request is rejected, the user enters the Approved Credentials state with the same role (not updated).

#### 6.2.3.3 *History:*

A feature worth highlighting is the ability of the user to view the history of his/her interactions with the app. This was done by adding data to the *history* collection after each interaction between the user and the app. The features that were stored and displayed were:

- Login of the user
- User account's requests, approvals and rejections
- Role change requests, approvals and rejections
- Addition of patients, update and deletion
- Addition of patient Records (i.e. ECGs), update and deletion
- Association of the patient to ECGs
- History of medical practitioner

#### 6.2.3.4 *Fetch and associate ECG to patient:*

The sequence diagram in Figure 34 highlights the process of generating an ECG diagnosis and associating the results to a certain patient. The interactions in this diagram are worth mentioning since all of the systems are involved in the interaction. In the diagram below, we assume that the patient selected is already in the database (just like in our product).

a) First, the user requests the retrieval of the ECG records from the Unity App which will communicate this request to the Server that will relay it to the Database. The Database returns the list of ECG recordings (that are not associated with any patient) to the application where the user can view them.

b) Second, the user selects one ECG record. Upon selecting it, a request for diagnosis is sent to the Server. The Server then asks the Diagnosis system to annotate the ECG signal sent (i.e. specify the P, Q, R, S, T peaks and the baseline). Once the annotations are returned, the Server requests a diagnosis from the Diagnosis system by passing the annotations as well as the ECG values. The Diagnosis system returns the list of ECG anomalies detected as well as a confidence score reflecting the frequency of the presence of each anomaly. Those results are sent to the Server which relay them to the UnityApp.

c) Third, the user has the option of adding this ECG and the diagnosis to a certain patient. To do so, the UnityApp sends a request to the Server which instructs the database to delete the ECG from the first collection (fetchedECG) and add it to the second collection with the patient's information (ecgRecords).

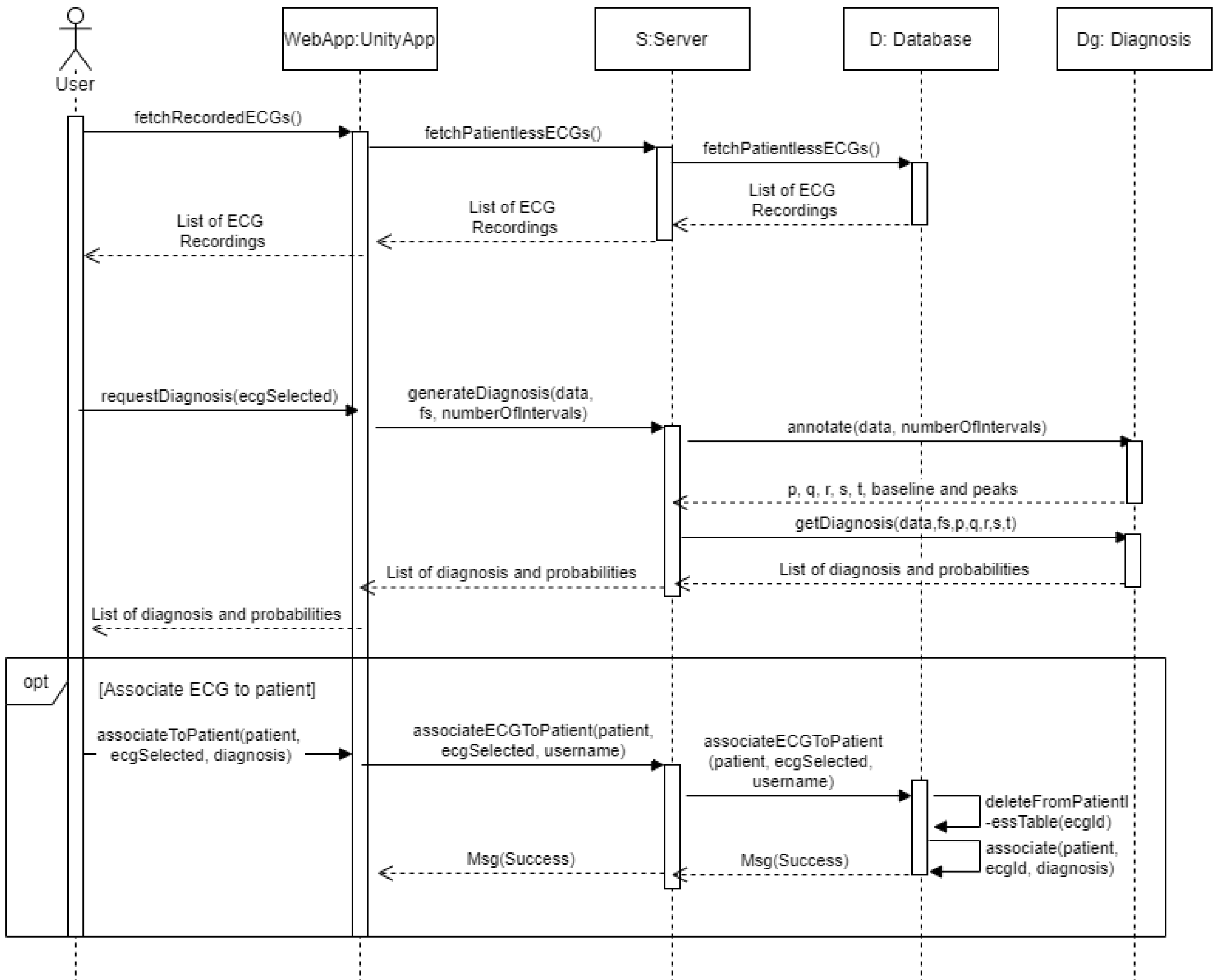



*Figure 35: Sequence diagram for Fetch and associate ECG to patient*

### 6.2.3.5 *ECG Diagnosis*

In this section, we will talk about the process in which the diagnosis of the patient was inferred.

*ECG ingestion*

To develop the diagnosis system, we proceeded by extracting random samples from both MIT-BIH and QT databases. The MIT-BIH Arrhythmia Database contains 48 half-hour excerpts of two-channel ambulatory ECG recordings, obtained from 47 subjects studied by the BIH Laboratory. The recordings were digitized at 360 samples per second per channel with 11-bit resolution over a 10-mV range. The QT database consists of over 100 fifteen-minute two-lead ECG recordings with onset, peak, and end markers for P, QRS and T waves of from 30 to 50 selected beats in each recording. We first downloaded each of those datasets and used the python library ***wfdb*** in order to access the data efficiently.

The data ingested in the *ecgFecthed* database was procured from two sources: the first source is the hardware components used by the ECG operator, while the second source is the random samples extracted from the record of a random patient from the databases that we have. We extracted the RR-Intervals by first smoothing the values using a gaussian wrapper, normalizing the ECG waves then using the pan-Topkins algorithm. After detecting the location of the R peaks, we calculated the mean value of the duration of one RR interval and we used it to retrieve *k* random RR intervals.

<u>*ECG Labelling*</u>

Several approaches were explored in order to detect the P, Q, R, S and T curves of the ECG signals. We have experimented with a Convolutional Neural Network, neural network and even LSTM. Such implementations did not scale well (overfit the training data).

Thus, an alternative unsupervised approach was explored. The pseudocode is showed below:

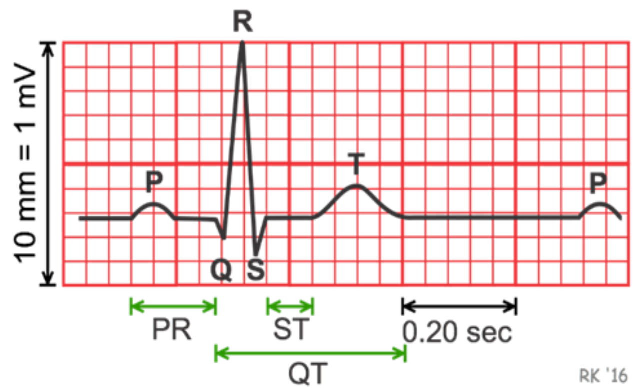


*Figure 36: ECG graph features*

```
Input: Y - Data points of one R-R ECG signal;
Output: L - Set of labels attributed;

Step 1: Cleaning the ECG signal
1.1 Correct the baseline of the signal
1.2 Interpolate the data using a cubic spline to produce reshapedY of M samples

Step 2: Detecting the QRS complex interval
2.1 Compute the derivative diffY of the reshaped signal
2.2 Retrieve the index of the top n peaks in diffY
2.3 Cluster the peaks using KMeans to filter out outliers
2.4 Identify the QRS complex as the cluster of points having the maximum peak

Step 3: Identifying the locations of the Q, R, and S labels
3.1 Identify the number of peaks N in the QRS complex by counting the number of intersections of a threshold with
    diffY
3.2 Identify the baseline level L of reshapedY
3.3 Attribute the labels based to the top N most significant peaks in the QRS interval based on their distance with
    respect to the computed baseline

Step 4: Identifying the locations of P and T labels
4.1 Identify the potential locations of P and T by computing the peaks occurring before and after the QRS complex
4.2. Attribute the labels to the peaks with maximum distance to the threshold
```

This pseudocode is elaborated in detail with a running example in Appendix 1.

*ECG Diagnosis*

After extensive research and after meeting with a medical expert, we were able to use the detected P, Q, R, S and T waves in order to detect abnormalities in the heart. The tables below summarize the rules and the anomalies as given by the medical practitioner. We should note that the diseases can be present simultaneously and are usually independent of one another. Due to the co-occurrence of those diseases, and the lack of labelled datasets covering non-arrhythmic diseases in the literature, we opted out for checking the conditions in the tables below to generate the diagnosis from the labels:

| Beats per minute | Anomaly |
|---|---|
| < 60 | Bradycardia |
| 60-100 | Normal |
| >100 | Tachycardia |

Table 11: Abnormal Detection using BPM

| P-wave | Anomaly |
|---|---|
| Inverted | Sinoatrial block: the electrical impulse is delayed or blocked on the way to the atria |
| | Dextrocardia: The heart is in an abnormal location within the chest |
| Greater than 2.5mm | Right atrial hypertrophy (or enlargement) |
| Invisible | Sinoatrial block: the electrical impulse is delayed or blocked on the way to the atria |
| Longer than 200ms | AV block between atria and ventricles |

Table 12: Abnormal ECG detection using P-wave

| QRS interval value | Anomaly |
|---|---|
| < 120 ms | Normal narrow QRS |
| > 120 ms | • VFrib<br>• Vtach<br>• Supraventricular tachycardia<br>• Ventricular pacing with pacemaker |

Table 13: Abnormal ECG detection using QRS interval

| QT interval value | Anomaly |
|---|---|
| >440ms | Long QT |
| < 440ms | Normal |

Table 14: Abnormal ECG detection using QT interval

| T wave | Anomaly |
|---|---|
| Negative | Abnormal (could lead to a heart attack) |

Table 15: Abnormal ECG detection using T-wave

Based on the labels inferred, we are able to generate a possible diagnosis for the patient. The diagnosis is carried per RR-interval on a voting basis (i.e. the more an anomaly occurs in RR intervals, the higher its occurrence probability). For instance, if three RR-intervals are detected, the labels and diagnosis of each RR-interval are generated. Then, the overall diagnosis is the aggregation of each one (i.e. an anomaly detected in one RR interval has a probability of presence of $\frac{1}{3}$, etc.).

#### 6.2.3.6 *Statistics*

In a world in which we are constantly surrounded by data, figures, and statistics, it is imperative to understand and to be able to use quantitative methods. The statistics section in our app exposes information about the stored and diagnosed data. It consists of six descriptive and inferential statistical charts which could help the Lebanese Red Cross in analyzing data, making decisions and planning. Five out of the six statistics describe the distribution of three main collections in the MongoDb database: *patients*, *ECG records* and *users*. For the *patients*, there is the gender distribution and the risk factors distribution; for the *ECG records*, there is the ECG abnormalities distribution and for the *users*, there is the user roles distribution and the user activity. The last statistic visualizes the accuracy of the software diagnosis by showing the number of correctly and incorrectly labeled diagnosis.

#### 6.2.3.7 *Statistics about the patients:*

The following two charts are built based on the data of all the patients stored in the database. The gender and the risk factors are considered two features which are highly correlated with heart diseases. Therefore, a statistical visualization of these factors and behaviors that pose a significant risk to the cardiac health is beneficial for decision making.

