Trabajo Fin de Grado

Grado en Ingeniería de las Tecnologías de Telecomunicación

Aplicación móvil y web para la monitorización de datos recogidos mediante la pulsera inteligente MiBand 2 usando BLE y la plataforma Fi-ware

Autor: María Peñate Garrido

Tutor: María Teresa Ariza Gómez

Dep. Ingeniería Telemática
Escuela Técnica Superior de Ingeniería
Universidad de Sevilla

Sevilla, 2017

Trabajo Fin de Grado
Grado en Ingeniería de las Tecnologías de Telecomunicación

Aplicación móvil y web para la monitorización de datos recogidos por la pulsera inteligente Xiaomi MiBand 2 usando BLE y la plataforma Fi-ware

Autor:
María Peñate Garrido

Tutor:
María Teresa Ariza Gómez
Profesor titular

Dep. Ingeniería Telemática
Escuela Técnica Superior de Ingeniería
Universidad de Sevilla

Sevilla, 2017

Trabajo Fin de Grado: Aplicación móvil y web para la monitorización de datos recogidos por la pulsera inteligente Xiaomi MiBand 2 usando BLE y la plataforma Fi-ware

Autor: María Peñate Garrido

Tutor: María Teresa Ariza Gómez

El tribunal nombrado para juzgar el Proyecto arriba indicado, compuesto por los siguientes miembros:

Presidente:

Vocales:

Secretario:

Acuerdan otorgarle la calificación de:

Sevilla, 2017

El Secretario del Tribunal

A mi familia

A mis maestros y profesores

A mis amigos

Agradecimientos

Después de cuatro años, hoy es el día. Escribo este apartado de agradecimientos para finalizar mi trabajo de fin de grado. Ha sido un período de aprendizaje intenso, no sólo en el campo científico, si no también a nivel personal. Realizar este trabajo ha tenido un gran impacto en mi y es por eso por lo que me gustaría agredecer a todas aquellas personas que me han ayudado y apoyado durante el proceso.

En primer lugar, me gustaría agraceder a mis padres por hacerme posible llegar hasta aquí. Agradecerles sus consejos durante estos cuatro años, el esfuerzo que han realizado y el acompañarme en todos y cada uno de los momentos, malos y buenos. Sólo puedo daros las gracias. Gracias por el apoyo que me habéis dado porque sin él probablemente no podría estar escribiendo esto.

También me gustaría agredecer a mis compañeras de carrera el haberme alegrado este camino, no duro, sino intenso que hemos escogido. Cuando empezamos no podíamos imaginar este momento, el momento de estar escribiendo tu último trabajo del grado, pero lo hemos conseguido. Gracias por acompañarme día a día durante estos años, desde el primero hasta el último. Gracias por aguantar mi rutina, mi forma de ser y por supuesto mis días malos.

No puedo dejar atrás a una persona que ha vivido conmigo tres de estos cuatro años. Gracias por apoyarme día tras día, por hacerme ver las cosas con otro punto de vista, por alegrarte de mis logros más que yo misma y por tener siempre una sonrisa para mi cuando lo necesitaba.

Es importante mencionar a todos los profesores que han formado parte de mi formación en la escuela. Vuestros consejos, explicaciones y experiencias me han hecho estar donde estoy y luchar por mis sueños. Gracias por no hacerlo todo fácil, por poner dificultades en el camino; clave fundamental para saber que, aunque no se pueda conseguir a la primera, no podemos dejar de intentarlo.

Por último, agradecer a la tutora de este proyecto, Mª Teresa Ariza Gómez, el tiempo y empeño dedicado en ayudar a su realización.

María Peñate Garrido

Sevilla

Resumen

Debido a la intensidad con la que las nuevas tecnologías están formando parte de nuestra vida cotidiana, hoy en día parece interesante pensar en nuevas funcionalidades, procedimientos o dispositivos en los que las nuevas tecnologías puedan ayudarnos a facilitar nuestro día a día.

Es un hecho ya conocido el que los dispositivos *wearables* conforman el **futuro de la tecnología**, ya que nos ofrecen todas las facilidades y ventajas de los dispositivos actuales, pero de una forma más cómoda, sin embargo aún no se ha llegado a la fórmula del éxito.

En este caso hemos diseñado e implementado un software destinado a la recogida de datos medidos por la pulsera inteligente Xiaomi MiBand 2. Este consiste en una aplicación Android que se comunica con la pulsera mediante **Bluetooth Low Energy** para realizar mediciones en tiempo real del usuario que la esté utilizando. Estos datos se guardarán en la nube usando una nueva plataforma llamada **Fi-ware** que se ha desarrollado para el nuevo concepto del internet de las cosas. Además, estos datos se mostrarán en una web con el fin de que un profesional pueda visualizar todas las mediciones de forma conjunta.

Se ha decidido realizar el desarrollo para la plataforma Android ya que es el sistema operativo líder en España, en 2017 su cuota de mercado es del 92,0%, frente al 7,5% de iOS. Esto significa que 9 de cada 10 teléfonos vendidos en España disponen del sistema operativo Android.

Abstract

Due to the intensity with which the new technologies are a part of our daily life, it is interesting to think of new functionalities, procedures or devices in which the new technologies can help to facilitate our day to day.

It is a well-known fact that wearable devices were **the future of technology**, since it offers all the facilities and advantages of the current devices, but in a more comfortable way, however, it has not yet reached the key to success.

In this case we have designed and implemented a software for the data collection measured by the smartband Xiaomi MiBand 2. This consists on an Android application that communicates with the smartband through **Bluetooth Low Energy** to make measurements in real time of the user who is using it. This data will be saved in the cloud using a new platform called **Fi-ware** that has been developed for the concept of internet of things. In addition, these data can be displayed on a web in order to a professional can visualize all the measurements together.

It has decided to carry out the development for the platform Android since it is the leader of the operating systems in Spain, in 2017 its market share is 92.0%, as opposed to 7.5% of iOS. This means that 9 out of 10 phones sold in Spain have the Android operating system.

Índice

ÍNDICE DE TABLAS

ÍNDICE DE ILUSTRACIONES

1 INTRODUCCIÓN

Todo empieza con un sueño. Sueñalo y podrás lograrlo.

- Walt Disney -

Como introducción a esta memoria podremos encontrar la motivación para llevar a cabo este trabajo, así como el problema planteado y la solución llevada a cabo. Encontraremos el alcance del proyecto y varios antecedentes tecnológicos que son conceptos clave para el desarrollo de éste.

1.1 Motivación

En este proyecto se ha afrontado un problema de diseño de software poniendo en práctica muchos de los conocimientos adquiridos en el grado. Se ha hecho uso de varias tecnologías como Android y Java, a las que se prevé un amplio futuro en el sector de las Telecomunicaciones.

Se ha optado por el desarrollo de una aplicación Android ya que resulta evidente el crecimiento de usuarios que usan *Smartphones* hoy en día, convirtiéndose este dispositivo en una extensión más del ser humano, siendo inconcebible, en muchas ocasiones, una vida sin éste. Según estudios estadísticos realizados por la compañía Ericsson, conocidos como *Mobility Report,* 7 de cada 10 usuarios estarían en posesión de un telefóno inteligente (*Smartphone*) en el año 2020. Esta necesidad se ha ido incrementando en los últimos años desde que toda esta tecnología se puede llevar puesta gracias a los dispositivos wearables (complementos digitales que son capaces de interactuar de forma continua con el ser humano).

Otro aspecto a tener en cuenta en la elección de este trabajo ha sido la oportunidad de adentrarme en el aprendizaje de las dos tecnologías mencionadas anteriormente, de las que no tenía gran conomiento antes de la realización del proyecto.

Además, la inclusión de conceptos de IoT (*Internet of Things*) fue uno de los motivos principales por el que opté en realizar este trabajo. Este es un campo que me resulta interesante de conocer debido al impacto que se espera que tenga sobre la forma en la que la gente vive, trabaja, viaja, se entretiene e interactúa con las otras personas alrededor del mundo.

1.2 Objetivos

El principal objetivo de este proyecto es la recolección de datos procedentes de los sensores de la pulsera inteligente ***Xiaomi MiBand 2*** a través del diseño y desarrollo de una aplicación Android.

Además, siempre que el dispositivo Android tenga conexión a Internet, estos datos se guardarán en la nube para poder visualizarlos a través de una aplicación web.

1.3 Soluciones existentes en el mercado

En este trabajo no hemos diseñado un nuevo tipo de aplicación ya que la recolección de datos de la pulsera inteligente escogida para el proyecto es un problema al que ya se le había dado solución. Por tanto, tenemos disponible en *Google Play Store* varias aplicaciones con esta finalidad.

Debido a que la MiBand anterior no tenía utilidad por si sola debido a la falta de una pantalla para visualizar los datos, Xiaomi Inc. se vio obligada al diseño de la aplicación Android **MiFit** compatible hoy día con las distintas versiones de la pulsera.

Por otro lado, una aplicación que compite a grandes rasgos con la oficial de Xiaomi es la conocida como **Notify & Fitness for Mi Band.** Con ella podríamos expandir las funciones de nuestra Mi Band añadiendo funcionalidades que no están disponibles en la aplicación oficial.

Cabe citar que por ahora, no se ha encontrado ningún software que coincida plenamente con el desarrollo que se ha realizado en este proyecto. Ninguna de las aplicaciones anteriormente mencionadas posee la funcionalidad de recoger los datos de las pulseras conectadas a ambas aplicaciones para poder visualizarlos de forma conjunta en una aplicación web.

1.4 Solución planteada

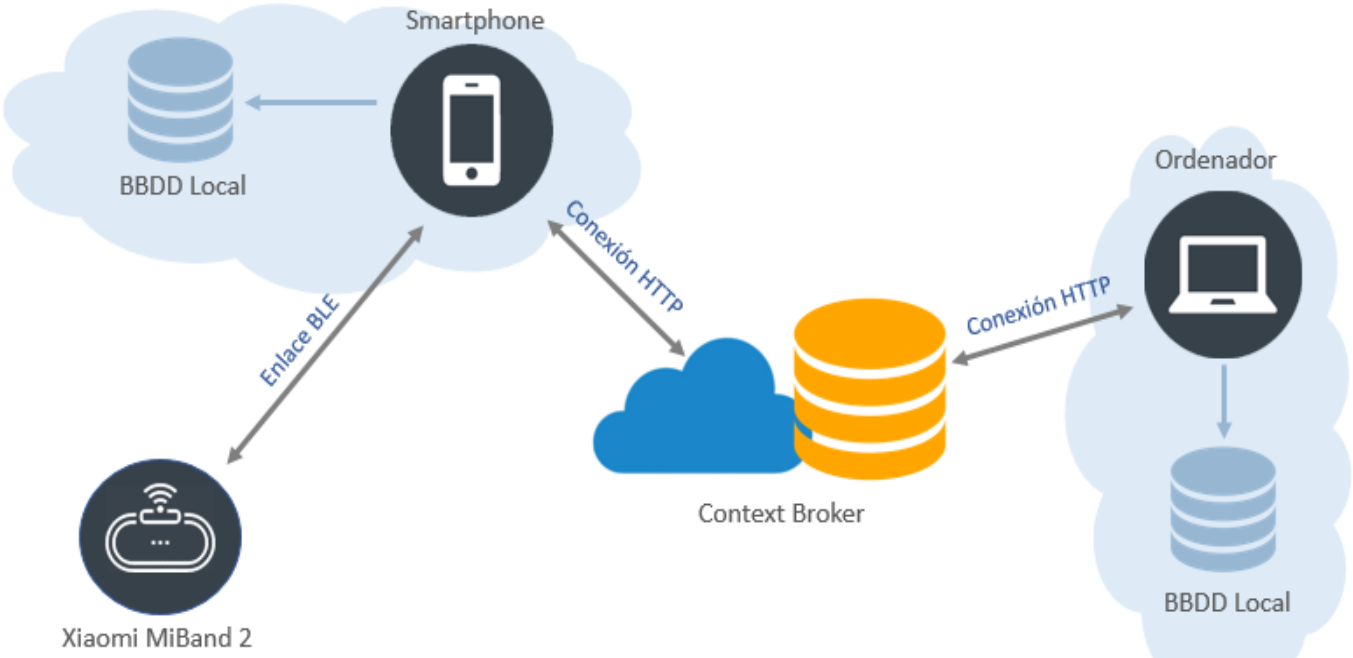


Ilustración 1: Solución planteada

En la ilustración 1 podemos observar la solución planteada a los objetivos de este proyecto cuya idea se desarrolla a continuación.

Dado que la Xiaomi MiBand 2 posee Bluetooth 4.0, es decir, Bluetooth Low Energy (BLE) y que este viene incorporado en todos los *Smartphones* diseñados a partir de 2013, se ha optado por esta tecnología para establecer la comunicación entre la pulsera y el *Smartphone* cuyas ventajas se desarrollan en un capítulo posterior.

Como hemos mencionado anteriormente, los datos recogidos por la pusera serán guardados en la nube. Para ello haremos uso de la plataforma FI-WARE que proporciona capacidades Cloud avanzadas, basadas en el estándar abierto OpenStack sobre las que adicionalmente se ofrece una gran librería de enablers genéricos (Generic Enablers) que facilitan el desarrollo de aplicaciones en múltiples sectores.

En concreto hemos utilizado uno de los componentes más importantes de Fi-Ware conocido como **Orion Context Broker**. Usando este *enabler* puedes registrarte a elementos del contexto y administrarlos a través de instrucciones de consulta y actualización.

La conexión con este componente tanto desde la aplicación Android como desde la aplicación web la realizaremos a través del protocolo de comunicación HTTP, en concreto mediante peticiones POST. Estas peticiones se enviarán desde ambos extremos. En nuestro caso la aplicación Android actualizará la información del Context Broker y la aplicación web recogerá dicha información para poder visualizarla.

1.5 Conocimientos a tener en cuenta para la realización del proyecto

1.5.1 Dispositos *Wearables*

La tecnología ha logrado avances sorprendentes en las últimas décadas. Ha sido capaz de aumentar su potencia a la par de disminuir su tamaño. Un tipo de tecnología que se está abriendo paso hoy en día es la tecnología que se puede vestir o llevar puesta, es decir, los conocidos como dispositivos *wearables*.

Los dispositivos *wearables* son aquellas prendas o complementos que incorporan un dispositivo electrónico que incluye un microprocesador, con la finalidad de ofrecer funciones dignas de productos de mayor tamaño sin apenas darnos cuenta de que los llevamos encima. Ahora mismo se encuentran en auge a pesar de llevar unos 500 años en el mundo.

La tecnología en los dispositivos *wearables* se encuentra en el fulgor de su historia, pues nos podemos remontar a la 4º década del siglo XVII donde comenzaron las ideas de esta tecnología. En la ilustración 2 podemos encontrar señalados los años donde aparecieron dispositivos clave para la revolución tecnológica que en la que nos encontramos hoy día.

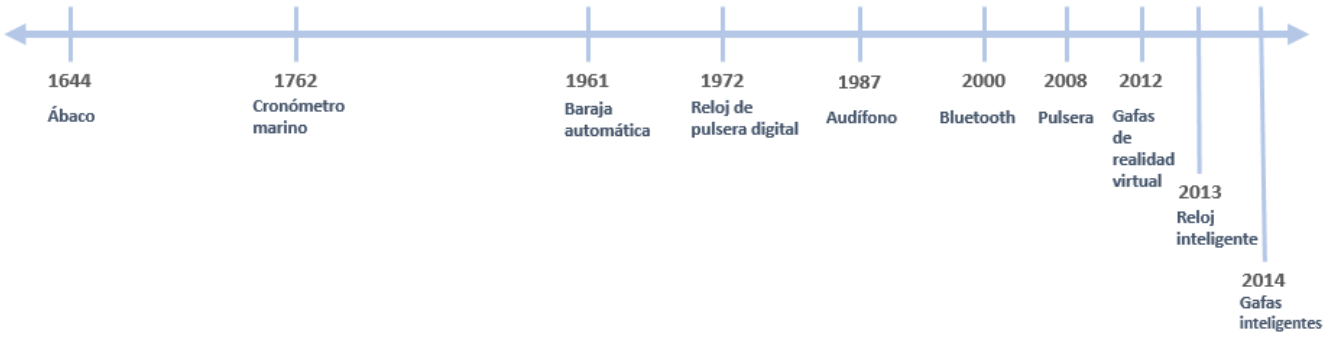


Ilustración 2: Esquema de la historia de los wearables

El objetivo principal de la existencia de estos dispositivos es la de ayudar al usuario a alcanzar un hábito de vida saludable, lo cual logran a través de diferentes funcionalidades utilizando los múltiples sensores con los que son equipados estos aparatos y que nunca dejan de monitorizar. El núcleo central de estos dispositivos, capaz de almacenar y analizar toda la información recopilada, es el teléfono inteligente (*Smartphone*).

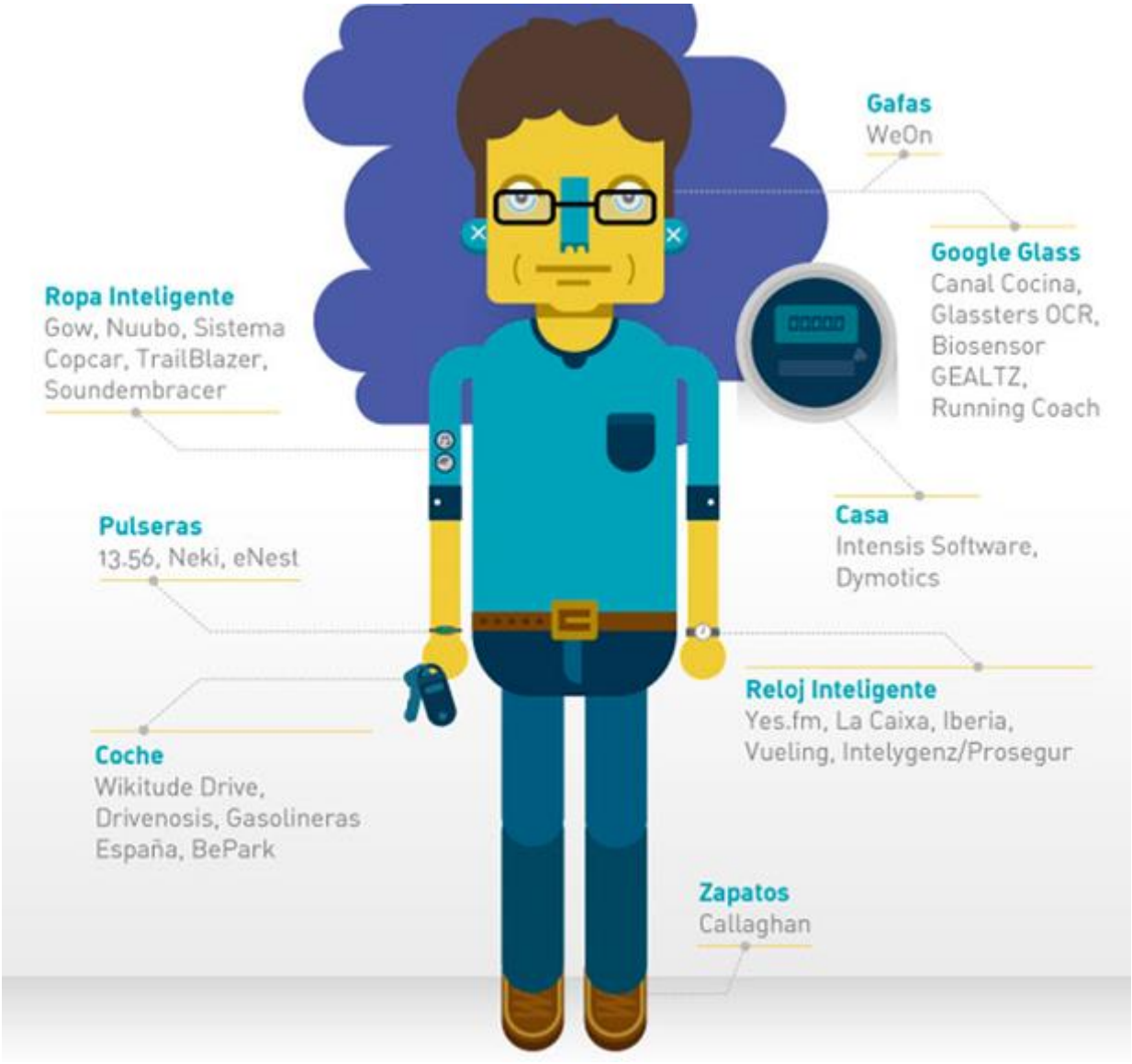


Ilustración 3: Diferentes tipos de dispositivos *wearables*

Como podemos observar en la ilustración 3, existen diferentes tipos de dispositivos wearables que se diferencian en la manera de "llevarlos", aunque todos los llevemos encima, pueden ser simples complementos como unas gafas o un reloj, o una prenda más como ropa o zapatos [1].

1.5.1.1 Pulseras inteligentes (*SmartBands*)

Los wearables que han logrado una mayor aceptación en la sociedad son los conocidos como relojes (smartwatch) o pulseras inteligentes (smartband) ya que estos dispositivos aparte de ofrecer un valor añadido al usuario, son capaces de aportar un análisis general sobre su salud gracias a los sensores que incorporan. Estos

sensores pueden ir desde simples cuantificadores de pasos que miden las calorías consumidas por el usuario, hasta una continua recopilación del pulso cardíaco, así como su temperatura y otros signos vitales significativos para mejorar el seguimiento de las enfermedades crónicas.

Las características generales de estos dispositivos que más tienden a tenerse en cuenta a la hora de adquirirlos son las dimensiones y el peso, dado que son de uso prolongado y debe ser mínimo el tiempo que el usuario prescinda de él.

Además, aparte de la multitud de sensores que pueden incorporar, deben ser capaces de incorporar varias tecnologías para la comunicación, algunas de carácter obligatorio, como las conexiones USB y Bluetooth, y otras adicionales, como WiFi o NFC, entre otras.

Gracias a los diversos estudios realizados se conoce que los jóvenes orientan el uso de estos dispositivos a mejorar su actividad física, mientras que en el sector adulto se utiliza en términos asociados a la salud. Además, si esta tecnología fuese capaz de basarse en el más básico comportamiento humano de acción-recompensa, como se indica en los principios de la e-Salud, aumentaría considerablemente el número de usuarios que harían uso de ella [2].

1.5.2 Internet de las cosas

Internet de las cosas (Internet of things, IoT) es un concepto que se refiere a la interconexión digital de objetos cotidianos con internet.

El concepto de internet de las cosas fue propuesto por Kevin Ashton en el Auto-ID Center del MIT en 1999 donde se realizaban investigaciones en el campo de la identificación por radiofrecuencia en red (RFID) y tecnologías de sensores.

Los IoT Ecosystems (o Internet de las cosas) se refieren a una red de objetos cotidianos conectados a través de Internet, lo que les permitirá recolectar e intercambiar datos de forma constante. Esta tendencia ha sido llamada "La siguiente Revolución Industrial", debido al impacto que se espera que tenga esta sobre la forma en la que la gente vive, trabaja, viaja, se entretiene e interactúa con las otras personas, los negocios y los gobiernos alrededor del mundo.

Podemos encontrar numerosas aplicaciones y ejemplos que hacen uso de este concepto. A continuación citamos algunos ejemplos:

- Los dispositivos wearables
- Aplicaciones para el hogar, por ejemplo, distribuyendo una serie de sensores y procesadores, podríamos automatizar el control de las ventanas, la temperatura del hogar, las luces, etc.
- Otros de los principales campos de aplicación del *Internet of Things* son las ciudades, haciéndolas más inteligentes y eficientes.
- Si llevamos el *Internet of Things* a terrenos más amplios como la seguridad nacional o las empresas, la trascendencia y las posibilidades son aún mayores. Por ejemplo: huertos automatizados, alumbrados inteligentes, supervisión de máquinas…

Como podemos ver, el *Internet of Things* es claramente el siguiente gran paso de la industria tecnológica. Abre un mundo de posibilidades incalculable, mayor incluso que el que abrió en su momento la era digital. Los primeros pasos ya se están dando (estándares, primeros prototipos y proyectos, etc.), pero, según diversos analistas, no será hasta 2020 cuando el *Internet of Things* comience a ser algo mucho más asentado y común entre los mortales [3].

1.5.3 e-Salud

El primer paso hacia la e-Salud se dió en el año 1998 por la Organización Mundial de la Salud (OMS) cuando reconoció la importancia de Internet en términos relacionados con la salud.

La e-salud consiste en brindar servicios sanitarios a pacientes, en los que el factor distancia sea muy relevante, mediante el uso de las tecnologías de la información y comunicación. Estos servicios pueden ser tanto en atención primaria como en hospitalaria y pasan por el manejo de información entre facultativos para el tratamiento o diagnóstico e incluso para solucionar dudas a padres con niños recien nacidos sanos sin necesidad de acudir a la consulta del centro de salud [4] [5].

1.5.3.1 Principios de la e-Salud.

Los principios que se van a citar a continuación deben ser el objetivo de todas las herramientas que se desarrollen con fines sanitarios. Estos principios son los siguientes:

- **Eficiencia.** La e-Salud debería evitar los diagnósticos duplicados o innecesarios.
- **Evidencia.** Someter los estudios a rigurosas evaluaciones científicas.
- **Mejorar la calidad de atención**. Al mejor la eficiencia, la calidad de atención debe mejorar.
- **Potenciación de los consumidores y pacientes.** El paciente podrá facilitar una mejor infomación al personal sanitario a través de Internet.
- **Fomentar una nueva relación entre el cliente y el profesional de la salud**. La toma de decisiones debe ser compartida.
- **Educación a través de fuentes digitales.** Los profesionales estarían en continuo aprendizaje gracias a las nuevas tecnologías.
- **Permitir el intercambio de información**.
- **Extender el foco de atención.** Extensión geográfica.
- **Ética.**
- **Equidad.** La atención sanitaria debe ser equitativa [2].

1.6 Estructura de la memoria

- **Capítulo 1: Introducción**

  En este capítulo podremos encontrar la motivación para llevar a cabo el proyecto asi como el problema y la solución planteada.

  También encontraremos unas breves introducciones a conceptos tecnológicos que están relacionados con el proyecto como, por ejemplo, los dispositivos *wearables* o el concepto del internet de las cosas.

- **Capítulo 2: Recursos**

  Este capítulo se centra en la descripción de los recursos utilizados para la realización del proyecto completo tanto a nivel de *software* como *hardware* indicando sus especificaciones (*hardware*) o, por el contrario, una descripción del mismo (*software*).

