UNIVERSIDAD DE VALLADOLID

ESCUELA TÉCNICA SUPERIOR DE INGENIEROS DE TELECOMUNICACIÓN

TRABAJO FIN DE GRADO

GRADO EN TECNOLOGÍAS ESPECÍFICAS DE TELECOMUNICACIÓN,
MENCIÓN EN SISTEMAS DE TELECOMUNICACIÓN

Sistemas de ayuda a la rehabilitación de pacientes basado en aplicación móvil

Autor:
Dña. Ana Santos Delgado

Tutor:
D. Alonso Alonso Alonso

TÍTULO:	**Sistema de ayuda a la rehabilitación de pacientes basado en aplicación móvil**
AUTOR:	**Dña. Ana Santos Delgado**
TUTOR:	**D. Alonso Alonso Alonso**
DEPARTAMENTO:	**Teoría de la Señal y Comunicaciones e Ingeniería Telemática**

Tribunal

PRESIDENTE:	**D. Alonso Alonso Alonso**
VOCAL:	**D. Javier Aguiar Pérez**
SECRETARIO:	**D. Ramón de la Rosa Steinz**
SUPLENTE 1:	**D. Jaime Gómez Gil**
SUPLENTE 2:	**D. Alberto Izquierdo Fuente**

FECHA:	**27 Septiembre 2023**
CALIFICACIÓN:	

Agradecimientos:

En primer lugar, a mi familia por confiar en mí incluso cuando ni yo misma lo hacía, por apoyarme y animarme continuamente durante todo este camino. Han supuesto mi mayor fortaleza a lo largo de este camino.

A mis amigos de Villalón y Villavelasco, que son sinónimo de hogar y familia, por animarme continuamente y por estar siempre a mi lado cuando más los he necesitado.

A los amigos que me ha regalado esta carrera que me los llevo para siempre y en especial a Carmen María y Karina por animarme en mis momentos de estrés y lágrimas.

Y por último y no por ello menos importante a mi tutor Alonso por guiarme y confiar en mí para la realización de este proyecto, gracias a su dedicación y continua disponibilidad.

Resumen

En este Trabajo de Fin de Grado se llevará a cabo un estudio de los diferentes tipos de aplicaciones móviles, junto con el entorno de desarrollo Flutter. Posteriormente se explorará la parte del hardware, centrándose en el análisis de dispositivos MEMS, y la evaluación de los sensores acelerómetro, giroscopio y magnetómetro. Además, se abordará el concepto de goniometría, una técnica de medición de ángulos, y se presentará el filtro complementario, una herramienta esencial en la fusión de datos del acelerómetro y giroscopio, con el fin de facilitar la comprensión de los aspectos técnicos del proyecto. Se desarrollará además un sistema basado en Arduino para comprobar de forma práctica el funcionamiento de un módulo acelerómetro y giroscopio, el MPU6050.

El objetivo de este proyecto será desarrollar una aplicación móvil diseñada para monitorizar por parte del profesional rehabilitador diferentes ejercicios de rehabilitación que involucran el codo y la muñeca, realizados fuera del entorno sanitario. A continuación, se expondrán las dos versiones que se plantearon durante el desarrollo de la aplicación. También se describirá el papel de la IA ChatGPT como herramienta de apoyo en la fase de desarrollo de la aplicación. Finalmente, se realizarán pruebas con personas para evaluar la eficacia de la aplicación y su utilidad en el contexto de ejercicios de rehabilitación. En las conclusiones se destacarán aspectos de mejora y la visión de futuro de este tipo de aplicaciones

Palabras clave

Aplicación, MHealth, Flutter, Dart, ángulo, sensores, MEMS, acelerómetro, giroscopio.

Abstract

In this final degree project, a study will be conducted on different types of mobile applications, along with the Flutter development environment. Subsequently, the hardware aspect will be explored, focusing on the analysis of MEMS devices and the evaluation of accelerometer, gyroscope, and magnetometer sensors. Furthermore, the concept of goniometry, a technique for measuring angles, will be addressed, and the complementary filter, an essential tool in fusing accelerometer and gyroscope data, will be introduced to facilitate the understanding of the technical aspects of the project. An Arduino-based system will also be developed to practically test the operation of an accelerometer and gyroscope module, the MPU6050.