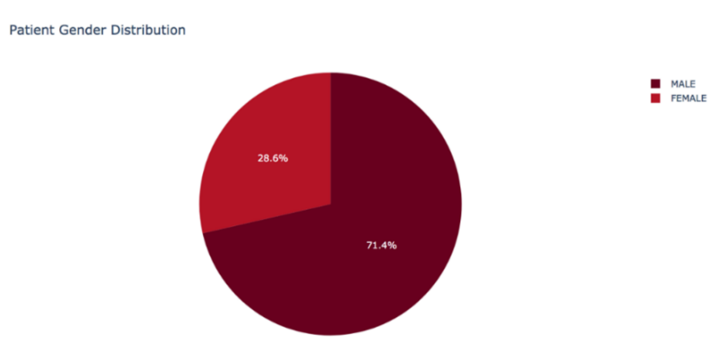



*Figure 38: Patient gender distribution statistics*

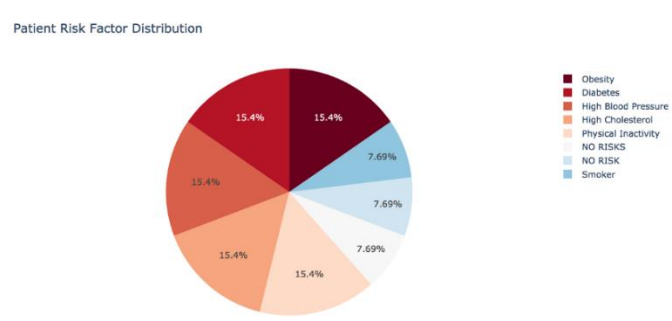



*Figure 37: Patient risk factor distribution statistics*

The following chart is based on the ECG records of all the patients in the database. It examines the different ECG abnormalities that were diagnosed correctly by the software. This statistic graph could be very valuable for the LRC since it shows the distribution of the different ECG abnormalities among the Lebanese population.

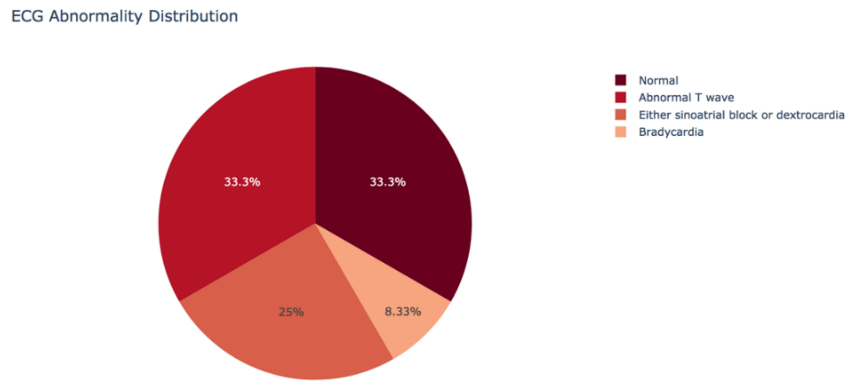
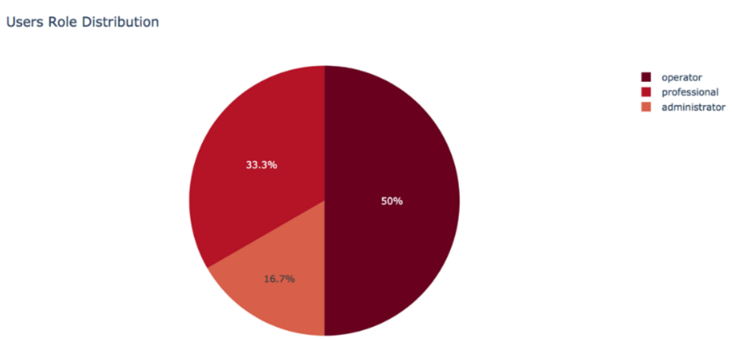



*Figure 39: ECG abnormality distribution statistics*

#### 6.2.3.8 *Statistics about the users*

The first chart is based on the role of the users stored on the database; it could be useful when planning the divisions of the LRC team between administrators, medical practitioners and ECG operators. As for the second chart, it displays the activity monitoring of the users by showing the number of daily active users which could be a measure of the growth rate of the product or of the behavior of the users.



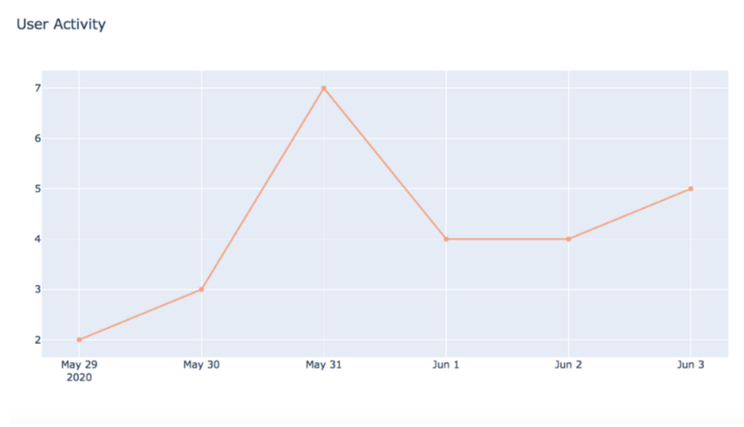




*Figure 40: User role distribution statistics statistics*

*Figure 41: User activity*

### 6.2.4 Statistics about the accuracy of the machine diagnosis

The last statistic evaluates the accuracy of the diagnoses predicted by the software. A predicted label is considered correct if it was approved by the medical practitioner, otherwise, it is incorrect. This statistical chart is effective in assessing the quality of the diagnosis made by our software.

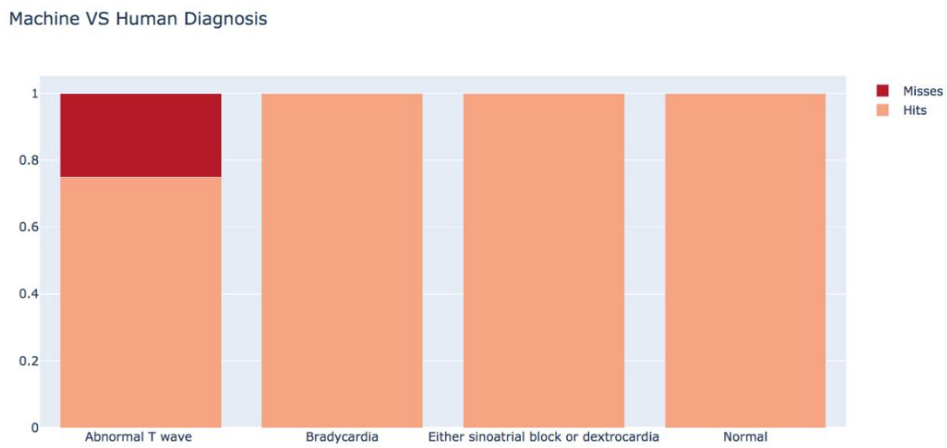



*Figure 42: Accuracy of the diagnosis's statistics*

### 6.2.5 Frontend of the application

The application structure follows a Model View Controller (MVC) architecture which arranges the code into three logical and interacting components: model component that manages the backend functionalities, view component that manages the front-end (GUI) and controller component that manages user interaction and passes these interactions to the Model and View. In addition, the Singleton pattern was used extensively since we need to keep a static reference of our patterns. The main difference between a static class and a singleton pattern is that the singleton reference of the class allows the class to have non static method while referencing at in a static environment. This will assure to only have a single copy of our class (Ex. One reference for requests to the server, A single reference for every Model View and Controller). Nevertheless, this pattern was beneficial since we know that we won't need to keep a reference for every object in an "Application Manager". Therefore, it's a great combination with the Model View Controller architecture in order to keep code concise and efficient.

In order to make the application easy to use for operators, medical practitioners and system administrators, we tried to minimize the number of clicks in order to perform a certain task. In addition, the design of the application follows a functional hierarchy, i.e. each layer or view englobes all related functionalities associated with its label in order to maximize the user experience. Nevertheless, we'll subdivide the UI, microservices and the Unity Application into several abstract layers. The microservices include:

- Connection to the web server (currently using the localhost endpoint, which can be practically hosted on any VPS or web hosting service).
- GET and POST requests to the server.
- Asynchronous feedback from the server.

In order to achieve an asynchronous feedback from the server using Unity and C#, we had to make use of several Object-Oriented Programming notions as well as some C# - only notations. Particularly, since we need to fill input fields or draw a graph or even move from a panel to another (Ex. Login Page to Main Page), whenever we receive a feedback from the server, then we need to set-up listeners on different events. Therefore, by using C# “IEnumerator” class we were able to mitigate the functionality of asynchronous methods. Briefly, we would assign a delegate anonymous class as a parameter with a generic data type of the form of data received. Hence, after a successful response of the server feedback we would call the “IEnumerator” with a custom invokable function as its parameter, in order to reuse the same asynchronous method for several operations. This task allowed an efficient development throught the application and only makes use of a Model and a Controller with the NotificationView associated for user feedback.

#### 6.2.5.1 *Graphical User Interface*

Since we’re following the MVC architecture, then the UI can change without any modifications in the code logic and vice versa. However, we’re using Unity Gaming Engine which is not designed for 2D Animations. Therefore, all effects from animations, panel transition and screen constraints with different scales of the view were created from scratch. The plotting was also done from scratch. This section contains several realms which we’ll breakdown into different components.

##### 6.2.5.1.1 Infrastructure

The components of our system adopt the MVC model. For example, the requests to server uses the model and the controller. In addition, the visualizations of the data from the database only requires a view and the model. Moreover, the entire code (and not only the requests to the server) uses asynchronous communication to maximize efficiency.

###### 6.2.5.1.2 Animations

To make the application more appealing for the users, we made use of animations. Adding them is very crucial since facilitating the flow would make them comfortable using the features (thus minimizing the training time required to master the application). Animations were mainly designed using a Finite State Machine pattern (FSM). Each UI component has a single state at any given point in time and the transition between states depends on the interaction of the user.

###### 6.2.5.1.3 User Functionalities and Panel Transition

This section discusses the transitions between the states of the application. In order to know which functionalities to activate or deactivate, we need to know the role of the user using the application. For this reason, we used the C# "Enum" data structure (created when launching the application) that builds several states storing binary values such as isAdmin, isOperator or isProfessional. Depending on the state, several functions are activated. As for panel transition, every time we go to a new panel, we keep track of the states of the previous panels using a stack. This panel can be accessed from multiple views. Panel could be used as a singleton that hold the previous state. For example, PatientECG could derive from associateECGRecord or from viewPatient (based on the state it arrived from, we will know which functionalities to activate).

###### 6.2.5.1.4 Plotting

The plotting has its Model (retrieving data points and the features from the database), View and Controller. The panel represents a raw image. We insert a raw image in the background, and we put an image on top of it in the foreground to plot the graph. Normalization of the values of x and y was required in order to fit them into a view of constant size regardless of the number of data points in the ECG signals. In order to plot the data points on the graph, each and every point is connected through a line using a raw image (transparent). In order to add the feature with the data points, event listeners are added at run time linking between the values of PQRST in the database and the actual value of the point on the plot.

# 7 Testing

## 7.1 Testing the User Interface and Database systems

In this section, we are going to test the User interface. However, we should note that the functionalities tested are implemented using Microservices in C# and python. Since the functionalities are heavily tied to the UI, testing was conducted to test both the Interface and the functionality behind it (communication, database related features, etc.).