- **Capítulo 3: Estimación temporal**

  En este capítulo encontraremos una comparación entre el tiempo que estimamos antes de comenzar el proyecto y el tiempo real, especificando horas concretas para cada tarea.

- **Capítulo 4: Requisitos del sistema**

  En este capítulo vamos a realizar el proceso conocido como Ingeniería de requisitos.

  Se realizará:

  - Descripción de los requisitos localizados en el sistema tanto funcionales como no funcionales.
  - Descripción de los casos de uso del sistema.
  - Definición de los actores del sistema.
  - Diagramas de actividad del sistema, tanto Android como Web.
  - Prototipos de ambas aplicaciones.

- **Capítulo 5: Aplicación Android**

  Este capítulo se centrará en la integración de la tecnología Bluetooth Low Energy con la plataforma Android. Se realizará una descripción de la tecnología y el código utilizado para la comunicación con el dispositivo wearable utilizado para la realización del proyecto.

  Además, se incluye una explicación completa de la aplicación desarrollada adjuntando capturas de la misma.

- **Capítulo 6: Fi-ware**

  En este capítulo podremos encontrar un estudio de esta nueva plataforma con la descripción completa del *Generic Enabler* utilizado para la realización del proyecto.

  También encontraremos la explicación del protocolo y de los estándares utilizados para la comunicación con ambas aplicaciones.

- **Capítulo 7: Aplicación web**

  Este capítulo contendrá información de los distintos lenguajes utilizados para la creación de la aplicación web y el código desarrollado para las partes más conflictivas de la misma.

  Además, se incluye una explicación completa de la aplicación desarrollada adjuntando capturas del comportamiento de la web.

- **Capítulo 8: Aplicación web**

  En este capítulo se describirán una serie de pruebas realizadas para validar el correcto funcionamiento del sistema.

- **Capítulo 9: Conclusiones**

  En este capítulo podremos encontrar tanto las conclusiones del proyecto como las líneas futuras de continuación.

- **Anexos**

  En este apartado podremos encontrar los anexos del trabajo. Se incluye la guía de instalación del Context Broker.

2 RECURSOS

Una vez que aceptamos nuestros límites, podemos ir más allá de ellos.

- Albert Einstein-

Actualmente el uso de recursos tecnológicos adecuados es imprescindible para ser eficiente, es decir, para lograr mejores resultados en menor tiempo. En concreto, para la realización de este proyecto hemos hecho uso de un conjunto de recursos humanos, hardware y software que especificaremos a continuación.

2.1 Recursos humanos

En la realización de este proyecto han intervenido dos personas: la autora del mismo, implementándolo y desarrollándolo y la tutora mencionada anteriormente como apoyo fundamental del proyecto.

2.2 Recursos Hardware

A continuación, se especifican los recursos hardware utilizados para la realización de este proyecto, indicando así sus características.

2.2.1 Ordenador Portatil ASUS x556ua

- Procesador Intel Core i5-6200U (2.3 Ghz ,3MB)
- Memoria RAM 4GB DDR3L 1600 MHz
- Disco duro 500GB (5400rpm)
- Sistema operativo Microsoft Windows 10 64bits

2.2.2 Xiaomi MiBand2

- Pantalla de 0,42″ OLED.

- Registro de pasos, calorías, distancia, ritmo cardíaco y sueño.
- Resistencia al polvo y al agua IP67.
- Bluetooth 4.0 BLE.
- 70 mAh de batería (hasta 20 días de uso).
- Notificación de mensajes y llamadas por vibración.
- 7 gramos de peso.
- Medidas: 40,3×15,7×10,5 mm.
- Materiales: plástico y aluminio para el cuerpo y copoliéster termoplástico para la correa.

2.2.3 Telefono móvil Alcatel Pixi 4

- Sistema operativo Android 6.0
- Procesador MediaTek MTK6735M
- CPU Cortex A53 (4 núcleos)
- Memoria RAM 1 GB
- Memoria Interna 8 GB
- Bluetooth 4.0
- WiFi 802.1 b/g/n

2.3 Recursos Software

2.3.1 Android Studio

Android Studio es el entorno de desarrollo integrado oficial para la plataforma Android. Fue anunciado el 16 de mayo de 2013 en la conferencia Google I/O, y reemplazó a Eclipse como el IDE oficial para el desarrollo de aplicaciones para Android. La primera versión estable fue publicada en diciembre de 2014.

Está basado en el software IntelliJ IDEA de JetBrains y ha sido publicado de forma gratuita a través de la Licencia Apache 2.0. Está disponible para las plataformas Microsoft Windows, macOS y GNU/Linux. Ha sido diseñado específicamente para el desarrollo de Android.

2.3.2 MySQL

MySQL es un Sistema de gestión de bases de datos relacional desarrollado bajo licencia dual GPL/Licencia comercial por Oracle Corporation y está considerada como la base de datos open source más popular del mundo para entornos de desarrollo web.

MySQL es una base de datos muy rápida en la lectura cuando utiliza el motor no transaccional MyISAM, pero puede provocar problemas de integridad en entornos de alta concurrencia en la modificación. En aplicaciones web hay baja concurrencia en la modificación de datos y en cambio el entorno es intensivo en lectura de datos, lo que hace a MySQL ideal para este tipo de aplicaciones.

2.3.3 Xampp

XAMPP es un paquete de instalación independiente de plataforma, software libre, que consiste principalmente en el sistema de gestión de bases de datos MySQL, el servidor web Apache y los intérpretes para lenguajes de script: PHP y Perl. El nombre proviene del acrónimo de X (para cualquiera de los diferentes sistemas operativos), Apache, MariaDB, PHP, Perl. El programa se distribuye bajo la licencia GNU y actúa como un servidor web libre, fácil de usar y capaz de interpretar páginas dinámicas.

2.3.4 Advanced REST Client

Advanced Rest Client es una extensión de Chrome que nos permite lanzar peticiones a servicios o APIs RestFul. Puedes hacer cualquier tipo de petición además de las habituales GET, POST, PUT y DELETE. Permite pasar parámetros a las peticiones y muestra el resultado devuelto por el servicio que queremos probar.

En este proyecto, esta extensión se ha utilizado para comprobar datos del Context Broker.

2.3.5 VM VirtualBox

VM VirtualBox es un software de virtualización para arquitecturas x86/amd64, creado originalmente por la empresa alemana innotek GmbH. Actualmente es desarrollado por Oracle Corporation como parte de su familia de productos de virtualización.

Por medio de esta aplicación es posible instalar sistemas operativos adicionales, conocidos como «sistemas invitados», dentro de otro sistema operativo «anfitrión», cada uno con su propio ambiente virtual.

En este caso, hemos instalado el sistema operativo CentOS en su versión 6.7. CentOS es una distribución Linux enfocada en entornos empresariales y derivada de los paquetes fuentes libremente publicados por Red Hat.

El objetivo de su instalación era alojar en él el servicio Context Broker que hemos utilizado para la realización de este proyecto.

3 ESTIMACIÓN TEMPORAL

Nunca dejes para mañana lo que puedes hacer hoy

- Lord Chesterfield -

-

Al comienzo del proyecto, este se dividió en una serie de tareas que fueron identificadas y se les asignó un tiempo. También, se planificó la secuencia de ejecución de las mismas para intentar conseguir que el tiempo de desarrollo fuese mínimo.

En este apartado compararemos la estimación temporal al comienzo del proyecto con el tiempo real dedicado.

3.1 Planificación temporal estimada

Curso Android Studio	**41 horas**
Búsqueda e inscripción	1 horas
Realización del curso	40 horas

Tabla 1: Tiempo estimado curso Android

Obtención de información	**41 horas**
Comprensión del problema	1 horas
Estudio de los objetivos	2 horas
Obtención de los requisitos	8 horas
Obtención de información sobre las tecnologías a utilizar	30 horas

Tabla 2: Tiempo estimado obtención de información

Diseño	**35 horas**
Diseño Android	25 horas
Diseño página web	10 horas

Tabla 3: Tiempo estimado diseño

Implementación	**70 horas**
Implementación Android	50 horas
Implementación web	30 horas

Tabla 4: Tiempo estimado implementación

Testeo y pruebas	**33 horas**
Pruebas manuales	3 horas
Corrección de errores	30 horas

Tabla 5: Tiempo estimado testeo y pruebas

Documentación	**70 horas**
Redacción de la memoria	70 horas

Tabla 6: Tiempo estimado documentación

Horas totales	**300 horas**

Tabla 7: Horas totales estimadas

3.2 Planificación temporal real

Curso Android Studio	**61 horas**
Búsqueda e inscripción	1 horas
Realización del curso	60 horas

Tabla 8: Tiempo real curso Android

Obtención de información	**81 horas**
Comprensión del problema	2 horas
Estudio de los objetivos	2 horas
Obtención de los requisitos	7 horas
Obtención de información adicional Android	20 horas
Obtención de información Context Broker	30 horas
Obtención de información Java	20 horas

Tabla 9: Tiempo real obtención de información

Diseño	**65 horas**
Diseño Android	40 horas
Diseño página web	25 horas

Tabla 10: Tiempo real diseño

Implementación	**130 horas**
Implementación Android	90 horas
Implementación web	40 horas

Tabla 11: Tiempo real implementación

Testeo y pruebas	**42 horas**
Pruebas manuales	2 horas
Corrección de errores	40 horas

Tabla 12: Tiempo real testeo y pruebas

Documentación	**90 horas**
Redacción de la memoria	90 horas

Tabla 13: Tiempo real documentación

Horas totales	**499 horas**

Tabla 14: Horas totales reales

4 REQUISITOS DEL SISTEMA

"Nuestra recompensa se encuentra en el esfuerzo y no en el resultado. Un esfuerzo total es una victoria completa"

- Mahatma Gandhi -

En este capítulo vamos a realizar el proceso conocido como Ingeniería de requisitos. Los requisitos que veremos a continuación nos darán información de cómo se comportará el sistema en su totalidad y las restricciones existentes. Se incluye un conjunto de casos de uso, requisitos funcionales y requisitos no funcionales. Estos requisitos están dirigidos tanto al cliente como al equipo de desarrollo.

4.1 Índice con los actores

Codigo	Nombre de los Actores	Descripción
ACT-001	Usuario de aplicación	Encargado de usar la aplicación. Será la persona que lleve la Xiaomi MiBand 2.
ACT-002	Gestor de datos	Encargado de gestionar los datos. Será la persona que tenga acceso a los datos recogidos por las aplicaciones MFit
ACT-003	Context Broker	Base de datos donde se guardarán los datos recogidos de las aplicaciones MFit.
ACT-004	SmartBand	Dispositivo que generará los datos

Tabla 15: Índice con los actores

4.2 Requisitos generales

Esta sección contiene la especificación de los requisitos generales del sistema, también denominados características del sistema u objetivos del sistema.

RGen-01	**Detectar SmartBand MiBand2**
Versión	1.0
Descripción	La aplicación Android deberá detectar las SmartBand MiBand2 cercanas al dispositivo móvil mediante Bluetooth BLE.
Prioridad	Esencial

Tabla 16. Detectar SmartBand MiBand2

RGen-02	**Información del usuario**
Versión	1.0
Descripción	La aplicación Android deberá preguntar la información del usuario que está utilizando la aplicación cuando este registre una pulsera.
Prioridad	Esencial

Tabla 17. Información del usuario

RGen-04	**Conexión SmartBand MiBand2**
Versión	1.0
Descripción	La aplicación Android deberá ser capaz de realizar la conexión mediante BLE con la pulsera.
Prioridad	Esencial

Tabla 18. Conexión SmartBand MiBand2

RGen-05	**Mediciones**
Versión	1.0
Descripción	La aplicación Android deberá tener una opción de ver las mediciones actuales (frecuencia cardíaca y pasos diarios) manualmente.
Prioridad	Esencial

Tabla 19. Mediciones

RGen-06	**Monitorización sesión de entrenamiento**
Versión	1.0
Descripción	La aplicación Android deberá tener una opción de monitorización automática de medidas en una sesión de entrenamiento.
Prioridad	Esencial

Tabla 20. Monitorizacion sesión entrenamiento

RGen-07	**Exportación al Context Broker**
Versión	1.0
Descripción	La aplicación Android deberá exportar la información recogida de la pulsera, tanto de mediciones como de entrenamiento al ContextBroker
Prioridad	Esencial

Tabla 21. Exportacion al Context Broker

RGen-08	**Detectar nueva smartband conectada al sistema**
Versión	1.0
Descripción	La aplicación web deberá notificar la adición de una nueva pulsera a la aplicación.
Prioridad	Esencial

Tabla 22: Detecta nueva smartband conectada al sistema

RGen-09	**Acceso a los datos de la aplicación Android**
Versión	1.0
Descripción	La aplicación web deberá poder acceder a todos los datos de la aplicación Android
Prioridad	Esencial

Tabla 23. Acceso a los datos de la aplicación Android.

RGen-10	**Actualización de datos**
Versión	1.0

Descripción	La aplicación web deberá actualizar automáticamente los datos recogidos de la aplicación Android
Prioridad	Esencial

Tabla 24. Actualización de datos.

RGen-11	**Mostar visualmente los datos recogidos**
Versión	1.0
Descripción	La aplicación web deberá mostrar visualmente los datos recogidos de la aplicación Android.
Prioridad	Esencial

Tabla 25. Mostrar visualmente los datos recogidos.

4.3 Casos de uso

En esta sección encontraremos la especificación de los casos de uso del sistema, incluyendo los correspondientes diagramas y la especificación de los propios casos de uso. Los casos de uso describen como se utilizará el sistema a desarrollar por sus futuros usuarios.

- Especificación

CaU-001	**Detectar MiBand2**
Precondición	El Bluetooth del dispositivo debe estar activado
Descripción	El sistema deberá detectar todas las Xiaomi MiBand 2 que estén a su alcance.
Postcondición	Después de detectar la pulsera, debemos emparejarla.

Tabla 26: Detectar MiBand2

CaU-002	**Emparejar MiBand2**
Precondición	El sistema debe haber detectado la pulsera
Dependencias	• CaU-001: Detectar MiBand2
Descripción	Una vez detectada la pulsera, estableceremos el emparejamiento.
Postcondición	Cuando la pulsera vibre debemos pulsarla para un correcto emparejamiento.

Tabla 27: Emparejar MiBand2

CaU-003	Recoger datos de frecuencia cardiaca
Precondición	La pulsera debe estar emparejada y activa.
Descripción	El sistema deberá recoger la frecuencia cardiaca actual y mostrarlo en la aplicación.
Dependencias	• CaU-001: Detectar MiBand2 • CaU-002: Emparejar MiBand2
Postcondición	No procede

Tabla 28: Recoger datos de frecuencia cardiaca

CaU-004	Recoger pasos diarios
Precondición	La pulsera debe estar emparejada y activa.
Descripción	El sistema deberá recoger los pasos actuales y mostrarlo en la aplicación.
Dependencias	• CaU-001: Detectar MiBand2 • CaU-002: Emparejar MiBand2
Postcondición	No procede

Tabla 29: Recoger pasos diarios

CaU-005	Activar monitorización entrenamiento personal
Precondición	• La pulsera debe estar emparejada y activa. • Los datos personales deben de estar completamente rellenos.
Descripción	El sistema deberá monitorizar un entrenamiento personal: 1. Se activará el cronómetro 2. Cada 20 segundos se realizarán mediciones de pulso y pasos 3. Con esos datos, calcularemos las calorias quemadas hasta ese momento.
Dependencias	• CaU-001: Detectar MiBand2 • CaU-002: Emparejar MiBand2
Postcondición	No procede

Tabla 30: Activar monitorización entrenamiento personal

CaU-006	**Desactivar monitorización entrenamiento personal**
Precondición	• La pulsera debe estar emparejada y activa. • Los datos personales deben de estar completamente rellenos.
Descripción	1. Se desactivará el cronómetro 2. Se dejarán de tomar mediciones 3. Se borrarán los datos anteriormente calculados.
Dependencias	• CaU-001: Detectar MiBand2 • CaU-002: Emparejar MiBand2
Postcondición	No procede

Tabla 31: Desactivar monitorización entrenamiento personal

CaU-007	**Mostrar gráfica de frecuencia cardiaca**
Precondición	• La pulsera debe estar emparejada y activa. • Los datos personales deben de estar completamente rellenos. • La sesión de entrenamiento debe estar activada
Descripción	El sistema debe mostrar una gráfica con los datos de frecuencia cardiaca tomados durante el entrenamiento.
Dependencias	• CaU-001: Detectar MiBand2 • CaU-002: Emparejar MiBand2 • CaU-005: Activar monitorización de entrenamiento
Postcondición	No procede

Tabla 32: Mostrar gráfica de frecuencia cardiaca

CaU-008	**Guardar datos en el Context Broker**
Precondición	• La pulsera debe estar emparejada y activa. • Los datos personales deben de estar completamente rellenos. • La sesión de entrenamiento debe estar activada o haber tomado mediciones manuales
Descripción	El sistema deberá guardar todos los datos tomados en el Context Broker, tanto de las mediciones manuales como los datos del entrenamiento.
Dependencias	• CaU-001: Detectar MiBand2

	• CaU-002: Emparejar MiBand2 • CaU-005: Activar monitorización de entrenamiento • CaU-003: Recoger datos de frecuencia cardiaca • CaU-004: Recoger pasos diarios
Postcondición	No procede

Tabla 33: Guardar datos en el Context Broker

CaU-009	**Recoger datos del context bróker**
Precondición	• Se deben haber medido datos con anterioridad desde la aplicación Android • El dispositivo que presente la aplicación web debe tener conexión a Internet.
Descripción	El sistema deberá recoger los datos del context bróker para así mostrarlos en la aplicación web
Dependencias	
Postcondición	No procede

Tabla 34: Recoger datos del context bróker

CaU-010	**Visualizar datos recogidos**
Precondición	• Se deben haber medido datos con anterioridad desde la aplicación Android • El dispositivo que presente la aplicación web debe tener conexión a Internet.
Descripción	El sistema deberá visualizar los datos que se recojan del context broker
Dependencias	• CaU-009. Recoger datos del context bróker
Postcondición	No procede

Tabla 35: Visualizar datos recogidos

- Diagramas

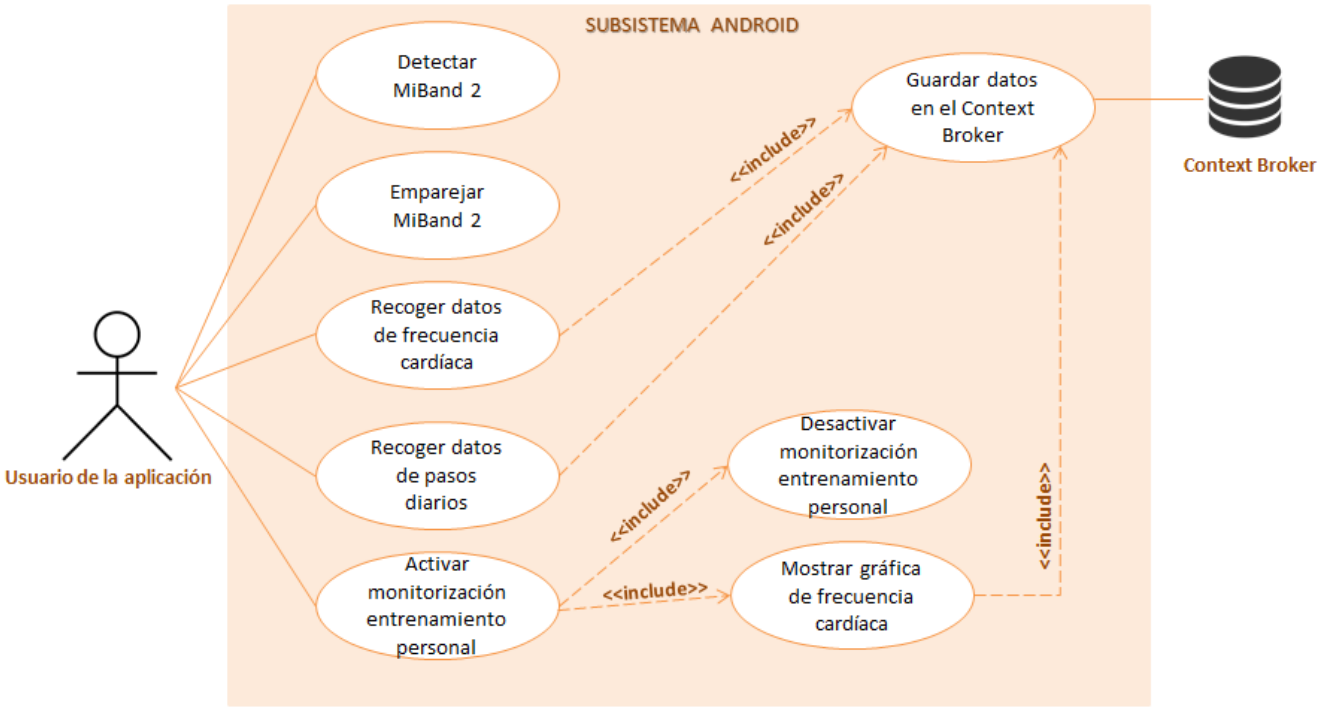


Ilustración 4: Casos de uso subsistema Android

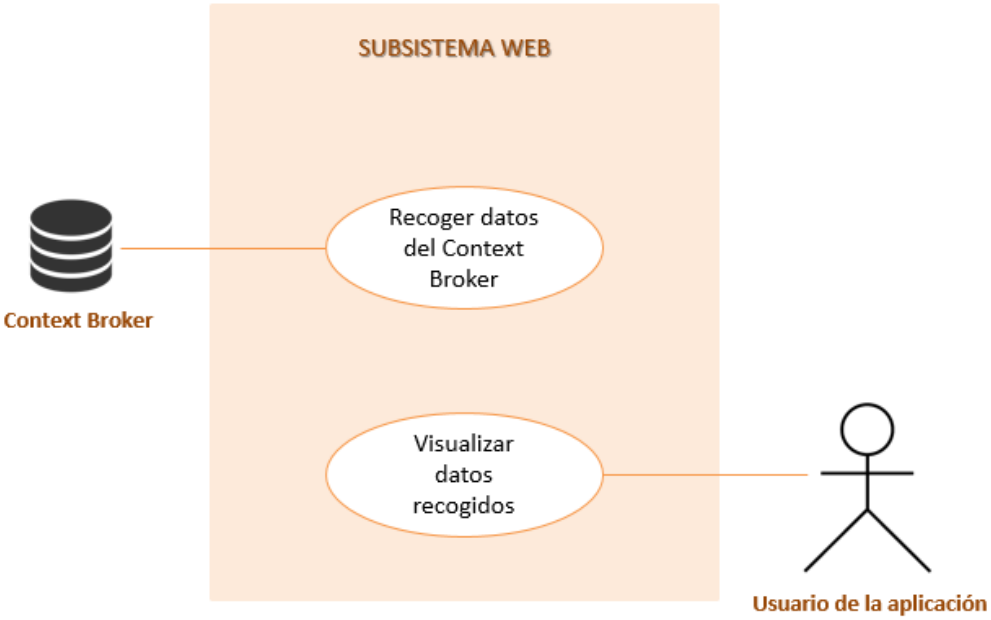


Ilustración 5: Casos de uso subsistema Web

4.3.1 Requisitos funcionales

En este apartado encontraremos los requisitos funcionales que se han identificado a partir de los requisitos generales y de los casos de uso.

4.3.2 Requisitos funcionales de información

Los requisitos de información deben contener información que debe almacenar el sistema para poder ofrecer la funcionalidad descrita en los casos de uso del sistema o en otros requisitos.

RFInfo-01	**Estructura base de datos local de aplicación Android**
Versión	1.0
Descripción	La aplicación Android debe disponer de una base de datos local en la que se recojan los datos medidos durante un día.
Prioridad	Esencial

Tabla 36. Estructura Base de Datos Local

RFInfo-02	**Información del usuario**
Versión	1.0
Descripción	El sistema dispondrá de la siguiente información relativa al usuario: • Nombre • Género • Peso • Edad • Objetivo de pasos al día
Prioridad	Esencial

Tabla 37. Información del usuario

RFInfo-03	**Estructura base de datos de la aplicación web**
Versión	1.0
Descripción	La base de datos hará uso de cuatro tablas donde se recogerá información relativa a: - Información de usuarios - Mediciones - Entrenamiento - Histórico de mediciones

Prioridad	Esencial

Tabla 38. Estructura Base de Datos de la aplicación web

4.3.3 Requisitos funcionales de Reglas de Negocio

Los requisitos de reglas de negocio especifican qué reglas de negocio debe respetar el sistema, evitando que se incumpla durante su funcionamiento.

RFRNego-01	**El usuario debe actuar manualmente antes de recoger el pulso en la aplicación Android**
Versión	1.0
Descripción	Antes de poder automatizar la monitorización del pulso, el usuario deberá tomar el pulso desde la pulsera manualmente.
Prioridad	Esencial

Tabla 39. El usuario debe actuar manualmente antes de recoger el pulso en la aplicación Android

RFRNego-02	**Dos usuarios no podrán conectarse a la misma pulsera simultáneamente.**
Versión	1.0
Descripción	El sistema deberá respetar que dos usuarios no se conecten a la misma pulsera a la misma vez
Prioridad	Esencial

Tabla 40. Dos usuarios no podrán conectarse a la misma pulsera simultáneamente

RFRNego-03	**El usuario no debe actualizar la página web para que se recarguen los datos.**
Versión	1.0
Descripción	El sistema se comporta de tal forma que los datos se recargarán automáticamente.
Prioridad	Esencial

Tabla 41. El usuario no debe actualizar la página web para que se recargen los datos.

RFRNego-04	**La aplicación web podrá visualizarse desde varias pantallas simultáneamente.**

Versión	1.0
Descripción	El sistema podrá soportar la visualización de datos desde varias pantallas simualtáneamente.
Prioridad	Esencial

Tabla 42. La aplicación web podrá visualizarse desde varias pantallas simultáneamente.

4.3.4 Requisitos funcionales de conducta

En los requisitos de conducta se incluyen aquellos comportamientos que no se hayan especificado mediante los casos de uso del sistema.

