UNIVERSIDAD TÉCNICA DEL NORTE

FACULTAD DE INGENIERÍA EN CIENCIAS APLICADAS

CARRERA DE INGENIERÍA EN MECATRÓNICA

TRABAJO DE GRADO PREVIO A LA OBTENCIÓN DEL TÍTULO DE INGENIERO EN MECATRÓNICA

TEMA

"EVALUADOR MIOELÉCTRICO MULTICANAL PARA PERSONAS CON AMPUTACIÓN TRANSTIBIAL"

AUTORA: Katherine Johana García Fierro

DIRECTORA: MSc. Ing. Luz María Tobar Contento

Ibarra – Ecuador

2019

UNIVERSIDAD TÉCNICA DEL NORTE

BIBLIOTECA UNIVERSITARIA

AUTORIZACIÓN DE USO Y PUBLICACIÓN

A FAVOR DE LA UNIVERSIDAD TÉCNICA DEL NORTE

1. IDENTIFICACIÓN DE LA OBRA

En cumplimiento del Art. 144 de la Ley de Educación Superior, hago la entrega del presente trabajo a la Universidad Técnica del Norte para que sea publicado en el Repositorio Digital Institucional, para lo cual pongo a disposición la siguiente información:

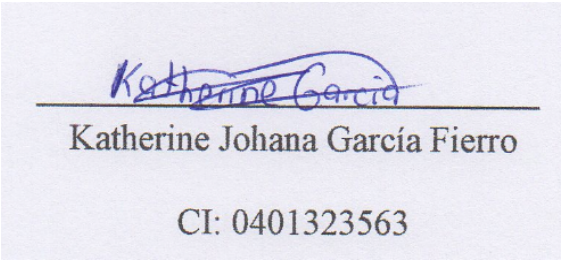
DATOS DE CONTACTO			
CÉDULA DE IDENTIDAD:	0401323563		
APELLIDOS Y NOMBRES:	García Fierro Katherine Johana		
DIRECCIÓN:	Ibarra, Pilanquí		
EMAIL:	kjgarcia@utn.edu.ec		
TELÉFONO FIJO:	062 953079	**TELÉFONO MÓVIL:**	0984113241

DATOS DE LA OBRA	
TÍTULO:	Evaluador Mioeléctrico Multicanal para Personas con Amputación Transtibial
AUTOR (ES):	García Fierro Katherine Johana
FECHA:	6/08/2019
PROGRAMA:	PREGRADO
TITULO POR EL QUE OPTA:	Ingeniera en Mecatrónica
ASESOR /DIRECTOR:	MSc. Ing. Luz María Tobar Subía Contento

2. CONSTANCIAS

El autor manifiesta que la obra objeto de la presente autorización es original y se la desarrolló, sin violar derechos de autor de terceros, por lo tanto, la obra es original y que es el titular de los derechos patrimoniales, por lo que asume la responsabilidad sobre el contenido de la misma y saldrá en defensa de la Universidad en caso de reclamación por parte de terceros.

Ibarra, a los 6 días del mes de agosto de 2019

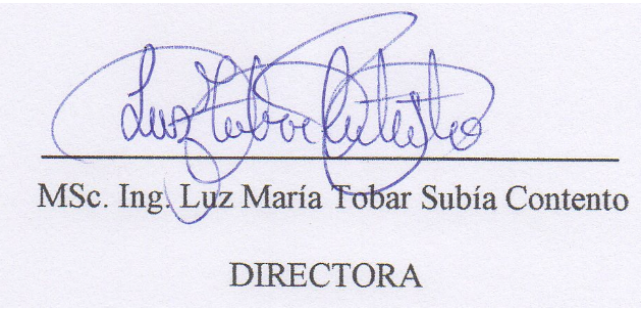


UNIVERSIDAD TÉCNICA DEL NORTE

FACULTAD DE INGENIERÍA EN CIENCIAS APLICADAS

CERTIFICACIÓN

Certifico que la tesis previa a la obtención del título de Ingeniera en Mecatrónica con el tema: "EVALUADOR MIOELÉCTRICO MULTICANAL PARA PERSONAS CON AMPUTACIÓN TANSTIBIAL", ha sido desarrollado y terminado en su totalidad por el Srta. Katherine Johana García Fierro, con cédula de identidad 040132356-3, bajo mi supervisión para lo cual firmo en constancia.



MSc. Ing. Luz María Tobar Subía Contento

DIRECTORA

AGRADECIMIENTO

Agradezco al destino, por haberme brindado una buena vida, experiencias que forjaron mi carácter, me permitieron aprender y ser feliz.

A mis padres Lilian Fierro y Juan Carlos García por ser pilares en mi vida, brindándome fortaleza para enfrentar las dificultades, apoyando mis decisiones y guiándome con cariño a través de ella.

A mi familia por apoyarme en cualquier situación, compartir varias experiencias conmigo y brindarme todo su cariño.

A mi amiga Gaby por compartir todas las locuras, emociones, alegrías y tristezas durante estos 9 años. Apoyarme y acompañarme en las dificultades.

A mis amigos Danny, Katy, Daniela, Mallus y Cristian por haber puesto el buen ambiente laboral durante mis años de universidad y permitirme disfrutarlos.

A Martin por apoyarme, escucharme, compartir y alegrar cada día durante estos años.

A la ing. Luz María Tobar por su ayuda durante la elaboración de este trabajo, aportando con sus conocimientos y la elaboración de diversos trámites.

A la fundación Prótesis Imbabura por asesorarme y colaborar con la culminación de este trabajo.

Al fisioterapeuta Jorge Zambrano docente de la carrera de fisioterapia UTN, por brindar asistencia técnica en la realización de las pruebas.

DEDICATORIA

Le dedico este trabajo a mis queridos padres Lilian y Juan Carlos, ya que todos mis logros han sido posibles gracias a su incansable apoyo, cariño y guía.

Y a mis hermanas Tanya y Nayely, que han sido una de las motivaciones más importantes para enfrentar los problemas.

RESUMEN

Las personas que pierden parcial o totalmente una extremidad realizan un proceso de rehabilitación que inicia desde el momento de su lesión hasta que utilizan una prótesis, procurando mantener condiciones físicas y psicológicas adecuadas para su adaptación. Los fisioterapeutas que se encargan de diagnosticar emplean metodologías manuales al realizar la evaluación de la condición del paciente, requiere mayor pericia y experiencia de este.

El trabajo consiste en la construcción de una herramienta tecnológica, que ayude al fisioterapeuta en el proceso de evaluación de la condición muscular en reposo, de personas con amputación transtibial, para lo cual se implementó un sistema de adquisición multicanal (tarjeta de biosensado) y se desarrolló un software que se adapte al mismo.

El sistema permite recolectar la información del paciente en una base de datos, adquirir, procesar y visualizar las señales electromiográficas provenientes de los músculos flexores y extensores de rodilla, para realizar una evaluación de fuerza muscular y un registro cronológico de la misma.

En la valoración se utiliza un protocolo comparativo. Donde, se establece el nivel de fuerza muscular de la extremidad sana en mV referenciando como normal (100%). Luego, se contrasta el de la extremidad amputada con este. Retornando un resultado porcentual (0-100) %, una puntuación cualitativa (normal, bien, regular, mal, escaso, nulo) en concordancia con la escala de Daniels; y un cuadro evolutivo de las medidas del paciente.

Se realizaron pruebas con personas sanas y amputadas. El promedio de la fuerza muscular concordó con el criterio del fisioterapeuta. Sin embargo, independientemente cada musculo marcará valores diferentes. Esto permite un análisis más a fondo y marca el inicio de otros estudios como la evaluación en marcha.

ABSTRACT

People who partially or totally lose a limb perform a rehabilitation process that begins from the moment of their injury until they use a prosthesis, trying to maintain adequate physical and psychological conditions for adaptation. The physiotherapists who are responsible for diagnosing employ manual methodologies to perform the evaluation of the patient's condition, requires greater expertise and experience of this.

The work consists in the construction of a technological tool that helps the physiotherapist in the process of assessing the resting muscle condition of people with transtibial amputation, so a multichannel acquisition system (biosensing card) was implemented and developed software that suits it.

The system allows the collection of patient information in a database, to acquire, process and visualize the electromyographic signals coming from the knee flexor and extensor muscles, to perform an evaluation of muscular strength and a chronological record of it.

The evaluation uses a comparative protocol. Where, the level of muscular strength of the healthy limb is established in mV by referencing as normal (100%). Then, the amputated limb is contrasted with it. Returning a percentage result (0-100) %, a qualitative score (normal, good, fair, bad, poor, null) in accordance with the Daniels scale; and an evolutionary picture.

Tests were performed with healthy and amputated people. The average muscle strength agreed with the physiotherapist's criteria, however, independently each muscle could mark different values. This allows a more in-depth analysis and marks the beginning of other studies.

ÍNDICE DE CONTENIDO

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

INTRODUCCIÓN

Planteamiento del problema

Las personas con amputación del miembro inferior sufren el síndrome de des acondicionamiento físico, que comprende un deterioro metabólico y sistemático del organismo como consecuencia de la inmovilización prolongada. En el sistema musculoesquelético se produce una pérdida de masa y fuerza muscular, en cuanto a la morfología se genera una atrofia por falta de uso muscular [1]; los músculos pierden flexibilidad y elasticidad.

"Los tratamientos pre protésicos buscan conseguir una independencia funcional respecto a los autocuidados y la movilidad" [2]. Sin embargo, este proceso llega a ser doloroso e incómodo, con un grado de dificultad elevado. Esto conlleva a que las personas dejen de utilizar prótesis o abandonen el proceso de rehabilitación [3]. La condición que enfrentan estas personas enmarca múltiples variables que van desde el rechazo social, psicológico y familiar hasta depresión y suicidio [4].

El fisioterapeuta necesita realizar una evaluación de la función muscular, que requiere parámetros como fuerza y potencia. Son investigados mediante diferentes metodologías, que van de las pruebas musculares manuales, que son las más simples pero imprecisas (requieren mayor pericia), a evaluaciones isocinéticas. El estudio de las señales electromiográficas analiza su comportamiento, contribuyendo al diagnóstico de posibles trastornos neuromusculares, al control de prótesis inteligentes y al proceso de rehabilitación muscular.

Según el CONADIS, a nivel nacional el 46,65% de las personas con discapacidad poseen una discapacidad física, de las cuales 56,99% son hombres y 52,84% se encuentran en el rango de edad de 30-65 años. De este porcentaje, con 4955 personas registradas, Imbabura representa el 2,44% [5]. En Imbabura, 255 personas poseen discapacidad a causa de accidentes de trabajo,

351 debido a accidentes de tránsito, 4232 por enfermedades congénitas y 3995 por enfermedad adquirida [6]. Las amputaciones de la extremidad inferior se consideran más importantes ya que su incidencia es del 85% de las realizadas [4].

Una investigación de las señales electromiográficas de los músculos del miembro inferior de personas con amputación transtibial durante la evaluación de función muscular optimizara el proceso de recuperación de las mismas.

Objetivos

Objetivo General

Construir un sistema que evalué la condición muscular de personas con amputación transtibial mediante el análisis de señales electromiográficas.

Objetivos Específicos

- Definir los músculos que intervienen en la flexión y extensión del miembro inferior y parámetros mínimos para la evaluación.
- Determinar una metodología para la adquisición y análisis de las señales electromiográficas.
- Implementar el sistema de adquisición multicanal.
- Desarrollar un software para el registro cronológico de las señales obtenidas a partir del sistema de adquisición multicanal.

Justificación

Las personas que pierden parcial o totalmente una extremidad realizan un proceso de rehabilitación que inicia desde el momento de su lesión hasta que utilizan una prótesis,

procurando mantener condiciones físicas y psicológicas adecuadas para su adaptación. Sin embargo, este proceso tiende a ser doloroso, incómodo y difícil. Ocasionando el abandono de este. Los fisioterapeutas que se encargan de diagnosticar emplean metodologías manuales para realizar la evaluación de la condición del paciente, lo que requiere mayor pericia y experiencia. Al ser un proceso sin retroalimentación, lo convierte en algo poco preciso, que conlleva tratamientos más demorosos.

Este trabajo surge del proyecto de tesis "Entrenador mioeléctrico para miembro superior e inferior", realizado por la Universidad Técnica del Norte. Se busca desarrollar una investigación sobre la evaluación de la función muscular empleando el análisis de señales electromiográficas, determinando una metodología para su adquisición y procesamiento, para desarrollar un software que utilice el fisioterapeuta.

"Según el objetivo 3 del Plan Nacional del Buen Vivir que es mejorar la calidad de vida de la población, este proyecto se relaciona con la creación de soluciones que ayuden a mejorar las necesidades de personas con discapacidad motriz o que necesiten rehabilitación en sus músculos y así motivar su inclusión en la sociedad" [3].

"Al desarrollar este proyecto se aprovechará los recursos y talento humano local, se permitirá alinearlo a la política del gobierno a través de la investigación, generación y beneficio del conocimiento con finalidad social y productiva. Además, esta investigación será un precedente para futuros proyectos relacionados al agotamiento muscular [3].

Alcance

En el proyecto de tesis se definirán los músculos que intervienen en los movimientos: flexión y extensión del miembro inferior y los parámetros necesarios para la evaluación de la condición muscular. Se describirá el procedimiento para la toma y análisis de las señales

electromiográficas. Con el uso de la tarjeta de adquisición se tomará las señales de los múltiples canales, que posteriormente serán registradas y tratadas por el software. En referencia al mismo, tendrá una pantalla de inicio, una pantalla de registro de pacientes, acceso a la información de cada paciente y una pantalla para la evaluación. La muestra para las pruebas será en personas sanas y un mínimo de 3 personas con amputación transtibial.

Hipótesis

Los resultados del protocolo propuesto concuerdan con la evaluación de la condición muscular realizada por el fisioterapeuta.

1. MARCO TEÓRICO

1.1. Anatomía y biomecánica del miembro inferior

La extremidad inferior se divide en región glútea, muslo, pierna y pie en función de las principales articulaciones, los componentes óseos y las referencias superficiales [7].

1.1.1. Funciones del miembro inferior

Una función fundamental de la extremidad inferior es soportar el peso del cuerpo con un gasto mínimo de energía. Cuando se está en posición erecta, el centro de gravedad es anterior al borde de la vértebra Sil de la pelvis (Figura 1.1). La línea vertical que pasa a través del centro de gravedad es ligeramente posterior a las articulaciones de la cadera y anterior a las de rodilla y tobillo [7].

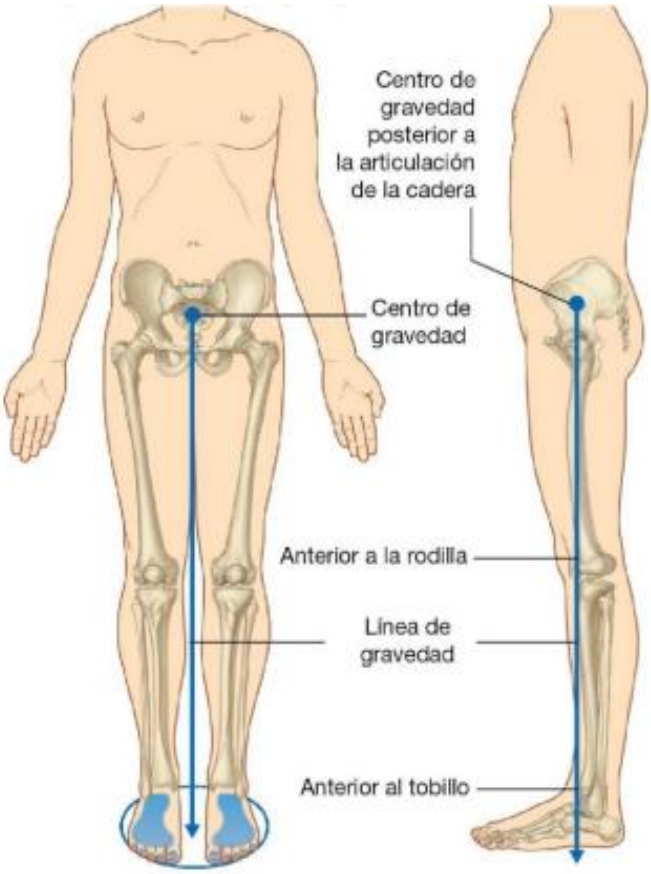


Figura 1.1. Centro y línea de gravedad [7].

La organización de los ligamentos y la forma de las articulaciones de cadera y rodilla facilita su “bloqueo” en posición erecta, reduciendo la energía muscular necesaria para mantener la bipedestación [7].

Una segunda función esencial es mover el cuerpo a través del espacio. Esto implica la integración de los movimientos de todas las articulaciones de la extremidad inferior para poner el pie sobre el suelo y mover el cuerpo sobre el pie [7].

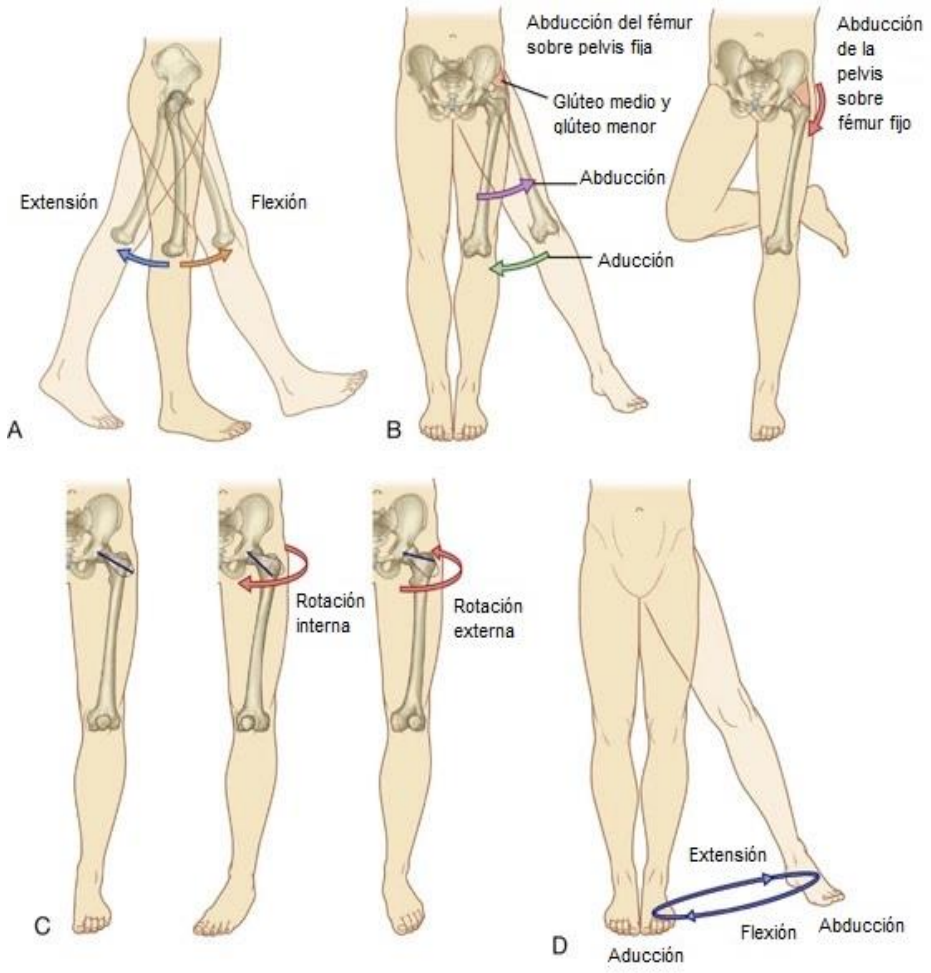


Figura 1.2. Movimientos de la articulación de la cadera. A. Flexión y extensión. B. Abducción y aducción. C. Rotaciones lateral y medial. D. Circunducción [7].

Los movimientos de la articulación de la cadera son la flexión, la extensión, la abducción, la aducción, las rotaciones medial y lateral y la circunducción (Figura 1.2). Las articulaciones

de la rodilla y el tobillo son sobre todo de tipo bisagra (gínglimo). Los principales movimientos de la rodilla son la flexión y la extensión. Los movimientos del tobillo son la flexión dorsal (movimiento de la cara dorsal del pie hacia la pierna) y la flexión plantar (Figura 1.3) [7].

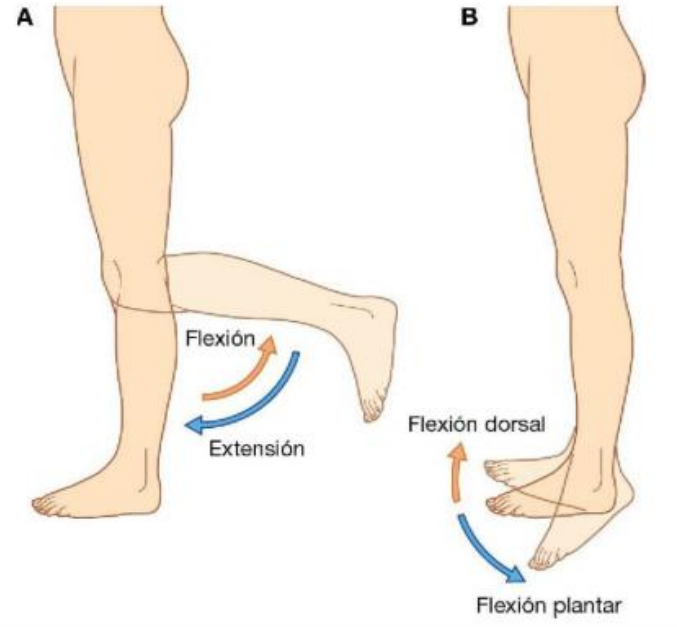


Figura 1.3. Movimientos de la rodilla y del tobillo. A. Flexión y extensión de la rodilla. B. Flexión dorsal y flexión plantar del tobillo [7].

1.1.2. Sistema muscular del muslo

Los músculos del muslo están organizados en tres compartimentos, mediante tabiques intermusculares. Los compartimentos son anterior o extensor, medial o aductor, y posterior o flexor, denominaciones que reciben según su localización o acción en la articulación de la rodilla [8].

1.1.2.1. Músculos anteriores del muslo

Los músculos anteriores del muslo son el pectíneo, el iliopsoas, el sartorio y el cuádriceps femoral (Figura 1.4).

Los músculos principales del compartimento anterior tienden a atrofiarse (disminuir) rápidamente con la enfermedad, y tras la inmovilización del muslo o la pierna suele necesitarse fisioterapia para restablecer la fuerza, el tono y la simetría con el miembro opuesto [8].

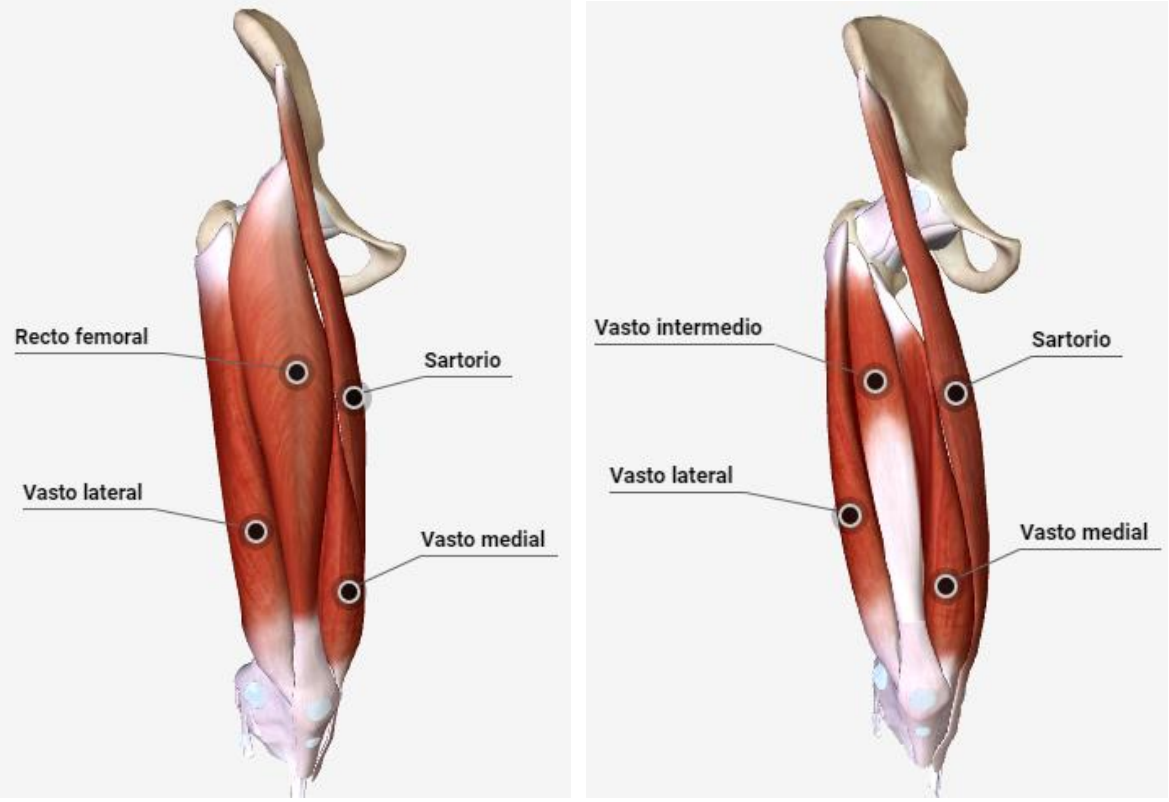


Figura 1.4. Músculos anteriores del muslo: extensores de la rodilla [9].

El cuádriceps es el mayor músculo extensor de la pierna (Tabla 1.1). La contracción concéntrica para extender la rodilla contra la fuerza de gravedad es importante al levantarse desde la posición sentada o en cuclillas, en la escalada y al subir escaleras, así como para la aceleración y proyección (carrera y salto) cuando está elevando o moviendo el peso corporal. En consecuencia, es tres veces más potente que su grupo muscular antagonista: los isquiotibiales [8].

En la marcha en llano, los cuádriceps están activos durante la finalización de la fase de oscilación, preparando la rodilla para aceptar el peso. Es el principal encargado de absorber el

impacto de la vibración del golpe de talón, y su actividad continúa cuando se asume el peso durante la fase de apoyo inicial (respuesta de carga) [8]. Se compone de:

- **Recto femoral.** El músculo recto femoral recibe este nombre por su trayecto recto descendente a lo largo del muslo. Cruza dos articulaciones, por lo que es capaz de flexionar el muslo en la articulación coxal y de extender la pierna en la articulación de la rodilla. Actúa durante las fases de preoscilación y oscilación inicial de la marcha [8].
- **Músculos vastos.** Los nombres de los tres grandes músculos vastos indican su posición alrededor del cuerpo del fémur, por tanto, son vasto lateral, medial e intermedio. Es difícil aislar la función de los tres músculos vastos [8].

Tabla 1.1. Músculos anteriores del muslo: Extensores de la rodilla [8].

Musculo	Inserción proximal	Inserción distal	Inervación	Acción principal
		Cuádriceps femoral		
Recto femoral	Espina iliaca anterior inferior e ilion, superior al acetábulo	Por medio del tendón común e intersecciones independientes en la base de la rótula; indirectamente, a través del ligamento rotuliano en la tuberosidad de la tibia; los vastos medial y lateral también se insertan en la tibia y en la rótula por medio de aponeurosis	Nervio femoral L2, L3, L4	Extienden la pierna en la articulación de la rodilla; el recto femoral. Estabiliza la articulación coxal y ayuda al iliopsoas a flexionar el muslo
Vasto lateral	Trocánter mayor y labio lateral de la línea áspera			
Vasto medial	Línea intertrocantérea y labio medial de la línea áspera			
Vasto intermedio	Caras anterior y lateral del cuerpo del fémur			

1.1.2.2. Región posterior del muslo

Tres de los cuatro músculos de la cara posterior del muslo (Figura 1.5) son músculos isquiotibiales: semitendinoso, semimembranoso y bíceps femoral (cabeza larga).

Las dos acciones de los isquiotibiales no deben realizarse de forma máxima todo el tiempo; la flexión completa de la rodilla (Tabla 1.2) requiere tal acortamiento de los isquiotibiales que no proporciona la contracción adicional que sería necesaria para la extensión completa simultánea del muslo; del mismo modo, la extensión completa de la cadera acorta los isquiotibiales de modo que no se contraen más para actuar totalmente sobre la rodilla [8].

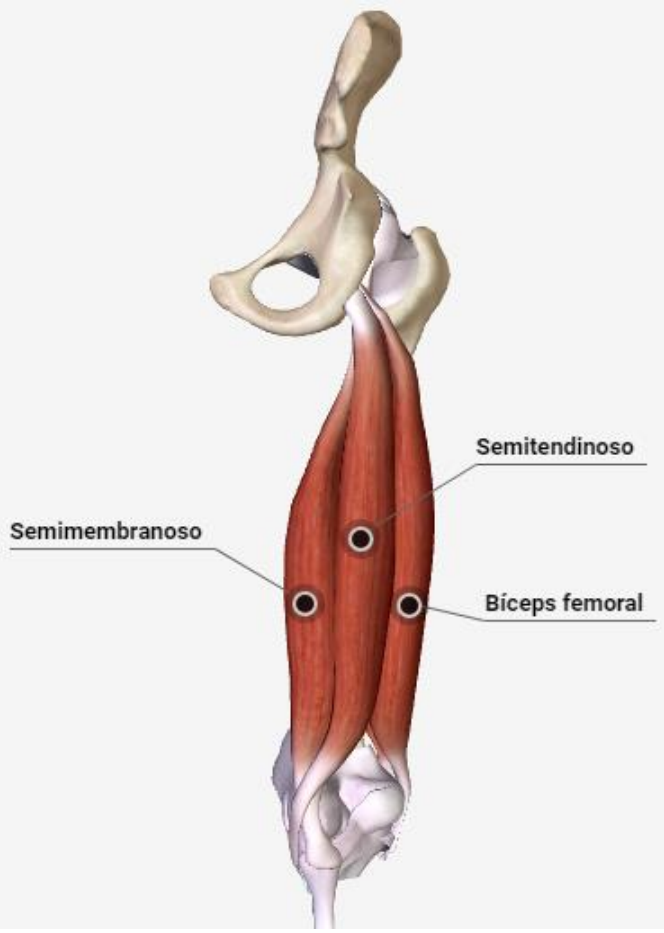


Figura 1.5. Músculos de la región posterior del muslo: extensores de la cadera y flexores de la rodilla [9].

Cuando los muslos y las piernas están fijos, los isquiotibiales ayudan a extender el tronco en la articulación coxal. Son activos en la extensión del muslo en cualquier situación, salvo en la flexión completa de la rodilla, incluyendo el mantenimiento de la postura erguida relajada. Una persona con parálisis de los isquiotibiales tiende a caer hacia delante porque los glúteos mayores no mantienen el tono muscular necesario para estar en pie en posición erguida [8].

Tabla 1.2. Músculos de la región posterior del muslo: extensores de la cadera y flexores de la rodilla [8].

Musculo	Inserción proximal	Inserción distal	Inervación	Acción principal
Semitendinoso	Tuberosidad isquiaca	Cara medial de la parte superior de la tibia	Componente tibial del nervio isquiático (L5, S1, S2)	Extienden el muslo; flexiona la pierna y la rotan medialmente cuando el muslo y la pierna están flexionados, estos músculos pueden extender el tronco
Semimembranoso		Parte posterior del cóndilo medial de la tibia; la inserción refleja la forma el ligamento poplíteo oblicuo (hacia el cóndilo lateral del fémur)		
Bíceps femoral	Cabeza larga: tuberosidad isquiática. Cabeza corta: línea áspera y línea supracondílea lateral del fémur	Lado lateral de la cabeza de la fíbula; el tendón está dividido en este punto por el ligamento colateral fibular de la rodilla	Cabeza larga: componente tibial del nervio isquiático (L5, S1, S2).	Flexiona la pierna y la rota lateralmente cuando la rodilla esta flexionada; extiende el muslo

1.1.2.3. Inervación motora del miembro inferior

Cada uno de los principales grupos musculares o compartimentos de las extremidades inferiores (Figura 1.6) está inervado sobre todo por uno o más de los nervios principales que se originan en los plexos lumbar y sacro [7]:

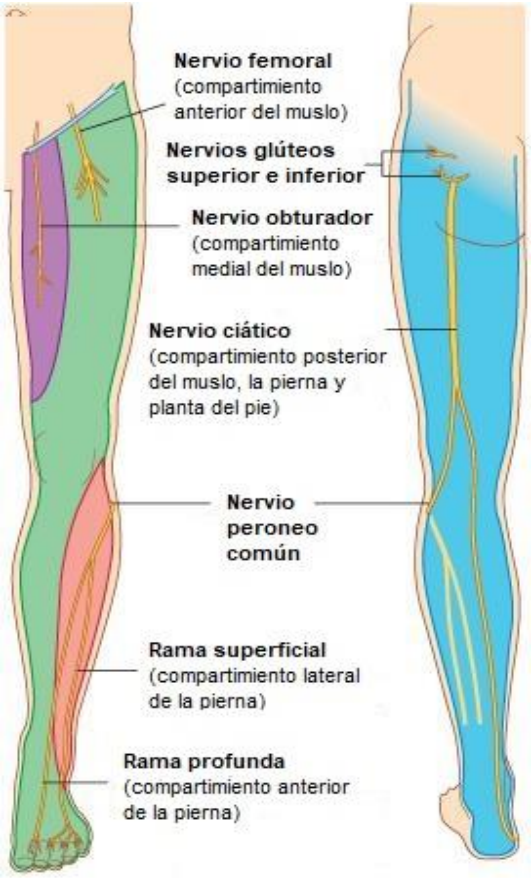


Figura 1.6. Principales nervios de la extremidad inferior (Los colores indican regiones de inervación motora) [7].

- Los grandes músculos de la región glútea (celeste) están inervados por los nervios glúteos superior e inferior [7].
- La mayoría de los músculos del compartimento anterior del muslo (verde) están inervados por el nervio femoral (excepto el tensor de la fascia lata, inervado por el nervio glúteo superior) [7].

- La mayoría de los músculos del compartimento medial (morado) están inervados por el nervio obturador (salvo el pectíneo, inervado por el nervio femoral, y parte del aductor mayor, inervado por la división tibial del nervio ciático) [7].
- La mayoría de los músculos del compartimento posterior del muslo y de la pierna (celeste), así como los de la planta del pie, están inervados por la porción tibial del nervio ciático (excepto la cabeza corta del bíceps femoral en la región posterior del muslo, inervada por la división peronea común del nervio ciático) [7].
- Los compartimentos anterior y lateral de la pierna y los músculos asociados con la superficie dorsal del pie (rojo) están inervados por la porción peronea común del nervio ciático [7].

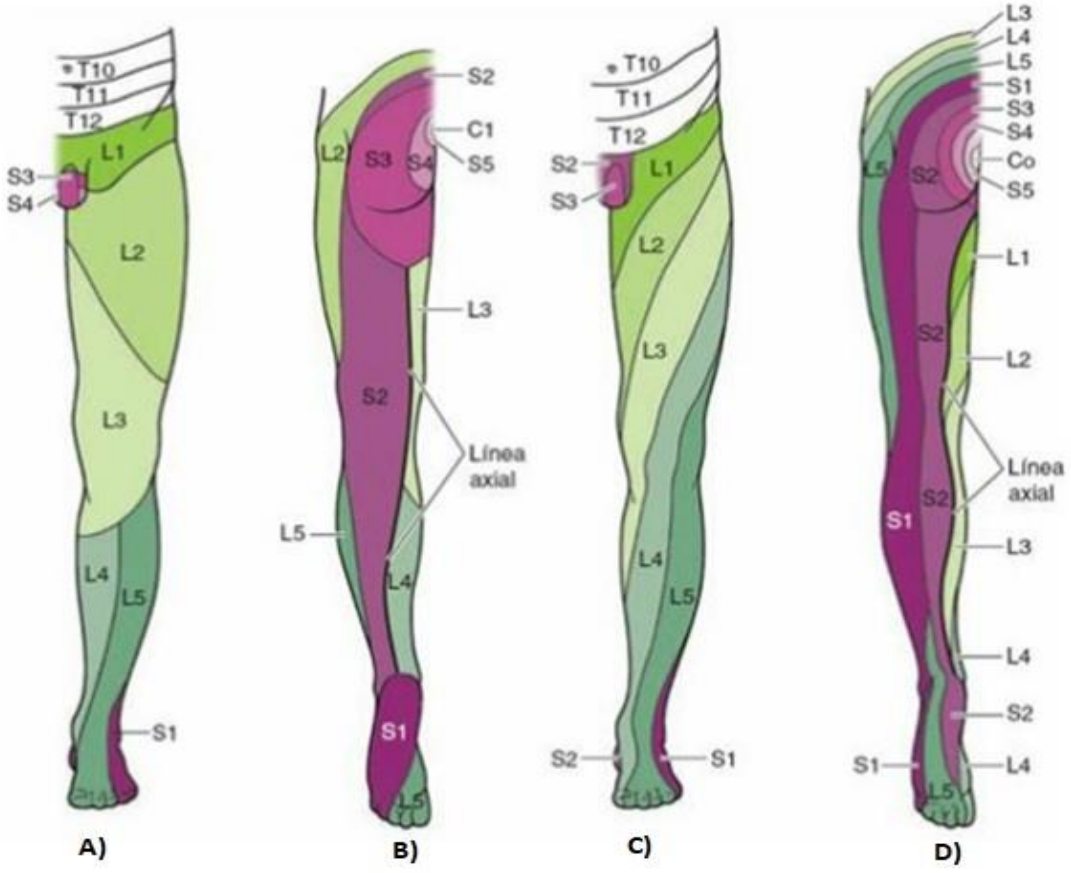


Figura 1.7. Dermatomas del miembro inferior. A y B) El patrón de dermatomas del miembro inferior de Foerster, C y D) El patrón de dermatomas de Keegan y Garrett [8].

Suelen utilizarse dos mapas diferentes de dermatomas. Muchos prefieren el patrón de dermatomas del miembro inferior de Foerster (1933), por su relación con los signos clínicos (figura 7, A y B). Otros prefieren el patrón de dermatomas de Keegan y Garrett (1948), por su uniformidad estética y su evidente relación con el desarrollo (Figura 1.7, C y D). Aunque se representan como zonas delimitadas, los dermatomas se superponen de manera considerable, excepto a lo largo de la línea axial [8].

En este caso se utiliza el patrón de Foerster (Figura 1.8) en donde:

- La flexión de la cadera está controlada sobre todo por L1 y L2.
- La extensión de la rodilla por L3 y L4.
- La flexión de la rodilla por L5 a S2.
- La flexión plantar del pie por S1 y S2.
- La aducción de los dedos por S2 y S3 [7].

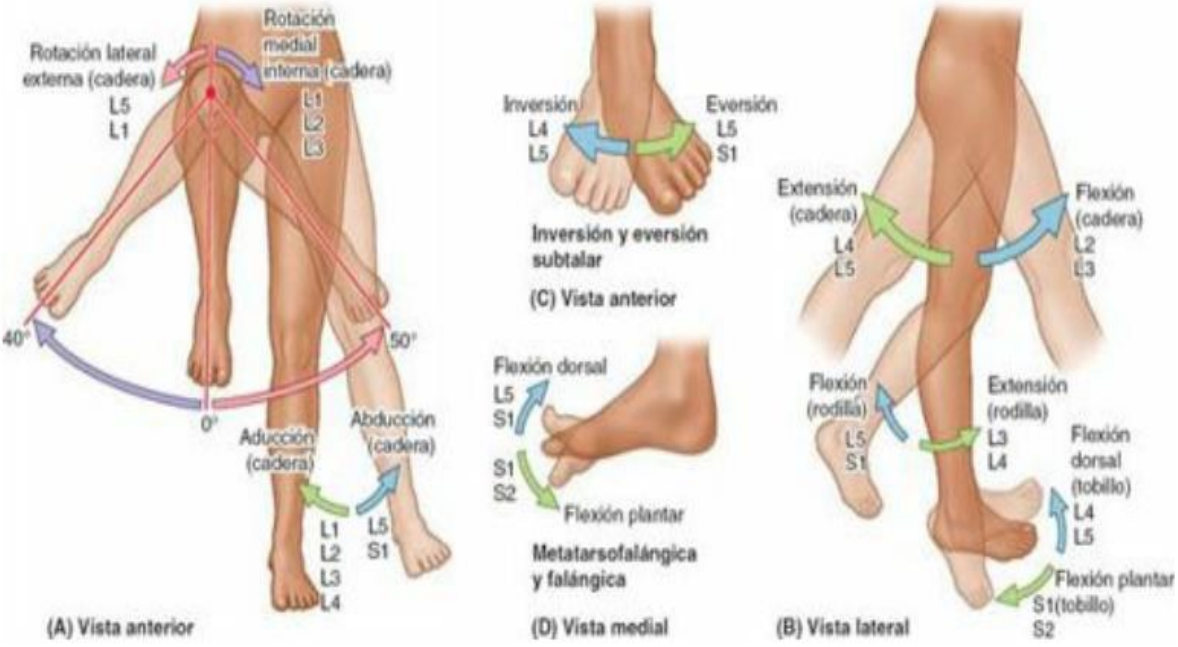


Figura 1.8. Miotomas: inervación segmentaria de los grupos musculares y movimientos del miembro inferior [8].

De esta forma, para conseguir la mejor respuesta de contracción muscular en un estudio electromiográfico se debe buscar el punto o placa de inervación del musculo. No es adecuado pensar que cada musculo posee un único punto motor, pues en ocasiones son varios. Si la zona del musculo es pequeña y el electrodo grande, responderán varios músculos o el grupo muscular. Las respuestas aisladas requieren un electrodo puntual o pequeño. Además, la localización precisa del mejor punto motor [10].

El electrodo pequeño es más selectivo, pero necesita mayor intensidad para conseguir la misma respuesta de uno grande. La resistencia de los electrodos no tiene mayor importancia siempre que el equipo de aplicación trabaje en intensidad constante [10].

En la Figura 1.9, las superficies negras cuadradas representan los puntos motores musculares, mientras que los círculos reflejan algunos puntos motores nerviosos.

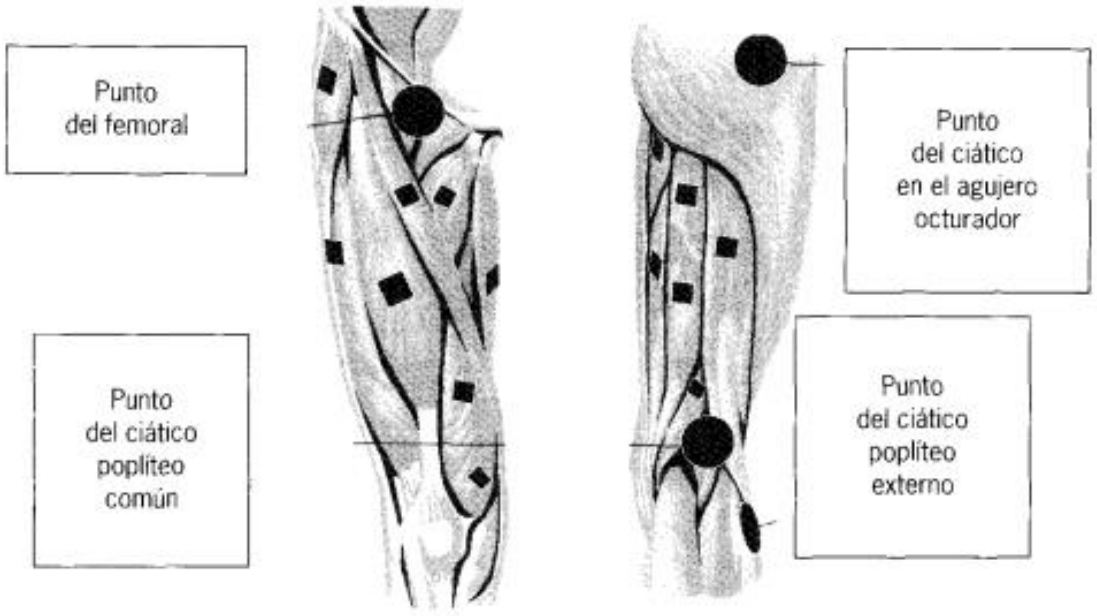


Figura 1.9. Puntos motores musculares y nerviosos del muslo [10].

Por tanto, en la Figura 1.10 se definen los mejores puntos motores para la colocación de los electrodos.

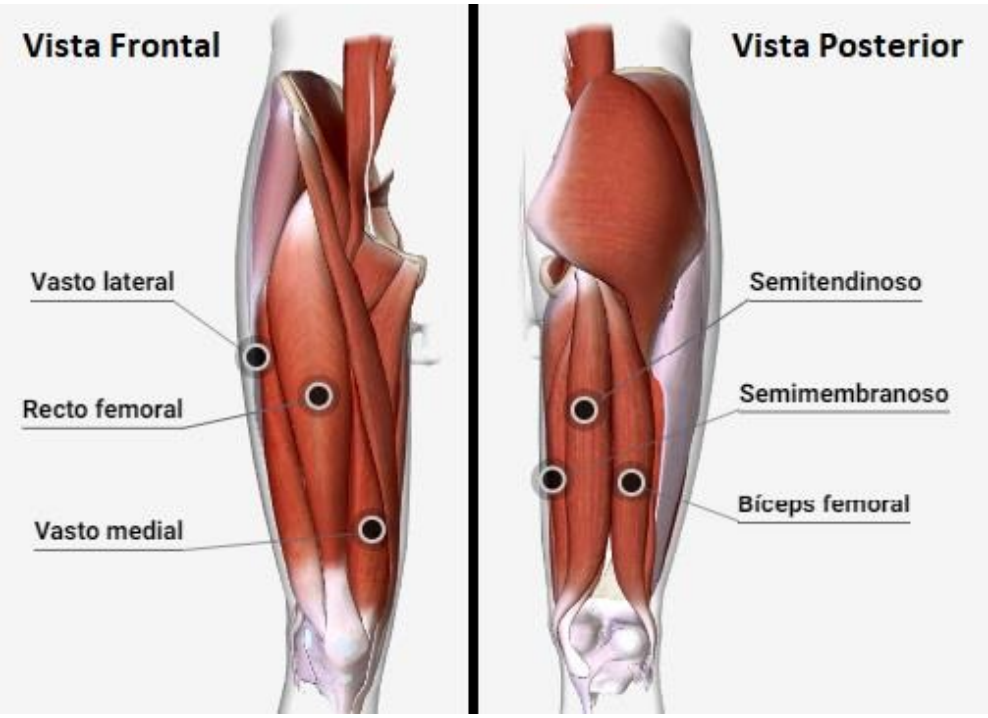


Figura 1.10. Puntos motores de los músculos superficiales del cuádriceps e isquiotibial [10].

1.2. Amputación del miembro inferior

La amputación es el procedimiento quirúrgico que consiste en la remoción, extirpación o resección de una parte o la totalidad de una extremidad a través de una o más estructuras óseas, en forma perpendicular al eje longitudinal del miembro. Cuando se efectúa a través de una interlínea articular se denomina desarticulación [4].

Tiene 2 metas: 1) Remover la porción de la extremidad para eliminar el estado patológico. 2) Crear un órgano distal optimo, desde el punto de vista motor y sensitivo, para el manejo protésico y la restauración de la función [1].

1.2.1. Niveles de amputación del miembro inferior

Los niveles son los lugares de amputación con el fin de obtener un muñón útil para la colocación de una prótesis (Figura 1.11). El nivel de amputación tiene que ser lo más distal

posible ya que la función de los muñones de amputación se reduce de forma progresiva al subir el nivel de la amputación [4].

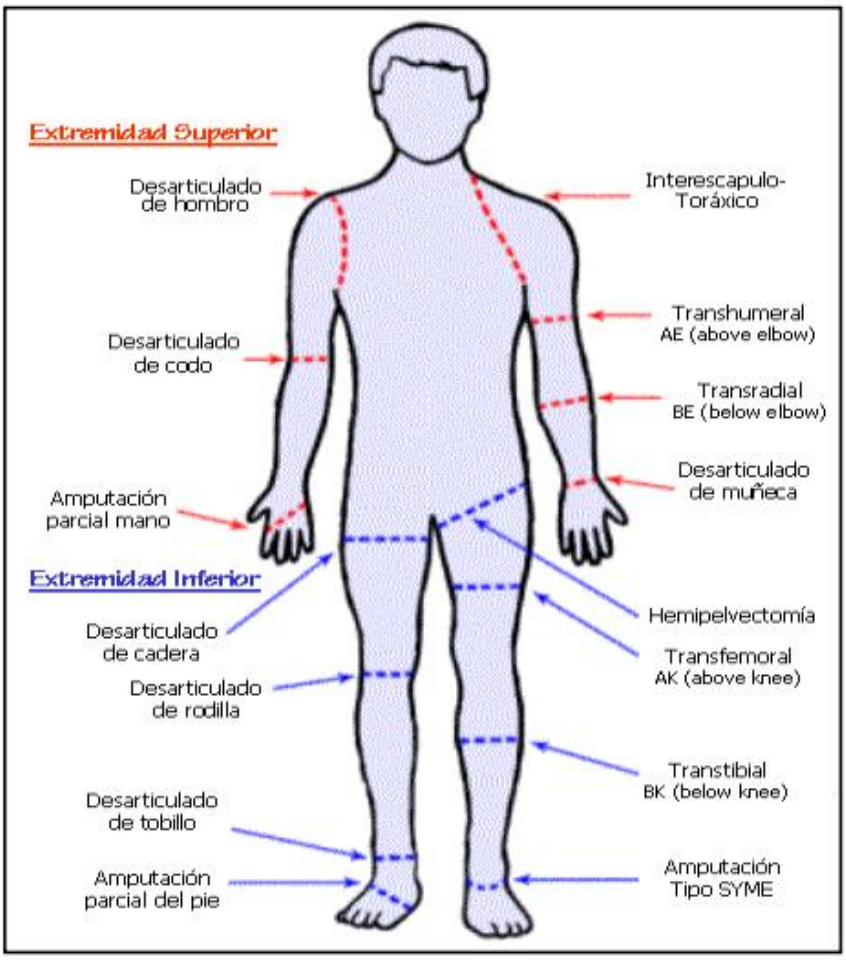


Figura 1.11. Niveles de amputación del cuerpo [4].

Los niveles de amputación del miembro inferior son:

- Desarticulación de cadera.
- Amputación transfemoral (arriba de la rodilla).
- Desarticulación de rodilla.
- Amputación transtibial (debajo de la rodilla).
- Amputación de Syme o transmaeolar (desarticulación del tobillo).
- Amputación de Lisfranc o tarsometatarsiana (amputación parcial del pie).

- Amputación de Chopart o transmetatarsiano (amputación parcial del pie) [4].

El caso de estudio de este trabajo es la amputación transtibial o debajo de la rodilla (Figura 1.12).

Figura 1.12. Amputación transtibial o infracondilia [11].

1.2.2. Cambios anatómicos y fisiológicos

"El nivel de amputación es relevante cuando la persona amputada entra en contacto con su prótesis, ya que cuando se conserva la articulación de la rodilla su uso demanda bajo gasto energético" [1]; adicionalmente, la rodilla es importante biomecánicamente para minimizar el desplazamiento del centro de gravedad en la marcha y los traslados.

Las personas con amputación del miembro inferior sufren síndrome de des acondicionamiento físico. Se define como el deterioro metabólico y sistemático del organismo como consecuencia de la inmovilización prolongada [1].

Este síndrome produce diversas alteraciones a nivel de diferentes sistemas:

- **Sistema musculo - esquelético:** Se produce una pérdida de masa y fuerza muscular, así como la reducción de la densidad mineral ósea, proporcionales a la duración del periodo de inmovilización. En cuanto a la morfología y función muscular se produce una atrofia por falta de uso muscular. Existe alteración del equilibrio de calcio [1].
- **Sistema metabólico y endocrino:** Se produce un incremento en la excreción de nitrógeno urinario, el cual conduce a hipoproteinemia, edema y pérdida de peso. El nivel de colesterol incrementa y el nivel de insulina es variante [1].
- **Sistema respiratorio:** Restricción mecánica de la ventilación. Reducción del volumen sanguíneo capilar pulmonar y la capacidad total de difusión pulmonar.
- **Sistema genitourinario:** Aumento del flujo sanguíneo renal y de la eliminación renal de agua. Incremento en la excreción de sodio y potasio. Se promueve la formación de cálculos renales [1].
- **Sistema nervioso central:** Se produce una privación sensitiva y psicosocial. El aislamiento social produce ansiedad, labilidad emocional, depresión, irritabilidad, etc. También se produce falta de concentración, motivación, problemas de memoria, etc. [1].
- **Sistema cardiovascular:** Reducciones del consumo máximo de oxígeno y de volumen sistólico [1].

1.2.3. Tratamientos para la rehabilitación de personas con amputación transtibial

Las personas que pierden parcial o totalmente una extremidad realizan un proceso de rehabilitación que inicia desde el momento de su lesión hasta que utilizan una prótesis, procurando mantener condiciones físicas y psicológicas adecuadas para su adaptación [3].

“El periodo post quirúrgico inmediato puede ser de dos semanas o más, dependiendo de la cicatrización del muñón. Aunque normalmente, tardará hasta tres meses en estabilizarse por completo. Por esto, cuando la herida ha cicatrizado y los puntos se retiran, la readaptación del amputado empieza (en la mayoría de los casos) con una prótesis provisional, que se irá modificando según los cambios del muñón” [11].

1.2.3.1. Objetivos del tratamiento

- Lucha contra el dolor.
- Disminuir el edema.
- Conseguir la autonomía con el menor gasto energético.
- Mantener el ángulo de movimiento en límites normales.
- Prevenir, o corregir, si se presentaran, las retracciones musculares causadas por posturas viciosas.
- Corregir los defectos de alineamiento.
- Mejorar la circulación y nutrición del muñón.
- Establecer el equilibrio muscular.
- Restaurar o aumentar la fuerza muscular, resistencia y coordinación.
- Prevenir la excesiva atrofia de tejidos.
- Mantener y mejorar las reacciones neuromusculares.
- Desensibilización y fortalecimiento de muñón.
- Control y manejo del dolor fantasma.
- Vendaje y moldeado de muñón.
- Reeducación del esquema corporal.
- Entrenamiento marcha con pilón [11].

1.3. Evaluación de la condición muscular del paciente

En la práctica de fisioterapia tradicional, la fuerza se mide de forma cualitativa mediante observación de la musculatura, palpación y aplicación de una prueba de Fuerza (Daniels o escala de Kendall).

La fuerza es una variable que en algunos casos intenta ser alterada por el paciente, pretendiendo realizar menor fuerza de la que es capaz. En esos casos es de utilidad tener algunas alternativas de valoración cuantitativa [12], como:

1.3.1. Electromiografía

Figura 1.13. Electromiografía en la fisioterapia [13].

Se utiliza como método o estrategia para objetivar la medición de la fuerza muscular gracias a que permite la identificación de la activación de uno o varios grupos musculares.

La principal limitación de este método consiste en que para uso en fisioterapia la técnica debe ser electromiografía de superficie (Figura 1.13), la cual no permite obtener de forma directa la fuerza ejercida por el grupo muscular evaluado. Sin embargo, brinda información complementaria a través del procesamiento de señales utilizando características en el dominio

del tiempo o de la frecuencia, usando diferentes algoritmos, como la transformada de Fourier o la transformada de Wavelet [12].

En personas sin amputación, para evaluar la condición del paciente se toma referencias individuales entre el miembro afectado y el sano, ya que al presentar las mismas características morfofisiologías se determina que el error resultante entre dos músculos sanos es inferior al error resultante entre un músculo sano y uno afectado [12]. Sin embargo, esto se aplica a personas con amputación al comparar la fuerza muscular de la pierna sana y la amputada.

Adicionalmente se procesa la señal mediante la fatiga muscular; ya que, al realizar un esfuerzo con un tipo de contracción y un tiempo determinado, la frecuencia y amplitud de la señal deberán ser similares en cada repetición, lo cual ayuda a identificar si el paciente está simulando, es decir, no ejerce la mayor fuerza posible [12].

1.3.2. Galga extensiométrica

Consiste en un dispositivo de medición que mide deformación e indirectamente presión, carga, torque, posición o fuerza. En el último caso, la medición se hace por tensión o compresión, variando la resistencia según el esfuerzo al que va a ser sometido [12].

Estos elementos son utilizados como principio de acción de sensores de fuerza, celdas de carga (Figura 1.14) y otras herramientas tecnológicas que permiten la valoración cuantitativa de la variable fuerza y que son aplicadas a grupos musculares en pacientes [12].

La galga extensiométrica específicamente tiene el inconveniente de la necesidad de experticia para su montaje y manipulación, ya que requiere definir adecuadamente los puntos donde será ubicada, ser adherida con resina o similar en la superficie de interés y construir o

adquirir un sistema de bioinstrumentación para transformar las micro deformaciones en variables eléctricas que sean fácilmente medidas [12].

Figura 1.14. Galga extensiométrica en plataforma de fuerza [14].

Adicionalmente, el proceso de selección de la galga suele ser riguroso y adaptado a la aplicación, y es susceptible a las variaciones de temperatura, que se compensa desde el circuito de bioinstrumentación [12].

1.3.3. Dinamometría

Figura 1.15. Dinamómetro universal NedDFM/IBV [15].

Es una herramienta tecnológica con el objetivo de medir cuantitativamente la capacidad muscular, expresada en fuerza máxima, potencia o fatiga muscular. Se utiliza como máquina de entrenamiento de potencia muscular o como herramienta para la valoración física del componente muscular e incluso articular (Figura 1.15) [12].

La evaluación se realiza de forma comparativa, y consiste en la ejecución de varias repeticiones de movimiento, estableciendo metodología y criterio de estudio entre una o varias variables, como la relación entre músculo sano/afectado, contracción concéntrica/excéntrica, fuerza máxima/velocidad de movimiento, comportamiento muscular en diferentes rangos de movilidad articular y/o fatiga muscular por contracción isométrica [12].

Esta prueba se complementa con el uso de electromiografía, ya que al realizar el movimiento se observa el aumento o disminución en la señal gráfica según el número de unidades motoras reclutadas [12].

La principal desventaja de los dinamómetros comerciales existentes consiste en que solo permiten hacer pruebas sobre una articulación y en un plano de movimiento determinado. Además, el diseño de fabricación hace que sea difícil realizar valoraciones del tronco; así mismo, por tratarse de un equipo robusto, sus costos son elevados, por lo que su uso no es tan frecuente, aunque en la actualidad se accede a equipos más económicos y mejores diseños [12].

1.3.4. Escalas de valoración de fuerza muscular

Asigna un valor en una escala del 0 al 5 o 0% al 100% (Tabla 1.3), que representan la ausencia de actividad o una respuesta normal, respectivamente. Estas pruebas se aplican a un movimiento, por lo cual la puntuación representa la actividad de todos los músculos en ese movimiento [16].

Tabla 1.3. Valoración de fuerza muscular [16] [17].