The objective of this project will be to develop a mobile application designed to monitor by the rehabilitation professional different rehabilitation exercises that involve the elbow and wrist, performed outside the healthcare environment. Next, the two versions considered during the application development process will be presented. The role of Al ChatGPT as a supporting tool in the application development phase will also be described. Finally, tests will be conducted with individuals to assess the effectiveness of the application and its utility in the context of rehabilitation exercises. The conclusions will highlight areas for improvement and the future vision of this type of application.

Keywords

Application, mHealth, Flutter, Dart, angle, sensors, MEMS, accelerometer, gyroscope.

ÍNDICE

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

1. INTRODUCCIÓN

En esta sección se expondrán las razones que motivaron la elección de este tema como trabajo de fin de grado, continuando con una enumeración de los objetivos que se tratarán de alcanzar a lo largo del proyecto, así como las fases y métodos a realizar para conseguir los objetivos y un contexto social que servirá para mostrar la situación actual de la rehabilitación y los pacientes.

Por último, se ha llevado a cabo una minuciosa investigación en páginas como Google schollar, IEEE Xplore, Pub Med, etc. Para la búsqueda de artículos de investigación relacionados con el tema de este proyecto. Los aspectos que se han tenido en cuenta para la selección de estos artículos han sido, la fecha de realización del estudio se ha pretendido seleccionar los artículos más recientes, también se han tenido en cuenta palabras claves como ehealth, mhealth, sensores, rehabilitación, fisioterapia, etc. Se han eliminado algunos de estos artículos debido a escasez de resultados y a que el acceso al artículo completo no era gratuito.

1.1 MOTIVACIÓN

Una de las causas que ha motivado la elección de este proyecto se basa en la necesidad de mejorar la eficacia y la accesibilidad de los servicios de rehabilitación. La rehabilitación es un proceso clave en la recuperación de lesiones o enfermedades, y su objetivo es ayudar a las personas a volver a su nivel de funcionamiento previo y mejorar su calidad de vida. Sin embargo, el proceso de rehabilitación a menudo puede ser tedioso y requiere un seguimiento constante y una supervisión por parte de un profesional de la salud.

La tecnología móvil ha proporcionado una oportunidad para mejorar la eficacia y la accesibilidad a los servicios de rehabilitación. Las aplicaciones móviles de ayuda a la rehabilitación permiten a los pacientes llevar a cabo ejercicios personalizados y programas basados en la evidencia científica, lo que ayuda a mejorar la recuperación de manera más eficiente. Además, estas aplicaciones suelen incluir un sistema de seguimiento y comunicación con el profesional de la salud, lo que permite una mejor colaboración y una mayor eficacia en el proceso de recuperación. Puesto que puede ser utilizada como una herramienta de evaluación y monitoreo a través de los datos obtenidos de la aplicación, evaluando así el progreso del paciente y si es necesario hacer ajustes en su plan de rehabilitación.

Muchas personas, especialmente aquellas con discapacidades o personas mayores, tienen dificultades para desplazarse a un centro de rehabilitación. Las aplicaciones móviles permiten a estas personas realizar los ejercicios en casa, lo que mejora la accesibilidad y la comodidad.

Este tipo de aplicaciones pueden tener un impacto económico positivo. Ya que ayudan a reducir el costo de la atención médica, puesto que permiten a los pacientes realizar ejercicios en casa en lugar de tener que acudir a un centro.

En conclusión, La tecnología móvil proporciona una oportunidad para mejorar la recuperación de los pacientes mediante el uso de ejercicios personalizados y el seguimiento de su progreso, además de permitir una comunicación eficaz entre el paciente y el profesional de la salud, pudiendo así proporcionar una atención bastante similar a la que se ofrece en una consulta presencial. También

contribuiría a mejorar la accesibilidad de los servicios de rehabilitación para personas mayores o con discapacidad, y tendría un impacto económico positivo al reducir costos y tiempo de inactividad laboral.

1.2 OBJETIVOS DEL PROYECTO

El objetivo principal de este proyecto es desarrollar una aplicación móvil de ayuda a la rehabilitación que haga uso de los sensores del propio dispositivo móvil. Para llevar a cabo este objetivo principal se fijaron una serie de objetivos que se enumeran a continuación.

