Escuela Superior
de Ingeniería y Tecnología
Universidad de La Laguna

Trabajo de Fin de Grado

Grado en Ingeniería Informática

Seguridad de dispositivos para detección de fatiga y somnolencia

Device security for fatigue and drowsiness detection

Iris Estefanía Pereira Domínguez

La Laguna, 13 de junio de 2022

Dña. **Pino Teresa Caballero Gil**, con N.I.F. 45.534.310-Z, Catedrática de Universidad adscrita al Departamento de Ingeniería Informática y de Sistemas de la Universidad de La Laguna, como tutora

Dña. **Jezabel Miriam Molina Gil**, con N.I.F. 78.507.682-B, profesora Ayudante Doctora de Universidad adscrita al Departamento de Ingeniería Informática y Sistemas de la Universidad de La Laguna, como cotutora.

C E R T I F I C A (N):

Que la presente memoria titulada:

"Seguridad de dispositivos para detección de fatiga y somnolencia"

ha sido realizada bajo su dirección por Dña. **Iris Estefanía Pereira Domínguez**, con N.I.F. 43.388.253-H.

Y para que así conste, en cumplimiento de la legislación vigente y a los efectos oportunos, firman la presente en La Laguna a 13 de junio de 2022

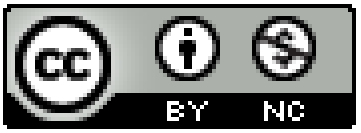
Agradecimientos

A mis padres, mi hermana y mis amigos y amigas, que han estado ahí, pese a todo, apoyándome incondicionalmente.

A Pino, Jezabel y Rubén por orientarme a lo largo de este proyecto.

A la empresa Metropolitano de Tenerife por el apoyo durante el desarrollo.

Licencia



Resumen

El propósito de este trabajo ha sido desarrollar una aplicación móvil para plataformas Android que, haciendo uso de la tecnología de área personal inalámbrica Bluetooth Low Energy, subconjunto del protocolo Bluetooth v4.0, y de un dispositivo wearable capaz de detectar mediante sensores de fotopletismografía (PPG) las constantes vitales del usuario que lleva puesto el sistema, sea capaz de inferir cuándo este se encuentra atravesando la primera fase del sueño: adormecimiento.

Estos dispositivos se colocan en la muñeca del usuario y registran el movimiento a través de un acelerómetro interno de tres ejes de manera no invasiva. Los datos recogidos son procesados por un algoritmo capaz de clasificar el tipo de movimiento realizado. También se encargan de registrar datos durante el descanso nocturno para ofrecer información acerca de la cantidad de horas de sueño y calidad del mismo. Además de todo esto, posee otras utilidades, como avisos inteligentes, pulsómetro, reloj y cronómetro, etc. Aparte de las pulseras de actividad, objeto de estudio en este proyecto, existe gran cantidad de dispositivos wearables, como pueden ser smartwatches, smartglasses, prendas de vestir, otros complementos, como anillos u otro tipo de joyas, dispositivos implantables… con una inmensa variedad de funciones adaptadas al tipo de sistema y lugar donde deban ir colocados.

A través de determinadas funcionalidades proporcionadas por Android Studio y del lenguaje de programación Java es posible, tras un proceso de autenticación para completar el emparejamiento, acceder programáticamente a aquellas características de la pulsera de actividad que el fabricante deja a libre disposición, como pueden ser la información del dispositivo, características tanto del hardware como del software del mismo, así como la información de la batería, cifras suministradas por el sensor de ritmo cardiaco, etc., si bien es cierto que existen muchas otras características a las que solo se puede acceder a través de la aplicación del fabricante y no desde aplicaciones o servicios de terceros.

Una vez obtenido el acceso a los servicios de la smartband, se pretendía deducir, mediante un algoritmo, el estado del usuario en cuanto a somnolencia se refiere, con el objetivo de prevenir cualquier tipo de accidente, ya sea laboral o de tráfico, así como de alertar a este mediante vibraciones continuadas del dispositivo, con el fin de concienciarlo de que atraviesa un estado que puede suponer un riesgo, tanto para su persona como para su entorno. Este algoritmo requiere de un periodo de aprendizaje, puesto que, al tratarse de variables fisiológicas, estas pueden verse alteradas en función de diversos factores, como pueden ser el género, la edad, la alteración en los hábitos de sueño, el padecimiento de enfermedades cardiovasculares, la diabetes, etc. del usuario, por lo que en muy pocos casos podrá aplicarse el mismo modelo a distintos usuarios. Este periodo resulta útil para registrar una tendencia en la frecuencia cardiaca del portador del dispositivo y para observar cómo varía a lo largo del tiempo, estableciendo así un umbral a partir del cual se determina que es posible que el usuario presente síntomas de cansancio y somnolencia.

Palabras clave: Somnolencia, smartband, wearable, Android Studio, Java

Abstract

The purpose of this project has been the development of a mobile application for Android platforms that, using Bluetooth Low Energy wireless personal area technology, a subset of the Bluetooth v4.0 protocol, and a wearable device capable of detecting the wearer's vital signs through photoplethysmography (PPG) sensors, is able to infer when the wearer is going through the first stage of sleep: drowsiness.

These devices are worn on the wrist and record movement through an internal three-axis accelerometer in a non-invasive way. The data collected is processed by an algorithm capable of classifying the type of movement performed. They are also responsible for recording data during the night's rest to provide information about the number of hours and the quality of sleep. In addition to all this, it has other utilities such as smart alerts, heart rate monitor, watch and stopwatch, etc. Apart from activity bracelets, the object of study in this project, there are a large number of wearable devices such as: smartwatches, smartglasses, clothing, other accessories such as rings or other types of jewellery, implantable devices… with a huge variety of functions adapted to the type of system and place where it is to be placed.

Through certain functionalities provided by Android Studio and the Java programming language it is possible, after an authentication process to complete the device pairing, to programmatically access those features of the activity wristband that the manufacturer leaves freely available, such as device information, hardware and software features of the device, as well as battery information, figures provided by the heart rate sensor, etc. However, there are many other features that can only be accessed through the manufacturer's application and not through third-party applications or services.

Once access to the smartband's services had been obtained, the aim was to use an algorithm to deduce the user's state of drowsiness in order to prevent any type of accident, whether at work or in traffic, and to alert the user through continuous vibrations of the device in order to make them aware that they are going through a state that could pose a risk both to themselves and to their surroundings. This algorithm requires a learning period since, as these are physiological variables, they can be altered depending on various factors such as the user's gender, age, altered sleep habits, cardiovascular disease, diabetes, etc., so that in very few cases can the same model be applied to different users. This period is useful to establish a trend in the wearer's heart rate and observe how it varies over time, thus establishing a threshold at which it is determined that the wearer is likely to show symptoms of tiredness and drowsiness.

Keywords: Drowsiness, smartband, wearable, Android Studio, Java

Índice general

Índice de figuras

Índice de tablas

1. Introducción

El presente documento tiene como objetivo recoger toda la información relativa al proceso de realización de este Trabajo de Fin de Grado. Concretamente, este primer capítulo se compone de una introducción al mismo, exponiendo en ella los objetivos que se han buscado con su realización, así como las fases que se han llevado a cabo hasta su finalización. Por último, se detalla la estructura de este documento y sus anexos, mostrando brevemente el contenido de cada sección.

1.1 Motivación

En la actualidad, los dispositivos wearables, en concreto smartbands y smartwatches son una tendencia en claro crecimiento.

El término **wearable** se refiere al conjunto de aparatos electrónicos que se incorporan en nuestro cuerpo e interactúan de forma continua con el usuario a la vez que con otros dispositivos, con el fin de realizar alguna tarea concreta. Algunos de estos dispositivos pueden ser relojes inteligentes (smartwatches), zapatillas, pulseras que controlan el estado de salud (smartbands), entre otros muchos.

La palabra *wearable* posee una raíz inglesa cuyo significado es “llevable” o “vestible”. En el argot tecnológico, este término señala a todos aquellos productos que contienen un microprocesador y que los usuarios agregan como parte de su vida cotidiana. Esta tecnología está presente en un amplio abanico de campos, con la finalidad de mejorar la calidad de vida de los usuarios, estando especialmente presente en el ámbito de la salud, la seguridad, el deporte, etc.

Una de las cualidades más aclamadas de estos dispositivos es la **monitorización de la calidad de sueño**, y esto es debido a que este tipo de dispositivos es capaz de detectar el momento en el que el usuario se queda dormido, cada vez que este se despierta a lo largo de la noche o la calidad y cantidad de sueño. Además, son capaces de estimar la cantidad de sueño ligero, profundo y REM, así como el tiempo que el usuario pasa despierto. Dependiendo del modelo de dispositivo, es posible que en función de los resultados se realicen recomendaciones para conseguir una mejora en la calidad del sueño, factor de gran importancia en nuestra vida diaria.

Como sabemos, un buen descanso se considera imprescindible para la salud física y mental del ser humano. Como indica la organización World Sleep Society [1], una cantidad de sueño insuficiente está íntimamente relacionada con el desarrollo de patologías y dolencias, como son la obesidad o la hipertensión, el deterioro cognitivo, una menor respuesta del sistema inmune, entre otros. Así pues, la falta de sueño o una mala calidad del mismo puede desembocar en graves problemas cuando se vuelve crónica. Sin embargo, algunas de las consecuencias más frecuentemente experimentadas y que pueden conllevar grandes riesgos son la fatiga y la somnolencia.

La definición más elemental de fatiga es “una sensación de falta de energía, de agotamiento o de cansancio” [2]. Este estado fisiológico está estrechamente relacionado con la somnolencia, cuya definición es “la propensión a dormirse” o “la habilidad de transición de la vigilia al sueño” [3]. En este último es posible experimentar una sensación de cansancio, pesadez, sueño y torpeza en los movimientos. A pesar de esto, son dos términos que difieren ligeramente, puesto que la fatiga puede llegar a conducir a un estado de somnolencia.
Por ello, es crucial saber reconocer los primeros síntomas de la fatiga, puesto que, a pesar de que el sueño no se produce de manera repentina, es un mecanismo que actúa rápidamente. Algunos de estos síntomas que representan la fatiga pueden ser:

- Aumento de la frecuencia de parpadeo, que a veces puede venir acompañado de pesadez en los párpados y picor en los ojos.
- Bostezos. Este es uno de los síntomas más claros y evidentes de fatiga, en especial si se da de manera recurrente o se prolonga más de lo habitual.
- Sensación de inquietud. Se cambiará frecuentemente de postura buscando la comodidad.
- Sacudidas espontáneas de la cabeza. De esta forma se trata involuntariamente de mantener un estado de alerta.
- Dificultad para concentrarse o permanecer alerta.

La aparición de alguno de estos factores durante la realización de actividades monótonas, como puede serlo la conducción o ciertas labores del día a día, puede llevar a experimentar microsueños, lo que aumenta desproporcionadamente el riesgo de sufrir accidentes.

Entendemos por microsueño [4] un breve episodio de desconexión con el entorno en el que se produce sueño involuntario. Estos pueden ocurrir en cualquier momento del día, incluso mientras se realiza una actividad que requiere de atención. Es por ello que está catalogado como una condición potencialmente peligrosa. Los microsueños pueden aparecer como consecuencia de un descanso deficiente durante los periodos nocturnos, aunque es posible experimentarlos también mientras se efectúa una actividad monótona y rutinaria a lo largo de un espacio de tiempo prolongado. Asimismo, algunos trastornos del sueño pueden provocar episodios de este tipo, como pueden ser apnea del sueño, que se caracteriza por la obstrucción de las vías respiratorias, lo que impide que la persona respire correctamente mientras duerme y que en consecuencia el cerebro no reciba la cantidad suficiente de oxígeno, o bien el insomnio, que impide el descanso adecuado durante la noche.