RFCond-01	**La aplicación Android deberá mostrar dos opciones de actuación**
Versión	1.0
Descripción	El sistema deberá mostrar dos opciones de actuación: - Opción de medir datos actuales - Opción de entrenamiento en el cual monitorizamos nuestros datos durante un cierto tiempo
Prioridad	Esencial

Tabla 43 : La aplicación Android deberá mostrar dos opciones de actuación

RFCond-02	**La aplicación web deberá mostrar los datos de tantas opciones de actuación como tenga la aplicación Android**
Versión	1.0
Descripción	La aplicación web deberá mostrar los datos recogidos de todas las opciones de actuación que tenga la aplicación Android.
Prioridad	Esencial

Tabla 44: La aplicación web deberá mostrar los datos de tantas opciones de actuación como tenga la aplicación Android

RFCond-03	**La aplicación web deberá generar un gráfico con las mediciones de pulso realizadas durante un día**
Versión	1.0
Descripción	La aplicación web deberá generar un gráfico con las mediciones del pulso realizadas durante un día

Prioridad	Esencial

Tabla 45: La aplicación web deberá generar un gráfico con las mediciones de pulso realizadas durante un día

4.4 Requisitos no funcionales

4.4.1 Requisitos no funcionales de fiabilidad

Los requisitos de fiabilidad establecen los niveles que debe cumplir el sistema a desarrollar en aspectos como recuperabilidad y tolerancia a fallos.

RNFFiab-01	**El sistema deberá tardar un máximo de 10 minutos para la recuperación tras una caída.**
Versión	1.0
Descripción	El sistema deberá iniciarse en un periodo de 10 minutos tras una caída, ya sea por razones externas (corte de la corriente eléctrica) o internas (desconexión por error humano)
Prioridad	Esencial

Tabla 46: El sistema deberá tardar un máximo de 10 minutos para la recuperación tras una caída.

RNFFiab-02	**El sistema no podrá estar inaccesible más de 1 hora al mes**
Versión	1.0
Descripción	El sistema deberá estar accesible 24h, con un máximo de caídas por motivos de software de 1h al mes.
Prioridad	Esencial

Tabla 47: El sistema no podrá estar inaccesible más de 1 hora al mes

RNFFiab-03	**Los datos recogidos por la pulsera no se podrán modificar**
Versión	1.0
Descripción	El sistema deberá impedir las modificaciones en los datos recogidos por la pulsera
Prioridad	Esencial

Tabla 48: Los datos recogidos por la pulsera no se podrán modificar

4.4.2 Requisitos no funcionales de usabilidad

Los requisitos de usabilidad establecen los niveles que debe cumplir el sistema a desarrollar en aspectos como facilidad de aprendizaje, comprensión, operatividad y atractividad.

RNFUsab-01	**El sistema deberá permitir en el 75% de las veces que con un máximo de 5 clicks sea suficiente para llegar a la información deseada**
Versión	1.0
Descripción	La totalidad del sistema deberá estar disponible con un máximo de 5 clicks desde la pantalla de inicio
Prioridad	Esencial

Tabla 49: El sistema deberá permitir en el 75% de las veces que con un máximo de 5 clicks sea suficiente para llegar a la información deseada

RNFUsab-01	**El sistema solo estará disponible en español**
Versión	1.0
Descripción	El único idioma disponible será el español.
Prioridad	Esencial

Tabla 50: El sistema sólo estará disponible en español

4.4.3 Requisitos no funcionales de eficiencia

Los requisitos de eficiencia establecen los niveles que debe cumplir el sistema a desarrollar en aspectos como tiempo de respuesta.

RNFEfic-01	**El sistema deberá tener un tiempo de respuesta inferior a 5 segundos**
Versión	1.0
Descripción	Una vez clicada una herramienta concreta, el sistema deberá ser capaz de cambiar de interfaz en un máximo de 5 segundos
Prioridad	Esencial

Tabla 51: El sistema deberá tener un tiempo de respuesta inferior a 5 segundos

4.4.4 Requisitos no funcionales de mantenibilidad

Los requisitos de mantenibilidad establecen los niveles que debe cumplir el sistema a desarrollar en aspectos como estabilidad, facilidad de análisis, facilidad de cambio y facilidad de pruebas.

RMant-01	**El código fuente que se implemente deberá cumplir las recomendaciones de la IEEE**
Versión	1.0
Descripción	El código que se implemente deberá adaptarse a los estándares con el fin de facilitar su entendimiento a los encargados de mantener el código
Prioridad	Esencial

Tabla 52: El código fuente que se implemente deberá cumplir las recomendaciones de la IEEE

4.4.5 Requisitos no funcionales de portabilidad

Los requisitos de portabilidad establecen los niveles que debe cumplir el sistema a desarrollar en aspectos relacionados con la escalabilidad.

RPorta-01	**La aplicación deberá evitar el uso de software propietario**
Versión	1.0
Descripción	Con el fin de evitar un exceso de presupuesto, el sistema deberá apoyarse en software libre cuando no pueda implementar alguna herramienta
Prioridad	Esencial

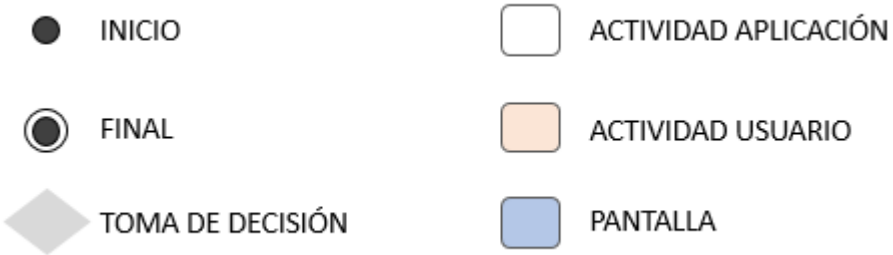
Tabla 53: La aplicación deberá evitar el uso de software propietario

4.5 Diagramas de actividad

Un diagrama de actividad se trata de otro tipo de diagrama UML que tiene como finalidad representar gráficamente un algoritmo o proceso. De esta forma, el desarrollador se asegura de haber realizado un análisis cuidado de lo que debe implementar, consiguiendo así que la tarea sea mucho más sencilla.

Durante la elaboración del diagrama, se debe identificar todas las actividades, tanto principales como secundarias, que están involucradas en la ejecución de un procedimiento y en el orden en el que operan. Además, no debe tenerse en cuenta sólo el escenario ideal, sino que se debe indentificar los puntos de decisión y los diferentes resultados que se pueden obtener según las condiciones.

A continuación, se muestra los elementos que intervienen en los diagramas de actividad de este proyecto



4.5.1 Diagramas de actividad para aplicación Android

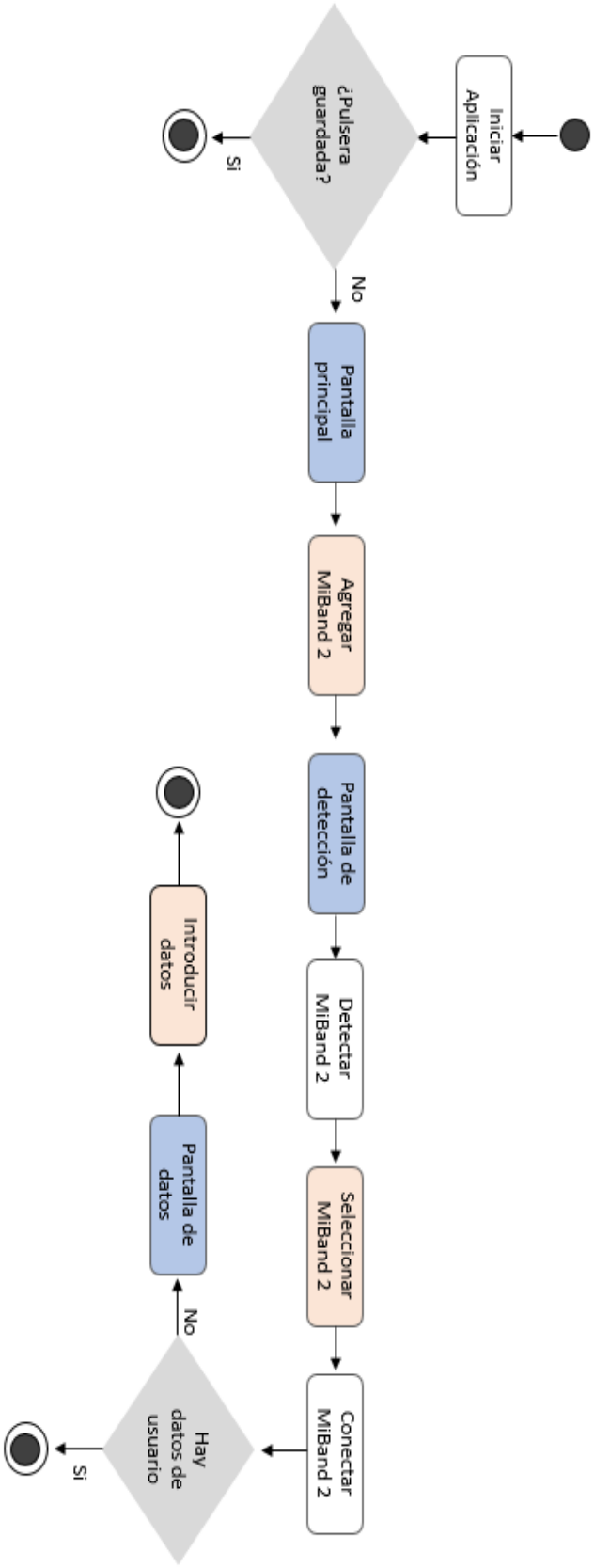


Ilustración 7: Diagrama de actividad inicio

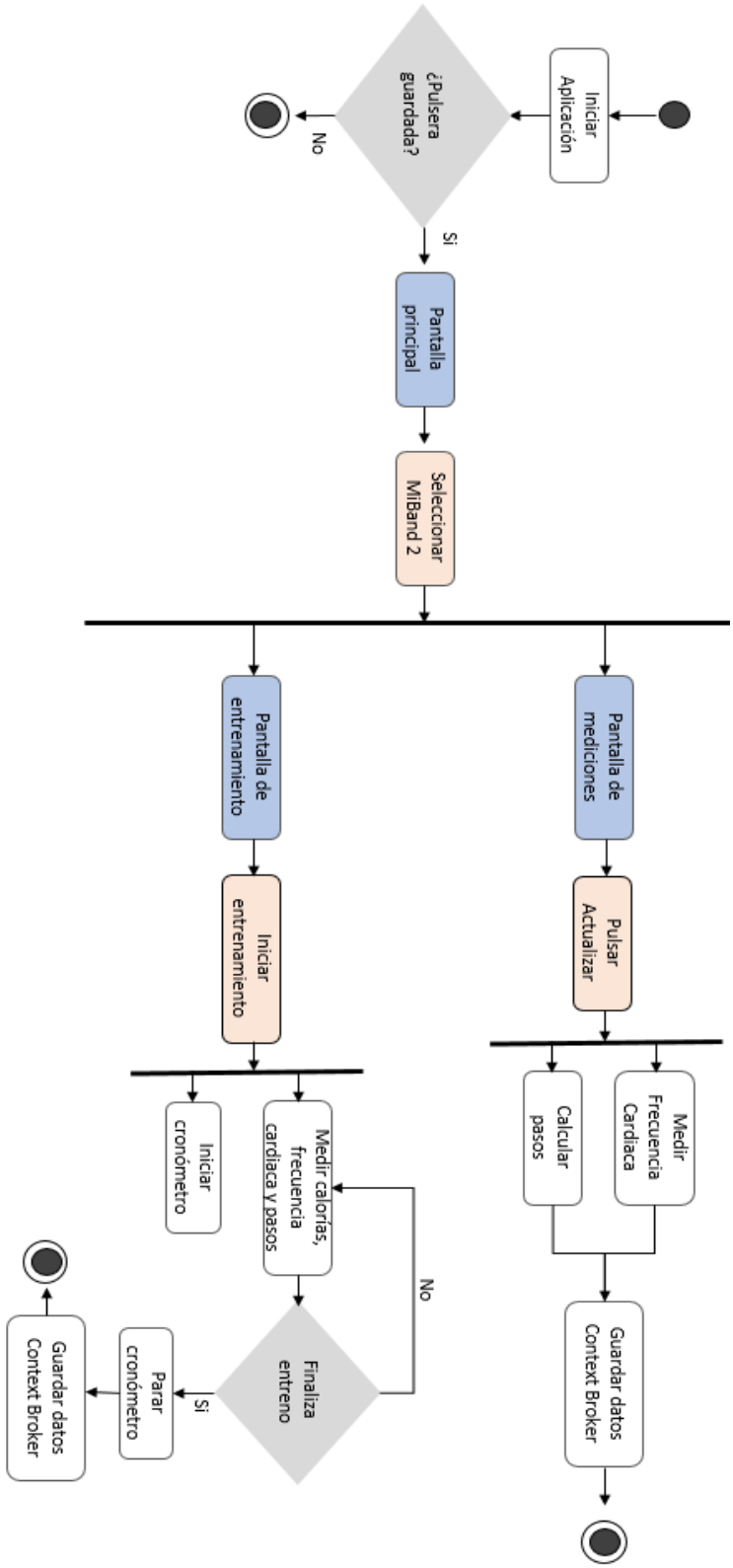


Ilustración 6: Diagrama de actividad opciones

4.5.2 Diagramas de actividad para aplicación Web

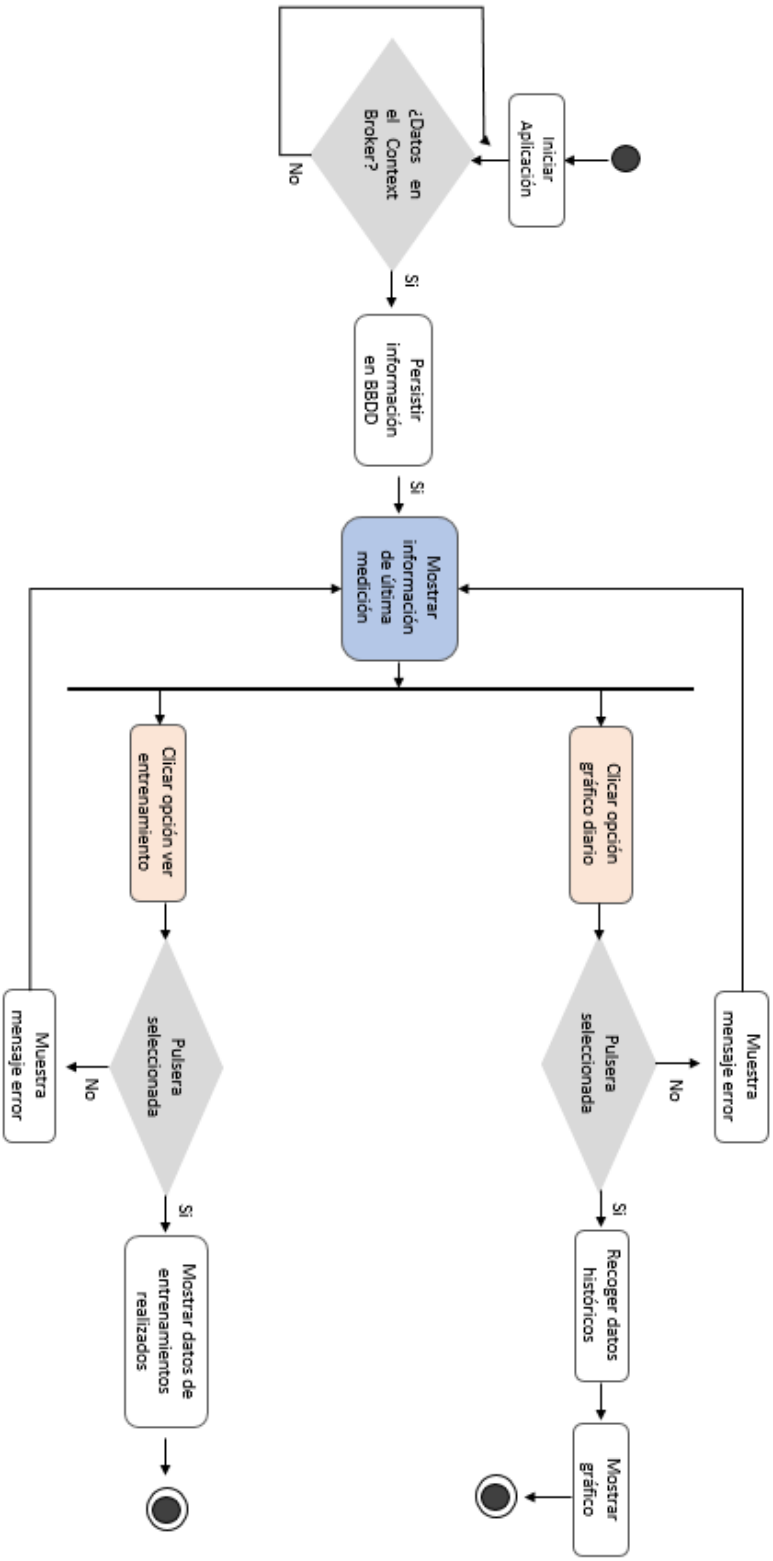


Ilustración 8: Diagrama de actividad aplicación Web

4.6 Prototipos

Por último, atendiéndose a los casos de usos y las especificaciones anteriormente mencionados, es aconsejable realizar un prototipo de las aplicaciones.

4.6.1 Prototipos de la aplicación Android

A continuación, se muestran los prototipos de las pantallas de la aplicación.

Ilustración 9: Prototipo Android (I)

Ilustración 10: Prototipo Android (II)

Según el prototipo, la pantalla principal de la aplicación sería similar a la Ilustración 9, donde tenemos tres opciones; agregar una nueva pulsera, visualizar una pulsera ya conectada y abrir las opciones de menú.

- Al añadir una nueva pulserá, navegaríamos hasta la Ilustración 10 donde al comenzar la detección nos irá apareciendo los dispositivos detectados.
- Al abrir el menú, podremos observar tres opciones; Ajustes, Sobre ti y Salir. En ajustes tendríamos un acceso directo a las Notificaciones de la aplicación y en Sobre ti tendríamos información relativa al usuario que está utilizando la pulsera.
- Por útlimo, al clicar sobre el dispositivo conectado en la pantalla principal navegaríamos hacia la pantalla que observamos en la Ilustración XX. Tendriamos dos opciones:
    - Mediciones: opción para visualizar nuestros datos actuales en la aplicación.
    - Entrenamiento: opción para monitorizar nuestro entrenamiento durante un tiempo.

Ilustración 14: Prototipo Android (III)

Ilustración 13: Prototipo Android (IV)

Ilustración 12: Prototipo Android (V)

Ilustración 11: Prototipo Android (VI)

Ilustración 15: Prototipo Android (VII)

4.6.2 Prototipos de la aplicación Web

A continuación, se muestra el prototipo de la aplicación web.

Ilustración 16: Prototipo Web (I)

Siempre que tengamos acceso a Internet desde el dispositivo donde se acceda a la aplicación web, la pantalla mostrada anteriormente será la visualizada cuando accedamos a la web. Según el prototipo, en la tabla inicial alojaremos información acerca del cliente y de las últimas mediciones realizadas con la aplicación, donde a través de iconos incluidos en la tabla podremos navegar hacia las siguientes opciones.

Si pulsamos sobre el icono incluido en la celda del campo Pulso se añadirá un gráfico a la pantalla con la información del pulso recogido durante un día.

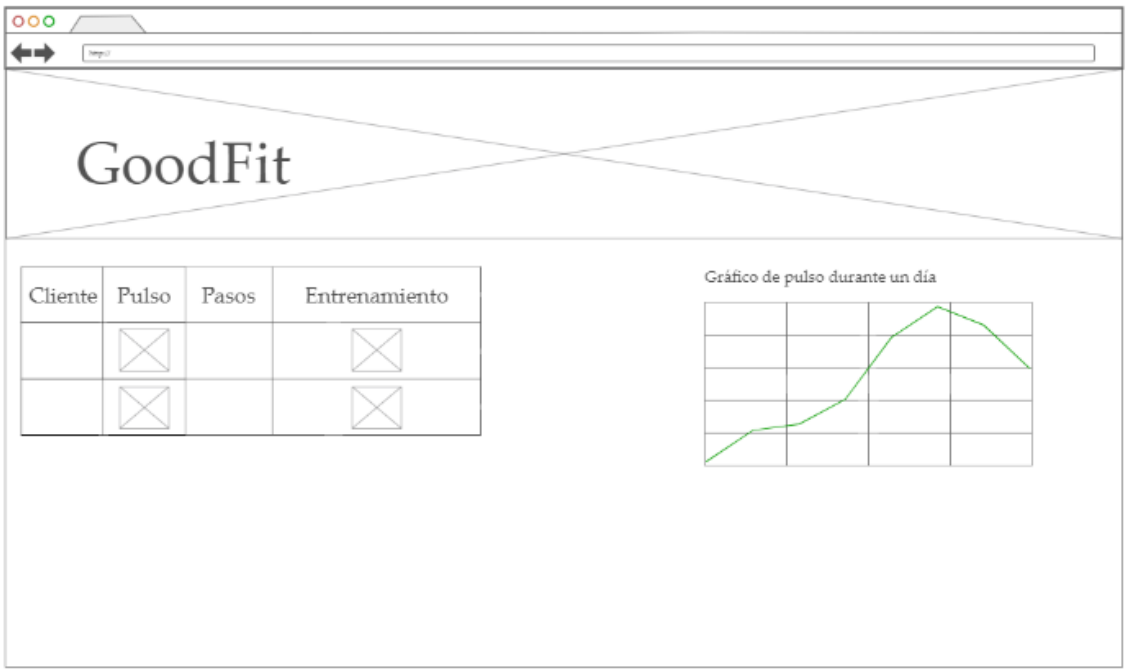


Ilustración 17: Prototipo Web (II)

Si, por el contrario, pulsamos en el icono incluido en la celda del campo Entrenamiento la aplicación muestra una tabla con los entrenamientos realizados por el usuario e información relevante de estos.

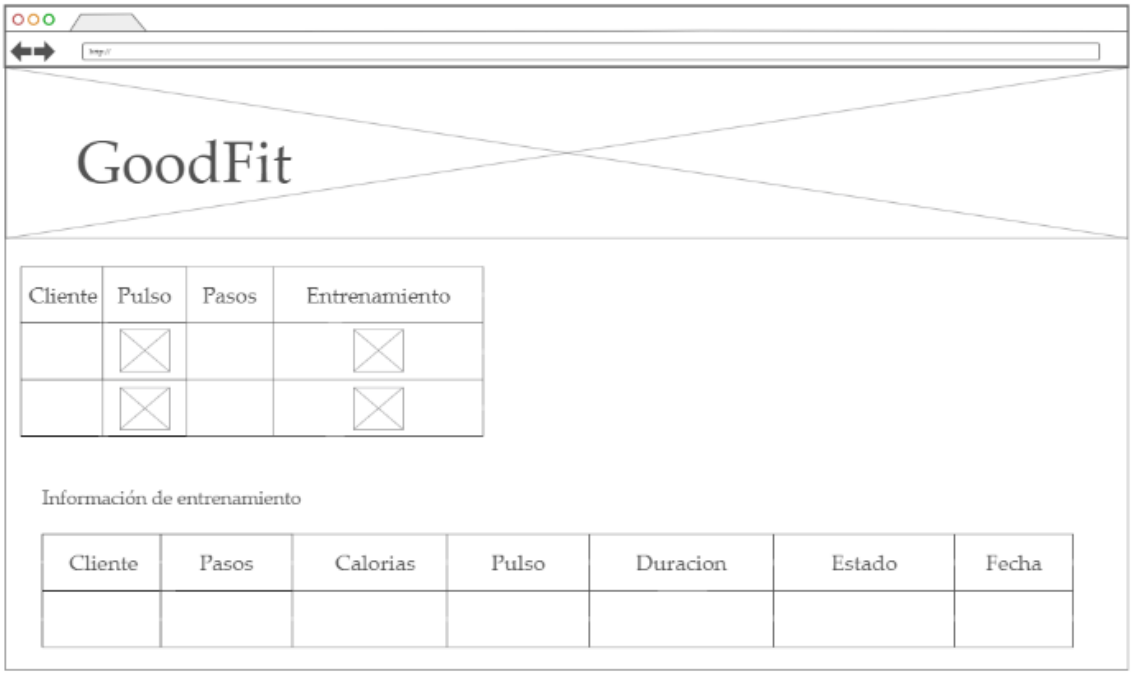


Ilustración 18: Prototipo Web (III)

5 APLICACIÓN ANDROID

El gran enemigo del conocimiento no es la ignorancia,
sino la ilusión de saber.

- Stephen Hawking-

En este capítulo podemos encontrar los conceptos estudiados para poder desarrollar correctamente la aplicación Android. Además, visualizaremos las distintas pantallas de la aplicación y como navegar entre ellas, así como distintas partes de código que son claves para el correcto funcionamiento de la aplicación.

5.1 Android

Android es un sistema operativo basado en el núcleo Linux. Fue diseñado principalmente para dispositivos móviles con pantalla táctil, como teléfonos inteligentes, tablets y también para relojes inteligentes, televisores y automóviles.

El éxito del sistema operativo se ha convertido en objeto de litigios sobre patentes en el marco de las llamadas «Guerras por patentes de teléfonos inteligentes» entre las empresas de tecnología.

Algunas características destacables de este sistema operativo son las siguientes:

- La plataforma es adaptable a pantallas de mayor resolución, VGA, biblioteca de gráficos 2D, biblioteca de gráficos 3D basada en las especificaciones de la OpenGL ES 2.0 y diseño de teléfonos tradicionales.

- Hace uso de SQLite, una base de datos liviana, que es usada para propósitos de almacenamiento de datos.

- Android soporta las siguientes tecnologías de conectividad: GSM/EDGE, IDEN, CDMA, EV-DO, UMTS, Bluetooth, Wi-Fi, LTE, HSDPA, HSPA+, NFC y WiMAX, GPRS, UMTS y HSDPA+ .

5.2 Bluetooth Low Energy

Bluetooth de baja energía (Bluetooth Low Energy, BLE) es una nueva tecnología digital de radio interoperable para pequeños dispositivos desarrollada por Bluetooth.

Este tipo de tecnología permite la comunicación entre dispositivos de pila de botón y dispositivos Bluetooth que opera en 2.4 GHz, con una tasa de transferencia de 1 Mbps en la capa física. Está basado en un microchip de bajo costo con opciones más amplias para su empleo en la industria. Tiene soporte para seguridad, ya que emplea el sistema de cifrados AES y esquemas de seguridad configurables.

El nuevo sistema bluetooth 4.0 viene incorporado en los smartphones a partir del 2013. Con esta tecnología Bluetooth ha mejorado el radio de transferencia (hasta 100 metros) y la velocidad de transferencia.