1. Realizar un estudio de las técnicas actuales de rehabilitación, investigando sobre los diferentes enfoques terapéuticos y técnicas, para así conocer las necesidades y expectativas de los pacientes y médicos o fisioterapeutas.
2. Investigar a cerca de los diferentes dispositivos hardware que se pueden emplear para registrar el movimiento del ejercicio como pueden ser los sensores de movimiento incluidos en los smartphones o wearables.
3. Consultar bibliografía sobre el tema para obtener una comprensión de las necesidades y desafíos en el campo de la rehabilitación.
4. Familiarización con el entorno de desarrollo flutter, donde se va a realizar la aplicación móvil.
5. Llevar a cabo un análisis de las diferentes aplicaciones de rehabilitación o fitness, que hay actualmente disponibles en la App Store y la Play Store.
6. Realizar un sistema basado en Arduino y PC para comprobar el funcionamiento de un sensor de acelerómetro + giroscopio comercial.
7. Realizar una interfaz de usuario dinámica, intuitiva y fácil de usar, que se adapte a las necesidades de cada usuario que pueden ser niños, adolescentes, adultos, ancianos o personas con algún tipo de discapacidad.
8. Llevar a cabo un proceso de prueba de la aplicación para asegurar un correcto funcionamiento y corregir posibles fallos. Dentro de este punto se deben cumplir los siguientes aspectos:
    a. Recoger los movimientos del ejercicio gracias a los sensores del dispositivo móvil.
    b. Procesar la información recogida por los sensores.
    c. Mostrar en tiempo real los resultados del movimiento y asesorar al paciente en remoto a la hora de realizar el ejercicio.
    d. Registrar todos los ejercicios realizados para que el médico o fisioterapeuta tenga acceso a ellos y pueda ver la evolución del paciente para realizar un seguimiento personalizado.
9. Realizar pruebas, con la versión final de la aplicación en sujetos de control y un paciente en rehabilitación tras operación para suprimir el síndrome del túnel carpiano.
10. Documentar y recopilar todo el proceso de desarrollo de la aplicación en una memoria.

1.3 FASES Y MÉTODOS

A continuación, se exponen las fases y métodos a seguir para conseguir los objetivos mostrados en la sección anterior.

1. Fase de investigación:
   - Revisión bibliográfica: llevar a cabo una revisión de artículos científicos relacionados con sistemas de ayuda a la rehabilitación de pacientes. Identificando los avances y tendencias en este ámbito, así como las metodologías empleadas en dichas investigaciones y los resultados obtenidos. Esta fase es importante para conocer el estado del arte y establecer las bases teóricas del proyecto.
   - Identificación de necesidades: identificar las necesidades y problemas actuales en la rehabilitación y como podría ayudar una aplicación que monitorice el movimiento del paciente para satisfacer los requerimientos de pacientes y profesionales en el ámbito de la rehabilitación.

2. Fase de diseño:
   - Diseño de la interfaz y funcionalidades: En este punto se debe diseñar la estructura y funcionalidades de la aplicación, teniendo en cuenta las necesidades y requerimientos identificados en la fase anterior, así como los avances y tendencias en el ámbito de la investigación. Se deben determinar las funcionalidades necesarias, como, por ejemplo, ejercicios personalizados, seguimiento de progreso, recordatorios de realización del ejercicio y/o medicación…
   - Creación de prototipos: En esta fase se desarrollarán prototipos de la aplicación para evaluar su usabilidad y accesibilidad.

3. Fase de desarrollo:
   - Desarrollo de la aplicación: En esta fase se deberá desarrollar la aplicación móvil según el diseño previo. Implementando las funcionalidades necesarias y con una interfaz de usuario dinámica y fácil de usar.
   - Pruebas y validación: Realizar pruebas de funcionamiento con voluntarios.
   - También se desarrollará un sistema basado en Arduino y PC empleando un sensor de acelerómetro + giroscopio comercial (MPU6050) para comprobar el funcionamiento de estos sensores y ensayar algoritmo.

4. Fase de evaluación:
   - Se realizarán pruebas de la versión final de la aplicación con sujetos de control y un paciente con algún tipo de dolencia.
   - Análisis de los resultados: se valorarán todos los resultados obtenidos tras las pruebas para mejorar la aplicación si es necesario.

5. Fase de conclusión:
   - Se recogerá en una memoria todo el proceso de desarrollo detallado, incluyendo la descripción de todas las características y funcionalidades de la aplicación, así como las fases y métodos llevados a cabo durante el proceso. Finalmente se recogerán los resultados obtenidos, las conclusiones y las posibles líneas futuras de investigación.

