UNIVERSIDAD DE TALCA
FACULTAD DE INGENIERÍA
ESCUELA DE INGENIERÍA CIVIL EN COMPUTACIÓN

Sistema de Monitoreo en el Hogar: Prototipo de monitoreo para el cuidado de personas que viven en hogares unipersonales

PATRICIO ALONSO VALDERRAMA ALIAGA

Profesor Guía: BENJAMÍN INGRAM

Profesor Informante: CÉSAR ALEJANDRO ASTUDILLO HERNÁNDEZ

Memoria para optar al título de
Ingeniero Civil en Computación

El presente documento fue calificado con nota: _____

Curicó – Chile

Noviembre, 2016

CONSTANCIA

La Dirección del Sistema de Bibliotecas a través de su encargado Biblioteca Campus Curicó certifica que el autor del siguiente trabajo de titulación ha firmado su autorización para la reproducción en forma total o parcial e ilimitada del mismo.

Curicó, 2019

Dedicado a mi familia y amigos.

AGRADECIMIENTOS

En primer lugar, agradecer a mi madre, padre, hermana y amigos que desde mi lugar de nacimiento, Chimbarongo, han confiado en mí, hasta el final de esta importante etapa.

Agradecer, además, a mis amigos, compañeros y profesores que hicieron agradable mi estadía en la universidad.

Finalmente, me gustaría agradecer a mi profesor guía Benjamin Ingram, que siempre estuvo disponible en este proyecto.

RESUMEN

En Chile, como en otros países en desarrollo o en vías de desarrollo, estamos viviendo un proceso de creciente envejecimiento poblacional, que implica nuevos desafíos en materia de política pública, especialmente en el area de salud, previsión social y uso del tiempo libre. A medida que el país avanza económicamente, las familias jóvenes han tenido más posibilidades de obtener una vivienda propia e independizarse de sus padres. Así, han aumentado las personas de la tercera edad que viven únicamente con sus parejas o solos.

Estos cambios han provocado que los adultos mayores cuenten con menos recursos humanos que les ayuden a enfrentar los principales cambios que les ocurren al envejecer, por ejemplo el deterioro de sus condiciones de salud y el aislamiento que sufren al dejar de trabajar o perder la pareja. Sin embargo cada vez son más las personas que prefieren llevar una vida independiente y tranquila al envejecer. En este sentido, y dadas las condiciones del proceso de envejecimiento, los familiares cercanos sufren la alarmante sensación de no conocer el estado en que se encuentran sus padres mientras no se encuentren cerca.

Actualmente, son muchos los estudios que se generan en relación al tema establecido. El Servicio Nacional del Adulto Mayor (SENAMA), realiza constantes indicadores sociodemográficos que afectan a los adultos mayores en el territorio nacional.

El objetivo de este proyecto de título, es implementar un prototipo de Sistema de Monitoreo que sirva como ayuda para esos familiares que se preocupan por las condiciones de estado en las cuales se encuentran los adultos mayores en sus hogares unipersonales.

El enfoque de esta memoria estará puesto en el procesamiento de información y condiciones actuales del hogar unipersonal de adultos mayores, generando datos y estadísticas que ayuden a monitorear y salvaguardar la vida de estos mismos.

SUMMARY

In Chile, like another countries, we are living a process of population aging, which implies new challenges in public policy terms, especially in health area, social welfare and use of free time. As the country progresses economically, young families have been possibilities to obtain an own home and become independent of their parents. So, the elderly people who live solely with their partners or alone have increased.

These changes have caused older adults have fewer human resources to help them cope with the major changes that occur as they get older, such as the deterioration of their health conditions and the isolation suffering when they stop work or lose their partner. However, more and more people prefer to have an independent and quiet life as they get older. In this sense, and with the conditions of the aging process, the close family suffer the alarming feeling of not knowing the parents state when they are not close.

Currently, there are many studies that are generated in relation to the established theme. The National Service for the Elderly (SENAMA, in spanish), performs constant sociodemographic indicators that affect older adults in the national territory.

The objective of this project is implement a prototype Monitoring System that serves as an aid to those relatives who are concerned about the state conditions in which the elderly are in their single-person homes.

The focus of this report will be on the processing of information and current conditions of the single-person household of older adults, generating data and statistics that help monitor and safeguard their lives.

PALABRAS CLAVE

Monitoreo; Adulto Mayor; Raspberry Pi; Servicio web; Base de Datos; Sensores; Bluetooth; BLE.

TABLA DE CONTENIDOS

página

Anexos

ÍNDICE DE FIGURAS

página

ÍNDICE DE TABLAS

página

1. Introducción

1.1. Descripción de la propuesta

1.1.1. Contexto del proyecto

Llega un momento en la vida en que las personas se hacen dependientes (por vejez, discapacidad o enfermedad), ya sean de sus hijos o algún familiar cercano. Sea cual sea el motivo, todos queremos tener cierta autonomía en nuestras vidas y en el hogar, pero existen riesgos de accidentes o problemas de salud que impiden esto.

Por lo anterior, mi propuesta de proyecto consiste en desarrollar un sistema de alerta y monitoreo[1] en un hogar, ideal para personas que requieren estar en constante cuidado, pero que por alguna razón se encuentran solos.

El sistema está conformado por sensores, tales como: movimiento, húmeda y temperatura; que estarán enviando estadísticas o indicadores de comportamiento del dependiente y señales que serán almacenadas en una base de datos a partir de un microcomputador, en este caso Raspberry Pi[2]. Este enviará los datos a una aplicación móvil en un dispositivo inteligente del usuario que quiere realizar el monitoreo.

1.1.2. Trabajos relacionados

Actualmente existen algunos sistemas que implementan este tipo de conceptos y realizan este propósito. Sin embargo las personas a monitorear tienen alguna discapacidad o en algunos casos no se dan cuenta de que puede ocurrir algún problema. Algunos sistemas requieren mucha interacción con la persona a monitorear, deben

[1] Proceso mediante el cual se reúne, observa, estudia y emplea información para luego poder realizar un seguimiento de un programa o hecho particular.

[2] https://www.raspberrypi.org/

presionar botones y tener un dispositivo móvil. Por ejemplo las personas de avanzada edad no se adaptan a las tecnologías y discrepan de aprender.

Otras tecnologías ayudan al monitoreo de enfermedades, por ejemplo la diabetes, sin embargo monitorean sus niveles de azúcar recibiendo constantes datos, pero si tienen algún problema específico en el hogar, no podrán llegar en auxilio.

Además los sistemas son costosos y algunos no se encuentran disponibles en Chile.

1.1.3. Definición del problema

Existen varios factores que generan la problemática que hace necesario el monitoreo de personas a larga distancia:

1. Las personas mayores sufren el no poder realizar una vida independiente, sin embargo en su condición susceptible a riesgos, es obvio que alguien se preocupe por ellos. En Chile el 11,8 % de los adultos mayores viven en un hogar unipersonal, según el Servicio Nacional del Adulto Mayor en el 2013. [6]

2. Las personas con algún trastorno psicológico, discapacidad u otro que se encuentre en riesgo de sufrir algún percance en el hogar.

3. El dinero o la disponibilidad de otra persona para realizar el cuidado a un familiar dependiente. Por ejemplo, si una persona de avanzada edad vive en la casa de su hijo y este tiene un trabajo regular, no se encontrará en la casa durante todo el día.

1.1.4. Propuesta de solución

La solución a la problemática consiste en desarrollar un sistema que le permita al usuario monitorear algunas señales básicas para el cuidado de un familiar dependiente que se encuentre en la casa.

Este sistema consta de cuatro partes:

- La primera parte es el hardware, referente a los sensores que envían información.

- La segunda parte es un microcomputador, que recibirá la información proveniente de los sensores y la enviará constantemente a un Servidor Web.

- La tercera parte, como se dijo anteriormente, tiene referencia al Servidor, el cuál almacenará la información en una base de datos, además de contener una serie de funciones para extraer estos mismos.
- La cuarta parte es una aplicación móvil que contiene gráficos e información del familiar dependiente.

En cuanto a los sensores, son fáciles de conseguir y tienen un precio moderado. También podemos usar una pulsera inteligente[3], que es básica y resistente al agua, tiene un precio moderado y sirve para algunos indicadores de importancia, ocupada por deportistas.

La conexión a internet se realizará utilizando dispositivos 3G[4], que tienen una amplia cobertura, además de un Bluetooth para la pulsera si es necesario.

La aplicación se encargará de informar cuando algunos niveles se encuentren en estado crítico o cuando una persona haya estado demasiado tiempo en un lugar gracias a los sensores de movimientos.

1.2. Hipótesis

- El uso de este sistema puede ayudar a un usuario preocupado por los riesgos de algún familiar dependiente a monitorear y asegurarse del bienestar de este.
- Las personas dependientes de una edad avanzada podrán vivir tranquilas en su hogar unipersonal.
- Los usuarios no deberán preocuparse de configuraciones.
- El prototipo de este sistema será realizado con tecnología existente.

1.3. Objetivos

Objetivo general

[3]Pulseras Bluetooth para dispositivos móviles que entregan información

[4]Dispositivo módem que utiliza un chip de teléfono, capaz de entregar conexión a internet a través de señal 3G, normalmente se conectan mediante USB

- El objetivo es diseñar y desarrollar un prototipo de bajo costo que será implementado para monitorear las señales de vitales y el ambiente de la casa unipersonal de una persona.

Objetivos específicos

- Diseñar y construir un prototipo de los dispositivos (sensores y otros).
- Diseñar y construir un prototipo del Servidor en un micro computador.
- Diseñar y construir un prototipo de un Servidor Web.
- Diseñar y construir un modelo de aplicación con estadísticas simples para el usuario (Software).

1.4. Alcances

- Los sensores necesarios serán básicos (Como ejemplo: temperatura, movimiento y pulsera inteligente) y se podrán modificar, mejorar o agregar más en un futuro con personas especializadas.
- El microcomputador recibirá la información de los sensores y las enviará a un Servidor Web.
- El Servidor Web será el encargado de obtener la información y conectarse a un Smartphone con Android a modelo de prueba.
- Se diseñará una aplicación móvil en Android que permita al o los usuarios visualizar estadísticas del familiar dependiente.
- Los usuarios no podrán hacer ninguna modificación.

1.5. Metodología

Objetivo 1: "Diseñar un prototipo de los dispositivos"

- Estudiar los diferentes dispositivos existentes.
- Seleccionar de acuerdo a su valor y utilidad.

- Formar el prototipo con los dispositivos.
- Probar el prototipo.

Objetivo 2: "Diseñar un prototipo del Servidor"

- Estudiar lenguaje de programación adecuado.
- Determinar variables que se utilizarán.
- Desarrollar modelo de Servidor.
- Conectar el Servidor con Prototipo de dispositivos.
- Probar funcionamiento.

Objetivo 3: "Diseñar un modelo de aplicación"

- Diseñar mockup de la aplicación.
- Desarrollo de la aplicación.
- Pruebas de la aplicación.

1.6. Plan de trabajo

Etapa 1: Desarrollar el objetivo 1 (6 de Mayo - 7 de Julio)

- Estudiar los diferentes dispositivos existentes (6 de Mayo - 19 de Mayo).
- Seleccionar de acuerdo a su valor y utilidad (20 de Mayo - 26 de Mayo).
- Formar el prototipo con los dispositivos (27 de Mayo - 23 de Junio).
- Probar el prototipo. (24 de Junio - 8 de Julio).

Etapa 2: Diseñar un prototipo del Servidor (8 de Julio - 8 de Septiembre)

- Estudiar lenguaje de programación Python (8 de Julio - 21 de Julio).
- Determinar variables que se utilizarán (22 de Julio - 28 de Julio).
- Desarrollar modelo de Servidor (29 de Julio - 11 de Agosto).
- Conectar el Servidor con Prototipo de dispositivos (12 de Agosto - 1 de Septiembre).
- Probar funcionamiento (2 de Septiembre - 8 de Septiembre).

Etapa 3: Diseñar un modelo de aplicación (9 de Septiembre - 10 de Noviembre)

- Diseñar mockup de la aplicación (9 de Septiembre - 15 de Septiembre).
- Desarrollo de la aplicación (16 de Septiembre - 20 de Octubre).
- Pruebas de la aplicación (21 de Octubre - 10 de Noviembre).

1.7. Resumen

En este capítulo se expuso la introducción del problema definido que se espera solucionar en este proyecto a través de hipótesis. Adicionalmente se muestran los objetivos general y específicos para el desarrollo de este sistema. Brevemente hablamos de los alcances, para finalizar con el plan de trabajo que utilizaremos para completar los objetivos anteriormente vistos.

En el siguiente capítulo veremos el Marco teórico del sistema, donde se profundizará acerca de los conceptos que tienen relación con el desarrollo del prototipo del sistema de monitoreo.

2. Marco teórico

En esta sección se profundizará en los conceptos básicos y principales que tienen relación al desarrollo del prototipo de un sistema de monitoreo a larga distancia. En primera instancia se definirán los componentes utilizados, usos y aplicaciones. Posteriormente se explicarán los tópicos más importantes con relación al monitoreo en el hogar.

2.1. Sistema de Monitoreo en el hogar

Un Sistema de Monitoreo, consiste en un sistema que siga, acompañe y oriente el comportamiento de una persona, dentro de límites de privacidad, con el fin de orientar metas y operaciones de manera correcta. En términos del proyecto computacional, el sistema o software debe tener las siguientes capacidades:

- Capturar datos.
- Menejo y almacenamiento de datos.
- Manipulación y análisis de datos.
- Envío y recepción de datos.
- Presentación de datos.

El sistema de monitoreo debe entregar información de datos relevantes, obtenidas a través de sensores, analizarla, enviarla y generar una presentación en forma de gráficos, reportes o notificaciones en un dispositivo móvil.

2.2. Componentes del Sistema de Monitoreo

El sistema de monitoreo tiene una serie de componentes importantes para su funcionamiento. A continuación se presentan:

1. **Sensores:** Son las fuentes de recepción de datos. Pueden variar dependiendo de la necesidad y su utilidad. Los sensores reciben información en un determinado tiempo para ser almacenada.

2. **Datos:** Son el componente esencial del sistema, se permite su recepción a través de los sensores en intervalos de tiempo, creando una secuencia de datos para ser analizada.

3. **Hardware:** Se habla de hardware por un servidor que almacene y transmita la información registrada desde los sensores a una aplicación móvil por señales 3G.

4. **Software:** El Software en el sistema consta de una serie de funciones para manipular y analizar los datos recibidos, de tal manera de enviarlos a la aplicación.

5. **Aplicación:** Aplicación de tipo móvil que recibe los datos y los presenta en forma de gráficos, reportes o notificaciones.

2.3. Programación del Monitoreo

La Programación para el monitoreo es esencial para el sistema, de manera tal que se realicen las notificaciones y estadísticas en la aplicación. Se realiza constantemente de acuerdo al almacenamiento de datos.

La información es esencial para el monitoreo a distancia de una persona. Se necesitan una serie de datos y estados de acuerdo al análisis para establecer índices de normalidad y riesgo.

En la programación se utilizarán varios sensores útiles para la medición de datos para monitorear a una persona dependiente: Sensores de Temperatura, Movimiento Infrarrojo, Pulseras Inteligente, etc. Dados los anteriores, se almacenarán los datos de estos sensores y se analizarán para tener índices y estados esenciales para la aplicación.

2.3.1. Estado de los datos

El estado de los datos es esencial en el monitoreo y en la presentación de estos en la aplicación. Los datos se analizan de acuerdo a los sensores a utilizar.

- Temperatura: Se mide en grados Celcius, la temperatura ideal puede variar de acuerdo a la estación del año, en verano el ideal es entre 24 °C y 26 °C, mientras que en invierno es entre 21 °C y 23 °C.
- Movimiento: Registra un valor 0 cuando no existe movimiento y valores variables de acuerdo a la actividad infrarroja. Se utilizará para saber si existe o no movimiento, registrando 0 u otro número como estados verdadero y falso.
- Ritmo Cardíaco: Se analiza de acuerdo a la medición de los latidos en un minuto.
- Pasómetro: Aumenta el contador de acuerdo a los pasos que se realizan. Podremos saber cuando una persona se encuentre caminando debido a la actividad de este contador.

Cada uno de estos datos presenta diferentes mediciones y estados de riesgo, necesarios para ser notificados, de acuerdo a sus niveles, tal como se aprecian en el *Cuadro 2.1*.

Cuadro 2.1: Tabla de estados y riesgo inicial para los datos.

Dato	**Métrica**	**ideal mín**	**ideal máx**	**crítico mín**	**crítico máx**
Temperatura	°C	18 °C	24 °C	15 °C	30 °C
Movimiento	boolean	-	-	-	-
Ritmo Cardíaco	latidos/min	60lpm	220lpm-edad	50lpm	200lpm
Pasómetro	pasos	-	-	-	-

2.3.2. Analizar los estados

Sabiendo los estados y sus límites, podemos ver las formas de representar nuestros datos en la aplicación.

Como ejemplo tomaremos el dato Temperatura. Sabemos que la temperatura mínima es 18 °C, pero la temperatura crítica mínima es 15 °C, por lo tanto tenemos intervalos que se representarán a través de colores en la presentación de la aplicación

móvil para entender mejor la información. Cuando la temperatura es menor o mayor a los índices críticos se realizará además una notificación en el dispositivo móvil, para que el usuario tenga una mayor advertencia de esto.

En cuanto a datos tales como el Movimiento y el Pasómetro, serán utilizados de manera distinta, en cuanto a estados de posicionamiento y actividad de la persona.

El ritmo cardíaco y la temperatura puede ser modificables en cuanto a sus estados críticos e ideales, debido a la edad de la persona, latidos por minuto, zona geográfica y la estación del año en la cual nos situemos, donde las temperaturas ideales y críticas pueden variar.

2.4. Problemas identificados

Existen problemas a la hora de realizar un monitoreo, ya sea en el análisis de los datos o en las molestias realizadas a la persona dependiente a monitorear.

- El análisis de información en estado erróneo afectaría de manera negativa al sistema, causando problemas de notificaciones a los usuarios. Por lo anterior es importante conocer los sensores y las mediciones que realizan.
- Los sensores utilizados no deben invadir la privacidad de la persona a monitorear, deben ser útiles, estadísticos y simples a la vez. Se podrían instalar cámaras de vigilancia en todas las habitaciones y el sistema ya no tendría ninguna utilidad, sin embargo esto invade la privacidad de las personas.
- Se debe estimar exactamente los índices máximos, mínimos y críticos, pudiendo ser modificados de acuerdo a un estudio de la persona, debido a factores climáticos, geográficos, enfermedades, edad y otros, son diferentes para el monitoreo.

2.5. Información necesaria

Dados los problemas anteriores, existe información necesaria para realizar un monitoreo adecuado de una persona, teniendo en cuenta los datos que se analizarán.

- **Clima y geografía:** En Chile existen climas de diferentes tipos, las personas al vivir un largo período de tiempo con estos climas, se adaptan con mayor

facilidad. Es por esto que los factores de temperatura ideales y críticos, varían dependiendo de la zona climática en la que se encuentren. Por lo tanto se analizarán los siguientes climas que modificarán los estados de temperatura de acuerdo a esto: Desértico, Semiárido, Mediterráneo, Andino, Templado y Templado oceánico.

- **Edad:** La edad es un factor importante en el análisis de los datos, como vimos en el *Cuadro 2.1*, el ritmo cardíaco ocupa una fórmula dependiendo de la edad de las personas

2.5.1. Trabajos Relacionados

A continuación se mencionan empresas y servicios que se utilizan actualmente para el monitoreo de personas.