Bluetooth LE también es la pieza clave del Internet de las Cosas, objetos cotidianos como los electrodomésticos, conectados permanentemente a Internet o directamente entre si, porque esta nueva tecnología consume tan poco que no tenemos que recargar el emisor o receptor de **Bluetooth LE** en muchos meses.

5.2.1 Integración BLE-Android

Android 4.3 (API 18) introduce soporte para Bluetooth Low Energy y proporciona funciones que las aplicaciones podrán utilizar para descubrir dispositivos BLE, consultar atributos de estos dispositivos y transmitir información.

A continuación, se describen los principales términos y conceptos de BLE:

- **Generic Attribute Profile (GATT)**. El perfil GATT es una especificación general para enviar y recibir fragmentos cortos de datos conocidos como "atributos" sobre un enlace BLE (entre un servidor y un cliente). Un perfil es una especificación de cómo funciona un dispositivo en una aplicación particular.
- **Attribute Protocol (ATT).** ATT está optimizado para funcionar en dispositivos BLE. Cada atributo se identifica de forma única por un identificador universalmente único (UUID), que es un formato estandarizado de 120 bits para un ID de cadena utilizado para identificar de forma exclusiva la información. Los atributos transportados por ATT son formateados como características y servicios.
- **Characteristic.** Una característica contiene un valor único y n descriptores que describen el valor de la característica.
- **Descriptor.** Los descriptores son atributos definidos que describen un valor característico.
- **Service.** Un servicio es una colección de características.

  Un perfil es una colección de uno o múltiples servicios. Cada servicio puede contener una o multiples características [6]

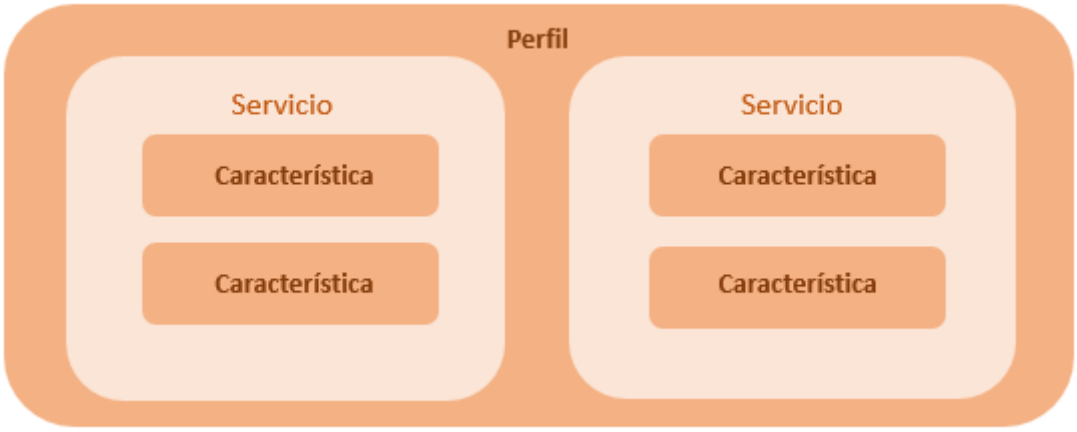


Ilustración 19: Esquema Bluetooth Low Enegery

5.3 Desarrollo de la aplicación Android.

Después de definir de la forma más completa posible tanto el comportamiento de la aplicación como sus objetivos y requisitos, en este apartado encontraremos el traslado de toda esa información a código.

5.3.1 Permisos

Nada más comenzar el desarrollo de la aplicación será necesario establecer los permisos necesarios en el fichero *AndroidManifest.xml*.

```
<uses-permission android:name="android.permission.INTERNET" />
<uses-permission android:name="android.permission.BLUETOOTH" />
<uses-permission android:name="android.permission.BLUETOOTH_ADMIN" />
```

Además, a partir de una determinada API de Android, no sería necesario solo con establecer los permisos, sino que es obligatorio hacer saber al usuario que la aplicación está requiriendo esos permisos. Por tanto, tendremos que incluir el siguiente código [7] [8].

```
if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.M) {
    checkAndRequestPermissions();
}

@TargetApi(Build.VERSION_CODES.M)
private void checkAndRequestPermissions() {
    List<String> wantedPermissions = new ArrayList<>();

    if (ContextCompat.checkSelfPermission(this,
Manifest.permission.BLUETOOTH) == PackageManager.PERMISSION_DENIED)
        wantedPermissions.add(Manifest.permission.BLUETOOTH);
    if (ContextCompat.checkSelfPermission(this,
Manifest.permission.BLUETOOTH_ADMIN) == PackageManager.PERMISSION_DENIED)
        wantedPermissions.add(Manifest.permission.BLUETOOTH_ADMIN);
    if (ContextCompat.checkSelfPermission(this,
```

```
    if (!wantedPermissions.isEmpty())
        ActivityCompat.requestPermissions(this, wantedPermissions.toArray(new
String[wantedPermissions.size()]), 0);
}
```

5.3.2 Conexión SmartBand Xioami MiBand 2

El desafío más importante al que nos sometíamos con la realización del proyecto es la conexión entre la *smartband* y el *smartphone* y la lectura correcta de los datos de los sensores en tiempo real.

5.3.2.1 Detectar SmartBand

La primera tarea que se debe realizar para iniciar una comunicación entre dispositivos BLE consiste en ejecutar un análisis (scan) para recuperar la referencia del dispositivo BLE correspondiente. Esta operación está asegurada por el objeto BluetoothAdapter, sin embargo, el procedimiento difiere en función de si la aplicación está destinada a un terminal que ejecuta la versión 21 de Android (Android Lollipop) o inferior (a partir de la versión de API 18, Jelly Bean).

Para las versiones de Android inferiores a Android 21, es directamente la instancia de BluetoothAdapter la que lanza el *scan*, invocando el método *startLeScan()*. Este método devolverá un valor booleano que indica si el scan se ha realizado con éxito o no. El parámetro de tipo BluetoothAdapter.LeScanCallback es una interfaz en la que la instancia proporcionada se invocará como retorno, cuando se haya detectado algún objeto BLE desde el adaptador. Para detener el análisis, invocaremos al método *stopLeScan()*.

A partir de la API 21, la búsqueda de objetos BLE se realiza mediante una instancia de la clase BluetoothLeScanner. Esta clase expone varias versiones del método *startScan()*, método que inicia la búsqueda de objetos BLE. El parámetro callback que incluimos dentro de la llamada al método *startScan()* es una instancia de ScanCallback, clase abstracta cuyo método onScanResult se invocará cuando se detecte algún objeto BLE.

```
public void startScan(List<ScanFilter> filters, ScanSettings settings,
        final ScanCallback callback) {
    startScan(filters, settings, null, callback, null);
}
```

Como podemos observar el método *startScan()* también permite especificar ciertos filtros para los objetos BLE detectados, así como parámetros suplementarios para la búsqueda.

El objeto de tipo ScanResult incluirá el conjunto de información relativa al resultado de la búsqueda. Entre los métodos que presenta se incluyen *getDevice()* y *getRssi()* [8] [7].

```
private ScanCallback getScanCallback() {
    if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.LOLLIPOP) {
        newLeScanCallback = new ScanCallback() {
            @TargetApi(Build.VERSION_CODES.LOLLIPOP)
            @Override
            public void onScanResult(int callbackType, ScanResult result) {
                super.onScanResult(callbackType, result);
                try {
                    ScanRecord scanRecord = result.getScanRecord();
```

```
                    ParcelUuid[] uuids = null;
                    if (scanRecord != null) {
                        //logMessageContent(scanRecord.getBytes());
                        List<ParcelUuid> serviceUuids =
scanRecord.getServiceUuids();
                        if (serviceUuids != null) {
                            uuids = serviceUuids.toArray(new ParcelUuid[0]);
                        }
                    }
                    LOG.warn(result.getDevice().getName() + ": " +
                            ((scanRecord != null) ?
scanRecord.getBytes().length : -1));
                    handleDeviceFound(result.getDevice(), (short)
result.getRssi(), uuids);
                } catch (NullPointerException e) {
                    LOG.warn("Error handling scan result", e);
                }
            }
        };
    }
    return newLeScanCallback;
}
```

5.3.2.2 Emparejar SmartBand

Para emparejar (*bond*) la smartBand deseada con el smartphone sólo tendremos que hacer uso de la instancia *BluetoothDevice* invocando el método *createBond()* el cual retornará un booleano que indica si el *bond* se ha realizado con éxito o no [8] [7].

```
@RequiresPermission(Manifest.permission.BLUETOOTH_ADMIN)
public boolean createBond() {
    if (sService == null) {
        Log.e(TAG, "BT not enabled. Cannot create bond to Remote Device");
        return false;
    }
    try {
        Log.i(TAG, "createBond() for device " + getAddress() +
                " called by pid: " + Process.myPid() +
                " tid: " + Process.myTid());
        return sService.createBond(this, TRANSPORT_AUTO);
    } catch (RemoteException e) {Log.e(TAG, "", e);}
    return false;
}
```

5.3.2.3 Intercambio de información/interacciones

Las interacciones entre un objeto BLE y un terminal Android se materializan mediante operaciones de lectura y escritura. Estas operaciones se basan ambas en el mismo esquema: la lectura (o escritura) se realiza mediante la instancia de BluetoothGatt, que representa el objeto BLE y el resultado se gestionará con una llamada de retorno mediante la instancia de BluetoothGattCallback.

Los datos expuestos por el objeto BLE, como se explicó anteriormente, se denominan **Characteristic**. Cada característica se identifica mediante un identificador único universal (UUID) atribuido por el consorcio responsable de la norma BLE.

Las características se agrupan por servicio, un servicio puede contener una o varias características. Cada servicio

se identifica a su vez con un UUID. Para acceder a una característica, previamente hay que obtener una referencia sobre el servicio correspondiente [8] [7].

5.3.2.3.1 Descubrir los servicios

Para poder acceder a un servicio, es necesario comprobar su disponibilidad sobre el objeto BLE. Para ello, la clase BluetoothGatt provee el método *discoverServices()*, cuya función es "descubrir" todos los servicios disponibles en el objeto BLE conectado.

Cuando se descubren los servicios, la lista de servicios disponibles está accesible mediante el método *getServices()* de la clase BluetoothGatt.

Para conocer el identificador único de un servicio concreto, hay que hacer referencia a la tabla provista por el consorcio encargado de la norma Bluetooth y seguir esta regla:

- El identificador único es una combinación del número de serie asignado por el consorcio (en 4 bytes) y el identificador básico (fijado por la norma) 0000xxxx-1000-8000-00805f9b34fb; las xxxx se reemplazan por el número de servicio.

Para leer una característica la clase BluetoothGatt expone el método *readCharacteristic()* [8] [7].

En nuestro caso, hemos usado una instancia de MiBand2Service con todos los servicios de la smartband MiBand 2 definidos. A continuación, tenemos varios ejemplos:

```
public static final UUID UUID_SERVICE_MIBAND_SERVICE =
UUID.fromString(String.format(BASE_UUID, "FEE0"));

public static final UUID UUID_SERVICE_MIBAND2_SERVICE =
UUID.fromString(String.format(BASE_UUID, "FEE1"));

public static final UUID UUID_SERVICE_HEART_RATE =
UUID.fromString(String.format(BASE_UUID, "180D"));

public static final UUID UUID_SERVICE_FIRMWARE_SERVICE =
UUID.fromString("00001530-0000-3512-2118-0009af100700");

public static final UUID UUID_CHARACTERISTIC_FIRMWARE =
UUID.fromString("00001531-0000-3512-2118-0009af100700");

public static final UUID UUID_CHARACTERISTIC_FIRMWARE_DATA =
UUID.fromString("00001532-0000-3512-2118-0009af100700");

public static final UUID UUID_CHARACTERISTIC_5_ACTIVITY_DATA =
UUID.fromString("00000005-0000-3512-2118-0009af100700");

public static final UUID UUID_CHARACTERISTIC_6_BATTERY_INFO =
UUID.fromString("00000006-0000-3512-2118-0009af100700");

public static final UUID UUID_CHARACTERISTIC_7_REALTIME_STEPS =
UUID.fromString("00000007-0000-3512-2118-0009af100700")
public static final UUID UUID_CHARACTERISTIC_AUTH =
UUID.fromString("00000009-0000-3512-2118-0009af100700");
public static final UUID UUID_CHARACTERISTIC_10_BUTTON =
UUID.fromString("00000010-0000-3512-2118-0009af100700");
```

Para leer los datos de estas características sólo sería necesario realizar la siguiente llamada:

```
BluetoothGattCharacteristic characteristic =
getCharacteristic(MiBand2Service.UUID_CHARACTERISTIC_3_CONFIGURATION);

Characteristic.getValue();
```

5.4 Diseño de la aplicación

La ilustración 20 muesta la pantalla principal de la aplicación GoodFit desarrollada para el proyecto. En ella podemos observar un *floating action button* cuya función sería dirigirnos al siguiente *activity* para proceder a la detección de la pulsera. Además, contamos con un *Navigation Drawer* que se mostrará si pulsamos en las tres líneas que aparecen en la esquina superior izquierda.

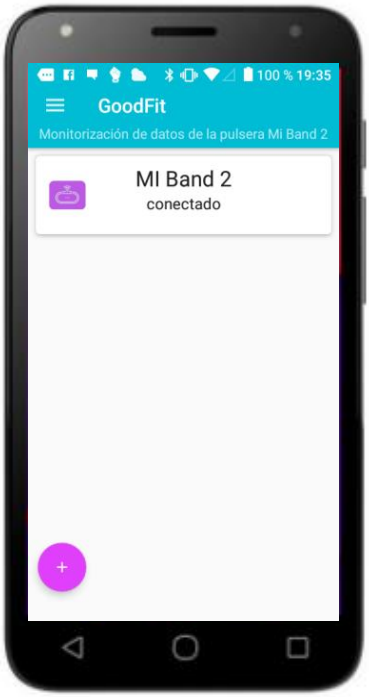


Ilustración 20: Pantalla Android principal

Esta función cuenta con tres opciones que podemos visualizar a continuación. Por un lado, en los **Ajustes** se nos mostrará un acceso directo a las notificaciones generales de la aplicación, mientras que en la opción **Sobre ti** contaremos con un cuestionario simple de datos del usuario que está haciendo uso de la smartband que son necesarios para el correcto uso de la aplicación. La gestión de estos datos se ha realizado mediante el uso de las *Preferences* de Android.

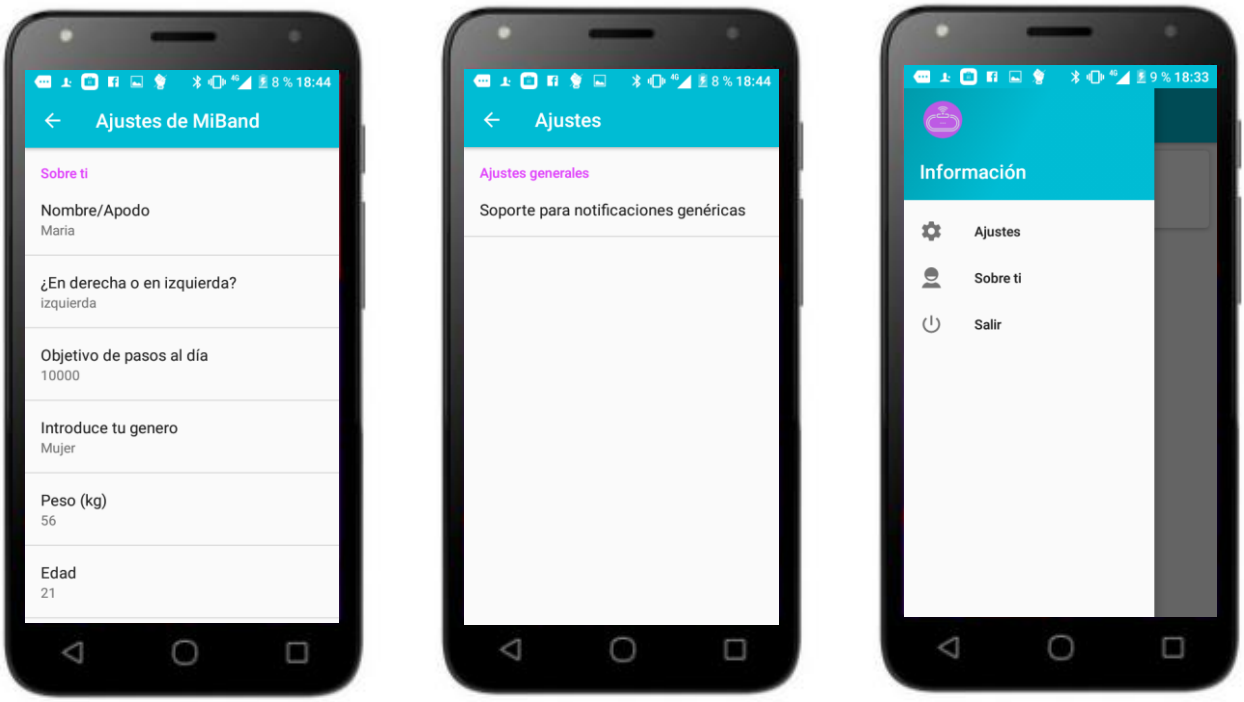


Ilustración 21: Pantalla Android Ajustes

Siguiendo con el proceso lógico de la aplicación, el siguiente paso sería detectar la *smartband* MiBand 2, *activity* al que accederemos como se ha explicado anteriormente. En él nos encontraremos un *Button* y un *ProgressBar* para indicar que la detección se está ejecutando.

A través de un *ListView* iremos mostrando los dispositivos que hayan sido detectados en la búsqueda. Con tan solo pulsar en la pulsera que queramos monitorizar, nos dirigiremos a otro *activity* para emparejar ambos dispositivos.

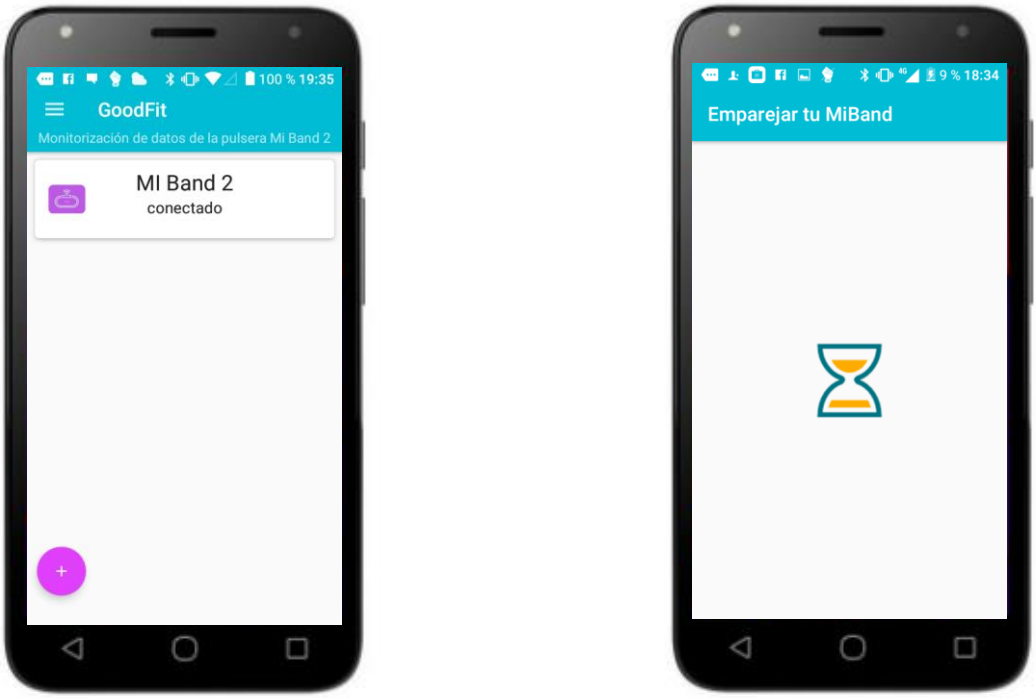


Ilustración 22: Pantalla Android conexión con *smartband*

Una vez emparejadas la pulsera con el dispositivo Android se nos mostrará una pantalla como la mostrada en la ilustración 23. Podemos observar que es la pantalla principal con un *ListView* donde visualizamos las pulseras conectadas a la aplicación. En el *Adapter* que usa este *ListView* encontramos la dirección MAC (identificador unívoco) como información relativa a la *smartband*.

Ilustración 23: Pantalla Android *smartband* conectada

Al pulsar sobre la pulsera que estemos usando, nos dirigiremos al siguiente *activity* donde encontraremos dos *fragments* con las dos funcionalidades principales de la aplicación.

El *fragment* que visualizamos por defecto es el relativo a las mediciones actuales. Podemos ver los pasos que hemos dado durante el día y la frecuencia cardíaca actual, actualizando la pantalla a través del botón que encontramos en la esquina superior derecha.

Los pasos realizados se mostrarán en un Integer simple mientras que la frecuencia cardíaca se visualiza mediante un *ProgressBar* circular. Aunque el valor de frecuencia cardíaca en el que consideramos taquicardia es a partir de 120, el valor máximo de este *ProgressBar* es 150 ya que es relativamente fácil llegar a 120 en una sesión de ejercicio.

Cuando actualizamos los datos de la pantalla y esta registra una frecuencia cardíaca de 0, se nos muestra una advertencia (*Toast*) que nos recuerda que nos ajustemos bien la pulsera.

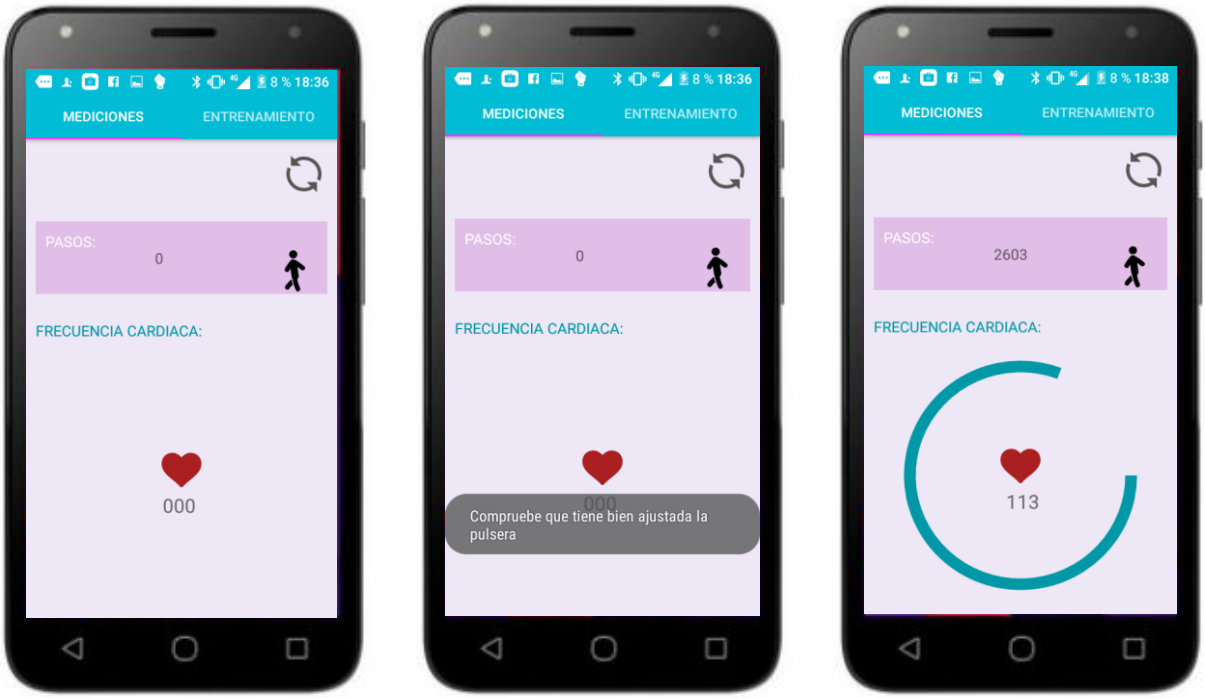


Ilustración 24: Pantalla Android opción mediciones

La segunda funcionalidad de la aplicación GoodFit es monitorizar nuestras mediciones en una sesión de entrenamiento, es decir, mientras que estamos haciendo ejercicio.

Para monitorizar las mediciones realizaremos un escaneo tanto de pasos como de la frecuencia cardiaca cada 40 segundos para que sea posible realizar unas mediciones correctas. Con estos datos calcularemos las calorías mediante las siguientes formulas diferenciadas por el género del usuario [9].

- Género femenino.

$$calorias = \left(\frac{-20.4022 + (0.4472 * frec\_cardiaca) + (0.278 * peso) + (0.074 * edad)}{4.184}\right) * min$$

- Género masculino.

$$calorias = \left(\frac{-55.0969 + (0.6309 * frec\_cardiaca) + (0.438 * peso) + (0.2017 * edad)}{4.184}\right) * min$$

Como podemos observar en la ilustración 25, con cada medición añadiremos un nuevo punto a la gráfica que visualizamos en el *activity*. Para parar el entrenamiento solo tendremos que pulsar sobre el *Button* y se reiniciarán todos los contadores.

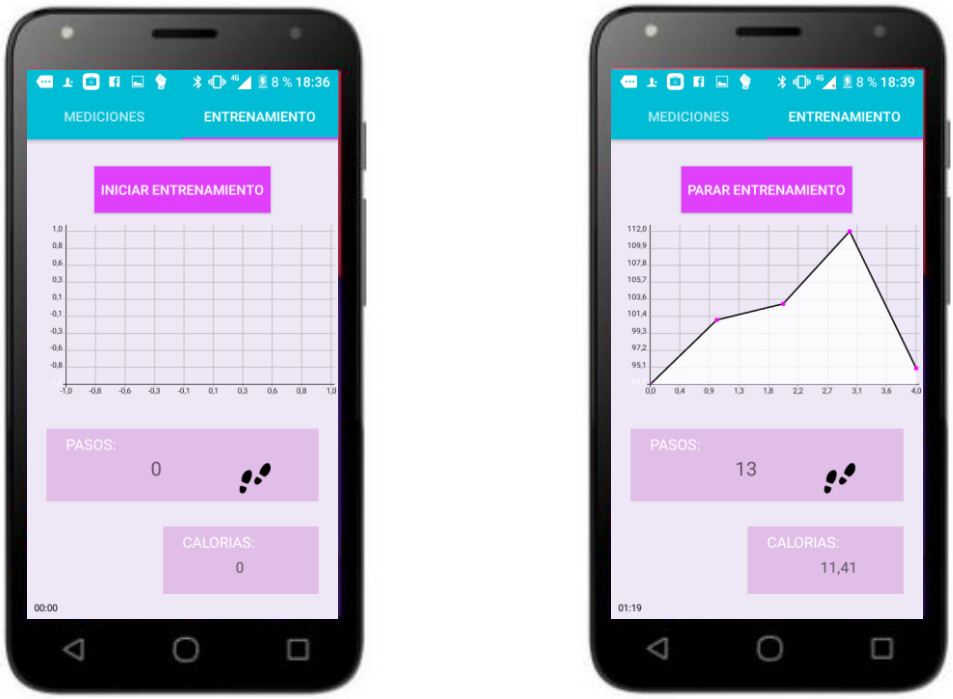


Ilustración 25: Pantalla Android opción entrenamiento

Por último, cabe citar que si emparejamos la pulsera con el dispositivo Android, esta se quedará guardada hasta que no la borremos manualmente.

