DESARROLLO DE UNA PLATAFORMA PARA LA TRANSMISIÓN Y RECEPCIÓN DE REGISTROS DE UN ESPIROMETRO DESDE UN TERMINAL MÓVIL.

TRABAJO ESPECIAL DE GRADO

Presentado ante la

UNIVERSIDAD CATÓLICA ANDRÉS BELLO

Como parte de los requisitos para optar al título de

INGENIERIO EN TELECOMUNICACIONES

REALIZADO POR:

Yenmari Díaz Lugo **María Fabiola Santiago O.**

TUTOR:

Iván Escalona

Caracas, febrero de 2013

UNIVERSIDAD CATÓLICA ANDRÉS BELLO.
FACULTAD DE INGENIERÍA.
ESCUELA DE INGENIERÍA DE TELECOMUNICACIONES

DESARROLLO DE UNA PLATAFORMA PARA LA TRANSMISIÓN Y RECEPCIÓN DE REGISTROS DE UN ESPIROMETRO DESDE UN TERMINAL MÓVIL.

TRABAJO ESPECIAL DE GRADO

Presentado ante la

UNIVERSIDAD CATÓLICA ANDRÉS BELLO

Como parte de los requisitos para optar al título de

INGENIERO DE TELECOMUNICACIONES

REALIZADO POR: Díaz Lugo, Yenmari

Santiago, María Fabiola

TUTOR: Escalona, Iván

FECHA: Febrero de 2013

UNIVERSIDAD CATÓLICA ANDRÉS BELLO.
FACULTAD DE INGENIERÍA.
ESCUELA DE INGENIERÍA DE TELECOMUNICACIONES

DESARROLLO DE UNA PLATAFORMA PARA LA TRANSMISIÓN Y RECEPCIÓN DE REGISTROS DE UN ESPIROMETRO DESDE UN TERMINAL MÓVIL.

Este jurado; una vez realizado el examen del presente trabajo ha evaluado su contenido con el resultado: _____

JURADO EXAMINADOR

Firma:	Firma:	Firma:
Nombre: _____	Nombre: _____	Nombre: _____

REALIZADO POR: Díaz Lugo, Yenmari
Santiago, María Fabiola

TUTOR: Escalona, Iván
FECHA: Febrero de 2013

RESUMEN

Díaz Lugo, Yenmari
yenmaridiaz @gmail.com

Santiago Ospina, María Fabiola
fabios45 @gmail.com

Actualmente en Venezuela existe una problemática en el área de la medicina por la falta de médicos especializados en las zonas rurales del país, una de las soluciones a esta problemática es la medicina a distancia mejor conocida como Telemedicina en donde se emplea Tecnologías de la Información para asistir a médicos o tratar a pacientes en lugares remotos. El presente Trabajo Especial de Grado se basó en el desarrollo de una plataforma para la transmisión de exámenes de espirometría y la recepción de un diagnóstico mediante el uso de un terminal móvil, Luego de la familiarización con el espirómetro y los equipos utilizados para el desarrollo de la plataforma, se estableció la conexión entre el servidor de la Escuela de Ingeniería en Telecomunicaciones de la UCAB (Universidad Católica Andrés Bello) y una computadora remota a través de una red VPN (Red Privada Virtual).

Durante el desarrollo se llevó a cabo el diseño de una página web donde se visualizó los pacientes clasificados por el tipo de disfunción respiratoria, la cual se almacenó en la base de datos del servidor ubicado en la UCAB.

Palabras claves: Telemedicina, espirometría, VPN, página web, base de datos.

DEDICATORIA

A mis padres y a mi familia por apoyarme incondicionalmente en todo momento

Yenmari Díaz Lugo

A mi mamá por impulsarme a alcanzar todas mis metas con su compañía y compresión, a mi papá por enseñarme a soñar en grande y a mi abuela por su cariño incondicional

María Fabiola Santiago

AGRADECIMIENTOS

En primer lugar queremos agradecerle a Dios por habernos permitido poder estudiar una carrera universitaria y culminarla con éxito, gracias a nuestros familiares por el apoyo brindado a lo largo de estos años, a nuestro tutor Iván Escalona, por apoyarnos y orientarnos en este proyecto y a nuestros profesores por brindarnos los conocimientos necesarios para el desarrollo de este Trabajo Especial de Grado.

Gracias a nuestros amigos Andrea Quintana, Oralip Sepúlveda y Carlos Abramovich por brindarnos su apoyo a lo largo de nuestra carrera y durante el desarrollo de este Trabajo Especial de Grado.

A todas aquellas personas que estuvieron de una u otra manera con nosotras durante el desarrollo de este Trabajo Especial de Grado y durante nuestra carrera universitaria.

Muchas gracias a todos

Yenmari Díaz Lugo

María Fabiola Santiago

ÍNDICE GENERAL

RESUMEN I

DEDICATORIA III

AGRADECIMIENTOS V

INTRODUCCIÓN XVII

CAPÍTULO I 1

Planteamiento del Proyecto 1

I.1 Planteamiento del problema 1

I.2 Objetivos 2

I.2.1 Objetivo General 2

I.2.2 Objetivos Específicos 2

I.3 Alcances y Limitaciones 3

I.3.1 Alcances 3

I.3.2 Limitaciones 3

I.4 Justificación 4

CAPÍTULO II 5

Marco Referencial 5

II. 1. Telemedicina 5

II. 2 Neumología 6

II. 3 Espirometría 7

II.3.1 Técnica e interpretación de los resultados 7

II.4 VPN (Virtual Private Network) 11

II.4.1 Protocolos de túnel 11

II.4.2 Tipos de VPN. 12

II.5 Base de Datos 13
II. 5.1 MySQL 14
II. 6 Página Web 14
II. 7 Cliente/Servidor 16
II.8 Protocolo SSH (Secure Shell) 17
II.9 Java 17
II. 10 Android 18
II.10.1 Eclipse 19
II. 11 Wi-Fi (Wireless Fidelity) 20
II.12 Formatos de Imagen 20
CAPÍTULO III 23
Marco Metodológico 23
CAPÍTULO IV 27
Desarrollo 27
IV.1 Configuración del servidor. 27
IV.2 Diseño de la base de datos. 30
IV.3 Diseño de la página web 34
IV.4 Aplicación en Java. 40
IV.5 Aplicación en Android. 47
CAPITULO V 49
Resultados 49
V.1 Red VPN 49
V.2 Base de datos 50
V.3 Página web 53

V.4 Aplicación en Java 60

V.5 Aplicación en Android 65

CAPÍTULO VI 69

Conclusiones y Recomendaciones 69

VI.1 Conclusiones 69

VI.2 Recomendaciones 71

BIBLIOGRAFÍA 73

APENDICE A 75

Código de Java para la conexión con la base de datos. 75

APENDICE B 91

Código de Android para el envió del examen de espirometría y la recepción de su diagnostico. 91

ÍNDICE DE FIGURAS

Figura 1. Esquema del Marco Teórico 5
Figura 2. Curva volumen-tiempo 8
Figura 3. Curva flujo-volumen 9
Figura 4. Esquema de la Metodología 23
Figura 5. Configuración de VPN Client 28
Figura 6.Mensaje de bienvenida al conectar con el servidor 28
Figura 7.Configuración de Putty 29
Figura 8. Usuario y Contraseña en Putty 29
Figura 9. Ingresando al servidor a través de Putty. 30
Figura 10.Ingresó a MySQL 31
Figura 11.Diseño de la Base de Datos. 31
Figura 12.Diseño del Acceso_Portal 32
Figura 13.Tablas en la Base de Datos en Putty 32
Figura 14.Conexión de MySQL Workbench 33
Figura 15.Tabla de Datos Personales del Paciente 33
Figura 16.Creación de las carpetas en el servidor 34
Figura 17.Configuración datos locales 35
Figura 18.Configuración datos remotos 36
Figura 19.Servidor de Prueba 36
Figura 20.Conectar al servidor remoto 37
Figura 21.Conexión con MySQL 38
Figura 22. Tablas de la base de datos en Dreamweaver 38
Figura 23.Validar usuarios 39
Figura 24.Tablas Dinámicas 40
Figura 25. Conexión a la base de datos 41
Figura 26. Busca archivo con los parámetros del examen de espirometría de un usuario. 41
Figura 27.Comparación de los valores para dar un diagnóstico del paciente. 42
Figura 28.Guarda los datos en la tabla Examen en la base de datos 42

Figura 29. Inserta los parámetros del examen en la tabla Examen de la base de datos. 43
Figura 30. Busca archivo txt con los datos personales del usuario 43
Figura 31.Guarda los datos en la tabla Datos_Personales en la base de datos 43
Figura 32. Inserta los datos personales del usuario en la tabla Datos_Personales de la base de datos. 44
Figura 33. Activación de los botones 44
Figura 34.Código para mostrar la IP de la maquina local 45
Figura 35.Inicialización del Servidor 45
Figura 36.Inicialización del cliente 46
Figura 37.Insertar hora 46
Figura 38.Guardar datos provenientes del terminal móvil 47
Figura 39.Conexión cliente Android 48
Figura 40.Envío de ping desde el cliente al servidor.....49
Figura 41.Envío de ping desde el servidor al cliente.....50
Figura 42.Tablas base de datos.....50
Figura 43.Tabla Acceso_Portal 51
Figura 44.Tabla Curvaespirométrica 51
Figura 45.Tabla Datos_Personales 52
Figura 46.Tabla Examen.....52
Figura 47.Página de Inicio.....53
Figura 48.Página de acceso 54
Figura 49.Página de Registro 55
Figura 50.Página de Consulta.....56
Figura 51.Página que presenta el diagnóstico normal 57
Figura 52.Página de Consulta.....58
Figura 53.Datos del Paciente.....58
Figura 54.Página de Información 59
Figura 55.Página de contactos. 60
Figura 56.Ventana de Inicio 61

Figura 57.Ventana de recepción de datos en la aplicación mostrando las datos del paciente. 62
Figura 58.Envío del diagnóstico al terminal móvil 62
Figura 59.Dirección donde se guarda el archivo en la computadora 63
Figura 60.Envió de diagnóstico 63
Figura 61.Insertar datos en el sistema. 64
Figura 62.Tablas de la base de datos 65
Figura 63. Icono de la aplicación 66
Figura 64. Página de inicio de la aplicación en Android 66
Figura 65. Página de Inicio de la aplicación de android. 67
Figura 66. Diagnóstico del paciente 68

ÍNDICE DE TABLAS

Tabla 1 Tipos de disfunción respiratoria..... 11

INTRODUCCIÓN

Las enfermedades que afectan al sistema respiratorio representan una causa importante de mortalidad en el mundo. El incremento de dichas enfermedades actualmente se debe a una multiplicidad de factores como son la emisión de productos contaminantes a la atmósfera, el humo emitido por las industrias sin un filtrado adecuado, los gases emitidos por vehículos automotores, el consumo de tabaco, el calentamiento global, entre otros. Uno de los métodos para reconocer dichas enfermedades en los pacientes es realizándoles un examen de espirometría.

Este examen consta de una evaluación analítica de la capacidad pulmonar por medio de un espirómetro, el cual es un instrumento que mide y registra el volumen de aire inhalado y exhalado del paciente, luego la información volumétrica se registra en un gráfico denominado espirograma. (OCEANO)

Uno de los problemas que enfrentan los médicos tanto generales como especialistas durante la práctica de la medicina ante éste y otros tipos de enfermedades, es poder estar en constante seguimiento del paciente debido a que éste puede encontrarse en áreas de difícil acceso en las cuales no hay médicos especializados en este tipo de enfermedades. Una solución es la aplicación de la medicina a distancia, la cual recibe el nombre de Telemedicina, en donde se emplea usualmente Tecnologías de la Información y las Comunicaciones para asistir a médicos o tratar pacientes en lugares remotos (Marcano & Ramirez, 2007).

Previamente a este Trabajo Especial de Grado, (Genovés, D. y Rizzo, D, 2010), desarrollaron una aplicación móvil para la transmisión y recepción de información espirométrica. En un principio el proyecto se planteó como una continuación al trabajo mencionado, pero luego sirvió como guía para el desarrollo de la aplicación móvil.

Es por esta razón que el siguiente Trabajo Especial de Grado desarrolla una plataforma para la transmisión de exámenes de espirometría y la recepción de un diagnóstico preliminar a través de un terminal móvil.

Este tomo se encuentra estructurado en 6 capítulos. En el primer capítulo se define el planteamiento del problema, sus objetivos, los alcances y limitaciones y la justificación e importancia de este proyecto.

En el capítulo dos se definen las bases teóricas de la investigación, en el capítulo tres se establecen los métodos y procedimientos para llevar a cabo este proyecto, este capítulo corresponde al marco metodológico.

En el capítulo cuatro se explica en forma detallada cada una de las fases de la investigación establecidas en el marco metodológico, dicho capítulo corresponde al desarrollo, donde se diseñó una plataforma la cual integra los exámenes de espirometría y el diagnóstico del mismo.

En el capítulo cinco corresponde a los resultados obtenidos durante la realización de este proyecto y finalmente en el capítulo seis se presentan las conclusiones y recomendaciones que dan respuesta al problema planteado, así como a los objetivos propuestos.

CAPÍTULO I

Planteamiento del Proyecto

En el siguiente capítulo se hace una descripción general del proyecto, definiendo el planteamiento del problema, los objetivos, la justificación, así como las limitaciones y alcances presentes.

I.1 Planteamiento del problema

El avance de las ciencias médicas a nivel mundial es vertiginoso, la transmisión de la información en tiempo real es vital, las herramientas para dicha transmisión es esencial. La telemedicina o más recientemente la medicina robótica teledirigida en tiempo real ha iniciado una de las eras más brillantes de la experiencia médica a distancia, capaz de sorprender al propio médico de cualquier especialidad.

Se plantea entonces el desarrollo de un sistema que permita al médico especialista en neumología analizar a distancia las diversas variables asociadas a pacientes que puedan presentar insuficiencia respiratoria, cuya vida pudiera estar comprometida si están en sitios distante de las unidades médicas de cuidados intensivo o críticos, todo con el fin de determinar la orientación adecuada a fin de aplicar los tratamientos más idóneos.

Para la realización de este trabajo se toman indicadores propios de la espirometría, y la posibilidad de enviar a distancia las diversas informaciones que de ellos se extraen, como son: la capacidad vital, la capacidad residual, el flujo respiratorio, las concentraciones de CO_2 (dióxido de carbono), O_2 (oxígeno), así como información sobre los diversos gases que funcionan en un proceso respiratorio normal o patológico.

En el presente trabajo se desarrolla un procedimiento práctico, eficaz, seguro, de fácil manejo y accesible a los diferentes centros de salud que requieran información

espirométrica de los pacientes que puedan presentar insuficiencia respiratoria por causas de enfermedades como por ejemplo el asma, que representa entre el 35 % al 45% de las enfermedades respiratorias de la población susceptible, así como también dentro de los hospitales que atienden niños, como el Hospital de Niños "J.M. de los Ríos" o el Hospital "Dr. José Ignacio Baldó" (El Algodonal en Antímano) entre otros.

Se diseña una plataforma en un servidor con ubicación en la Universidad Católica Andrés Bello que recibe de forma remota la información proveniente de un dispositivo móvil con sistema operativo Android, la almacena y genera un diagnóstico preliminar del examen.

I.2 Objetivos

A continuación se presentan el objetivo general y los objetivos específicos propuestos para la realización de este proyecto.

I.2.1 Objetivo General

Desarrollar una plataforma en un servidor que se implementará en el área de neumología para la recepción, procesamiento, almacenamiento y envío de señales provenientes de un espirómetro a través de un dispositivo móvil.

I.2.2 Objetivos Específicos

- Evaluar el comportamiento de la señal proveniente del espirómetro, para su posterior procesamiento y almacenamiento.
- Indagar acerca de los datos y las características que deben estar presentes en la graficación de la señal emitida por el espirómetro.
- Estudiar los posibles formatos de imagen en los cuales se puede presentar el examen de espirometría.

- Determinar el ancho de banda óptimo para el envío de este tipo de información.
- Desarrollar una página web.
- Construir una base de datos con distintos casos proveniente del espirómetro.
- Comparar la información proveniente del espirómetro con la base datos que permita emitir un diagnóstico preliminar del paciente.

I.3 Alcances y Limitaciones

I.3.1 Alcances

Este Trabajo Especial de Grado incluye el diseño de una página web donde cada médico autorizado podrá tener acceso a la lectura de los resultados del examen de espirometría, así como del diagnóstico preliminar del paciente.

Para facilitar el análisis de resultados del examen se incluye una base de datos y la configuración de un modelo cliente servidor que permita la comunicación del servidor de la UCAB con una computadora personal de forma remota, también comprende una aplicación móvil el cual envía los datos del paciente así como el examen.

I.3.2 Limitaciones

Este Trabajo Especial de Grado no incluirá la modificación del espirómetro pues la disponibilidad del equipo viene dada por la facilidad que tenga el personal médico contactado por la Universidad Católica Andrés Bello de proporcionarlo,

tampoco incluirá el funcionamiento en aplicaciones similares para otros sistemas operativos distintos a Android.

Otra de las limitaciones presentes en el proyecto es que para la conexión con la plataforma es necesaria la utilización de una red WiFi debido a que es la única manera de acceder al servidor.

La conexión con el servidor de la UCAB debe realizarse fuera de las instalaciones de la misma, debido a que no se puede acceder a la VPN desde la misma red debido a restricciones hechas por la Dirección de Tecnologías de Información DTI así mismo tampoco se puede acceder al servidor mediante dispositivos como celulares y tabletas.

El médico que realiza el examen debe de tener el archivo de texto formato .txt con los parámetros de los exámenes y contar con los datos personales del paciente para obtener un diagnóstico del mismo.

I.4 Justificación

El desarrollo de una plataforma para la transmisión de exámenes de espirometría se plantea con el objetivo de lograr un avance en el área de neumología especialmente debido a la carencia de médicos especialista en las zonas rurales del país. Este proyecto permite a los médicos observar los diferentes casos de disfunción respiratoria por medio de una red WiFi evitando el traslado de los pacientes a los centros especializados.

CAPÍTULO II

Marco Referencial

Este capítulo presenta la información necesaria para llevar a cabo el desarrollo de una plataforma para la transmisión y recepción de registros de un espirómetro desde un terminal móvil. En la Figura 1 se muestra los puntos teóricos a tratar durante la realización de este proyecto.

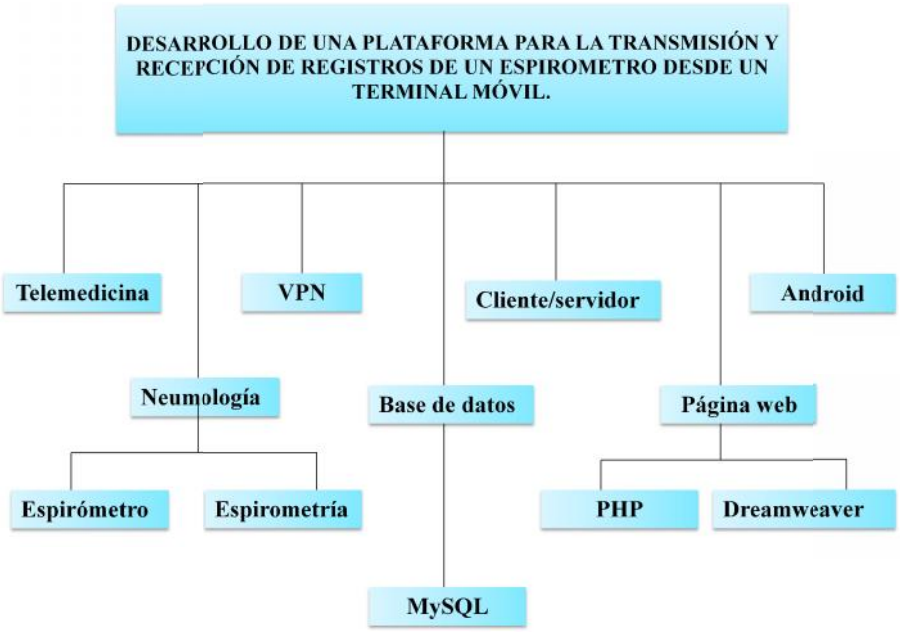


Figura 1. Esquema del Marco Teórico

Fuente: Elaboración Propia

II. 1. Telemedicina

La Telemedicina se define como el ofrecimiento de atención médica a personas en todo el mundo mediante la combinación de la tecnología avanzada en telecomunicaciones con los conocimientos médicos. Su objetivo es proporcionar atención médica de alta calidad a pacientes en todo el mundo a un costo relativamente bajo. Proporciona rápido acceso a servicios médicos de alta calidad, a un precio

razonable, a pacientes que de lo contrario quizá no recibirían atención alguna. (Course Tecnology, 1999)

Según la ATA (American Telemedicine Association) Telemedicina es el intercambio de información médica mediante comunicaciones electrónicas para la salud y la educación de pacientes ó profesionales sanitarios y con el fin de mejorar la asistencia. Uno de los términos que están asociados con telemedicina es Telesalud que se utiliza a menudo para incluir una definición más amplia de ayuda médica remota que no siempre involucra servicios clínicos, entre los servicios se puede encontrar videoconferencias, transmisión de imágenes fijas incluyendo portales de pacientes, monitoreo remoto de signos vitales, educación y capacitación medica continua además de centros de llamadas de enfermería, todos estos servicios son considerados parte de la Telemedicina y la Telesalud. (American Telemedicine Association, 2012)

II. 2 Neumología

Es la parte de la medicina que se ocupa de las enfermedades del pulmón y por ende se encarga del estudio y tratamiento médico de las enfermedades que aquejan al aparato respiratorio y al tórax. Estas incluyen padecimientos de la vía respiratoria (tráquea, bronquios y pulmones), pleuras (membranas que recubren al pulmón), la caja torácica con todos sus elementos (pared torácica: músculos, costillas, vértebras). De igual forma y con excepción de algunos órganos, el mediastino (espacio comprendido entre ambos pulmones), también es de interés en dicha especialidad. Las enfermedades respiratorias y del tórax pueden ser primarias (originadas en los pulmones y el propio tórax), o ser secundarias a enfermedades sistémicas (por ejemplo enfermedades inmunológicas, insuficiencia de órganos como el corazón y el riñón, tumores e infecciones originadas en otro órgano). (Reyes, 2012)

II. 3 Espirometría

La espirometría es un método de valoración pulmonar que permite hallar el volumen del aire que puede alojar los pulmones. El volumen de aire pulmonar o capacidad vital (CV) se obtiene tras una expulsión de aire hasta alcanzar el volumen residual, expulsión que va precedida siempre de una inspiración máxima. La espirometría es de gran ayuda en la exploración y diagnóstico de la función pulmonar. Revela el estado de las vías aéreas (problemas obstructivos), si existe disminución de la capacidad pulmonar (problemas restrictivos), o si se trata de problemas mixtos. Permite conocer igualmente el grado de adaptación pulmonar frente al esfuerzo, mediante el conocimiento de la frecuencia respiratoria y el volumen máximo por minuto que puede movilizar un sujeto al realizar un ejercicio. (Fernández, 2006).