2.5.2. Lively

Lively es un sistema de alerta de emergencias médicas 24/7 el cual fue desarrollado por Lively Inc. Cuenta con aparatos sensores que se instalan en diferentes lugares de la casa y también con un reloj inteligente, vinculándolos entre si y el centro de llamados de Lively[15].

Ventajas:

- Su reloj inteligente controla y alerta los sensores, además de poseer un botón de ayuda.
- Posee un servicio de çall center"de Lively que monitorea a larga distancia.
- Dispositivo inteligente para píldoras.
- Control de puertas.

Desventajas:

- El çall center"se encarga de todos los monitoreos.
- Necesita mucha interacción con la persona a monitorear.
- Sólo está disponible en 6 países.

2.5.3. Digital Life

Es un sistema de seguridad en el hogar 24/7 que realiza un monitoreo en el hogar y su seguridad a partir de diferentes dispositivos sensores, controlando la temperatura, luz y agua. Además de poder controlar aparatos inteligentes solo con el smartphone[14].

Ventajas:

- Tiene diferentes planes con diferentes dispositivos.
- Controla la seguridad y realiza alertas para cuando no se encuentre en el hogar.
- Posee una variedad de dispositivos

Desventajas:

- Posee planes con un precio elevado, que van desde $40 a $55 dolares mensuales.
- Los costos de equipamiento varían dependiendo del plan, desde $700 a $1.200 dolares.
- No se encuentra disponible en todos los países.

2.6. Hardware

Para la construcción del sistema de monitoreo son necesarios una serie de componentes físicos, tales como computadores y sensores, que formarán parte del correcto funcionamiento del sistema.

2.6.1. Raspberry Pi

La Raspberry Pi es un ordenador de bajo costo, del tamaño de una tarjeta de crédito que se conecta a un monitor de ordenador o televisor, y utiliza un teclado o un ratón estandar. Se trata de un pequeño dispositivo capaz de permitir a las personas de todas las edades explorar la computación, y aprender a programar en lenguajes como Scratch y Python. Es capaz de hacer todo lo que espera una computadora de escritorio, desde la navegación por internet y reproducción de videos de alta definición, a la toma de hojas de cálculo, procesador de textos, y jugar juegos.[10]

Figura 2.1: Raspberry Pi

2.6.2. Servidor de Base de datos

Un Servidor de Base de datos es un programa que provee servicios de base de datos a otros programas u computadoras, definido así en el modelo cliente-servidor. Algunas de las funciones básicas son almacenar, recuperar y administrar los datos de una base de datos. Se requiere un servidor confiable que pueda soportar el almacenamiento y peticiones de datos constantes.[19] MySQL es la base de datos de código abierto más popular del mundo. Con su rendimiento, confiabilidad y facilidad de uso comprobados, MySQL se ha convertido en la principal opción de base de datos para aplicaciones basadas en la Web, utilizada por propiedades web de alto perfil como Facebook, Twitter, Youtube y los cinco principales sitios web. Además, es una alternativa extremadamente popular como base de datos integrada, distribuida por miles de ISV y OEM.[16]

2.6.3. Sensor de Temperatura

Es un sensor básico de humedad y temperatura, mida temperaturas entre 0 y 50 °C con una incerteza de 2 °C y para medir humedad entre 20 % y 80 % con una precisión de 5 % con periodos de muestreo de 2 segundo. [20]

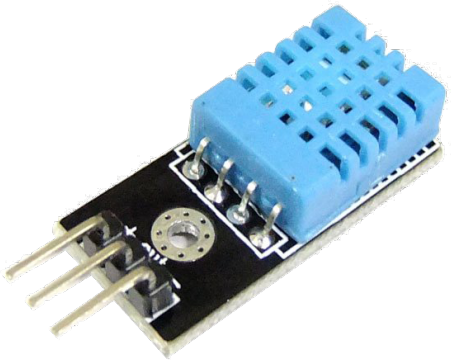


Figura 2.2: Sensor de Temperatura y Humedad DHT11.

2.6.4. Sensor de Movimiento

Es un sensor de movimiento pasivo infrarrojo (PIR). Recibe las radiaciones detectando la diferencia entre el calor emitido por un cuerpo y el del espacio alrededor. En el proyecto servirá para revisar si una persona entra o sale de una habitación, así podemos saber el tiempo y otras variables.[12]

Figura 2.3: Sensor de movimiento Pir Módulo Hc-sr501.

2.6.5. Pulsera Inteligente Xiaomi

Es una pulsera inteligente esencial para deportistas, pero que entrega diferentes estadísticas necesarias. Tiene monitoreo de ritmo cardíaco, pasómetro, rastreador de sueño y ánimo, etc. En el proyecto servirá para revisar estadísticas de la persona a

monitorear, ya no solo a la casa, sino que a esta misma. Además al saber si alguien está caminando, podrá poner en funcionamiento alguno de los otros dispositivo.[7]

Figura 2.4: Pulsera Inteligente Xiaomi Bracelet

2.6.6. Módem 3G

Un módem 3G es un terminal de red inalámbrica basada en la tecnología CDMA2000. Con este dispositivo se puede acceder a internet desde cualquier lugar y momento. Raspberry Pi tiene puertos USB en los que este dispositivo será conectado. En el proyecto servirá para comunicar la aplicación con el servidor donde se almacenarán los datos.

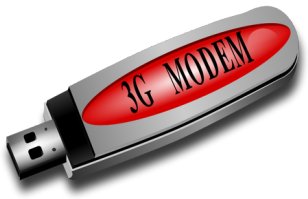


Figura 2.5: Módem 3G para la conexión a internet

2.6.7. Costo de Componentes

En el Cuadro 2.2 se detallan los costos de cada componente necesario para la construcción de este prototipo de sistema de monitoreo.

Cuadro 2.2: Tabla de Componentes para conocer el precio total del Sistema

Componente	**Precio (pesos)**
Raspberry Pi	$28.400
Sensor de Temperatura	$2.990
Sensor de Movimiento	$3.490
Pulsera Inteligente Xiaomi	$15.890
Adaptador Bluetooth 4.0	$9.990
Cargador	$1.000
Total	**$61.760**

2.7. Software

2.7.1. Python

Python[1] es un maravilloso y poderoso lenguaje de programación fácil de utilizar (leer y escribir). La sintaxis de Python es muy clara y sencilla, además de utilizar palabras clave en inglés estándar.[11]

Librería GPIO[2]: Es un módulo para el control de los canales de Raspberry Pi. Es una pequeña librería de Python que toma algo de la complejidad conducida de los pines GPIO (Entrada/Salida de Propósito General). Una vez que se instala un LED puede ser iluminado con 3 lineas en Python. La instalación de la biblioteca es igual de simple.

2.7.2. Raspbian

Rasbian[3] es un sistema operativo libre basado en Debian, optimizado para el hardware de Raspberry Pi. Un sistema operativo es la modificación de programas básicos y utilidades para hacer que funcione Raspberry Pi. Sin embargo Raspbian entrega mas que un Sistema operativo puro: Viene con cerca de 35.000 paquetes de software pre-compilados en un formato que hace más fácil la instalación en el Raspberry Pi.

[1]Lenguaje de Programación dinámico y tipeado diseñado para enfatizar la usabilidad
[2]https://sourceforge.net/p/raspberry-gpio-python/wiki/install/
[3]https://www.raspberrypi.org/downloads/raspbian/

2.8. Resumen

En este capítulo se definieron los temas más relevantes para el desarrollo del sistema. Se abordaron temas como la definición del monitoreo y sus componentes. Posteriormente se interiorizó en cómo se realizará la programación del monitoreo y cómo se estructurarán los datos para su análisis. También se explicaron algunos problemas identificados y cada uno de los componentes de Hardware y Software que se utilizarán.

En el siguiente capítulo se analizará la metodología de desarrollo que se utilizará a lo largo de este proyecto.

3. Metodología

En este capítulo se mostrarán algunos aspectos relevantes en cuanto al control, planificación y estructura del proceso de desarrollo del prototipo del sistema de monitoreo.

Debido a que el proyecto es relativamente largo, es necesario implementar una Metodología que se adapte correctamente a las necesidades de desarrollo.

Tal como vimos en el *Sección 1.5* del *Capítulo 1*, tenemos una Metodología que cumplir para los Objetivos del Sistema desarrollados en un Plan de trabajo visto en la *Sección 1.6*.

3.1. SCRUM

Scrum es una metodología ágil en el que se aplican de manera regular un conjunto de buenas prácticas para trabajar colaborativamente.[1] En Scrum se realizan entregas parciales y regulares del producto final, priorizadas por el beneficio que aportan al receptor del proyecto. Scrum está especialmente indicado para proyectos con las siguientes características:

- Entornos Complejos
- Obtener resultados pronto
- Requisitos cambiantes o poco definidos
- Innovación, competitividad, flexibilidad y productividad son fundamentales.

[1] https://proyectosagiles.org/que-es-scrum/

En el caso de este proyecto, se complementa perfectamente con las características de Scrum, debido a que es un proyecto de titulación innovativo y único, que posee requisitos poco definidos y cambiantes, dado que somos nosotros quienes lo estamos desarrollando. Una ventaja importante de Scrum en cuanto al proceso, es que se ejecuta en bloques temporales cortos y fáciles, ideales para este proyecto y el trabajo recurrente y semanal que estamos desarrollando. Estos bloques temporales se llaman *Sprint* y son desplegados en varias iteraciones para su finalización.

3.1.1. Product Owner

El *Product Owner* es la persona Dueña del Producto y responsable de gestionar la Lista del Producto, además de maximizar el valor del producto y del trabajo del Equipo de Desarrollo. En el caso de nuestro proyecto el Product Owner será nuestro profesor guía Benjamin Ingram, a quien debemos rendirle cuenta de los avances de nuestro proyecto.[24]

3.1.2. Eventos de Scrum

En Scrum existen eventos predefinidos con el fin de crear regularidad y minimizar la necesidad de reuniones no definidas en Scrum. Estos eventos constituyen una oportunidad formal para la inspección y adaptación.[24]

Sprint

Es el corazón de Scrum. Son bloques de tiempo que al ser terminados crean un incremento en el desarrollo del Producto final. Los Sprint contienen una serie de Reuniones de Planificación, además de eventos desarrollados en documentos para el desarrollo del Sprint.[24]

Sprint Review

Al final del Sprint se entrega este documento llamado Revisión del Sprint, donde se explica lo que se realizó durante el Sprint. Este documento es revisado a través de una reunión en la cual se revisan los productos terminados y cuales no se han terminado, los problemas ocurridos en el Sprint y sus soluciones.[24]

Sprint Retrospective

La importancia de la Retrospectiva del Sprint, es inspeccionarse a sí mismo y crear un plan de mejoras que sean abordados durante el siguiente Sprint.[24]

3.2. Scrum en el Proyecto de Sistema de Monitoreo

Debido a ser un proyecto unipersonal, la metodología Scrum no fue aprovechada al máximo, dado a que no existe un equipo de trabajo que la siga. Sin embargo se adoptó un plan de trabajo con Sprint de aproximadamente dos semanas, donde se realizaban las reuniones de planificación, además de documentar la finalización del Sprint a través de Sprint Review y Sprint Retrospective para mejorar en el siguiente Sprint. Esta metodología funcionó correctamente debido al Plan de trabajo separado en objetivos que hacía mas sencillo el desarrollo.

3.3. Resumen

En este capítulo se describió parte de la metodología utilizada para completar el problema del proyecto. En primera instancia explicamos de qué trata la metodología de desarrollo Scrum, para terminar explicando cómo fue utilizada en este proyecto. En el siguiente capítulo se describirá el diseño de la solución del proyecto.

4. Diseño de la Solución

En este capítulo se mostrarán algunos aspectos relevantes en cuanto al diseño del prototipo del sistema de monitoreo. En primer lugar se presentarán las herramientas y tecnologías que se utilizaron. Luego, se presentarán historias de usuario, para indagar en los requisitos que deben ser cumplidos. Para finalizar con la arquitectura de los hardware, software y el diseño de la aplicación.

4.1. Herramientas

En esta sección describiremos las herramientas necesarias para el desarrollo del sistema.

4.1.1. Herramientas de Desarrollo

- **Sublime Text**[1]: Es un editor de texto para código fuente.
- **Wamp Server**[2]: Es un ambiente de desarrollo web para Windows. Donde podemos crear aplicaciones web con Apache2, PHP y base de datos MySQL.
- **PhpMyAdmin**: Es un manejador de base de datos utilizado por Wamp Server.
- **Python**[3]: Es un lenguaje de programación interpretado, cuyo propósito hace hincapié en una sintaxis cómoda y un código legible. Utilizado principalmente para el desarrollo de aplicaciones en Raspberry Pi para este proyecto.
- **PhoneGap**[4]: Es una herramienta de Adobe, utilizada para desarrollar apli-

[1] http://www.sublimetext.com
[2] http://www.wampserver.com/
[3] https://www.python.org/
[4] http://phonegap.com/

caciones móviles multiplataforma con una sola base de código a través de la tecnología HTML5, CSS y Javascript.

4.1.2. Herramientas de Control

- **PHP**[5]: Es un lenguaje interpretado de alto nivel, utilizado para paginas HTML y que es ejecutado por el lado del servidor. Esta herramienta es utilizada para la comunicación entre la base de datos y el Cliente (Aplicación Móvil).

4.1.3. Herramientas de Visualización

- **HTML**[6]: Hyper Text Markup Language, es un estándar para la creación de páginas web. En este proyecto se utilizará para la creación de la interfaz de la aplicación móvil.

- **Javascript**[7]: Es un lenguaje de programación interpretado, orientado a objetos, basado en prototipos, no tipado y dinámico. Javascript es una gran herramienta en el lado del cliente, donde implementaremos funciones que modifican directamente la vista por lado del cliente en HTML.

- **Ajax**[8]: Ajax es una técnica de desarrollo web para crear aplicaciones interactivas. Ejecutadas por lado del cliente, se utilizará para comunicar y desplegar datos desde el provenientes del servidor, gracias a su comunicación con PHP.

- **CSS**[9]: Es un lenguaje de hojas de estilo creado para el control de aspectos de presentación, debido a índices HTML. En este proyecto se utilizó para la presentación de la interfaz de la aplicación.

- **Boostrap**[10]: Es un framework para el diseño de sitios y aplicaciones web. Posee elementos basados en HTML y CSS, además de extensiones JavaScript para manejar estos elementos. En este proyecto se utilizó para el completo desarrollo del diseño de la aplicación móvil.

[5]https://www.php.net/
[6]http://www.w3schools.com/html/
[7]http://www.javascript.com
[8]http://www.w3schools.com/ajax/
[9]http://www.w3schools.com/css/
[10]http://www.getbootstrap.com/

4.2. Historias de Usuario

En esta sección se describirán los requisitos del sistema a través de historias de usuario. Estos requisitos fueron obtenidos a partir de la idea en conjunto con el profesor guía, algunas referencias de usuarios interesados en la aplicación e información recopilada de profesionales en el tema.

- Los usuarios deberán visualizar en la aplicación el nivel de temperatura actual de la habitación de la persona a monitorear.
- Los usuarios deberán visualizar en la aplicación el nivel de humedad actual de la habitación de la persona a monitorear.
- Los usuarios deberán visualizar en la aplicación el número de pasos que ha realizado la persona a monitorear en el día.
- Los usuarios deberán visualizar el ritmo cardíaco actual de la persona a monitorear.
- Los usuarios deberán visualizar en la aplicación si existe movimiento actualmente en la habitación de la persona a monitorear.
- Los usuarios podrán modificar la información personal de la persona a monitorear (edad y sexo).
- Los usuarios podrán modificar las reglas de estado de la temperatura de la habitación de la persona a monitorear.
- Los usuarios podrán modificar las reglas de estado del ritmo cardíaco de la persona a monitorear.
- Los usuarios podrán visualizar en la aplicación los estados de la temperatura a través de colores característicos.
- Los usuarios podrán visualizar en la aplicación los estados del ritmo cardíaco a través de colores característicos.
- Los usuarios podrán visualizar en la aplicación los estados de la humedad de la habitación a través de colores característicos.

- Los usuarios podrán visualizar en la aplicación los estados de movimiento en la habitación a través de colores característicos.

4.3. Diseño de la Arquitectura

La Arquitectura del sistema para este proyecto se centra en la arquitectura Cliente-Servidor, en este caso se centran principalmente en la visualización de datos a través de la aplicación móvil.

En el proyecto se pueden apreciar la división de cuatro capas tal como se ve en la *Figura 4.1*.

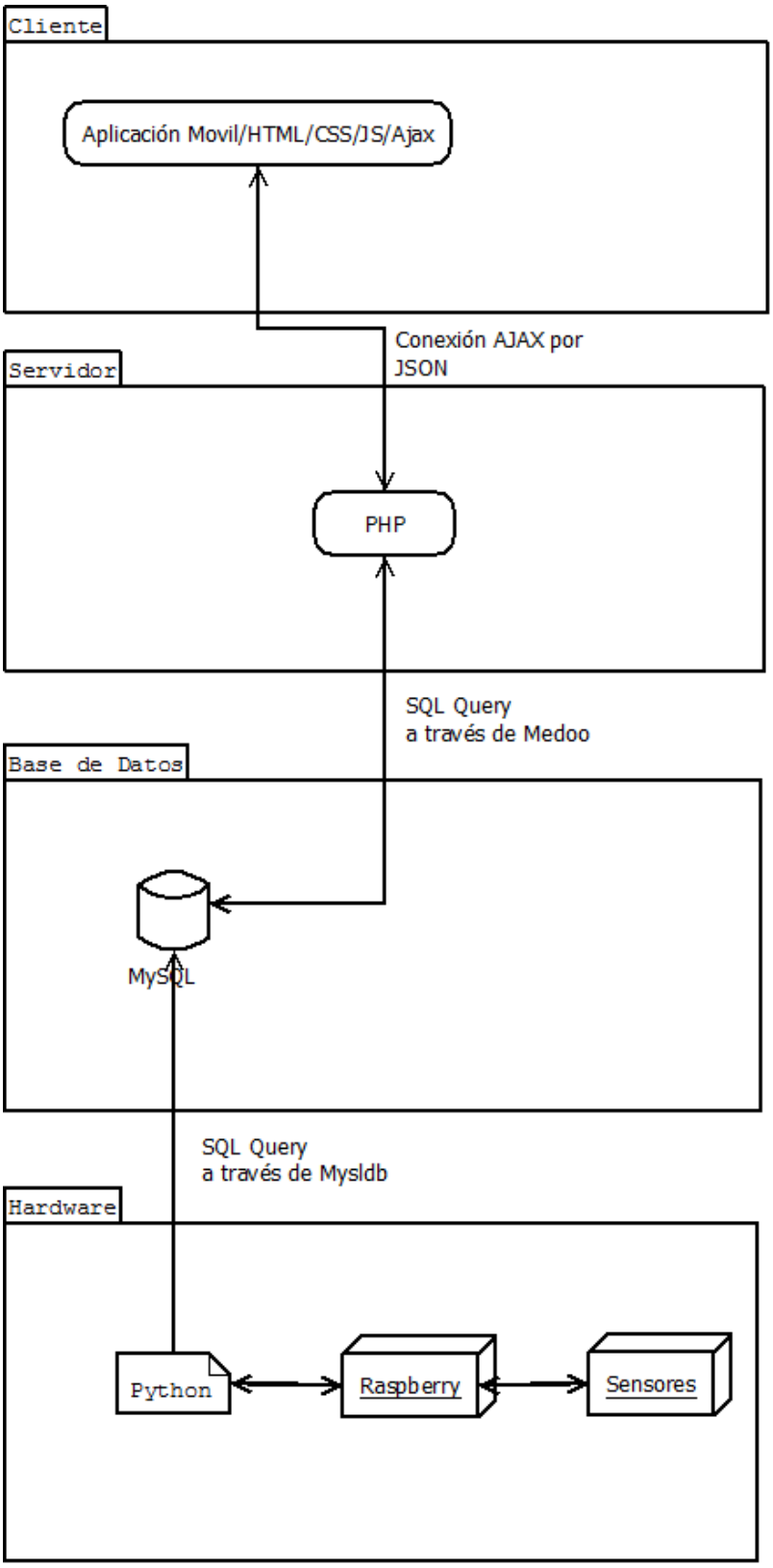


Figura 4.1: Arquitectura en Capas del prototipo de Sistema

La *Figura 4.1* muestra cuatro capas divididas que se explicarán a continuación:

- **Cliente**: La Capa del cliente tiene toda la aplicación móvil, donde se encuentran los archivos HTML, CSS y JS, esta se conecta a través de la herramienta Ajax de Javascript al Servidor Web para la obtención e inserción de datos en formato JSON.
- **Servidor**: El Servidor Apache es una capa que contiene una serie de archivos PHP, los cuales se encargan de dar resultado a la obtención e inserción de datos almacenados en la base de datos. El Servidor web es un intermediario entre el Cliente y la Base de Datos vía internet.
- **Base de Datos**: La capa Base de Datos nos permite almacenar a través de la plataforma MySQL los datos de manera espacial. Se comunica con el Servidor Web para entregar los datos al Cliente, pero también se comunica con la capa de Hardware para obtener estos mismos.
- **Hardware**: La capa de Hardware contiene el Raspberry Pi, en conjunto con todos los sensores que entregan información. Esta capa se comunica con la capa de Base de Datos a través de la librería MySqlDB de Python, para almacenar la información que es obtenida de los sensores.