En la pantalla principal, se nos mostrará la pulsera tal y como se explicó anteriormente pero el estado será no conectado. Si pulsamos sobre ella, el sistema pasará a conectarse con la pulsera automáticamente.

Para borrar la pulsera, dejaremos pulsada la pulsera y se nos mostrará un *pop-up* para confirmar el borrado de la misma. Podemos comprobar esta opción en las siguientes ilustraciones.

Ilustración 26: Pantalla Android borrado de pulsera

5.5 Diagramas de secuencia de la aplicación.

A continuación, se muestran tres diagramas de secuencia de la aplicación. Estos facilitan la visualización y el entendimiento de las interacciones entre las distintas partes del sistema.

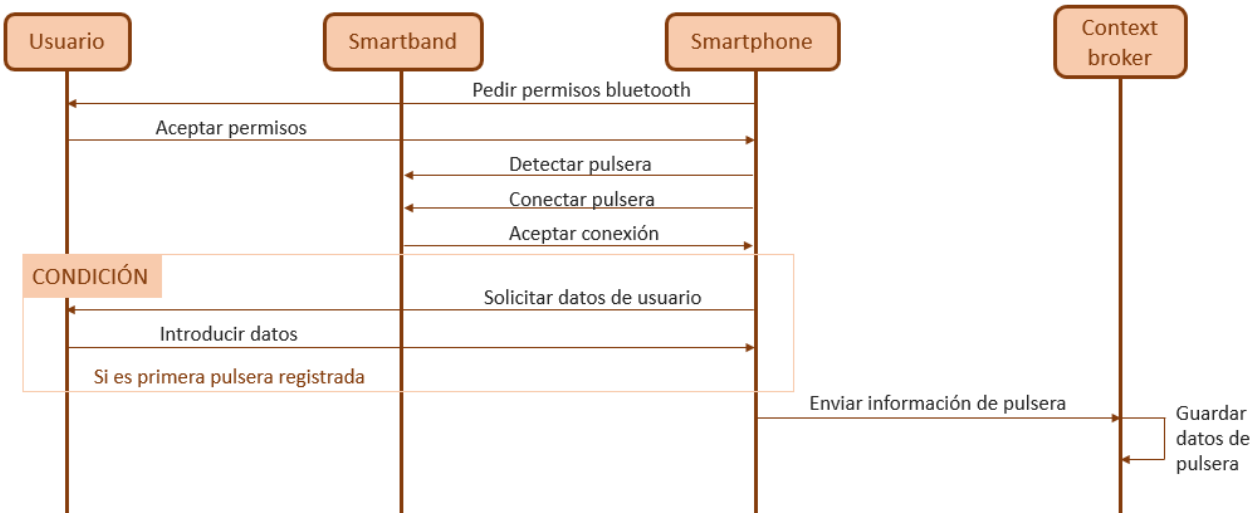


Ilustración 27: Diagrama de secuencia - Conexión con la pulsera

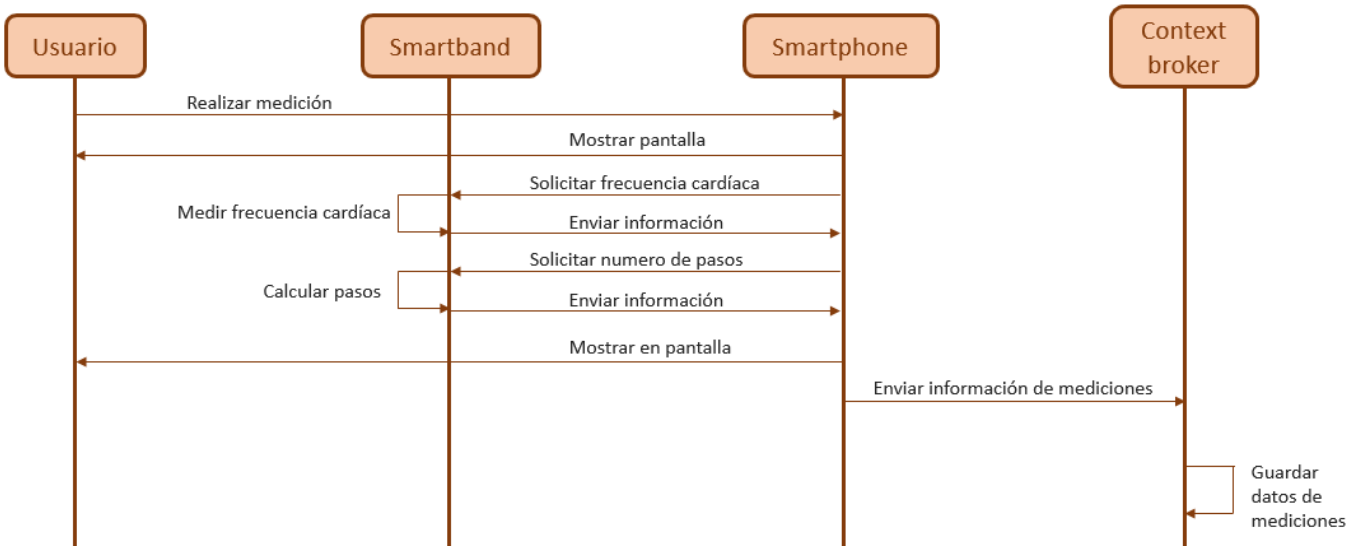


Ilustración 28: Diagrama de secuencia - Mediciones

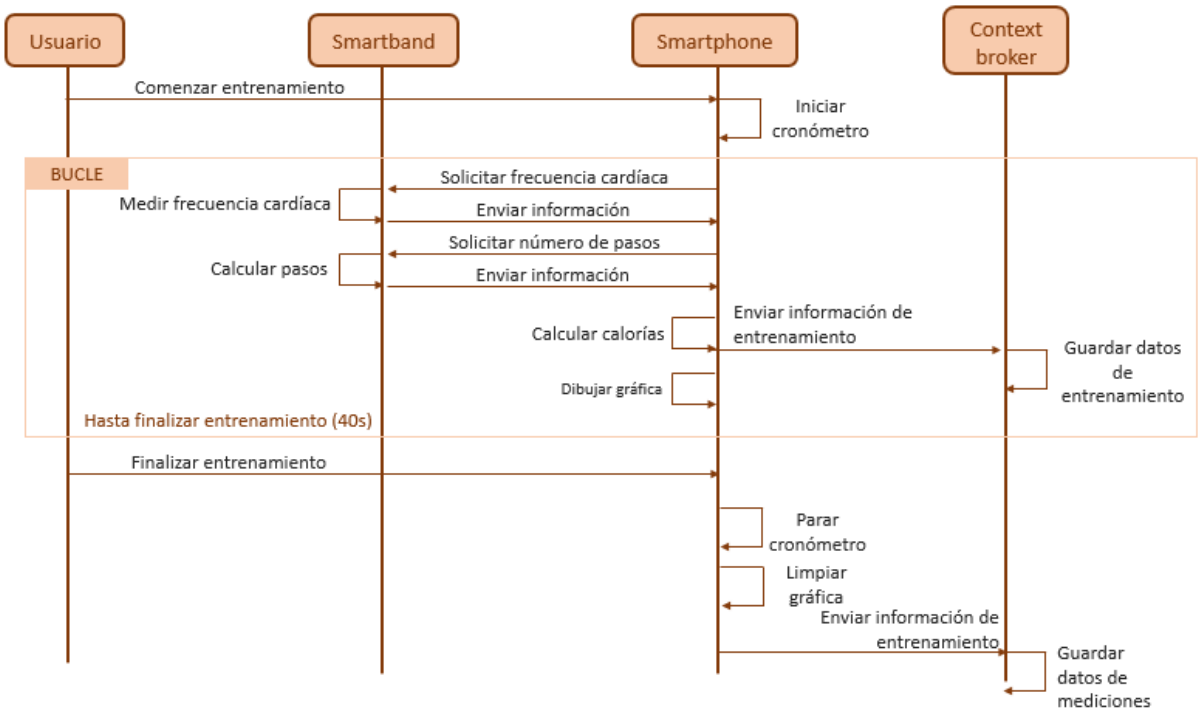


Ilustración 29: Diagrama de secuencia – Entrenamiento

6 FIWARE

La inteligencia es la capacidad de adaptarse al cambio.

- Stephen Hawking-

En este capítulo se encuentra la explicación de la tecnología actualmente en auge que hemos utilizado para la realización del proyecto. Además, podremos encontrar partes de código de la aplicación Android que son indispensables para la correcta comunicación entre ambas partes del proyecto.

6.1 La plataforma FIWARE y el proyecto FICORE.

La plataforma FIWARE está siendo desarrollada como parte del proyecto Fi-ware, financiado por el VII Programa Marco de la Unión Europea dentro del proyecto de colaboración público-privada para el Internet del Futuro (FI-PPP – "Future Internet Public-Private Partnetship"). El objetivo de la plataforma Fi-ware es el que forma el núcleo del Internet del Futuro, ofreciendo para ello una infraestructura que permita el desarrollo y despliegue de aplicaciones que conforman el denominado "ecosistema de las ciudades inteligentes". Para lograr este objetivo, la plataforma ofrece un conjunto de especificaciones y una arquitectura abierta, que permiten el desarrollo y la distribución de aplicaciones y servicios digitales.

Dentro de esta serie de recursos se encuentran:

- **FIWARE Lab**: se trata de una infaestructura virtual que dispone de todas las tecnologías desarrolladas por FIWARE.
- **FIWARE Academy**: ofrece cursos, tutoriales y lecciones para ponerse en contacto con las tecnologías desarrolladas por FIWARE.
- **FIWARE Ops**: se trata de un conjunto de herramientas que facilitan el despliegue, configuración y manejo de las instancias FIWARE [10]

La plataforma FICORE comprende un conjunto de habilitadores tecnológicos genéricos (GE: Generic Enablers), los cuales ofecen una gran cantidad de funcionalidades de propósito general mediante APIs públicas, gratuitas y bien definidas.

La arquitectura FIWARE se organiza en un conjunto de sistemas (más conocido como capítulos) que actúan como agrupadores de GEs:

- **Data/Context Management System**. Este capítulo tiene como objetivo dar soporte para el tratamiento y almacenamiento de datos y eventos y proporcionar medios para producir, recopilar, publicar y consumir información de contexto a gran escala. El presente trabajo se enmarca en este capítulo.
- **Internet of Things (IoT) Services Enablement**. Este capítulo tiene como objetivo proporcionar GEs que conviertan a cualquier objeto, organismo o persona que pueda tener uno o varios sensores en recursos de contexto accesibles, localizables y usables que permitan a las aplicaciones de FIWARE la interacción con objetos de la vida real.
- **Application/Services Ecosystem and Delivery Framework**. Este capítulo permite a las aplicaciones ser accesibles por usuarios finales desde cualquier dispositivo. Además, incorpora capacidades de Services MarketPlace y de publicación por diferentes canales.
- **Security**: Este capítulo tiene como objetivo proporcionar GEs que permitan la identificación, autentificación y autorización. La seguridad abarca desde la insfraestructura hasta la capa de aplicación.
- **Cloud Hosting**: Permite a proveedores almacenar sus aplicaciones en una infraestructura Cloud consiguiendo que los recursos se asignen dinámicamente conforme a la demanda para cumplir requisitos de negocio y SLAs.
- **Interface to Networks and Devices (I2ND)**. El objetivo de este capítulo es proporcionar interfaces para el control de calidad de servicio y la asignación de recusos de red, interfaces necesarias para el desarrollo de los componentes de la plataforma y, por último, interfaces que envuelvan el acceso a los facilitadores de red [11].

6.2 Data/Context Management System

La plataforma FIWARE permite la realización de aplicaciones y servicios inteligentes a través de un conjunto de GEs capaces de recopilar, publicar, intercambiar, procesar y analizar datos masivos de forma rápida y eficiente.

Hoy en día, varios servicios gratuitos se basan en modelos de negocio que explotan datos masivos proporcionados por usuarios finales. Estos datos se explotan en publicidad o se ofrecen a terceros para que puedan construir aplicaciones innovadoras. Twitter, Facebook, Amazon y muchos otros son ejemplos de esto.

Este capítulo tiene como objetivo proporcionar GEs que faciliten el desarrollo de aplicaciones que requieran la gestión, procesamiento y explotación de la información de contexto. Combinado con GEs procedentes de los capítulos de Marco de Aplicación y Prestación de Servicios, los proveedores de aplicaciones podrán construir modelos de negocio innovadores como los de las compañías mencionadas anteriormente.

Los GEs incluidos en este capítulo permitirán:

- Generar, suscribirse para ser notificado y consultar información de contexto procedente de diferentes fuentes.
- Procesamiento de grandes cantidades de información de contexto de forma agregada, utilizando técnicas *BigData Map & Reduce*, con el fin de generar nuevos conocimientos.
- Gestionar y publicar datos en tiempo real.
- Procesamiento, correlación y distribución de grandes volúmenes de datos.

Los GEs que componen este módulo son:

- **Publish/Subscribe Context Broker GE** que permite a las aplicaciones intercambiar eventos siguiendo el patrón publicador/suscriptor. En este GE se basa la realización del trabajo.
- **Complex Event Processing GE** que permite el procesamiento de streams en tiempo real.
- **BigData Analysis GE** que permite realizar análisis Map-Reduce de grandes volúmenes de datos.
- **Multimedia Analysis Generation GE**, que es capaz de extraer meta-información (conocimiento) de forma automtica y semiautomática para el análisis de contenido multimedia.
- **Unstructured data analysis GE**, que permite la extracción de metadatos en el análisis de información sin estructurar.
- **Meta-data pre-processing GE**, que facilita la generación de objetos desde diversos formatos de metadata.
- **Location GE**, que provee información geolocalizada desde los dispositivos.
- **Query Broker GE**, que ofrece mecanismos de query uniformes en diferentes repositorios.
- **Semantic Annotation GE**, que permite enriquecer información con meta-data para ser explotada por aplicaciones.
- **Semantic ApplicationSupport GE**, que prove soporte para trabajar con aplicaciones semánticas [11].

6.2.1 Publish/Subscribe Context Broker GE

6.2.1.1 Definiciones previas

A continuación, podemos ver una serie de vocabulario relacionado con este Generic Enabler para el correcto entendimiento de este apartado.

- Evento: Algo que ha ocurrido. Los cambios en la información de contexto se consideran eventos.
- Context Broker: Es el principal componente y funciona como interfaz entre los actores de la arquitectura. Este debe conocer los productores de contexto de la arquitectura.
- Productor de Contexto: Actor que puede generar y actualizar un contexto.
- Proveedor de Contexto: Tipo especial de productor de contexto, que proporciona información de contexto bajo demanda; eso significa que el Context Broker o incluso el consumidor de contexto puede invocar al proveedor de contexto con el fin de adquirir información.
- Consumidor de Contexto: Actor que explota la información de contexto.

6.2.1.2 Descripción funcional del Context Broker (CB)

Un Context Broker (CB) permite gestionar todo el ciclo de vida de la información de contexto incluyendo actualizaciones, consultas, registros y suscripciones. Además, utilizando CB podemos suscribirnos a la información de contexto por lo que cuando se produzca alguna condición se reciba una notificación.

El Context Broker nos permite publicar la información de contexto por medio de entidades denominadas productores de contexto. La información que estos publiquen estará disponible para otras entidades denominadas consumidores de contexto.

El principio fundamental en el que se basa un Context Broker es la disociación total entre productores y consumidores de contexto. El concepto de disociación total lo podemos entender de manera que los productores publican datos sin saber dónde y cuándo los consumidores consumen dicha información, por tanto, no necesitan estar conectados entre ellos. Por otro lado, el consumidor se interesa por el evento en sí no por quién lo generó.

Un gestor como este es imprescindible cuando se quiere desarrollar un escenario de gestión de contexto. En este tipo de escenarios se necesita un componente entre los consumidores de contexto (ej. smartphone) y los productores (ej. sensor). El Context Broker cumple esa funcionalidad intermedia en la arquitectura [12].

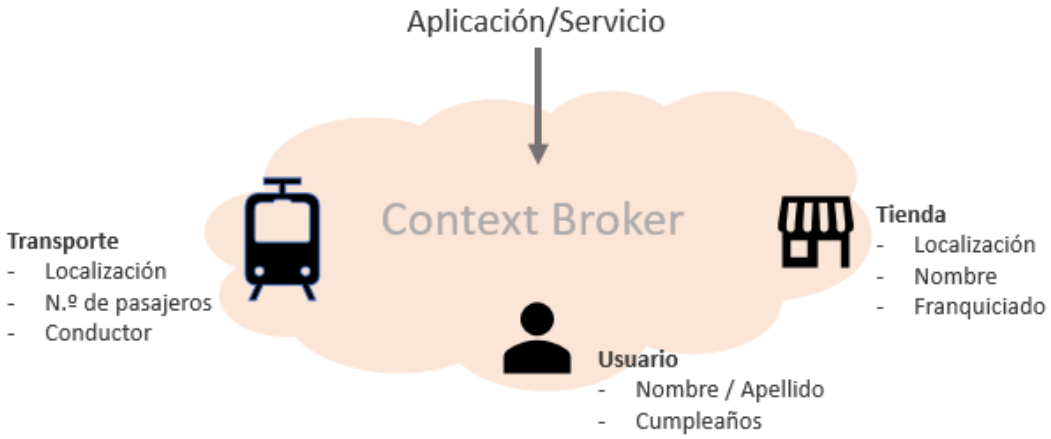


Ilustración 30: Ejemplo Context Broker

6.2.1.3 Estándares empleados por el Context Broker

El Context Broker está basado en los principios de diseño **REST**. Las tecnologías y especificaciones utilizadas en este GE son las siguientes.

- **NGSI10**

El Context Broker basa sus interfaces en los estándares NGSI definidos en el marco de la Open 2 Mobile Alliance (OMA). Estos estándares tienen como objetivo la gestión del contexto. Sin embargo, las especificaciones FIWARE sobre NGSI no implementan algunos aspectos definidos por OMA los cuales se consideran poco útiles o innecesariamente complejos.

a. Interacciones básicas y relación de entidades

Ilustración 31: Interacciones entre entidades (Context Broker) (I)

- Los productores de contexto publican elementos de contexto invocando la operación *updateContext* en el ContextBroker.
- Los consumidores de contexto reciben la información invocando la operación *queryContext* en el Content Broker [11].

b. Interacciones relacionadas con los productores de contexto

Los productores de contexto publican los elementos de contexto invocando la operación *updateContext* en una publicación o suscripción sobre el Context Broker. Algunos productores de contexto (denominados proveedores de contexto) también pueden exportar una operación queryContext de publicación o suscripción al Context Broker cuando se les solicita.

Los consumidores de contexto pueden recuperar elementos de contexto invocando a la operación *queryContext* en una publicación o suscripción al Context Broker. Al contrario de la Interacción básica, el Context Broker reenvía la consulta al proveedor de contexto apropiado y devuelve el resultado al consumidor de contexto originario [11].

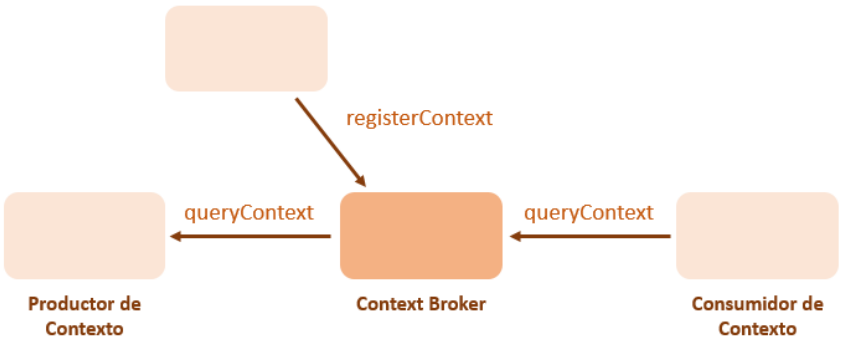


Ilustración 32: Interacciones entre entidades (Context Broker) (II)

c. Interacciones relacionadas con la subscripción de los consumidores de contexto.

Los consumidores de contexto pueden subscribirse para la recepción de datos que cumpla unas determinadas condiciones usando la operación *subscribeContext.* Los consumidores subscritos recibirán elementos de contexto a través de la operación *notifyContext* [11].

Ilustración 33: Interacciones entre entidades (Context Broker) (III)

- **HTTP**

Hypertext Transfer Protocol o HTTP es el protocolo de comunicación que permite las transferencias de información en la World Wide Web. HTTP define la sintaxis y la semántica que utilizan los elementos de software de la arquitectura web (clientes, servidores, proxies) para comunicarse. HTTP es un protocolo sin estado, es decir, no guarda ninguna información sobre conexiones anteriores.

Es un protocolo orientado a transacciones y sigue el esquema petición-respuesta entre un cliente y un servidor. El cliente realiza una petición enviando un mensaje, con cierto formato al servidor. El servidor le envía un mensaje de respuesta.

La forma de identificar servicios o documentos en la web es a través de cadenas de caracteres denominados localizadores uniformes de recursos (URL por sus siglas en inglés).

Las URL revelan la identidad del servicio al que queremos acceder, pero la acción que se debe realizar se especifica a través de los denominados verbos de HTTP. Los métodos que se pueden emplear en las peticiones HTTP son:

- GET: recoger un recurso existente. La URL contiene toda la información necesaria que el servidor necesita para localizar y devolver el recurso.
- POST: crear un nuevo recurso. Las peticiones POST suelen llevar una carga útil que especifica los datos para el nuevo recurso.
- PUT: actualizar un recurso existente. La carga útil puede contener los datos actualizados para el recurso.
- DELETE: Eliminar un recurso existente [13].

- **Formatos de Serialización de Datos.**

Entendemos como serialización el proceso de convertir un objeto en una forma que pueda ser transportada a través de un protocolo de intercambio de mensajes.

En concreto, el Context Broker GE de FIWARE soporta tanto la serialización en formato "eXtensible Markup Language" (XML) como en formato "JavaScript Object Notation" (JSON).

Por un lado, XML define un conjunto de reglas para la codificación de documentos en un formato que sea legible y de lectura mecánica.

Por otro lado, JSON se basa en la dupla atributo-valor. Originalmente derivada del lenguaje de programación JavaScript, JSON es un formato de datos independiente del lenguaje.

6.3 Utilización del Context Broker

Como resumen de lo explicado anteriormente, sabemos que los mensajes dirigidos al Context Broker son peticiones POST codificadas en lenguaje JSON.

A continuación, se explicará las peticiones POST que realiza la aplicación Android para un correcto funcionamiento de la aplicación siempre que tengamos acceso a internet en nuestro dispositivo.

6.3.1 Creación de entidades

Para un mejor procesamiento de los datos, hemos diferenciado dos tipos de entidades, la entidad de tipo *SmartBand* y la entidad de tipo *SmartBandTrain*, que como su propio nombre indica recogerá los datos de la opción del entrenamiento que ofrecemos en la aplicación *GoodFit*.

La creación de ambos tipos de entidades se realizará mediante una petición POST dirigida a la IP donde tengamos alojado el Context Broker con los siguientes cuerpos:

- Entidad ***SmartBand***

```
{"contextElements": [
      {"type":"SmartBand",
       "isPattern":"false",
       "id":"MiBandMAC",
       "attributes": [
                      {"name":"hearthrate","type":"int","value":"0"},
                      {"name":"steps","type":"int","value":"0"}
                     ]
       }
                    ],
"updateAction":"APPEND"}
```

- Entidad ***SmartBandTrain***

```
{"contextElements": [
      {"type":"SmartBandTrain",
       "isPattern":"false",
       "id":"MiBandMAC",
       "attributes": [
                      {"name":"hearthrate","type":"int","value":"0"},
                      {"name":"steps","type":"int","value":"0"}
                      {"name":"calories","type":"double","value":"0.0"}
                      {"name":"train","type":"boolean","value":"false"}
                      {"name":"duration","type":"double","value":"0.0"}
```

```
                    ]
        }
                   ],
"updateAction":"APPEND"}
```

A parte de los campos **tipo** e **id** que definen el tipo de entidad y el identificador, la carga de la petición, también llamada *payload,* contiene un conjunto de atributos. Cada atributo contiene un valor y un tipo. En el caso de la entidad ***SmartBand*** sólo tendremos los atributos *heartrate* y *steps*, mientras en la entidad ***SmartBandTrain*** tendremos los atributos *heartrate, steps, calories, train, duration.*

Estas entidades serán creadas en el momento en el cual una pulsera se empareje correctamente con un dispositivo Android haciendo uso de la aplicación *GoodFit*. Para ello, se debe ejecutar la siguiente parte del código.