II.3.1 Técnica e interpretación de los resultados

La espirometría es un registro que depende del esfuerzo realizado y, por lo tanto requiere una compresión y una cooperación adecuadas por parte del paciente. La maniobra comienza con una inhalación máxima de aire seguida inmediatamente por una exhalación realizada en forma rápida y también máxima. Se debe estimular al paciente para que espire de manera tan rápida y completa como le sea posible. En forma rutinaria la prueba se efectúa en posición de sentado y con una pinza nasal.

La maniobra debe repetirse 3 veces como mínimo. Si los trazados espirométricos son técnicamente aceptables y reproducibles, se puede dar como finalizada la prueba.

La interpretación de los resultados se realiza comparando los valores medidos en el paciente con otros de referencia obtenidos de estudios en poblaciones de personas sanas no fumadoras. La Sociedad Americana del Tórax (ATS) ha publicado

recomendaciones para la aceptabilidad y reproducibilidad de las maniobras espirométricas y para el mantenimiento, calibración y control de calidad de los equipos.

Una vez realizada la maniobra se puede obtener dos tipos de registro: la curva volumen-tiempo y la curva flujo-volumen.

La curva volumen-tiempo es el registro de volumen de aire espirado en función del tiempo empleado en la espiración; el gráfico está compuesto por un sistema de ejes vertical y horizontal, donde se registra el volumen en litros en el eje vertical y el tiempo en segundos en el eje horizontal.(ver Figura 2)

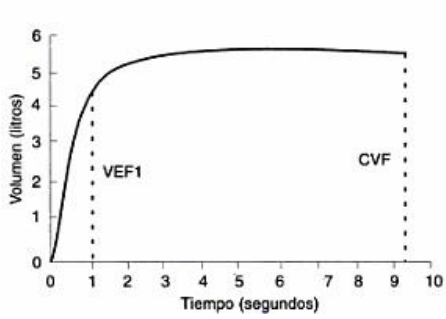


Figura 2. Curva volumen-tiempo

Fuente: (Alvarez, 2008)

La curva flujo-volumen registra los flujos espirados e inspirados en litros por segundo, en el eje vertical, y el volumen de aire en litros, en el eje horizontal.(ver Figura 3)

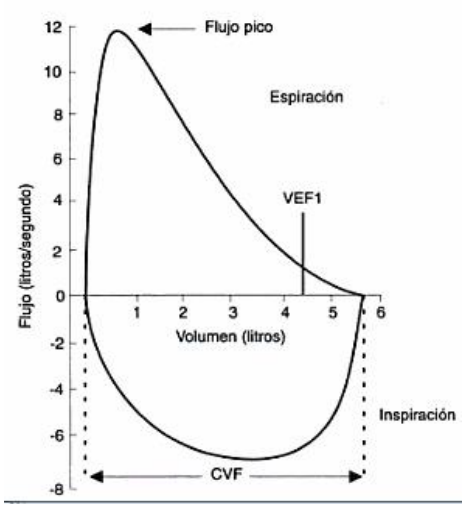


Figura 3. Curva flujo-volumen

Fuente: (Alvarez, 2008)

Los valores espirométricos de mayor relevancia obtenidos a partir de estas curvas son la capacidad vital forzada (CVF), el volumen espiratorio forzado en el primer segundo (VEF1), el flujo espiratorio forzado 25-75 (FEF25-75) y el flujo pico o flujo máximo espiratorio forzado

La capacidad vital forzada (CVF) es el máximo volumen de aire que se puede exhalar de los pulmones luego de una inhalación máxima.

El volumen espiratorio forzado en el primer segundo (VEF1) es el máximo volumen de aire que es espirado en el primer segundo.

La relación VEF1/CVF, también llamado índice de Tiffenau, esta generalmente entre 0,70-0,80, lo cual indica que entre el 70% y el 80% de la capacidad vital forzada es espirada en el primer segundo en la mayoría de las personas.

El flujo espiratorio forzado 25-75 (FEF25-75) es el flujo espiratorio entre el 25% y el 75% de la curva de capacidad vital forzada, y evalúa la vía aérea de pequeño calibre.

El flujo pico o flujo máximo espiratorio forzado corresponde al mayor flujo espiratorio medido que se registra tempranamente en la maniobra espiratoria forzada, puede obtenerse de la curva flujo volumen, o también de un medidor de flujo pico portátil. (Alvarez, 2008).

El espirómetro es un instrumento que consta de una boquilla y un tubo conectado a un dispositivo de registro. Este instrumento de registro mide el volumen de aire inspirado o espirado y la duración de cada respiración. (Pasquier, 2013).

La espirometría puede detectar dos tipos básicos de disfunción respiratoria: el patrón obstructivo y el patrón restrictivo.

El criterio primario para definir un patrón obstructivo es el descenso de la relación VEF1/CVF; el VEF1, el flujo pico y el FEF25-75 también se encuentran disminuidos en mayor o menor proporción de acuerdo con la gravedad de la obstrucción. La CVF puede ser normal o estar disminuida cuando el grado de obstrucción es más grave.

El patrón restrictivo se caracteriza por un descenso de la capacidad vital forzada con relación VEF1/CVF normal o aumentada; el VEF1 se encuentra reducido proporcionalmente a la reducción de la CVF y el FEF 25-75 es variable. (Alvarez, 2008).

El patrón mixto comparte características de los patrones anteriores.

A continuación se muestra una tabla que presenta los tipos de disfunciones respiratorias presentes en un examen de espirometría en donde las flechas representan la disminución de los parámetros de su valor normal.

	Obstructivo	Restrictivo	Mixto
CVF	Normal	↓	↓
VEF1	↓	↓	↓
VEF1/CVF	↓	Normal	↓

Tabla 1 Tipos de disfunción respiratoria

Fuente: (Alvarez, 2008).

II.4 VPN (Virtual Private Network)

La Red Privada Virtual permite simular una red local sobre una red pública. Una vez establecida la conexión de la red privada virtual los datos viajan encriptados de forma que el emisor y el receptor son capaces de leerlo. Para hacerlo posible de manera segura es necesario proporcionar los medios para garantizar la autenticación, integridad y confidencialidad. (Piquero, 2009)

II.4.1 Protocolos de túnel

Los principales protocolos de túnel son:

- Protocolo PPTP (Protocolo de túnel punto a punto) consiste en crear tramas con el protocolo PPP (Punto a Punto) y después encapsularlos mediante un datagrama de IP. Por lo tanto, con este tipo de conexión, los equipos remotos en dos redes de área local se conectan con una conexión de igual a igual (con un sistema de autenticación/cifrado) y el paquete se envía dentro de un datagrama de IP. De esta manera, los datos de la red de área local así como las direcciones de los equipos que se encuentran en el encabezado del mensaje no serán visibles.

- Protocolo L2TP: el protocolo de túnel de capa 2 es un protocolo muy similar al PPTP. L2TP encapsula tramas PPP que a su vez encapsulan otros protocolos (como IP, IPX o NetBIOS).

- Protocolo IPSEC (*Internet Protocol security*): es un protocolo definido por el IETF (Internet Engineering Task Force), que se usa para transferir datos de manera segura en la capa de red. En realidad es un protocolo que mejora la seguridad del protocolo IP para garantizar la privacidad, integridad y autenticación de los datos enviados. Está basado en tres módulos.

- Encabezado de autenticación IP (AH, *Authentication Header*), que incluye integridad, autenticación y protección contra ataques de REPLAY a los paquetes.

- Carga útil de seguridad encapsulada (ESP, *Encapsulating Security Payload*), que define el cifrado del paquete. ESP brinda privacidad, integridad, autenticación y protección contra ataques de REPLAY.

- Asociación de seguridad (SA), define configuraciones de seguridad e intercambio de clave. Las SA incluyen toda la información acerca de cómo procesar paquetes IP (los protocolos AH y/o ESP, el modo de transporte o túnel, los algoritmos de seguridad utilizados por los protocolos, las claves utilizadas, etc.). El intercambio clave se realiza manualmente o con el protocolo de intercambio IKE (internet key exchange en la mayoría de los casos), lo que permite que ambas partes se escuchen entre sí. (Romero, 2010).

II.4.2 Tipos de VPN.

Básicamente existen tres arquitecturas de conexión VPN:

- VPN de acceso remoto: consiste en usuarios que se conectan con la empresa desde sitios remotos utilizando Internet.
- VPN punto a punto: está compuesto por en servidor fijo que posee un vínculo permanente a Internet, acepta las conexiones vía Internet provenientes de los sitios y establece túnel VPN, generalmente utilizados para conectar oficinas remotas con la sede central de la organización.
- VPN interna WLAN: es una variante de acceso remoto pero, en vez de utilizar Internet como medio de conexión, emplea la misma red de área local (LAN) de la empresa. (Piquero, 2009)

II.5 Base de Datos

Una base de datos es un conjunto de datos almacenados sin redundancia innecesaria en un soporte informático y accesible simultáneamente por distintos usuarios y aplicaciones. Los datos deben estar estructurados y almacenados de forma totalmente independiente de las aplicaciones que la utilizan.

Los componentes de una base de datos son:

- Datos: es el componente fundamental de la base de datos, están relacionados entre sí formando un conjunto con mínimas redundancias. Los datos por sí mismo no aportan conocimiento hay que procesarlos y transformarlos.
- Software SGBD (Sistema de Gestión de Bases de Datos): es un software o un conjunto de programas que permite crear y mantener una base de datos. El SGBD actúa como interfaz entre los programas de aplicación (usuarios) y el sistema operativo. El objetivo principal es proporcionar un entorno eficiente a la hora de almacenar y recuperar la información de la base de datos, este facilita el proceso de definir, construir y manipular la base de datos.
- Usuarios: existen tres tipos los programadores de aplicación, estos son los que escriben programas de aplicación que utiliza la base de datos, los usuarios poco experimentados que utilizan las aplicaciones escritas previamente y los usuarios

casuales que utilizan consultas formuladas en un lenguaje de consulta de base de datos.

- Administrador de la base de datos (DBA): son las personas encargados del control del sistema, entre sus funciones se pueden mencionar definir y modificar el esquema de la base de datos, así como crear y modificar las estructuras de almacenamiento físicas y los métodos de acceso, autorizar el acceso a la base de datos, entre otras. (Yera, 2007).

II. 5.1 MySQL

Es un sistema de administración de bases de datos relacionales rápido, sólido y flexible. Es ideal para crear base de datos con acceso desde páginas web dinámicas, para la creación de transacciones o para cualquier solución que implique almacenar datos, teniendo la posibilidad de realizar múltiples y rápidas consultas, MySQL ofrece varias ventajas respecto a otros sistemas gestores de bases de datos ya que tiene licencia pública, permitiendo no solo la utilización del programa sino también la consulta y modificación de su código fuente, el programa está desarrollado en C y C++, lo que facilita su integración en otras aplicaciones desarrolladas igualmente en esos lenguajes, utiliza el lenguaje SQL (lenguaje de consulta estructurado) que es el lenguaje de consulta más usado y estandarizado para acceder a base de datos relacionales. (Gómez, 2005)

II. 6 Página Web

Las páginas web son un medio para transmitir información, la cual se crea mediante un lenguaje especial denominado HTML (Hypertext Markup Language, lenguaje de etiquetado de documentos hipertextual). Este lenguaje permite crear documentos multimedia que pueden incluir texto, imágenes, sonido, videos, animaciones, entre otros. (Ideas Propias, 2005)

Una página web puede constar de dos partes: la primera es la cabecera, que representa la primera parte de la página web, y contiene el título de la página y otros parámetros que el navegador utilizará y la segunda parte es el cuerpo, que contiene el contenido real de la página, incluye el texto y las etiquetas. Mientras que el texto es la información real contenida en una página, las etiquetas definen la apariencia del documento.

Para facilitar la creación de páginas web se han desarrollados unos programas denominados editores de HTML el cual pueden ser clasificados en dos grandes grupos:

- Editores WYSIWYG (What You See What You Get): con este tipo de editores el usuario no trabaja directamente sobre el código, sino sobre un documento que muestra la apariencia de la página tal como llegaría al usuario final. Se trata de editores ideales para personas que comienzan a utilizar el lenguaje HTML y quieren llegar a crear una página web sin necesidad de dominar el lenguaje, ya que estos editores se encargan de generar automáticamente ese código de forma oculta para el usuario. Por supuesto, el usuario tiene la posibilidad de consultar el código generado y, en algunos casos, dependiendo del editor, también de modificarlo.

Editores no WYSIWYG: en este tipo de editores, el usuario en todo momento puede ver y editar el código fuente. Estos editores disponen de opciones que permiten generar automáticamente la estructura de los diferentes elementos de la página y ayudan a la utilización de los comandos sin necesidad de que el usuario los conozca con todo detalle.

PHP (Hyper Text Processor) es un lenguaje interpretado del lado del servidor que se caracteriza por su potencia, versatilidad, robustez y modularidad. Los programas escritos en PHP se pueden incorporar directamente en el código HTML y ejecutados por el servidor web a través de un intérprete antes de transferir al cliente que lo ha solicitado, un resultado en forma de HTML puro. (Gómez, 2005)

Dreamweaver es un editor visual de páginas web, por lo que no es necesario conocer el lenguaje de programación para editarlas. Pero el programa también permite la edición directa del código, debido a que los programadores pueden introducir directamente las instrucciones. La codificación de los documentos HTML se basa en una estructura de etiqueta. Las páginas cuentan con dos secciones de código: head (cabecera) y body (cuerpo). La primera contiene las etiquetas necesarias para que la página se vea correctamente en Internet y la segunda, los elementos que conforman la página. (MEDIAactive, 2010)

II. 7 Cliente/Servidor

El modelo cliente/servidor (C/S), la tecnología cliente/servidor y la arquitectura cliente/servidor se refieren a un modelo de diseño que se puede pensar como aplicaciones que se ejecutan en una red de área local (LAN). En términos muy básicos, puede describir que el cliente solicita y que el servidor ejecuta o de alguna forma realiza las solicitudes de trabajo. Las computadoras en la red se programan para desempeñar eficazmente el trabajo dividiendo las tareas de procesamiento entre clientes y servidores. (Kendall, 2005)

Entre las ventajas de la arquitectura de cliente/servidor se encuentra

- Permite que la información se procese cerca de donde se ha generado.
- Dado que las funciones de software quedan repartidas entre varias máquinas, es posible utilizar computadoras que no necesiten un alto poder de procesamiento para los clientes, lo cual baja los costos ya que únicamente el servidor es el que va a necesitar ser una máquina más poderosa, por lo cual reduce los costos.
- El crecimiento del hardware es gradual, debido a que se puede aumentar el número de clientes sin que sea necesario cambiar cada vez el servidor, se puede sustituir el servidor sin que los clientes se vean afectados, se puede aumentar la capacidad de un cliente sin tener que cambiar ni el servidor ni a los otros clientes.
- Facilita el uso de interfaces gráficas de usuario y aplicaciones multimedia.

Entre los inconvenientes de la arquitectura cliente/servidor se puede encontrar: (Falgueras, 2003)

- El software distribuido es más complejo que el no distribuido y también más difícil de probarlo y depurar errores en él; también la administración y la problemática de seguridad son más complejas.
- Cualquier sistema distribuido tiende a fallar con más frecuencia que en un sistema centralizado, ya que tiene más componentes que pueden fallar independientemente.

II.8 Protocolo SSH (Secure Shell)

Es un protocolo de intérprete de órdenes segura, se utiliza para acceder a equipos de forma remota a través de una red informática, a los que se puede controlar mediante un intérprete de órdenes o comandos. Además del acceso remoto tiene otras funciones, como transferir ficheros de forma segura, es decir, utilizando los protocolos SFTP (Secure File Transfer Protocol), o crear un canal seguro para intercambiar información entre equipos. Trabaja de forma similar al protocolo Telnet pero de forma segura, cifrando la información para que cuando esta circule por la red de un equipo a otro, vaya encriptado y dificulte así ser interceptada. Para poderlo usar es necesario tener instalado un programa para tal fin en ambos equipos, que suele tener el mismo nombre que el protocolo SSH. (Ramos, 2011).

II.9 Java

Es un lenguaje de programación orientado a objetos desarrollado por Sun MicroSystems, una compañía reconocida por sus estaciones de trabajo UNIX de alta calidad. Fundamentado en C++, el lenguaje Java se diseñó para ser pequeño, sencillo y portátil a través de plataformas y sistemas operativos, tanto a nivel de código fuente como binario, lo que significa que los programas Java (aplicaciones), pueden ejecutarse en cualquier computadora que tenga instalada un máquina virtual de Java. (Flores).

II. 10 Android

Android es el sistema operativo para smartphones desarrollado por Google. Entre sus principales características se puede destacar que está basado en Linux (*software* libre) y que cuenta con aplicaciones o programas para añadir funcionalidades al smartphone. Estas aplicaciones en su mayoría no vienen instaladas en el teléfono, por lo que es indispensable disponer de conexión a Internet (Wi-Fi o 3G) para sacar el máximo partido al Smartphone. (Cano, 2011).

Proporciona una plataforma de desarrollo libre para aplicaciones con gran riqueza e innovaciones (sensores, localización, servicios, barras de notificaciones). También se conoce como Android SDK.

Una de las mayores fortalezas del entorno de aplicación de Android es que se aprovecha el lenguaje de programación Java. El SDK de Android no acaba de ofrecer todo lo disponible para su estándar del entorno de ejecución Java (JRE), pero es compatible con una fracción muy significativa de la misma.

Normalmente las aplicaciones Android están escritas en Java. Para desarrollar aplicaciones en Java se puede utilizar el Android SDK. Existe otra opción consiste en desarrollar las aplicaciones utilizando C/C++. Para esta opción se puede utilizar el Android NDK (Native Development Kit) . (Girones, 2011)

La plataforma de Android está compuesta de varias capas: El kernel de Linux, librerías nativas, la máquina virtual de Dalvik y un framework de aplicación. El kernel de Linux proporciona los servicios básicos del sistema operativo básico y de abstracción de hardware para las pilas de software superior. Las librerías nativas apoyan las diversas funcionalidades de los buscadores web, procesamiento de datos multimedia, acceso a bases de datos y la recepción de GPS optimizado para un entorno de recursos limitados de hardware. Los registros basados en la máquina virtual de Dalvik ejecutan código Java con una demanda de memoria.

Los desarrolladores tienen pleno acceso a la API (Interfaz de programación de aplicaciones) de un mismo framework utilizado por el núcleo de aplicaciones. La arquitectura de la aplicación está diseñada para simplificar la reutilización de componentes, y cualquier aplicación puede publicar sus capacidades y cualquier otra aplicación podrá entonces hacer uso de esas capacidades (sujeto a restricciones de seguridad impuestas por el framework). Este mismo mecanismo permite que los componentes se remplacen por el usuario baja. En la parte superior de las capas, Android proporciona un framework de programación basado en componentes para que los usuarios puedan crear fácilmente sus propias aplicaciones. (Osorio, 2011)

Los sockets stream ofrecen un servicio orientado a conexión, donde los datos se transfieren como un flujo continuo, sin encuadrarlos en registros o bloques. Este tipo de socket se basa en el protocolo TCP, que es un protocolo orientado a conexión. Esto implica que antes de transmitir información hay que establecer una conexión entre los dos sockets. Mientras uno de los sockets atiende peticiones de conexión (servidor), el otro solicita la conexión (cliente). Una vez que los dos sockets están conectados, ya se pueden transmitir datos en ambas direcciones. El protocolo incorpora de manera transparente al programador la corrección de errores. Es decir, si detecta que parte de la información no llegó a su destino correctamente, esta volverá a ser transmitida. Además, no limita el tamaño máximo de información a transmitir. (Girones, 2011).