4.3.1. Arquitectura de Hardware

En esta sección se presenta, como vimos anteriormente, la arquitectura de la capa de Hardware, donde se encuentran el microcomputador, en conjunto con los sensores conectados que entregan información. La Arquitectura de estos está compuesta por una serie de dispositivos conectados a los pines del microcomputador y los sensores a través de cables, en intervención de una Protoboard[11]. Además, se conecta una Pulsera a través de un dispositivo Bluetooth 4.0 conectado en el USB del Raspberry. El diseño de la arquitectura del Hardware puede verse en la *Figura 4.2*.

[11]Tablero con orificios conectados eléctricamente entre sí

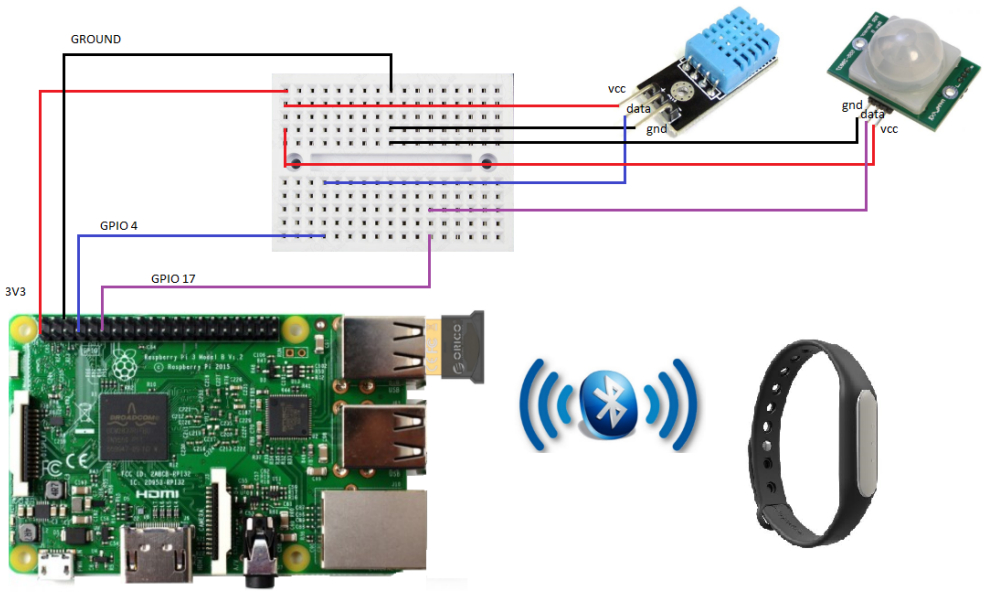


Figura 4.2: Arquitectura del Hardware del Sistema

4.4. Diseño de Interfaz de Usuario

En esta etapa se mostrarán los mockup para nuestro prototipo de aplicación móvil. La idea es centrarse en una visualización sencilla de los datos y que a su vez permita escalar en funcionalidades para el futuro.

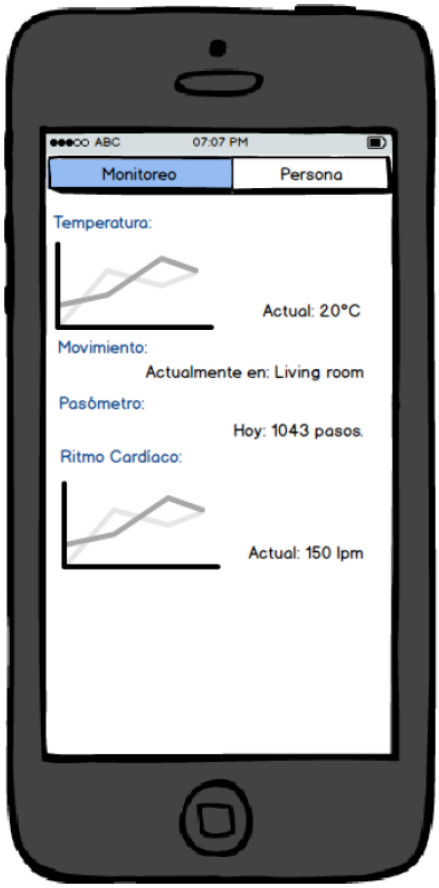


Figura 4.3: MockUp de la interfaz de Monitoreo de la Aplicación Móvil

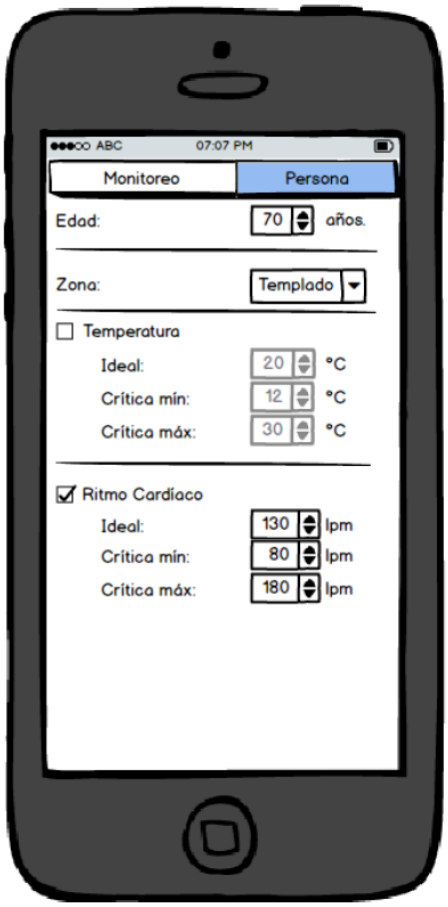


Figura 4.4: MockUp de la interfaz de Configuración de Persona de la Aplicación Móvil

Figura 4.5: MockUp de la interfaz de Configuración de la Aplicación Móvil

Figura 4.6: MockUp de la interfaz de Acerca De de la Aplicación Móvil

4.5. Resumen

En este capítulo pudimos apreciar los componentes más importantes del diseño de la solución de nuestro proyecto. Se definieron las herramientas que se van a utilizar que dieron paso al diseño de la arquitectura del sistema en capas, donde las capas se comunican a través del modelo de arquitectura Cliente-Servidor. Finalmente se mostraron los diseños iniciales para la interfaz de la aplicación móvil, en conjunto a sus mockup.

En el siguiente capítulo se describirá paso a paso el desarrollo de la solución que hemos diseñado para el sistema y su implementación.

5. Desarrollo de la Solución

En este capítulo se mostrarán los pasos necesarios para la construcción y funcionamiento del sistema de monitoreo. Para comenzar se analizarán los componentes de Hardware junto a todas sus partes y posteriormente el desarrollo de la programación correspondiente al software para este dispositivo. Adicionalmente se mostrará y explicará el desarrollo de la aplicación móvil.

5.1. Hardware

El hardware más importante utilizado en este proyecto es Raspberry Pi, el cual es un ordenador de bajo costo, capaz de utilizar un Sistema Operativo Linux.

Una de las más poderosas características de Raspberry Pi, es su columna de GPIO (General Purpose Input/Output). Estos pines son una interfaz física entre Raspberry Pi y el mundo exterior. En el nivel más simple, se puede pensar en ellos como interruptores que se pueden activar o desactivar (input) o que el Raspberry Pi puede activar o desactivar (output). De los 40 pines, 26 son pines GPIO y las otras son de alimentación o tierra.[21]

Los pines anteriormente nombrados nos servirán para realizar las conexiones de nuestros sensores y obtener los datos. Sin embargo en cada modelo de Raspberry Pi se cambia el número de pines, por lo que es necesario conocer estos mismos. En nuestro caso utilizaremos un Raspberry Pi 2 model B, el cual posee los 40 pines nombrados anteriormente. La funcionalidad de cada uno de estos pines puede ser observada con mayor detalle en la *Figura 5.1.*

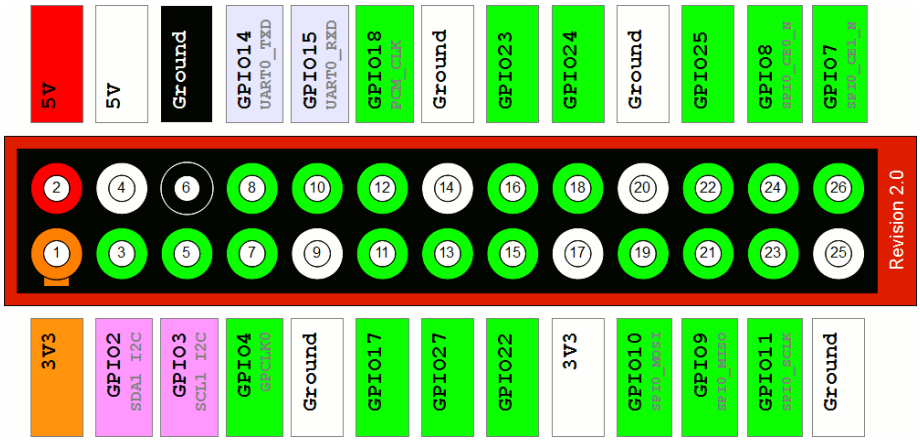


Figura 5.1: GPIO del Raspberry Pi 2B

5.1.1. Raspberry Pi

Para utilizar el Raspberry Pi, se necesita instalar un Sistema Operativo en la tarjeta SD que tiene disponible. En este caso el Sistema Operativo utilizado fue Raspbian[1], basado en Linux.

Rasbian es el Sistema Operativo oficial de la Fundación. Puede instalarlo con NOOBS o descargar la imagen. Raspbian viene preinstalado con muchos softwares para la educación, programación y uso general. Tiene Python, Scratch, Sonic Pi, Java, Mathematica y mucho más.

Instalación de Raspbian

Para instalar Raspbian debemos hacer que la tarjeta SD de nuestro Raspberry Pi contenga la imagen de Raspbian booteable, es decir que al iniciar el Raspberry proceda a iniciar el Sistema Operativo que se encuentra en la tarjeta SD.

Para esto, a modo de ejemplo, utilizaremos en Windows el programa Win32DiskImager, el cual es recomendado por Raspberry para este proceso. El programa es sencillo y bastante intuitivo de utilizar, pero debemos tener cuidado en seleccionar la imagen correcta que descargamos y la tarjeta SD montada.[2]

[1] https://www.raspberrypi.org/downloads/raspbian/
[2] https://www.raspberrypi.org/documentation/installation/installing-images/windows.md

5.1.2. Conexión de Sensores

Como se ha mencionado anteriormente los sensores que se utilizarán en este sistema son: Sensor DHT11 (temperatura y humedad), Sensor de Movimiento PIR y una Pulsera Inteligente Xiaomi 1S.

Para utilizar los sensores, es necesaria la conexión de estos mismos en los pines GPIO del Raspberry Pi tal como se muestra en la *Figura 5.2*.

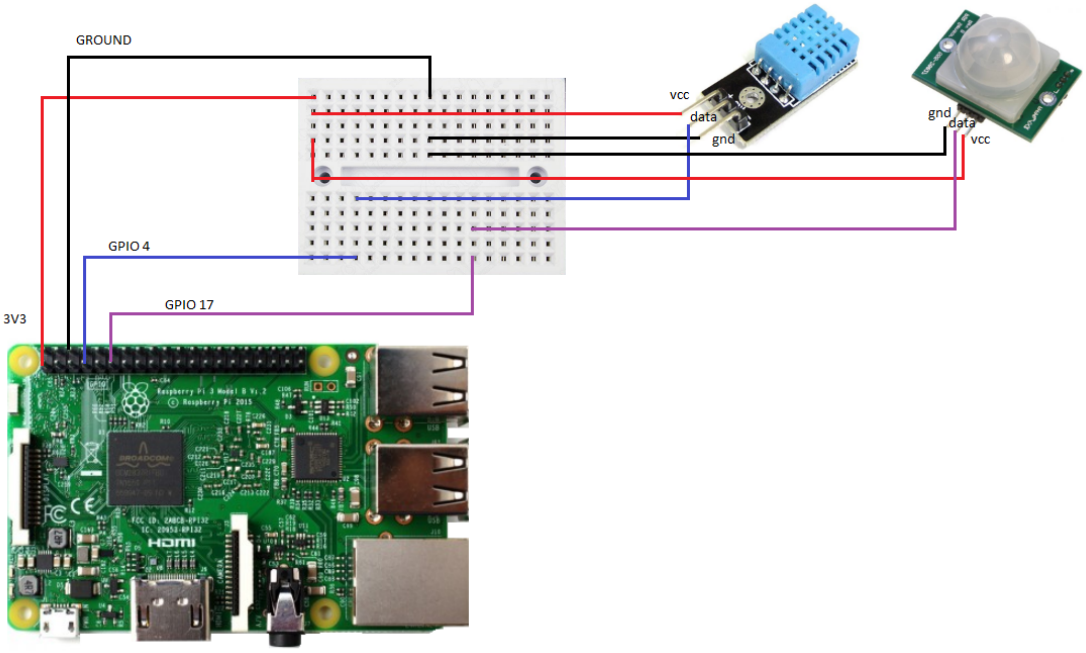


Figura 5.2: Arquitectura de Raspberry Pi con Sensores de DHT11 y PIR

Programación de Sensores

Para tener acceso a los sensores vamos a controlar nuestro Raspberry Pi a través de un programa realizado en Python. Para esto necesitamos instalar la librería **RPi.GPIO**[5] que instalaremos con el siguiente comando:

sudo apt-get install python-rpi.gpio python3-rpi.gpio

Adicionalmente utilizaremos una librería pura de Python para utilizar el sensor DHT11 de temperatura y humedad, esta contiene una forma sencilla de utilizar nuestro sensor en un programa Python[18]. Esta librería puede ser descargada desde la plataforma de desarrollo Github.

Ahora podemos crear un programa en Python para nuestros sensores que leerán la información de los sensores DHT11 de temperatura y humedad, y el sensor de movimiento conectados al Raspberry Pi. El programa utilizado se muestra a continuación
.

Listing 5.1: Código Sensores en Python

```
import RPi.GPIO as GPIO
import dht11
import time

GPIO.setwarnings(False)
GPIO.setmode(GPIO.BCM)
GPIO.cleanup()

GPIO.setmode(GPIO.BCM)
GPIO.setup(4, GPIO.IN)

while True:
instance = dht11.DHT11(pin = 17)
result = instance.read()

if result.is_valid():
print ("Last valid input: " + str(datetime.datetime.now()))
print ("Temperature: %d C" %result.temperature)
print ("Humidity: %d %%" %result.humidity)

if GPIO.input(4) == True:
print "Hay Movimiento"
else:
print "No hay Movimiento"

time.sleep(2)
```

Como vemos en el código anterior, estamos obteniendo desde el pin 4 la información del sensor de movimiento, y desde el pin 7, a través de la librería dht11, la información del sensor de temperatura y humedad. También podemos apreciar que estamos obteniendo la información cada 2 segundos. Los resultados de este programa pueden ser vistos en la siguiente *Figura 5.3*.

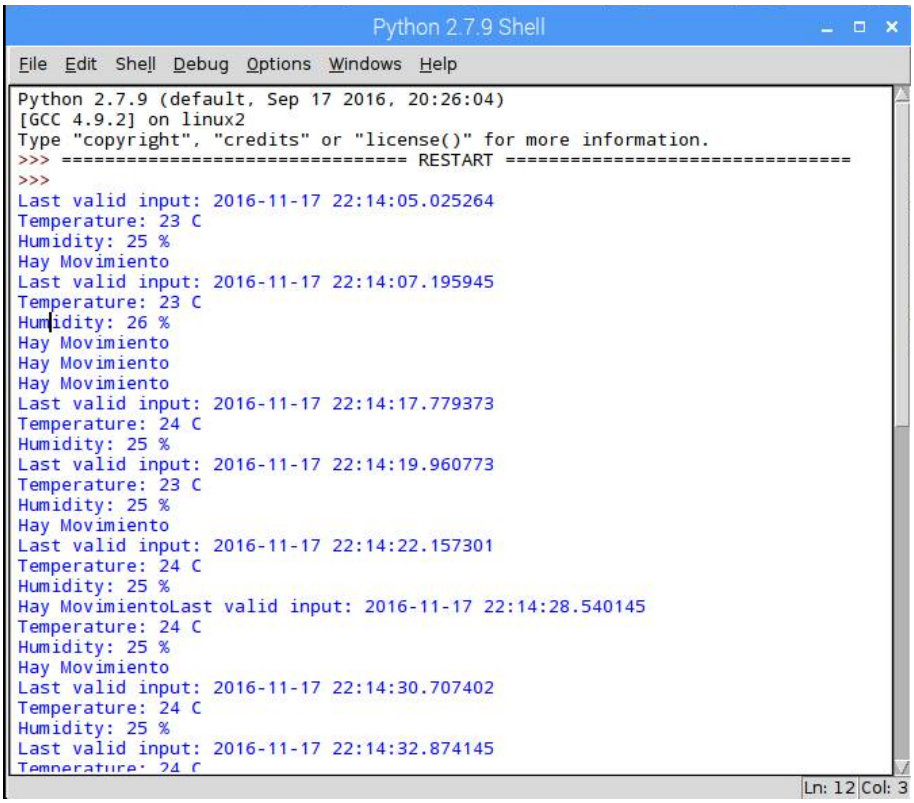


Figura 5.3: Resultados de Programa Python de sensores

5.1.3. Pulsera Inteligente

Uno de los procesos más importantes de este sistema es poder monitorear a una persona, sin embargo la información de una persona no puede ser sacada de un sensor conectado al microcomputador. El problema significaba algo crítico para el desarrollo del proyecto, sin embargo se analizó la Pulsera Inteligente Xiaomi 1S [7].