- URL de la petición

```
String requestUrl="http://IP_CxtBroker:1026/v1/updateContext";
```

- Envío de la petición POST

Una de las formas de implementar la comunicación cliente servidor con la API de Android es haciendo uso de la clase HttpURLConnection. Para ello utilizaremos el código propuesto a continuación.

```
try {
    ClientREST.sendPostRequest(requestUrl, peticion);
} catch (IOException e) {
    e.printStackTrace();
}


public class ClientREST {
    public static String sendPostRequest(final String
requestUrl, final String payload) throws IOException {
        final StringBuffer[] jsonString = new StringBuffer[1];

        AsyncTask.execute(new Runnable() {
            @Override
            public void run() {
                try {


                    URL url = new URL(requestUrl);
                    HttpURLConnection connection =
(HttpURLConnection) url.openConnection();

                    connection.setDoInput(true);
                    connection.setDoOutput(true);
                    connection.setRequestMethod("POST");
                    connection.setRequestProperty("Accept",
"application/json");
                    connection.setRequestProperty("Content-
```

```
Type", "application/json; charset=UTF-8");
                    OutputStreamWriter writer = new
OutputStreamWriter(connection.getOutputStream(), "UTF-8");
                    writer.write(payload);
                    writer.close();
                    BufferedReader br = new BufferedReader(new
InputStreamReader(connection.getInputStream()));
                    jsonString[0] = new StringBuffer();
                    String line;
                    while ((line = br.readLine()) != null) {
                        jsonString[0].append(line);
                    }
                    br.close();
                    connection.disconnect();

                } catch (IOException e) {
                    e.printStackTrace();
                }
            }
        });
if (jsonString[0] == null)
    return null;
        else {
    return jsonString[0].toString();
}
    }
}
```

6.3.2 Actualización de entidades

El procedimiento para actualizar las entidades ya creadas es similar al anterior. Para, ello en el cuerpo de la petición deberíamos modificar el valor del campo *updateAction* a **UPDATE**. El *payload* de la petición será la misma que en la creación de entidades. En nuestro caso, en la creación de la entidad todos los atributos tendrán valor 0, sin embargo, en su actualización ya se hará uso de datos reales [14].

7 APLICACIÓN WEB

Daría todo lo que sé, por la mitad de lo que ignoro.

- René Descartes-

En este capítulo podemos encontrar los conceptos estudiados para poder desarrollar correctamente la aplicación web. Además, visualizaremos la pantalla de la aplicación y como navegar en ella, así como distintas partes de código que son claves para el correcto funcionamiento de la aplicación.

7.1 Tecnologías utilizadas

7.1.1 HTML

HTML, sigla en inglés de HyperText Markup Language, hace referencia al lenguaje de marcado para la elaboración de páginas web. Es un estándar que sirve de referencia del software que conecta con la elaboración de páginas web en sus diferentes versiones, define una estructura básica y un código para la definición de contenido de una página web, como texto, imágenes, videos, juegos, entre otros.

Es un estándar a cargo del World Wide Web Consortium, organización dedicada a la estandarización de casi todas las tecnologías ligadas a la web, sobre todo en lo referente a su escritura e interpretación. Se considera el lenguaje web más importante siendo su invención crucial en la aparición, desarrollo y expansión de la World Wide Web (WWW).

El lenguaje HTML basa su filosofía de desarrollo en la diferenciación. Para añadir un elemento externo a la página, este no se incrusta directamente en el código de la página, sino que se hace una referencia a la ubicación de dicho elemento mediante texto. De este modo, la página web contiene solamente texto mientras que recae en el navegador web la tarea de unir todos los elementos y visualizar la página final.

El HTML también puede describir, hasta un cierto punto, la apariencia de un documento, y puede incluir o hacer referencia a un tipo de programa llamado script, el cual puede afectar el comportamiento de navegadores web y otros procesadores de HTML.

El lenguaje HTML puede ser creado y editado con cualquier editor de textos básico, como puede ser Gedit en GNU/Linux, el Bloc de notas de Windows, o cualquier otro editor que admita texto sin formato como GNU Emacs, Microsoft Wordpad, TextPad, Vim, Notepad++, entre otros [15]

7.1.2 CSS

Hojas de estilo en cascada (CSS) es un lenguaje de diseño gráfico para definir y crear la presentación de un documento estructurado escrito en un lenguaje de marcado. Es muy usado para establecer el diseño visual de las páginas web, e interfaces de usuario escritas en HTML.

7.1.3 JavaScript

JavaScript es un lenguaje de programación interpretado. Se define como orientado a objetos, basado en prototipos, imperativo, débilmente tipado y dinámico.

Se utiliza principalmente en su forma del lado del cliente (client-side), implementado como parte de un navegador web permitiendo mejoras en la interfaz de usuario y páginas web dinámicas.

JavaScript se diseñó con una sintaxis similar a C, aunque adopta nombres y convenciones del lenguaje de programación Java. Sin embargo, Java y JavaScript tienen semánticas y propósitos diferentes [16].

7.1.4 PHP

PHP es un lenguaje de programación de uso general de código del lado del servidor originalmente diseñado para el desarrollo web de contenido dinámico. Fue uno de los primeros lenguajes de programación del lado del servidor que se podían incorporar directamente en el documento HTML en lugar de llamar a un archivo externo que procese los datos. Puede ser usado en la mayoría de los servidores web al igual que en casi todos los sistemas operativos y plataformas sin ningún costo [17].

7.1.5 Ajax

AJAX, acrónimo de Asynchronous JavaScript And XML (JavaScript asíncrono y XML), es una técnica de desarrollo web para crear aplicaciones interactivas o RIA (Rich Internet Applications). Estas aplicaciones se ejecutan en el cliente, es decir, en el navegador de los usuarios mientras se mantiene la comunicación asíncrona con el servidor en segundo plano.

Ajax es una tecnología asíncrona, en el sentido de que los datos adicionales se solicitan al servidor y se cargan en segundo plano sin interferir con la visualización ni el comportamiento de la página.

JavaScript es un lenguaje de programación (scripting language) en el que normalmente se efectúan las funciones de llamada de Ajax mientras que el acceso a los datos se realiza mediante XMLHttpRequest, objeto disponible en los navegadores actuales [18].

7.2 Diagrama Base de Datos.

Para tener un diseño conceptual de la base de datos a implementar, es necesario realizar un diagrama de entidad-relación, en el cual se representa la composición de las entidades y sus relaciones.

Cada entidad simboliza un objeto o concepto que se describe en la base de datos de la aplicación. En otras palabras, son elementos que intervienen directamente en el sistema de datos que se quiere crear y están compuestos de atributos que describen las características de dicho objeto.

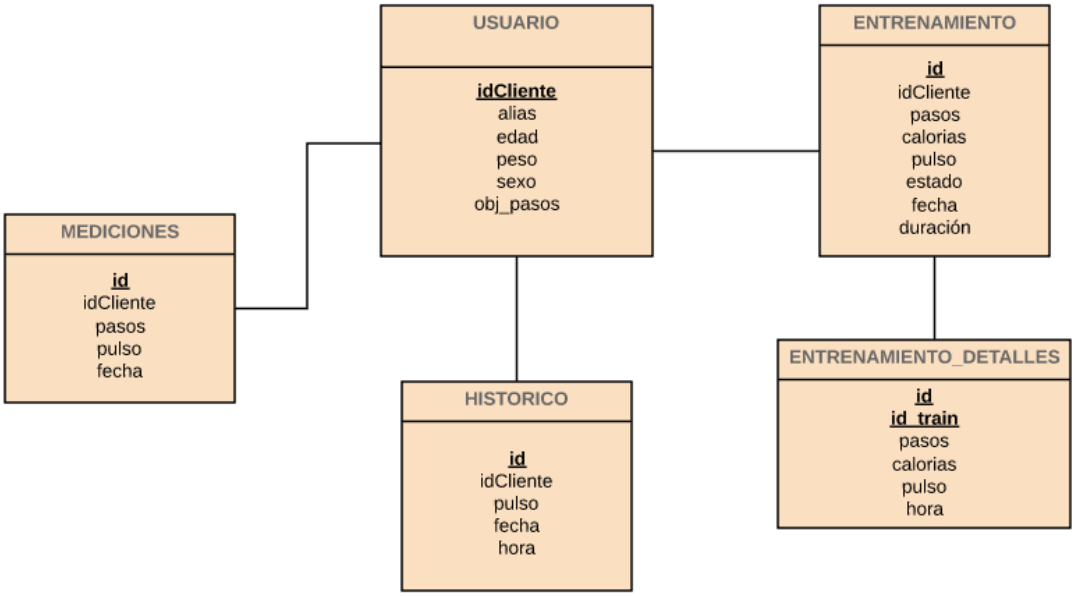


Ilustración 34: Diagrama Base de Datos aplicación Web

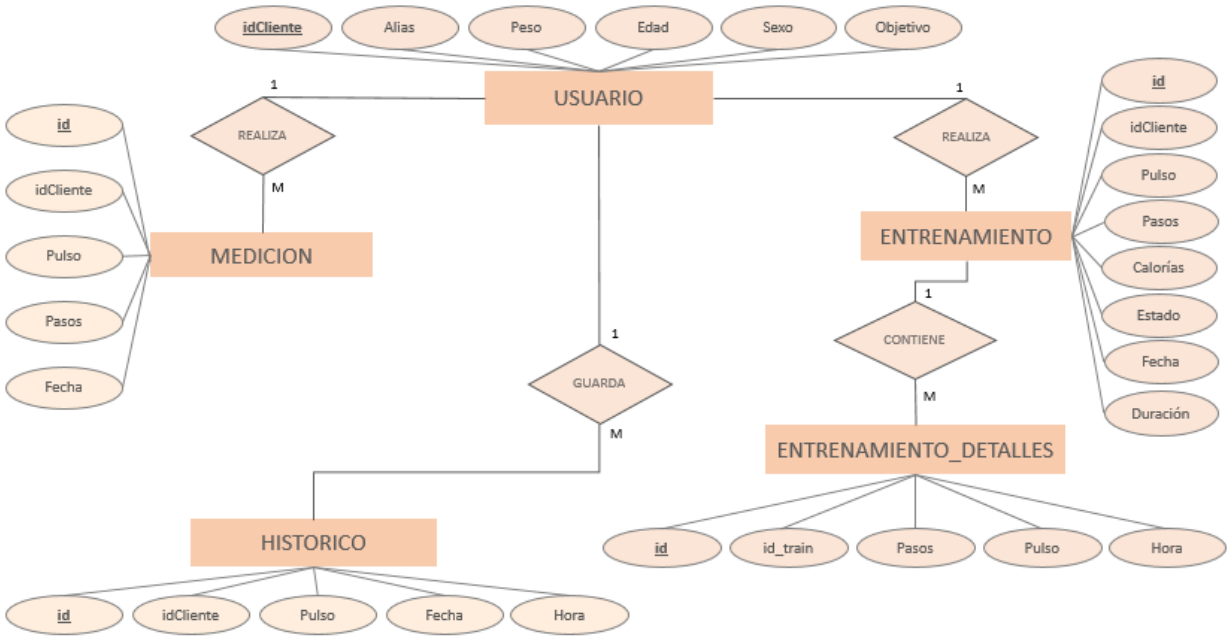


Ilustración 35: Diagrama entidad-relación

Como podemos observar en la ilustración anterior, todas las entidades que intervienen se relacionan directamente como subclases de la entidad Usuario, esto se debe a que el objetivo de la aplicación no es otro que el de ofrecer un estudio de cada usuario particular.

7.3 Desarrollo e implementación de la aplicación Web.

Después de definir de la forma más completa posible tanto el comportamiento de la aplicación como sus objetivos y requisitos, en este apartado encontraremos el traslado de toda esa información a código.

Como hemos mencionado en un capítulo anterior, hemos implementado y desarrollado la página web a través de XAMPP y el editor de texto Notepad++.

En la siguiente imagen podemos observar cual será la primera vista de la aplicación. Comprobamos que, sin datos, la tabla se muestra vacía.

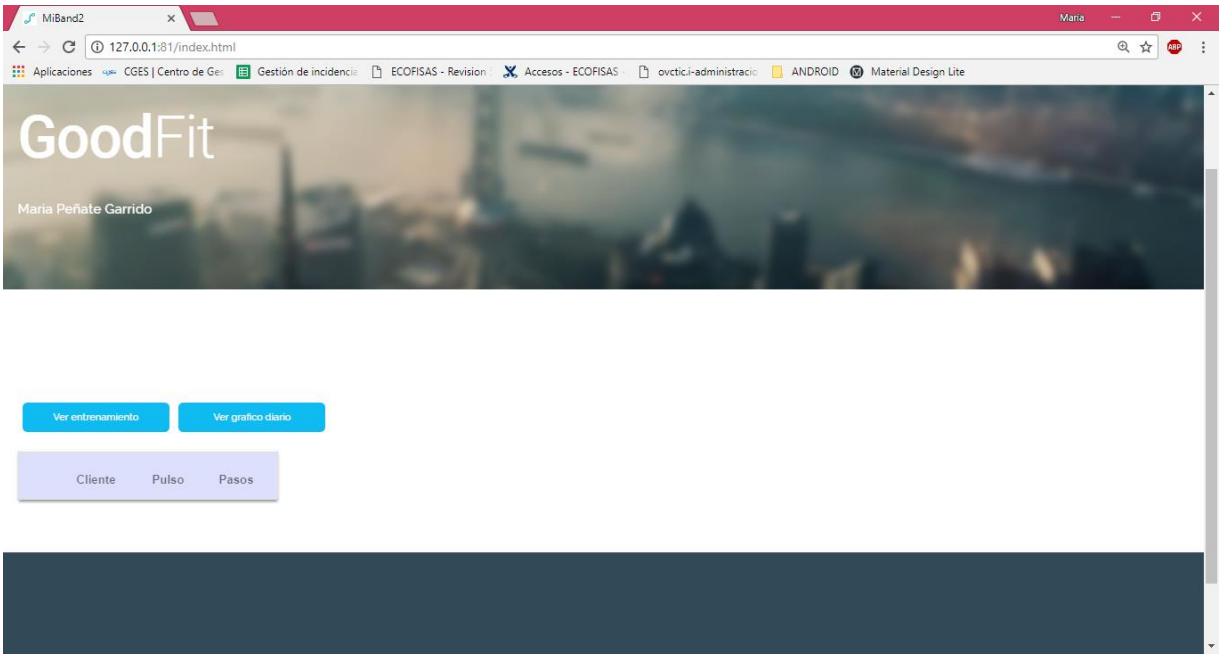


Ilustración 36: Página Web – Pantalla principal sin datos

Una vez que las aplicaciones conectadas a las pulseras realicen mediciones, el Context Broker ya tendrá datos guardados y, por tanto, ya podremos visualizar estos datos si recargamos la página. Podemos observar en la siguiente imagen, dos pulseras conectadas a la misma o distinta aplicación cuya última medición de datos se recoge en la tabla que visualizamos.

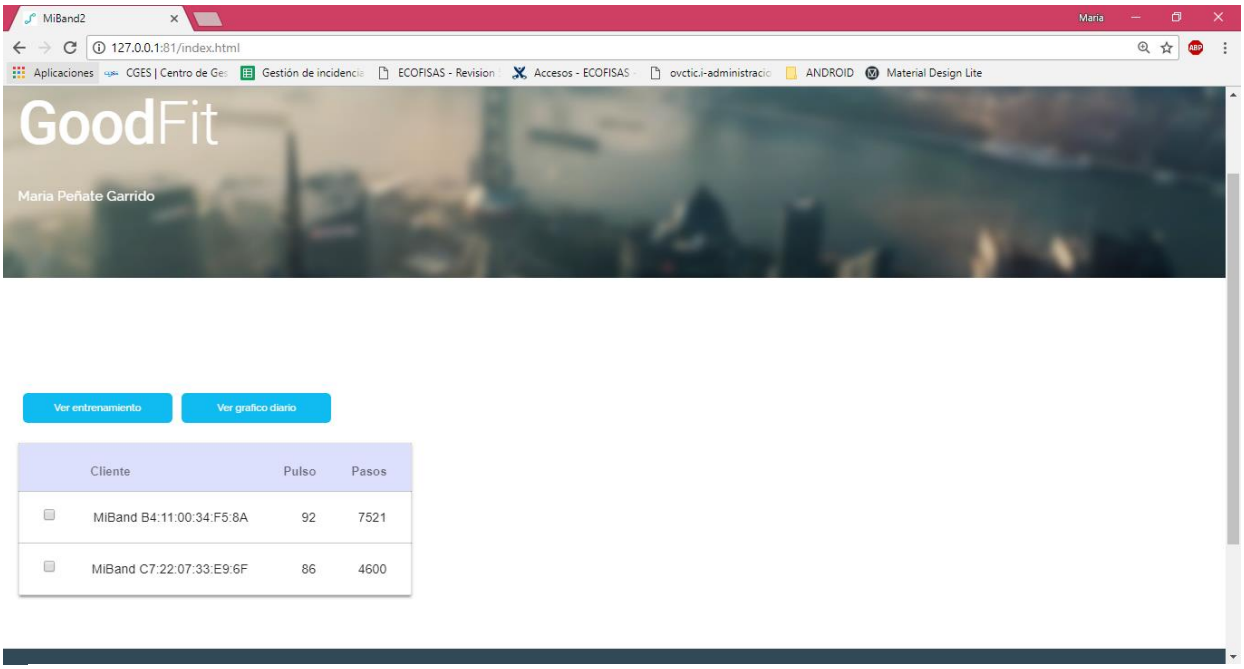


Ilustración 37: Página Web - Pantalla principal con datos

En la parte posterior a la tabla, encontramos dos botones con distintas funcionalidades.

- El primero comenzando por la derecha, **Ver gráfico diario**, nos mostrará en la parte lateral de la página un gráfico del pulso recogido por la aplicación durante un día.

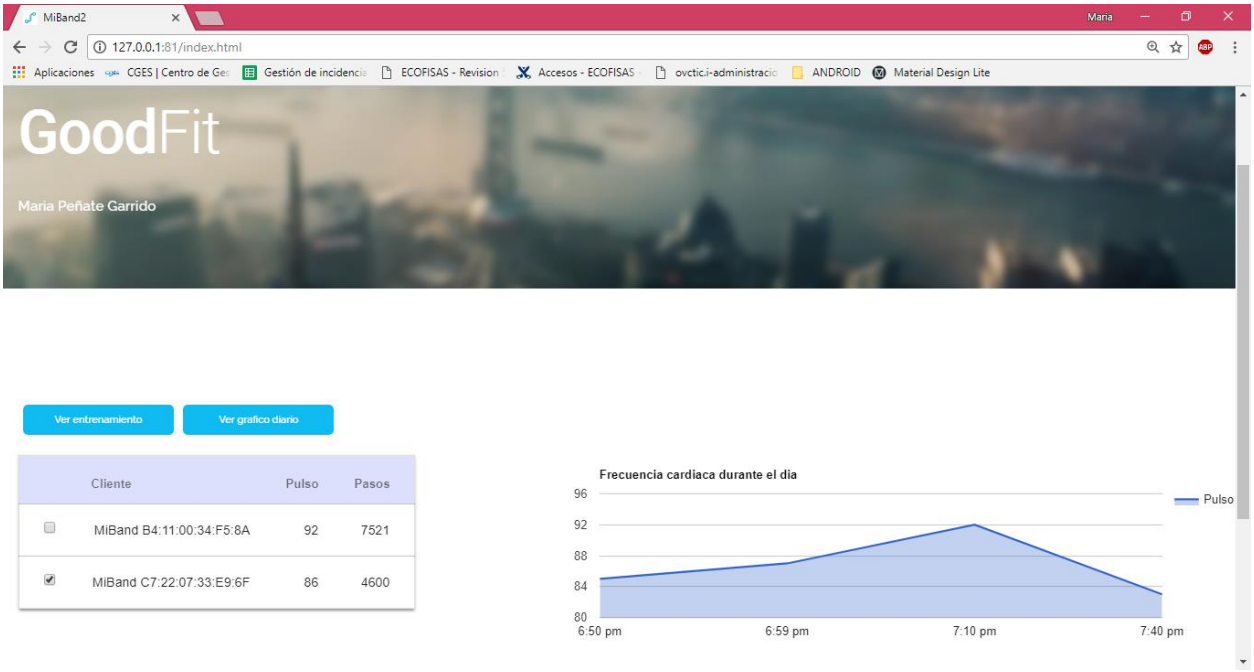


Ilustración 38: Página Web – Gráfico diario

- El siguiente, **Ver entrenamiento**, nos mostrará en la parte inferior de la página la información recogida, relativa a las sesiones de entrenamiento realizadas por el usuario seleccionado. Es necesario citar que cuando el estado tiene valor 1, el entrenamiento no ha finalizado. Podemos

comprobar en la ilustración 39 que es necesario filtrar por fecha para visualizar los datos del entrenamiento.

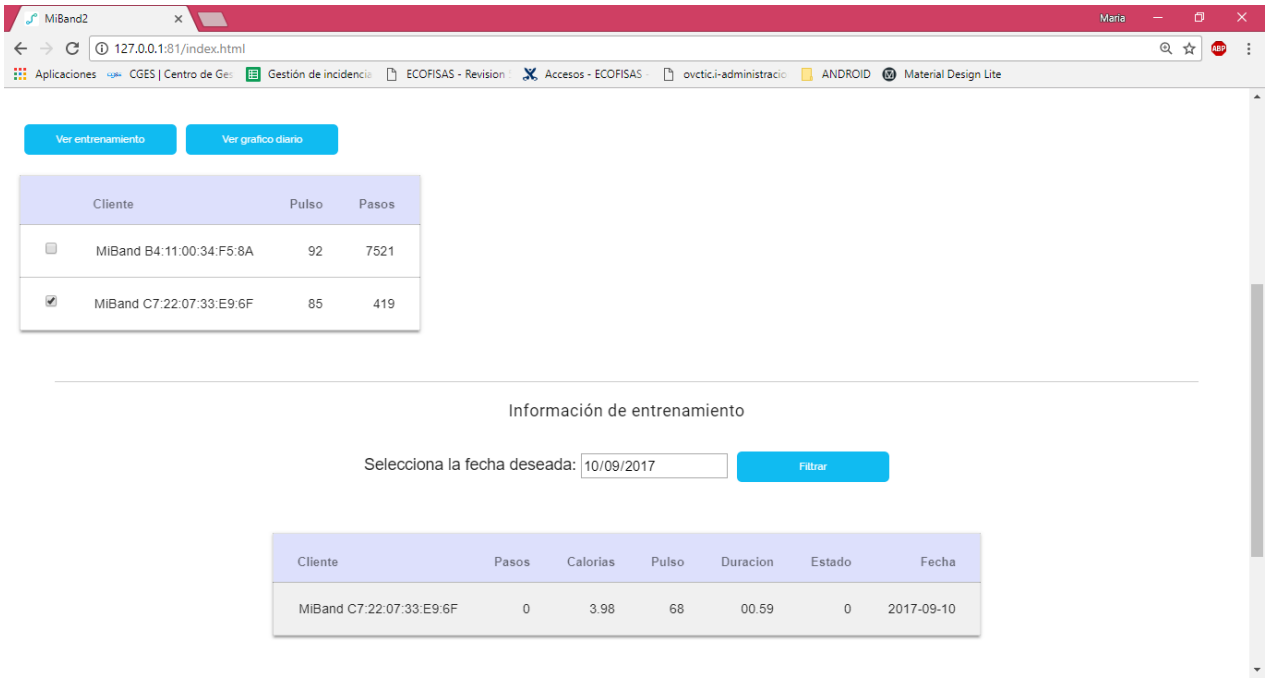


Ilustración 39: Página Web – Tabla entrenamiento

También que, si no seleccionamos ninguna pulsera y pulsamos algún botón, la aplicación lanzará un pop-up informando del error.

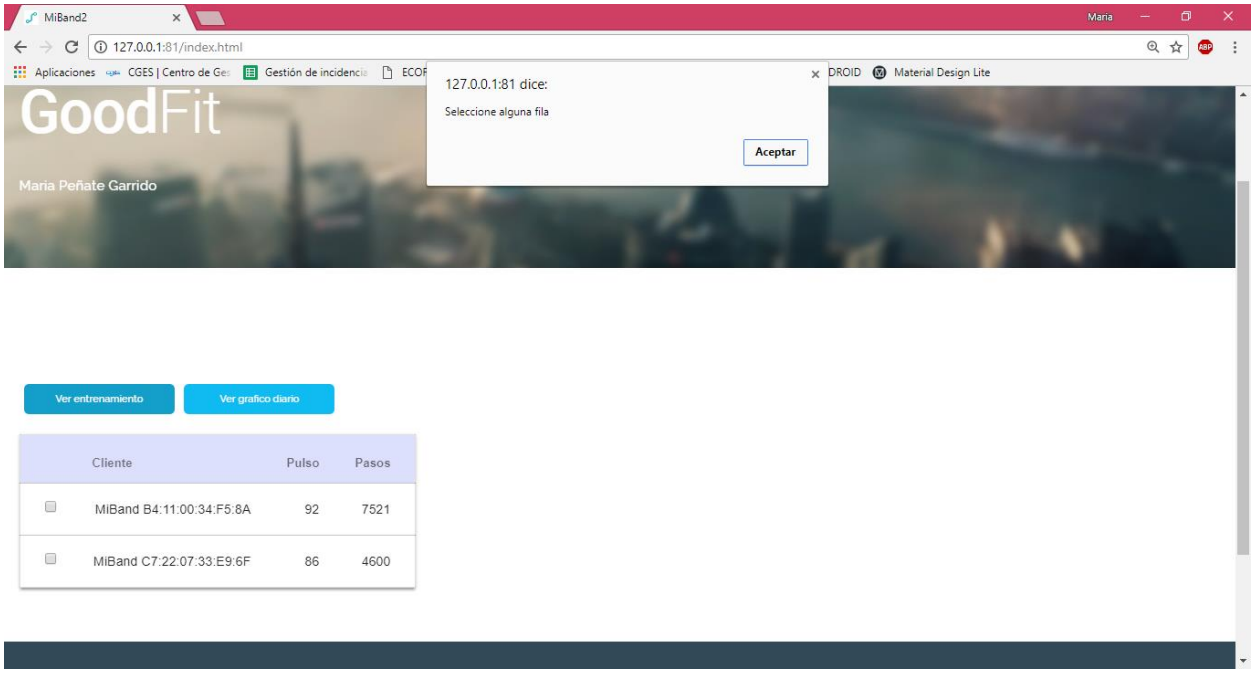


Ilustración 40: Página Web – Error selección

7.3.1 Conexión con la BBDD

Hemos utilizado varios lenguajes para la creación de la web, en concreto, la conexión con la base de datos y las peticiones a esta se han realizado en PHP. A continuación, se muestran dichas partes del código.

- Conexión base de datos [19].

```
$enlace = mysql_connect('127.0.0.1:3306',  'root', 'fiware');
if  (!$enlace) {
    die('No pudo conectarse: ' . mysql_error()); }
```

- Peticiones [19]

```
$sql = "select * from fiware.datos where idCliente like 'MiBand%'";
$hay_cliente=mysql_query($sql);
```

7.3.2 Comunicación con el Context Broker

En este apartado encontraremos la porción del código relativo a la comunicación con el Context Broker, desde que se lanza la petición hasta que se registra la respuesta. Esta comunicación es necesaria ya que queremos visualizar los datos recogidos por el *smartphone* en la aplicación web. Por tanto, necesitaremos realizar peticiones al Context Broker para que este nos informe acerca de los datos actuales.