Proceeding with all test cases below was successful with ***no error***.

| |
|---|
| Input(s):<br>The required fields are: u*sername* and *password* (that need to be in the database in order to login). |
| Test(s):<br>1- Input correct *username* and *password*<br>2- Input a wrong *username* and a correct *password*<br>3- Input a correct *username* and a wrong *password*<br>4- Input wrong *username* and *password*<br>5- Input *username* and *password* that were not checked yet by the Administrator (after signing up)<br>6- Input *username* and *password* that were rejected by the Administrator |
| Output(s):<br>Four possible notification messages: "Invalid username and/or password", "Credential Request was not checked yet", "Request was granted! Successful login" and "Credential request was not approved. Reason: not specified". If the login was successful, the user is redirected to the main page comprising the functionalities. |

*Table 16: Signing in test card*

| |
|---|
| Input(s):<br>The required fields are: *full name, username, password, email, gender, role, dateofBirth* and the optional field is: *extra information* |
| Test(s):<br>1- Input all the fields correctly<br>2- Input all the fields except the optional field<br>3- Leave all the fields empty<br>4- Input all the required fields except one (test it for each one of the required field)<br>5- Input a numerical *full name*<br>6- Input a *full name* with less than 3 characters<br>7- Input a *username* that belongs to a user or that was used to request an account<br>8- Input a *username* with less than 2 characters<br>9- Input a *password* having less than 3 characters<br>10- Input an *email* having a wrong format |

| |
|---|
| 11- Input an *email* that was used by another user<br>12- Input a *date of birth* with a format that is different from DD/MM/YYYY |
| Output(s):<br>Two possible notification messages: "Your request has been sent" or "Kindly fix your login details". In both cases, the user is redirected to the login page. |

*Table 17: Signing up in test cards*

| |
|---|
| Input(s):<br>The user's fields that could be updated are: *role*, *extra information*, *password*, *gender* and *date of birth.* |
| Test(s):<br>1- Try to update *full name* or *username* (should be locked)<br>2- Update each one of the allowed fields (one by one and all together)<br>3- Update the *password* with another one having less than 3 characters<br>4- Update the *date of birth* while changing the format DD/MM/YYYY<br>5- Update the *role*<br>6- Update the *role* multiple times successively |
| Output(s):<br>Two possible notification messages: "Profile update success" and "Change role success". If user changes the role, it won't be changed until the administrator accepts the request If the user is done with updating the profile, she/he is redirected to the main page. |

*Table 18: Updating user's information test cards*

| |
|---|
| Input(s):<br>No input (Administrator can access all the user requests ) |
| Test(s):<br>1- Accept/reject a request and check if it still present in the list of requests<br>2- Approve an authentication request and try to login using the accepted credentials<br>3- Reject an authentication request and try to login using the rejected credentials<br>4- Approve a user role change request and check if it was updated for the user<br>5- Reject a user role change request and check if it was updated for the user |
| Output(s):<br>Two possible notification messages: "Rejected success", "Approved success". If the user is done with accepting/rejecting request, she/he can return to the main page. |

*Table 19: Administrator's responding to requests test card*

| |
|---|
| Input(s):<br>The user's optional fields are: *full name, diseases, blood type* and *gender*. |
| Test(s):<br>1- Leave all the fields empty<br>2- Fill some fields<br>3- Fill all fields |

| |
|---|
| Output(s):<br>Notification message: "Add patient success". In all cases, the patient is stored in the database (without constraints) and it can be visualized by any user by pressing to the "Patients" button. |

*Table 20: Adding patient test card*

| |
|---|
| Input(s):<br>The restricted inputs (for Professional users) are: *human diagnosis*, *PQRST* values (relocate from the graph) and the other optional input is a patient to be associated to the ECG record. |
| Test(s):<br>1- Input a *human diagnosis* (try if it is possible when the user is not a professional)<br>2- Update *human diagnosis* (try if it is possible when the user is not a professional)<br>3- Update *PQRST* values by relocating the points on the graph (try if it is possible when the user is not a professional)<br>4- Associate ECG record to a patient in the database<br>5- Associate multiple ECG records to the same patient<br>6- Try to associate an ECG record that is already associated to a patient (blocked) |
| Output(s):<br>Notification message: "Patient Selected *nameOfPatient*" when the ECG record is associated to a patient. Notification message: "Save success" when you change any other field. The user is redirected to the previous page. |

*Table 21: Manipulating ECG records test card*

| |
|---|
| Input(s):<br>The required fields are: *full name, diseases, blood type* and *gender* (for ***patient*** search); *full name, username, email, date of birth, gender* and *role* (for ***user*** search); *date* (for ***patientless ECG*** search), *date, machine diagnosis and human diagnosis* (for ***ECG record*** search); *full name , username, email, date of birth, gender* and *role* (for ***user account request*** search) and username and role (for ***user role change request*** search). |
| Test(s):<br>1- Input all search fields together<br>2- Leave all the search fields empty<br>3- Write only the beginning of a search text field<br>4- Search in a field for a specific output then search again in the same field for another output |
| Output(s):<br>If the user presses on the search button, she/he is redirected to the page before the search page that displays the results of the search: list view of patients, users, patientless ECG, ECG records, user account requests and user role change request. |

*Table 22: Searching test cards*

| |
| --- |
| Input(s):<br>No required fields. |
| Test(s):<br>1- Add a new patient (in order to check its effect on the Patient Gender Distribution and Patient Risk Factor Distribution charts)<br>2- Diagnose a new ECG record and approve the ECG abnormalities (in order to check its effect on the ECG Abnormality Distribution and Machine Vs Human Diagnosis charts)<br>3- Add a user or change the role of user (in order to check its effect on the User Role Distribution chart)<br>4- Sign out then Sign in (in order to check the effect on the User Activity chart) |
| Output(s):<br>The user visualizes six charts: Patient Gender Distribution, Patient Risk Factor Distribution, ECG Abnormality Distribution, Machine VS Human Diagnosis, User Role Distribution and User Activity. |

*Table 23: Statistics test cards*

| |
| --- |
| Input(s):<br>No required fields. |
| Test(s):<br>1- Sign in using a specific user and perform any functionality in the application<br>2- Sign in using a specific user, perform any functionality, sign out, sign in using another user and then perform any functionality<br>3- Sign in using a specific user, perform any functionality, sign out, sign in using same user and then perform any functionality |
| Output(s):<br>If the user presses on the history button, she/he visualizes all the activities done by any user. |

*Table 24: History test cards*

## 7.2 Testing the ECG Diagnosis system

To test the accuracy of the labelling, we proceed by randomly extracting RR-intervals from the QT database. We extracted 40 signals with a random number of RR intervals, and we annotated them using the algorithm proposed. After doing that, we proceeded by sending the original ECG signals with their original labels to Miss Grace Akiki, and asked her to fill an excel sheet that confirms the number of P, Q, R, S and T peaks in the curves by specifying the true positives, false positives and false negatives. An excerpt from this excel sheet is showed here:

| *P* | | | *Q* | | | *R* | | | *S* | | | *T* | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| *TP* | *FP* | *FN* | *TP* | *FP* | *FN* | *TP* | *FP* | *FN* | *TP* | *FP* | *FN* | *TP* | *FP* | *FN* |
| 2 | 0 | 0 | 0 | 0 | 0 | 2 | 0 | 0 | 1 | 0 | 1 | 2 | 0 | 0 |
| 2 | 0 | 0 | 0 | 0 | 0 | 2 | 0 | 0 | 1 | 0 | 1 | 2 | 0 | 0 |
| 2 | 0 | 0 | 0 | 0 | 0 | 2 | 0 | 0 | 2 | 0 | 0 | 2 | 0 | 0 |
| 2 | 0 | 0 | 0 | 0 | 0 | 2 | 0 | 0 | 2 | 0 | 0 | 2 | 0 | 0 |
| 2 | 0 | 0 | 2 | 0 | 0 | 0 | 0 | 0 | 2 | 0 | 0 | 2 | 0 | 0 |
| 2 | 0 | 0 | 0 | 0 | 0 | 2 | 0 | 0 | 2 | 0 | 0 | 2 | 0 | 0 |
| 2 | 0 | 0 | 0 | 0 | 0 | 2 | 0 | 0 | 2 | 0 | 0 | 2 | 0 | 0 |
| 2 | 0 | 0 | 0 | 0 | 0 | 2 | 0 | 0 | 2 | 0 | 0 | 2 | 0 | 0 |

*Table 25: Excerpt from the predicted and expected P, Q, R, S and T labels*

Some of the sample graphs provided are also showed below:

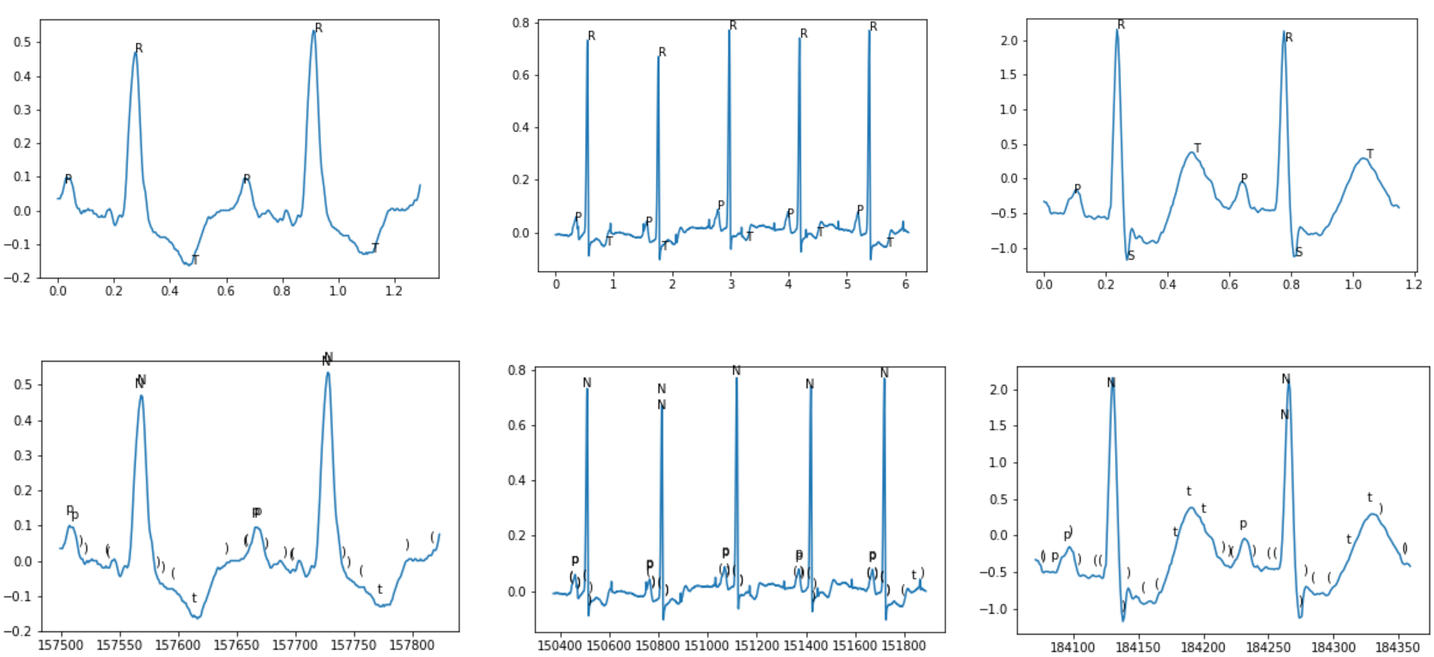


*Figure 43: Some of the ECGs assessed by the medical expert (the upper row shows the predicted labels while the lower row shows the labels provided in the QT dataset)*

We should note that the professional was provided with labelled plots from the QT dataset (refer to Figure 43 to minimize the labelling error while also taking into account the baseline provided by the dataset.  From the values provided by the medical expert, we calculated the accuracy by computing the precision, recall and F-values. The results are showed below:

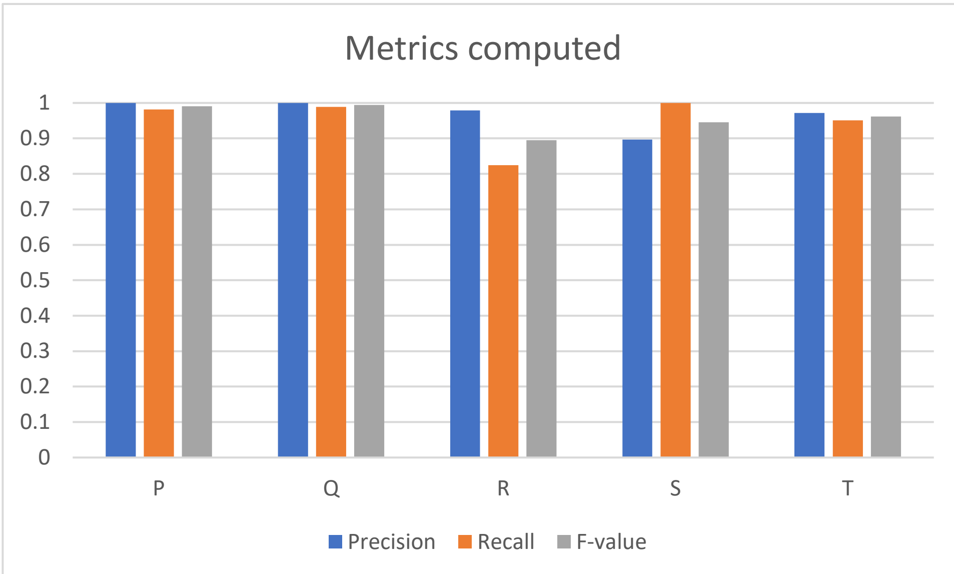



*Figure 44: Bar chart showing the precision, recall and f-values of the P, Q, R, S and T labelling*

- Figure 44 shows that the proposed approach performs very well on the QT dataset. The recall of the R peaks is less than the other peaks, which means that the algorithm was not detecting all the R peaks present in the dataset.
- With the random selection of the data and the different shapes of the P, Q, R, S, T waves, we should mention that the number of predictions is only an indicative of the performance of the algorithm. Further testing should be conducted on a larger set of data (the study was restricted to 40 samples due to the unavailability of the professional within the time constraints of the project).

