

ULL

Universidad
de La Laguna

Escuela Superior de
Ingeniería y Tecnología



Trabajo de Fin de Grado

Grado en Ingeniería Informática

Open Smartband

Open Smartband

José Yeray Díaz Rodríguez

La Laguna, 05 de septiembre de 2017

D. **Jesús Alberto González Martínez**, con N.I.F. 43779378-M profesor Colaborador de Universidad adscrito al Departamento de Ingeniería Informática y de Sistemas de la Universidad de La Laguna, como tutor

C E R T I F I C A

Que la presente memoria titulada:

“Open Smartband.”

ha sido realizada bajo su dirección por D. **José Yeray Díaz Rodríguez**, con N.I.F. 54052235-G.

Y para que así conste, en cumplimiento de la legislación vigente y a los efectos oportunos firma la presente en La Laguna a 05 de septiembre de 2017.

Agradecimientos

A la Universidad de La Laguna y especialmente a la Escuela Superior de Ingeniería y Tecnología por haberme permitido iniciar mi futuro profesional, así como a todos los docentes y compañeros, por los conocimientos y ayuda transmitida.

A Jesús Alberto por su labor como tutor de Trabajo de Fin de Grado , por su ayuda, dedicación y capacidad para la resolución de problemas, búsqueda de alternativas y motivación para cumplir con los objetivos.

A Pedro, Luis, Jonay, Juanjo, Víctor, Alfonso y Borja por todo su apoyo y dedicación para que jamás me rindiera y cumpliera mi objetivo, convirtiéndose en más que compañeros de universidad en amigos para toda la vida.

Y en especial, por su inagotable esfuerzo durante todos estos años que me han dado siempre fuerzas para no rendirme y lograr llegar a la meta, a mi familia: mis padres, los que admiro, mis tíos Daniel y Pino, mi primo Alexander y mi pareja Carla, mi principal apoyo.

Licencia



© Esta obra está bajo una licencia de Creative Commons Reconocimiento 4.0 Internacional.

Resumen

*Este proyecto está enmarcado en el ámbito de los llamados “sistemas vestibles”, comúnmente llamados **sistemas wearables**. Tiene como objetivo, crear una pulsera open source para que los usuarios puedan fabricar y personalizar según sus necesidades su propia **pulsera de actividad**, y productos derivados del uso de smartbands o pulseras de actividad de código abierto como fuente principal de obtención de datos, ya sea para el ámbito personal, deportivo o médico.*

*Esto se consigue principalmente a través del uso de dos tecnologías, por un lado la placa de desarrollo hardware abierto, **arduino** y por otro el uso de un smartphone.*

*Para alcanzar el objetivo principal de éste proyecto, construir una **smartband**, haciendo uso de una placa arduino y sensores arduino, y una aplicación tanto para **android** como **iOS**, para recibir mediante **bluetooth** los datos recogidos y que puedan ser tratados y visualizados en el smartphone.*

En el futuro, la base de este proyecto podría ser usada como sistema recomendador experto para la creación de rutinas de entrenamiento, calentamiento, estiramiento y alimentación , o incluso como sistema médico de monitorización de constantes y valores corporales.

Palabras clave: sistemas wearables, pulsera de actividad, arduino, smartband, android, iOS, bluetooth.

Abstract

This project is framed within the scope of so-called “wearable systems”. This project’s purpose was to create an open source wristband so that users can manufacture and customize their own activity bracelet. Products derived from the use of open source activity bracelets, as the main source of data collection, whether for personal, sports or medical use.

This is achieved through the use of two technologies, an open hardware development arduino board and a smartphone.

Using an arduino board and arduino sensors, data is collected through Bluetooth and received by an application for both. Android and iOS, processes and displays on a smartphone.

In the future, the basis of this project could be used as an expert recommendation system for creating training, warm-up, stretching and dietary routines, or even as a medical system monitoring patient vitals.

Keywords: wearable systems, activity bracelet, *arduino*, *smartband*, *iOS*, *android*, *Bluetooth*.

Índice General

Capítulo 1. Introducción.	1
1.1 Antecedentes y estado del arte.	1
1.2 Terminología.	2
1.2.1 Wearable.	2
1.2.2 Arduino.	2
1.2.3 Open source.	3
1.3 Objetivos y tareas a realizar.	5
1.4 Tecnologías empleadas.	7
Capítulo 2. Diseño de la solución.	11
2.1 Arduino.	11
2.1.1 Sensores.	12
2.1.2 Conexiones.	14
2.1.3 Implementación.	16
2.2 Aplicación móvil:	17
2.2.1 Ionic 2.	17
2.2.2 Ionic native Bluetooth serial.	17
2.2.3 Sqlite.	19
2.2.4 ChartJS.	20
Capítulo 3. Aplicación Open Smartband.	21
3.1 Conexión.	22
3.2 Dashboard.	22
3.3 Alertas.	23
Capítulo 4. Resultados	24
4.1 Arduino.	24
4.2 Datos arduino.	25
4.3 Open Smartband.	26
Capítulo 5. Conclusiones y líneas futuras	34
Capítulo 6. Summary and Conclusions	36
Capítulo 7. Presupuesto.	38

7.1 Presupuesto total.....	39
Apéndice A. Repositorio del proyecto.	40
Bibliografía	41

Índice de figuras

Figura 1 : OpenSource	4
Figura 2 : Cuota de mercado Android.....	5
Figura 3 : Arduino NANO.....	12
Figura 4 : Mapa de conexiones arduino	15
Figura 5 : Diseño aplicación android.	21
Figura 6 : Arduino Smartband.....	24
Figura 7 : Consola de paquete de datos Arduino	25
Figura 8 : Pulsaciones en reposo	26
Figura 9 : Vista home Open Smartband	27
Figura 10 : Escaneando dispositivos	28
Figura 11 : Dispositivos disponibles	29
Figura 12 : Pestaña de visualización de datos	30
Figura 13 : Datos sincronizados	31
Figura 14 : Visualización de gráficas	32
Figura 15 : Configuración de alertas.....	33

Índice de tablas

Tabla 1 : Resumen de gastos..... 39

Capítulo 1. Introducción.

1.1 Antecedentes y estado del arte.

No es un secreto que en el mundo el dispositivo móvil más vendido es el Smartphone, que se ha convertido en el centro de la comunicación y el ocio personal.

Desde 2014, hay otra tecnología complementaria totalmente en auge de dispositivos, gafas, relojes, pulseras o anillos, todos etiquetados como “inteligentes”, pretenden aprovechar la tecnología y llevar el uso de estos objetos cotidianos un poco más allá. Se conocen con el término anglosajón “wearable”, tecnologías que se llevan puestas, como una prenda o un complemento.

Sin duda uno de estos dispositivos más comunes en el mercado, son las smartband, se pueden encontrar diferentes marcas y modelos.

Cuando se planteó este proyecto, la idea principal, era utilizar una de ellas, y crear un sistema experto que realizará sugerencias sobre alimentación, rutinas de entrenamiento, etc. ... obteniendo la información de entrada los datos de dicho dispositivo. Pero para sorpresa encontrar una smartband con una API abierta, que se pudiera utilizar no era una misión fácil, y en algunos casos mucho menos barata.

Fue entonces cuando surgieron una batería de preguntas:

- > ¿Por qué no hacer una?
- > ¿Y si hay más gente que quiera hacer este tipo de proyectos pero se encuentran con la misma problemática?
- > ¿Y si añadiera sensores diferentes a los que hay en el mercado?
- > ¿Y si pudiera añadirle una funcionalidad nueva a las smartbands?

Para dar respuestas a todas esas preguntas y abordar una solución a esta carencia en el mercado nació este proyecto. La idea es crear una smartband open source y una aplicación móvil que sirvan como primer eslabón para conseguir un mejor dispositivo open source que los que ya existen en el mercado.

1.2 Terminología.

1.2.1 Wearable.

El concepto wearable quizás sea el más importante de los conceptos a tratar en el proyecto. Hace referencia al conjunto de aparatos y dispositivos electrónicos que se incorporan en alguna parte de nuestro cuerpo interactuando continuamente con el usuario y con otros dispositivos con la finalidad de realizar alguna función específica. Tiene como raíz la lengua inglesa, cuya traducción significa “llevable” o “vestible”, en el argot tecnológico hace referencia a ordenadores corporales o llevables.

Aunque se puede datar los orígenes de la tecnología en la década de 1970, no ha sido hasta la década del 2010 cuando ha evolucionado lo suficiente para poder atraer un amplio abanico de consumidores. Como fecha clave, cabe destacar la feria internacional de consumo electrónico CES del año 2014, cuando las grandes compañías hacen la presentación oficial de su apuesta por esta tecnología.

La tecnología wearable se halla presente en un extenso abanico de campos que satisfacen nuestras necesidades, con la finalidad de mejorar nuestra calidad de vida, estando muy presente en la salud de los pacientes, la seguridad de las personas que se exponen a determinados peligros en su trabajo diario, el adiestramiento de atletas que se preparan para una competición, etc...

El futuro de la tecnología pasa por esta nueva concepción de ordenadores que nos acompañan a todas partes, en un futuro próximo veremos teléfonos móviles que se encuentran implantados en nuestras orejas, o exoesqueletos robóticos que nos facilitaran el trabajo que anteriormente nos producía grandes esfuerzos y fatigas ... Sin duda la tecnología wearable ha llegado para quedarse entre nosotros.

Para el desarrollo del sistema wearable, la smartband, no sería posible, sin arduino.

1.2.2 Arduino.

Arduino es una plataforma de prototipos electrónica de código abierto (open source) basada en hardware y software flexibles y fáciles de usar. Está pensado para artistas, diseñadores, como hobby y para cualquier interesado en crear objetos o entornos interactivos. Arduino puede sentir el entorno mediante la recepción de entradas desde una variedad de sensores y puede

afectar a su alrededor mediante controles de luces, motores y otros artefactos.

Hay muchos otros microcontroladores y plataformas disponibles para computación física. Arduino simplifica el proceso de trabajo con microcontroladores, pero ofrece algunas ventajas sobre otros sistemas para profesores, estudiantes y aficiones interesados:

- Barato: las placas arduino son relativamente baratas comparadas con otros plataformas.
- Multiplataforma: el software de arduino puede ser ejecutado sobre sistemas operativos Windows, macOS y GNU/Linux.
- Entorno de programación simple y claro: el entorno de programación es fácil de usar para principiantes, pero suficientemente flexible para que usuarios avanzados puedan aprovecharlo también.
- Código abierto: El software de arduino está publicado como herramienta de código abierto, disponible para extensión por programadores experimentados. El arduino está basado en microcontroladores ATMEGA8 y ATMEGA168 de Atmel. Los planos para los módulos están publicados bajo licencia Creative Commons, por lo que diseñadores experimentados de circuitos pueden realizar su propia versión.

Su origen se remonta a 2005, como un proyecto para acercar los micro controladores a los estudiantes, en Italia. Hoy en día se han convertido en una de las primeras opciones open source para el desarrollo hardware.

1.2.3 Open source.

El open source, o código abierto, es el software desarrollado y distribuido libremente. Se focaliza más en los beneficios prácticos (acceso al código fuente) que en cuestiones éticas o de libertad que se destacan en el software libre.