1.4 CONTEXTO SOCIAL

La fisioterapia y la rehabilitación son fundamentales para la recuperación y el bienestar de las personas que sufren lesiones o enfermedades, ya que ayudan a mejorar la movilidad, la fuerza, el equilibrio, la coordinación y la función general del paciente. Sin embargo, existen varias barreras que dificultan el acceso a estos servicios, como la falta de recursos, la falta de profesionales cualificados, la falta de cobertura médica y la problemática de aislamiento en el entorno rural.

La tecnología móvil ha revolucionado la forma en que vivimos y nos relacionamos con el mundo que nos rodea. En los últimos años, el aumento del uso de dispositivos móviles y wearables ha tenido un impacto significativo en la actividad física. Debido a que proporcionan acceso a aplicaciones de rehabilitación y ejercicios en línea. Este tipo de aplicaciones se encuentran dentro del campo de las aplicaciones de salud también conocidas como mHealth. Algunas de estas aplicaciones ofrecen ejercicios de terapia física y rehabilitación guiados, mientras que otras proporcionan seguimiento y monitoreo de la actividad física, el progreso de la recuperación y los recordatorios de medicación [1][2].

El desarrollo de una aplicación móvil de ayuda a la rehabilitación tiene un gran potencial para mejorar la atención médica y la calidad de vida de las personas que sufren lesiones o enfermedades, ya que permite una mayor flexibilidad y accesibilidad en el proceso de rehabilitación. Además, al ser una herramienta tecnológica, puede ofrecer una mayor personalización y adaptabilidad a las necesidades individuales de cada paciente.[3]

Sin embargo, es importante tener en cuenta que el desarrollo de este tipo de aplicaciones debe ser realizado por profesionales cualificados y con experiencia en el campo de la fisioterapia y la rehabilitación, para garantizar su eficacia y seguridad [4]. Asimismo, se deben tener en cuenta las barreras tecnológicas y económicas para garantizar el acceso a la aplicación de todos los pacientes que la necesiten.

En resumen, la sociedad cada vez es más consciente de la importancia de la fisioterapia y la rehabilitación, pero con barreras para el acceso a estos servicios. La tecnología móvil es una herramienta valiosa para mejorar la atención médica y la calidad de vida de las personas que sufren lesiones o enfermedades, siempre y cuando se desarrolla de manera adecuada. Por lo tanto, el desarrollo de una aplicación móvil de ayuda a la rehabilitación es una oportunidad para contribuir a la mejora de la salud y el bienestar de las personas.

1.5 ESTADO DEL ARTE

Las lesiones del sistema musculoesquelético involucran daños en uno o más componentes del sistema locomotor y sus tejidos asociados. Estas lesiones representan la mayor proporción de condiciones de dolor persistente no relacionadas con el cáncer. Según la Organización Mundial de la Salud (OMS), aproximadamente entre el 20% y el 33% de la población mundial vive con una afección musculoesquelética dolorosa, convirtiéndose en la principal causa de discapacidad a nivel global, siendo el dolor lumbar la principal causa de discapacidad a nivel mundial. Este tipo de lesiones representan aproximadamente el 25% de los gastos en salud a nivel global, lo que implica una significativa carga para los recursos destinados a la atención médica [4].

La fisioterapia musculoesquelética despliega una amplia variedad de enfoques terapéuticos, incluyendo ejercicios de fortalecimiento y flexibilidad, consejos sobre postura y ergonomía, terapia manual, y educación para la autogestión del dolor. Esta disciplina resulta fundamental para aliviar el dolor y la discapacidad a corto plazo, agilizando la recuperación y, por ende, aliviando la presión sobre el sistema de atención médica [1].

Cuando este tipo de afecciones requiere cirugía, se busca reducir al máximo la estancia hospitalaria del paciente. Dado que el tiempo en el hospital es más breve, el papel del autocontrol por parte de los pacientes se ha vuelto crucial para mejorar los resultados. Según la Organización Mundial de la Salud (OMS), el autocontrol postoperatorio se refiere a la capacidad de las personas y sus familias para hacer frente a la enfermedad, ya sea con o sin el respaldo del personal médico. El personal médico brinda al paciente una educación para que pueda adaptarse adecuadamente a su nueva situación después de la cirugía en su propio hogar [5].