II.10.1 Eclipse

Es un IDE multiplataforma libre (software que, una vez obtenido, puede ser usado, copiado, estudiado, modificado y redistribuido libremente) para crear aplicaciones de cualquier tipo que emplea módulos (en ingles plug-in) para proporcionar toda su funcionalidad. Eclipse originalmente fue creado por IBM (International Business Machines) y en la actualidad lo desarrolla la Fundación Eclipse (organización independiente sin ánimo de lucro que fomenta una comunidad de código abierto). La definición que le da el proyecto Eclipse a su software es : “una

especie de herramienta universal – un IDE abierto y extensible para todo y nada en particular". (Flores, 2008)

II. 11 Wi-Fi (Wireless Fidelity)

Las tecnologías inalámbricas más comunes para la prestación de servicios de telecomunicaciones son aquellas que se basan en el uso del espectro radioeléctrico (radio), tales como la telefonía móvil, las redes Wi-Fi, las redes WiMAX, la tecnología Bluetooth, las comunicaciones por satélite entre otras. Estas tecnologías inalámbricas, lo son porque las uniones, conexiones o enlaces entre los diferentes terminales no se basan en el contacto físico a través de cables, y los diferentes aparatos se comunican utilizando ondas radioeléctricas mediante elementos de emisión y recepción incorporados.

La tecnología de acceso directo inalámbrico comúnmente denominada Wireless Fidelity (Wi-Fi) se aplica a redes de ordenadores o terminales de datos conectados mediante ondas de radio basadas en el estándar 802.11 del Instituto de Ingenieros Electricos y Electronicos (IEEE), con un funcionamiento parecido al de los teléfonos móviles y que permite una conexión inalámbrica entre distintos equipos en un radio geográfico restringido determinado por una estación base. La tecnología inalámbrica Wi-Fi permite conexiones entre los propios terminales para la transmisión de datos a alta velocidad y, así mismo, permite conexiones a Internet de banda ancha a gran velocidad. (Belvehí, 2004)

II.12 Formatos de Imagen

Existen varios tipos de formato de imagen entre los más utilizados podemos encontrar GIF (Graphics Interchange Format), PNG (Portable Network Graphics) y JPEG (Joint Photographic Experts Group), estos son los más recomendables debido a

que ocupan menos memoria, cada uno de estos formatos poseen ciertas características que las diferencian entre sí como son la transparencia, animación, comprensión, gamma, entre otros.

El formato GIF emplea el esquema de compresión Lempel Ziv Welch (LZW) o también conocido como esquema de compresión sin perdidas, este se utiliza para reducir su tamaño, y la paleta de color se limita a 8 bits, por lo que permite un máximo de 256 colores. Además permite definir un color como transparente, normalmente se asigna la trasparencia al color de fondo de la imagen y la forma de descarga del navegador se puede definir como normal o entrelazada. En el modo entrelazada, las líneas que forman la imagen no se almacenan secuencialmente, sino entrelazadas por grupos, de este modo como se descargue la imagen se van visualizando las líneas entrelazadas, lo que proporciona una idea global de la imagen final.

El formato JPEG permite comprimir enormemente las imágenes fotográficas, estas son imágenes a todo color (24 bits por punto de la imagen). Emplea JPEG progresivo este es un mecanismo similar al entrelazado en el formato GIF ya que se pueden descargar progresivamente las imágenes, lo que facilita una previsualización rápidas de las imágenes. El esquema de compresión de JPEG es una sofisticada técnica matemática denominada transformada discreta de cosenos, mediante esta técnica se pueden elegir distintos grados de compresión y de calidad, también emplea compresión con perdidas ya que los datos no necesarios se descartan a medida que se comprime cada vez más la imagen.

El formato PNG fue desarrollado por W3C (World Wide Web Consortium) como una alternativa de carácter público al formato GIF tradicional, este formato fue creado específicamente para internet y otras redes de ordenadores. Entre sus características más importantes podemos mencionar:

- Soporte de imágenes basadas en paleta (1, 2, 4, 8-bit), como GIF.

- Soporte de escala de grises (1, 2, 4, 8, 16-bit)
- Soporte de color real (24, 48-bit), como JPEG.
- Transparencia binaria, como GIF.
- Corrección gamma.
- Algoritmo de compresión sin pérdidas.

El formato PNG no permite animaciones como GIF, pero se ha desarrollado un nuevo formato basado en PNG que si permite crear animaciones. (Mora, 2002)

CAPÍTULO III

Marco Metodológico

Este Trabajo Especial de Grado se basa en el desarrollo de una plataforma para la transmisión y recepción de registros de un espirómetro desde un terminal móvil. A continuación se presentan los distintos métodos y procedimientos empleados en la investigación una vez formulado el planteamiento del problema y definido las bases teóricas. En la Figura 4 se muestra el esquema de las diferentes fases que fueron empleadas para la elaboración de este proyecto.

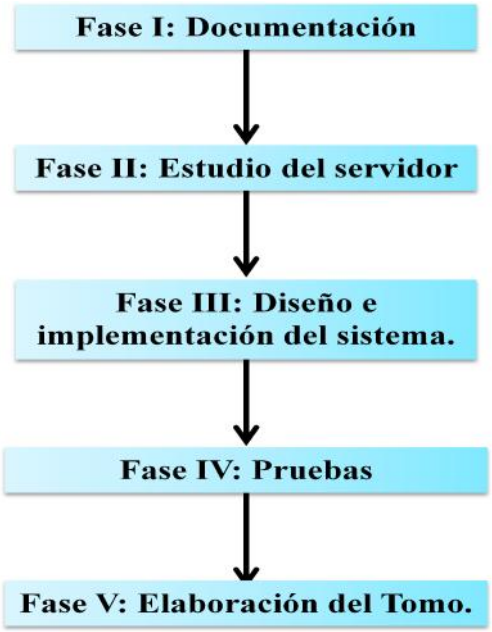


Figura 4. Esquema de la Metodología

Fuente: Elaboración Propia

- **Fase I: Documentación**

En esta fase se investigó y recopiló toda la teoría necesaria para la realización de este proyecto.

Inicialmente se investigó sobre los Trabajos Especiales de Grado relacionados con el proyecto, también se averiguó acerca de la espirometría y de los parámetros y problemas respiratorios presentes en este tipo de examen así como la lectura del mismo para poder determinar un diagnóstico del paciente, por otra parte se recopiló información sobre MySQL Workbench, Dreamweaver8 y Eclipse ya que estos fueron los software utilizados para la creación de la base de datos, página web y aplicación en el terminal móvil.

- **Fase II: Estudio del Servidor**

En esta fase se estudió las características generales presentes en el servidor como la capacidad de almacenamiento, la dirección IP, sistema operativo, entre otras, ya que no fue necesario la configuración del mismo debido a que por razones de seguridad el personal de la DTI (Dirección de Tecnologías de Información) es el encargado de gestionar y configurar los servidores existentes y dándo un espacio de memoria para realizar este proyecto.

- **Fase III: Diseño e implementación del sistema**

Para el desarrollo de este proyecto era necesario la creación de una base de datos la cual contenía tanto los datos personales del paciente como el examen de espirometría, por lo que se procedió a realizar las tablas necesarias con la ayuda del software MySQL Workbench.

Para que los médicos revisaran de una manera más fácil los resultados del examen se realizó una página web mediante el software Dreamweaver8, en donde se

observó todos los pacientes a los que le realizaron estás pruebas clasificados por el tipo de disfunción respiratoria que poseían.

Para poder obtener un diagnóstico del paciente se elaboró una aplicación en Java la cual recibía del terminal móvil los datos del examen, y una vez obtenidos los datos, la aplicación generó el diagnóstico preliminar del paciente enviando esta información a la base de datos, página web y al terminal móvil.

En el terminal móvil con sistema operativo Android se realizó una aplicación mediante el software eclipse en el cual, se podían enviar el examen de espirometría y los datos personales del paciente a la aplicación en Java.

- **Fase IV: Pruebas**

Para comprobar el funcionamiento del sistema se realizaron pruebas enviando cada uno de los tipos de disfunciones respiratorias (obstructivos, restrictivos, normales), desde el terminal móvil recibiendo el diagnóstico preliminar de cada uno de los pacientes almacenados en la base de datos dentro del servidor de la UCAB.

Se comprobó que la página web cumpliera con las necesidades de los médicos y que facilitara la búsqueda dentro de la base de datos de los pacientes.

- **Fase V: Elaboración del Tomo**

Una vez realizadas las pruebas que conectan la aplicación en el terminal móvil con la aplicación en la PC para obtener un diagnóstico del paciente y almacenar sus datos en la base de datos visualizados en la página web se procedió a la redacción y elaboración del tomo para sustentar la investigación y resultados del proyecto demostrando el cumplimiento de los objetivos siguiendo la metodología planteada al comienzo del mismo.

CAPÍTULO IV

Desarrollo

En el siguiente capítulo se desarrollan las fases establecidas en la metodología, para el cumplimiento de cada uno de los objetivos establecidos.

IV.1 Configuración del servidor.

La configuración de la red VPN se realizó entre una computadora personal que funcionaba como cliente y el servidor ubicado en la Universidad Católica Andrés Bello.

Por medidas de seguridad el personal de la DTI de la UCAB fue el encargado de realizar esta conexión virtual permitiéndo el acceso al servidor.

Para obtener la conexión remota a la VPN se procedió a instalar un programa el cual se llama VPN Client el cual permitió tener acceso privado al servidor, este fue descargado mediante el siguiente link proporcionado por la DTI.

http://helpdesk.ugent.be/vpn/download/VPNclient5_UGent.msi

Los datos facilitados por la DTI fueron una dirección IP, el grupo, la contraseña de grupo, el nombre de usuario y una clave

Por medio del programa VPN Client y de los datos proporcionados por la DTI se logró una nueva conexión VPN llamada accesoUCAB entre el servidor ubicado en la UCAB y una computadora personal.

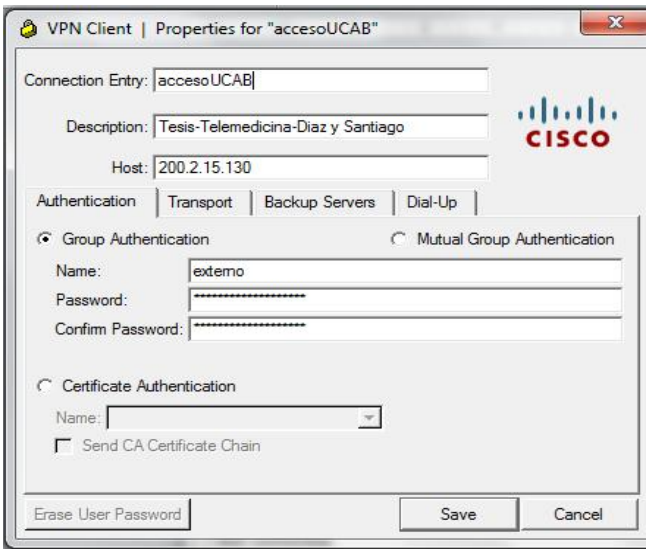


Figura 5. Configuración de VPN Client

Fuente Elaboración Propia

Luego de salvar los cambios se procedió a realizar la conexión la cual fue exitosa debido a que en la computadora apareció un mensaje de bienvenida.

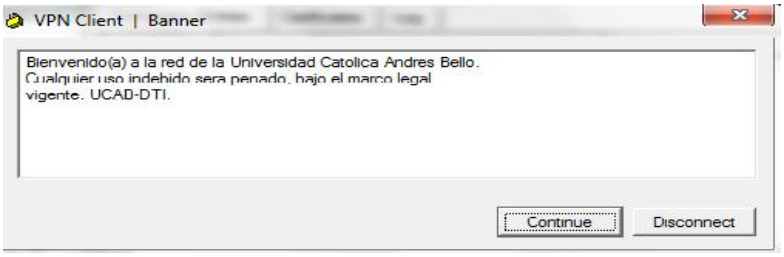


Figura 6.Mensaje de bienvenida al conectar con el servidor

Fuente Elaboración Propia

Al tener confirmada la conexión VPN se debió buscar una manera para interactuar con el servidor y manejar la base de datos, para lograr esto fue necesario descargar el programa llamado Putty, el cual es un programa para realizar conexiones con servidores remotos por línea de comandos.

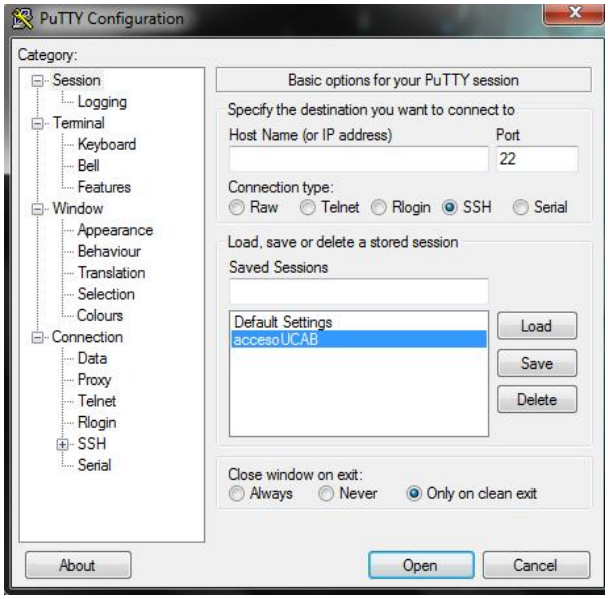


Figura 7.Configuración de Putty

Fuente Elaboración Propia

Luego de realizar las configuraciones necesarias se ingresó al servidor con un usuario y contraseña proporcionada por el personal la DTI, a continuación se muestra la interfaz del Putty en el cual se ingresó el usuario y contraseña.

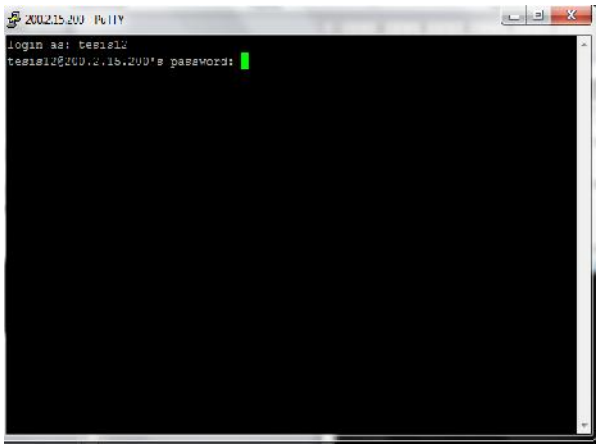


Figura 8. Usuario y Contraseña en Putty

Fuente Elaboración Propia

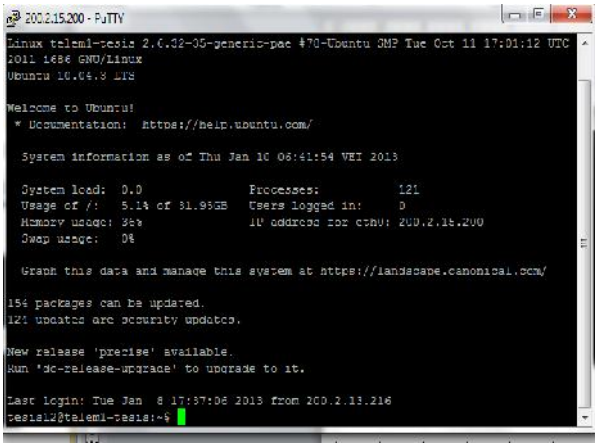


Figura 9. Ingresando al servidor a través de Putty.

Fuente Elaboración Propia

IV.2 Diseño de la base de datos.

Los datos de los pacientes utilizados para la creación de la base de datos fueron suministrados por el doctor Andrés Octavio y el ITMS (International Telemedical Systems) de Venezuela.

Luego de que se confirmó el ingreso al servidor y se recaudaron las muestras, se procedió a realizar el diseño y la creación de las tablas que están contenidas en la base de datos.

Los datos proporcionados por la DTI sobre la base datos que tenían que ver con el presente trabajo fueron los siguientes: nombre de la base de datos, usuario y clave.

Para acceder a la base de datos en el servidor se utilizó el siguiente comando

mysql -u usr_tesis12 –p

En la Figura 10 se puede observar que se logró ingresar exitosamente a MySQL.

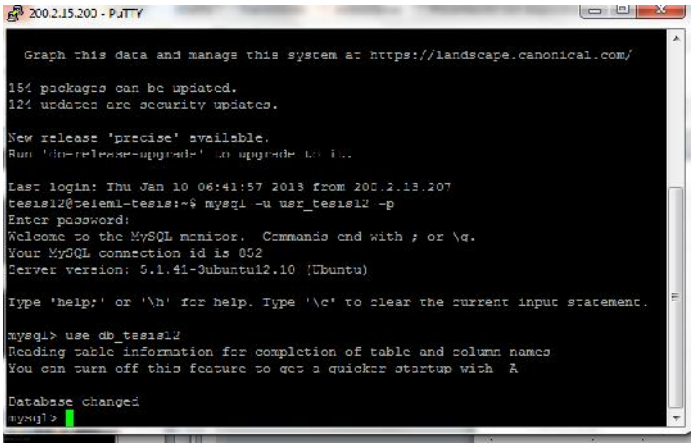


Figura 10.Ingresó a MySQL

Fuente Elaboración Propio

Para el diseño de la base de datos se utilizó el programa de MySQL Workbench en la Figura 11 se observa el diseño el cual incluye 3 tablas, los datos personales del paciente, los parámetros del examen de espirometría y la curva generada mediante dicho examen.

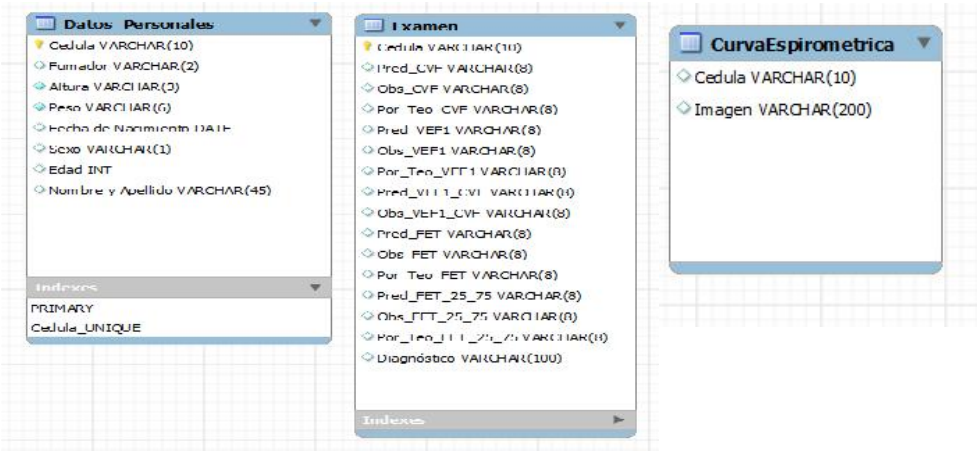


Figura 11.Diseño de la Base de Datos.

Fuente Elaboración Propia.

Se diseñó una tabla denominada Acceso_Portal la cual contenía la cédula y la contraseña del médico para acceder a la página web, en la Figura 12 se muestra el diseño.

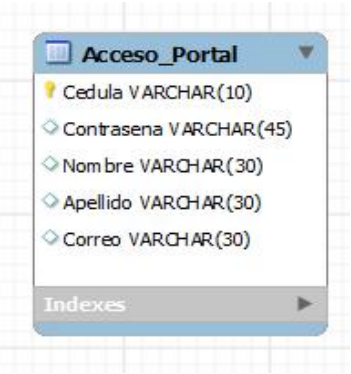


Figura 12.Diseño del Acceso_Portal

Fuente: Elaboración Propia.

Para acceder a las tablas de la base de datos en el servidor se utilizó el siguiente comando *show tables* en la Figura 13 se muestra las tablas.

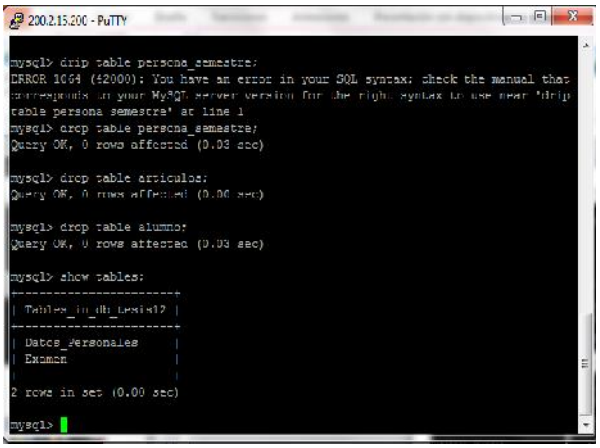


Figura 13.Tablas en la Base de Datos en Putty

Fuente Elaboración Propia

En el Figura 14 se muestra la conexión para poder interconectar el diseño de la base de datos creada en MySQL Workbench con Putty

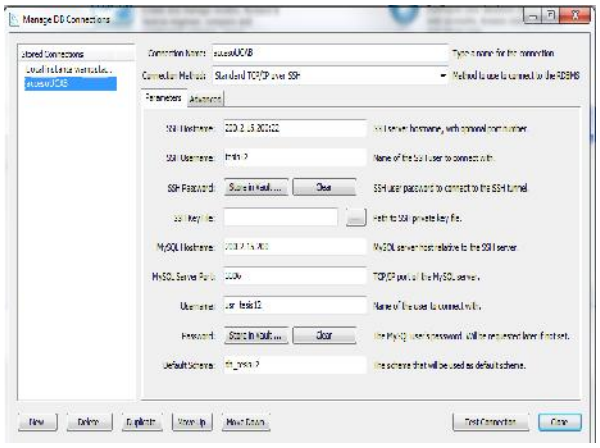


Figura 14.Conexión de MySQL Workbench

Fuente Elaboración Propia

En la Figura 15 se muestra la tabla de la base de datos que contiene los datos personales del paciente en el cual tiene cédula, nombre, altura sexo, peso, edad, fecha de nacimiento, fumador.