La pulsera Xiaomi 1S tenía la ventaja de analizar variables como el Pulso Cardíaco y los pasos del día que realizaba una persona. Lo más importante de la Xiaomi 1S es que es una pulsera resistente al agua, además de ser pequeña y de goma, por lo que tampoco es invasiva para la persona que monitorearemos. Otra ventaja muy importante tiene relación con la batería, la cuál dura aproximadamente 30 días.

La pulsera Xiaomi 1S puede ser utilizada con una aplicación móvil (Android/iOS), sin embargo agregar un smartphone a nuestra arquitectura no es una posibilidad, debido a que las personas mayores no se adecuan a estas tecnologías tan fácilmente.

La pulsera Xiaomi 1S cuenta con tecnología Bluetooth Low Energy (BLE). La tecnología de BLE[3] está diseñada para proveer un menor y significativo uso de la energía. Esta tecnología fue introducida al soporte de desarrollo a partir de Android 4.3 (API Level 18).[8]

Bluetooth LE se fundamenta en la reducción del consumo y, por tanto, en minimizar la potencia de transmisión de la señal radio utilizada y el radio de cobertura. La sustancial reducción del consumo y la consiguiente extensión de la autonomía, no solo redunda en una mejora para smartphones o tablets, sino que también es importante para el desarrollo de sensores.[23]

Servicios GATT

Bluetooth LE tiene una serie de características, los cuales son tipos de atributos definidos que contienen un único valor lógico, estas características se denominan Servicios GATT. Estos servicios son utilizados de acuerdo a las características que presenten los dispositivos con BLE.[4]

En nuestro caso, la pulsera Xiaomi posee bastantes características, sin embargo las que se utilizarán en este proyecto serán la medición de pasos diarios y el pulso cardíaco de una persona.

Cuadro 5.1: Tabla de Características de Servicios GATT

SpecificationName	**AssignedNumber**
Heart Rate	0x180D
Heart Rate Control Point	0x2A39
Heart Rate Max	0x2A8D
Heart Rate Measurement	0x2A37
Steps	0xff06

[3]

5.1.4. Acceso a Servicios GATT

Como vimos anteriormente en el *Cuadro 5.1*, podemos apreciar que existe un número asignado para cada una de las características de los Servicios GATT. Existen dos formas de utilizar estos servicios, sin embargo comparten atributos generales.

[3]Bluetooth Low Energy, tecnología diseñada por Bluetooth para proveer un menor y significativo uso de la energía.

UUID

Universal Unique Identifier o Identificador Único Universal es un número de 128 bits (16 bytes) que está garantizado (o tiene una alta probabilidad) para ser globalmente único. UUID se utilizan en muchos protocolos y aplicaciones que no sean Bluetooth, y su formato, el uso y la generación se especifican en UIT-T Rec. X.667, también conocida como ISO 9834-8. [22]

Los UUID están construidos de la siguiente forma:

xxxxxxx-0000-1000-8000-00805F9B34FB

Como vimos anteriormente podemos apreciar la forma constituida de nuestro UUID introduciendo un valor corto de 16 o 32 bits (indicado por xxxxxxxx), incluyendo ceros a la izquierda en el UUID Base Bluetooth.[13] Es decir el UUID que se utilizará para el GATT de Heart Rate Measurement será:

00002A37-0000-1000-8000-00805F9B34FB

Handle

Handle, por su parte, es un atributo identificador único de 16 bits para cada atributo en un servidor GATT particular. Es la parte de cada atributo que hace que sea direccionable, y se garantiza que no se cambie (con las salvedades descritas en el atributo de caché) entre transacciones o, para los dispositivos enlazados, incluso a través de conexiones.

Los Handle están construidos de la siguiente forma:

0x0000

Sin embargo la cantidad de Handle a disposición de todos los servidores GATT es *0xFFFE* (65535), aunque en la práctica, el número de atributos en un servidor es típicamente más cercano a unas pocas docenas.[13]

Permisos de acceso

Los valores de las características de un Servicio GATT, pueden ser escritas y/o leídas, al igual que en los permisos de archivos estos permisos determinan lo que el cliente puede realizar.[13]

Cuadro 5.2: Tabla de Permisos de Acceso para Servicios GATT

Permisos de Acceso	**Descripción**
None	El atributo no puede ser leído o escrito por un cliente
Readable	El atributo puede ser leído por el cliente
Writable	El atributo puede ser escrito por el cliente
Readable and writable	El atributo puede ser leído y escrito por el cliente

Encriptación

Los servicios pueden determinar si se requiere un cierto nivel de cifrado para los atributos accedidos por el cliente. Los permisos de cifrado permitidos, según lo definido por el GATT son los siguientes:

Cuadro 5.3: Tabla de Encriptación para Servicios GATT

Cifrado	**Descripción**
No encryption required (Security Mode 1, Level 1)	El atributo es accesible en un texto plano de conexión, no cifrada.
Unauthenticated encryption required (Security Mode 1, Level 2)	La conexión debe ser encriptada para acceder a este atributo, pero las claves de cifrado no tiene que ser autenticados (aunque pueden ser).
Authenticated encryption required (Security Mode 1, Level 3)	La conexión debe estar encriptada con una clave de acceso autenticado a este atributo.

Adicional a esto, existe la autorización para los servicios GATT permitidos para el cliente. Los cuales se representan como No autorizado y autorizado solamente. Sin embargo esto puede ser crucial para el desempeño.[13]

5.1.5. Programación de Servicios GATT

La programación correspondiente a la extracción de los datos se realiza a través de recepción de Servicios GATT.

Para este caso, vamos a utilizar Python y algunas librerías para realizar este proceso. En este proceso usaremos una serie de librerías y sus dependencias que nos

servirán para realizar lecturas en tiempo real de las características BLE de la pulsera Xiaomi 1S.

Arquitectura

Para comenzar a realizar la lectura de nuestra pulsera Xiaomi 1S, lo primero que debemos tener es un adaptador Bluetooth usb como el de la *Figura 5.4*.

Figura 5.4: Adaptador Bluetooth 4.0 USB

Este dispositivo nos ayudará a la recepción de las señales, para conectar nuestro dispositivo al microcomputador. Cabe destacar que BLE es una especificación básica de Bluetooth 4.0, 4.1 y 4.2, por lo tanto es necesario que el adaptador Bluetooth tenga una versión igual o superior a 4.0.[2]

Teniendo lo anterior en cuenta, podemos conectar nuestro adaptador a uno de los puertos USB de nuestro microcomputador. Al conectarlo y tener nuestra pulsera cerca, la arquitectura sería la mostrada en la *Figura 5.5*.

Figura 5.5: Arquitectura Raspberry Pi, Bluetooth y Pulsera

Ya que tenemos la arquitectura procederemos a encontrar la conexión que se genera entre nuestro Raspberry Pi y la pulsera Xiaomi 1S a través del Bluetooth.

Encontrando el Dispositivo

Para revisar la conexión de nuestra pulsera Xiaomi 1S debemos ingresar por la consola del Raspberry Pi y utilizar la herramienta *hcitool*.

Hcitool es una librería utilizada para configurar las conexiones Bluetooth y enviar algunos comandos especiales a dispositivos Bluetooth. Tenemos una serie de comandos disponibles para utilizar.

Por lo tanto podemos utilizar: *sudo hcitool dev*, para conocer el adaptador Bluetooth y dónde se encuentra disponible, en este caso podemos observar los resultados en la *Figura 5.6*.

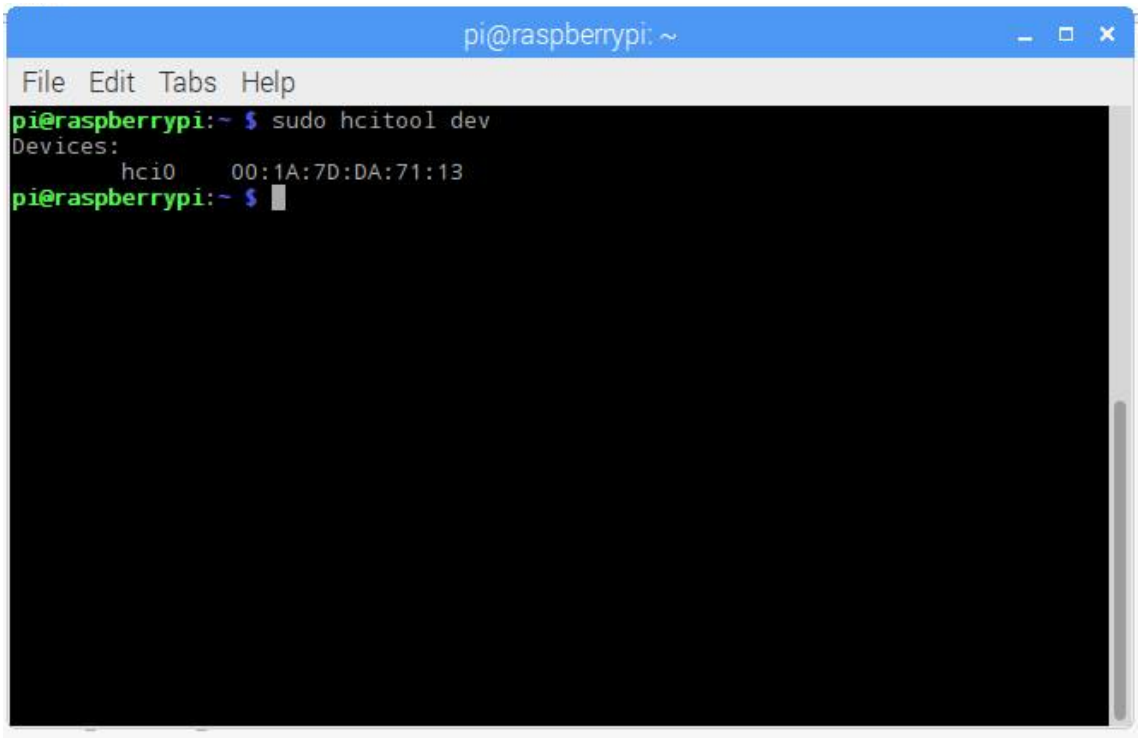


Figura 5.6: Resultado del comando: sudo hcitool dev

Ahora solo nos queda realizar un escaneo con la librería para saber que dispositivos se encuentran disponibles para la conexión.

sudo hcitool lescan

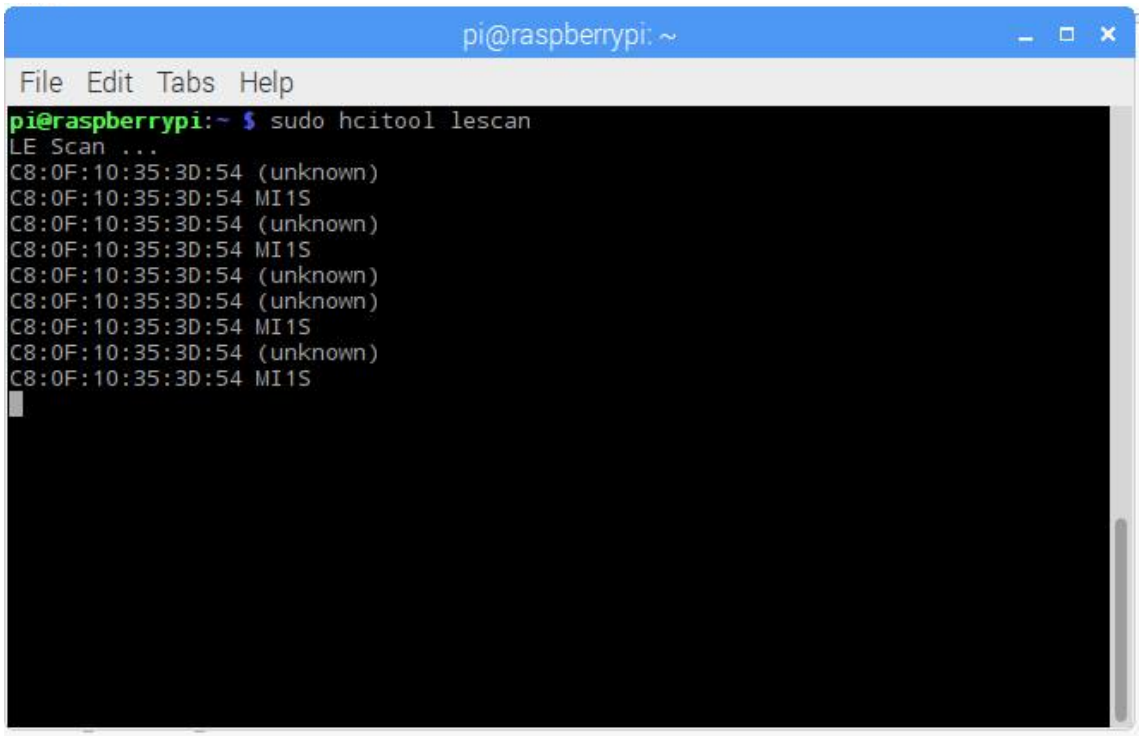


Figura 5.7: Resultado del comando: sudo hcitool lescan

Como podemos apreciar en la *Figura 5.7*, existe un dispositivo que fue analizado, en este podemos ver su *mac* y su *nombre*, afortunadamente se trata de nuestro Xiaomi 1S, por lo tanto ya estamos listos para la conexión.

Generando Conexión

Existen varias formas para conectar un dispositivo a través de Bluetooth LE, sin embargo en nuestro caso utilizaremos una librería esencial que se encontró y que tiene como principal función la conexión de Bandas Inteligentes Xiaomi. La librería en cuestión es *mibanda*, una librería pura de Python para acceder a la pulsera Xiaomi Mi Band. [4]

Como podemos apreciar, la librería está diseñada para la conexión de una pulsera Xiaomi, pero de una versión anterior a la que nosotros tenemos.

[4]https://bitbucket.org/OscarAcena/mibanda

Instalando MiBanda y sus dependencias

MiBanda es una librería que tiene dependencia de la conocida librería de conexiones BLE llamada Gattlib.[5] Para poder utilizarla realizaremos los siguientes comando en nuestro Raspberry Pi.

sudo apt-get install pkg-config sudo apt-get install libboost-python-dev sudo apt-get install libboost-thread-dev sudo apt-get install libbluetooth-dev >= 4.101 sudo apt-get install libglib2.0-dev sudo apt-get install python-dev

Algunas de estas librerías pueden demorar bastante tiempo en su instalación, pero debemos ser pacientes para su utilización.

Posteriormente debemos instalar Gattlib.

sudo apt-get install python-gattlib

Y ahora procedemos a instalar por último MiBanda.

sudo apt-get install python-mibanda

Programando con MiBanda

Como paso final, debemos crear un programa en Python para recibir los datos de la pulsera

Listing 5.2: Código Pasómetro y Pulso Cardíaco en Python

```
import time
import datetime
import mibanda
import gattlib

sd = mibanda.DiscoveryService()
device = mibanda.BandDevice("C8:0F:10:35:3D:54", "MI1S")
device.connect()

while True:
        date = time.strftime('%Y-%m-%d %H:%M:%S')
        print date
        steps = device.getSteps()
        print "Pasos: ",steps
        heartrate = device.requester.read_by_uuid("0000180d
            -0000-1000-8000-00805f9b34fb")[0]
```

[5] https://bitbucket.org/OscarAcena/pygattlib/src

```
pulso = ord(data[0]) + (ord(data[1])<<8)
print "Pulso: ", pulso

time.sleep(2)
```

Como podemos apreciar en el código anterior, nos conectamos automáticamente a nuestra pulsera Xiaomi 1S a través de la función *BandDevice()* ingresando la mac y el nombre del dispositivo anteriormente encontrados, para luego proceder a conectarnos a través de la función *connect()*.

Utilizamos *steps = decive.getSteps()*, la cual es una función definida en la librería *mibanda*, para calcular los pasos que se han realizado en el día.

Sin embargo también podemos apreciar que el pulso cardíaco no se puede obtener de la misma forma, esto es debido a que *mibanda* fue diseñado para una pulsera Xiaomi de versiones anteriores, que no posee sensor de pulso cardíaco. Es por esto que realizamos la lectura manual del Servicio GATT que queremos leer.

Si vemos el código de *mibanda* podremos apreciar lo siguiente:

Listing 5.3: Código MiBanda

```
class BandDevice(object):
def __init__(self, address, name):
self.address = address
self.name = name

self.requester = gattlib.GATTRequester(address, False)

def connect(self):
self.requester.connect(True)

def getAddress(self):
return self.address

def getName(self, cached=True):
if cached:
return self.name

data = self.requester.read_by_uuid(UUID_DEVICE_NAME)
return data[0]
```

```
def getBatteryInfo(self):
data = self.requester.read_by_uuid(UUID_BATTERY)
return BatteryInfo(data[0])
```

Aquí podemos apreciar algunas de las funcionalidades de la librería *mibanda*, y también que *self.requester*, contiene un atributo muy importante de la librería *gattlib*.

Listing 5.4: Código Gattlib

```
#include <boost/thread/thread.hpp>
#include <boost/python/dict.hpp>
#include <boost/python/extract.hpp>
#include <sys/ioctl.h>
#include <iostream>

#include <bluetooth/bluetooth.h>
#include <bluetooth/l2cap.h>
#include <bluetooth/hci.h>
#include <bluetooth/hci_lib.h>

#include "gattlib.h"

class PyThreadsGuard {
public:
PyThreadsGuard() : _save(NULL) {
_save = PyEval_SaveThread();
};

~PyThreadsGuard() {
PyEval_RestoreThread(_save);
};

private:
PyThreadState* _save;
};

boost::python::list
GATTRequester::read_by_handle(uint16_t handle) {
GATTResponse response;
read_by_handle_async(handle, &response);
```

```
        if (not response.wait(MAX_WAIT_FOR_PACKET))
        // FIXME: now, response is deleted, but is still registered on
        // GLIB as callback!!
        throw std::runtime_error("Device is not responding!");
        return response.received();
        }

        boost::python::list
        GATTRequester::read_by_uuid(std::string uuid) {
        PyThreadsGuard guard;
        GATTResponse response;

        read_by_uuid_async(uuid, &response);

        if (not response.wait(MAX_WAIT_FOR_PACKET))
        // FIXME: now, response is deleted, but is still registered on
        // GLIB as callback!!
        throw std::runtime_error("Device is not responding!");

        return response.received();
        }
```

Aunque parezca complejo, el anterior código nos dice que Gattlib utiliza las funciones de *read_by_handle* y *read_by_uuid* para recibir los Servicios GATT a partir de estas mismas, enviándoles el correspondiente handle o uuid del servicio a través de un string.

Por lo tanto en nuestro archivo Python para leer el servicio del pulso cardíaco podemos utilizar:

device.requester.read_by_uuid("0000180d-0000-1000-8000-00805f9b34fb")

La forma de imprimir lo que se recibe es genérica, debido a que se recibe un arreglo del servicio, de forma encriptada.

5.2. Software

Como vimos anteriormente en la sección de Hardware, hemos creado programas en Python que nos permiten obtener la información de nuestros sensores, ya sea sensores GPIO de nuestra Raspberry o sensores conectados a través de Bluetooth, como es el caso de nuestra pulsera. Sin embargo necesitamos almacenar esta información

para que se registre con el tiempo.

5.2.1. Base de Datos

Para almacenar la información pertinente al monitoreo de una persona en su hogar unipersonal, debemos en primera instancia tener una Base de Datos que nos permita este trabajo.

