ESCUELA POLITÉCNICA NACIONAL

FACULTAD DE INGENIERÍA ELÉCTRICA Y ELECTRÓNICA

DESARROLLO DE UN SIMULADOR DE SEÑALES CARDÍACAS COMO HERRAMIENTA PARA EL ENTRENAMIENTO MÉDICO

TRABAJO DE TITULACIÓN PREVIO A LA OBTENCIÓN DEL TÍTULO DE INGENIERO EN ELECTRÓNICA Y TELECOMUNICACIONES

CRISTIAN LEONARDO CHACHA ZHUNIO

cristian.chacha@epn.edu.ec

DIRECTOR: Ph.D. FELIPE LEONEL GRIJALVA ARÉVALO

felipe.grijalva@epn.edu.ec

CODIRECTOR: Ph.D. ROBIN GERARDO ÁLVAREZ RUEDA

robin.alvarez@epn.edu.ec

Quito, marzo 2020

AVAL

Certificamos que el presente trabajo fue desarrollado por Cristian Leonardo Chacha Zhunio, bajo nuestra supervisión.

Ph.D. FELIPE LEONEL GRIJALVA ARÉVALO
DIRECTOR DEL TRABAJO DE TITULACIÓN

Ph.D. ROBIN GERARDO ÁLVAREZ RUEDA
CODIRECTOR DEL TRABAJO DE TITULACIÓN

DECLARACIÓN DE AUTORÍA

Yo CRISTIAN LEONARDO CHACHA ZHUNIO, declaro bajo juramento que el trabajo aquí descrito es de mi autoría; que no ha sido previamente presentado para ningún grado o calificación profesional; y, que he consultado las referencias bibliográficas que se incluyen en este documento.

A través de la presente declaración dejo constancia de que la Escuela Politécnica Nacional podrá hacer uso del presente trabajo según los términos estipulados en la Ley, Reglamentos y Normas vigentes.

CRISTIAN LEONARDO CHACHA ZHUNIO

DEDICATORIA

Este trabajo esta dedicado a mi familia,
Karla, Xavier y Amelia mi madre
quien con su apoyo, sacrificio y bendición
a largo de mi vida y de mi formación
me ha permitido llegar donde estoy.

AGRADECIMIENTO

En primer lugar quiero agradecer a Dios por las valiosas lecciones de vida y por darme la fortaleza para seguir adelante ya que en un tramo del desarrollo de este trabajo se presentaron situaciones difíciles que influyeron el tiempo de realización de esta tesis.

A mi familia quienes son mi motivación día a día y por darme el apoyo incondicional todos estos años. Los quiero mucho.

A todas aquellas personas que me apoyaron y me brindaron su ayuda a lo largo de esta etapa en la universidad. Dios bendiga su generosidad.

Al Ph.D. Felipe Grijalva por su ayuda, paciencia y por la confianza que tuvo en mí para realizar este trabajo.

ÍNDICE DE CONTENIDO

RESUMEN

La simulación médica es una parte importante en la etapa de formación de los futuros profesionales de la salud y debería ser implementada en las Universidades y Centros de Educación Médica pero actualmente el costo de los equipos y la infraestructura sofisticada limita su despliegue.

En este trabajo se presenta una aplicación con fines educativos como una alternativa accesible, simple y de bajo costo. La aplicación permite simular señales cardíacas y escenarios clínicos.

La aplicación fue desarrollada en Python, donde se diseñaron y codificaron dos interfaces gráficas: una que corresponde para el profesor y otra para el estudiante las cuales se comunican entre sí mediante sockets. Además, se modeló la señal ECG (Electrocardiograma) normal con la posibilidad de modificar los valores de sus amplitudes e intervalos que la conforman. También se implementaron seis patologías cardíacas a partir de bases de datos disponibles en *PhysioNet*. Finalmente, se codificó funciones para realizar desfibrilación, cardioversión eléctrica y aplicación de diferentes medicamentos.

La aplicación fue revisada por dos médicos quienes verificaron las funcionalidades del software. Los médicos manifestaron que la interfaz gráfica es sencilla de utilizar, validaron la calidad e integridad de la señal ECG y también contribuyeron con recomendaciones para potenciar la aplicación.

PALABRAS CLAVE: ECG, Python, Simulación Médica, GUI, Sockets, PyQt.

ABSTRACT

Medical simulation is an important part in the training stage of future health professionals and should be implemented in Universities and Medical Education Centers but currently the cost of equipment and sophisticated infrastructure limits their deployment.

This project presents an application for educational purposes as an accessible, simple and low-cost alternative. The application allows simulating cardiac signals and clinical scenarios.

The application was developed in Python, where two interfaces were designed and coded: one that corresponds to the teacher and another to the student, which communicate with each other through sockets. In addition, the normal ECG (Electrocardiogram) signal was modeled with the possibility of modifying the values of its amplitudes and intervals that comprise it. Six cardiac pathologies were also implemented from databases available on *PhysioNet*. In addition, functions for defibrillation, electrical cardioversion and medication were implemented.

The application was reviewed by two doctors who verified the functionality of the software. The doctors said that the graphical interface is simple to use, validated the quality and integrity of the ECG signal and also contributed with recommendations to enhance the application.

KEYWORDS: ECG, Python, Medical Simulation, GUI, Sockets, PyQt.

1. INTRODUCCIÓN

La simulación en el campo de la salud consiste en situar al estudiante en un contexto que reproduzca un aspecto de la realidad y establecer en ese ambiente situaciones similares a las que deberá afrontar con pacientes sanos o enfermos [1]. La simulación en la Educación Médica ha experimentado un gran desarrollo a nivel mundial, esta se ha convertido en una herramienta que favorece la adquisición de habilidades médicas previo al contacto real con el paciente y fomenta la seguridad para el estudiante, disminuyendo así la posibilidad de cometer errores o complicaciones en la realización de procedimientos médicos [2].

Los simuladores médicos hay de dos tipos: los que están destinados para la enseñanza médica y los que se utilizan para el servicio técnico o prueba de equipos médicos. En el Ecuador se ha estado incorporando la simulación al panorama educativo a través de Clínicas de simulación para las Escuelas de Medicina. Estos centros de simulación se encuentran equipados con equipos de simulación de alta fidelidad y maniquíes que simulan varias partes del organismo humano. Los costos de los equipos de simulación bloquean la expansión de su utilización impidiendo un alcance masivo [3].

Profesores y estudiantes necesitan un entrenamiento que proporcione una modalidad educativa empírico-práctica permitiendo la participación proactiva de los alumnos a escenarios clínicos comunes, complejos y poco frecuentes con las prácticas tradicionales y teóricas. Una de las grandes diferencias en la enseñanza de la medicina con el modelo tradicional y la enseñanza basada en la simulación, es durante el entrenamiento clínico en pacientes reales donde los alumnos deben de estar continuamente supervisados para evitar que cometan errores y corregirlos de manera inmediata. Esto con el fin de cuidar la integridad y seguridad del paciente. En contraste, dentro de una simulación, los errores son permitidos por el instructor, con el fin de que el alumno aprenda de las consecuencias de su error, rectifique y vuelva a realizar el procedimiento de manera correcta, reforzando así sus conocimientos [1].

El problema que vamos a enfrentar en el presente proyecto es la escasa incorporación de herramientas basadas en simulación para la educación en la medicina, que se convierte en un aspecto importante en los currículums para los futuros profesionales de la salud. Si no se realiza este proyecto los estudiantes no podrán poner en práctica sus conocimientos lo cual no favorecerá tanto a la adquisición de habilidades y a la potencialización en la formación de los estudiantes. El presente proyecto proporciona una herramienta didáctica orientada a la enseñanza y aprendizaje de futuros profesionales de la salud, que sea capaz de simular señales cardíacas y seis señales patológicas. La herramienta se compone de dos interfaces de usuario correspondientes para el uso del profesor y del estudiante. Las interfaces mencionadas muestran gráficamente las señales cardíacas que surgen de la generación y ajuste de los parámetros básicos realizados por el profesor. Esta herramienta permite crear diferentes escenarios que ayudará a la formación de los estudiantes.

1.1. OBJETIVOS

El objetivo general de este Proyecto Técnico es desarrollar una aplicación en Python que permita simular señales eléctricas del corazón de forma que su utilización este orientada a la enseñanza médica.

Los objetivos específicos de este Proyecto Técnico son:

- Estudiar las señales eléctricas del corazón.
- Desarrollar un algoritmo que genere y muestre gráficamente la señal eléctrica básica del corazón.
- Implementar un generador de señales patológicas cardíacas. Las señales que se implementarán son: Arritmia Sinusal, Fibrilación Auricular, Fibrilación Ventricular, Taquicardia Ventricular, Taquicardia Supraventricular y Ectopia Ventricular Maligna.
- Diseñar e implementar la interfaz gráfica del médico docente.
- Diseñar e implementar la interfaz gráfica del estudiante.

1.2. ALCANCE

El proyecto está orientado a la implementación de un simulador de señales cardíacas mediante software cuya función principal está destinada a la enseñanza médica. Para la simulación de las señales cardíacas se ha elegido el lenguaje de programación Python. Cabe mencionar que el sistema sólo considerará de un médico docente y un estudiante.
El simulador presentará las siguientes características:

- Permitirá generar la señal eléctrica básica de un corazón sano con ajuste de parámetros como la frecuencia cardíaca, intervalos de duración y amplitudes de las ondas.
- Permitirá cargar bases de datos para graficar las siguientes señales patológicas cardíacas: arritmia sinusal, fibrilación auricular, taquicardia ventricular, arritmia supraventricular, fibrilación ventricular y ectopia ventricular maligna.
- Constará de una interfaz gráfica para el profesor y otra para el estudiante. Los parámetros básicos de las simulaciones serán establecidas en la interfaz del profesor y se enviarán a través de una conexión de red a la interfaz del estudiante.
- La interfaz del estudiante permitirá simular la desfibrilación, cardioversión y medicación mediante controles ubicados en su interfaz.

1.3. MARCO TEÓRICO

En esta sección se abordarán los conceptos necesarios para la realización del proyecto. En la sección 1.3.1 se revisará los conceptos relacionados con la fisiología del corazón. En

la sección 1.3.2 revisaremos fundamentos sobre el electrocardiograma en el que describen las diferentes derivaciones del electrocardiograma. Posteriormente, en la sección 1.3.3 se analizará los componentes que conforman la señal ECG (Electrocardiograma). Por otra parte, en la sección 1.3.4 se revisan las señales patológicas utilizadas en este estudio. En la sección 1.3.5 se indica los fármacos implementados en el proyecto. En la sección 1.3.6 analizaremos el procedimiento de desfibrilación y cardioversión eléctrica. Finalmente en la sección 1.3.7 nos enfocaremos en detallar las herramientas utilizadas en el proyecto.

1.3.1. Fisiología del Corazón

El corazón es un órgano importante que forma parte del sistema cardiovascular funciona como una bomba para proporcionar una circulación continua de sangre por todo el cuerpo. El corazón está dividido en 4 cavidades: 2 en el lado derecho y 2 en el izquierdo (ver Figura1.1). Cada cámara superior se conoce como aurícula y cada cámara inferior como ventrículo. Los 4 compartimentos se conocen como: aurícula derecha, ventrículo derecho, aurícula izquierda y ventrículo izquierdo. La sangre ingresa al corazón a través de las aurículas que son las cámaras más pequeñas y se bombea a través de las más grandes que son los ventrículos [4].

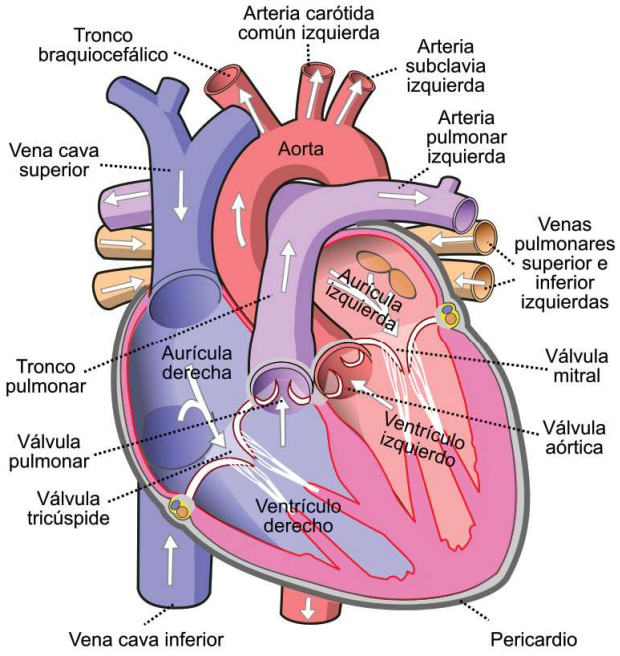

Figura 1.1: Cavidades del Corazón. Fuente [5]

1.3.1.1. Funcionamiento del Corazón

El corazón se contrae a diferentes velocidades dependiendo de muchos factores. En reposo, puede latir alrededor de 60 veces por minuto, pero puede aumentar a 100 latidos por minuto o más. El ejercicio, las emociones, la fiebre, las enfermedades y algunos medicamentos pueden influir en la frecuencia cardíaca. Los lados izquierdo y derecho del corazón funcionan al unísono. El lado derecho del corazón recibe sangre desoxigenada y la envía a los

pulmones; El lado izquierdo del corazón recibe sangre de los pulmones y la bombea al resto del cuerpo. Las aurículas y los ventrículos se contraen y se relajan a su vez, produciendo un latido rítmico. Cada latido se puede dividir en dos partes [6]:

- **Diástole :** las aurículas y los ventrículos se relajan y se llenan de sangre.
- **Sístole :** las aurículas se contraen (sístole auricular) y empujan la sangre hacia los ventrículos; luego, cuando las aurículas comienzan a relajarse, los ventrículos se contraen (sístole ventricular) y bombean sangre fuera del corazón.

1.3.1.2. Frecuencia Cardíaca

La frecuencia cardíaca es uno de los signos vitales o indicadores de salud más importantes en el cuerpo humano. La frecuencia cardíaca representa la cantidad de veces que el corazón late en el espacio de un minuto. La frecuencia cardíaca depende de factores como: la actividad física, el estado psicológico, las respuestas emocionales, condiciones ambientales, enfermedad, la edad, etc. La frecuencia cardíaca en reposo se denomina ritmo sinusal y se refiere a la frecuencia cardíaca cuando una persona está relajada [7].

Es importante identificar si la frecuencia cardíaca se encuentra dentro del rango normal. Si una enfermedad o lesión debilita el corazón, los órganos no recibirán suficiente sangre para funcionar normalmente. La frecuencia cardíaca normal en reposo para personas mayores de 10 años incluidos los adultos mayores, es de entre 60 y 100 latidos por minuto (lpm). Cuando esta disminuye por debajo de 60 recibe el nombre de bradicardia y si aumenta por encima de 100 se denomina taquicardia [7].

1.3.1.3. Actividad Eléctrica del Corazón

Para bombear sangre a todo el cuerpo, los músculos del corazón deben coordinarse perfectamente, comprimiendo la sangre en la dirección correcta, en el momento correcto, a la presión correcta. La actividad del corazón está coordinada por impulsos eléctricos y viajan por un camino especial a través del corazón [8].

- La señal eléctrica comienza en el nodo sinoauricular (SA) conocido como el marcapasos natural del corazón, ubicado en la parte superior de la aurícula derecha. Esta señal hace que las aurículas se contraigan, empujando la sangre hacia los ventrículos [8].
- Posteriormente la señal eléctrica viaja a un área de células en la parte inferior de la aurícula derecha llamada nodo auriculoventricular. Estas células actúan como una puerta; disminuyen la velocidad de la señal para que las aurículas y los ventrículos no se contraigan al mismo tiempo [8].
- Desde aquí, la señal se transporta a lo largo de fibras de Purkinje dentro de las paredes del ventrículo; transmiten el impulso al músculo cardíaco haciendo que los ventrículos se contraigan [8].
- El nodo SA envía otro impulso eléctrico para contraerse y el ciclo inicia nuevamente [8].

1.3.2. Electrocardiograma

Un electrocardiograma (ECG) es una prueba que se usa para la identificación de trastornos del ritmo cardíaco y para el diagnóstico de anomalías del corazón. Esta actividad se registra en hojas de gráficos o en algunos tipos de monitores colocando los electrodos en ubicaciones específicas del cuerpo de una persona. El registro muestra una serie de ondas eléctricas que ocurren durante cada latido del corazón. Las ondas registradas tienen picos y valles; normalmente están representadas por las letras P, Q, R, S, T y U. La onda U no es consistente y es invisible entre el 70 % de las personas. Por lo tanto, clínicamente la onda U no es tan importante como las otras ondas [9].

1.3.2.1. Derivaciones del Electrocardiograma

El electrocardiograma estándar de 12 derivaciones muestra la actividad eléctrica del corazón a través de 12 perspectivas diferentes. Estas 12 vistas se recopilan colocando electrodos en el pecho, muñecas y tobillos para que puedan registrar con precisión desde las diferentes perspectivas las imágenes de la actividad eléctrica del corazón (ver Figura1.2). En este estudio nos concentraremos en las derivaciones bipolares [9].

- **Derivaciones de las Extremidades Estándar o Bipolares: I**, II Y III.
- **Derivaciones Aumentadas de las Extremidades:** aVR, aVL y aVF.
- **Derivaciones Torácicas o Precordiales:** V1, V2, V3, V4, V5 y V6.

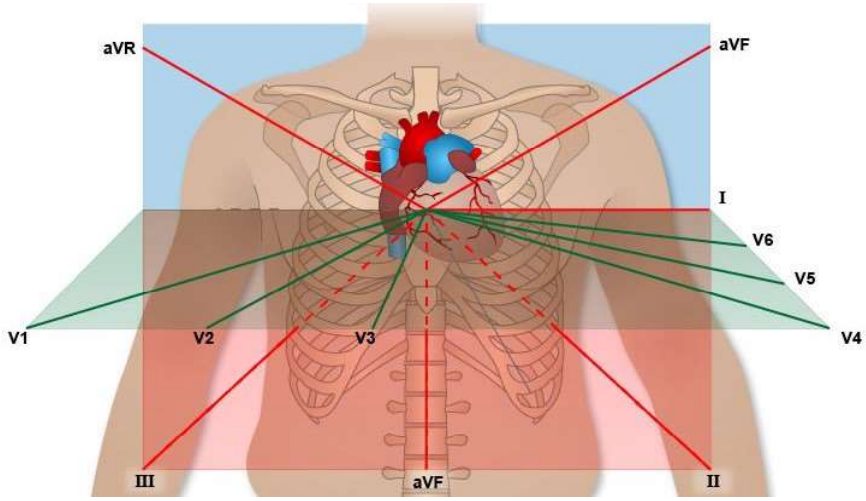

Figura 1.2: Derivaciones del Electrocardiograma. Fuente [10]

1.3.2.2. Derivaciones Bipolares

Son las derivaciones cardíacas clásicas del electrocardiograma descritas por Einthoven. Registran la diferencia de potencial entre dos electrodos ubicados en extremidades diferentes. Consta de las derivaciones DI, DII y DII. Estas tres derivaciones forman un triángulo equilátero en el que el corazón se encuentra en el centro [9].

- **D I:** muestra la diferencia eléctrica entre el brazo izquierdo (electrodo positivo) y el brazo derecho (electrodo negativo) Figura1.3a.
- **D II:** muestra la diferencia eléctrica entre el brazo derecho (electrodo negativo) y el pie izquierdo (electrodo positivo) Figura1.3b.
- **D III:** muestra la diferencia eléctrica entre el brazo izquierdo (electrodo negativo) y el pie izquierdo (electrodo positivo) Figura1.3c.