Escala de Daniels	Escala de Kendall	Puntuación Cualitativa	Interpretación
5	100%	**Normal (N):** Ejecuta el movimiento completo o mantiene una posición limite contra la máxima resistencia.	Musculo normal
4	80%	**Bien (B):** Ejecuta el movimiento completo contra la fuerza de la gravedad y tolera una resistencia fuerte sin modificar su postura. Sin embargo, cuando la resistencia es máxima se desplaza.	Déficit de movimiento voluntario
3	50%	**Regular (R):** Ejecuta el movimiento completo solo frente a la fuerza de la gravedad. Cualquier resistencia adicional impide el movimiento.	
2	20%	**Mal (M):** Ejecuta el movimiento completo en una posición horizontal al movimiento (Fuerza de la gravedad mínima).	
1	5%	**Actividad escasa (E):** No existe desplazamiento, pero se siente o se visualiza la tensión del tendón por tratar de ejecutar el movimiento.	Parálisis parcial
0	0%	**Nula (Sin actividad):** Carente de actividad a la palpación o inspección visual.	Parálisis total

1.3.4.1. Evaluación de flexión de la rodilla

Se evalúan los músculos posteriores del muslo, mediales (semitendinoso y semimembranoso) y lateral (bíceps femoral).

- **Grados 5 (normal), 4 (bien) y 3 (regular)**

El examinador explora primero los 3 músculos en conjunto (con el pie en la línea media). Solo cuando se produce una desviación (o asimetría) en el movimiento o existe duda se recurre a la exploración de los músculos de forma independiente [16].

- **Músculos en conjunto:** El paciente debe estar en posición decúbito prono, con los brazos estirados y los pies sobresaliendo del borde de la mesa, con la rodilla flexionada 45°. El fisioterapista se posiciona al lado de la extremidad que se explora, con la mano que ejerce resistencia en la superficie posterior de la pierna sobre el tobillo. El paciente flexiona la rodilla mientras mantiene la pierna en rotación de equilibrio (Figura 1.16) [16].

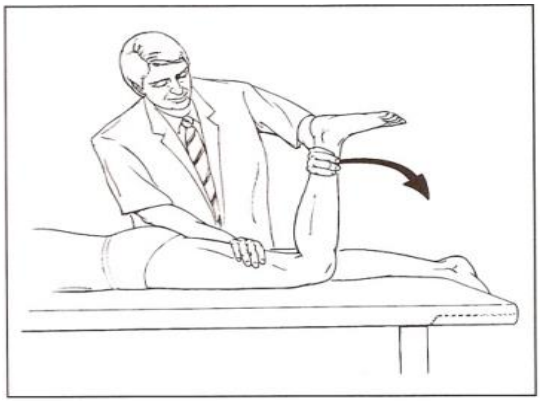


Figura 1.16. Evaluación de los músculos que intervienen en la flexión de rodilla en conjunto para grados 5, 4 y 3 [16].

- **Músculos mediales (semitendinoso y semimembranoso):** El paciente debe estar en posición decúbito prono, con la rodilla flexionada hasta al menos 90° y la pierna en rotación

interna. El fisioterapista sostiene la pierna por encima del tobillo y ejerce resistencia en sentido oblicuo (abajo y afuera) hacia la extensión de la rodilla. El paciente flexiona la rodilla, mientras mantiene la pierna en rotación interna (Figura 1.17) [16].

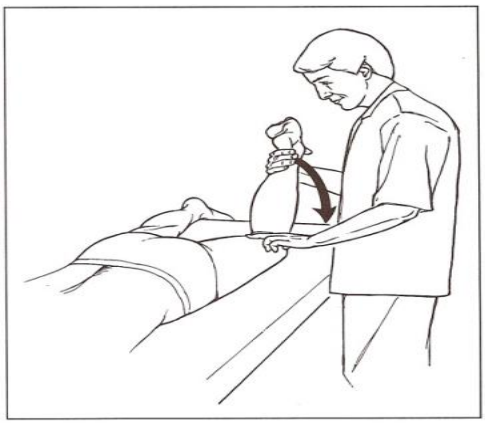


Figura 1.17. Evaluación de los músculos mediales que intervienen en la flexión de rodilla para grados 5, 4 y 3 [16].

- **Musculo lateral (bíceps femoral):** El paciente debe estar en posición decúbito prono, con la rodilla flexionada hasta al menos 90° y la pierna en rotación externa. El fisioterapista ejerce resistencia que se opone a la flexión de la rodilla, aplicada sobre el tobillo, hacia abajo y hacia dentro. El paciente flexiona la rodilla, mientras mantiene la pierna en rotación externa (Figura 1.18) [16].

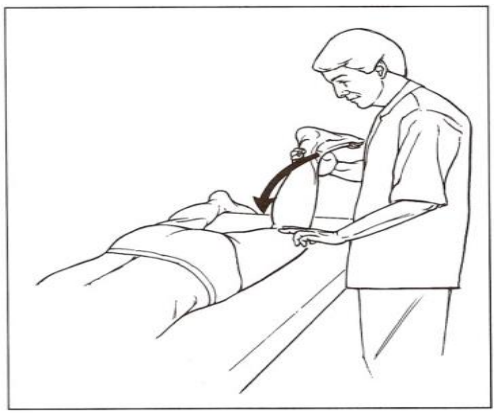


Figura 1.18. Evaluación del musculo lateral que interviene en la flexión de rodilla para grados 5, 4 y 3 [16].

- **Grado 2 (mal)**

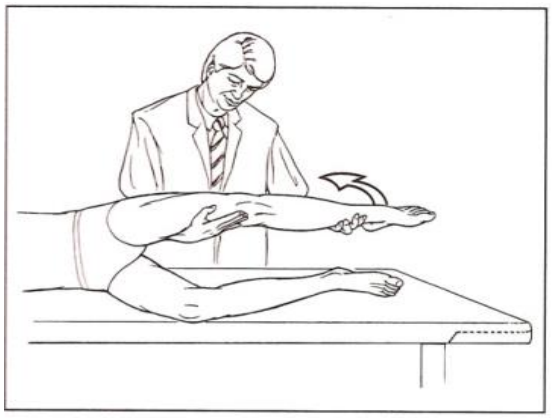


Figura 1.19. Evaluación de los músculos que intervienen en la flexión de rodilla para grado 2 [16].

El paciente debe estar en posición decúbito lateral, con la pierna inferior flexionada para mantener la estabilidad. El fisioterapista se ubica detrás del paciente a la altura de la rodilla, sosteniendo la pierna superior (a examinarse), una mano sujeta el muslo en el lado medial de la rodilla y la otra sostiene la pierna por el tobillo. El paciente flexiona la rodilla con toda la amplitud posible del movimiento (Figura 1.19) [16].

- **Grado 1 (escaso) y grado 0 (nulo)**

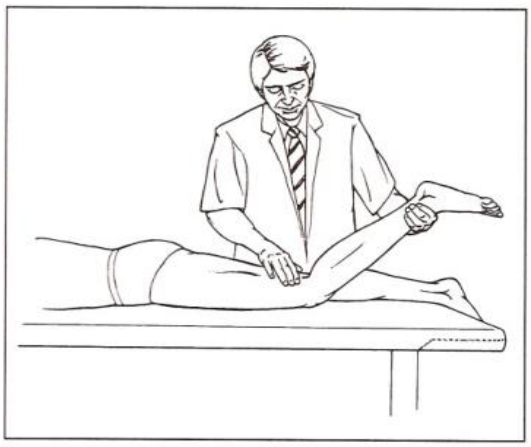


Figura 1.20. Evaluación de los músculos que intervienen en la flexión de rodilla para grado 1 y 0 [16].

El paciente debe estar en posición decúbito prono, con los brazos y los pies estirados sobre el borde de la mesa, la rodilla parcialmente flexionada. El fisioterapista se ubica en el lado de la extremidad a examinarse a la altura de la rodilla, una mano sostiene el miembro flexionado por el tobillo y la otra palpa ambos tendones medial y lateral, inmediatamente por encima de la parte posterior de la rodilla. El paciente intenta flexionar la rodilla (Figura 1.20) [16].

1.3.4.2. Evaluación de extensión de la rodilla

Los músculos del cuádriceps femoral se exploran de forma conjunta como grupo funcional. No es posible examinarlos individualmente en una exploración manual [16].

- **Grados 5 (normal), 4 (bien) y 3 (regular)**

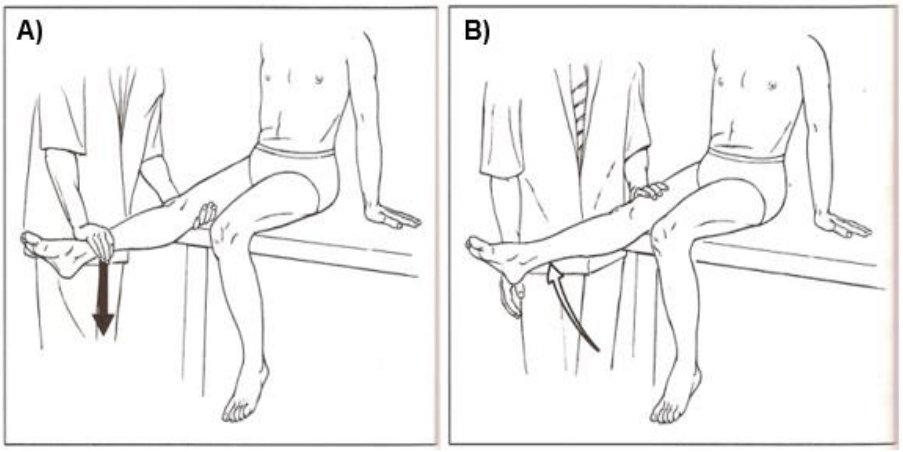


Figura 1.21. Evaluación de los músculos que intervienen en la extensión de rodilla. A) Grados 5 y 4. B) Grado 3 [16].

El paciente debe estar sentado. Se coloca un cojín o la mano por debajo de la porción distal del muslo, para conservar el fémur en posición horizontal. Las manos descansan sobre la mesa a los lados para mantener la estabilidad, o sujetarse a la mesa. Se permite al paciente que se incline hacia atrás, para disminuir la tirantez de los músculos posteriores. No se admite que el paciente realice una hiperextensión de la rodilla, porque esto la bloquea en esta posición [16].

El fisioterapista está al lado de la extremidad examinada, la mano que ejerce resistencia sobre la superficie anterior de la pierna, por encima del tobillo. En las pruebas de los grados 5 y 4 se ejerce la resistencia hacia abajo, en sentido de la flexión de la rodilla. El paciente extiende la rodilla realizando el movimiento completo, pero no sobrepasa los 0° en hiperextensión (Figura 1.21) [16].

- **Grado 2 (mal)**

El paciente debe estar decúbito lateral, con la extremidad a examinar encima. La otra extremidad se flexiona para mantener la estabilidad. El examinador mantiene la rodilla de la paciente flexionada a 90°. El fisioterapista se ubica alado a la altura de la rodilla. Con un brazo sostiene la extremidad, rodeando el muslo, con la mano por debajo de la rodilla y con el otro sostiene la pierna por el tobillo. El paciente extiende la rodilla con toda la amplitud posible del movimiento. Al sostener la extremidad el examinador no debe ofrecer resistencia ni ayudar al movimiento voluntario del paciente (Figura 1.22) [16].

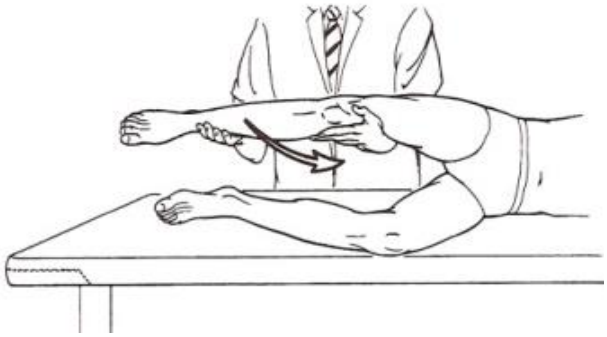


Figura 1.22. Evaluación de los músculos que intervienen en la extensión de rodilla para grado 2 [16].

- **Grado 1 (escaso) y 0 (nulo)**

El paciente está en posición decúbito supino. El fisioterapista se ubica alado de la extremidad a examinarse a la altura de la rodilla. Una mano se utiliza para palpar el tendón del

cuádriceps por encima de la rodilla, sujetando suavemente el tendón entre el dedo pulgar y los dedos. El examinador palpa el tendón de la rótula, con dos o cuatro dedos por debajo de la rodilla, mientras el paciente intenta estirar la rodilla [16].

Un test alternativo consiste en colocar una mano debajo de la rodilla, que está ligeramente flexionada, se palpa el cuádriceps o el tendón rotuliano, mientras el paciente intenta estirar la rodilla (Figura 1.23) [16].

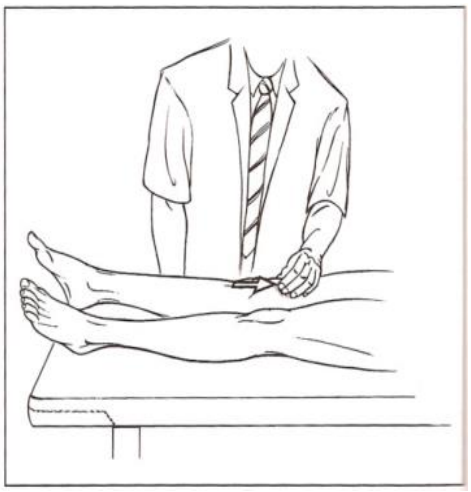


Figura 1.23. Evaluación de los músculos que intervienen en la extensión de rodilla para grados 1 y 0 [16].

1.4. La unidad motora y la señal electromiográfica

Una unidad motora (Figura 1.24) es la responsable del movimiento producido en un músculo. Está compuesta por una única neurona motora, sus dendritas y las diversas ramificaciones de su axón y las distintas fibras musculares que son accionadas por esta. En una unidad motora básica se encuentra desde pocas unidades de estas fibras (para movimientos precisos como en un ojo o un dedo) hasta miles de ellas (para grandes músculos como el bíceps braquial o el músculo gastrocnemio) [18].

Estas neuronas motoras son las encargadas de transportar los impulsos eléctricos generados en el cerebro y la espina dorsal hasta el músculo, utilizando para ello complejas reacciones químicas en las terminaciones axónicas que se encuentran en contacto con las fibras musculares, llamadas sinapsis neuromusculares. Estas reacciones consisten en el intercambio de iones y otras sustancias capaces de polarizar y despolarizar la unión entre fibra y neurona, generando así una diferencia de potencial de varias decenas de mV que se mide con sencillez, incluso en la superficie de la piel [18].

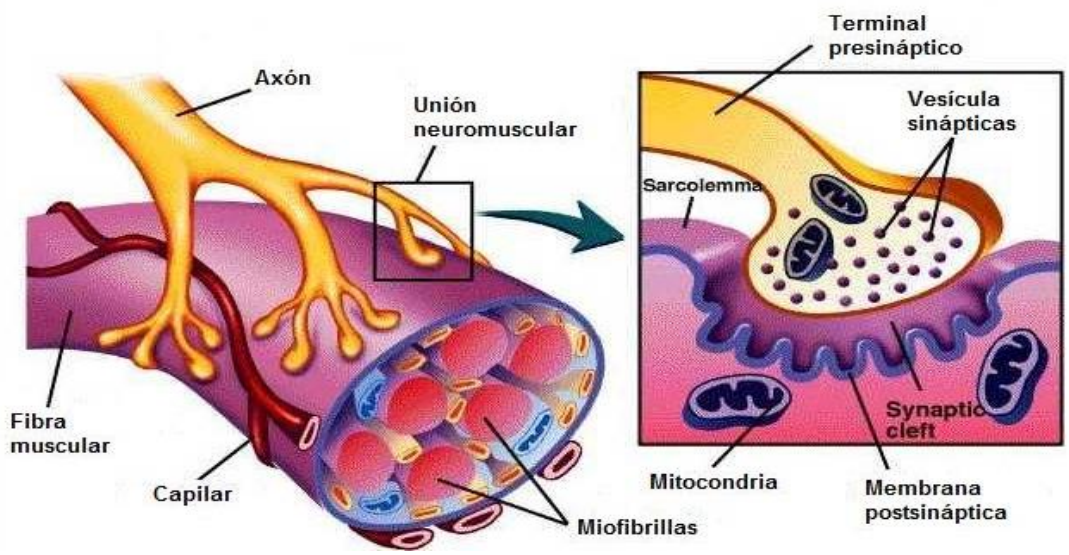


Figura 1.24. Unidad motora [19].

El registro de los cambios producidos por la descarga de fibras musculares de una determinada unidad motora se conoce como potencial de acción de la unidad motora (PAUM). En condiciones normales, la amplitud media de un PAUM está entre 0.5mV y su duración es de 8 a 14 ms dependiendo del tamaño de la UM. Si la abundancia de sodio excede un cierto nivel de umbral, la despolarización de la membrana causa un potencial de acción que rápidamente cambia de -80mV a +30mV; después de la excitación, esta diferencia de potencial se desplaza por la fibra muscular a una velocidad de 2-6m/s [20].

Una vez esta señal eléctrica ha llegado a las fibras musculares controladas por dicha neurona motora, estas se contraen generando un movimiento.

Una señal electromiográfica (Figura 1.25) es fruto de la superposición de todas las señales eléctricas creadas por las diversas unidades motoras en contacto con un transductor especializado para la captura de las mismas, llamado comúnmente electrodo, en un momento discreto de tiempo [21].

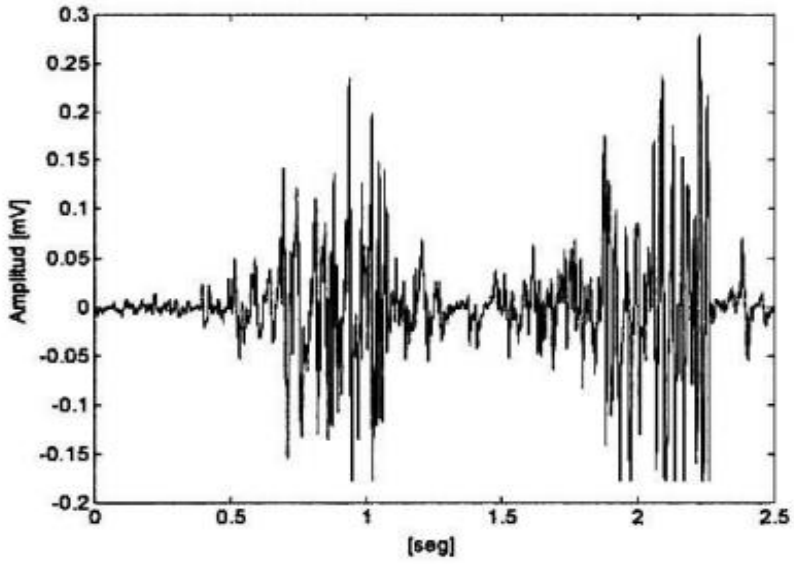


Figura 1.25. EMG de superficie durante contracciones intermitentes del músculo extensor de la muñeca [21].

2. METODOLOGÍA

2.1. Metodología de investigación

El proceso de investigación que se requiere es de tipo cuantitativo. El orden es riguroso, parte de una idea que va acotándose y, una vez delimitada, se derivan objetivos y preguntas de investigación, se revisa la literatura y se construye un marco o una perspectiva teórica. De las preguntas se establecen hipótesis y determinan variables; se traza un plan para probarlas (diseño); se miden las variables en un determinado contexto; y se extrae una serie de conclusiones respecto de la o las hipótesis [22].

La metodología manejada en este trabajo de grado se puede visualizar en la Figura 2.1.

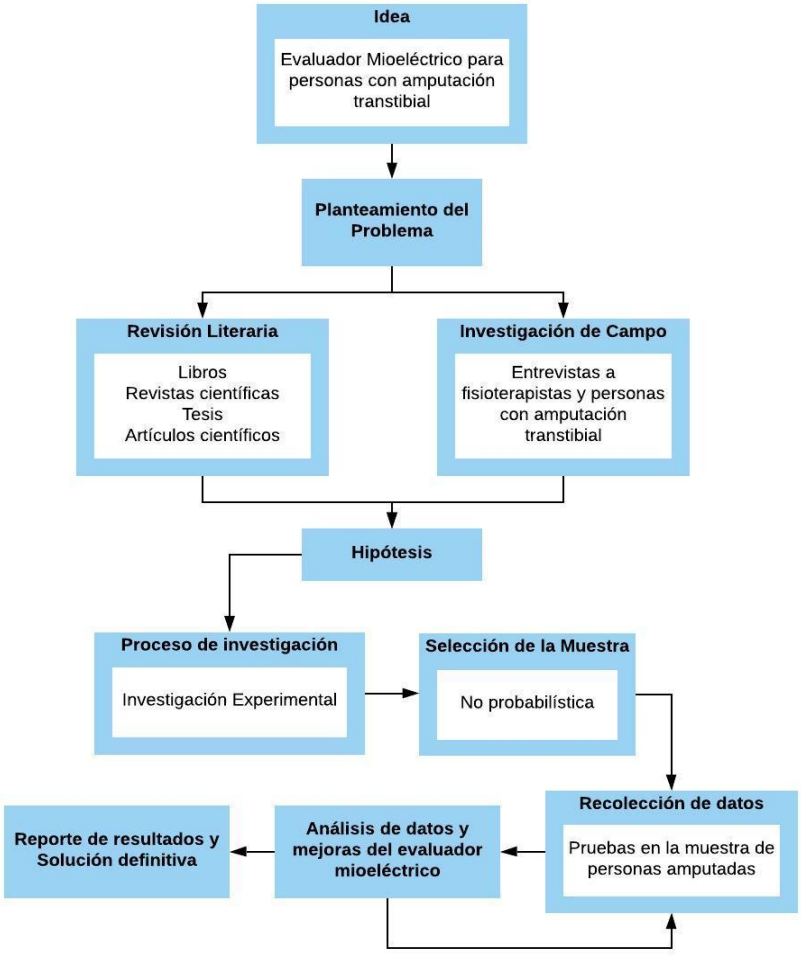


Figura 2.1. Metodología de la investigación

2.1.1. Proceso de investigación

El proceso de investigación requiere una investigación experimental. Las personas con amputación transtibial en la etapa de rehabilitación pre protésica requieren un control cronológico por parte del fisioterapista, debido a los diversos cuidados que requieren. Este proceso de rehabilitación toma hasta 3 meses.

Usualmente en esta etapa los fisioterapistas realizan pruebas de carácter cualitativo (Daniels o Kendall). Estas pruebas dependen netamente de la experiencia y pericia del evaluador, que asigna una valoración en una escala del 0-5 o 0%-100%, sin información de retroalimentación, haciéndolo poco preciso. Por tanto, es muy importante determinar un protocolo para la ejecución de esta etapa mediante la implementación de señales electromiográficas.

2.1.1.1. Protocolo de adquisición y evaluación de la condición muscular

- Realizar la medición del nivel de la señal en los músculos que intervienen en la flexión y extensión de la rodilla en la pierna sana (sin amputación), al nivel obtenido en la flexión y extensión se le asigna el grado normal (100%) respectivamente.
- En la pierna que presenta la amputación se realiza la toma de señal en los mismos músculos tanto en flexión como extensión de rodilla.
- Se compara el nivel normal con los obtenidos en el segundo paso, para cada musculo que interviene en los movimientos y a nivel general del cuádriceps e isquiotibial, mediante la obtención de un promedio. Se les asigna una valoración dentro de la escala del 0% - 100%, tanto para flexión como para extensión de la rodilla.
- Finalmente se asigna una puntuación cualitativa (normal, bien, regular, mal, escasa, nula) en base a la escala de Daniels y Kendall.

Este protocolo permite tener una prueba que registra el valor cuantitativo desde el punto de vista electromiográfico y lo interpreta para mejor comprensión por parte del fisioterapeuta.

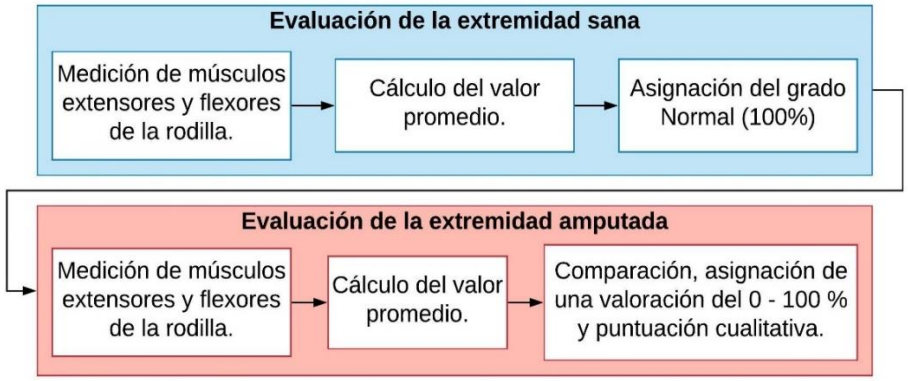


Figura 2.2. Protocolo de evaluación.

2.1.1.2. Investigación Experimental

Mediante el protocolo antes mencionado se obtiene el nivel de fuerza muscular del paciente acorde a la escala de evaluación de Daniels o Kendall. Para comprobar su validez se requiere que el fisioterapeuta realice la prueba manual y contrastar su criterio con los valores obtenidos.

2.2. Requisitos del Evaluador de la condición muscular mioeléctrico

El evaluador utiliza señales electromiografías para contribuir en la preparación del paciente para el uso de una prótesis normal, en la etapa pre protésica o después de pasar mucho tiempo sin utilizarla. Por tanto, debe estar compuesto por un sistema electrónico (hardware) y un programa que permita operarlo (software).

2.2.1. Hardware

Las señales biológicas producidas por los músculos no son tomadas y analizadas tal cual son adquiridas. Al tratarse de diferencias de potencial muy pequeñas, estas se acondicionan mediante filtros que eliminen las bandas de frecuencia que no aportan información relevante,

amplificadores diferenciales con un alto rechazo en modo común (CMRR) y posteriormente digitalizadas para su uso final [18]. Por tanto, todo evaluador mioeléctrico tiene el sistema de la Figura 2.3.

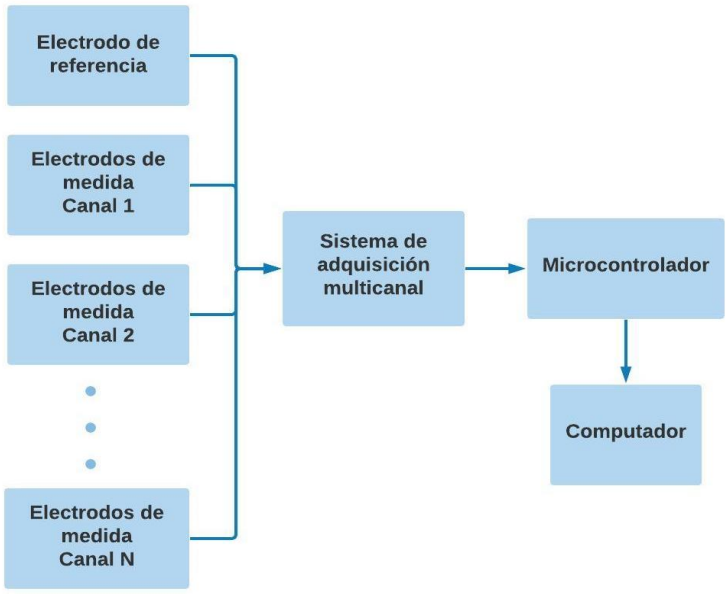


Figura 2.3. Diagrama de bloques del sistema de adquisición electromiográfico

En el mercado se encuentra multitud de dispositivos profesionales capaces de realizar las funciones antes descritas, así como mostrar las señales adquiridas por pantalla mediante software propio, ya sea en un PC embebido en la propia estructura del sistema o de forma modular en un PC portátil. El principal problema de todos estos dispositivos es su alto precio, solo apto para su uso constante en clínicas o centros especializados [18].

Existen dispositivos de bajo coste con características similares, sin embargo, para considerarse candidatos, deben cumplir con normativas. La Asociación para el Avance de la Instrumentación Médica (AAMI), ha promulgado normas para el diseño y ha presentado una serie de estándares para el etiquetado, colores, seguridad y el rendimiento que el instrumento médico está obligado a cumplir. Las características para el hardware se muestran en la Tabla 2.1 [23].

Tabla 2.1. Recomendaciones para equipos médicos profesionales de electromiografía según AAMMI [23].

Ancho de banda	De 20 a 500 Hz
Rechazo a la atenuación	Mínimo 80 dB/dec
Impedancia electrodo-tierra	> 5 MΩ
Impedancia electrodo-piel	< 30 Ω
El rechazo en modo común (CMRR)	Mínimo 100 dB
Amplitud diferencial	De 20 a 500 μV
Corriente electrodo- tierra	< 20 μA

El teorema de muestreo de Nyquist-Shannon afirma que, para digitalizar una señal sin perder información, se muestrea a una frecuencia de al menos el doble de la frecuencia fundamental de la señal analógica [20].

Sin embargo, en el tratamiento de la señal se utilizan diversos filtros que descartan información, esto se muestra en diversas investigaciones con señales electromiográficas, en donde se concluye que la información correspondiente a la fuerza muscular se encuentra alrededor de los 150 Hz.

2.2.2. Software

Además, de contar con un hardware adecuado, se requiere un software que realice el tratamiento de la señal electromiográfica recibida en el computador, obteniendo y almacenando en una base de datos la información necesaria para evaluar el estado del paciente y su avance en el tratamiento. Ayudando al fisioterapista en el proceso de rehabilitación, al tener una retroalimentación del estado del paciente y un mejor diagnóstico.

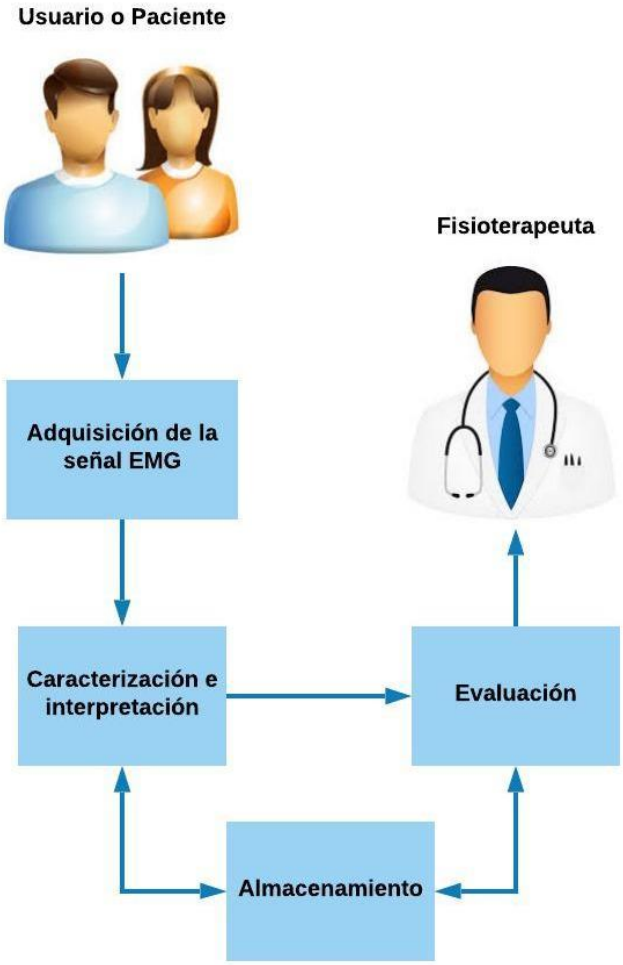


Figura 2.4. Estructura funcional de un evaluador mioeléctrico.

Contribuyendo a una mejora rápida del paciente. Por tanto, la estructura del software de un evaluador mioeléctrico se muestra en la Figura 2.4.

2.3. Metodología para la selección del hardware

Para la selección del hardware se puso a consideración las características de diferentes tarjetas de adquisición con su software, mediante el método de evaluación de criterios ponderados. En general estas no se centran sobre un determinado elemento, sino que se ponderan distintos aspectos del sistema en base a criterios que a menudo implican juicios de valor [24]. Para tomar una decisión siempre están presentes los dos elementos siguientes:

- **Alternativas:** Como mínimo se dispone de dos alternativas (lo más adecuado es entre 3 y 6) cuyas características son diferentes.
- **Criterios:** Hay que establecer los criterios en base a los cuales las alternativas serán evaluadas, así como también la ponderación relativa entre ellas [24].

Tabla 2.2. Valores asignados a los criterios

1	Si el criterio (o solución) de las filas es superior (o mejor; >) que el de las columnas
0.5	Si el criterio (o solución) de las filas es equivalente (=) al de las columnas
0	Si el criterio (o solución) de las filas es inferior (o peor; <) que el de las columnas

El método se basa en unas tablas donde cada criterio (o solución, para un determinado criterio) se confronta con los restantes criterios (o soluciones) y se asignan los valores mostrados en la Tabla 2.2 [24].

Luego, para cada criterio (o solución), se suman los valores asignados con relación a los restantes criterios (o soluciones) al que se le añade una unidad (para evitar que el criterio o solución menos favorable tenga una valoración nula); después, en otra columna se calculan los valores ponderados para cada criterio (o solución) [24].

Finalmente, la valoración total para cada solución resulta de la suma de productos de los pesos específicos de cada solución por el peso específico del respectivo criterio [24].

2.3.1. Criterios de evaluación

Para la selección del hardware se consideraron 4 criterios:

- **Precio:** Se busca el menor precio de compra y envió.
- **Funcionalidad:** El equipo debe cumplir con los requerimientos mínimos según las normas AAMMI. Es necesario que los canales de adquisición de la tarjeta coincidan

inicialmente con el número de músculos involucrados en la flexión y extensión de la rodilla (8 canales). Se requiere que sea inalámbrica para mayor facilidad de movimiento y trabajar con baterías para mayor seguridad.

- **Usabilidad:** El software y el hardware deben ser fáciles de aprender a usar. Con licencia libre.
- **Flexibilidad:** El software o el hardware son modificados por los desarrolladores.

Las alternativas de solución consideradas son:

1. Tarjeta de biosensado Cyton.
2. Sensor Bluetooth FlexVolt (8 canales).
3. Muscle SpikerShield Pro (6 canales) y Arduino.
4. Olimex (1 canal) y Stm32 Discovery.

2.3.2. Método de evaluación de criterios ponderados

Después de comparar la importancia de cada criterio con los demás, se determinó que la funcionalidad de la tarjeta y la flexibilidad del hardware o software son los criterios más importantes (Tabla 2.3).

Tabla 2.3. Comparación de los criterios de evaluación

Criterio	Precio	Funcionalidad	Usabilidad	Flexibilidad	∑+1	**Ponderación**
Precio		0	0	0	1	0.100
Funcionalidad	1		1	0.5	3.5	0.350
Usabilidad	1	0		0	2	0.200
Flexibilidad	1	0.5	1		3.5	0.350
				Suma	10	1

Se compararon las alternativas en base a los diferentes criterios en la Tabla 2.4.

Tabla 2.4. Comparación de las alternativas

Precio	Alternativa 1	Alternativa 2	Alternativa 3	Alternativa 4	**∑+1**	**Ponderación**
Alternativa 1		0	0.5	0	1.5	0.150
Alternativa 2	1		1	1	4	0.400
Alternativa 3	0.5	0		0	1.5	0.150
Alternativa 4	1	0	1		3	0.300
Funcionalidad	Alternativa 1	Alternativa 2	Alternativa 3	Alternativa 4	**∑+1**	**Ponderación**
Alternativa 1		0.5	1	1	3.5	0.350
Alternativa 2	0.5		1	1	3.5	0.350
Alternativa 3	0	0		0	1	0.100
Alternativa 4	0	0	1		2	0.200
Usabilidad	Alternativa 1	Alternativa 2	Alternativa 3	Alternativa 4	**∑+1**	**Ponderación**
Alternativa 1		1	0	1	3	0.300
Alternativa 2	0		0	1	2	0.200
Alternativa 3	1	1		1	4	0.400
Alternativa 4	0	0	0		1	0.100
Flexibilidad	Alternativa 1	Alternativa 2	Alternativa 3	Alternativa 4	**∑+1**	**Ponderación**
Alternativa 1		1	1	1	4	0.400
Alternativa 2	0		0.5	0.5	2	0.200
Alternativa 3	0	0.5		0.5	2	0.200
Alternativa 4	0	0.5	0.5		2	0.200

Finalmente, con la suma del producto de las ponderaciones se determina que la alternativa 1 es la solución más fiable en base a los criterios antes establecidos (Tabla 2.5).

Tabla 2.5. Comparación de las alternativas con los criterios

Criterio	Precio	Funcionalidad	Usabilidad	Flexibilidad	∑	**Prioridad**
Alternativa 1	0.015	0.123	0.060	0.140	0.338	1
Alternativa 2	0.040	0.123	0.040	0.070	0.273	2
Alternativa 3	0.015	0.035	0.080	0.070	0.200	3
Alternativa 4	0.030	0.070	0.020	0.070	0.190	4
				Suma	1.000	

2.3.3. Tarjeta de biosensado Cyton

Es una interfaz neuronal de 8 canales compatible con Arduino con un procesador de 32 bits. En su núcleo, el OpenBCI Cyton (Figura 2.5) implementa el microcontrolador PIC32MX250F128B, lo que le brinda una gran cantidad de memoria local y velocidades de procesamiento rápidas [25].

La placa se comunica de forma inalámbrica a una computadora a través del dongle USB OpenBCI utilizando los módulos de radio RFDuino. También, se comunica a cualquier dispositivo móvil o tableta compatible con Bluetooth Low Energy (BLE) [25].

La placa y el dongle vienen precargados con el último firmware, lo que hace que OpenBCI sea accesible para los investigadores y desarrolladores con poca o ninguna experiencia en hardware. Los tableros OpenBCI tienen una lista creciente de formatos de salida de datos, lo que los hace compatibles con una colección en expansión de aplicaciones y herramientas de bio retroalimentación existentes (Anexo 1) [25].

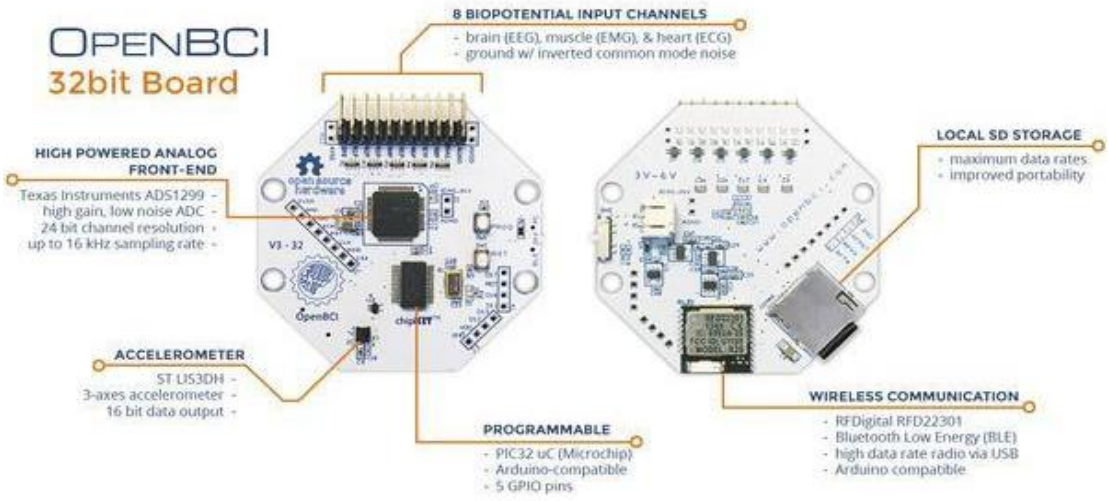


Figura 2.5. Tarjeta de biosensado Cyton [25].

2.3.4. Electrodos

Para una mejor adquisición de datos se escogió electrodos desechables de espuma blanca, 30 mm. Diámetro, forma circular, chasquido formado, centrado con columna del gel sólido. Pegamento de grado medico agresivo. Esto elimina el lio de geles líquidos, ofreciendo la impedancia baja, sostiene el contenido acuoso en una formulación de hidrogel polimerizada, que es no sensible, no irritante y no citotóxico a la piel, libre de látex.

2.4. Metodología para la adquisición y análisis de señales EMG

La tarjeta OpenBCI Cyton se encarga de realizar la adquisición de los impulsos eléctricos originados por el musculo en sus diversos canales, amplificarlos, filtrarlos, codificarlos, transmitirlos mediante bluetooth a su dongle, el cual los decodifica. De esta forma la tarjeta brinda la señal cruda que debe ser procesada digitalmente para extraer la información relevante en la evaluación de la condición muscular del paciente.

En este procesamiento se utilizó los filtros:

- Respuesta invariante de impulso o filtro de Notch (IIR)
- Filtro pasa-bandas
- Filtro pasa-bajos

Existen varios métodos para la caracterización de la señal EMG que dependen de parámetros temporales, frecuenciales y tempo – frecuenciales. Para el este trabajo solo se requiere el nivel de contracción muscular. Por tanto, se utiliza:

Raíz Media Cuadrática (RMS): Es una de las técnicas más utilizadas debido a que permite detectar y medir los niveles de contracción muscular de acuerdo con el estimado de energía, relacionada con la amplitud en ciertos intervalos de tiempo. Su expresión viene dada por [26]:

$$RMS = \sqrt{\frac{1}{N}\sum_{i=1}^{N} {x_i}^2}$$

Ecuación 1. Raíz Media Cuadrática

Donde:

N es el número total de muestras, I el número de muestra y X el valor EMG.

2.5. Metodología para la construcción de interfaz gráfica de usuario

El pensamiento de diseño es una metodología utilizada para crear ideas innovadoras que centra su eficacia en entender y plantear soluciones a las necesidades reales de los usuarios. En términos sencillos, es una disciplina que usa la sensibilidad y los métodos del diseñador para hacer coincidir las necesidades de las personas con lo que es tecnológicamente factible [27].

Posee cinco factores diferenciales a otras metodologías: la generación de empatía, donde se busca entender los problemas, necesidades y deseos de los usuarios implicados en la solución;

el trabajo en equipo interdisciplinario, la generación de prototipos, que ayudan a la validación y mejoras de las ideas [27].

La metodología de pensamiento de diseño define las siguientes cinco etapas:

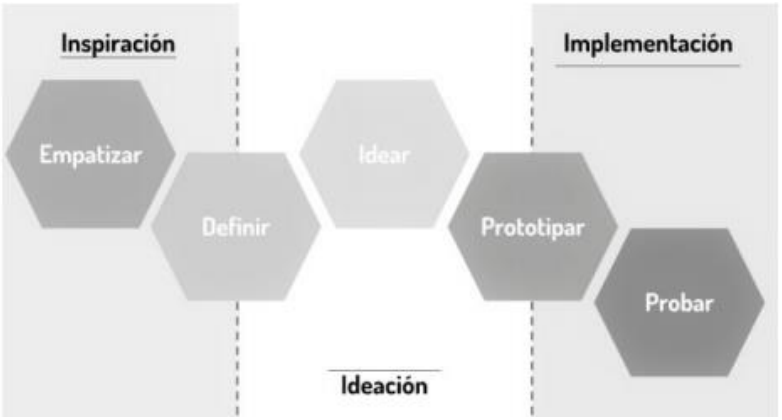


Figura 2.6. Etapas de la metodología del pensamiento de diseño [27].

- **Empatizar**: Este proceso busca comprender las necesidades de los usuarios implicados en la solución y su entorno.
- **Definir**: Se procesan y sintetizan todas revelaciones encontradas en la etapa anterior con el fin de formar una perspectiva clara para la creación de la solución [27].
- **Idear**: Se busca explorar una amplia variedad de soluciones posibles a través de la generación de ideas [27].
- **Prototipar**: En esta etapa se transforman las ideas en una forma física, siendo posible experimentar e interactuar con la solución propuesta, para validar las ideas y aprender del proceso mejorando la solución [27].
- **Probar**: Se realizan pruebas de los prototipos con los usuarios implicados en la solución con el fin de utilizar sus observaciones y comentarios para refinar los prototipos, identificar mejoras significativas, fallos a resolver y posibles carencias [27].

Para diseñar correctamente la interfaz gráfica de usuario es necesario basarse en la norma ISO 9241, que trata sobre los aspectos ergonómicos en puestos de trabajo con PVD, enfocada en la calidad de la usabilidad y ergonomía tanto del hardware como del software. También se tomó en consideración los principios de usabilidad heurística de Jakob Nielsen.

2.5.1. Principios de diseño de una Interfaz gráfica de Usuario

En el diseño de una GUI existen varios conceptos y principios a tomar en cuenta, sin embargo, los principios de usabilidad heurística de Jakob Nielsen son los más utilizados (Figura 2.7).

Figura 2.7. Principios de usabilidad heurísticos de Jakob Nielsen [28].

De esta forma se propone un modelo de GUI que cumpla con:

- **Visibilidad del estatus del sistema:** Mediante mensajes de confirmación o error, activación y desactivación de widgets y en procesos largos indicando la espera en el puntero del mouse.
- **Alineación entre el sistema y el mundo real:** La utilización de imágenes e iconos que asocien cada acción con la realidad; por ejemplo, para guardar se usa el icono de un

disquete. También empleando palabras conocidas por el usuario como registro, inicio, conectar, etc.

- **Control y libertad para el usuario:** Brindando la posibilidad de corregir errores, al editar la información ingresada en el registro del paciente, eliminar un paciente registrado, realizar nuevamente la adquisición de las señales EMG en la evaluación o sustituir la valoración de la fecha actual.
- **Consistencia y estándares:** Los colores se atienen a estándares preestablecidos; iniciar es verde y parar rojo. La información se mantiene a los extremos laterales de la pantalla. Se mantiene una misma gama de colores adecuados para la vista.
- **Reconocimiento antes que reacción:** Se muestra toda la información necesaria para evitar que el usuario tenga que memorizarla. En la pantalla de inicio se muestran todos los pacientes registrados, en la de evaluación una imagen de los puntos motores para flexión y extensión de rodilla y en la historia clínica se visualiza toda la información del paciente.
- **Prevención de errores:** Se confirma el ingreso de toda la información solicitada, que sea correcto el número de cedula, se delimitan campos numéricos, se comprueba la conexión del dispositivo, que la valoración de la extremidad sana haya sido realizada, el almacenamiento de la valoración realizada y la selección de un paciente para evaluar, visualizar su historia clínica, editar información o eliminar.
- **Estética y diseño minimalista:** La GUI debe mantenerse simple y concisa. Solo se incluye la información necesaria, de tal forma que las pantallas no se vean sobrecargadas de información.
- **Ayudar a los usuarios con los errores:** Los errores se muestran con lenguaje entendible por el usuario y existen sugerencias de solución en los botones de ayuda.
- **Ayuda y documentación:** Botones de ayuda y el manual de usuario.

3. DESARROLLO DEL SISTEMA DE EVALUACIÓN DE LA CONDICIÓN MUSCULAR

En el desarrolla del sistema de evaluación primero, se configuró la tarjeta electrónica Cyton para la adquisición multicanal de las señales mioeléctricas y la fuente de alimentación como parte del hardware. Segundo, la interfaz gráfica del usuario como parte del software. Finalmente, se hicieron pruebas con personas sanas y amputadas, siguiendo la metodología antes planteada en la sección 2.1.1.1.

3.1. Implementación del sistema de adquisición multicanal para señales mioeléctricas

La placa de biosensado Cyton, ver Figura 2.5, se usa en proyectos de investigación concernientes al área de bioingeniería, es decir, posee una aplicación médica. Por tanto, requiere algunas adaptaciones que permitan un fácil manejo por parte del fisioterapeuta (el usuario).

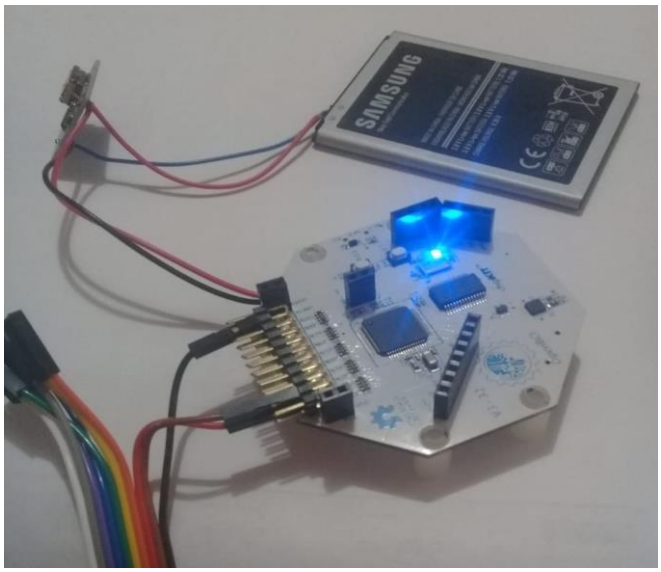


Figura 3.1. Conexión del circuito de carga.

Se adapto una batería de ion-litio con un circuito de carga para ser empleado por varias horas (Figura 3.1), y así evitar manipulación del equipo por terceros, caídas y daños. se elaboró

una caja de PLA mediante impresión 3D. Los planos se encuentran en el Anexo 2. El prototipo final se muestra en la Figura 3.2.

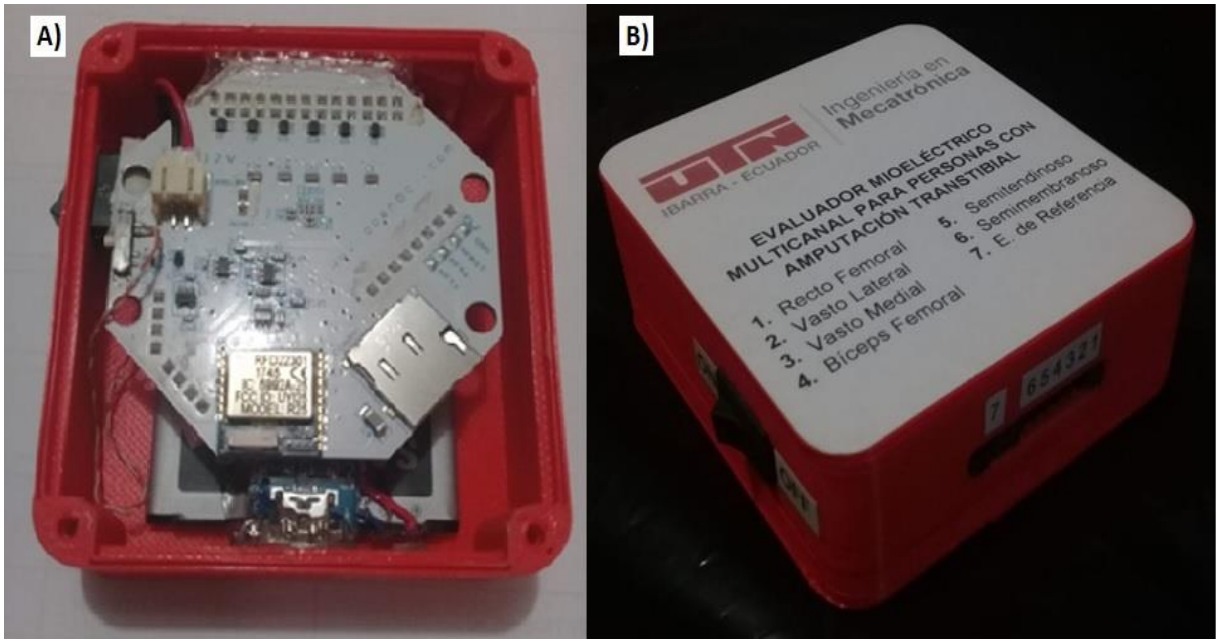


Figura 3.2. Evaluador mioeléctrico multicanal. A. Vista interna. B. Vista isométrica.

3.2. Registro cronológico de las señales obtenidas a partir del sistema de adquisición multicanal

La empresa OpenBCI elabora tarjetas con software y hardware libre, brindando varias herramientas para desarrolladores de software. La página web oficial presenta información referente al SDK en 3 lenguajes diferentes de programación como son: Processing (java), Python y JsNode, ver Figura 3.3.

Por las facilidades que brinda el lenguaje Python (versión 2.7) se eligió para el desarrollo de este proyecto. Su tipado dinámico, programación orientada a objetos con una sintaxis de simple aprendizaje, soporte para varias bases de datos, librerías y funciones preestablecidas. También posee un gran soporte técnico gracias a su comunidad.

El SDK de Python brinda una librería para la comunicación con la tarjeta Cyton. Retorna un arreglo con los datos adquiridos por sus 8 canales y otro con 3 canales auxiliares. Para realizar el proceso de acondicionamiento de la señal, se utilizó la librería pyseeg (basada en Cyton). Esta librería realiza el filtrado IIR, pasa-banda y pasa-bajo.

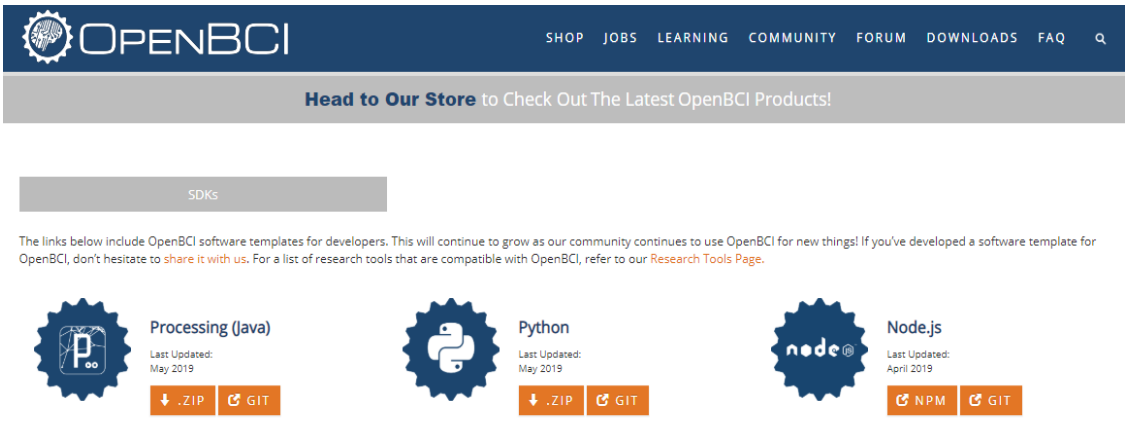


Figura 3.3. Página web de descarga de la empresa Open BCI.

El proyecto fue realizado con PyQt 4, compatible con Python 2.7. Esta biblioteca incluye abstracciones de sockets de red, subprocesos, Unicode, expresiones regulares, bases de datos SQL, SVG, OpenGL, XML, un navegador web completamente funcional, un sistema de ayuda, un marco multimedia, así como una amplia colección de widgets GUI. Además, brinda facilidad en el desarrollo de interfaz gráfica gracias a su QtDesigner [29].

La base de datos utilizada fue SQLite. Es una biblioteca en lenguaje C que implementa un motor de base de datos SQL pequeño, rápido, autónomo, de alta fiabilidad y completo. SQLite es el motor de base de datos más utilizado en el mundo. SQLite está integrado en todos los teléfonos móviles y en la mayoría de las computadoras, y se incluye en innumerables aplicaciones que las personas usan todos los días [30].

Para lograr que el programa funcione en cualquier computadora con un sistema operativo Windows o Linux, sin la necesidad de instalar Python y sus librerías, se utilizó Cx_Freeze. Es

un conjunto de scripts y módulos para congelar scripts de Python en ejecutables de la misma manera que lo hacen py2exe y py2app. A diferencia de estas dos herramientas, Cx Freeze es multiplataforma y debería funcionar en cualquier plataforma en la que Python funcione. Requiere Python 2.7 o superior y funciona con Python 3 [31].

Figura 3.4. Herramientas utilizadas para el desarrollo del software.

3.2.1. Lógica del software

Mediante diagramas de flujo se muestra la lógica general del programa en el Anexo 3, utilizada para el diseño de este; cada proceso que la compone se detalla en otro diagrama de flujo. El proceso de evaluación de la extremidad sana y la amputada es el mismo.

Las librerías empleadas en el proyecto son: Pyseeg (OpenBCI), sys, os, datetime, PyQt4, QtCore y QtGui, sqlite3, pickle, numpy, serial, matplotlib y reportlab.

Durante el avance del proyecto se implementaron varias soluciones para hacer posible la lógica antes mencionada. Uno de los inconvenientes más significativo fue presentar en la interfaz gráfica la señal electromiográfica adquirida y procesada en tiempo real.

El SDK de Python que se utilizó para la conexión con la tarjeta de biosensado Cyton cuenta con una librería denominada *"open_bci_v3"*. Esta recibe los datos EMG por un lapso preestablecido de segundos. Sin embargo, para la aplicación no es posible establecer un tiempo de toma de datos ya que esto depende del ritmo del paciente. Por tanto, fue necesario contar con una señal de inicio y paro representadas por botones en la interfaz.

Pero estas señales no realizaban su función a causa de un bucle existente en la librería, que se detenía al concluir el lapso o al activarse una función denominada *"stop"*. Esto genera que el programa no responda durante la adquisición.

Después de investigar el problema a fondo se llegó a 2 conclusiones: 1) Se crea un bucle infinito que abarca por completo el procesador del computador e impide la ejecución del resto del software. 2) El procesador no soporta una adquisición en tiempo real.

Entonces, se dio solución mediante la implementación de otro hilo de programación para el libre desempeño de la interfaz (Figura 3.5). Esto fragmenta las tareas permitiendo que se ejecuten al mismo tiempo, accediendo a los recursos de manera compartida y reduciendo su consumo.

También se segmento la adquisición de datos, es decir, cada 20 se realiza su procesamiento y se muestra en la gráfica. Visualizando la información en cuasi tiempo real y almacenando en otras variables toda la información adquirida.

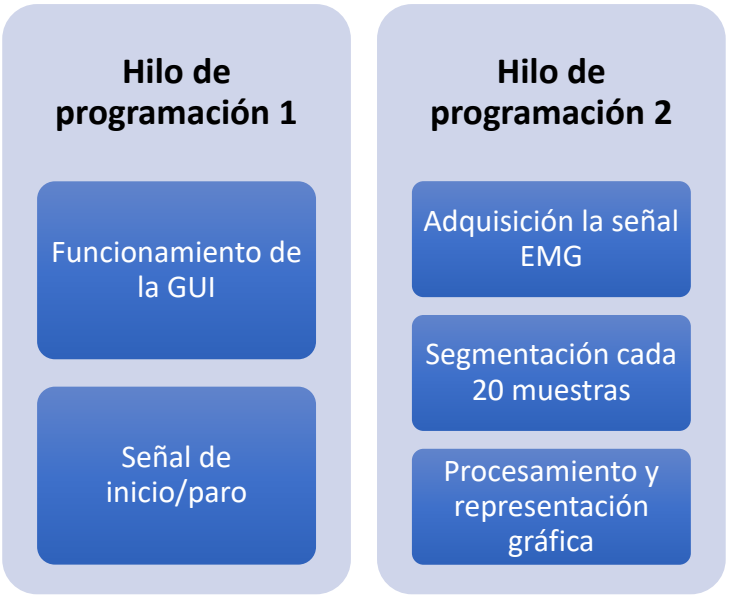


Figura 3.5. Lógica para el funcionamiento de las gráficas en cuasi tiempo real.

Al momento de convertir el script de Python a un archivo ejecutable. Existieron dificultades de compatibilidad con las librerías utilizadas. Se intento con varias herramientas, sin embargo, la única que funciono fue Cx_Freeze. Esta posee algunos inconvenientes con las librerías de numpy, scipy, pyqtgraph o matplotlib. Por ende, fue necesario declarar manualmente en el s*etup* aquellas que no eran reconocidas en la conversión.