Este proyecto constará de una Base de Datos MySQL. MySQL es la base de datos de código abierto más popular del mundo. Con su rendimiento, confiabilidad y facilidad de uso comprobados, MySQL se ha convertido en la principal opción de base de datos para aplicaciones basadas en la Web, utilizada por propiedades web de alto perfil como Facebook, Twitter, Youtube y los cinco principales sitios web. Además, es una alternativa extremadamente popular como base de datos integrada, distribuida por miles de ISV y OEM.[16]

Creando una Base de Datos MySQL

En este proyecto utilizaremos cinco datos importantes, los cuales son: Pasos, Pulso, Movimiento, Temperatura y Humedad, entregados por nuestros sensores.

Para crear nuestra base de datos debemos tener principal cuidado en qué y cómo recibiremos estos datos. Como prueba utilizaremos una base de datos de MySQL del servidor local WampServer.[6] Comenzaremos creando tres diferentes tablas para los tres sensores que tenemos a nuestra disposición. Tal como se muestra en la *Figura 5.8*, las tres tablas tendrán de nombre *movimiento*, *temperatura* y *pulsera*.

[6]http://www.wampserver.com/en/

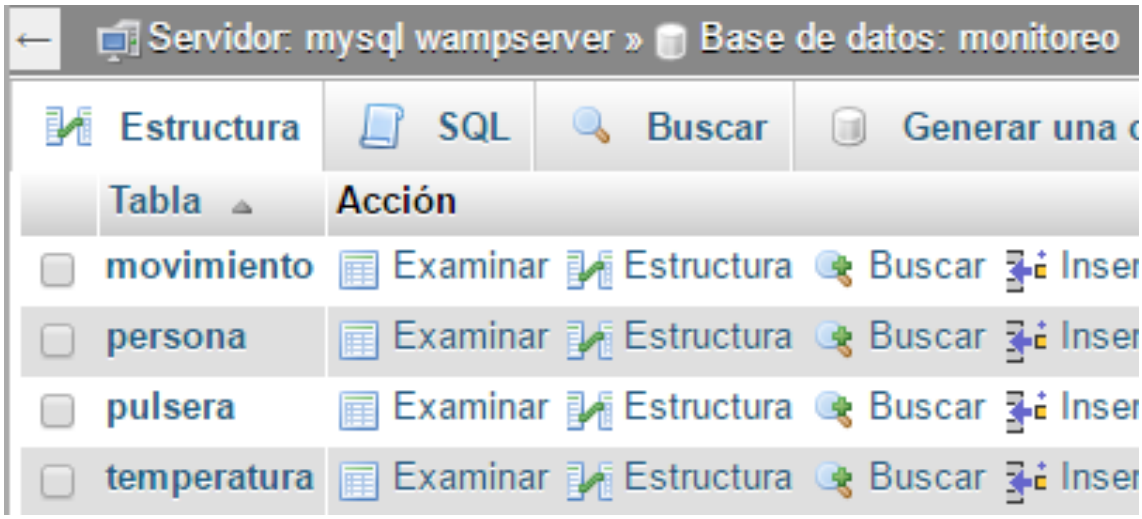


Figura 5.8: Tablas creadas de MySQL

La estructura de cada una de las tablas será analizada en las figuras *Figura 5.9*, *Figura 5.10* y *Figura 5.11*:

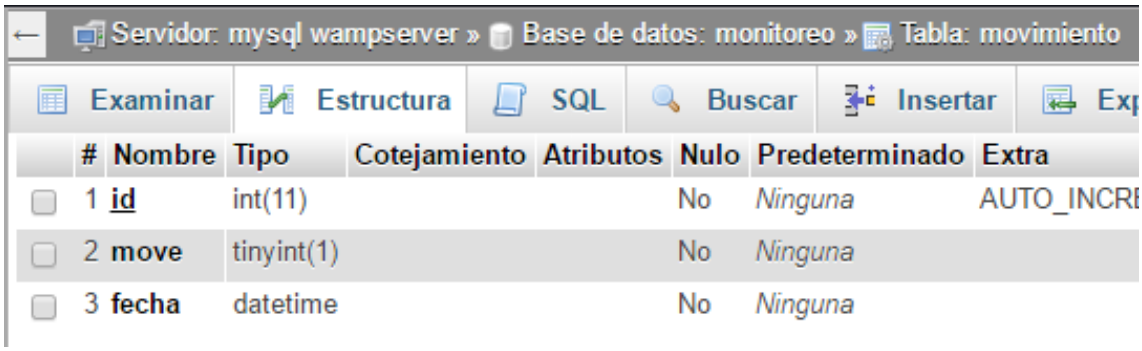
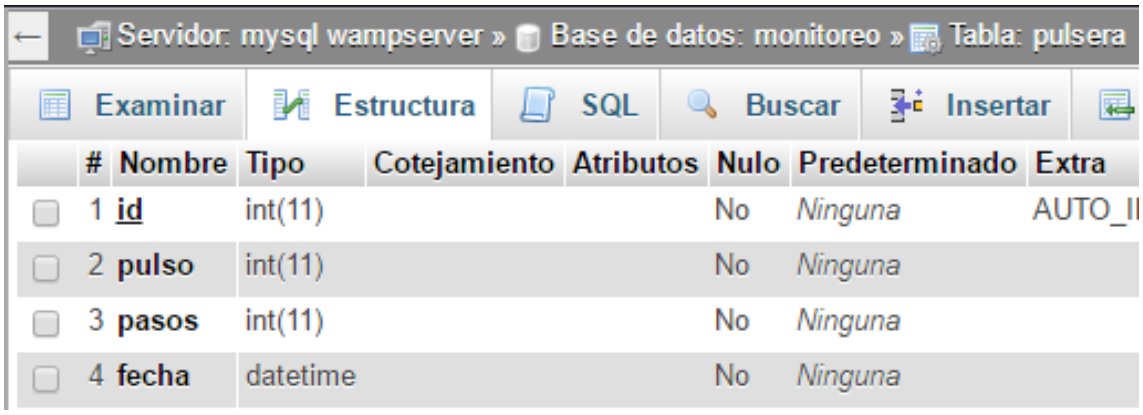
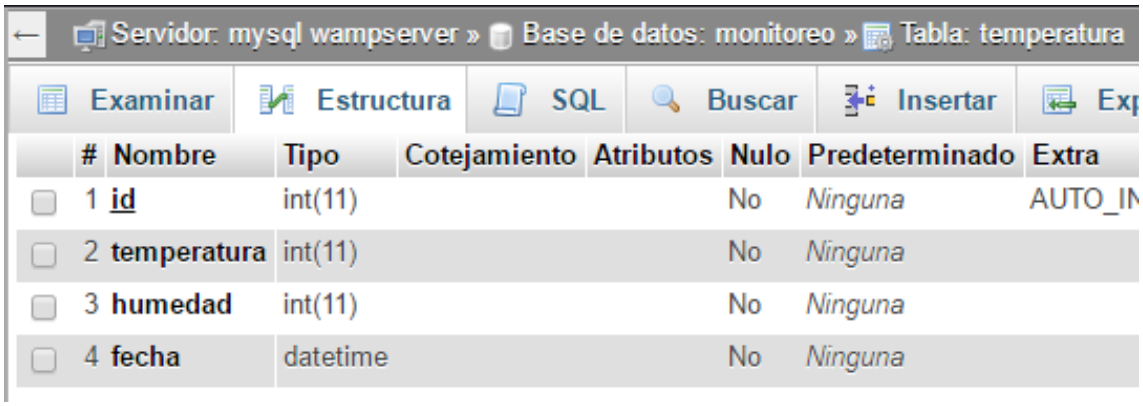


Figura 5.9: Estructura de la Tabla Movimiento



	#	Nombre	Tipo	Cotejamiento	Atributos	Nulo	Predeterminado	Extra
☐	1	id	int(11)			No	*Ninguna*	AUTO_I
☐	2	pulso	int(11)			No	*Ninguna*	
☐	3	pasos	int(11)			No	*Ninguna*	
☐	4	fecha	datetime			No	*Ninguna*	

Figura 5.10: Estructura de la Tabla Pulsera



	#	Nombre	Tipo	Cotejamiento	Atributos	Nulo	Predeterminado	Extra
☐	1	id	int(11)			No	*Ninguna*	AUTO_IN
☐	2	temperatura	int(11)			No	*Ninguna*	
☐	3	humedad	int(11)			No	*Ninguna*	
☐	4	fecha	datetime			No	*Ninguna*	

Figura 5.11: Estructura de la Tabla Temperatura

En estas tablas encontramos una serie de atributos, sin embargo cada una de ellas tiene un *id* representativo, además de una *fecha*. La fecha nos será muy útil al momento de realizar los reportes de nuestros datos.

Enviando la información con MySQL en Python

Para comenzar con el envío de información desde Python a MySQL es necesario utilizar la librería *Mysqldb* de Python, la que podemos utilizar instalándola a través de la linea de comandos:

sudo apt-get python-mysqldb

Y para utilizarla en nuestro código Python es bastante sencillo.

Listing 5.5: Código Gattlib

```
import MySQLdb

def connection():
try:
db = MySQLdb.connect(host="xx.xx.xx.xx",
user="xxx",
passwd="xxx",
db="monitoreo")
print "Connected"
return db

except MySQLdb.Error, e:
print "Error al Conectarse a la base de datos"
```

Con el código anterior estamos listos para enviar la información de nuestros sensores a las diferentes tablas incluidas en la base de datos *monitoreo*.

Envío de datos

El envío de los datos, como pudimos apreciar en la sección anterior, se envía a través del mismo programa que se encuentra recibiendo la información de los sensores. Esto debido a que debemos entregar los datos actuales y en tiempo real.

Para enviar los datps utilizaremos el siguiente código en nuestros programas Python.

Listing 5.6: Envío a MySQL de Sensores en Python

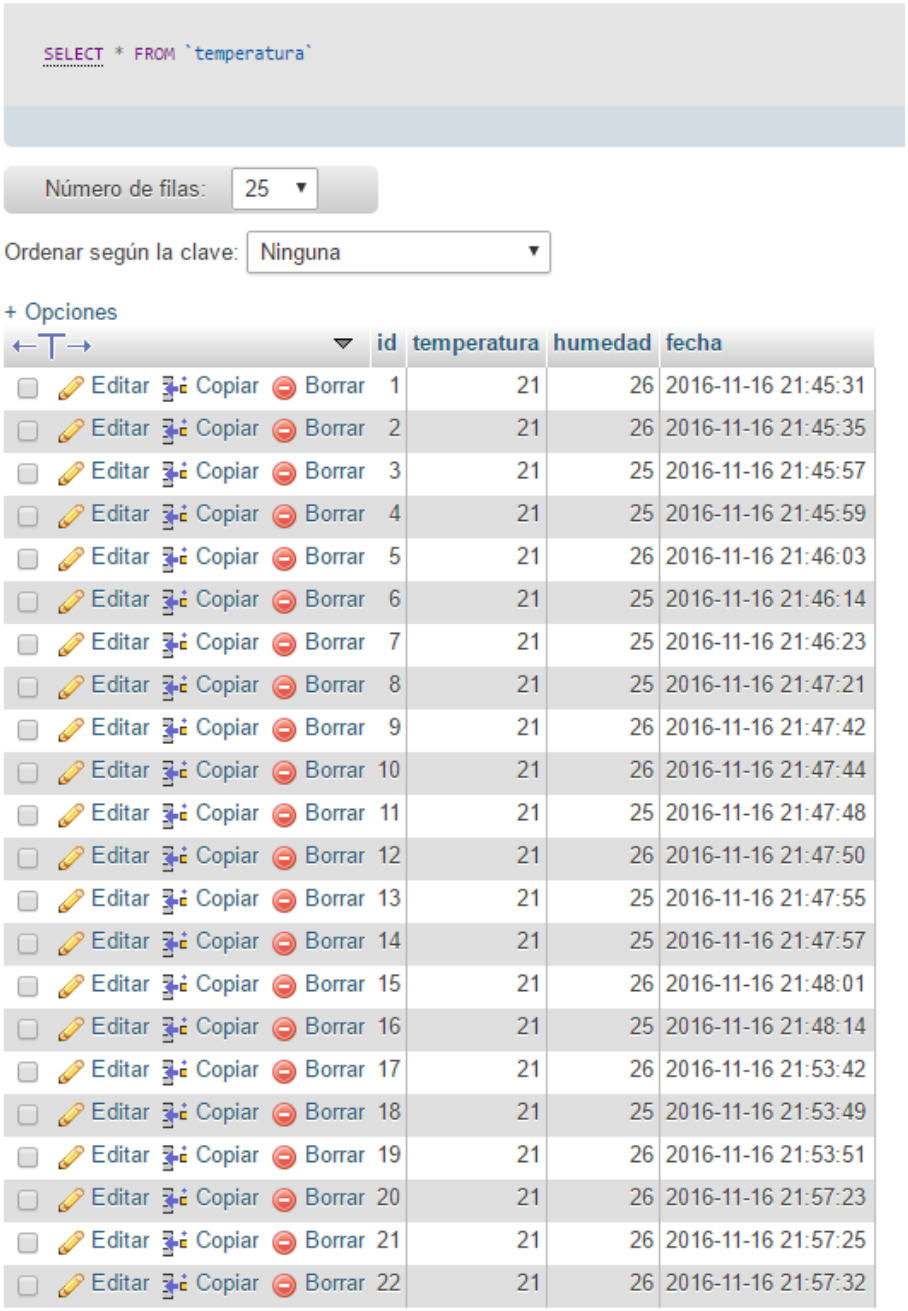
```
import RPi.GPIO as GPIO
import dht11
```

```
import time

GPIO.setwarnings(False)
GPIO.setmode(GPIO.BCM)
GPIO.cleanup()

GPIO.setmode(GPIO.BCM)
GPIO.setup(4, GPIO.IN)

while True:
db = connection()
cur = db.cursor()
date = time.strftime('%Y-%m-%d %H:%M:%S')

instance = dht11.DHT11(pin = 17)
result = instance.read()

if result.is_valid():
print ("Last valid input: " + str(datetime.datetime.now()))
print ("Temperature: %d C" % result.temperature)
print ("Humidity: %d %%" % result.humidity)
cur.execute("""INSERT INTO temperatura VALUES (%s,%s,%s)""", (
    result.temperature,result.humidity,date))

if GPIO.input(4) == True:
print "Hay Movimiento"
cur.execute("""INSERT INTO movimiento VALUES (%s,%s)""", (1,
    date))
else:
print "No Hay Movimiento"
cur.execute("""INSERT INTO movimiento VALUES (%s,%s)""", (0,
    date))

time.sleep(2)
db.commit()
db.close()
```

El código anteriormente descrito nos muestra que realizamos la conexión a la base de datos a partir de la función mostrada en *Listing 5.6* para luego en cada una de las lecturas de los sensores realizar una inserción SQL de las tablas correspondientes

a partir de los resultados. También se puede apreciar que se utiliza la librería de Python *time* para obtener en el formato *datetime* de MySQL, la fecha y hora actual. Este proceso es igual para los atributos de la Pulsera Xiaomi.

Una vez realizado este proceso nuestros datos ya se encontrarán en la base de datos MySQL tal como se muestra en la *Figura 5.12*.



```
SELECT * FROM `temperatura`
```

Número de filas: 25

Ordenar según la clave: Ninguna

+ Opciones

	id	temperatura	humedad	fecha
Editar Copiar Borrar	1	21	26	2016-11-16 21:45:31
Editar Copiar Borrar	2	21	26	2016-11-16 21:45:35
Editar Copiar Borrar	3	21	25	2016-11-16 21:45:57
Editar Copiar Borrar	4	21	25	2016-11-16 21:45:59
Editar Copiar Borrar	5	21	26	2016-11-16 21:46:03
Editar Copiar Borrar	6	21	25	2016-11-16 21:46:14
Editar Copiar Borrar	7	21	25	2016-11-16 21:46:23
Editar Copiar Borrar	8	21	25	2016-11-16 21:47:21
Editar Copiar Borrar	9	21	26	2016-11-16 21:47:42
Editar Copiar Borrar	10	21	26	2016-11-16 21:47:44
Editar Copiar Borrar	11	21	25	2016-11-16 21:47:48
Editar Copiar Borrar	12	21	26	2016-11-16 21:47:50
Editar Copiar Borrar	13	21	25	2016-11-16 21:47:55
Editar Copiar Borrar	14	21	25	2016-11-16 21:47:57
Editar Copiar Borrar	15	21	26	2016-11-16 21:48:01
Editar Copiar Borrar	16	21	25	2016-11-16 21:48:14
Editar Copiar Borrar	17	21	26	2016-11-16 21:53:42
Editar Copiar Borrar	18	21	25	2016-11-16 21:53:49
Editar Copiar Borrar	19	21	26	2016-11-16 21:53:51
Editar Copiar Borrar	20	21	26	2016-11-16 21:57:23
Editar Copiar Borrar	21	21	26	2016-11-16 21:57:25
Editar Copiar Borrar	22	21	26	2016-11-16 21:57:32

Figura 5.12: Registro en Tabla de Temperatura con datos del sensor DHT11

5.2.2. Servidor Web

El servidor web es una programa que procesa una aplicación desde el lado del Servidor y que puede ser accedida por el Cliente para obtener la información desde

cualquier parte vía Internet.

Apache

Es un servidor HTTP de código abierto para sistemas operativos modernos, incluyendo UNIX y Windows. El objetivo de este, es proporcionar un servidor seguro, eficiente y extensible que proporcione servicios HTTP en sincronización con los estándares HTTP actuales.[9]

En este proyecto se utilizará Apache desde un servidor local con WampServer en el cuál se incluirán archivos en formato PHP para generar consultas a la base de datos.

Los archivos que utilizaremos en este proceso tienen relación con las lecturas de las tablas, en este caso se utilizará un archivo por cada una de las tablas.

Medoo

Medoo es extremadamente ligero con un solo archivo, fácil de usar, fácil de aprender y optimizado con un alto rendimiento para aumentar la experiencia de desarrollo y la experiencia del usuario para aplicaciones web. Es adecuado para cada proyecto de desarrollo de PHP con base de datos SQL necesaria.[17]

Utilizaremos Medoo para la conexión a la Base de datos, de forma segura y confiable. Medoo contiene una forma simple de conectar y utilizar las consultas SQL de cualquier base de datos en el desarrollo de PHP, sin embargo dejo en claro que puede existir un sinnúmero de formas de acceder a bases de datos a través de consultas SQL.

Para esto solo descargamos el archivo php de Medoo desde su página oficial [7] en nuestro servidor web, y agregamos los atributos necesarios de nuestra base de datos, tal como se muestra en el *Listing 5.7*.

Listing 5.7: Conexión a la base de datos con Medoo

```
<?php
/*! Medoo 0.9.8.3 - Copyright 2015, Angel Lai - MIT license -
    http://medoo.in */
```

[7] http://medoo.in/

```
class medoo{
protected $database_type = 'mysql';
protected $charset = 'utf8';
protected $database_name = "monitoreo";
protected $server = "xxxx";
protected $username ="xxxx";
protected $password = "xxxx";
protected $database_file;
protected $socket;
protected $port = 3306;
protected $option=array();
protected $logs=array(); ...
}
?>
```

[8]

Leer la Base de datos desde el Servidor

Para leer la base de datos, como se dijo anteriormente utilizaremos Medoo. Existirá un archivo por cada una de las tablas de la base de datos. Los archivos, escritos en lenguaje PHP, nos ayudarán a que el cliente lea el último dato ingresado a la base de datos, de esta forma podremos tener una composición de datos actualizados a la realidad actual del monitoreo.

Un ejemplo de cómo obtener el último dato de la tabla *temperatura* se muestra en el código implementado en el *Listing 5.8*.