1.2 Definición del problema

En la actualidad no es complicado encontrar proyectos realizados sobre pulseras de actividad destinados a obtener datos de las mismas. Sin embargo, el principal inconveniente radica en que la mayoría de ellos no posee un propósito específico, sino que por el contrario están destinados a exponer los datos a modo de demostración al usuario, sin el consiguiente estudio de los mismos. Al mismo tiempo, es recalcable el hecho de que buena parte de estas aplicaciones con funcionalidades limitadas están dirigidas a dispositivos concretos, que hoy en día es poco probable el poder conseguirlos en el

mercado, puesto que han quedado obsoletos o las empresas dedicadas a la venta de aparatos electrónicos optan por ofrecer productos más novedosos.

Por otra parte, surge un problema a la hora de extraer la información recabada en la aplicación desarrollada por los fabricantes de este tipo de dispositivos, debido a que en ella es posible visualizarla, pero no obtener registros exportables, por ejemplo, en formato Excel o CSV. Esto dificulta considerablemente la confección de datasets de cara a realizar estudios comparativos de una muestra de sujetos.

1.3 Estado del arte

A continuación se realiza un estudio sobre las aplicaciones análogas al proyecto que han sido encontradas:

1.3.1 Copiloto Mutua

Durante el año 2015, las compañías Samsung Electronics y Mutua Madrileña colaboraron en el desarrollo de una aplicación pensada para la compatibilidad con el smartwatch *Samsung Gear S*. Su función consistía en, gracias a los potentes sensores que este posee, aprender de la forma de conducir del usuario para así detectar posibles situaciones de riesgo durante la conducción. De este modo, la aplicación se convirtió en una especie de asistente para la conducción responsable

Esta aplicación aprende de los hábitos de conducción del usuario, como son la postura, los movimientos y rutinas, la frecuencia cardíaca, etc., gracias a la información proporcionada por el acelerómetro, el giroscopio y el monitor de frecuencia cardiaca del dispositivo, a fin de poder localizar y evitar posibles situaciones de riesgo para el conductor. De esta manera, esta aplicación es capaz de alertar al conductor en caso de detectarse riesgo de somnolencia, pulso cardiaco demasiado elevado o de haberse superado el tiempo máximo de conducción recomendado por la Dirección General de Tráfico (DGT). Para ello, se requiere de una serie de pasos previos, como la detección de pulsaciones en reposo.

Copiloto Mutua nace como fruto del trabajo realizado en la Universidad Pontificia de Salamanca por el profesor Sergio Ríos y su equipo de investigadores, encargados no solo de la programación del software, sino también de la implementación del algoritmo capaz de conocer cuándo el conductor comienza a sentirse cansado a partir de los movimientos que efectúa.

1.3.2 Gadgetbridge

Gadgetbridge es una aplicación de código abierto en continuo desarrollo para plataformas Android, que permite al usuario la configuración y la utilización de una amplia lista de smartwatches y smartbands compatibles, sin necesidad de instalar la aplicación oficial o transmitir datos personales a los servidores del fabricante. Es por esto por lo que en esta aplicación prima la privacidad de los usuarios, ya que, al no existir la necesidad de utilizar aplicaciones oficiales de los dispositivos, los usuarios no tendrán que crear cuentas a través de estas, ni compartir su información privada, que las industrias productoras almacenan y es, en definitiva, de la que se nutren. Por consiguiente, es una potencial alternativa al

software proporcionado por fabricantes como Pebble, Xiaomi, Samsung, etc. con facilidad para gestionar la información recabada de los wearables dedicados.

Gadgetbridge brinda un ecosistema cerrado que posibilita la gestión integral de todos los ajustes de los dispositivos, ya que contiene un gran número de características entre las que se encuentran notificaciones de llamada, sincronización de calendario, alarmas inteligentes, rastreo de la actividad y el sueño, monitorización del ritmo cardiaco, control de la reproducción de música y muchas más.

En los dispositivos que han experimentado actualizaciones en su firmware se han introducido una serie de cambios, siendo el más importante la restricción en la configuración de la clave de autenticación por parte de aplicaciones de terceros. Lo que sucede es que, una vez completado el emparejamiento, la aplicación oficial asigna al reloj o pulsera una clave de autenticación necesaria para obtener la funcionalidad plena de este. Ello obliga al usuario a emparejar el dispositivo con la aplicación oficial en primer lugar, lo que contradice el principio subyacente de Gadgetbridge, aunque desafortunadamente no se ha descubierto aún una solución alternativa.

1.4 Objetivos

El objetivo principal de este proyecto consiste en el desarrollo de una aplicación destinada a plataformas Android, orientada a la detección de fatiga y somnolencia gracias al empleo de dispositivos *wearables,* como los mencionados anteriormente.

Los sistemas seleccionados permiten medir y contrastar ciertas variables fisiológicas del usuario que los lleva puestos, como pueden ser la frecuencia cardiaca, el nivel de oxígeno en sangre, la calidad del sueño, el nivel de estrés, etc., de manera poco invasiva, puesto que el usuario solo deberá llevarlos en su muñeca mientras realice la actividad pertinente.

Los objetivos de este trabajo se desglosan en detalle a continuación:

- Estudio y análisis de los diferentes modelos de smartbands que se encuentran actualmente en el mercado, haciendo hincapié en las diversas variables que estos permiten medir.
- Estudio de otros proyectos similares al que se busca desarrollar.
- Creación de un prototipo de aplicación a través de la cual sea posible detectar la pulsera elegida mediante la tecnología Bluetooth Low Energy (estudiada en profundidad más adelante), realizar una conexión estable y obtener lecturas de los diferentes sensores.
- Implementación de un algoritmo de detección de somnolencia que, usando como fuente de apoyo principal las medidas tomadas del dispositivo, sea capaz de determinar el estado del usuario y actuar en consecuencia, enviando, por ejemplo, vibraciones continuas para alertar al usuario.
- Realización de un breve estudio como apoyo a la implementación a un grupo de usuarios a los que se les ha colocado un dispositivo tomando continuas medidas mientras realizan actividades de su vida cotidiana.

1.5 Conceptualización de la propuesta

El planteamiento llevado a cabo está pensado para su implantación en una empresa o institución dedicada al transporte. Por tanto, se tiene presente al usuario a la hora de crear una interfaz sencilla y amigable para él.

Asimismo, se pretende obtener una aplicación plenamente funcional y de configuración sencilla que permita recabar datos de una pulsera de actividad que el usuario lleve puesta. De esta forma y en tiempo real, la aplicación aprenderá cómo fluctúan las variables fisiológicas del portador del monitor de actividad durante un periodo determinado, y posteriormente detectar los momentos en los que podría estar sufriendo síntomas de cansancio y somnolencia para alertarle de ello y alentarle a tomar un descanso de la actividad que se encuentre realizando.

1.6 Fases

En este apartado se ofrece una visión general de las fases llevadas a cabo a lo largo de la evolución del proyecto.

- Estudio, consistente en la investigación sobre las tecnologías que nos disponemos a utilizar, que incluye la indagación y pruebas sobre aplicaciones relacionadas, con el objeto de extraer la mayor cantidad de información posible acerca del alcance de proyectos similares.
- Análisis, que define los requisitos con el objetivo de precisar exactamente qué se pretende construir.
- Diseño: Tras finalizar las fases previas y con el listado de requisitos bien definido, procedemos a diseñar la arquitectura de la aplicación. Esto incluye diversos diagramas que definirán el diseño de pantallas, interfaz, base de datos, etc.
- Implementación, que corresponde a la programación de la aplicación móvil en Java.
- Pruebas: Con el objetivo de comprobar la calidad y efectividad del software desarrollado, se realizará una fase de comprobación en diferentes dispositivos, a fin de ratificar el correcto funcionamiento de los componentes, ajuste de la interfaz a cada tamaño de dispositivo, entre otros.
- Toma de muestras: Con el fin de ajustar el umbral a partir del cual se detecta un estado de somnolencia, se registra un histórico de pulsaciones a un grupo de usuarios.

1.7 Estructura del documento

Este documento se divide en dos partes: la primera de ellas contiene la memoria que resume el proyecto, mientras que la segunda incluye los anexos que especifican en detalle algunos aspectos del mismo.

La memoria principal consta los capítulos que se presentan a continuación:

Capítulo 1 – Introducción. Este capítulo presenta brevemente la motivación, definición del problema, estado del arte y objetivos generales del proyecto, así como sus distintas fases. Como último apartado, se presenta la estructura de la memoria.

Capítulo 2 – Tecnologías empleadas. En este capítulo se comentan las tecnologías empleadas tanto hardware como software, así como IDE y lenguajes de programación utilizados.

Capítulo 3 – Aplicación Stay Awake. En este capítulo está dedicado a detallar el desarrollo de la aplicación y sus diversas funcionalidades.

Capítulo 4 – Algoritmo para detección de fatiga y somnolencia. A lo largo de este capítulo se describe el algoritmo implementado para la detección de fatiga y somnolencia.

Capítulo 5 – Implementación. Este capítulo se utiliza para hacer énfasis en detalles de la implementación de relevancia.

Capítulo 6– Presupuesto. En este capítulo se detalla el presupuesto para este proyecto en cuanto a costes de material y de recursos humanos.

Capítulo 7 – Conclusiones y líneas futuras. En este capítulo se finaliza la descripción del proyecto con una serie de conclusiones, así como un conjunto de objetivos a futuro que incluir y/o mejorar en la aplicación.

Este documento contiene, además, una serie de anexos que completan y amplían la información ya presentada en los apartados anteriores. Los anexos son los siguientes:

Anexo 1 – Enlace al repositorio de código. Este anexo presenta el enlace al repositorio donde se aloja el código de la aplicación.

Anexo 2 – Enlace a wireframes de la aplicación. En este anexo se incluye el enlace a los primeros bocetos de la aplicación.

Anexo 3 – Enlace a mockups de la aplicación. En este anexo se incluye el enlace a un prototipo de la aplicación realizado como prueba de diseño de la interfaz.

2. Tecnologías empleadas

En este capítulo se lleva a cabo un estudio del hardware y software empleados en el desarrollo de este proyecto, así como los inconvenientes encontrados y soluciones adoptadas. En primer lugar, se realiza un análisis detallado de los dispositivos y otras tecnologías empleados. A continuación, se comentan los problemas encontrados durante el avance del proyecto y, en el caso de haberla, la solución empleada.

2.1 Hardware

Los monitores de actividad son sistemas que registran el movimiento mediante un acelerómetro interno de tres ejes. Los datos que aporta el acelerómetro interno son procesados por un algoritmo, que es capaz de categorizar si el movimiento se debe a un paso dado por el usuario o a un giro, por ejemplo, en la cama. En resumen, detectan movimiento y lo transforman en "pasos", una unidad de medida estándar para estimar la actividad a lo largo del día. Con estos pasos se calculará la distancia recorrida teórica y las calorías consumidas a lo largo del día, en función de la edad, peso, altura y sexo del usuario. Sin embargo, es posible que las pulseras, al llevarlas en la muñeca, contabilicen como pasos ficticios algunas actividades en las que el usuario está quieto y sólo mueve los brazos.

Las pulseras deportivas, además de ser extremadamente útiles para obtener datos sobre el día a día, también se utilizan para programar objetivos. En ellas es posible introducir la cantidad de pasos que se desea andar, por ejemplo, y activar notificaciones que avisen al usuario de cuándo se han o no conseguido. Asimismo, muchas pulseras de actividad permiten controlar las alertas del teléfono y redes sociales, para ver la predicción meteorológica e incluso para descargar aplicaciones. La razón de ello se encuentra en la posibilidad de sincronizar el teléfono móvil con la pulsera.

No obstante, este tipo de tecnologías, que pueden tener una utilidad relativa en algunos casos y siempre bajo supervisión médica, no ofrecen resultados lo suficientemente fiables, por lo que no son aptos para usarse como herramientas de autodiagnóstico.