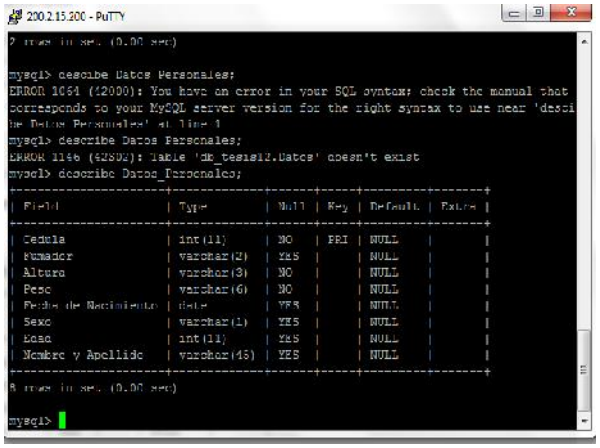


Figura 15.Tabla de Datos Personales del Paciente

Fuente Elaboración Propia.

IV.3 Diseño de la página web

La creación de la página web se realizó mediante el programa Dreamweaver ya que es una aplicación que permite controlar manualmente el código HTML mediante un entorno de diseño visual, permitiendo el fácil manejo y administración de sitios de páginas web.

Un sitio web es un conjunto de archivos y carpetas, relacionados entre sí, con un diseño similar o un objetivo común por lo que para crear un nuevo sitio en Dreamweaver y conectarlo con el servidor se tuvo que crear carpetas para poder almacenar las páginas generadas en este sitio, esto se realizó mediante el programa Putty utilizando la línea de comando *mkdir portal1* en esta carpeta se almacenó toda la información referente a la página web, también se debió crear otra carpeta dentro de portal1 llamada imágenes utilizando el mismo comando, como se muestra a continuación

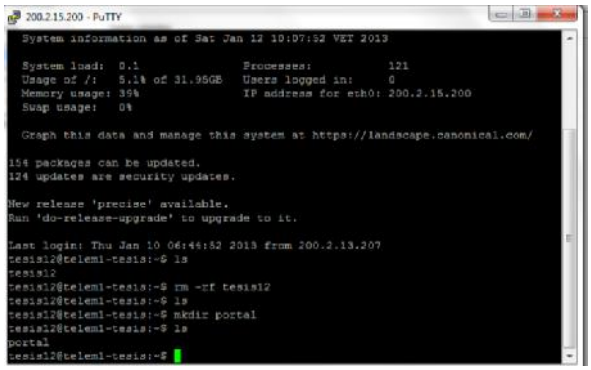


Figura 16.Creación de las carpetas en el servidor

Fuente: Elaboración Propia

Una vez realizadas las carpetas se procedió a crear el nuevo sitio, por lo que se realizó la configuración con el servidor web. Para la configuración de los datos locales se colocó el nombre el cual hace referencia al sitio y se busca la carpeta creada en el computador donde se cargaran los archivos que se sincronizarán con los

archivos del servidor; este mismo procedimiento se realizó con la carpeta de imágenes en el cual se guardarán las imágenes almacenadas en el sitio; en la dirección HTTP, se colocó la dirección de la carpeta con el mismo nombre que está ubicada en el servidor, como se muestra en la Figura 17.

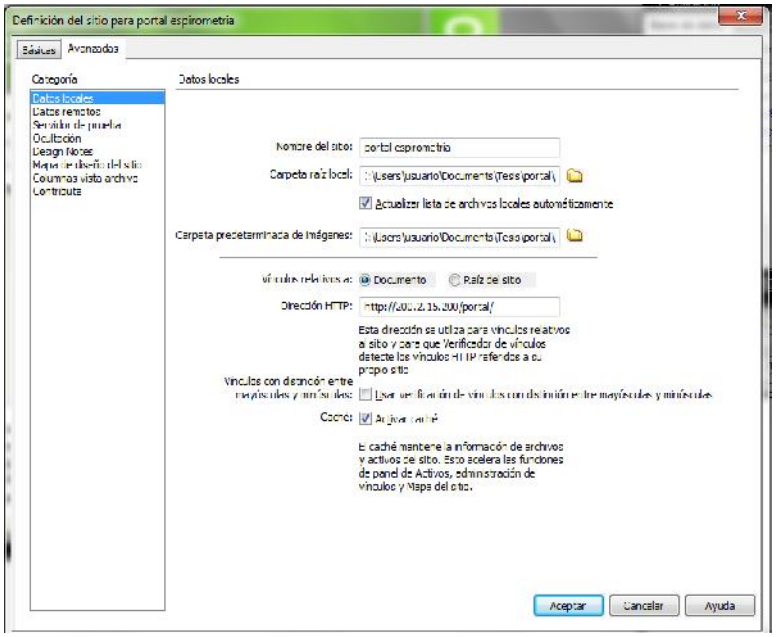


Figura 17.Configuración datos locales

Fuente: Elaboración Propia

Para la configuración de los datos remotos se seleccionó como método de acceso FTP para transferir archivos al directorio portal1 que está localizado en el servidor de la UCAB; una vez establecido este método se ingresó el usuario y contraseña establecidos por la DTI, además de seleccionar la opción “Utilizar_FTP seguro(SFTP)” como se observa en la Figura 18.

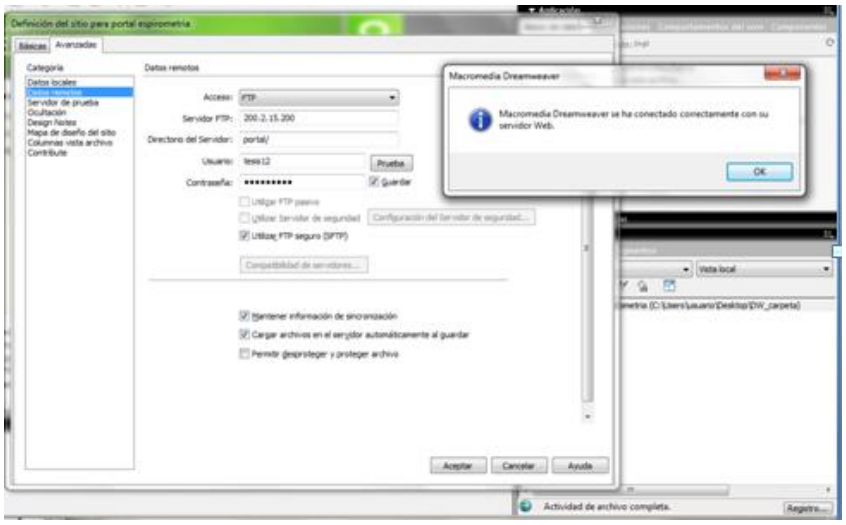


Figura 18.Configuración datos remotos

Fuente: Elaboración Propia

Luego para la configuración el servidor de prueba se utilizó como modelo de servidor PHP MySQL, mediante el acceso FTP a los archivos localizados en el servidor UCAB se ingresó con el mismo usuario y contraseña que en datos remotos y utilizando la opción FTP seguro (SFTP), el prefijo URL es la dirección del directorio de la carpeta donde se guardarán los archivos de la página web, como se observa en la Figura 19.

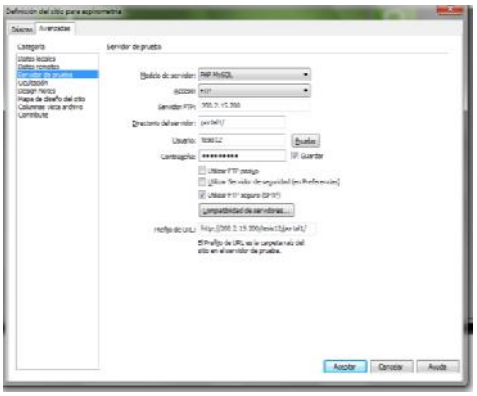


Figura 19.Servidor de Prueba

Fuente: Elaboración Propia

Para terminar el establecimiento de la conexión con el sitio en la sección de archivo, en el menú izquierdo de la pantalla se seleccionó el sitio y se le dio el botón azul para conectar al servidor remoto (Ver Figura 20).

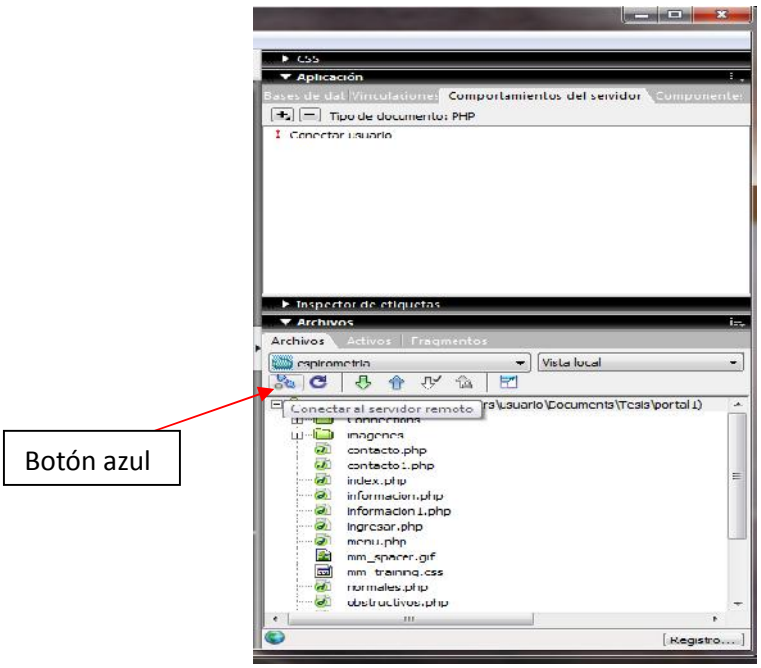


Figura 20.Conexión al servidor remoto

Fuente: Elaboración Propia

Una vez conectado al sitio se procedió a crear los archivos PHP que son las que poseen el diseño del contenido de las páginas web.

Para manejar la base de datos desde la página web se procedió a establecer la conexión; esto se realizó como se muestra en la Figura 21, colocando el nombre de conexión accesoUCAB, la IP del servidor MySQL y utilizando el nombre de usuario y contraseña proporcionados por la DTI.

Figura 21.Conexión con MySQL

Fuente: Elaboración Propia

En la Figura 22 se muestra todas las tablas contenidas en la base de datos que se puede utilizar en la página web mediante la conexión establecida en la Figura 21.

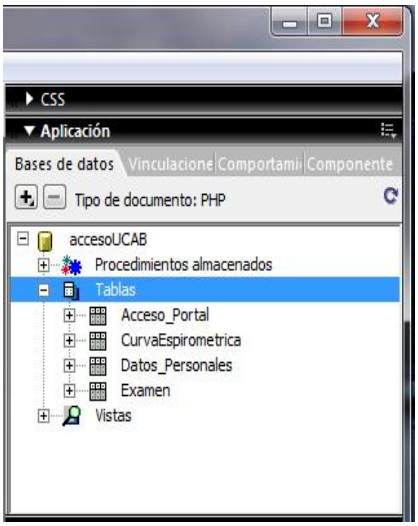


Figura 22. Tablas de la base de datos en Dreamweaver

Fuente: Elaboración Propia

Para validar los usuarios de entrada se utilizó formularios, autenticándolos mediante la verificación de sus datos en la base de datos; para esto, en la sección de aplicación se seleccionó comportamiento del servidor/autenticación de usuario/conectar usuario, donde apareció una ventana con diferentes opciones como se muestra en la Figura 23.

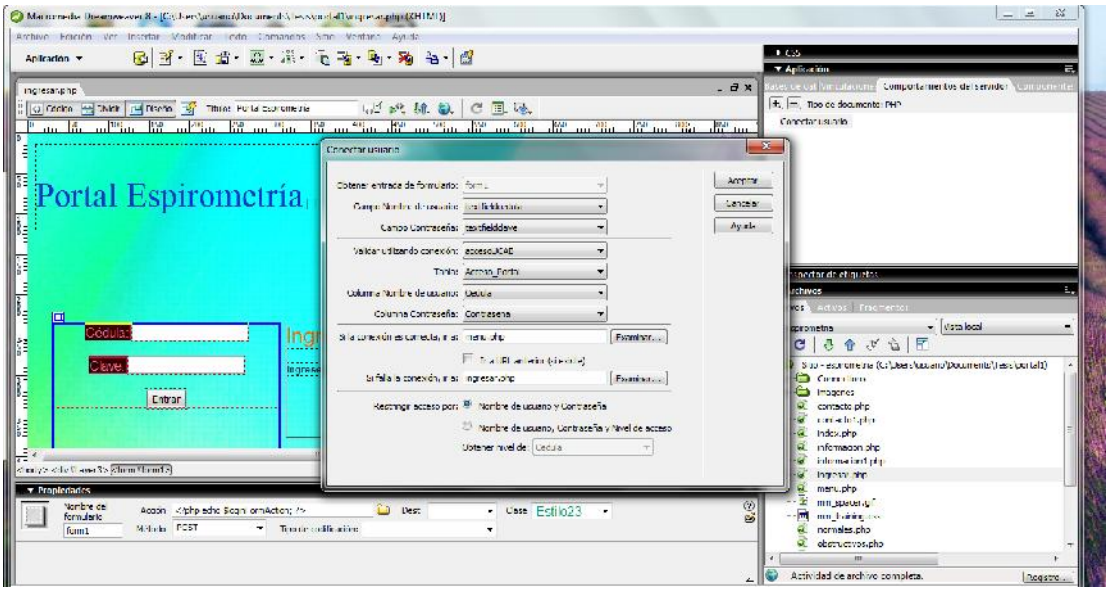


Figura 23.Validar usuarios

Fuente: Elaboración Propia

En la Figura 23 en el campo de usuario y contraseña se seleccionó textfile correspondientes, se validó utilizando la conexión a la base de datos creada previamente, se seleccionó la tabla en la base de datos y la columna de la tabla a la cual se hará referencia y con las cuales se verificaron su existencia o no con los valores ingresados, luego si la conexión fue exitosa se le asigna el archivo al cual se quiere redireccionar y de la misma forma se aplica el mismo procedimiento si la conexión es fallida.

Para consultar los casos de los pacientes según su diagnóstico se crearon hipervínculos que llevaran a páginas donde se mostró la tabla de exámenes con un diagnóstico en específico; para ello se crearon tablas dinámicas las cuales necesitan juegos de registros que se crean en la sección de aplicación/vinculaciones, luego se selecciona el icono con la opción juego de registros (consulta) y aparecerá lo que se muestra en la Figura 24.

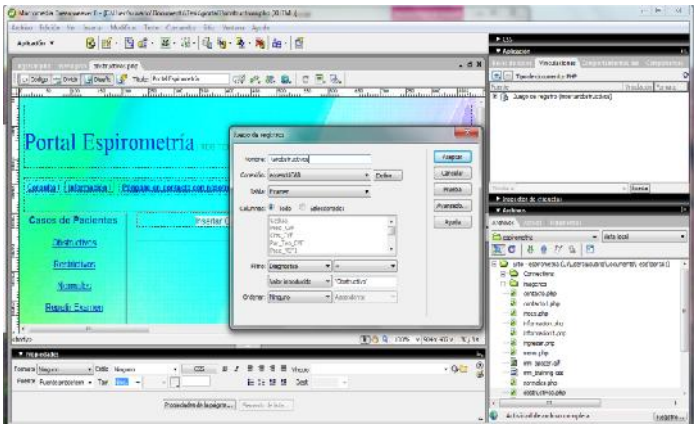


Figura 24.Tablas Dinámicas

Fuente: Elaboración Propia.

IV.4 Aplicación en Java.

Se realizó una aplicación en Java para leer los datos del paciente provenientes de una aplicación en el terminal móvil.

Para guardar los datos provenientes del terminal móvil en la base de datos se procedió a crear la conexión, se importaron los drivers de MySQL y se creó la autenticación del usuario con los datos proporcionados por la DTI, como se puede apreciar en la Figura 25.

```
public ConexionBD() {
    try {
        Class.forName("org.gjt.mm.mysql.Driver"); // se importan los driver mysql
        String basededatos = "db_tesis12";
        String usuario = "usr_tesis12";
        String password = "#tele389#";
        //Comando que me permite comunicarme por medio del driver sin try catch hay error de reporte
        conexion = DriverManager.getConnection("jdbc:mysql://200.2.15.200:3306/" + basededatos, usuario, password);
        conexion2= DriverManager.getConnection("jdbc:mysql://200.2.15.200:3306/" + basededatos, usuario, password);
        System.out.println("conectado");

    } catch (ClassNotFoundException ex) {
        System.out.println("Error en el Driver");

    } catch (SQLException ex) {

        System.out.println("Error en la Conexion");
    }



}
```

Figura 25. Conexión a la base de datos

Fuente: Elaboración Propia

Una vez establecida la conexión con la base de datos se buscó el archivo que contiene los parámetros del examen de espirometría, los leyó y realizó la comparación de los parámetros para establecer el diagnóstico del paciente y guarda los datos en la base de datos. (Ver Figura 26 y 27)

```
// busca archivo con los parámetros del examen de espirometria de un usuario
File archivo = new File("C:/Users/usuario/Documents/Tesis/muestras/examenpaciente2.txt");
FileInputStream archivostream = new FileInputStream(archivo);
DataInputStream entrada = new DataInputStream(archivostream);
BufferedReader lector = new BufferedReader(new InputStreamReader(entrada));
String strLinea;
```

Figura 26. Busca archivo con los parámetros del examen de espirometría de un usuario.

Fuente: Elaboración Propia

```
//      comparación de los valores para dar un diagnostico aproximado
        if (Obs_VEF1_CVF_decimal <= 70) {

            resultado_diagnostico = "Obstructivo.";
            if (Por_Teo_CVF_decimal >= 60 && Por_Teo_VEF1_decimal >= 60) {
                resultado_diagnostico = "Obstructivo Leve.";
                aux = 77;
            }
        }
        if ((Obs_VEF1_CVF_decimal > 70) && (Por_Teo_CVF_decimal < 80) && (Por_Teo_VEF1_decimal < 80)) {
            if ((Por_Teo_CVF_decimal >= 60) && (Por_Teo_VEF1_decimal >= 60)) {
                resultado_diagnostico = "Restrictivo Leve.";
                aux = 77;
            }
            if ((Por_Teo_CVF_decimal < 60) && (Por_Teo_CVF_decimal >= 40) && (Por_Teo_VEF1_decimal >= 60) && (Por_Teo_VEF1_decimal >= 40)) {
                resultado_diagnostico = "Restrictivo Moderado.";
                aux = 77;
            }
            if ((Por_Teo_CVF_decimal < 40) && (Por_Teo_VEF1_decimal < 40)) {
                resultado_diagnostico = "Restrictivo Severo.";
                aux = 77;
            }
        }
        if ((Obs_VEF1_CVF_decimal < 70) && (Por_Teo_CVF_decimal < 80) && (Por_Teo_VEF1_decimal < 80)) {
            resultado_diagnostico = "Mixto";
            aux = 77;
        }
        if ((Obs_VEF1_CVF_decimal > 70) && (Por_Teo_CVF_decimal > 80) && (Por_Teo_VEF1_decimal > 80)) {
            resultado_diagnostico = "Normal";
            aux = 77;
        }
        if (aux != 77) {
            resultado_diagnostico = "El examen no cumple criterios de aceptabilidad y reproducibilidad se sugiere repetir.";
        }
        return resultado_diagnostico;
    }
```

Figura 27.Comparación de los valores para dar un diagnóstico del paciente.

Fuente: Elaboración Propia

```
//          guardar los datos en la tabla Examen en la base de datos
                conectar.insertarTabla(cedula, Pred_CVF, Obs_CVF, Por_Teo_CVF, Pred_VEF1, Obs_VEF1, Por_Teo_VEF1,
                        Pred_VEF1_CVF, Obs_VEF1_CVF, Pred_FEF, Obs_FEF, Por_Teo_FEF, Pred_FEF_25_75,
                        Obs_FEF_25_75, Por_Teo_FEF_25_75, Diagnostico);
```

Figura 28.Guarda los datos en la tabla Examen en la base de datos

Fuente: Elaboración Propia

```
//          inserta los parametros del examen en la tabla Examen de la base de datos
            try {Statement consultaInsertar = conexion.createStatement();}
               consultaInsertar.executeUpdate("INSERT into Examen VALUES('" + cedula + "','" + Pred_CVF + "','" + Obs_CVF + "','"
                       + Por_Tco_CVF + "','" + Pred_VEF1 + "','" + Obs_VEF1 + "','" + Por_Tco_VEF1 + "','" + Pred_VEF1_CVF + "','"
                       + Obs_VEF1_CVF + "','" + Pred_FET + "','" + Obs_FET + "','" + Por_Tco_FET + "','" + Pred_FET_25_75 + "','"
                       + Obs_FET_25_75 + "','" + Por_Tco_FET_25_75 + "','" + diagnosticoresultado + "')");
//             verifico que la insercion fue exitosa
               System.out.println("Se inserto los datos  del Examen");
            }
        } catch (SQLException ex) {
            System.out.println("Error en la insersion de Examen");
        }
```

Figura 29. Inserta los parámetros del examen en la tabla Examen de la base de datos.