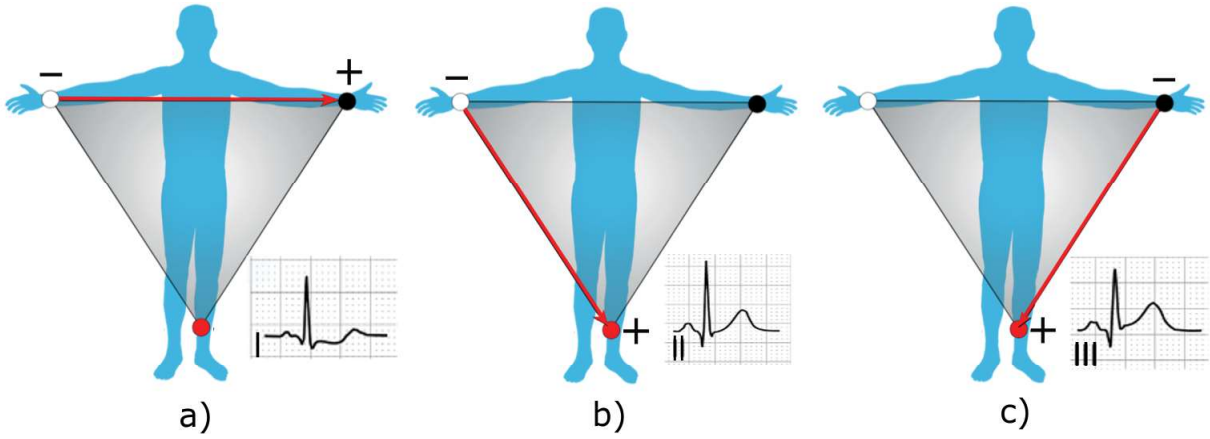

Figura 1.3: Derivaciones Bipolares. a) Derivación I. b) Derivación II. c) Derivación III. Adaptado de [11]

1.3.3. Señal ECG

El patrón básico del electrocardiograma se compone de: tres ondas (onda P, onda T y en ocasiones una onda U), el complejo QRS, dos intervalos (PR y QT) y tres segmentos (PQ, ST y TP) [12] (ver Figura 1.4).

1.3.3.1. Ondas, Segmentos e Intervalos de la Señal ECG

Todas las ondas del electrocardiograma y los intervalos entre ellas tienen una duración de tiempo, un rango de amplitudes (voltajes) aceptables y una morfología típica. Cualquier variación del trazado normal es potencialmente patológica y, por lo tanto, de consideración clínica [12].

Onda P

La onda P es la primera onda del ciclo cardíaco. Representa la despolarización de las aurículas. La duración normal de la onda P es menor de 0.12 s y una amplitud máxima de 0.25 mV. Es positiva en las derivaciones I y II, excepto en la derivación aVR donde es negativa [12].

Intervalo PR

Es el período de tiempo desde el inicio de la onda P hasta el comienzo del complejo QRS. Este intervalo representa el tiempo entre el inicio de la despolarización auricular y el inicio de la despolarización ventricular. Normalmente varía de 0,12 a 0,20 segundos de duración [12].

Segmento PR

Es linea plana entre el final de la onda P y el inicio del complejo QRS. El segmento PR sirve como línea base (conocida también como línea de referencia o linea isoeléctrica) de la curva ECG. La amplitud de cualquier onda o deflexión se mide utilizando el segmento PR como línea de referencia [12].

Complejo QRS

El complejo QRS está formado por tres ondas: Q, R y S. Su duración oscila entre 0.06 y 0.12 segundos. Toma varias morfologías dependiendo de la derivación [12].

- **Onda Q:** representa la despolarización septal, suele ser estrecha y poco profunda. Tiene una duración menor o igual a 0.04 s de ancho y de 0.2 mV de profundidad, en general no supera el 25 % de la onda R [12].
- **Onda R:** representa el estímulo eléctrico a medida que pasa a través de la porción principal de las paredes ventriculares. La pared de los ventrículos es muy gruesa debido a la cantidad de trabajo que tienen que hacer, en consecuencia, se requiere más voltaje. La amplitud en las derivaciones I, II y III debe ser menor o igual a 2 mV [12].
- **Onda S:** es cualquier deflexión negativa que siga a la onda R, representa la despolarización de las fibras de Purkinje. Tiene una duración no mayor a 0.06 s y su amplitud varía entre 0.2 mV a 1.5 mV [12].

Segmento ST

El segmento ST es una línea isoeléctrica, un período de tiempo sin actividad eléctrica del corazón. Representa el período de tiempo entre el fin de la despolarización ventricular y el inicio de la repolarización. Debe estar en el mismo nivel que el intervalo PR. Cada elevación o depresión de esta línea es patológica. Tiene una duración de 5 a 150 ms [12].

Onda T

Representa la repolarización de los ventrículos. Generalmente es de menor amplitud que el QRS que le precede. Su amplitud máxima es menor de 0.5 mV y su duración es de 0.10 a 0.25 segundos [12].

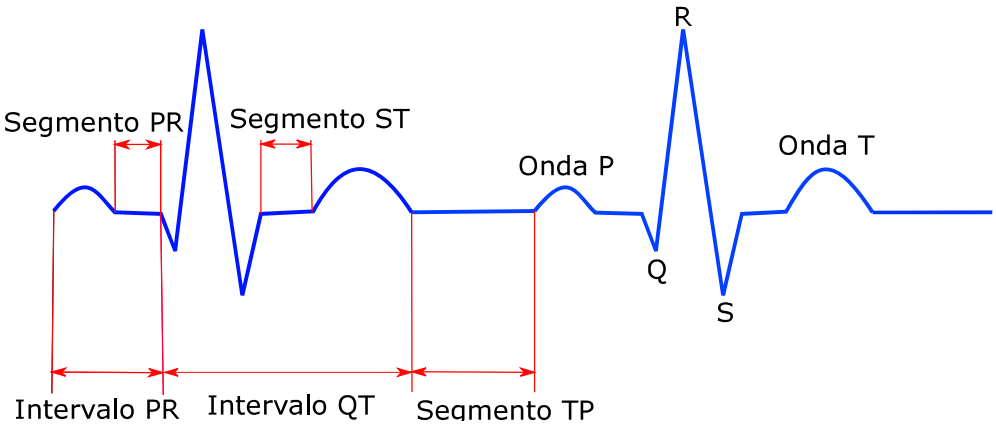

Figura 1.4: Componentes de la ECG estándar. Fuente Propia

Intervalo QT

El intervalo QT se mide desde el inicio del complejo QRS hasta el final de la onda T. Representa la duración de la sístole ventricular (el conjunto de la despolarización y repolarización). El intervalo QT no varía con la edad, pero sí con la frecuencia cardiaca. Sus duraciones son normalmente menores o iguales a 0.40 segundos para los hombres y 0.44 segundos para las mujeres [12].

Segmento TP

Es el intervalo isoeléctrico en el electrocardiograma comprendido entre el final de la onda T de un ciclo cardíaco y el comienzo de la onda P del ciclo siguiente. El intervalo TP se acorta cuando aumenta la frecuencia cardíaca y viceversa. [12].

1.3.4. Arritmias Cardíacas

El ECG normal consiste en series repetitivas de ondas P, Q, R, S y T, que se ajustan a los estándares establecidos para el tamaño y la forma; ocurren de 60 a 100 veces por minuto. Si estas condiciones prevalecen, el corazón está en ritmo sinusal normal. Cuando la velocidad de cualquiera de las ondas individuales es anormal, el trastorno se llama arritmia. Las arritmias se pueden clasificar por el lugar de origen de la arritmia, por su frecuencia cardíaca y por su modo de presentación [13].

Tabla 1.1: Clasificación de las Arritmias

ORIGEN	FRECUENCIA CARDÍACA	MODO DE PRESENTACIÓN
Supraventriculares: se originan en las aurículas	Taquicardias: mayor a 100 lpm	Paroxísticas: de carácter ocasional
Ventriculares: se originan en los ventrículos	Bradicardias: menor a 60 lpm	Crónicas: de naturaleza permanente

1.3.4.1. Arritmia Sinusal

La arritmia sinusal es una variación del ritmo sinusal normal. La arritmia sinusal se caracteriza por tener una frecuencia irregular en la que la variación en el intervalo RR es mayor de 0.12 segundos (ver Figura 1.5). Además, las ondas P tienen una morfología normal (i.e. sin evidencia de contracciones auriculares prematuras) [14].

Figura 1.5: Arritmia Sinusal. Fuente [15]

1.3.4.2. Fibrilación Auricular

Esta arritmia ocurre debido a múltiples impulsos eléctricos que salen de la aurícula. En el electrocardiograma las ondas P aparecen como líneas ondulantes sin una onda P visible. Las ondas P normales se reemplazan con una pequeña deflexión de amplitud variable ya que la frecuencia cardiaca es extremadamente rápida (mayor a 300 lpm) [14] (ver Figura 1.6).

Figura 1.6: Fibrilación Auricular. Fuente [15]

1.3.4.3. Taquicardia Ventricular

Es una arritmia muy grave. Los pacientes que presentan taquicardia ventricular a menudo presentan una frecuencia cardíaca entre 100 y 250 lpm. Las ondas P pueden estar presentes o ausentes y estas pueden ser positivas o negativas. Los complejos QRS se describen como "de aspecto salvaje" con grandes oscilaciones y superan los 0.12 segundos (ver Figura 1.7). Es potencialmente peligrosa porque puede desencadenar fibrilación ventricular o muerte súbita [14].

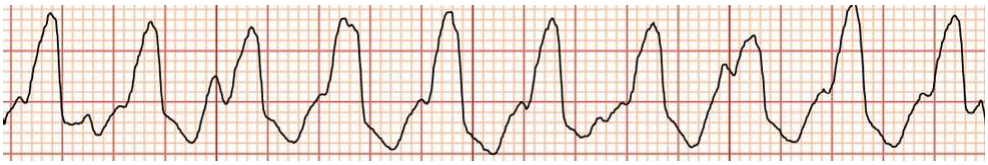

Figura 1.7: Taquicardia Ventricular. Fuente [16]

1.3.4.4. Fibrilación Ventricular

La fibrilación ventricular es una de las causas más comunes de paro cardíaco. En el electrocardiograma se visualiza una señal irregular de morfología caótica. No se pueden ver complejos QRS, no hay ondas P y no hay intervalos PR (ver Figura 1.8). Puede ser mortal por lo que necesita atención médica inmediata [14].

Figura 1.8: Fibrilación Ventricular. Fuente [17]

1.3.4.5. Taquicardia Supraventricular

La taquicardia supraventricular ocurre cuando las señales eléctricas en las cámaras superiores del corazón se disparan de manera anormal, lo que interfiere con las señales eléctricas que provienen del nodo SA (el marcapasos natural del corazón). La frecuencia cardíaca se acelera a 150 lpm o más [14] (ver Figura 1.9).

Figura 1.9: Taquicardia Supraventricular. Fuente [15]

1.3.4.6. Ectopia Ventricular Maligna

La ectopia ventricular es una presentación clínica común en pacientes con arritmias del tracto de salida del ventrículo idiopático. Las opciones de tratamiento incluyen vigilancia clínica, terapia médica con agentes antiarrítmicos o ablación con catéter. La terapia médica puede ofrecer un beneficio sintomático, pero puede tener efectos secundarios y, por lo general, resulta en una reducción de la carga en lugar de la erradicación de la ectopia [18]. Su forma de onda se puede observar en la Figura 1.10.

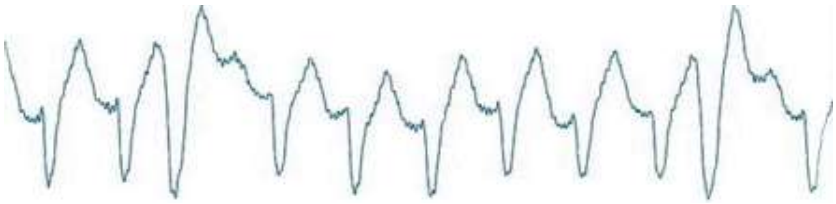

Figura 1.10: Ectopia Ventricular Maligna. Fuente [19]

1.3.5. Fármacos en Cardiología

La utilización de sustancias farmacológicas es parte del tratamiento que se le puede dar a un paciente para enfrentar diversas anormalidades que se presentan. Con un tratamiento adecuado se puede prevenir, atenuar los síntomas e incluso curar alguna enfermedad. Es importante conocer los principales efectos positivos, la correcta dosificación, la forma de administración y los efectos secundarios que se puedan generar. Se puede ocasionar empeoramiento o inducir nuevas complicaciones si se incumple el correcto tratamiento farmacológico [20].

Existen diversos tipos de fármacos antiarrítmnicos; algunos de ellos son beneficiosos simultáneamente para varias enfermedades. Dependiendo de la dolencia del paciente, el médico elegirá aquel que resulte más eficaz y seguro. Los fármacos cardiovasculares se agrupan por categorías. Los pertenecientes a una misma categoría son similares entre sí, pero con pequeñas diferencias [20]. Las principales categorías de los fármacos antiarrítmicos son las siguientes:

- **Clase I:** Bloqueantes de los canales de sodio, se subdividen en IA (bloqueo intermedio), IB (bloqueo rápido) y IC (bloqueo lento). Ejemplos: Quinidina, Lidocaína, Procainamida, Disopiramida, etc [21].
- **Clase II:** Son betabloqueantes disminuyen las pulsaciones por minuto y el trabajo que el corazón necesita para bombear la sangre. Ejemplos: Atenolol, Propanolol, Bisoprolol, Carvedilol [21].

- **Clase III:** Permiten controlar y regular el ritmo del corazón. Prolongan la duración del potencial cardíaco y en consecuencia el período refractario efectivo. Ejemplos: Amiodarona, Sotalol, Bretilio, Ibutilida, etc [21].

- **Clase IV:** Bloqueantes de los canales de calcio. Ejemplos: Verapamilo, Diltiazem [21].

1.3.5.1. Epinefrina

La epinefrina también llamada adrenalina es un fármaco clase II, se usa como medicamento para estimular el corazón durante un paro cardíaco (e.g. fibrilación ventricular, taquicardia ventricular sin pulso y asistolia), para la bradicardia sintomática como alternativa a la infusión de dopamina. También se usa para aumentar la presión arterial en casos de shock, como broncodilatador y antiespasmódico en el asma bronquial. Se administra por vía intravenosa [22] [23].

Dosis
La dosis habitual varía de 1-5 mg [23].

Efectos Adversos
Puede producir latidos cardíacos acelerados o anormales en raras ocasiones, nerviosismo, sudoración, náuseas, vómitos, dificultad para respirar, dolor de cabeza, mareos, ansiedad, temblores o piel pálida [24].

1.3.5.2. Lidocaína

La lidocaína es un fármaco antiarrítmico de clase IB que se ha convertido en uno de los medicamentos más utilizados en el tratamiento de las arritmias ventriculares como la taquicardia ventricular, fibrilación ventricular y fibrilación auricular [25].

Dosis
Se administra por vía intravenosa en una dosis de 1 a 3 mg/kg [24].

Efectos Adversos
Puede producir bradicardia, mareos, náuseas, convulsiones, espasmos musculares o extrasístoles ventriculares [24].

1.3.5.3. Quinidina

Antiarrítmico de clase IA derivado de la quinina. Este medicamento se usa para tratar y prevenir muchos tipos de latidos cardíacos irregulares como arritmias cardíacas supraventriculares, fibrilación ventricular, taquicardia ventricular, ectopia ventricular maligna y fibrilación auricular [26].

Dosis
Se administra al paciente de 100 a 600 mg por dosis por vía oral [26].

Efectos Adversos
Las reacciones adversas con mayor frecuencia han sido consistentemente gastrointestinales como diarrea, náuseas, vómitos y ardor de estómago. Manifestaciones cardíacas co-

mo taquiarritmias, torsade de pointes, alteraciones de la conducción e insuficiencia cardíaca [26].

1.3.5.4. Amiodarona

La amiodarona es uno de los fármacos antiarrítmicos más utilizados y muy eficaz en una de las arritmias cardíacas más frecuentes como lo es la fibrilación auricular. Se usa también para el tratamiento de la fibrilación ventricular, taquicardia supraventricular y taquicardia ventricular [27].

Dosis
La amiodarona está disponible para administración oral e intravenosa. Una dosis típica vía oral es de 300-350 mg [27].

Efectos Adversos
La amiodarona suele ser bien tolerada, pero puede causar muchos efectos secundarios, como dolor de cabeza, mareo y alteraciones gastrointestinales; con menos frecuencia puede producir alteraciones de la tiroides, del hígado, de los pulmones o de los ojos [27].

1.3.6. DESFIBRILACIÓN y CARDIOVERSIÓN

La cardioversión eléctrica y la desfibrilación son procedimientos en el tratamiento de pacientes con arritmias cardíacas mediante la aplicación de corriente eléctrica. El objetivo en ambos es entregar una cantidad de energía al corazón para aturdir momentáneamente el corazón y así permitir que se active un ritmo sinusal normal [28]. En la cardioversión eléctrica la energía aplicada es menor y se realiza de forma sincronizada, es decir, se aplica la descarga cuando el equipo detecta la onda R del ECG, mientras que la desfibrilación la energía aplicada es mayor y se realiza de forma no sincronizada [19].

1.3.6.1. Desfibrilación

Es el tratamiento para las arritmias potencialmente mortales con las cuales el paciente no tiene pulso, es decir, para las patologías como fibrilación ventricular, taquicardia ventricular sin pulso o la ectopia ventricular. Los niveles de energía para la desfibrilación para un desfibrilador bifásico el algoritmo de RCP recomienda aplicar descargas inicialmente de 150-200 Joules y descargas posteriores de 150-360 Joules [29].

1.3.6.2. Cardioversión

Es un procedimiento que restaura el ritmo cardíaco normal (i.e. ritmo sinusal) en personas que tienen determinados tipos de arritmias tales como: taquicardia, taquicardia ventricular, fibrilación auricular, taquicardia supraventricular, etc. En la cardioversión, el choque debe sincronizarse adecuadamente, de modo que no ocurra durante el período vulnerable, es decir, durante la onda T. El nivel de energía recomendado para realizar la cardioversión es de 100-200 Juoles [30].

1.3.6.3. Riesgos

La energía eléctrica puede terminar un ritmo anormal, pero si se entrega de manera inapropiada, también puede inducir arritmias graves. Esto puede suceder cuando no se selecciona el modo síncrono en la cardioversión, cuando se aplica el choque eléctrico sobre la onda T o cuando se aplica sobre el segmento ST. Al realizar la desfibrilación se debe verificar el ritmo cardíaco para asegurarse de que una descarga esté realmente indicada, y así, evitar daños miocárdicos y quemaduras superficiales [30].

1.3.7. HERRAMIENTAS UTILIZADAS

1.3.7.1. Python

Python es un lenguaje de programación de código abierto de uso general, multiplataforma, orientado a objetos y de alto nivel. Incorpora módulos, excepciones, tipos de datos dinámicos y clases. Su sintaxis es simple y fácil de aprender ya que Python tiene una estructura simple y sintaxis claramente definida. Python utiliza módulos y paquetes, lo que fomenta la modularidad del programa y la reutilización de código. La biblioteca de Python es extensa cuenta con módulos con funcionalidades para: servicios de res, multiprocesamiento, bases de datos, navegadores web, correo electrónico, XML, HTML, procesamiento de audio y video, GUIs y muchas otras funciones específicas del sistema [31].

1.3.7.2. Anaconda Distribution

Anaconda es una plataforma gratuita y de código abierto para la gestión e implementación de paquetes. Facilita el desarrollo para la ciencia de datos y Machine Learning a través de Python y R. Posee una colección de más de 1500 mas paquetes de código abierto. Anaconda simplifica el proceso de gestión e implementación de paquetes. Posee un sistema de gestión de paquetes llamado Conda para instalar, ejecutar y actualizar paquetes junto con sus dependencias. También proporciona una función integrada para crear propios paquetes de Python y subirlos a los servidores de Anaconda. Anaconda Distribution también incluye una interfaz gráfica de usuario llamada Anaconda Navigator que simplifica el uso de conda sin el uso de comandos [32].

Características de Anaconda Distribution
Anaconda cuenta con una gran cantidad de características entre las que podemos resaltar las siguientes [32]:

- Posee una amplia y detallada documentación.
- Multiplataforma (Linux, MacOS y Windows).
- Bibliotecas de ciencia de datos como: Numpy, Numba, Dask y Pandas.
- Posee bibliotecas para Machine Learning como TensorFlow , Scikit-Learn y Theano.
- Utiliza diversos IDE como Jupyter, JupyterLab, Spyder y RStudio.
- Posee bibliotecas de visualización como: Matplotlib, Bokeh, Holoviews y Datashader.

- Elimina problemas de dependencia y control de versiones de librerías.

1.3.7.3. PyQt

PyQt es un conjunto de herramientas de Python para el marco de aplicaciones multiplataforma que combina todas las ventajas de Qt y Python. Con PyQt se puede incluir bibliotecas Qt en el código Python lo que le permite escribir aplicaciones GUI en Python. PyQt fue desarrollado por Riverbank Computing Limited. PyQt está disponible en dos ediciones: PyQt4 que es compatible con Qt v4 y PyQt5 que es compatible con Qt v5. Ambas ediciones se pueden construir para Python 2 y 3 [33].
Qt mas que un kit de herramientas GUI incluye también abstracciones de sockets, hilos, Unicode, bases de datos SQL, soporte SVG, OpenGL, analizador XML, navegador web, un sistema de ayuda, un marco multimedia, así como una variedad de widgets GUI. el Módulo QtCore contiene una funcionalidad no GUI para trabajar con archivos y directorios. El módulo QtGui contiene todos los controles gráficos. Las clases Qt emplean un mecanismo de signal/slot para la comunicación entre objetos. Qt también incluye Qt Designer, un diseñador gráfico de interfaz de usuario. PyQt es capaz de generar código Python desde Qt Designer [33].