3.2.2. Desarrollo de la interfaz gráfica de usuario

Contiene 3 ventanas principales y un cuadro de dialogo. A continuación, se describe cada una:

- **Pantalla principal:** Es la primera pantalla que se muestra al abrir el programa. Permite la navegación mediante sus botones *Registro*, *Evaluación* e *Historia clínica* como se muestra en Figura 3.6 sección A. Además, muestra los pacientes registrados en la nómina (Figura 3.6 sección B), posee un explorador de los pacientes (Figura 3.6 sección C) con botones para Editar, Eliminar y Ayuda.

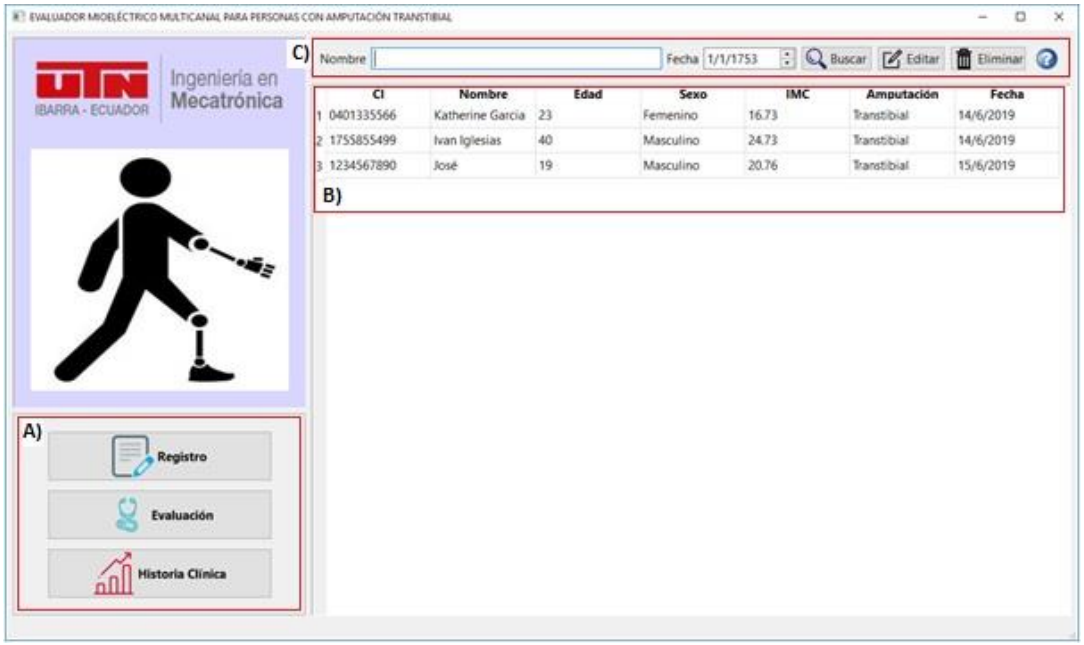


Figura 3.6. Pantalla Principal.

- **Pantalla de registro:** Es un cuadro de dialogo, donde se ingresa los datos del paciente para ser almacenados en la base de datos, ver Figura 3.7.

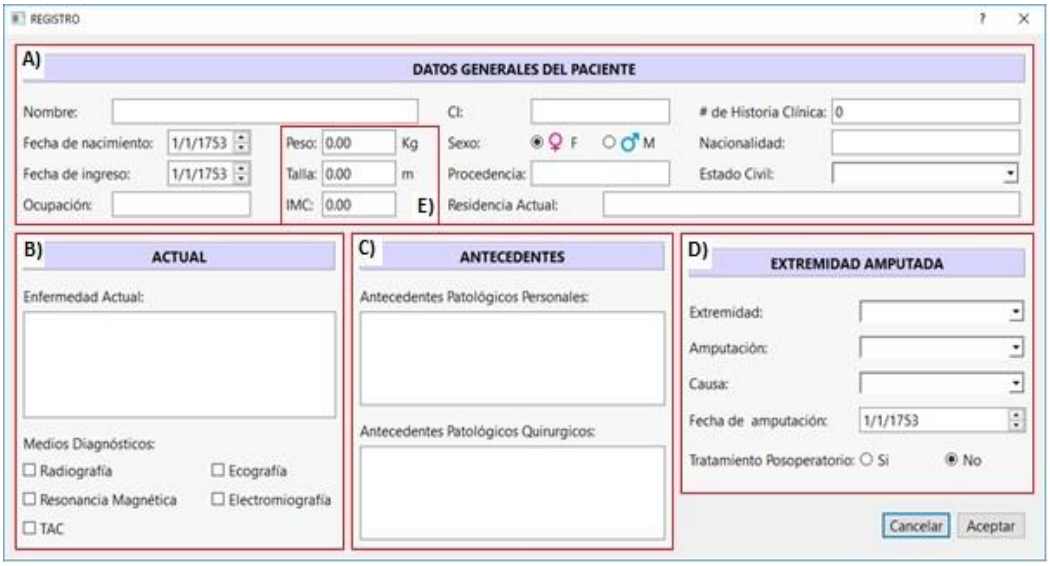


Figura 3.7. Pantalla de Registro

Se solicitan los datos generales (Figura 3.7 sección A): el nombre, cedula, numero de historia clínica, sexo, fecha de nacimiento, peso, talla, nacionalidad, procedencia, residencia actual, ocupación, fecha de ingreso; datos de la extremidad amputada (Figura 3.7 sección D):

extremidad, tipo de amputación, causa, fecha de amputación, tratamiento posoperatorio; información médica actual (Figura 3.7 sección B): enfermedad actual, medios diagnósticos; antecedentes patológicos personales y quirúrgicos (Figura 3.7 sección C).

Se calcula el índice de masa corporal (IMC) después de haber ingresado la talla y el peso (Figura 3.7 sección E).

- **Pantalla de evaluación:** Realiza la evaluación de la condición muscular de la extremidad amputada, para los movimientos de flexión y extensión de la rodilla, como se presenta en la Figura 3.8. Donde se muestra el nivel de fuerza muscular en mV, la comparación de la extremidad sana del 0% al 100% y la puntuación cualitativa, ver Figura 3.8 sección B. Se visualiza una imagen con los puntos motores de los músculos a evaluarse (Figura 3.8 sección A) para un mejor entendimiento con el fisioterapeuta.

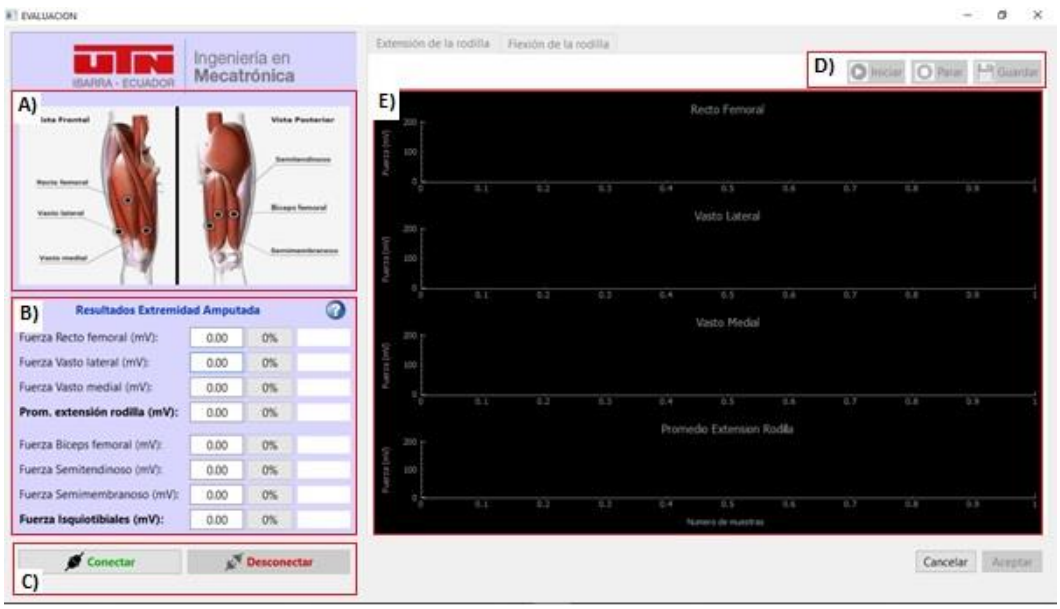


Figura 3.8. Pantalla de Evaluación

Los botones de conexión y desconexión (Figura 3.8 sección C) admiten que el usuario entienda que es necesario conectar el equipo al paciente para comenzar la medición. Cada pestaña contiene botones de inicio, parar y guardar (Figura 3.8 sección D). Estos dan comienzo

y fin a la valoración, mostrando gráficas de la activación muscular. Al guardar, aparecen los resultados de la valoración. Durante la adquisición de la información las gráficas presentan las señales EMG procesadas como un indicativo de lo que está sucediendo (Figura 3.8 sección E).

Se muestra la misma pantalla para la evaluación de la extremidad sana, sin el nivel de fuerza en porcentaje ni la puntuación cualitativa.

- **Pantalla de historia clínica:** Se visualiza la información del paciente y los resultados de las evaluaciones de forma cronológica. En la Figura 3.9 sección A se muestran los datos generales del paciente, sección B la información médica, y sección D los datos de la extremidad amputada.

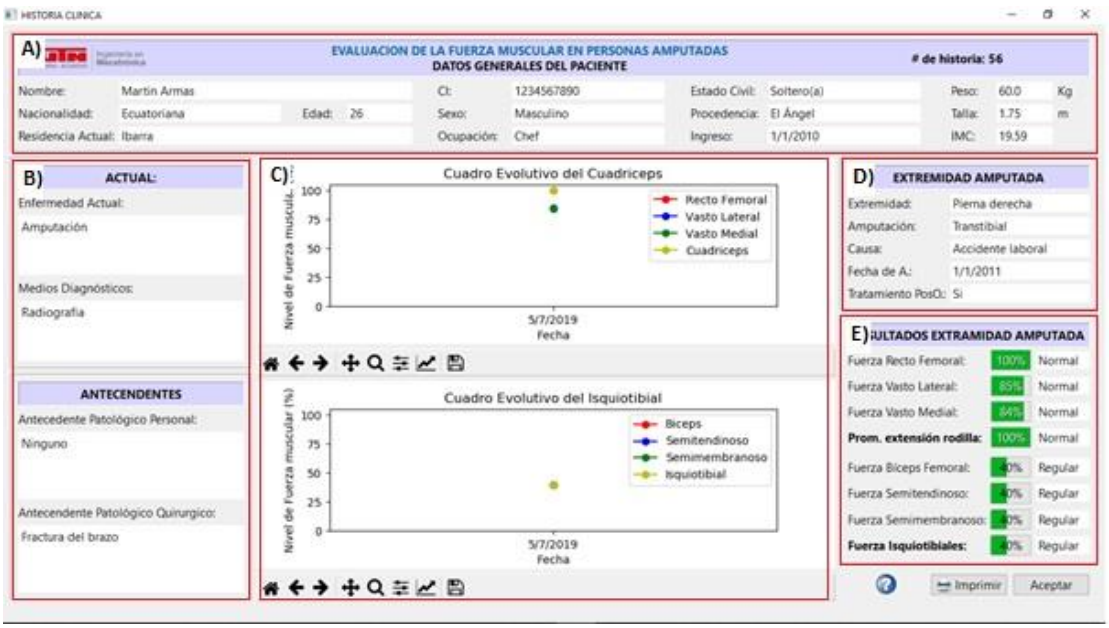


Figura 3.9. Pantalla de Historia Clínica

En el centro de la pantalla se visualizan los cuadros evolutivos correspondiente a las diferentes evaluaciones de la extremidad amputada a lo largo del tiempo, tanto para los músculos que componen el cuádriceps como del isquiotibial (Figura 3.9 sección C). Para una mejor visualización en la Figura 3.9 sección E se muestran los resultados de la posición

seleccionada en el cuadro evolutivo. Posee un botón *Imprimir* que genera un archivo PDF con la información.

El reporte en formato PDF se guarda en la carpeta de ubicación del programa, en la Figura 3.10 se visualiza un ejemplo real.

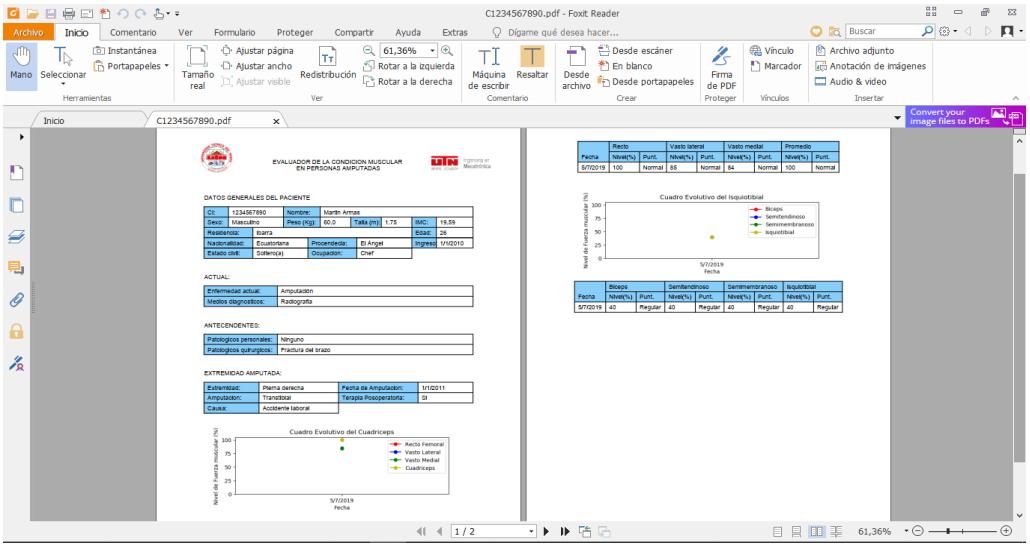


Figura 3.10. Reporte en formato PDF

3.3. Pruebas y análisis de resultados

A lo largo del proceso de investigación se contó con la colaboración de personal de salud como fisioterapeuta y protesista que supervisaron el protocolo de pruebas a los pacientes. Contribuyendo con su diagnóstico en base a la evaluación manual para realizar una comparación y la opinión sobre el software desarrollado en la ejecución de las correcciones pertinentes. El protocolo de las pruebas es el siguiente:

- Registro de la información de paciente.
- Limpieza de la piel del paciente y si se requiere depilación de la zona donde se ubicarán los electrodos.
- Ubicación de los electrodos en los puntos motores de la extremidad (Figura 3.11).

- Ejecución de la evaluación en la extremidad sana y posteriormente en la amputada.
- Visualización de los resultados y comparación con el diagnóstico del fisioterapeuta.

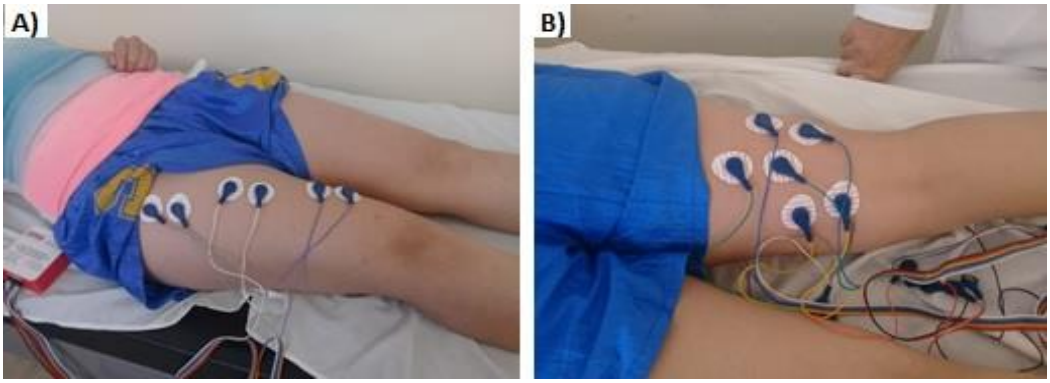


Figura 3.11. Colocación de los electrodos. A. Músculos del cuádriceps para extensión de rodilla. B. Músculos del isquiotibial para flexión de rodilla.

En la evaluación de extensión de rodilla el paciente debe estar sentado (Figura 3.12). Las manos descansan sobre la mesa a los lados para mantener la estabilidad, o se sujeta a la mesa. El paciente se incline hacia atrás, para disminuir la tirantez de los músculos posteriores. No es admisible que el paciente realice una hiperextensión de la rodilla, porque esto bloquea en la posición.

Figura 3.12. Posición del paciente para la evaluación de extensión de rodilla sin carga.

En la valoración de flexión el paciente debe estar en posición decúbito prono, con los brazos estirados y los pies sobresaliendo del borde de la mesa. Flexionando la rodilla mientras mantiene la pierna en rotación de equilibrio (Figura 3.13).

Figura 3.13. Posición del paciente para la evaluación de flexión de rodilla.

La adquisición de datos se realiza durante 3 contracciones con intervalos de 3 segundos para cada movimiento, ya que posterior a esto los músculos sufren fatiga y la información no contribuiría a la evaluación.

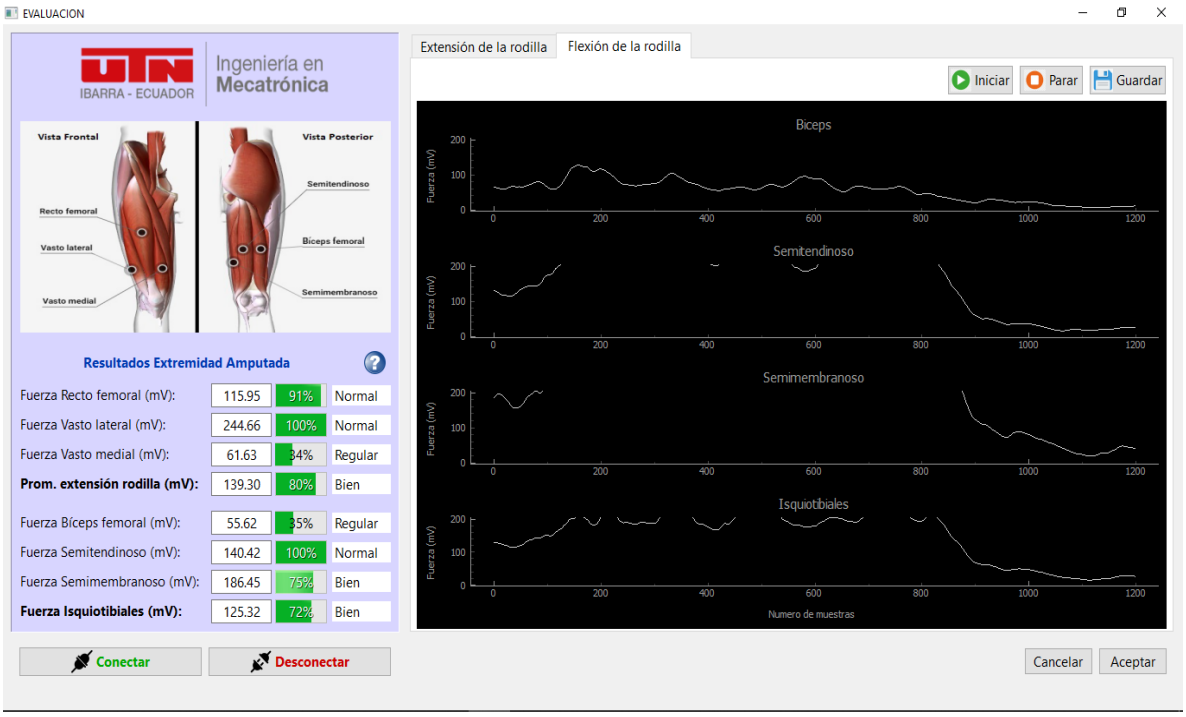


Figura 3.14. Evaluación en una persona sana.

Con el fin de comprobar la funcionalidad del programa se realizaron pruebas previas a lo largo del desarrollo, se ejecutaron mediciones en músculos del brazo, únicamente para corroborar que el software adquiere y procesa la información. Así, antes de realizar la evaluación en personas amputadas se evaluó a personas sanas, comparando los resultados con la escala de Daniels en base a la opinión del fisioterapeuta (Figura 3.14).

Las pruebas se realizaron en personas sanas y a 3 personas con amputación transtibial. En la muestra se consideró personas con amputación unilateral debido al protocolo de evaluación y un IMC entre 18 y 30. Se excluyeron personas que posean alguna lesión en la extremidad sana o que posean enfermedades que influyan en la valoración como la diabetes, parálisis cerebral, entre otras. En la Tabla 3.1 se muestran los datos de las personas evaluadas. En donde, el género está representado por F para femenino y M para masculino; el estado con S para sano y A para amputado.

Tabla 3.1. Datos generales de los pacientes

Pac.	**IMC**	**Edad**	**Genero**	**Estado**	**Fecha de amputación**	**Causa**	**Terapia PosO.**
1	24.44	22	F	S	-	-	-
2	22.58	17	F	S	-	-	-
3	20.34	26	M	S	-	-	-
4	19.61	31	M	S	-	-	-
5	20.76	22	M	A	1/6/2007	Accidente de transito	Si
6	25.86	48	M	A	1/2/2014	Enfermedad adquirida	Si
7	29.24	49	F	A	1/1/1985	Enfermedad adquirida	Si

Los datos obtenidos durante la valoración de los extensores y flexores de rodilla se muestran en la Tabla 3.2 y Tabla 3.3. En donde, se muestra el nivel de fuerza muscular en mV, porcentaje y la puntuación cualitativa (N: Normal, B: Bien, R: Regular, M: Mal, E: Escaso, S: Sin actividad). La primera fila de cada paciente corresponde a los valores de referencia, que se señala con la letra R.

Tabla 3.2. Datos obtenidos en la evaluación de los extensores de rodilla

Pac.	Extremidad	Recto F.			Vasto Lateral			Vasto Medial			Promedio		
		mV	%	P	mV	%	P	mV	%	P	mV	%	P
1	Derecha(R)	126.33	100	N	230.83	100	N	180.72	100	N	173.08	100	N
	Izquierda	115.95	91	N	244.66	100	N	61.63	34	R	139.3	80	B
2	Derecha(R)	182.37	100	N	192.76	100	N	263.69	100	N	211.98	100	N
	Derecha	114.96	63	R	30.71	15	R	174.73	66	B	105.64	49	R
3	Izquierda(R)	96.38	100	N	123.4	100	N	115.65	100	N	109.56	100	N
	Izquierda	115.28	100	N	108.44	87	B	81.71	70	B	100.57	91	N
4	Izquierda(R)	166.74	100	N	93.85	100	N	72.33	100	N	109.81	100	N
	Derecha	110.15	66	B	70.85	75	B	126.18	100	N	101.24	92	N

En la Tabla 3.4 se comparan las puntuaciones obtenidas con el grado de la prueba manual realizada por el fisioterapeuta. En base a toda la información se deduce lo siguiente:

La primera se realizó tomando valores de referencia en la pierna derecha y se evaluó para grado 5 la pierna izquierda. Siendo la derecha dominante, se considera lógico obtener valores menores en la izquierda.

Tabla 3.3. Datos obtenidos en la evaluación de los flexores de rodilla

Pac.	Extremidad	Bíceps F.			Semitendinoso			Semimembranoso			Isquiotibial		
		mV	%	P	mV	%	P	mV	%	P	mV	%	P
1	Derecha(R)	155.82	100	N	115.17	100	N	248.44	100	N	171.89	100	N
	Izquierda	55.62	35	R	140.42	100	N	186.45	75	B	125.32	72	B
2	Derecha(R)	285.45	100	N	163.01	100	N	229.56	100	N	224.49	100	N
	Derecha	92.02	32	R	68.11	41	R	85.67	37	R	80.63	35	R
3	Izquierda(R)	107.3	100	N	156.01	100	N	74.44	100	N	111.43	100	N
	Izquierda	97.93	91	N	126	80	B	69.8	93	N	96.78	86	B
4	Izquierda(R)	239.25	100	N	74.52	100	N	155.03	100	N	155.2	100	N
	Derecha	126.25	52	R	198.62	100	N	130.41	84	B	149.1	96	N

Para la segunda persona se considera la pierna derecha como referencia y se evalúa en la misma. Se realizo la prueba para grado 3 por tanto la puntuación obtenida concuerda.

Tabla 3.4. Resumen y comparación de la evaluación realizada con la prueba manual en personas sanas

Pac.	Puntuación Extensores	Puntuación Flexores	Grado valoración de Daniels
1	Bien	Bien	5
2	Regular	Regular	3
3	Normal	Bien	5
4	Normal	Normal	5

En la tercera persona se realiza la evaluación en la pierna izquierda para grado 5. Se observa una discrepancia en los resultados de los flexores, sin embargo, se considera que la persona no utilizo toda su fuerza.

En los pacientes amputados se realizaron pruebas bajo resistencia máxima, es decir se aplicó fuerza al muñón, y pruebas solo con la prótesis. En el último caso no es posible ejercer resistencia debido a que la prótesis no posee buena fijación a la extremidad. Todos los casos presentan amputación tercio proximal, llevan mucho tiempo utilizando prótesis y no poseen hipersensibilidad en el muñón. Esto permitió realizar las pruebas sin malestar en el paciente.

3.3.1. Caso 1:

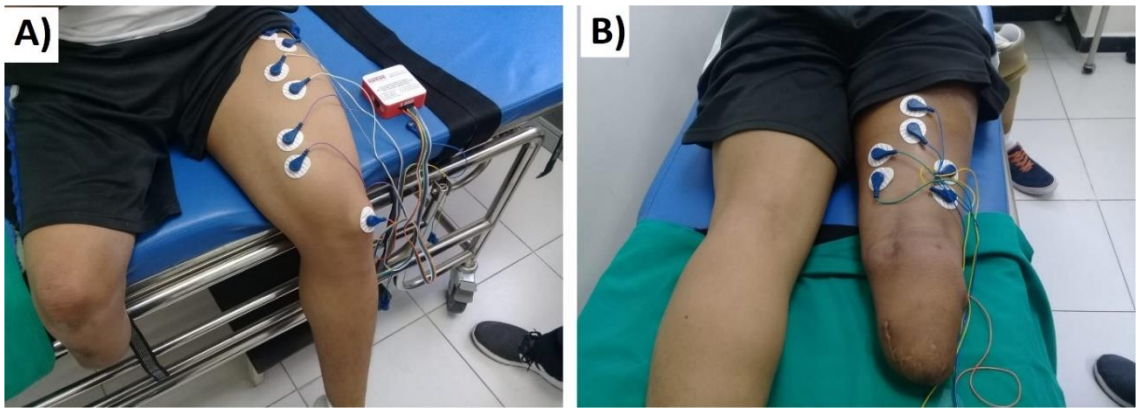


Figura 3.15. Evaluación de fuerza en el paciente 5. A. Evaluación en extensión de rodilla extremidad sana. B. Evaluación en flexión de rodilla extremidad amputada.

El nivel de fuerza muscular en milivoltios se observa en la Figura 3.16 y en porcentaje en la Figura 3.17. En ambos casos se aprecia que existe una gran diferencia entre la extremidad sana y la amputada. La extremidad sana obtuvo valores cercanos a los 300 mV; la extremidad amputada bajo resistencia máxima cantidades aproximadas a los 70 mV y solo con la prótesis cercanos a los 50 mV.

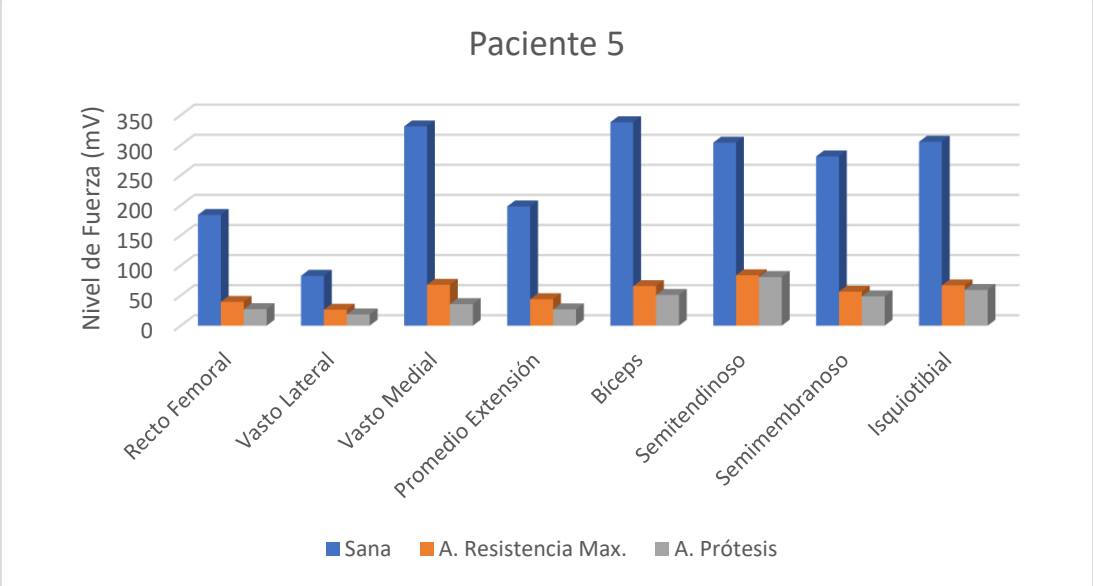


Figura 3.16. Resultados del paciente 5 en mV.

Por lo cual, la extremidad amputada bajo resistencia máxima en promedio de extensión obtuvo 22% y en flexión 21%. Solo con la prótesis en extensión 13% y flexión 19%.

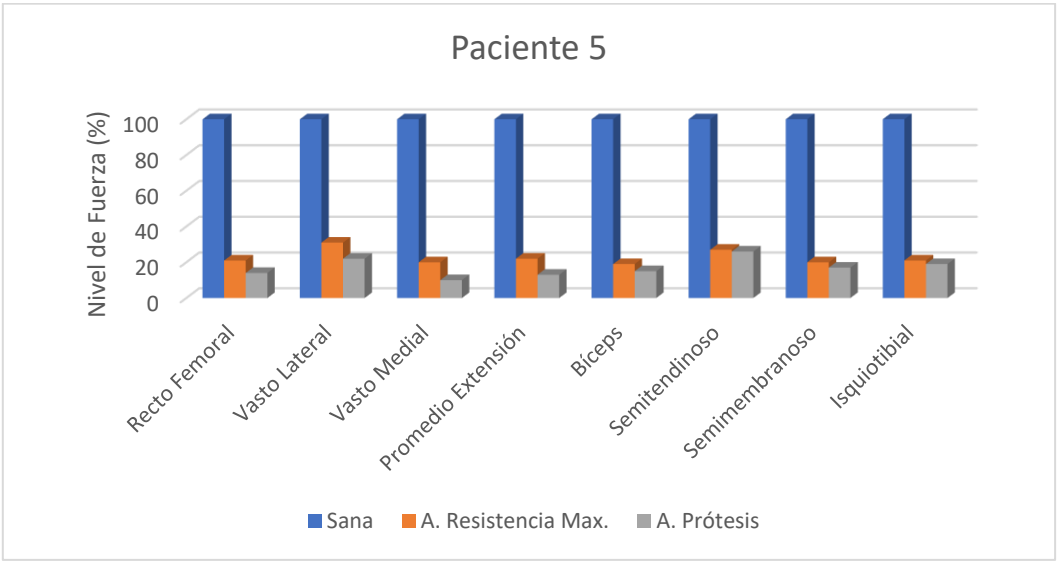


Figura 3.17. Resultados del paciente 5 en porcentaje.

3.3.2. Caso 2:

El nivel de fuerza muscular en milivoltios se observa en la Figura 3.18 y en porcentaje en la Figura 3.19. La extremidad sana obtuvo valores cercanos a los 140 mV; la extremidad

amputada bajo resistencia máxima cantidades aproximadas a los 80 mV y solo con la prótesis cercanos a los 40 mV.

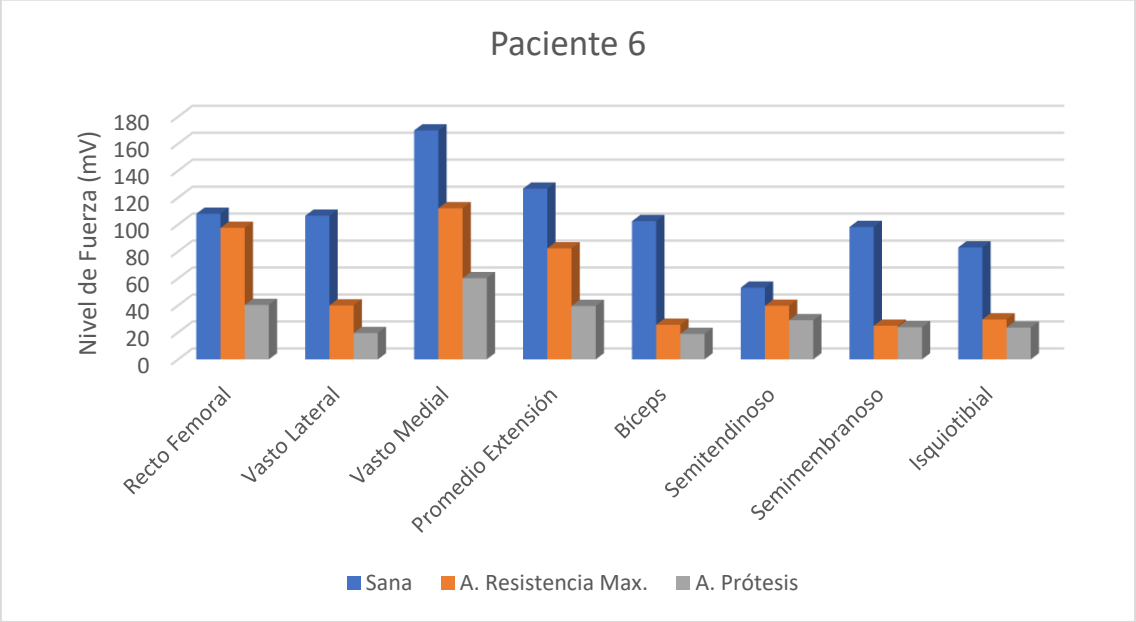


Figura 3.18. Resultados del paciente 6 en mV.

Por lo cual, la extremidad amputada bajo resistencia máxima en promedio de extensión obtuvo 65% y en flexión 35%. Solo con la prótesis en extensión 31% y flexión 28%.

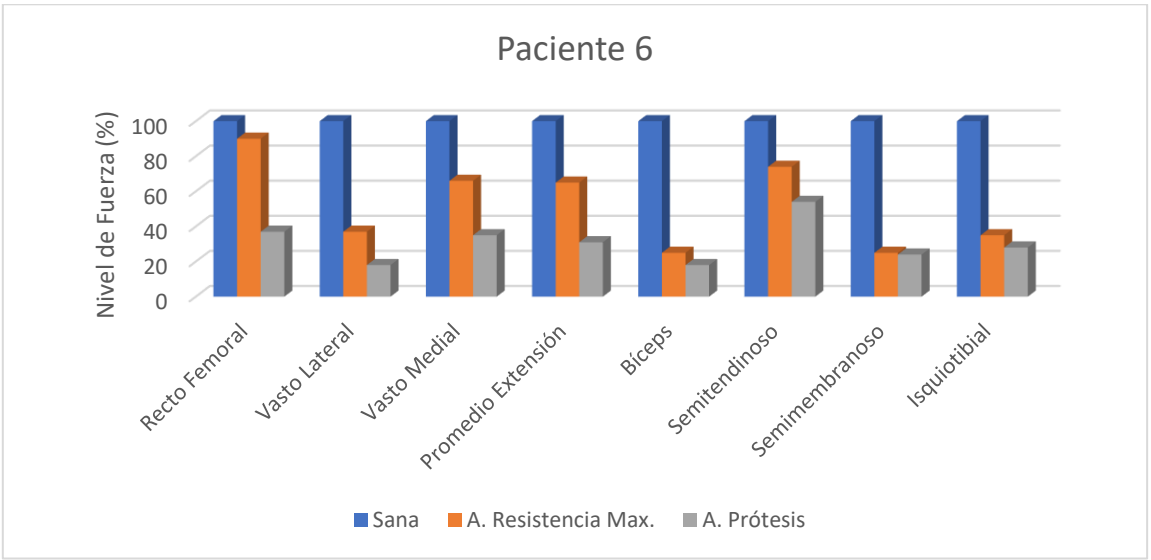


Figura 3.19. Resultados del paciente 6 en porcentaje.

3.3.3. Caso 3:

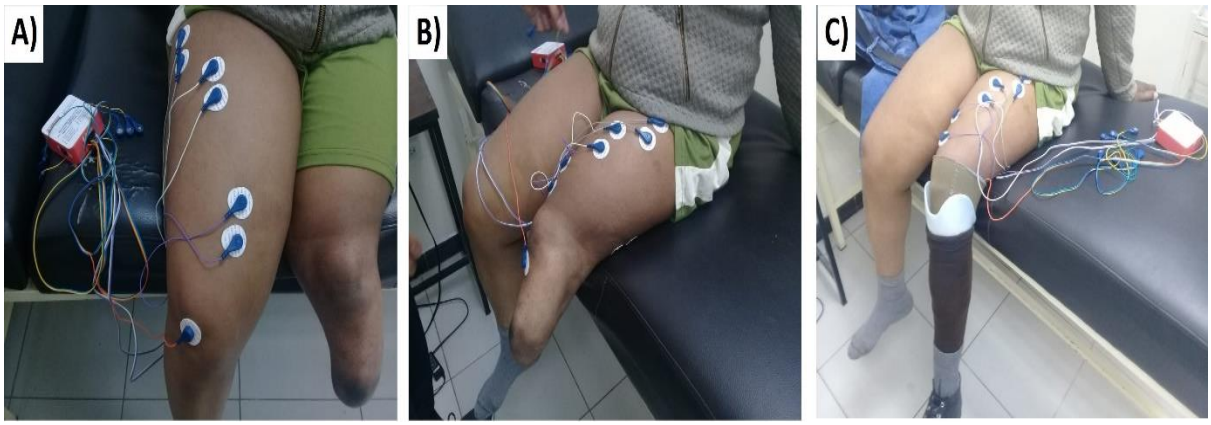


Figura 3.20. Evaluación de fuerza en extensión al paciente 7. A. Extremidad sana. B. Extremidad amputada bajo resistencia máxima. C. Extremidad amputada con prótesis.

El nivel de fuerza muscular en milivoltios se observa en la Figura 3.21 y en porcentaje en la Figura 3.22. La extremidad sana obtuvo valores próximos a los 50 mV; la extremidad amputada bajo resistencia máxima cantidades cercanas a los 20 mV y solo con la prótesis 20 mV.

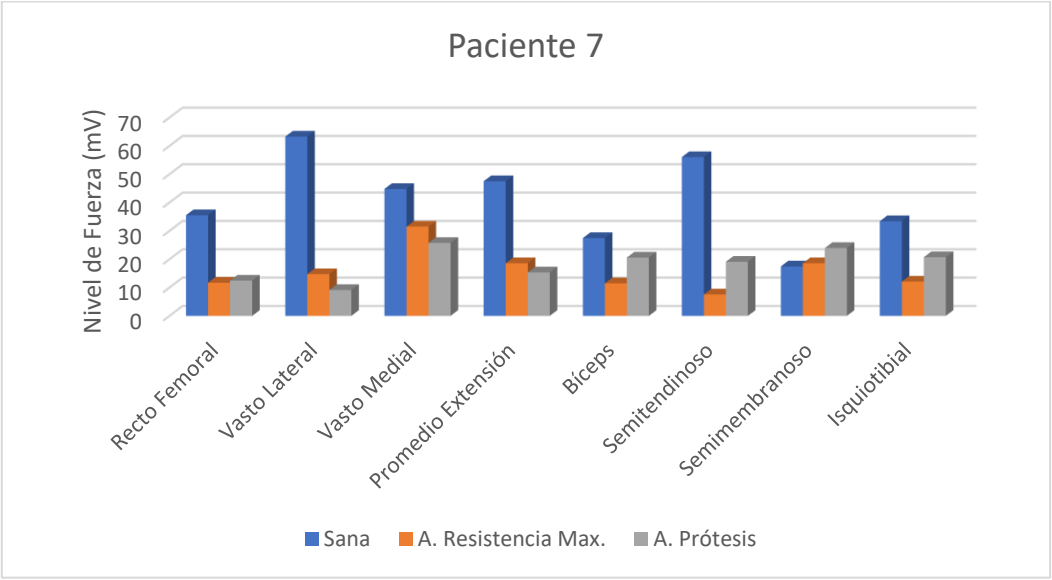


Figura 3.21. Resultados del paciente 7 en mV.

En este caso la persona posee mayor tejido adiposo, provocando la disminución de los valores. Por lo cual, la extremidad amputada bajo resistencia máxima en promedio de extensión obtuvo 39% y en flexión 32%. Solo con la prótesis en extensión 36% y flexión 61%.

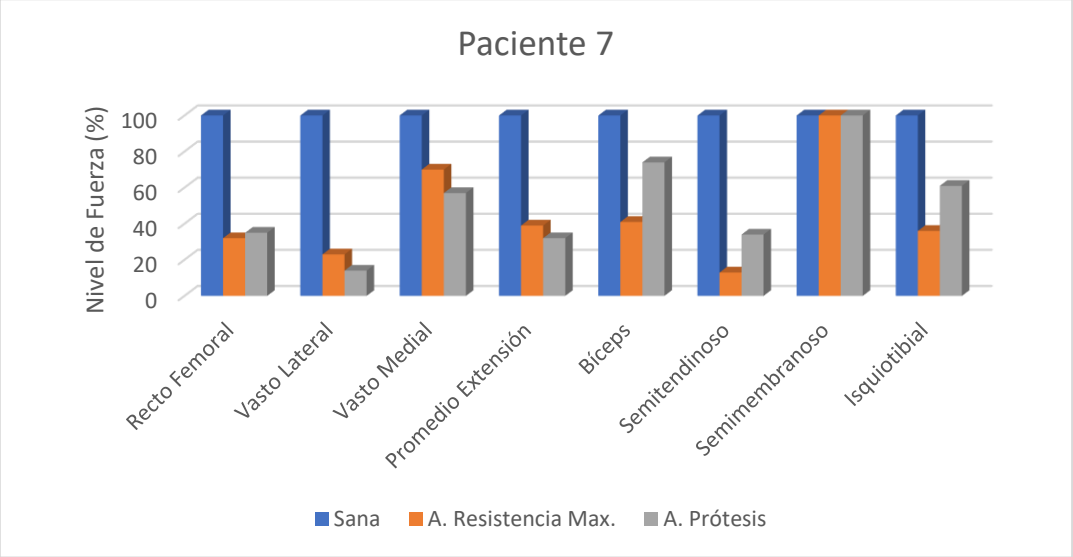


Figura 3.22. Resultados del paciente 7 en porcentaje.

Los resultados indican que en personas con amputación transtibial la extremidad amputada posee un nivel de fuerza muscular mucho menor que la sana. Es posible que esto se deba a que las personas en su condición usualmente no recuperan a ese punto la musculatura, sin importar el tiempo de rehabilitación que lleven. En todos los casos los pacientes llevan varios años utilizando su prótesis.

Los datos obtenidos dan a notar que la evaluación bajo resistencia máxima no supera por mucho a la que solo utiliza prótesis. La última busca simular una valoración de 3 según las pruebas de Daniels. Esto indica que existe la posibilidad de que el paciente no use toda su fuerza, siendo este el caso se debería a la corta distancia existente entre el punto de aplicación de la resistencia y la rodilla, que genera poco torque.

CONCLUSIONES

- La revisión literaria y consultas realizadas a expertos coinciden que los músculos extensores de la rodilla a los que se tiene acceso son: recto femoral, vasto lateral y vasto medio, es decir los músculos que conforman el cuádriceps. Los músculos flexores de rodilla son los isquiotibiales; se tiene acceso al bíceps femoral, semitendinoso y semimembranoso.
- En base al criterio del personal médico se definen los parámetros mínimos para la evaluación en reposo, como: el índice de masa corporal (IMC), el tipo de amputación, enfermedades que influyan en la misma como diabetes, parálisis cerebral, entre otras; los músculos que intervienen en el movimiento y el nivel de fuerza muscular de la extremidad sana.
- En colaboración del fisioterapeuta se determinó un protocolo comparativo entre el nivel de fuerza muscular de la extremidad sana con la amputada, a causa de la variación en los niveles de fuerza de una persona a otra. La metodología de adquisición y procesamiento de las señales EMG extrae únicamente el nivel de fuerza, acorde a lo antes mencionado.
- La investigación de diferentes tarjetas electrónicas para lectura de señales mioeléctricas, concluyó en la implementación de un sistema de adquisición de señales EMG de 8 canales inalámbrico (Cyton), que fue acondicionado para su aplicación.
- Se desarrollo un software que registra la información del paciente, juntamente con los parámetros para la valoración. Ejecutando el protocolo de evaluación se muestra las gráficas de la señal EMG en tiempo real (con un retraso de 20 muestras más el tiempo del procesador), gracias a un segundo hilo de programación. Se presenta el nivel de fuerza muscular en mV y en porcentajes (0-100%) con su puntuación cualitativa. La

información se almacena en la base de datos con fechas, visualizando en la historia clínica mediante un cuadro evolutivo.

- En base a los resultados obtenidos se concluye que la información reunida concuerda con la condición de los pacientes, a excepción de la evaluación bajo resistencia máxima. En este caso los valores deben ser más altos, sin embargo, la distancia entre el punto de aplicación y la rodilla es muy corta, impidiendo que se realice el suficiente torque.

RECOMENDACIONES

- El trabajo únicamente realiza la evaluación en reposo para los movimientos de flexión y extensión de la rodilla, es recomendable para investigaciones a futuro agregar un estudio de los músculos flexores y extensores de cadera.
- Se recomienda añadir un estudio sobre la activación muscular en las fases de la marcha, es decir, la evaluación durante la marcha.
- En base a los resultados es necesario implementar un sistema de carga para obtener una mejor valoración contra resistencia máxima.
- Realizar pruebas con muestras que incluyan personas con diferentes condiciones físicas o que tengan una enfermedad que influya en los resultados como la diabetes y compararlas.

REFERENCIAS BIBLIOGRÁFICAS

[1] M. L. Ocampo, L. M. Henao y L. Vásquez, Amputación de miembro inferior: Cambios funcionales, inmovilización y actividad física, Bogotá: Universidad del Rosario, 2010.

[2] Y. Govantes Bacallao, C. J. Alba Gelabert y A. Arias Cantalapiedra, «Protocolo de actuación en la rehabilitación de pacientes amputados de miembro inferior,» *Revista Cubana de Medicina Física y Rehabilitación,* pp. 33-43, 2016.

[3] J. Cuasapaz, «Entrenador mioeléctrico para personas amputadas del miembro,» 2017.

[4] Programa de medicina y cirugía de la Universidad Tecnológica de Pereira, Persona con amputación. Guía de rehabilitación, Pereira, 2013.

[5] CONADIS, «Personas con discapacidad registradas,» 2018. [En línea]. Available: http://www.consejodiscapacidades.gob.ec/estadistica/index.html.

[6] CONADIS, «Causas de discapacidad,» mayo 2013. [En línea]. Available: http://www.consejodiscapacidades.gob.ec/wp-content/uploads/downloads/2015/05/causas_discapacidad_conadis.pdf.

[7] R. Drake, W. Vogl y A. Mitchell, Anatomía de Gray para estudiantes, Barcelona: Elsevier España, S.L.U., 2015.

[8] K. Moore, A. Dalley II y A. Agur, Anatomía de Moore con orientación clínica, Philadelphia: Wolters Kluwer Health, S.A., Lippincott Williams & Wilkins, 2013.

[9] Human Biodigital, «Músculos flexión y extensión de la rodilla,» 2019. [En línea]. Available: https://human.biodigital.com/view?id=2s15&type=bookmark.

[10] J. M. Rodríguez Martín, «Aplicación de corrientes de baja y media frecuencia,» de *Electroterapia en Fisioterapia*, Buenos Aires, Bogotá, Caracas, Madrid, México, Editorial Médica Panamericana, 2014, p. 113.

[11] G. Ñahuincopa Pariona, «Tratamiento fisioterapeútico en amputados debajo de rodilla del miembro inferior,» Perú, Universidad Inca Garcilaso De La Vega, 2017, pp. 32-33.

[12] V. Perez y V. Montoya Leal, «Valoración cuantitativa para la reincorporación ocupacional,» *Salud Uninorte,* vol. 32, nº 2, pp. 319-336, 2016.

[13] Sportplus center, «¿Qué es la electromiografía?,» 9 octubre 2014. [En línea]. Available: https://www.sportpluscenter.com/que-es-la-electromiografia/. [Último acceso: 1 noviembre 2018].

[14] Biomech Solutions, «Plataformas de fuerza,» 2016. [En línea]. Available: https://www.biomech-solutions.com/plataformas-fuerza-bertec.html. [Último acceso: 8 agosto 2019].

[15] Intituto de Biomecánica de Valencia, «NedDiscapacidad/IBV,» [En línea]. Available: http://analisisbiomecanico.ibv.org/index.php/productos/software-valoracion-biomecanica/neddiscapacidad-ibv. [Último acceso: 1 noviembre 2018].

[16] H. Hislop, J. Montgomery y B. Connolly, Daniels-Wothingham´s. Pruebas funcionales musculares, California: Marban.

[17] P. Arancibia Contalba, «Examen muscular, articular y mimica facia,» 22 mayo 2014. [En línea]. Available: https://es.slideshare.net/PELOINDAHOUSE/examen-muscular-articular-y-mimica-facial. [Último acceso: 1 noviembre 2018].

[18] Á. M. García Engeños, Sistema de adquisición multicanal para señales mioeléctricas, Madrid, 2015.

[19] R. Bednar, «Placa neuromuscular: partes, funciones y patologías.,» 17 abril 2017. [En línea]. Available: https://www.lifepersona.com/neuromuscular-plaque-parts-functions-and-pathologies. [Último acceso: 2 noviembre 2018].

[20] E. V. Cabrera Ávila y E. I. Montes Fernandez, Obtencion y análisis de señales electromiográficas de las articulaciones tibiofemoral y femororotuliana aplicada a la detección y rehabilitación de problemas musculares en el movimiento de la rodilla, Cuenca: Universidad Politécnica Sede Cuenca, 2012.

[21] J. Fernández, R. Acevedo y C. Tabernig, «Influencia de la fatiga muscular en la señal electromiográfica de músculos estimulados eléctricamente,» *Revista EIA*, nº 7, pp. 111-119, 2007.

[22] R. Hernández Sampieri, C. Fernández Collado y M. d. P. Baptista Lucio, Metodología de la investigación, México D.F.: McGRAW-HILL / Interamericana editores, S.A. DE C.V., 2014.

[23] D. F. Yépez Ponce, Tarjeta de adquisición y tratamiento de señales eléctricas provenientes del cuerpo humano para el accionamiento de prótesis transradiales, Ibarra, 2016.

[24] C. Riva Romeva, Diseño concurrente, Barcelona: Edicions UPC, 2002.

[25] OpenBCI, «Cyton Biosensing Board (8 canales),» 2018. [En línea]. Available: https://shop.openbci.com/products/cyton-biosensing-board-8-channel?variant=38958638542. [Último acceso: 23 11 2018].

[26] H. Caluguillin, Análisis de características de señales electromiográficas para la determinación de movimientos de una mano., Ibarra, 2019.

[27] S. Sastoque, C. Narváez y G. Garnica, «Metodología para la construcción de Interfaces Gráficas Centradas en el Usuario,» *Nuevas Ideas en Informática Educativa*, vol. 12, pp. 314 - 324, 2016.

[28] M. Angulo, «Heurísticas de usabilidad,» 2019 marzo 14. [En línea]. Available: https://www.tesseractspace.com/blog/heuristicas-de-usabilidad/.

[29] Riverbank Computing Limited, «¿Qué es PyQt?,» [En línea]. Available: https://riverbankcomputing.com/software/pyqt/intro. [Último acceso: 2 junio 2019].

[30] SQLite, «¿Qué es SQLite?,» [En línea]. Available: https://www.sqlite.org/index.html. [Último acceso: 2 junio 2019].

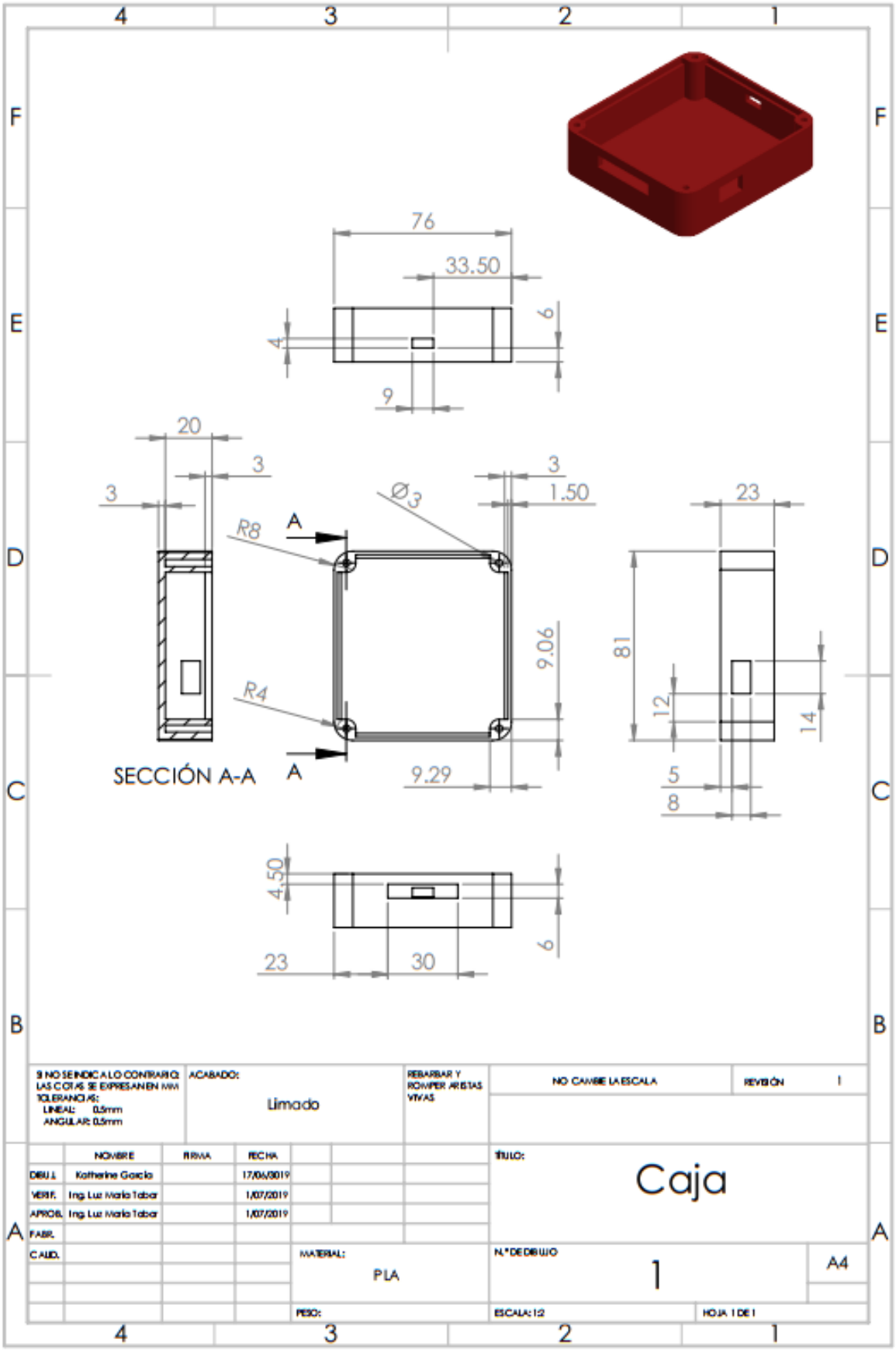
[31] A. Tuininga, «¡Bienvenido a la documentación de cx_Freeze!,» 2017. [En línea]. Available: https://cx-freeze.readthedocs.io/en/latest/. [Último acceso: 25 julio 2019].