2.1.1 Mi Band 3

La pulsera de actividad Mi Band 3, del fabricante multinacional Xiaomi es la tercera generación de pulseras de actividad más vendidas. Posee grandes ventajas frente a modelos anteriores, como pueden ser la integración de una pantalla táctil, resistencia al agua, lectura de notificaciones, etc. Esto se muestra en la figura 1.

Por otra parte, se sostiene que el dispositivo no es del todo preciso a la hora de cuantificar la actividad del usuario que lo lleva, es decir, se cuantifican pasos falsos, y ocurre algo similar con las calorías quemadas cuando se realiza una actividad que la pulsera no contempla, pues la medición de este es aproximada. Además, algunos usuarios comentan

también que el sensor de frecuencia cardiaca no es apto para llevar un control estricto, pues los datos no son del todo fiables.

En cuanto a la autonomía de ésta, el fabricante promete una autonomía teórica de 20 días, gracias a la batería de 110mAh. En la práctica se destaca que el hecho de mantener el sensor de frecuencia cardiaca activado de manera continua junto a otras funciones acelera considerablemente el consumo de energía.

Figura 1: Mi Band 3

La tabla 1 muestra la ficha técnica.

PANTALLA	OLED 0,78" táctil, 120 x 80 píxeles
DIMENSIONES Y PESO	17.9 x 46.9 x 12 mm, 20 g
CORREA	Longitud: 247mm, Compatible con 155-216mm
BATERÍA	110 mAh
RESISTENCIA	IP68, hasta 50 metros bajo el agua, -10º a 50º C
CONEXIÓN CON EL MÓVIL	Android 4.4 / iOS 9.0 o superior Mínimo bluetooth 4.0
SENSORES Y CONECTIVIDAD	Bluetooth 4.2, acelerómetro, pulsómetro, NFC (opcional)
PRECIO (APROX.)	**8,33 euros**

Tabla 1: Ficha técnica del modelo Mi Band 3

2.1.2 Mi Smart Band 5

El modelo de pulsera Xiaomi Mi Smart Band 5 es uno de los modelos más novedosos de la marca actualmente. Con un diseño muy similar a los modelos previos, como puede apreciarse en su imagen comercial en la figura 2, cuenta con dos elementos principales, la correa y el dispositivo electrónico propiamente dicho. Los sensores de ritmo cardiaco se sitúan en una leve protuberancia en la parte trasera, además de incorporar un conector magnético que facilita la tarea de recarga.

La pulsera incluye sensores PPG (fotopletismografía) para la lectura del ritmo cardiaco. Además, implementa un sistema de acelerómetro de tres ejes, giroscopio de tres ejes y un motor de rotación para los avisos y despertador. La lectura de pulso tiene un rango de error razonablemente pequeño, lo que permite que esta sea bastante cercana a la real. No obstante, ha de tenerse en cuenta que no es recomendable recurrir a ésta en caso de sufrir alguna cardiopatía.

La principal mejora respecto a modelos anteriores es la compatibilidad con once nuevos modos deportivos en el apartado de entrenamiento, como pueden ser: "ciclismo", "cinta de correr", "elíptica", "yoga", etc.

En este modelo, Xiaomi reduce ligeramente la autonomía, a fin de equipar el producto con más tecnología en idéntico espacio. La autonomía teórica en este caso es de veinte días, aunque resulta prácticamente imposible alcanzar esta cifra si se emplea el dispositivo con todas sus funciones activas.

Figura 2: Mi Smart Band 5

La tabla 2 muestra la ficha técnica.

PANTALLA	OLED a color 1,1 pulgadas (126 x 294 píxeles)
SENSORES	Frecuencia cardíaca (PPG) Acelerómetro de 3 ejes Giroscopio de 3 ejes
RESISTENCIA AL AGUA	5 ATM
AUTONOMÍA	Batería de 125 mAh Autonomía mayor de 14 días (según el fabricante)
CONECTIVIDAD	Bluetooth 5.0
COMPATIBILIDAD	iOS y Android
OTROS	Reconocimiento de 11 deportes Carga magnética Modo de salud femenina Control remoto para hacer fotos
DIMENSIONES	47,4 x 18,6 x 12,7 mm
PRECIO (APROX.)	**24,99 euros**

Tabla 2: Ficha técnica del modelo Mi Smart Band 5

2.1.3 Mi Smart Band 6

La Xiaomi Mi Smart Band 6 es la versión más reciente de la popular gama de pulseras inteligentes del fabricante Xiaomi. Esta incluye algunas mejoras, como el crecimiento de la pantalla o la integración de más modos deportivos. Sin embargo, mantiene el diseño que tanta importancia tiene en su éxito, con una estética cómoda, tal y como se muestra en la figura 3, y peso muy ligero, que permite utilizarla sin molestia alguna para el usuario.

En contraposición a modelos anteriores, los sensores de este dispositivo son bastante precisos. Incluso, este nuevo modelo incluye pulsioxímetro, esto es, un sensor que permite medir el nivel de oxígeno en sangre del usuario. A diferencia del modelo descrito anteriormente, este cuenta con treinta modos deportivos que permiten un mejor ajuste de los sensores en función de la actividad que se realiza.

Además, una gran mejora es el incremento en la capacidad de la batería, que pasa a ser de 125mAh. A pesar de esto, un aumento en las funciones, así como en el tamaño de la pantalla, conllevan también el aumento en el consumo de energía, lo que ofrece como resultado un total de aproximadamente catorce días de autonomía.

Figura 3: Mi Smart Band 6

La tabla 3 muestra la ficha técnica.

PANTALLA	OLED a color 1,56 pulgadas (152 x 486 píxeles) 326 PPP, Brillo máximo de 450 nits
SENSORES	Frecuencia cardíaca (PPG) Saturación de oxígeno en sangre (SpO2) Registro de la respiración durante el sueño Acelerómetro y giroscopio de 3 ejes
RESISTENCIA AL AGUA	5 ATM
AUTONOMÍA	Batería de 125 mAh Autonomía mayor de 14 días (según el fabricante)
CONECTIVIDAD	Bluetooth 5.0
COMPATIBILIDAD	iOS y Android
OTROS	Reconocimiento de 30 deportes Carga magnética

	Modo de salud femenina Control remoto para hacer fotos
DIMENSIONES	47,4 x 18,6 x 12,7 mm
PRECIO (APROX.)	**39,99 euros**

Tabla 3: Ficha técnica del modelo Mi Smart Band 6

2.2 Software

2.2.1 Bluetooth Low Energy

Bluetooth Low Energy [5] es una tecnología de red de área personal inalámbrica diseñada e impulsada por Bluetooth Special Interest Group, destinada a aplicaciones innovadoras, basadas principalmente en IoT (Internet of Things), en diversas industrias, como por ejemplo, en atención médica, entrenamiento, seguridad y entretenimiento en el hogar. Algunas de las características de la tecnología inalámbrica Bluetooth Smart (Low Energy) son:

- Requerimiento de potencia muy bajo.
- Seguridad en las transmisiones gracias al cifrado AES 128.
- Bajo coste.
- Interoperabilidad de múltiples proveedores.
- Radio y velocidad de transferencia mejorados.

Como se muestra en la figura 4, es similar a la pila de protocolos TCP/IP.

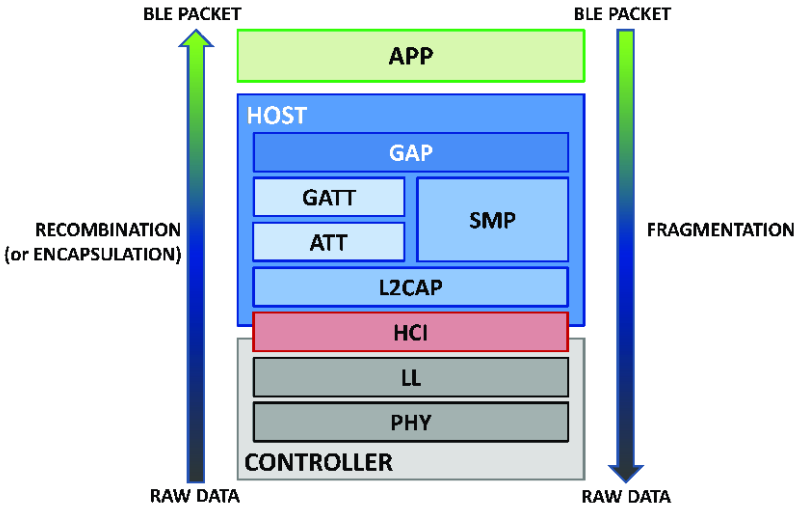


Figura 4: Representación del stack Bluetooth Low Energy

La capa inferior está diseñada para hardware, emparejamiento y encriptación. Las capas superiores (capas de host y aplicación) están encapsuladas para aplicaciones u otros

sistemas operativos. Estas últimas son de especial relevancia para el proyecto, por lo que se detallan a continuación [6]:

Perfil de acceso genérico (GAP): Se utiliza principalmente para controlar la conexión y transmisión de dispositivos. Define cómo los dispositivos BLE son capaces de controlar la detección de otros dispositivos, conexión, etc., lo cual garantiza el buen funcionamiento entre dispositivos. Para regular a bajo nivel el funcionamiento de todos los dispositivos que emplean tecnología BLE, se establecen una serie de modos y procedimientos. Un modo es un estado en el que un dispositivo periférico puede cambiar durante un periodo de tiempo, con el fin de lograr un objetivo o permitir que otro dispositivo realice un procedimiento. Algunos ejemplos de modos son:

- Modos de descubrimiento: No detectable, detectable por tiempo limitado, general-detectable.
- Modos de conexión: No conectable, conectable dirigido, no dirigido-conectable.

En cambio, un procedimiento es una secuencia de acciones llevadas a cabo por un dispositivo central para lograr un objetivo, como puede ser el descubrimiento de otros dispositivos o el establecimiento de conexión con un dispositivo específico. Algunos ejemplos de procedimientos pueden ser:

- Procedimientos de descubrimiento: Descubrimiento limitado, descubrimiento general.
- Procedimientos de conexión: General-conexión, conexión directa.

Los dispositivos BLE pueden operar en uno o más roles del perfil GAP al mismo tiempo, siempre que la capa de enlace lo permita. Este rol impone restricciones y aplica un determinado comportamiento, por lo que generalmente se fija en la etapa del diseño del dispositivo. Se definen dos pares de roles que permiten que los dispositivos se comuniquen entre sí:

Locutor/Observador: Se implementan comunicaciones unidireccionales sin conexión de manera similar a la figura 5.

- Locutor: Envía periódicamente paquetes publicitarios con datos y utiliza la función de anunciante de la capa de enlace.
- Observador: Realiza exploraciones de las emisoras a la escucha de datos publicitarios. Emplea el rol de escáner de capa de enlace.

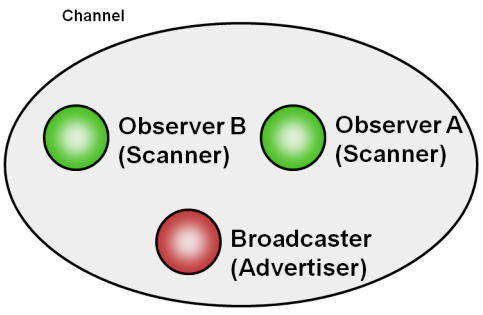


Figura 5: Topología de red en comunicaciones sin conexión

Periférico/Central: Se implementan comunicaciones bidireccionales orientadas a conexión como en la figura 6.

- Periférico: Utiliza el rol de esclavo de capa de enlace. Anuncia mediante el uso de paquetes publicitarios conectables. Está optimizado para consumir la menor cantidad de potencia de procesamiento y memoria, por lo que permite un diseño de bajo coste.
- Central: Emplea el rol de maestro de capa de enlace. Es capaz de establecer y administrar una conexión, además de permitir la conexión a varios dispositivos de manera simultánea.