Listing 5.8: Conexión a la base de datos con Medoo

```
<?php
require "/medoo.min.php";

$database = new medoo();

$data = $database->select('temperatura', '*', [
"ORDER" => "id_DESC",
"LIMIT"=> 1
```

[8]Medoo es un archivo PHP complejo, en la última linea de este código, representado con ... el archivo original contiene más de 10000 caracteres que no pueden ser representados aquí.

```
    ]);

    echo json_encode($data);

    ?>
```

En este caso nos conectamos con Medoo y creamos un nuevo atributo a partir de la clase *medoo* con el que podremos utilizar llamadas SQL de una forma más sencilla. La consulta SQL realizada pide que desde la tabla *temperatura*, se obtengan todos los datos, donde el orden de la tabla se rija por la *id* en forma descendente, con un límite de una tupla.

Luego porsupuesto a partir de nuestro archivo PHP codificamos el resultado de la consulta SQL en forma de JSON, para que la lectura por parte del cliente sea más sencilla.

5.3. Aplicación Móvil

El uso de aplicaciones móviles se está haciendo cada vez más recurrente en la vida cotidiana, debido a que los smartphone llegaron a ser dispositivos indispensables para cualquier persona.

Para una persona que necesita monitorear a un familiar dependiente cuando no se encuentra cerca, no hay nada más conveniente que tener una aplicación móvil desde la cual estar atento en cualquier momento del día. Además de esta razón, es necesario tener una aplicación compatible con cualquier tipo de dispositivos.

5.3.1. PhoneGap

La creación de aplicaciones para cada plataforma (iPhone, Android, Windows y más) requiere diferentes marcos e idiomas. PhoneGap resuelve esto mediante el uso de tecnologías web basadas en estándares para conectar aplicaciones web y dispositivos móviles. Dado que las aplicaciones de PhoneGap son compatibles con los estándares, están preparadas para trabajar con los navegadores a medida que evolucionan.[1]

Phonegap es un framework para el desarrollo de aplicaciones móviles a través de herramientas genéricas tales como Javascript, HTML5 y CSS. El resultado de utilizar estas herramientas es poder tener una compatibilidad con todos los dispositivos

móviles. Además, maneja una serie de herramientas en forma de API[9] que permiten el acceso a características de los dispositivos móviles, tales como acelerómetro, cámara, notificaciones, etc.

5.3.2. Programación de Aplicación Móvil

Utilizando PhoneGap la primera parte es desarrollar la interfaz gráfica que será mostrada al Cliente, para esto utilizaremos HTML5[10]. Para darle un estilo definido a nuestra aplicación Móvil utilizaremos Bootstrap[11], el cuál nos ayudará a utilizar de forma ordenada y con una compatibilidad para los distintos tipos de dispositivos.

Javascript

Javascript se ha convertido en un lenguaje de programación por defecto para la comunicación con HTML. Se define como un lenguaje orientado a objetos que tiene un gran potencial para el desarrollo de aplicaciones. Actualmente existen muchos frameworks que lo utilizan como potencial herramienta en el desarrollo de aplicaciones que se concentren en el lado del Cliente, para tener una mayor respuesta de las aplicaciones sin que el Servidor cargue con la responsabilidad de procesar, enviar y recargar la aplicación por cada cambio.

Usaremos Javascript como una herramienta potencial que nos permita a través de los datos extraídos solamente en forma de JSON[12], para que sean juzgados en archivos Javascript que se conectarán directamente a la Vista del Cliente, realizando los cambios necesarios en tiempo real.

Vista de la aplicación

Lo que el Cliente ve en la aplicación móvil debe ser una parte esencial en el desarrollo. El archivo principal de la aplicación *index.html* implementado en el *Listing 5.9*, contiene una serie de parámetros registrados con atributos HTML que podrán ser leídos y rellenados con la herramienta Javascript.

[9] Application Programming Interface, es el conjunto de subrutinas, funciones y procedimientos (o métodos, en la programación orientada a objetos) que ofrece cierta biblioteca para ser utilizado por otro software como una capa de abstracción.

[10] es la quinta revisión importante del lenguaje básico de la World Wide Web, HTML.

[11] Bootstrap es el framework HTML, CSS y JS más popular para desarrollar proyectos móviles.

[12] JSON es un formato de texto ligero para el intercambio de datos

Listing 5.9: Extracto de HTML5 para aplicación móvil

```
<div class="page-content" id="page_content">
<ul class='list-group'>
<h4>Temperatura:</h4>
<li class='list-group-item'>
Actualmente
<span id="temperatura" class='badge'></span>
</li>
<h4>Humedad:</h4>
<li class='list-group-item'>
Actualmente
<span id="humedad" class='badge'></span>
</li>
<h4>Movimiento:</h4>
<span id="movimiento" class='badge'></span>
<li class='list-group-item'>
Actualmente en
<span id="movimiento" class='badge'></span>
</li>
<h4>Pasómetro:</h4>
<li class='list-group-item'>
Hoy
<span id="pasometro" class='badge'></span>
</li>
<h4>Ritmo Cardíaco:</h4>
<li class='list-group-item'>
Actualmente
<span id="ritmo_cardiaco" class='badge'></span>
</li>
</ul>
</div>
```

Recepción de datos del Servidor

Para recibir los datos del Servidor, se utilizará la función *Ajax*[13] de *JQuery*[14] como se muestra en el código implementado en el *Listing 5.10*.

[13]Función de JQuery para realizar solicitudes asíncronas HTTP.

[14]Librería de Javascript que facilita la manipulación de atributos HTML, eventos, animaciones, etc.

Listing 5.10: Extracto de Javascript para recibir datos del Servidor

```
$(document).ready(function() {
$.ajax({
url: 'http://localhost/monitoreo/getPulsera.php',
success: function(data){
page_content.find("#ritmo_cardiaco").html(data[0]['pulso']+' lpm');
page_content.find("#pasometro").html(data[0]['pasos']+' pasos');
},
dataType:'json'
});
}
```

Como vemos en el código ejemplo anterior, obtenemos los datos del Servidor a través del archivo *getPulsera.php* el cual nos entrega el último dato agregado a la base de datos en forma de JSON, como se muestra en la *Figura 5.13*, que podemos leer con Javascript y realizar cambios en el HTML de la aplicación.

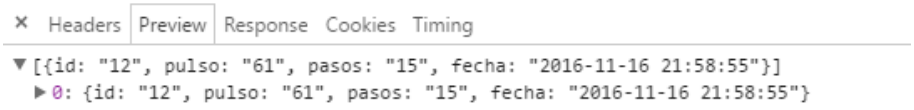


Figura 5.13: Respuesta de getPulsera.php en el Servidor

5.4. Manipulación de Datos

Como dijimos anteriormente, vamos a realizar la manipulación en el lado del Cliente, es decir la misma aplicación será la encargada, a través de *Javascript*, de juzgar y utilizar los datos de la forma que se estime conveniente. Vamos a representar los estados de una forma sencilla a través de colores representativos para cada situación tal como se muestran en la *Figura 5.14*.

Figura 5.14: Clases Bootstrap para representar estados de un Label

5.4.1. Reglas para la Representación de Información

Las reglas utilizadas para la representación de la información se basan en tres datos importantes que se analizarán tal como se muestran en el *Cuadro 5.4*.

Cuadro 5.4: Tabla de Reglas para la Representación de Información

Dato	**Métrica**	**ideal mín**	**ideal máx**	**crítico mín**	**crítico máx**
Temperatura	°C	18 °C	24 °C	15 °C	30 °C
Rimo Cardíaco	latidos/min	80lpm	110lpm	50lpm	150lpm
Humedad	%	30 %	60 %	15 %	70 %

[15]

En este caso los dos datos que abordaremos en la aplicación para la representación de estados son la *Temperatura* y el *Ritmo Cardíaco*. Como vemos en el *Cuadro 5.4*, estos ya poseen reglas para valores ideales y críticos, sin embargo es necesaria su representación adecuada en la aplicación.

Por lo tanto en el *Cuadro 5.5*, *Cuadro 5.6* y *Cuadro 5.7* veremos una representación de reglas.

Reglas para Temperatura

Cuadro 5.5: Tabla de Reglas de información para la Temperatura

Condición	**Representación**
If Temperatura <24 and Temperatura >18	Success
If Temperatura <18 and Temperatura >15	Warning
If Temperatura >24 and Temperatura <30	Warning
If Temperatura <15	Danger
If Temperatura >30	Danger

[15]La medición del ritmo cardíaco se analizará para una mujer mayor de 60 años

Reglas para Ritmo Cardíaco

Cuadro 5.6: Tabla de Reglas de información para el Ritmo Cardíaco

Condición	Representación
If Pulso <110 and Pulso >80	Success
If Pulso <80 and Pulso >50	Warning
If Pulso >110 and Pulso <150	Warning
If Temperatura <50	Danger
If Temperatura >150	Danger

Reglas para Humedad

Cuadro 5.7: Tabla de Reglas de información para la Humedad

Condición	Representación
If Humedad <60 and Humedad >30	Success
If Humedad <30 and Humedad >15	Warning
If Humedad >60 and Humedad <70	Warning
If Humedad <15	Danger
If Humedad >70	Danger

5.4.2. Representación de Reglas en la Aplicación

Ya teniendo las Reglas para la representación de la información definidas para la aplicación, solo nos queda utilizarlas de manera adecuada en en la aplicación móvil. Para esto veremos su código en *Javascript* en el *Listing 5.11* y el resultado en la *Figura 5.15*.

Listing 5.11: Javascript de la Aplicación Móvil con Reglas de representación

```
$(document).ready(function() {
$.ajax({
url: 'http://localhost/monitoreo/getPulsera.php',
success: function(data){
```

```
        page_content.find("#ritmo_cardiaco").html(data[0]['pulso']+' lpm');

        var pulso = data[0]['pulso'];

        page_content.find("#ritmo_cardiaco").removeClass();

        if(pulso < 110 && pulso > 80){
        page_content.find("#ritmo_cardiaco").addClass("pull-right label label-success");
        }
        if(pulso < 80 && pulso > 50){
        page_content.find("#ritmo_cardiaco").addClass("pull-right label label-warning");
        }
        if(pulso > 110 && pulso < 150){
        page_content.find("#ritmo_cardiaco").addClass("pull-right label label-warning");
        }
        if(pulso < 50){
        page_content.find("#ritmo_cardiaco").addClass("pull-right label label-danger");
        }
        if(pulso > 150){
        page_content.find("#ritmo_cardiaco").addClass("pull-right label label-danger");
        }
        page_content.find("#pasometro").html(data[0]['pasos']+' pasos')
            ;
        },
        dataType:'json'
        });
        }
```

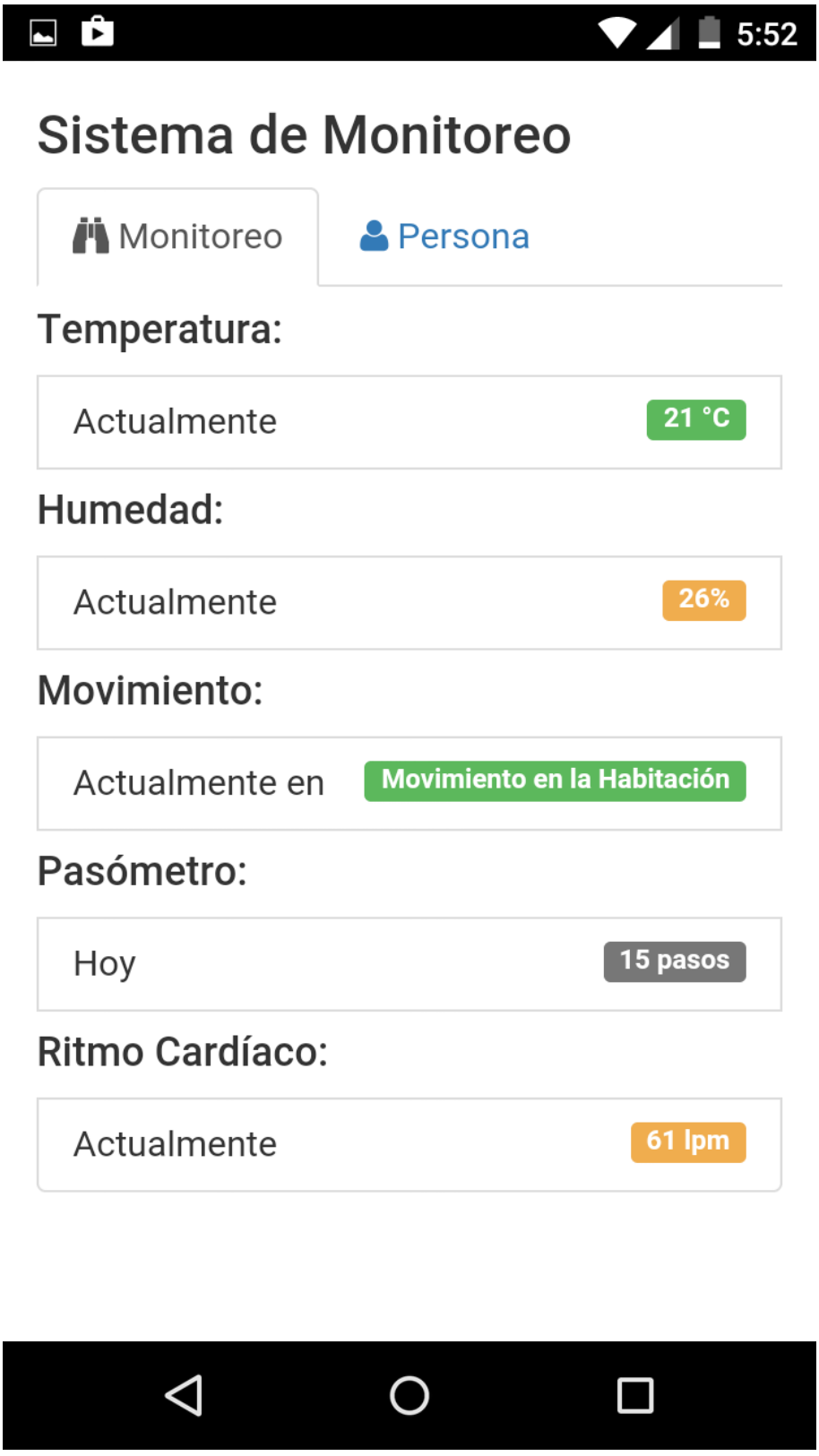


Figura 5.15: Captura de Pantalla de Aplicación Movil - Sección Monitoreo

5.4.3. Reglas con datos reales

Como vimos anteriormente en el *Cuadro 5.4*, los datos de la medición fueron analizados para una persona concreta, de una edad aproximada a 80 años y con valores estáticos, sin embargo esto no refleja la realidad de lo que queremos realizar.

Por lo anteriormente explicado, es necesario que el usuario pueda monitorear a la persona con valores que estime conveniente, debido a que no todas las personas que vamos a monitorear comparten la misma edad y por ende, tampoco el ritmo cardíaco estático que nombramos. Además de esto, la temperatura ideal y crítica en algunos lugares o zonas varía, también estas personas pueden no sentirse cómodas con una temperatura que nosotros estimemos conveniente.

Base de Datos Persona

Para esto crearemos una nueva tabla en nuestra base de datos que nos ayude a obtener los valores entregados por el usuario. La nueva tabla de nombre *persona* contiene las reglas necesarias de *Temperatura* y *Ritmo Cardíaco*, tal como se muestra en la *Figura 5.16*.

#	Nombre	Tipo	Cotejamiento	Atributos	Nulo	Predeterminado	Extra
1	id	int(11)			No	Ninguna	AUTO
2	edad	int(11)			No	Ninguna	
3	sexo	int(11)			No	Ninguna	
4	ideal_min_t	int(11)			No	Ninguna	
5	ideal_max_t	int(11)			No	Ninguna	
6	crit_min_t	int(11)			No	Ninguna	
7	crit_max_t	int(11)			No	Ninguna	
8	ideal_min_rc	int(11)			No	Ninguna	
9	ideal_max_rc	int(11)			No	Ninguna	
10	crit_min_rc	int(11)			No	Ninguna	
11	crit_max_rc	int(11)			No	Ninguna	

Figura 5.16: Estructura de la Tabla Persona

Formulario de Reglas de datos reales

En la vista de nuestra aplicación podremos realizar cambios de los valores reales para las Reglas de Información, en este caso, simplemente creamos un formulario que ya contiene valores por defecto almacenados en la base de datos para la persona, sin embargo pueden ser modificados dependiendo de la persona que necesitamos monitorear y la zona geográfica en la que nos encontremos, para el caso de la temperatura.

En este caso, el código HTML del *Listing 5.12* cambia, para agregar la nueva funcionalidad del formulario de la Persona, que se muestra como resultado en la *Figura 5.17* y *5.18*.