Comenzaremos definiendo la URL de la petición que dependerá de los casos explicados en el capítulo anterior.

```
$url = 'http://192.168.1.30:1026/v1/queryContext';
```

cURL es un proyecto de software consistente en una biblioteca (libcurl) y un intérprete de comandos (curl) orientado a la transferencia de archivos. El principal propósito y uso para cURL es automatizar transferencias de archivos o secuencias de operaciones no supervisadas. En concreto, lo hemos usado para simular acciones de usuarios en un navegador web, es decir, para enviar las solicitudes POST a una URL concreta.

```
$ch = curl_init($url);
```

Procedemos a crear el dato que cargaremos como *payload* de la petición. Esto se modificará dependiendo tanto del tipo de petición como de los datos que estemos solicitando.

```
$jsonData  =  '{  "entities"  :  [{  "type":"SmartBand"  ,  "isPattern":"true",
"id":"MiBand .*"}]}';
```

A continuación, indicaremos que se va a realizar una petición **POST**.

```
curl_setopt($ch, CURLOPT_POST, 1);
```

Una vez aquí, alteraremos la configuraciôm para poder manipular el resultado de la petición correctamente, cargamos el payload en la petición y agregamos las cabeceras oportunas.

```
curl_setopt($ch, CURLOPT_RETURNTRANSFER, 1);
curl_setopt($ch, CURLOPT_POSTFIELDS, $jsonData);
curl_setopt($ch, CURLOPT_HTTPHEADER, array('Content-Type:application/json'));
```

Por último, ejecutamos la petición cuyo resultado se guardará en la variable indicada para poder manipular los datos.

```
$result = curl_exec($ch);
```

Una vez obtenido el resultado, mediante manipulación de strings y arrays podremos obtener los datos que insertaremos en la base de datos.

Tanto la conexión con la base de datos como la comunicación con el Context Broker se harán en un fichero distinto al principal de nuestra web. Llegaremos a ejecutar este fichero programado en PHP tras una petición AJAX lanzada desde la página principal.

```
$.ajax({
                                    type: "POST",
                                    url: "query.php",
                                    async: true,
                                   data: {cliente: idCliente},
                                    success: function(datos){
                                        dataJson = eval(datos);

                                          for(var      i      in      dataJson){
//en este bucle manipularemos los datos
                                              }
                                    },
                                    error: function (obj, error, objError){
                                        //avisar que ocurrió un error
                                    } });
```

El campo url contendrá la dirección del fichero a ejecutar y el campo data los parámetros que pasamos desde el fichero principal hacia el fichero definido en la URL [19].

7.4 Diagrama de secuencia de la aplicación

En este diagrama mostraremos como se actualiza la base de datos. Podemos observar como una vez recocogidos los datos en el Context Broker, la web se refrescará cada 40 segundos, solicitando los datos.

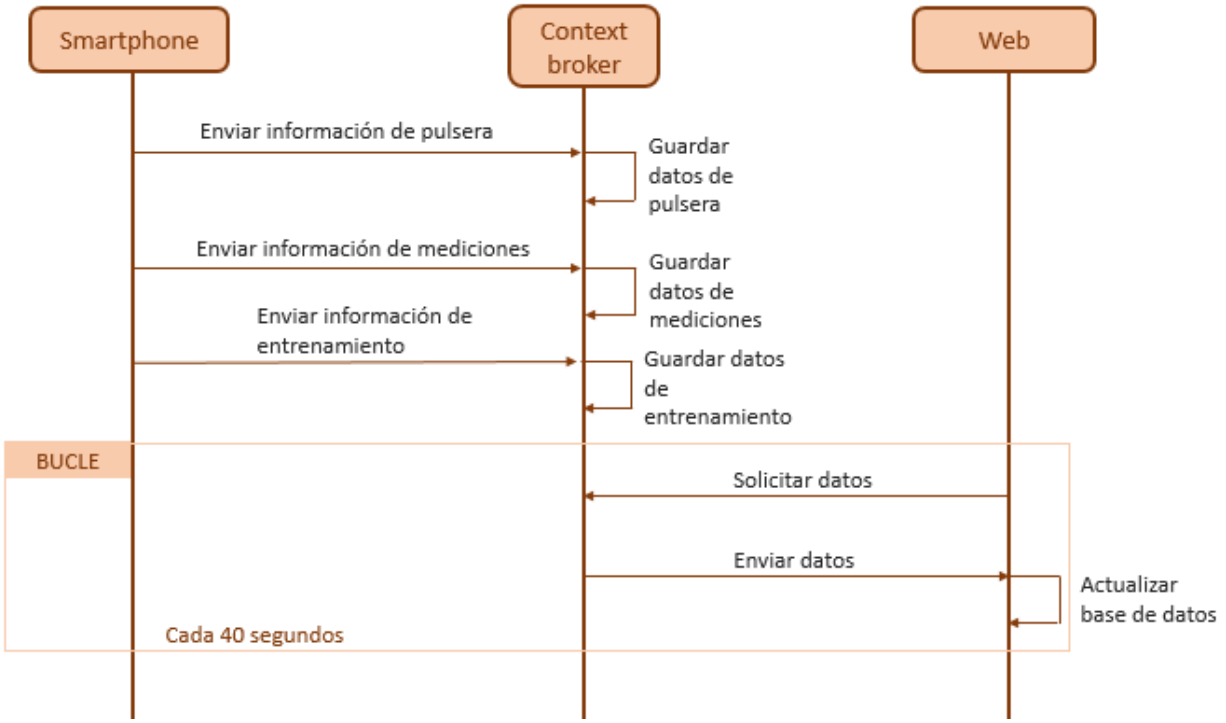


Ilustración 41: Diagrama de actividad - Aplicación web

8 PRUEBAS Y VALIDACIÓN

Da igual. Prueba otra vez. Fracasa otra vez. Fracasa mejor.

- Samuel Beckett-

En este capítulo podremos encontrar una serie de pruebas manuales que se han realizado para verificar el correcto funcionamiento del sistema completo.

8.1 Pruebas manuales

Prueba 01	**Apertura y navegación correcta en la aplicación**
Descripción	Se abrirá la aplicación y se navegará por todas las partes que podamos acceder sin tener pulsera registrada, comprobando que no afectan ni al diseño ni al funcionamiento de la misma.
Resultado	Superada

Tabla 54: Pruebas manuales - Apertura y navegación correcta en la aplicación

Prueba 02	**Comprobación de acceso a notificaciones**
Descripción	Nos dirigimos a la opción de ajustes y comprobamos que, al pulsar sobre la opción de notificaciones, la aplicación nos redirige a la opción correcta.
Resultado	Superada

Tabla 55: Pruebas manuales - Comprobación de acceso a notificaciones.

Prueba 03	**Solicitud de permisos al iniciar por primera vez.**
Descripción	Comprobamos que al usar por primera vez la aplicación se nos mostrará un pop-up para aceptar los permisos necesarios para el uso correcto de la aplicación.
Resultado	Superada

Tabla 56: Pruebas manuales – Solicitud de permisos al iniciar por primera vez

Prueba 04	**Conectar con smartband**
Descripción	A través de la pantalla principal de la aplicación, detectaremos una nueva smartband y nos emparejaremos con ella. Comprobamos que esta se añade a la lista de la pantalla principal en estado **conectado**.
Resultado	Superada

Tabla 57: Pruebas manuales - Conectar con smartband

Prueba 05	**Solicitud de información de usuario al registrar una pulsera por primera vez**
Descripción	Se comprueba que la aplicación solicita la información de usuario cuando emparejamos la primera pulsera con la aplicación. Comprobamos que los datos son modificables en cualquier momento.
Resultado	Superada

Tabla 58: Pruebas manuales – Solicitud de información de usuario al registrar una pulsera por primera vez

Prueba 06	**Comprobación de que los datos medidos son correctos.**
Descripción	Se comprueba tras 5 mediciones cada 5 minutos que los datos recogidos y visualizados por la aplicación coinciden con los marcados en la pulsera.
Resultado	Superada

Tabla 59: Pruebas manuales – Comprobación de que los datos medidos son correctos

Prueba 07	**Datos correctos durante la monitorización**
Descripción	Se comprueba que los datos recogidos durante una monitorización de 3 minutos son datos que concuerdan con la actividad física del momento.
Resultado	Superada

Tabla 60: Pruebas manuales – Datos correctos durante la monitorización

Prueba 08	**Eliminación de smartband en la lista**
Descripción	Nos situaremos en la pantalla principal y borraremos una pulsera. Comprobamos que al hacer esto, el contacto no aparece en la lista de la cual se borró.
Resultado	Superada

Tabla 61: Pruebas manuales – Eliminación de smartband en la lista

Prueba 09	**Funcionamiento correcto de la aplicación sin acceso a Internet**
Descripción	Comprobamos que la aplicación cumple con sus funcionalidades y funciona correctamente quitando los datos y la Wi-Fi del dispositivo.
Resultado	Superada

Tabla 62: Pruebas manuales – Funcionamiento correcto de la aplicación sin acceso a Internet.

Prueba 10	**Funcionamiento correcto del sistema completo con acceso a Internet**
Descripción	Comprobamos que cuando tenemos acceso a Internet, la web visualiza todos los datos de las mediciones realizadas de las pulseras conectadas a la aplicación.
Resultado	Superada

Tabla 63: Pruebas manuales – Funcionamiento correcto del sistema completo con acceso a Internet

8.2 Pruebas de aceptación

Prueba 11	**Validación del producto final por parte del tutor**
Descripción	Se le entregará el producto final al tutor del proyecto, de manera que éste lo pruebe, y determine si realmente cumple las expectativas planteadas.
Resultado	

Tabla 64: Pruebas de aceptación - Validación del producto por parte del tutor

9 CONCLUSIONES

Todo lo que puedas imaginar es real.
- Pablo Picasso -

9.1 Conclusiones

Considero que este proyecto ha sido muy positivo tanto para mi formación como para mi futuro laboral ya que como hemos comentado a lo largo del proyecto, las tecnologías usadas en el mismo están en pleno auge y supondrán un gran cambio en distintas áreas, mayormente en el área sanitaria. No obstante, es evidente que aún tienen obstáculos que solventar para alcanzar una mayor notoriedad; la desconfianza de los profesionales y el propio mercado son unos de los más importantes.

La realización del proyecto ha sido una experiencia satisfactoria ya que se han podido cumplir todos los objetivos y requisitos propuestos al comienzo del mismo.

Durante su realización se ha tenido que combinar conocimientos adquiridos en el grado con conocimientos adquiridos recientementes para poder desarrollarlo correctamente. En estos se incluyen tanto el lenguaje de programación Android como los dispositivos *wearables*, ambos desconocidos para mí antes de comenzar el proyecto. Además, he podido ampliar mis conocimientos en algunos lenguajes de programación importantes hoy en día como, por ejemplo, Java de los que tenía conocimientos básicos al ser impartidos en el grado.

Queda reflejado en el párrafo anterior, pero es necesario destacar la gran cantidad de tiempo empleado en el aprendizaje de la programación en Android. Se ha tenido que aprender de forma autónoma sin tener destreza con el lenguaje en el que está basado.

Desde mi punto de vista, una de las partes más complicadas del proyecto fue implementar correctamente la comunicación entre un dispositivo *Xioami MiBand 2* y un *Smartphone,* ya que maneja una gran cantidad de conceptos abstractos. Además, la inclusión de *Bluetooth Low Energy* en el proyecto ha supuesto un estudio exhaustivo de esta tecnología para comprender su funcionamiento.

9.2 Línea de operación

Este proyecto tiene un gran margen de mejora y actualmente, dado su planteamiento, se trata de una buena base sobre la que seguir trabajando en busca de más funcionalidades con el fin de mejorar el servicio ofrecido a los usuarios de esta smartband. Entre una multitud de ideas, se destacan:

- **Login de usuarios.** Introducir la opción de un login de usuarios con dos finalidades; que el smartphone pueda ser usado por varios usuarios sin necesidad de modificar sus datos personales cada vez que vayan a usar la aplicación y por otro lado, que el usuario tenga la posibilidad de cerrar su sesión.

- **Identificación de usuarios.** Actualmente el sistema registra un usuario por cada dirección MAC conectada a la aplicación. La mejora propuesta sería introducir una opción para poder asociar dos direcciones MAC al mismo usuario para no perder el registro en el caso de que el usuario cambie de smartband.

- **Detectar diferentes tipos de pulseras.** Realizar un estudio de las más usadas y conseguir que la aplicación pueda comunicarse con diferentes tipos de pulseras con el fin de ampliar los destinatarios de la misma.

- **Mantenimiento de la aplicación**. Dada la evolución continua del sistema operativo de Android, se debe mantener actualizada siempre la aplicación para una mayor optimización

- **Funcionalidad web.** Ampliar la funcionalidad web de modo que el gestor de datos, es decir, el usuario de la aplicación web pueda realizar una medición o una monitorización de una pulsera concreta.

- **Login de usuarios en la web.** Ampliar la web para que se haga uso de un usuario y contraseña para acceder a los datos de la web.

- **Realizar estudio de sueño.** Usar los datos facilitados por la pulsera para realizar estudios de sueño.

- **Relación de usuarios con profesional.** Añadir la funcionalidad de que un usuario pueda asociarse a un profesional. Así, cada profesional podrá monitorizar sólo a sus propios pacientes.

Podemos observar que las mejoras propuestas son nuevos requisitos que implementar en la aplicación.

ANEXOS

ANEXO 1. Instalación Context Broker

Para usar el Context Broker se ha hecho uso de una máquina virtual con sistema operativo CentOS 6.7 donde se facilitaba su instalación, previniendo así bastantes errores.

En el catálogo de FIWARE (https://catalogue.fiware.org/enablers/publishsubscribe-context-broker-orion-context-broker/downloads) podremos encontrar el enlace para la descarga. Aunque se muestran cinco enlaces, se ha utilizado http://bit.ly/fiware-orion024-vbox para la descarga como se muestra en la siguiente imagen.

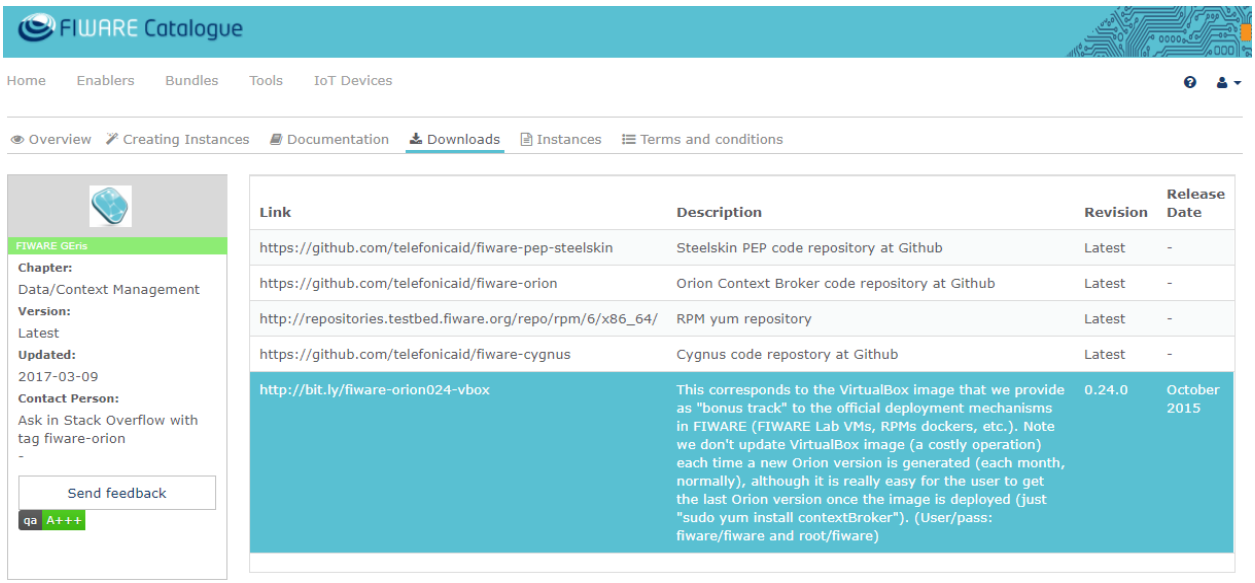


Ilustración 42: Interfaz catálogo FIWARE

Accedemos a la dirección anterior y automáticamente se nos descarga el paquete Orion024-CentOS67.ova, el paquete que contiene la página web a desplegar.

Si no disponemos de VirtualBox en nuestro ordenador, tendremos que acceder a la siguiente web https://www.virtualbox.org/wiki/Downloads y seleccionar la opción que más nos convenga según el SO del ordenador que estemos utilizando.

Una vez llegados hasta aquí, procedemos a desplegar la máquina virtual accediendo al administrador de VirtualBox.

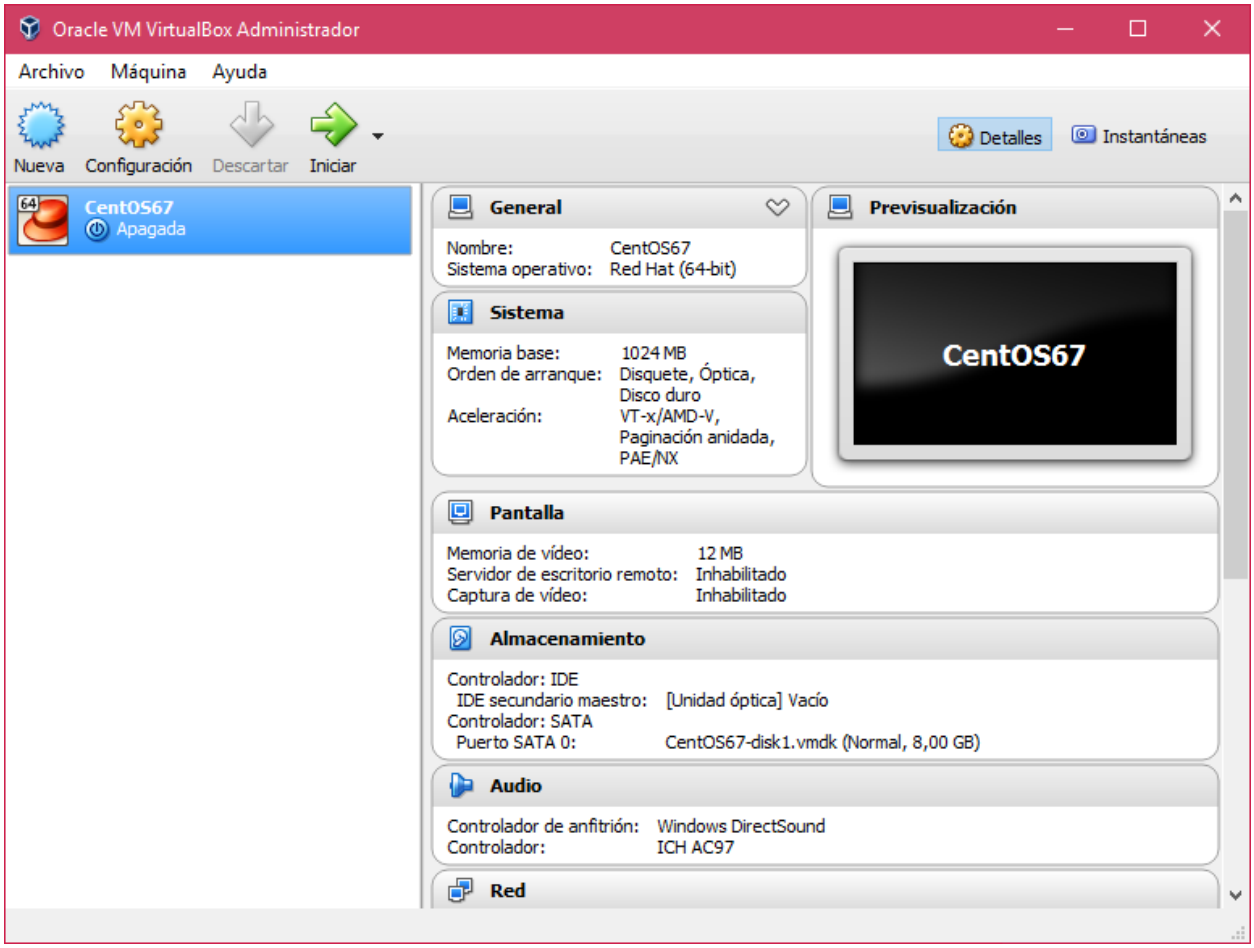


Ilustración 43: Instalación Context Broker (I)

Un fichero OVA contiene los ficheros necesarios para desplegar la máquina (los ficheros del disco virtual (.VMDK o .VHD) y los ficheros .mf (manifest file)) comprimidos con TAR.

Como podemos observar, en nuestro entorno ya lo tenemos desplegado, sin embargo, a continuación, se muestra una guía para el correcto despliegue. Para iniciar el despliegue navegaremos hasta la opción de **Importar Servicio Virtualizado** que nos aparece en el desplegable de **Archivo**.

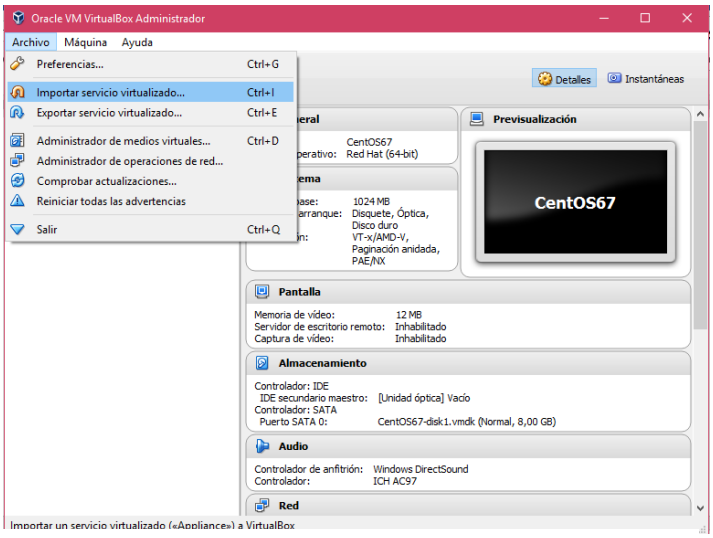


Ilustración 44: Instalación Context Broker (II)

Una vez llegados aquí, seleccionamos el **paquete .ova** descargado anteriormente.

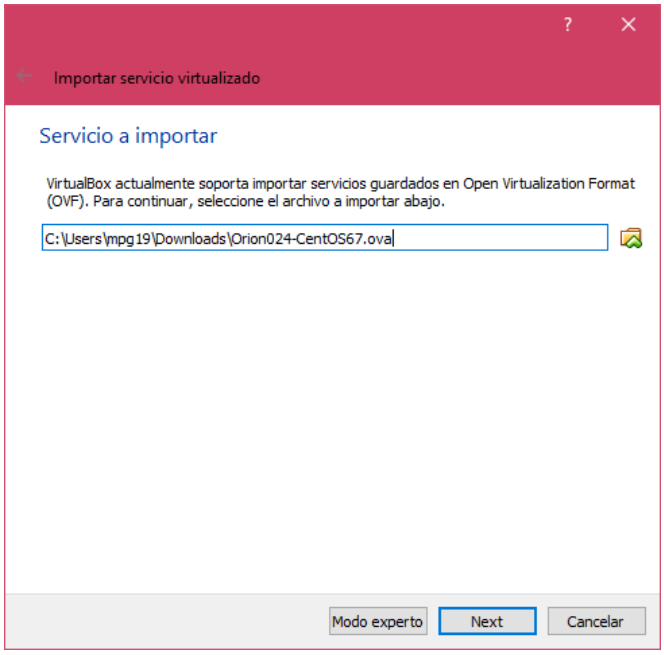


Ilustración 45: Instalación Context Broker (III)

Clicamos en siguiente y dejamos toda la configuración con sus valores por defecto y finalizamos. Depués de unos dos minutos tendremos la máquina desplegada y lista para su uso.

Debido a que necesitamos que la máquina virtual sea parte de nuestra red física para poder acceder a los recursos del context broker de manera directa, tenemos que configurar la conexión de red del equipo virtual en modo ***bridge.*** Para configurar este parámetro simplemente seleccionamos la opción de **Configuración**, posteriormente la pestaña de **Avanzado** y configuramos los valores conforme a la siguiente imagen.

+

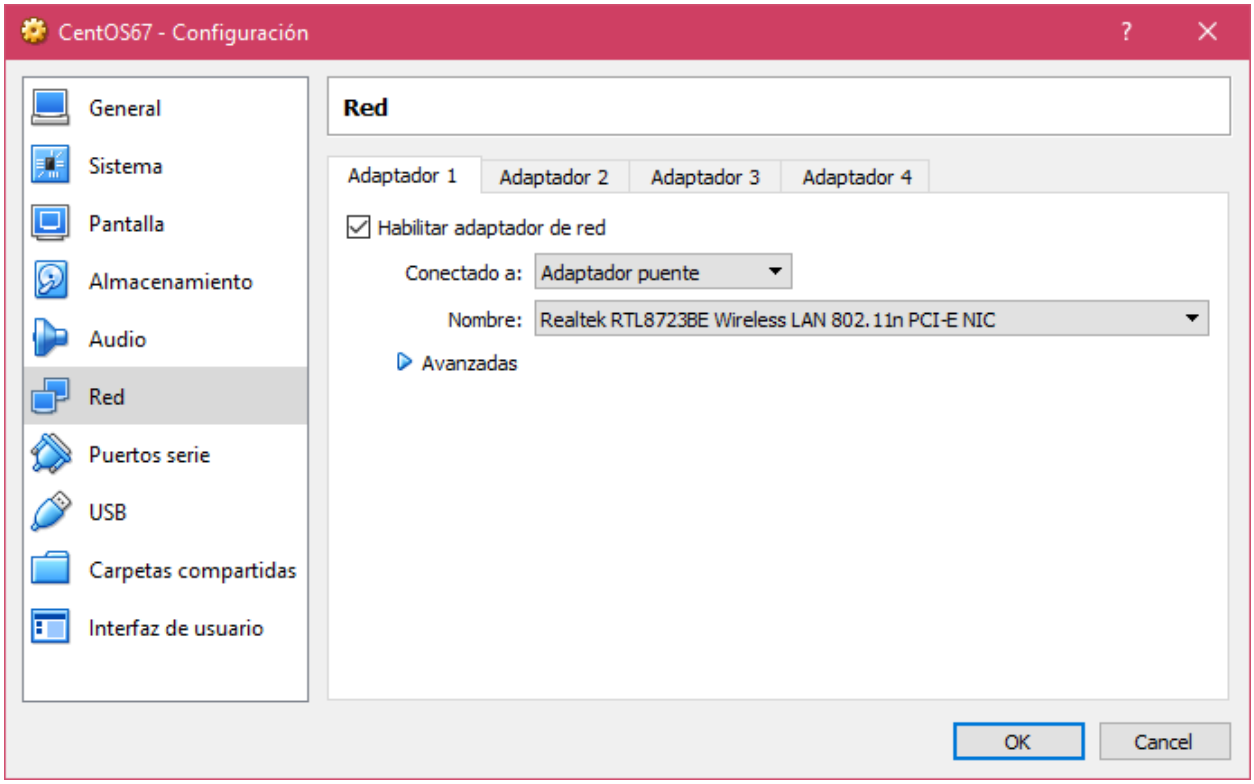


Ilustración 46: Instalación Context Broker (IV)

Ya tendremos lista la máquina virtual para poder usarla correctamente, por tanto, procedemos a encenderla haciendo doble clic sobre ella. Despues de un tiempo de espera la máquina nos solicita un usuario y una contraseña que nos facilita la página donde hemos realizado la descarga (usuario: ***root***, contraseña: ***fiware***).