## 7.3 Testing the ECG detection system

In order to test the ECG detection system, we proceeded by capturing the data and viewing it before it enters the database. We should note that this system requires even more testing. Due to the limited number of disposable patches (originally 10, now 3), we had few occasions to test the whole system. Results shown here are used to establish an initial confidence level in the system.

### 7.3.1 Testing the filtering

To test whether the filters chosen work correctly, we inspected the acquired signal before and after filtering. Based on our experiments, the filters in a non-noisy environment (i.e. at home) could be more relaxed compared to filters in a noisy environment (i.e. in the labs, surrounded with electric equipment). Thus, the filters in the lab had to be narrower to accommodate for the environment. Choosing the software filters was indeed a good choice since opting out for RC hardware filters would have been proven to be inefficient.

We could view sample signals with their time domain representation and the frequency domain representation before and after filtering.

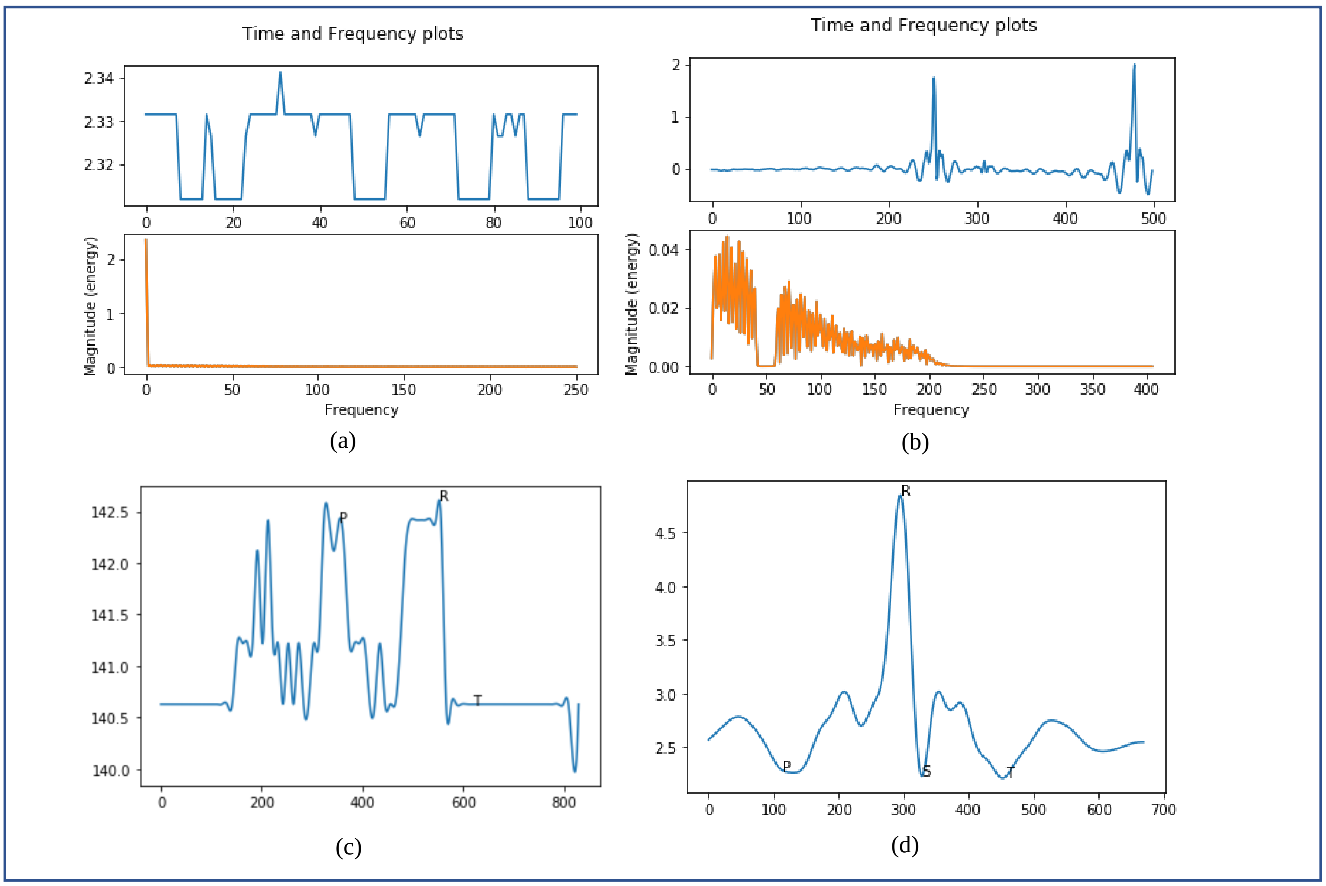



*Figure 45: a) Time and frequency spectrums of the signal before filtering, b) Time and frequency spectrums of the signal after filtering, c) Sample of the signal in the time domain before filtering, d)Sample of the signal in the time domain after filtering*

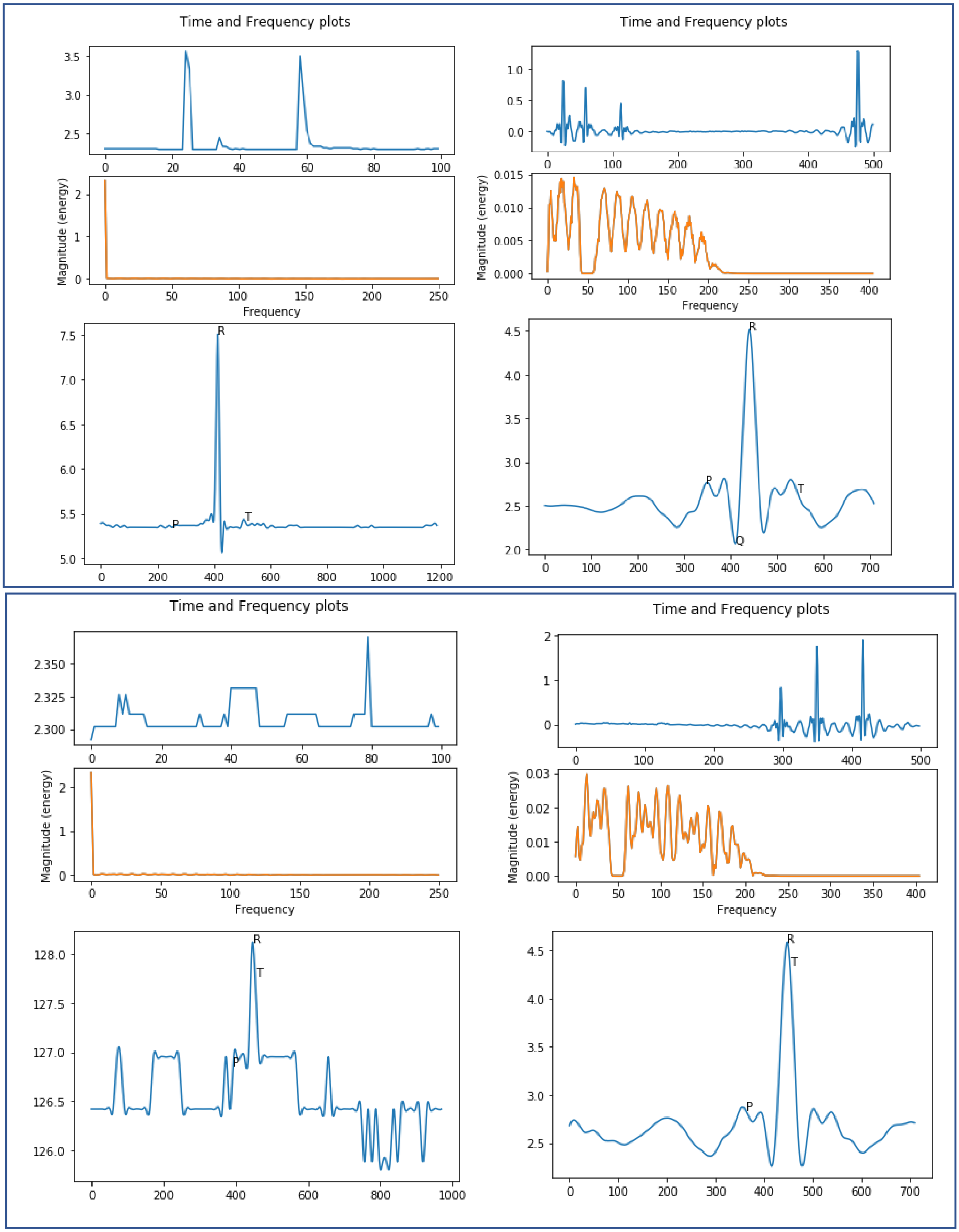



*Figure 46: Time and frequency plots before and after filtering (Same as figure 44)*

Since these results were taken from home in a non-noisy environment, the filter used were a bandpass filter from 1Hz to 200Hz and a band stop filter from 50 to 60Hz. The frequency diagrams clearly show the effect of the filters on the frequency domain, as well as their effect on the resulting wave in the time domain. From the results, we could confirm that the Bandpass and Band stop filters are working properly.

### 7.3.2 Testing the acquisition

To test whether the hardware performs as required, we proceeded by capturing the ECG signal when the electrodes are connected to the body then proceeded by capturing the signal when the electrodes are not connected (i.e. catching noise from the room, with no patient connected to it). We were able to view an ECG signal only in the case where the electrodes were connected to the patient. This is shown below:

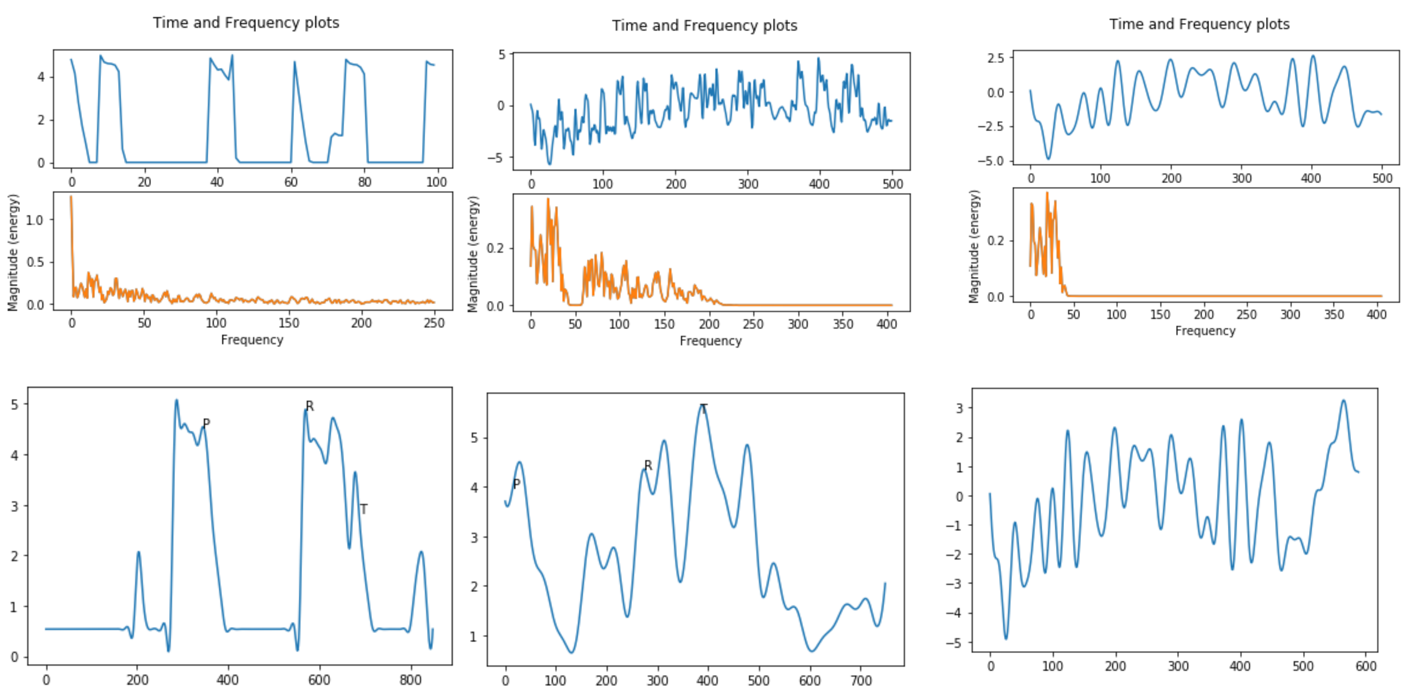



*Figure 47: ECG signals outputted when the electrodes are not connected from the patient: the first column represents the unfiltered signal, the second column represents the same signal after a bandpass filter from 1-200Hz, the third column represents the same signal after a bandpass filter from 1-40Hz*

Looking at the noisy signals still being noisy after narrowing the bandpass filter, we can clearly observe that the system here did not detect any ECG. Thus, the signal detected earlier when the electrodes were connected to the patient’s body is indeed an ECG signal, and we can assume that the data acquisition system works as expected.