La idea del código abierto se centra en la premisa de que al compartir el código, el programa resultante tiende a ser de calidad superior al software propietario, es una visión técnica.

Al igual que el software libre, el código abierto u open source tiene una serie de requisitos que los podemos resumir en este diagrama.

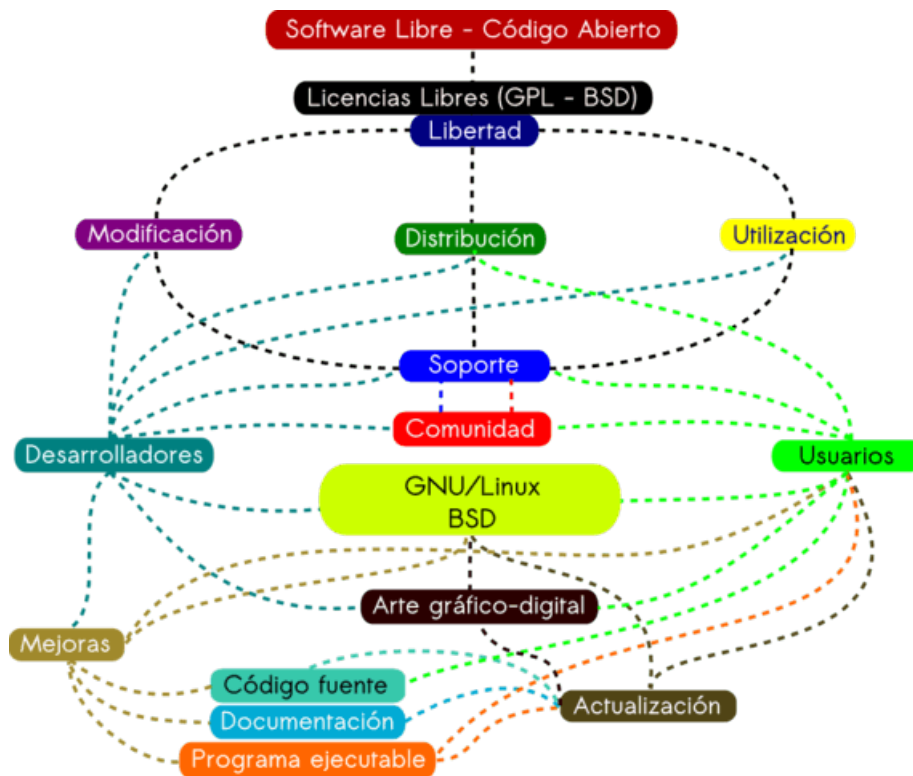


Figura 1 : OpenSource[29]

Basado en el open source se encuentra el sistema operativo Linux, creado por Linus Torvalds en 1991, que es la base de android sistema operativo para el cual desarrollaremos nuestra aplicación móvil.

Android es un sistema operativo libre, gratuito y multiplataforma que fue diseñado para dispositivos móviles con pantalla táctil, tablets, relojes o televisores. Fue creado por la empresa Android Inc. , empresa que respaldó Google económicamente, y más tarde en 2005 compró.

Android es el sistema líder del mercado mundial, la cuota de mercado varía según el país, en nuestro país la cuota de mercado es de un 92 por ciento para android y un 7,5 por ciento para iOS, porque éstos son argumentos más que suficientes para que el desarrollo de nuestra aplicación fuera enfocada al mercado android.



Figura 2 : Cuota de mercado Android[30]

1.3 Objetivos y tareas a realizar.

Con el desarrollo de este proyecto se quiere abrir un camino open source en los sistemas wearables y sobre todo en las smartbands. Para satisfacer este proyecto los objetivos que se han definido son:

- Crear una smartband funcional
- Conectar la smartband con un dispositivo móvil
- Crear base de datos
- Obtener datos de la base de datos
- Crear aplicación móvil

En un futuro con todas las contribuciones al proyecto, quizás podamos conseguir una pulsera de actividad que ofrezca mucha más información útil y personalización de lo que podemos obtener hoy en día al comprar una pulsera de actividad comercial. Y que se promueva el uso de las **smartband** a nivel médico, para el control y seguimiento de las variables vitales del cuerpo humano, y que sirva de apoyo para los profesionales y amateur del mundo deportivo que tanto necesitan medir y cuantificar su trabajo, su esfuerzo y su progreso.

Para el desarrollo de este proyecto, el primer paso ha sido documentarse sobre metodologías, tecnologías, herramientas y lenguajes de programación a utilizar, para tener los conocimientos necesarios para poder enfrentarse

con las capacidades oportunas a este reto que se presenta. Por todo ello a continuación se detallan los objetivos a cumplir para llevar a cabo esta empresa:

- Documentación:
 - Analizar y aprender el uso de un sistema arduino, tanto a nivel de hardware como software.
 - Analizar el uso de un sensor de aceleración y giroscopio para arduino.
 - Analizar la incorporación de un sensor de pulso al sistema arduino.
 - Analizar el desarrollo de la incorporación de un sensor de temperatura corporal a arduino.
 - Analizar una solución para incorporar un módulo bluetooth a arduino.
- Desarrollo:
 - Implementar el uso de un sensor de aceleración y giroscopio para arduino.
 - Implementar la unión de un sensor de pulso al sistema arduino.
 - Implementar el desarrollo de la incorporación de un sensor de temperatura corporal a arduino.
 - Desarrollar una solución para incorporar un módulo bluetooth a arduino.
 - Desarrollar un programa en arduino que nos permitiera la recogida de todos los datos de los diferentes sensores y los empaqueta en un JSON que pudiéramos enviar por bluetooth.
 - Implementar el desarrollo para enviar un paquete de datos por el módulo bluetooth.
 - Desarrollar un componente en android para buscar y conectarse a un módulo bluetooth para poder leer datos.
 - Desarrollar un componente en android para leer los datos recibidos por bluetooth.
 - Desarrollar una base de datos en android para almacenar cada uno de los mensajes recibidos.
 - Desarrollar un aplicación android para consumir los datos de la base de datos, procesarlos y mostrarlos de forma entendible y útil para el usuario final.

- Pruebas:
 - Probar los datos enviados por los sensores arduinos con un sniffer bluetooth.
 - Probar la conexión entre arduino y el smartphone mediante bluetooth.
 - Probar la recepción de los datos desde nuestra aplicación móvil.
 - Comprobar el correcto almacenamiento de los datos en nuestra base de datos.
 - Comprobar el funcionamiento de la recopilación de datos desde nuestra base de datos.
 - Comprobar el funcionamiento de la aplicación móvil y el visualizado correcto de los datos.

Para llevar a cabo todos estos objetivos, el desarrollo se ha realizado en dos fases, por un lado la fabricación e implementación de la smartband con arduino, y por último, el desarrollo de la aplicación android.

1.4 Tecnologías empleadas.

A continuación se detallan todas las herramientas, lenguajes y utilidades que han hecho posible la realización de este proyecto, así como la organización y comunicación entre el alumno y el tutor, en dos partes: por un lado se detallarán las herramientas y utilidades para el desarrollo propio del proyecto, y en el segundo lugar todas las demás utilidades necesarias:

- Utilidades para el desarrollo:



Arduino IDE^[16]: es el entorno de desarrollo integrado oficial para la plataforma **Arduino**, de fácil uso, con facilidades para incluir librerías y uso de ejemplos.

Se ha utilizado para el desarrollo y test del programa **arduino**.



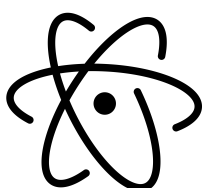
Adafruit^[17]: es una compañía hardware **open-source** que proporciona, entre otras cosas, librerías para componentes **arduino**, a parte de fabricar sus propios componentes.

Se ha utilizado las librerías, como modelo de referencia y sus ejemplos para aprender a utilizar los diferentes sensores.



Android Studio^[18]: es el IDE oficial para android. Es potente y fácil de usar y permite hacer un buen trabajo de análisis y debugging en el proceso de desarrollo.

Utilizado para comprobar el funcionamiento de nuestra de aplicación, a través del uso de consola, y del depurador.



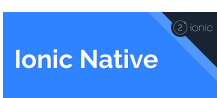
Atom^[19]: es un editor de código abierto con soporte para plug-ins escrito en Node.js. Incrustando git control, desarrollado por GitHub.

Editor donde se ha desarrollado toda la implementación de nuestra aplicación móvil.



SQLite^[20]: es un sistema de gestión de base de datos relacional, contenido en una relativamente pequeña biblioteca escrita en C. Es de las más usadas para implementar base datos en aplicaciones móviles, sobre todo en ionic.

Ha sido el sistema de basado utilizado para incluir una base de datos en nuestro proyecto. Por su ligereza y su plugin para ionic lo hacen una herramienta casi esencial en los proyectos con ionic.



Ionic Native^[21] - Bluetooth Serial: plugin de ionic para comunicaciones bluetooth con dispositivos **android** e **iOS**.

Ha sido el plugin utilizado para poder conectar nuestro módulo bluetooth de arduino con nuestra aplicación android.



Ionic^[22]: es un framework MVC(model-view-controller), gratuita y open source para el desarrollo de aplicaciones híbridas basadas en html5, CSS y JS, está construido con SASS y optimizado con Angular JS. En el caso de este proyecto, utilizamos Ionic 3 y angular2.

Es el framework elegido para el desarrollo de nuestra aplicación android, por facilidad de uso y comodidad, y también porque ofrece la posibilidad de realizar un

aplicación para android e ios con la misma implementación desarrollada.



Angular 2^[23]: framework para aplicaciones web de typescript de código abierto, mantenido por google, que se utiliza para crear y mantener aplicaciones web de una sola página.

Es el framework de desarrollo incluido en los proyectos ionic, y ha sido también uno de los argumentos de peso por el que nos hemos decantado por ionic para el desarrollo de nuestra aplicación, por el uso de angular.



TypeScript^[24]: es un lenguaje de programación libre y de código abierto desarrollado y mantenido por Microsoft. Es un superconjunto de JavaScript, que esencialmente añade tipado estático y objetos basados en clases.

Lenguaje de desarrollo utilizado durante la implementación de nuestra aplicación móvil.



Arduino^[25]: está basado en c++ y aunque la referencia para el lenguaje de programación está en la web oficial de arduino, también es posible usar comandos estándar de c++ para la programación de arduino.

Lenguaje de programación utilizado durante el desarrollo de nuestro smartband.



SQL^[26]: es un lenguaje específico que da acceso a un sistema de gestión de bases de datos relacionales que permite especificar diversos tipos de operaciones en ellos.

Lenguaje utilizada para la configuración y gestión de los datos de nuestra base de datos sqlite.

- Herramientas de realización y organización:



GitHub^[27]: es una plataforma de desarrollo colaborativo para alojar proyectos utilizando el sistema de control de versiones Git. Utiliza el framework ruby on rails.

Lo hemos utilizado como repositorio online para el uso de git como sistema de control de versiones, por comodidad y facilidad de uso, y porque será el lugar donde luego quede el proyecto alojado y documentado para toda la comunidad open source.