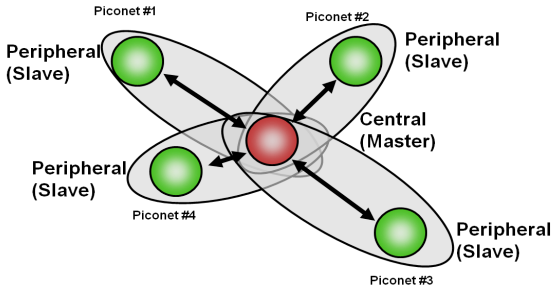


Figura 6: Topología de red en comunicaciones orientadas a conexión

Perfil de atributos genéricos (GATT): El perfil GATT es una especificación que establece cómo se organizan e intercambian pequeñas fracciones de datos, denominados "atributos", a través de un vínculo BLE. Gran parte de los perfiles de aplicaciones de bajo consumo actuales se basan en GATT y están estandarizados por el Grupo de Interés Especial (SIG) de Bluetooth.

Protocolo de atributos (ATT): GATT está basado en el protocolo de atributos (ATT). Por eso, también se denomina GATT/ATT. ATT está optimizado para funcionar en dispositivos BLE. Por esa razón, usa la menor cantidad de bytes posible.

Un servidor GATT contiene datos organizados en forma de atributos. Un atributo es una pieza de datos direccionables y etiquetados; o metadatos sobre el atributo. Cada atributo se identifica de forma exclusiva e inmutable mediante un identificador único universal (UUID), que es un formato estandarizado de 128 bits, empleado para hacer direccionable el atributo. El perfil de atributos genérico establece una jerarquía para la organización de los atributos y se agrupan de la siguiente manera indicada en la figura 7.

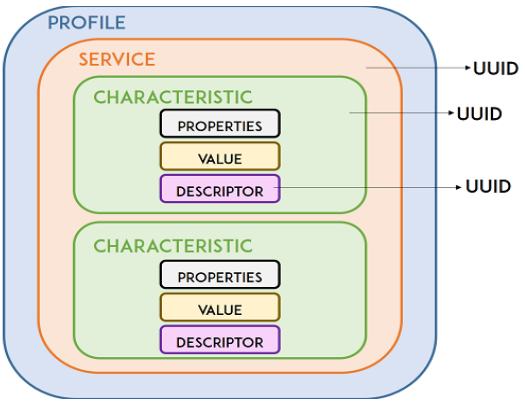


Figura 7: Representación gráfica de servicios y características del GATT

- Servicio: Un servicio es una colección de datos y comportamientos para lograr una función particular. Por ejemplo, se podría encontrar un servicio denominado "Monitor de ritmo cardiaco" que incluya características como "medición del ritmo cardiaco". Se puede clasificar en públicos (definidos por el Bluetooth SIG [7]) y privados (definidos por el proveedor).
- Característica: Una característica es en esencia un contenedor para los datos que ha de contener un mínimo de dos atributos, esto es, atributo de declaración de la característica y atributo de valor de la característica. Opcionalmente puede contener también atributos descriptores.
- Descriptor: Los descriptores son atributos especiales empleados para ampliar los metadatos contenidos en el atributo de declaración. Los descriptores más comunes incluyen:
    - Propiedades extendidas.
    - Descripción legible de la característica para el usuario.
    - Descriptor de configuración de características del cliente (CCCD). Es un descriptor con especial interés, puesto que proporciona un mecanismo para habilitar las actualizaciones iniciadas por un servidor, donde podrá enviar de forma asíncrona valores de características al cliente sin que este tenga que sondearlos. Estas actualizaciones pueden ser de dos tipos: notificaciones (sin acuse de recibo) o indicaciones (con acuse de recibo).

2.2.2 Java

Java es un lenguaje de programación, así como una plataforma informática comercializada y fundada por Sun Microsystems en 1995, y que se ha mantenido vigente hasta la fecha gracias a sus diversas características, que son:

- Simplicidad: A pesar de derivar de los lenguajes C y C++, Java no cuenta con las características más confusas y menos utilizadas de estos.
- Orientado a objetos: El enfoque orientado a objetos es uno de los estilos más populares de programación. Permite diseñar el software de manera que los diferentes tipos de datos estén asociados a sus operaciones respectivas.
- Distribuido e independiente de la plataforma: Java proporciona una gran biblioteca estándar, así como las herramientas necesarias para que los software sean distribuidos.
- Independiente de la plataforma: Los programas escritos en lenguaje Java pueden ejecutarse en cualquier hardware, lo que lo hace portátil y accesible.
- Seguro y sólido: La plataforma que proporciona es muy segura, además de garantizar canales de comunicación fiables, capaces de proteger la privacidad de los datos.
- Multihilo: Es posible llevar a cabo varias tareas simultáneamente en un mismo programa, lo que permite mejorar el rendimiento y la velocidad de ejecución.

Algunas de las ventajas que aporta Java son:

- Adaptabilidad a cualquier dispositivo.
- Posibilidad de diseñar cualquier aplicación o elemento gracias al uso de plantillas.
- Creación de páginas web dinámicas.
- Posibilidad de añadir audio, elementos multimedia, bases de datos y otras funciones.

2.2.3 Android Studio

Android Studio es un entorno de desarrollo integrado (IDE) oficial para la plataforma Android, dedicada a la implementación de aplicaciones, basado en IntelliJ IDEA, un IDE para el desarrollo de programas informáticos. Además de las diferentes funcionalidades y herramientas para desarrolladores, Android Studio ofrece otras ventajas como:

- Un sistema de compilación flexible basado en Gradle, una herramienta que permite la automatización de compilación de código abierto.
- Compatibilidad con sistemas operativos Windows, Mac, Linux y Chrome OS
- Un emulador con diversas funciones.
- Un entorno unificado para todos los dispositivos Android.
- Posibilidad de insertar cambios en el código y recursos a la app en ejecución, sin necesidad de reiniciarla.
- Integración con GitHub.
- Herramientas de Lint para la detección de problemas de rendimiento, usabilidad y compatibilidad entre versiones.
- Compatibilidad con C++ Java y Kotlin.

2.2.4 SQLite

De cara a mantener la persistencia de los datos recogidos de los dispositivos, y por consecuencia obtener un aumento en la fiabilidad de la aplicación, se tomó la decisión de incorporar un sistema de gestión de bases de datos relacional denominado **SQLite**. Esta base de datos resulta ideal para las necesidades específicas de la aplicación en desarrollo, gracias a sus características:

- SQLite posee una pequeña memoria y es totalmente autocontenida, es decir, que apenas incluye dependencias externas.
- Su librería está escrita en C.
- Su código fuente es de dominio público.
- Realiza operaciones de manera eficiente, mucho más rápido que otras plataformas como MySQL y PostgreSQL.
- Es multiplataforma y sus bases de datos pueden ser portadas de manera sencilla, sin emplear ningún tipo de configuración o administración.
- Es compatible con los principios ACID, Atomicidad, Consistencia, Aislamiento y Durabilidad.

3. Aplicación Stay Awake

Este capítulo está destinado a describir el procedimiento seguido en el transcurso de la implementación de la aplicación, elemento principal de este proyecto.

3.1 Diseño de la interfaz de usuario

Para obtener un diseño sencillo y amigable al usuario, se optó por una interfaz que, en primera instancia, contenga cuatro botones diferentes como se aprecia en la figura 8, cada uno de los cuales tiene una funcionalidad asociada, además de un badge que indica el estado de la aplicación.

- **Bluetooth on:** Al ser pulsado, este botón dispara un Intent de Java que solicita encender el Bluetooth al teléfono móvil en el que se ejecuta la aplicación, siempre y cuando se disponga de dicha tecnología.
- **Bluetooth off:** Al pulsarse, de manera similar al anterior, se desactiva esta función.
- **Badge de estado:** Por cada operación que la aplicación esté realizando, se mostrará en pantalla el estado de la misma, por ejemplo: “Discovering”, “Bluetooth enabled”, “Connecting”, etc.
- **Show bonded devices:** Cuando se escoge esta opción, se listan a continuación todos los dispositivos que han sido previamente asociados vía Bluetooth con el teléfono, y que este tiene almacenados para ser vinculados automáticamente si son detectados.
- **Scan devices:** Al ser elegida esta opción, en primer lugar debe comprobarse que se ha habilitado el servicio Bluetooth previamente, y, a continuación, se muestra un diálogo de alerta que permite navegar a los ajustes de la aplicación y activar la localización en caso de que no lo esté, para luego regresar a la página principal. A continuación, se inicia el escaneo, por lo que se listarán todos aquellos dispositivos que contengan en su nombre la palabra “Band”, de tal manera que no interfieran otros dispositivos no compatibles con la aplicación.

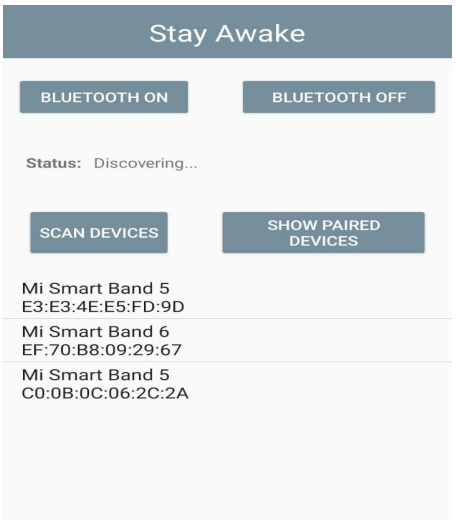


Figura 8: Pantalla principal de la aplicación

Al seleccionarse un dispositivo, se realiza una conexión con este, por lo que se navega a la pantalla específica para el dispositivo, similar a la mostrada en la figura 9, que incluye los siguientes elementos:

- **Nombre del dispositivo:** En la cabecera de la pantalla se indica el nombre del dispositivo al que se ha conectado la aplicación.
- **Disconnect:** Si este botón es pulsado se finaliza inmediatamente la conexión con el dispositivo y se regresa a la pantalla principal. Además, el fichero que contiene los datos recogidos es trasladado al directorio raíz del almacenamiento interno del teléfono móvil.
- **Nivel de batería:** Junto al botón anterior, se puede observar el nivel de batería obtenido de la pulsera junto a un icono representativo.
- **Ritmo cardiaco:** Posteriormente se muestra bajo el icono correspondiente el nivel de pulsaciones que mide la pulsera de actividad. Este valor cambia en tiempo real según las medidas tomadas, y en caso de no recibirse ninguna, se sustituye por el símbolo "-".

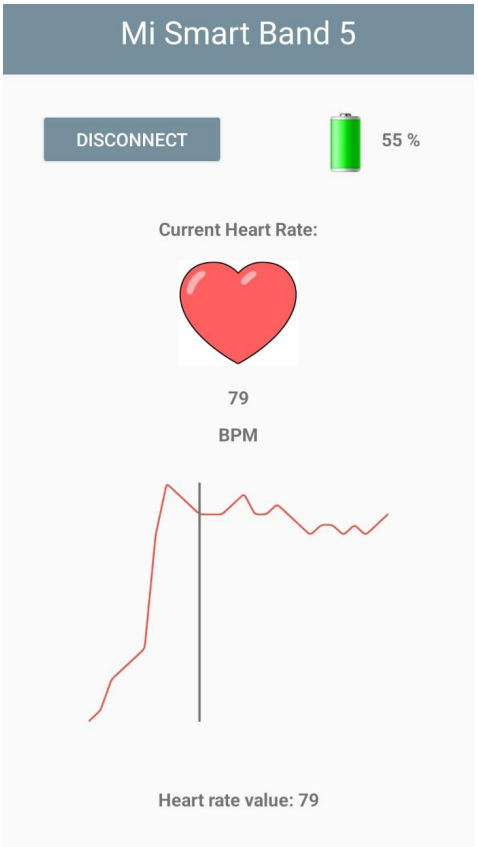


Figura 9: Pantalla de un dispositivo

Posteriormente se incluyó una pantalla intermedia mostrada en la figura 10, para aquellos dispositivos que requieren autenticación, en la que se incluye un cuadro para entrada de texto, donde se debe escribir la clave correspondiente, y un botón para enviar la clave y cambiar de pantalla.