## 7.4 Performance

- The ECG management system's performance has been tested using Unity. The graphs below show the time occupied by each of the agents in the system (it is seen to be very small).

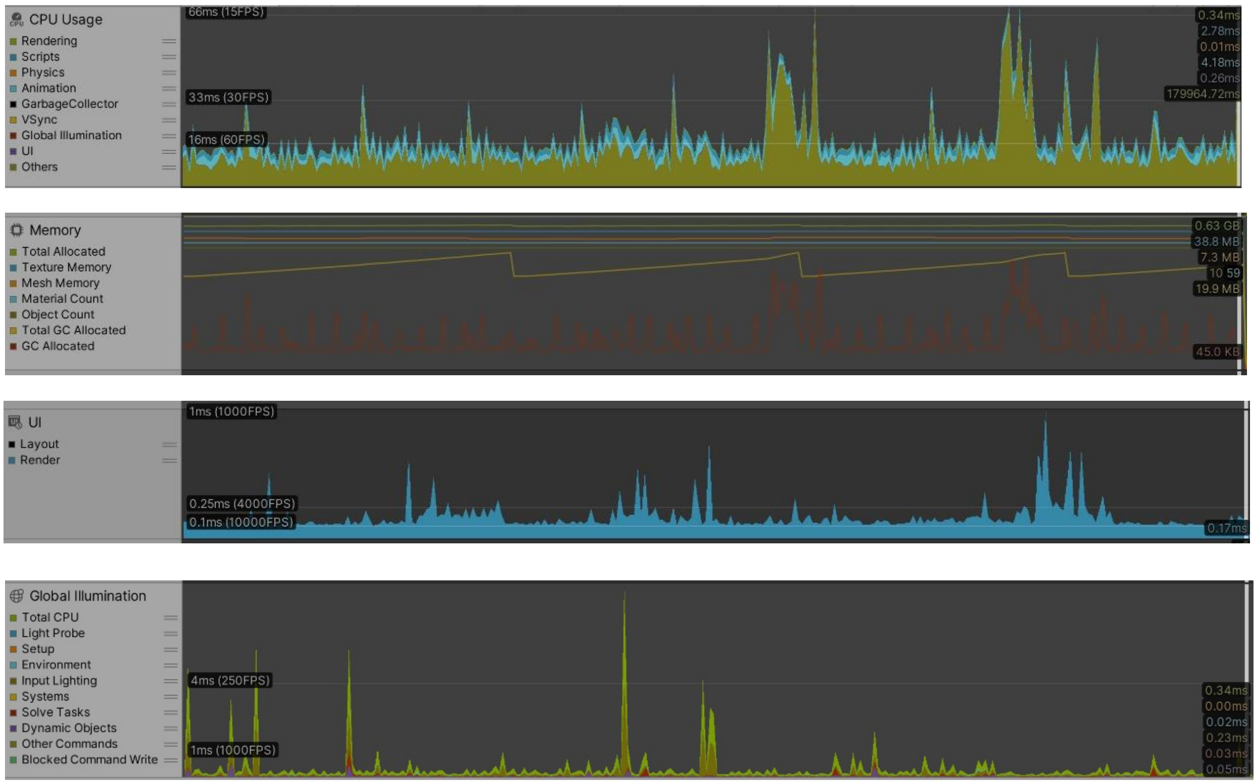



*Figure 48: Unity System Performance*

- The average time to acquire the ECG data points was measured to be approximately 35 seconds (averaged over 7 trials). This is reasonable and acceptable, given that the current device is a prototype.

# 8 Conclusion

According to the World Health Organization, cardiovascular diseases are the number 1 cause of death globally [16]. They are among of the most dangerous illnesses(i.e. stroke, trachea, bronchus, etc.) because they might cause unpredicted death if not monitored correctly.

In this context, ECGs are extensively used in emergency operations to visualize the patient's electrical heart activity. Currently, the Lebanese emergency medical service (Lebanese Red Cross) is facing a communication challenge between the first responders and medical specialists. Our proposal titled *Check My Heart* is an ambitious project that aims at filling some of the gaps that currently exist in this field by providing both real-time diagnostics functionality and live transmission of the first responders' ambulance to the LRC or hospital servers. Our solution consists of implementing an ECG machine with wireless transmission capabilities, electronic storage abilities and heart abnormality predictions. Providing LRC emergency officers with tools that enable them to diagnose heart problems faster and more effectively, with the help of analysis and the live assistance of medical officers connected to the system, could help them save more lives.

As future directions, we are going to work on improving several aspects of our project. First, we will be improving the hardware design. With more resources and more time, we will implement a 3-lead ECG to perform field testing. Furthermore, we will be performing a large-scale testing of both labelling and diagnosis approaches by using a larger and more diverse dataset. Then, we will be meeting with the Lebanese Red Cross to deliver our solution and adjust it based on their needs.

# 9 Appendix 1

## 9.1 Running example for the labelling approach

Let’s elaborate on the pseudocode using a running example. For this example, we will be setting the parameters as set in the product: M=50, n=4 , k=5, and N=4.

**Step 0:** The input signal is showed in Figure 49.

**Step 1:** The signal is processed to remove the false peaks.

Step 1.1 The baseline of the signal is adjusted to make sure that the overall signal is horizontal. This step is necessary to be able to compare the peaks with the baseline at a later stage.

Step 1.2.1 The cubic spline interpolation is a special case for interpolation that is used to provide an interpolating polynomial that is smoother than the original. Furthermore, this interpolation is known for having smaller error when compared to other interpolating polynomials such as Lagrange polynomial and Newton polynomial. This method is used to remove the ripples in the signal by calculating the function that approximates the signal using the interpolation. To reduce the size of the datapoints to M=50, we generate 50 values between the original boundaries of the signal and compute the outputs using the interpolation.

The output of the steps is showed below:

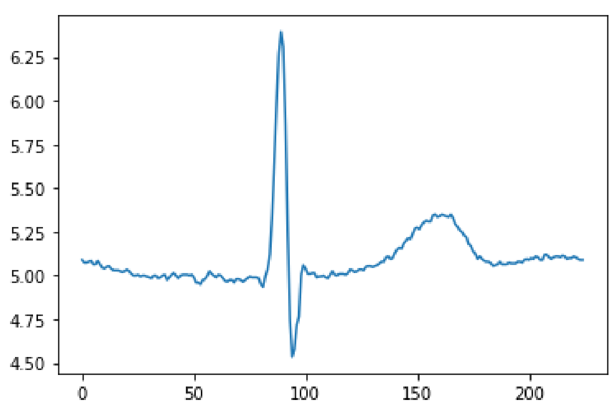


*Figure 49: Input ECG signal*

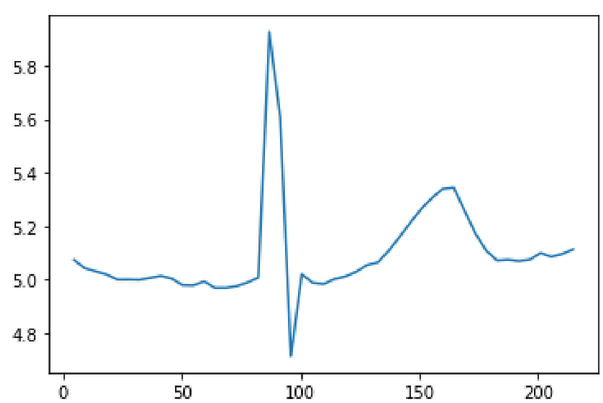


*Figure 50: ECG signal after cubic spline interpolation*

**Step 2:**

Steps 2.1 – 2.2: The derivative of the signal is computed to identify the rate of change of the signal. The QRS complex occurs due to sudden change in the signal, thus could be determined by identifying the points in which the signal's value changes the fastest, i.e. in which derivatives are maximized. Thus, retrieving the location of the top 8 peaks (maximum 4 peaks and minimum 4 peaks) in the derivative is an indicative of the location of the QRS complex. The output of those steps is showed in Figure 51.

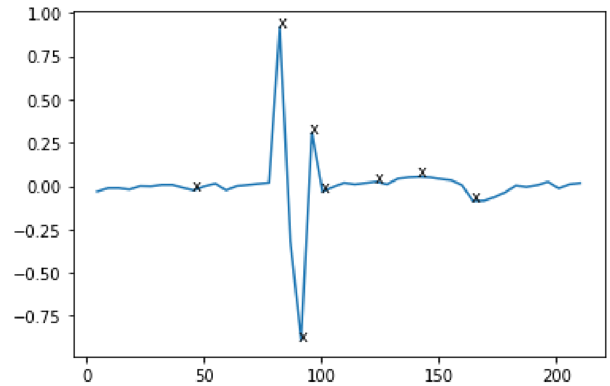


*Figure 51: Derivative of the signal*

Steps 2.3 – 2.4: The detected peaks are now clustered in order to filter out outlier peaks and identify the location of the QRS complex. The clustering was performed using the KMeans algorithm since it is relatively simple to implement, and it produces tighter clusters than the more complex hierarchical clustering. Furthermore, it is well suitable to compare data points in space using the conventional distance measure. We should note that the peaks were clustered using their temporal value (index in the time axis). The peaks in the QRS complex are the ones present in the cluster that contains the maximum value of peak (i.e. the R peak, which is by definition, the longest positive peak). The number of clusters chosen was $k$=***5*** since there are mainly 5 events in the ECG signals (no event, P, QRS, T and no event). The results of clusters is showed below:

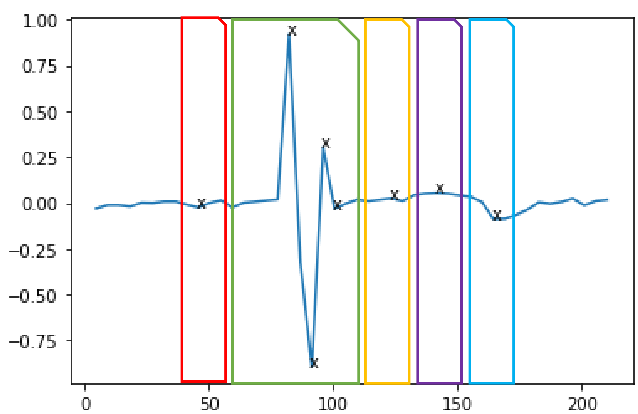


*Figure 53: Peak clustering using K-means*

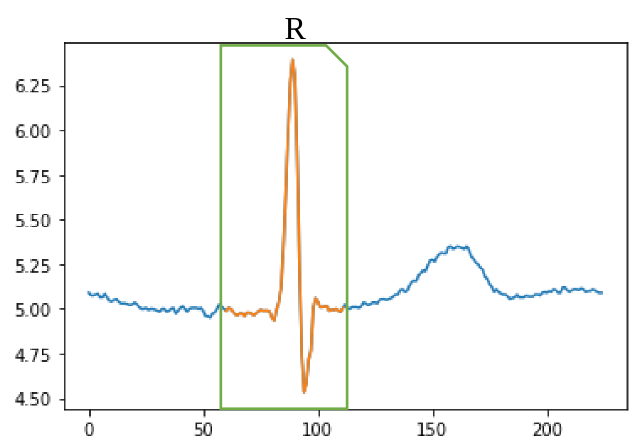


*Figure 52: Detection of the QRS complex based on the cluster containing the maximum values of peaks*

**Step 3:**

Step 3.1: In order to identify the number of peaks inside the QRS complex, we detect the number of intersections of the derivative of the signal with a threshold that we set to be 0.17. The number of intersections determine the number of peaks. In the example below, there are mainly 6 points of intersection, thus 3 variations in the signals and 2 peaks in the original curve. Therefore, we will have either the Q and R peaks or the R and S peaks in the ECG signal.