Trello^[28]: herramienta de gestión de proyectos, da una perspectiva de todas las tareas, las realizadas, las que están en ejecución y las que quedan por hacer.

Ha sido la herramienta para la organización, y control de las tareas por parte del alumno para el desarrollo del proyecto.



Google Drive: es el sistema cloud de almacenamiento y uso compartido de datos de google.

Herramienta utilizada para el uso compartido de documentación entre el alumno y el tutor durante la ejecución de nuestro proyecto.

Capítulo 2.

Diseño de la solución.

Como se ha citado anteriormente, para acometer ese proyecto, se ha necesitado organizar en dos fases, por un lado el diseño e implementación de la smartband y por otro lado el desarrollo de la aplicación móvil. En este capítulo se explicarán todas las fases y partes del desarrollo del proyecto.

2.1 Arduino.

La primera fase del desarrollo del proyecto ha sido la parte hardware. Lo primero ha consistido en un análisis del mercado, de los diferentes modelos de arduino disponibles y tomar la decisión de cuál era el más adecuado para este proyecto. Me gustaría mencionar, que sin duda ésta ha sido la parte más difícil para mí, debido sobre todo, a que el hardware es lo que más se aleja de mi itinerario de especialización, y que nunca antes había tenido contacto con esta tecnología.

Para dar solución a este problema se ha contado con el modelo NANO pro V3.0, por su potencia, disponibilidad de salidas y tamaño parecía la opción que mejor se ajustaba a los requisitos del proyecto. Entre los argumentos para esta elección cabe destacar su tamaño, 0,70" x 1,70", su amplia conectividad de pines, 8 analógicos y 14 digitales, conexión mini USB de serie, botón reset y dos pines de uso exclusivo para comunicaciones con otros dispositivos los pines TX y RX de los que se hablará posteriormente.

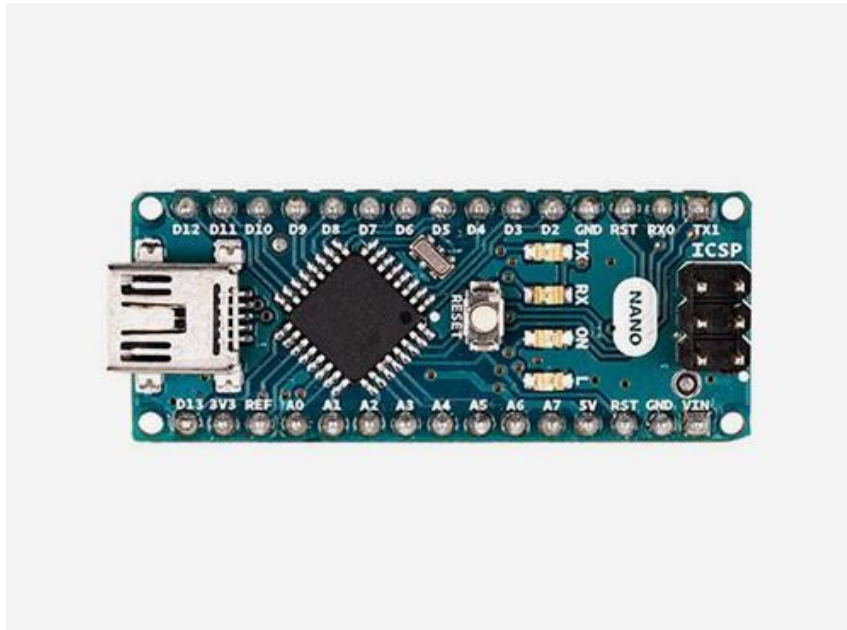


Figura 3 : Arduino NANO

Además se ha tenido que adquirir todos los sensores para poder leer los datos del entorno y obtener la información necesaria. Se ha considerado incluir, un sensor de pulso, es muy común encontrar este tipo de sensores en cualquier smartband, y es una función muy valiosa, no simplemente como dato estadístico para un mejor rendimiento deportivo, sino también para persona con problemas cardíacos que necesiten un mayor seguimiento de esta constante vital; un sensor de temperatura, con este sensor incluido una novedad en el mercado, ya que casi ningún, por no decir ninguno, aunque en nuestro estudio del mercado no se ha encontrado ninguna referencia; un sensor de aceleración y giroscopio, nuevamente es un sensor muy común, ya que es el método más usual para obtener la cantidad de movimiento del individuo; y por último un módulo bluetooth, para poder comunicarse con la aplicación móvil.

2.1.1 Sensores.

- Sensor de pulso: la primera opción fue el sensor MAX30100, ya que según la documentación y los comentarios de los usuarios, poseía una buena calibración, una buena librería para la lectura de los datos y unas muy buenas críticas en cuanto a su usabilidad y estabilidad.

Llegados a este punto se encontró un problema. Tras una semana de pruebas, intentos, ejemplos, librerías y tutoriales ... seguía sin recibir ningún tipo de datos. Al final se llegó a la conclusión de que era el propio sensor y se decidió comprar el sensor de pulsesensor.com, era más nuevo, y había menos información en la

red, pero en la página oficial había una buena documentación, y tenía muy buenas críticas.

- Sensor de temperatura: como se ha comentado anteriormente este sensor es una novedad en las smartband, está pensado para conocer nuestro cuerpo y cual es la temperatura normal, para el nivel deportivo, pero sobre todo también para el seguimiento médico en caso de enfermedades.

Para este elemento se ha seleccionado el sensor GY906, que proporciona temperatura corporal, al estar en contacto con el cuerpo y temperatura ambiente, aunque esta última se ha despreciado al procesar los datos porque no se le ha visto la utilidad.

- Sensor acelerómetro y giroscopio: este es el sensor más común para calcular el movimiento, medir los pasos, las horas de sueño, calcular las calorías etc ... Ciertamente es, que no es un sensor diseñado específicamente para calcular los pasos, sino que es el resultado de un gran algoritmo y procesamiento de datos, que en algunos casos llega al punto de estar patentado.

Es aquí, donde ha aparecido otro contratiempo. Cuando se realizó el análisis del problema, se vio que era posible determinar el conteo de pasos de una persona atendiendo a dos de las tres variables del acelerómetro, atendiendo a una variación de un umbral, ya que cuando esas dos mismas variables son muy cercanas a cero y la última es la única que tiene valor, lo que quiere decir es que solo actúa la aceleración de la gravedad, por lo tanto el individuo está parado.

Una vez, metido en el desarrollo se cayó en cuenta que el cálculo de los pasos, no era algo muy trivial, y que casi era un propio trabajo de fin de grado, el implementar un gran algoritmo que despreciara movimientos de la mano, sin moverse el individuo, ajustar los umbrales para contar pasos correctamente ... por lo que la aproximación del conteo de pasos, es bastante pobre, y sería una línea futura a mejorar, ya que se podría incorporar un algoritmo que fuera contando pasos según la variación de los valores del acelerómetro, y añadir una memoria, y que ya estén pre procesados los pasos a la hora de enviar los paquetes por bluetooth.

En el proceso de análisis del proyecto, se decidió incorporar el sensor de aceleración y giroscopio por el gran abanico de posibilidades que ofrecía entre ellas la posibilidad de calcular los pasos, pero también hacer análisis de parámetros del movimiento en busca de un movimiento casi perfecto en un mundo deportivo de competición.

- Sensor bluetooth: el sensor bluetooth elegido fue el HC-05, quizás el sensor más usado y más común para arduino para esta función, y como se quería que fuera estable, se tomó la decisión en base a lo más comercializado y estable del mercado.

A la hora del desarrollo se descubrieron algunas peculiaridades, como por ejemplo que otros sensores, ya ponen el '\n' ellos solos, para indicar el final del paquete de datos, y en este caso se ha tenido que hacer a la hora del procesamiento de datos. También se pensó en ir a por un módulo bluetooth de bajo consumo, LTE, pero priorizamos la confianza de un módulo estable en el mercado, a un modelo más nuevo y menos testeado.

2.1.2 Conexiones.

Con tantos sensores y módulos conectados, cuando se empezó a montar los jumpers y conectarlos con el arduino, se descubrió que no iba a resultar un juego de niños, ya que se tenía que diseñar un mapa de conexiones, porque algunos sensores repetían pines del bus de datos y todos utilizaban el mismo pin de alimentación. El esquema o mapa resultante de configuración de la smartband, es el siguiente:

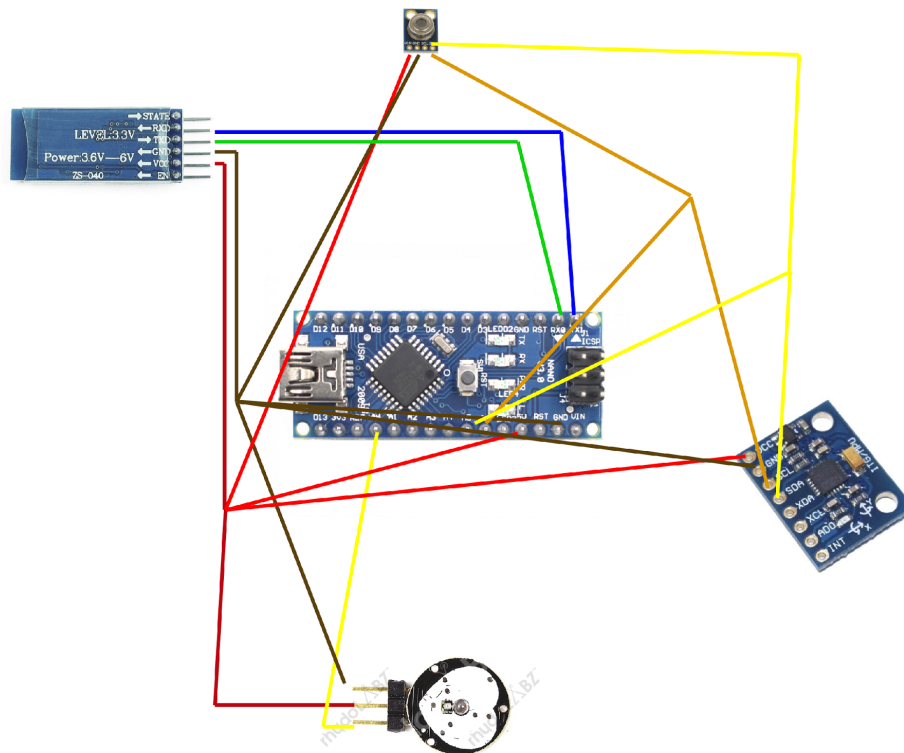


Figura 4 : Mapa de conexiones arduino

Como se puede observar todos los sensores comparte el cable rojo, alimentación, que se conecta al pin V5 de arduino, al igual, que el cable marrón, que es la toma de tierra, que se conecta al pin GND.

Para el sensor de bluetooth, parte superior izquierda, el cable azul conecta RX con TX de placa arduino, y el cable verde, a la inversa TX con RX.

En la parte superior, el sensor de temperatura, conecta SDA con A4 con el cable amarillo, y SCL con el cable naranja con A5, exactamente, de igual forma que el acelerómetro, parte inferior derecha, que utiliza los mismo pines SDA y SCL y se ha tenido que realizar un puente para compartir el mismo bus de datos.