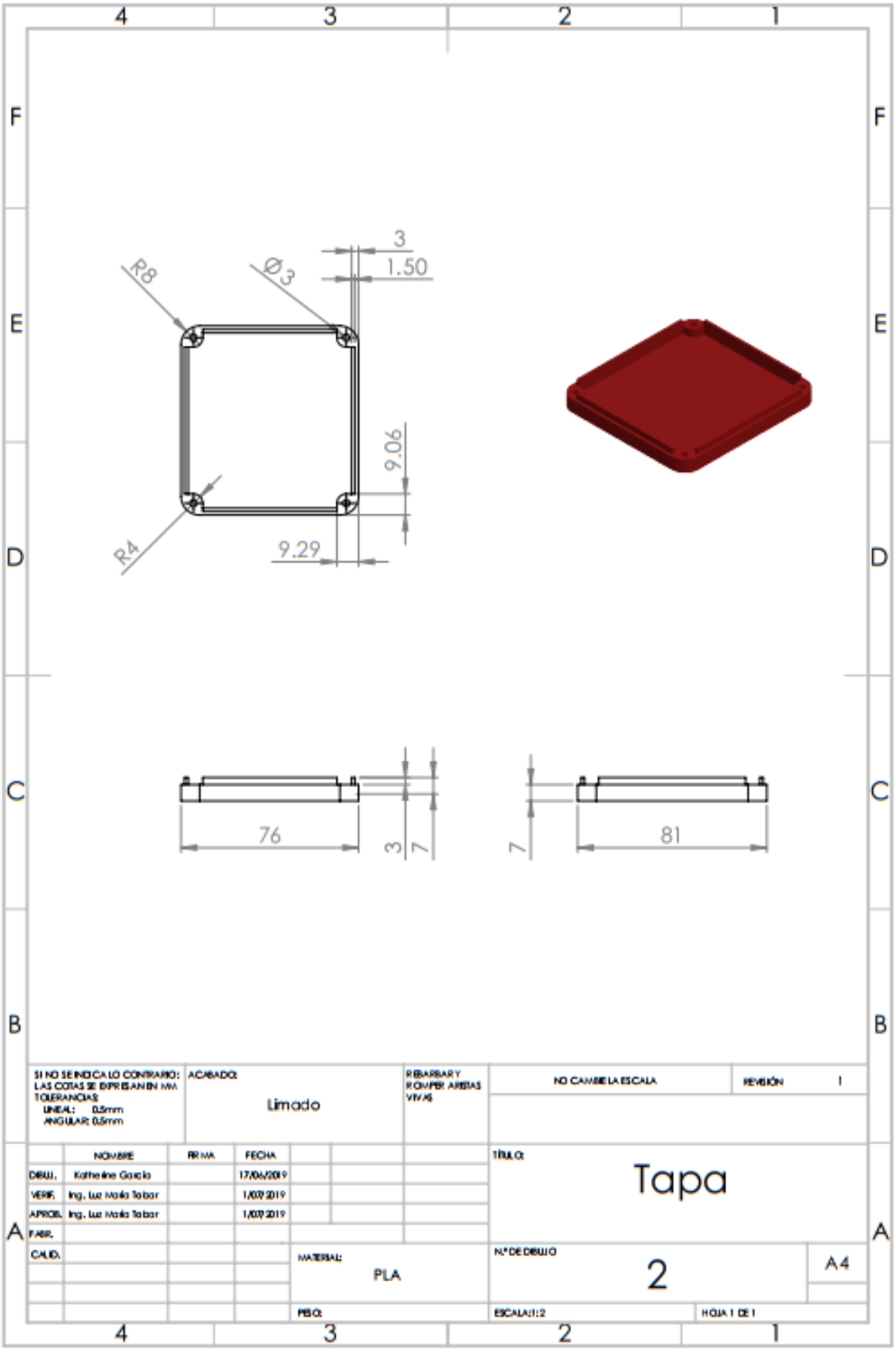
ANEXOS

Anexo 1. Especificaciones técnicas Tarjeta de biosensado Cyton

OpenBCI 32bit Board	
Canales	8
Compatibilidad	Electrodos activos y pasivos
Adquisición	ADS1299 ADC de Texas Instruments. Acelerómetro LIS3DH
Microcontrolador	PIC32MX250F128B con cargador de arranque chipKIT ™ (50MHz)
Conexión	Radio RFduino ™ de baja potencia Bluetooth ™
Resolución de datos del canal de bits	24
Ganancia programable	1, 2, 4, 6, 8, 12, 24
Tensión de operación digital	3.3V
Tensión de operación analógica	+/- 2.5V
Tensión de entrada	~ 3.3-12V
Periféricos	Ranura para tarjeta microSD
Pines	5 pines GPIO, 3 son analógicos
OpenBCI Dongle	
Conexión	RFD22301 módulo de radio de RFdigital ™
Convertidor	FT231X Convertidor de USB a serie de FTDI

Anexo 2. Planos de la caja




4
3
2
1
F
E
D
C
B
A
R8
Ø3
3
1.50
9.06
R4
9.29
76
3
7
7
81
SI NO SE INDICA LO CONTRARIO:
LAS COTAS SE EXPRESAN EN MM
TOLERANCIAS:
LINEAL: 0,5mm
ANGULAR: 0,5mm
ACABADO:
Limado
REBARBAR Y
ROMPER ARISTAS
VIVAS
NO CAMBIE LA ESCALA
REVISIÓN 1
NOMBRE
FIRMA
FECHA
DIBUJ. Katherine Garcia 17/06/2019
VERIF. Ing. Luz Maria Tobar 1/07/2019
APROB. Ing. Luz Maria Tobar 1/07/2019
FABR.
CALID.
TÍTULO:
Tapa
MATERIAL:
PLA
Nº DE DIBUJO
2
A4
PESO:
ESCALA:1:2
HOJA 1 DE 1

Anexo 3. Diagramas de flujo

Diagrama de flujo General

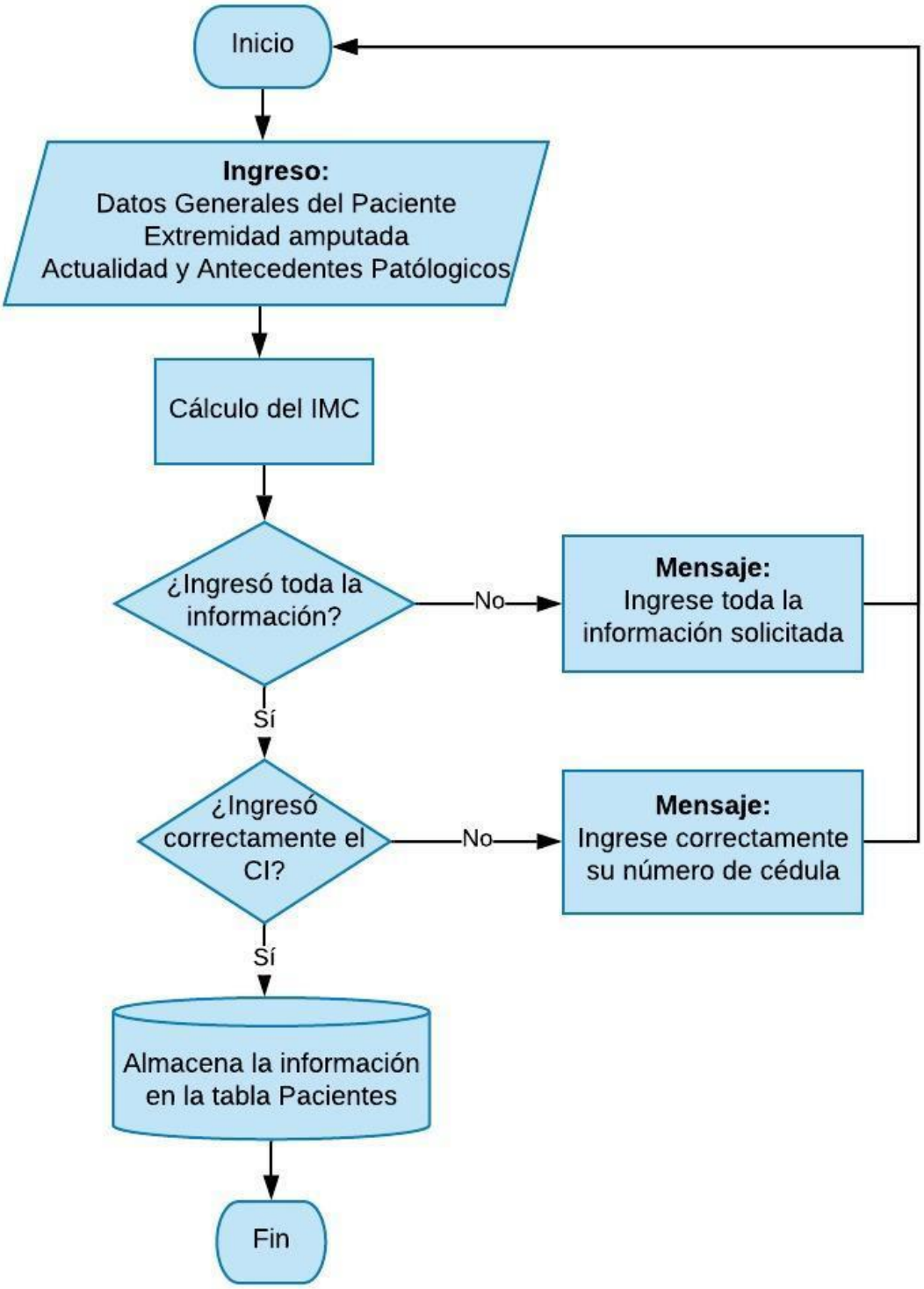


Diagrama de flujo del Registro de un paciente

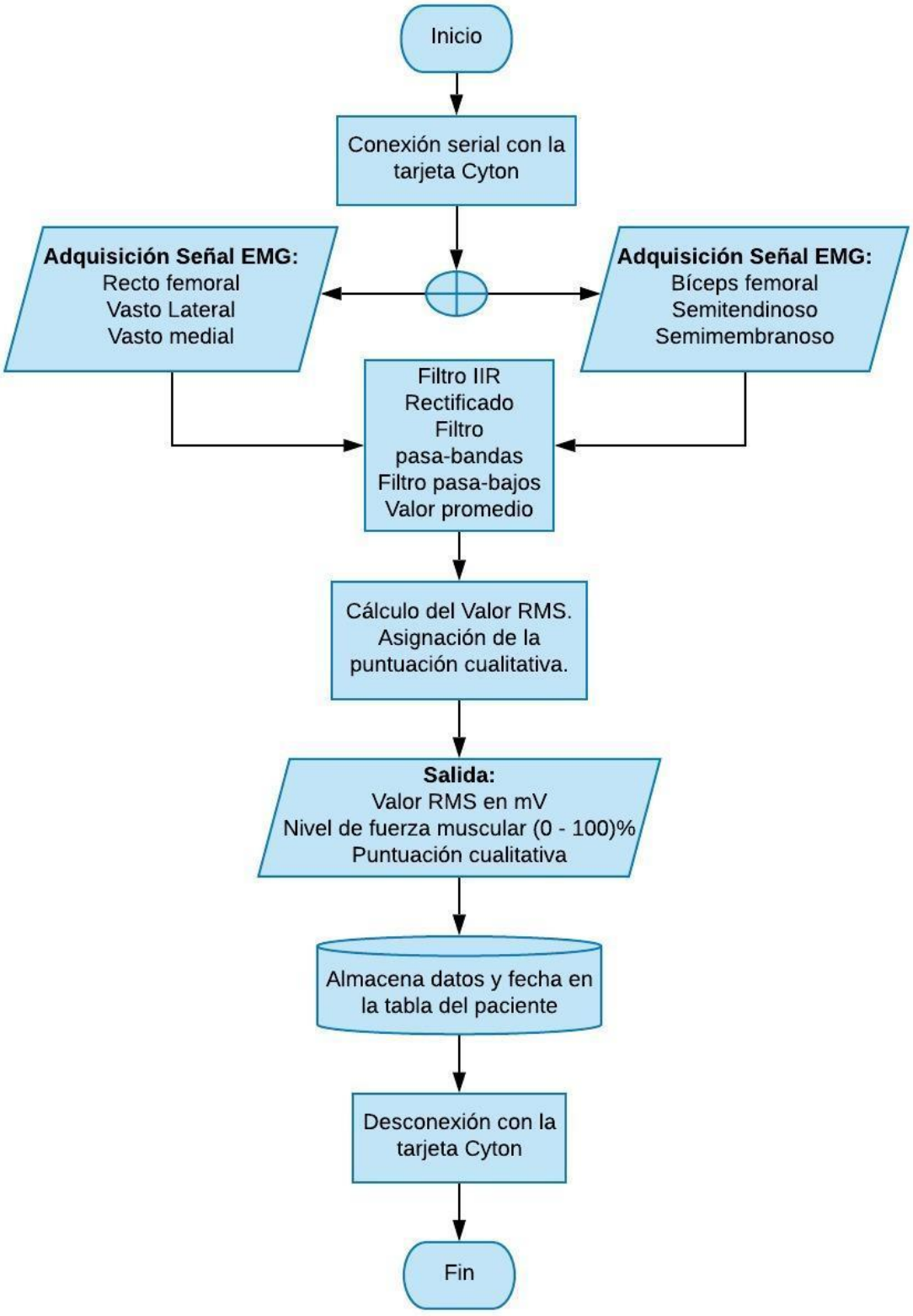


Diagrama de flujo de la Evaluación de un paciente

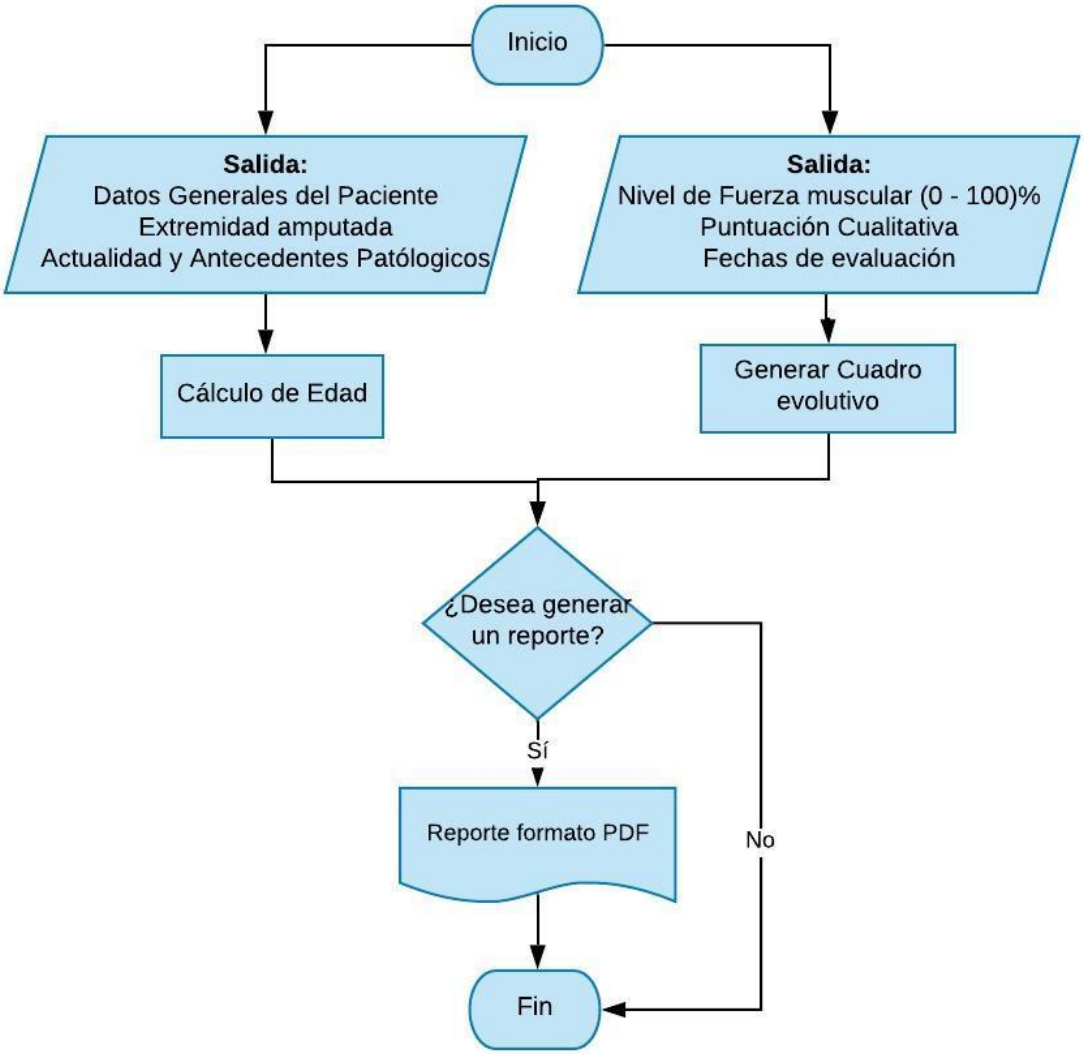


Diagrama de flujo de la Visualización de los resultados (Historia Clínica)

Anexo 4. Manual de Usuario

NOMBRE: Evaluador Multicanal Mioeléctrico para Personas con Amputación Transtibial

VERSION DEL SISTEMA: 1

FECHA DE ELABORACIÓN: 10/07/2019

ÁREA: Fisioterapia

PROPÓSITO

El manual tiene como finalidad ser una guía básica de operación del Evaluador multicanal mioeléctrico para personas con amputación transtibial; permitiendo al lector adquirir las destrezas y conocimientos indispensables en una operación adecuada del sistema. Así, ser una herramienta de consulta de primera mano para el usuario en cualquier momento.

CONOCIMIENTOS BÁSICOS

- Manejo del ordenador, e interfaces.
- Dominio en el área de fisioterapia y electromiografía.

INTRODUCCIÓN

El sistema Evaluador multicanal mioeléctrico para personas con amputación transtibial fue creado con el objetivo de cuantificar el diagnóstico del personal de salud en la evaluación de la condición muscular, al brindar una herramienta tecnológica que otorga precisión y retroalimentación al proceso.

El funcionamiento de este consiste en evaluar el nivel muscular de la extremidad amputada y comparar con el de la extremidad sana. Además, se realiza un registro cronológico de la

información de cada paciente. Así, se presenta un cuadro evolutivo de la condición muscular y se genera un informe en un archivo PDF.

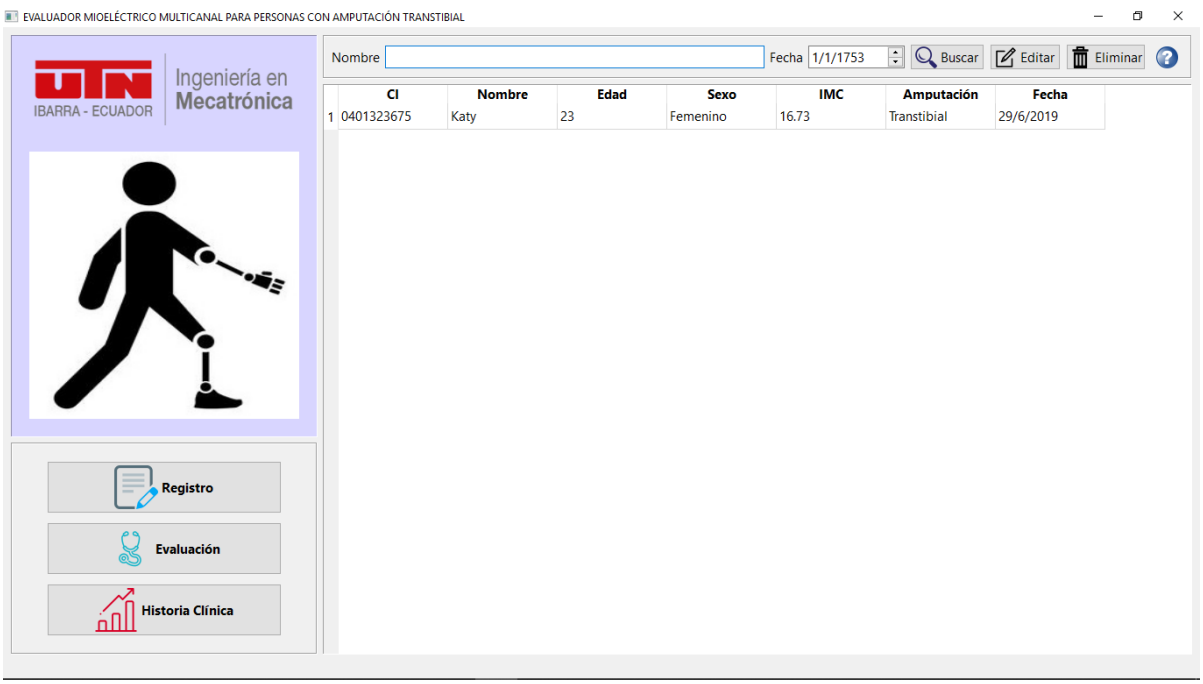
INSTALACIÓN

Para inicializar la instalación se realizará los siguientes pasos:

- Copie el archivo *Evaluador.zip* en su computador en el disco local y extráigalo.
- En la carpeta nueva, realice doble clic en el *archivo Evaluador.exe* para que empiece la ejecución.
- Al finalizar, tendrá acceso al portable del programa.

DESARROLLO DEL MANUAL DE USUARIO

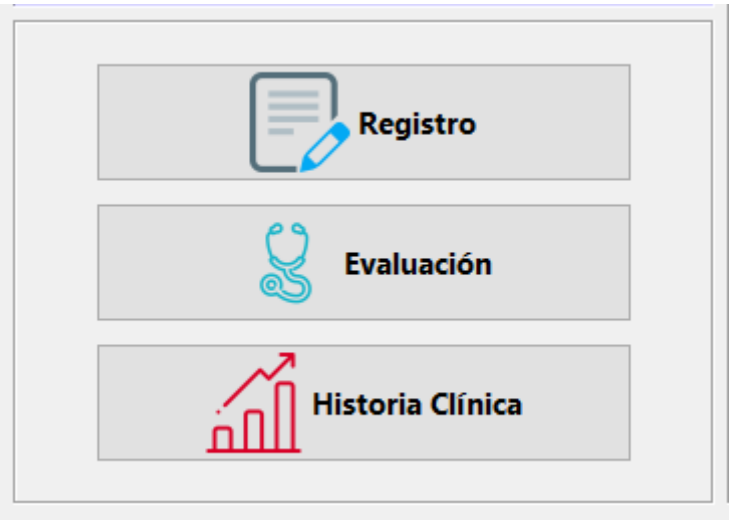
Pantalla principal



Al abrir el programa, la primera pantalla se muestra. Permite la navegación mediante sus botones Registro, Evaluación e Historia clínica. Muestra los pacientes registrados en una tabla, posee un buscador, botones Editar, Eliminar y Ayuda.

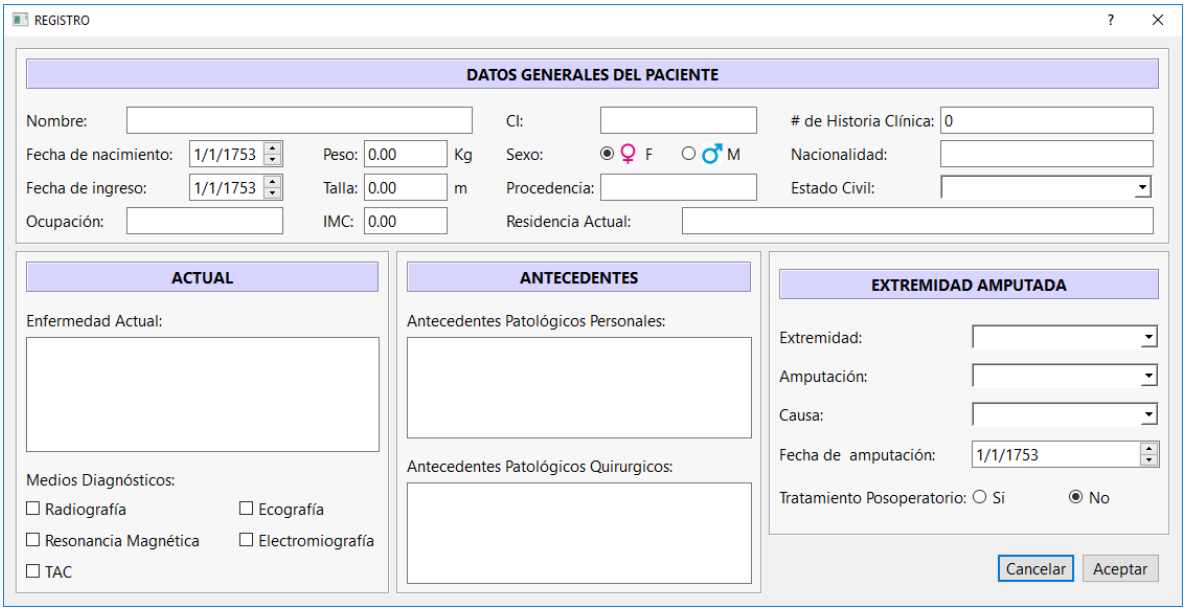
Registro de un paciente

En la pantalla principal, haga clic en el botón Registro y se abrirá la pantalla de registro.



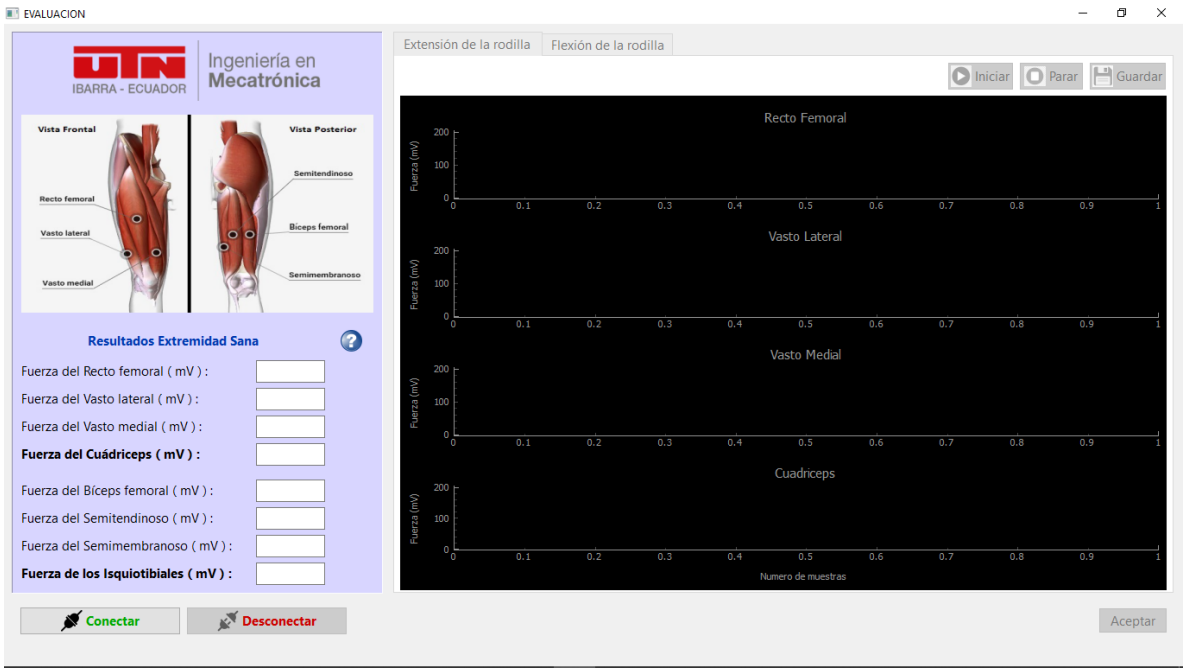
Ingrese la información del paciente: Nombre, numero de cedula (CI), fecha de nacimiento, sexo, peso (Kg), talla (m), nacionalidad, procedencia, lugar de residencia actual, ocupación, estado civil, fecha de ingreso, la información sobre la enfermedad actual, medios diagnósticos, antecedentes patológicos personales y quirúrgicos. Una vez se ingrese el peso y la talla, el índice de masa corporal se calculará automáticamente.

También ingrese la información de la extremidad amputada: Extremidad, tipo de amputación, causa, fecha de la amputación y tratamiento posoperatorio.

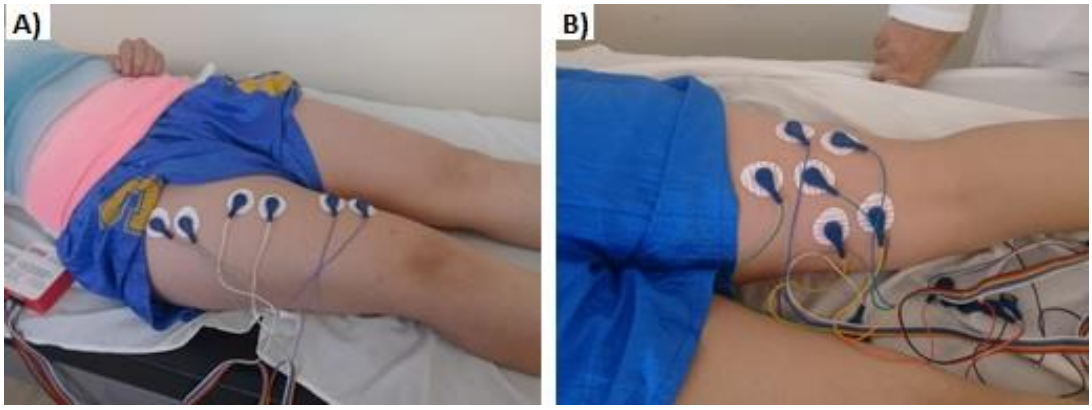


Una vez sea correcta la información, haga clic en aceptar. Se abrirá la pantalla de evaluación de la extremidad sana. Es necesario adquirir estos valores de referencia para concluir con el registro del paciente.

Evaluación de la extremidad sana



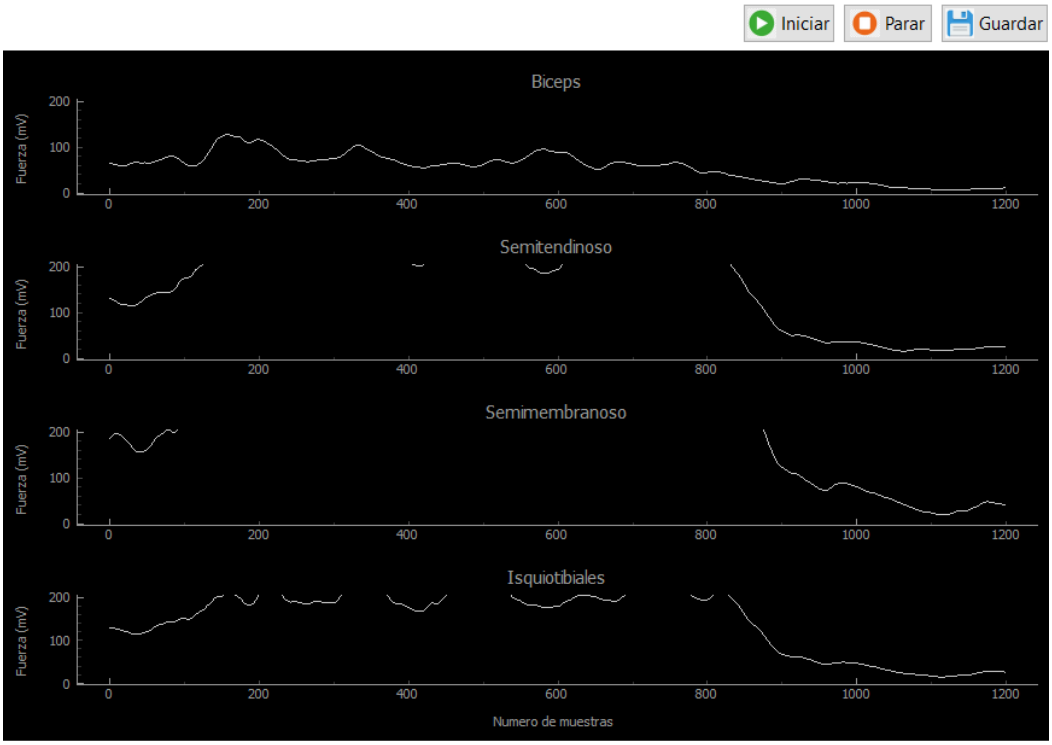
Conecte los electrodos en los puntos motores de los músculos del cuádriceps (A) y del isquiotibial (B) de la extremidad sana. Conecte el cableado al equipo como se indica en la etiqueta de la parte superior del mismo.



Una vez esté conectado, haga clic en "**Conectar"** y espere hasta que aparezca el mensaje de confirmación. Ahora puede realizar la evaluación.

Es necesario que en la extremidad sana se realice la evaluación para extensión y flexión de rodilla. Por tanto, asegúrese de repetir el proceso para el movimiento faltante. A continuación, se muestra la posición de las contracciones a realizar para cada movimiento.

Haga clic en iniciar. Una vez visualice que la gráfica da inicio y se estabilice, realice 3 contracciones con una espera de 3 segundos para cada una. Al finalizar, haga clic en parar.



Si los datos están correctos, haga clic en guardar y aparecerá el valor de la fuerza muscular en mV. Caso contrario, realice nuevamente la adquisición.

Haga clic en Desconectar y espere hasta que aparezca el mensaje de confirmación.

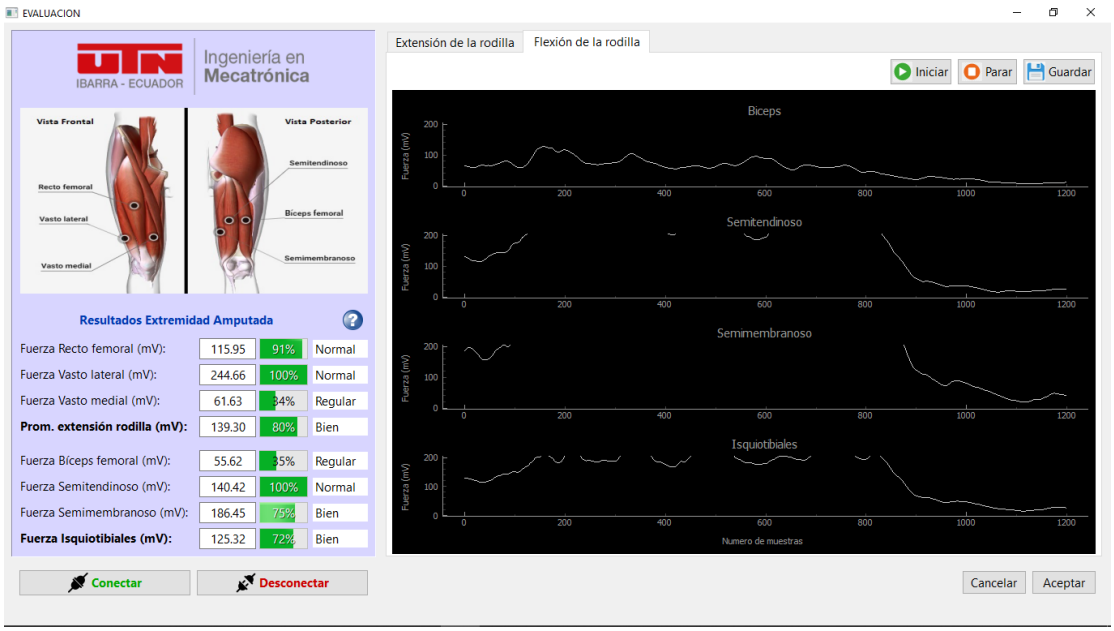
Finalmente, haga clic en aceptar. Para regresar a la pantalla principal.

Evaluación de la extremidad amputada

En la pantalla principal, Seleccione el paciente del que desee realizar la evaluación y haga clic en el botón evaluación.

Siga el mismo proceso para la evaluación del miembro sano. En este caso es posible realizar la evaluación solo para flexión o extensión de rodilla o ambos.

Al dar clic en guardar se mostrará el nivel de fuerza muscular en mV, en un intervalo (0-100) % y la puntuación cualitativa.

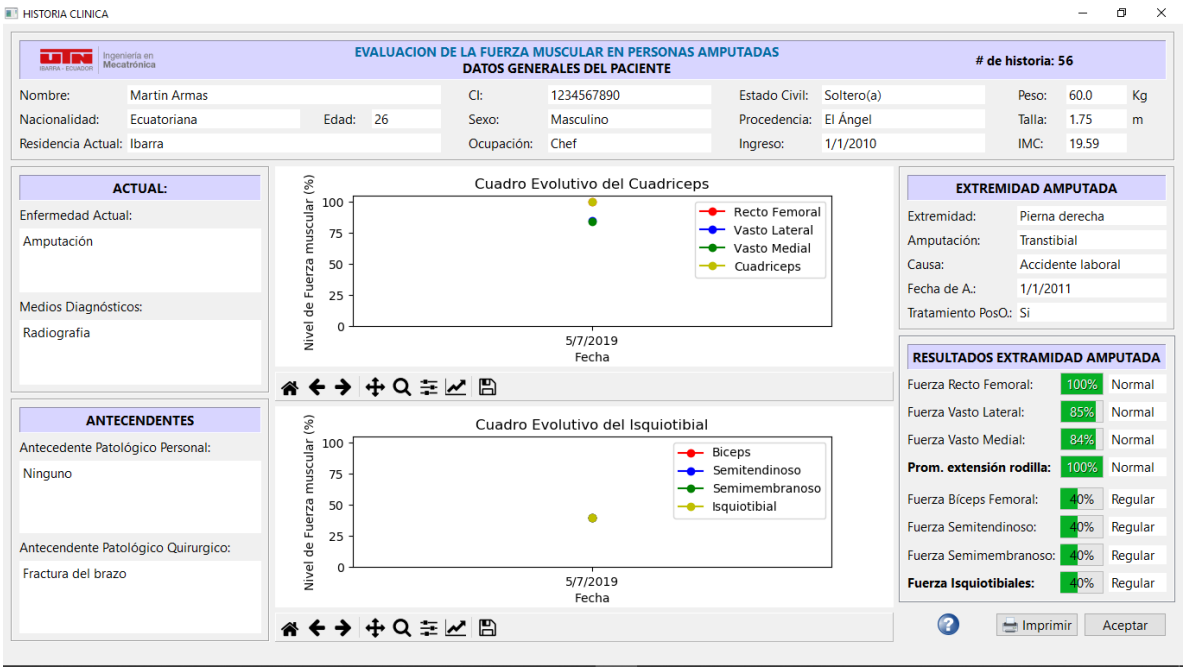


Una vez realizada la evaluación ingrese en la Historia Clínica para visualizar la información.

Historia Clínica

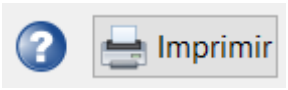
En la pantalla principal, seleccione el paciente del que desee visualizar la información y haga clic en el botón Historia Clínica. Se abrirá la pantalla historia clínica.

En esta ventana se muestra toda la información del paciente, un cuadro evolutivo con la información de las evaluaciones realizadas.



Para visualizar con mayor precisión los niveles de fuerza, haga clic en la gráfica del cuadro evolutivo, en la posición de fecha que desea visualizar. Si desea manipular la gráfica, en la parte inferior se encuentra una barra que le permite hacer zoom y mover la gráfica.

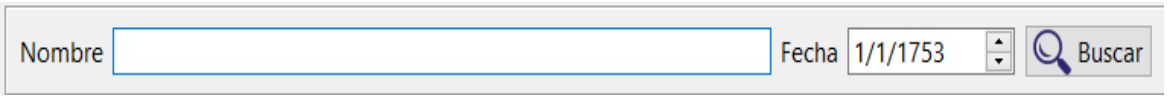
Para generar un reporte en formato PDF, haga clic en el botón imprimir. El reporte se guardará en la carpeta contenedora del programa.



Haga clic en aceptar para regresar a la pantalla principal.

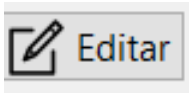
Buscar en la tabla de pacientes

En la pantalla principal, ingrese un nombre o las iniciales y haga clic en buscar. O ingrese la fecha de la última visita y haga clic en buscar.



Editar la información del paciente

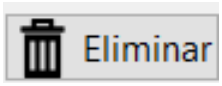
En la pantalla principal, seleccione el paciente del que desea cambiar la información. Se abrirá la pantalla de registro con la información anteriormente ingresada.



Modifique la información que desea y haga clic en aceptar.

Eliminar la información de un paciente

En la pantalla principal, seleccione el paciente del que desea eliminar y haga clic en el botón eliminar. Se mostrará un mensaje de confirmación, presione aceptar.



POSIBLES PROBLEMAS Y SOLUCIONES

Sección	Error	Acción a tomar
Evaluación	La tarjeta no se conecta al programa	Desconecte el USB y apague la tarjeta. Vuelva a conectar el USB y después encienda la tarjeta
	Después de presionar Iniciar, la gráfica se paraliza.	Presione Parar y luego cancelar. La información guardada no se alterará. Vuelva a ingresar en la pantalla de evaluación y realice la evaluación pendiente

	La señal aparece en una gráfica que no corresponde al musculo	Conecte los cables de acuerdo al diagrama que se muestra en la explicación
	Los valores no concuerdan con la realidad	Asegúrese de que los electrodos estén correctamente conectados en los puntos motores
Registro	Intento registrar un paciente, pero me aparece que ya está registrado	Asegúrese en el buscador que el paciente no se encuentre en la tabla
	No puedo editar el número de cedula	El programa permite editar toda la información excepto el número de cedula. Si ese es el caso, elimine al paciente y vuelva a registrarlo

Anexo 5: Entrevista a los fisioterapeutas

Fecha: 2/08/2019

Nombre del entrevistado: Lic. Jairo Collaguazo

Empresa: Prótesis Imbabura

OBJETIVO:

Conocer la opinión del usuario (fisioterapeuta) acerca del sistema desarrollado en el trabajo de grado "Evaluador Mioeléctrico Multicanal para Personas con Amputación Transtibial" para posibles mejoras a futuro.

PREGUNTAS:

1) ¿Cómo calificaría la información que brinda el programa?

a) Muy Buena
b) Buena
c) Regular
d) Mala

2) ¿Como considera usted la forma en que se presenta la información en el programa?

a) Muy Buena
b) Buena
c) Regular
d) Mala

3) ¿Qué mejoras sugeriría?

Se recomienda aumentar información sobre la evaluación física del muñón, fisioterapéutica y protésica. El programa debería mostrar el tipo de ejercicio a realizarse (isométrico o isotónico). Es necesaria la implementación de un sistema de carga para precisión. Basándose en el RM de una persona.

4) ¿Considera que el evaluador mioeléctrico podría ser de gran utilidad en el campo de la medicina o investigativo?

Si es de gran utilidad.

Fecha: 2/08/2019

Nombre del entrevistado: JORGE L. ZAMBRANO FT.

Empresa: UNIVERSIDAD TÉCNICA DEL NORTE

OBJETIVO:

Conocer la opinión del usuario (fisioterapeuta) acerca del sistema desarrollado en el trabajo de grado "Evaluador Mioeléctrico Multicanal para Personas con Amputación Transtibial" para posibles mejoras a futuro.

PREGUNTAS:

1) ¿Cómo calificaría la información que brinda el programa?

a) Muy Buena
b) Buena
c) Regular
d) Mala
e) Insuficiente

2) ¿Como considera usted la forma en que se presenta la información en el programa?

a) Muy Buena
b) Buena
c) Regular
d) Mala

3) ¿Qué mejoras sugeriría?

Sugiero hacer una mejora en el programa para que la hoja de resultados sea más explícita, es decir sea de fácil entendimiento para el profesional de la salud que podría recibirla como un estudio clínico de referencia. (Yo entiendo los resultados porque participé en el estudio y el proceso de realización de la electromiografía al paciente, pero si quiero referir a este paciente a una interconsulta con otro profesional va a tener dificultades para la interpretación de los resultados). Por ejemplo, en la hoja de resultados debería mencionar que para emitir el porcentaje de fuerza nos basamos en un estudio previo en el miembro sano.

4) ¿Considera que el evaluador mioeléctrico podría ser de gran utilidad en el campo de la medicina o investigativo?

Si, ya que la electromiografía es un estudio clínico objetivo de fundamental importancia, pero por su alto costo es poco accesible.

Fecha: 5/08/2019

Nombre del entrevistado: Daniela Zurita

Empresa: Universidad Técnica del Norte

OBJETIVO:

Conocer la opinión del usuario (fisioterapeuta) acerca del sistema desarrollado en el trabajo de grado "Evaluador Mioeléctrico Multicanal para Personas con Amputación Transtibial" para posibles mejoras a futuro.

PREGUNTAS:

1) ¿Cómo calificaría la información que brinda el programa?

a) Muy Buena
b) Buena
c) Regular
d) Mala
e) Insuficiente

2) ¿Como considera usted la forma en que se presenta la información en el programa?

a) Muy Buena
b) Buena
c) Regular
d) Mala

3) ¿Qué mejoras sugeriría?

Permitir la incorporación de pruebas complementarias en base de archivos. Separar en fases el cuadro evolutivo. Información sobre tratamiento preoperatorio.

4) ¿Considera que el evaluador mioeléctrico podría ser de gran utilidad en el campo de la medicina o investigativo?

Si porque permite hacer un seguimiento del tratamiento y poder saber si la terapia está encaminada a la necesidad del paciente.