Sin embargo, para muchos pacientes, llevar a cabo los ejercicios de rehabilitación en casa puede resultar desafiante. Sin la supervisión de un terapeuta, existe el riesgo de ejecutar los ejercicios de manera incorrecta, lo que incluye una velocidad de movimiento inadecuada y una calidad de movimiento deficiente, lo que podría tener un impacto negativo en la eficacia del ejercicio. Además, el grado de compromiso del paciente con el programa de ejercicios también influye significativamente en el éxito del proceso de rehabilitación [6].

La rehabilitación remota, utilizando tecnologías de salud móvil (mHealth), ha surgido como una herramienta con un inmenso potencial para mejorar la adherencia y facilitar la ejecución efectiva de ejercicios en el entorno doméstico [7]. La Organización Mundial de la Salud (OMS) define la salud móvil como "una práctica médica y de salud pública respaldada por dispositivos móviles, dispositivos de monitoreo de pacientes, asistentes digitales personales y otros dispositivos inalámbricos" [4]. El crecimiento exponencial del número de dispositivos móviles, con más de 3.800 millones de personas poseyendo smartphones en la actualidad [4], y los avances tecnológicos, incluyendo sensores portátiles, han impulsado significativamente el desarrollo de mHealth [8]. Este progreso ha dado lugar a un aumento notable en la cantidad de aplicaciones de salud disponibles en los últimos años [4]. Gracias a los sensores portátiles, se logra un seguimiento continuo de la actividad del paciente, lo que ha llevado a una notable mejora en la adherencia a los programas de rehabilitación en el hogar [8].

Sin embargo, la incorporación de aplicaciones de mHealth en la práctica clínica no ha seguido el mismo ritmo que el aumento en su disponibilidad en las principales tiendas de aplicaciones. Esto se debe a la falta de conocimiento o confianza por parte de los profesionales médicos en relación con la tecnología, así como a la posible carencia de contenido médico preciso y de calidad en estas aplicaciones. Además, existe el riesgo de que la privacidad sea insuficiente y de que las aplicaciones no ofrezcan beneficios terapéuticos suficientes. A pesar de que las aplicaciones de mHealth deben cumplir con ciertos criterios específicos para ser distribuidas a través de las tiendas de aplicaciones, estos criterios suelen centrarse en aspectos tecnológicos y no requieren una supervisión médica adecuada.[9]

Se ha realizado una investigación de aplicaciones que emplean sensores para monitorizar o realizar un seguimiento de la actividad de los pacientes. Se mencionan algunas:

- Evaluación de un sistema portátil de biorretroalimentación de ejercicios basado en sensores: Este sistema incluye una IMU ubicada en el muslo izquierdo del usuario, que monitorea su

movimiento durante la realización de una serie de ejercicios enfocados en las piernas. La aplicación móvil se encarga de procesar las señales provenientes de la IMU, registrando las repeticiones y empleando métodos de clasificación personalizados para evaluar la corrección de cada repetición. Durante el ejercicio, el usuario recibe información en tiempo real acerca de su progreso hacia la finalización de las repeticiones. Una vez que se completa el ejercicio, el usuario puede consultar sus resultados. Cabe mencionar que este sistema experimentó algunos inconvenientes en la funcionalidad de la aplicación, aunque los resultados generales reflejaron un alto grado de usabilidad. Los usuarios destacaron especialmente la facilidad de uso y configuración del sistema [10].

- Estudio del impacto de una aplicación en el cuidado postoperatorio diario de pacientes sometidos a reemplazo de rodilla: En esta investigación, se analizó el efecto de una aplicación diseñada para la instrucción activa de los pacientes durante las primeras cuatro semanas de su proceso de recuperación después de la cirugía de reemplazo de rodilla. Los resultados obtenidos revelaron una reducción del dolor en reposo, durante la actividad física y durante la noche en los usuarios de la aplicación, además de indicar una tendencia hacia una disminución en la necesidad de atención médica [5].
- Estudio de la Aplicación HospitalFit: Es una aplicación diseñada específicamente para pacientes hospitalizados. Esta aplicación móvil se integra con un acelerómetro que cumple la función de distinguir entre las posturas de acostado, sentado y caminando. Proporciona información valiosa tanto para los pacientes como para los fisioterapeutas. Para los pacientes, ofrece datos sobre su progreso, además de un programa de ejercicios completo respaldado por videos instructivos. Los resultados del estudio sugieren que esta aplicación contribuye de manera efectiva a mejorar los niveles de actividad de los pacientes, aunque es importante destacar que la población participante en la investigación fue relativamente reducida [3].
- Estudio de Viabilidad y Experiencia del Paciente en un Programa de Rehabilitación en el Hogar: En este estudio, se exploró la viabilidad y la experiencia de los pacientes participantes en un programa de rehabilitación en el entorno domiciliario. Para llevar a cabo este programa, se utilizó una Tablet que permitía a los pacientes acceder a videos de ejercicios y un sensor de movimiento que se llevaba como un collar para registrar la actividad relacionada con el movimiento. Los pacientes se ponían en contacto telefónicamente con sus terapeutas para recibir orientación y entrenamiento. El dispositivo del cuello incluía un acelerómetro y un sensor de presión barométrica. Los resultados obtenidos indicaron una alta adherencia por parte de los pacientes al programa de ejercicios y una experiencia positiva en relación con la tecnología empleada. No obstante, se reconoce la necesidad de llevar a cabo investigaciones adicionales para evaluar con mayor profundidad la efectividad de este enfoque [8].