Por último, en la parte inferior, el sensor de pulso, utiliza su pin S con el cable amarillo y lo conecta con el pin A0 de arduino.

2.1.3 Implementación.

El desarrollo del software se ha realizado de forma incremental. De tal manera que se ha ido conectando sensor a sensor, se ha incluido las diferentes librerías disponibles para conocer su funcionamiento, ejecutar los ejemplos, para descubrir el formato en el que llegaban los valores y probar el rendimiento y estabilidad del sensor.

Una vez alcanzado el punto, en que se sabía que librería utilizar para cada uno de ellos, y ya se conocía el funcionamiento de todos los sensores, se decidió a realizar una implementación donde se pudiera leer todos los sensores y mostrar sus valores por consola. Para ello se creó una función por cada sensor donde se almacenan los datos en una variable por sensor, y en la función principal del programa se llama a cada una de esas funciones para obtener el valor de todos los sensores e imprimirlos todos juntos.

Superada esta parte, llegó una parte sensible del proyecto, se tenía que enviar toda esa información, en un único paquete, cuyo formato fuera posible, conseguir en arduino, fuera posible enviar por bluetooth y su posible lectura luego desde la aplicación móvil.

Tras muchas pruebas e intentos, se descubrió cómo realizar un JSON de datos, con cada una de los sensores, en formato clave – valor, y en el caso de los sensores con múltiples valores, por ejemplo el sensor de aceleración que proporciona valores para las coordenadas x, y ,z un formato clave – array valor.

Además de los valores, también se incluye un timestamp del momento en el que se envía el paquete, para ello se ha tenido que añadir una librería también. Puesto que arduino no tiene reloj, en cada ejecución se tiene que hacer un set de la fecha actual, y luego construir un array de valores, donde se almacena, día, mes, hora, minutos y segundos.

De esta manera, el paquete que envía el módulo bluetooth es un JSON de datos, seguido de un “\n” para indicar el final del paquete de datos.

Ejemplo de un paquete de datos:

```
{“temperatura”:24.829987, “pulse”: 519, “accel”: [-108,2352,-9540], “gyro”: [561,-808,-368], “timeStamp”:[10,7,17,56,0] }.
```

2.2 Aplicación móvil:

Cuando se llegó a este punto tocaba decidir cuál era la mejor alternativa para desarrollar la aplicación móvil. Se tenía claro que el objetivo era una aplicación móvil para android, pero no se tenía claro que la mejor opción fuera hacerla nativa, ya que con la existencia de framework que te permiten desarrollar una app y que luego puede ser compilada para android e ios, aunque el objetivo principal fuera android, pensando en el futuro, lo mejor era hacer un aplicación híbrida.

Para continuar se tenía que seleccionar un framework entre los disponibles para realizar nuestra aplicación, quizás los más comunes sean Ionic y React Native. Aquí se abre un gran batalla entre dos grandes frameworks, quizás en busca de un mejor rendimiento la mejor opción era React, pero con conocimientos de la tecnología, se acercaba más Ionic, así que finalmente la opción elegida fue Ionic 2, también un poco como punto de experiencia profesional para enfrentarse y conocer los cambios de angularjs frente a angular 2 y la diferencia de incluir typescript, eran alicientes como para motivarse a apostar y actualizar el repertorio de frameworks y el lenguajes conocidos.

2.2.1 Ionic 2.

El proceso de desarrollo de la aplicación móvil, empezó por crear un proyecto vacío en ionic, y empezar a analizar la estructura, la organización de la información , y se descubrió que el cambio con angular 2 y typescript el cambio era bastante significante.

El siguiente paso, fue incluir en ionic la plataforma de desarrollo para android, y probar a compilar para asegurarse de que al menos el proyecto vacío se lanzaba en el móvil, no se tenía ningún tipo de problema en conflicto de dependencias, versiones y todo iba bien.

Tras obtener un resultado satisfactorio había llegado el momento de empezar a desarrollar nuestra aplicación móvil.

2.2.2 Ionic native Bluetooth serial.

Una vez se tuvo la smartband funcionando y era estable, quedaba la segunda parte, poder recibir todos esos datos en la aplicación y procesarlos.

Se empezó por documentarnos sobre el funcionamiento de ionic native y ver algún tutorial para visualizar el camino a seguir.

La primera tarea a la que afrontar, fue conseguir conectarse con el módulo bluetooth arduino. Esta ha sido, sin duda alguna, la etapa más compleja y difícil del proyecto. En primer lugar, como se ha comentado anteriormente, arduino dispone de dos pines Tx y Rx para conexiones con otros dispositivos, pero en toda la documentación que se encontraba para el envío de datos por el sensor bluetooth nos recomendaba otros dos pines. Por lo que cuando empezábamos a leer paquetes de datos, solo se recibía en código ASCII.

Por lo que se intentó realizar un algoritmo para volver a descifrar el json, pero se hacía una tarea muy larga y tediosa. Así que se decidió hacer un alto en el camino, y volver sobre los pasos, hasta caer en cuenta de que el funcionamiento correcto se conseguía utilizando los pines anteriormente nombrados.

Ahora sí, se podía leer el paquete de datos desde una consola en una aplicación android, era el momento de poder conectarse y leer desde la aplicación.

Se encontró un plug-in ionic para la lectura, se siguió la documentación, y se empezó la implementación del código. Pero por más que lo se intentaba, no se conseguía parearse con el módulo bluetooth. Hasta que se cayó en el error de que el plug-in solo funcionaba para bluetooth LTE, por lo que teníamos que conseguir otro plug-in para bluetooth classic. Se descubrió el plug-in de ionic-native bluetooth serial, el cual fue utilizado para la ejecución del proyecto.

Fácilmente se consiguió buscar los dispositivos bluetooth del entorno, incluso parearse a uno de ellos. Pero no resultó tan fácil leer los datos del bluetooth. No se sabía cuando llegan los paquetes, por lo tanto no se conocía cuando se tenía que abrir la conexión ni cuando escuchar para ver si venían mensajes...

Tras días de pruebas, se cayó en la cuenta de que los paquetes acaban con un "\n" que se ponía al final de cada paquete bluetooth, por lo que ya se tenía un elemento por el que diferenciar cuando terminaba un mensaje. Así que se revisó las funciones de la librería, y la solución fue subscribirse a la función mientras no llegara un "\n" y almacenar todos los datos que llegaran. Y así se consiguió leer los primeros paquetes de datos desde la aplicación móvil.

2.2.3 Sqlite.

Para almacenar los datos, se decidió utilizar Local Storage, el procedimiento era leer un paquete, recuperar lo que había almacenado en el local storage, concatenarlo, y volver a guardarlo. En principio, funcionaba muy bien, rápido y sencillo, sin tener que montar un servicio de base de datos y parecía la opción más sencilla.

Con el tiempo, y sobre todo a la hora de procesar los datos, se cayó en la cuenta de que la aplicación se volvía lenta y poco dinámica, porque se perdía mucho tiempo en el procesamiento de datos, si el número de datos seguía creciendo, la escalabilidad de la aplicación se vería afectada. Por lo que se trató de buscar una solución, pero no se vio una forma de optimizarla, al final se tenía un súper string con muchos samples de mensajes que se tenía que recorrer para recuperar cualquier dato.

Ante esta problemática, se cayó en la cuenta de que de esta manera se podrían perder datos, crear una aplicación muy lenta y poco dinámica, se decidió parar y buscar alternativas de almacenamiento de datos, que pudiera dar más rapidez a la hora de obtener datos, y se decidió elegir sqlite, una base de datos ligera, por toda la potencia de sql y con un plug-in en ionic que iba a ayudar a incorporar este gestor de base de datos al proyecto.

Para utilizarlo, se realizó un servicio, con las funciones, “open data base”, “create data base”, “create table”, y los métodos necesarios para obtener información, y guardar, los métodos getter y setter.

La estructura de base es sencilla, se tiene una tabla llamada btsamples, donde se almacena los mensajes directamente que recibimos de arduino, en los campos, temperatura, pulso, ax, ay, az, gx, gy, gz y timestamp ; y además se tiene otro tabla para contador de pasos, con los campos step y timestamp, que almacena los pasos cada fecha según hayan sido procesados.

Este cambio llevó algunos días, entre configuración, implementación y prueba – error, pero una vez incorporada la base de datos al proyecto, la fluidez de la aplicación fue considerable, en datos se pasó de tardar unos 2 segundos de media en obtener los datos de pulso, temperatura y pasos actuales, a tenerlos en medio segundo de media, por lo que estos datos da la confianza de saber que la aplicación ha ganado en fluidez y escalabilidad para el futuro.

2.2.4 ChartJS.

Para la representación de los datos de forma gráfica, tras analizar el mercado se eligió la librería chartjs para ionic, por su diseño responsive, su facilidad de interacción la hora de seleccionar puntos de las gráficas, su facilidad de configuración, y su variedad de gráficas y configuraciones, pensando en un futuro en la cantidad de datos que datos que podría ser necesario representar en las diferentes formas que sería necesario representar.

Capítulo 3. Aplicación Open Smartband.

Open Smartband es la aplicación implementada para este proyecto. La aplicación se ha diseñado muy sencilla y usable. En ella se puede diferenciar 3 grandes menús, por un lado se tiene la parte de conexión, donde se busca los sensores bluetooth y se conectará, un dashboard de visualización de datos y una última parte de configuración de alertas.

Para diferenciar estos tres apartados, se tiene tres tabs diferentes en el fondo de la aplicación, según se puede visualizar en la siguiente imagen:

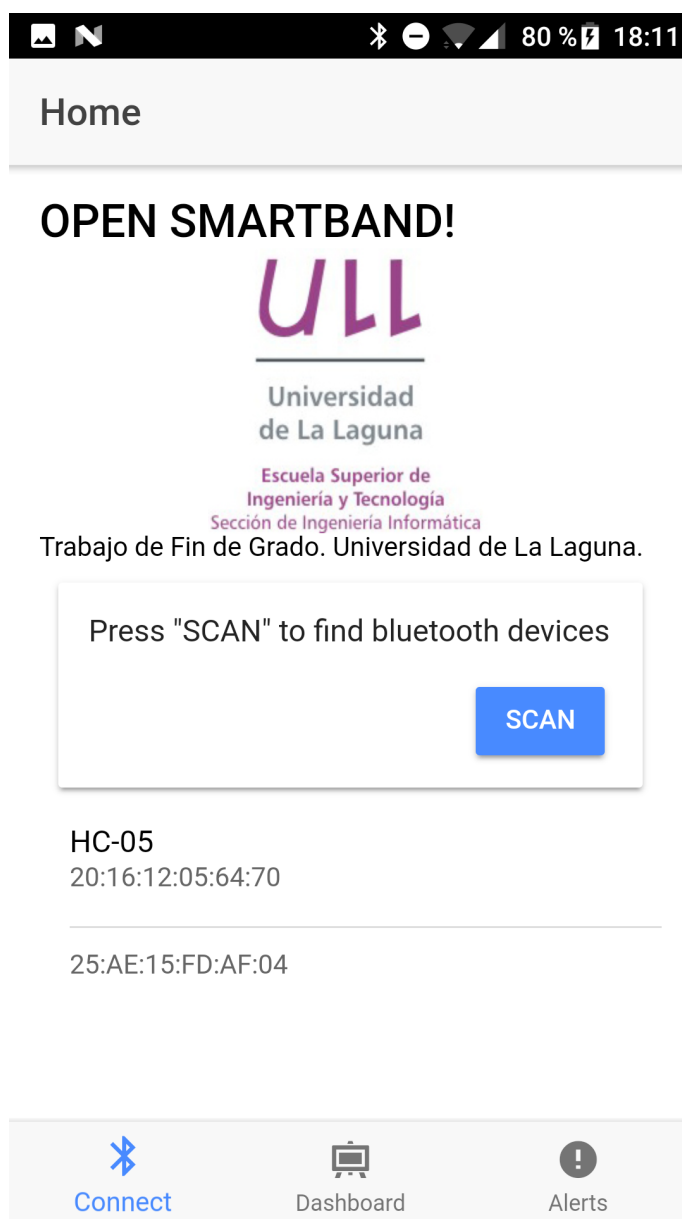


Figura 5 : diseño aplicación android.