Qt Designer

Qt Designer es la herramienta de Qt para diseñar y construir interfaces gráficas de usuario (GUI) con Qt Widgets. El usuario puede diseñar sus ventanas o cuadros de diálogo de una manera que se lo pueda ir visualizando y probarlos con diferentes estilos y resoluciones [34]. En la Figura1.11 se observa el entorno de desarrollo de Qt Designer.

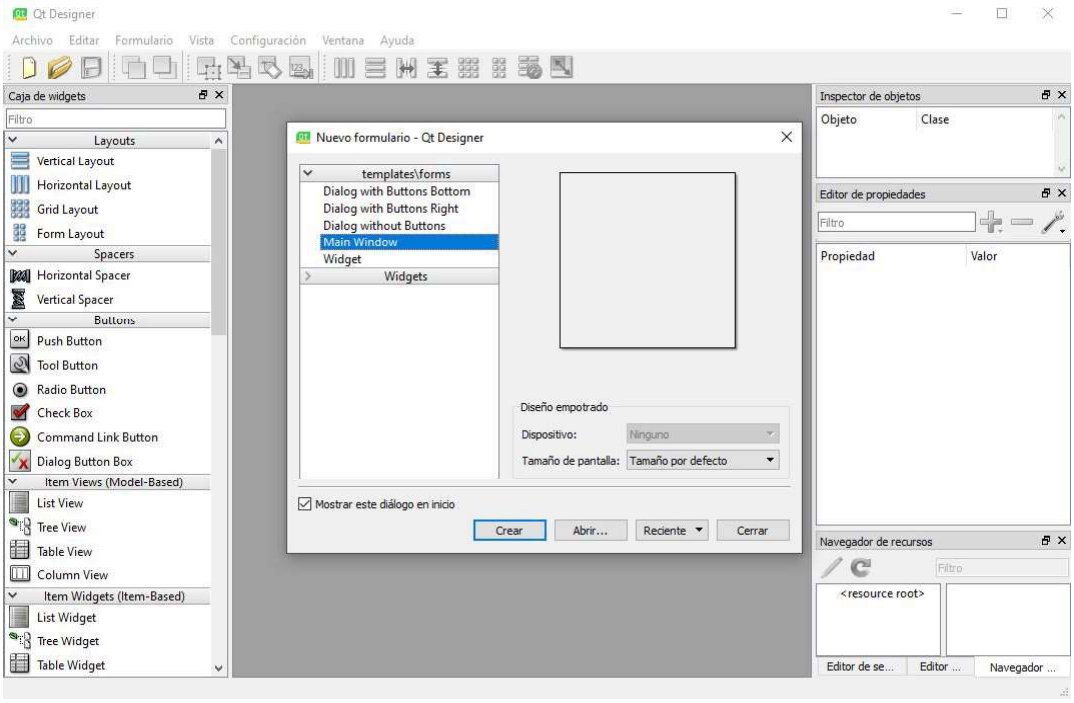

Figura 1.11: Entorno de Qt Designer. Fuente [35]

1.3.7.4. PhysioNet

PhysioNet es un recurso disponible en la web cuyo objetivo es estimular y apoyar a la investigación en el área que estudia las señales biomédicas y fisiológicas. Tiene tres componentes interdependientes [36]:

- **PhysioBank** es un repositorio grande de grabaciones digitales de señales fisiológicas, series de tiempo y datos relacionados para uso de la comunidad de investigación biomédica. Incluye colecciones de señales cardiopulmonares, neuronales y de otras señales biomédicas de pacientes sanos y pacientes con una variedad de afecciones que incluyen muerte cardíaca súbita, insuficiencia cardíaca congestiva, epilepsia, apnea del sueño, etc. Estas colecciones implican desarrollos y aportes de los miembros de la comunidad de investigación [36].

- **PhysioToolkit** es una herramienta de software de código abierto para el análisis, procesamiento y simulación de señales fisiológicas [36].

- **PhysioNetWorks** es un laboratorio virtual donde pueden trabajar en conjunto los profesionales de la salud de cualquier parte del mundo para la creación, evaluación, mejoramiento, documentación y preparación de nuevas bases de datos y software para su publicación en PhysioNet [36].

El sitio web de PhysioNet fue creado como mecanismo para la difusión e intercambio libre de señales biomédicas y software de código abierto. Se caracteriza por proporcionar acceso digital gratuito a los datos de PhysioBank, software PhysioToolkit, y a espacios de trabajo seguros para el desarrollo colaborativo de nuevos datos y software dentro de PhysioNetWorks [36].

PhysioBank actualmente contiene más de 36,000 grabaciones de señales fisiológicas y series cronológicas digitalizadas. Estas grabaciones están organizadas en más de 50 bases de datos. Todas estas bases de datos están disponibles gratuitamente en la web de PhysioNet. Al hablar de una gran cantidad de datos de alrededor de 4 TeraBytes, se complica la tarea a la hora de buscar algún registro entre toda esa información disponible. Para ello PhysioNet ha incorporado la herramienta PhysioBank ATM [36].

PhysioBank ATM

PhysioBank ATM es una herramienta en la web para explorar PhysioBank. Es una herramienta versátil para una visión general y rápida de las bases de datos disponibles. Por ejemplo, PhysioBank ATM actualmente permite obtener señales digitalizadas de 100000 muestras de un minuto de duración en el formato de salida de nuestra elección. La caja de herramientas de PhysioBank ATM permite al usuario mostrar de una forma más organizada las formas de onda, series de tiempo de intervalo RR, histogramas y anotaciones en archivos de texto, CSV, EDF o .mat (para su utilización con MATLAB u Octave) [36]. En la Figura1.12 se observa su entrono en la web.

Figura 1.12: Herramienta web PhysioNet ATM. Fuente [37]

2. METODOLOGÍA

Para el desarrollo de este trabajo se elaboró un modelo de capas en el que constan los componentes implementados de la aplicación, con el objetivo de que si se realiza algún cambio en un componente de una capa, este solo afecte a la capa en cuestión sin tener que modificar el código fuente de otra capa del modelo.
En el presente capítulo se describen los elementos que conforman el software ECG System, la función de cada uno de ellos y su implementación en Python. En la sección 2.1 se hablará del modelo de capas de nuestro software en el que se detalla todos los componentes de cada capa del software. En la sección 2.2 se hablará de la implementación del proyecto en Python.

2.1. ARQUITECTURA DEL SISTEMA DE SIMULACIÓN ECG

La arquitectura del software desarrollado está compuesta por la capa Lógica del Software y capa Interfaz Gráfica de Usuario (GUI). La Capa Lógica del Software realiza las tareas de procesamiento y carga de bases de datos. La capa GUI se encarga de mostrar en las respectivas interfaces los procesos acordes a cada rol que se tome en la simulación. El funcionamiento y la estructura de capas de la aplicación se lo muestra en el diagrama de la Figura 2.1.

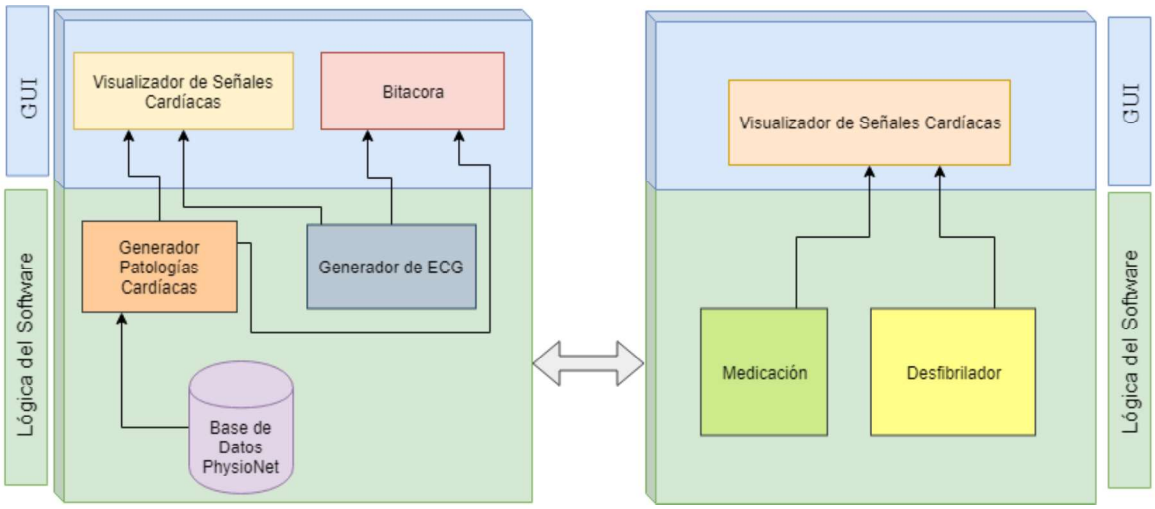

Figura 2.1: Diagrama de la Arquitectura de la Aplicación ECG System

El funcionamiento del simulador se describe en los siguientes pasos:

- Inicialmente se ejecuta la interfaz del estudiante, la cual se mantiene a la espera de peticiones de conexión.
- Posteriormente se ejecuta la interfaz del profesor que tiene la información de la dirección IP y puerto al cual se conectará.
- Las dos interfaces establecen una conexión bidireccional entre las dos interfaces.

- Una vez conectados se podrá crear y enviar señales del corazón a la interfaz del estudiante.
- Permitirá al instructor observar las acciones del estudiante en una bitácora.
- La interfaz del estudiante dispone de un módulo que permite simular desfibrilación, cardioversión y medicación.

2.1.1. LÓGICA DEL SOFTWARE

Esta capa se encarga de la funcionalidad del software, recibe las peticiones de la capa Interfaz de Usuario, esta capa contiene la mayor cantidad de código de programación.

2.1.1.1. Generador de la Señal ECG

El modelo aplicado en este trabajo se basa en la descripción matemática de cada componente que conforma la señal ECG. Con la implementación de este modelo nos permitirá ajustar los valores en la GUI del profesor con el fin de obtener varias señales cardíacas. Los componentes principales de la señal ECG se los lista a continuación:

- Amplitud de la onda P (A_P) y duración de la onda P (T_P).
- Intervalo PR (t_{PR}).
- Amplitud del complejo QRS e intervalo QRS.
- Longitud del segmento ST (t_{ST}).
- Amplitud de la onda T (A_T) y duración la onda T (T_T).
- Intervalo del segmento TP (t_{TP}).

Las ondas P y T se las pueden consideraran como ondas seno periódicas y las pude representar por las ecuaciones 2.1 y 2.2.
Ecuación de la onda P:

$$f_P(t) = A_P \cdot \sin\left(\frac{\pi}{T_P} \cdot t\right), \qquad Si\ 0 \leqslant t \leqslant T_P \tag{2.1}$$

Ecuación de la onda T:

$$f_T(t) = A_T \cdot \sin\left(\frac{\pi}{T_T} \cdot t\right), \qquad Si\ 0 \leqslant t \leqslant T_T \tag{2.2}$$

Las ondas Q, R y S pueden considerarse como ondas triangulares periódicas. En este trabajo para facilitar el modelado un enfoque es usar ecuaciones lineales que representen las rectas ascendentes y descendentes del complejo QRS. Con la ecuación 2.3 podemos representar cada recta del complejo QRS que en el contexto de la Geometría Analítica consta de 2 rectas de pendientes positivas y 2 rectas pendientes negativas.

$$y = mx + b \tag{2.3}$$

Donde, el valor de la pendiente es:

$$m = \frac{y_2 - y_1}{x_2 - x_1} \tag{2.4}$$

Y el punto de intercepción con el eje vertical es:

$$b = y_1 - m \cdot x_1 \tag{2.5}$$

Adaptándolas estas ecuaciones para nuestro propósito, podemos representar el complejo QRS mediante la ecuación 2.6:

$$f_{QRS}(t) = \begin{cases} m_Q t_Q + c_Q, & Si\ t_{PR} \leq t \leq t_Q; \\ m_{R1} t_{R1} + c_{R1}, & Si\ t_Q \leq t \leq t_{R1}; \\ m_{R2} t_{R2} + c_{R2}, & Si\ t_{R1} \leq t \leq t_{R2}; \\ m_S t_S + c_S, & Si\ t_{R2} \leq t \leq t_S \end{cases} \tag{2.6}$$

Cabe recalcar que los tiempos t_{PR} y t_s son los tiempos donde inicia y termina el complejo QRS respectivamente. t_Q es el tiempo donde termina la primera recta descendente, t_{R1} el tiempo donde termina la segunda recta ascendente, t_{R2} el tiempo donde termina la tercera recta descendente.

Finalmente, el resultado de combinar las señales P, QRS y T obtenemos un pulso básico ECG de acuerdo a la ecuación 2.7. Con este modelo un pulso básico ECG puede ser construido, se muestra gráficamente el resultado en la Figura 2.2.

$$f_{ECG}(t) = \begin{cases} A_P \cdot \sin\left(\frac{\pi}{T_P} \cdot t\right), & Si\ 0 \leq t \leq T_P; \\ 0, & Si\ T_P \leq t \leq t_{PR}; \\ m_Q t_Q + c_Q, & Si\ t_{PR} \leq t \leq t_Q; \\ m_{R1} t_{R1} + c_{R1}, & Si\ t_Q \leq t \leq t_{R1}; \\ m_{R2} t_{R2} + c_{R2}, & Si\ t_{R1} \leq t \leq t_{R2}; \\ m_S t_S + c_S, & Si\ t_{R2} \leq t \leq t_S; \\ 0, & Si\ t_S \leq t \leq t_{ST}; \\ A_T \cdot \sin\left(\frac{\pi}{T_T} \cdot t - t_{ST}\right), & Si\ t_{ST} \leq t \leq t_T; \\ 0, & Si\ t_T \leq t \leq t_{ECG} \end{cases} \tag{2.7}$$

Ahora, sabemos que una señal ECG real es una señal periódica y tiene una frecuencia fundamental que está determinada por el ritmo cardíaco. El período en segundos de un pulso simple se determina mediante la ecuación 2.8.

$$T_{Latido} = \frac{60}{BPM} \tag{2.8}$$

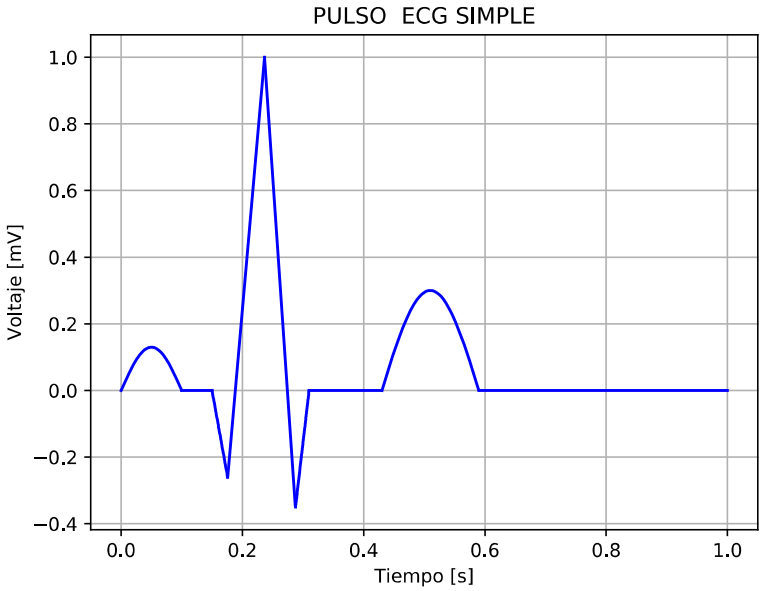

Figura 2.2: Forma de Onda de un Pulso ECG Simulado

Donde el parámetro BPM representa la frecuencia cardiaca en latidos por minuto. En este trabajo para conseguir que el pulso básico ECG se convierta en una señal periódica se ha utilizado el algoritmo Karplus-Strong.

Algoritmo Karplus-Strong

Es un modelo físico realmente simple pero muy efectivo en la generación de una variedad de sonidos de cuerda pulsada, llamado así por sus inventores principales, Kevin Karplus y Alex Strong. Los componentes principales que hacen que funcione el algoritmo Karplus-Strong son el mecanismo de retroalimentación y la operación de promediado. El mecanismo de retroalimentación del buffer en anillo modela el medio en el que la energía viaja de un lado a otro. La longitud del buffer de anillo determina la frecuencia fundamental del sonido resultante. El promedio actúa como un filtro paso bajo en la señal debido a que está en el camino de la retroalimentación, esto tiene el efecto de atenuar gradualmente los armónicos más altos mientras se mantienen los más bajos.El resultado es una forma de onda periódica [38].

El funcionamiento del Algoritmo Karplus-Strong para crear una onda periódica se basa en la ecuación 2.9 , se observa en el diagrama de la Figura 2.3.

$$y[n] = x[n] - \alpha y[n - M] \tag{2.9}$$

En donde:

- M es el tamaño en muestras de la señal que se desea hacer periódica.
- x_n es la señal de entrada para el intervalo $0 \leq n < M$.
- y_n es la señal periódica.
- α es la atenuación, en nuestro caso su valor es igual a 1.

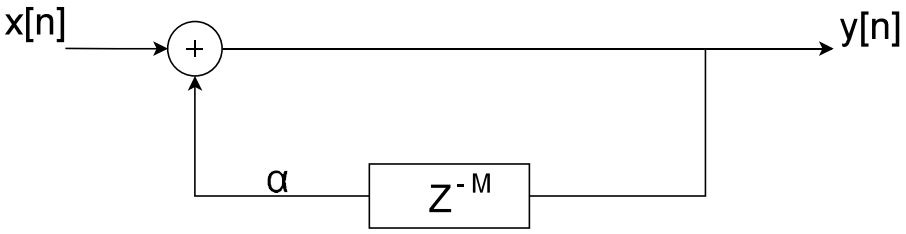

Figura 2.3: Esquema Básico del Algoritmo Karplus-Strong

Aplicando el Algoritmo Karplus-Strong a la señal ECG simple obtenemos una señal periódica como el ejemplo de la Figura 2.4, en donde, se observa una señal ECG periódica de 10 segundos.

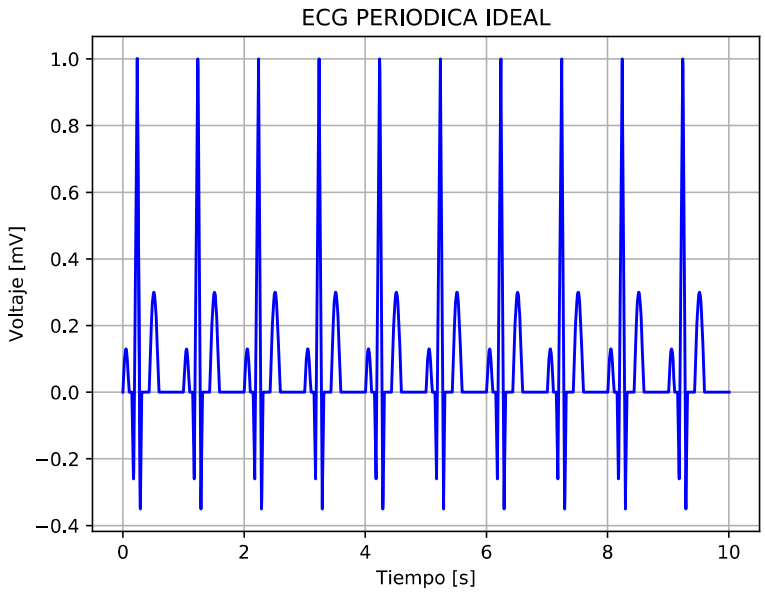

Figura 2.4: Señal ECG Periódica de Duración t = 10 segundos

Para convertir a nuestra señal ECG acorde a la vida real es necesario analizar que en la práctica existen muchos factores que afectan la forma de onda de la señal ECG. La señal de ECG es susceptible a varios tipos de ruidos, estos pueden proceder del propio sistema o de factores externos [39]. Las principales fuentes de interferencia y ruido que afectan a un equipo electrocardiograma y que se considerarán en este trabajo son: la interferencia eléctrica y el ruido interno o inherente de los equipo electrónicos.