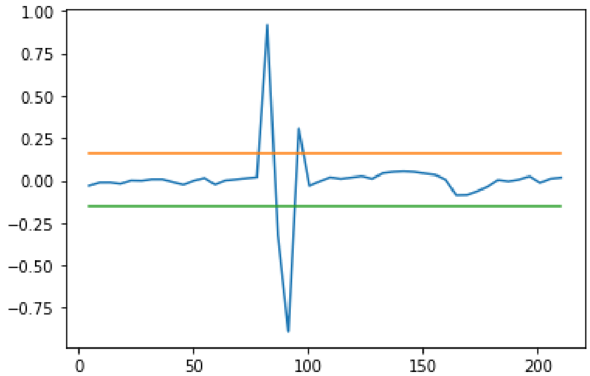



*Figure 54: Detection of the number of peaks in the QRS interval*

Step 3.2: The baseline of the signal should be identify in order to be able to correctly label the peak detected. The baseline is identified by determining the horizontal line that crosses the most values (since the baseline is in the middle, thus should cross the curve multiple times). The result of our example is showed below:

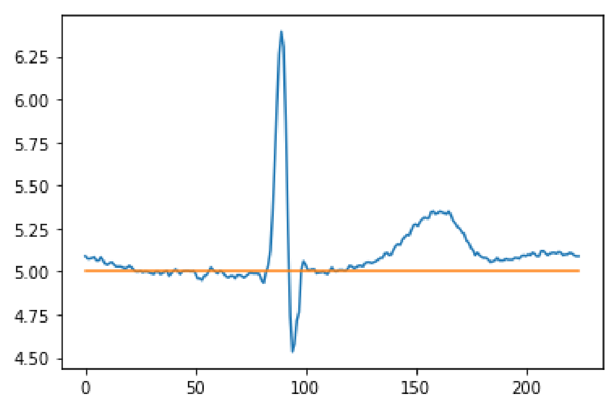



*Figure 55: Detection of the baseline*

<u>Step 3.3:</u> First, we identify the top N peaks inside the QRS complex. (N was determined from Step 3.2). Based on N, we are able to annotate the curve using some predefined rules. The following activity diagram explains the annotation process (for simplicity, some activities were omitted such as the end activity as well as the ones which produce no additional annotation).

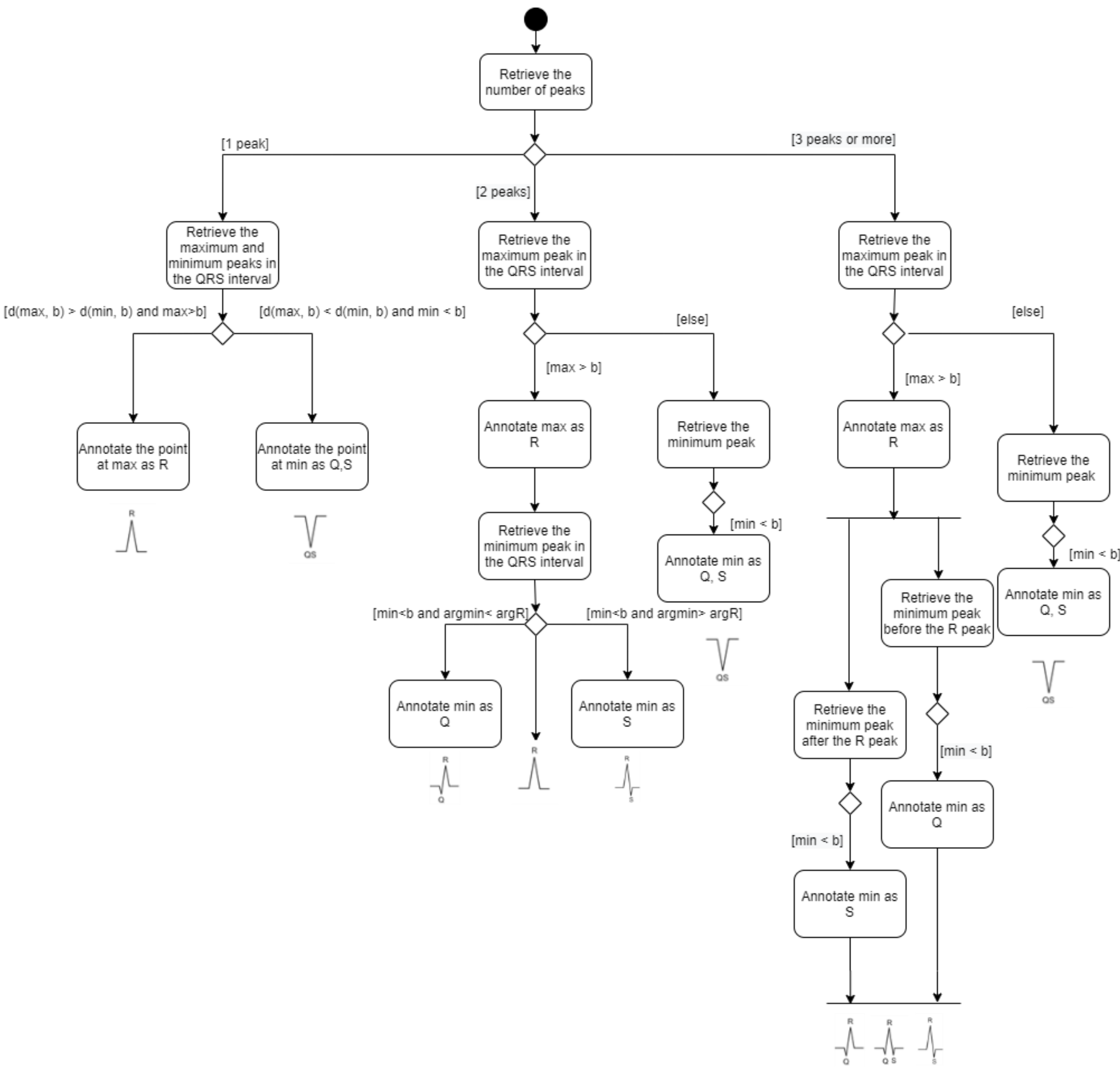



*Figure 56: Activity Diagram showing the annotation process of the PQRST peaks*

Based on the diagram above, we annotate the following curve:

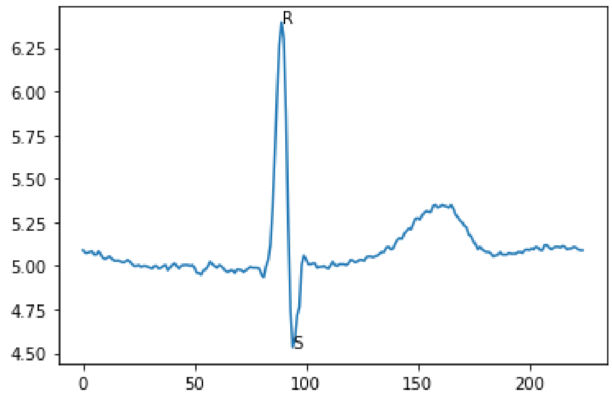



*Figure 57: ECG annotated with the QRS complex (output of the activity diagram above)*

**Step 4:**

Steps 4.1-4.2: To identify the P and T peaks, we identify the peaks before and after the annotated peaks and we attribute the labels to the ones with the maximum distance to the baseline.

Thus, the following labels are produced:

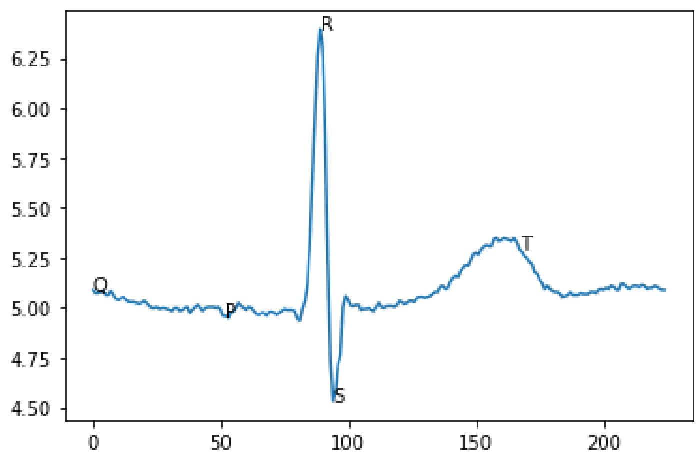



*Figure 58: ECG annotated with the PQRST peaks (output of the activity diagram above)*

We should note that unidentified peaks (such as Q in our example) are labelled at the origin, to allow the medical practitioner to change their value from the User Interface, in case of mislabeling.

# 10 Appendix 2

## 10.1 PIC18F4550

### 10.1.1 Programming the PIC

The PIC18F4550 is an 8-bit microcontroller member of the PIC18F family. In order to program the pic, we used the Pickit3 alongside MPLAB X IDE. Pictures and pinout of the pic and Pickit 3 are showed below:



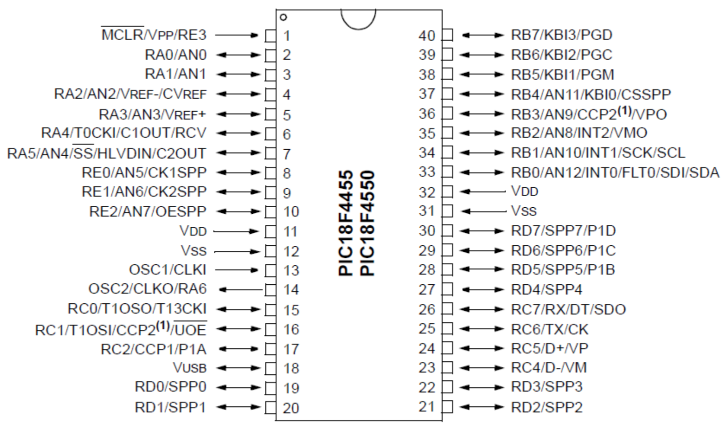



Figure 59: PIC18F4550 configuration

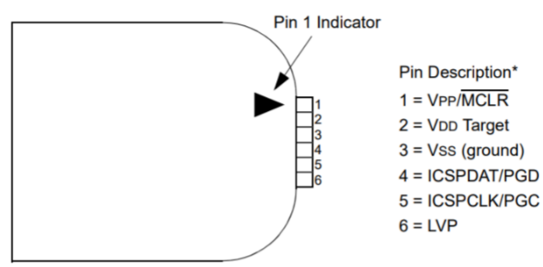



Figure 60: Pickit 3 configuration

The connections needed in order to connect the pic to the Pickkit3 are showed below:

| **Pickit3 pins** | | **Pic18F4550 pins** | | **Function** |
|---|---|---|---|---|
| Vpp/$\overline{MCLR}$ | 1 | MCLR/Vpp/RE3 | 1 | To reset the MCU before programming |
| VDD | 2 | Vdd | 11/ 32 | Target Voltage of PIC (5V) |
| Vss | 3 | Vss | 12 /31 | Ground pin of the whole circuit |
| ICSPDAT/PGD | 4 | RB7/KBI3/PGD | 40 | Program Data (PDG) is connected to the in-Circuit Serial Programming (ICSP) data pin |
| ICSPCLK/PGC | 5 | RB6/KBI2/PGC | 39 | Program Clock (PGC) is connected to In Circuit Serial Programming (ICSP) clock pin |

Table 26: PIC and Pickit3 pin out

### 10.1.2 Setting up the A/D module of the PIC

We configured the A/D module of the PIC in order to be able to do the analog to digital conversion. PIC18F4550 has 13 channels which mean that 13 analog input signals can be converted simultaneously using the module. It uses a successive approximation with a resolution of 10 bits (i.e. the range of bits is 0 to $2^{10} - 1 = 1023$).

In order to setup the conversion, we configure the following registers:

1- ADCON0 (A/D Control Register 0):
   - This register is used to select the input channels.
   - Since we have 13 channels, the bits CH3: CH0 bits (bits 5-2) are used to select the channel multiplexed with the digital I/O pins.
   - The GO bit (bit 1) starts the conversion of the input analog signal when set and is used to check the status of the A/D conversion.

2- ADCON1 (A/D control register 1):
   - The bits PCFG0: PCFG3 (bits 3-0) determine which pins of the controller should be configured as analog inputs.
   - The bits VCGF1: VCGF0 (bits 5-4) are used to select the reference voltage for conversion.

3- ADCON2 (A/D Control Register 2):
   - The bits ADCS2: ADCS0 (bits 2-0) configure the prescaler used for the A/D clock.
   - The bits ACQT2: ACQT0 (bits 5-3) are used to set the acquisition time of the ADC.
   - The ADFM bit is used to select the format by which result should be stored in the ADRESH and ADRESL.

4- ADRESH & ADRESL: Those registers are used to store the result of the conversion (which is 10 bits).