Fuente: Elaboración Propia

Y luego guarda los datos personales del paciente en la tabla Datos_Personales de la base de datos.

```
//          busca archivo txt con los datos personales del usuario que se realizó el examen de espirometria
            File archivopaciente = new File("C:/Users/usuario/Documents/Tesis/muestras/datospaciente1.txt");
            FileInputStream archivostream = new FileInputStream(archivopaciente);
            DataInputStream entrada = new DataInputStream(archivostream);
            BufferedReader lector = new BufferedReader(new InputStreamReader(entrada));
            String strLinea;
```

Figura 30. Busca archivo txt con los datos personales del usuario

Fuente: Elaboración Propia

```
//          guardar los datos en la tabla Examen en la base de datos
            conectar.insertarTabla2(Cedula, Fumador, Altura, Peso, FechaNacimiento, Sexo, Edad, NombreApellido);
```

Figura 31.Guarda los datos en la tabla Datos_Personales en la base de datos

Fuente: Elaboración Propia

```
inserta los datos personales del usuario en la tabla Datos_Personales de la base de datos

Statement consultaInsertar = conexion2.createStatement();
consultaInsertar.executeUpdate("INSERT into Datos_Personales VALUES('" + cedula + "','" + fumador + "','" + altura + "','" + peso
        + "','" + fechanacimiento + "','" + sexo + "','" + edad + "','" + nombreapellido + "')");
 verifica que la inserción fue exitosa
System.out.println("=====================================");
System.out.println("Se inserto paciente");
```

Figura 32. Inserta los datos personales del usuario en la tabla Datos_Personales de la base de datos.

Fuente: Elaboración Propia

Se creó una interfaz en Java para el fácil manejo de esta aplicación, esta interfaz contó con tres botones que se van a ir activando a medida que se envían los exámenes, a continuación se muestra como se deshabilitaron los botones.

```
public TEGEspirometria() {
    initComponents();
    getIPLocalhost();

    //Deshabilita los botones y direccion IP.
    enviarCelButton.setEnabled(false);
    cargarBDButton.setEnabled(false);
    dirIPTextField.setEnabled(false);
}
```

Figura 33. Activación de los botones

Fuente: Elaboración Propia

La ventana creada en la aplicación muestra la dirección IP de la computadora para que trabaje como servidor y reciba los datos personales y el examen de espirometría proveniente del terminal móvil. En la Figura 34 se puede apreciar el código utilizado.

```
//Da la IP de la máquina local.
public void getIPLocalhost(){

    InetAddress localHost = null;
        try {
            localHost = InetAddress.getLocalHost();
            dirIPTextField.setText(localHost.getHostAddress());

        } catch (UnknownHostException ex) {

        }
}
```

Figura 34.Código para mostrar la IP de la maquina local

Fuente: Elaboración Propia.

En la Figura 35 se muestra como la computadora funciona como servidor recibiendo los datos provenientes del cliente; lo primero que se realizó fue crear el socket, y se esperó la conexión por parte del cliente, una vez que se confirmó la conexión con el cliente, leyó y guardó todos los datos provenientes del cliente en mensaje recibido.

```
    public void initServer() {
BufferedReader entrada;


try {
so = new ServerSocket(PUERTO );/* Crea socket servidor que escuchara en puerto 5000*/
so=new Socket();
System.out.println("Esperando una conexion:");
so = so.accept();
//Inicia el socket, ahora esta esperando una conexión por parte del cliente
System.out.println("Un cliente se ha conectado.");
//Canal de entrada  de datos
entrada = new BufferedReader(new InputStreamReader(so.getInputStream()));
System.out.println("Confirmando conexion al cliente....");
mensajeRecibido = entrada.readLine();
guardarDatos(mensajeRecibido);
System.out.println(mensajeRecibido);
horactual=getTiempo();
informacionTextArea.append("\n"+"["+horactual+"]"+":"+"Sus datos han sido recibido exitósamente ");
System.out.println("Cerrando conexión...");
so.close();//Aqui se cierra la conexión con el cliente
}catch(Exception e ) {
    horactual=getTiempo();
informacionTextArea.append("\n"+"["+horactual+"]"+"Error: "+e.getMessage());
System.out.println("["+horactual+"]"+"Error: "+e.getMessage());
}
}
```

Figura 35.Inicialización del Servidor

Fuente: Elaboración Propia

```
public void initClient(String resultadoexamen) {
try
{
HOST=dirIPTextField.getText().toString();
so = new Socket(HOST,PUERTO); /*conectar a un servidor en localhost con puerto 5000*/
//creamos el flujo de datos por el que se enviara un mensaje
mensaje = new DataOutputStream(so.getOutputStream());
//enviamos el mensaje
mensaje.write(resultadoexamen.getBytes());
//cerramos la conexión
so.close();
informacionTextArea.append("\n"+"["+horactual+"]"+":"+"Su diágnostico fue enviado.");
}catch(Exception e )
{
horactual=getTiempo();
informacionTextArea.append("\n"+"["+horactual+"]"+"Error: "+e.getMessage());
System.out.println("\n"+"["+horactual+"]"+"Error: "+e.getMessage());
}
```

Figura 36.Inicialización del cliente

Fuente: Elaboración Propia

Todos los mensajes que aparecen en la interfaz contienen la hora en la cual se realizó la actividad; en la Figura 37 se muestra el código utilizado para esto.

```
public String getTiempo() {
        Calendar Cal = Calendar.getInstance();
        String tiempo = Cal.get(Cal.HOUR_OF_DAY) + ":" + Cal.get(Cal.MINUTE);
        String hora = tiempo;
        return hora;
    }
```

Figura 37.Insertar hora

Fuente: Elaboración Propia

En la Figura 38 se muestra como los datos recibidos en la computadora desde el terminal móvil son guardados en la dirección de la computadora en un archivo txt.

```
public void guardardatoscel(String datosrecibidoscel){
        {
        FileWriter fichero = null;
        PrintWriter pw = null;
        try
        {
            fichero = new FileWriter("C:/Users/usuario/Documents/Tesis/muestras/datoscel.txt");
            pw = new PrintWriter(fichero);

                pw.println(datosrecibidoscel);
```

Figura 38.Guardar datos provenientes del terminal móvil

Fuente: Elaboración Propia

IV.5 Aplicación en Android.

Se desarrolló una aplicación en Android que funcionó como cliente/ servidor la cual envió los datos personales del paciente y los parámetros del examen de espirometría a la aplicación en Java en donde guardó los datos en la base de datos y recibió en el terminal móvil un diagnóstico preliminar, para esto se utilizó sockets usando el puerto 46000. En la Figura 39 se muestra como se abrió el socket desde Android, se crea el sockets *sock* y se le pasa la dirección IP proveniente de un EditText donde se introduce la dirección IP de la computadora que funcionó como servidor y se le asignó el puerto 46000 de forma aleatoria el cual tiene que ser el mismo que se estableció en la aplicación de Java, se creó un *OutputStream* donde se van enviar en una cadena de bytes los strings de los datos del paciente.

La red WiFi utilizada para abrir lo sockets fue una red local proveniente de un router conectado a un modem ABA de CANTV

```
sock = new Socket(DirIP.getText().toString(), 46000);

OutputStream os = sock.getOutputStream();

String  enviardatos= nombre.getText().toString()+";"+cedula.getText().toString()+";"+
                     edad.getText().toString()+";"+altura.getText().toString()+";"+peso.getText().toString()+";"+
                     fenacimiento.getText().toString()+";"+sexo.getText().toString()+";"+fumador.getText().toString();
os.write(enviardatos.getBytes());
os.flush();
Log.e("cliente", "flush finalizado");
os.close();
```

Figura 39.Conexión cliente Android

Fuente: Elaboración Propia

CAPITULO V

Resultados

En el siguiente capítulo se presenta los resultados obtenidos para el cumplimiento de este Trabajo Especial de Grado, se muestra la conexión de la VPN, la base de dato, y la página web.

V.1 Red VPN

Una vez realizada la configuración tanto del lado del servidor como del lado del cliente se procedió a verificar la conexión haciendo Ping entre ambos. En la Figura 40 se comprobó la conexión entre cliente servidor haciendo ping al servidor mediante command prompt (Símbolo del Sistema) con dirección IP 200.2.15.200.

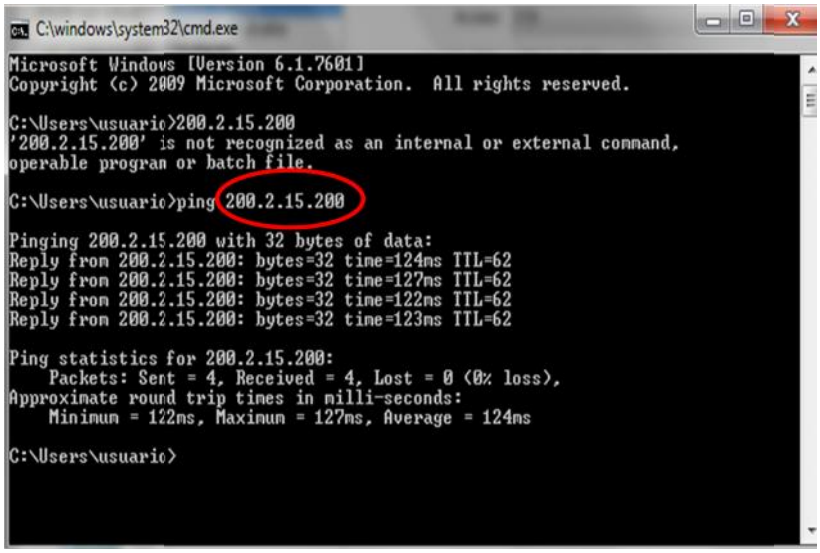


Figura 40.Envió de ping desde el cliente al servidor

Fuente: Elaboración Propia

En la Figura 41 se comprobó la conexión entre servidor cliente haciendo ping al cliente con dirección IP 200.2.15.130 desde el programa Putty.

```
200.2.15.200 - PuTTY
  Memory usage: 40%                IP address for eth0: 200.2.15.200
  Swap usage:   0%

  Graph this data and manage this system at https://landscape.canonical.com/

161 packages can be updated.
131 updates are security updates.

New release 'precise' available.
Run 'do-release-upgrade' to upgrade to it.

Last login: Sat Feb  9 11:45:33 2013 from 200.2.13.216
tesis12@telem1-tesis:~$ ping 200.2.15.130
PING 200.2.15.130 (200.2.15.130) 56(84) bytes of data.
64 bytes from 200.2.15.130: icmp_seq=1 ttl=128 time=0.262 ms
64 bytes from 200.2.15.130: icmp_seq=2 ttl=128 time=0.235 ms
64 bytes from 200.2.15.130: icmp_seq=3 ttl=128 time=0.255 ms
64 bytes from 200.2.15.130: icmp_seq=4 ttl=128 time=0.239 ms
64 bytes from 200.2.15.130: icmp_seq=5 ttl=128 time=0.258 ms
64 bytes from 200.2.15.130: icmp_seq=6 ttl=128 time=0.253 ms
64 bytes from 200.2.15.130: icmp_seq=7 ttl=128 time=0.262 ms
64 bytes from 200.2.15.130: icmp_seq=8 ttl=128 time=0.312 ms
64 bytes from 200.2.15.130: icmp_seq=9 ttl=128 time=0.261 ms
```

Figura 41.Envió de ping desde el servidor al cliente

Fuente: Elaboración Propia

V.2 Base de datos

Este proyecto está compuesto por cuatro bases de datos creadas en MySQL como se muestra en la Figura 42.

```
200.2.15.200 - PuTTY
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 3214
Server version: 5.1.41-3ubuntu12.10 (Ubuntu)

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> use db_tesis12
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
mysql> show tables;
+----------------------+
| Tables_in_db_tesis12 |
+----------------------+
| Acceso_Portal        |
| CurvaEspirometrica   |
| Datos_Personales     |
| Examen               |
+----------------------+
4 rows in set (0.00 sec)

mysql>
```

Figura 42.Tablas base de datos

Fuente: Elaboración Propia

En la base de datos 1 llamada Acceso_Portal están contenidas todas las personas que .pueden acceder a la aplicación.

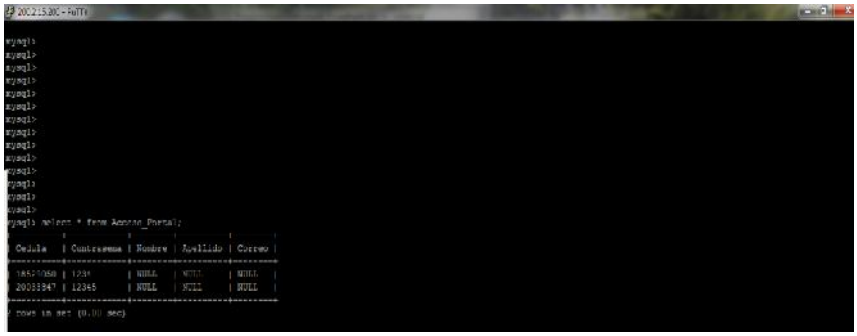


Figura 43.Tabla Acceso_Portal

Fuente: Elaboración Propia

En la base de datos 2 llamada CurvaEspirométrica se muestra la imagen de cada examen espirométrico realizado a los pacientes.

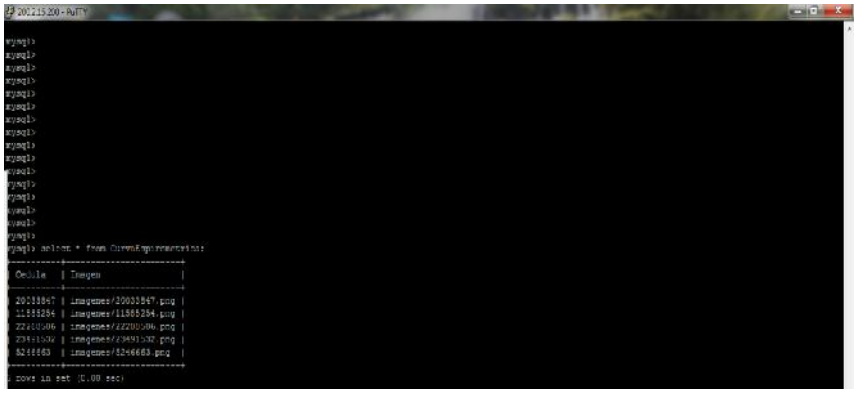


Figura 44.Tabla Curvaespirométrica

Fuente: Elaboración Propia

En la base de datos 3 llamada Datos_Personales se muestra los datos de cada uno de los pacientes a los cuales se les realizó este examen.

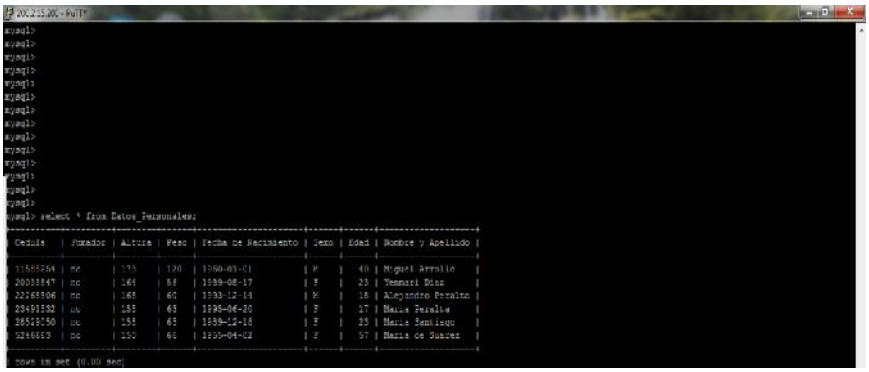


Figura 45.Tabla Datos_Personales

Fuente: Elaboración Propia

En la base de datos 4 llamada Examen se muestra los parámetros y el posible diagnóstico de cada examen espirométrico realizados a los pacientes.

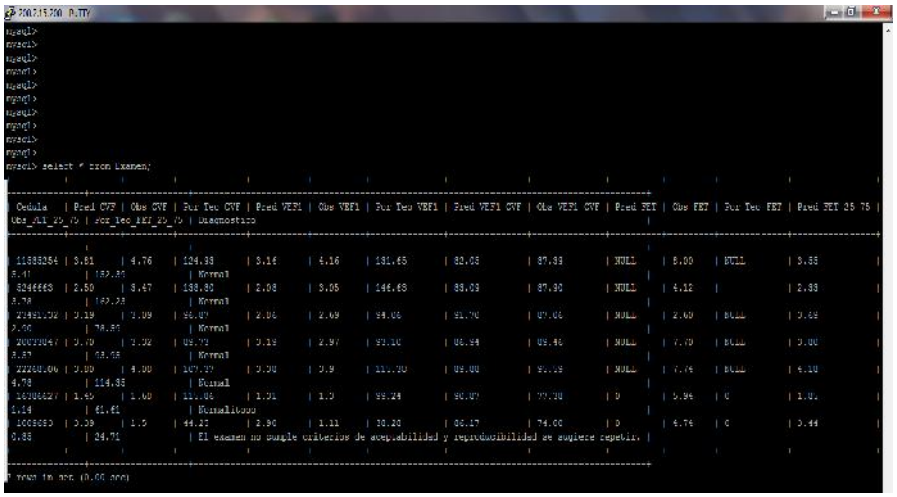


Figura 46.Tabla Examen

Fuente: Elaboración Propia.

V.3 Página web

Una vez realizada la base de datos y establecida la conexión cliente servidor se procedió a crear la página web, el resultado de la misma se muestra a continuación.

En la Figura 47 se muestra la página de inicio, esta es la primera página que tiene acceso el usuario cuando desea ingresar a la plataforma

Figura 47.Página de Inicio

Fuente: Elaboración Propia.

Si se desea ingresar al sistema se presentará una ventana solicitando la cédula y contraseña como se muestra en la Figura 48, sino se introducen los datos correctamente no se tendrá acceso a la aplicación.

Figura 48.Página de acceso

Fuente: Elaboración Propia.

Si el usuario no posee clave y desea ingresar tendrá que registrarse, colocando sus datos personales como se puede apreciar en la Figura 49.

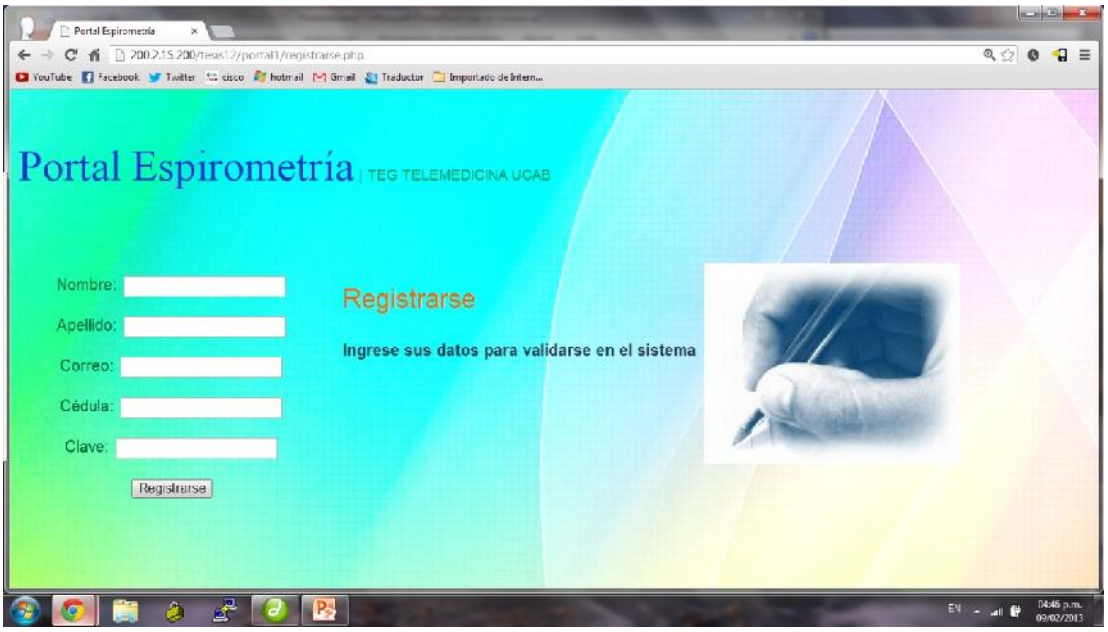


Figura 49.Página de Registro

Fuente: Elaboración Propia.

Todas las personas que accedan correctamente a la aplicación podrán tener acceso a todas las páginas restantes.

En el extremo superior izquierdo se encontrará tres botones los cuales se encontrarán fijos para el resto de las páginas.

El botón 1 es el de consulta en el cual se podrán consultar los datos y exámenes realizados a pacientes.

El botón 2 es el de información en el cual mostrará las definiciones de los parámetros existentes dentro del examen espirométrico.

El botón 3 es el de póngase en contacto con nosotros, en este se mostrará los datos de los administradores por si existe algún problema.

En la página de consulta en el lado izquierdo se mostrará un menú en el cual se puede apreciar los diferentes casos de disfunciones respiratorias presentes dentro de un examen de espirometría, como se observa a continuación.