La red eléctrica es la principal fuente de interferencia, se genera como consecuencia del acoplamiento capacitivo, conducción o inducción magnética. Tiene naturaleza periódica y para nuestro caso el valor de la frecuencia es igual a 60 Hz y se lo representa con la ecuación 2.10. Dónde $f = 60Hz$.

$$f_{RedElctrica}(t) = A \cdot \sin(2\pi \cdot f \cdot t) \tag{2.10}$$

Otro tipo de ruido encontrado en las señales ECG es el ruido inherente que generan los propios componentes electrónicos de un circuito o sistema. El ruido inherente es de naturaleza aleatoria y lo representamos como $\sigma(t)$.

Por lo tanto, podemos expresar nuestra señal ECG ruidosa mediante la ecuación 2.11.

$$f_{N_{ECG}}(t) = f_{RedElctrica}(t) + \sigma(t) + f_{ECG}(t) \tag{2.11}$$

En este punto, la eliminación de ruidos de la señal ECG se convierte en una parte vital en el procesamiento de la señal y esto juega un papel importante en la detección y diagnóstico de enfermedades del corazón. Para minimizar los efectos del ruido aleatorio y la separación de la interferencia de la red eléctrica aplicamos un filtro pasabajos con el objetivo de eliminar las componentes de alta frecuencia. Se implementó un filtro digital pasabajos IIR de tercer orden tipo Butterworth de manera que se produzca una respuesta lo más plana que sea posible en la banda de paso.

Mediante un análisis del espectral verificamos que las componentes significativas de la señal ECG se encuentran hasta los 20 Hz como se puede observar en la Figura 2.5. Adicionalmente verificamos que el 99 % de la energía de la señal ECG simulada se concentra aproximadamente hasta los 20 Hz.

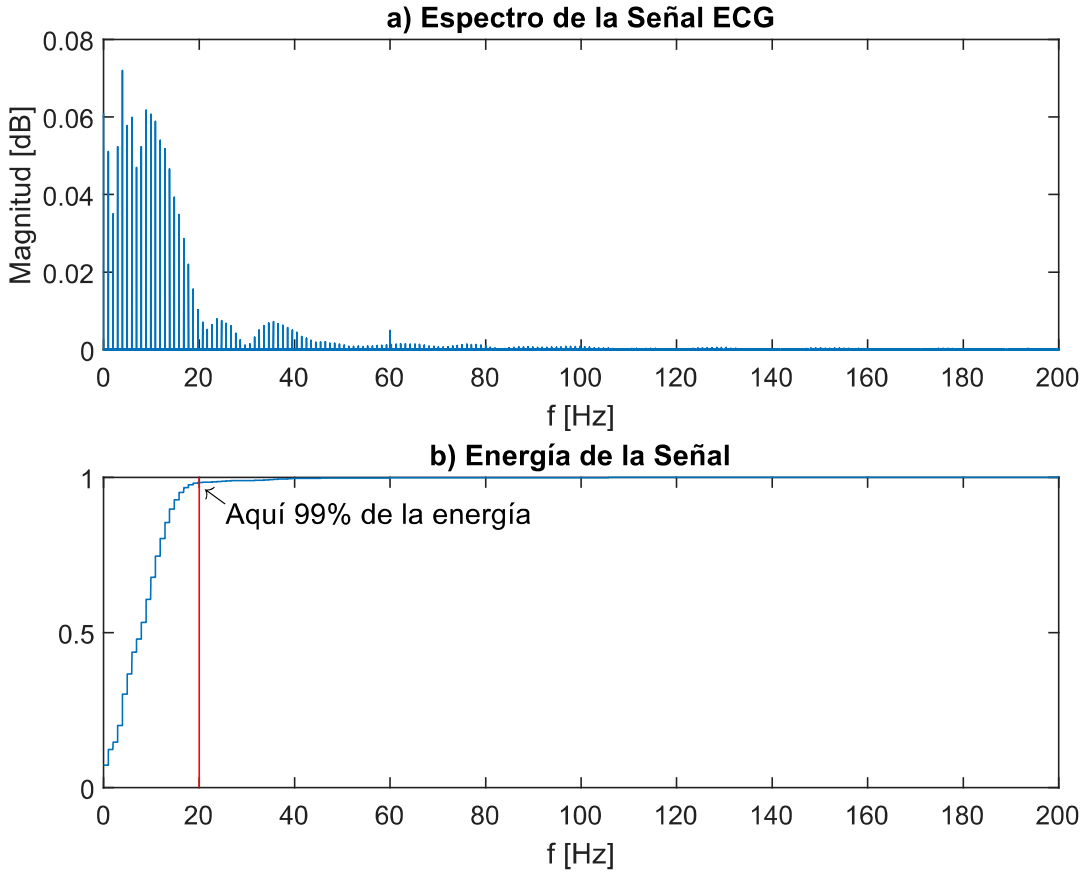

Figura 2.5: Espectro y Energía de la Señal ECG Simulada

Por lo tanto, para remover el ruido de la señal ECG simulada establecemos la frecuencia de corte igual a 36 Hz, debido a que las componentes representativas de nuestra señal se encuentran por debajo de los 20 Hz y la componente de la red eléctrica deseamos filtrar esta en 60 Hz. En la Figura 2.6a se observa la señal ECG ruidosa y en la Figura 2.6b se observa la señal ECG restaurada luego de aplicar el filtro pasabajos.

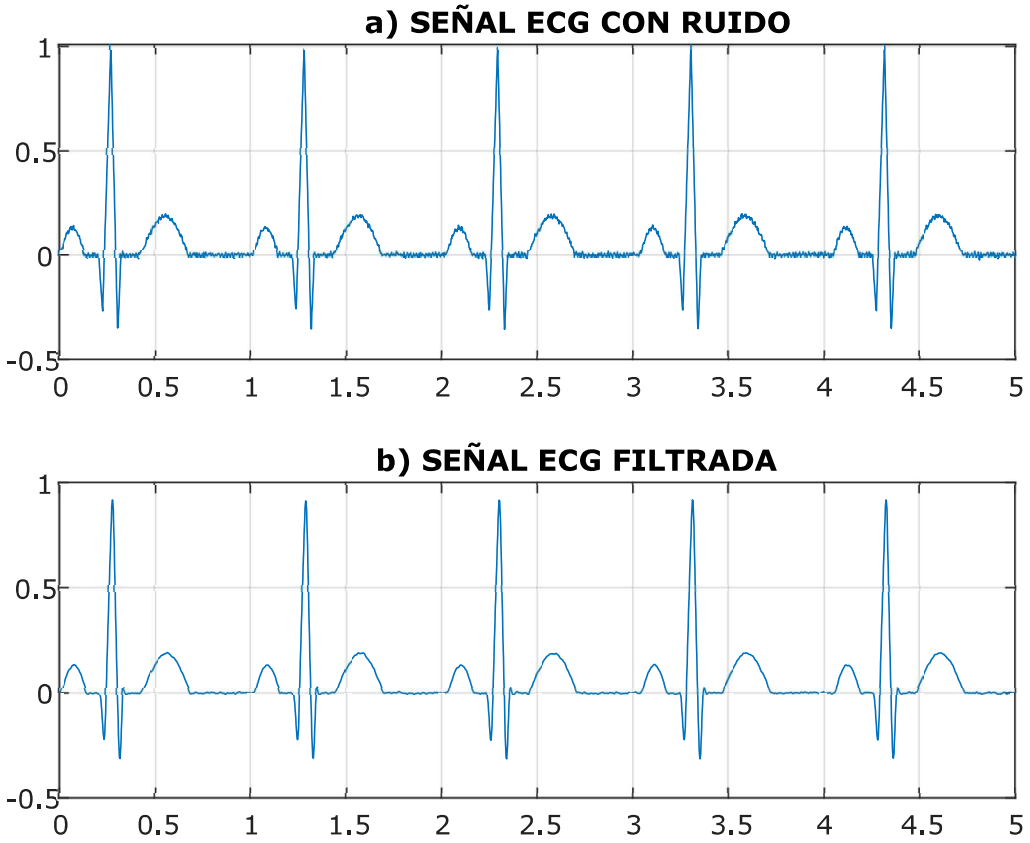

Figura 2.6: Proceso de Filtrado de la señal ECG. a) Señal ECG con Ruido. b) Señal ECG Filtrada

2.1.1.2. Generador de las Señales Patológicas

Este módulo de la aplicación nos permitirá cargar registros de diferentes patologías del corazón desde una fuente confiable. Para este trabajo se ha utilizado la base de datos del sitio web de PhysioNet. Las bases se encuentran disponibles en `https://archive.physionet.org/cgi-bin/atm/ATM`, las señales cardiológicas seleccionadas para este trabajo fueron las siguientes:

- MIT-BIH Arrhythmia Database.
- MIT-BIH Atrial Fibrillation Database.
- MIT-BIH Malignant Ventricular Ectopy Database.
- MIT-BIH Supraventricular Arrhythmia Database.
- CU Ventricular Tachyarrhythmia Database.

- Spontaneous Ventricular Tachyarrhythmia.

MIT-BIH Arrhythmia Database
Esta base de datos contiene 48 registros de media hora obtenidos de 47 sujetos estudiados por el Laboratorio del BIH entre 1975 y 1979. Las grabaciones están digitalizadas a 360 muestras por segundo con una resolución de 11 bits en un rango de 10 milivoltios [40].

MIT-BIH Atrial Fibrillation Database
Las grabaciones individuales tienen una duración de 10 horas cada una y contienen dos señales de ECG, cada una de ellas muestreada a 250 muestras por segundo con una resolución de 12 bits en un rango de $\pm$ 10 milivoltios [41].

MIT-BIH Malignant Ventricular Ectopy Database
Esta base de datos incluye 22 registros de ECG de media hora de sujetos que experimentaron episodios de taquicardia ventricular sostenida, aleteo ventricular y fibrilación ventricular [42].

MIT-BIH Supraventricular Arrhythmia Database
Incluye 78 registros de ECG de media hora elegidos para complementar los registros de las arritmias supraventriculares en la base de datos de arritmias MIT-BIH [43].

CU Ventricular Tachyarrhythmia Database
Esta base de datos incluye 35 registros de ECG de 8 minutos de sujetos que experimentaron episodios de taquicardia ventricular sostenida, aleteo ventricular y fibrilación ventricular [44].

Spontaneous Ventricular Tachyarrhythmia
Esta base de datos contiene 135 pares de series de tiempo de intervalo RR, registradas por desfibriladores automáticos implantados en 78 sujetos. Cada serie contiene entre 986 y 1022 intervalos RR. Cada par incluye un episodio espontáneo de taquicardia ventricular o fibrilación ventricular y una muestra del ritmo sinusal [45].
Varios de estos registros son de larga duración y que podrían generar problemas de memoria al procesar un conjunto grande de datos. Para nuestro propósito requerimos obtener registros de corta duración para ello utilizamos la herramienta de PhysioBank ATM, donde tenemos la posibilidad de descargar los registros a intervalos de tiempo pequeños. Dichos tiempos van desde 10 segundos, 1 minuto, 1 hora, etc.

El tipo de archivo que se descarga de la página web de las señales patológicas son archivos .mat que son cargadas a Matlab para realizar el procesamiento correspondiente. Este procesamiento consta en extraer la señal de la estructura de datos obtenida desde la pagina y guardar en un archivo .txt para leerla en Python. Parte del procesamiento realizado en MatLab es convertir la señal digitalizada obtenida desde PhysioNet a una señal con las unidades físicas (segundos y milivoltios).

Observamos en la Figura 2.7a la gráfica de la señal correspondiente a una Arritmia Sinusal descargada desde PhysioNet. En esta gráfica observamos que el eje horizontal corresponde a las muestras de la señal y el eje vertical corresponde a los valores digitalizados de la

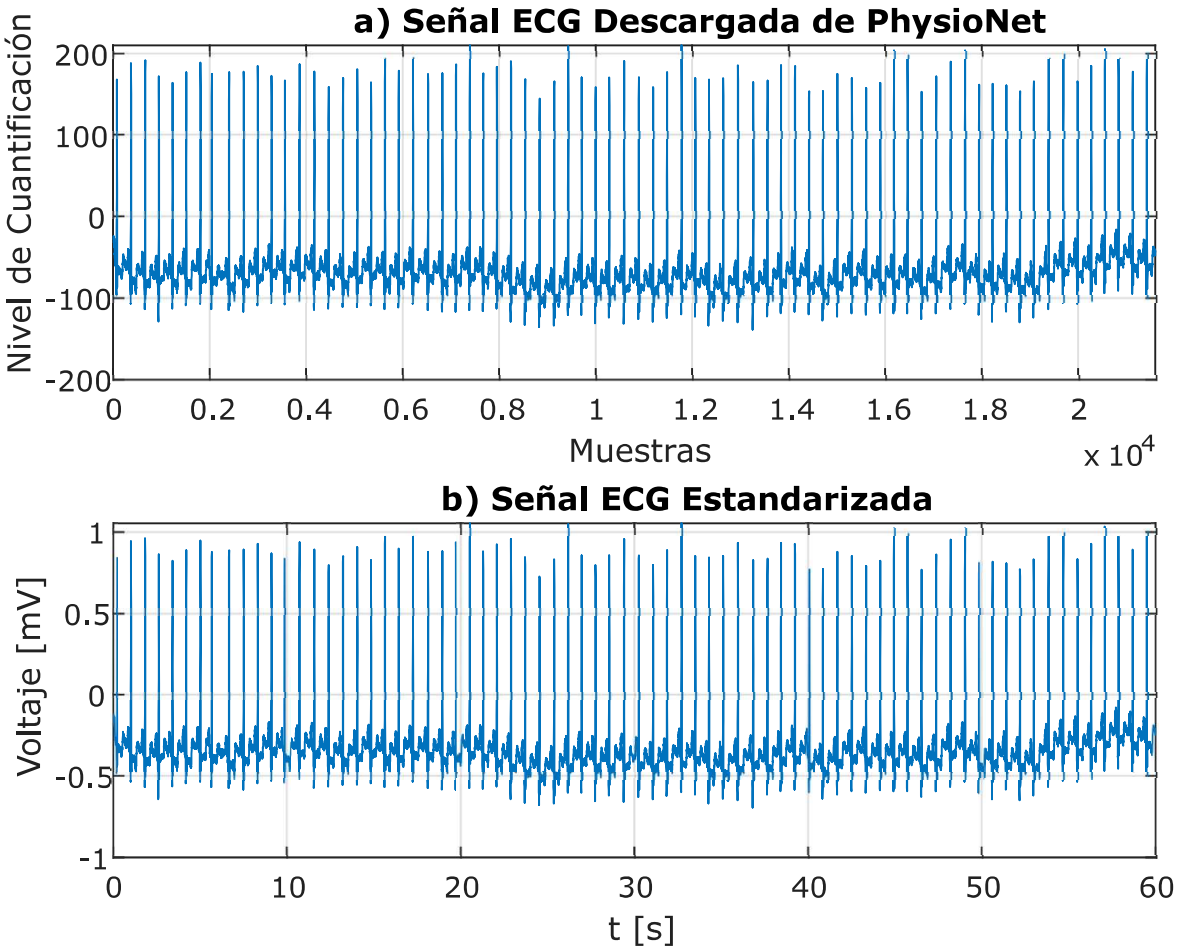

Figura 2.7: Procedimiento Conversión de unidades. a) Señal ECG Muestreada Obtenida de PhysioNet. b) Señal ECG en unidades Estandarizadas

señal. Para poder convertir las señales descargadas de PhysioNet a las unidades adecuadas en el que el eje horizontal represente el tiempo en segundos y el eje vertical represente la amplitud de la señal en milivoltios como se muestra en la Figura 2.7b utilizamos el archivo .info que se encuentra disponible junto a cada registro de la señal que se selecciona en la herramienta PhysioBank al momento de descargar determinada señal. El archivo .info contiene información detallada como: el nombre de las señal, el número de muestras, duración de la señal, frecuencia de muestreo, tipo de derivación de ECG y el número de derivaciones de ECG, ganancia, base de muestreo.

En cada archivo .info nos indica cómo convertir a unidades físicas la respectiva señal. Por ejemplo, para convertir la señal correspondiente a la Arritmia Sinusal, el archivo nos indica que debemos restar a la señal muestreada la base de muestreo y todo esto dividirlo para el valor de la ganancia. Esto lo podemos representar mediante la Ecuación 2.12 [36].

$$Signal = \frac{ArritmiaSinusal - Base}{Ganancia} \tag{2.12}$$

En donde, $Signal$ representa la señal cuyas unidades de la amplitud están en milivoltios, la variable $ArritmiaSinusal$ en este caso representa la señal descargada de Physionet,

$Base$ es igual a 1024 y la $Ganancia$ es igual a 200. Estos parámetros se los encuentra en el archivo .info de la señal que fue descargada y difieren para cada tipo de señal.
Aplicando este procedimiento a cada señal correspondiente a las patologías seleccionadas las transformamos a las unidades respectivas y las guardamos en archivos .txt para exportarlas a Python. En la Figura 2.8 observamos las señales patológicas seleccionadas luego de realizar el procedimiento de conversión de unidades.

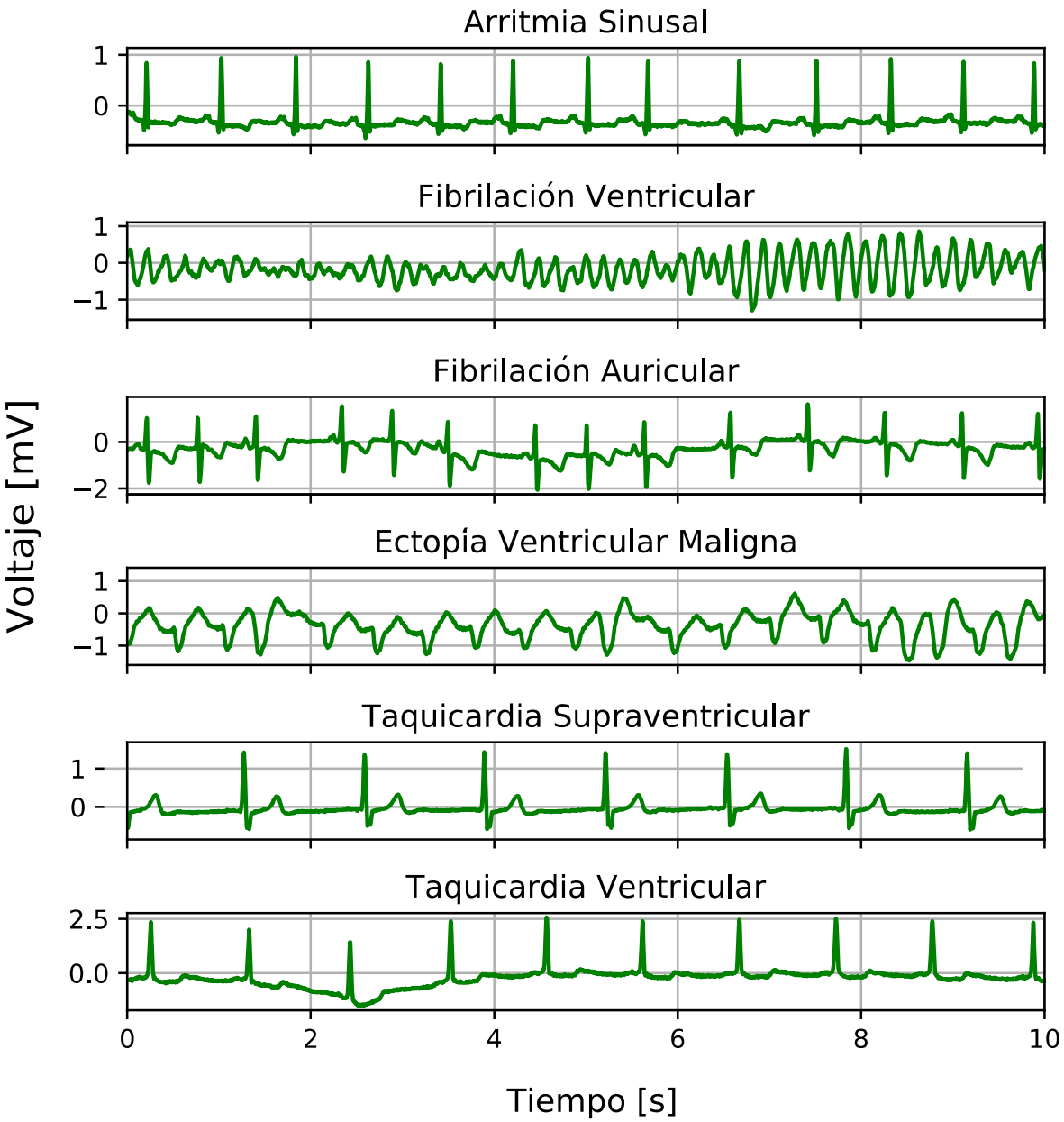

Figura 2.8: Forma de Onda de las Patologías Implementadas en la Aplicación

2.1.1.3. Módulo Medicación

Esta sección de la aplicación ECG System permite al estudiante simular la aplicación de fármacos para tratar determinada anomalía en el ritmo cardíaco de manera que se pueda reanudar el ritmo cardíaco normal. En nuestra aplicación se ha incluido los principales

fármacos antiarrítmicos más utilizados: Epinefrina, Lidocaina, Amiodarona, Quinidina. Para implementar sus efectos en nuestro simulador consideramos los siguientes parámetros en cada fármaco:

- Dosis necesaria del fármaco.
- Efectos secundarios del fármaco.
- Sobredosis y contraindicaciones.
- Patología a la cual se debe aplicar el fármaco.