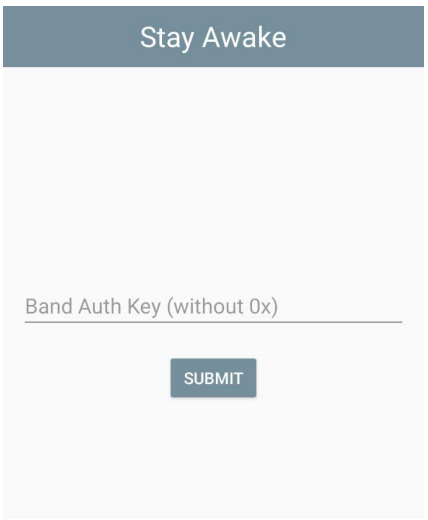


Figura 10: Pantalla de autenticación

3.2 Escaneo y conexión

Para facilitar al usuario la tarea de detectar los dispositivos cercanos que se encuentran en modo "detectable", así como para establecer una conexión con ellos, se emplea una estructura de tipo Bluetooth Adapter, que proporciona los métodos adecuados para estas funciones. Esta estructura representa el adaptador Bluetooth del dispositivo local. El Bluetooth Adapter permite realizar tareas fundamentales de Bluetooth, como iniciar la detección de dispositivos, consultar una lista de dispositivos vinculados (emparejados), o instanciar un dispositivo Bluetooth utilizando una dirección MAC conocida. Básicamente, este es el punto de partida para todas las acciones de Bluetooth.

3.3 Autenticación en smartbands

Generalmente, para acceder a determinadas características de ciertos modelos de smartband, es preciso realizar un proceso de autenticación, que además puede variar en función del modelo de firmware que esta emplee.

Dentro del perfil GATT, el servicio dedicado a la autenticación, entre otras cosas, es aquel con UUID "0xfee1". De entre las características que este posee, aquella específica para este procedimiento es la de UUID "00000009-0000-3512-2118-0009af100700". Además, antes de comenzar el envío de datos, es necesario habilitar las notificaciones de esta característica mediante el descriptor con UUID "0x2902". Esto se consigue enviando los bytes {0x01, 0x00} al descriptor correspondiente.

En dispositivos cuyo firmware no ha sido actualizado, como pueden ser **Mi Band 2** o **Mi Band 3**, se debe realizar un proceso sencillo de autenticación, que se describe a continuación, que, una vez realizado, hará que las conexiones próximas sean más rápidas:

Una vez habilitadas las notificaciones, se envía a la característica de autenticación un array encabezado con los bytes {0x01, 0x00}, seguido de los 16 bytes que conforman la clave secreta de la pulsera, es decir, 128 bits. En este caso particular se envía la siguiente clave,

que corresponde a la secuencia “0123456789@ABCDE”: “{0x01, 0x08, 0x30, 0x31, 0x32, 0x33, 0x34, 0x35, 0x36, 0x37, 0x38, 0x39, 0x40, 0x41, 0x42, 0x43, 0x44, 0x45}”, ya que este modelo no requiere una clave específica.

El dispositivo debe enviar una respuesta. Esta debe ser {0x10, 0x01, 0x01}, lo cual indica que el primer paso se ha completado con éxito. La respuesta se compone de tres dígitos en hexadecimal: el primero, 16 en decimal, indica que se está recibiendo una respuesta; a continuación, 1 en decimal, que indica que se ha completado el **primer** paso de la autenticación, y, por último, 1 en decimal, que señala el éxito de la operación. Cabe mencionar que esta respuesta se obtiene al presionar el botón de la pulsera, o en caso contrario, la respuesta no será exitosa.

Una vez obtenida esta respuesta, se realiza el segundo paso de la autenticación, es decir, realizar una petición de la clave aleatoria que genera la pulsera. Para ello, se deben enviar los siguientes bytes: {0x02, 0x08}.

La respuesta, esta vez del segundo paso, se compone por: “{0x10, 0x02, 0x01, (16 bytes de clave secreta)}”. El primer dígito, 16 en decimal, indica que se está recibiendo una respuesta; a continuación, 2 en decimal, señala que se trata de una respuesta referente al **segundo** paso de la autenticación, y, por último, 1 en decimal, indica el éxito de la operación.

Una vez recibida la clave generada por la pulsera, se procede a realizar el tercer y último paso de la autenticación, que consiste en enviar a la característica un array con los siguientes elementos: “{0x03, 0x08, (clave encriptada)}”, donde *clave encriptada* (encrypted key) corresponde a la clave enviada en el primer paso de autenticación cifrada, junto a la *clave secreta*, enviada en el segundo paso empleando un cifrado de tipo AES . El cifrado de tipo AES se sirve de un algoritmo basado en sustituir, permutar y transformar linealmente bloques de datos de 16 bytes. En cada una de las rondas de este se calcula una clave circular única a partir de la clave de cifrado y se incorpora en los cálculos. Para este caso concreto de autenticación se emplea un modo de operación llamado ECB (Electronic Code Book), en el cual los mensajes se dividen en bloques y cada uno de ellos se cifra por separado, usando una misma clave K.

La última respuesta debe ser {0x10, 0x03, 0x01}, que apuntan a que el **tercer** paso de la autenticación se ha completado con éxito y, por tanto, la pulsera se encuentra vinculada. En el caso de que la respuesta estuviera conformada por otra combinación de dígitos, significaría que la autenticación no se ha completado con éxito.

<u>Autenticación basada en servidor:</u>

En aquellos dispositivos con firmware actualizado se requiere de una autenticación basada en servidor, esto quiere decir que es necesario emplear previamente la aplicación oficial del fabricante (Mi Fit, Zepp, Amazfit…) para realizar el emparejamiento inicial y obtener la clave de autenticación. Actualmente, algunos de los dispositivos que emplean este

tiempo de autenticación son **Mi Band 4, Mi Smart Band 5, Mi Smart Band 6, Amazfit Bip Lite, Amazfit GTR**, entre otros.

Cabe mencionar que inicialmente pretendía emplearse el modelo **Mi Smart Band 6** para la realización del proyecto. Sin embargo, se comprobó que el método de autenticación necesario para este emplea un protocolo criptográfico para el intercambio de claves denominado *Diffie-Hellman* [8], basado en la idea de la utilización de una clave secreta generada por los interlocutores, que cada uno de ellos puede conocer gracias a que cada uno posee una clave pública y una secreta. Este protocolo emplea, además, criptografía de curvas elípticas, lo cual dificulta considerablemente la autenticación. Por tanto, al sobrepasarse los límites del alcance del proyecto, se tomó la decisión de emplear el modelo **Mi Band 3** y posteriormente, **Mi Smart Band 5**, debido a que el anterior es un modelo ya obsoleto y difícil de obtener en el mercado.

Para adquirir dicha clave se ha empleado una versión modificada de la aplicación *Mi Fit* [9]. Una vez instalada, se debe pulsar el botón "+" para realizar la búsqueda de dispositivos. Al realizarse el emparejamiento de este, se encontrará un directorio en la raíz del almacenamiento interno del teléfono, denominado "freemyband", que contendrá un fichero con la dirección MAC del dispositivo emparejado y la clave de autenticación de este.

Una vez conocida la clave, comienza el proceso de autenticación, realizando una petición de la clave aleatoria que genera la pulsera. Para ello, se deben enviar los siguientes bytes: {0x02, 0x00}.

La respuesta que debe recibirse es: "{0x10, 0x02, 0x01, (16 bytes de clave secreta)}". El primer dígito, 16 en decimal, indicará que se está recibiendo una respuesta; a continuación, 2 en decimal, señalará que se trata de una respuesta referente al **segundo** paso de la autenticación, y, por último, 1 en decimal, indicará el éxito de la operación.

Al recibir la clave generada por la pulsera, se procederá a realizar el tercer y último paso de la autenticación, que consistirá en enviar a la característica, un array con los siguientes elementos: "{0x03, 0x00, (clave encriptada)}", donde la *clave encriptada* corresponde a la clave enviada en el primer paso de autenticación cifrada, junto a la *clave secreta*, enviada en el segundo paso, empleando un cifrado de tipo AES en modo de operación ECB (Electronic Code Book).

La última respuesta debe ser {0x10, 0x03, 0x01}, que apuntan a que el **tercer** paso de la autenticación se ha completado con éxito y, por tanto, la pulsera se encuentra vinculada. En caso de recibir algún otro dígito como respuesta, indicará que la autenticación no ha sido exitosa.

3.4 Lectura de valores de la pulsera de actividad

Una vez completada la autenticación del dispositivo, es posible acceder a las siguientes características:

- **Información de la batería:** Esta característica, perteneciente al servicio GATT “0xfee0”, informa de todo lo relativo al estado de la misma, cuyo UUID es “00000006-0000-3512-2118-0009af100700”. Al leer de esta característica se recibe una secuencia similar a la siguiente:

  0F-3B-00-E3-07-04-0E-10-27-11-0C-E3-07-04-0D-13-19-16-0C-64

  - Donde **0F** concuerda con la cabecera de la respuesta.
  - En este caso **3B** corresponde al nivel de batería en el momento en el que se recibe la información (59%).
  - Los siguientes dígitos (**00**) indican que se está utilizando la batería. En caso de que el dispositivo estuviera cargando, cambiaría a **01**.
  - A continuación la secuencia **E3-07-04-0E-10-27-11** se refiere a la fecha y hora en el momento del envío de información.
  - El siguiente valor, **0C** informa del número de ciclos de carga.
  - Por último, el valor **64** corresponde al porcentaje de batería al que se llegó la última vez que cargó el dispositivo.

- **Monitor de ritmo cardiaco:** Para esta funcionalidad se van a emplear las siguientes características pertenecientes al servicio "0000180d-0000-1000-8000-00805f9b34fb":
  - **Característica de control:** "00002a39-0000-1000-8000-00805f9b34fb"
  - **Característica de medida:** "00002a37-0000-1000-8000-00805f9b34fb"

  Y esta otra relativa al servicio “0xfee0”.

  - **Sensor:** "00000001-0000-3512-2118-0009af100700"

  Además de la característica de En primer lugar, se ha de enviar la siguiente secuencia de bytes al sensor: {0x01, 0x03, 0x19}. Al recibir la respuesta, se habilitarán las notificaciones para la característica de medida. Una vez hecho esto, se envía una secuencia de bytes {0x15, 0x01, 0x01} a la característica de control para comenzar la toma de medidas. Al cabo de unos segundos, se empezarán a recibir valores. Cada vez que se obtenga un nuevo valor, se recibirá un evento de cambio en la característica 0x2a37. Estos valores recibidos se almacenarán de manera persistente en una base de datos, a la vez que se mostrarán y actualizarán en tiempo real en la aplicación.

  A continuación, se ha de establecer un contador que cada 12 segundos envíe un byte {0x16} a la característica de control. Esto mantiene el sensor continuamente en funcionamiento.

  Cuando se finaliza la conexión con el dispositivo, se ha de enviar la secuencia {0x15, 0x01, 0x00} a la característica de control para desconectar el sensor.

- **Notificaciones:** Esto se realiza mediante un método muy sencillo. Basta con escribir los bytes con el valor {0x02, 0x01} para el caso de la Mi Band 3, o {0x03, 0x01} para la Mi Smart Band 5, a la característica perteneciente al servicio GATT “0x1802”, cuyo

UUID es “00002a06-0000-1000-8000-00805f9b34fb”. De esta manera, la pulsera comenzará a vibrar del mismo modo que si se estuviera recibiendo una llamada.