Anexo 6: Código del software desarrollado

1) Código principal (Evaluador)

```
import pyseeg.openbci.open_bci_v3 as bci
import pyseeg.modules.filterlib as flt
import pyqtgraph as pg
import sys
import os
import pickle
import sqlite3
import datetime
from PyQt4.QtGui import *
from PyQt4.QtCore import *
from principal import *
from registro import *
from historia import *
from evaluacion import *
from ayudaP import *
from ayudaE import *
from ayudaH import *
import numpy as np
import serial
from matplotlib.backends.backend_qt4agg import FigureCanvasQTAgg as FigureCanvas
from matplotlib.backends.backend_qt4agg import NavigationToolbar2QT as
NavigationToolbar
from matplotlib.figure import Figure
from reportlab.lib.pagesizes import A4
from reportlab.lib import colors
from reportlab.lib.styles import getSampleStyleSheet
from reportlab.platypus import SimpleDocTemplate, Image, Spacer, Paragraph, Table,
TableStyle, PageBreak
import re

# Creacion y conexion con la base de datos
class base_datos():
    def __init__(self):
        self.error = QtGui.QMessageBox()
        self.error.setIcon(QMessageBox.Warning)
        self.error.setWindowTitle("ADVERTENCIA")
        # Busca la direccion de la carpeta contenedora y establece la conexion con
la base de datos
        app_path = os.getcwd()
        db_path = app_path+'/DB_Pacientes.db'
        self.conexion = sqlite3.connect(db_path)
        self.conexion.text_factory = str
        self.cursor = self.conexion.cursor()
        try:
            # Crea la tabla pacientes
            self.cursor.execute("""
                CREATE TABLE PACIENTES(
                CI              TEXT     PRIMARY KEY    NOT NULL,
                NOMBRE          TEXT                    NOT NULL,
                NACIMIENTO      BLOB,
                SEXO            TEXT,
                PESO            REAL,
                TALLA           REAL,
                IMC             REAL,
                HC              INT,
                NACIONALIDAD    TEXT,
                PROCEDENCIA     TEXT,
                RESIDENCIA      TEXT,
                ECIVIL          TEXT,
                OCUPACION       TEXT,
```

```
                FECHINGRESO     BLOB,
                MIEMBRO         TEXT,
                AMPUTACION      TEXT,
                CAUSA           TEXT,
                FECHAMPU        BLOB,
                TERAPIA         TEXT,
                ENFERMEDAD      TEXT,
                MEDIOS          BLOB,
                PATOLOGICOP     TEXT,
                PATOLOGICOQ     TEXT,
                FECHA           BLOB
                )""")
        except Exception as e:
            pass

    def DB_datos(self, ci):
        try:
            # Crea una tabla para cada paciente, en donde se guardan la informacion
de la evaluacion
            self.cursor.execute("""
                CREATE TABLE """+'C'+str(ci)+""" (
                FECHA               BLOB     NOT NULL,
                EXTREMIDAD          TEXT,
                RECTO_CRUDA         REAL,
                VLATERAL_CRUDA      REAL,
                VMEDIO_CRUDA        REAL,
                CUADRICEPS_CRUDA    REAL,
                BICEPS_CRUDA        REAL,
                SEMITEN_CRUDA       REAL,
                SEMIMEM_CRUDA       REAL,
                ISQUIOTIBIAL_CRUDA  REAL,
                RECTO               INT,
                VLATERAL            INT,
                VMEDIO              INT,
                CUADRICEPS          INT,
                BICEPS              INT,
                SEMITEN             INT,
                SEMIMEM             INT,
                ISQUIOTIBIAL        INT,
                SENAL_RECTO         BLOB,
                SENAL_VLATERAL      BLOB,
                SENAL_VMEDIO        BLOB,
                SENAL_CUADRICEPS    BLOB,
                SENAL_BICEPS        BLOB,
                SENAL_SEMITEN       BLOB,
                SENAL_SEMIMEM       BLOB,
                SENAL_ISQUIOTIBIAL  BLOB,
                PUNT_RECTO          TEXT,
                PUNT_VLATERAL       TEXT,
                PUNT_VMEDIO         TEXT,
                PUNT_CUADRICEPS     TEXT,
                PUNT_BICEPS         TEXT,
                PUNT_SEMITEN        TEXT,
                PUNT_SEMIMEM        TEXT,
                PUNT_ISQUIOTIBIAL   TEXT
                )""")
        except Exception as e:
            pass


# Pantalla principal
class V_principal(QtGui.QMainWindow):
    def __init__(self, parent=None):
        QtGui.QWidget.__init__(self, parent)
        self.ui = Ui_principal()
        self.ui.setupUi(self)
        self.ui.tabla.horizontalHeader().setResizeMode(1, QHeaderView.Stretch)
        self.showMaximized()
```

```
        # inicializacion de variables
        self.ci = 0
        self.edit = False
        self.P = V_ayudaP()
        self.error = QtGui.QMessageBox()
        self.error.setIcon(QMessageBox.Warning)
        self.error.setWindowTitle("ADVERTENCIA")
        # Conexion con los botones
        self.ui.btn_registro.clicked.connect(self.abrirRegistro)
        self.ui.btn_eval.clicked.connect(self.abrirEval)
        self.ui.btn_buscar.clicked.connect(self.buscar)
        self.ui.btn_editar.clicked.connect(self.editar)
        self.ui.btn_eliminar.clicked.connect(self.eliminar)
        self.ui.btn_HC.clicked.connect(self.abrirDiagnostico)
        self.ui.btn_ayuda.clicked.connect(self.ayuda)
        self.mostrarTabla()

    def ayuda(self):
        # muestra el mensaje de ayuda
        self.P.show()

    def mostrarTabla(self):
        # muestra la informacion de la base de datos en una tabla
        DB.cursor.execute('SELECT CI,NOMBRE,SEXO,IMC,NACIMIENTO,AMPUTACION,FECHA
FROM PACIENTES')
        datos = DB.cursor.fetchall()
        self.Tabla(datos)

    def Tabla(self, datos):
        # ubica la informacion en la tabla
        cont = 0
        self.ui.tabla.setRowCount(len(datos))
        fecha = datetime.datetime.now()
        for fila in datos:
            nombre = fila[1].decode(' utf-8 ')
            self.ui.tabla.setItem(cont, 0, QTableWidgetItem(fila[0]))
            self.ui.tabla.setItem(cont, 1, QTableWidgetItem(nombre))
            self.ui.tabla.setItem(cont, 3, QTableWidgetItem(fila[2]))
            self.ui.tabla.setItem(cont, 4, QTableWidgetItem(str(fila[3])))
            nac = pickle.loads(fila[4])
            f = pickle.loads(fila[6])
            if nac[1] > fecha.month:
                aux = fecha.year - nac[2] - 1
            else:
                aux = fecha.year - nac[2]
            self.ui.tabla.setItem(cont, 2, QTableWidgetItem(str(aux)))
            self.ui.tabla.setItem(cont, 5, QTableWidgetItem(fila[5]))
            self.ui.tabla.setItem(cont, 6, QTableWidgetItem(str(f[0]) + '/' +
str(f[1]) + '/' + str(f[2])))
            cont = cont + 1

    def abrirRegistro(self):
        # abre la pantalla de registro
        self.edit = False
        reg.__init__()
        self.close()
        reg.show()

    def abrirEval(self):
        # abre la pantalla de evaluacion
        self.seleccionar()
        if self.ci != 0:
            eval.__init__()
            DB.cursor.execute("SELECT FECHA FROM C" + str(self.ci) + " WHERE
EXTREMIDAD = ?", ["Sana"])
            dat = DB.cursor.fetchall()
            # verifica si se realizo la evaluacion en la extremidad sana
            if len(dat) == 0:
```

```
                # abre la pantalla de evaluacion para la extremidad sana
                reg.registro = "Sana"
                eval.ui.resultados.setText("Resultados Extremidad Sana")
                eval.ui.pb_biceps.close()
                eval.ui.pb_cuadriceps.close()
                eval.ui.pb_isquiotibial.close()
                eval.ui.pb_recto.close()
                eval.ui.pb_semimem.close()
                eval.ui.pb_semiten.close()
                eval.ui.pb_vastoL.close()
                eval.ui.pb_vastoM.close()
                eval.ui.btn_cancelar.close()
                eval.ui.RectoFP.close()
                eval.ui.VastoLP.close()
                eval.ui.VastoMP.close()
                eval.ui.CuadricepsP.close()
                eval.ui.BicepsP.close()
                eval.ui.SemimemP.close()
                eval.ui.SemitenP.close()
                eval.ui.IsquiotibialesP.close()
            else:
                # abre la pantalla de evaluacion para la extremidad amputada
                eval.ui.resultados.setText("Resultados Extremidad Amputada")
                reg.registro = "Amputada"
            self.close()
            eval.show()
            eval.showMaximized()

    def abrirDiagnostico(self):
        # abre la pantalla de la historia clinica
        self.seleccionar()
        if self.ci != 0:
            historia.__init__()
            historia.mostrar()
            self.close()
            historia.show()
            historia.showMaximized()

    def seleccionar(self):
        # extrae el numero de cedula de la fila seleccionada
        fila = self.ui.tabla.selectedItems()
        if len(fila) == 1:
            self.ci = self.ui.tabla.item(fila[0].row(), 0).text()
        else:
            self.error.setText('Seleccione un paciente de la lista')
            self.error.exec_()
            self.ci = 0

    def buscar(self):
        # se realiza la busqueda en la base de datos acorde al nombre o fecha
        nombre = str(self.ui.nombre.text())
        dia = self.ui.fecha.date().day()
        mes = self.ui.fecha.date().month()
        a = self.ui.fecha.date().year()
        f = [dia, mes, a]
        fecha = pickle.dumps(f)
        datos = []
        aux = False
        self.ui.tabla.clearContents()
        # busqueda por nombre y fecha
        if nombre != '' and a > 1753:
            DB.cursor.execute("""SELECT CI, NOMBRE, SEXO, IMC, NACIMIENTO,
AMPUTACION, FECHA FROM PACIENTES WHERE NOMBRE LIKE ? AND FECHA LIKE ?""",
["%{}%".format(nombre), fecha])
            datos = DB.cursor.fetchall()
        elif nombre != '':
            # busqueda por nombre
            DB.cursor.execute("""SELECT CI, NOMBRE, SEXO, IMC, NACIMIENTO,
```

```
AMPUTACION, FECHA FROM PACIENTES WHERE NOMBRE LIKE ?""", ["%{}%".format(nombre)])
            datos = DB.cursor.fetchall()
        elif a > 1753:
            # busqueda por fecha
            DB.cursor.execute("""SELECT CI, NOMBRE, SEXO, IMC, NACIMIENTO,
AMPUTACION, FECHA FROM PACIENTES WHERE FECHA LIKE ?""", [fecha])
            datos = DB.cursor.fetchall()
        else:
            aux = True
            self.mostrarTabla()
            self.error.setText('Ingrese un nombre o fecha')
            self.error.exec_()
        # comprueba si se encontro algun resultado
        if len(datos) == 0 and aux is False:
            self.error.setText('Busqueda no encontrada')
            self.error.exec_()
        elif aux is False:
            self.Tabla(datos)
        self.ui.fecha.setDate(QtCore.QDate(1753, 1, 1))

    def editar(self):
        # abre la pantalla de registro con la informacion del paciente seleccionado
        self.seleccionar()
        if self.ci != 0:
            self.edit = True
            self.close()
            reg.show()
            reg.editar()

    def eliminar(self):
        # elimina de la base de datos al paciente seleccionado
        self.seleccionar()
        if self.ci != 0:
            msg = QtGui.QMessageBox()
            msg.setIcon(QMessageBox.Warning)
            msg.setText("Esta seguro de eliminar")
            msg.setWindowTitle("ADVERTENCIA")
            msg.setStandardButtons(QMessageBox.Ok | QMessageBox.Cancel)
            msg.button(QMessageBox.Ok).clicked.connect(self.acept_elim)
            msg.exec_()

    def acept_elim(self):
        # en caso de aceptar elimina al paciente de la base de datos
        DB.cursor.execute("""DELETE FROM PACIENTES WHERE CI LIKE ?""",
[str(self.ci)])
        DB.cursor.execute('DROP TABLE '+'C'+str(self.ci))
        DB.conexion.commit()
        self.ui.tabla.clearContents()
        self.mostrarTabla()



class hilo2(QThread):
    # creacion de un segundo hilo de programacion
    def __init__(self):
        QThread.__init__(self)
        self.cont = 0
        self.emg = [[], [], []]

    def __del__(self):
        self.wait()

    def run(self):
        # comienza la adquisicion de datos EMG
        self.cont = 0
        self.emg = [[], [], []]
        self.frt1 = flt.FltRealTime()
        self.frt2 = flt.FltRealTime()
```

```
        self.frt3 = flt.FltRealTime()
        eval.board.start_streaming(plotdata)


def plotdata(sample):
    # adquisicion y procesamiento de los valores EMG
    aux = 20
    suma = [0]*aux
    smp = sample.channel_data
    # seleccion de los canales a adquirir acorde al movimiento de extension o
flexion
    # filtrado IIR
    if eval.aux == 1:
        filtro1 = hilo.frt1.filterIIR(smp[0], 0)
        filtro2 = hilo.frt2.filterIIR(smp[1], 0)
        filtro3 = hilo.frt3.filterIIR(smp[2], 0)
    else:
        filtro1 = hilo.frt1.filterIIR(smp[3], 0)
        filtro2 = hilo.frt2.filterIIR(smp[4], 0)
        filtro3 = hilo.frt3.filterIIR(smp[5], 0)
    hilo.emg[0].append(filtro1)
    hilo.emg[1].append(filtro2)
    hilo.emg[2].append(filtro3)
    hilo.cont = hilo.cont + 1
    # segmentacion y procesamiento
    if hilo.cont == aux:
        for x in range(0, 3):
            # filtrado pasa_banda y pasa_bajos de cada canal
            filtro_pbanda = flt.butter_bandpass_filter(hilo.emg[x], lowcut=5,
highcut=110, fs=250)
            rect = abs(filtro_pbanda)
            filtro_pbajo = flt.butter_lowpass_filter(rect, lowcut=2, fs=250)
            smax = len(filtro_pbajo)
            smin = smax - aux
            emg1 = filtro_pbajo[smin:smax]
            for y in range(0, aux):
                eval.valor[x].pop(0)
                eval.valor[x].append(emg1[y])
                eval.emg[x].append(emg1[y])
                suma[y] = suma[y] + emg1[y]
                if x == 2:
                    # obtencion del valor promedio de los 3 canales
                    suma[y] = suma[y]/3
                    eval.valor[3].pop(0)
                    eval.valor[3].append(suma[y])
                    eval.emg[3].append(suma[y])
        # graficas de cada canal y el valor promedio
        eval.curve1.setData(eval.valor[0])
        eval.curve2.setData(eval.valor[1])
        eval.curve3.setData(eval.valor[2])
        eval.curve4.setData(eval.valor[3])
        hilo.cont = 0


# Pantalla de evaluacion
class V_evaluacion(QtGui.QMainWindow):
    def __init__(self, parent=None):
        QtGui.QWidget.__init__(self, parent)
        self.ui = Ui_Evaluacion()
        self.ui.setupUi(self)
        # inicializacion de las valiables
        self.E = V_ayudaE()
        self.valor = [[0] * 1200, [0] * 1200, [0] * 1200, [0] * 1200]
        self.emg = [[], [], [], []]
        self.aux = 0
        self.RMS = [0.]*8
        self.emgC = [0.]*4
```

```
        self.emgI = [0.]*4
        self.guardar1 = False
        self.guardar2 = False
        fecha = datetime.datetime.now()
        f = [fecha.day, fecha.month, fecha.year]
        self.fec = pickle.dumps(f)
        self.repeticion = False
        self.error = QtGui.QMessageBox()
        self.error.setIcon(QMessageBox.Warning)
        self.error.setWindowTitle("ADVERTENCIA")
        self.msg = QtGui.QMessageBox()
        self.msg.setIcon(QMessageBox.Information)
        self.msg.setWindowTitle("INFORMACION")
        # creacion de las graficas
        self.p = pg.GraphicsWindow()
        self.p1 = pg.GraphicsWindow()
        self.plot1 = self.p.addPlot(0, 0, title='Recto Femoral')
        self.plot2 = self.p.addPlot(1, 0, title='Vasto Lateral')
        self.plot3 = self.p.addPlot(2, 0, title='Vasto Medial')
        self.plot4 = self.p.addPlot(3, 0, title='Promedio Extension Rodilla')
        self.plot5 = self.p1.addPlot(0, 0, title='Biceps')
        self.plot6 = self.p1.addPlot(1, 0, title='Semitendinoso')
        self.plot7 = self.p1.addPlot(2, 0, title='Semimembranoso')
        self.plot8 = self.p1.addPlot(3, 0, title='Isquiotibiales')
        self.plot1.setYRange(0, 350)
        self.plot2.setYRange(0, 350)
        self.plot3.setYRange(0, 350)
        self.plot4.setYRange(0, 350)
        self.plot5.setYRange(0, 350)
        self.plot6.setYRange(0, 350)
        self.plot7.setYRange(0, 350)
        self.plot8.setYRange(0, 350)
        self.plot1.setLabel('left', "Fuerza (mV)")
        self.plot2.setLabel('left', "Fuerza (mV)")
        self.plot3.setLabel('left', "Fuerza (mV)")
        self.plot4.setLabel('left', "Fuerza (mV)")
        self.plot4.setLabel('bottom', "Numero de muestras")
        self.plot5.setLabel('left', "Fuerza (mV)")
        self.plot6.setLabel('left', "Fuerza (mV)")
        self.plot7.setLabel('left', "Fuerza (mV)")
        self.plot8.setLabel('left', "Fuerza (mV)")
        self.plot8.setLabel('bottom', "Numero de muestras")
        self.ui.grafica_E.addWidget(self.p)
        self.ui.grafica_F.addWidget(self.p1)
        # deshabilitacion de widgets
        self.ui.btn_guardar_E.setEnabled(False)
        self.ui.btn_guardar_F.setEnabled(False)
        self.ui.btn_aceptar.setEnabled(False)
        self.ui.btn_Desconectar.setEnabled(False)
        self.ui.tab_Eval_Inicial.setEnabled(False)
        # conexion con las acciones de cada boton
        self.ui.btn_aceptar.clicked.connect(self.aceptar)
        self.ui.btn_Conectar.clicked.connect(self.conectar)
        self.ui.btn_Desconectar.clicked.connect(self.desconectar)
        self.ui.btn_cancelar.clicked.connect(self.cancelar)
        self.ui.btn_Inicio_E.clicked.connect(self.inicioE)
        self.ui.btn_inicio_F.clicked.connect(self.inicioF)
        self.ui.btn_guardar_E.clicked.connect(self.guardarE)
        self.ui.btn_guardar_F.clicked.connect(self.guardarF)
        self.ui.btn_parar_E.clicked.connect(self.pararE)
        self.ui.btn_parar_F.clicked.connect(self.pararF)
        self.ui.btn_ayuda.clicked.connect(self.ayuda)

    def ayuda(self):
        # muestra el mensaje de ayuda
        self.E.show()

    def conectar(self):
```

```
        # conexion de la comunicacion serial del programa con la tarjeta Cyton
        QApplication.setOverrideCursor(Qt.WaitCursor)
        encontrado = False
        puerto = 0
        for puerto in range(0, 127):
            # deteccion del puerto USB
            try:
                p = 'COM' + str(puerto)
                velocidad = '115200'
                conexion = serial.Serial(p, velocidad)
                conexion.close()
                encontrado = True
                break
            except:
                pass
        # confirmacion de que el puerto fue encontrado y conexion
        QApplication.restoreOverrideCursor()
        if encontrado:
            port = 'COM' + str(puerto)
            baud = 115200
            self.board = bci.OpenBCIBoard(port=port, baud=baud)
            self.ui.btn_Desconectar.setEnabled(True)
            self.ui.tab_Eval_Inicial.setEnabled(True)
            self.msg.setText("Conexion correctamente realizada")
            self.msg.exec_()
        else:
            self.error.setText("Error: Puerto no encontrado")
            self.error.exec_()

    def desconectar(self):
        # desconexion de la comunicacion serial con la tarjeta Cyton
        QApplication.setOverrideCursor(Qt.WaitCursor)
        self.ui.btn_aceptar.setEnabled(True)
        self.board.disconnect()
        QApplication.restoreOverrideCursor()
        self.msg.setText("Desconexion correctamente realizada")
        self.msg.exec_()

    def aceptar(self):
        # retorno a la pantalla principal
        if reg.registro == "Sana":
            # si es la evaluacion de la extremidad sana confirma que se guardara
extension y flexion
            if self.guardar1 is True and self.guardar2 is True:
                self.cancelar()
                DB.cursor.execute("""UPDATE PACIENTES SET FECHA = ? WHERE CI LIKE
?""", [self.fec, str(myapp.ci)])
                DB.conexion.commit()
            else:
                self.error.setText("Evaluacion incompleta: Realice la evaluacion
para flexion y extension de rodilla. Presione Guardar")
                self.error.exec_()
        else:
            # confirma que se guardara extension o flexion
            if self.guardar1 is True or self.guardar2 is True:
                self.cancelar()
                DB.cursor.execute("""UPDATE PACIENTES SET FECHA = ? WHERE CI LIKE
?""", [self.fec, str(myapp.ci)])
                DB.conexion.commit()
            else:
                self.error.setText("Evaluacion incompleta: Presione Guardar")
                self.error.exec_()

    def puntuacion(self, porcentaje):
        # asignacion de la puntuacion cualitativa
        if porcentaje <= 1:
            punt = "Nula"
        elif porcentaje <= 7:
```

```
            punt = "Escasa"
        elif porcentaje <= 25:
            punt = "Mal"
        elif porcentaje <= 65:
            punt = "Regular"
        elif porcentaje <= 90:
            punt = "Bien"
        else:
            punt = "Normal"
        return punt

    def inicioE(self):
        # inicio de la adquisicion para extension
        self.valor = [[0] * 1200, [0] * 1200, [0] * 1200, [0] * 1200]
        self.emg = [[], [], [], []]
        self.aux = 1
        self.ui.tab_Eval_Inicial.setTabEnabled(1, False)
        self.plot1.clear()
        self.plot2.clear()
        self.plot3.clear()
        self.plot4.clear()
        self.curve1 = self.plot1.plot()
        self.curve2 = self.plot2.plot()
        self.curve3 = self.plot3.plot()
        self.curve4 = self.plot4.plot()
        hilo.start()

    def pararE(self):
        # parar la adquisicion y calcular el valor RMS
        self.board.stop()
        self.ui.tab_Eval_Inicial.setTabEnabled(1, True)
        self.ui.btn_guardar_E.setEnabled(True)
        for i in range(0, 4):
            n = len(self.emg[i])
            e = self.emg[i][400:n]
            rms = np.sqrt(np.sum(x ** 2 for x in e) / n)
            self.RMS.insert(i, round(rms, 2))
            self.emgC[i] = pickle.dumps(e)

    def guardarE(self):
        # guardar la informacion en la base de datos del paciente
        DB.cursor.execute("SELECT FECHA FROM C" + str(myapp.ci) + " WHERE
EXTREMIDAD = ? AND FECHA = ?", [reg.registro,self.fec])
        dat = DB.cursor.fetchall()
        # comprueba si existe repeticion de fechas
        if len(dat) == 0:
            self.repeticion = False
        else:
            self.repeticion = True
        DB.cursor.execute("SELECT * FROM C"+ str(myapp.ci))
        datos = DB.cursor.fetchall()
        valores = self.RMS[0:4]
        valores.append(self.emgC[0])
        valores.append(self.emgC[1])
        valores.append(self.emgC[2])
        valores.append(self.emgC[3])
        if reg.registro is "Amputada":
            # comprueba que la extremidad es amputada para realizar la comparacion
            fila = datos[0]
            for i in range(2, 6):
                # calculo del porcentaje
                aux = self.RMS[i-2]*100/float(fila[i])
                if aux > 100:
                    aux = 100
                punt = self.puntuacion(int(aux))
                valores.append(int(aux))
                valores.append(punt)
            valores.append(reg.registro)
```

```
            valores.append(self.fec)
            if self.repeticion is True:
                # se actualiza la informacion si ya existe la fecha en la base de
datos
                DB.cursor.execute("UPDATE C" + str(myapp.ci) + """ SET RECTO_CRUDA
= ?, VLATERAL_CRUDA = ?, VMEDIO_CRUDA =?, CUADRICEPS_CRUDA = ?, SENAL_RECTO = ?,
SENAL_VLATERAL = ?, SENAL_VMEDIO = ?, SENAL_CUADRICEPS = ?, RECTO = ?, PUNT_RECTO =
?, VLATERAL = ?, PUNT_VLATERAL = ?, VMEDIO = ?, PUNT_VMEDIO = ?, CUADRICEPS = ?,
PUNT_CUADRICEPS = ? WHERE EXTREMIDAD = ? AND FECHA = ?""", valores)
            else:
                # se inserta nueva informacion en la base de datos
                DB.cursor.execute("INSERT INTO C" + str(myapp.ci) + """
(RECTO_CRUDA, VLATERAL_CRUDA, VMEDIO_CRUDA, CUADRICEPS_CRUDA, SENAL_RECTO,
SENAL_VLATERAL, SENAL_VMEDIO, SENAL_CUADRICEPS, RECTO, PUNT_RECTO, VLATERAL,
PUNT_VLATERAL, VMEDIO, PUNT_VMEDIO, CUADRICEPS, PUNT_CUADRICEPS, EXTREMIDAD, FECHA)
VALUES (?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?)""", valores)
            self.ui.pb_recto.setValue(valores[8])
            self.ui.pb_vastoL.setValue(valores[10])
            self.ui.pb_vastoM.setValue(valores[12])
            self.ui.pb_cuadriceps.setValue(valores[14])
            self.ui.RectoFP.setText(valores[9])
            self.ui.VastoLP.setText(valores[11])
            self.ui.VastoMP.setText(valores[13])
            self.ui.CuadricepsP.setText(valores[15])
        else:
            # evaluacion de la extremidad sana
            valores.append(reg.registro)
            valores.append(self.fec)
            if self.repeticion is True:
                # se actualiza la informacion si ya existe la fecha en la base de
datos
                DB.cursor.execute("UPDATE C" + str(myapp.ci) + """ SET RECTO_CRUDA
= ?, VLATERAL_CRUDA = ?, VMEDIO_CRUDA = ?, CUADRICEPS_CRUDA = ?, SENAL_RECTO = ?,
SENAL_VLATERAL = ?, SENAL_VMEDIO = ?, SENAL_CUADRICEPS = ?  WHERE EXTREMIDAD = ?
AND FECHA = ?""", valores)
            else:
                # se inserta nueva informacion en la base de datos
                DB.cursor.execute("INSERT INTO C"+ str(myapp.ci)+""" (RECTO_CRUDA,
VLATERAL_CRUDA, VMEDIO_CRUDA, CUADRICEPS_CRUDA, SENAL_RECTO, SENAL_VLATERAL,
SENAL_VMEDIO, SENAL_CUADRICEPS, EXTREMIDAD, FECHA) VALUES (?, ?, ?, ?, ?, ?, ?, ?,
?, ?)""", valores)
        DB.conexion.commit()
        self.guardar1 = True
        self.ui.RectoF.setValue(self.RMS[0])
        self.ui.VastoL.setValue(self.RMS[1])
        self.ui.VastoM.setValue(self.RMS[2])
        self.ui.Cuadriceps.setValue(self.RMS[3])

    def inicioF(self):
        # inicio de la adquisicion para flexion
        self.valor = [[0] * 1200, [0] * 1200, [0] * 1200, [0] * 1200]
        self.emg = [[], [], [], []]
        self.aux = 2
        self.ui.tab_Eval_Inicial.setTabEnabled(0, False)
        self.plot5.clear()
        self.plot6.clear()
        self.plot7.clear()
        self.plot8.clear()
        self.curve1 = self.plot5.plot()
        self.curve2 = self.plot6.plot()
        self.curve3 = self.plot7.plot()
        self.curve4 = self.plot8.plot()
        hilo.start()

    def pararF(self):
        # parar la adquisicion y calcular el valor RMS
        self.board.stop()
        self.ui.tab_Eval_Inicial.setTabEnabled(0, True)
```

```
        self.ui.btn_guardar_F.setEnabled(True)
        for i in range(0, 4):
            n = len(self.emg[i])
            e = self.emg[i][400:n]
            rms = np.sqrt(np.sum(x ** 2 for x in e) / n)
            self.RMS.insert(i+4, round(rms, 2))
            self.emgI[i] = pickle.dumps(e)

    def guardarF(self):
        # guardar la informacion en la base de datos del paciente
        DB.cursor.execute("SELECT FECHA FROM C" + str(myapp.ci) + " WHERE
EXTREMIDAD = ? AND FECHA = ?", [reg.registro, self.fec])
        dat = DB.cursor.fetchall()
        # comprueba si existe repeticion de fecha
        if len(dat) == 0:
            self.repeticion = False
        else:
            self.repeticion = True
        DB.cursor.execute("SELECT * FROM C" + str(myapp.ci))
        datos = DB.cursor.fetchall()
        valores = self.RMS[4:8]
        valores.append(self.emgI[0])
        valores.append(self.emgI[1])
        valores.append(self.emgI[2])
        valores.append(self.emgI[3])
        if reg.registro is "Amputada":
            # comprueba que la extremidad es amputada para realizar la comparacion
            fila = datos[0]
            for i in range(6, 10):
                # calculo del porcentaje
                aux = self.RMS[i - 2] * 100 / fila[i]
                if aux > 100:
                    aux = 100
                punt = self.puntuacion(int(aux))
                valores.append(int(aux))
                valores.append(punt)
            valores.append(reg.registro)
            valores.append(self.fec)
            if self.repeticion is True:
                # se actualiza la informacion si ya existe la fecha en la base de
datos
                DB.cursor.execute("UPDATE C" + str(myapp.ci) + """ SET BICEPS_CRUDA
= ?, SEMITEN_CRUDA = ?, SEMIMEM_CRUDA = ?, ISQUIOTIBIAL_CRUDA = ?, SENAL_BICEPS =
?, SENAL_SEMITEN = ?, SENAL_SEMIMEM = ?, SENAL_ISQUIOTIBIAL = ?, BICEPS = ?,
PUNT_BICEPS = ?, SEMITEN = ?, PUNT_SEMITEN = ?, SEMIMEM = ?, PUNT_SEMIMEM = ?,
ISQUIOTIBIAL = ?, PUNT_ISQUIOTIBIAL = ? WHERE EXTREMIDAD = ? AND FECHA = ?""",
valores)
            else:
                # se inserta nueva informacion en la base de datos
                DB.cursor.execute("INSERT INTO C" + str(myapp.ci) +
"""(BICEPS_CRUDA, SEMITEN_CRUDA, SEMIMEM_CRUDA, ISQUIOTIBIAL_CRUDA, SENAL_BICEPS,
SENAL_SEMITEN, SENAL_SEMIMEM, SENAL_ISQUIOTIBIAL, BICEPS, PUNT_BICEPS, SEMITEN,
PUNT_SEMITEN, SEMIMEM, PUNT_SEMIMEM, ISQUIOTIBIAL, PUNT_ISQUIOTIBIAL, EXTREMIDAD,
FECHA) VALUES (?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?)""", valores)
                self.ui.pb_biceps.setValue(valores[8])
                self.ui.pb_semiten.setValue(valores[10])
                self.ui.pb_semimem.setValue(valores[12])
                self.ui.pb_isquiotibial.setValue(valores[14])
                self.ui.BicepsP.setText(valores[9])
                self.ui.SemitenP.setText(valores[11])
                self.ui.SemimemP.setText(valores[13])
                self.ui.IsquiotibialesP.setText(valores[15])
        else:
            # evaluacion de la extremidad sana
            valores.append(reg.registro)
            valores.append(self.fec)
            if self.repeticion is True:
                # se actualiza la informacion si ya existe la fecha en la base de
```

```
datos
                DB.cursor.execute("UPDATE C" + str(myapp.ci) + """ SET BICEPS_CRUDA
= ?, SEMITEN_CRUDA = ?, SEMIMEM_CRUDA = ?, ISQUIOTIBIAL_CRUDA = ?, SENAL_BICEPS =
?, SENAL_SEMITEN = ?, SENAL_SEMIMEM = ?, SENAL_ISQUIOTIBIAL = ? WHERE EXTREMIDAD =
? AND FECHA = ?""", valores)
            else:
                # se inserta nueva informacion en la base de datos
                DB.cursor.execute("INSERT INTO C" + str(myapp.ci) +
"""(BICEPS_CRUDA, SEMITEN_CRUDA, SEMIMEM_CRUDA, ISQUIOTIBIAL_CRUDA, SENAL_BICEPS,
SENAL_SEMITEN, SENAL_SEMIMEM, SENAL_ISQUIOTIBIAL, EXTREMIDAD, FECHA) VALUES (?, ?,
?, ?, ?, ?, ?, ?, ?, ?)""", valores)
        DB.conexion.commit()
        self.guardar2 = True
        self.ui.Biceps.setValue(self.RMS[4])
        self.ui.Semiten.setValue(self.RMS[5])
        self.ui.Semimem.setValue(self.RMS[6])
        self.ui.Isquiotibial.setValue(self.RMS[7])

    def cancelar(self):
        # permite salir de la pantalla sin concluir la evaluacion
        self.close()
        myapp.__init__()
        myapp.show()
        myapp.mostrarTabla()


# Pantalla de historia clinica
class V_historia(QtGui.QMainWindow):
    def __init__(self, parent=None):
        QtGui.QWidget.__init__(self, parent)
        self.ui = Ui_Diagnostico()
        self.ui.setupUi(self)
        self.error = QtGui.QMessageBox()
        self.error.setIcon(QMessageBox.Warning)
        self.error.setWindowTitle("ADVERTENCIA")
        # inicializacion de variables
        self.H = V_ayudaH()
        self.fecha = datetime.datetime.now()
        f = [self.fecha.day, self.fecha.month, self.fecha.year]
        self.fec = pickle.dumps(f)
        self.cuadri = [[], [], [], [], [], [], [], [], []]
        self.isquio = [[], [], [], [], [], [], [], [], []]
        self.edad = 0
        self.Medios = ""
        # creacion de graficas
        self.figure = Figure()
        self.canvas1 = FigureCanvas(self.figure)
        self.figure1 = Figure()
        self.canvas2 = FigureCanvas(self.figure1)
        toolbar1 = NavigationToolbar(self.canvas1, self)
        toolbar2 = NavigationToolbar(self.canvas2, self)
        self.figure.subplots_adjust(top=0.85, bottom=0.2)
        self.figure1.subplots_adjust(top=0.85, bottom=0.2)
        self.ui.Grafica.addWidget(self.canvas1)
        self.ui.Grafica.addWidget(toolbar1)
        self.ui.Grafica.addWidget(self.canvas2)
        self.ui.Grafica.addWidget(toolbar2)
        # conectar las acciones con los botones
        self.ui.btn_aceptar.clicked.connect(self.aceptar)
        self.ui.btn_imprimir.clicked.connect(self.imprimir)
        self.ui.ayuda.clicked.connect(self.ayuda)

    def ayuda(self):
        # muestra el mensaje de ayuda
        self.H.show()

    def imprimir(self):
        # permite generar un documento PDF con la informacion del paciente
```

```
        c = SimpleDocTemplate("C"+str(myapp.ci)+".pdf", pagesize=A4)
        elements = []
        # encabezado
        utn = Image('UTN.png', width=50, height=50)
        cime = Image('LOGO.png', width=100, height=30)
        data0 = [[utn, 'EVALUADOR DE LA CONDICION MUSCULAR \nEN PERSONAS
AMPUTADAS', cime]]
        encabezado = Table(data0, colWidths=[100, 300, 100])
        encabezado.setStyle(TableStyle([('ALIGN', (0, 0), (2, 0), 'CENTER'),
                                        ('FONTSIZE', (0, 0), (2, 0), 10)]))
        elements.append(encabezado)
        elements.append(Spacer(0, 30))
        # datos generales del paciente
        estilo = getSampleStyleSheet()
        p1 = Paragraph('DATOS GENERALES DEL PACIENTE ', estilo["BodyText"])
        elements.append(p1)
        elements.append(Spacer(0, 10))
        data = [['CI: ', str(myapp.ci), '', 'Nombre: ',
str(self.datos[0][0]).decode(' utf-8 '), '', '', '', ''],
                ['Sexo: ', str(self.datos[0][2]), '', 'Peso (Kg): ',
str(self.datos[0][3]), 'Talla (m): ', str(self.datos[0][4]), 'IMC: ',
str(self.datos[0][5])],
                ['Residencia: ', '', str(self.datos[0][10]).decode(' utf-8 '), '',
'', '', '', 'Edad:', str(self.edad)]
                ]
        t = Table(data, colWidths=[40, 40, 50, 60, 50, 50, 50, 45, 55])
        t.setStyle(TableStyle([('BACKGROUND', (0, 0), (0, 2), colors.lightskyblue),
                               ('BACKGROUND', (3, 0), (3, 1), colors.lightskyblue),
                               ('BACKGROUND', (0, 2), (1, 2), colors.lightskyblue),
                               ('BACKGROUND', (7, 1), (7, 2), colors.lightskyblue),
                               ('BACKGROUND', (5, 1), (5, 1), colors.lightskyblue),
                               ('GRID', (0, 0), (8, 2), 0.5, colors.black),
                               ('FONTSIZE', (0, 0), (8, 2), 10),
                               ('SPAN', (4, 0), (8, 0)),
                               ('SPAN', (1, 0), (2, 0)),
                               ('SPAN', (1, 1), (2, 1)),
                               ('SPAN', (0, 2), (1, 2)),
                               ('SPAN', (2, 2), (6, 2))]))
        f_i = pickle.loads(self.datos[0][12])
        ingreso = str(f_i[0]) + "/" + str(f_i[1]) + "/" + str(f_i[2])
        data1 = [
            ['Nacionalidad:', str(self.datos[0][7]).decode(' utf-8 '),
'Procendecia:', str(self.datos[0][8]).decode(' utf-8 '), 'Ingreso:', ingreso],
            ['Estado civil:', str(self.datos[0][11]).decode(' utf-8 '),
'Ocupacion:', str(self.datos[0][9]).decode(' utf-8 '),'','']
        ]
        t1 = Table(data1, colWidths=[80, 90, 80, 90, 45, 55])
        t1.setStyle(TableStyle([('BACKGROUND', (0, 0), (0, 1),
colors.lightskyblue),
                                ('BACKGROUND', (2, 0), (2, 1),
colors.lightskyblue),
                                ('BACKGROUND', (4, 0), (4, 0),
colors.lightskyblue),
                               ('GRID', (0, 0), (5, 0), 0.5, colors.black),
                               ('GRID', (0, 1), (3, 1), 0.5, colors.black),
                               ('FONTSIZE', (0, 0), (5, 1), 10)]))
        elements.append(t)
        elements.append(t1)
        # informacion medica actual
        elements.append(Spacer(0, 20))
        p2 = Paragraph('ACTUAL: ', estilo["BodyText"])
        elements.append(p2)
        elements.append(Spacer(0, 10))
        e = str(self.datos[0][18]).decode(' utf-8 ')
        enf = self.salto(e, 80)
        data2 = [
            ['Enfermedad actual:', enf],
            ['Medios diagnosticos:', self.Medios[0:len(self.Medios)-2]]
```

```
        ]
        t2 = Table(data2, colWidths=[120, 320])
        t2.setStyle(TableStyle([('BACKGROUND', (0, 0), (0, 1),
colors.lightskyblue),
                                ('GRID', (0, 0), (1, 1), 0.5, colors.black),
                                ('FONTSIZE', (0, 0), (1, 1), 10)]))
        elements.append(t2)
        # antecedentes
        elements.append(Spacer(0, 20))
        p3 = Paragraph('ANTECENDENTES: ', estilo["BodyText"])
        elements.append(p3)
        elements.append(Spacer(0, 10))
        patp = str(self.datos[0][20]).decode(' utf-8 ')
        patologicop = self.salto(patp, 80)
        patq = str(self.datos[0][21]).decode(' utf-8 ')
        patologicoq = self.salto(patq, 80)
        data3 = [
            ['Patologicos personales:', patologicop],
            ['Patologicos quirurgicos:', patologicoq]
        ]
        t3 = Table(data3, colWidths=[120, 320])
        t3.setStyle(TableStyle([('BACKGROUND', (0, 0), (0, 1),
colors.lightskyblue),
                                ('GRID', (0, 0), (1, 1), 0.5, colors.black),
                                ('FONTSIZE', (0, 0), (1, 1), 10)]))
        elements.append(t3)
        # informacion de la extremidad amputada
        elements.append(Spacer(0, 20))
        p4 = Paragraph('EXTREMIDAD AMPUTADA: ', estilo["BodyText"])
        elements.append(p4)
        elements.append(Spacer(0, 10))
        f_amp = pickle.loads(self.datos[0][16])
        f = str(f_amp[0])+"/"+str(f_amp[1])+"/"+str(f_amp[2])
        data4 = [['Extremidad: ', str(self.datos[0][13]), 'Fecha de Amputacion: ',
f],
                 ['Amputacion: ', str(self.datos[0][14]), 'Terapia Posoperatoria:
', str(self.datos[0][17])],
                 ['Causa: ', str(self.datos[0][15]), '', '']]
        t4 = Table(data4, colWidths=[90, 130, 130, 90])
        t4.setStyle(TableStyle([('BACKGROUND', (0, 0), (0, 2),
colors.lightskyblue),
                                ('BACKGROUND', (2, 0), (2, 1), colors.lightskyblue),
                                ('GRID', (0, 0), (3, 1), 0.5, colors.black),
                                ('GRID', (0, 2), (1, 2), 0.5, colors.black),
                                ('FONTSIZE', (0, 0), (3, 2), 10)]))
        elements.append(t4)
        # grafica e informacion de los musculos que componen al cuadriceps
        self.figure.savefig('plotC.png')
        self.figure1.savefig('plotI.png')
        imagen = Image('plotC.png', width=450, height=150)
        elements.append(Spacer(0, 20))
        elements.append(imagen)
        elements.append(PageBreak())
        data5 = [['Fecha', 'Recto', '', 'Vasto lateral', '', 'Vasto medial', '',
'Promedio', ''],
                 ['', 'Nivel(%)', 'Punt.', 'Nivel(%)', 'Punt.', 'Nivel(%)',
'Punt.', 'Nivel(%)', 'Punt.']]
        for cont in range(0, len(self.cuadri[0])):
            aux = [self.cuadri[4][cont], self.cuadri[0][cont],
self.cuadri[5][cont], self.cuadri[1][cont], self.cuadri[6][cont],
                   self.cuadri[2][cont], self.cuadri[7][cont],
self.cuadri[3][cont], self.cuadri[8][cont]]
            data5.append(aux)
        t5 = Table(data5)
        t5.setStyle(TableStyle([('BACKGROUND', (0, 0), (8, 1),
colors.lightskyblue),
                                ('GRID', (0, 0), (8, len(data5)-1), 0.5,
colors.black),
```

```
                                ('FONTSIZE', (0, 0), (8, len(data5)-1), 10),
                                ('ALIGN', (0, 0), (8, len(data5)-1), 'CENTER'),
                                ('SPAN', (0, 0), (0, 1)),
                                ('SPAN', (1, 0), (2, 0)),
                                ('SPAN', (3, 0), (4, 0)),
                                ('SPAN', (5, 0), (6, 0)),
                                ('SPAN', (7, 0), (8, 0))]))
        elements.append(t5)
        # grafica e informacion de los musculos que componen al isquiotibial
        elements.append(Spacer(0, 30))
        imagen1 = Image('plotI.png', width=450, height=150)
        elements.append(imagen1)
        elements.append(Spacer(0, 10))
        data6 = [['Fecha', 'Biceps', '', 'Semitendinoso', '', 'Semimembranoso', '',
'Isquiotibial', ''],
                 ['', 'Nivel(%)', 'Punt.', 'Nivel(%)', 'Punt.', 'Nivel(%)',
'Punt.', 'Nivel(%)', 'Punt.']]
        for cont in range(0, len(self.isquio[0])):
            aux = [self.isquio[4][cont], self.isquio[0][cont],
self.isquio[5][cont], self.isquio[1][cont], self.isquio[6][cont],
                   self.isquio[2][cont], self.isquio[7][cont],
self.isquio[3][cont], self.isquio[8][cont]]
            data6.append(aux)
        t6 = Table(data6)
        t6.setStyle(TableStyle([('BACKGROUND', (0, 0), (8, 1),
colors.lightskyblue),
                                ('GRID', (0, 0), (8, len(data5) - 1), 0.5,
colors.black),
                                ('FONTSIZE', (0, 0), (8, len(data5) - 1), 10),
                                ('ALIGN', (0, 0), (8, len(data5) - 1), 'CENTER'),
                                ('SPAN', (0, 0), (0, 1)),
                                ('SPAN', (1, 0), (2, 0)),
                                ('SPAN', (3, 0), (4, 0)),
                                ('SPAN', (5, 0), (6, 0)),
                                ('SPAN', (7, 0), (8, 0))]))
        elements.append(t6)
        # construccion del PDF
        c.build(elements)

    def aceptar(self):
        # retorna a la pantalla principal
        try:
            os.remove('plotC.png')
            os.remove('plotI.png')
            os.remove('plotC1.png')
            os.remove('plotI1.png')
        except:
            pass
        self.close()
        myapp.__init__()
        myapp.show()
        myapp.mostrarTabla()

    def mostrar(self):
        # muesta la informacion del paciente almacenada en la base de datos
        DB.cursor.execute("""SELECT NOMBRE, NACIMIENTO, SEXO, PESO, TALLA, IMC, HC,
NACIONALIDAD, PROCEDENCIA, OCUPACION, RESIDENCIA, ECIVIL, FECHINGRESO, MIEMBRO,
AMPUTACION, CAUSA, FECHAMPU, TERAPIA, ENFERMEDAD, MEDIOS, PATOLOGICOP, PATOLOGICOQ
FROM PACIENTES WHERE CI = ?""", [str(myapp.ci)])
        self.datos = DB.cursor.fetchall()
        for fila in self.datos:
            # datos generales del paciente
            nombre = fila[0].decode(' utf-8 ')
            self.ui.nombre.setText(nombre)
            self.ui.ci.setText(str(myapp.ci))
            nac = pickle.loads(fila[1])
            if nac[1] > self.fecha.month:
                self.edad = self.fecha.year - nac[2] - 1
```

```
            else:
                self.edad = self.fecha.year - nac[2]
            self.ui.edad.setText(str(self.edad))
            self.ui.sexo.setText(fila[2])
            self.ui.peso.setText(str(fila[3]))
            self.ui.talla.setText(str(fila[4]))
            self.ui.IMC.setText(str(fila[5]))
            self.ui.HC.setText("# de historia: "+str(fila[6]))
            nacion = fila[7].decode(' utf-8 ')
            self.ui.nacionalidad.setText(nacion)
            procedencia = fila[8].decode(' utf-8 ')
            self.ui.procedencia.setText(procedencia)
            ocupacion = fila[9].decode(' utf-8 ')
            self.ui.ocupacion.setText(ocupacion)
            residencia = fila[10].decode(' utf-8 ')
            self.ui.residencia.setText(residencia)
            self.ui.civil.setText(fila[11])
            f_i = pickle.loads(fila[12])
            self.ui.ingreso.setText(str(f_i[0]) + "/" + str(f_i[1]) + "/" +
str(f_i[2]))
            # informacion de la extremidad amputada
            self.ui.extremidad.setText(fila[13])
            self.ui.amputacion.setText(fila[14])
            self.ui.causa.setText(fila[15])
            f_amp = pickle.loads(fila[16])

self.ui.fecha_a.setText(str(f_amp[0])+"/"+str(f_amp[1])+"/"+str(f_amp[2]))
            self.ui.tratamiento.setText(fila[17])
            # informacion medica actual
            enf = fila[18].decode(' utf-8 ')
            self.ui.enfermedad.setPlainText(enf)
            medios = pickle.loads(fila[19])
            self.Medios = ""
            if medios["Radiografia"] is True:
                self.Medios += "Radiografia, "
            if medios["Resonancia"] is True:
                self.Medios += "Resonancia electromagnetica, "
            if medios["Electromiografia"] is True:
                self.Medios += "Electromiografia, "
            if medios["TAC"] is True:
                self.Medios += "TAC, "
            if medios["Ecografia"] is True:
                self.Medios += "Ecografia, "
            self.ui.medios.setPlainText(self.Medios[0:len(self.Medios)-2])
            # antecedentes
            patologicop = fila[20].decode(' utf-8 ')
            self.ui.antecedenteP.setPlainText(patologicop)
            patologicoq = fila[21].decode(' utf-8 ')
            self.ui.antecedenteQ.setPlainText(patologicoq)
        # genera las graficas evolutivas de la condicion del paciente
        self.plot()
        self.canvas1.draw()
        self.canvas2.draw()
        # conecta una accion al hacer clic en la grafica
        self.canvas1.mpl_connect('button_press_event', onclick)
        self.canvas2.mpl_connect('button_press_event', onclick1)

    def salto(self, nums, elems):
        # retorna un parrafo a partir de un texto lineal
        final = ''
        palabra = ''
        steps = 0
        for num in nums:
            steps += 1
            palabra += num
            if num == " ":
                if steps >= elems:
                    final += '\n'
```

```
                    steps = 0
                else:
                    final += palabra
                    palabra = ''
        if nums[len(nums)-1] != " ":
            final += palabra
        return final

    def plot(self):
        # crea las graficas evolutivas a partir de la informacion en la base de
datos
        self.cuadri = [[], [], [], [], [], [], [], [], []]
        self.isquio = [[], [], [], [], [], [], [], [], []]
        DB.cursor.execute("""SELECT RECTO, VLATERAL, VMEDIO, CUADRICEPS, BICEPS,
SEMITEN, SEMIMEM, ISQUIOTIBIAL, FECHA, PUNT_RECTO, PUNT_VLATERAL, PUNT_VMEDIO,
PUNT_CUADRICEPS, PUNT_BICEPS, PUNT_SEMITEN, PUNT_SEMIMEM, PUNT_ISQUIOTIBIAL FROM
C""" + str(myapp.ci))
        dato = DB.cursor.fetchall()
        for row in dato[1:len(dato)]:
            fech = pickle.loads(row[8])
            f = str(fech[0]) + '/' + str(fech[1]) + '/' + str(fech[2])
            if row[3] is not None:
                # informacion del cuadriceps
                self.cuadri[0].append(row[0])
                self.cuadri[1].append(row[1])
                self.cuadri[2].append(row[2])
                self.cuadri[3].append(row[3])
                self.cuadri[4].append(f)
                self.cuadri[5].append(row[9])
                self.cuadri[6].append(row[10])
                self.cuadri[7].append(row[11])
                self.cuadri[8].append(row[12])
            if row[7] is not None:
                # informacion del isquiotibial
                self.isquio[0].append(row[4])
                self.isquio[1].append(row[5])
                self.isquio[2].append(row[6])
                self.isquio[3].append(row[7])
                self.isquio[4].append(f)
                self.isquio[5].append(row[13])
                self.isquio[6].append(row[14])
                self.isquio[7].append(row[15])
                self.isquio[8].append(row[16])
        # configuracion de las graficas
        ax = self.figure.add_subplot(111)
        ax.clear()
        ax.plot_date(x=self.cuadri[4], y=self.cuadri[0], color='r', ls='solid',
label='Recto Femoral')
        ax.plot_date(x=self.cuadri[4], y=self.cuadri[1], color='b', ls='solid',
label='Vasto Lateral')
        ax.plot_date(x=self.cuadri[4], y=self.cuadri[2], color='g', ls='solid',
label='Vasto Medial')
        ax.plot_date(x=self.cuadri[4], y=self.cuadri[3], color='y', ls='solid',
label='Promedio')
        ax.legend()
        ax.set_ylabel('Nivel de Fuerza muscular (%)')
        ax.set_xlabel('Fecha')
        ax.set_ylim(0, 105)
        ax.set_title("Cuadro Evolutivo del Cuadriceps")
        bx = self.figure1.add_subplot(111)
        bx.clear()
        bx.plot_date(x=self.isquio[4], y=self.isquio[0], color='r', ls='solid',
label='Biceps')
        bx.plot_date(x=self.isquio[4], y=self.isquio[1], color='b', ls='solid',
label='Semitendinoso')
        bx.plot_date(x=self.isquio[4], y=self.isquio[2], color='g', ls='solid',
label='Semimembranoso')
        bx.plot_date(x=self.isquio[4], y=self.isquio[3], color='y', ls='solid',
```

```
label='Isquiotibial')
        bx.legend()
        bx.set_ylabel('Nivel de Fuerza muscular (%)')
        bx.set_xlabel('Fecha')
        bx.set_ylim(0, 105)
        bx.set_title("Cuadro Evolutivo del Isquiotibial")



def onclick(event):
    # retorna el nivel de fuerza en porcentaje y la puntuacion del punto
seleccionado en la grafica del cuadriceps
    ix = event.xdata
    try:
        ix = int(round(ix, 0))
        historia.ui.recto.setValue(historia.cuadri[0][ix])
        historia.ui.vasto_L.setValue(historia.cuadri[1][ix])
        historia.ui.vasto_M.setValue(historia.cuadri[2][ix])
        historia.ui.cuadriceps.setValue(historia.cuadri[3][ix])
        historia.ui.rectoP.setText(historia.cuadri[5][ix])
        historia.ui.vasto_LP.setText(historia.cuadri[6][ix])
        historia.ui.vasto_MP.setText(historia.cuadri[7][ix])
        historia.ui.cuadricepsP.setText(historia.cuadri[8][ix])
    except:
        pass

def onclick1(event):
    # retorna el nivel de fuerza en porcentaje y la puntuacion del punto
seleccionado en la grafica del isquiotibial
    ix = event.xdata
    try:
        ix = int(round(ix, 0))
        historia.ui.biceps.setValue(historia.isquio[0][ix])
        historia.ui.semiten.setValue(historia.isquio[1][ix])
        historia.ui.semimem.setValue(historia.isquio[2][ix])
        historia.ui.isquiotibiales.setValue(historia.isquio[3][ix])
        historia.ui.bicepsP.setText(historia.isquio[5][ix])
        historia.ui.semitenP.setText(historia.isquio[6][ix])
        historia.ui.semimemP.setText(historia.isquio[7][ix])
        historia.ui.isquiotibialesP.setText(historia.isquio[8][ix])
    except:
        pass



# Pantalla de registro
class V_registro(QtGui.QDialog):
    def __init__(self, parent=None):
        QtGui.QWidget.__init__(self, parent)
        self.ui = Ui_Registro()
        self.ui.setupUi(self)
        self.registro = False
        self.error = QtGui.QMessageBox()
        self.error.setIcon(QMessageBox.Warning)
        self.error.setWindowTitle("ADVERTENCIA")
        # inicializacion de variables
        self.IMC = 0
        self.informacion = []
        # conexion de acciones con los botones
        self.ui.btn_aceptar.clicked.connect(self.aceptar)
        self.ui.btn_cancelar.clicked.connect(self.cancelar)
        self.ui.talla.valueChanged.connect(self.cambiaValor)
        self.ui.peso.valueChanged.connect(self.cambiaValor)

    def editar(self):
        # muestra la informacion del paciente almacenada en la base de datos para
ser editada
        DB.cursor.execute("""SELECT NOMBRE, NACIMIENTO, SEXO, PESO, TALLA, HC,
```

```
NACIONALIDAD, PROCEDENCIA, OCUPACION, RESIDENCIA, ECIVIL, FECHINGRESO, MIEMBRO,
AMPUTACION, CAUSA, FECHAMPU, TERAPIA, ENFERMEDAD, MEDIOS, PATOLOGICOP, PATOLOGICOQ
FROM PACIENTES WHERE CI LIKE ?""", [str(myapp.ci)])
        datos = DB.cursor.fetchall()
        for fila in datos:
            # datos generales del paciente
            self.ui.CI.setText(myapp.ci)
            self.ui.CI.setEnabled(False)
            nombre = fila[0].decode(' utf-8 ')
            self.ui.nombre.setText(nombre)
            nac = pickle.loads(fila[1])
            self.ui.nacimiento.setDate(QtCore.QDate(nac[2], nac[1], nac[0]))
            if fila[2] == 'Femenino':
                self.ui.rb_fem.setChecked(True)
            else:
                self.ui.rb_masc.setChecked(True)
            self.ui.peso.setValue(fila[3])
            self.ui.talla.setValue(fila[4])
            self.ui.HC.setValue(fila[5])
            nacion = fila[6].decode(' utf-8 ')
            self.ui.nacionalidad.setText(nacion)
            procedencia = fila[7].decode(' utf-8 ')
            self.ui.procedencia.setText(procedencia)
            ocupacion = fila[8].decode(' utf-8 ')
            self.ui.ocupacion.setText(ocupacion)
            residencia = fila[9].decode(' utf-8 ')
            self.ui.residencia.setText(residencia)
            civil = self.ui.civil.findText(fila[10])
            self.ui.civil.setCurrentIndex(civil)
            f_i = pickle.loads(fila[11])
            # informacion de la extremidad amputada
            self.ui.ingreso.setDate(QtCore.QDate(f_i[2], f_i[1], f_i[0]))
            im = self.ui.extremidad.findText(fila[12])
            self.ui.extremidad.setCurrentIndex(im)
            ia = self.ui.amputacion.findText(fila[13])
            self.ui.amputacion.setCurrentIndex(ia)
            ic = self.ui.causa.findText(fila[14])
            self.ui.causa.setCurrentIndex(ic)
            f_amp = pickle.loads(fila[15])
            self.ui.fecha_a.setDate(QtCore.QDate(f_amp[2], f_amp[1], f_amp[0]))
            if fila[16] == 'Si':
                self.ui.T_si.setChecked(True)
            else:
                self.ui.T_no.setChecked(True)
            # informacion medica actual
            enf = fila[17].decode(' utf-8 ')
            self.ui.enfermedad.setPlainText(enf)
            medios = pickle.loads(fila[18])
            if medios["Radiografia"] is True:
                self.ui.radiografia.setChecked(True)
            if medios["Resonancia"] is True:
                self.ui.resonancia.setChecked(True)
            if medios["Electromiografia"] is True:
                self.ui.electromiografia.setChecked(True)
            if medios["TAC"] is True:
                self.ui.TAC.setChecked(True)
            if medios["Ecografia"] is True:
                self.ui.ecografia.setChecked(True)
            # antecedentes
            patologiap = fila[19].decode(' utf-8 ')
            self.ui.patologicoP.setPlainText(patologiap)
            patologiaq = fila[20].decode(' utf-8 ')
            self.ui.patologicoQ.setPlainText(patologiaq)

    def aceptar_edit(self):
        # actualiza la informacion del paciente en la base de datos
        DB.cursor.execute("""
                          UPDATE PACIENTES SET NOMBRE = ?, NACIMIENTO = ?, SEXO =
```

```
?, PESO = ?, TALLA = ?, IMC = ?, HC = ?, NACIONALIDAD = ?, PROCEDENCIA = ?,
OCUPACION = ?, RESIDENCIA = ?, ECIVIL = ?, FECHINGRESO = ?, MIEMBRO = ?, AMPUTACION
= ?, CAUSA = ?, FECHAMPU = ?, TERAPIA = ?, ENFERMEDAD = ?, MEDIOS = ?, PATOLOGICOP
= ?, PATOLOGICOQ = ?, FECHA = ? WHERE CI LIKE ?""", self.informacion)
        DB.conexion.commit()
        self.close()
        myapp.__init__()
        myapp.show()
        myapp.mostrarTabla()

    def cambiaValor(self):
        # calculo del IMC
        peso = self.ui.peso.value()
        talla = self.ui.talla.value()
        if talla != 0 and peso != 0:
            aux = peso / (talla**2)
            self.IMC = round(aux, 2)
            self.ui.IMC.setValue(self.IMC)

    def cancelar(self):
        # regresa a la pantalla principal sin guardar ningun dato
        self.close()
        myapp.show()
        myapp.mostrarTabla()

    def aceptar(self):
        # adquiere la informacion llenada en la pantalla de registro
        fecha = datetime.datetime.now()
        f = [fecha.day, fecha.month, fecha.year]
        nombre = str(self.ui.nombre.text().toUtf8())
        ci = str(self.ui.CI.text())
        dia_nac = self.ui.nacimiento.date().day()
        mes_nac = self.ui.nacimiento.date().month()
        a_nac = self.ui.nacimiento.date().year()
        nac = [dia_nac, mes_nac, a_nac]
        sexo = ''
        if self.ui.rb_fem.isChecked() is True:
            sexo = 'Femenino'
        if self.ui.rb_masc.isChecked() is True:
            sexo = 'Masculino'
        peso = self.ui.peso.value()
        talla = self.ui.talla.value()
        hc = self.ui.HC.value()
        nacion = str(self.ui.nacionalidad.text().toUtf8())
        procedencia = str(self.ui.procedencia.text().toUtf8())
        ocupacion = str(self.ui.ocupacion.text().toUtf8())
        residencia = str(self.ui.residencia.text().toUtf8())
        civil = str(self.ui.civil.currentText())
        dia_i = self.ui.ingreso.date().day()
        mes_i = self.ui.ingreso.date().month()
        a_i = self.ui.ingreso.date().year()
        i = [dia_i, mes_i, a_i]
        # extremidad amputada
        miembro = str(self.ui.extremidad.currentText())
        amputacion = str(self.ui.amputacion.currentText())
        causa = str(self.ui.causa.currentText())
        dia_a = self.ui.fecha_a.date().day()
        mes_a = self.ui.fecha_a.date().month()
        a_amp = self.ui.fecha_a.date().year()
        fecha_amp = [dia_a, mes_a, a_amp]
        terapia = ''
        if self.ui.T_si.isChecked() is True:
            terapia = 'Si'
        if self.ui.T_no.isChecked() is True:
            terapia = 'No'
        # conversion de los vectores a bloob
        nacimiento = pickle.dumps(nac)
        fec = pickle.dumps(f)
```

```
        f_amp = pickle.dumps(fecha_amp)
        ingreso = pickle.dumps(i)
        # informacion medica actual y antecedentes
        enfermedad = str(self.ui.enfermedad.toPlainText().toUtf8())
        patologicop = str(self.ui.patologicoP.toPlainText().toUtf8())
        patologicoq = str(self.ui.patologicoQ.toPlainText().toUtf8())
        medios = {"Radiografia": False, "Ecografia": False, "TAC": False,
"Electromiografia": False, "Resonancia": False}
        if self.ui.radiografia.isChecked() is True:
            medios["Radiografia"] = True
        if self.ui.ecografia.isChecked() is True:
            medios["Ecografia"] = True
        if self.ui.TAC.isChecked() is True:
            medios["TAC"] = True
        if self.ui.electromiografia.isChecked() is True:
            medios["Electromiografia"] = True
        if self.ui.resonancia.isChecked() is True:
            medios["Resonancia"] = True
        medio = pickle.dumps(medios)
        # comprobacion de que la informacion mas importante fue ingresada
        if (nombre == '' or ci == '' or a_nac <= 1753 or miembro == '' or
amputacion == '' or  causa == '' or self.IMC == 0
            or a_amp <= 1753 or a_i <= 1753 or nacion == '' or procedencia == '' or
ocupacion == '' or residencia == ''
            or hc == 0 or civil == ''):
            self.error.setText('Ingrese toda la informacion solicitada')
            self.error.exec_()
        elif a_amp < a_nac or a_i < a_nac:
            # comprobacion de la concordacia de las fechas
            self.error.setText('Ingrese correctamente las fechas')
            self.error.exec_()
        else:
            # comprobacion de la concordancia del numero de cedula
            self.informacion = [nombre, nacimiento, sexo, peso, talla, self.IMC,
hc, nacion, procedencia, ocupacion,
                                residencia, civil, ingreso, miembro, amputacion,
causa, f_amp, terapia, enfermedad,
                                medio, patologicop, patologicoq, fec, ci]
            if myapp.edit is False:
                cadena = re.sub(r'\d', '', ci)
                if len(ci) != 10 or len(cadena) > 0:
                    self.error.setText('Ingrese correctamente su numero de cedula')
                    self.error.exec_()
                else:
                    self.aceptar_reg()
            else:
                self.aceptar_edit()

    def aceptar_reg(self):
        # inserta la informacion de un nuevo paciente en la tabla paciente de la
base de datos
        try:
            DB.cursor.execute("""INSERT INTO PACIENTES(NOMBRE, NACIMIENTO, SEXO,
PESO, TALLA, IMC, HC, NACIONALIDAD, PROCEDENCIA, OCUPACION, RESIDENCIA, ECIVIL,
FECHINGRESO, MIEMBRO, AMPUTACION, CAUSA, FECHAMPU, TERAPIA, ENFERMEDAD, MEDIOS,
PATOLOGICOP, PATOLOGICOQ, FECHA, CI) VALUES(?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?,
?, ?, ?, ?, ?, ?, ?, ?, ?, ?)""", self.informacion)
            DB.conexion.commit()
            aux = False
        except:
            self.error.setText('La persona ya esta registrada')
            self.error.exec_()
            aux = True
        if aux is False:
            # muestra la pantalla de evaluacion inicial
            DB.DB_datos(self.informacion[23])
            myapp.ci = self.informacion[23]
            self.close()
```

```
            self.registro = "Sana"
            eval.__init__()
            eval.ui.resultados.setText("Resultados Extremidad Sana")
            eval.ui.pb_biceps.close()
            eval.ui.pb_cuadriceps.close()
            eval.ui.pb_isquiotibial.close()
            eval.ui.pb_recto.close()
            eval.ui.pb_semimem.close()
            eval.ui.pb_semiten.close()
            eval.ui.pb_vastoL.close()
            eval.ui.pb_vastoM.close()
            eval.ui.btn_cancelar.close()
            eval.ui.RectoFP.close()
            eval.ui.VastoLP.close()
            eval.ui.VastoMP.close()
            eval.ui.CuadricepsP.close()
            eval.ui.BicepsP.close()
            eval.ui.SemimemP.close()
            eval.ui.SemitenP.close()
            eval.ui.IsquiotibialesP.close()
            eval.show()
            eval.showMaximized()


# Pantalla de ayuda para la pantalla principal
class V_ayudaP(QtGui.QDialog):
    def __init__(self, parent=None):
        QtGui.QWidget.__init__(self, parent)
        self.ui = Ui_ayudaP()
        self.ui.setupUi(self)

# Pantalla de ayuda para la pantalla de evaluacion
class V_ayudaE(QtGui.QDialog):
    def __init__(self, parent=None):
        QtGui.QWidget.__init__(self, parent)
        self.ui = Ui_ayudaE()
        self.ui.setupUi(self)

# Pantalla de ayuda para la pantalla historia clinica
class V_ayudaH(QtGui.QDialog):
    def __init__(self, parent=None):
        QtGui.QWidget.__init__(self, parent)
        self.ui = Ui_ayudaH()
        self.ui.setupUi(self)


if __name__ == '__main__':
    app = QtGui.QApplication(sys.argv)
    hilo = hilo2()
    DB = base_datos()
    historia = V_historia()
    eval = V_evaluacion()
    reg = V_registro()
    myapp = V_principal()
    myapp.show()
    sys.exit(app.exec_())
    DB.conexion.close()
```

1) Código de la pantalla principal

```
# -*- coding: utf-8 -*-

# Form implementation generated from reading ui file 'principal.ui'
#
# Created by: PyQt4 UI code generator 4.11.4
#
# WARNING! All changes made in this file will be lost!

from PyQt4 import QtCore, QtGui

try:
    _fromUtf8 = QtCore.QString.fromUtf8
except AttributeError:
    def _fromUtf8(s):
        return s

try:
    _encoding = QtGui.QApplication.UnicodeUTF8
    def _translate(context, text, disambig):
        return QtGui.QApplication.translate(context, text, disambig, _encoding)
except AttributeError:
    def _translate(context, text, disambig):
        return QtGui.QApplication.translate(context, text, disambig)

class Ui_principal(object):
    def setupUi(self, principal):
        principal.setObjectName(_fromUtf8("principal"))
        principal.resize(1280, 729)
        font = QtGui.QFont()
        font.setFamily(_fromUtf8("Leelawadee UI"))
        font.setPointSize(11)
        principal.setFont(font)
        self.centralwidget = QtGui.QWidget(principal)
        self.centralwidget.setObjectName(_fromUtf8("centralwidget"))
        self.gridLayout = QtGui.QGridLayout(self.centralwidget)
        self.gridLayout.setObjectName(_fromUtf8("gridLayout"))
        self.frame = QtGui.QFrame(self.centralwidget)
        sizePolicy = QtGui.QSizePolicy(QtGui.QSizePolicy.Minimum,
QtGui.QSizePolicy.Preferred)
        sizePolicy.setHorizontalStretch(0)
        sizePolicy.setVerticalStretch(0)
        sizePolicy.setHeightForWidth(self.frame.sizePolicy().hasHeightForWidth())
        self.frame.setSizePolicy(sizePolicy)
        self.frame.setMaximumSize(QtCore.QSize(350, 16777215))
        self.frame.setStyleSheet(_fromUtf8("background-color: rgb(216, 213,
255);"))
        self.frame.setFrameShape(QtGui.QFrame.Panel)
        self.frame.setFrameShadow(QtGui.QFrame.Sunken)
        self.frame.setLineWidth(1)
        self.frame.setObjectName(_fromUtf8("frame"))
        self.verticalLayout = QtGui.QVBoxLayout(self.frame)
        self.verticalLayout.setMargin(20)
        self.verticalLayout.setSpacing(30)
        self.verticalLayout.setObjectName(_fromUtf8("verticalLayout"))
        self.label = QtGui.QLabel(self.frame)
        sizePolicy = QtGui.QSizePolicy(QtGui.QSizePolicy.Preferred,
QtGui.QSizePolicy.Preferred)
        sizePolicy.setHorizontalStretch(0)
        sizePolicy.setVerticalStretch(0)
        sizePolicy.setHeightForWidth(self.label.sizePolicy().hasHeightForWidth())
        self.label.setSizePolicy(sizePolicy)
        self.label.setText(_fromUtf8(""))
        self.label.setPixmap(QtGui.QPixmap(_fromUtf8("LOGO.png")))
        self.label.setScaledContents(True)
```

```
        self.label.setObjectName(_fromUtf8("label"))
        self.verticalLayout.addWidget(self.label)
        self.label_2 = QtGui.QLabel(self.frame)
        sizePolicy = QtGui.QSizePolicy(QtGui.QSizePolicy.Preferred,
QtGui.QSizePolicy.Expanding)
        sizePolicy.setHorizontalStretch(0)
        sizePolicy.setVerticalStretch(0)
        sizePolicy.setHeightForWidth(self.label_2.sizePolicy().hasHeightForWidth())
        self.label_2.setSizePolicy(sizePolicy)
        self.label_2.setText(_fromUtf8(""))
        self.label_2.setPixmap(QtGui.QPixmap(_fromUtf8("amputado.jpg")))
        self.label_2.setScaledContents(True)
        self.label_2.setObjectName(_fromUtf8("label_2"))
        self.verticalLayout.addWidget(self.label_2)
        self.gridLayout.addWidget(self.frame, 0, 0, 4, 1)
        self.frame_2 = QtGui.QFrame(self.centralwidget)
        self.frame_2.setMaximumSize(QtCore.QSize(16777215, 50))
        self.frame_2.setFrameShape(QtGui.QFrame.Box)
        self.frame_2.setFrameShadow(QtGui.QFrame.Sunken)
        self.frame_2.setObjectName(_fromUtf8("frame_2"))
        self.horizontalLayout = QtGui.QHBoxLayout(self.frame_2)
        self.horizontalLayout.setObjectName(_fromUtf8("horizontalLayout"))
        self.label_3 = QtGui.QLabel(self.frame_2)
        font = QtGui.QFont()
        font.setBold(False)
        font.setWeight(50)
        self.label_3.setFont(font)
        self.label_3.setObjectName(_fromUtf8("label_3"))
        self.horizontalLayout.addWidget(self.label_3)
        self.nombre = QtGui.QLineEdit(self.frame_2)
        self.nombre.setObjectName(_fromUtf8("nombre"))
        self.horizontalLayout.addWidget(self.nombre)
        self.label_4 = QtGui.QLabel(self.frame_2)
        font = QtGui.QFont()
        font.setBold(False)
        font.setWeight(50)
        self.label_4.setFont(font)
        self.label_4.setObjectName(_fromUtf8("label_4"))
        self.horizontalLayout.addWidget(self.label_4)
        self.fecha = QtGui.QDateEdit(self.frame_2)
        self.fecha.setDate(QtCore.QDate(1753, 1, 1))
        self.fecha.setObjectName(_fromUtf8("fecha"))
        self.horizontalLayout.addWidget(self.fecha)
        self.btn_buscar = QtGui.QPushButton(self.frame_2)
        icon = QtGui.QIcon()
        icon.addPixmap(QtGui.QPixmap(_fromUtf8("buscar.png")), QtGui.QIcon.Normal,
QtGui.QIcon.Off)
        self.btn_buscar.setIcon(icon)
        self.btn_buscar.setIconSize(QtCore.QSize(30, 25))
        self.btn_buscar.setObjectName(_fromUtf8("btn_buscar"))
        self.horizontalLayout.addWidget(self.btn_buscar)
        self.btn_editar = QtGui.QPushButton(self.frame_2)
        icon1 = QtGui.QIcon()
        icon1.addPixmap(QtGui.QPixmap(_fromUtf8("editar.png")), QtGui.QIcon.Normal,
QtGui.QIcon.Off)
        self.btn_editar.setIcon(icon1)
        self.btn_editar.setIconSize(QtCore.QSize(30, 25))
        self.btn_editar.setObjectName(_fromUtf8("btn_editar"))
        self.horizontalLayout.addWidget(self.btn_editar)
        self.btn_eliminar = QtGui.QPushButton(self.frame_2)
        icon2 = QtGui.QIcon()
        icon2.addPixmap(QtGui.QPixmap(_fromUtf8("elimi.png")), QtGui.QIcon.Normal,
QtGui.QIcon.Off)
        self.btn_eliminar.setIcon(icon2)
        self.btn_eliminar.setIconSize(QtCore.QSize(25, 25))
        self.btn_eliminar.setObjectName(_fromUtf8("btn_eliminar"))
        self.horizontalLayout.addWidget(self.btn_eliminar)
        self.btn_ayuda = QtGui.QPushButton(self.frame_2)
```

```
        self.btn_ayuda.setText(_fromUtf8(""))
        icon3 = QtGui.QIcon()
        icon3.addPixmap(QtGui.QPixmap(_fromUtf8("ayuda.png")), QtGui.QIcon.Normal,
QtGui.QIcon.Off)
        self.btn_ayuda.setIcon(icon3)
        self.btn_ayuda.setIconSize(QtCore.QSize(25, 25))
        self.btn_ayuda.setCheckable(False)
        self.btn_ayuda.setAutoDefault(False)
        self.btn_ayuda.setDefault(False)
        self.btn_ayuda.setFlat(True)
        self.btn_ayuda.setObjectName(_fromUtf8("btn_ayuda"))
        self.horizontalLayout.addWidget(self.btn_ayuda)
        self.gridLayout.addWidget(self.frame_2, 0, 1, 1, 1)
        self.frame_3 = QtGui.QFrame(self.centralwidget)
        self.frame_3.setFrameShape(QtGui.QFrame.Box)
        self.frame_3.setFrameShadow(QtGui.QFrame.Sunken)
        self.frame_3.setObjectName(_fromUtf8("frame_3"))
        self.verticalLayout_2 = QtGui.QVBoxLayout(self.frame_3)
        self.verticalLayout_2.setContentsMargins(40, 20, 40, 20)
        self.verticalLayout_2.setSpacing(10)
        self.verticalLayout_2.setObjectName(_fromUtf8("verticalLayout_2"))
        self.btn_registro = QtGui.QPushButton(self.frame_3)
        self.btn_registro.setMinimumSize(QtCore.QSize(0, 60))
        font = QtGui.QFont()
        font.setBold(True)
        font.setWeight(75)
        self.btn_registro.setFont(font)
        icon4 = QtGui.QIcon()
        icon4.addPixmap(QtGui.QPixmap(_fromUtf8("reg.png")), QtGui.QIcon.Normal,
QtGui.QIcon.Off)
        self.btn_registro.setIcon(icon4)
        self.btn_registro.setIconSize(QtCore.QSize(50, 50))
        self.btn_registro.setObjectName(_fromUtf8("btn_registro"))
        self.verticalLayout_2.addWidget(self.btn_registro)
        self.btn_eval = QtGui.QPushButton(self.frame_3)
        self.btn_eval.setMinimumSize(QtCore.QSize(0, 60))
        font = QtGui.QFont()
        font.setBold(True)
        font.setWeight(75)
        self.btn_eval.setFont(font)
        icon5 = QtGui.QIcon()
        icon5.addPixmap(QtGui.QPixmap(_fromUtf8("eval2.png")), QtGui.QIcon.Normal,
QtGui.QIcon.Off)
        self.btn_eval.setIcon(icon5)
        self.btn_eval.setIconSize(QtCore.QSize(50, 50))
        self.btn_eval.setObjectName(_fromUtf8("btn_eval"))
        self.verticalLayout_2.addWidget(self.btn_eval)
        self.btn_HC = QtGui.QPushButton(self.frame_3)
        self.btn_HC.setMinimumSize(QtCore.QSize(0, 60))
        font = QtGui.QFont()
        font.setBold(True)
        font.setWeight(75)
        self.btn_HC.setFont(font)
        icon6 = QtGui.QIcon()
        icon6.addPixmap(QtGui.QPixmap(_fromUtf8("resul.png")), QtGui.QIcon.Normal,
QtGui.QIcon.Off)
        self.btn_HC.setIcon(icon6)
        self.btn_HC.setIconSize(QtCore.QSize(50, 50))
        self.btn_HC.setObjectName(_fromUtf8("btn_HC"))
        self.verticalLayout_2.addWidget(self.btn_HC)
        self.gridLayout.addWidget(self.frame_3, 4, 0, 1, 1)
        self.tabla = QtGui.QTableWidget(self.centralwidget)
        font = QtGui.QFont()
        font.setFamily(_fromUtf8("Leelawadee UI"))
        font.setPointSize(11)
        self.tabla.setFont(font)
        self.tabla.setFrameShape(QtGui.QFrame.Box)
```

```
self.tabla.setEditTriggers(QtGui.QAbstractItemView.AnyKeyPressed|QtGui.QAbstractIte
mView.EditKeyPressed)
        self.tabla.setDragDropOverwriteMode(False)
        self.tabla.setSelectionMode(QtGui.QAbstractItemView.SingleSelection)
        self.tabla.setRowCount(0)
        self.tabla.setObjectName(_fromUtf8("tabla"))
        self.tabla.setColumnCount(7)
        item = QtGui.QTableWidgetItem()
        font = QtGui.QFont()
        font.setBold(True)
        font.setWeight(75)
        item.setFont(font)
        self.tabla.setHorizontalHeaderItem(0, item)
        item = QtGui.QTableWidgetItem()
        font = QtGui.QFont()
        font.setBold(True)
        font.setWeight(75)
        item.setFont(font)
        self.tabla.setHorizontalHeaderItem(1, item)
        item = QtGui.QTableWidgetItem()
        font = QtGui.QFont()
        font.setBold(True)
        font.setWeight(75)
        item.setFont(font)
        self.tabla.setHorizontalHeaderItem(2, item)
        item = QtGui.QTableWidgetItem()
        font = QtGui.QFont()
        font.setBold(True)
        font.setWeight(75)
        item.setFont(font)
        self.tabla.setHorizontalHeaderItem(3, item)
        item = QtGui.QTableWidgetItem()
        font = QtGui.QFont()
        font.setBold(True)
        font.setWeight(75)
        item.setFont(font)
        self.tabla.setHorizontalHeaderItem(4, item)
        item = QtGui.QTableWidgetItem()
        font = QtGui.QFont()
        font.setBold(True)
        font.setWeight(75)
        item.setFont(font)
        self.tabla.setHorizontalHeaderItem(5, item)
        item = QtGui.QTableWidgetItem()
        font = QtGui.QFont()
        font.setBold(True)
        font.setWeight(75)
        item.setFont(font)
        self.tabla.setHorizontalHeaderItem(6, item)
        self.tabla.horizontalHeader().setCascadingSectionResizes(False)
        self.tabla.horizontalHeader().setDefaultSectionSize(125)
        self.gridLayout.addWidget(self.tabla, 1, 1, 4, 1)
        principal.setCentralWidget(self.centralwidget)
        self.statusbar = QtGui.QStatusBar(principal)
        self.statusbar.setObjectName(_fromUtf8("statusbar"))
        principal.setStatusBar(self.statusbar)

        self.retranslateUi(principal)
        QtCore.QMetaObject.connectSlotsByName(principal)

    def retranslateUi(self, principal):
        principal.setWindowTitle(_translate("principal", "EVALUADOR MIOELÉCTRICO
MULTICANAL PARA PERSONAS CON AMPUTACIÓN TRANSTIBIAL", None))
        self.label_3.setText(_translate("principal", "Nombre", None))
        self.label_4.setText(_translate("principal", "Fecha", None))
        self.fecha.setDisplayFormat(_translate("principal", "d/M/yyyy", None))
        self.btn_buscar.setText(_translate("principal", "Buscar", None))
        self.btn_editar.setText(_translate("principal", "Editar", None))
```

```
        self.btn_eliminar.setText(_translate("principal", "Eliminar", None))
        self.btn_registro.setText(_translate("principal", "Registro", None))
        self.btn_eval.setText(_translate("principal", "Evaluación", None))
        self.btn_HC.setText(_translate("principal", "Historia Clínica", None))
        item = self.tabla.horizontalHeaderItem(0)
        item.setText(_translate("principal", "CI", None))
        item = self.tabla.horizontalHeaderItem(1)
        item.setText(_translate("principal", "Nombre", None))
        item = self.tabla.horizontalHeaderItem(2)
        item.setText(_translate("principal", "Edad", None))
        item = self.tabla.horizontalHeaderItem(3)
        item.setText(_translate("principal", "Sexo", None))
        item = self.tabla.horizontalHeaderItem(4)
        item.setText(_translate("principal", "IMC", None))
        item = self.tabla.horizontalHeaderItem(5)
        item.setText(_translate("principal", "Amputación", None))
        item = self.tabla.horizontalHeaderItem(6)
        item.setText(_translate("principal", "Fecha", None))
```