El estudiante elegirá el medicamento que considera más adecuado y seguro para cada caso. La tabla 2.1 muestra un resumen de acuerdo con lo visto en la sección 1.3.5 de los tipos de fármacos para determinada patología, la dosis efectiva y los efectos secundarios.

Tabla 2.1: Resumen de los Fármacos Implementados

FÁRMACO	PATOLOGÍAS	DOSIS	REACCIONES ADVERSAS
Epinefrina	Bradicardia,Taquicardia Supraventricular, Taquicardia Ventricular, Arritmia Sinusal	1-5 mg	Acelera el ritmo cardíaco
Amiodarona	Arritmia Sinusal, Taquicardia Supraventricular, Fibrilación Ventricular, Taquicardia Ventricular.	300-350 mg	Empeora las arritmias ventriculares
Quinidina	Fibrilación Auricular, Taquicardia Ventricular, Arritmia Ventricular Maligna	100-600 mg	Mareos, náuseas
Lidocaína	Fibrilación Ventricular, Taquicardia Ventricular	100-300 mg	Náuseas, mareos, convulsiones

2.1.1.4. Módulo Desfibrilador

La aplicación ECG System dispone de una sección de desfibrilación que se encuentra en la interfaz del estudiante y permite simular los procedimientos de desfibrilación y cardioversion eléctrica para los casos de emergencia que sea necesaria su aplicación.
Para el proceso de desfibrilación el estudiante tiene la posibilidad de ajustar la cantidad de energía a aplicar y visualizar a través de un StatusBar ubicado en la parte inferior de su interfaz el progreso de carga que necesita el desfibrilador para que pueda ser aplicada la energía seleccionada. Por ejemplo, cuando el profesor envía al alumno una fibrilación ventricular, el estudiante podrá aplicar una desfibrilación como parte de la terapia que se debe realizar al paciente. La desfibrilación será eficaz si la cantidad de energía es la adecuada para el tipo de arritmia. Puede llegar a ser necesario aplicar incrementos graduales de energía subsecuentes si el ritmo no se restablece.

Para la cardioversión eléctrica se incluye el modo sincrónico en donde luego de ser activado un marcador QRS aparecerá en la pantalla y permitirá al estudiante aplicar la descarga eléctrica de forma sincrónica con la onda R. Por ejemplo, cuando el profesor envía al estudiante una señal de fibrilación auricular este activará el modo sincrónico y podrá aplicar la descarga de forma sincronizada con la onda R y así restablecer el ritmo normal. Será necesario repetir el procedimiento para que se restablezca el ritmo cardíaco normal si la descarga no fue aplicada sobre la onda R. La aplicación podrá reconocer a las patologías

Tabla 2.2: Resumen de Ritmos Viables a Desfibrilación y Cardioversión

DESFIBRILACIÓN	CARDIOVERSIÓN
Fibrilación Ventricular Taquicardia Ventricular Ectopia Ventricular Maligna	Taquicardia Arritmia Sinusal Fibrilación Auricular Taquicardia Supraventricular

que sean viables a la aplicación de la cardioversion, de no ser así no se podrá aplicar este tratamiento.

En la tabla 2.2 se resume de acuerdo a lo explicado en la sección 1.3.6 los ritmos cardiológicos a los que se puede aplicar desfibrilación o cardioversión.
Tanto para la desfibrilación como para la cardioversión eléctrica se tomará en cuenta las contraindicaciones y complicaciones, es decir, si no es eficaz el tratamiento se puede agravar o dar paso a la aparición de otro tipo de arritmias.

2.1.1.5. Comunicación entre las Interfaces

En este punto necesitamos que los equipos en los se ejecutan las interfaces del profesor y del estudiante se comuniquen entre sí. Para lograr este objetivo utilizaremos un modelo Cliente/Servidor a través de una red TCP/IP. La forma más común de trabajar con TCP/IP directamente en código es con un socket API. Un socket es un objeto similar a un archivo que representa un único punto de conectividad de red para el sistema. Cada socket está definido por la dirección IP de la máquina, el puerto en el que escucha, y el protocolo que utiliza. Configuramos un socket en cada extremo y permitimos que el Cliente interactúe con el Servidor. El Servidor y el Cliente deben tener asociado el mismo puerto para comunicarse.

El servidor se configurará en una red de área local y usando una dirección IPv4 privada. Para el puerto se debe tener cuidado al elegir uno que no esté en uso. Por ejemplo los puertos menores a 1024 que se asignan a servicios comunes y que se los denominan puertos conocidos. Por lo que deberíamos escoger un puerto entre 1024 y 65535.

Los servidores básicamente hacen lo siguiente:

- Abren un socket.
- Enlazan a una dirección y puerto.
- Escuchan las conexiones entrantes.
- Aceptan conexiones.
- Leen y envian datos.

El script del lado del cliente simplemente intentará acceder al socket del servidor creado, en la dirección IP y el puerto especificado. Una vez que se conecta, empezará a transmitir información. En este trabajo hemos establecido el Servidor a la interfaz del estudiante y el Cliente a la interfaz del profesor.

2.1.2. INTERFAZ GRÁFICA

Esta capa de acuerdo a nuestro modelo recibe información del usuario y la muestra muestra directamente a través de los componentes visuales disponibles. El sistema de simulación ECG proporciona dos interfaces: una interfaz para que se utiliza para el rol de profesor Figura 2.9 y otra interfaz que se usa cuando toma el rol de estudiante Figura 2.10.

Figura 2.9: Mockup de la Interfaz Gráfica del Profesor. Fuente Propia

2.1.2.1. Componentes de la Interfaz Gráfica del Profesor

La ventana del profesor consta de 5 paneles principales que describimos a continuación:

- **Ajustador de Frecuencia Cardíaca:** dispone de un slider que permite variar la frecuencia cardíaca en un rango de 20 a 200 latidos por minuto y cuyo valor está inicializado en 60 latidos por minuto.
- **Panel de Ajustes de Parámetros Básicos:** permiten al profesor crear señales ECG personalizadas modificando parámetros como: la magnitud de Onda P, duración de la onda P, Intervalo PR, amplitud de la onda QRS, duración del complejo QRS, amplitud de la onda T y duración de la onda T.
- **Panel de Visualización de Señales:** simula la pantalla de un electrocardiograma. Muestra las señales cardíacas en el dominio del tiempo. Se encuentra disponible tanto para la ventana del profesor como para la del estudiante y fue implementada con la herramienta PyQtGraph que ofrece un mejor desempeño para la visualización de datos en forma de animación que los que se obtenía con la librería Matplotlib.
- **Panel de Patologías Cardíacas:** Contiene las seis bases de datos correspondientes a las señales patológicas descargadas de PhysioNet.

- **La Bitácora:** es un elemento de la interfaz del profesor que proporciona los registros de manera cronológica de los eventos que se desarrollan en la simulación. Dispone de cuatro columnas que corresponden a la hora, frecuencia cardíaca de la señal actual, nombre del ritmo cardiaco y una columna que muestra detalles de la simulación. El reporte se puede exportar como un archivo csv.

2.1.2.2. Componentes de la Interfaz Gráfica del Estudiante

La interfaz gráfica del estudiante dispone de una pantalla al igual a la ventana del profesor que representa la pantalla del electrocardiograma. Esta pantalla muestra de forma animada las señales cardíacas establecidas por el médico docente o por determinada acción realizada por el estudiante que modifique el ritmo cardiaco actual.

Figura 2.10: Mockup de la Interfaz Gráfica del Estudiante. Fuente Propia

En la parte lateral izquierda de la pantalla dispone de un cuadro que muestra el ritmo cardíaco de la señal actual. También incluye un StatusBar en la parte inferior de la ventana que se utiliza para mostrar al usuario información de estado de los procesos que se está llevando a cabo.

2.2. IMPLEMENTACIÓN DEL SOFTWARE EN PYTHON

Para realizar la programación de la aplicación se utilizó el IDE Spyder que viene incorporado en Ananconda Distribution el cual facilitó la administración de los paquetes. Anaconda viene con mas de 1500 paquetes entre los cuales fueron necesarios para este proyectos los siguientes:

- NumPy es el paquete fundamental para el cálculo numérico. Define la matriz numérica y los tipos de matriz y las operaciones básicas en ellos.
- SciPy es una colección de algoritmos numéricos y cajas de herramientas específicas de dominio, que incluye procesamiento de señales, optimización, estadísticas, etc.
- Pandas es una librería destinada para la manipulación y análisis de datos.
- Sockets nos proporciona los servicios de red básicos, ejemplo nos permite crear un modelo Servidor/Cliente.
- Thread permite trabajar con múltiples subprocesos que comparten su espacio de datos global.

2.2.1. Implementación de la Señal ECG

Para implementar la señal ECG simulada nos basamos en el modelo visto en la sección 2.1.1.1. A partir de este modelo se lo implementó en Python. En primer lugar adquirimos los parámetros básicos de la señal ECG que ingresa el profesor desde su interfaz gráfica. Los parámetros básicos son: la frecuencia cardíaca, amplitud e intervalo de la onda P, intervalo del segmento PR, amplitud e intervalo del complejo QRS, amplitud e intervalo de la onda T.

```
PeriodoLat = 60/ bmp            #Frecuencia Cardica
#tiempos de constantes
Pint = self.interP              #intervalo de la onda P
PRint = self.interPR            #intervalo PR
STseg = 0.12*PeriodoLat         #intervalo segmento PR
QRSint = self.interQRS          #intervalo ndel complejo QRS
q1 = PRint + 0.16*QRSint        #tiempo de la recta Q
r1 = PRint + 0.54*QRSint        #tiempo de la recta R creciente
r2 = PRint + 0.87*QRSint        #tiempo de la recta R decreciente
s = PRint + 1*QRSint            #tiempo de la recta s
st = PRint + QRSint + STseg     #longitud del segmento ST
t2 = st + self.interT           #tiempo de la la onda t
#constantes de amplitud
pPeak = self.amP                #Amplitud onda P
rPeak = self.ampQRS             #Amplitud onda R
qPeak = 0.26 * rPeak            #Amplitud onda Q
sPeak = 0.35 * rPeak            #Amplitud onda S
tPeak = self.amT                #Amplitud onda T
```

Segmento de código 1: Parámetros básicos para simular la señal ECG

El Segmento de Código 1 muestra como se realiza la asignación de los valores ingresados de la interfaz gráfica del profesor. Observamos que las variables, STseg, q1, r1 y r2 van acompañados de ciertos factores numéricos. Estos factores sirven para relacionar los parámetros ingresados a través de la interfaz gráfica del profesor con las variables creadas para generar el pulso básico ECG. Estos factores se calculan utilizando los valores normales de la señal ECG revisados en la sección 1.3.3.

Para el segmento ST:

$$ST = \frac{T_{Latido}}{t_{ST}} = \frac{0{,}12}{1{,}00} = 0{,}12 \tag{2.13}$$

Tomando en cuenta que $q1$ es el tiempo cuando la onda Q está en su amplitud máxima.

$$q1 = \frac{t_Q}{t_{QRS}} = \frac{0{,}02}{0{,}12} = 0{,}16 \tag{2.14}$$

De igual forma, $r1$ es el tiempo cuando la onda R está en su amplitud máxima.

$$r1 = \frac{t_Q}{t_{QRS}} = \frac{0{,}04 + 0{,}025}{0{,}12} = 0{,}54 \tag{2.15}$$

Por ultimo, con el mismo criterio determinamos el factor para la variable $r2$:

$$r2 = \frac{t_Q}{t_{QRS}} = \frac{0{,}09 + 0{,}015}{0{,}12} = 0{,}87 \tag{2.16}$$

Luego procedemos a crear separadamente cada onda con sus respectivos vectores de tiempo. El Segmento de Código 2 muestra como se simuló el complejo QRS a partir de la composición de las 4 rectas ascendentes y descendentes que conforman la señal ECG básica.

```
#constantes de líneas QRS
#y = mx+c
#m = (y2-y1)/(x2-x1)
#c = y1-m*x1
mq = -qPeak/(q1-PRint)              #Pendiente de la recta Q
cq = -mq*PRint                      #Parametro c de la recta Q
mrUp = (rPeak+qPeak)/(r1-q1)        #Pendiemte de la recta creciente R
crUp = rPeak-mrUp*r1                #Parametro c de la recta creciente R
mrDown = (-sPeak-rPeak)/(r2-r1)     #Pendiemte de la recta decreciente R
crDown = rPeak-mrDown*r1            #Parametro c de la recta decreciente R
ms = sPeak/(s-r2)                   #Pendiemte de la recta S
cs = -ms*s                          #Parametro c de la recta s

#Generacion del complejo QRS
tQ = np.arange(PRint, q1, Ts)
Q = mq*tQ + cq                      #Funcion de la recta Q
tR = np.arange(q1, r1, Ts)
R = mrUp*tR + crUp                  #Funcion de la recta creciente R
tR2 = np.arange(r1, r2, Ts)
R2 = mrDown*tR2 + crDown            #Funcion de la recta creciente R
ts = np.arange(r2, s, Ts)
S = ms*ts + cs                      #Funcion de la recta S

#Generacion de onda P
tP = np.arange(0, Pint, Ts)        #eje de tiempo
P = pPeak*sp.sin(tP * sp.pi/Pint)  #Funcion de la onda P
```

Segmento de código 2: Simulación del complejo QRS

Para la simulación de las ondas P y Q como se lo había explicado en la sección 2.1.1.1, generamos una onda senoidal cuyo período es la duración de la onda P o T según sea el caso. En tanto para crear los segmentos PR, ST y TP simplemente creamos un vector de ceros. Observamos el Segmento de Código 3 el procedimiento para generar la onda P y el segmento PR.

```
#Generacion de onda P
tP = np.arange(0, Pint, Ts)        #eje de tiempo
P = pPeak*sp.sin(tP * sp.pi/Pint)  #Funcion de la onda P
#Generacion del segmento PR
tPR = np.arange(Pint, PRint, Ts)   #eje de tiempo
PRseg = np.zeros(len(tPR))         #segmento PR
```

Segmento de código 3: Simulación de la onda P y segmento PR

Finalmente, para obtener el pulso básico ECG establecemos la duración de la señal ECG y concatenamos cada componente en su respectivo orden en un vector. Este vector será la señal de entrada del Algoritmo Karplus-Strong para convertirla en una señal periódica. Para nuestro propósito la duración de la señal ECG simulada es de 60 segundos.

```
duracion = 20                                        #Duracion ECG periodica
#Senial ECG simple
ecg = np.concatenate((P, PRseg, Q, R, R2, S, STseg, T, TPseg))
M = len(ecg)                                        #Longitud del pulso ECG
n = int(np.ceil((duracion*M)/PeriodoLat))           #tamanio la senial eCG periodica
alfa = 1                                            #factor de atenuacion
B = [0.1*10**1]                                     #
A = np.concatenate((1, np.zeros(M-1), -alfa), axis=None)
xzeropad = np.concatenate((ecg, np.zeros(n-M)), axis=None)
ecgPer = sp.signal.lfilter(B, A, xzeropad)          #Señal ECG periodica
```

Segmento de código 4: Obtención del pulso ECG simple

Ahora implementamos el ruido presente en la señal ECG real, para esto incluiremos el ruido producido por la red eléctrica y el ruido inherente de los dispositivos electrónicos.

```
WN = 0.005*np.random.randn(len(ecgPer))          #Ruido blanco
EN_N = 0.005*sp.sin(2*sp.pi*60*t);               #Ruido de la red electrica
ECG_N = ecgPer + EN_N + WN                       #Señal ruidosa ECG
```

Segmento de código 5: Adición de ruido a la señal ECG

Finalmente para obtener una señal ECG real removemos el ruido mediante la implementación de un filtro digital pasabajos.

```
fc = 36                                          #Frecuencia de corte
fnorm = fc /(Fs/2)                               #Frecuencia normalizada
b,a= sp.signal.butter(3, fnorm, 'low')           #Coeficientes del filtro
ecgFilt = sp.signal.lfilter(b, a, ECG_N)         #Señal ECG filtrada
```

Segmento de código 6: Filtrado de la señal ECG

2.2.2. Implementación de las Señales Patológicas

Para poder generar las señales patológicas importamos los archivos .txt y mediante la librería Pandas leemos dichos archivos desde Python. El funcionamiento de la librería Pandas se describe a continuación.

- La lectura de los archivos se especifica el nombre y la ruta, en nuestro caso la ruta es la misma del programa principal.
- Se indica si el archivo dispone de una cabecera, por lo general la primera linea del fichero se la utiliza como cabecera, en nuestro caso no dispone de cabecera, lo cuál asignamos None a la propiedad Head.
- Se especifica que no se utiliza los espacios en blanco como separador mediante el parámetro `delim_whitespace`.
- Finalmente al DataFrame creado seleccionamos el número de filas y columnas que corresponden a los datos, nuestros archivos tienen un formato de 1 fila por N columnas.

El Segmento de Código 7 ilustra el procedimiento implementado para cargar las bases de datos por medio de la librería Pandas.

```
data = pd.read_csv('Arritmia.txt', header = 0, delim_whitespace=False)
senial_ecg = data.iloc[:,0]
```

Segmento de código 7: Procedimiento para cargar las bases de datos

2.2.3. Animación de las Señales Cardíacas

En este punto hemos conseguido mostrar gráficamente las señales cardíacas de forma estática sin ningún tipo de efecto que simule un monitor de electrocardiograma. Necesitamos que las señales cardíacas se grafiquen de tal forma que se asemeje a un electrocardiograma que adquiere los datos y los muestra gráficamente en tiempo real.

PyqthGraph es ideal para realizar esta tarea debido a que se revisó los ejemplos dados en la documentación de PyQtGraph y se verificó que uno de los métodos disponibles se basa en actualizar los datos de la matriz de la señal cardíaca de modo que la gráfica parezca desplazarse. Este método es conocido como el de Scrolling Plots (Gráficos de desplazamiento). En nuestro caso haremos desplazar los datos de la matriz de la señal cardíaca una posición a la izquierda. Para completar el efecto del electrocardiograma implementamos el método `update()` que actualizará los datos cada 15 milisegundos y permitirá ver a la señal desplazarse hacia la izquierda de la pantalla. En el Segmento de Código 9 se muestra el ejemplo para animar la señal del ritmo sinusal.

En primer lugar, importamos la librería PyqtGraph.

```
import pyqtgraph as pg
from pyqtgraph.Qt import QtCore, QtGui
```

Segmento de código 8: Importación de la librería PyqtGraph

```
#Cargamos el archivo de la base de datos
data = pd.read_csv('Ritmo_sinusal.txt', header = 0, delim_whitespace=False)
#Creamos un DataFrame que contendrá los datos
senial_ecg = data.iloc[:,0]
fs = 128                              #Frecuencia de muestreo
L = len(senial_ecg)                   #Longitud de la señal
t = (np.arange(0, L)) / fs            #vector de tiempo
Color = pg.mkPen('g', width=1)        #Parametros de la grafica: color, ancho
self.plotWidget.setYRange (-1, 3)             #Fijamos limites del eje x
curve = self.plot(t,senial_ecg, pen=Color)    #Creamos una curva para graficar

# Metodo para actualizar los datos
# Cada vez que se llama a esta función, la pantalla de datos se actualiza
def update():
    #desplaza los datos en la matriz una muestra a la izquierda.
    senial_ecg[:-1] = senial_ecg[1:]
    #establecemos la curva con estos datos
    curve.setData(t, senial_ecg)
#Actualizamos los datos llamando al Metodo Update cada 15 milisegundos
timer = pg.QtCore.QTimer()
timer.timeout.connect(update)
timer.start(15)
```

Segmento de código 9: Animación de la señales

2.2.4. Comunicación entre las Interfaces

Los equipos del profesor y estudiante se comunicaran por medio de un modelo Cliente/-Servidor. La aplicación del estudiante será el servidor y la aplicación del profesor será el cliente. Crearemos dos scripts uno para el Servidor y otra para el cliente. Para ambos casos las librerías que se utilizan son las mismas.

```
import socket
from threading import Thread
```

Segmento de código 10: Importación de las librerías a utilizar

Un problema común al crear aplicaciones GUI es “bloquear” la interfaz al intentar realizar tareas en segundo plano. En nuestro caso el proceso de comunicación entre el cliente y el servidor hace que se congele la GUI. La solución es sacar su trabajo del hilo de la GUI y crear un hilo individual utilizando el módulo Thread.