- **Giroscopio:** De cara a obtener los datos relativos al movimiento de la pulsera, se emplean dos características correspondientes al servicio “0xfee0”:
  - **Característica de sensor:** "00000001-0000-3512-2118-0009af100700"
  - **Característica sensor de medida:** "00000002-0000-3512-2118-0009af100700"

De manera similar a la toma de medidas de frecuencia cardíaca, en primer lugar se ha de enviar la siguiente secuencia de bytes al sensor: {0x01, 0x01, 0x19}. Al recibir la respuesta correspondiente, se envía el byte {0x02} a la misma característica. Junto a la siguiente respuesta, se deben habilitar las notificaciones para el sensor de medida, donde se recibirán los datos tomados del giroscopio. El procedimiento anterior deberá ser repetido cada 30 o 60 segundos para mantener el sensor en funcionamiento.

Se reciben 10 paquetes de datos por segundo, que deben ser interpretados de la siguiente manera, siendo un ejemplo de secuencia la siguiente:

01-20-39-00-CE-00-47-00-5C-00-9F-00-33-00-88-00-83-00-58-00

- Donde el primer byte siempre será **01**, que corresponde al tipo de cabecera.
- En este caso el segundo byte es un contador que indica el número del paquete, **20** en este caso.
- Los siguientes bytes son los valores hexadecimales para los vectores X, Y, Z, donde cada valor está compuesto por dos bytes. El primer byte de cada vector es el valor, y el segundo byte es **00** o **ff**, lo que representa valores positivos o negativos.
- En un paquete de datos puede haber 1, 2 o 3 valores de vector XYZ.

3.5 Persistencia de datos

Como se mencionó en el capítulo anterior, se emplea el sistema SQLite para preservar los datos recogidos de la pulsera en el almacenamiento interno del teléfono. A continuación, se muestra en la figura 11 un esquema simple de cómo se estructura dicha base de datos:

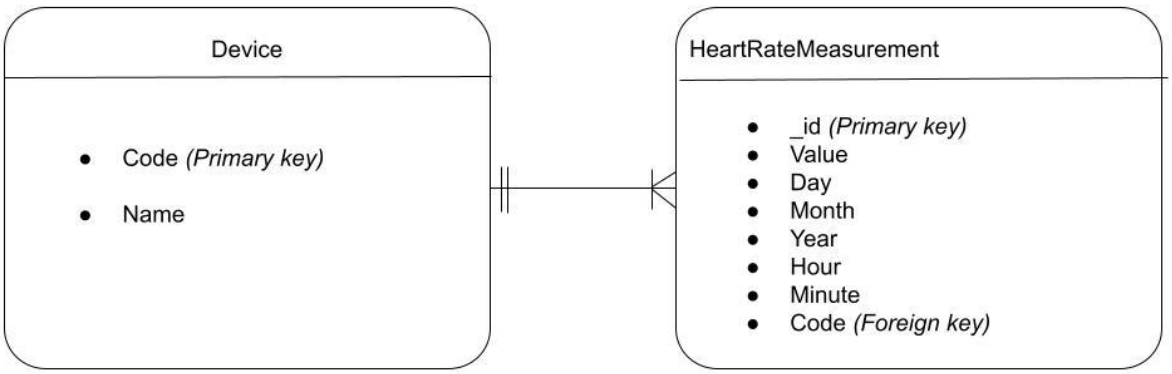


Figura 11: Esquema de base de datos para las medidas recogidas.

La primera parte de la figura 11 corresponde a un dispositivo que posee un identificador único (que podría corresponder a la dirección MAC de este) y su nombre asociado.

La segunda parte de la figura 11 corresponde a una medida realizada por el sensor de frecuencia cardiaca, que contiene los siguientes campos:

- Un identificador único como clave primaria.
- El valor recogido de la medida.
- El día en el que se tomó la medida.
- El mes en el que se tomó la medida.
- El año en el que se tomó la medida.
- La hora a la que se realizó la medida.
- El minuto exacto en el que se realizó la medida.
- La dirección del dispositivo desde el que se realiza la medida.

Se puede apreciar una relación entre ambas partes de tipo “uno a muchos”, que señala que a un dispositivo pueden corresponderle gran cantidad de datos relativos a mediciones de frecuencia cardiaca.

Además, se ha incluido en la base de datos una partes con la siguiente información relativa a la primera aproximación del algoritmo para detección de somnolencia:

- Número de grupos de medidas.
- Número de medidas por grupo.
- Número de medidas descartadas.

Como se comentó previamente, estos datos se guardan en el almacenamiento interno del teléfono, y por tanto no son visibles al usuario, a menos que este haya obtenido permisos de superusuario previamente. Es por esto por lo que esos datos han de ser trasladados al almacenamiento externo, visible para el usuario, que puede ser una tarjeta SD o la memoria del móvil, antes de finalizar la ejecución de la aplicación, así como solicitar permiso al usuario para escribir en dicho almacenamiento.

3.6 Visualización de datos

Para facilitar la visualización de los datos obtenidos se dibuja un gráfico de líneas compuesto por una lista enlazada, una estructura de datos de tipo *first in, first out* (FIFO) donde se almacenan las primeras 200 medidas, una cantidad que simplifica la tarea de comparar la tendencia de los valores recabados, y, una vez sobrepasado el límite de medidas se comienzan a eliminar las primeras para ser sustituidas por las nuevas que se reciben, y así es posible actualizar el gráfico en tiempo real.

El tipo de gráfico que se ha incluido, como puede verse en la figura 12, ha sido elegido por cuestiones de compatibilidad con la versión de SDK de la aplicación. Esto imposibilita añadir los valores recibidos en los ejes x e y. En cambio, al pulsar y arrastrar el dedo por el gráfico se puede ver una línea que muestra el valor en el punto en el que el usuario la sitúa.

Figura 12: Gráfico de valores para frecuencia cardiaca

4. Algoritmo para detección de fatiga y somnolencia

En el capítulo anterior se indicó el empleo de un algoritmo capaz de inferir el estado del usuario en cuanto a somnolencia, gracias a las variables fisiológicas que es posible extraer de la pulsera de actividad.

En este capítulo se pretende enfatizar en la implementación de este algoritmo así como en las variables que influyen en él y las distintas aproximaciones a las que se ha llegado a lo largo del proyecto.

4.1 Pulsaciones

Es bien conocido que la frecuencia cardiaca recomendada en adultos [10] debe situarse entre 60 y 100 pulsaciones por minuto. Esta cantidad puede ser variable en función de respuestas emocionales a situaciones de miedo, adrenalina, etc., posición del cuerpo, medicamentos consumidos, condición física, así como otros factores. Sin embargo, una frecuencia inferior a 60 suele denominarse "bradicardia", y suele darse en momentos de inactividad, así como en personas con muy buena forma física. Por otra parte, superar las 100 pulsaciones por minuto se denomina "taquicardia", y suele conllevar riesgo cardiaco, ya que implica un sobreesfuerzo del corazón, lo que aumenta su desgaste.

Según expertos [11], mientras una persona duerme, lo más adecuado es que su ritmo cardiaco se encuentre alrededor de 50 pulsaciones por minuto, un 8% menos que en estado de vigilia o reposo despierto. En cambio, durante el reposo se estima que el ritmo cardiaco debe estar en torno a las 60 pulsaciones por minuto si la salud cardiovascular es adecuada.

4.1.1 Primera aproximación al algoritmo

Conociendo estos datos, en un primer acercamiento al algoritmo definitivo, se procedió de la siguiente manera:

- **Descarte de medidas**

Durante las primeras tomas de medidas se detectaron irregularidades en los primeros valores obtenidos del sensor. Esto puede deberse a que sean necesarios para la pulsera unos segundos que permitan la estabilización del sensor de la misma. Por ello, se tomó la decisión de descartar las primeras 15 o 20 medidas, para así evitar anomalías que puedan afectar al correcto funcionamiento del algoritmo.

- **Creación de grupos**

Completada la fase de estabilización del sensor, se busca crear grupos donde se almacenan las medidas de frecuencia cardiaca. Cada grupo debe contener el mismo número de medidas, que puede variar en función de cuán grandes se deseen los grupos. En este caso,

se crean aproximadamente 10 grupos de medidas con el fin de obtener una tendencia que pueda considerarse fiable.

- **Cálculo de la media**

Una vez el grupo alcance el valor máximo de medidas que puede albergar, se descarta el 25% de valores más grandes, así como el 25% de valores más pequeños. Esto evita la posibilidad de obtener falsos positivos por medidas erróneas. A continuación, se calcula la media aritmética de las medidas restantes y se almacena en una estructura de datos empleada en el siguiente paso.

- **Cálculo de la tendencia**

Obtenidos los valores medios de cada uno de los grupos, se procede a calcular la tendencia por el método de las dos medias [12]. Este método se basa en separar las observaciones en dos grupos de tamaño similar y calcular la media aritmética de cada uno de estos, obteniéndose así dos puntos. La línea de tendencia se dibuja haciendo pasar una recta por los dos puntos calculados. Esta recta se obtiene mediante la expresión general donde P_1 (x_1, y_1), P_2 (x_2, y_2): $y - y1 = \frac{y1-y2}{x1-x2} \cdot (x - x1)$

Seguidamente, se separan las observaciones en dos grupos, calculando la media de cada uno de ellos, como en el supuesto incluido en la tabla 4:

	Grupo	**Valor medio**	**Grupo**	**Valor medio**
	1	70	4	66
	2	68	5	63
	3	64	6	61
Media	**2**	**67**	**5**	**63**

Tabla 4: Ejemplo para el cálculo de la tendencia

De esta manera se calcula que P_1 (2, 67), P_2 (5, 63), y es posible aplicar la fórmula, tal que:

$$y = \frac{63-67}{5-2} \cdot (x - 2) + 67 \Rightarrow y = -1.333x + 69.666$$

Donde el valor -1.333 corresponde a la pendiente de la recta, y posibilita el hecho de realizar predicciones dando valores a la variable x.

En este punto surgió un problema a la hora de determinar cuál debe ser el mínimo valor que puede tomar la pendiente que indica el instante en que el usuario presenta síntomas de somnolencia. Esto intentó mediante los valores recabados en las distintas pruebas a un grupo de sujetos.

4.1.2 Segunda aproximación al algoritmo

Para implementar una mejora en este algoritmo, se modificó el procedimiento de esta forma:

En primer lugar, del mismo modo que anteriormente, se descartan las primeras medidas en concepto de estabilización del sensor. A continuación, se almacenan valores de frecuencia cardiaca en una estructura de datos durante un periodo de tiempo correspondiente a la fase de aprendizaje de aproximadamente 30 minutos. Por ello, se ha maximizado este primer periodo, que llega a ocupar la totalidad del intervalo de tiempo en el que, según [13], existe menor probabilidad de presentar síntomas de somnolencia al volante. Esto se debe a que se pretende determinar una medida que represente de la manera más fiable posible el estado del ritmo cardiaco del usuario.

Una vez expirado este tiempo, se calcula la media de los valores, así como la desviación típica. Esto se hace con el fin de detectar picos de valores que puedan corresponder a incrementos por determinadas actividades o emociones que pueda experimentar el usuario que pudieran invalidar la muestra. En el caso de que esta desviación sea inferior a 10, es decir, todos los valores de la muestra estén comprendidos en un rango de 20 valores, se tomará como una muestra aceptable y se procederá con el siguiente paso. Se toma el valor 10 para la desviación, ya que está relacionado con la mitad de lo que se considera el índice de recuperación más bajo registrado [14] para asegurar que el usuario se encuentra en estado de reposo:

- Índice de recuperación de la frecuencia cardíaca superior a 70 es el de atletas de élite que se encuentran en plena forma.
- Índice de recuperación entre las 50-70 pulsaciones corresponden a atletas de subélite o aficionados que están en muy buena forma.
- Los valores comprendidos entre las 40-50 pulsaciones corresponden a atletas aficionados bien entrenados.
- Índice de recuperación de 30-40 pulsaciones se dan en personas que realizan un entrenamiento poco regular o al principio de temporada.
- De 20 a 30 pulsaciones corresponden a personas que entrenan de forma irregular o que están sobreentrenando.
- Un índice de recuperación por debajo de las 20 pulsaciones corresponde a personas en muy mala forma física.