3.1 Conexión.

En el primer tab(“pestaña”) se tiene la vista de conexión. En ella en primer lugar se encuentra el nombre de la aplicación y el logo corporativo de la ULL y la descripción de que este proyecto nace con el objetivo de superar un Trabajo de Fin de Grado.

Lo siguiente que se puede visualizar es un botón SCAN, con el cual la aplicación empezaría la búsqueda de dispositivos cercanos. Al hacer clic sobre la acción scan, aparece un spinner para dar la impresión al usuario de que se está ejecutando la búsqueda y no piense que la aplicación se ha quedado colgada o no hace nada durante un tiempo.

Una vez la aplicación termina de encontrar todos los dispositivos disponible, muestra una lista con todas opciones disponibles encontradas.

Al seleccionar alguno de ellos, si es la primera vez que se sincroniza con ese dispositivo, aparecerá un aviso de si se quiere aparear con ese dispositivo, y que se escriba el código de sincronización para confirmar la sincronización. En caso de la smartband el dispositivo a conectarnos viene identificado como HC-05

Finalizado el proceso de conexión y sincronización, la smartband y la aplicación están en conexión, y los mensajes de arduino, llegan y se procesarán de forma totalmente transparente para el usuario.

3.2 Dashboard.

Este apartado es la vista donde se puede visualizar todos los datos resultantes de analizar los mensajes recibidos de la pulsera de actividad.

La primera vez que se entra, vemos que no se refleja ningún dato, solo las cabeceras de las gráficas y elementos visuales a pintar. En la cabecera de la vista se visualiza un botón SYNC, al accionar este botón se puede ver al momento los datos más actuales disponibles en la base datos, de manera que se puede actualizar la vista en cada momento.

En el dashboard se puede ver de manera clara, y precisa, sin rebuscar mucho, diferenciados de distintos colores y con un icono representativo para cada medición, las 3 principales variables tratadas por el sistema, pulso, temperatura y pasos.

En los tres principales “cards” donde se puede apreciar la temperatura, el pulso y los pasos, corresponden directamente con las últimas mediciones almacenadas en la base de datos.

Posteriormente se puede ver una gráfica lineal, donde se refleja la temperatura y el pulso, y dos líneas simultáneas, de los últimos samples obtenidos, donde se puede hacer clic sobre cada punto y se observa la temperatura y pulso de ese momento, junto con la fecha y hora de esa medición.

En la parte superior de la gráfica se ve dos leyendas de colores que hacen referencia a cada una de las líneas dibujadas, al hacer clic sobre una de ellas, se desactiva el label y se desdibuja la línea seleccionada, por si se quiere centrar la visualización en una medición en concreto.

Por último se observa una gráfica de barras de pasos. Esta gráfica lo que muestra es el total de pasos dados cada día en la última semana, de esta manera se puede determinar qué día se ha hecho más ejercicio que otros.

3.3 Alertas.

En esta última vista, se puede configurar los valores personalizados para los que se quiere que se avise, si se supera el umbral en cada una de esas mediciones, temperatura, pulso y pasos, de manera que al sincronizar la base de datos, si alguna de las mediciones superarse ese valor establecido por el usuario, se visualizará una señal de alerta para esa medición informando de que se ha superado ese umbral, para que se revise el valor por si fuera importante o peligroso para esa persona.

Esta parte está sobre todo pensada para las personas con problemas o enfermedades que necesitan saber que su temperatura corporal o su pulso no pueden sobrepasar un umbral que ellos mismo personalizar en cada momento.

Capítulo 4.

Resultados

Una vez descrito los procesos y requerimientos necesarios para llevar a cabo este proyecto, es esta sección se explicará detalladamente cómo se obtienen los resultados, podremos visualizar el funcionamiento de la smartband y de la aplicación móvil Open Smartband corriendo en un dispositivo android.

4.1 Arduino.

En primer lugar, se va a analizar la siguiente imagen, donde se puede ver el arduino y los sensores conectados y los mensajes que envía, en este caso se muestra cada 10 segundos.

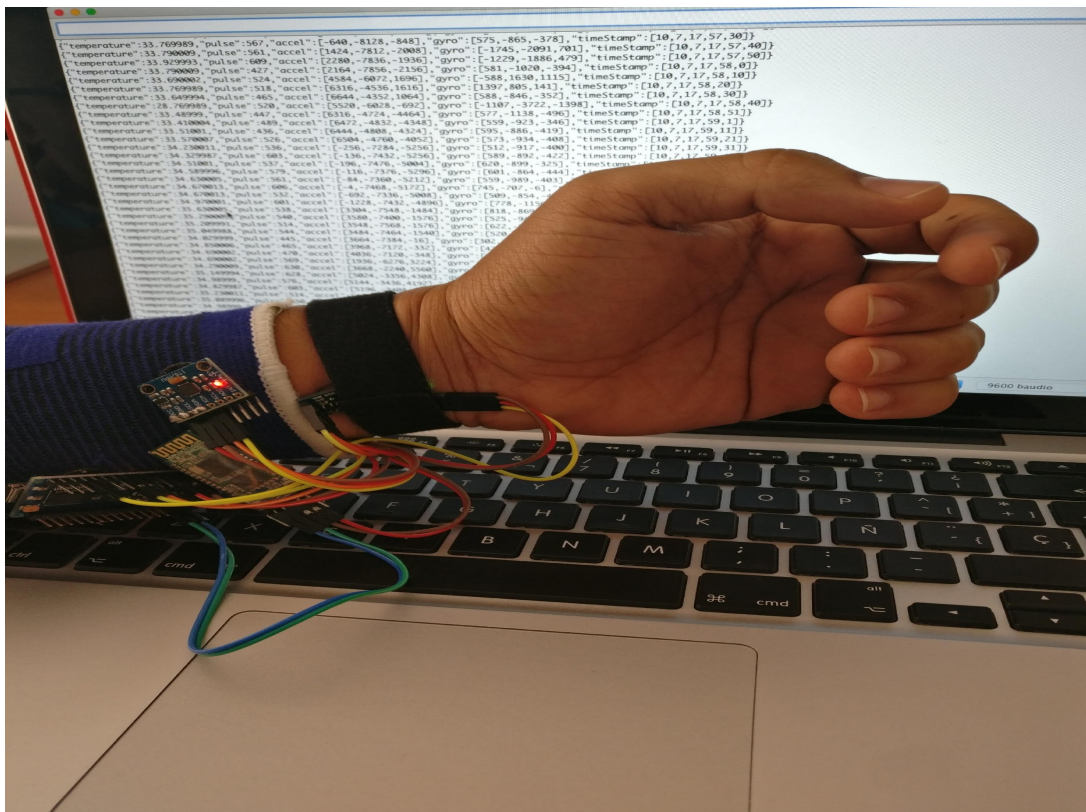


Figura 6 : Arduino Smartband

Como se ve en la imagen, bajo la cinta negra, están los dos sensores que se necesitan estén en contacto con la piel, son el sensor de temperatura y el sensor de pulso. Pegados a la muñequera se puede observar en un posición superior el acelerómetro y giroscopio y justo por debajo de éste último el sensor bluetooth y por último en el fondo izquierdo de la imagen, la placa arduino.

En el fondo de la imagen se ve en una consola los paquetes de datos que están enviando en cada momento, ahora se va a analizar esa consola de datos desde una mejor visualización de los datos.

```

/dev/cu.usbserial-00000000
{"temperature":34.230011,"pulse":536,"accel":[-256,-7284,-5256],"gyro":[512,-917,-400],"timeStamp":[10,7,17,59,31]}
{"temperature":34.329987,"pulse":603,"accel":[-136,-7432,-5256],"gyro":[589,-892,-422],"timeStamp":[10,7,17,59,41]}
{"temperature":34.51001,"pulse":537,"accel":[-196,-7476,-5004],"gyro":[620,-899,-325],"timeStamp":[10,7,17,59,51]}
{"temperature":34.589996,"pulse":579,"accel":[-116,-7376,-5296],"gyro":[601,-864,-444],"timeStamp":[10,7,18,0,1]}
{"temperature":34.630005,"pulse":561,"accel":[-84,-7360,-5212],"gyro":[559,-989,-403],"timeStamp":[10,7,18,0,11]}
{"temperature":34.670013,"pulse":606,"accel":[-4,-7468,-5172],"gyro":[745,-707,-6],"timeStamp":[10,7,18,0,21]}
{"temperature":34.670013,"pulse":532,"accel":[-692,-7336,-5008],"gyro":[509,-854,-462],"timeStamp":[10,7,18,0,31]}
{"temperature":34.970001,"pulse":601,"accel":[-1228,-7432,-4896],"gyro":[778,-1150,-245],"timeStamp":[10,7,18,0,41]}
{"temperature":35.630005,"pulse":538,"accel":[3304,-7548,-1484],"gyro":[818,-869,-224],"timeStamp":[10,7,18,0,51]}
{"temperature":35.290009,"pulse":540,"accel":[3580,-7400,-1576],"gyro":[525,-946,-439],"timeStamp":[10,7,18,1,1]}

```

Figura 7 : Consola de paquete de datos Arduino

4.2 Datos arduino.

Como se observa en la imagen superior, se ve un conjunto de mensajes enviados por la smartband, cada 10 segundos. Si se analiza los mensajes, en primer lugar se puede observar que el formato, como ya se había mencionado anteriormente, es un formato JSON cada uno de ellos, donde se puede distinguir las etiquetas claves, temperature, pulse, accel, gyro y timeStamp.

Se observa que la temperatura según van llegando los mensajes se va estabilizando en torno a los 36 grados de temperatura. Son unos 36-37 grados la temperatura normal para una persona en condiciones de salud normales.