2.2.5. Programación del Socket

Para establecer una conexión entre el servidor y el cliente necesitaremos un objeto socket. Para crear el objeto de socket, debemos utilizar el método `socket()` disponible en el módulo socket que consta de los siguientes argumentos.

- **Dirección del Socket:** La dirección del socket se representa usando ciertas familias de direcciones. Cada familia de direcciones requiere ciertos parámetros para establecer una conexión. Utilizaremos la familia de direcciones `AF_INET` en esta aplicación. La

familia de direcciones `AF_INET` necesita un par (host, port) para establecer una conexión donde el parámetro host es la dirección IPv4 que está en formato string, port es un número entero que representa el número de puerto utilizado para la comunicación [34].

- **Tipo del Socket:** El tipo de socket se representa a través de varias constantes: `SOCK_STREAM` , `SOCK_DGRAM`, `SOCK_RAW`, `SOCK_RDM` y `SOCK_SEQPACKET`. Utilizaremos el tipo de socket más utilizado `SOCK_STREAM` que corresponde al protocolo TCP [34].

```
1        tcpServer = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
2        tcpServer.setsockopt(socket.SOL_SOCKET, socket.SO_REUSEADDR, 1)
```

Segmento de código 11: Parámetros para la creación del socket

El método `setsockopt()` se usa en la aplicación para establecer el valor de la opción de socket dada. Incluye los siguientes dos parámetros esenciales:

- **SOL_SOCKET:** Este parámetro es la capa de socket en sí. Se utiliza para opciones independientes del protocolo [34].
- **SO_REUSEADDR:** Este parámetro permite que otros sockets se unan a este puerto a menos que ya haya un socket de escucha activo vinculado al puerto [34].

En el caso del servidor, vinculará un socket a algún puerto del servidor. En el caso del cliente, conectará un socket a ese servidor, en el mismo puerto que está usando el código del lado del servidor.

Después de crear el socket en el lado del servidor se utiliza el método `bind()` para vincularle una dirección, el método `listen()` para colocarlo en el estado de escucha y, finalmente, el método `accept()` para aceptar una nueva conexión del cliente. Esto se indica en el Segmento de Código 12.

```
1        #Vincula EL socket a una dirección y puerto en particular
2        tcpServer.bind((TCP_IP, TCP_PORT))
3        #Escucha las conexiones entrantes, permite hasta 1 conexion
4        tcpServer.listen(1)
5        #Establecemos la conexión con el cliente
6        (ip,port) = tcpServer.accept()
```

Segmento de código 12: Socket lado servidor

Para crear el socket en el lado del cliente, el procedimiento es similar con la diferencia de que en lugar de utilizar el método `bind()` lo hacemos con el método `connect()` tal y como se indica en el Segmento de Código 13.

```
1       #Creamos el socket
2       tcpClient = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
3       #Coneccion con el Servidor
4       tcpClient.connect((host, port))
```

Segmento de código 13: Socket lado del cliente

Una vez que tenemos un socket inicializado podemos usar algunos métodos para enviar datos, recibir datos y finalmente cerrar la conexión.

```
        #Enviar datos, este método se puede llamar varias veces
        tcpClient.send("Datos a enviar")
        #Recibe hasta 2048 bytes del otro extremo
        tcpClient.recv(2048)
        #Cerrar la conexión del socket
        tcpClient.close()
```

Segmento de código 14: Métodos de un socket

2.2.5.1. Implementación del Servidor

Creamos una clase `ServerThread()` que hereda la clase Thread del módulo de subprocesos de Python. El método `run()` se anula cuando se definen las variables `TCP_IP` y `TCP_HOST` y la variable tcpServer está vinculado con estas variables.

```
class ServerThread(Thread):
    def __init__(self,window):
        Thread.__init__(self)
        self.window=window

    def run(self):
        TCP_IP = '0.0.0.0'          #Direccion de host
        TCP_PORT = 5000             #Puerto arbitrario
        BUFFER_SIZE = 32            #Tamanio del Buffer
        # Creamos un objeto socket tipo TCP
        tcpServer = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
        tcpServer.setsockopt(socket.SOL_SOCKET, socket.SO_REUSEADDR, 1)
        #Vincula EL socket a una dirección y puerto en particular
        tcpServer.bind((TCP_IP, TCP_PORT))
        #Escucha las conexiones entrantes, permite hasta 1 conexion
        tcpServer.listen(1)
```

Segmento de código 15: Creación de la Clase Server

Posteriormente, el servidor se mantiene a la espera de las conexiones entrantes. Para cada nueva conexión de cliente, el servidor crea un nuevo subproceso de cliente dentro del bucle While. Esto es para que al momento de crear un nuevo subproceso para cada cliente, dicho proceso no bloquee la funcionalidad de la GUI. Para nuestra aplicación el servidor manejará una sola conexión al mismo tiempo.

```
        while True:
            print("Python Server: Waiting for connections from TCP clients...")
            global conn
            #Establecemos la conexión con el cliente
            (conn, (ip,port)) = tcpServer.accept()
            newthread = ClientThread(ip,port,window)
            newthread.start()
            threads.append(newthread)
```

Segmento de código 16: Conexión de Clientes

2.2.5.2. Implementación del Cliente

La idea principal es explicar cómo se envía un mensaje, cómo el servidor escucha el puerto y cómo se establece la comunicación entre los dos. Se implementa una clase `ClientThread()` en el script del servidor que maneja la comunicación con el cliente, el método `run()` se anula y espera los datos recibidos del cliente. Se pasa un objeto de clase `Window()` a la clase `ClientThread()` que lo usa para acceder al contenido de la clase de la GUI del estudiante. Los datos recibidos se decodifican porque están en formato de bytes y deben convertirse en cadenas utilizando la codificación UTF-8.

```
class ClientThread(Thread):

    def __init__(self,ip,port,window):
        Thread.__init__(self)
        self.window=window
        self.ip = ip            # Direccion IP del Cliente
        self.port = port        # Puerto
        print("New server socket thread started for " + ip + ":" + str(port))
```

Segmento de código 17: Clase ClientThread

Para el script del lado del cliente tenemos de manera similar una clase `ClientThread()` que es una clase que hereda la clase Thread y anula el método `run()`. En el método `run()` para conectarse al servidor se introduce la dirección IP y el puerto 5000.

```
    def run(self):
       host ='127.0.0.1'        #Direccion del Servidor
       port = 5000              #Puerto del servidor
       BUFFER_SIZE = 2048       # Tamanio del buffer
       global tcpClient
       #Creamos el socket
       tcpClient = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
       #Coneccion con el Servidor
       tcpClient.connect((host, port))
```

Segmento de código 18: Método run del script del cliente

Una vez que se establece una conexión con el servidor, el cliente intenta recibir datos del servidor dentro del bucle while.

```
       while True:
           #Espera a recibir los datos
           data = tcpClient.recv(BUFFER_SIZE)
           # Convertimos str a bytes
           window.chat.append(data.decode("utf-8"))
       tcpClient.close()
```

Segmento de código 19: Proceso para la recepción de datos

Al recibir los datos del servidor, los datos se convierten en formato de cadena del formato de bytes.

2.2.5.3. Transmisión de Datos

Gracias al bucle While podemos enviar y recibir mensajes indefinidamente a menos que cerremos el socket. Como ya lo revisamos enviar un mensaje es muy sencillo, lo podemos hacer desde cualquier parte del código simplemente referenciando a la variable del socket ya sea del servidor o el cliente. En el Segmento de Código 20 podemos observar cómo se envían los parámetros para construir la señal ECG.

```
#Parámetros para la onda P
self.amP = self.sbPW.value()
self.interP = self.sbPint.value()
#Parámetros para el intervalo PR
self.interPR = self.sbInterPR.value()
#Parámetros para el complejo QRS
self.interQRS = self.sbQRSint.value()
self.ampQRS = self.sbQRSamp.value()
#Parámetros para para la onda T
self.amT = self.sbTW.value()
self.interT = self.sbTint.value()

#Almaceno los valores en una Lista
self.stream_data = [self.BMP, self.amP, self.interP, self.interPR, self.ampQRS,
    self.interQRS, self.amT, self.interT]
data = self.stream_data

#Envío de los parámetros básicos de la ECG Normal
if data:
    for entry in data:
        new_data = str("%s\n" %entry).encode("utf-8")
        tcpClientA.send(new_data)
        time.sleep(0.1)   #Pausa de 1 ms
```

Segmento de código 20: Proceso para el envío de datos

Observamos que guardamos los valores ingresados por el profesor desde su GUI en una lista y enviamos cada elemento con una pausa de 1 ms entre cada elemento de la lista, esto es para que los valores lleguen uno por línea y no se junten los valores cuando lleguen al otro extremo.

Para recibir los datos, se lo hace dentro del bucle While tanto en el lado del Servidor como del Estudiante, utilizamos una estructura de control mediante `if`, `elif` y `else` para procesar los datos recibidos y realizar las acciones respectivas del programa.

```
while True :
    global conn
    data = conn.recv(2048)
    data_recv = data.decode('utf-8')

    if  data_recv == 'arritmiasinusal':
        window.patologias(data_recv)

    elif data_recv == 'fibriventricular':
```

```
        window.patologias(data_recv)

    elif data_recv == 'fibriauricular':
        window.patologias(data_recv)

    elif data_recv == 'ectopya':
        window.patologias(data_recv)

    elif data_recv == 'taquisupraventricular':
        window.patologias(data_recv)

    elif data_recv == 'taquiventricular':
        window.patologias(data_recv)

    elif data_recv == 'ParadaCardioR':
        window.patologias(data_recv)

    elif data_recv == 'reset':
        window.Clear()
    else:
        #Convertimos de string a entero el primer parámetro que corresponde
        #a la frecuencia cardiaca
        data = int (data_recv)
        #Guardamos este valore en la primera posiciond de la lista
        datos[0] = data
        #Recibimos los parametros de la ECG uno a uno.
        #Guardamos en una lista y llamamos al metodo que construye la ECG
        for j in range(1, L):
            msg = conn.recv(1024)
            data = float( msg.decode('utf-8') )
            datos[j] = data
        window.ConstruirEcg(datos)
```

Segmento de código 21: Control de los datos recibidos

2.2.6. Implementación del Simulador de Desfibrilación

Una vez que hemos establecido la conexión entre la interfaz del profesor y estudiante mediante un modelo Cliente/Servidor ahora pueden interactuar entre sí. Entonces para implementar la desfibrilación y cardioversion eléctrica en primer lugar, el profesor enviará al estudiante alguna señal cardíaca. El estudiante valorará la arritmia, es decir, antes de aplicar el choque eléctrico, verificará que la arritmia sea susceptible de recibir esta terapia.

Se ha creado el método `AplicarShock()` dentro de la clase `Window()` del script del estudiante para aplicar desfibrilación y cardioversión. Las variables se obtienen en el caso de la energía de la GUI del estudiante. El tipo de arritmia se estable en los métodos donde se grafican las señales cardíacas y el número de intentos se asigna en esta sección.

```
def AplicarShock(self):
    arritmia = self.tipo                        #Tipo de Arritmia
    energia = self.spinBox_Energia.value()  #Valor de la Energia
    intentos = 2                                #Numeros de intentos del Shock
```

Segmento de código 22: Variables necesarias para aplicar desfibrilación

Para la desfibrilación hemos tomado en cuenta la energía en Joules y el número de shocks necesarios para restablecer el ritmo normal o ritmo sinusal. Por ejemplo si el estudiante aplica desfibrilación a un ritmo que no es viable para aplicar esta tipo de terapia, el ritmo cardíaco actual se verá afectado, como se indica en el Segmento de Código 23.

```
if arritmia == "RitmoNormal":
    print ('ritmo normal')
    window.statusbar.showMessage('Desfibrilacion mal aplicada....')
    self.plotWidget.clear()          #Limpia la pantalla del Electrocardiogram
    entry = 'cambio2'                #Variable de control
    conn.send(entry.encode("utf-8"))  #Enviamos informacion al profesor
    #Cargamos una base de Datos
    data = pd.read_csv('MuerteCardiaca.txt', header = 0, delim_whitespace=False)
    self.senial_ecg = data.iloc[:,0]
    fs = 250
    L = len(self.senial_ecg)
    self.t = (np.arange(0, L)) / fs
    self.plotWidget.setYRange (-1.5, 3.1)
    Color = pg.mkPen('g', width=1)
    #Graficamos la señal
    self.curve = self.plotWidget.plot(self.t, self.senial_ecg, pen=Color)
    #Llamamos al metodo Update para animar el grafico
    self.timer.timeout.connect(self.update)
```

Segmento de código 23: Desfibrilación a un ritmo no aplicable

Cuando la señal sea aplicable a esta terapia, el ritmo se restablecerá cuando el números de shocks aplicados sea igual a 3 y la energía suministrada este dentro del rango establecido para cada patología. De no ser así, se mostrará un mensaje en el StatusBar indicando que es necesario aplicar un shok adicional.

```
elif arritmia == "fibriventricular":
    if 200 <= energia <= 300:
        if self.cont == intentos:
            window.statusbar.showMessage('Ritmo Normal Restablecido........')
            self.plotWidget.clear()
            entry = 'cambio'
            conn.send(entry.encode("utf-8"))
            data = pd.read_csv('Ritmo_sinusal.txt', header = 0,
                                                    ,delim_whitespace=False)
            self.senial_ecg = data.iloc[:,0]
            fs = 128
            L = len(self.senial_ecg)
            self.t = (np.arange(0, L)) / fs
            self.plotWidget.setYRange (-1, 3)
            Color = pg.mkPen('g', width=1)
```

```
            self.curve = self.plotWidget.plot(self.t, self.senial_ecg,
                                                                    ,pen=Color)
            self.timer.timeout.connect(self.update)
        else:
            window.statusbar.showMessage('Patologia Persiste es necesario
                                        "aplicar otro shock con mayor energia')
    self.cont += 1
```

Segmento de código 24: Desfibrilación a un ritmo aplicable

2.2.7. Implementación del Simulador de Cardioversión

De igual forma que la desfibrilación, la cardioversión eléctrica es una terapia en la que debemos aplicar una descarga eléctrica que permita restablecer el ritmo normal pero de forma sincronizada. Para lograr esto, en primer lugar hemos implementado el método `MostarSinc()` donde se utiliza una variable booleana y definimos `True` cuando el modo sincrónico esta activado y `False` cuando el modo sincrónico esta desactivado. Adicional este método muestra una marca en el centro de la pantalla que sirve de referencia para aplicar la descarga. Para los ritmos que son aplicables la cardioversión como por ejemplo la fibrilación ventricular, el método `MostrarSinc()` no tendrá ningún efecto.

Ahora para encontrar el punto de descarga que coincida con el pico de la onda R, fijamos un umbral de tal forma que nos permita tener una cierta anchura (duración), así como cierta altura (amplitud). Sabemos que la marca de descarga se encuentra en la mitad, y el eje del tiempo comprende entre $[0;3]$ s, por lo tanto el punto donde tenemos que verificar se encuentra dentro del umbral es en $t = 1{,}5$ s. Realizamos un pequeño cálculo para encontrar el número de la muestra.

$$Muestra = f_s \cdot t \tag{2.17}$$

En dónde f_s es la frecuencia de muestreo de la señal cardíaca y t es igual a 1.5 segundos. Ejemplo, para la Arritmia Sinusal el número de muestra es:

$$Muestra = 250 \cdot 1{,}5 = 375 \tag{2.18}$$

De igual forma si el punto de la señal donde se aplique la descarga no se encuentra dentro del umbral, aparecerá un mensaje en el StatusBar de la GUI del estudiante indicando que es necesario aplicar una nueva descarga.

```
if self.ModoSinc == True:                #Variable Booleana
    punto = self.senial_ecg[375]         #Numero de Muestra para t=1.5s
    if punto > -0.4:
        window.statusbar.showMessage('Ritmo Normal Restablecido.........')
        self.plotWidget.clear()
        entry = 'cambio'
        conn.send(entry.encode("utf-8"))
        data = pd.read_csv('Ritmo_sinusal.txt', header = 0
                                                    ,delim_whitespace=False)
        self.senial_ecg = data.iloc[:,0]
        fs = 250
```

```
        L = len(self.senial_ecg)
        self.t = (np.arange(0, L)) / fs
        self.plotWidget.setYRange (-1, 3)
        Color = pg.mkPen('g', width=1)
        self.curve = self.plotWidget.plot(self.t, self.senial_ecg, pen=Color)
        self.timer.timeout.connect(self.update)
    else:
        window.statusbar.showMessage('Patologia Persiste es necesario aplicar
                                     "otro shock con mayor energia')
```

Segmento de código 25: Proceso para simular la Cardioversión

2.2.8. Implementación del Simulador de Medicación

La sección de Medicación permite aplicar un tipo de terapia frente a las arritmias mediante el suministro de fármacos. Para esto creamos el método `Medicar()`, que no tiene argumentos de entrada pero necesita de la variable compartida `self.tipo` y del valor de la dosis ingresada desde la interfaz del estudiante. De acuerdo a la arritmia enviada por el profesor se comprobará en el condicional si la dosis se encuentra dentro del rango para restablecer el ritmo normal, si no cumple se verificará las demás condiciones y según sea el caso se aplicaran los efectos adversos de cada patología.

```
if arritmia == "arritmiasinusal":
    if dosis < 300:
        # Muestra mensaje en el StatusBar
        window.statusbar.showMessage("Dosis Ineficaz")

    elif 300 <= dosis <= 350:
        window.statusbar.showMessage("Dosis efectiva")
        self.plotWidget.clear()
        #Se envia la informacion al profesor
        streamdts = [dosis, 'cambio']
        if streamdts:
            for valores in streamdts:
                new_data = str ("%s" %valores).encode("utf-8")
                conn.send(new_data)
                time.sleep(0.1)

        #Se cambia el ritmo cardiaco
        data = pd.read_csv('Ritmo_sinusal.txt', header = 0,
                                                    ,delim_whitespace=False)
        self.senial_ecg = data.iloc[:,0]
        fs = 250, L = len(self.senial_ecg)
        self.t = (np.arange(0, L)) / fs
        self.plotWidget.setYRange (-1, 3), Color = pg.mkPen('g', width=1)
        self.curve = self.plotWidget.plot(self.t, self.senial_ecg, pen=Color)
        self.timer.timeout.connect(self.update)

    else:   #Para dosis mayores a 350mg se considera como Sobredosis
        window.statusbar.showMessage("dosis erronea.... Arritmia Inducida")
        self.plotWidget.clear()
```

```
        streamdts = [dosis, 'cambio6']
        if streamdts:
            for valores in streamdts:
                new_data = str ("%s" %valores).encode("utf-8")
                conn.send(new_data)
                time.sleep(0.1)

        #Se cambia de ritmo
        data = pd.read_csv('taquiVentricular.txt', header = 0,
                                                        ,delim_whitespace=False)
        self.senial_ecg = data.iloc[:,0]
        fs = 250, L = len(self.senial_ecg)
        self.t = (np.arange(0, L)) / fs
        self.plotWidget.setYRange (-1.5, 1.5), Color = pg.mkPen('g', width=1)
        self.curve = self.plotWidget.plot(self.t, self.senial_ecg, pen=Color)
        self.timer.timeout.connect(self.update)
```

Segmento de código 26: Método para simular la Medicación

3. RESULTADOS Y DISCUSIÓN (APLICACIÓN METODOLÓGICA)

En esta sección se presentan los resultados obtenidos de nuestro proyecto. En primer lugar se realizó una verificación para comprobar que todas las funciones del software se encuentren operando correctamente. Luego se verificó la funcionalidad del software con usuarios (i.e. médicos del área) que nos ayuda como retroalimentación y aprobación de las funcionalidades del software.