En caso contrario, se aumentará esta ventana de tiempo en 5 minutos, tiempo que se corresponde con una medición de corto plazo [15] hasta obtener una desviación admisible.

En el siguiente paso, se almacenarán valores durante un intervalo de 10 minutos, correspondiente a dos mediciones a corto plazo y, como anteriormente, se calcularán la media y la desviación estándar de estos. Si esta resulta inferior a 10, se comparará la diferencia entre las medias de los dos períodos en cuestión de porcentajes. En el caso de que el porcentaje en el que difieren sea superior a un 6%, que corresponde con un porcentaje ligeramente inferior al que se indica en [16], así como que la tendencia calculada en este periodo resulte negativa, se activará una alerta simultáneamente en la pulsera y en el teléfono móvil. En caso de que la desviación indique una fluctuación muy grande de los valores, se aumentará el tiempo de la ventana en 5 minutos.

Si persiste este estado, es decir, si el porcentaje sigue siendo superior, así como la tendencia negativa, se volverá a activar la alarma en el próximo intervalo de tiempo.

4.2 Movimiento

Tal y como se propuso para la aplicación “Copiloto Mutua” se pretendía aprender acerca de los hábitos de conducción del usuario, postura, movimientos y/o rutinas empleando la información proporcionada por el acelerómetro y el giroscopio.

Para ello, es necesario extraer y procesar una serie de características y calcular la correlación entre ellas como se indica en [17].

Se concluyó que este apartado permanecerá en fase experimental, y no se agregará al algoritmo debido a que, en primer lugar, sólo ha sido posible obtener esta información en el modelo Mi Band 3. Esto se debe a que en este modelo, sólo se incluye un acelerómetro de tres ejes que, en el caso de no existir aceleración, los datos que proporciona equivalen a los de un giroscopio. En cambio en modelos superiores se proporcionan por separado acelerómetro y giroscopio.

En segundo lugar, no se han encontrado proyectos análogos para el modelo principal que se está empleando, Mi Smart Band 5, y por tanto no se ha encontrado una forma de extraer estos datos.

5. Implementación

En los capítulos anteriores se describió la aplicación desarrollada para este proyecto. En este capítulo se pretende enfatizar en los detalles más importantes del código de la misma haciendo uso de capturas de pantalla.

5.1 UUIDs de servicios y características

Como se muestra en la figura 13, en un fichero aparte se creó una clase Java destinada a contener los identificadores de los servicios, características y descriptores necesarios para la implementación y de esta forma hacer más sencillo el acceso a estos en el código principal.

```
public static class MiBand {
    //Services
    public static UUID MiBand_Service_UUID = UUID.fromString("0000fee1-0000-1000-8000-00805f9b34fb");
    public static UUID MiBand1_Service_UUID = UUID.fromString("0000fee0-0000-1000-8000-00805f9b34fb");
    public static UUID HR_Service_UUID = UUID.fromString("0000180d-0000-1000-8000-00805f9b34fb");
    public static UUID Alert_Service_UUID = UUID.fromString("00001802-0000-1000-8000-00805f9b34fb");

    //Characteristics
    public static UUID Alert_Characteristic_UUID = UUID.fromString("00002a06-0000-1000-8000-00805f9b34fb");
    public static UUID Auth_Characteristic_UUID = UUID.fromString("00000009-0000-3512-2118-0009af100700");
    public static UUID Battery_Level_UUID = UUID.fromString("00000006-0000-3512-2118-0009af100700");
    public static UUID Measurement_Characteristic_UUID = UUID.fromString("00002a37-0000-1000-8000-00805f9b34fb");
    public static UUID Sensor_Characteristic_UUID = UUID.fromString("00000001-0000-3512-2118-0009af100700");
    public static UUID Sensor_Measure_Characteristic_UUID = UUID.fromString("00000002-0000-3512-2118-0009af100700");
    public static UUID Control_Characteristic_UUID = UUID.fromString("00002a39-0000-1000-8000-00805f9b34fb");

    //Descriptor
    public static UUID Descriptor_UUID = UUID.fromString("00002902-0000-1000-8000-00805f9b34fb");
```

Figura 13: Clase para almacenar UUIDs

5.2 Autenticación

El código que se puede apreciar en la figura 14 corresponde al paso de autenticación descrito en el *capítulo 4* una vez recibida la clave aleatoria que genera la pulsera. Se pueden distinguir dos partes, que concuerdan con los dos tipos de autenticación ya comentados.

```
private void executeAuthorisationSequence(BluetoothGatt gatt, BluetoothGattCharacteristic characteristic) {
    byte[] value = characteristic.getValue();
    if (value[0] == 16 && value[1] == 2 && value[2] == 1) {
        try {
            if (name.contains("Mi Band 2") || name.contains("Mi Band 3")) {
                secret_key = Arrays.copyOfRange(value, start: 3, end: 19);
                @SuppressLint("GetInstance") Cipher cipher = Cipher.getInstance("AES/ECB/NoPadding");
                String CIPHER_NAME = "AES";
                SecretKeySpec key = new SecretKeySpec(mi_band_3_key, CIPHER_NAME);
                cipher.init(Cipher.ENCRYPT_MODE, key);
                byte[] bytes = cipher.doFinal(secret_key);
                byte[] rq = Arrays.copyOf(new byte[]{0x03, 0x08}, newLength: 2 + bytes.length);
                System.arraycopy(bytes, i: 0, rq, i1: 2, bytes.length);

                characteristic.setValue(rq);
                gatt.writeCharacteristic(characteristic);
            } else if (name.contains("Mi Band 4") || name.contains("Mi Smart Band 5")) {
                byte[] mValue = Arrays.copyOfRange(value, start: 3, end: 19);
                @SuppressLint("GetInstance") Cipher eCipher = Cipher.getInstance("AES/ECB/NoPadding");
                SecretKeySpec newKey = new SecretKeySpec(mi_band_key, algorithm: "AES");
                eCipher.init(Cipher.ENCRYPT_MODE, newKey);
                byte[] bytes = eCipher.doFinal(mValue);
                byte[] rq = Arrays.copyOf(new byte[]{0x03, 0x00}, newLength: 2 + bytes.length);
                System.arraycopy(bytes, i: 0, rq, i1: 2, bytes.length);

                characteristic.setValue(rq);
                gatt.writeCharacteristic(characteristic);
            }
```

Figura 14: Código para autenticación

En la primera de ellas, se realiza la autenticación para los modelos Mi Band 2 y 3, empleando una constante para el cifrado AES denominada *mi_band_3_key*, que es en esencia un vector con los bytes correspondientes a la secuencia “0123456789@ABCDE”.

En cambio la segunda parte, que se emplea para los modelos Mi Band 4 y Mi Smart Band 5, requieren de una variable *mi_band_key*. Esto es, la clave de autenticación que el usuario debe introducir manualmente en un campo de texto, y que se recupera para este proceso.

En ambos casos, una vez cifrada la clave se confecciona la respuesta concatenando los bytes 0x03, 0x00 con la clave y enviándola a la característica respectiva.

5.3 Lectura de valores

El método “onCharacteristicChanged", mostrado en la figura 15, forma parte de una clase denominada *BluetoothGattCallback*, necesaria para implementar todos los métodos necesarios para gestionar las conexiones bluetooth.

Figura 15: Detección de cambios en una característica

En concreto, este método es capaz de detectar cuándo se recibe una notificación en una característica determinada y, en el caso de que esta haya sido en la referente al monitor de ritmo cardiaco, se invoca a una función llamada *HRCallback* con el valor recibido donde se va a decidir cómo actuar en consecuencia al valor recibido.

En la figura 16 se muestra la primera parte del algoritmo para detección de fatiga, donde al iniciar el método, el primer paso es crear un timestamp para la medida recibida, que se empleará más adelante. A continuación se comprueba que el valor recibido sea superior a 35. Esto es debido a que ningún ser humano podría tener 35 o menos pulsaciones por minuto, y sería un indicativo de un error en el sensor. Seguidamente se comprueba si el valor debe ser ignorado gracias a un contador inicializado a cero que se compara con una variable global, que indica el número de valores que deben ser descartados.

```
private void HRCallback (byte currentHrValue) {
    Calendar measureTime = Calendar.getInstance();
    if (currentHrValue > 35) { //Check if sensor is working properly
        final String heartRateValue = String.valueOf(currentHrValue);
        Log.d( tag: "HR",  msg: "Received: " + heartRateValue);

        // Ignore first measurements in order to stabilize sensor
        if (ignored < maxIgnored) {
            ignored++;
            mHandler.post(() -> mHrValue.setText(heartRateValue));
            Log.d( tag: "CALLBACK",  msg: "Ignored");
        } else if (firstPeriod) {
            if ((measureTime.getTimeInMillis() - t) < (delay * ONE_MINUTE_IN_MILLIS)) {
                heartRateValues.add((int) currentHrValue);
                saveData(heartRateValue, measureTime);
                mHandler.post(() -> mHrValue.setText(heartRateValue));
            } else {
                referenceMean = meanOfGroup();
                double std = standardDeviation(referenceMean);
                if (std < 10) {
                    firstPeriod = false;
                    delay = 10;
                    heartRateValues.clear();
                    referenceDate = Calendar.getInstance();
                    t = referenceDate.getTimeInMillis();
                } else {
                    delay += 5;
                }
            }
```

Figura 16: (Primera parte) Gestión de valores de frecuencia cardiaca

Una vez se ha finalizado el descarte de medidas, se comprueba en qué periodo se encuentra el programa, que inicialmente debe ser el primero. En primer lugar se verifica si se ha excedido o no el tiempo límite establecido para ese periodo. En caso negativo, se almacenan las medidas en una estructura de datos. En otro caso, se calculan la media y la varianza con métodos externos y se comprueba que esta última sea inferior a lo establecido, y entonces se pasará al siguiente periodo, con un tiempo límite diferente.

En la segunda parte del algoritmo, mostrada en la figura 17 se puede ver como en los periodos posteriores se realiza un procedimiento similar, almacenando las medidas mientras no se sobrepase el tiempo estipulado. A continuación se calculan la media y la desviación estándar de la muestra como se hizo anteriormente, así como la diferencia entre la media del primer periodo y la del actual y la tendencia. Se establece una condición donde si la tendencia obtenida es negativa, así como el valor que indica la diferencia entre las medias es menor a -6, valor justificado previamente, se invoca a un método que activa las notificaciones en la pulsera programáticamente, y además se crea una estructura de datos que genera una notificación en el teléfono móvil.

```
} else if (!firstPeriod) {
    if ((measureTime.getTimeInMillis() - t) < (delay * ONE_MINUTE_IN_MILLIS)) {
        heartRateValues.add((int) currentHrValue);
        saveData(heartRateValue, measureTime);
        mHandler.post(() -> mHrValue.setText(heartRateValue));
    } else {
        periodMean = meanOfGroup();
        Log.d( tag: "CALLBACK",  msg: "New Period");
        double std = standardDeviation(periodMean);
        if (std < 10) {
            double diffPercentage = meanDifference(referenceMean, periodMean);
            Pair<Double, Double> trend = calculateTrend();
            Log.d( tag: "CALLBACK",  msg: "Calculated trend: " + trend);
            if ((trend.first < 0) && (diffPercentage < -6)) {
                startVibrate();
                ToneGenerator toneGenerator = new ToneGenerator(AudioManager.STREAM_MUSIC,  volume: 200);
                toneGenerator.startTone(ToneGenerator.TONE_CDMA_EMERGENCY_RINGBACK,  i1: 3000);
            }
            heartRateValues.clear();
            referenceDate = Calendar.getInstance();
            t = referenceDate.getTimeInMillis();
            delay = 10;
        } else {
            delay += 5;
        }
    }
}
```