En segundo lugar se ve el pulso, donde se tiene unos valores de entre 53 y 60 pulsaciones (el valor que se observa hay que dividirlo entre 10 para tener el nivel de precisión correcto, es decir, 536 corresponde 53,6 pulso por minuto). Las pulsaciones de una persona en reposo oscila entre los 50 y 100, pero esas pulsaciones puede ser inferior en función de la edad, y del estado físico de la persona y el sexo, como se puede observar en la siguiente tabla:

HOMBRES: VALORES DE REFERENCIA PULSACIONES EN REPOSO POR MINUTO				
EDAD (AÑOS)	INADECUADO (PPM)	NORMAL (PPM)	BUENO (PPM)	EXCELENTE (PPM)
20 - 29	86 o más	70 - 84	62 - 68	60 o menos
30 - 39	86 o más	72 - 84	64 - 70	62 o menos
40 - 49	90 o más	74 - 88	66 - 72	64 o menos
50 o más	90 o más	76 - 88	68 - 74	66 o menos

MUJERES: VALORES DE REFERENCIA PULSACIONES EN REPOSO POR MINUTO				
EDAD (AÑOS)	INADECUADO (PPM)	NORMAL (PPM)	BUENO (PPM)	EXCELENTE (PPM)
20 - 29	96 o más	78 - 94	72 - 76	70 o menos
30 - 39	98 o más	80 - 96	72 - 78	70 o menos
40 - 49	100 o más	80 - 98	74 - 78	72 o menos
50 o más	104 o más	84 - 102	76 - 82	74 o menos

Figura 8 : pulsaciones en reposo

La siguiente etiqueta es accel, correspondiente al acelerómetro, que da las coordenadas X,Y,Z. De igual manera que el gyro, giroscopio.

Por último se tiene la etiqueta timeStamp, donde se puede ver que los mensajes se envían con una separación de 10 segundo entre ellos. El formato es un array compuesto por día, mes, hora, minutos y segundos, de tal forma que [10,7,18,0,21] corresponde a las 18:00:21 horas del día 10 de julio.

4.3 Open Smartband

Una vez comprobado el funcionamiento de la smartband, ahora se procederá a observar el funcionamiento de la aplicación móvil android, Open Smartband.

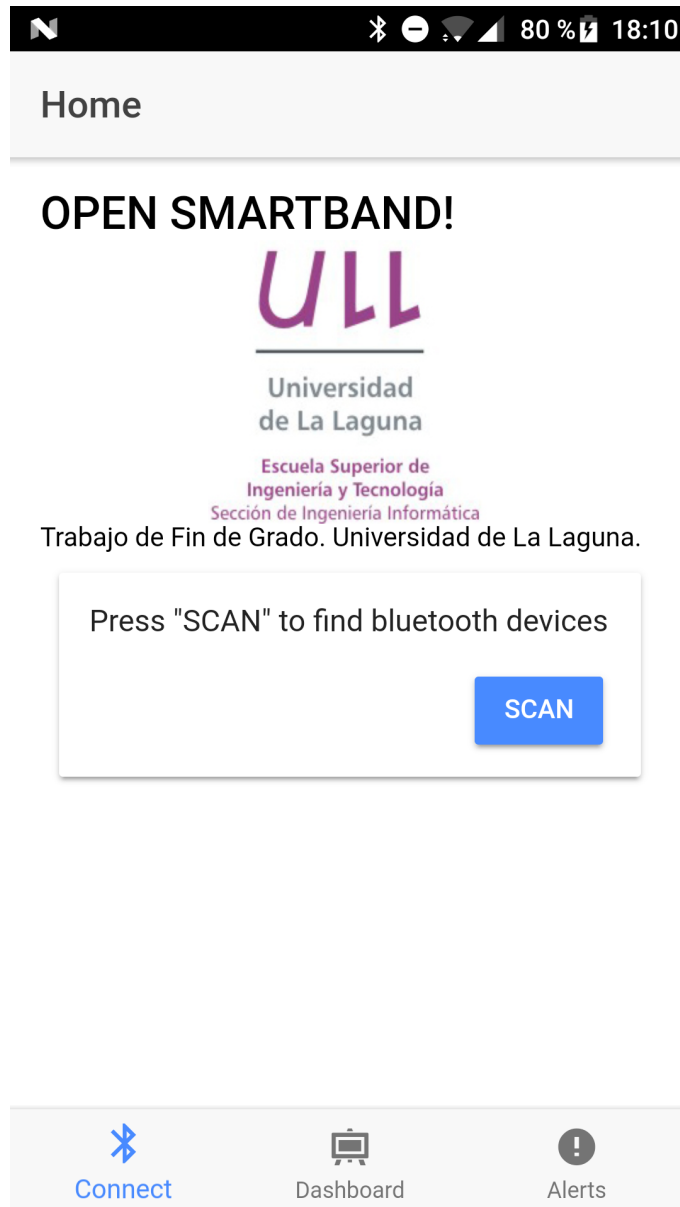


Figura 9 : vista home Open Smartband

En la anterior se puede ver la primera pestaña de nuestra aplicación, como se ha citado anteriormente, en la parte superior se puede observar el nombre de la aplicación, Open Smartband, y objetivo porque el que se ha desarrollado, para cumplir con el plan docente de la Universidad de La Laguna, en la Escuela Superior de Ingeniería y Tecnología, como Trabajo de Fin de Grado.

A continuación en el centro de la vista se puede observar un recuadro que indica que se haga clic en el botón SCAN, para buscar todos los dispositivos bluetooth cercanos.

Por último, se puede observar también, en la parte inferior de la imagen, el menú de las 3 pestañas, que ya se han citado anteriormente. En este caso

se observa que se está situado en la pestaña Connect destacada de las demás en un color azul sobre los gris de las dos restantes e identificada con un icono con el símbolo de bluetooth.

Al ejecutar la búsqueda de dispositivos, aparecerá un spinner en la vista para indicar de que el proceso de búsqueda está en proceso, este proceso se demora unos 30 segundos.

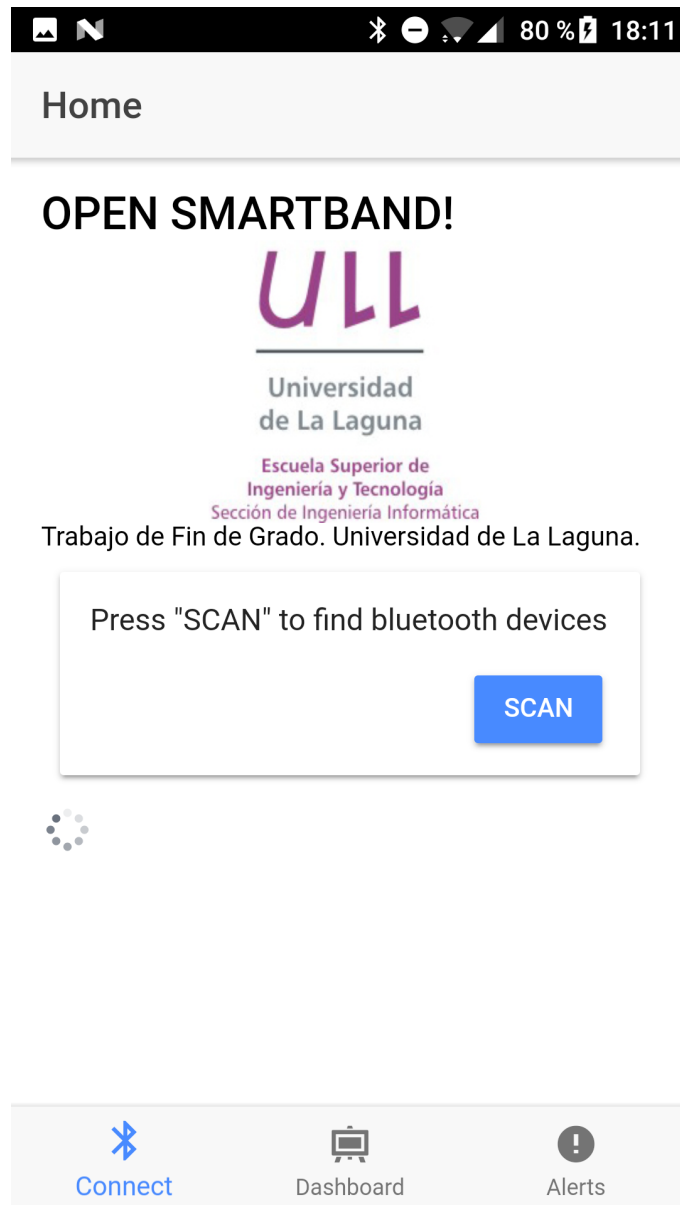


Figura 10 : Escaneando dispositivos

Una vez , transcurridos esos 30 segundos, aparecerá en la vista un listado con todos los dispositivos disponible para poder establecer una conexión.

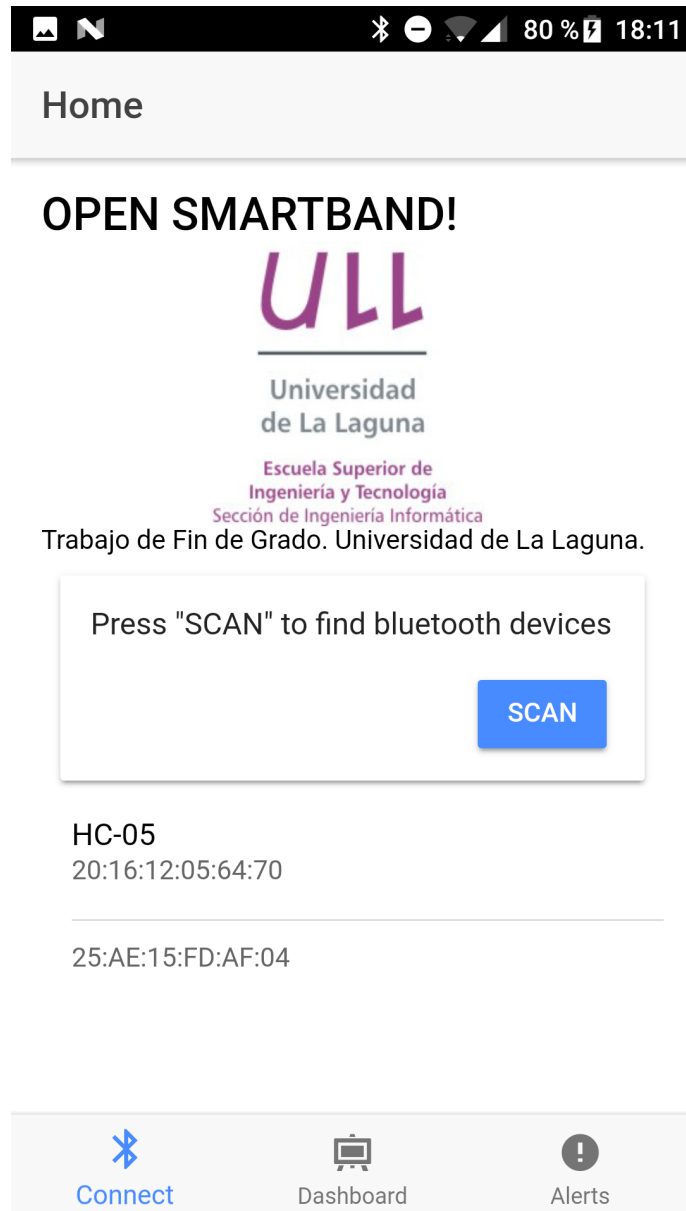


Figura 11 : dispositivos disponibles

Como es visible en la imagen anterior, una de las opciones es HC-05, que es el nombre del módulo bluetooth de nuestra smartband, simplemente al hacer click sobre la opción deseada, en este caso la pulsera de actividad, ya quedaremos emparejados con ella(en caso de ser la primera, vez saldrá la ventana de confirmación, estableciendo el código de conexión, que por defecto es 1234).