3.1. Pruebas de Funcionalidad

Aquí es donde se ejecutan los diferentes componentes que han sido diseñados en la aplicación con el fin de comprobar su correcto funcionamiento. El objetivo es asegurarse de que cumpla con las especificaciones que fueron diseñadas. Se empezará con el proceso inicial, es decir, la conexión entre las dos interfaces. Luego se comprobará la simulación de la señal ECG y de las señales patológicas. Posterior se verificará el proceso de desfibrilación y cardioversión. Y finalmente se verificará el proceso de medicación.

3.1.1. Resultados de Pruebas de la Conexión de las Interfaces

La prueba de conexión se realizó ejecutando los scripts en dos máquinas diferentes conectadas en una LAN. En primer lugar se ejecuta el script del estudiante que se mantiene a la espera de conexiones entrantes. Luego se ejecuta el script del profesor y se establece la conexión entre las dos interfaces. En la Figura 3.1 se observa el resultado de este proceso. Podemos observar que en la interfaz del estudiante nos muestra un mensaje sobre el estado de la conexión.

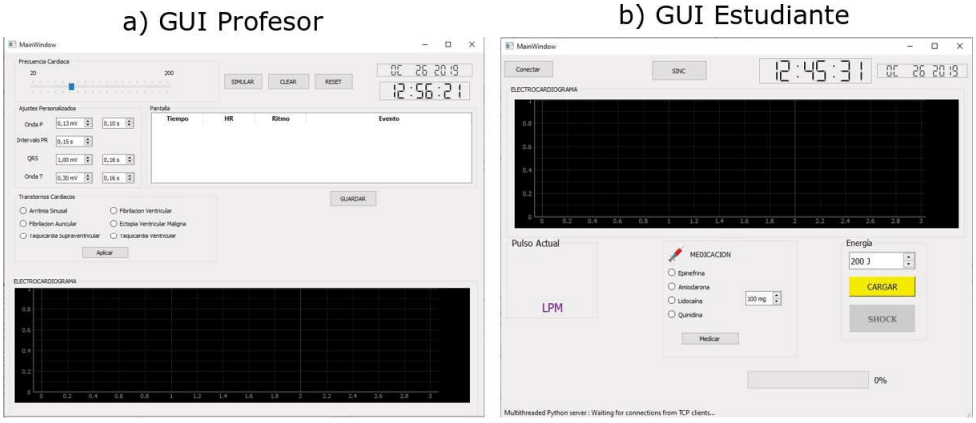

Figura 3.1: Prueba de la conexión de las interfaces

3.1.2. Resultados de la Simulación de la Señal ECG

Una vez establecida la conexión entre las interfaces se realizó las pruebas de la señal ECG simulada. En la Figura 3.2 se observa el resultado de simular un ritmo sinusal, en la Figura 3.3a se observa el resultado de simular una bradicardia y en la Figura 3.3b se observa

el resultado de simular una taquicardia. Cabe recalcar que además de variar la frecuencia cardíaca se modificó los otros parámetros básicos como lo son las amplitudes e intervalos.

a) GUI Profesor b) GUI Estudiante

Figura 3.2: Simulación de una señal ECG básica

a) Bradicardia b) Taquicardia

Figura 3.3: Simulación de bradicardia y taquicardia

3.1.3. Resultados de la Simulación de las Señales Patológicas

A continuación se presenta como se cargan las señales patológicas implementadas. Para enviar las señales patológicas simplemente se selecciona cualquier base de datos del panel de las señales patológicas y se presiona aplicar, automáticamente se envía a la interfaz del estudiante. En la bitácora implementa den la GUI del profesor se muestra la información de la señal cargada.

3.1.3.1. Arritmia Sinusal

Observamos en la Figura 3.4 el resultado de cargar la base de datos de la señal arritmia sinusal.

3.1.3.2. Fibrilación Ventricular

Observamos en la Figura 3.5 el resultado de cargar la base de datos de la señal fibrilación ventricular.

a) GUI Profesor

b) GUI Estudiante

Figura 3.4: Arritmia sinusal

a) GUI Profesor

b) GUI Estudiante

Figura 3.5: Fibrilación ventricular

3.1.3.3. Fibrilación Auricular

Observamos en la Figura 3.6 el resultado de cargar la base de datos de la señal fibrilación auricular.

a) GUI Profesor

b) GUI Estudiante

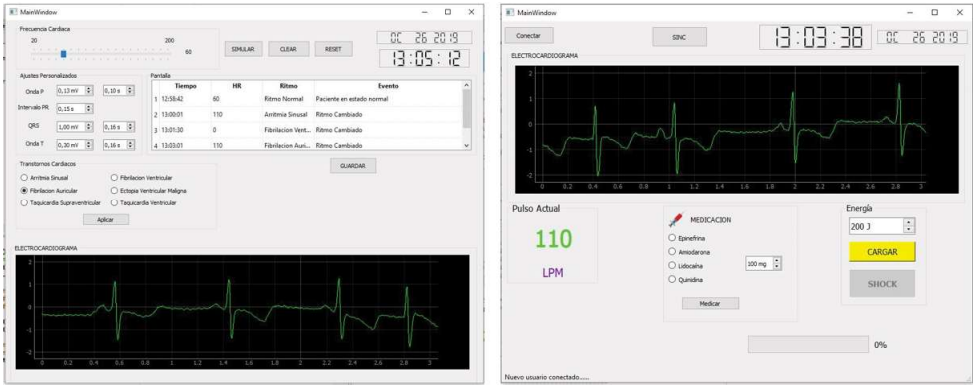

Figura 3.6: Fibrilación auricular

3.1.3.4. Ectopia Ventricular Maligna

Observamos en la Figura 3.7 el resultado de cargar la base de datos de la señal ectopia ventricular maligna.

a) GUI Profesor b) GUI Estudiante

Figura 3.7: Ectopia ventricular maligna

3.1.3.5. Taquicardia Supraventricular

Observamos en la Figura 3.8 el resultado de cargar la base de datos de la señal taquicardia supraventricular.

a) GUI Profesor b) GUI Estudiante

Figura 3.8: Taquicardia supraventricular

3.1.3.6. Taquicardia Ventricular

Observamos en la Figura 3.9 el resultado de cargar la base de datos de la señal taquicardia ventricular.

3.1.4. Resultados de la Simulación de Desfibrilación

En este punto el profesor podrá proponer al estudiante cualquier escenario, uno de estos escenarios es dónde el estudiante puede aplicar la desfibrilación como terapia a determinada arritmia.

En primer lugar se muestra el resultado en la Figura 3.10 cuando se aplica una descarga eléctrica a una bradicardia, dicha señal cardíaca no es susceptible a recibir esta terapia, por

a) GUI Profesor

b) GUI Estudiante

Figura 3.9: Taquicardia ventricular

lo que provoca una muerte cardíaca súbita. El resultado será similar para cualquier ritmo cardíaco que no sea aplicable a esta terapia.

a) GUI Profesor

b) GUI Estudiante

Figura 3.10: Resultado de desfibrilación a un ritmo no aplicable

Ahora veamos cuando el ritmo cardíaco es factible a desfibrilación. En la Figura 3.11 se observa cuando estamos en un caso fibrilación ventricular y se aplica una descarga de 200 J. Para que este procedimiento sea exitoso el estudiante debe que aplicar 3 descargas consecutivas y con crecimientos graduales de la energía.

a) GUI Profesor

b) GUI Estudiante

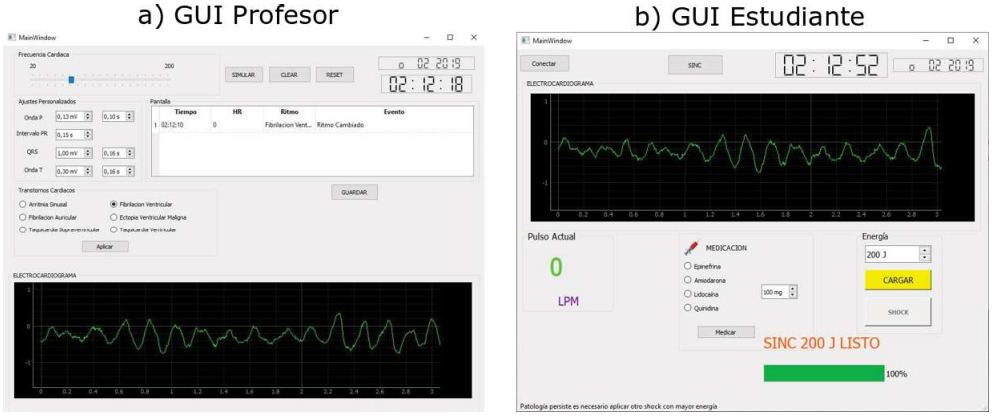

Figura 3.11: Desfibrilación a un ritmo aplicable

En la bitácora de la interfaz del profesor se registra la información de la energía aplicada y de los ritmos cardíacos. En la Figura 3.12 se observa como se restablece el ritmo normal luego del proceso de desfibrilación.

a) GUI Profesor b) GUI Estudiante

Figura 3.12: Resultado de una desfibrilación exitosa

3.1.5. Resultados de la Simulación de la Cardioversión

La cardioversión es un procedimiento similar a la desfibrilación con la diferencia que en la cardioversión la descarga se realiza de forma sincronizada con el complejo QRS específicamente con la onda R. Para aplicar la descarga se necesita activar el modo sincrónico que muestra en pantalla un marcador. Si el ritmo cardíaco no es aplicable a recibir este tipo de tratamiento el modo sincrónico no se activará de modo que no habrá la posibilidad de aplicar una descarga. En la Figura 3.13.a se observa cuando estamos ante un caso de ectopia ventricular maligna, esta señal al tener ausencia de latidos y presentar una morfología caótica no es posible activar el marcador el modo sincrónico. En la Figura 3.13.b se observa una arritmia sinusal con el marcador que permite aplicar la descarga en el momento que el pico de la onda R pase por ese punto.

a) Ritmo no Aplicable b)Ritmo Aplicable

Figura 3.13: Resultado para ritmos aplicables y no aplicables a cardioversión

Cuando el modo sincrónico se activa, la marca sirve como referencia para que el estudiante pueda aplicar la descarga en el momento que el complejo QRS pase por este punto. Si

la descarga no sucede en en el complejo QRS aparecerá un mensaje en el StatusBar del estudiante indicando que es necesario aplicar otra descarga. El ritmo se restablece cuando la descarga ocurre en el complejo QRS como se observa en la Figura 3.14.

a) GUI Profesor b) GUI Estudiante

Figura 3.14: Resultado de una cardioversión exitosa

3.1.6. Resultados de la Simulación de Medicación

La medicación permite combatir contra las arritmias utilizando sustancias farmacológicas. De igual manera que la desfibrilación y cardioversión se debe tomar en cuenta el ritmo cardíaco que se esta tratando. Si aplicamos una determinado fármaco a un ritmo que no sea susceptible a recibirlo se reflejarán los efectos secundarios.

3.1.6.1. Epinefrina

Se presenta los resultados para los casos posibles cuando se suministra esta medicación. Cabe recalcar que la dosis adecuada para la epinefrina es 1 a 5 mg. En primer lugar se muestra el resultado en la Figura 3.15 cuando se suministra correctamente el medicamento, en este caso la patología se trata de una bradicardia.

a) GUI Profesor b) GUI Estudiante

Figura 3.15: Resultado cuando la epinefrina es correctamente suministrada

Si la dosis suministrada no es la correcta, en este caso se observa en la Figura 3.16 que se produce una arritmia sinusal.

a) GUI Profesor b) GUI Estudiante

Figura 3.16: Resultado cuando se suministra una dosis incorrecta

3.1.6.2. Amiodarona

Tomando en cuenta la dosis recomendada y los efectos secundarios, se presentan los resultados obtenidos cuando se aplica la amiodarona. En primer lugar cuando se aplica un cantidad por debajo de la dosis recomendada, observamos en la Figura 3.17 el resultado.

a) GUI Profesor b) GUI Estudiante

Figura 3.17: Resultado cuando se suministra una dosis ineficaz

Se presenta el resultado cuando se aplica la dosis correcta ante un caso de fibrilación ventricular. Se observa en la Figura 3.18 que el ritmo sinusal se restablece y además en la bitácora de la GUI del profesor se registra la cantidad suministrada por el estudiante.
Se presenta el resultado cuando se suministra una sobredosis. En la Figura 3.19 se observa que al aplicar 400 mg de Amiodarona a un paciente con arritmia sinusal se induce una taquicardia ventricular.

3.1.6.3. Lidocaína

Se presentan los resultados de la pruebas ejecutadas para los posibles escenarios cuando se aplica la lidocaína como parte del tratamiento de una arritmia. En primer lugar se presenta el resultado en la Figura 3.20 cuando la dosis aplicada a cualquier señal cardíaca está por debajo de la dosis recomendada y por lo tanto no se ve afectada.

Luego, se presenta el resultado cuando se suministra una dosis correcta a una arritmia. Se

a) GUI Profesor b) GUI Estudiante

Figura 3.18: Resultado cuando se suministra una dosis correcta

a) GUI Profesor b) GUI Estudiante

Figura 3.19: Resultado cuando se suministra una sobredosis

a) GUI Profesor b) GUI Estudiante

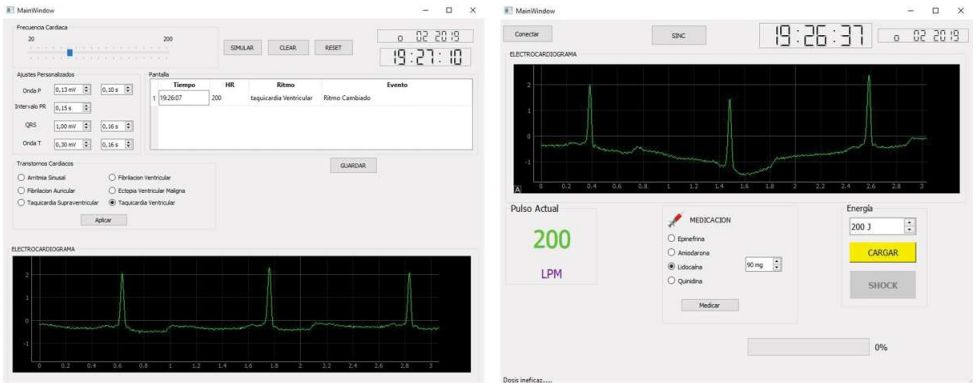

Figura 3.20: Resultado cuando se dosifica una baja cantidad

observa en la Figura 3.21 que ante un caso de taquicardia ventricular al aplicar 120 mg del medicamento se restablece el ritmo normal del paciente.

Por último, se presenta el resultado cuando se suministra una sobredosis. En la Figura 3.22 se observa que al aplicar 320 mg de lidocaína a un paciente con taquicardia ventricular se induce una arritmia sinusal.

a) GUI Profesor b) GUI Estudiante

Figura 3.21: Resultado cuando se aplica una dosis correcta

a) GUI Profesor b) GUI Estudiante

Figura 3.22: Resultado cuando se aplica una sobredosis

3.1.6.4. Quinidina

Se presentan los resultados de la pruebas ejecutadas para los posibles escenarios cuando se aplica quinidina como parte del tratamiento de una arritmia.

En primer lugar se presenta cuando la dosis aplicada a cualquier señal cardíaca esta por debajo de la dosis recomendada. En la Figura 3.23.a se observa cuando se suministra al ritmo sinusal y se muestra un mensaje de alerta y en la Figura 3.23.b se observa cuando se aplica a una arritmia sinusal y se indica a través de un mensaje que la dosis es ineficaz. Para ambos casos el ritmo cardíaco no cambia en el electrocardiograma.

Luego, se presenta el resultado cuando se suministra una dosis correcta a una arritmia. Se observa en la Figura 3.24 que ante un caso de taquicardia supraventricular al aplicar 150 mg del medicamento se muestra en la pantalla del profesor la base de datos que describe los efectos de la quinidina.

Por último, se presenta el resultado cuando se suministra una sobredosis. En la Figura 3.25 se observa que al aplicar 630 mg de quinidina a un paciente con fibrilación auricular se induce una arritmia sinusal.

a) Baja Dosificación a Ritmo Normal

b) Baja Dosificación a Arritmia

Figura 3.23: Resultado cuando se aplica una baja cantidad de dosis

a) GUI Profesor

b) GUI Estudiante

Figura 3.24: Resultado cuando se aplica una dosis correcta

a) GUI Profesor

b) GUI Estudiante

Figura 3.25: Resultado cuando se suministra una sobredosis

3.2. Pruebas con Usuarios

Con el fin validar la funcionalidad y conocer la opinión de personas calificadas en el área de la salud se presentó nuestra aplicación y se realizaron pruebas con la ayuda de dos médicos.

Figura 3.26: Validación del software realizada con médicos

La primera validación del software se realizó con la ayuda del Dr. Luis Ojeda, Médico General de la Clínica Santa Lucía de la Ciudad de Macas. La segunda validación se realizó con la ayuda del Dr. Juan Pablo Román, Médico Internista de la Clínica Santa Fe de la Ciudad de Macas.

Los pruebas que se efectuaron fueron las siguientes:

- Se simuló una señal ECG a tres diferentes frecuencias de tal forma de obtener una taquicardia, bradicardia y un ritmo sinusal.
- Se seleccionaron las seis diferentes patologías y se las presentaron en las dos interfaces para que los médicos evaluaran las formas de onda.
- Se presentó un ejemplo del procedimiento de desfibrilación y un ejemplo del procedimiento de cardioversión eléctrica, para mostrar a los médicos la forma como se había implementado estos procesos.
- Finalmente, se presentó los medicamentos seleccionados en nuestra aplicación y los efectos que se implementaron.

Los escenarios y las funciones presentadas se evaluaron por medio de una encuesta que tiene por objetivo conocer la opinión de los médicos a través de indicadores que indican el nivel de satisfacción. En esta encuesta se definieron 12 preguntas, 11 de opción múltiple

que nos permite evaluar el nivel de aceptación del software, una pregunta para conocer la opinión de los médicos sobre la utilización de la simulación como herramienta educación y una pregunta abierta que ayuda a la búsqueda de mejoras a la aplicación.

La escala de las respuestas esta compuesta por las siguientes opciones: Extremadamente satisfecho/Muy desacuerdo, Muy satisfecho/De acuerdo, algo satisfecho/No estoy seguro, no tan satisfecho/En desacuerdo y nada satisfecho/Muy en desacuerdo.

3.2.1. Valoración de los Resultados

Las preguntas 1,2,3,4 y 8 se han agrupado con el objetivo de conocer la satisfacción del uso y apariencia del software. Las preguntas 5,6,7,9 y 10 se han agrupado para determinar la opinión de los médicos con respecto a las funcionalidades de la aplicación.

En lineas generales, los encuestados se encuentran muy satisfechos con la apariencia y facilidad de uso de la aplicación. En cuanto a la funcionalidad de la aplicación los encuestados indican que están de acuerdo en que la aplicación responde acorde como se quiere. Los dos médicos responden que están de acuerdo que la ejecución del programa es correcta, no se detiene inesperadamente ni se ejecuta lentamente.

Para la pregunta 11, como se esperaba los dos médicos contestaron en la encuesta que se encuentran muy de acuerdo que se utilicen este tipo de herramientas en el proceso de aprendizaje para los estudiantes del área de la salud. Ellos comentan que no todos los centros educativos cuentan con la facilidad de integrar este tipo de herramientas y esta aplicación podría ser útil para dichos centros de estudio.

En cuanto a la pregunta abierta sobre cómo mejorar este software los encuestados mencionaron grandes ideas ya que por obvias razones son personas calificadas y tienen gran experiencia en el área, tal es el caso del Dr. Ojeda que indicó que en su etapa de estudiante en la Facultad de Medicina él tenía que recurrir directamente a los hospitales e interactuar directamente con pacientes. El Dr. Ojeda considera que esta herramienta se podría complementar incluyendo mayor cantidad de registros patológicos, añadiendo las 12 derivaciones que son imprescindibles para un profesional que analiza la actividad eléctrica del corazón de tal forma que permita al estudiante determinar otros parámetros como lo son los ejes cardíacos. También recomienda incluir efectos de sonido acorde con cada tipo de señal electrocardiográfica permitiendo simular un entorno acorde con lo real beneficiando el aprendizaje del estudiante.

Por su parte, el Dr. Juan Pablo Román dada su experiencia como docente en el área sugiere implementar en la interfaz del estudiante un cuadro para que el estudiante argumente y describa el tipo de señal planteada por el profesor esto permitirá al médico docente evaluar los conocimientos del estudiante de forma más completa. También indica escoger adecuadamente las arritmias ya que algunas comparten las mismas características, es decir, pertenecen a un mismo grupo.