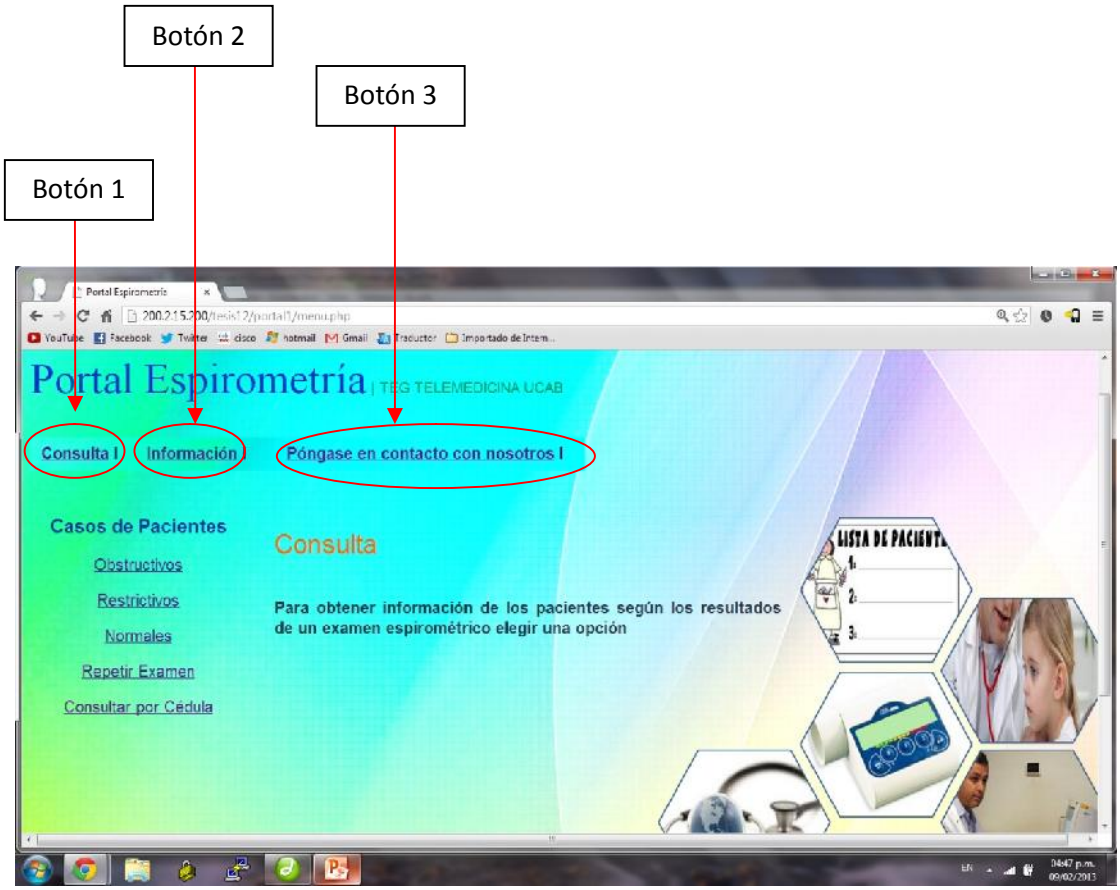
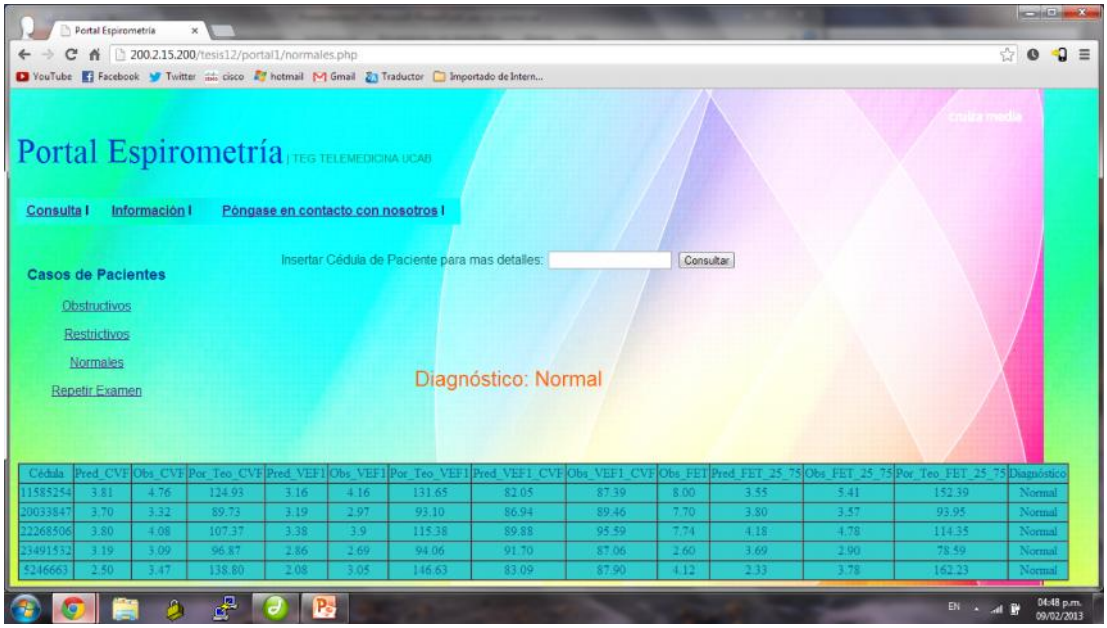


Figura 50.Página de Consulta

Fuente: Elaboración Propia

Una vez seleccionado uno de los casos aparecerá una tabla con la cédula de todos los pacientes a quienes se le diagnosticó ese tipo de disfunción respiratoria con los respectivos parámetros presentes dentro del examen de espirometría, si se desea ver con más detalle a un paciente en particular en la parte superior se podrá colocar el número de cédula del paciente como se puede apreciar en la Figura 51.



Cédula	Pred_CVF	Obs_CVF	Por_Teo_CVF	Pred_VEF1	Obs_VEF1	Por_Teo_VEF1	Pred_VEF1_CVF	Obs_VEF1_CVF	Obs_FET	Pred_FET_25_75	Obs_FET_25_75	Por_Teo_FET_25_75	Diagnóstico
11585254	3.81	4.76	124.93	3.16	4.16	131.65	82.05	87.39	8.00	3.55	5.41	152.39	Normal
20033847	3.70	3.32	89.73	3.19	2.97	93.10	86.94	89.46	7.70	3.80	3.57	93.95	Normal
22268506	3.80	4.08	107.37	3.38	3.9	115.38	89.88	95.59	7.74	4.18	4.78	114.35	Normal
23491532	3.19	3.09	96.87	2.86	2.69	94.06	91.70	87.06	2.60	3.69	2.90	78.59	Normal
5246663	2.50	3.47	138.80	2.08	3.05	146.63	83.09	87.90	4.12	2.33	3.78	162.23	Normal

Figura 51.Página que presenta el diagnóstico normal

Fuente: Elaboración Propia.

En el menú izquierdo se muestra un botón en el cual se podrá consultar por cédula a los pacientes como se puede apreciar en la Figura 52 y así ver con más detalle su examen espirométrico en el cual mostrará la curva flujo volumen y la curva volumen tiempo, así como los datos personales del paciente y los parámetros presentes en el examen como se observa en la Figura 53.

Figura 52.Página de Consulta

Fuente: Elaboración Propia

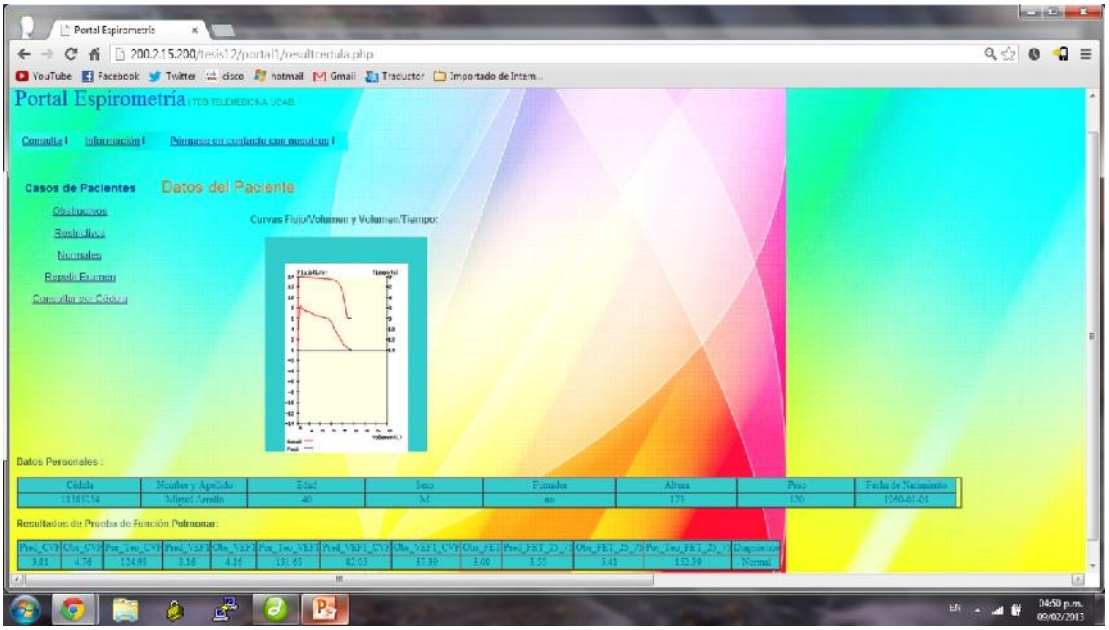


Figura 53.Datos del Paciente

Fuente: Elaboración Propia.

En la Figura 54 se muestra la página de información la cual contiene como están representadas las curvas flujo volumen y volumen tiempo, con su respectivo significado además cuenta con las definiciones de los principales parámetros que se obtienen en un examen de espirometría.

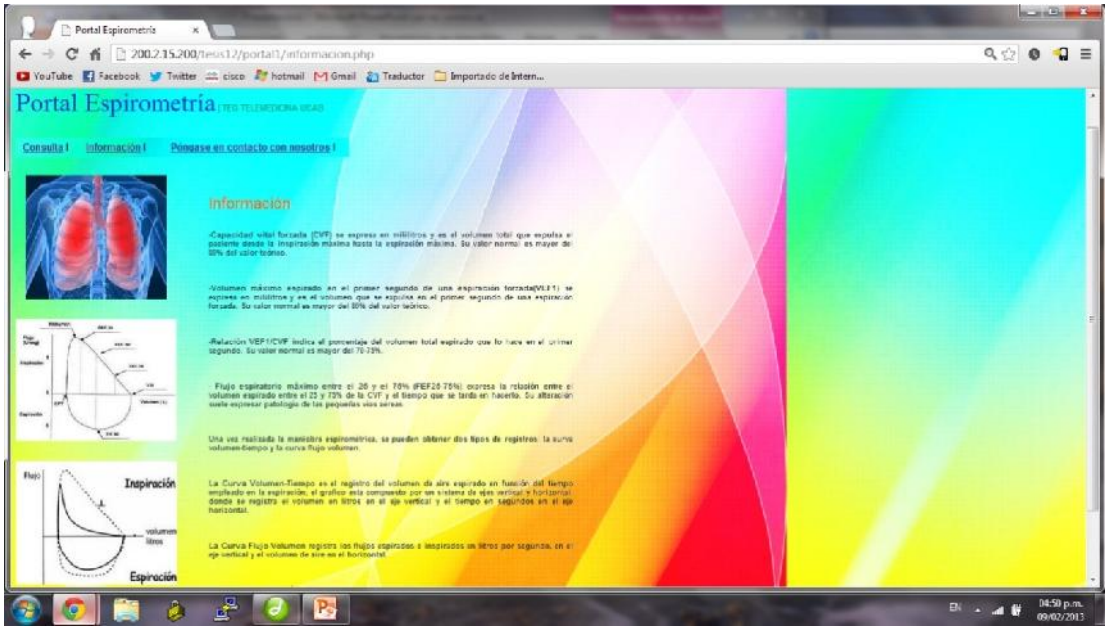


Figura 54.Página de Información

Fuente: Elaboración Propia

En la Figura 55 se puede apreciar la página de contactos, la cual contiene los datos tanto de la institución como de los creadores de la aplicación.

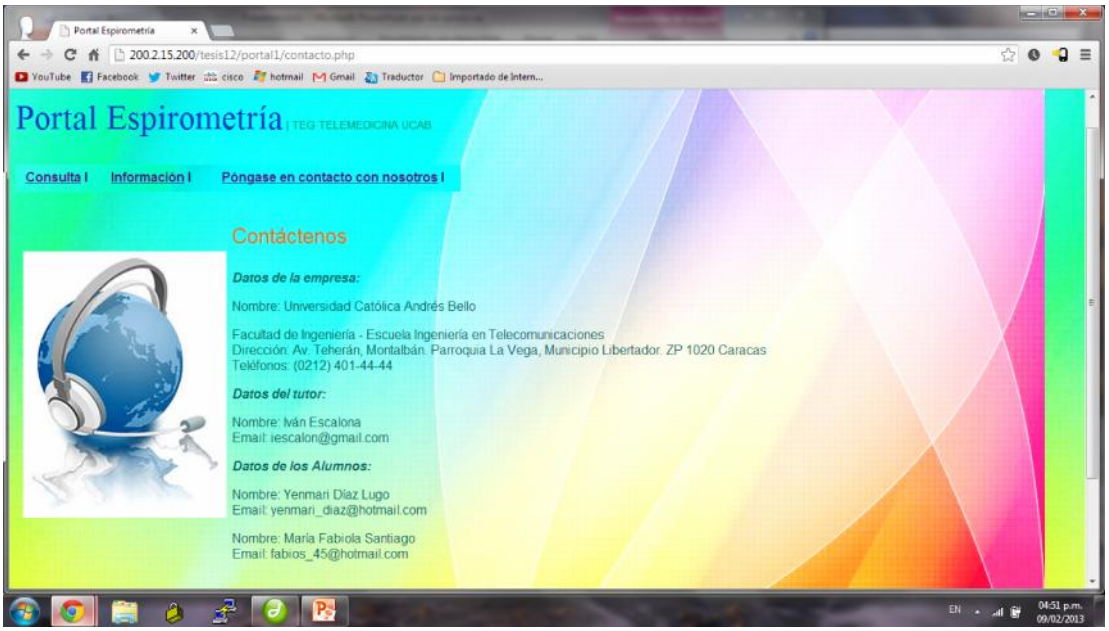


Figura 55.Página de contactos.

Fuente: Elaboración Propia

V.4 Aplicación en Java

Una vez realizada la base de datos y la página web se procedió a crear la aplicación en Java; esta aplicación permitió establecer una conexión entre el teléfono móvil y la computadora, logrando enviar los exámenes de espirometría y recibir de ella el diagnóstico preliminar del mismo. Los resultados de esta aplicación se presentan a continuación.

En la Figura 56 se muestra la ventana de inicio de la aplicación en la cual se colocó la dirección IP modo servidor para recibir los datos del paciente y el examen de espirometría provenientes del terminal móvil; esta ventana contiene tres botones fijos los cuales se van activando a medida de que corra la aplicación. El botón de

Recibir Datos se presionó una vez que se colocó la dirección IP de modo servidor y se esperó a recibir los datos provenientes del terminal móvil.

Figura 56.Ventana de Inicio

Fuente: Elaboración propia

Una vez enviados los datos del terminal móvil, apareció en la ventana de la aplicación la hora a la cual fue realizada el envío de los datos y un mensaje diciendo que los datos fueron recibidos de manera exitosa como se muestra a continuación.

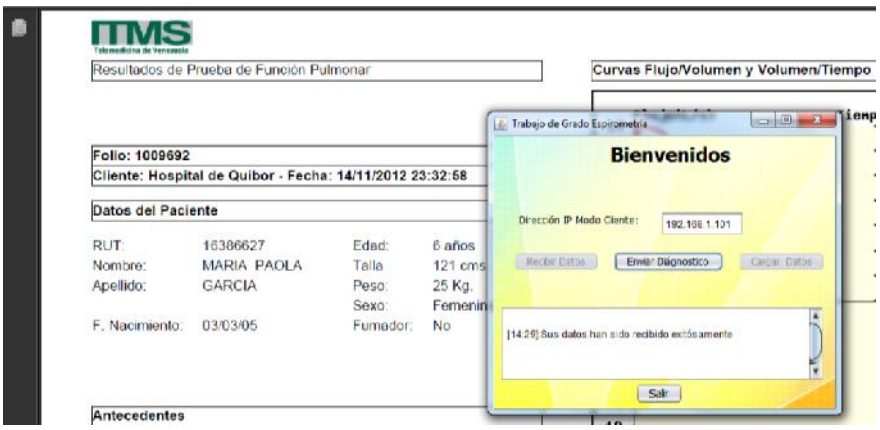


Figura 57.Ventana de recepción de datos en la aplicación mostrando las datos del paciente.

Fuente: Elaboración Propia

Para enviar el diagnóstico al terminal móvil, se debió colocar en la pantalla de inicio la dirección IP del terminal móvil y presionar el botón de enviar diagnóstico como se muestra en la Figura 58.

Figura 58.Envió del Diagnostico al terminal móvil

Fuente: Elaboración Propia

En la Figura 59 se muestra la dirección en la computadora, donde se guarda todos los datos enviados desde el terminal móvil.

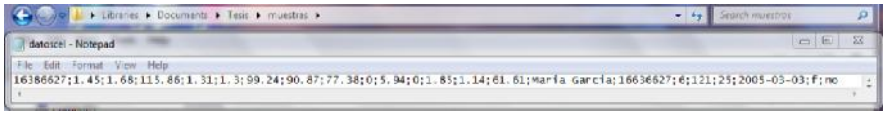


Figura 59.Dirección donde se guarda el archivo en la computadora

Fuente Elaboración Propia

Una vez enviado el diagnóstico se mostró en pantalla que el mismo fue enviado con éxito al terminal móvil y apareció el resultado, que en el caso de este paciente tiene un diagnostico normal, y automáticamente se activa el último botón el cual se refiere a cargar los datos en la base de datos.

Figura 60.Envió de diagnostico

Fuente Elaboración Propia.

Para enviar los datos a la base de datos es necesario primero conectarse con la VPN, una vez establecida la conexión aparecerá en pantalla un mensaje diciendo que se insertaron los datos en el sistema como se muestra a continuación.

Figura 61.Insertar datos en el sistema.

Fuente: Elaboración Propia.

En la Figura 62 se muestra que los datos enviados por el terminal móvil en este caso la del paciente de 6 años son insertados de manera exitosa en la base de datos.

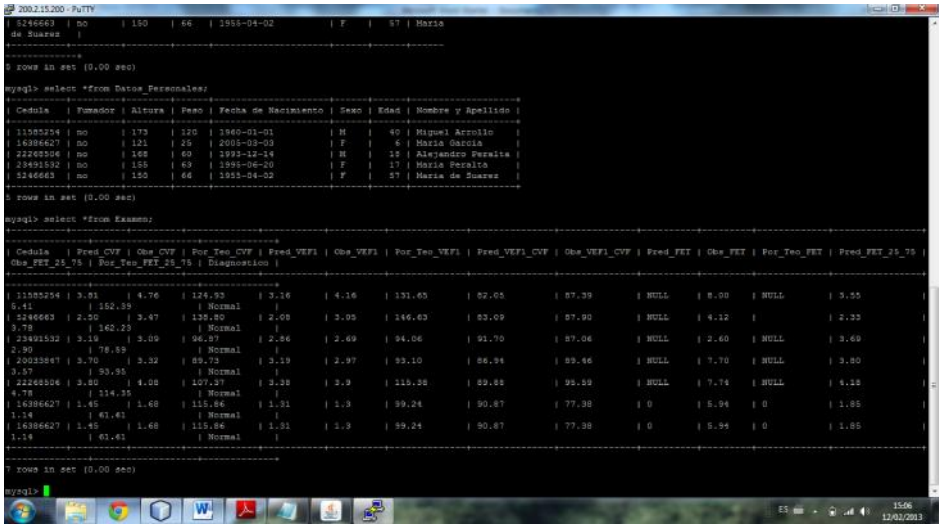


Figura 62.Tablas de la base de datos

Fuente: Elaboración Propia.

V.5 Aplicación en Android

Para el envío del examen de espirometría y de los datos personales del paciente y la recepción del diagnóstico preliminar se realizó una aplicación en Android como se muestra a continuación.

En la Figura 63 se muestra el icono de la aplicación en un terminal móvil Samsung SIII mientras que en la Figura 64 se puede apreciar la página de inicio de la aplicación en la cual hay que introducir los datos personales del paciente como nombre y apellido, cédula, edad, altura, peso, fecha de nacimiento, sexo y fumador.

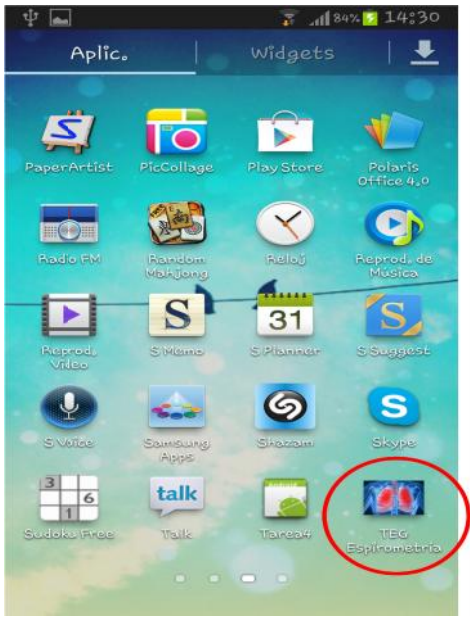


Figura 63. Icono de la aplicación

Fuente: Elaboración Propia

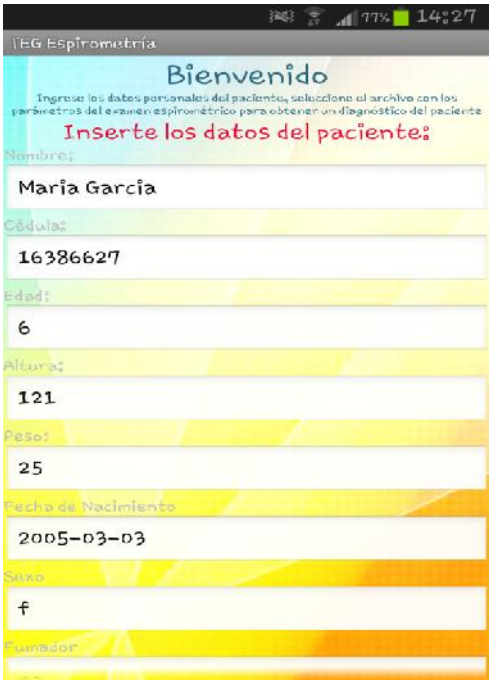


Figura 64. Página de inicio de la aplicación en Android

Fuente: Elaboración Propia

En la Figura 65 se muestra la dirección IP del servidor al cual se envió los datos funcionando el terminal móvil como cliente y dos botones uno para enviar los datos personales del paciente y del examen de espirometría y el otro por el cual el terminal móvil funciona como servidor y la computadora como cliente y solicita su diagnóstico.