The basic configuration used is showed in the table below:

| Register | Value | Explanation |
|---|---|---|
| TRISA | 0xFF | Set Port A (Analog) as input |
| ADCON1 | 0x0E | Set channels as analog, reference voltages as 0 and 5V |
| ADCON2 | 0x80 | Set sampling rate to fosc/2, acquisition time to 0 |
| ADRESH | 0 | Empty the output registers |
| ADRESL | 0 | |

*Table 27: Configuration of the PIC in order to set up the A/D module*

In order to find the analog voltage measured, we used the following formula:

$$\frac{\text{Maximum value outputed by the A/D}}{\text{Maximum voltage}} = \frac{\text{A/D Reading}}{\text{Analog Voltage Measured}}$$

### 10.1.3 Serial communication

PIC18F4550 has an in-built Universal Synchronous/Asynchronous Receiver/Transmitter (USART module). With the help of USART, we can send / receive data using a transmitter and receive data using a receiver. There is no need for feedback from the Raspberry pi, thus the communication adopted is a one-way asynchronous communication. Since we are using an asynchronous communication, each character (byte) is placed in between a start bit (low voltage 0) and stop bit (high voltage 1). Furthermore, the baud rate (i.e. the rate of the data transfer) needs to be set up. As we have mentioned previously, we will be transmitting a batch of readings from the pic.

In order to setup the communication on the PIC side, we configure the following registers:

1- TXSTA (Transmit Status and Control Register):

- The TXEN transmit enable bit (bit 5) is used to enable the transmission.
- The BRGH High baud rate select bit (bit 2) is used to select whether the speed of transmission is high or low.

2- SPBRG (Serial Port Baud Rate Generator): This register contains a calculated value that is set to generate the desired baud rate. It is calculated based on the BRGH bit:

- If BRGH =0, then $SPBRG = \frac{f_{osc}}{64 \times BaudRate} - 1$

- If BRGH =1, then $SPBRG = \frac{f_{osc}}{16 \times BaudRate} - 1$

3- TXREG (transmitter register): this register is used to store the data to be transmitted. Once the transmission is complete, the **TXIF** (transmit interrupt flag) is set to confirm the transmission and to denote that the receiver is ready for another transmission.

The basic configuration used is showed in the table below:

| Register | Value | Explanation |
|---|---|---|
| TRISC6 | 0 | Set the transmitter Tx pin as output |
| TXSTA | 0x24 | Enable transmitter, enable high speed |
| SPBRG | Calculated using the formulas above | Set the value in the SPBRG to provide a baud rate of 22,000 bps |

*Table 28: Configuration of the PIC for communication with Pi*

### 10.1.4 Code

```
/*
 * File:   newmain.c
 * Author: Joseph Attieh
 *
 * Created on May 30, 2020, 11:48 PM
 */

#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include <xc.h>
#include "header.h"
#define _XTAL_FREQ  8000000
#define F_CPU 8000000/16
#define vref 5.0
#define buffSize 1637


void USART_Init(long);
void USART_TxChar(char);


void USART_Init(long baud_rate)
{
    float temp;
    TRISC6=0;                    /*Make Tx pin as output*/
    TRISC7=1;                    /*Make Rx pin as input*/
    temp=(((float)(F_CPU)/(float)baud_rate)-1);
    SPBRG=(int)temp;
    TXSTA = 0x24;}
/*****************TRANSMIT FUNCTION*****************************************/
void USART_TxChar(char out)
{
        while(TXIF==0);           /*wait for transmit interrupt flag*/
        TXREG=out;               /*transmit data via TXREG register*/
}

```

```
void ADC_Init()
{
   TRISA = 0xFF;            /* Set as input port */
   ADCON1 = 0x0E;           /* Ref vtg is VDD and Configure pin as analog pin */
   ADCON2 = 0x80;           /* Right Justified, 4Tad and Fosc/32. */
   ADRESH=0;                                 /* Flush ADC output Register */
   ADRESL=0;
}
int ADC_Read(int channel)
{
   int digital;
   /* Channel 0 is selected i.e.(CHS3CHS2CHS1CHS0=0000) & ADC is disabled */
   ADCON0 =(ADCON0 & 0b11000011)|((channel<<2) & 0b00111100);
   ADCON0 |= ((1<<ADON)|(1<<GO));                   /*Enable ADC and start conversion*/
   /* Wait for End of conversion i.e. Go/done'=0 conversion completed */
   while(ADCON0bits.GO_nDONE==1);
   digital = (ADRESH*256) | (ADRESL);               /*Combine 8-bit LSB and 2-bit MSB*/
   return(digital);
}
void main(void) {

   TRISB=0;
   ADC_Init();
   OSCCON=0x72;
   int digital, digital1, digital2,digital3, digital4, digital5, digital6, digital7;
   float voltage, voltage1, voltage2,voltage3, voltage4,voltage5, voltage6,voltage7;
    USART_Init(22000);
   __delay_ms(50);
   int i=0;
   static char v[buffSize];

   while(1)
   {
      digital=ADC_Read(0);
      digital1=ADC_Read(1);
      digital2=ADC_Read(2);
      digital3=ADC_Read(3);
      digital4=ADC_Read(4);
      digital5=ADC_Read(5);
      digital6=ADC_Read(6);
      digital7=ADC_Read(7);

      voltage= digital*((float)vref/(float)1023);
      voltage1= digital1*((float)vref/(float)1023);
      voltage2= digital2*((float)vref/(float)1023);
      voltage3= digital3*((float)vref/(float)1023);
      voltage4= digital4*((float)vref/(float)1023);
      voltage5= digital5*((float)vref/(float)1023);
      voltage6= digital6*((float)vref/(float)1023);
      voltage7= digital7*((float)vref/(float)1023);

      char data[7];
      sprintf(data,"%.4f",voltage);
      data[6]=' ';
      for(int j=0; j< sizeof(data)&&i<buffSize; j++){
         v[i] = data[j];
         i+=1;
      }

```

```
        sprintf(data,"%.4f",voltage1);
        data[6]=' ';
        for(int j=0; j< sizeof(data)&&i<buffSize; j++){
            v[i] = data[j];
            i+=1;
        }
        sprintf(data,"%.4f",voltage2);
        data[6]=' ';
        for(int j=0; j< sizeof(data) && i<buffSize; j++){
            v[i] = data[j];
            i+=1;
        }

        sprintf(data,"%.4f",voltage3);
        data[6]=' ';
        for(int j=0; j< sizeof(data)&&i<buffSize; j++){
            v[i] = data[j];
            i+=1;
        }
        sprintf(data,"%.4f",voltage4);
        data[6]=' ';
        for(int j=0; j< sizeof(data)&&i<buffSize; j++){
            v[i] = data[j];
            i+=1;
        }
        sprintf(data,"%.4f",voltage5);
        data[6]=' ';
        for(int j=0; j< sizeof(data) && i<buffSize; j++){
            v[i] = data[j];
            i+=1;
        }
        sprintf(data,"%.4f",voltage6);
        data[6]=' ';
        for(int j=0; j< sizeof(data)&&i<buffSize; j++){
            v[i] = data[j];
            i+=1;
        }
        sprintf(data,"%.4f",voltage7);
        data[6]=' ';
        for(int j=0; j< sizeof(data)&&i<buffSize; j++){
            v[i] = data[j];
            i+=1;
        }
        if (i>=buffSize){

        for(int k=0; k< sizeof(v); k++)
            USART_TxChar(v[k]);

        i=0;

        }
    }
}
```

## 10.2 Raspberry Pi

The Raspberry Pi is a low cost, small single-board computer that plugs into a computer monitor or TV and uses a standard keyboard and mouse. This small board contains a memory, CPU, GPU, Ethernet Port, GPIO Pins, XBee Socket, Power Source Connector, HDMI and others. It can be connected to a Pi Camera and different types of sensors and motors. In order to set it up, you will need to download the Raspbian OS, save it on an SD card then insert the SD card inside the Raspberry Pi. The Raspberry Pi supports coding in python.

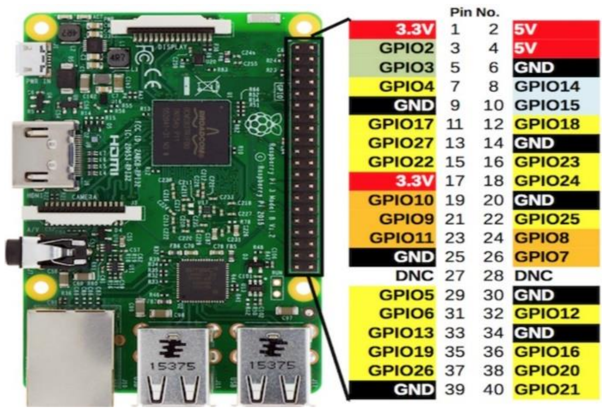



*Figure 61: Raspberry pi pinout*

### 10.2.1 Serial communication

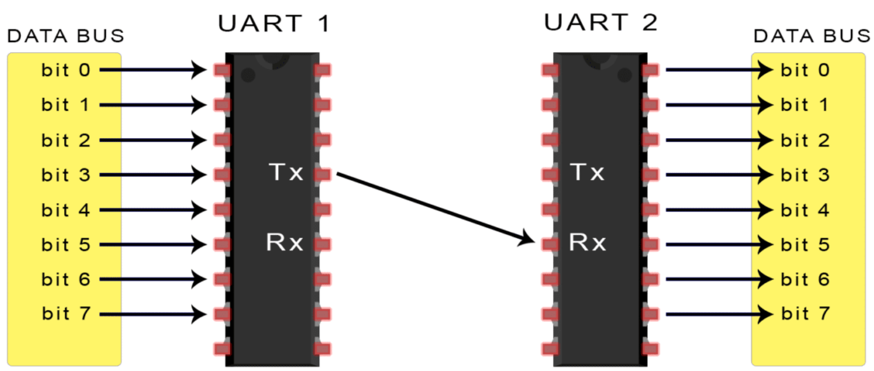



*Figure 62: UART Communication*

In order to accomplish a reliable connection with the PIC microcontroller we made use of the universal Asynchronous Receiver/Transmitter (UART) serial communication protocol. Using the raspberry pinout, we found that GPIO pin number 8 supports UART communication. UART requires two wires to transmit data between devices. With the transmitter (TX) of the PIC microcontroller being plugged into the receiver (RX) of the Raspberry Pi. Hence, by using this type of communication we were able to transmit the ECG readings reliably from the PIC to the raspberry Pi.

### 10.2.2 Parsing the ECG Data

At the PIC microcontroller, the data was collected from all of the Analog to Digital converters, then sent as batches of UART transmits. Hence, in order to ensure a correct parsing of the data, we made sure to read every batch without adding any transmission overhead. We also made sure to filter out any junk characters (if any) from the stream of data.

### 10.2.3 Transmission of the ECG Data

At the end of each successful UART transmission, we saved the ECG data as a JSON file which we transmitted to the server through an intermediary endpoint using an HTTP POST request. At the server side, the data is filtered and then sent to the database.

### 10.2.4 Overview

We should note that the user initiates the acquisition of data from the PIC by pressing on a push button connected as an input to the raspberry pi. At the start of the communication between the Pic and raspberry pi, the raspberry pi lights up the LED. The data received is cleaned from non-numerical characters and then appended to a string. Once the transmission is over, the raspberry Pi turns off the LED and initiates a POST request to send the data to the server.

We should state that we attached an event handler to the button, so that each time the user clicks on the button, we make sure to call the function that gets the data from our microcontroller. Furthermore, the serial receiver of the Raspberry Pi is named ***ttyS0***. It communicates with the PIC at a baud rate of 22,000 bits per second.