Una vez sincronizados con la smartband, de manera totalmente transparente al usuario, los mensajes de datos están siendo recibidos por la aplicación, procesados y almacenados en nuestro sistema de base de datos.

Posteriormente se va a visualizar todos los datos disponible, para ello, debemos hacer click en la pestaña dashboard.

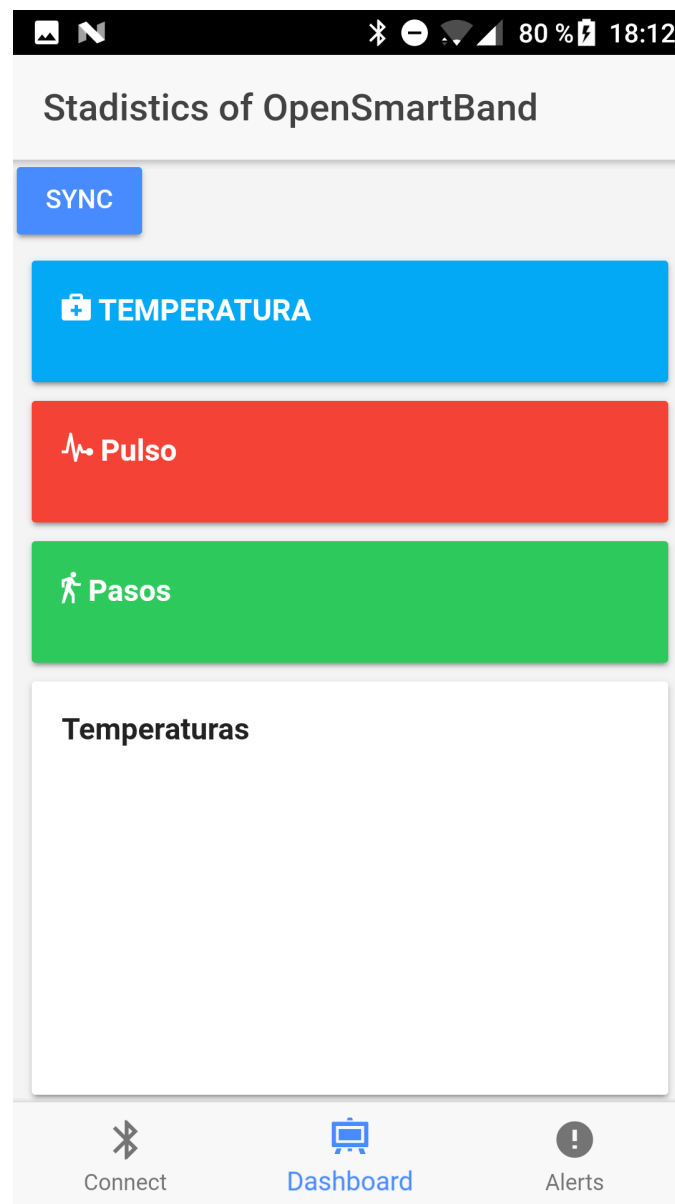


Figura 12 : pestaña de visualización de datos

Al entrar vista, se ve resaltados y de manera llamativa, dos cuadros distinguidos por colores y por un icono diferente y representativo de cada una de las variables principales procesadas por la aplicación, temperatura, pulso y pasos. También es visible que no hay datos, como se ha citado anteriormente, a la aplicación siguen llegando datos, por lo que cuando se quiera actualizar los datos y ver los datos más actuales, simplemente, se debe accionar el botón SYNC situado en la parte superior izquierda de la vista.



Figura 13 : Datos sincronizados

Una vez actualizado los datos, se observan datos en los 3 recuadros, comentados anteriormente, donde cada de uno de ellos corresponde al último mensaje procesado por la aplicación y recibido de la smartband.

Debajo de los 3 recuadros, se puede observar dos graficas, como se puede observar en la siguiente imagen:

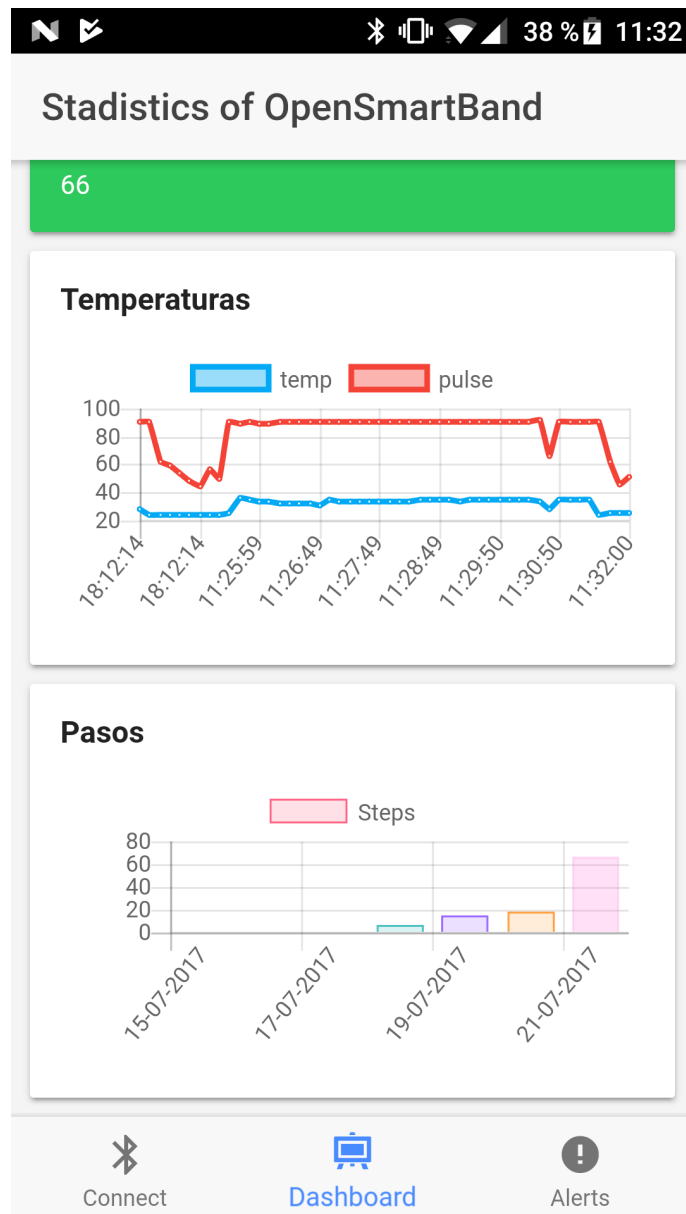


Figura 14 : visualización de gráficas

Como se puede observar, en la parte superior se tiene una gráfica, donde se tiene dos datos representados en una gráfica de líneas. De color rojo, según se puede observar en la leyenda, se tiene representado los últimos datos del pulso, y por otro lado de color azul, se tiene representados los últimos samples de temperatura corporal.

Y en la gráfica final, se puede ver una gráfica de barras, donde se obtiene el recuento total de pasos por día realizados durante la última semana.

En la última pestaña, alerts, se tiene opciones de configuración para establecer valores de alerta personalizados.

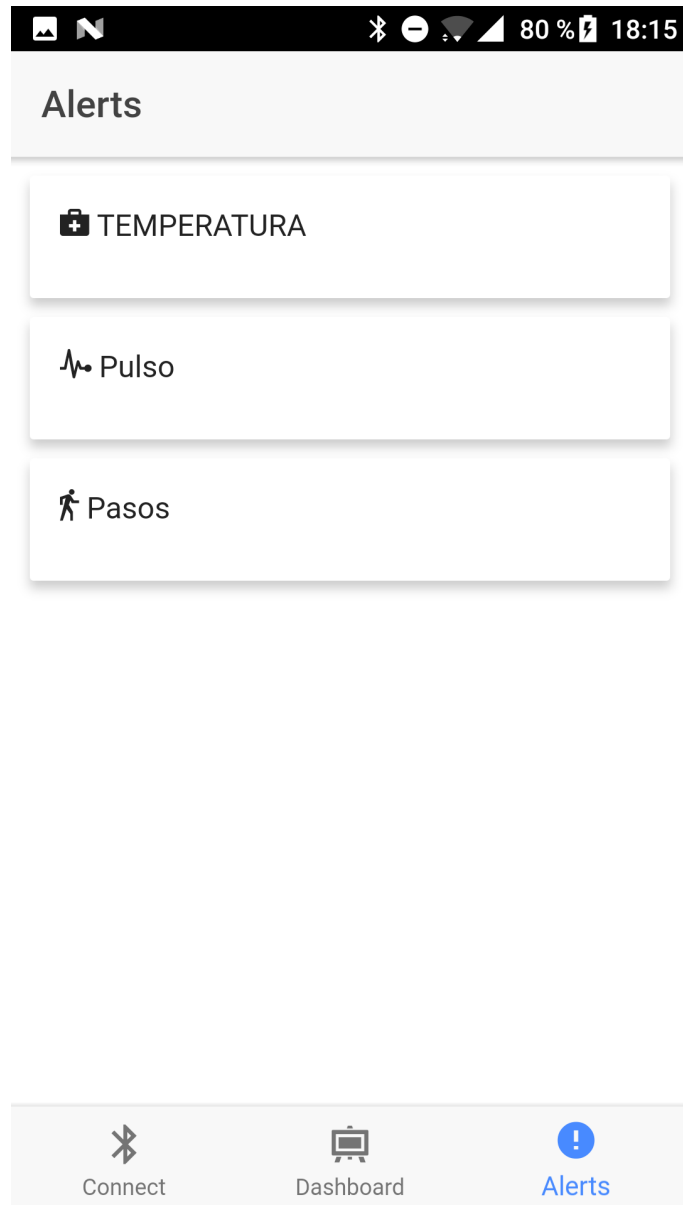


Figura 15 : configuración de alertas

Al hacer clic en una de la opciones, se puede establecer valores numéricos para cada una de las distintas variable, para que nos avise si al leer el último sample de datos, es mayor al valor que se ha establecido en la alerta.

Capítulo 5.

Conclusiones y líneas futuras

En la actualidad la smartband es funcional y estable, envía todos los datos de los sensores en un paquete de datos, y permite conectarse con la aplicación mediante bluetooth para transmitir los datos leídos de los diferentes módulos.

Y la aplicación móvil es instalable y ejecutable en android e ios, y permite establecer una conexión bluetooth con la pulsera de actividad, y recibir los datos enviados por esta. Se puede analizar, procesar y guardar todos esos datos en la base de datos. También se puede visualizar y hacer un análisis de los datos recibidos en la vista de gráficas y datos recibidos. Y por último se tiene la opción de establecer valores personalizados de alerta.