Solicitamos la ip de la máquina con **`#ifconfig`** donde tendremos que hacer las peticiones POST y por último, activamos el servicio con **`#service contextBroker start`**

ANEXO 2. Manual de usuario

Al iniciar la aplicación nos mostrará la pantalla que visualizamos en la ilustración 43. En ella, podemos abrir el menú clicando sobre el icono de la parte superior izquierda o agregar una nueva pulsera clicando en signo + de la parte inferior.

Ilustración 47: Manual de usuario – Pantalla Android principal

Al pulsar sobre el botón de menú, la aplicación mostrará la siguiente pantalla con las tres opciones de menú que podemos observar.

Ilustración 48: Manual de usuario – Pantalla Android información

- En la opción de **Ajustes** podemos encontrar un enlace directo a las notificaciones generales del teléfono.

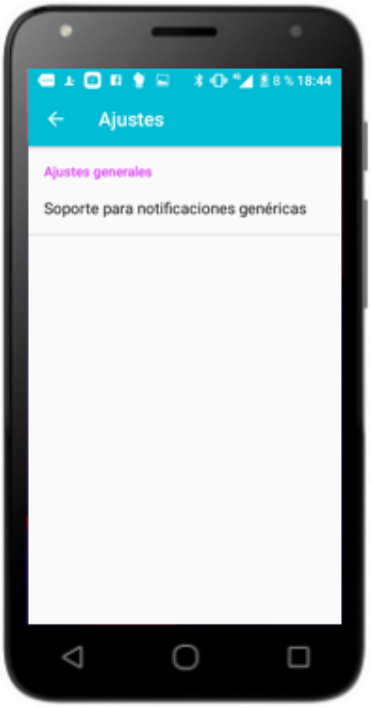


Ilustración 49: Manual de usuario – Pantalla Android Ajustes

- Al pulsar en la opción **Sobre ti**, la aplicación nos mostrará información a rellenar relativa al usuario, como su nombre, su peso, edad, etc…

Ilustración 50: Manual de usuario – Pantalla Android Sobre mi

- Por último, tenemos la opción de salir de la aplicación que cerrará automáticamente la misma.

Volvemos a la pantalla principal y si clicamos en la opción de Agregar nueva pulsera, la aplicación mostrará la pantalla que visualizamos en la ilustración. Mientras esté realizando la detección se nos mostrará el semicírculo que podemos ver en la ilustración 51. A la misma vez que se vaya detectando pulseras MiBand 2 se irán agregando a la lista como muestra en la ilustración 51. La información que podems visualizar es el nombre de la pulsera y la dirección **MAC** de esta.

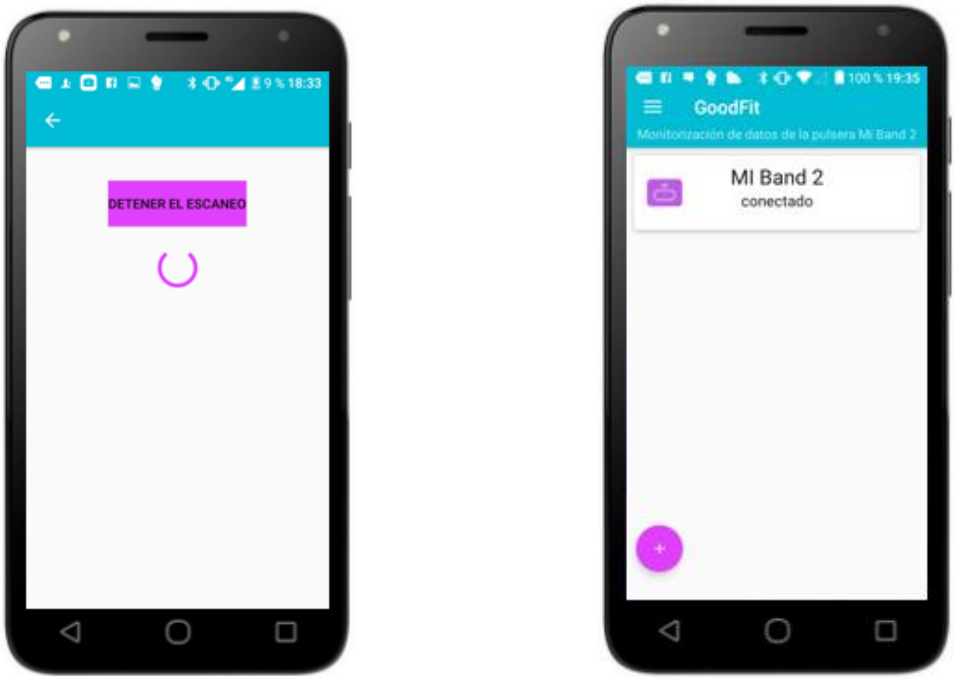


Ilustración 51: Manual de usuario – Pantalla Android detección Smartband

Al clicar sobre el dispositivo detectado comenzará el emperajamiento de la pulsera con el móvil, es decir, la conexión. La pantalla mostrará un gif de un reloj de arena hasta que la conexión sea satisfactoria. Para ello, es importante dejar pulsado el botón táctil de la pulsera cuando esta vibre.

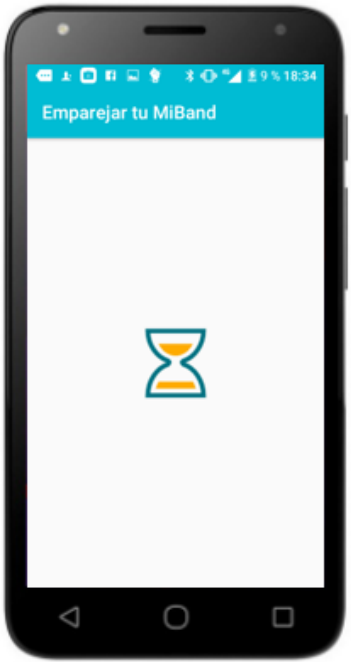


Ilustración 52: Manual de usuario – Pantalla Android GIF

Una vez que la conexión se haya realizado correctamente visualizaremos la siguiente pantalla. Podemos comprobar que se trata de la pantalla principal de la aplicación.

Ilustración 53: Manual de usuario – Pantalla Android smartband conectada

Si clicamos en el dispositvo conectado, accederemos a las funcionalidades de la aplicación. Para navegar entre las distintas funcionalidades clicaremos en los botones que se muestran en la parte superior de la pantalla. En la primera pantalla, llamada mediciones recogeremos los datos actuales de la pulsera cuando le demos al botón de actualizar, que se muestra en la esquina superior derecha. Cuando los datos del pulso recogido sean cero se mostrará un aviso tal y como se muestra en la ilustración 55.

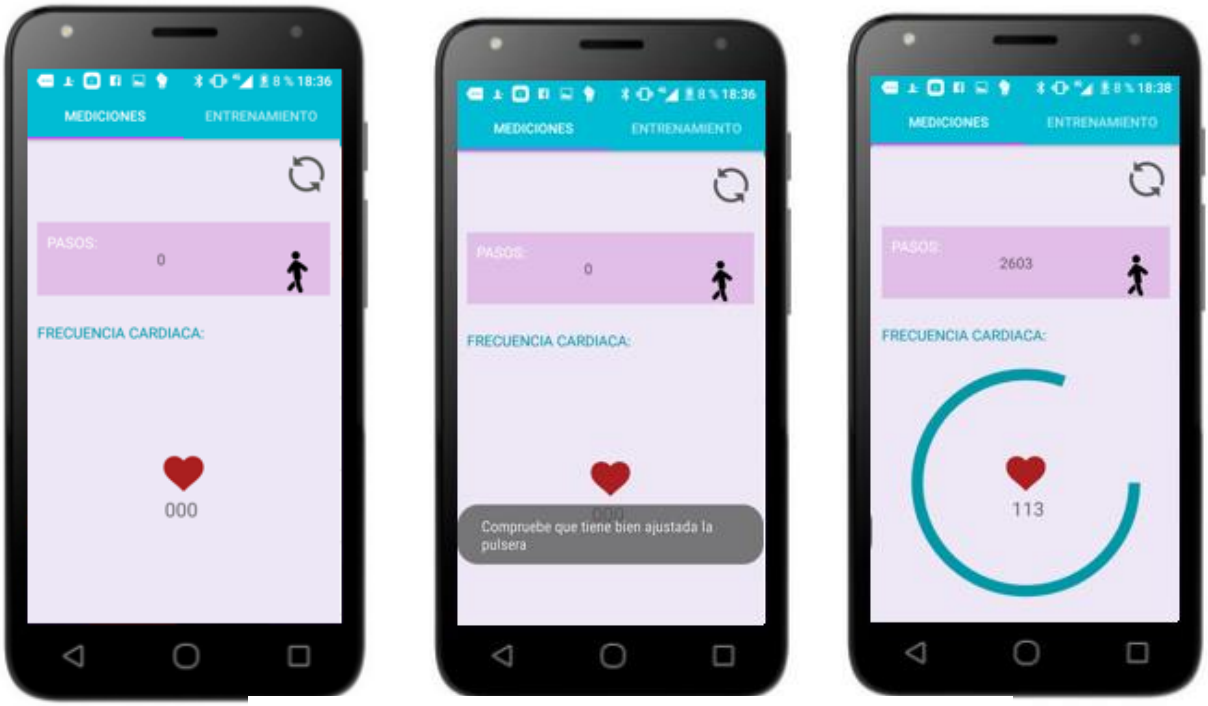


Ilustración 54: Manual de usuario Pantalla Android mediciones

Por otro lado, la opción de entrenamiento permitirá monitorizar nuestra actividad física durante un tiempo. En la pantalla se nos mostrará el tiempo transcurrido desde que pulsamos el botón de Iniciar entrenamiento. Este será el momento en el que se comiencen a recoger mediciones cada 20 segundos tanto de nuestra frecuencia cardiaca como de nuestros pasos. Con las mediciones anteriores y la información proporcionada en la pantalla de Ajustes será suficiente para poder calcular las calorías gastadas.

Ilustración 55: Manual de usuario – Pantalla Android Entrenamiento

Una vez que tengamos una pulsera conectada a la aplicación Android y esta tenga conexión a Internet podemos hacer uso de la aplicación web para visualizar los datos. La pantalla mostrada en la ilustración será la pantalla principal donde podemos encontrarnos una tabla con la información de las pulseras conectadas y la información de su última medición realizada en la opción de **Mediciones** que hemos mencionado anteriormente.

En esta pantalla encontramos dos botones con dos funcionalidades distintas, **Ver entrenamiento** y **Ver gráfico diario.**

Para hacer uso de estos botones tendremos que clicar dentro de la caja que aparece en el lado izquierdo de la celda de la pulsera que queramos consultar. Si clicamos en cualquier botón sin haber seleccionado antes alguna de las pulseras, la aplicación nos mostrará una advertencia tal y como se puede ver en la ilustración 60.

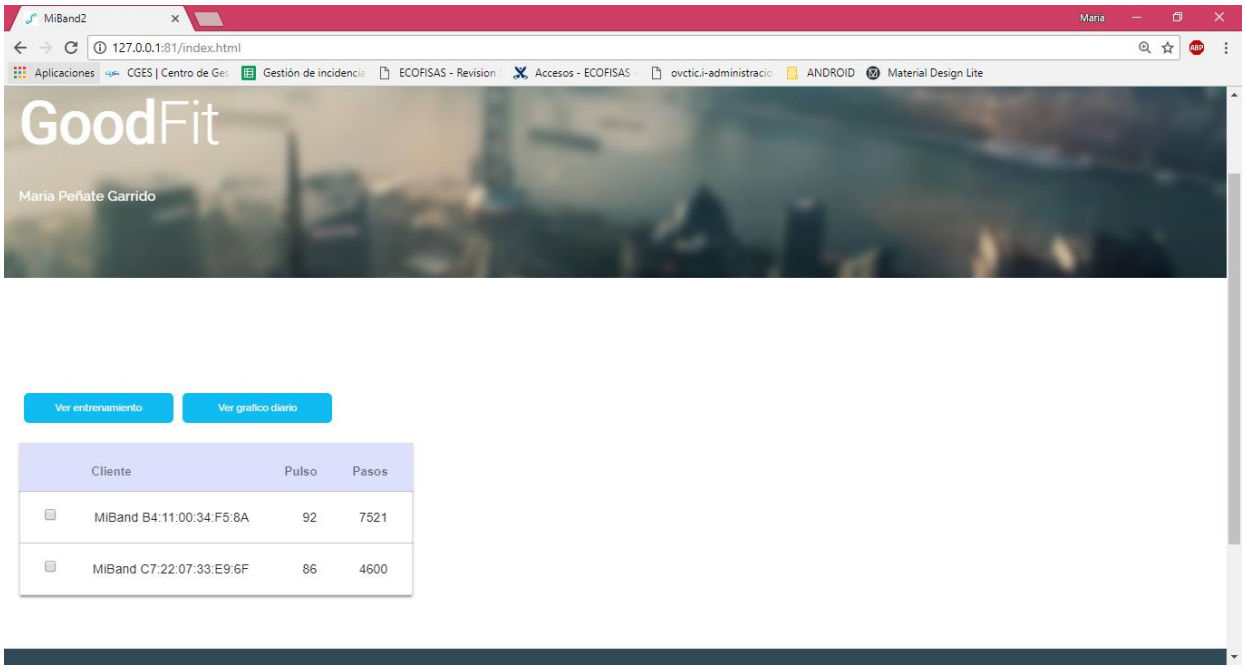


Ilustración 56: Manual de usuario – Pantalla principal web

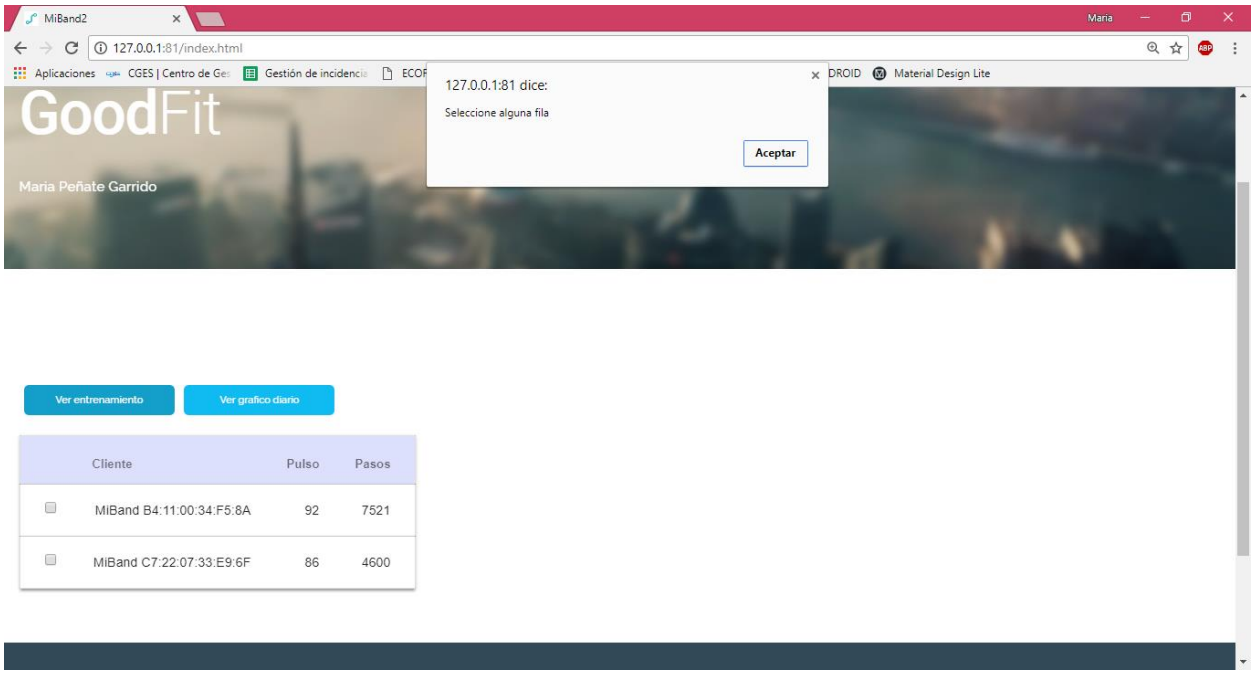


Ilustración 57: Manual de usuario – Pantalla error de selección web

Si por el contrario seleccionamos correctamente la pulsera, podremos hacer uso de ambos botones.

- Clicando en la opción de **Ver gráfico diario** la aplicación mostrará un gráfico en la parte derecha de la pantalla. Esta muestra las mediciones de frecuencia cardíaca realizadas por el usuario durante un día indicando la hora en la que se midió. Podemos ver un ejemplo de ello en la ilustración 59.

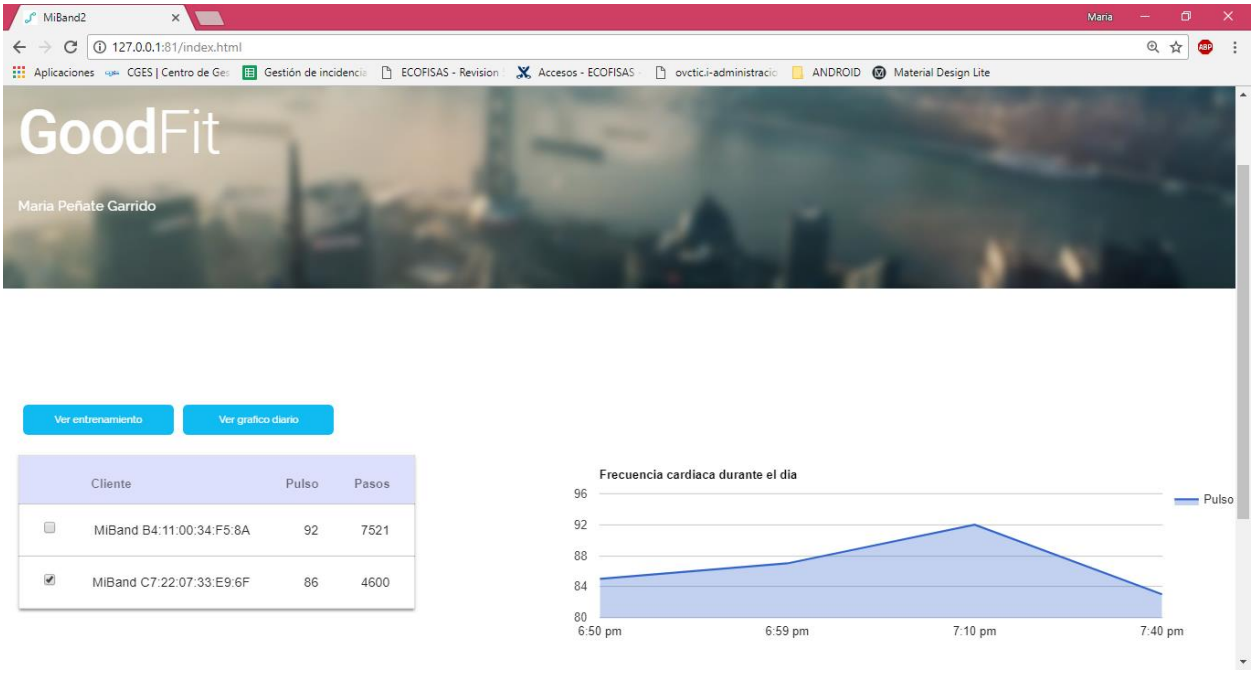


Ilustración 58: Manual de usuario – Pantalla gráfico web

- Por el contrario, si clicamos en **Ver entrenamiento** se nos mostrará una tabla con los entrenamientos realizados por el usuario. Indicando los pasos, calorías, el último pulso registrado, la duración, el estado del entrenamiento, es decir, si está finalizado o no y la fecha de realización. Podemos ver un ejemplo de ello en la ilustración 60.

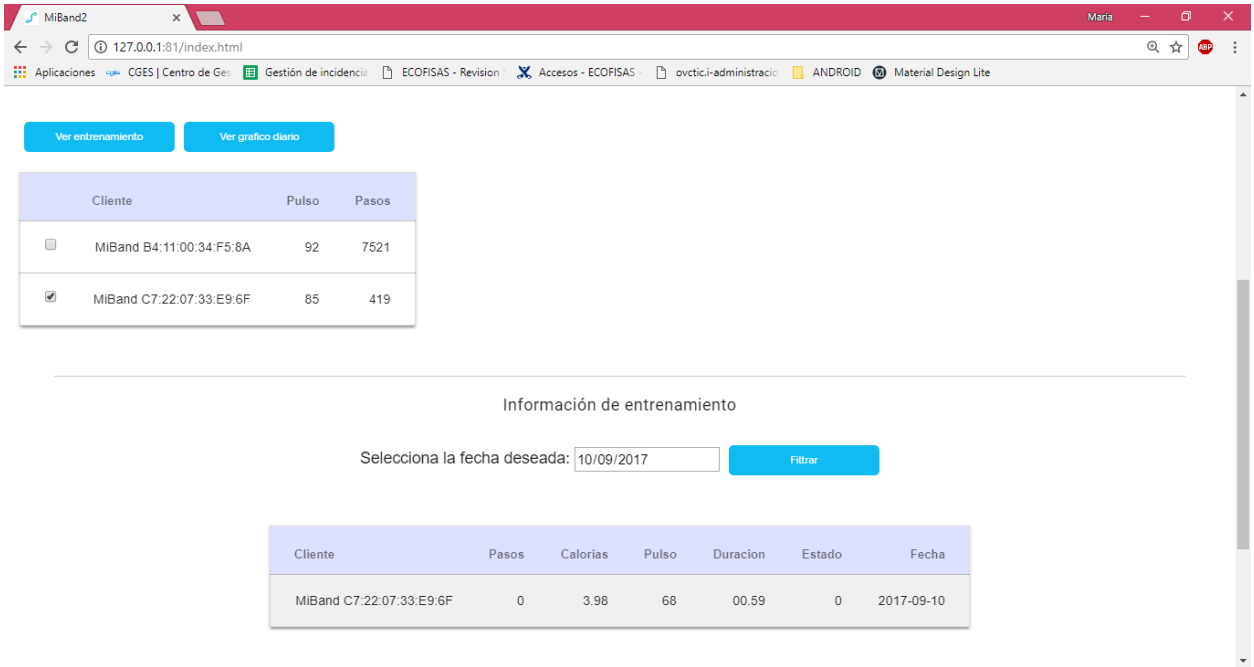


Ilustración 59: Manual de usuario – Pantalla entrenamiento web

10 REFERENCIAS

[1] R. Muñoz, «Código Nexo,» [En línea]. Available: http://www.codigonexo.com/blog/actualidad-blog/que-son-los-wearables/.

[2] G. García Sánchez, «Monitorización de pacientes con pulseras inteligente,» *TFG,* 2016.

[3] P. Rojas, «idx,» [En línea]. Available: https://idx.com.co/sabes-qu%C3%A9-es-iot-ecosystems-y-c%C3%B3mo-puede-mejorar-tu-vida-ab413f564e54.

[4] J. Rabanales Sotos, «Telemedina,» *Revista Clínica Médica Familiar,* vol. 4, nº 1, pp. 42-48, 2010.

[5] C. M. Cervantes Guijarro, «Uso de las nuevas tecnologías y telemedicina en el seguimiento de recién nacido sano,» *Revista Pediátrica de Atención Primaria,* vol. 16, pp. 305-310, 2014.

[6] «Bluetooth,» [En línea]. Available: https://www.bluetooth.com/specifications/gatt/generic-attributes-overview.

[7] «Gadgebridge,» [En línea]. Available: https://github.com/Freeyourgadget/Gadgetbridge.

[8] S. HEBUTERNE, Android: guía de desarrollo de aplicaciones Java para smartphones y tabletas, Eni Ediciones, 2016.

[9] «muyfitness,» [En línea]. Available: https://muyfitness.com/cuantas-calorias-queman-hechos_7181/.

[10] V. Rampérez Martín, «Gestión del uso de aplicaciones y servicios en frameworks de negocio digital,» 2016.

[11] «FIWARE,» [En línea]. Available: https://www.fiware.org/.

[12] S. Juzaino Zarate, «Monitoreo de Temperatura Ambiental del Laboratorio de Mecatrónica del ITN, basados en la Arquitectura Básica de Fiware para IOT (Internet Of Things),» *Artículo Revista de Sistemas Computacionales y TIC's,* vol. 2, nº 6, pp. 36-44, 2016.

[13] «HTTP,» [En línea]. Available: https://es.wikipedia.org/wiki/Protocolo_de_transferencia_de_hipertexto.

[14] «FIWARE - Orion,» [En línea]. Available: https://fiware-orion.readthedocs.io/en/master/user/walkthrough_apiv1/index.html.

[15] «HTML,» [En línea]. Available: https://es.wikipedia.org/wiki/HTML.

[16] «JavaScript,» [En línea]. Available: https://es.wikipedia.org/wiki/JavaScript.

[17] «PHP,» [En línea]. Available: https://es.wikipedia.org/wiki/PHP.

[18] «AJAX,» [En línea]. Available: https://es.wikipedia.org/wiki/AJAX.

[19] «Manual PHP,» [En línea]. Available: http://php.net/manual/es/.

[20] F. Rives Hurtado, «Mis cosas, inquietudes y demás,» [En línea]. Available: http://www.franriveshurtado.com/blog/advanced-rest-client-para-probar-api/.