2) Código de la pantalla de Registro

```
# -*- coding: utf-8 -*-

# Form implementation generated from reading ui file 'registro.ui'
#
# Created by: PyQt4 UI code generator 4.11.4
#
# WARNING! All changes made in this file will be lost!

from PyQt4 import QtCore, QtGui

try:
    _fromUtf8 = QtCore.QString.fromUtf8
except AttributeError:
    def _fromUtf8(s):
        return s

try:
    _encoding = QtGui.QApplication.UnicodeUTF8
    def _translate(context, text, disambig):
        return QtGui.QApplication.translate(context, text, disambig, _encoding)
except AttributeError:
    def _translate(context, text, disambig):
        return QtGui.QApplication.translate(context, text, disambig)

class Ui_Registro(object):
    def setupUi(self, Registro):
        Registro.setObjectName(_fromUtf8("Registro"))
        Registro.resize(1126, 544)
        font = QtGui.QFont()
        font.setFamily(_fromUtf8("Leelawadee UI"))
        font.setPointSize(11)
        Registro.setFont(font)
        Registro.setSizeGripEnabled(False)
        self.gridLayout_3 = QtGui.QGridLayout(Registro)
        self.gridLayout_3.setObjectName(_fromUtf8("gridLayout_3"))
        self.frame_6 = QtGui.QFrame(Registro)
        self.frame_6.setMinimumSize(QtCore.QSize(350, 0))
        self.frame_6.setMaximumSize(QtCore.QSize(400, 16777215))
        self.frame_6.setStyleSheet(_fromUtf8("background-color: rgb(247, 247,
247);"))
```

```
        self.frame_6.setFrameShape(QtGui.QFrame.Box)
        self.frame_6.setFrameShadow(QtGui.QFrame.Sunken)
        self.frame_6.setObjectName(_fromUtf8("frame_6"))
        self.gridLayout_5 = QtGui.QGridLayout(self.frame_6)
        self.gridLayout_5.setObjectName(_fromUtf8("gridLayout_5"))
        self.radiografia = QtGui.QCheckBox(self.frame_6)
        font = QtGui.QFont()
        font.setFamily(_fromUtf8("Leelawadee UI"))
        font.setPointSize(11)
        self.radiografia.setFont(font)
        self.radiografia.setObjectName(_fromUtf8("radiografia"))
        self.gridLayout_5.addWidget(self.radiografia, 6, 0, 1, 1)
        self.resonancia = QtGui.QCheckBox(self.frame_6)
        self.resonancia.setMinimumSize(QtCore.QSize(200, 0))
        font = QtGui.QFont()
        font.setFamily(_fromUtf8("Leelawadee UI"))
        font.setPointSize(11)
        self.resonancia.setFont(font)
        self.resonancia.setObjectName(_fromUtf8("resonancia"))
        self.gridLayout_5.addWidget(self.resonancia, 7, 0, 1, 1)
        self.extremidad_5 = QtGui.QLabel(self.frame_6)
        self.extremidad_5.setMinimumSize(QtCore.QSize(0, 30))
        font = QtGui.QFont()
        font.setFamily(_fromUtf8("Leelawadee UI"))
        font.setPointSize(11)
        font.setBold(True)
        font.setWeight(75)
        self.extremidad_5.setFont(font)
        self.extremidad_5.setStyleSheet(_fromUtf8("background-color: rgb(216, 213,
255)"))
        self.extremidad_5.setFrameShape(QtGui.QFrame.Box)
        self.extremidad_5.setFrameShadow(QtGui.QFrame.Sunken)
        self.extremidad_5.setAlignment(QtCore.Qt.AlignCenter)
        self.extremidad_5.setObjectName(_fromUtf8("extremidad_5"))
        self.gridLayout_5.addWidget(self.extremidad_5, 0, 0, 1, 2)
        self.enfermedad = QtGui.QTextEdit(self.frame_6)
        font = QtGui.QFont()
        font.setFamily(_fromUtf8("Leelawadee UI"))
        font.setPointSize(11)
        self.enfermedad.setFont(font)
        self.enfermedad.setStyleSheet(_fromUtf8("background-color: rgb(255, 255,
255);"))
        self.enfermedad.setObjectName(_fromUtf8("enfermedad"))
        self.gridLayout_5.addWidget(self.enfermedad, 3, 0, 1, 2)
        spacerItem = QtGui.QSpacerItem(20, 10, QtGui.QSizePolicy.Minimum,
QtGui.QSizePolicy.Fixed)
        self.gridLayout_5.addItem(spacerItem, 1, 0, 1, 1)
        self.electromiografia = QtGui.QCheckBox(self.frame_6)
        font = QtGui.QFont()
        font.setFamily(_fromUtf8("Leelawadee UI"))
        font.setPointSize(11)
        self.electromiografia.setFont(font)
        self.electromiografia.setObjectName(_fromUtf8("electromiografia"))
        self.gridLayout_5.addWidget(self.electromiografia, 7, 1, 1, 1)
        self.label_17 = QtGui.QLabel(self.frame_6)
        self.label_17.setMinimumSize(QtCore.QSize(90, 0))
        font = QtGui.QFont()
        font.setFamily(_fromUtf8("Leelawadee UI"))
        font.setPointSize(11)
        font.setBold(False)
        font.setWeight(50)
        self.label_17.setFont(font)
        self.label_17.setObjectName(_fromUtf8("label_17"))
        self.gridLayout_5.addWidget(self.label_17, 2, 0, 1, 2)
        self.ecografia = QtGui.QCheckBox(self.frame_6)
        font = QtGui.QFont()
        font.setFamily(_fromUtf8("Leelawadee UI"))
        font.setPointSize(11)
```

```
        self.ecografia.setFont(font)
        self.ecografia.setObjectName(_fromUtf8("ecografia"))
        self.gridLayout_5.addWidget(self.ecografia, 6, 1, 1, 1)
        self.TAC = QtGui.QCheckBox(self.frame_6)
        font = QtGui.QFont()
        font.setFamily(_fromUtf8("Leelawadee UI"))
        font.setPointSize(11)
        self.TAC.setFont(font)
        self.TAC.setObjectName(_fromUtf8("TAC"))
        self.gridLayout_5.addWidget(self.TAC, 8, 0, 1, 1)
        self.label_25 = QtGui.QLabel(self.frame_6)
        self.label_25.setMinimumSize(QtCore.QSize(90, 0))
        font = QtGui.QFont()
        font.setFamily(_fromUtf8("Leelawadee UI"))
        font.setPointSize(11)
        font.setBold(False)
        font.setWeight(50)
        self.label_25.setFont(font)
        self.label_25.setObjectName(_fromUtf8("label_25"))
        self.gridLayout_5.addWidget(self.label_25, 5, 0, 1, 2)
        spacerItem1 = QtGui.QSpacerItem(20, 10, QtGui.QSizePolicy.Minimum,
QtGui.QSizePolicy.Fixed)
        self.gridLayout_5.addItem(spacerItem1, 4, 0, 1, 1)
        self.gridLayout_3.addWidget(self.frame_6, 1, 1, 2, 1)
        self.frame_4 = QtGui.QFrame(Registro)
        self.frame_4.setMinimumSize(QtCore.QSize(350, 0))
        self.frame_4.setMaximumSize(QtCore.QSize(350, 16777215))
        self.frame_4.setStyleSheet(_fromUtf8("background-color: rgb(247, 247,
247);"))
        self.frame_4.setFrameShape(QtGui.QFrame.Box)
        self.frame_4.setFrameShadow(QtGui.QFrame.Sunken)
        self.frame_4.setObjectName(_fromUtf8("frame_4"))
        self.gridLayout_2 = QtGui.QGridLayout(self.frame_4)
        self.gridLayout_2.setObjectName(_fromUtf8("gridLayout_2"))
        spacerItem2 = QtGui.QSpacerItem(20, 10, QtGui.QSizePolicy.Minimum,
QtGui.QSizePolicy.Fixed)
        self.gridLayout_2.addItem(spacerItem2, 1, 0, 1, 1)
        self.patologicoP = QtGui.QTextEdit(self.frame_4)
        font = QtGui.QFont()
        font.setFamily(_fromUtf8("Leelawadee UI"))
        font.setPointSize(11)
        self.patologicoP.setFont(font)
        self.patologicoP.setStyleSheet(_fromUtf8("background-color: rgb(255, 255,
255);"))
        self.patologicoP.setObjectName(_fromUtf8("patologicoP"))
        self.gridLayout_2.addWidget(self.patologicoP, 3, 0, 1, 1)
        self.extremidad_3 = QtGui.QLabel(self.frame_4)
        self.extremidad_3.setMinimumSize(QtCore.QSize(0, 30))
        self.extremidad_3.setMaximumSize(QtCore.QSize(16777215, 30))
        font = QtGui.QFont()
        font.setFamily(_fromUtf8("Leelawadee UI"))
        font.setPointSize(11)
        font.setBold(True)
        font.setWeight(75)
        self.extremidad_3.setFont(font)
        self.extremidad_3.setStyleSheet(_fromUtf8("background-color: rgb(216, 213,
255)"))
        self.extremidad_3.setFrameShape(QtGui.QFrame.Box)
        self.extremidad_3.setFrameShadow(QtGui.QFrame.Sunken)
        self.extremidad_3.setAlignment(QtCore.Qt.AlignCenter)
        self.extremidad_3.setObjectName(_fromUtf8("extremidad_3"))
        self.gridLayout_2.addWidget(self.extremidad_3, 0, 0, 1, 1)
        self.patologicoQ = QtGui.QTextEdit(self.frame_4)
        font = QtGui.QFont()
        font.setFamily(_fromUtf8("Leelawadee UI"))
        font.setPointSize(11)
        self.patologicoQ.setFont(font)
        self.patologicoQ.setStyleSheet(_fromUtf8("background-color: rgb(255, 255,
```

```
255);"))
        self.patologicoQ.setObjectName(_fromUtf8("patologicoQ"))
        self.gridLayout_2.addWidget(self.patologicoQ, 6, 0, 1, 1)
        self.label_24 = QtGui.QLabel(self.frame_4)
        self.label_24.setMinimumSize(QtCore.QSize(90, 0))
        font = QtGui.QFont()
        font.setFamily(_fromUtf8("Leelawadee UI"))
        font.setPointSize(11)
        font.setBold(False)
        font.setWeight(50)
        self.label_24.setFont(font)
        self.label_24.setObjectName(_fromUtf8("label_24"))
        self.gridLayout_2.addWidget(self.label_24, 2, 0, 1, 1)
        self.label_26 = QtGui.QLabel(self.frame_4)
        self.label_26.setMinimumSize(QtCore.QSize(90, 0))
        font = QtGui.QFont()
        font.setFamily(_fromUtf8("Leelawadee UI"))
        font.setPointSize(11)
        font.setBold(False)
        font.setWeight(50)
        self.label_26.setFont(font)
        self.label_26.setObjectName(_fromUtf8("label_26"))
        self.gridLayout_2.addWidget(self.label_26, 5, 0, 1, 1)
        spacerItem3 = QtGui.QSpacerItem(20, 10, QtGui.QSizePolicy.Minimum,
QtGui.QSizePolicy.Fixed)
        self.gridLayout_2.addItem(spacerItem3, 4, 0, 1, 1)
        self.gridLayout_3.addWidget(self.frame_4, 1, 2, 2, 1)
        self.frame_5 = QtGui.QFrame(Registro)
        self.frame_5.setStyleSheet(_fromUtf8("background-color: rgb(247, 247,
247);"))
        self.frame_5.setFrameShape(QtGui.QFrame.Box)
        self.frame_5.setFrameShadow(QtGui.QFrame.Sunken)
        self.frame_5.setObjectName(_fromUtf8("frame_5"))
        self.gridLayout_4 = QtGui.QGridLayout(self.frame_5)
        self.gridLayout_4.setObjectName(_fromUtf8("gridLayout_4"))
        spacerItem4 = QtGui.QSpacerItem(20, 20, QtGui.QSizePolicy.Fixed,
QtGui.QSizePolicy.Minimum)
        self.gridLayout_4.addItem(spacerItem4, 2, 8, 1, 1)
        self.civil = QtGui.QComboBox(self.frame_5)
        font = QtGui.QFont()
        font.setFamily(_fromUtf8("Leelawadee UI"))
        font.setPointSize(11)
        self.civil.setFont(font)
        self.civil.setStyleSheet(_fromUtf8("background-color: rgb(255, 255,
255);"))
        self.civil.setFrame(True)
        self.civil.setObjectName(_fromUtf8("civil"))
        self.civil.addItem(_fromUtf8(""))
        self.civil.setItemText(0, _fromUtf8(""))
        self.civil.addItem(_fromUtf8(""))
        self.civil.addItem(_fromUtf8(""))
        self.civil.addItem(_fromUtf8(""))
        self.civil.addItem(_fromUtf8(""))
        self.gridLayout_4.addWidget(self.civil, 5, 14, 1, 1)
        self.label = QtGui.QLabel(self.frame_5)
        self.label.setMinimumSize(QtCore.QSize(90, 0))
        font = QtGui.QFont()
        font.setFamily(_fromUtf8("Leelawadee UI"))
        font.setPointSize(11)
        font.setBold(False)
        font.setWeight(50)
        self.label.setFont(font)
        self.label.setObjectName(_fromUtf8("label"))
        self.gridLayout_4.addWidget(self.label, 2, 0, 1, 1)
        self.label_3 = QtGui.QLabel(self.frame_5)
        font = QtGui.QFont()
        font.setFamily(_fromUtf8("Leelawadee UI"))
        font.setPointSize(11)
```

```
        font.setBold(False)
        font.setWeight(50)
        self.label_3.setFont(font)

self.label_3.setAlignment(QtCore.Qt.AlignLeading|QtCore.Qt.AlignLeft|QtCore.Qt.Alig
nVCenter)
        self.label_3.setObjectName(_fromUtf8("label_3"))
        self.gridLayout_4.addWidget(self.label_3, 4, 9, 1, 1)
        self.label_20 = QtGui.QLabel(self.frame_5)
        self.label_20.setMinimumSize(QtCore.QSize(90, 0))
        font = QtGui.QFont()
        font.setFamily(_fromUtf8("Leelawadee UI"))
        font.setPointSize(11)
        font.setBold(False)
        font.setWeight(50)
        self.label_20.setFont(font)
        self.label_20.setObjectName(_fromUtf8("label_20"))
        self.gridLayout_4.addWidget(self.label_20, 6, 0, 1, 1)
        self.residencia = QtGui.QLineEdit(self.frame_5)
        sizePolicy = QtGui.QSizePolicy(QtGui.QSizePolicy.Minimum,
QtGui.QSizePolicy.Fixed)
        sizePolicy.setHorizontalStretch(0)
        sizePolicy.setVerticalStretch(0)

sizePolicy.setHeightForWidth(self.residencia.sizePolicy().hasHeightForWidth())
        self.residencia.setSizePolicy(sizePolicy)
        self.residencia.setMaximumSize(QtCore.QSize(16777215, 16777215))
        font = QtGui.QFont()
        font.setFamily(_fromUtf8("Leelawadee UI"))
        font.setPointSize(11)
        self.residencia.setFont(font)
        self.residencia.setStyleSheet(_fromUtf8("background-color: rgb(255, 255,
255);"))
        self.residencia.setObjectName(_fromUtf8("residencia"))
        self.gridLayout_4.addWidget(self.residencia, 6, 11, 1, 4)
        self.label_22 = QtGui.QLabel(self.frame_5)
        font = QtGui.QFont()
        font.setFamily(_fromUtf8("Leelawadee UI"))
        font.setPointSize(11)
        font.setBold(False)
        font.setWeight(50)
        self.label_22.setFont(font)
        self.label_22.setObjectName(_fromUtf8("label_22"))
        self.gridLayout_4.addWidget(self.label_22, 6, 9, 1, 2)
        self.label_18 = QtGui.QLabel(self.frame_5)
        font = QtGui.QFont()
        font.setFamily(_fromUtf8("Leelawadee UI"))
        font.setPointSize(11)
        font.setBold(False)
        font.setWeight(50)
        self.label_18.setFont(font)
        self.label_18.setObjectName(_fromUtf8("label_18"))
        self.gridLayout_4.addWidget(self.label_18, 4, 13, 1, 1)
        self.ocupacion = QtGui.QLineEdit(self.frame_5)
        sizePolicy = QtGui.QSizePolicy(QtGui.QSizePolicy.Minimum,
QtGui.QSizePolicy.Fixed)
        sizePolicy.setHorizontalStretch(0)
        sizePolicy.setVerticalStretch(0)

sizePolicy.setHeightForWidth(self.ocupacion.sizePolicy().hasHeightForWidth())
        self.ocupacion.setSizePolicy(sizePolicy)
        self.ocupacion.setMaximumSize(QtCore.QSize(16777215, 16777215))
        font = QtGui.QFont()
        font.setFamily(_fromUtf8("Leelawadee UI"))
        font.setPointSize(11)
        self.ocupacion.setFont(font)
        self.ocupacion.setStyleSheet(_fromUtf8("background-color: rgb(255, 255,
255);"))
```

```
        self.ocupacion.setObjectName(_fromUtf8("ocupacion"))
        self.gridLayout_4.addWidget(self.ocupacion, 6, 1, 1, 2)
        self.label_2 = QtGui.QLabel(self.frame_5)
        font = QtGui.QFont()
        font.setFamily(_fromUtf8("Leelawadee UI"))
        font.setPointSize(11)
        font.setBold(False)
        font.setWeight(50)
        self.label_2.setFont(font)
        self.label_2.setObjectName(_fromUtf8("label_2"))
        self.gridLayout_4.addWidget(self.label_2, 4, 0, 1, 2)
        spacerItem5 = QtGui.QSpacerItem(20, 20, QtGui.QSizePolicy.Fixed,
QtGui.QSizePolicy.Minimum)
        self.gridLayout_4.addItem(spacerItem5, 4, 4, 1, 1)
        self.rb_masc = QtGui.QRadioButton(self.frame_5)
        font = QtGui.QFont()
        font.setFamily(_fromUtf8("Leelawadee UI"))
        font.setPointSize(11)
        self.rb_masc.setFont(font)
        icon = QtGui.QIcon()
        icon.addPixmap(QtGui.QPixmap(_fromUtf8("ELLOS-Y-ELLAS.jpg")),
QtGui.QIcon.Normal, QtGui.QIcon.Off)
        self.rb_masc.setIcon(icon)
        self.rb_masc.setIconSize(QtCore.QSize(20, 20))
        self.rb_masc.setObjectName(_fromUtf8("rb_masc"))
        self.gridLayout_4.addWidget(self.rb_masc, 4, 11, 1, 1)
        self.IMC = QtGui.QDoubleSpinBox(self.frame_5)
        self.IMC.setMaximumSize(QtCore.QSize(16777215, 16777215))
        font = QtGui.QFont()
        font.setFamily(_fromUtf8("Leelawadee UI"))
        font.setPointSize(11)
        self.IMC.setFont(font)
        self.IMC.setStyleSheet(_fromUtf8("background-color: rgb(255, 255, 255);"))
        self.IMC.setReadOnly(True)
        self.IMC.setButtonSymbols(QtGui.QAbstractSpinBox.NoButtons)
        self.IMC.setMaximum(150.0)
        self.IMC.setObjectName(_fromUtf8("IMC"))
        self.gridLayout_4.addWidget(self.IMC, 6, 6, 1, 1)
        self.label_14 = QtGui.QLabel(self.frame_5)
        self.label_14.setMinimumSize(QtCore.QSize(30, 0))
        font = QtGui.QFont()
        font.setFamily(_fromUtf8("Leelawadee UI"))
        font.setPointSize(11)
        font.setBold(False)
        font.setWeight(50)
        self.label_14.setFont(font)
        self.label_14.setStyleSheet(_fromUtf8(""))

self.label_14.setAlignment(QtCore.Qt.AlignLeading|QtCore.Qt.AlignLeft|QtCore.Qt.Ali
gnVCenter)
        self.label_14.setWordWrap(True)
        self.label_14.setObjectName(_fromUtf8("label_14"))
        self.gridLayout_4.addWidget(self.label_14, 6, 5, 1, 1)
        spacerItem6 = QtGui.QSpacerItem(20, 20, QtGui.QSizePolicy.Fixed,
QtGui.QSizePolicy.Minimum)
        self.gridLayout_4.addItem(spacerItem6, 2, 12, 1, 1)
        self.ingreso = QtGui.QDateEdit(self.frame_5)
        self.ingreso.setMaximumSize(QtCore.QSize(90, 16777215))
        font = QtGui.QFont()
        font.setFamily(_fromUtf8("Leelawadee UI"))
        font.setPointSize(11)
        self.ingreso.setFont(font)
        self.ingreso.setStyleSheet(_fromUtf8("background-color: rgb(255, 255,
255);"))
        self.ingreso.setDate(QtCore.QDate(1753, 1, 1))
        self.ingreso.setObjectName(_fromUtf8("ingreso"))
        self.gridLayout_4.addWidget(self.ingreso, 5, 2, 1, 1)
        self.rb_fem = QtGui.QRadioButton(self.frame_5)
```

```
        font = QtGui.QFont()
        font.setFamily(_fromUtf8("Leelawadee UI"))
        font.setPointSize(11)
        self.rb_fem.setFont(font)
        icon1 = QtGui.QIcon()
        icon1.addPixmap(QtGui.QPixmap(_fromUtf8("ELLAS.png")), QtGui.QIcon.Normal,
QtGui.QIcon.Off)
        self.rb_fem.setIcon(icon1)
        self.rb_fem.setIconSize(QtCore.QSize(20, 20))
        self.rb_fem.setChecked(True)
        self.rb_fem.setObjectName(_fromUtf8("rb_fem"))
        self.gridLayout_4.addWidget(self.rb_fem, 4, 10, 1, 1)
        self.nacimiento = QtGui.QDateEdit(self.frame_5)
        self.nacimiento.setMaximumSize(QtCore.QSize(90, 16777215))
        font = QtGui.QFont()
        font.setFamily(_fromUtf8("Leelawadee UI"))
        font.setPointSize(11)
        self.nacimiento.setFont(font)
        self.nacimiento.setStyleSheet(_fromUtf8("background-color: rgb(255, 255,
255);"))
        self.nacimiento.setDate(QtCore.QDate(1753, 1, 1))
        self.nacimiento.setObjectName(_fromUtf8("nacimiento"))
        self.gridLayout_4.addWidget(self.nacimiento, 4, 2, 1, 2)
        self.label_8 = QtGui.QLabel(self.frame_5)
        font = QtGui.QFont()
        font.setFamily(_fromUtf8("Leelawadee UI"))
        font.setPointSize(11)
        font.setBold(False)
        font.setWeight(50)
        self.label_8.setFont(font)
        self.label_8.setObjectName(_fromUtf8("label_8"))
        self.gridLayout_4.addWidget(self.label_8, 2, 9, 1, 1)
        self.label_21 = QtGui.QLabel(self.frame_5)
        font = QtGui.QFont()
        font.setFamily(_fromUtf8("Leelawadee UI"))
        font.setPointSize(11)
        font.setBold(False)
        font.setWeight(50)
        self.label_21.setFont(font)
        self.label_21.setObjectName(_fromUtf8("label_21"))
        self.gridLayout_4.addWidget(self.label_21, 5, 9, 1, 1)
        self.nacionalidad = QtGui.QLineEdit(self.frame_5)
        font = QtGui.QFont()
        font.setFamily(_fromUtf8("Leelawadee UI"))
        font.setPointSize(11)
        self.nacionalidad.setFont(font)
        self.nacionalidad.setStyleSheet(_fromUtf8("background-color: rgb(255, 255,
255);"))
        self.nacionalidad.setObjectName(_fromUtf8("nacionalidad"))
        self.gridLayout_4.addWidget(self.nacionalidad, 4, 14, 1, 1)
        self.peso = QtGui.QDoubleSpinBox(self.frame_5)
        self.peso.setMinimumSize(QtCore.QSize(80, 0))
        self.peso.setMaximumSize(QtCore.QSize(100, 16777215))
        font = QtGui.QFont()
        font.setFamily(_fromUtf8("Leelawadee UI"))
        font.setPointSize(11)
        self.peso.setFont(font)
        self.peso.setStyleSheet(_fromUtf8("background-color: rgb(255, 255, 255);"))
        self.peso.setButtonSymbols(QtGui.QAbstractSpinBox.NoButtons)
        self.peso.setMaximum(150.0)
        self.peso.setObjectName(_fromUtf8("peso"))
        self.gridLayout_4.addWidget(self.peso, 4, 6, 1, 1)
        self.label_10 = QtGui.QLabel(self.frame_5)
        font = QtGui.QFont()
        font.setFamily(_fromUtf8("Leelawadee UI"))
        font.setPointSize(11)
        font.setBold(False)
        font.setWeight(50)
```

```
        self.label_10.setFont(font)
        self.label_10.setObjectName(_fromUtf8("label_10"))
        self.gridLayout_4.addWidget(self.label_10, 2, 13, 1, 1)
        self.label_19 = QtGui.QLabel(self.frame_5)
        font = QtGui.QFont()
        font.setFamily(_fromUtf8("Leelawadee UI"))
        font.setPointSize(11)
        font.setBold(False)
        font.setWeight(50)
        self.label_19.setFont(font)
        self.label_19.setObjectName(_fromUtf8("label_19"))
        self.gridLayout_4.addWidget(self.label_19, 5, 13, 1, 1)
        self.talla = QtGui.QDoubleSpinBox(self.frame_5)
        self.talla.setMinimumSize(QtCore.QSize(80, 0))
        self.talla.setMaximumSize(QtCore.QSize(100, 16777215))
        font = QtGui.QFont()
        font.setFamily(_fromUtf8("Leelawadee UI"))
        font.setPointSize(11)
        self.talla.setFont(font)
        self.talla.setStyleSheet(_fromUtf8("background-color: rgb(255, 255,
255);"))
        self.talla.setButtonSymbols(QtGui.QAbstractSpinBox.NoButtons)
        self.talla.setMaximum(150.0)
        self.talla.setObjectName(_fromUtf8("talla"))
        self.gridLayout_4.addWidget(self.talla, 5, 6, 1, 1)
        self.nombre = QtGui.QLineEdit(self.frame_5)
        sizePolicy = QtGui.QSizePolicy(QtGui.QSizePolicy.Minimum,
QtGui.QSizePolicy.Fixed)
        sizePolicy.setHorizontalStretch(0)
        sizePolicy.setVerticalStretch(0)
        sizePolicy.setHeightForWidth(self.nombre.sizePolicy().hasHeightForWidth())
        self.nombre.setSizePolicy(sizePolicy)
        font = QtGui.QFont()
        font.setFamily(_fromUtf8("Leelawadee UI"))
        font.setPointSize(11)
        self.nombre.setFont(font)
        self.nombre.setStyleSheet(_fromUtf8("background-color: rgb(255, 255,
255);"))
        self.nombre.setObjectName(_fromUtf8("nombre"))
        self.gridLayout_4.addWidget(self.nombre, 2, 1, 1, 7)
        self.procedencia = QtGui.QLineEdit(self.frame_5)
        sizePolicy = QtGui.QSizePolicy(QtGui.QSizePolicy.Minimum,
QtGui.QSizePolicy.Fixed)
        sizePolicy.setHorizontalStretch(0)
        sizePolicy.setVerticalStretch(0)

sizePolicy.setHeightForWidth(self.procedencia.sizePolicy().hasHeightForWidth())
        self.procedencia.setSizePolicy(sizePolicy)
        self.procedencia.setMaximumSize(QtCore.QSize(16777215, 16777215))
        font = QtGui.QFont()
        font.setFamily(_fromUtf8("Leelawadee UI"))
        font.setPointSize(11)
        self.procedencia.setFont(font)
        self.procedencia.setStyleSheet(_fromUtf8("background-color: rgb(255, 255,
255);"))
        self.procedencia.setObjectName(_fromUtf8("procedencia"))
        self.gridLayout_4.addWidget(self.procedencia, 5, 10, 1, 2)
        self.label_13 = QtGui.QLabel(self.frame_5)
        font = QtGui.QFont()
        font.setFamily(_fromUtf8("Leelawadee UI"))
        font.setPointSize(11)
        font.setBold(False)
        font.setWeight(50)
        self.label_13.setFont(font)
        self.label_13.setStyleSheet(_fromUtf8(""))

self.label_13.setAlignment(QtCore.Qt.AlignLeading|QtCore.Qt.AlignLeft|QtCore.Qt.Ali
gnVCenter)
```

```
        self.label_13.setWordWrap(True)
        self.label_13.setObjectName(_fromUtf8("label_13"))
        self.gridLayout_4.addWidget(self.label_13, 5, 7, 1, 1)
        self.label_11 = QtGui.QLabel(self.frame_5)
        self.label_11.setMinimumSize(QtCore.QSize(30, 0))
        font = QtGui.QFont()
        font.setFamily(_fromUtf8("Leelawadee UI"))
        font.setPointSize(11)
        font.setBold(False)
        font.setWeight(50)
        self.label_11.setFont(font)
        self.label_11.setStyleSheet(_fromUtf8(""))

self.label_11.setAlignment(QtCore.Qt.AlignLeading|QtCore.Qt.AlignLeft|QtCore.Qt.Ali
gnVCenter)
        self.label_11.setWordWrap(True)
        self.label_11.setObjectName(_fromUtf8("label_11"))
        self.gridLayout_4.addWidget(self.label_11, 5, 5, 1, 1)
        self.label_12 = QtGui.QLabel(self.frame_5)
        font = QtGui.QFont()
        font.setFamily(_fromUtf8("Leelawadee UI"))
        font.setPointSize(11)
        font.setBold(False)
        font.setWeight(50)
        self.label_12.setFont(font)
        self.label_12.setStyleSheet(_fromUtf8(""))

self.label_12.setAlignment(QtCore.Qt.AlignLeading|QtCore.Qt.AlignLeft|QtCore.Qt.Ali
gnVCenter)
        self.label_12.setWordWrap(True)
        self.label_12.setObjectName(_fromUtf8("label_12"))
        self.gridLayout_4.addWidget(self.label_12, 4, 7, 1, 1)
        self.label_7 = QtGui.QLabel(self.frame_5)
        self.label_7.setMinimumSize(QtCore.QSize(30, 0))
        font = QtGui.QFont()
        font.setFamily(_fromUtf8("Leelawadee UI"))
        font.setPointSize(11)
        font.setBold(False)
        font.setWeight(50)
        self.label_7.setFont(font)
        self.label_7.setStyleSheet(_fromUtf8(""))

self.label_7.setAlignment(QtCore.Qt.AlignLeading|QtCore.Qt.AlignLeft|QtCore.Qt.Alig
nVCenter)
        self.label_7.setWordWrap(True)
        self.label_7.setObjectName(_fromUtf8("label_7"))
        self.gridLayout_4.addWidget(self.label_7, 4, 5, 1, 1)
        self.CI = QtGui.QLineEdit(self.frame_5)
        font = QtGui.QFont()
        font.setFamily(_fromUtf8("Leelawadee UI"))
        font.setPointSize(11)
        self.CI.setFont(font)
        self.CI.setStyleSheet(_fromUtf8("background-color: rgb(255, 255, 255);"))
        self.CI.setObjectName(_fromUtf8("CI"))
        self.gridLayout_4.addWidget(self.CI, 2, 10, 1, 2)
        self.label_23 = QtGui.QLabel(self.frame_5)
        font = QtGui.QFont()
        font.setFamily(_fromUtf8("Leelawadee UI"))
        font.setPointSize(11)
        font.setBold(False)
        font.setWeight(50)
        self.label_23.setFont(font)
        self.label_23.setObjectName(_fromUtf8("label_23"))
        self.gridLayout_4.addWidget(self.label_23, 5, 0, 1, 2)
        spacerItem7 = QtGui.QSpacerItem(20, 10, QtGui.QSizePolicy.Minimum,
QtGui.QSizePolicy.Fixed)
        self.gridLayout_4.addItem(spacerItem7, 1, 5, 1, 1)
        self.extremidad_2 = QtGui.QLabel(self.frame_5)
```

```
        self.extremidad_2.setMinimumSize(QtCore.QSize(0, 30))
        self.extremidad_2.setMaximumSize(QtCore.QSize(16777215, 30))
        font = QtGui.QFont()
        font.setFamily(_fromUtf8("Leelawadee UI"))
        font.setPointSize(11)
        font.setBold(True)
        font.setWeight(75)
        self.extremidad_2.setFont(font)
        self.extremidad_2.setStyleSheet(_fromUtf8("background-color: rgb(216, 213,
255)"))
        self.extremidad_2.setFrameShape(QtGui.QFrame.Box)
        self.extremidad_2.setFrameShadow(QtGui.QFrame.Sunken)
        self.extremidad_2.setAlignment(QtCore.Qt.AlignCenter)
        self.extremidad_2.setObjectName(_fromUtf8("extremidad_2"))
        self.gridLayout_4.addWidget(self.extremidad_2, 0, 0, 1, 16)
        self.HC = QtGui.QSpinBox(self.frame_5)
        self.HC.setMinimumSize(QtCore.QSize(80, 0))
        self.HC.setMaximumSize(QtCore.QSize(16777215, 16777215))
        font = QtGui.QFont()
        font.setFamily(_fromUtf8("Leelawadee UI"))
        font.setPointSize(11)
        self.HC.setFont(font)
        self.HC.setStyleSheet(_fromUtf8("background-color: rgb(255, 255, 255);"))
        self.HC.setButtonSymbols(QtGui.QAbstractSpinBox.NoButtons)
        self.HC.setMaximum(999999999)
        self.HC.setObjectName(_fromUtf8("HC"))
        self.gridLayout_4.addWidget(self.HC, 2, 14, 1, 1)
        self.gridLayout_3.addWidget(self.frame_5, 0, 1, 1, 3)
        self.frame = QtGui.QFrame(Registro)
        self.frame.setMaximumSize(QtCore.QSize(16777215, 50))
        self.frame.setFrameShape(QtGui.QFrame.StyledPanel)
        self.frame.setFrameShadow(QtGui.QFrame.Raised)
        self.frame.setObjectName(_fromUtf8("frame"))
        self.horizontalLayout = QtGui.QHBoxLayout(self.frame)
        self.horizontalLayout.setObjectName(_fromUtf8("horizontalLayout"))
        spacerItem8 = QtGui.QSpacerItem(40, 20, QtGui.QSizePolicy.Expanding,
QtGui.QSizePolicy.Minimum)
        self.horizontalLayout.addItem(spacerItem8)
        self.btn_cancelar = QtGui.QPushButton(self.frame)
        self.btn_cancelar.setObjectName(_fromUtf8("btn_cancelar"))
        self.horizontalLayout.addWidget(self.btn_cancelar)
        self.btn_aceptar = QtGui.QPushButton(self.frame)
        self.btn_aceptar.setObjectName(_fromUtf8("btn_aceptar"))
        self.horizontalLayout.addWidget(self.btn_aceptar)
        self.gridLayout_3.addWidget(self.frame, 2, 3, 1, 1)
        self.sw_extremidad = QtGui.QFrame(Registro)
        self.sw_extremidad.setMinimumSize(QtCore.QSize(300, 0))
        self.sw_extremidad.setMaximumSize(QtCore.QSize(400, 16777215))
        self.sw_extremidad.setStyleSheet(_fromUtf8("background-color: rgb(247, 247,
247);"))
        self.sw_extremidad.setFrameShape(QtGui.QFrame.Box)
        self.sw_extremidad.setFrameShadow(QtGui.QFrame.Sunken)
        self.sw_extremidad.setObjectName(_fromUtf8("sw_extremidad"))
        self.gridLayout = QtGui.QGridLayout(self.sw_extremidad)
        self.gridLayout.setObjectName(_fromUtf8("gridLayout"))
        self.label_6 = QtGui.QLabel(self.sw_extremidad)
        self.label_6.setMinimumSize(QtCore.QSize(0, 30))
        self.label_6.setMaximumSize(QtCore.QSize(16777215, 30))
        font = QtGui.QFont()
        font.setFamily(_fromUtf8("Leelawadee UI"))
        font.setPointSize(11)
        font.setBold(True)
        font.setWeight(75)
        self.label_6.setFont(font)
        self.label_6.setStyleSheet(_fromUtf8("background-color: rgb(216, 213,
255)"))
        self.label_6.setFrameShape(QtGui.QFrame.Box)
        self.label_6.setFrameShadow(QtGui.QFrame.Sunken)
```

```
        self.label_6.setAlignment(QtCore.Qt.AlignCenter)
        self.label_6.setObjectName(_fromUtf8("label_6"))
        self.gridLayout.addWidget(self.label_6, 0, 0, 1, 3)
        spacerItem9 = QtGui.QSpacerItem(20, 10, QtGui.QSizePolicy.Minimum,
QtGui.QSizePolicy.Fixed)
        self.gridLayout.addItem(spacerItem9, 1, 1, 1, 1)
        self.label_9 = QtGui.QLabel(self.sw_extremidad)
        font = QtGui.QFont()
        font.setFamily(_fromUtf8("Leelawadee UI"))
        font.setPointSize(11)
        font.setBold(False)
        font.setWeight(50)
        self.label_9.setFont(font)
        self.label_9.setObjectName(_fromUtf8("label_9"))
        self.gridLayout.addWidget(self.label_9, 2, 0, 1, 1)
        self.extremidad = QtGui.QComboBox(self.sw_extremidad)
        self.extremidad.setMaximumSize(QtCore.QSize(180, 16777215))
        font = QtGui.QFont()
        font.setFamily(_fromUtf8("Leelawadee UI"))
        font.setPointSize(11)
        self.extremidad.setFont(font)
        self.extremidad.setStyleSheet(_fromUtf8("background-color: rgb(255, 255,
255);"))
        self.extremidad.setFrame(True)
        self.extremidad.setObjectName(_fromUtf8("extremidad"))
        self.extremidad.addItem(_fromUtf8(""))
        self.extremidad.setItemText(0, _fromUtf8(""))
        self.extremidad.addItem(_fromUtf8(""))
        self.extremidad.addItem(_fromUtf8(""))
        self.gridLayout.addWidget(self.extremidad, 2, 2, 1, 1)
        self.label_4 = QtGui.QLabel(self.sw_extremidad)
        font = QtGui.QFont()
        font.setFamily(_fromUtf8("Leelawadee UI"))
        font.setPointSize(11)
        font.setBold(False)
        font.setWeight(50)
        self.label_4.setFont(font)
        self.label_4.setObjectName(_fromUtf8("label_4"))
        self.gridLayout.addWidget(self.label_4, 3, 0, 1, 2)
        self.amputacion = QtGui.QComboBox(self.sw_extremidad)
        self.amputacion.setMaximumSize(QtCore.QSize(180, 16777215))
        font = QtGui.QFont()
        font.setFamily(_fromUtf8("Leelawadee UI"))
        font.setPointSize(11)
        self.amputacion.setFont(font)
        self.amputacion.setStyleSheet(_fromUtf8("background-color: rgb(255, 255,
255);"))
        self.amputacion.setFrame(True)
        self.amputacion.setObjectName(_fromUtf8("amputacion"))
        self.amputacion.addItem(_fromUtf8(""))
        self.amputacion.setItemText(0, _fromUtf8(""))
        self.amputacion.addItem(_fromUtf8(""))
        self.gridLayout.addWidget(self.amputacion, 3, 2, 1, 1)
        self.label_5 = QtGui.QLabel(self.sw_extremidad)
        font = QtGui.QFont()
        font.setFamily(_fromUtf8("Leelawadee UI"))
        font.setPointSize(11)
        font.setBold(False)
        font.setWeight(50)
        self.label_5.setFont(font)
        self.label_5.setObjectName(_fromUtf8("label_5"))
        self.gridLayout.addWidget(self.label_5, 4, 0, 1, 1)
        self.causa = QtGui.QComboBox(self.sw_extremidad)
        self.causa.setMaximumSize(QtCore.QSize(180, 16777215))
        font = QtGui.QFont()
        font.setFamily(_fromUtf8("Leelawadee UI"))
        font.setPointSize(11)
        self.causa.setFont(font)
```

```
        self.causa.setStyleSheet(_fromUtf8("background-color: rgb(255, 255,
255);"))
        self.causa.setFrame(True)
        self.causa.setObjectName(_fromUtf8("causa"))
        self.causa.addItem(_fromUtf8(""))
        self.causa.setItemText(0, _fromUtf8(""))
        self.causa.addItem(_fromUtf8(""))
        self.causa.addItem(_fromUtf8(""))
        self.causa.addItem(_fromUtf8(""))
        self.causa.addItem(_fromUtf8(""))
        self.gridLayout.addWidget(self.causa, 4, 2, 1, 1)
        self.label_15 = QtGui.QLabel(self.sw_extremidad)
        font = QtGui.QFont()
        font.setFamily(_fromUtf8("Leelawadee UI"))
        font.setPointSize(11)
        font.setBold(False)
        font.setWeight(50)
        self.label_15.setFont(font)
        self.label_15.setObjectName(_fromUtf8("label_15"))
        self.gridLayout.addWidget(self.label_15, 5, 0, 1, 2)
        self.fecha_a = QtGui.QDateEdit(self.sw_extremidad)
        self.fecha_a.setMaximumSize(QtCore.QSize(180, 16777215))
        font = QtGui.QFont()
        font.setFamily(_fromUtf8("Leelawadee UI"))
        font.setPointSize(11)
        self.fecha_a.setFont(font)
        self.fecha_a.setStyleSheet(_fromUtf8("background-color: rgb(255, 255,
255);"))
        self.fecha_a.setDate(QtCore.QDate(1753, 1, 1))
        self.fecha_a.setObjectName(_fromUtf8("fecha_a"))
        self.gridLayout.addWidget(self.fecha_a, 5, 2, 1, 1)
        self.label_16 = QtGui.QLabel(self.sw_extremidad)
        self.label_16.setMinimumSize(QtCore.QSize(150, 0))
        font = QtGui.QFont()
        font.setFamily(_fromUtf8("Leelawadee UI"))
        font.setPointSize(11)
        font.setBold(False)
        font.setWeight(50)
        self.label_16.setFont(font)
        self.label_16.setObjectName(_fromUtf8("label_16"))
        self.gridLayout.addWidget(self.label_16, 6, 0, 1, 2)
        self.frame_2 = QtGui.QFrame(self.sw_extremidad)
        self.frame_2.setMaximumSize(QtCore.QSize(180, 30))
        self.frame_2.setFrameShape(QtGui.QFrame.StyledPanel)
        self.frame_2.setFrameShadow(QtGui.QFrame.Raised)
        self.frame_2.setObjectName(_fromUtf8("frame_2"))
        self.horizontalLayout_2 = QtGui.QHBoxLayout(self.frame_2)
        self.horizontalLayout_2.setMargin(0)
        self.horizontalLayout_2.setObjectName(_fromUtf8("horizontalLayout_2"))
        self.T_si = QtGui.QRadioButton(self.frame_2)
        font = QtGui.QFont()
        font.setFamily(_fromUtf8("Leelawadee UI"))
        font.setPointSize(11)
        self.T_si.setFont(font)
        self.T_si.setIconSize(QtCore.QSize(20, 20))
        self.T_si.setChecked(False)
        self.T_si.setObjectName(_fromUtf8("T_si"))
        self.horizontalLayout_2.addWidget(self.T_si)
        self.T_no = QtGui.QRadioButton(self.frame_2)
        font = QtGui.QFont()
        font.setFamily(_fromUtf8("Leelawadee UI"))
        font.setPointSize(11)
        self.T_no.setFont(font)
        self.T_no.setIconSize(QtCore.QSize(20, 20))
        self.T_no.setChecked(True)
        self.T_no.setObjectName(_fromUtf8("T_no"))
        self.horizontalLayout_2.addWidget(self.T_no)
        self.gridLayout.addWidget(self.frame_2, 6, 2, 1, 1)
```

```
        self.gridLayout_3.addWidget(self.sw_extremidad, 1, 3, 1, 1)

        self.retranslateUi(Registro)
        QtCore.QMetaObject.connectSlotsByName(Registro)

    def retranslateUi(self, Registro):
        Registro.setWindowTitle(_translate("Registro", "REGISTRO", None))
        self.radiografia.setText(_translate("Registro", "Radiografía", None))
        self.resonancia.setText(_translate("Registro", "Resonancia Magnética",
None))
        self.extremidad_5.setText(_translate("Registro", "ACTUAL", None))
        self.electromiografia.setText(_translate("Registro", "Electromiografía",
None))
        self.label_17.setText(_translate("Registro", "Enfermedad Actual:", None))
        self.ecografia.setText(_translate("Registro", "Ecografía", None))
        self.TAC.setText(_translate("Registro", "TAC", None))
        self.label_25.setText(_translate("Registro", "Medios Diagnósticos:", None))
        self.extremidad_3.setText(_translate("Registro", "ANTECEDENTES", None))
        self.label_24.setText(_translate("Registro", "Antecedentes Patológicos
Personales:", None))
        self.label_26.setText(_translate("Registro", "Antecedentes Patológicos
Quirurgicos:", None))
        self.civil.setItemText(1, _translate("Registro", "Casado(a)", None))
        self.civil.setItemText(2, _translate("Registro", "Soltero(a)", None))
        self.civil.setItemText(3, _translate("Registro", "Divorciado(a)", None))
        self.civil.setItemText(4, _translate("Registro", "Viudo(a)", None))
        self.label.setText(_translate("Registro", "Nombre:", None))
        self.label_3.setText(_translate("Registro", "Sexo:", None))
        self.label_20.setText(_translate("Registro", "Ocupación:", None))
        self.label_22.setText(_translate("Registro", "Residencia Actual:", None))
        self.label_18.setText(_translate("Registro", "Nacionalidad:", None))
        self.label_2.setText(_translate("Registro", "Fecha de nacimiento:", None))
        self.rb_masc.setText(_translate("Registro", "M", None))
        self.label_14.setText(_translate("Registro", "IMC:", None))
        self.ingreso.setDisplayFormat(_translate("Registro", "d/M/yyyy", None))
        self.rb_fem.setText(_translate("Registro", "F", None))
        self.nacimiento.setDisplayFormat(_translate("Registro", "d/M/yyyy", None))
        self.label_8.setText(_translate("Registro", "CI: ", None))
        self.label_21.setText(_translate("Registro", "Procedencia:", None))
        self.label_10.setText(_translate("Registro", "# de Historia Clínica:",
None))
        self.label_19.setText(_translate("Registro", "Estado Civil:", None))
        self.label_13.setText(_translate("Registro", "m", None))
        self.label_11.setText(_translate("Registro", "Talla:", None))
        self.label_12.setText(_translate("Registro", "Kg", None))
        self.label_7.setText(_translate("Registro", "Peso:", None))
        self.label_23.setText(_translate("Registro", "Fecha de ingreso:", None))
        self.extremidad_2.setText(_translate("Registro", "DATOS GENERALES DEL
PACIENTE", None))
        self.btn_cancelar.setText(_translate("Registro", "Cancelar", None))
        self.btn_aceptar.setText(_translate("Registro", "Aceptar", None))
        self.label_6.setText(_translate("Registro", "EXTREMIDAD AMPUTADA", None))
        self.label_9.setText(_translate("Registro", "Extremidad:", None))
        self.extremidad.setItemText(1, _translate("Registro", "Pierna derecha",
None))
        self.extremidad.setItemText(2, _translate("Registro", "Pierna izquierda",
None))
        self.label_4.setText(_translate("Registro", "Amputación:", None))
        self.amputacion.setItemText(1, _translate("Registro", "Transtibial", None))
        self.label_5.setText(_translate("Registro", "Causa: ", None))
        self.causa.setItemText(1, _translate("Registro", "Accidente laboral",
None))
        self.causa.setItemText(2, _translate("Registro", "Accidente de transito",
None))
        self.causa.setItemText(3, _translate("Registro", "Enfermedad congenita",
None))
        self.causa.setItemText(4, _translate("Registro", "Enfermedad adquirida",
None))
```

```
        self.label_15.setText(_translate("Registro", "Fecha de  amputación:",
None))
        self.fecha_a.setDisplayFormat(_translate("Registro", "d/M/yyyy", None))
        self.label_16.setText(_translate("Registro", "Tratamiento Posoperatorio:",
None))
        self.T_si.setText(_translate("Registro", "Si", None))
        self.T_no.setText(_translate("Registro", "No", None))
```