La investigación llevada a cabo durante la realización de éste proyecto deja evidencia del hecho de que se puede obtener información de gran valor de las smartbands, y que se puede ampliar los datos que se recogen del entorno en comparación de las pulseras de actividad comerciales, y por lo tanto es posible extender el uso de las smartband en el uso diario, en un uso deportivo y en un entorno médico también. No obstante, se considera que este proyecto es solo el principio de lo que se podría llegar a conseguir siguiendo la línea de investigación y actuación de dicho proyecto, realizando la incorporación de nuevos sensores, la calibración y ajuste de los sensores disponibles para obtener datos más precisos y seguir mejorando la aplicación para dar más opciones de configuración y la posibilidad de mostrar más datos de interés para el usuario.

Por otro lado, cabe destacar que los objetivos fijados al comienzo de este trabajo no se han cumplido en su totalidad. Se han cumplido objetivos como la creación, de la smartband, el almacenamiento de los datos en una base de datos y la creación de una aplicación móvil capaz de sincronizarse con la pulsera de actividad y poder mostrar los datos obtenidos de los sensores de la smartband. Aunque a la hora de realizar este proyecto han surgido ciertos inconvenientes y el proceso no se ha caracterizado por su sencillez, el balance del cumplimiento de los objetivos es positivo. Aunque se hace constar también, que sabemos que la aproximación que hemos realizado del podómetro es poca precisa, y que es una de líneas futuras a trabajar para conseguir una mejor pulsera de actividad, y que finalmente no se ha conseguido llegar a la cúspide de los objetivos descrito al inicio del proyecto, el sistema experto, pero queda evidencia de los inconvenientes y de los cambios que se han tenido que desarrollar para poder llegar a este punto.

Como continuación de este proyecto, considero que se puede realizar mejoras en la estimación de pasos, creando un algoritmo de detección de movimiento y cálculo de pasos, añadiendo nuevos sensores, como por ejemplo un sensor de glucosa o creando mejoras en la aplicación móvil, por ejemplo incorporando un sistema experto que nos recomendará alimentación o estiramientos y ejercicios en función de nuestra actividad diaria.

A nivel personal el hecho de haber trabajado en este proyecto me ha brindado la oportunidad de formarme en un campo de la informática desconocido para mi, el hardware, y de actualizarme en el entorno de desarrollo de aplicaciones móviles, adaptándome a la últimas versiones de los frameworks más actuales del mercado. Cabe destacar que el desarrollo de este proyecto me ha ayudado a descubrir cuál podría ser la vertiente profesional por la que decantarme una vez finalizados mis estudios universitarios, ya que visto de primera mano que se pueden desarrollar aplicaciones con un gran abanico de posibilidades diferentes que son de utilidad para la sociedad actual.

Capítulo 6.

Summary and Conclusions

Currently the smartband is functional and stable. It sends all the sensor's data in a data packet, connecting via Bluetooth to the application to transmit the obtained data from the different modules. The mobile application is installable and executable in Android and iOS facilitates a Bluetooth connection with the activity bracelet and receives the data sent by it.

Data can be analyzed, processed, and saved in the database. Data can also be analyzed and received in the through graphs and received data. Additionally, customized alerts are also a feature.

The research carried out during the course of this project concludes that valuable information can be obtained by the use of smartbands. Apart from that, it is possible to collect more information from the environment if needed. Therefore, it is possible to expand the use of smartbands in daily life, athletic use, and in a medical environment as well.

However, this project is only the beginning of what could be achieved if the investigation is continued. The incorporation of new sensors, calibration, and the adjustment of sensors could help to improve the use of the smartbands because the users could obtain more accurate information.

It is important to mention that the objectives set at the beginning of this research have not been fully achieved.

The accomplished objectives in this research project are: the creation of the smartband, the storage of the data in database, and the creation of a mobile application capable of synchronizing with the wristband and the ability to display the data obtained from the sensors of the smartband.

During this investigation, some difficulties arose and although the process was not simple, I consider that the achievement of the objectives of this project have been successfully resolved.

Although it is also noted that the approach to the pedometer is inaccurate, I consider that it can be a future line to work with in order to obtain a better activity bracelet. Finally, there is an objective described at the beginning of the project (the expert system) that has not been developed because of the rise of different problems that made it impossible to achieve.

As a continuation of this project, I consider it possible to improve the footstep estimation by creating an algorithm of motion detection and footstep calculation, adding new sensors, such as a glucose sensor, or by sensor

mobile application enhancements, e.-g by incorporating an expert system that will recommend food or stretching and exercises according to our daily activity.

The development of this project has given me the oppotunity to become familiar with hardware, a field formerly unknown to me. In addition, I've kept myself up-to-date in the mobile application environment, which is of ever-growing importance.

The development of this project has helped me elucidate a possible work activity for me in the future when I conclude my studies because I have discovered that it is possible to and that I am capable of developing very useful applications for the society.

Capítulo 7.

Presupuesto.

En este capítulo se especifica un presupuesto que indica cuánto costaría realizar este Trabajo de Fin de Grado si se tratase de un trabajo encargado por un cliente para una empresa profesional.

Teniendo en cuenta que para la realización de éste proyecto no ha sido necesaria ningún tipo de licencia de software ya que las herramientas a utilizar han sido de software libre, el presupuesto que se detalla a continuación descubrirá gastos de equipos, recursos y equipo humano necesario para llevarlo a cabo.

EQUIPO HUMANO: Dado que éste Proyecto de Fin de Grado, cuenta con un reconocimiento de 12 ECTS, y que según el reconocimiento europeo cada ECTS deberían de ser 25 horas de trabajo, en éste proyecto se han realizado más de 300 horas, entre documentación, desarrollo, test y redacción de la memoria. Si se calcula la hora de trabajo a unos 40€ la hora, esto daría un total de 12.000€ en mano de obra.

AMORTIZACIÓN DE EQUIPOS INFORMÁTICOS: Según la aplicación de los coeficientes de amortización fijados en la tabla oficialmente aprobada por la Ley 27/2014, de 27 de noviembre, del Impuesto sobre Sociedades, el porcentaje a aplicar para Equipos para procesos de información es de un 25 por ciento. Por lo tanto, si se estima el valor del equipo a 1000 euros, en un año se amortizaría 250€, si ahora se calcula el dinero amortizado en las 300 horas de trabajo de un valor total de 31,24 euros.

GASTO DE LUZ e INTERNET: De igual manera, se va calcular los gastos de luz e internet que han supuesto estas 300 horas de trabajo. En primer lugar, se calcula el gasto de luz, el precio hora de luz es de 0,114 euros a día 19 de julio de 2017, por lo tanto sería un total de 34,2 euros de luz por las 300 horas de trabajo. Por otro lado, según el diario el País, en España el coste medio por el servicio de Internet banda ancha es de 43,10 euros mensuales, lo que calculado daría unos 21.55 euros para el desarrollo del proyecto.

COMPONENTES: Para terminar, se va calcular los gastos de los componentes que se ha necesitado comprar para la construcción de la smartband.

1. Jumpers: 3,99€
2. Arduino nano v3: 8,90€

- 3. Sensor de pulso: 11,00€
 - 4. Sensor de temperatura: 15,50€
 - 5. Sensor acelerómetro/giroscopio defectuoso: 8,83€
 - 6. Sensor acelerómetro/giroscopio: 3,10€
 - 7. Sensor bluetooth: 5,50€
- Total : 56,82€

7.1 Presupuesto total.

Tipos	Gasto
Equipo Humano	12.000
Amortización equipos	31,24
Internet y Luz	34,2 + 21,55
Componentes	56,82
TOTAL :	12.143,81€

Tabla 1 : resumen de gastos

Apéndice A.

Repositorio del proyecto.

Todo el código de este proyecto se puede encontrar alojado en github de manera open source para su uso en el siguiente enlace:

<https://github.com/alu0100499162/TFG-OpenSmartband>

Bibliografía

- [1] ACM LaTeX Style. [http://www.acm.org/publications/latex style/](http://www.acm.org/publications/latex%20style/).
- [2] Arduino. <https://www.arduino.cc/en/Serial/Read>.
- [3] Arduino pinout. <https://www.luisllamas.es/esquema-de-patillaje-de-arduino-pinout/>
- [4] Arduino librería para MAX30100. <https://github.com/oxullo/Arduino-MAX30100>.
- [5] Arduino ejemplos sensor de pulso. <https://github.com/WorldFamousElectronics>, <https://github.com/WorldFamousElectronics/PulseSensorStarterProject>, <http://www.instructables.com/id/Pulse-Sensor-With-Arduino-Tutorial/>
- [6] Documentación para el pulso. <https://porque-se.com/pulso-normal-cuales-frecuencia-cardiaca/>
- [7] Ejemplos sensor Bluetooth HC-05. <http://www.instructables.com/id/Arduino-AND-Bluetooth-HC-05-Connecting-easily/> , <http://www.prometec.net/bt-hc05/>
- [8] Documentación para acelerómetro y giroscopio. <https://github.com/jrowberg/i2cdevlib/tree/master/Arduino/MPU6050>
- [9] Crear json de datos en Arduino. <https://bblanchon.github.io/ArduinoJson/doc/encoding/>
- [10] Ionic primeros pasos. <https://www.adictosaltrabajo.com/tutoriales/empezando-con-ionic-2/> , <https://andresbaredes.wordpress.com/2016/08/31/ionic-framework-2-primeros-pasos/> , <http://blog.enriqueoriol.com/ionic-course> , <https://desarrolloweb.com/manuales/manual-ionic2.html>
- [11] Ionic 2 y Android Studio. <https://www.christianengvall.se/add-android-platform-in-ionic-2/>
- [12] Local storage vs SQLite. <https://www.thepolyglotdeveloper.com/2015/12/use-sqlite-in-ionic-2-instead-of-local-storage/>
- [13] SQLite. <https://ionicframework.com/docs/native/sqlite/>, <https://reviblog.net/tag/ionic-2-sqlite/>
- [14] Ionic native. <http://ionicframework.com/docs/native/>.
- [15] Angular ChartJS. <http://jtblin.github.io/angular-chart.js/>
- [16] <https://www.arduino.cc/en/Main/OldSoftwareReleases>.
- [17] https://en.wikipedia.org/wiki/Adafruit_Industries

- [18] <http://android-studio.uptodown.com/windows>
- [19] [https://es.wikipedia.org/wiki/Atom_\(editor_de_textos\)](https://es.wikipedia.org/wiki/Atom_(editor_de_textos))
- [20] <https://developer.android.com/reference/android/database/sqlite/package-summary.html>
- [21] <https://www.ion-book.com/blog/ionic2/ionic-native/>
- [22] <https://ionicframework.com/>
- [23] [https://es.wikipedia.org/wiki/Angular_\(framework\)](https://es.wikipedia.org/wiki/Angular_(framework))
- [24] <https://es.wikipedia.org/wiki/TypeScript>
- [25] <https://es.wikipedia.org/wiki/Arduino>
- [26] <https://es.wikipedia.org/wiki/SQL>
- [27] <https://es.wikipedia.org/wiki/GitHub>
- [28] <http://es.gizmodo.com/como-organizar-toda-tu-vida-utilizando-trello-1684529913>
- [29] <https://www.gpsos.es/soluciones-open-source/definicion-de-open-source/>
- [30] <http://www.ituser.es/en-cifras/2015/04/android-se-encuentra-en-el-876-de-los-smartphones-en-espana>