- Estudio de Evaluación del Ejercicio de Rehabilitación mediante el Uso de Sensores Inerciales: El propósito de esta investigación fue explorar la capacidad de utilizar datos recopilados por sensores inerciales y técnicas de aprendizaje automático para evaluar el rendimiento de ejercicios de extremidades inferiores en rehabilitación. Como resultado, se logró demostrar la posibilidad de clasificar la ejecución de los ejercicios como correcta o incorrecta, así como identificar errores específicos cometidos durante cada ejercicio. Además, se observó que la reducción en el número de sensores no afectó la precisión de los resultados. No obstante, se concluyó que se requiere una mayor investigación para profundizar en este campo [6].

- Evaluación de la Precisión del Sistema de Medición del Ángulo de la Rodilla en Tres Ejercicios Comunes de Fisioterapia: En este estudio, se implementaron dos Unidades de Medición Inercial (IMU) inalámbricas que se conectaron a través de una aplicación móvil interactiva. La retroalimentación en tiempo real permitió a los usuarios monitorear y ajustar su postura durante la realización de los ejercicios. Se observó un incremento en el error de medición a medida que se ampliaba el rango de movimiento (RoM) entre las IMU. Los resultados indicaron que la retroalimentación visual mejoró la capacidad de los usuarios para ejecutar los ejercicios de manera adecuada. Es importante señalar que la población de estudio fue relativamente pequeña y estuvo compuesta por individuos jóvenes y saludables [7].

2. REVISIÓN DE LA TECNOLOGÍA

2.1 TIPO DE APLICACIONES MÓVILES

En los últimos años el desarrollo de aplicaciones móviles ha crecido al mismo nivel que lo ha hecho la evolución de la tecnología.

Las horas que se pasan frente al teléfono móvil no dejan de aumentar conforme pasan los años, según un estudio realizado por informe Mobile 2022, los usuarios llegan a dedicar hasta 1752 horas al año a las aplicaciones móviles, lo que ha llevado a una evolución en la forma de conectarse y relacionarse. La pandemia que sufrimos en 2020 debida al covid-19 contribuyó también al aumento del uso de dispositivos móviles, para realizar compras online, en busca de entretenimiento en redes sociales, juegos o para realizar deporte mediante aplicaciones fitness [11].

Por lo tanto, se puede decir que la navegación a través del teléfono móvil está ganando terreno a la navegación a través de ordenadores de escritorio.

Para los desarrolladores de aplicaciones móviles una de sus principales preguntas es que tecnología de desarrollo de aplicaciones es más adecuada. Actualmente existen tres tipos de aplicaciones móviles: aplicaciones web, aplicaciones nativas y aplicaciones híbridas, aunque la primera no entra estrictamente dentro de la tecnología de desarrollo de aplicaciones móviles.
En esta sección se abordarán las características de cada tipo y se decidirá cuál es la más conveniente para desarrollar la aplicación de ayuda a la rehabilitación.