4. CONCLUSIONES Y RECOMENDACIONES

4.1. CONCLUSIONES

En este trabajo se ha implementado una aplicación que simula señales electrocardiografías y permite crear escenarios donde se puede variar propiedades como el ritmo cardíaco, amplitudes y duraciones de las ondas a los cuales un estudiante se tendrá enfrentar sin el temor de afectar a un paciente.

La utilización de esta aplicación es de gran importancia para la formación de profesionales en el área de la salud, ya que cuenta con una interfaz gráfica intuitiva y amigable que facilita la puesta en práctica de los conocimientos teóricos adquiridos, permitiendo a los estudiantes elevar su confianza ante estas situaciones clínicas y así desarrollar nuevas habilidades que le serán útiles en la vida profesional.

El modelo matemático implementado para simular la señal ECG fue un modelo básico ya que existen otros modelos matemáticos más sofisticados que describen una señal ECG. La ventaja de utilizar este modelo es que nos proporciona una forma de onda clara y fácil de interpretar a aquellas personas que inician con el aprendizaje de esta rama.

Los resultados de la validación del software con los médicos fueron satisfactorios y obtuvo una buena aceptación, debido a que esta herramienta cuenta con funcionalidades elementales que un estudiante requiere para el aprendizaje en la rama de la cardiología. Y de acuerdo con la validación realizada con los médicos se verificó fidelidad de la señal ECG simulada, dicha señal fue modelada tomando en cuenta los valores reales de una señal ECG.

También se puede mencionar que la aplicación tiene ciertas limitaciones debido a que la aplicación no es instalable y para poderlo ejecutar es necesario tener instalado de Anaconda Distribution y PyQtGraph ya que al momento de encapsular en un archivo ejecutable se obtuvieron problemas por la cantidad de dependencias que se utilizaron en la aplicación.

La sincronización fue un aspecto que se logró mejorar de cierta forma debido a que la aplicación presentaba un pequeño desface en la animación de las señales en las interfaces del profesor y del estudiante. Al principio se detectaron problemas en la sincronización de las señales, el efecto de animación de las señales cardiológicas presentaban un ligero desfase, es decir, el efecto de animación para la interfaz del estudiante se percibía a simple vista con cierta lentitud con el efecto de animación de la interfaz del profesor. Esto se solucionó modificando manualmente los tiempos de la función Update que controla la velocidad de este efecto.

4.2. RECOMENDACIONES

Este proyecto se considera como una fase preliminar y se deja abierto para la integración de nuevas funcionalidades y la potencialización de esta herramienta a través trabajos futuros. Por ejemplo, se puede ampliar la base de datos de las señales patológicas. Adicionalmente

tomando en cuenta las recomendaciones por parte de los médicos, se podría incluir un sección donde el estudiante pueda describir a la señal cardíaca que se le ha propuesto, de forma que el profesor se le facilite evaluar los conocimientos adquiridos por el estudiante.

De igual forma, se puede mejorar el módulo de medicación de manera que se tome en cuenta aspectos como el tiempo de efectos de los medicamentos y la posibilidad de combinar entre dos o mas medicamentos para el tratamiento de las arritmias.

El desarrollo de una App conectado mediante Wi-Fi o Bluetooth que permita la portabilidad de la aplicación y posibilite ejecutarlo sin necesidad de conectarse a internet.

Se deja abierta la posibilidad de implementar la conexión entre un profesor y varios alumnos. En este trabajo realizado existe la posibilidad de ampliar el numero de dispositivos que trabajen como estudiantes, pero se optó por realizar entre un profesor y un estudiante.

También se puede desarrollar una App Web de este proyecto de manera que se pueda abrir la aplicación desde cualquier navegador web y desde cualquier sistema operativo sin tener que recurrir a la instalación del software en la máquina que va a trabajar.

5. REFERENCIAS BIBLIOGRÁFICAS

[1] A. Dávila-Cervantes, "Simulación en educación médica," *Investigación en educación médica*, vol. 3, no. 10, pp. 100–105, 2014.

[2] J. Veletanga, "UCE inauguró Centro de Simulación Médica más grande del Ecuador," citado el 11/11/2019. [Online]. Disponible en: https://www.redaccionmedica.ec/secciones/profesionales/uce-inaugur-centro-de-simulaci-n-m-dica-m-s-grande-del-ecuador-87305

[3] M. Amitai Ziv, "Las simulaciones en educación médica," *Educ Médica*, vol. 10, no. 3, pp. 147–8, 2007.

[4] A. M. Katz, *Physiology of the Heart Fourth Edition*. Lippincott Williams & Wilkins, 2006. [Online]. Disponible en: https://books.google.com.ec/books?id=V1S1WU3J4ioC

[5] G. J. Tortora and B. H. Derrickson, *Principles of anatomy and physiology*. John Wiley & Sons, 2008. [Online]. Disponible en: https://books.google.com.ec/books?id=C2iBQgAACAAJ

[6] J. T. Catalano, *Guide to ECG analysis, Second Edition*. Ada, Oklahoma: Lippincott Williams & Wilkins, 2002. [Online]. Disponible en: https://books.google.com.ec/books?id=V1S1WU3J4ioC

[7] A. D. Jose and D. Collison, "The normal range and determinants of the intrinsic heart rate in man," *Cardiovascular research*, vol. 4, no. 2, pp. 160–167, 1970.

[8] J. Sundnes, G. T. Lines, X. Cai, B. F. Nielsen, K.-A. Mardal, and A. Tveito, *Computing the electrical activity in the heart*. Springer Science & Business Media, 2007, vol. 1.

[9] S. A. Jones, *ECG notes: Interpretation and Management Guide 2nd Edition*. Philadelphia, PA 19103: FA Davis, 2009. [Online]. Disponible en: https://books.google.com.ec/books?hl=es&lr=&id=TWf2AAAAQBAJ

[10] V. Autores, "HEARTe | 12 lead ECG," citado el 11/11/2019. [Online]. Disponible en: https://www.heartelearning.org/labyrinths?id=47637

[11] B. E. Jin, H. Wulff, J. H. Widdicombe, J. Zheng, D. M. Bers, and J. L. Puglisi, "A simple device to illustrate the einthoven triangle," *Advances in physiology education*, vol. 36, no. 4, pp. 319–324, 2012.

[12] K. Wesley, *Huszar. Interpretación del ECG: monitorización y 12 derivaciones: Guía práctica para la interpretación y el tratamiento, Quinta Edición*. Elsevier Health Sciences, 2017. [Online]. Disponible en: https://books.google.com.ec/books?id=USQmDwAAQBAJ

[13] O. G. Sotelo, *Manual de Arritmias Cardiacas : Guía Diagnóstica Terapéutica*. Editorial Universidad de Costa Rica, 2002. [Online]. Disponible en: https://books.google.com.ec/books?id=3epznYBfQpMC

[14] D. Martin and D. M. M. Rn, *The Complete Study Guide to Learning the Electrocardiogram*. Lulu.com, Jun. 2007. [Online]. Disponible en: https://books.google.com.ec/books?id=vm8HI4cWNEkC

[15] C. S. Education, "Atrial Rhythms | Lessons and Quiz," 2019. [Online]. Disponible en: https://ekg.academy/atrial-rhythms

[16] G. L. K. Sarah Schaeffer, "Ventricular tachycardia," 2017. [Online]. Disponible en: https://www.cancertherapyadvisor.com/home/decision-support-in-medicine/hospital-medicine/ventricular-tachycardia/

[17] M. Training and S. LLC, "Fibrilación Ventricular guía de referencia," 2019. [Online]. Disponible en: https://www.practicalclinicalskills.com/ekg-reference-guide-details-es?lessonid=26

[18] A. Jadhav, A. Ingole, and A. Chockalingam, "Ventricular ectopic beats: An overview of management considerations: Retraction," *The American journal of the medical sciences*, vol. 344, no. 3, 2012.

[19] J. Tintinalli, *Tintinallis Emergency Medicine A Comprehensive Study Guide*. McGraw-Hill Education, 2015. [Online]. Disponible en: https://lib.hpu.edu.vn/handle/123456789/32417

[20] R. B. Freire and A. M. González, "Fármacos cardiovasculares," in *Libro de la salud cardiovascular del Hospital Clínico San Carlos y la Fundación BBVA*. Fundación BBVA, 2009, pp. 87–100.

[21] L. B. Mitchel, "Drugs for Arrhythmias - Cardiovascular Disorders." [Online]. Disponible en: https://www.msdmanuals.com/professional/cardiovascular-disorders/arrhythmias-and-conduction-disorders/drugs-for-arrhythmias

[22] C. W. Callaway, "Epinephrine for cardiac arrest," *Current Opinion in Cardiology*, vol. 28, no. 1, p. 36, Jan. 2013. [Online]. Disponible en: https://journals.lww.com/co-cardiology/Fulltext/2013/01000/Epinephrine_for_cardiac_arrest.7.aspx

[23] S. Chowdhry, L. Seidenstricker, D. S. Cooney, R. Hazani, and B. J. Wilhelmi, "Do not use epinephrine in digital blocks: myth or truth? part ii. a retrospective review of 1111 cases," *Plastic and reconstructive surgery*, vol. 126, no. 6, pp. 2031–2034, 2010.

[24] G. W. Abbott and R. Levi, "Antiarrhythmic Drugs," in *Pharmacology and Physiology for Anesthesia (Second Edition)*, H. C. Hemmings and T. D. Egan, Eds. Philadelphia: Elsevier, Jan. 2019, pp. 556–574. [Online]. Disponible en: http://www.sciencedirect.com/science/article/pii/B9780323481106000272

[25] G. Mccann, “Pharmacological treatment of significant cardiac arrhythmias,” *British Journal of Sports Medicine*, vol. 34, no. 5, pp. 401–402, Oct. 2000. [Online]. Disponible en: https://bjsm.bmj.com/content/34/5/401

[26] A. A. Grace and A. J. Camm, “Quinidine,” *New England Journal of Medicine*, vol. 338, no. 1, pp. 35–45, Jan. 1998. [Online]. Disponible en: https://doi.org/10.1056/NEJM199801013380107

[27] J. B. Florek and D. Girzadas, “Amiodarone,” in *StatPearls [Internet]*. StatPearls Publishing, 2018. [Online]. Disponible en: https://www.ncbi.nlm.nih.gov/books/NBK482154/

[28] B. Lown, “Defibrillation and cardioversion,” 2002. [Online]. Disponible en: https://academic.oup.com/cardiovascres/article/55/2/220/354826

[29] A. J. Adgey, M. S. Spence, and S. J. Walsh, “Theory and practice of defibrillation:(2) defibrillation for ventricular fibrillation,” *Heart*, vol. 91, no. 1, pp. 118–125, 2005.

[30] P. J. Podrid, S. Lévy, and B. Downey, “Basic principles and techniques of cardioversion and defibrillation,” *UpToDate. Last Updated May*, vol. 28, 2009.

[31] P. S. Foundation, “General Python FAQ - Python 3.8.0 documentation,” citado el 11/12/2019. [Online]. Disponible en: https://docs.python.org/3/faq/general.html#general-information

[32] A. Inc., “Anaconda Documentation - Anaconda documentation,” citado el 11/12/2019. [Online]. Disponible en: https://docs.anaconda.com/

[33] R. C. Limited, “Introduction - PyQt v5.14b1 Reference Guide,” citado el 11/11/2019. [Online]. Disponible en: https://www.riverbankcomputing.com/static/Docs/PyQt5/introduction.html

[34] B. M. Harwani, *Qt5 Python GUI Programming Cookbook: Building responsive and powerful cross-platform applications with PyQt*. Packt Publishing Ltd, Jul. 2018. [Online]. Disponible en: https://books.google.com.ec/books?hl=es&lr=&id=YO1mDwAAQBAJ

[35] Qt Development Frameworks, “Qt designer,” 2018. [Online]. Disponible en: https://www.qt.io

[36] A. L. Goldberger, L. A. Amaral, L. Glass, J. M. Hausdorff, P. C. Ivanov, R. G. Mark, J. E. Mietus, G. B. Moody, C.-K. Peng, and H. E. Stanley, “Physiobank, physiotoolkit, and physionet: components of a new research resource for complex physiologic signals,” *Circulation*, vol. 101, no. 23, pp. e215–e220, 2000.

[37] V. Autores, “PhysioBank ATM,” citado el 11/11/2019. [Online]. Disponible en: https://archive.physionet.org/cgi-bin/atm/ATM

[38] D. A. Jaffe and J. O. Smith, “Extensions of the karplus-strong plucked-string algorithm,” *Computer Music Journal*, vol. 7, no. 2, pp. 56–69, 1983.

[39] H. Nagendra, S. Mukherjee, and V. Kumar, “Application of wavelet techniques in ecg signal processing: an overview,” *Int J Eng Sci Technol*, vol. 3, no. 10, pp. 7432–7443, 2011.

[40] G. B. Moody and R. G. Mark, “The impact of the mit-bih arrhythmia database,” *IEEE Engineering in Medicine and Biology Magazine*, vol. 20, no. 3, pp. 45–50, 2001.

[41] G. Moody, “A new method for detecting atrial fibrillation using rr intervals,” *Computers in Cardiology*, pp. 227–230, 1983.

[42] S. D. Greenwald, “The development and analysis of a ventricular fibrillation detector,” Ph.D. dissertation, Massachusetts Institute of Technology, 1986.

[43] S. D. Greenwald, R. S. Patil, and R. G. Mark, *Improved detection and classification of arrhythmias in noise-corrupted electrocardiograms using contextual information*. IEEE, 1990.

[44] F. Nolle, F. Badura, J. Catlett, R. Bowser, and M. Sketch, “Crei-gard, a new concept in computerized arrhythmia monitoring systems,” *Computers in Cardiology*, vol. 13, pp. 515–518, 1986.

[45] A. Goldberger, L. Amaral, L. Glass, J. Hausdorff, P. Ivanov, R. Mark, J. Mietus, G. Moody, C.-K. Peng, and H. Stanley, “Spontaneous Ventricular Tachyarrhythmia Database,” 2003. [Online]. Disponible en: https://physionet.org/content/mvtdb/1.0/

ANEXOS

ANEXO A. Archivos de la interfaz del Profesor.
(Se encuentran en el CD adjunto a este documento)

ANEXO B. Archivos de la interfaz del Estudiante.
(Se encuentran en el CD adjunto a este documento)

ANEXO C. Encuestas realizadas

Anexo C: Encuestas Realizadas

ENCUESTA DE SATISFACCIÓN DE USUARIO

Con el propósito de determinar la importancia y calidad del Software, deseamos conocer su punto de vista. El éxito del Software depende de su objetividad y colaboración. Por favor complete el siguiente cuestionario.

1. **¿Qué tan satisfecho/a está con la facilidad de uso de este software?**

   ◯ Extremadamente satisfecho/a

   ⊘ Muy satisfecho/a

   ◯ Algo satisfecho/a

   ◯ No tan satisfecho/a

   ◯ Nada satisfecho/a

2. **¿Qué tan satisfecho/a está con la apariencia de este software?**

   ◯ Extremadamente satisfecho/a

   ⊘ Muy satisfecho/a

   ◯ Algo satisfecho/a

   ◯ No tan satisfecho/a

   ◯ Nada satisfecho/a

3. **La interfaz es amigable/intuitiva.**

   ⊘ Muy de acuerdo

   ◯ De acuerdo

   ◯ No estoy seguro

   ◯ En desacuerdo

   ◯ Muy en desacuerdo

4. **La organización de los menús, botones, etc. es lógica.**

   ◯ Muy de acuerdo

   ⊘ De acuerdo

   ◯ No estoy seguro

- ◯ En desacuerdo
- ◯ Muy en desacuerdo

5. ¿Los mensajes de error son adecuados y entendibles?

- ◯ Muy de acuerdo
- ⊘ De acuerdo
- ◯ No estoy seguro
- ◯ En desacuerdo
- ◯ Muy en desacuerdo

6. ¿Cree usted que el software se ejecuta lentamente?

- ◯ Muy de acuerdo
- ◯ De acuerdo
- ◯ No estoy seguro
- ⊘ En desacuerdo
- ◯ Muy en desacuerdo

7. ¿Se ha detenido inesperadamente en algún momento?

- ◯ Muy de acuerdo
- ◯ De acuerdo
- ◯ No estoy seguro
- ⊘ En desacuerdo
- ◯ Muy en desacuerdo

8. ¿La información se presenta de manera clara y entendible?

- ◯ Muy de acuerdo
- ⊘ De acuerdo
- ◯ No estoy seguro
- ◯ En desacuerdo

○ Muy en desacuerdo

9. ¿Es fácil hacer que el software haga exactamente lo que quiero?

○ Muy de acuerdo

⊘ De acuerdo

○ No estoy seguro

○ En desacuerdo

○ Muy en desacuerdo

10. No siempre responde como yo quiero.

○ Muy de acuerdo

⊘ De acuerdo

○ No estoy seguro

○ En desacuerdo

○ Muy en desacuerdo

11. Está de acuerdo con que esta herramienta sea utilizada para el entrenamiento médico?

⊘ Muy de acuerdo

○ De acuerdo

○ No estoy seguro

○ En desacuerdo

○ Muy en desacuerdo

12. ¿Tiene algunas ideas sobre cómo mejorar este software?

Observaciones de forma en la ficha estudiante colocar las gestiones de las patologías para selección y un espacio para argumentar la propuesta.
Observación de fondo podría escoger de mejor manera el tipo de amputación

ENCUESTA DE SATISFACCIÓN DE USUARIO

Con el propósito de determinar la importancia y calidad del Software, deseamos conocer su punto de vista. El éxito del Software depende de su objetividad y colaboración. Por favor complete el siguiente cuestionario.

1. **¿Qué tan satisfecho/a está con la facilidad de uso de este software?**

   ◯ Extremadamente satisfecho/a

   ◯ Muy satisfecho/a

   ◯ Algo satisfecho/a

   ◯ No tan satisfecho/a

   ◯ Nada satisfecho/a

2. **¿Qué tan satisfecho/a está con la apariencia de este software?**

   ◯ Extremadamente satisfecho/a

   ◯ Muy satisfecho/a

   ◯ Algo satisfecho/a

   ◯ No tan satisfecho/a

   ◯ Nada satisfecho/a

3. **La interfaz es amigable/intuitiva.**

   ◯ Muy de acuerdo

   ◯ De acuerdo

   ◯ No estoy seguro

   ◯ En desacuerdo

   ◯ Muy en desacuerdo

4. **La organización de los menús, botones, etc. es lógica.**

   ◯ Muy de acuerdo

   ◯ De acuerdo

   ◯ No estoy seguro

- ◯ En desacuerdo
- ◯ Muy en desacuerdo

5. ¿Los mensajes de error son adecuados y entendibles?

- ◯ Muy de acuerdo
- ◯ De acuerdo
- ◯ No estoy seguro
- ◯ En desacuerdo
- ◯ Muy en desacuerdo

6. ¿Cree usted que el software se ejecuta lentamente?

- ◯ Muy de acuerdo
- ◯ De acuerdo
- ◯ No estoy seguro
- ◯ En desacuerdo
- ◯ Muy en desacuerdo

7. ¿Se ha detenido inesperadamente en algún momento?

- ◯ Muy de acuerdo
- ◯ De acuerdo
- ◯ No estoy seguro
- ◯ En desacuerdo
- ◯ Muy en desacuerdo

8. ¿La información se presenta de manera clara y entendible?

- ◯ Muy de acuerdo
- ◯ De acuerdo
- ◯ No estoy seguro
- ◯ En desacuerdo

- ○ Muy en desacuerdo

9. **¿Es fácil hacer que el software haga exactamente lo que quiero?**

- ⊘ Muy de acuerdo
- ○ De acuerdo
- ○ No estoy seguro
- ○ En desacuerdo
- ○ Muy en desacuerdo

10. **No siempre responde como yo quiero.**

- ○ Muy de acuerdo
- ⊘ De acuerdo
- ○ No estoy seguro
- ○ En desacuerdo
- ○ Muy en desacuerdo

11. **Está de acuerdo con que esta herramienta sea utilizada para el entrenamiento médico?**

- ⊘ Muy de acuerdo
- ○ De acuerdo
- ○ No estoy seguro
- ○ En desacuerdo
- ○ Muy en desacuerdo

12. **¿Tiene algunas ideas sobre cómo mejorar este software?**

COMPLEMENTAR: OBTENER EL EJE, MAYOR CANTIDAD DE REGISTROS PATOLOGICOS SONIDO CARDIACO, COMPLENTAR CON LAS 12 DERIVACIONES.

ORDEN DE EMPASTADO