Listing 5.12: HTML del formulario para Persona

```
<div id="persona" class="tab-pane fade">
<form class="form" id="fom_persona">
<h3> <i class="fa fa-user" aria-hidden="true"></i> Información
    Persona</h3>
<div class="form-group">
<label for="edad"> Edad: </label>
<input type="number" class="form-control" id="edad">
</div>
<div class="form-group">
<label for="sexo">Sexo</label>
<select class="form-control" id="sexo">
<option value="0">Femenino</option>
<option value="1">Masculino</option>
</select>
</div>

<h3> <i class="fa fa-line-chart" aria-hidden="true"></i>
    Temperatura</h3>
<div class="form-group">
<label for="zona"> Ideal Mínima: </label>
<input type="number" class="form-control" id="ideal_min_t">
</div>
<div class="form-group">
<label for="zona"> Ideal Máxima: </label>
<input type="number" class="form-control" id="ideal_max_t">
</div>
<div class="form-group">
```

```
<label for="zona"> Crítica Mínima: </label>
<input type="number" class="form-control" id="crit_min_t">
</div>
<div class="form-group">
<label for="zona"> Crítica Máxima: </label>
<input type="number" class="form-control" id="crit_max_t">
</div>

<h3> <i class="fa fa-heartbeat" aria-hidden="true"></i> Ritmo
    Cardíaco</h3>
<div class="form-group">
<label for="zona"> Ideal Mínima: </label>
<input type="number" class="form-control" id="ideal_min_rc">
</div>
<div class="form-group">
<label for="zona"> Ideal Máxima: </label>
<input type="number" class="form-control" id="ideal_max_rc">
</div>
<div class="form-group">
<label for="zona"> Crítica Mínima: </label>
<input type="number" class="form-control" id="crit_min_rc">
</div>
<div class="form-group">
<label for="zona"> Crítica Máxima: </label>
<input type="number" class="form-control" id="crit_max_rc">
</div>

<button type="submit" id="formSubmit" class="btn btn-success">
    Guardar Datos</button>
<form>
</div>
```

Figura 5.17: Captura de Pantalla Aplicación Movil - Sección Persona

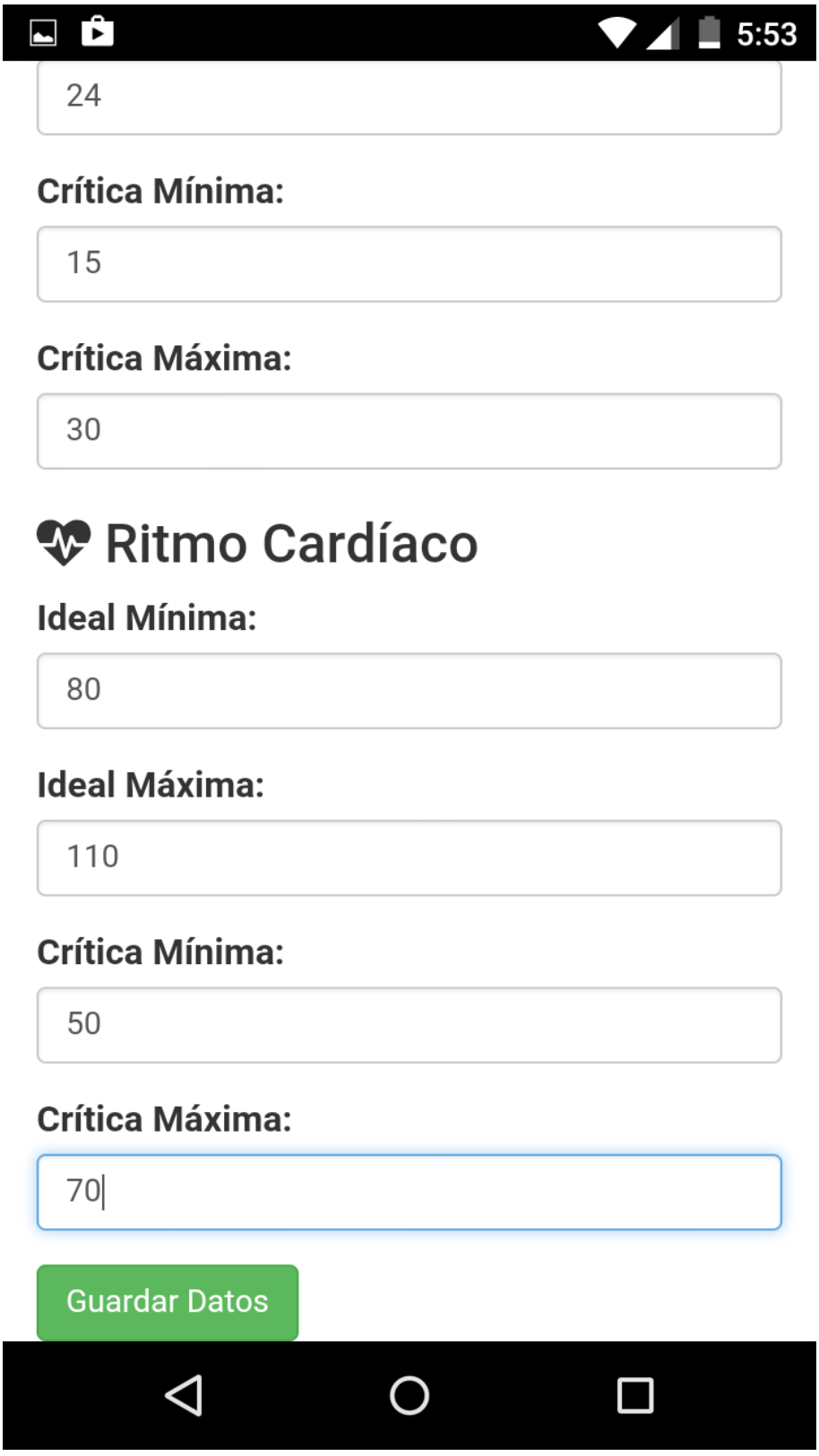


Figura 5.18: Captura de Pantalla Aplicación Movil - Sección Persona

Para guardar la información correspondiente vamos a utilizar la función *HTTP Post* la cuál nos permitirá comunicarnos con el Servidor Web a partir de Javascript con *Ajax*, tal como se muestra en el *Listing 5.13*.

Listing 5.13: Función para enviar los datos del formulario Persona

```
$('#formSubmit').click(function(e){
e.preventDefault();
var edad = $("#edad").val();
var sexo = $("#sexo").val();
var ideal_min_t = $("#ideal_min_t").val();
var ideal_max_t = $("#ideal_max_t").val();
var crit_min_t = $("#crit_min_t").val();
var crit_max_t = $("#crit_max_t").val();
var ideal_min_rc = $("#ideal_min_rc").val();
var ideal_max_rc = $("#ideal_max_rc").val();
var crit_min_rc = $("#crit_min_rc").val();
var crit_max_rc = $("#crit_max_rc").val();
var data = {
'edad': edad,
'sexo': sexo,
'ideal_min_t': ideal_min_t,
'ideal_max_t': ideal_max_t,
'crit_min_t': crit_min_t,
'crit_max_t': crit_max_t,
'ideal_min_rc': ideal_min_rc,
'ideal_max_rc': ideal_max_rc,
'crit_min_rc': crit_min_rc,
'crit_max_rc': crit_max_rc
};
$.ajax({
type:'POST',
data: data,
url:'http://localhost/monitoreo/updatePersona.php',
success:function(data) {
console.log(data);
}
});
});
```

Por parte del Servidor utilizaremos la función de Medoo para actualizar los datos

recibidos a partir de Post, *$database->update()*, que se muestra en el *Listing 5.14*.

Listing 5.14: PHP del Servidor Web para actualizar la tabla Persona

```
<?php
require "/medoo.min.php";

$database = new medoo();

$edad = $_POST['edad'];
$sexo = $_POST['sexo'];
$ideal_min_t = $_POST['ideal_min_t'];
$ideal_max_t = $_POST['ideal_max_t'];
$crit_min_t = $_POST['crit_min_t'];
$crit_max_t = $_POST['crit_max_t'];
$ideal_min_rc = $_POST['ideal_min_rc'];
$ideal_max_rc = $_POST['ideal_max_rc'];
$crit_min_rc = $_POST['crit_min_rc'];
$crit_max_rc = $_POST['crit_max_rc'];

var_dump($edad);

$lastauser = $database->update('persona', [
'edad' => $edad,
'sexo' => $sexo,
'ideal_min_t' => $ideal_min_t,
'ideal_max_t' => $ideal_max_t,
'crit_min_t' => $crit_min_t,
'crit_max_t' => $crit_max_t,
'ideal_min_rc' => $ideal_min_rc,
'ideal_max_rc' => $ideal_max_rc,
'crit_min_rc' => $crit_min_rc,
'crit_max_rc' => $crit_max_rc
],
[ 'id' => 1]
);
?>
```

Con esto ya podemos tener los datos actualizados de la Persona y pueden ser modificados fácilmente desde la aplicación por el usuario.

Representación de Información real

Una vez que ya contamos con las reglas modificadas por el usuario, es necesario utilizar estas para la representación de la información de Monitoreo en la aplicación. Para esto será necesario realizar algunos ajustes en el código Javascript para la representación de las reglas en la aplicación que vimos en el *Listing 5.11*, ahora implementados en el *Listing 5.15*.

Listing 5.15: Javascript de la Aplicación Móvil con Reglas de representación reales

```
var edad = $("#edad").val();
var sexo = $("#sexo").val();
var ideal_min_t = $("#ideal_min_t").val();
var ideal_max_t = $("#ideal_max_t").val();
var crit_min_t = $("#crit_min_t").val();
var crit_max_t = $("#crit_max_t").val();
var ideal_min_rc = $("#ideal_min_rc").val();
var ideal_max_rc = $("#ideal_max_rc").val();
var crit_min_rc = $("#crit_min_rc").val();
var crit_max_rc = $("#crit_max_rc").val();

$.ajax({
url: 'http://localhost/monitoreo/getPulsera.php',
success: function(data){
monitoreo.find("#ritmo_cardiaco").html(data[0]['pulso']+' lpm')
    ;
var pulso = data[0]['pulso'];
monitoreo.find("#ritmo_cardiaco").removeClass();

if(pulso < ideal_max_rc && pulso > ideal_min_rc){
monitoreo.find("#ritmo_cardiaco").addClass("pull-right label
    label-success");
}
if(pulso < ideal_min_rc && pulso > crit_min_rc){
monitoreo.find("#ritmo_cardiaco").addClass("pull-right label
    label-warning");
}
if(pulso > ideal_max_rc && pulso < crit_max_rc){
monitoreo.find("#ritmo_cardiaco").addClass("pull-right label
    label-warning");
```

```
        }
        if(pulso < crit_min_rc){
        monitoreo.find("#ritmo_cardiaco").addClass("pull-right label
            label-danger");
        }
        if(pulso > crit_max_rc){
        monitoreo.find("#ritmo_cardiaco").addClass("pull-right label
            label-danger");
        }
        monitoreo.find("#pasometro").html(data[0]['pasos']+' pasos');
        },
        dataType:'json'
        });
```

Como podemos ver se realizaron los cambios en las reglas de acuerdo a los ideales y críticos definidos por el usuario. Esto ahorrará que el usuario contenga notificaciones o alertas de situaciones que no son preocupantes.

5.5. Resumen del Capítulo

En este capítulo pudimos ahondar en el desarrollo e implementación de la solución del proyecto. En primera instancia comenzamos a visualizar la implementación de los Hardware utilizados en el sistema, tanto por parte del Raspberry Pi, en conjunto con los sensores que se utilizaron y conectaron a este mismo, a través del desarrollo de programas que nos permitían obtener la información. Posteriormente se especificaron los Software utilizados en este sistema, para el almacenado y extracción de datos en un Servidor Web. En estas secciones se explica paso a paso el proceso de desarrollo del sistema. Por último se abordó la implementación de un prototipo de Aplicación Movíl, para permitir la visualización de la información al cliente.

En el siguiente capítulo se abordarán las pruebas realizadas del prototipo del sistema de monitoreo, para validar su correcto funcionamiento en un ambiente real.

6. Pruebas

El prototipo de monitoreo fue desarrollado en casi la totalidad estimada y planificada, los componentes contienen un correcto funcionamiento y entregan los datos indispensables para el monitoreo de una persona. En este capítulo se verificarán las funcionalidades principales y el correcto funcionamiento del prototipo de sistema de monitoreo, con el fin de asegurar que el conjunto de datos recibidos es óptimo para monitorear a una persona, por lo tanto se aplicaron las siguientes medidas:

- Se seleccionó una persona mayor de edad.
- Se conectó el prototipo de sistema de monitoreo en la casa.
- Se colocó la pulsera en su mano derecha, previamente probada con su aplicación móvil para su correcto funcionamiento.
- Se le entregó a su hijo el smartphone para configurar la Información Personal y las reglas de Temperatura y Ritmo Cardíaco.
- En este capítulo se presentan los resultados de las pruebas generadas a partir de los datos obtenidos en un tiempo definido.

6.1. Datos de Prueba

6.1.1. Persona a monitorear

Como dijimos anteriormente la persona a monitorear es mayor de edad, los datos correspondientes a esta persona son entregados en el *Cuadro 6.1.*

Cuadro 6.1: Tabla de Datos de Persona a Monitorear

Información	**Definición**
Nombre	Blanca Rosa Parraguez Vera
Edad	81 años
Fecha de Nacimiento	12 de Julio de 1935
Dirección	Villa Convento Viejo, Miraflores número 53
Enfermedades	Diabetes, Hipersensibilidad y Vértigo
Doctor Tratante	Jaime Cabezas. Geriatra.
Establecimiento de atención	Clínica San Francisco, ciudad San Fernando.

6.1.2. Información Personal y Reglas

Para realizar esto el hijo de la señora Blanca Parraguez, realizó los cambios en la sección *Persona* de la aplicación móvil para modificar la edad y el sexo de la señora Blanca, además de la Temperatura y el Ritmo Cardíaco ideal y crítico, los cuales se muestran en el *Cuadro 6.2*.

Cuadro 6.2: Tabla de Información personal y reglas de la persona a monitorear

	Parámetro	**Valor**
Información Personal		
	Edad	81
	Sexo	Femenino
Temperatura		
	Ideal Mínima	18
	Ideal Máxima	24
	Crítica Mínima	15
	Crítica Máxima	30
Ritmo Cardíaco		
	Ideal Mínima	70
	Ideal Máxima	100
	Crítica Mínima	50
	Crítica Máxima	150

6.2. Resultados

A continuación se presentan los resultados que entregó el sistema de monitoreo a partir de las pruebas a la Señora Blanca Parraguez en un tiempo de 15 minutos. Los datos no pueden ser mostrados gráficamente en la aplicación, sin embargo se mostrará un gráfico creado a partir de los datos reales almacenados en la base de datos, ordenados por el parámetro *fecha*, que nos entrega la fecha y el tiempo exacto en el que fueron almacenados.

6.2.1. Nivel de Temperatura

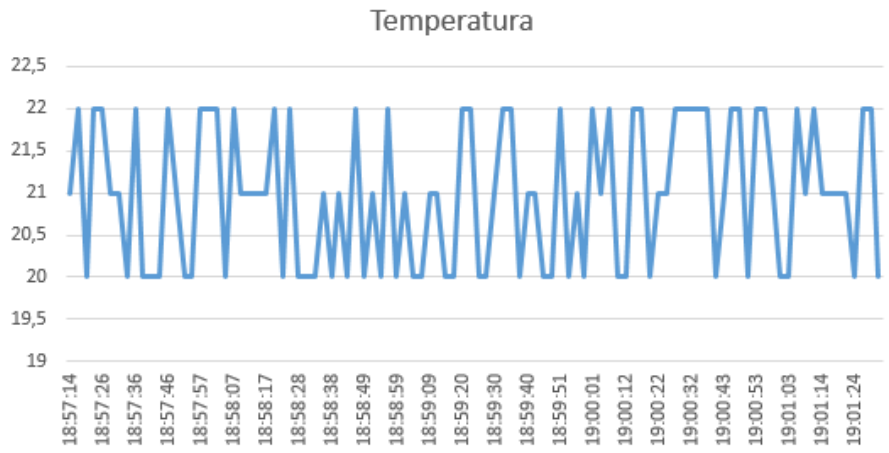


Figura 6.1: Gráfico representativo de los Datos de Temperatura en la Prueba

6.2.2. Ritmo Cardíaco

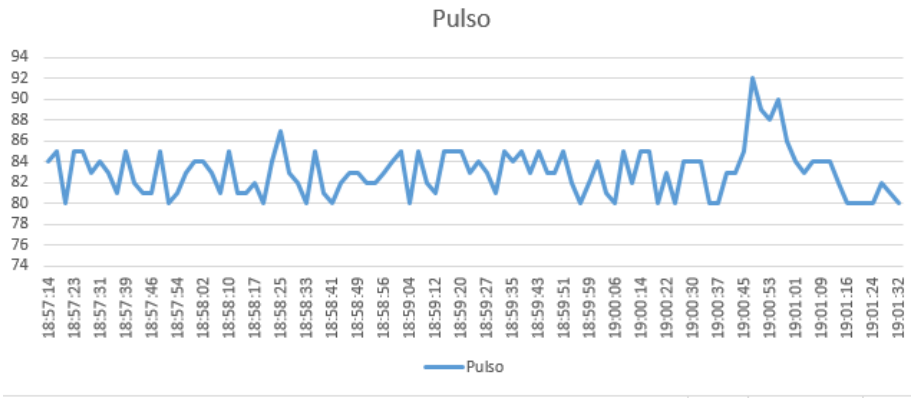


Figura 6.2: Gráfico representativo de los Datos de Ritmo Cardíaco en la Prueba

6.2.3. Observaciones

A partir de las pruebas realizadas podemos concretar que la información entregada por el sensor de temperatura es correcta, aunque en el gráfico de temperatura no se muestra toda la información capturada en los 15 minutos, debido a que fueron aproximadamente 150 datos, cabe destacar que los datos de temperatura no son almacenados cada dos segundos como esperábamos que sucediera, esto debido a las consultas y su velocidad de transmisión.

En cuanto al Ritmo Cardíaco, al igual en el gráfico de temperatura no se muestra toda la información capturada y los datos tampoco fueron almacenados cada dos segundos, además la pulsera Xiaomi 1S se desconectó en un momento deteniendo el programa, lo que puede causar problemas en su utilización a futuro.

6.3. Resumen del Capítulo

En este capítulo se presentaron las pruebas realizadas en un hogar real y con el cual se esperaba que el prototipo del sistema generara información fidedigna de una persona que vive en un hogar unipersonal y que es constantemente visitada por sus familiares. Mediante el conjunto de datos realizado por los usuarios, se validó que la información de la aplicación móvil mostraba correctamente la información entregada por los sensores.

En el siguiente capítulo se expondrán las conclusiones generadas a partir de la realización de este proyecto y de las pruebas realizadas. Posteriormente pasaremos a revisar los trabajos futuros que podrían ser de gran ayuda en el sistema.

7. Conclusiones

El proyecto desarrollado permitió desempeñar todos los conocimientos del área de la computación a un sistema social, esencialmente para la persona mayor de edad, que necesita ser monitoreada constantemente, ya sea por enfermedades o algún problema en su hogar. Mediante este proyecto se pretende demostrar que es posible estar alerta en cualquier momento de una persona dependiente sin importar la distancia a la cuál se encuentre. Además, se pretende que las personas dependientes vivan un proceso de adultés con la mayor calma posible, sin tener que ser enviados a un acilo para ser vigilados, teniendo una vida autónoma.

En cuanto al proyecto, se pudo demostrar que es posible implementar un sistema con un precio módico, el cual, a través de sensores comunes pueda entregar información esencial de una persona y su hogar unipersonal. Los datos pueden ser visualizados fácilmente a través de la aplicación móvil, sin la necesidad que profesionales gasten el tiempo vigilando directamente a la persona. Además, la visualización de la aplicación permite interpretar de manera sencilla los datos asociados al estudio, y también modificar los rangos de riesgo de las características monitoreadas, de acuerdo a lo que se estima conveniente con un doctor especialista.

En relación a la parte tecnológica, queda demostrado que es posible implementar el sistema de monitoreo mediante el envío de datos a través de JSON, relegando casi todo el trabajo del Servidor a la Aplicación del Cliente, para que así la información que sea enviada y recibida a través de internet, sea de datos simples y livianos, ahorrando el costo de la conexión a internet. De esta forma con un dispositivo 3G con un plan de datos relativamente barato y con una velocidad de transmisión baja, tendrá un funcionamiento adecuado en el sistema. Esto además fue pensado para que los hogares que se ubiquen en zonas sin cobertura puedan ser alcanzados por esta nueva tecnología.

Queda en manifiesto que es posible llevar este proyecto a magnitudes más elaboradas, no solo a datos de Temperatura, Movimiento, Humedad y Ritmo Cardíaco. Es posible la incorporación de sensores de todo tipo, por ejemplo se pueden agregar sensores de movimiento en todas las habitaciones de un hogar; se pueden instalar sensores de flujo de agua en los baños; la instalación de sensores de monóxido de carbono o combustible para cocinas y estufas o cualquier tipo de sensores que puedan ser compatibles con microcomputador.

En cuanto a los objetivos del sistema, se cumplieron íntegramente, desde la implementación del sistema (arquitectura y base de datos), hasta el desarrollo de una aplicación móvil para la visualización de los datos por parte de los usuarios que desean monitorear.

A partir de las pruebas se puede concluir que la pulsera Xiaomi 1S no es un dispositivo invasivo para la persona, después de unos momentos la persona a monitorear siquiera notó que aún la tenía. La aplicación fue sencilla de utilizar, de acuerdo a los comentarios de los familiares de la persona a monitorear, sin embargo se encontraron algunos detalles que debían ser confirmados con el médico tratante. Además expresaron que esperan que el sistema pueda ser completado con éxito a futuro.

Las pruebas fueron desarrolladas en base al estudio de monitoreo de una persona de avanzada edad como se meciona en el objetivo general de este documento. Sin embargo se hace necesario el desarrollo de pruebas de monitoreo en diferentes personas y lugares para conocer futuros problemas en el desarrollo de este trabajo. Se hace enfasis en que las características monitoreadas, principalmente el ritmo cardíaco, deben ser modificadas en los campos de riesgo de estas mismas por un profesional de la salud.