Figura 65. Página de Inicio de la aplicación de android.

Fuente: Elaboración Propia.

En la Figura 66 se observa que el Diagnóstico preliminar fue recibido en el terminal móvil con éxito desde la computadora.

Figura 66. Diagnóstico del paciente

Fuente: Elaboración Propia

CAPÍTULO VI

Conclusiones y Recomendaciones

VI.1 Conclusiones

En el desarrollo del Trabajo Especial de Grado se diseñó una plataforma para la transmisión de exámenes de espirometría y la recepción del diagnóstico provenientes de un terminal móvil, lo que ayuda a solucionar la carencia de médicos especialistas en neumología en las zonas rurales del país, con esta plataforma los médicos podrán obtener una orientación del resultado, producto de una base de datos con parámetros teóricos para su diagnóstico.

Gracias al apoyo del Dr. Andrés Octavio y al ITMS que nos suministraron las muestras para la creación de la base de datos se desarrolló una aplicación en Java la cual daba un diagnóstico preliminar del paciente a quien se le realizó el examen de espirometría.

Esta plataforma incluye una página web en la cual los médicos podrán tener acceso y encontrarán los casos de pacientes a los cuales se le realizó el examen de espirometría, clasificados por las distintas disfunciones respiratorias existente, permitiendo así un fácil acceso a través de una conexión a Internet siempre y cuando se tengan los permisos establecidos por parte de la DTI de la UCAB para conectarse al servidor.

El funcionamiento de esta plataforma contempla varios pasos para obtener un diagnóstico partiendo del terminal móvil, llegar a una computadora y de ahí guardar los datos en la base de datos. Esto se debió a que el terminal móvil no puede conectarse directamente al servidor de la UCAB debido a los permisos y el proxi que se utiliza.

La red VPN suministrada por la UCAB se utilizó para el intercambio de información de forma segura ya que envía la información cifrada a través de tunneling, lo cual proporciona una ventaja debido a que las personas autorizadas pueden acceder a la información del paciente.

El uso de las redes inalámbricas WiFi y su conexión a Internet permite conectar el terminal móvil con una aplicación en Java y enviar el examen de espirometría y recibir el diagnóstico preliminar del mismo.

El ancho de banda óptimo para la utilización de esta plataforma se considera de 2Mbps; no se recomienda trabajar con velocidades menores ya que la conexión con la VPN puede fallar y no permitirá el acceso a la base de datos y a la página web.

El formato utilizado para guardar las imágenes de los pacientes obtenidos por el ITMS almacenados en el servidor de la UCAB son formatos PNG. La curva generada por los exámenes de espirometría realizados en futuros pacientes por medio de la aplicación que se desarrolló no contará con el respaldo en la base de datos de dicha imagen, porque para cargarla al servidor de la UCAB se necesitan programas externos a la plataforma desarrollada en este Trabajo Especial de Grado.

El Trabajo Especial de Grado de (Genovés, D. y Rizzo, D, 2010), contempló la evaluación del comportamiento de la señal proveniente del espirómetro, para su posterior procesamiento y almacenamiento, por lo cual no fue necesario su análisis a profundidad en este proyecto.

VI.2 Recomendaciones

Las recomendaciones que se presentan en este Trabajo Especial de Grado son las siguientes:

Para obtener una eficacia a la hora del manejo de la plataforma se recomienda capacitar al personal en el uso de todas las aplicaciones, tanto al momento de aplicar el examen de espirometría como al enviar los datos para obtener un diagnóstico.

Para que la plataforma tenga un buen funcionamiento es necesaria una conexión a Internet estable de buena calidad en todos los equipos utilizados como el terminal móvil y el ordenador.

Se recomienda tener un respaldo actualizado de la información almacenada en el servidor por medidas de seguridad por si se presenta alguna falla y se pierde información.

BIBLIOGRAFÍA

Alvarez, A. (2008). *Semiología Médica.* Argentina: Panamericana.

American Telemedicine Association. (2012). Recuperado el 2012 de junio de 2012, de http://www.americantelemed.org/learn/what-is-telemedicine

Belvehí, X. M. (2004). *Derecho de las Telecomunicaciones.* Barcelona: Legal Link.

Cano, J. P. (15 de marzo de 2011). *Nuevas Tecnologías*. Recuperado el 20 de enero de 2013, de Android: el sistema operativo para smartphones: http://suite101.net/article/android-nuevo-sistema-operativo-para-smartphones-a44342#axzz2KSWYT4dV

Course Tecnology. (1999). *Principios de Sistemas de Información* . México: Thomson.

Falgueras, B. C. (2003). *Ingeniería del Software.* Barcelona: UOC.

Fernández, S. (2006). *Método de tratamiento de la escoiosis, cifosis y lordosis.* Barcelona.

Flores, J. J. (2008). *Método de las 6'D modelamiento- algoritmo - programación* .

Flores, J. J. *Metodo para la solucion de problemas utilizando la progracion orientada a objetos.* Lima: Universidad de San Martin de Porres.

Genovés D. y Rizzo D.(2011) "Desarrollo de una aplicación móvil para la transmisión y recepción de información espirométrica". Trabajo Especial de Grado, Universidad Católica Andrés Bello.

Girones, J. T. (2011). *El gran libro de Android.* España: marcombo.

Gómez, Á. C. (2005). *PHP y MySQL. Tecnologías para el desarrollo de aplicaciones web.* Españ: Díaz de Santos.

Ideas Propias. (2005). *Diseño de Páginas Web y Diseño Gráfico.* España: Vigo.

Kendall, K. (2005). *Analisis y Diseño de Sistemas.* Mexico : Pearson.

Marcano, A., & Ramirez, C. (24 de Septiembre de 2007). *Diseño de un modelo de interconexión de ambulatorios remotos de una red de Telemedicina.* Recuperado el 28 de Mayo de 2012, de http://biblioteca2.ucab.edu.ve: http://biblioteca2.ucab.edu.ve/anexos/biblioteca/marc/texto/AAR2046.pdf

MEDIAactive. (2010). *Dreamweaver CS5.* Barcelona: marcombo.

Mora, S. L. (2002). *Programacion de aplicaciones web: historia, principios básicos y clientes web.* . España: Club Universitario.

OCEANO. *Diccionario de Medicina. Oceano Mosby.* españa: Oceano.

Osorio, R. D. (2011). *Características y parámetros de la seguridad para los smartphones con sistema operativo Android.*

Pasquier, R. M. (9 de enero de 2013). *Medicina Preventiva Santa Fe.* Recuperado el 15 de enero de 2013, de La Espirometría: www.medicinapreventiva.com.ve/espirometria.htm

Piquero, J. V. (2009). *Prácticas de Redes.*

Ramos, M. d. (2011). *Seguridad Informatica.* España: Paraninfo.

Reyes, J. C. (2012). *Bene Hospital.* Recuperado el 8 de enero de 2013, de Neumólogo y las enfermedades respiratorias: http://www.bene.com.mx/notas_detalle.php?id_nota=7

Romero, M. d. (2010). *Servicios en Red.* España: Paraninfo.

Yera, Á. C. (2007). *Diseño y Programación de Bases de Datos.* Madrid: Visión Libros.

APENDICE A
Código de Java para la conexión con la base de datos.

ConexionBD.java

```
package edu.ucab.espirometria;

import java.sql.*;
import java.io.*;
/**
 *
 * @author Maria Fabiola Santiago, Yenmari Diaz
 */
public class ConexionBD {
//esta es la variable para realizar la conexion.

   Connection conexion = null;
   Connection conexion2 = null;
   String cedula;

   public ConexionBD() {
      try {
         //Importar DriverMySql para realizar la conexion >.< ----- link
http://www.youtube.com/watch?v=aVhrUpgmYqc
         Class.forName("org.gjt.mm.mysql.Driver");
         String basededatos = "db_tesis12";
         String usuario = "usr_tesis12";
         String password = "#tele389#";
         //Comando que me permite comunicarme por medio del driver sin try catch hay error de reporte
         conexion = DriverManager.getConnection("jdbc:mysql://200.2.15.200:3306/" + basededatos,
usuario, password);
         conexion2= DriverManager.getConnection("jdbc:mysql://200.2.15.200:3306/" + basededatos,
usuario, password);
         System.out.println("Conectado");

      } catch (ClassNotFoundException ex) {
         System.out.println("Error en el Driver");

      } catch (SQLException ex) {

         System.out.println("Error en la Conexion");
      }
   }

   //Inserta el registro del usuario en la tabla paciente.

   public void insertarTabla(String cedula, String Pred_CVF, String Obs_CVF, String Por_Teo_CVF,
String Pred_VEF1,
```

```
                    String Obs_VEF1, String Por_Teo_VEF1, String Pred_VEF1_CVF,String
Obs_VEF1_CVF,
                    String Pred_FET,String Obs_FET,String Por_Teo_FET,String Pred_FET_25_75,String
Obs_FET_25_75,
                    String Por_Teo_FET_25_75,String Diagnostico ) {
        {
        System.out.println("==================================================");
           try (Statement consultaInsertar = conexion.createStatement()) {
              consultaInsertar.executeUpdate("INSERT into Examen VALUES('" + cedula + "','" +
Pred_CVF + "','" + Obs_CVF + "','"
                     + Por_Teo_CVF + "','" + Pred_VEF1 + "','" + Obs_VEF1 + "','" + Por_Teo_VEF1 +
"','" + Pred_VEF1_CVF + "','"
                     + Obs_VEF1_CVF + "','" + Pred_FET + "','" + Obs_FET + "','" + Por_Teo_FET + "','"
+ Pred_FET_25_75 + "','"
                     + Obs_FET_25_75 + "','" + Por_Teo_FET_25_75 + "','" + Diagnostico + "')");
              System.out.println("Se inserto los datos Examen.");
           }
       catch (SQLException ex) {
          System.out.println("Error en la insersion Examen.");
       }
   }
   }

 public void insertarTabla2(String cedula, String fumador, String altura, String peso, String
fechanacimiento,
                    String sexo, String edad, String nombreapellido) {
    System.out.println("entro a insrtartabla2.");
      try {

        // Crea una instancia de consulta especial para insertar.
        Statement consultaInsertar = conexion2.createStatement();
        consultaInsertar.executeUpdate("INSERT into Datos_Personales VALUES('" + cedula + "','" +
fumador + "','" + altura + "','" + peso
              + "','" + fechanacimiento + "','" + sexo + "','" + edad + "','" + nombreapellido + "')");
//          verifica que la inserción fue exitosa
            System.out.println("==================================================");
           System.out.println("Se inserto paciente.");
       } catch (SQLException ex) {
          System.out.println("Error en la insersion paciente. ");
       }
   }


}
```

Ejecutable.java

```
package edu.ucab.espirometria;
import java.io.BufferedReader;
import java.io.DataInputStream;
import java.io.File;
import java.io.FileInputStream;
import java.io.FileNotFoundException;
import java.io.IOException;
import java.io.InputStreamReader;
import java.text.ParseException;
import java.util.ArrayList;
import java.util.LinkedList;
import java.util.List;
import javax.swing.JFileChooser;
import javax.swing.JOptionPane;

/**
 *
 * @author Maria Fabiola Santiago, Yenmari Diaz
 */
public class Ejecutable {
   static String cedula;
   static String Pred_CVF;
   static String Obs_CVF;
   static String Por_Teo_CVF;
   static String Pred_VEF1;
   static String Obs_VEF1;
   static String Por_Teo_VEF1;
   static String Pred_VEF1_CVF;
   static String Obs_VEF1_CVF;
   static String Pred_FET;
   static String Obs_FET;
   static String Por_Teo_FET;
   static String Pred_FET_25_75;
   static String Obs_FET_25_75;
   static String Por_Teo_FET_25_75;
   static String Diagnostico;
   static Double Obs_VEF1_CVF_decimal;
   static Double Por_Teo_VEF1_decimal;
   static Double Por_Teo_CVF_decimal;
   static String resultado_diagnostico;
   static  int aux = 0;
   static String Cedula;
```

```
    static String NombreApellido;
    static String Edad;
    static String Peso;
    static String FechaNacimiento;
    static String Sexo;
    static String Fumador;
    static String Altura;

    public String resultadodiagnostico()   {

        try {
          File archivo = new File("C:/Users/usuario/Documents/Tesis/muestras/datoscel.txt");
          FileInputStream archivostream = new FileInputStream(archivo);
          DataInputStream entrada = new DataInputStream(archivostream);
          BufferedReader lector = new BufferedReader(new InputStreamReader(entrada));
          String strLinea;

          try {
//              Leer el archivo linea por linea
            while ((strLinea = lector.readLine()) != null) {
              String[] palabraseparada = strLinea.split(";");

              cedula = palabraseparada[0];
              Pred_CVF = palabraseparada[1];
              Obs_CVF = palabraseparada[2];
              Por_Teo_CVF = palabraseparada[3];
              Pred_VEF1 = palabraseparada[4];
              Obs_VEF1 = palabraseparada[5];
              Por_Teo_VEF1 = palabraseparada[6];
              Pred_VEF1_CVF = palabraseparada[7];
              Obs_VEF1_CVF=palabraseparada[8];
              Pred_FET = palabraseparada[9];
              Obs_FET = palabraseparada[10];
              Por_Teo_FET = palabraseparada[11];
              Pred_FET_25_75 = palabraseparada[12];
              Obs_FET_25_75 = palabraseparada[13];
              Por_Teo_FET_25_75 = palabraseparada[14];
              Diagnostico=diagnosticoresult(Por_Teo_CVF, Por_Teo_VEF1, Obs_VEF1_CVF);
        }
          } catch (IOException ex) {
          }
        } catch (FileNotFoundException ex) {
        }
        return Diagnostico;
      }
    public String  diagnosticoresult(String Por_Teo_CVF, String Por_Teo_VEF1, String
Obs_VEF1_CVF) {
```

```
    Por_Teo_CVF_decimal =  Double.parseDouble(Por_Teo_CVF);
    Por_Teo_VEF1_decimal = Double.parseDouble(Por_Teo_VEF1);
    Obs_VEF1_CVF_decimal = Double.parseDouble(Obs_VEF1_CVF);

    if (Obs_VEF1_CVF_decimal <= 70) {

      resultado_diagnostico = "Obstructivo.";
      if (Por_Teo_CVF_decimal >= 60 && Por_Teo_VEF1_decimal >= 60) {
        resultado_diagnostico = "Obstructivo Leve.";
        aux = 77;
      }
    }
    if ((Obs_VEF1_CVF_decimal > 70) && (Por_Teo_CVF_decimal < 80) &&
(Por_Teo_VEF1_decimal < 80)) {
      if ((Por_Teo_CVF_decimal >= 60) && (Por_Teo_VEF1_decimal >= 60)) {
        resultado_diagnostico = "Restrictivo Leve.";
        aux = 77;
      }
      if ((Por_Teo_CVF_decimal < 60) && (Por_Teo_CVF_decimal >= 40) &&
(Por_Teo_VEF1_decimal >= 60) && (Por_Teo_VEF1_decimal >= 40)) {
        resultado_diagnostico = "Restrictivo Moderado.";
        aux = 77;
      }
      if ((Por_Teo_CVF_decimal < 40) && (Por_Teo_VEF1_decimal < 40)) {
        resultado_diagnostico = "Restrictivo Severo.";
        aux = 77;
      }
    }
    if ((Obs_VEF1_CVF_decimal < 70) && (Por_Teo_CVF_decimal < 80) &&
(Por_Teo_VEF1_decimal < 80)) {

      resultado_diagnostico = "Mixto";
      aux = 77;
    }
    if ((Obs_VEF1_CVF_decimal > 70) && (Por_Teo_CVF_decimal > 80) &&
(Por_Teo_VEF1_decimal > 80)) {

      resultado_diagnostico = "Normal";
      aux = 77;
    }
    if (aux != 77) {
      resultado_diagnostico = "El examen no cumple criterios de aceptabilidad y reproducibilidad se
sugiere repetir.";
    }
    return resultado_diagnostico;
  }
```

```
    public void cargaexamenpaciente(){
        ConexionBD conectar = new ConexionBD();
        try {
          File archivo = new File("C:/Users/usuario/Documents/Tesis/muestras/datoscel.txt");
          FileInputStream archivostream = new FileInputStream(archivo);
          DataInputStream entrada = new DataInputStream(archivostream);
          BufferedReader lector = new BufferedReader(new InputStreamReader(entrada));
          String strLinea;
          try {
//              Leer el archivo linea por linea
            while ((strLinea = lector.readLine()) != null) {
              String[] palabraseparada = strLinea.split(";");

              cedula = palabraseparada[0];
              Pred_CVF = palabraseparada[1];
              Obs_CVF = palabraseparada[2];
              Por_Teo_CVF = palabraseparada[3];
              Pred_VEF1 = palabraseparada[4];
              Obs_VEF1 = palabraseparada[5];
              Por_Teo_VEF1 = palabraseparada[6];
              Pred_VEF1_CVF = palabraseparada[7];
              Obs_VEF1_CVF=palabraseparada[8];
              Pred_FET = palabraseparada[9];
              Obs_FET = palabraseparada[10];
              Por_Teo_FET = palabraseparada[11];
              Pred_FET_25_75 = palabraseparada[12];
              Obs_FET_25_75 = palabraseparada[13];
              Por_Teo_FET_25_75 = palabraseparada[14];
              Diagnostico=diagnosticoresult(Por_Teo_CVF, Por_Teo_VEF1, Obs_VEF1_CVF);
              Cedula = palabraseparada[15];
              NombreApellido = palabraseparada[16];
              Edad = palabraseparada[17];
              Altura = palabraseparada[18];
              Peso = palabraseparada[19];
              FechaNacimiento = palabraseparada[20];
              Sexo = palabraseparada[21];
              Fumador = palabraseparada[22];

          conectar.insertarTabla(cedula, Pred_CVF, Obs_CVF,
Por_Teo_CVF,Pred_VEF1,Obs_VEF1,Por_Teo_VEF1,
          Pred_VEF1_CVF,Obs_VEF1_CVF,Pred_FET,Obs_FET,Por_Teo_FET,Pred_FET_25_75,
          Obs_FET_25_75,Por_Teo_FET_25_75,Diagnostico);
          conectar.insertarTabla2(Cedula, Fumador, Altura, Peso, FechaNacimiento, Sexo, Edad,
NombreApellido);
      }
```

```
            } catch (IOException ex) {
            }
        } catch (FileNotFoundException ex) {
        }
    }

    public void cargapaciente(){
        ConexionBD conectar = new ConexionBD();
          try {
            File archivopaciente = new File("C:/Users/usuario/Documents/Tesis/muestras/datoscel.txt");
            FileInputStream archivostream = new FileInputStream(archivopaciente);
            DataInputStream entrada = new DataInputStream(archivostream);
            BufferedReader lector = new BufferedReader(new InputStreamReader(entrada));
            String strLinea;

            try {
//              Leer el archivo linea por linea
                while ((strLinea = lector.readLine()) != null) {
                    String[] palabraseparada = strLinea.split(";");

                Cedula = palabraseparada[15];
                Fumador = palabraseparada[16];
                Altura = palabraseparada[17];
                Peso = palabraseparada[18];
                FechaNacimiento = palabraseparada[19];
                Sexo = palabraseparada[20];
                Edad = palabraseparada[21];
                NombreApellido = palabraseparada[22];

            conectar.insertarTabla2(Cedula, Fumador, Altura, Peso, FechaNacimiento, Sexo, Edad,
NombreApellido);
                }
            } catch (IOException ex) {
            }
        } catch (FileNotFoundException ex) {
        }
}
}
```