3) Código de la pantalla de Evaluación

```
# -*- coding: utf-8 -*-

# Form implementation generated from reading ui file 'evaluacion.ui'
#
# Created by: PyQt4 UI code generator 4.11.4
#
# WARNING! All changes made in this file will be lost!

from PyQt4 import QtCore, QtGui

try:
    _fromUtf8 = QtCore.QString.fromUtf8
except AttributeError:
    def _fromUtf8(s):
        return s

try:
    _encoding = QtGui.QApplication.UnicodeUTF8
    def _translate(context, text, disambig):
        return QtGui.QApplication.translate(context, text, disambig, _encoding)
except AttributeError:
    def _translate(context, text, disambig):
        return QtGui.QApplication.translate(context, text, disambig)

class Ui_Evaluacion(object):
    def setupUi(self, Evaluacion):
        Evaluacion.setObjectName(_fromUtf8("Evaluacion"))
        Evaluacion.resize(1329, 788)
        font = QtGui.QFont()
        font.setFamily(_fromUtf8("Leelawadee UI"))
        font.setPointSize(11)
        Evaluacion.setFont(font)
        self.centralwidget = QtGui.QWidget(Evaluacion)
        self.centralwidget.setObjectName(_fromUtf8("centralwidget"))
        self.gridLayout_5 = QtGui.QGridLayout(self.centralwidget)
        self.gridLayout_5.setObjectName(_fromUtf8("gridLayout_5"))
        self.tab_Eval_Inicial = QtGui.QTabWidget(self.centralwidget)
        self.tab_Eval_Inicial.setEnabled(True)
        self.tab_Eval_Inicial.setAutoFillBackground(False)
        self.tab_Eval_Inicial.setElideMode(QtCore.Qt.ElideNone)
        self.tab_Eval_Inicial.setUsesScrollButtons(True)
        self.tab_Eval_Inicial.setObjectName(_fromUtf8("tab_Eval_Inicial"))
        self.extension = QtGui.QWidget()
        self.extension.setObjectName(_fromUtf8("extension"))
        self.gridLayout_4 = QtGui.QGridLayout(self.extension)
        self.gridLayout_4.setContentsMargins(6, 6, 6, 0)
        self.gridLayout_4.setObjectName(_fromUtf8("gridLayout_4"))
        self.btn_guardar_E = QtGui.QPushButton(self.extension)
        icon = QtGui.QIcon()
        icon.addPixmap(QtGui.QPixmap(_fromUtf8("guardar.jpg")), QtGui.QIcon.Normal,
QtGui.QIcon.Off)
```

```
        self.btn_guardar_E.setIcon(icon)
        self.btn_guardar_E.setIconSize(QtCore.QSize(25, 25))
        self.btn_guardar_E.setObjectName(_fromUtf8("btn_guardar_E"))
        self.gridLayout_4.addWidget(self.btn_guardar_E, 0, 3, 1, 1)
        spacerItem = QtGui.QSpacerItem(40, 20, QtGui.QSizePolicy.Expanding,
QtGui.QSizePolicy.Minimum)
        self.gridLayout_4.addItem(spacerItem, 0, 0, 1, 1)
        self.btn_parar_E = QtGui.QPushButton(self.extension)
        icon1 = QtGui.QIcon()
        icon1.addPixmap(QtGui.QPixmap(_fromUtf8("stop.png")), QtGui.QIcon.Normal,
QtGui.QIcon.Off)
        self.btn_parar_E.setIcon(icon1)
        self.btn_parar_E.setIconSize(QtCore.QSize(25, 25))
        self.btn_parar_E.setObjectName(_fromUtf8("btn_parar_E"))
        self.gridLayout_4.addWidget(self.btn_parar_E, 0, 2, 1, 1)
        spacerItem1 = QtGui.QSpacerItem(0, 0, QtGui.QSizePolicy.Minimum,
QtGui.QSizePolicy.Fixed)
        self.gridLayout_4.addItem(spacerItem1, 2, 1, 1, 1)
        self.btn_Inicio_E = QtGui.QPushButton(self.extension)
        icon2 = QtGui.QIcon()
        icon2.addPixmap(QtGui.QPixmap(_fromUtf8("play.png")), QtGui.QIcon.Normal,
QtGui.QIcon.Off)
        self.btn_Inicio_E.setIcon(icon2)
        self.btn_Inicio_E.setIconSize(QtCore.QSize(25, 25))
        self.btn_Inicio_E.setObjectName(_fromUtf8("btn_Inicio_E"))
        self.gridLayout_4.addWidget(self.btn_Inicio_E, 0, 1, 1, 1)
        self.grafica_E = QtGui.QVBoxLayout()
        self.grafica_E.setObjectName(_fromUtf8("grafica_E"))
        self.gridLayout_4.addLayout(self.grafica_E, 1, 0, 1, 4)
        self.tab_Eval_Inicial.addTab(self.extension, _fromUtf8(""))
        self.flexion = QtGui.QWidget()
        self.flexion.setObjectName(_fromUtf8("flexion"))
        self.gridLayout = QtGui.QGridLayout(self.flexion)
        self.gridLayout.setContentsMargins(6, 6, 6, 0)
        self.gridLayout.setObjectName(_fromUtf8("gridLayout"))
        self.btn_guardar_F = QtGui.QPushButton(self.flexion)
        self.btn_guardar_F.setIcon(icon)
        self.btn_guardar_F.setIconSize(QtCore.QSize(25, 25))
        self.btn_guardar_F.setObjectName(_fromUtf8("btn_guardar_F"))
        self.gridLayout.addWidget(self.btn_guardar_F, 0, 3, 1, 1)
        self.btn_inicio_F = QtGui.QPushButton(self.flexion)
        self.btn_inicio_F.setIcon(icon2)
        self.btn_inicio_F.setIconSize(QtCore.QSize(25, 25))
        self.btn_inicio_F.setObjectName(_fromUtf8("btn_inicio_F"))
        self.gridLayout.addWidget(self.btn_inicio_F, 0, 1, 1, 1)
        self.btn_parar_F = QtGui.QPushButton(self.flexion)
        self.btn_parar_F.setIcon(icon1)
        self.btn_parar_F.setIconSize(QtCore.QSize(25, 25))
        self.btn_parar_F.setObjectName(_fromUtf8("btn_parar_F"))
        self.gridLayout.addWidget(self.btn_parar_F, 0, 2, 1, 1)
        spacerItem2 = QtGui.QSpacerItem(40, 20, QtGui.QSizePolicy.Expanding,
QtGui.QSizePolicy.Minimum)
        self.gridLayout.addItem(spacerItem2, 0, 0, 1, 1)
        self.grafica_F = QtGui.QVBoxLayout()
        self.grafica_F.setObjectName(_fromUtf8("grafica_F"))
        self.gridLayout.addLayout(self.grafica_F, 1, 0, 1, 4)
        self.tab_Eval_Inicial.addTab(self.flexion, _fromUtf8(""))
        self.gridLayout_5.addWidget(self.tab_Eval_Inicial, 1, 2, 1, 1)
        self.frame_4 = QtGui.QFrame(self.centralwidget)
        sizePolicy = QtGui.QSizePolicy(QtGui.QSizePolicy.Minimum,
QtGui.QSizePolicy.Preferred)
        sizePolicy.setHorizontalStretch(0)
        sizePolicy.setVerticalStretch(0)
        sizePolicy.setHeightForWidth(self.frame_4.sizePolicy().hasHeightForWidth())
        self.frame_4.setSizePolicy(sizePolicy)
        self.frame_4.setMinimumSize(QtCore.QSize(400, 0))
        self.frame_4.setMaximumSize(QtCore.QSize(450, 16777215))
        self.frame_4.setStyleSheet(_fromUtf8("background-color: rgb(216, 213,
```

```
255);"))
        self.frame_4.setFrameShape(QtGui.QFrame.Panel)
        self.frame_4.setFrameShadow(QtGui.QFrame.Sunken)
        self.frame_4.setObjectName(_fromUtf8("frame_4"))
        self.gridLayout_2 = QtGui.QGridLayout(self.frame_4)
        self.gridLayout_2.setContentsMargins(10, -1, 10, -1)
        self.gridLayout_2.setObjectName(_fromUtf8("gridLayout_2"))
        spacerItem3 = QtGui.QSpacerItem(20, 10, QtGui.QSizePolicy.Minimum,
QtGui.QSizePolicy.Fixed)
        self.gridLayout_2.addItem(spacerItem3, 2, 0, 1, 1)
        spacerItem4 = QtGui.QSpacerItem(5, 20, QtGui.QSizePolicy.Expanding,
QtGui.QSizePolicy.Minimum)
        self.gridLayout_2.addItem(spacerItem4, 9, 2, 1, 1)
        spacerItem5 = QtGui.QSpacerItem(20, 10, QtGui.QSizePolicy.Minimum,
QtGui.QSizePolicy.Fixed)
        self.gridLayout_2.addItem(spacerItem5, 16, 0, 1, 1)
        self.label_8 = QtGui.QLabel(self.frame_4)
        sizePolicy = QtGui.QSizePolicy(QtGui.QSizePolicy.Expanding,
QtGui.QSizePolicy.Preferred)
        sizePolicy.setHorizontalStretch(0)
        sizePolicy.setVerticalStretch(0)
        sizePolicy.setHeightForWidth(self.label_8.sizePolicy().hasHeightForWidth())
        self.label_8.setSizePolicy(sizePolicy)
        self.label_8.setMinimumSize(QtCore.QSize(190, 0))
        self.label_8.setMaximumSize(QtCore.QSize(16777215, 25))
        font = QtGui.QFont()
        font.setFamily(_fromUtf8("Leelawadee UI"))
        font.setPointSize(11)
        font.setBold(False)
        font.setWeight(50)
        self.label_8.setFont(font)

self.label_8.setAlignment(QtCore.Qt.AlignLeading|QtCore.Qt.AlignLeft|QtCore.Qt.Alig
nVCenter)
        self.label_8.setObjectName(_fromUtf8("label_8"))
        self.gridLayout_2.addWidget(self.label_8, 9, 0, 1, 2)
        self.label_9 = QtGui.QLabel(self.frame_4)
        sizePolicy = QtGui.QSizePolicy(QtGui.QSizePolicy.Expanding,
QtGui.QSizePolicy.Preferred)
        sizePolicy.setHorizontalStretch(0)
        sizePolicy.setVerticalStretch(0)
        sizePolicy.setHeightForWidth(self.label_9.sizePolicy().hasHeightForWidth())
        self.label_9.setSizePolicy(sizePolicy)
        self.label_9.setMinimumSize(QtCore.QSize(190, 0))
        self.label_9.setMaximumSize(QtCore.QSize(16777215, 25))
        font = QtGui.QFont()
        font.setFamily(_fromUtf8("Leelawadee UI"))
        font.setPointSize(11)
        font.setBold(False)
        font.setWeight(50)
        self.label_9.setFont(font)

self.label_9.setAlignment(QtCore.Qt.AlignLeading|QtCore.Qt.AlignLeft|QtCore.Qt.Alig
nVCenter)
        self.label_9.setObjectName(_fromUtf8("label_9"))
        self.gridLayout_2.addWidget(self.label_9, 10, 0, 1, 2)
        self.label_14 = QtGui.QLabel(self.frame_4)
        sizePolicy = QtGui.QSizePolicy(QtGui.QSizePolicy.Expanding,
QtGui.QSizePolicy.Preferred)
        sizePolicy.setHorizontalStretch(0)
        sizePolicy.setVerticalStretch(0)

sizePolicy.setHeightForWidth(self.label_14.sizePolicy().hasHeightForWidth())
        self.label_14.setSizePolicy(sizePolicy)
        self.label_14.setMinimumSize(QtCore.QSize(210, 0))
        self.label_14.setMaximumSize(QtCore.QSize(16777215, 25))
        font = QtGui.QFont()
        font.setFamily(_fromUtf8("Leelawadee UI"))
```

```
        font.setPointSize(11)
        font.setBold(False)
        font.setWeight(50)
        self.label_14.setFont(font)

self.label_14.setAlignment(QtCore.Qt.AlignLeading|QtCore.Qt.AlignLeft|QtCore.Qt.Ali
gnVCenter)
        self.label_14.setObjectName(_fromUtf8("label_14"))
        self.gridLayout_2.addWidget(self.label_14, 19, 0, 1, 2)
        self.VastoL = QtGui.QDoubleSpinBox(self.frame_4)
        self.VastoL.setMinimumSize(QtCore.QSize(70, 0))
        font = QtGui.QFont()
        font.setFamily(_fromUtf8("Leelawadee UI"))
        font.setPointSize(11)
        self.VastoL.setFont(font)
        self.VastoL.setStyleSheet(_fromUtf8("background-color: rgb(255, 255,
255);"))
        self.VastoL.setWrapping(False)
        self.VastoL.setAlignment(QtCore.Qt.AlignCenter)
        self.VastoL.setReadOnly(True)
        self.VastoL.setButtonSymbols(QtGui.QAbstractSpinBox.NoButtons)
        self.VastoL.setSpecialValueText(_fromUtf8(""))
        self.VastoL.setMaximum(99999.99)
        self.VastoL.setObjectName(_fromUtf8("VastoL"))
        self.gridLayout_2.addWidget(self.VastoL, 10, 3, 1, 1)
        self.label_12 = QtGui.QLabel(self.frame_4)
        sizePolicy = QtGui.QSizePolicy(QtGui.QSizePolicy.Expanding,
QtGui.QSizePolicy.Preferred)
        sizePolicy.setHorizontalStretch(0)
        sizePolicy.setVerticalStretch(0)

sizePolicy.setHeightForWidth(self.label_12.sizePolicy().hasHeightForWidth())
        self.label_12.setSizePolicy(sizePolicy)
        self.label_12.setMinimumSize(QtCore.QSize(190, 0))
        self.label_12.setMaximumSize(QtCore.QSize(16777215, 25))
        font = QtGui.QFont()
        font.setFamily(_fromUtf8("Leelawadee UI"))
        font.setPointSize(11)
        font.setBold(False)
        font.setWeight(50)
        self.label_12.setFont(font)

self.label_12.setAlignment(QtCore.Qt.AlignLeading|QtCore.Qt.AlignLeft|QtCore.Qt.Ali
gnVCenter)
        self.label_12.setObjectName(_fromUtf8("label_12"))
        self.gridLayout_2.addWidget(self.label_12, 17, 0, 1, 2)
        self.RectoF = QtGui.QDoubleSpinBox(self.frame_4)
        self.RectoF.setMinimumSize(QtCore.QSize(70, 0))
        font = QtGui.QFont()
        font.setFamily(_fromUtf8("Leelawadee UI"))
        font.setPointSize(11)
        self.RectoF.setFont(font)
        self.RectoF.setStyleSheet(_fromUtf8("background-color: rgb(255, 255,
255);"))
        self.RectoF.setWrapping(False)
        self.RectoF.setAlignment(QtCore.Qt.AlignCenter)
        self.RectoF.setReadOnly(True)
        self.RectoF.setButtonSymbols(QtGui.QAbstractSpinBox.NoButtons)
        self.RectoF.setSpecialValueText(_fromUtf8(""))
        self.RectoF.setMaximum(99999.99)
        self.RectoF.setObjectName(_fromUtf8("RectoF"))
        self.gridLayout_2.addWidget(self.RectoF, 9, 3, 1, 1)
        self.label_15 = QtGui.QLabel(self.frame_4)
        sizePolicy = QtGui.QSizePolicy(QtGui.QSizePolicy.Expanding,
QtGui.QSizePolicy.Preferred)
        sizePolicy.setHorizontalStretch(0)
        sizePolicy.setVerticalStretch(0)
```

```
sizePolicy.setHeightForWidth(self.label_15.sizePolicy().hasHeightForWidth())
        self.label_15.setSizePolicy(sizePolicy)
        self.label_15.setMinimumSize(QtCore.QSize(190, 0))
        self.label_15.setMaximumSize(QtCore.QSize(16777215, 25))
        font = QtGui.QFont()
        font.setFamily(_fromUtf8("Leelawadee UI"))
        font.setPointSize(11)
        font.setBold(True)
        font.setWeight(75)
        self.label_15.setFont(font)

self.label_15.setAlignment(QtCore.Qt.AlignLeading|QtCore.Qt.AlignLeft|QtCore.Qt.Ali
gnVCenter)
        self.label_15.setObjectName(_fromUtf8("label_15"))
        self.gridLayout_2.addWidget(self.label_15, 20, 0, 1, 2)
        self.label_11 = QtGui.QLabel(self.frame_4)
        sizePolicy = QtGui.QSizePolicy(QtGui.QSizePolicy.Expanding,
QtGui.QSizePolicy.Preferred)
        sizePolicy.setHorizontalStretch(0)
        sizePolicy.setVerticalStretch(0)

sizePolicy.setHeightForWidth(self.label_11.sizePolicy().hasHeightForWidth())
        self.label_11.setSizePolicy(sizePolicy)
        self.label_11.setMinimumSize(QtCore.QSize(210, 0))
        self.label_11.setMaximumSize(QtCore.QSize(16777215, 25))
        font = QtGui.QFont()
        font.setFamily(_fromUtf8("Leelawadee UI"))
        font.setPointSize(11)
        font.setBold(True)
        font.setWeight(75)
        self.label_11.setFont(font)

self.label_11.setAlignment(QtCore.Qt.AlignLeading|QtCore.Qt.AlignLeft|QtCore.Qt.Ali
gnVCenter)
        self.label_11.setObjectName(_fromUtf8("label_11"))
        self.gridLayout_2.addWidget(self.label_11, 14, 0, 1, 2)
        spacerItem6 = QtGui.QSpacerItem(20, 10, QtGui.QSizePolicy.Minimum,
QtGui.QSizePolicy.Fixed)
        self.gridLayout_2.addItem(spacerItem6, 5, 0, 1, 1)
        self.Isquiotibial = QtGui.QDoubleSpinBox(self.frame_4)
        self.Isquiotibial.setMinimumSize(QtCore.QSize(70, 0))
        font = QtGui.QFont()
        font.setFamily(_fromUtf8("Leelawadee UI"))
        font.setPointSize(11)
        self.Isquiotibial.setFont(font)
        self.Isquiotibial.setStyleSheet(_fromUtf8("background-color: rgb(255, 255,
255);"))
        self.Isquiotibial.setWrapping(False)
        self.Isquiotibial.setAlignment(QtCore.Qt.AlignCenter)
        self.Isquiotibial.setReadOnly(True)
        self.Isquiotibial.setButtonSymbols(QtGui.QAbstractSpinBox.NoButtons)
        self.Isquiotibial.setSpecialValueText(_fromUtf8(""))
        self.Isquiotibial.setMaximum(99999.99)
        self.Isquiotibial.setObjectName(_fromUtf8("Isquiotibial"))
        self.gridLayout_2.addWidget(self.Isquiotibial, 20, 3, 1, 1)
        self.label_3 = QtGui.QLabel(self.frame_4)
        self.label_3.setMaximumSize(QtCore.QSize(16777215, 250))
        self.label_3.setText(_fromUtf8(""))
        self.label_3.setPixmap(QtGui.QPixmap(_fromUtf8("puntos motores.jpg")))
        self.label_3.setScaledContents(True)
        self.label_3.setObjectName(_fromUtf8("label_3"))
        self.gridLayout_2.addWidget(self.label_3, 4, 0, 1, 6)
        self.VastoM = QtGui.QDoubleSpinBox(self.frame_4)
        self.VastoM.setMinimumSize(QtCore.QSize(70, 0))
        font = QtGui.QFont()
        font.setFamily(_fromUtf8("Leelawadee UI"))
        font.setPointSize(11)
        self.VastoM.setFont(font)
```

```
        self.VastoM.setStyleSheet(_fromUtf8("background-color: rgb(255, 255,
255);"))
        self.VastoM.setWrapping(False)
        self.VastoM.setAlignment(QtCore.Qt.AlignCenter)
        self.VastoM.setReadOnly(True)
        self.VastoM.setButtonSymbols(QtGui.QAbstractSpinBox.NoButtons)
        self.VastoM.setSpecialValueText(_fromUtf8(""))
        self.VastoM.setMaximum(99999.99)
        self.VastoM.setObjectName(_fromUtf8("VastoM"))
        self.gridLayout_2.addWidget(self.VastoM, 12, 3, 1, 1)
        self.label_13 = QtGui.QLabel(self.frame_4)
        sizePolicy = QtGui.QSizePolicy(QtGui.QSizePolicy.Expanding,
QtGui.QSizePolicy.Preferred)
        sizePolicy.setHorizontalStretch(0)
        sizePolicy.setVerticalStretch(0)

sizePolicy.setHeightForWidth(self.label_13.sizePolicy().hasHeightForWidth())
        self.label_13.setSizePolicy(sizePolicy)
        self.label_13.setMinimumSize(QtCore.QSize(190, 0))
        self.label_13.setMaximumSize(QtCore.QSize(16777215, 25))
        font = QtGui.QFont()
        font.setFamily(_fromUtf8("Leelawadee UI"))
        font.setPointSize(11)
        font.setBold(False)
        font.setWeight(50)
        self.label_13.setFont(font)

self.label_13.setAlignment(QtCore.Qt.AlignLeading|QtCore.Qt.AlignLeft|QtCore.Qt.Ali
gnVCenter)
        self.label_13.setObjectName(_fromUtf8("label_13"))
        self.gridLayout_2.addWidget(self.label_13, 18, 0, 1, 2)
        self.Cuadriceps = QtGui.QDoubleSpinBox(self.frame_4)
        self.Cuadriceps.setMinimumSize(QtCore.QSize(70, 0))
        font = QtGui.QFont()
        font.setFamily(_fromUtf8("Leelawadee UI"))
        font.setPointSize(11)
        self.Cuadriceps.setFont(font)
        self.Cuadriceps.setStyleSheet(_fromUtf8("background-color: rgb(255, 255,
255);"))
        self.Cuadriceps.setWrapping(False)
        self.Cuadriceps.setAlignment(QtCore.Qt.AlignCenter)
        self.Cuadriceps.setReadOnly(True)
        self.Cuadriceps.setButtonSymbols(QtGui.QAbstractSpinBox.NoButtons)
        self.Cuadriceps.setSpecialValueText(_fromUtf8(""))
        self.Cuadriceps.setMaximum(99999.99)
        self.Cuadriceps.setObjectName(_fromUtf8("Cuadriceps"))
        self.gridLayout_2.addWidget(self.Cuadriceps, 14, 3, 1, 1)
        self.label_10 = QtGui.QLabel(self.frame_4)
        sizePolicy = QtGui.QSizePolicy(QtGui.QSizePolicy.Expanding,
QtGui.QSizePolicy.Preferred)
        sizePolicy.setHorizontalStretch(0)
        sizePolicy.setVerticalStretch(0)

sizePolicy.setHeightForWidth(self.label_10.sizePolicy().hasHeightForWidth())
        self.label_10.setSizePolicy(sizePolicy)
        self.label_10.setMinimumSize(QtCore.QSize(190, 0))
        self.label_10.setMaximumSize(QtCore.QSize(16777215, 25))
        font = QtGui.QFont()
        font.setFamily(_fromUtf8("Leelawadee UI"))
        font.setPointSize(11)
        font.setBold(False)
        font.setWeight(50)
        self.label_10.setFont(font)

self.label_10.setAlignment(QtCore.Qt.AlignLeading|QtCore.Qt.AlignLeft|QtCore.Qt.Ali
gnVCenter)
        self.label_10.setObjectName(_fromUtf8("label_10"))
        self.gridLayout_2.addWidget(self.label_10, 12, 0, 1, 2)
```

```
        self.Semiten = QtGui.QDoubleSpinBox(self.frame_4)
        self.Semiten.setMinimumSize(QtCore.QSize(70, 0))
        font = QtGui.QFont()
        font.setFamily(_fromUtf8("Leelawadee UI"))
        font.setPointSize(11)
        self.Semiten.setFont(font)
        self.Semiten.setStyleSheet(_fromUtf8("background-color: rgb(255, 255,
255);"))
        self.Semiten.setWrapping(False)
        self.Semiten.setAlignment(QtCore.Qt.AlignCenter)
        self.Semiten.setReadOnly(True)
        self.Semiten.setButtonSymbols(QtGui.QAbstractSpinBox.NoButtons)
        self.Semiten.setSpecialValueText(_fromUtf8(""))
        self.Semiten.setMaximum(99999.99)
        self.Semiten.setObjectName(_fromUtf8("Semiten"))
        self.gridLayout_2.addWidget(self.Semiten, 18, 3, 1, 1)
        self.Semimem = QtGui.QDoubleSpinBox(self.frame_4)
        self.Semimem.setMinimumSize(QtCore.QSize(70, 0))
        font = QtGui.QFont()
        font.setFamily(_fromUtf8("Leelawadee UI"))
        font.setPointSize(11)
        self.Semimem.setFont(font)
        self.Semimem.setStyleSheet(_fromUtf8("background-color: rgb(255, 255,
255);"))
        self.Semimem.setWrapping(False)
        self.Semimem.setAlignment(QtCore.Qt.AlignCenter)
        self.Semimem.setReadOnly(True)
        self.Semimem.setButtonSymbols(QtGui.QAbstractSpinBox.NoButtons)
        self.Semimem.setSpecialValueText(_fromUtf8(""))
        self.Semimem.setMaximum(99999.99)
        self.Semimem.setObjectName(_fromUtf8("Semimem"))
        self.gridLayout_2.addWidget(self.Semimem, 19, 3, 1, 1)
        self.horizontalLayout = QtGui.QHBoxLayout()
        self.horizontalLayout.setObjectName(_fromUtf8("horizontalLayout"))
        self.label_6 = QtGui.QLabel(self.frame_4)
        self.label_6.setMaximumSize(QtCore.QSize(300, 70))
        self.label_6.setFrameShape(QtGui.QFrame.NoFrame)
        self.label_6.setFrameShadow(QtGui.QFrame.Raised)
        self.label_6.setText(_fromUtf8(""))
        self.label_6.setPixmap(QtGui.QPixmap(_fromUtf8("LOGO.png")))
        self.label_6.setScaledContents(True)
        self.label_6.setObjectName(_fromUtf8("label_6"))
        self.horizontalLayout.addWidget(self.label_6)
        self.gridLayout_2.addLayout(self.horizontalLayout, 1, 0, 1, 6)
        self.Biceps = QtGui.QDoubleSpinBox(self.frame_4)
        self.Biceps.setMinimumSize(QtCore.QSize(70, 0))
        font = QtGui.QFont()
        font.setFamily(_fromUtf8("Leelawadee UI"))
        font.setPointSize(11)
        self.Biceps.setFont(font)
        self.Biceps.setStyleSheet(_fromUtf8("background-color: rgb(255, 255,
255);"))
        self.Biceps.setWrapping(False)
        self.Biceps.setAlignment(QtCore.Qt.AlignCenter)
        self.Biceps.setReadOnly(True)
        self.Biceps.setButtonSymbols(QtGui.QAbstractSpinBox.NoButtons)
        self.Biceps.setSpecialValueText(_fromUtf8(""))
        self.Biceps.setMaximum(99999.99)
        self.Biceps.setObjectName(_fromUtf8("Biceps"))
        self.gridLayout_2.addWidget(self.Biceps, 17, 3, 1, 1)
        self.CuadricepsP = QtGui.QLineEdit(self.frame_4)
        self.CuadricepsP.setMinimumSize(QtCore.QSize(50, 0))
        self.CuadricepsP.setMaximumSize(QtCore.QSize(70, 16777215))
        font = QtGui.QFont()
        font.setFamily(_fromUtf8("Leelawadee UI"))
        font.setPointSize(11)
        self.CuadricepsP.setFont(font)
        self.CuadricepsP.setStyleSheet(_fromUtf8("background-color: rgb(255, 255,
```

```
255);"))
        self.CuadricepsP.setFrame(False)
        self.CuadricepsP.setReadOnly(True)
        self.CuadricepsP.setObjectName(_fromUtf8("CuadricepsP"))
        self.gridLayout_2.addWidget(self.CuadricepsP, 14, 5, 1, 1)
        self.VastoMP = QtGui.QLineEdit(self.frame_4)
        self.VastoMP.setMinimumSize(QtCore.QSize(50, 0))
        self.VastoMP.setMaximumSize(QtCore.QSize(70, 16777215))
        font = QtGui.QFont()
        font.setFamily(_fromUtf8("Leelawadee UI"))
        font.setPointSize(11)
        self.VastoMP.setFont(font)
        self.VastoMP.setStyleSheet(_fromUtf8("background-color: rgb(255, 255,
255);"))
        self.VastoMP.setFrame(False)
        self.VastoMP.setReadOnly(True)
        self.VastoMP.setObjectName(_fromUtf8("VastoMP"))
        self.gridLayout_2.addWidget(self.VastoMP, 12, 5, 1, 1)
        self.SemitenP = QtGui.QLineEdit(self.frame_4)
        self.SemitenP.setMinimumSize(QtCore.QSize(50, 0))
        self.SemitenP.setMaximumSize(QtCore.QSize(70, 16777215))
        font = QtGui.QFont()
        font.setFamily(_fromUtf8("Leelawadee UI"))
        font.setPointSize(11)
        self.SemitenP.setFont(font)
        self.SemitenP.setStyleSheet(_fromUtf8("background-color: rgb(255, 255,
255);"))
        self.SemitenP.setFrame(False)
        self.SemitenP.setReadOnly(True)
        self.SemitenP.setObjectName(_fromUtf8("SemitenP"))
        self.gridLayout_2.addWidget(self.SemitenP, 18, 5, 1, 1)
        self.horizontalLayout_5 = QtGui.QHBoxLayout()
        self.horizontalLayout_5.setContentsMargins(-1, 0, -1, -1)
        self.horizontalLayout_5.setObjectName(_fromUtf8("horizontalLayout_5"))
        self.resultados = QtGui.QLabel(self.frame_4)
        sizePolicy = QtGui.QSizePolicy(QtGui.QSizePolicy.Expanding,
QtGui.QSizePolicy.Preferred)
        sizePolicy.setHorizontalStretch(0)
        sizePolicy.setVerticalStretch(0)

sizePolicy.setHeightForWidth(self.resultados.sizePolicy().hasHeightForWidth())
        self.resultados.setSizePolicy(sizePolicy)
        self.resultados.setMinimumSize(QtCore.QSize(0, 0))
        self.resultados.setMaximumSize(QtCore.QSize(16777215, 25))
        font = QtGui.QFont()
        font.setFamily(_fromUtf8("Leelawadee UI"))
        font.setPointSize(11)
        font.setBold(True)
        font.setWeight(75)
        self.resultados.setFont(font)
        self.resultados.setStyleSheet(_fromUtf8("color: rgb(0, 59, 177);"))
        self.resultados.setAlignment(QtCore.Qt.AlignCenter)
        self.resultados.setObjectName(_fromUtf8("resultados"))
        self.horizontalLayout_5.addWidget(self.resultados)
        self.btn_ayuda = QtGui.QPushButton(self.frame_4)
        self.btn_ayuda.setMaximumSize(QtCore.QSize(50, 16777215))
        self.btn_ayuda.setLayoutDirection(QtCore.Qt.RightToLeft)
        self.btn_ayuda.setText(_fromUtf8(""))
        icon3 = QtGui.QIcon()
        icon3.addPixmap(QtGui.QPixmap(_fromUtf8("ayuda.png")), QtGui.QIcon.Normal,
QtGui.QIcon.Off)
        self.btn_ayuda.setIcon(icon3)
        self.btn_ayuda.setIconSize(QtCore.QSize(25, 25))
        self.btn_ayuda.setAutoDefault(False)
        self.btn_ayuda.setDefault(False)
        self.btn_ayuda.setFlat(True)
        self.btn_ayuda.setObjectName(_fromUtf8("btn_ayuda"))
        self.horizontalLayout_5.addWidget(self.btn_ayuda)
```

```
        self.gridLayout_2.addLayout(self.horizontalLayout_5, 6, 0, 1, 6)
        self.pb_isquiotibial = QtGui.QProgressBar(self.frame_4)
        sizePolicy = QtGui.QSizePolicy(QtGui.QSizePolicy.Fixed,
QtGui.QSizePolicy.Fixed)
        sizePolicy.setHorizontalStretch(0)
        sizePolicy.setVerticalStretch(0)

sizePolicy.setHeightForWidth(self.pb_isquiotibial.sizePolicy().hasHeightForWidth())
        self.pb_isquiotibial.setSizePolicy(sizePolicy)
        self.pb_isquiotibial.setMinimumSize(QtCore.QSize(60, 25))
        self.pb_isquiotibial.setMaximumSize(QtCore.QSize(60, 16777215))
        font = QtGui.QFont()
        font.setFamily(_fromUtf8("Leelawadee UI"))
        font.setPointSize(11)
        self.pb_isquiotibial.setFont(font)
        self.pb_isquiotibial.setProperty("value", 0)
        self.pb_isquiotibial.setAlignment(QtCore.Qt.AlignCenter)
        self.pb_isquiotibial.setObjectName(_fromUtf8("pb_isquiotibial"))
        self.gridLayout_2.addWidget(self.pb_isquiotibial, 20, 4, 1, 1)
        self.pb_biceps = QtGui.QProgressBar(self.frame_4)
        sizePolicy = QtGui.QSizePolicy(QtGui.QSizePolicy.Fixed,
QtGui.QSizePolicy.Fixed)
        sizePolicy.setHorizontalStretch(0)
        sizePolicy.setVerticalStretch(0)

sizePolicy.setHeightForWidth(self.pb_biceps.sizePolicy().hasHeightForWidth())
        self.pb_biceps.setSizePolicy(sizePolicy)
        self.pb_biceps.setMinimumSize(QtCore.QSize(60, 25))
        self.pb_biceps.setMaximumSize(QtCore.QSize(60, 16777215))
        font = QtGui.QFont()
        font.setFamily(_fromUtf8("Leelawadee UI"))
        font.setPointSize(11)
        self.pb_biceps.setFont(font)
        self.pb_biceps.setProperty("value", 0)
        self.pb_biceps.setAlignment(QtCore.Qt.AlignCenter)
        self.pb_biceps.setObjectName(_fromUtf8("pb_biceps"))
        self.gridLayout_2.addWidget(self.pb_biceps, 17, 4, 1, 1)
        self.VastoLP = QtGui.QLineEdit(self.frame_4)
        self.VastoLP.setMinimumSize(QtCore.QSize(50, 0))
        self.VastoLP.setMaximumSize(QtCore.QSize(70, 16777215))
        font = QtGui.QFont()
        font.setFamily(_fromUtf8("Leelawadee UI"))
        font.setPointSize(11)
        self.VastoLP.setFont(font)
        self.VastoLP.setStyleSheet(_fromUtf8("background-color: rgb(255, 255,
255);"))
        self.VastoLP.setFrame(False)
        self.VastoLP.setReadOnly(True)
        self.VastoLP.setObjectName(_fromUtf8("VastoLP"))
        self.gridLayout_2.addWidget(self.VastoLP, 10, 5, 1, 1)
        self.BicepsP = QtGui.QLineEdit(self.frame_4)
        self.BicepsP.setMinimumSize(QtCore.QSize(50, 0))
        self.BicepsP.setMaximumSize(QtCore.QSize(70, 16777215))
        font = QtGui.QFont()
        font.setFamily(_fromUtf8("Leelawadee UI"))
        font.setPointSize(11)
        self.BicepsP.setFont(font)
        self.BicepsP.setStyleSheet(_fromUtf8("background-color: rgb(255, 255,
255);"))
        self.BicepsP.setFrame(False)
        self.BicepsP.setReadOnly(True)
        self.BicepsP.setObjectName(_fromUtf8("BicepsP"))
        self.gridLayout_2.addWidget(self.BicepsP, 17, 5, 1, 1)
        self.pb_vastoL = QtGui.QProgressBar(self.frame_4)
        sizePolicy = QtGui.QSizePolicy(QtGui.QSizePolicy.Fixed,
QtGui.QSizePolicy.Fixed)
        sizePolicy.setHorizontalStretch(0)
        sizePolicy.setVerticalStretch(0)
```

```
sizePolicy.setHeightForWidth(self.pb_vastoL.sizePolicy().hasHeightForWidth())
        self.pb_vastoL.setSizePolicy(sizePolicy)
        self.pb_vastoL.setMinimumSize(QtCore.QSize(60, 25))
        self.pb_vastoL.setMaximumSize(QtCore.QSize(60, 16777215))
        font = QtGui.QFont()
        font.setFamily(_fromUtf8("Leelawadee UI"))
        font.setPointSize(11)
        self.pb_vastoL.setFont(font)
        self.pb_vastoL.setProperty("value", 0)
        self.pb_vastoL.setAlignment(QtCore.Qt.AlignCenter)
        self.pb_vastoL.setObjectName(_fromUtf8("pb_vastoL"))
        self.gridLayout_2.addWidget(self.pb_vastoL, 10, 4, 1, 1)
        self.IsquiotibialesP = QtGui.QLineEdit(self.frame_4)
        self.IsquiotibialesP.setMinimumSize(QtCore.QSize(50, 0))
        self.IsquiotibialesP.setMaximumSize(QtCore.QSize(70, 16777215))
        font = QtGui.QFont()
        font.setFamily(_fromUtf8("Leelawadee UI"))
        font.setPointSize(11)
        self.IsquiotibialesP.setFont(font)
        self.IsquiotibialesP.setStyleSheet(_fromUtf8("background-color: rgb(255,
255, 255);"))
        self.IsquiotibialesP.setFrame(False)
        self.IsquiotibialesP.setReadOnly(True)
        self.IsquiotibialesP.setObjectName(_fromUtf8("IsquiotibialesP"))
        self.gridLayout_2.addWidget(self.IsquiotibialesP, 20, 5, 1, 1)
        self.SemimemP = QtGui.QLineEdit(self.frame_4)
        self.SemimemP.setMinimumSize(QtCore.QSize(50, 0))
        self.SemimemP.setMaximumSize(QtCore.QSize(70, 16777215))
        font = QtGui.QFont()
        font.setFamily(_fromUtf8("Leelawadee UI"))
        font.setPointSize(11)
        self.SemimemP.setFont(font)
        self.SemimemP.setStyleSheet(_fromUtf8("background-color: rgb(255, 255,
255);"))
        self.SemimemP.setFrame(False)
        self.SemimemP.setReadOnly(True)
        self.SemimemP.setObjectName(_fromUtf8("SemimemP"))
        self.gridLayout_2.addWidget(self.SemimemP, 19, 5, 1, 1)
        self.pb_semimem = QtGui.QProgressBar(self.frame_4)
        sizePolicy = QtGui.QSizePolicy(QtGui.QSizePolicy.Fixed,
QtGui.QSizePolicy.Fixed)
        sizePolicy.setHorizontalStretch(0)
        sizePolicy.setVerticalStretch(0)

sizePolicy.setHeightForWidth(self.pb_semimem.sizePolicy().hasHeightForWidth())
        self.pb_semimem.setSizePolicy(sizePolicy)
        self.pb_semimem.setMinimumSize(QtCore.QSize(60, 25))
        self.pb_semimem.setMaximumSize(QtCore.QSize(60, 16777215))
        font = QtGui.QFont()
        font.setFamily(_fromUtf8("Leelawadee UI"))
        font.setPointSize(11)
        self.pb_semimem.setFont(font)
        self.pb_semimem.setProperty("value", 0)
        self.pb_semimem.setAlignment(QtCore.Qt.AlignCenter)
        self.pb_semimem.setObjectName(_fromUtf8("pb_semimem"))
        self.gridLayout_2.addWidget(self.pb_semimem, 19, 4, 1, 1)
        self.pb_recto = QtGui.QProgressBar(self.frame_4)
        sizePolicy = QtGui.QSizePolicy(QtGui.QSizePolicy.Fixed,
QtGui.QSizePolicy.Fixed)
        sizePolicy.setHorizontalStretch(0)
        sizePolicy.setVerticalStretch(0)

sizePolicy.setHeightForWidth(self.pb_recto.sizePolicy().hasHeightForWidth())
        self.pb_recto.setSizePolicy(sizePolicy)
        self.pb_recto.setMinimumSize(QtCore.QSize(60, 25))
        self.pb_recto.setMaximumSize(QtCore.QSize(60, 16777215))
        font = QtGui.QFont()
```

```
        font.setFamily(_fromUtf8("Leelawadee UI"))
        font.setPointSize(11)
        self.pb_recto.setFont(font)
        self.pb_recto.setProperty("value", 0)
        self.pb_recto.setAlignment(QtCore.Qt.AlignCenter)
        self.pb_recto.setObjectName(_fromUtf8("pb_recto"))
        self.gridLayout_2.addWidget(self.pb_recto, 9, 4, 1, 1)
        self.pb_vastoM = QtGui.QProgressBar(self.frame_4)
        sizePolicy = QtGui.QSizePolicy(QtGui.QSizePolicy.Fixed,
QtGui.QSizePolicy.Fixed)
        sizePolicy.setHorizontalStretch(0)
        sizePolicy.setVerticalStretch(0)

sizePolicy.setHeightForWidth(self.pb_vastoM.sizePolicy().hasHeightForWidth())
        self.pb_vastoM.setSizePolicy(sizePolicy)
        self.pb_vastoM.setMinimumSize(QtCore.QSize(60, 25))
        self.pb_vastoM.setMaximumSize(QtCore.QSize(60, 16777215))
        font = QtGui.QFont()
        font.setFamily(_fromUtf8("Leelawadee UI"))
        font.setPointSize(11)
        self.pb_vastoM.setFont(font)
        self.pb_vastoM.setProperty("value", 0)
        self.pb_vastoM.setAlignment(QtCore.Qt.AlignCenter)
        self.pb_vastoM.setObjectName(_fromUtf8("pb_vastoM"))
        self.gridLayout_2.addWidget(self.pb_vastoM, 12, 4, 1, 1)
        self.pb_semiten = QtGui.QProgressBar(self.frame_4)
        sizePolicy = QtGui.QSizePolicy(QtGui.QSizePolicy.Fixed,
QtGui.QSizePolicy.Fixed)
        sizePolicy.setHorizontalStretch(0)
        sizePolicy.setVerticalStretch(0)

sizePolicy.setHeightForWidth(self.pb_semiten.sizePolicy().hasHeightForWidth())
        self.pb_semiten.setSizePolicy(sizePolicy)
        self.pb_semiten.setMinimumSize(QtCore.QSize(60, 25))
        self.pb_semiten.setMaximumSize(QtCore.QSize(60, 16777215))
        font = QtGui.QFont()
        font.setFamily(_fromUtf8("Leelawadee UI"))
        font.setPointSize(11)
        self.pb_semiten.setFont(font)
        self.pb_semiten.setProperty("value", 0)
        self.pb_semiten.setAlignment(QtCore.Qt.AlignCenter)
        self.pb_semiten.setObjectName(_fromUtf8("pb_semiten"))
        self.gridLayout_2.addWidget(self.pb_semiten, 18, 4, 1, 1)
        self.pb_cuadriceps = QtGui.QProgressBar(self.frame_4)
        sizePolicy = QtGui.QSizePolicy(QtGui.QSizePolicy.Fixed,
QtGui.QSizePolicy.Fixed)
        sizePolicy.setHorizontalStretch(0)
        sizePolicy.setVerticalStretch(0)

sizePolicy.setHeightForWidth(self.pb_cuadriceps.sizePolicy().hasHeightForWidth())
        self.pb_cuadriceps.setSizePolicy(sizePolicy)
        self.pb_cuadriceps.setMinimumSize(QtCore.QSize(60, 25))
        self.pb_cuadriceps.setMaximumSize(QtCore.QSize(60, 16777215))
        font = QtGui.QFont()
        font.setFamily(_fromUtf8("Leelawadee UI"))
        font.setPointSize(11)
        self.pb_cuadriceps.setFont(font)
        self.pb_cuadriceps.setProperty("value", 0)
        self.pb_cuadriceps.setAlignment(QtCore.Qt.AlignCenter)
        self.pb_cuadriceps.setObjectName(_fromUtf8("pb_cuadriceps"))
        self.gridLayout_2.addWidget(self.pb_cuadriceps, 14, 4, 1, 1)
        self.RectoFP = QtGui.QLineEdit(self.frame_4)
        sizePolicy = QtGui.QSizePolicy(QtGui.QSizePolicy.Preferred,
QtGui.QSizePolicy.Fixed)
        sizePolicy.setHorizontalStretch(0)
        sizePolicy.setVerticalStretch(0)
        sizePolicy.setHeightForWidth(self.RectoFP.sizePolicy().hasHeightForWidth())
        self.RectoFP.setSizePolicy(sizePolicy)
```

```
        self.RectoFP.setMinimumSize(QtCore.QSize(50, 0))
        self.RectoFP.setMaximumSize(QtCore.QSize(70, 16777215))
        font = QtGui.QFont()
        font.setFamily(_fromUtf8("Leelawadee UI"))
        font.setPointSize(11)
        self.RectoFP.setFont(font)
        self.RectoFP.setStyleSheet(_fromUtf8("background-color: rgb(255, 255,
255);"))
        self.RectoFP.setFrame(False)
        self.RectoFP.setReadOnly(True)
        self.RectoFP.setObjectName(_fromUtf8("RectoFP"))
        self.gridLayout_2.addWidget(self.RectoFP, 9, 5, 1, 1)
        self.gridLayout_5.addWidget(self.frame_4, 1, 0, 1, 1)
        self.frame = QtGui.QFrame(self.centralwidget)
        self.frame.setFrameShape(QtGui.QFrame.NoFrame)
        self.frame.setFrameShadow(QtGui.QFrame.Sunken)
        self.frame.setObjectName(_fromUtf8("frame"))
        self.horizontalLayout_2 = QtGui.QHBoxLayout(self.frame)
        self.horizontalLayout_2.setContentsMargins(0, 0, -1, 0)
        self.horizontalLayout_2.setObjectName(_fromUtf8("horizontalLayout_2"))
        self.frame_2 = QtGui.QFrame(self.frame)
        self.frame_2.setMinimumSize(QtCore.QSize(450, 0))
        self.frame_2.setFrameShape(QtGui.QFrame.NoFrame)
        self.frame_2.setFrameShadow(QtGui.QFrame.Sunken)
        self.frame_2.setObjectName(_fromUtf8("frame_2"))
        self.horizontalLayout_3 = QtGui.QHBoxLayout(self.frame_2)
        self.horizontalLayout_3.setObjectName(_fromUtf8("horizontalLayout_3"))
        self.btn_Conectar = QtGui.QPushButton(self.frame_2)
        self.btn_Conectar.setMinimumSize(QtCore.QSize(187, 0))
        font = QtGui.QFont()
        font.setFamily(_fromUtf8("Leelawadee UI"))
        font.setPointSize(11)
        font.setBold(True)
        font.setWeight(75)
        self.btn_Conectar.setFont(font)
        self.btn_Conectar.setStyleSheet(_fromUtf8("color: rgb(0, 170, 0);"))
        icon4 = QtGui.QIcon()
        icon4.addPixmap(QtGui.QPixmap(_fromUtf8("conectar.png")),
QtGui.QIcon.Normal, QtGui.QIcon.Off)
        self.btn_Conectar.setIcon(icon4)
        self.btn_Conectar.setIconSize(QtCore.QSize(25, 25))
        self.btn_Conectar.setObjectName(_fromUtf8("btn_Conectar"))
        self.horizontalLayout_3.addWidget(self.btn_Conectar)
        self.btn_Desconectar = QtGui.QPushButton(self.frame_2)
        self.btn_Desconectar.setMinimumSize(QtCore.QSize(187, 0))
        font = QtGui.QFont()
        font.setBold(True)
        font.setWeight(75)
        self.btn_Desconectar.setFont(font)
        self.btn_Desconectar.setStyleSheet(_fromUtf8("color: rgb(204, 0, 0);"))
        icon5 = QtGui.QIcon()
        icon5.addPixmap(QtGui.QPixmap(_fromUtf8("desconectar.png")),
QtGui.QIcon.Normal, QtGui.QIcon.Off)
        self.btn_Desconectar.setIcon(icon5)
        self.btn_Desconectar.setIconSize(QtCore.QSize(25, 25))
        self.btn_Desconectar.setObjectName(_fromUtf8("btn_Desconectar"))
        self.horizontalLayout_3.addWidget(self.btn_Desconectar)
        self.btn_Desconectar.raise_()
        self.btn_Conectar.raise_()
        self.horizontalLayout_2.addWidget(self.frame_2)
        spacerItem7 = QtGui.QSpacerItem(665, 20, QtGui.QSizePolicy.Expanding,
QtGui.QSizePolicy.Minimum)
        self.horizontalLayout_2.addItem(spacerItem7)
        self.btn_cancelar = QtGui.QPushButton(self.frame)
        self.btn_cancelar.setMinimumSize(QtCore.QSize(80, 30))
        self.btn_cancelar.setMaximumSize(QtCore.QSize(90, 16777215))
        self.btn_cancelar.setObjectName(_fromUtf8("btn_cancelar"))
        self.horizontalLayout_2.addWidget(self.btn_cancelar)
```

```
        self.btn_aceptar = QtGui.QPushButton(self.frame)
        self.btn_aceptar.setMinimumSize(QtCore.QSize(80, 30))
        self.btn_aceptar.setMaximumSize(QtCore.QSize(90, 16777215))
        self.btn_aceptar.setObjectName(_fromUtf8("btn_aceptar"))
        self.horizontalLayout_2.addWidget(self.btn_aceptar)
        self.gridLayout_5.addWidget(self.frame, 3, 0, 1, 3)
        Evaluacion.setCentralWidget(self.centralwidget)
        self.statusbar = QtGui.QStatusBar(Evaluacion)
        self.statusbar.setObjectName(_fromUtf8("statusbar"))
        Evaluacion.setStatusBar(self.statusbar)

        self.retranslateUi(Evaluacion)
        self.tab_Eval_Inicial.setCurrentIndex(0)
        QtCore.QMetaObject.connectSlotsByName(Evaluacion)

    def retranslateUi(self, Evaluacion):
        Evaluacion.setWindowTitle(_translate("Evaluacion", "EVALUACION", None))
        self.btn_guardar_E.setText(_translate("Evaluacion", "Guardar", None))
        self.btn_parar_E.setText(_translate("Evaluacion", "Parar", None))
        self.btn_Inicio_E.setText(_translate("Evaluacion", "Iniciar", None))

self.tab_Eval_Inicial.setTabText(self.tab_Eval_Inicial.indexOf(self.extension),
_translate("Evaluacion", "Extensión de la rodilla", None))
        self.btn_guardar_F.setText(_translate("Evaluacion", "Guardar", None))
        self.btn_inicio_F.setText(_translate("Evaluacion", "Iniciar", None))
        self.btn_parar_F.setText(_translate("Evaluacion", "Parar", None))

self.tab_Eval_Inicial.setTabText(self.tab_Eval_Inicial.indexOf(self.flexion),
_translate("Evaluacion", "Flexión de la rodilla", None))
        self.label_8.setText(_translate("Evaluacion", "Fuerza Recto femoral (mV):",
None))
        self.label_9.setText(_translate("Evaluacion", "Fuerza Vasto lateral (mV):",
None))
        self.label_14.setText(_translate("Evaluacion", "Fuerza Semimembranoso
(mV):", None))
        self.label_12.setText(_translate("Evaluacion", "Fuerza Bíceps femoral
(mV):", None))
        self.label_15.setText(_translate("Evaluacion", "Fuerza Isquiotibiales
(mV):", None))
        self.label_11.setText(_translate("Evaluacion", "Prom. extensión rodilla
(mV):", None))
        self.label_13.setText(_translate("Evaluacion", "Fuerza Semitendinoso
(mV):", None))
        self.label_10.setText(_translate("Evaluacion", "Fuerza Vasto medial (mV):",
None))
        self.resultados.setText(_translate("Evaluacion", "RESULTADOS", None))
        self.btn_Conectar.setText(_translate("Evaluacion", "Conectar", None))
        self.btn_Desconectar.setText(_translate("Evaluacion", "Desconectar", None))
        self.btn_cancelar.setText(_translate("Evaluacion", "Cancelar", None))
        self.btn_aceptar.setText(_translate("Evaluacion", "Aceptar", None))
```