Queda claro que es una idea muy viable, ya que en la actualidad existe un problema social en cuánto a las personas dependientes, ya sean mayores de edad o personas con alguna discapacidad o enfermedad. La idea no es intervenir agresivamente con sus vidas, sino que el monitoreo puede ser una alternativa para que puedan continuar con sus vidas normalmente.

Por último y a modo personal, señalo que fue un agrado y un gran desafío el desarrollo de este proyecto, debido a la gran cantidad de conocimientos abordados, como también tecnologías computacionales, tales como base de datos, microcomputadores, Servidores Web, varios lenguajes de programación, metodologías, aplicaciones móviles y web, etc. También los dejo invitados, gracias a este proyecto, a enfocarse

directamente a la resolución de problemas sociales de nuestro país, debido a que con nuestros conocimientos pueden efectuarse soluciones como las de este proyecto, para cambiar las vidas de personas.

8. Trabajo futuro

A continuación se presenta un listado con funcionalidades que debido al corto tiempo de este proyecto y al tratarse de un prototipo no fueron incluidas, pero que a futuro agregarían un valor sustancial al Sistema de Monitoreo:

- Una mejora importante sería agregar una mayor cantidad de sensores de movimiento más completos que nos permitan conocer de forma exacta en qué lugar del hogar se encuentra la persona a monitorear.
- Otra mejora importante sería el desarrollo de una aplicación web, para desligar a la aplicación móvil de formularios extensos para el control de monitoreo.
- En cuanto a la seguridad, se podría ahondar más en el tema, generando permisos, tanto para los dispositivos y la aplicación móvil que permitan que el monitoreo de las personas no puedan ser visualizadas por terceros.
- De la mano de lo anterior, sería de gran importancia encriptar la información registrada en la base de datos para que la información que navega por internet sea más segura y confiable.
- También referente a la seguridad, podría ser indispensable que la aplicación web tuviera un registro de usuarios a través de tokens, tanto como la aplicación móvil tuviera un registro unipersonal para el dispositivo de la persona que quiere monitorear.
- Otra mejora importante sería permitir la exportación de datos a través de archivos tipo CSV o Excel, para que la aplicación sea utilizada por centros de cuidado, hospital, acilos, etc. Generando reportes para familiares.

- Realizar un estudio con el cual se vea la posibilidad de una optimización de las consultas SQL que reciben la información, aunque Medoo es una plataforma bastante segura al respecto.
- Agregar cualquier sensor que sea conveniente en el hogar, aunque la información sea mínima, en cualquier momento puede ser utilizada para prevenir algún desastre o salvar la vida de una persona.

Bibliografía

[1] Adobe. Adobe phonegap about - a high-level summary of what phonegap is all about. http://phonegap.com/about/.

[2] Argenox. A guide to selecting a bluetooth chipset. http://www.argenox.com/bluetooth-low-energy-ble-v4-0-development/library/a-guide-to-selecting-a-bluetooth-chipset/, Mayo 19, 2016.

[3] Inc Bluetooth SIG. Gatt characteristics. https://www.bluetooth.com/specifications/gatt/character

[4] Inc Bluetooth SIG. Gatt services. https://www.bluetooth.com/specifications/gatt/services.

[5] Croston. raspberry-gpio-python a python module to control the gpio on a raspberry pi. https://sourceforge.net/projects/raspberry-gpio-python/.

[6] División de Planificación y Desarrollo Servicio Nacional del Adulto Mayor. Indicadores sociodemográficos de las personas mayores a nivel territorial. http://www.senama.cl/filesapp/, Julio, 2013.

[7] DealExtreme. Xiaomi waterproof sports smart bluetooth v4.0 bracelet. http://www.dx.com/p/xiaomi-waterproof-sports-smart-bluetooth-v4-0-bracelet-w-silicone-wristband-black-blue-372526.

[8] Android Developers. Bluetooth low energy. https://developer.android.com/guide/topics/connectivi le.html.

[9] Apache Software Foundation. Apache, http server proyect. https://httpd.apache.org/.

[10] Raspberry Pi Foundation. What is a raspberry pi? https://www.raspberrypi.org/help/what-is-a-raspberry-pi.

[11] Python Software Fundation. About python - application for python. https://www.python.org/about/apps/.

[12] javiersb. Sensor de movimiento pir hc-sr501. http://zygzax.com/sensor-de-movimiento-pir-hc-sr501/, Noviembre 17, 2012.

[13] Akiba Robert Davidson Kevin Townsend, Carles CufÃ. Getting started with bluetooth low energy, chapter 4: Gatt (services and characteristics). https://www.safaribooksonline.com/library/view/getting-started-with/9781491900550/ch04.html.

[14] Digital Life. Digital life - shop. https://my-digitallife.att.com/learn/shop.

[15] Lively. Lively - how it works? http://www.mylively.com/how-it-works.

[16] Oracle. Oracle mysql: la base de datos de código abierto más popular del mundo. https://www.oracle.com/lad/mysql/index.html.

[17] The Medoo Project. About medoo. http://medoo.in/about.

[18] szaso. Dht11 sensor reading on raspberry pi 2. http://szazo.github.io/blog/2016/01/18/dht11-sensor-reading-on-raspberry-pi-2/.

[19] Sam Turner. Una introduccion a servidores de base de datos. http://blog.iweb.com/es/2014/04/servidores-de-bases-de-datos/2487.html, Abril 14, 2014.

[20] D-Robotics UK. Dht11 humidity and temperature sensor. http://www.micropik.com/PDF/dht11.pdf, Julio 30, 2010.

[21] RASPBERRY PI FOUNDATION UK. Gpio: Models a+, b+, raspberry pi 2 b and raspberry pi 3 b. https://www.raspberrypi.org/documentation/usage/gpio-plus-and-raspi2/.

[22] International Telecommunication Union. Information technology - procedures for the operation of object identifier registration authorities. http://www.itu.int/rec/T-REC-X.667/en.

[23] JJ Velasco. En qué consiste bluetooth le?, howpublished =.

[24] Ken Schwaber y Jeff Sutherland. La guía definitiva de scrum: Las reglas del juego. http://www.scrumguides.org/docs/scrumguide/v1/Scrum-Guide-ES.pdf, Julio, 2013.

ANEXOS

A. Código Fuente

A.1. raspberry-sensores.py

Listing A.1: Código Python para la recepción y almacenado de datos de Sensores de Movimiento PIR, DHT11 y Pulsera Xiaomi 1S

```
import RPi.GPIO as GPIO
import dht11
import time
import datetime
import MySQLdb
import mibanda
import gattlib

def connection():
try:
        db = MySQLdb.connect(host="xxx.xxx.xxx.xxx",
        user="",
        passwd="",
        db="monitoreo")
        print "Connected"
        return db

except MySQLdb.Error, e:
        print "Error al Conectarse a la base de datos"

sd = mibanda.DiscoveryService()
device = mibanda.BandDevice("C8:0F:10:35:3D:54", "MI1S")
```

```
device.connect()

GPIO.setwarnings(False)
GPIO.setmode(GPIO.BCM)
GPIO.cleanup()

GPIO.setmode(GPIO.BCM)
GPIO.setup(4, GPIO.IN)

while True:
        db = connection()
        cur = db.cursor()

        instance = dht11.DHT11(pin = 17)
        result = instance.read()

        date = time.strftime('%Y-%m-%d %H:%M:%S')

        if result.is_valid():
                print ("Last valid input: " + str(datetime.
                    datetime.now()))
                print ("Temperature: %d C" % result.temperature
                    )
                print ("Humidity: %d %%" % result.humidity)
                cur.execute("""INSERT INTO temperatura VALUES
                    (%s,%s,%s)""", (result.temperature, result.
                    humidity,date))

        if GPIO.input(4) == True:
                print "Hay Movimiento"
                cur.execute("""INSERT INTO movimiento VALUES (%
                    s,%s)""", (1,date))

        steps = device.getSteps()
        print "Pasos: ",steps
        heartrate = device.requester.read_by_uuid("0000180d
            -0000-1000-8000-00805f9b34fb")[0]
        pulso = ord(data[0]) + (ord(data[1])<<8)
        print "Pulso: ",pulso
```

```
        cur.execute("""INSERT INTO pulsera VALUES (%s,%s,%s)"""
            , (pulso,steps,date))

        time.sleep(2)
        db.commit()
        db.close()
```

A.2. index.html

Listing A.2: Código HTML para la vista de la aplicación Móvil

```
<!DOCTYPE html>
<html>

<head>
<meta charset="utf-8" />
<meta name="format-detection" content="telephone=no" />
<meta name="msapplication-tap-highlight" content="no" />
<meta name="viewport" content="user-scalable=no, initial-scale
    =1, maximum-scale=1, minimum-scale=1, width=device-width" />
<meta http-equiv="Content-Security-Policy" content="default-src
    * 'unsafe-inline'; style-src 'self' 'unsafe-inline'; media-
    src *" />

<link rel="stylesheet" type="text/css" href="css/bootstrap.css"
    >
<link rel="stylesheet" type="text/css" href="css/jquery.mobile
    -1.4.5.min.css">
<link rel="stylesheet" type="text/css" href="css/my-style.css">
<link rel="stylesheet" type="text/css" href="css/font-awesome.
    min.css">
<title>Sistema de Monitoreo</title>
</head>

<body>
<div class="container">
<h3>Sistema de Monitoreo</h3>

<ul class="nav nav-tabs">
```

```
<li class="active" id="monitoreoTab"><a data-toggle="tab" href=
    "#monitoreo"><i class="fa fa-binoculars"></i> Monitoreo</a><
    /li>
<li id="personaTab"><a data-toggle="tab" href="#persona"><i
    class="fa fa-user"></i> Persona</a></li>
</ul>

<div class="tab-content">
<div id="monitoreo" class="tab-pane fade in active">
<ul class='list-group'>
<h4>Temperatura:</h4>
<li class='list-group-item'>
Actualmente
<span id="temperatura" class='pull-right label label-default'><
    /span>
</li>
<h4>Humedad:</h4>
<li class='list-group-item'>
Actualmente
<span id="humedad" class='pull-right label label-default'></
    span>
</li>
<h4>Movimiento:</h4>
<li class='list-group-item'>
Actualmente en
<span id="movimiento" class='pull-right label label-default'></
    span>
</li>
<h4>Pasómetro:</h4>
<li class='list-group-item'>
Hoy
<span id="pasometro" class='pull-right label label-default'></
    span>
</li>
<h4>Ritmo Cardíaco:</h4>
<li class='list-group-item'>
Actualmente
<span id="ritmo_cardiaco" class='pull-right label label-default
    '></span>
</li>
</ul>
```

```
</div>
<div id="persona" class="tab-pane fade">
<form class="form" id="fom_persona">
<h3> <i class="fa fa-user" aria-hidden="true"></i> Información
    Persona</h3>

<div class="form-group">
<label for="edad"> Edad: </label>
<input type="number" class="form-control" id="edad">
</div>
<div class="form-group">
<label for="sexo">Sexo</label>
<select class="form-control" id="sexo">
<option value="0">Femenino</option>
<option value="1">Masculino</option>
</select>
</div>

<h3> <i class="fa fa-line-chart" aria-hidden="true"></i>
    Temperatura</h3>

<div class="form-group">
<label for="zona"> Ideal Mínima: </label>
<input type="number" class="form-control" id="ideal_min_t">
</div>
<div class="form-group">
<label for="zona"> Ideal Máxima: </label>
<input type="number" class="form-control" id="ideal_max_t">
</div>
<div class="form-group">
<label for="zona"> Crítica Mínima: </label>
<input type="number" class="form-control" id="crit_min_t">
</div>
<div class="form-group">
<label for="zona"> Crítica Máxima: </label>
<input type="number" class="form-control" id="crit_max_t">
</div>

<h3> <i class="fa fa-heartbeat" aria-hidden="true"></i> Ritmo
    Cardíaco</h3>
```

```
<div class="form-group">
<label for="zona"> Ideal Mínima: </label>
<input type="number" class="form-control" id="ideal_min_rc">
</div>
<div class="form-group">
<label for="zona"> Ideal Máxima: </label>
<input type="number" class="form-control" id="ideal_max_rc">
</div>
<div class="form-group">
<label for="zona"> Crítica Mínima: </label>
<input type="number" class="form-control" id="crit_min_rc">
</div>
<div class="form-group">
<label for="zona"> Crítica Máxima: </label>
<input type="number" class="form-control" id="crit_max_rc">
</div>

<button type="submit" id="formSubmit" class="btn btn-success">
    Guardar Datos</button>
<form>
</div>
</div>
</div>
<script type="text/javascript" src="js/jquery.min.js"></script>
<script type="text/javascript" src="js/bootstrap.min.js"></
    script>
<script type="text/javascript" src="js/jquery.mobile-1.4.5.min.
    js"></script>
<script type="text/javascript" src="js/my-js.js"></script>
</body>

</html>
```

A.3. my-js.js

Listing A.3: Código Javascript para la recepción y envío de datos, además de las reglas para el muestreo de la información

```
var persona = $('#persona');
```

```
var monitoreo = $('#monitoreo');
var formulario = $('#form_persona');

$('li').click(function(){
$('li').removeClass('active');
$(this).addClass('active');
var att = $(this).attr('id');
console.log("ATTR? "+att);
if (att == 'personaTab') {
console.log('Persona');
$('.tab-pane').removeClass('in active');
$('#persona').addClass('in active');
}
if (att == 'monitoreoTab'){
console.log('Monitoreo');
$('.tab-pane').removeClass('in active');
$('#monitoreo').addClass('in active');
}
})

$('#formSubmit').click(function(e){
e.preventDefault();
var edad = $("#edad").val();
var sexo = $("#sexo").val();
var ideal_min_t = $("#ideal_min_t").val();
var ideal_max_t = $("#ideal_max_t").val();
var crit_min_t = $("#crit_min_t").val();
var crit_max_t = $("#crit_max_t").val();
var ideal_min_rc = $("#ideal_min_rc").val();
var ideal_max_rc = $("#ideal_max_rc").val();
var crit_min_rc = $("#crit_min_rc").val();
var crit_max_rc = $("#crit_max_rc").val();
var data = {
'edad': edad,
'sexo': sexo,
'ideal_min_t': ideal_min_t,
'ideal_max_t': ideal_max_t,
'crit_min_t': crit_min_t,
'crit_max_t': crit_max_t,
'ideal_min_rc': ideal_min_rc,
'ideal_max_rc': ideal_max_rc,
```

```
        'crit_min_rc': crit_min_rc,
        'crit_max_rc': crit_max_rc
        };

        $.ajax({
        type:'POST',
        data: data,
        url:'http://localhost/monitoreo/updatePersona.php',
        success:function(data) {
        console.log(data);
        }
        });

        });

        $(document).ready(function() {
        var edad = $("#edad").val();
        var sexo = $("#sexo").val();
        var ideal_min_t = $("#ideal_min_t").val();
        var ideal_max_t = $("#ideal_max_t").val();
        var crit_min_t = $("#crit_min_t").val();
        var crit_max_t = $("#crit_max_t").val();
        var ideal_min_rc = $("#ideal_min_rc").val();
        var ideal_max_rc = $("#ideal_max_rc").val();
        var crit_min_rc = $("#crit_min_rc").val();
        var crit_max_rc = $("#crit_max_rc").val();

        $.ajax({
        url: 'http://localhost/monitoreo/getPulsera.php',
        success: function(data){
        monitoreo.find("#ritmo_cardiaco").html(data[0]['pulso']+' lpm')
            ;

        var pulso = data[0]['pulso'];

        monitoreo.find("#ritmo_cardiaco").removeClass();

        if(pulso < ideal_max_rc && pulso > ideal_min_rc){
        monitoreo.find("#ritmo_cardiaco").addClass("pull-right label
            label-success");
        }
```

```
if(pulso < ideal_min_rc && pulso > crit_min_rc){
monitoreo.find("#ritmo_cardiaco").addClass("pull-right label
    label-warning");
}
if(pulso > ideal_max_rc && pulso < crit_max_rc){
monitoreo.find("#ritmo_cardiaco").addClass("pull-right label
    label-warning");
}
if(pulso < crit_min_rc){
monitoreo.find("#ritmo_cardiaco").addClass("pull-right label
    label-danger");
}
if(pulso > crit_max_rc){
monitoreo.find("#ritmo_cardiaco").addClass("pull-right label
    label-danger");
}
monitoreo.find("#pasometro").html(data[0]['pasos']+' pasos');
},
dataType:'json'
});

$.ajax({
url: 'http://localhost/monitoreo/getTemperatura.php',
success: function(data){
monitoreo.find("#temperatura").html(data[0]['temperatura']+' °C
    ');
var temperatura = data[0]['temperatura'];

monitoreo.find("#temperatura").removeClass();

if(temperatura < 24 && temperatura > 18){
monitoreo.find("#temperatura").addClass("pull-right label label
    -success");
}
if(temperatura < 18 && temperatura > 15){
monitoreo.find("#temperatura").addClass("pull-right label label
    -warning");
}
if(temperatura > 24 && temperatura < 30){
monitoreo.find("#temperatura").addClass("pull-right label label
    -warning");
```

```
        }
        if(temperatura < 15){
        monitoreo.find("#temperatura").addClass("pull-right label label
            -danger");
        }
        if(temperatura > 30){
        monitoreo.find("#temperatura").addClass("pull-right label label
            -danger");
        }
        monitoreo.find("#humedad").html(data[0]['humedad']+'%');
        var humedad = data[0]['humedad'];

        monitoreo.find("#humedad").removeClass();

        if(humedad < 60 && humedad > 30){
        monitoreo.find("#humedad").addClass("pull-right label label-
            success");
        }
        if(humedad < 30 && humedad > 15){
        monitoreo.find("#humedad").addClass("pull-right label label-
            warning");
        }
        if(humedad > 60 && humedad < 70){
        monitoreo.find("#humedad").addClass("pull-right label label-
            warning");
        }
        if(humedad < 15){
        monitoreo.find("#humedad").addClass("pull-right label label-
            danger");
        }
        if(humedad > 70){
        monitoreo.find("#humedad").addClass("pull-right label label-
            danger");
        }
        },
        dataType:'json'
        });

        $.ajax({
        url: 'http://localhost/monitoreo/getMovimiento.php',
        success: function(data){
```

```
        monitoreo.find("#movimiento").removeClass();
        if(data[0]['move']=="1"){
        monitoreo.find("#movimiento").html('Movimiento en la Habitación
            ');
        monitoreo.find("#movimiento").addClass("pull-right label label-
            success");
        }
        else{
        monitoreo.find("#movimiento").html('No Hay Movimiento');
        monitoreo.find("#movimiento").addClass("pull-right label label-
            default");
        }
        },
        dataType:'json'
        });

        $.ajax({
        url: 'http://localhost/monitoreo/getPersona.php',
        success: function(data){
        persona.find('#edad').val(data[0]['edad']);
        persona.find('#sexo').val(data[0]['sexo']);
        persona.find('#ideal_min_t').val(data[0]['ideal_min_t']);
        persona.find('#ideal_max_t').val(data[0]['ideal_max_t']);
        persona.find('#crit_min_t').val(data[0]['crit_min_t']);
        persona.find('#crit_max_t').val(data[0]['crit_max_t']);
        persona.find('#ideal_min_rc').val(data[0]['ideal_min_rc']);
        persona.find('#ideal_max_rc').val(data[0]['ideal_max_rc']);
        persona.find('#crit_min_rc').val(data[0]['crit_min_rc']);
        persona.find('#crit_max_rc').val(data[0]['crit_max_rc']);
        },
        dataType:'json'
        });

        });
```