Figura 17: (Segunda parte) Gestión de valores de frecuencia cardiaca

5.4 Almacenamiento de datos

El método mostrado en la figura 18 está pensado para ser llamado cada vez que sea necesario insertar valores en la base de datos. Se recibe el valor y una instancia de calendario, que indica la fecha y hora a la que fue recibida la medida. De esta manera se crea una estructura que contenga el valor recibido junto a los distintos campos de la fecha, día, mes, año, hora, minuto y segundo, así como la dirección MAC del dispositivo desde el que se recibió.

```
private void saveData(String value, Calendar cal) {
    ContentValues cv = new ContentValues();
    cv.put(HeartRateContract.HeartRateEntry.COLUMN_VALUE, value);
    cv.put(HeartRateContract.HeartRateEntry.COLUMN_DAY, cal.get(Calendar.DAY_OF_MONTH));
    cv.put(HeartRateContract.HeartRateEntry.COLUMN_MONTH, cal.get(Calendar.MONTH) + 1);
    cv.put(HeartRateContract.HeartRateEntry.COLUMN_YEAR, cal.get(Calendar.YEAR));
    cv.put(HeartRateContract.HeartRateEntry.COLUMN_HOUR, cal.get(Calendar.HOUR_OF_DAY));
    cv.put(HeartRateContract.HeartRateEntry.COLUMN_MINUTE, cal.get(Calendar.MINUTE));
    cv.put(HeartRateContract.HeartRateEntry.COLUMN_SECOND, cal.get(Calendar.SECOND));
    cv.put(HeartRateContract.HeartRateEntry.COLUMN_ADDRESS, address);

    db.insert(HeartRateContract.HeartRateEntry.TABLE_NAME,  nullColumnHack: null, cv);
}
```

Figura 18: Almacenamiento de valores en SQLite

Por último, se inserta esta estructura en la parte correspondiente de la base de datos creada.

6. Presupuesto

En este capítulo se presenta el presupuesto de este Trabajo de Fin de Grado. El presupuesto se ha dividido en dos apartados: costes asociados a materiales y costes relativos al desarrollo (recursos humanos).

6.1 Costes tecnológicos

La tabla 5 presenta los costes que corresponden a los materiales y/o equipamiento necesario para el desarrollo de este proyecto.

Tipos	Descripción	Producto	Cantidad	Coste unitario
Pulseras de actividad	Compra de varios modelos de pulseras para las distintas fases del desarrollo	Mi Smart Band 6	1	43.70€
		Mi Smart Band 5	3	24.99€
		Mi Band 3	1	--
Total:				118,67€

Tabla 5: Costes tecnológicos

6.2 Costes de desarrollo

La tabla 6 presenta los costes correspondientes al desarrollo del proyecto y recursos humanos necesarios para el mismo, así como el tiempo empleado en cada uno de los periodos que este comprende.

Tipo de actividad	Cantidad	Coste unitario	Coste total
Estudio preliminar de dispositivos	30 horas	12€/h	360€
Análisis de proyectos similares	24 horas	12€/h	288€
Análisis de la tecnología	22 horas	12€/h	264€
Desarrollo de	80 horas	12€/h	960€

métodos básicos para el funcionamiento de la app			
Desarrollo de métodos para la toma de datos	140 horas	12€/h	1680€
Toma de medidas y estudio de las mismas	30 horas	5€/h	150€
Documentación del proyecto	50 horas	9€/h	450€
Total:	376 horas		4152€

Tabla 6: Costes de desarrollo

7. Conclusiones y líneas futuras

A pesar de los múltiples contratiempos encontrados durante el progreso del proyecto, se ha conseguido una muy buena aproximación a una aplicación para la detección de somnolencia mediante pulseras de actividad, basada principalmente en la monitorización del ritmo cardiaco, tal y como se planteó en el alcance inicial. Además se ha incluido de manera experimental la detección de movimientos. Una vez obtenidos estos conocimientos fundamentales para trabajar haciendo uso de las características de la pulsera, es probable que la inclusión de nuevas características se vea considerablemente simplificada.

En conclusión, este trabajo ha resultado ser una experiencia enriquecedora, a la par que un gran reto debido a que se cuenta con una limitada cantidad de información sobre la tecnología Bluetooth y su variante de baja energía. Por ello, existen muy pocos proyectos realmente orientados al tipo de dispositivos que se han empleado para este desarrollo, y muchos de ellos se limitan a ampliar el trabajo realizado en otros. Por tanto, ha sido una tarea complicada recopilar información relevante para este trabajo.

De la misma manera, a lo largo de la implementación surgieron una serie de inconvenientes respecto a los modelos de dispositivos empleados, primeramente en cuanto a la autenticación. Como se comentó, la autenticación previa resultó ser un proceso fundamental para el correcto funcionamiento de las funciones posteriores. Esto supuso un grave retraso en el cronograma del proyecto. Sin embargo, pudo ser solventado recurriendo a diferentes modelos de monitores de actividad.

Otro de los contratiempos encontrados fue la dificultad debida a que algunos de los métodos cruciales para la aplicación, como la forma de autenticación o la toma de datos de movimiento, entre otros, difieren significativamente entre los distintos modelos de dispositivo, lo cual derivó en otro retraso considerable en el avance del trabajo. Asimismo, el hecho de que algunas de las variables fisiológicas que es posible medir con la aplicación del fabricante estén ocultas por este para aplicaciones de terceros supuso una gran limitación, ya que estas habrían aportado valiosa información que podría haber sido usada en el algoritmo implementado.

Por otra parte, la labor de desarrollar un proyecto usando un lenguaje y herramientas desconocidas supuso una excelente manera de adquirir nuevos conocimientos y afianzar los adquiridos durante el grado.

Como futuros avances en el proyecto, podrían mencionarse los siguientes:

- Solventar la autenticación mediante el empleo de curvas elípticas para el modelo Mi Smart Band 6.
- Ampliar el rango de dispositivos admitidos por la aplicación para incluir otros, como smartwatches, así como modelos pertenecientes a otros fabricantes.
- Obtener modelos de dispositivos cuyas características sean de libre acceso y así extender el algoritmo a la par que su fiabilidad.

8. Conclusions and future works

Despite the many setbacks encountered during the progress of the project, a very good approximation of an application for sleepiness detection using activity wristbands has been achieved, based mainly on heart rate monitoring, as proposed in the initial scope, as well as having experimentally included movement detection. Once this fundamental knowledge for working with the features of the wristband has been obtained, it is likely that the inclusion of new features will be considerably simplified.

In conclusion, this work has proved to be both an enriching and challenging experience. This is because Bluetooth technology, as well as its low energy variant, has a limited amount of information. Therefore, there are very few projects really oriented to the type of devices that have been used for this development and many of them are limited to extend the work done in others, and consequently it has been a complicated task to extract relevant information for this work.

In the same way, throughout the implementation, a series of inconveniences arose with respect to the device models used, firstly in terms of authentication. As mentioned, prior authentication turned out to be a fundamental process for the correct operation of the subsequent functions. This was a serious delay in the project schedule. However, it could be solved as soon as possible by resorting to other models of activity monitors.

Another setback encountered was the finding that some of the methods crucial to the application, such as the form of authentication, motion data collection, etc. differed significantly between device models, which resulted in another considerable delay in the progress of the work. Also, the fact that some of the physiological variables that can be measured with the manufacturer's application are hidden by the manufacturer from third-party applications was a limitation, as these would have provided valuable information that could have been used in the implemented algorithm.

On the other hand, the work of developing a project using a language and tools unknown to the programmer is an excellent way to learn and consolidate the knowledge acquired during the degree.

The following could be mentioned as future advances in the project:

- Solving the authentication using elliptic curves for the Mi Smart Band 6 model.
- Expand the range of devices supported by the application to include others such as smartwatches, as well as models belonging to other manufacturers.
- Obtain device models whose features are freely available, and thus extend the algorithm and its reliability.

9. Bibliografía

[1] “World Sleep Society”. worldsleepsociety.org. https://worldsleepsociety.org/

[2] MedlinePlus en español [Online]. Bethesda (MD): Biblioteca Nacional de Medicina (EE. UU.). “Fatiga”. Available: Fatiga: MedlinePlus enciclopedia médica

[3] R. M., Edmundo y R. C. M., Jorge. “Somnolencia: Qué es, qué la causa y cómo se mide. Acta méd. peruana” [Online]. 2010, vol.27, n.2, pp.137-143. Available: Somnolencia: Qué es, qué la causa y cómo se mide

[4] "¿Qué son los microsueños y cómo controlarlos?" Mejor con Salud. https://mejorconsalud.as.com/microsuenos-controlarlos/

[5] Wikipedia Collaborators. “Bluetooth Low Energy” [Online]. Wikipedia, the free encyclopedia, 2021. Available:

https://es.wikipedia.org/w/index.php?title=Bluetooth_de_baja_energ%C3%ADa&oldid=140049462

[6] "Generic Attribute Profile (GATT) Overview - Developer Help". Home - Developer Help. Generic Attribute Profile (GATT) Overview - Developer Help

[7] Bluetooth SIG 16-bit UUID Numbers. 16-bit UUID Numbers Document

[8] Diffie, W. y M. E. Hellman. “New directions in cryptography”, IEEE Transactions on Information Theory 22 (1976), pp. 644-654.

[9] Freemyband: https://www.freemyband.com/

[10] A. C. Angulo. (2021, Oct 29). “Frecuencia cardiaca: ¿Cuál es el nivel de pulsaciones recomendado?” [Online]. Available: Frecuencia cardiaca: ¿Cuál es el nivel de pulsaciones recomendado? • Farmacia Angulo

[11] “La frecuencia cardíaca del corazón en reposo y actividad”. Desfibrilador.com. La frecuencia cardíaca del corazón en reposo y actividad

[12] D. Espinosa. (2011, Oct 11). “Estimación de la tendencia de una serie temporal” [Online]. Available: Estimación de la tendencia de una serie temporal

[13] N. Zhang, M. Fard, M. H. U. Bhuiyan, D. Verhagen, M. F. Azari & S. R. Robinson (2018) “The effects of physical vibration on heart rate variability as a measure of drowsiness”, Ergonomics, vol 61, no. 9, pp. 1259-1272, July 17, 2018, doi: 10.1080/00140139.2018.1482373

[14] R. Martí. (2021, Aug. 20). “Índice de recuperación de la frecuencia cardiaca, ¿estás tan en forma como te crees?” [Online]. Available: Índice de recuperación de la frecuencia cardiaca, ¿estás tan en forma como te crees?

[15] Shaffer F, Ginsberg JP. “An Overview of Heart Rate Variability Metrics and Norms”. Front Public Health. 2017; vol 5, no. 258, Sep 28, 2017. doi:10.3389/fpubh.2017.00258

[16] S.-H. Jo, J.-M. Kim, y D. K. Kim, “Heart rate change while drowsy driving”, J. Korean Med. Sci., vol. 34, n.o 8, p. e56, 2019.

[17] B. Lee, B. Lee and W. Chung, "Standalone Wearable Driver Drowsiness Detection System in a Smartwatch," in IEEE Sensors Journal, vol. 16, no. 13, pp. 5444-5451, July 1, 2016, doi: 10.1109/JSEN.2016.2566667.

[18] Github. https://github.com/

[19] Mockflow. https://wireframepro.mockflow.com/

[20] Figma. https://www.figma.com/

Apéndice 1. Enlace al repositorio de código

Repositorio en GitHub [18]: https://github.com/alu0101205953/StayAwake.git

Apéndice 2. Enlace a Wireframes de la aplicación

Wireframes en Mockflow [19]: https://wireframepro.mockflow.com/view/MYqWs10gBnb

Apéndice 3. Enlace a Mockups de la aplicación

Mockups en Figma [20]:
https://www.figma.com/file/MOBBuTj7BET1i3r6C0J3Fz/StayAwakePrototype?node-id=0%3A1