4) Código de la pantalla de Historia Clínica

```
# -*- coding: utf-8 -*-

# Form implementation generated from reading ui file 'historia.ui'
#
# Created by: PyQt4 UI code generator 4.11.4
#
# WARNING! All changes made in this file will be lost!

from PyQt4 import QtCore, QtGui

try:
    _fromUtf8 = QtCore.QString.fromUtf8
except AttributeError:
    def _fromUtf8(s):
        return s

try:
    _encoding = QtGui.QApplication.UnicodeUTF8
    def _translate(context, text, disambig):
        return QtGui.QApplication.translate(context, text, disambig, _encoding)
except AttributeError:
    def _translate(context, text, disambig):
        return QtGui.QApplication.translate(context, text, disambig)

class Ui_Diagnostico(object):
    def setupUi(self, Diagnostico):
        Diagnostico.setObjectName(_fromUtf8("Diagnostico"))
        Diagnostico.resize(1318, 774)
        font = QtGui.QFont()
        font.setFamily(_fromUtf8("Leelawadee UI"))
        font.setPointSize(11)
        Diagnostico.setFont(font)
        self.centralwidget = QtGui.QWidget(Diagnostico)
        self.centralwidget.setObjectName(_fromUtf8("centralwidget"))
        self.gridLayout = QtGui.QGridLayout(self.centralwidget)
        self.gridLayout.setObjectName(_fromUtf8("gridLayout"))
        self.frame_7 = QtGui.QFrame(self.centralwidget)
        self.frame_7.setMinimumSize(QtCore.QSize(300, 0))
        self.frame_7.setMaximumSize(QtCore.QSize(300, 300))
        self.frame_7.setFrameShape(QtGui.QFrame.Box)
        self.frame_7.setFrameShadow(QtGui.QFrame.Sunken)
        self.frame_7.setObjectName(_fromUtf8("frame_7"))
        self.verticalLayout_3 = QtGui.QVBoxLayout(self.frame_7)
        self.verticalLayout_3.setObjectName(_fromUtf8("verticalLayout_3"))
        self.label_40 = QtGui.QLabel(self.frame_7)
        self.label_40.setMinimumSize(QtCore.QSize(0, 30))
        self.label_40.setMaximumSize(QtCore.QSize(16777215, 30))
        font = QtGui.QFont()
        font.setBold(True)
        font.setWeight(75)
        self.label_40.setFont(font)
        self.label_40.setStyleSheet(_fromUtf8("background-color: rgb(216, 213,
255);"))
        self.label_40.setFrameShape(QtGui.QFrame.Panel)
        self.label_40.setFrameShadow(QtGui.QFrame.Sunken)
        self.label_40.setAlignment(QtCore.Qt.AlignCenter)
        self.label_40.setObjectName(_fromUtf8("label_40"))
        self.verticalLayout_3.addWidget(self.label_40)
        self.label_24 = QtGui.QLabel(self.frame_7)
        font = QtGui.QFont()
        font.setBold(False)
        font.setWeight(50)
        self.label_24.setFont(font)
        self.label_24.setObjectName(_fromUtf8("label_24"))
```

```
        self.verticalLayout_3.addWidget(self.label_24)
        self.antecedenteP = QtGui.QPlainTextEdit(self.frame_7)
        self.antecedenteP.setFrameShape(QtGui.QFrame.NoFrame)
        self.antecedenteP.setReadOnly(True)
        self.antecedenteP.setObjectName(_fromUtf8("antecedenteP"))
        self.verticalLayout_3.addWidget(self.antecedenteP)
        self.label_41 = QtGui.QLabel(self.frame_7)
        font = QtGui.QFont()
        font.setBold(False)
        font.setWeight(50)
        self.label_41.setFont(font)
        self.label_41.setObjectName(_fromUtf8("label_41"))
        self.verticalLayout_3.addWidget(self.label_41)
        self.antecedenteQ = QtGui.QPlainTextEdit(self.frame_7)
        self.antecedenteQ.setFrameShape(QtGui.QFrame.NoFrame)
        self.antecedenteQ.setReadOnly(True)
        self.antecedenteQ.setObjectName(_fromUtf8("antecedenteQ"))
        self.verticalLayout_3.addWidget(self.antecedenteQ)
        self.gridLayout.addWidget(self.frame_7, 3, 0, 2, 1)
        self.frame_6 = QtGui.QFrame(self.centralwidget)
        self.frame_6.setMinimumSize(QtCore.QSize(300, 0))
        self.frame_6.setMaximumSize(QtCore.QSize(320, 200))
        self.frame_6.setFrameShape(QtGui.QFrame.Box)
        self.frame_6.setFrameShadow(QtGui.QFrame.Sunken)
        self.frame_6.setObjectName(_fromUtf8("frame_6"))
        self.gridLayout_6 = QtGui.QGridLayout(self.frame_6)
        self.gridLayout_6.setHorizontalSpacing(6)
        self.gridLayout_6.setObjectName(_fromUtf8("gridLayout_6"))
        self.tratamiento = QtGui.QLineEdit(self.frame_6)
        self.tratamiento.setMinimumSize(QtCore.QSize(170, 0))
        self.tratamiento.setMaximumSize(QtCore.QSize(180, 16777215))
        font = QtGui.QFont()
        font.setFamily(_fromUtf8("Leelawadee UI"))
        font.setPointSize(11)
        self.tratamiento.setFont(font)
        self.tratamiento.setStyleSheet(_fromUtf8("background-color: rgb(255, 255,
255);"))
        self.tratamiento.setFrame(False)
        self.tratamiento.setReadOnly(True)
        self.tratamiento.setObjectName(_fromUtf8("tratamiento"))
        self.gridLayout_6.addWidget(self.tratamiento, 5, 1, 1, 1)
        self.fecha_a = QtGui.QLineEdit(self.frame_6)
        self.fecha_a.setMinimumSize(QtCore.QSize(170, 0))
        self.fecha_a.setMaximumSize(QtCore.QSize(180, 16777215))
        font = QtGui.QFont()
        font.setFamily(_fromUtf8("Leelawadee UI"))
        font.setPointSize(11)
        self.fecha_a.setFont(font)
        self.fecha_a.setStyleSheet(_fromUtf8("background-color: rgb(255, 255,
255);"))
        self.fecha_a.setFrame(False)
        self.fecha_a.setReadOnly(True)
        self.fecha_a.setObjectName(_fromUtf8("fecha_a"))
        self.gridLayout_6.addWidget(self.fecha_a, 4, 1, 1, 1)
        self.causa = QtGui.QLineEdit(self.frame_6)
        self.causa.setMinimumSize(QtCore.QSize(170, 0))
        self.causa.setMaximumSize(QtCore.QSize(180, 16777215))
        font = QtGui.QFont()
        font.setFamily(_fromUtf8("Leelawadee UI"))
        font.setPointSize(11)
        self.causa.setFont(font)
        self.causa.setStyleSheet(_fromUtf8("background-color: rgb(255, 255,
255);"))
        self.causa.setFrame(False)
        self.causa.setReadOnly(True)
        self.causa.setObjectName(_fromUtf8("causa"))
        self.gridLayout_6.addWidget(self.causa, 3, 1, 1, 1)
        self.label_34 = QtGui.QLabel(self.frame_6)
```

```
        font = QtGui.QFont()
        font.setBold(False)
        font.setWeight(50)
        self.label_34.setFont(font)
        self.label_34.setObjectName(_fromUtf8("label_34"))
        self.gridLayout_6.addWidget(self.label_34, 5, 0, 1, 1)
        self.label_23 = QtGui.QLabel(self.frame_6)
        font = QtGui.QFont()
        font.setBold(False)
        font.setWeight(50)
        self.label_23.setFont(font)
        self.label_23.setObjectName(_fromUtf8("label_23"))
        self.gridLayout_6.addWidget(self.label_23, 2, 0, 1, 1)
        self.label_36 = QtGui.QLabel(self.frame_6)
        font = QtGui.QFont()
        font.setBold(False)
        font.setWeight(50)
        self.label_36.setFont(font)
        self.label_36.setObjectName(_fromUtf8("label_36"))
        self.gridLayout_6.addWidget(self.label_36, 1, 0, 1, 1)
        self.label_38 = QtGui.QLabel(self.frame_6)
        font = QtGui.QFont()
        font.setBold(False)
        font.setWeight(50)
        self.label_38.setFont(font)
        self.label_38.setObjectName(_fromUtf8("label_38"))
        self.gridLayout_6.addWidget(self.label_38, 3, 0, 1, 1)
        self.amputacion = QtGui.QLineEdit(self.frame_6)
        self.amputacion.setMinimumSize(QtCore.QSize(170, 0))
        self.amputacion.setMaximumSize(QtCore.QSize(180, 16777215))
        font = QtGui.QFont()
        font.setFamily(_fromUtf8("Leelawadee UI"))
        font.setPointSize(11)
        self.amputacion.setFont(font)
        self.amputacion.setStyleSheet(_fromUtf8("background-color: rgb(255, 255,
255);"))
        self.amputacion.setFrame(False)
        self.amputacion.setReadOnly(True)
        self.amputacion.setObjectName(_fromUtf8("amputacion"))
        self.gridLayout_6.addWidget(self.amputacion, 2, 1, 1, 1)
        self.label_35 = QtGui.QLabel(self.frame_6)
        self.label_35.setMinimumSize(QtCore.QSize(0, 30))
        self.label_35.setMaximumSize(QtCore.QSize(16777215, 30))
        font = QtGui.QFont()
        font.setBold(True)
        font.setWeight(75)
        self.label_35.setFont(font)
        self.label_35.setStyleSheet(_fromUtf8("background-color: rgb(216, 213,
255);"))
        self.label_35.setFrameShape(QtGui.QFrame.Panel)
        self.label_35.setFrameShadow(QtGui.QFrame.Sunken)
        self.label_35.setAlignment(QtCore.Qt.AlignCenter)
        self.label_35.setObjectName(_fromUtf8("label_35"))
        self.gridLayout_6.addWidget(self.label_35, 0, 0, 1, 2)
        self.extremidad = QtGui.QLineEdit(self.frame_6)
        self.extremidad.setMinimumSize(QtCore.QSize(170, 0))
        self.extremidad.setMaximumSize(QtCore.QSize(180, 16777215))
        font = QtGui.QFont()
        font.setFamily(_fromUtf8("Leelawadee UI"))
        font.setPointSize(11)
        self.extremidad.setFont(font)
        self.extremidad.setStyleSheet(_fromUtf8("background-color: rgb(255, 255,
255);"))
        self.extremidad.setFrame(False)
        self.extremidad.setReadOnly(True)
        self.extremidad.setObjectName(_fromUtf8("extremidad"))
        self.gridLayout_6.addWidget(self.extremidad, 1, 1, 1, 1)
        self.label_37 = QtGui.QLabel(self.frame_6)
```

```
        font = QtGui.QFont()
        font.setBold(False)
        font.setWeight(50)
        self.label_37.setFont(font)
        self.label_37.setObjectName(_fromUtf8("label_37"))
        self.gridLayout_6.addWidget(self.label_37, 4, 0, 1, 1)
        self.gridLayout.addWidget(self.frame_6, 1, 2, 1, 1)
        self.frame_2 = QtGui.QFrame(self.centralwidget)
        self.frame_2.setMinimumSize(QtCore.QSize(300, 0))
        self.frame_2.setMaximumSize(QtCore.QSize(320, 16777215))
        self.frame_2.setFrameShape(QtGui.QFrame.StyledPanel)
        self.frame_2.setFrameShadow(QtGui.QFrame.Sunken)
        self.frame_2.setObjectName(_fromUtf8("frame_2"))
        self.gridLayout_3 = QtGui.QGridLayout(self.frame_2)
        self.gridLayout_3.setObjectName(_fromUtf8("gridLayout_3"))
        self.label_26 = QtGui.QLabel(self.frame_2)
        self.label_26.setMinimumSize(QtCore.QSize(0, 30))
        self.label_26.setMaximumSize(QtCore.QSize(16777215, 30))
        font = QtGui.QFont()
        font.setBold(True)
        font.setWeight(75)
        self.label_26.setFont(font)
        self.label_26.setStyleSheet(_fromUtf8("background-color: rgb(216, 213,
255);"))
        self.label_26.setFrameShape(QtGui.QFrame.Panel)
        self.label_26.setFrameShadow(QtGui.QFrame.Sunken)
        self.label_26.setAlignment(QtCore.Qt.AlignCenter)
        self.label_26.setObjectName(_fromUtf8("label_26"))
        self.gridLayout_3.addWidget(self.label_26, 0, 0, 1, 3)
        self.label = QtGui.QLabel(self.frame_2)
        font = QtGui.QFont()
        font.setBold(False)
        font.setWeight(50)
        self.label.setFont(font)
        self.label.setObjectName(_fromUtf8("label"))
        self.gridLayout_3.addWidget(self.label, 1, 0, 1, 1)
        self.label_17 = QtGui.QLabel(self.frame_2)
        font = QtGui.QFont()
        font.setBold(False)
        font.setWeight(50)
        self.label_17.setFont(font)
        self.label_17.setObjectName(_fromUtf8("label_17"))
        self.gridLayout_3.addWidget(self.label_17, 8, 0, 1, 1)
        self.label_12 = QtGui.QLabel(self.frame_2)
        font = QtGui.QFont()
        font.setBold(True)
        font.setWeight(75)
        self.label_12.setFont(font)
        self.label_12.setObjectName(_fromUtf8("label_12"))
        self.gridLayout_3.addWidget(self.label_12, 6, 0, 1, 1)
        self.semiten = QtGui.QProgressBar(self.frame_2)
        self.semiten.setMaximumSize(QtCore.QSize(50, 16777215))
        self.semiten.setProperty("value", 0)
        self.semiten.setAlignment(QtCore.Qt.AlignCenter)
        self.semiten.setObjectName(_fromUtf8("semiten"))
        self.gridLayout_3.addWidget(self.semiten, 9, 1, 1, 1)
        self.label_15 = QtGui.QLabel(self.frame_2)
        font = QtGui.QFont()
        font.setBold(True)
        font.setWeight(75)
        self.label_15.setFont(font)
        self.label_15.setObjectName(_fromUtf8("label_15"))
        self.gridLayout_3.addWidget(self.label_15, 12, 0, 1, 1)
        self.isquiotibiales = QtGui.QProgressBar(self.frame_2)
        self.isquiotibiales.setMaximumSize(QtCore.QSize(50, 16777215))
        self.isquiotibiales.setProperty("value", 0)
        self.isquiotibiales.setAlignment(QtCore.Qt.AlignCenter)
        self.isquiotibiales.setObjectName(_fromUtf8("isquiotibiales"))
```

```
        self.gridLayout_3.addWidget(self.isquiotibiales, 12, 1, 1, 1)
        self.recto = QtGui.QProgressBar(self.frame_2)
        self.recto.setMaximumSize(QtCore.QSize(50, 16777215))
        self.recto.setProperty("value", 0)
        self.recto.setAlignment(QtCore.Qt.AlignCenter)
        self.recto.setObjectName(_fromUtf8("recto"))
        self.gridLayout_3.addWidget(self.recto, 1, 1, 1, 1)
        self.label_14 = QtGui.QLabel(self.frame_2)
        font = QtGui.QFont()
        font.setBold(False)
        font.setWeight(50)
        self.label_14.setFont(font)
        self.label_14.setObjectName(_fromUtf8("label_14"))
        self.gridLayout_3.addWidget(self.label_14, 3, 0, 1, 1)
        self.label_16 = QtGui.QLabel(self.frame_2)
        font = QtGui.QFont()
        font.setBold(False)
        font.setWeight(50)
        self.label_16.setFont(font)
        self.label_16.setObjectName(_fromUtf8("label_16"))
        self.gridLayout_3.addWidget(self.label_16, 10, 0, 1, 1)
        self.label_13 = QtGui.QLabel(self.frame_2)
        font = QtGui.QFont()
        font.setBold(False)
        font.setWeight(50)
        self.label_13.setFont(font)
        self.label_13.setObjectName(_fromUtf8("label_13"))
        self.gridLayout_3.addWidget(self.label_13, 2, 0, 1, 1)
        self.cuadriceps = QtGui.QProgressBar(self.frame_2)
        self.cuadriceps.setMaximumSize(QtCore.QSize(50, 16777215))
        self.cuadriceps.setProperty("value", 0)
        self.cuadriceps.setAlignment(QtCore.Qt.AlignCenter)
        self.cuadriceps.setObjectName(_fromUtf8("cuadriceps"))
        self.gridLayout_3.addWidget(self.cuadriceps, 6, 1, 1, 1)
        spacerItem = QtGui.QSpacerItem(5, 5, QtGui.QSizePolicy.Minimum,
QtGui.QSizePolicy.Fixed)
        self.gridLayout_3.addItem(spacerItem, 7, 0, 1, 1)
        self.biceps = QtGui.QProgressBar(self.frame_2)
        self.biceps.setMaximumSize(QtCore.QSize(50, 16777215))
        self.biceps.setProperty("value", 0)
        self.biceps.setAlignment(QtCore.Qt.AlignCenter)
        self.biceps.setObjectName(_fromUtf8("biceps"))
        self.gridLayout_3.addWidget(self.biceps, 8, 1, 1, 1)
        self.vasto_M = QtGui.QProgressBar(self.frame_2)
        self.vasto_M.setMaximumSize(QtCore.QSize(50, 16777215))
        self.vasto_M.setProperty("value", 0)
        self.vasto_M.setAlignment(QtCore.Qt.AlignCenter)
        self.vasto_M.setObjectName(_fromUtf8("vasto_M"))
        self.gridLayout_3.addWidget(self.vasto_M, 3, 1, 1, 1)
        self.vasto_L = QtGui.QProgressBar(self.frame_2)
        self.vasto_L.setMaximumSize(QtCore.QSize(50, 16777215))
        self.vasto_L.setProperty("value", 0)
        self.vasto_L.setAlignment(QtCore.Qt.AlignCenter)
        self.vasto_L.setObjectName(_fromUtf8("vasto_L"))
        self.gridLayout_3.addWidget(self.vasto_L, 2, 1, 1, 1)
        self.semimem = QtGui.QProgressBar(self.frame_2)
        self.semimem.setMaximumSize(QtCore.QSize(50, 16777215))
        self.semimem.setProperty("value", 0)
        self.semimem.setAlignment(QtCore.Qt.AlignCenter)
        self.semimem.setObjectName(_fromUtf8("semimem"))
        self.gridLayout_3.addWidget(self.semimem, 10, 1, 1, 1)
        self.label_18 = QtGui.QLabel(self.frame_2)
        font = QtGui.QFont()
        font.setBold(False)
        font.setWeight(50)
        self.label_18.setFont(font)
        self.label_18.setObjectName(_fromUtf8("label_18"))
        self.gridLayout_3.addWidget(self.label_18, 9, 0, 1, 1)
```

```
        self.rectoP = QtGui.QLineEdit(self.frame_2)
        self.rectoP.setMaximumSize(QtCore.QSize(70, 16777215))
        font = QtGui.QFont()
        font.setFamily(_fromUtf8("Leelawadee UI"))
        font.setPointSize(11)
        self.rectoP.setFont(font)
        self.rectoP.setStyleSheet(_fromUtf8("background-color: rgb(255, 255,
255);"))
        self.rectoP.setFrame(False)
        self.rectoP.setReadOnly(True)
        self.rectoP.setObjectName(_fromUtf8("rectoP"))
        self.gridLayout_3.addWidget(self.rectoP, 1, 2, 1, 1)
        self.vasto_LP = QtGui.QLineEdit(self.frame_2)
        self.vasto_LP.setMaximumSize(QtCore.QSize(70, 16777215))
        font = QtGui.QFont()
        font.setFamily(_fromUtf8("Leelawadee UI"))
        font.setPointSize(11)
        self.vasto_LP.setFont(font)
        self.vasto_LP.setStyleSheet(_fromUtf8("background-color: rgb(255, 255,
255);"))
        self.vasto_LP.setFrame(False)
        self.vasto_LP.setReadOnly(True)
        self.vasto_LP.setObjectName(_fromUtf8("vasto_LP"))
        self.gridLayout_3.addWidget(self.vasto_LP, 2, 2, 1, 1)
        self.vasto_MP = QtGui.QLineEdit(self.frame_2)
        self.vasto_MP.setMaximumSize(QtCore.QSize(70, 16777215))
        font = QtGui.QFont()
        font.setFamily(_fromUtf8("Leelawadee UI"))
        font.setPointSize(11)
        self.vasto_MP.setFont(font)
        self.vasto_MP.setStyleSheet(_fromUtf8("background-color: rgb(255, 255,
255);"))
        self.vasto_MP.setFrame(False)
        self.vasto_MP.setReadOnly(True)
        self.vasto_MP.setObjectName(_fromUtf8("vasto_MP"))
        self.gridLayout_3.addWidget(self.vasto_MP, 3, 2, 1, 1)
        self.cuadricepsP = QtGui.QLineEdit(self.frame_2)
        self.cuadricepsP.setMaximumSize(QtCore.QSize(70, 16777215))
        font = QtGui.QFont()
        font.setFamily(_fromUtf8("Leelawadee UI"))
        font.setPointSize(11)
        self.cuadricepsP.setFont(font)
        self.cuadricepsP.setStyleSheet(_fromUtf8("background-color: rgb(255, 255,
255);"))
        self.cuadricepsP.setFrame(False)
        self.cuadricepsP.setReadOnly(True)
        self.cuadricepsP.setObjectName(_fromUtf8("cuadricepsP"))
        self.gridLayout_3.addWidget(self.cuadricepsP, 6, 2, 1, 1)
        self.bicepsP = QtGui.QLineEdit(self.frame_2)
        self.bicepsP.setMaximumSize(QtCore.QSize(70, 16777215))
        font = QtGui.QFont()
        font.setFamily(_fromUtf8("Leelawadee UI"))
        font.setPointSize(11)
        self.bicepsP.setFont(font)
        self.bicepsP.setStyleSheet(_fromUtf8("background-color: rgb(255, 255,
255);"))
        self.bicepsP.setFrame(False)
        self.bicepsP.setReadOnly(True)
        self.bicepsP.setObjectName(_fromUtf8("bicepsP"))
        self.gridLayout_3.addWidget(self.bicepsP, 8, 2, 1, 1)
        self.semitenP = QtGui.QLineEdit(self.frame_2)
        self.semitenP.setMaximumSize(QtCore.QSize(70, 16777215))
        font = QtGui.QFont()
        font.setFamily(_fromUtf8("Leelawadee UI"))
        font.setPointSize(11)
        self.semitenP.setFont(font)
        self.semitenP.setStyleSheet(_fromUtf8("background-color: rgb(255, 255,
255);"))
```

```
        self.semitenP.setFrame(False)
        self.semitenP.setReadOnly(True)
        self.semitenP.setObjectName(_fromUtf8("semitenP"))
        self.gridLayout_3.addWidget(self.semitenP, 9, 2, 1, 1)
        self.semimemP = QtGui.QLineEdit(self.frame_2)
        self.semimemP.setMaximumSize(QtCore.QSize(70, 16777215))
        font = QtGui.QFont()
        font.setFamily(_fromUtf8("Leelawadee UI"))
        font.setPointSize(11)
        self.semimemP.setFont(font)
        self.semimemP.setStyleSheet(_fromUtf8("background-color: rgb(255, 255,
255);"))
        self.semimemP.setFrame(False)
        self.semimemP.setReadOnly(True)
        self.semimemP.setObjectName(_fromUtf8("semimemP"))
        self.gridLayout_3.addWidget(self.semimemP, 10, 2, 1, 1)
        self.isquiotibialesP = QtGui.QLineEdit(self.frame_2)
        self.isquiotibialesP.setMaximumSize(QtCore.QSize(70, 16777215))
        font = QtGui.QFont()
        font.setFamily(_fromUtf8("Leelawadee UI"))
        font.setPointSize(11)
        self.isquiotibialesP.setFont(font)
        self.isquiotibialesP.setStyleSheet(_fromUtf8("background-color: rgb(255,
255, 255);"))
        self.isquiotibialesP.setFrame(False)
        self.isquiotibialesP.setReadOnly(True)
        self.isquiotibialesP.setObjectName(_fromUtf8("isquiotibialesP"))
        self.gridLayout_3.addWidget(self.isquiotibialesP, 12, 2, 1, 1)
        self.label.raise_()
        self.recto.raise_()
        self.label_12.raise_()
        self.cuadriceps.raise_()
        self.label_13.raise_()
        self.vasto_L.raise_()
        self.label_14.raise_()
        self.vasto_M.raise_()
        self.label_15.raise_()
        self.label_17.raise_()
        self.label_18.raise_()
        self.label_16.raise_()
        self.biceps.raise_()
        self.semiten.raise_()
        self.semimem.raise_()
        self.isquiotibiales.raise_()
        self.label_26.raise_()
        self.rectoP.raise_()
        self.vasto_LP.raise_()
        self.vasto_MP.raise_()
        self.cuadricepsP.raise_()
        self.bicepsP.raise_()
        self.semitenP.raise_()
        self.semimemP.raise_()
        self.isquiotibialesP.raise_()
        self.gridLayout.addWidget(self.frame_2, 2, 2, 2, 1)
        self.frame_5 = QtGui.QFrame(self.centralwidget)
        self.frame_5.setMinimumSize(QtCore.QSize(0, 150))
        self.frame_5.setMaximumSize(QtCore.QSize(16777215, 160))
        self.frame_5.setStyleSheet(_fromUtf8(""))
        self.frame_5.setFrameShape(QtGui.QFrame.Box)
        self.frame_5.setFrameShadow(QtGui.QFrame.Sunken)
        self.frame_5.setObjectName(_fromUtf8("frame_5"))
        self.gridLayout_2 = QtGui.QGridLayout(self.frame_5)
        self.gridLayout_2.setContentsMargins(9, 9, 9, -1)
        self.gridLayout_2.setObjectName(_fromUtf8("gridLayout_2"))
        self.procedencia = QtGui.QLineEdit(self.frame_5)
        self.procedencia.setMinimumSize(QtCore.QSize(100, 0))
        self.procedencia.setMaximumSize(QtCore.QSize(16777215, 16777215))
        font = QtGui.QFont()
```

```
        font.setFamily(_fromUtf8("Leelawadee UI"))
        font.setPointSize(11)
        self.procedencia.setFont(font)
        self.procedencia.setStyleSheet(_fromUtf8("background-color: rgb(255, 255,
255);"))
        self.procedencia.setFrame(False)
        self.procedencia.setReadOnly(True)
        self.procedencia.setObjectName(_fromUtf8("procedencia"))
        self.gridLayout_2.addWidget(self.procedencia, 5, 15, 1, 1)
        self.label_11 = QtGui.QLabel(self.frame_5)
        self.label_11.setMinimumSize(QtCore.QSize(90, 0))
        self.label_11.setMaximumSize(QtCore.QSize(50, 16777215))
        font = QtGui.QFont()
        font.setFamily(_fromUtf8("Leelawadee UI"))
        font.setPointSize(11)
        font.setBold(False)
        font.setWeight(50)
        self.label_11.setFont(font)
        self.label_11.setObjectName(_fromUtf8("label_11"))
        self.gridLayout_2.addWidget(self.label_11, 5, 14, 1, 1)
        self.label_9 = QtGui.QLabel(self.frame_5)
        self.label_9.setMinimumSize(QtCore.QSize(50, 0))
        self.label_9.setMaximumSize(QtCore.QSize(50, 16777215))
        font = QtGui.QFont()
        font.setFamily(_fromUtf8("Leelawadee UI"))
        font.setPointSize(11)
        font.setBold(False)
        font.setWeight(50)
        self.label_9.setFont(font)
        self.label_9.setObjectName(_fromUtf8("label_9"))
        self.gridLayout_2.addWidget(self.label_9, 5, 8, 1, 1)
        spacerItem1 = QtGui.QSpacerItem(20, 20, QtGui.QSizePolicy.Fixed,
QtGui.QSizePolicy.Minimum)
        self.gridLayout_2.addItem(spacerItem1, 3, 7, 1, 1)
        self.civil = QtGui.QLineEdit(self.frame_5)
        self.civil.setMinimumSize(QtCore.QSize(100, 0))
        self.civil.setMaximumSize(QtCore.QSize(16777215, 16777215))
        font = QtGui.QFont()
        font.setFamily(_fromUtf8("Leelawadee UI"))
        font.setPointSize(11)
        self.civil.setFont(font)
        self.civil.setStyleSheet(_fromUtf8("background-color: rgb(255, 255,
255);"))
        self.civil.setFrame(False)
        self.civil.setReadOnly(True)
        self.civil.setObjectName(_fromUtf8("civil"))
        self.gridLayout_2.addWidget(self.civil, 3, 15, 1, 1)
        self.nombre = QtGui.QLineEdit(self.frame_5)
        font = QtGui.QFont()
        font.setFamily(_fromUtf8("Leelawadee UI"))
        font.setPointSize(11)
        self.nombre.setFont(font)
        self.nombre.setStyleSheet(_fromUtf8("background-color: rgb(255, 255,
255);"))
        self.nombre.setFrame(False)
        self.nombre.setReadOnly(True)
        self.nombre.setObjectName(_fromUtf8("nombre"))
        self.gridLayout_2.addWidget(self.nombre, 3, 1, 1, 6)
        self.residencia = QtGui.QLineEdit(self.frame_5)
        self.residencia.setMinimumSize(QtCore.QSize(100, 0))
        font = QtGui.QFont()
        font.setFamily(_fromUtf8("Leelawadee UI"))
        font.setPointSize(11)
        self.residencia.setFont(font)
        self.residencia.setStyleSheet(_fromUtf8("background-color: rgb(255, 255,
255);"))
        self.residencia.setFrame(False)
        self.residencia.setReadOnly(True)
```

```
        self.residencia.setObjectName(_fromUtf8("residencia"))
        self.gridLayout_2.addWidget(self.residencia, 6, 1, 1, 6)
        self.label_32 = QtGui.QLabel(self.frame_5)
        self.label_32.setMaximumSize(QtCore.QSize(150, 16777215))
        font = QtGui.QFont()
        font.setFamily(_fromUtf8("Leelawadee UI"))
        font.setPointSize(11)
        font.setBold(False)
        font.setWeight(50)
        self.label_32.setFont(font)
        self.label_32.setObjectName(_fromUtf8("label_32"))
        self.gridLayout_2.addWidget(self.label_32, 6, 0, 1, 1)
        self.edad = QtGui.QLineEdit(self.frame_5)
        self.edad.setMinimumSize(QtCore.QSize(80, 0))
        self.edad.setMaximumSize(QtCore.QSize(80, 16777215))
        font = QtGui.QFont()
        font.setFamily(_fromUtf8("Leelawadee UI"))
        font.setPointSize(11)
        self.edad.setFont(font)
        self.edad.setStyleSheet(_fromUtf8("background-color: rgb(255, 255, 255);"))
        self.edad.setFrame(False)
        self.edad.setReadOnly(True)
        self.edad.setObjectName(_fromUtf8("edad"))
        self.gridLayout_2.addWidget(self.edad, 5, 6, 1, 1)
        self.ocupacion = QtGui.QLineEdit(self.frame_5)
        self.ocupacion.setMinimumSize(QtCore.QSize(80, 0))
        self.ocupacion.setMaximumSize(QtCore.QSize(16777215, 16777215))
        font = QtGui.QFont()
        font.setFamily(_fromUtf8("Leelawadee UI"))
        font.setPointSize(11)
        self.ocupacion.setFont(font)
        self.ocupacion.setStyleSheet(_fromUtf8("background-color: rgb(255, 255,
255);"))
        self.ocupacion.setFrame(False)
        self.ocupacion.setReadOnly(True)
        self.ocupacion.setObjectName(_fromUtf8("ocupacion"))
        self.gridLayout_2.addWidget(self.ocupacion, 6, 11, 1, 1)
        self.label_30 = QtGui.QLabel(self.frame_5)
        self.label_30.setMinimumSize(QtCore.QSize(40, 0))
        self.label_30.setMaximumSize(QtCore.QSize(40, 16777215))
        font = QtGui.QFont()
        font.setFamily(_fromUtf8("Leelawadee UI"))
        font.setPointSize(11)
        font.setBold(False)
        font.setWeight(50)
        self.label_30.setFont(font)
        self.label_30.setObjectName(_fromUtf8("label_30"))
        self.gridLayout_2.addWidget(self.label_30, 3, 18, 1, 1)
        self.label_10 = QtGui.QLabel(self.frame_5)
        self.label_10.setMaximumSize(QtCore.QSize(150, 16777215))
        font = QtGui.QFont()
        font.setFamily(_fromUtf8("Leelawadee UI"))
        font.setPointSize(11)
        font.setBold(False)
        font.setWeight(50)
        self.label_10.setFont(font)
        self.label_10.setObjectName(_fromUtf8("label_10"))
        self.gridLayout_2.addWidget(self.label_10, 5, 0, 1, 1)
        self.ingreso = QtGui.QLineEdit(self.frame_5)
        self.ingreso.setMinimumSize(QtCore.QSize(100, 0))
        self.ingreso.setMaximumSize(QtCore.QSize(16777215, 16777215))
        font = QtGui.QFont()
        font.setFamily(_fromUtf8("Leelawadee UI"))
        font.setPointSize(11)
        self.ingreso.setFont(font)
        self.ingreso.setStyleSheet(_fromUtf8("background-color: rgb(255, 255,
255);"))
        self.ingreso.setFrame(False)
```

```
        self.ingreso.setReadOnly(True)
        self.ingreso.setObjectName(_fromUtf8("ingreso"))
        self.gridLayout_2.addWidget(self.ingreso, 6, 15, 1, 1)
        self.IMC = QtGui.QLineEdit(self.frame_5)
        self.IMC.setMaximumSize(QtCore.QSize(70, 16777215))
        font = QtGui.QFont()
        font.setFamily(_fromUtf8("Leelawadee UI"))
        font.setPointSize(11)
        self.IMC.setFont(font)
        self.IMC.setStyleSheet(_fromUtf8("background-color: rgb(255, 255, 255);"))
        self.IMC.setFrame(False)
        self.IMC.setReadOnly(True)
        self.IMC.setObjectName(_fromUtf8("IMC"))
        self.gridLayout_2.addWidget(self.IMC, 6, 19, 1, 1)
        spacerItem2 = QtGui.QSpacerItem(15, 20, QtGui.QSizePolicy.Fixed,
QtGui.QSizePolicy.Minimum)
        self.gridLayout_2.addItem(spacerItem2, 5, 4, 1, 1)
        self.label_4 = QtGui.QLabel(self.frame_5)
        self.label_4.setMinimumSize(QtCore.QSize(50, 0))
        self.label_4.setMaximumSize(QtCore.QSize(50, 16777215))
        font = QtGui.QFont()
        font.setFamily(_fromUtf8("Leelawadee UI"))
        font.setPointSize(11)
        font.setBold(False)
        font.setWeight(50)
        self.label_4.setFont(font)
        self.label_4.setObjectName(_fromUtf8("label_4"))
        self.gridLayout_2.addWidget(self.label_4, 3, 8, 1, 3)
        self.label_7 = QtGui.QLabel(self.frame_5)
        self.label_7.setMaximumSize(QtCore.QSize(150, 16777215))
        font = QtGui.QFont()
        font.setFamily(_fromUtf8("Leelawadee UI"))
        font.setPointSize(11)
        font.setBold(False)
        font.setWeight(50)
        self.label_7.setFont(font)
        self.label_7.setObjectName(_fromUtf8("label_7"))
        self.gridLayout_2.addWidget(self.label_7, 3, 0, 1, 1)
        self.label_19 = QtGui.QLabel(self.frame_5)
        self.label_19.setMinimumSize(QtCore.QSize(40, 0))
        self.label_19.setMaximumSize(QtCore.QSize(40, 16777215))
        font = QtGui.QFont()
        font.setFamily(_fromUtf8("Leelawadee UI"))
        font.setPointSize(11)
        font.setBold(False)
        font.setWeight(50)
        self.label_19.setFont(font)
        self.label_19.setObjectName(_fromUtf8("label_19"))
        self.gridLayout_2.addWidget(self.label_19, 5, 18, 1, 1)
        self.label_33 = QtGui.QLabel(self.frame_5)
        self.label_33.setMinimumSize(QtCore.QSize(90, 0))
        self.label_33.setMaximumSize(QtCore.QSize(50, 16777215))
        font = QtGui.QFont()
        font.setFamily(_fromUtf8("Leelawadee UI"))
        font.setPointSize(11)
        font.setBold(False)
        font.setWeight(50)
        self.label_33.setFont(font)
        self.label_33.setObjectName(_fromUtf8("label_33"))
        self.gridLayout_2.addWidget(self.label_33, 6, 14, 1, 1)
        self.label_8 = QtGui.QLabel(self.frame_5)
        self.label_8.setMinimumSize(QtCore.QSize(50, 0))
        self.label_8.setMaximumSize(QtCore.QSize(100, 16777215))
        font = QtGui.QFont()
        font.setFamily(_fromUtf8("Leelawadee UI"))
        font.setPointSize(11)
        font.setBold(False)
        font.setWeight(50)
```

```
        self.label_8.setFont(font)
        self.label_8.setObjectName(_fromUtf8("label_8"))
        self.gridLayout_2.addWidget(self.label_8, 6, 8, 1, 1)
        self.label_20 = QtGui.QLabel(self.frame_5)
        self.label_20.setMinimumSize(QtCore.QSize(50, 0))
        self.label_20.setMaximumSize(QtCore.QSize(50, 16777215))
        font = QtGui.QFont()
        font.setFamily(_fromUtf8("Leelawadee UI"))
        font.setPointSize(11)
        font.setBold(False)
        font.setWeight(50)
        self.label_20.setFont(font)
        self.label_20.setObjectName(_fromUtf8("label_20"))
        self.gridLayout_2.addWidget(self.label_20, 6, 18, 1, 1)
        self.label_29 = QtGui.QLabel(self.frame_5)
        self.label_29.setMinimumSize(QtCore.QSize(50, 0))
        self.label_29.setMaximumSize(QtCore.QSize(30, 16777215))
        font = QtGui.QFont()
        font.setFamily(_fromUtf8("Leelawadee UI"))
        font.setPointSize(11)
        font.setBold(False)
        font.setWeight(50)
        self.label_29.setFont(font)
        self.label_29.setObjectName(_fromUtf8("label_29"))
        self.gridLayout_2.addWidget(self.label_29, 5, 5, 1, 1)
        self.talla = QtGui.QLineEdit(self.frame_5)
        self.talla.setMaximumSize(QtCore.QSize(70, 16777215))
        font = QtGui.QFont()
        font.setFamily(_fromUtf8("Leelawadee UI"))
        font.setPointSize(11)
        self.talla.setFont(font)
        self.talla.setStyleSheet(_fromUtf8("background-color: rgb(255, 255,
255);"))
        self.talla.setFrame(False)
        self.talla.setReadOnly(True)
        self.talla.setObjectName(_fromUtf8("talla"))
        self.gridLayout_2.addWidget(self.talla, 5, 19, 1, 1)
        self.nacionalidad = QtGui.QLineEdit(self.frame_5)
        self.nacionalidad.setMinimumSize(QtCore.QSize(200, 0))
        font = QtGui.QFont()
        font.setFamily(_fromUtf8("Leelawadee UI"))
        font.setPointSize(11)
        self.nacionalidad.setFont(font)
        self.nacionalidad.setStyleSheet(_fromUtf8("background-color: rgb(255, 255,
255);"))
        self.nacionalidad.setFrame(False)
        self.nacionalidad.setReadOnly(True)
        self.nacionalidad.setObjectName(_fromUtf8("nacionalidad"))
        self.gridLayout_2.addWidget(self.nacionalidad, 5, 1, 1, 3)
        self.label_28 = QtGui.QLabel(self.frame_5)
        self.label_28.setMinimumSize(QtCore.QSize(90, 0))
        self.label_28.setMaximumSize(QtCore.QSize(30, 16777215))
        font = QtGui.QFont()
        font.setFamily(_fromUtf8("Leelawadee UI"))
        font.setPointSize(11)
        font.setBold(False)
        font.setWeight(50)
        self.label_28.setFont(font)
        self.label_28.setObjectName(_fromUtf8("label_28"))
        self.gridLayout_2.addWidget(self.label_28, 3, 14, 1, 1)
        self.peso = QtGui.QLineEdit(self.frame_5)
        self.peso.setMaximumSize(QtCore.QSize(70, 16777215))
        font = QtGui.QFont()
        font.setFamily(_fromUtf8("Leelawadee UI"))
        font.setPointSize(11)
        self.peso.setFont(font)
        self.peso.setStyleSheet(_fromUtf8("background-color: rgb(255, 255, 255);"))
        self.peso.setFrame(False)
```

```
        self.peso.setReadOnly(True)
        self.peso.setObjectName(_fromUtf8("peso"))
        self.gridLayout_2.addWidget(self.peso, 3, 19, 1, 1)
        spacerItem3 = QtGui.QSpacerItem(20, 20, QtGui.QSizePolicy.Fixed,
QtGui.QSizePolicy.Minimum)
        self.gridLayout_2.addItem(spacerItem3, 3, 16, 1, 1)
        self.ci = QtGui.QLineEdit(self.frame_5)
        self.ci.setMinimumSize(QtCore.QSize(80, 0))
        self.ci.setMaximumSize(QtCore.QSize(16777215, 16777215))
        font = QtGui.QFont()
        font.setFamily(_fromUtf8("Leelawadee UI"))
        font.setPointSize(11)
        self.ci.setFont(font)
        self.ci.setStyleSheet(_fromUtf8("background-color: rgb(255, 255, 255);"))
        self.ci.setFrame(False)
        self.ci.setReadOnly(True)
        self.ci.setObjectName(_fromUtf8("ci"))
        self.gridLayout_2.addWidget(self.ci, 3, 11, 1, 1)
        self.sexo = QtGui.QLineEdit(self.frame_5)
        self.sexo.setMinimumSize(QtCore.QSize(80, 0))
        self.sexo.setMaximumSize(QtCore.QSize(16777215, 16777215))
        font = QtGui.QFont()
        font.setFamily(_fromUtf8("Leelawadee UI"))
        font.setPointSize(11)
        self.sexo.setFont(font)
        self.sexo.setStyleSheet(_fromUtf8("background-color: rgb(255, 255, 255);"))
        self.sexo.setFrame(False)
        self.sexo.setReadOnly(True)
        self.sexo.setObjectName(_fromUtf8("sexo"))
        self.gridLayout_2.addWidget(self.sexo, 5, 11, 1, 1)
        spacerItem4 = QtGui.QSpacerItem(20, 20, QtGui.QSizePolicy.Fixed,
QtGui.QSizePolicy.Minimum)
        self.gridLayout_2.addItem(spacerItem4, 3, 12, 1, 2)
        self.label_31 = QtGui.QLabel(self.frame_5)
        self.label_31.setMinimumSize(QtCore.QSize(40, 0))
        self.label_31.setMaximumSize(QtCore.QSize(40, 16777215))
        font = QtGui.QFont()
        font.setFamily(_fromUtf8("Leelawadee UI"))
        font.setPointSize(11)
        font.setBold(False)
        font.setWeight(50)
        self.label_31.setFont(font)
        self.label_31.setObjectName(_fromUtf8("label_31"))
        self.gridLayout_2.addWidget(self.label_31, 3, 20, 1, 1)
        self.label_39 = QtGui.QLabel(self.frame_5)
        self.label_39.setMinimumSize(QtCore.QSize(40, 0))
        self.label_39.setMaximumSize(QtCore.QSize(40, 16777215))
        font = QtGui.QFont()
        font.setFamily(_fromUtf8("Leelawadee UI"))
        font.setPointSize(11)
        font.setBold(False)
        font.setWeight(50)
        self.label_39.setFont(font)
        self.label_39.setObjectName(_fromUtf8("label_39"))
        self.gridLayout_2.addWidget(self.label_39, 5, 20, 1, 1)
        self.frame = QtGui.QFrame(self.frame_5)
        self.frame.setMinimumSize(QtCore.QSize(0, 30))
        self.frame.setMaximumSize(QtCore.QSize(16777215, 50))
        self.frame.setStyleSheet(_fromUtf8("background-color: rgb(216, 213,
255);"))
        self.frame.setFrameShape(QtGui.QFrame.Panel)
        self.frame.setFrameShadow(QtGui.QFrame.Sunken)
        self.frame.setObjectName(_fromUtf8("frame"))
        self.gridLayout_5 = QtGui.QGridLayout(self.frame)
        self.gridLayout_5.setContentsMargins(20, 6, 20, 6)
        self.gridLayout_5.setObjectName(_fromUtf8("gridLayout_5"))
        self.HC = QtGui.QLabel(self.frame)
        self.HC.setMaximumSize(QtCore.QSize(200, 16777215))
```

```
        font = QtGui.QFont()
        font.setFamily(_fromUtf8("Leelawadee UI"))
        font.setPointSize(11)
        font.setBold(True)
        font.setWeight(75)
        self.HC.setFont(font)
        self.HC.setObjectName(_fromUtf8("HC"))
        self.gridLayout_5.addWidget(self.HC, 0, 2, 2, 1)
        self.label_6 = QtGui.QLabel(self.frame)
        self.label_6.setMinimumSize(QtCore.QSize(0, 30))
        self.label_6.setMaximumSize(QtCore.QSize(140, 50))
        self.label_6.setText(_fromUtf8(""))
        self.label_6.setPixmap(QtGui.QPixmap(_fromUtf8("LOGO.png")))
        self.label_6.setScaledContents(True)
        self.label_6.setObjectName(_fromUtf8("label_6"))
        self.gridLayout_5.addWidget(self.label_6, 0, 0, 2, 1)
        self.label_27 = QtGui.QLabel(self.frame)
        font = QtGui.QFont()
        font.setFamily(_fromUtf8("Leelawadee UI"))
        font.setPointSize(11)
        font.setBold(True)
        font.setWeight(75)
        self.label_27.setFont(font)
        self.label_27.setStyleSheet(_fromUtf8(""))
        self.label_27.setAlignment(QtCore.Qt.AlignCenter)
        self.label_27.setObjectName(_fromUtf8("label_27"))
        self.gridLayout_5.addWidget(self.label_27, 1, 1, 1, 1)
        self.label_3 = QtGui.QLabel(self.frame)
        font = QtGui.QFont()
        font.setFamily(_fromUtf8("Leelawadee UI"))
        font.setPointSize(11)
        font.setBold(True)
        font.setWeight(75)
        self.label_3.setFont(font)
        self.label_3.setStyleSheet(_fromUtf8("Color:rgb(0, 107, 161)"))
        self.label_3.setAlignment(QtCore.Qt.AlignCenter)
        self.label_3.setObjectName(_fromUtf8("label_3"))
        self.gridLayout_5.addWidget(self.label_3, 0, 1, 1, 1)
        self.gridLayout_2.addWidget(self.frame, 0, 0, 1, 21)
        self.gridLayout.addWidget(self.frame_5, 0, 0, 1, 3)
        self.frame_3 = QtGui.QFrame(self.centralwidget)
        self.frame_3.setMinimumSize(QtCore.QSize(300, 0))
        self.frame_3.setMaximumSize(QtCore.QSize(300, 300))
        self.frame_3.setFrameShape(QtGui.QFrame.Box)
        self.frame_3.setFrameShadow(QtGui.QFrame.Sunken)
        self.frame_3.setObjectName(_fromUtf8("frame_3"))
        self.verticalLayout = QtGui.QVBoxLayout(self.frame_3)
        self.verticalLayout.setObjectName(_fromUtf8("verticalLayout"))
        self.label_25 = QtGui.QLabel(self.frame_3)
        self.label_25.setMinimumSize(QtCore.QSize(0, 30))
        self.label_25.setMaximumSize(QtCore.QSize(16777215, 30))
        font = QtGui.QFont()
        font.setBold(True)
        font.setWeight(75)
        self.label_25.setFont(font)
        self.label_25.setStyleSheet(_fromUtf8("background-color: rgb(216, 213,
255);"))
        self.label_25.setFrameShape(QtGui.QFrame.Panel)
        self.label_25.setFrameShadow(QtGui.QFrame.Sunken)
        self.label_25.setAlignment(QtCore.Qt.AlignCenter)
        self.label_25.setObjectName(_fromUtf8("label_25"))
        self.verticalLayout.addWidget(self.label_25)
        self.label_2 = QtGui.QLabel(self.frame_3)
        font = QtGui.QFont()
        font.setBold(False)
        font.setWeight(50)
        self.label_2.setFont(font)
        self.label_2.setObjectName(_fromUtf8("label_2"))
```

```
        self.verticalLayout.addWidget(self.label_2)
        self.enfermedad = QtGui.QPlainTextEdit(self.frame_3)
        self.enfermedad.setFrameShape(QtGui.QFrame.NoFrame)
        self.enfermedad.setReadOnly(True)
        self.enfermedad.setObjectName(_fromUtf8("enfermedad"))
        self.verticalLayout.addWidget(self.enfermedad)
        self.label_5 = QtGui.QLabel(self.frame_3)
        font = QtGui.QFont()
        font.setBold(False)
        font.setWeight(50)
        self.label_5.setFont(font)
        self.label_5.setObjectName(_fromUtf8("label_5"))
        self.verticalLayout.addWidget(self.label_5)
        self.medios = QtGui.QPlainTextEdit(self.frame_3)
        self.medios.setFrameShape(QtGui.QFrame.NoFrame)
        self.medios.setReadOnly(True)
        self.medios.setObjectName(_fromUtf8("medios"))
        self.verticalLayout.addWidget(self.medios)
        self.gridLayout.addWidget(self.frame_3, 1, 0, 2, 1)
        self.frame_4 = QtGui.QFrame(self.centralwidget)
        self.frame_4.setMaximumSize(QtCore.QSize(16777215, 40))
        self.frame_4.setFrameShape(QtGui.QFrame.StyledPanel)
        self.frame_4.setFrameShadow(QtGui.QFrame.Raised)
        self.frame_4.setObjectName(_fromUtf8("frame_4"))
        self.horizontalLayout_3 = QtGui.QHBoxLayout(self.frame_4)
        self.horizontalLayout_3.setContentsMargins(-1, 0, -1, 0)
        self.horizontalLayout_3.setObjectName(_fromUtf8("horizontalLayout_3"))
        self.ayuda = QtGui.QPushButton(self.frame_4)
        self.ayuda.setText(_fromUtf8(""))
        icon = QtGui.QIcon()
        icon.addPixmap(QtGui.QPixmap(_fromUtf8("ayuda.png")), QtGui.QIcon.Normal,
QtGui.QIcon.Off)
        self.ayuda.setIcon(icon)
        self.ayuda.setIconSize(QtCore.QSize(25, 25))
        self.ayuda.setFlat(True)
        self.ayuda.setObjectName(_fromUtf8("ayuda"))
        self.horizontalLayout_3.addWidget(self.ayuda)
        self.btn_imprimir = QtGui.QPushButton(self.frame_4)
        self.btn_imprimir.setMinimumSize(QtCore.QSize(0, 25))
        icon1 = QtGui.QIcon()
        icon1.addPixmap(QtGui.QPixmap(_fromUtf8("imprimir.png")),
QtGui.QIcon.Normal, QtGui.QIcon.Off)
        self.btn_imprimir.setIcon(icon1)
        self.btn_imprimir.setIconSize(QtCore.QSize(20, 20))
        self.btn_imprimir.setFlat(False)
        self.btn_imprimir.setObjectName(_fromUtf8("btn_imprimir"))
        self.horizontalLayout_3.addWidget(self.btn_imprimir)
        self.btn_aceptar = QtGui.QPushButton(self.frame_4)
        self.btn_aceptar.setMinimumSize(QtCore.QSize(0, 25))
        self.btn_aceptar.setObjectName(_fromUtf8("btn_aceptar"))
        self.horizontalLayout_3.addWidget(self.btn_aceptar)
        self.gridLayout.addWidget(self.frame_4, 4, 2, 1, 1)
        self.Grafica = QtGui.QVBoxLayout()
        self.Grafica.setObjectName(_fromUtf8("Grafica"))
        self.gridLayout.addLayout(self.Grafica, 1, 1, 4, 1)
        Diagnostico.setCentralWidget(self.centralwidget)
        self.statusbar = QtGui.QStatusBar(Diagnostico)
        self.statusbar.setObjectName(_fromUtf8("statusbar"))
        Diagnostico.setStatusBar(self.statusbar)

        self.retranslateUi(Diagnostico)
        QtCore.QMetaObject.connectSlotsByName(Diagnostico)

    def retranslateUi(self, Diagnostico):
        Diagnostico.setWindowTitle(_translate("Diagnostico", "HISTORIA CLINICA",
None))
        self.label_40.setText(_translate("Diagnostico", "ANTECENDENTES", None))
        self.label_24.setText(_translate("Diagnostico", "Antecedente Patológico
```

```
Personal:", None))
        self.label_41.setText(_translate("Diagnostico", "Antecendente Patológico
Quirurgico:", None))
        self.label_34.setText(_translate("Diagnostico", "Tratamiento PosO.:",
None))
        self.label_23.setText(_translate("Diagnostico", "Amputación:", None))
        self.label_36.setText(_translate("Diagnostico", "Extremidad:", None))
        self.label_38.setText(_translate("Diagnostico", "Causa:", None))
        self.label_35.setText(_translate("Diagnostico", "EXTREMIDAD AMPUTADA",
None))
        self.label_37.setText(_translate("Diagnostico", "Fecha de A.:", None))
        self.label_26.setText(_translate("Diagnostico", "RESULTADOS EXTRAMIDAD
AMPUTADA", None))
        self.label.setText(_translate("Diagnostico", "Fuerza Recto Femoral:",
None))
        self.label_17.setText(_translate("Diagnostico", "Fuerza Bíceps Femoral:",
None))
        self.label_12.setText(_translate("Diagnostico", "Prom. extensión rodilla:",
None))
        self.label_15.setText(_translate("Diagnostico", "Fuerza Isquiotibiales:",
None))
        self.label_14.setText(_translate("Diagnostico", "Fuerza Vasto Medial:",
None))
        self.label_16.setText(_translate("Diagnostico", "Fuerza Semimembranoso:",
None))
        self.label_13.setText(_translate("Diagnostico", "Fuerza Vasto Lateral:",
None))
        self.label_18.setText(_translate("Diagnostico", "Fuerza Semitendinoso:",
None))
        self.label_11.setText(_translate("Diagnostico", "Procedencia:", None))
        self.label_9.setText(_translate("Diagnostico", "Sexo:", None))
        self.label_32.setText(_translate("Diagnostico", "Residencia Actual:",
None))
        self.label_30.setText(_translate("Diagnostico", "Peso:", None))
        self.label_10.setText(_translate("Diagnostico", "Nacionalidad:", None))
        self.label_4.setText(_translate("Diagnostico", "CI:", None))
        self.label_7.setText(_translate("Diagnostico", "Nombre:", None))
        self.label_19.setText(_translate("Diagnostico", "Talla:", None))
        self.label_33.setText(_translate("Diagnostico", "Ingreso:", None))
        self.label_8.setText(_translate("Diagnostico", "Ocupación:", None))
        self.label_20.setText(_translate("Diagnostico", "IMC:", None))
        self.label_29.setText(_translate("Diagnostico", "Edad:", None))
        self.label_28.setText(_translate("Diagnostico", "Estado Civil:", None))
        self.label_31.setText(_translate("Diagnostico", "Kg", None))
        self.label_39.setText(_translate("Diagnostico", "m", None))
        self.HC.setText(_translate("Diagnostico", "# de Historia:", None))
        self.label_27.setText(_translate("Diagnostico", "DATOS GENERALES DEL
PACIENTE", None))
        self.label_3.setText(_translate("Diagnostico", "EVALUACION DE LA FUERZA
MUSCULAR EN PERSONAS AMPUTADAS", None))
        self.label_25.setText(_translate("Diagnostico", "ACTUAL:", None))
        self.label_2.setText(_translate("Diagnostico", "Enfermedad Actual:", None))
        self.label_5.setText(_translate("Diagnostico", "Medios Diagnósticos:",
None))
        self.btn_imprimir.setText(_translate("Diagnostico", "Imprimir", None))
        self.btn_aceptar.setText(_translate("Diagnostico", "Aceptar", None))
```

5) Código de la ventana de ayuda para la pantalla Principal

```
# -*- coding: utf-8 -*-

# Form implementation generated from reading ui file 'ayudaP.ui'
#
# Created by: PyQt4 UI code generator 4.11.4
#
# WARNING! All changes made in this file will be lost!

from PyQt4 import QtCore, QtGui

try:
    _fromUtf8 = QtCore.QString.fromUtf8
except AttributeError:
    def _fromUtf8(s):
        return s

try:
    _encoding = QtGui.QApplication.UnicodeUTF8
    def _translate(context, text, disambig):
        return QtGui.QApplication.translate(context, text, disambig, _encoding)
except AttributeError:
    def _translate(context, text, disambig):
        return QtGui.QApplication.translate(context, text, disambig)

class Ui_ayudaP(object):
    def setupUi(self, ayudaP):
        ayudaP.setObjectName(_fromUtf8("ayudaP"))
        ayudaP.resize(728, 504)
        font = QtGui.QFont()
        font.setFamily(_fromUtf8("Leelawadee UI"))
        font.setPointSize(11)
        ayudaP.setFont(font)
        self.verticalLayout = QtGui.QVBoxLayout(ayudaP)
        self.verticalLayout.setObjectName(_fromUtf8("verticalLayout"))
        self.plainTextEdit = QtGui.QPlainTextEdit(ayudaP)
        font = QtGui.QFont()
        font.setFamily(_fromUtf8("Leelawadee UI"))
        font.setPointSize(11)
        self.plainTextEdit.setFont(font)
        self.plainTextEdit.setStyleSheet(_fromUtf8("background-color: rgb(216, 213,
255);"))
        self.plainTextEdit.setFrameShape(QtGui.QFrame.NoFrame)
        self.plainTextEdit.setTabChangesFocus(False)
        self.plainTextEdit.setUndoRedoEnabled(True)
        self.plainTextEdit.setReadOnly(True)
        self.plainTextEdit.setBackgroundVisible(False)
        self.plainTextEdit.setCenterOnScroll(False)
        self.plainTextEdit.setObjectName(_fromUtf8("plainTextEdit"))
        self.verticalLayout.addWidget(self.plainTextEdit)

        self.retranslateUi(ayudaP)
        QtCore.QMetaObject.connectSlotsByName(ayudaP)

    def retranslateUi(self, ayudaP):
        ayudaP.setWindowTitle(_translate("ayudaP", "AYUDA", None))
        self.plainTextEdit.setPlainText(_translate("ayudaP", "REGISTRO DE UN
PACIENTE\n"
"Haga clic en el botón Registro y se abrirá la pantalla de registro.\n"
"\n"
"EVALUACION DE LA EXTREMIDAD AMPUTADA\n"
"Seleccione el paciente del que desee realizar la evaluación y haga clic en el
botón evaluación.\n"
"\n"
"ACCEDER A LA HISTORIA CLINICA\n"
```

```
"Seleccione el paciente del que desee visualizar la información y haga clic en el
botón Historia Clínica. Se abrirá la pantalla historia clínica.\n"
"\n"
"BUSCAR UN PACIENTE\n"
"Ingrese un nombre o las iniciales y haga clic en buscar. O ingrese la fecha de la
última visita y haga clic en buscar.\n"
"\n"
"EDITAR INFORMACION DEL PACIENTE\n"
"Seleccione el paciente del que desea cambiar la información. Se abrirá la pantalla
de registro con la información anteriormente ingresada.\n"
"ADVERTENCIA: No se puede editar el número de cédula de un paciente. Si desea
hacerlo, deberá eliminar al paciente y volver a registrarlo.\n"
"\n"
"ELIMINAR LA INFORMACION DE UN PACIENTE\n"
"Seleccione el paciente del que desea eliminar la información y haga clic en el
botón eliminar. Se mostrará un mensaje de confirmación, presione aceptar.", None))
```

6) Código de la ventana de ayuda para la pantalla de Evaluación

```
# -*- coding: utf-8 -*-

# Form implementation generated from reading ui file 'ayudaE.ui'
#
# Created by: PyQt4 UI code generator 4.11.4
#
# WARNING! All changes made in this file will be lost!

from PyQt4 import QtCore, QtGui

try:
    _fromUtf8 = QtCore.QString.fromUtf8
except AttributeError:
    def _fromUtf8(s):
        return s

try:
    _encoding = QtGui.QApplication.UnicodeUTF8
    def _translate(context, text, disambig):
        return QtGui.QApplication.translate(context, text, disambig, _encoding)
except AttributeError:
    def _translate(context, text, disambig):
        return QtGui.QApplication.translate(context, text, disambig)

class Ui_ayudaE(object):
    def setupUi(self, ayudaE):
        ayudaE.setObjectName(_fromUtf8("ayudaE"))
        ayudaE.resize(705, 623)
        font = QtGui.QFont()
        font.setFamily(_fromUtf8("Leelawadee UI"))
        font.setPointSize(11)
        ayudaE.setFont(font)
        self.verticalLayout = QtGui.QVBoxLayout(ayudaE)
        self.verticalLayout.setSpacing(15)
        self.verticalLayout.setObjectName(_fromUtf8("verticalLayout"))
        self.plainTextEdit = QtGui.QPlainTextEdit(ayudaE)
        self.plainTextEdit.setMaximumSize(QtCore.QSize(16777215, 260))
        font = QtGui.QFont()
        font.setFamily(_fromUtf8("Leelawadee UI"))
        font.setPointSize(11)
        self.plainTextEdit.setFont(font)
        self.plainTextEdit.setAutoFillBackground(False)
        self.plainTextEdit.setStyleSheet(_fromUtf8("background-color: rgb(216, 213,
255);"))
```

```
        self.plainTextEdit.setFrameShape(QtGui.QFrame.NoFrame)
        self.plainTextEdit.setTabChangesFocus(False)
        self.plainTextEdit.setUndoRedoEnabled(True)
        self.plainTextEdit.setReadOnly(True)
        self.plainTextEdit.setBackgroundVisible(False)
        self.plainTextEdit.setCenterOnScroll(False)
        self.plainTextEdit.setObjectName(_fromUtf8("plainTextEdit"))
        self.verticalLayout.addWidget(self.plainTextEdit)
        self.tableWidget = QtGui.QTableWidget(ayudaE)
        font = QtGui.QFont()
        font.setFamily(_fromUtf8("Leelawadee UI"))
        font.setPointSize(11)
        self.tableWidget.setFont(font)
        self.tableWidget.setFrameShape(QtGui.QFrame.NoFrame)
        self.tableWidget.setEditTriggers(QtGui.QAbstractItemView.AnyKeyPressed)
        self.tableWidget.setRowCount(5)
        self.tableWidget.setObjectName(_fromUtf8("tableWidget"))
        self.tableWidget.setColumnCount(2)
        item = QtGui.QTableWidgetItem()
        font = QtGui.QFont()
        font.setFamily(_fromUtf8("Leelawadee UI"))
        font.setPointSize(11)
        font.setBold(True)
        font.setWeight(75)
        item.setFont(font)
        self.tableWidget.setHorizontalHeaderItem(0, item)
        item = QtGui.QTableWidgetItem()
        font = QtGui.QFont()
        font.setFamily(_fromUtf8("Leelawadee UI"))
        font.setPointSize(11)
        font.setBold(True)
        font.setWeight(75)
        item.setFont(font)
        self.tableWidget.setHorizontalHeaderItem(1, item)
        item = QtGui.QTableWidgetItem()
        self.tableWidget.setItem(0, 0, item)
        item = QtGui.QTableWidgetItem()
        self.tableWidget.setItem(0, 1, item)
        item = QtGui.QTableWidgetItem()
        self.tableWidget.setItem(1, 0, item)
        item = QtGui.QTableWidgetItem()
        self.tableWidget.setItem(1, 1, item)
        item = QtGui.QTableWidgetItem()
        self.tableWidget.setItem(2, 0, item)
        item = QtGui.QTableWidgetItem()
        self.tableWidget.setItem(2, 1, item)
        item = QtGui.QTableWidgetItem()
        self.tableWidget.setItem(3, 0, item)
        item = QtGui.QTableWidgetItem()
        self.tableWidget.setItem(3, 1, item)
        item = QtGui.QTableWidgetItem()
        self.tableWidget.setItem(4, 0, item)
        item = QtGui.QTableWidgetItem()
        self.tableWidget.setItem(4, 1, item)
        self.tableWidget.horizontalHeader().setDefaultSectionSize(250)
        self.tableWidget.horizontalHeader().setSortIndicatorShown(False)
        self.tableWidget.horizontalHeader().setStretchLastSection(True)
        self.tableWidget.verticalHeader().setDefaultSectionSize(60)
        self.verticalLayout.addWidget(self.tableWidget)

        self.retranslateUi(ayudaE)
        QtCore.QMetaObject.connectSlotsByName(ayudaE)

    def retranslateUi(self, ayudaE):
        ayudaE.setWindowTitle(_translate("ayudaE", "AYUDA", None))
        self.plainTextEdit.setPlainText(_translate("ayudaE", "EVALUACION DE LA
FUERZA MUSCULAR\n"
"1. Conecte los electrodos en los puntos motores de los músculos del cuádriceps y
```

```
del isquiotibial, que se visualizan en la imagen.\n"
"2. Conecte el USB, los cables acorde al manual y encienda la tarjeta.\n"
"3. Haga clic en Conectar y espere hasta que aparezca el mensaje de confirmación.
Ahora ya puede realizar la evaluación. \n"
"4. Haga clic en iniciar. Una vez visualice que la gráfica da inicio y se
estabilice, realice 3 contracciones. Al finalizar, haga clic en parar.\n"
"5. Si los datos están correctos, haga clic en guardar y aparecerá el valor de la
fuerza muscular en mV. Caso contrario, realice nuevamente la adquisición.\n"
"6. Haga clic en Desconectar y espere hasta que aparezca el mensaje de
confirmación. Finalmente, haga clic en aceptar.", None))
        item = self.tableWidget.horizontalHeaderItem(0)
        item.setText(_translate("ayudaE", "ERROR", None))
        item = self.tableWidget.horizontalHeaderItem(1)
        item.setText(_translate("ayudaE", "ACCION A TOMAR", None))
        __sortingEnabled = self.tableWidget.isSortingEnabled()
        self.tableWidget.setSortingEnabled(False)
        item = self.tableWidget.item(0, 0)
        item.setText(_translate("ayudaE", "La tarjeta no se conecta al programa",
None))
        item = self.tableWidget.item(0, 1)
        item.setText(_translate("ayudaE", "Desconecte el USB y apague la tarjeta.
Vuelva a conectar el USB y después encienda la tarjeta", None))
        item = self.tableWidget.item(1, 0)
        item.setText(_translate("ayudaE", "Después de presionar Iniciar, la gráfica
se paraliza.", None))
        item = self.tableWidget.item(1, 1)
        item.setText(_translate("ayudaE", "Presione Parar y luego cancelar. La
información guardada no se alterará. Vuelva a ingresar en la pantalla de evaluación
y realice la evaluación pendiente", None))
        item = self.tableWidget.item(2, 0)
        item.setText(_translate("ayudaE", "La señal aparece en una grafica que no
corresponde al musculo", None))
        item = self.tableWidget.item(2, 1)
        item.setText(_translate("ayudaE", "Conecte los cables acorde al diagrama
que se muestra en el manual", None))
        item = self.tableWidget.item(3, 0)
        item.setText(_translate("ayudaE", "Los valores no concuerdan con la
realidad", None))
        item = self.tableWidget.item(3, 1)
        item.setText(_translate("ayudaE", "Asegúrese de que los electrodos estén
correctamente conectados en los puntos motores", None))
        item = self.tableWidget.item(4, 0)
        item.setText(_translate("ayudaE", "Al guardar no paso nada", None))
        item = self.tableWidget.item(4, 1)
        item.setText(_translate("ayudaE", "Asegúrese de haber realizado la
evaluación de la extremidad sana durante el registro", None))
        self.tableWidget.setSortingEnabled(__sortingEnabled)
```

7) Código de la ventana de ayuda para la pantalla Historia Clínica

```
# -*- coding: utf-8 -*-

# Form implementation generated from reading ui file 'ayudaH.ui'
#
# Created by: PyQt4 UI code generator 4.11.4
#
# WARNING! All changes made in this file will be lost!

from PyQt4 import QtCore, QtGui

try:
```

```
    _fromUtf8 = QtCore.QString.fromUtf8
except AttributeError:
    def _fromUtf8(s):
        return s

try:
    _encoding = QtGui.QApplication.UnicodeUTF8
    def _translate(context, text, disambig):
        return QtGui.QApplication.translate(context, text, disambig, _encoding)
except AttributeError:
    def _translate(context, text, disambig):
        return QtGui.QApplication.translate(context, text, disambig)

class Ui_ayudaH(object):
    def setupUi(self, ayudaH):
        ayudaH.setObjectName(_fromUtf8("ayudaH"))
        ayudaH.resize(522, 295)
        font = QtGui.QFont()
        font.setFamily(_fromUtf8("Leelawadee UI"))
        font.setPointSize(11)
        ayudaH.setFont(font)
        self.verticalLayout = QtGui.QVBoxLayout(ayudaH)
        self.verticalLayout.setSpacing(15)
        self.verticalLayout.setObjectName(_fromUtf8("verticalLayout"))
        self.plainTextEdit = QtGui.QPlainTextEdit(ayudaH)
        self.plainTextEdit.setMaximumSize(QtCore.QSize(16777215, 16777215))
        font = QtGui.QFont()
        font.setFamily(_fromUtf8("Leelawadee UI"))
        font.setPointSize(11)
        self.plainTextEdit.setFont(font)
        self.plainTextEdit.setAutoFillBackground(False)
        self.plainTextEdit.setStyleSheet(_fromUtf8("background-color: rgb(216, 213,
255);"))
        self.plainTextEdit.setFrameShape(QtGui.QFrame.NoFrame)
        self.plainTextEdit.setTabChangesFocus(False)
        self.plainTextEdit.setUndoRedoEnabled(True)
        self.plainTextEdit.setReadOnly(True)
        self.plainTextEdit.setBackgroundVisible(False)
        self.plainTextEdit.setCenterOnScroll(False)
        self.plainTextEdit.setObjectName(_fromUtf8("plainTextEdit"))
        self.verticalLayout.addWidget(self.plainTextEdit)

        self.retranslateUi(ayudaH)
        QtCore.QMetaObject.connectSlotsByName(ayudaH)

    def retranslateUi(self, ayudaH):
        ayudaH.setWindowTitle(_translate("ayudaH", "AYUDA", None))
        self.plainTextEdit.setPlainText(_translate("ayudaH", "DESCRIPCION\n"
"En esta ventana se muestra toda la información del paciente y un cuadro evolutivo
con la información de las evaluaciones realizadas.\n"
"\n"
"Para visualizar con mayor precisión los niveles de fuerza, haga clic en la gráfica
del cuadro evolutivo, en la posición de fecha que desea visualizar. Si desea
manipular la gráfica, en la parte inferior se encuentra una barra que le permite
hacer zoom y mover la gráfica.\n"
"\n"
"IMPRIMIR\n"
"Haga clic en el botón imprimir. El reporte se guardará en la carpeta contenedora
del programa en formato PDF. Abra el archivo e imprimalo.", None))
```

8) Código del Setup

```
from cx_Freeze import setup, Executable
import os
import scipy

includefiles_list=[]
scipy_path = os.path.dirname(scipy.__file__)
includefiles_list.append(scipy_path)

build_options = dict(packages=['matplotlib'], #this line solves an issue w/
matplotlib
                 include_files=includefiles_list, #this line is for scipy issue
                 includes=['matplotlib.backends.backend_qt5agg']) #this line solves
another issue w/ matplotlib

additional_mods = ['numpy.core._methods', 'numpy.lib.format',
'scipy.sparse.csgraph._validation', 'scipy.ndimage._ni_support',
                   'pyqtgraph.debug', 'pyqtgraph.ThreadsafeTimer']
setup(name="Evaluador",
      version="0.1",
      description="",
      options = {'build_exe': {'includes': additional_mods}},
      executables=[Executable("Evaluador.py")])
```