### 10.2.5 Code:

```
import serial, json, re, requests
from time import sleep
import RPi.GPIO as GPIO
import time

GPIO.cleanup()
GPIO.setmode(GPIO.BOARD)
GPIO.setwarnings(False)
GPIO.setup(18,GPIO.OUT)
GPIO.setup(16,GPIO.IN, pull_up_down=GPIO.PUD_DOWN)

b = True
def removeUnicode(string):
    return re.sub("[^\. 0-9]", "", string)
print("Starting...")
ser = serial.Serial ("/dev/ttyS0", 22000)    #Open port with baud rate

url = "http://192.168.56.1/receiveData"
```

```
def button_callback(channel):
    array =[]
    st =""
    t=0
    print("starting")
    GPIO.output(18,GPIO.HIGH)
    while (len(st.replace(" ",""))/7)<1000:
        print("t", t)
        sleep(0.01)
        received_data = ser.read()            #read serial port
        data_left = ser.inWaiting()           #check for remaining byte
        received_data += ser.read(data_left)
        received_data = str(received_data)
        print(received_data.replace("\n",""))
        #print received data
        st+=removeUnicode(received_data.strip())
        t+=1
    GPIO.output(18,GPIO.LOW)
    with open('output2.txt', 'w') as filehandle:
            json.dump([st], filehandle)
    print("Done")
    print(requests.post(url , data = json.dumps({"data":st}), headers={"Content-Type":"application/json"}).json())
    return

GPIO.add_event_detect(16, GPIO.RISING, callback=button_callback);

while(True):
    hh=0
```

# 11 Appendix 3

This section will visualize some of the main GUI components of our capstone project. The following screenshots were taken from the application:

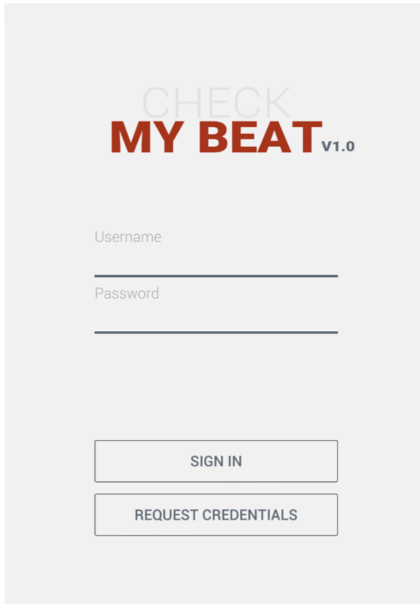



*Figure 63: Sign in panel*

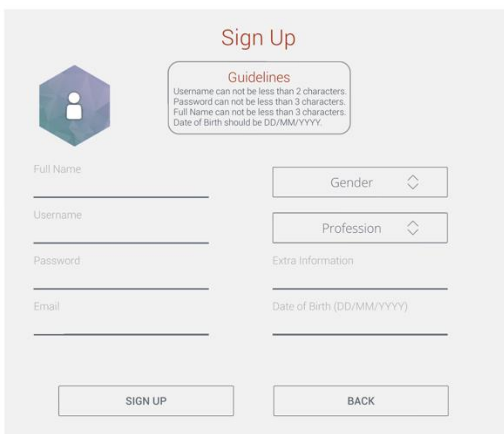



*Figure 64: Sign up panel*

*Figure 65: Menu Panel*

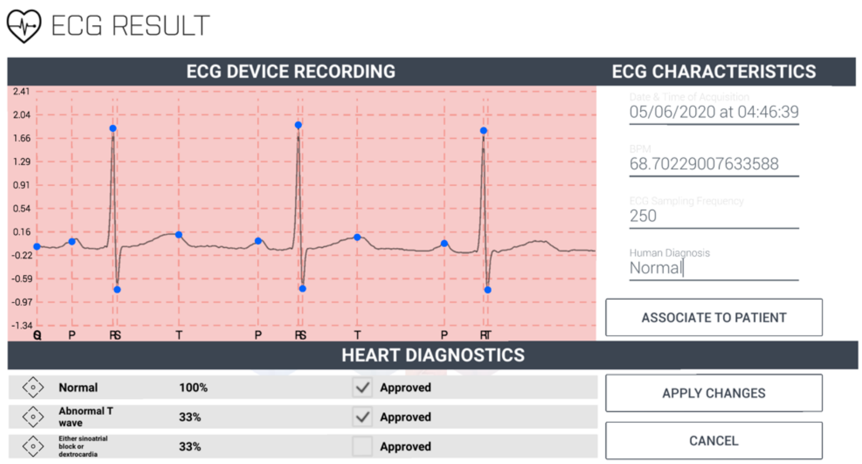



*Figure 66: ECG visualization panel*

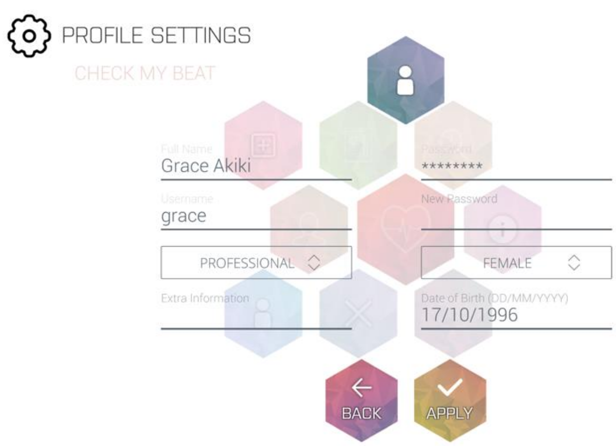



*Figure 67: User Profile Panel*

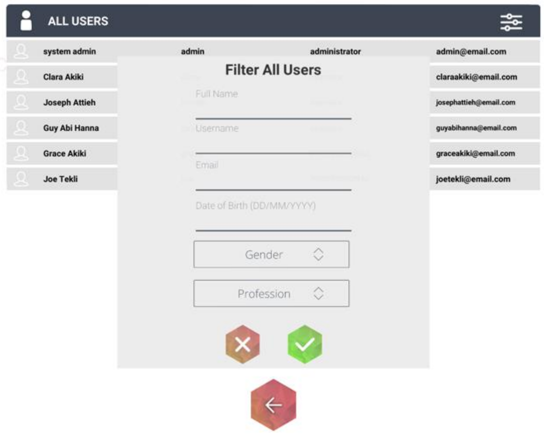



*Figure 68: Searching panel*

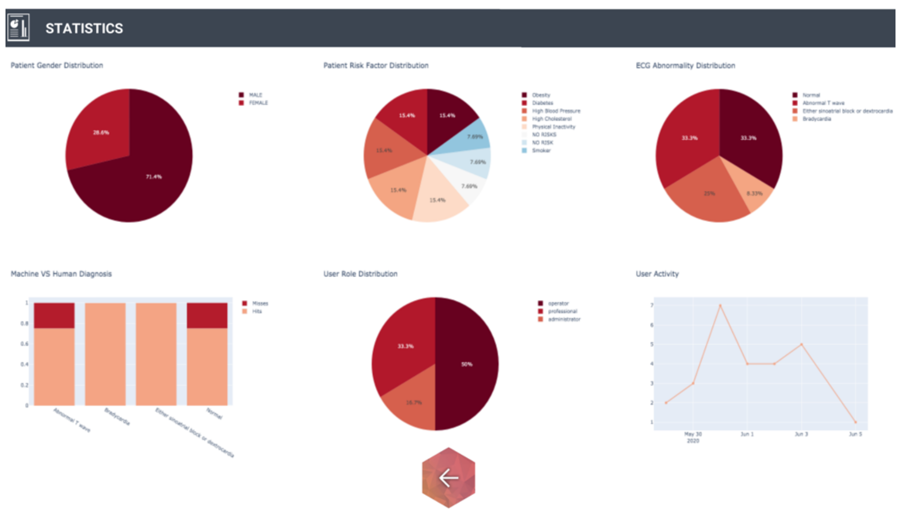



*Figure 69: Statistics panel*

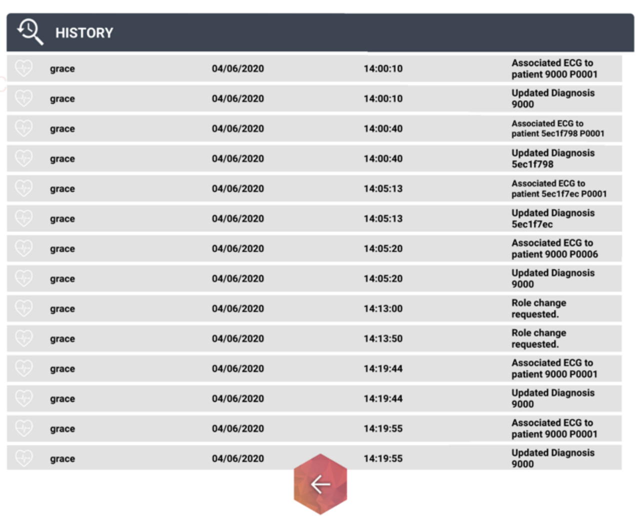



*Figure 70: History panel*


*Figure 71: Information panel*

# References


[1] F. El-Jardali , L. El Bawab , Y. Faraj , M. El Sayed and N. Jabr, "K2P Policy Brief: Strengthening Emergency Medical Services in Lebanon," Beirut, 2017.

[2] E. J. Benjamin , P. Muntner , A. Alonso and et al., "Heart Disease and Stroke Statistics—2019 Update: A Report From the American Heart Association," Circulation, 2019.

[3] M. Imam, "PhD Thesis: Analysis of beat-to-beat ventricular repolarization duration variability from electrocardiogram signal," 2016.

[4] Cables and Sensors, "12-Lead ECG Placement Guide with Illustrations," [Online]. Available: https://www.cablesandsensors.com/pages/12-lead-ecg-placement-guide-with-illustrations.

[5] "Depolarization," [Online]. Available: https://commons.wikimedia.org/wiki/File:1221_Action_Potential.jpg#/media/File:1221_Action_Potential.jpg.

[6] "ECG Reading," [Online]. Available: https://www.amperordirect.com/pc/help-ecg-monitor/z-interpreting-ecg-2.html.

[7] R. E. Klabunde, "Electrocardiogram (EKG, ECG)," [Online]. Available: https://www.cvphysiology.com/Arrhythmias/A009.

[8] R. E. Klabunde, "Electrophysiological Changes During Cardiac Ischemia," [Online]. Available: https://www.cvphysiology.com/CAD/CAD012.

[9] "Components of ECG Signal," [Online]. Available: https://www.ncbi.nlm.nih.gov/books/NBK2214/ .

[10] Physio-Control, Inc., "Lifepak 12 Defibrillator/Monitor," Medtronic, 2007. [Online]. Available: https://www.transmed-usa.com/physio_pdfs/lp12/LP12_Brochure_3207916-000.pdf.

[11] V. N. Dhruva, S. I. Abdelhad, A. Anis, W. Gluckman, D. Hom, W. Dougan, E. Kaluski, B. Haider and M. Klapholz, "ST-segment analysis using wireless technology in acute myocardial infarction (STAT-MI) trial," *Journal of the American College of Cardiology,* vol. 50, no. 6, pp. 509-513, 2007.

[12] P. and P. Singh, "Zigbee and GSM based patient health monitoring system," *International Conference on Electronics and Communication Systems (ICECS),* pp. 1-5, 2014.

[13] S. Anil, "Method and apparatus for a hand-held computer EKG device". U.S. Patent 6,654,631, 25 November 2003.

[14] Physio-Control, Inc., "Lifepak 1000 Defibrillator Product Brochure," 2015. [Online]. Available: https://www.physio-control.com/uploadedFiles/Physio85/Contents/Workplace_and_Community/Products/3308966_D_LR.pdf.

[15] M. G. Rockwell, G. D. Brink, J. N. Andrews, D. L. Burto, P. A. Arand, N. H. Forman, K. S. Rucker, D. J. Powers and Kent, "Defibrillator with wireless communication of ECG signals". Washington, DC Patent 6,405,083, 11 June 2002.

[16] Y. Lina and M. Sriyudthsaka, "Design and Development of Standard 12-Lead ECG Data," in *2016 International Electrical Engineering Congress, iEECON2016, 2-4 March 2016*, Chiang Mai, 2016.

[17] "Wet, dry, active and passive electrodes. What are they, and what to choose?," 09 11 2019. [Online]. Available: https://www.brainlatam.com/blog/wet-dry-active-and-passive-electrodes.-what-are-they-and-what-to-choose-413.

[18] E. Richard and A. D. C. Chan, "Design of a Gel-less Two-Electrode ECG Monitor," IEEE, 2010.

[19] W. M. Noormanshah, P. N. Nohuddin and Z. Zaino, "Document Categorization Using Decision Tree: Preliminary," *International Journal of Engineering & Technology,* vol. 7, pp. 437-440, 2018.

[20] N. Patel and S. Upadhyay, "Study of Various Decision Tree Pruning Methods with their Empirical Comparison in WEKA," *International Journal of Computer Applications,* vol. 60, no. 12, pp. 20-25, 2012.

[21] I. Rish, "An Empirical Study of the Naïve Bayes Classifier," 2001.

[22] L. Jiang, H. Zhang and Z. Cai, "A Novel Bayes Model: Hidden Naive Bayes," *IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING,* vol. 21, no. 10, pp. 1361-1371, 2009.

[23] A. Goyal, D. Bathla, P. Sharma, M. Sahay and S. Sood, "MRI image based patient specific computational model reconstruction of the left ventricle cavity and myocardium," in *2016 International Conference on Computing, Communication and Automation (ICCCA)*, 2016.

[24] N. Meziane, J. G. Webster, M. Attari and A. J. Nimunkar, "Dry electrodes for electrocardiography," *Physiological Measurement,* vol. 34, pp. 47-69, 2013.

[25] P. Selvey, "ECG Filters," 27 02 2017. [Online]. Available: https://www.medteq.net/article/2017/4/1/ecg-filters.