TEGEspirometria.java

```
package edu.ucab.espirometria;
import java.io.*;
import java.net.*;
import java.util.Calendar;
import java.awt.Image;
```

```
import java.awt.Toolkit;

/**
 *
 * @author Maria Fabiola Santiago, Yenmari Diaz
 */
public class TEGEspirometria extends javax.swing.JFrame {

String HOST;
final int PUERTO=46000;
ServerSocket sc;
Socket so;
DataOutputStream mensaje;
String mensajeRecibido;
Ejecutable ejecutable= new Ejecutable();
String diagnostico;
String horactual;

    /**
     * Creates new form TEGEspirometria
     */
    public TEGEspirometria() {
        initComponents();
        getIPLocalhost();

        //Deshabilita los botones y direccion IP.
        enviarCelButton.setEnabled(false);
        cargarBDButton.setEnabled(false);
        dirIPTextField.setEnabled(false);
    }

    @SuppressWarnings("unchecked")
    // <editor-fold defaultstate="collapsed" desc="Generated Code">
    private void initComponents() {

        cargarCelButton = new javax.swing.JButton();
        enviarCelButton = new javax.swing.JButton();
        cargarBDButton = new javax.swing.JButton();
        jScrollPane1 = new javax.swing.JScrollPane();
        informacionTextArea = new javax.swing.JTextArea();
        bienvenidosLabel = new javax.swing.JLabel();
        salirButton = new javax.swing.JButton();
        ipLabel = new javax.swing.JLabel();
        dirIPTextField = new javax.swing.JTextField();
        fondoLabel = new javax.swing.JLabel();

        setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);
```

```
setIconImage(getIconImage());
setPreferredSize(new java.awt.Dimension(427, 365));
getContentPane().setLayout(null);

cargarCelButton.setFont(new java.awt.Font("Tahoma", 0, 12)); // NOI18N
cargarCelButton.setText("Recibir Datos");
cargarCelButton.addActionListener(new java.awt.event.ActionListener() {
   public void actionPerformed(java.awt.event.ActionEvent evt) {
      cargarCelButtonActionPerformed(evt);
   }
});
getContentPane().add(cargarCelButton);
cargarCelButton.setBounds(23, 139, 103, 23);

enviarCelButton.setFont(new java.awt.Font("Tahoma", 0, 12)); // NOI18N
enviarCelButton.setText("Enviar Diágnostico");
enviarCelButton.addActionListener(new java.awt.event.ActionListener() {
   public void actionPerformed(java.awt.event.ActionEvent evt) {
      enviarCelButtonActionPerformed(evt);
   }
});
getContentPane().add(enviarCelButton);
enviarCelButton.setBounds(144, 139, 131, 23);

cargarBDButton.setFont(new java.awt.Font("Tahoma", 0, 12)); // NOI18N
cargarBDButton.setText("Cargar  Datos");
cargarBDButton.addActionListener(new java.awt.event.ActionListener() {
   public void actionPerformed(java.awt.event.ActionEvent evt) {
      cargarBDButtonActionPerformed(evt);
   }
});
getContentPane().add(cargarBDButton);
cargarBDButton.setBounds(293, 139, 105, 23);

informacionTextArea.setEditable(false);
informacionTextArea.setColumns(20);
informacionTextArea.setRows(5);
jScrollPane1.setViewportView(informacionTextArea);

getContentPane().add(jScrollPane1);
jScrollPane1.setBounds(7, 204, 390, 90);

bienvenidosLabel.setFont(new java.awt.Font("Tahoma", 1, 24)); // NOI18N
bienvenidosLabel.setText("Bienvenidos");
getContentPane().add(bienvenidosLabel);
bienvenidosLabel.setBounds(140, 10, 144, 23);
```

```
        salirButton.setFont(new java.awt.Font("Tahoma", 0, 14)); // NOI18N
        salirButton.setText("Salir");
        salirButton.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                salirButtonActionPerformed(evt);
            }
        });
        getContentPane().add(salirButton);
        salirButton.setBounds(170, 296, 57, 25);

        ipLabel.setFont(new java.awt.Font("Tahoma", 0, 12)); // NOI18N
        ipLabel.setText("Dirección IP Modo Servidor :");
        getContentPane().add(ipLabel);
        ipLabel.setBounds(30, 80, 160, 40);

        dirIPTextField.setText("192.168.1.100");
        getContentPane().add(dirIPTextField);
        dirIPTextField.setBounds(200, 90, 100, 30);

        fondoLabel.setIcon(new
javax.swing.ImageIcon(getClass().getResource("/imagenes/fondo2.png"))); // NOI18N
        getContentPane().add(fondoLabel);
        fondoLabel.setBounds(0, 0, 410, 340);

        pack();
    }// </editor-fold>

 //Da la IP de la máquina local.
public void getIPLocalhost(){

    InetAddress localHost = null;
        try {
            localHost = InetAddress.getLocalHost();
            dirIPTextField.setText(localHost.getHostAddress());

        } catch (UnknownHostException ex) {

        }
}



    private void cargarCelButtonActionPerformed(java.awt.event.ActionEvent evt) {
      initServer();
      enviarCelButton.setEnabled(true);
      dirIPTextField.setEnabled(true);
      cargarCelButton.setEnabled(false);
```

```
    ipLabel.setText("Dirección IP Modo Cliente:");


  }

  private void salirButtonActionPerformed(java.awt.event.ActionEvent evt) {
    System.exit(0);
  }

  private void enviarCelButtonActionPerformed(java.awt.event.ActionEvent evt) {

        horactual=getTiempo();
        diagnostico=ejecutable.resultadodiagnostico();
        informacionTextArea.append("\n"+"["+horactual+"]"+"Diágnostico"+":"+diagnostico);
        System.out.println(diagnostico);
        initClient(diagnostico);
        cargarBDButton.setEnabled(true);
        dirIPTextField.setEnabled(false);
        enviarCelButton.setEnabled(false);
        ipLabel.setText("Dirección IP Modo Servidor:");
        getIPLocalhost();

  }

  private void cargarBDButtonActionPerformed(java.awt.event.ActionEvent evt) {
      ejecutable.cargaexamenpaciente();
      informacionTextArea.append("\n"+"["+horactual+"]"+":"+"Sus datos han sido ingresado
sistema.");
      cargarBDButton.setEnabled(false);
      cargarCelButton.setEnabled(true);


  }

  public void initServer() {
BufferedReader entrada;


try {
sc = new ServerSocket(PUERTO );/* crea socket servidor que escuchara en puerto 5000*/
so=new Socket();
System.out.println("Esperando una conexión:");
so = sc.accept();
//Inicia el socket, ahora esta esperando una conexión por parte del cliente
System.out.println("Un cliente se ha conectado.");
//Canal de entrada  de datos
entrada = new BufferedReader(new InputStreamReader(so.getInputStream()));
```

```
System.out.println("Confirmando conexion al cliente....");
mensajeRecibido = entrada.readLine();
guardardatoscel(mensajeRecibido);
System.out.println(mensajeRecibido);
horactual=getTiempo();
informacionTextArea.append("\n"+"["+horactual+"]"+":"+"Sus datos han sido recibido exitósamente
");
System.out.println("Cerrando conexión...");
sc.close();//Aqui se cierra la conexión con el cliente
}catch(Exception e ) {
   horactual=getTiempo();
informacionTextArea.append("\n"+"["+horactual+"]"+"Error: "+e.getMessage());
System.out.println("["+horactual+"]"+"Error: "+e.getMessage());
}
}

   /*ejecuta este metodo para correr el cliente */
public void initClient(String resultadoexamen) {
try
{
HOST=dirIPTextField.getText().toString();
so = new Socket(HOST,PUERTO); /*conectar a un servidor en localhost con puerto 5000*/
//creamos el flujo de datos por el que se enviara un mensaje
mensaje = new DataOutputStream(so.getOutputStream());
//enviamos el mensaje
mensaje.write(resultadoexamen.getBytes());
//cerramos la conexión
sc.close();
informacionTextArea.append("\n"+"["+horactual+"]"+":"+"Su diágnostico fue enviado.");
}catch(Exception e )
{
horactual=getTiempo();
informacionTextArea.append("\n"+"["+horactual+"]"+"Error: "+e.getMessage());
System.out.println("\n"+"["+horactual+"]"+"Error: "+e.getMessage());
}
}

 public String getTiempo() {
      Calendar Cal = Calendar.getInstance();
      String tiempo = Cal.get(Cal.HOUR_OF_DAY) + ":" + Cal.get(Cal.MINUTE);
      String hora = tiempo;
      return hora;
   }

 public void guardardatoscel(String datosrecibidoscel){
      {
      FileWriter fichero = null;
```

```
        PrintWriter pw = null;
        try
        {
            fichero = new FileWriter("C:/Users/usuario/Documents/Tesis/muestras/datoscel.txt");
            pw = new PrintWriter(fichero);

                pw.println(datosrecibidoscel);

        } catch (Exception e) {
            e.printStackTrace();
        } finally {
            try {
            // Nuevamente aprovechamos el finally para
            // asegurarnos que se cierra el fichero.
            if (null != fichero)
                fichero.close();
            } catch (Exception e2) {
                e2.printStackTrace();
            }
        }
    }

  }
    /**
     * @param args the command line arguments
     */
    public static void main(String args[]) {
        /* Set the Nimbus look and feel */
        //<editor-fold defaultstate="collapsed" desc=" Look and feel setting code (optional) ">
        /* If Nimbus (introduced in Java SE 6) is not available, stay with the default look and feel.
         * For details see http://download.oracle.com/javase/tutorial/uiswing/lookandfeel/plaf.html
         */
        try {
            for (javax.swing.UIManager.LookAndFeelInfo info :
javax.swing.UIManager.getInstalledLookAndFeels()) {
                if ("Nimbus".equals(info.getName())) {
                    javax.swing.UIManager.setLookAndFeel(info.getClassName());
                    break;
                }
            }
        } catch (ClassNotFoundException ex) {

java.util.logging.Logger.getLogger(TEGEspirometria.class.getName()).log(java.util.logging.Level.SE
VERE, null, ex);
        } catch (InstantiationException ex) {
```

```
java.util.logging.Logger.getLogger(TEGEspirometria.class.getName()).log(java.util.logging.Level.SE
VERE, null, ex);
        } catch (IllegalAccessException ex) {

java.util.logging.Logger.getLogger(TEGEspirometria.class.getName()).log(java.util.logging.Level.SE
VERE, null, ex);
        } catch (javax.swing.UnsupportedLookAndFeelException ex) {

java.util.logging.Logger.getLogger(TEGEspirometria.class.getName()).log(java.util.logging.Level.SE
VERE, null, ex);
        }
        //</editor-fold>

        /* Create and display the form */
        java.awt.EventQueue.invokeLater(new Runnable() {
            public void run() {
//              new TEGEspirometria().setVisible(true);
                TEGEspirometria frame = new TEGEspirometria();
                frame.setLocationRelativeTo(null);
                frame.setTitle("Trabajo de Grado Espirometría");
                frame.setIconImage(null);
                frame.setVisible(true);
            }
        });
    }
    // Variables declaration - do not modify
    private javax.swing.JLabel bienvenidosLabel;
    private javax.swing.JButton cargarBDButton;
    private javax.swing.JButton cargarCelButton;
    private javax.swing.JTextField dirIPTextField;
    private javax.swing.JButton enviarCelButton;
    private javax.swing.JLabel fondoLabel;
    private javax.swing.JTextArea informacionTextArea;
    private javax.swing.JLabel ipLabel;
    private javax.swing.JScrollPane jScrollPane1;
    private javax.swing.JButton salirButton;
    // End of variables declaration
```

APENDICE B
Código de Android para el envió del examen de espirometría y la recepción de su diagnostico.

Código main.xml android

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
   xmlns:tools="http://schemas.android.com/tools"
   android:layout_width="fill_parent"
   android:layout_height="fill_parent"
   android:background="@drawable/fondo2"
   android:orientation="vertical" >

   <TextView
      android:layout_width="fill_parent"
      android:layout_height="wrap_content"
      android:text="Bienvenido"
      android:gravity="center_horizontal"
      android:textColor="#276880"
      android:textSize="25dp" />

   <TextView
      android:layout_width="fill_parent"
      android:layout_height="wrap_content"
      android:gravity="center_horizontal"
      android:text="Ingrese los datos personales del paciente, seleccione el archivo con los parámetros del examen
espirométrico para obtener un diagnóstico del paciente"
      android:textColor="#276880"
      android:textSize="10dp" />

   <TextView
      android:layout_width="fill_parent"
      android:layout_height="wrap_content"
      android:gravity="center_horizontal"
      android:text="Inserte los datos del paciente:"
      android:textColor="#ED0034"
      android:textSize="20dp" />
   <!-- inicia el ScrollView -->

   <ScrollView
      android:layout_width="fill_parent"
      android:layout_height="fill_parent" >

      <!-- se colocal el LinearLayout que contendra el formulario -->

      <LinearLayout
         android:layout_width="fill_parent"
         android:layout_height="fill_parent"
         android:orientation="vertical" >

         <TextView
```

```
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:text="Nombre:" />

<EditText
    android:id="@+id/etnombre"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:hint="nombre y apellido" />

<TextView
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:text="Cédula:" />

<EditText
    android:id="@+id/etcedula"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:hint="número de cedula" />

<TextView
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:text="Edad:" />

<EditText
    android:id="@+id/etedad"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:hint="edad" />

<TextView
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:text="Altura:" />

<EditText
    android:id="@+id/etaltura"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:hint="altura en cm" />

<TextView
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:text="Peso:" />

<EditText
    android:id="@+id/etpeso"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
```

```
    android:hint="peso en Kg" />

<TextView
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:text="Fecha de Nacimiento" />

<EditText
    android:id="@+id/etfnacimiento"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:hint="aaaa-mm-dd" />

<TextView
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:text="Sexo" />

<EditText
    android:id="@+id/etsexo"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:hint="F o M" />

<TextView
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:text="Fumador" />

<EditText
    android:id="@+id/etfumador"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:hint="si o no" />

<TextView
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:gravity="center_horizontal"
    android:text="....................." />

<TextView
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:text="    " />

<TextView
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:text="Dirección IP del Servidor al cual se enviaran los datos"
    android:gravity="center_horizontal"/>
```

```
<EditText
   android:id="@+id/etip"
   android:layout_width="fill_parent"
   android:layout_height="wrap_content"
   android:hint="192.168.0.110" />

<Button
   android:id="@+id/btenviar"
   android:layout_width="fill_parent"
   android:layout_height="wrap_content"
   android:layout_weight="80"
   android:text="Enviar Datos" />

<TextView
   android:layout_width="fill_parent"
   android:layout_height="wrap_content"
   android:gravity="center_horizontal"
   android:text="....................." />

<TextView
   android:layout_width="fill_parent"
   android:layout_height="wrap_content"
   android:text="    " />

<TextView
   android:layout_width="fill_parent"
   android:layout_height="wrap_content"
   android:text="Dirección IP del Móvil para recibir diagnóstico" />

<TextView
   android:id="@+id/tvmovil"
   android:layout_width="fill_parent"
   android:layout_height="wrap_content"
   android:textSize="20dp"
   android:gravity="center_horizontal"
   android:textColor="#000000"
   android:text="192.168.1.102" />

<Button
   android:id="@+id/btsolicitar"
   android:layout_width="fill_parent"
   android:layout_height="wrap_content"
   android:layout_weight="80"
   android:text="Solicitar Diagnóstico" />

<TextView
   android:layout_width="fill_parent"
   android:layout_height="wrap_content"
   android:gravity="center_horizontal"
   android:text="El Diagnóstico es:"
   android:textColor="#ED0034"
   android:textSize="20dp" />
```

```
        <TextView
            android:id="@+id/tvdiagnostico"
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:gravity="center_horizontal"
            android:text="Normal"
            android:textColor="#276880"
                    android:textSize="25dp" />

        <TextView
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:gravity="center_horizontal"
            android:text="....................." />

        <TextView
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:text="    " />
         <Button
            android:id="@+id/btsalir"
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:layout_weight="80"
            android:text="Salir de la aplicación" />
      </LinearLayout>
    </ScrollView>

</LinearLayout>
```

Codigo MainActivity.java android

```
package com.ucab.clientefinal;


import java.io.BufferedReader;
import java.io.DataInputStream;
import java.io.File;
import java.io.FileInputStream;
import java.io.IOException;
import java.io.InputStream;
import java.io.InputStreamReader;
import java.io.OutputStream;
import java.net.ServerSocket;
import java.net.Socket;
import java.net.UnknownHostException;
import android.app.Activity;
import android.app.AlertDialog;
import android.content.DialogInterface;
import android.net.wifi.WifiInfo;
```

```
import android.net.wifi.WifiManager;
import android.os.Bundle;
import android.os.Environment;
import android.util.Log;
import android.view.View;
import android.view.View.OnClickListener;
import android.widget.Button;
import android.widget.EditText;
import android.widget.ImageView;
import android.widget.TextView;
import android.widget.Toast;


public class MainActivity extends Activity implements OnClickListener {

        Button enviar,selecttxt,solicitar;
        EditText  etip,nombre,cedula,edad,altura,peso,fenacimiento,sexo,fumador;
        Socket sock, skCliente;
        ServerSocket skServidor;
        String texto;
        TextView diagnostico, ipmovil;

        /** Called when the activity is first created. */

        public void onCreate(Bundle savedInstanceState) {
                super.onCreate(savedInstanceState);
                setContentView(R.layout.main);

                enviar = (Button) findViewById(R.id.btenviar);
                enviar.setOnClickListener(this);

                solicitar = (Button) findViewById(R.id.btsolicitar);
                solicitar.setOnClickListener(this);

                selecttxt = (Button) findViewById(R.id.btsalir);
                selecttxt.setOnClickListener(this);

                etip = (EditText) findViewById(R.id.etip);
                nombre = (EditText) findViewById(R.id.etnombre);
                cedula = (EditText) findViewById(R.id.etcedula);
          edad= (EditText) findViewById(R.id.etedad);
          altura= (EditText) findViewById(R.id.etaltura);
          peso= (EditText) findViewById(R.id.etpeso);
          fenacimiento= (EditText) findViewById(R.id.etfnacimiento);
          sexo= (EditText) findViewById(R.id.etsexo);
          fumador= (EditText) findViewById(R.id.etfumador);

          diagnostico=(TextView)findViewById(R.id.tvdiagnostico);
    ipmovil = (TextView) findViewById(R.id.tvmovil);

    WifiManager wifiManager = (WifiManager) getSystemService(WIFI_SERVICE);
    WifiInfo wifiInfo = wifiManager.getConnectionInfo();
```

```
int ipAddress = wifiInfo.getIpAddress();
String IP = String.format("%d.%d.%d.%d",
                (ipAddress & 0xff),
                (ipAddress >> 8 & 0xff),
                (ipAddress >> 16 & 0xff),
                (ipAddress >> 24 & 0xff));
ipmovil.setText("Direccion IP: "+IP);


        }

        public void onClick(View v) {
                // TODO Auto-generated method stub

                switch(v.getId()){

                case R.id.btenviar:

                        Toast.makeText(this, "enviando datos", Toast.LENGTH_SHORT ).show();

                        try {

//                              lee el contenido del archivo txt con los parametros del examen de
espirometria

                                        try
                                        {
                                                File ruta_sd =
Environment.getExternalStorageDirectory();

                                                File f = new File(ruta_sd.getAbsolutePath(),
"prueba_sd.txt");

                                                BufferedReader fin =new BufferedReader(new
InputStreamReader(new FileInputStream(f)));

                                                texto = fin.readLine();
                                                fin.close();

                                        Toast.makeText(this, texto, Toast.LENGTH_SHORT).show();

                                        }
                                        catch (Exception ex)
                                        {
                                                Log.e("Ficheros", "Error al leer fichero desde tarjeta
SD");
                                                Toast.makeText(this, "Error al leer fichero desde tarjeta
SD", Toast.LENGTH_SHORT).show();
                                        }

//                                      envio información al servidor
                                sock = new Socket(etip.getText().toString(), 46000);
```

```
                    OutputStream os = sock.getOutputStream();

                    String enviardatos = texto+ ";"+ nombre.getText().toString() + ";"
                                + cedula.getText().toString() + ";"
                                + edad.getText().toString() + ";"
                                + altura.getText().toString() + ";"
                                + peso.getText().toString() + ";"
                                + fenacimiento.getText().toString() + ";"
                                + sexo.getText().toString() + ";"
                                + fumador.getText().toString();

                    os.write(enviardatos.getBytes());
                    os.flush();
                    Log.e("cliente", "flush finalizado");
                    os.close();

                } catch (UnknownHostException e) {
                    // TODO Auto-generated catch block
                    Toast.makeText(this, "Error de Host",Toast.LENGTH_SHORT ).show();
                } catch (IOException e) {
                    // TODO Auto-generated catch block
                    e.printStackTrace();
                    Toast.makeText(this, "Error en el socket",Toast.LENGTH_SHORT
).show();
                }
                break;

            case R.id.btsolicitar:

                Toast.makeText(this, "solicitando diagnóstico", Toast.LENGTH_SHORT ).show();

                try {
                    skServidor = new ServerSocket(46000);
                    skCliente = skServidor.accept(); // acepta al cliente
                    Toast.makeText(this, "espera conexion", Toast.LENGTH_SHORT
).show();

                    InputStream in = skCliente.getInputStream();
                    DataInputStream inputdata = new DataInputStream (in);

                    String strLinea=inputdata.readLine();
                    diagnostico.setText(strLinea);
                    in.close();


                } catch (IOException e) {
                    // TODO Auto-generated catch block
                    e.printStackTrace();
                }

                break;
```

```
                    case R.id.btsalir:
//                          salida del sistema

                            Toast.makeText(this, "salida del sistema", Toast.LENGTH_SHORT).show();

                    AlertDialog.Builder builder= new AlertDialog.Builder(this);
                    builder.setMessage("ESTA SEGURO QUE DESEA SALIR?")
                        .setCancelable(false)
                        .setPositiveButton("YES", new DialogInterface.OnClickListener() {

                                    public void onClick(DialogInterface dialog, int id) {
                                            // TODO Auto-generated method stub
                                            MainActivity.this.finish();
                                    }
                            })
                            .setNegativeButton("No", new DialogInterface.OnClickListener() {

                                    public void onClick(DialogInterface dialog, int id) {
                                            // TODO Auto-generated method stub
                                            dialog.cancel();
                                    }
                            });

                    AlertDialog alert = builder.create();
                    alert.show();

                            break;
                    }

            }

    }
```