2.1.1 APLICACIONES NATIVAS

Las aplicaciones nativas son aplicaciones desarrolladas específicamente para un sistema operativo y se ejecutan directamente en el hardware del dispositivo. A continuación, se desarrollan en detalle algunas de las características más importantes de estas aplicaciones:

- Adaptadas para funcionar en un sistema operativo específico y también a los dispositivos, lo que les permite aprovechar al máximo los recursos del hardware [16].
- Pueden aprovechar las características y funciones exclusivas de la plataforma móvil, lo que les permite ofrecer una experiencia de usuario más personalizada.
- Pueden acceder directamente a las características del hardware del dispositivo, como la cámara, el GPS y los sensores de movimiento, en la *Fig.1* aparece un esquema con todos los sensores disponibles en el móvil a los que pueden tener acceso este tipo de aplicaciones.
- Pueden implementar medidas de seguridad adicionales, como el cifrado de datos y la autenticación de usuarios.
- Pueden funcionar sin conexión a Internet, lo que las convierte en una opción popular para las aplicaciones que necesitan estar disponibles incluso cuando no hay una conexión a Internet disponible.
- Pueden integrarse con otros sistemas y tecnologías, lo que las convierte en una opción popular para las empresas que buscan desarrollar aplicaciones empresariales.

Estas aplicaciones trabajan con los lenguajes de programación propios de cada sistema operativo, en el caso de Android emplearan Java o Kotlin y en iOS emplearan Swift o Objective-C. Por lo tanto, si se quiere que una aplicación funcione en ambas plataformas iOS y Android, se necesitaran generar dos códigos diferentes, lo que supondrá un mayor tiempo de desarrollo.

La distribución de este tipo de aplicaciones se realiza a través de la App Store para iOS y de Google Play para Android. [13][14][15][16][17]

Fig. 1 Sensores disponibles en los smartphones[12]

2.1.1.1 VENTAJAS Y DESVENTAJAS

Como ventajas tiene:

- Mayor rendimiento y velocidad en comparación con otras tecnologías de desarrollo de aplicaciones.
- Mayor experiencia de usuario personalizada debido a la integración con las características exclusivas de cada plataforma móvil.
- Mayor seguridad debido a las medidas de seguridad implementadas en el sistema operativo móvil y en la propia aplicación.
- Mayor accesibilidad para los usuarios móviles, incluso cuando no hay una conexión a Internet disponible.
- Mayor capacidad de integración con otros sistemas empresariales.

Como desventajas presenta:

- Costo de desarrollo y mantenimiento más elevado que otras tecnologías de desarrollo de aplicaciones.
- Limitaciones en la portabilidad de la aplicación a diferentes plataformas móviles.
- Tiempos de desarrollo más largos debido a la necesidad de desarrollar aplicaciones separadas para cada plataforma móvil.

- Mayor complejidad en comparación con otras tecnologías de desarrollo de aplicaciones.

En resumen, las aplicaciones nativas ofrecen una serie de ventajas significativas, incluyendo mayor rendimiento, experiencia de usuario personalizada, mayor seguridad y accesibilidad sin conexión a Internet. Sin embargo, presentan unos costes de mantenimiento más elevados y tiempos de desarrollo más largos. [13][16][17]

2.1.2 APLICACIONES WEB

Las aplicaciones web son aplicaciones informáticas que se ejecutan en un servidor web y se acceden a través de un navegador web.

Las tecnologías más comunes utilizadas para el desarrollo de este tipo de aplicaciones son [19]:

1. Lenguajes de programación: los más empleados para el desarrollo de aplicaciones web son HTML, CSS, JavaScript y PHP. HTML se emplea para la estructura de la página, CSS para el estilo y la apariencia, JavaScript para la interactividad y PHP para la programación del lado del servidor.
2. Frameworks y bibliotecas: son herramientas software que se utilizan para facilitar el desarrollo de las aplicaciones web. Algunos de los más populares son AngularJS, React, Vue.js y Laravel.
3. Bases de datos: Las bases de datos se utilizan para almacenar información en las aplicaciones web. Las más comunes son MySQL, PostgreSQL y MongoDB.
4. Servidores web: son programas que se ejecutan en un servidor y que permiten que las aplicaciones web sean accesibles a través de internet. Algunos de los servidores web más comunes son HTTP, HTTPS y WebSockets.

Las aplicaciones web permiten el acceso desde cualquier dispositivo con conexión a internet y un navegador web, lo que aumenta su accesibilidad y portabilidad. Son fáciles de actualizar, ya que se actualizan automáticamente en el servidor, y no requieren instalación en el dispositivo del usuario.[17][18]

2.1.2.1 VENTAJAS Y DESVENTAJAS

En primer lugar, se enumeran las ventajas de este tipo de aplicaciones [14][17][18]:

- Disponibilidad: puede ser accedida desde cualquier dispositivo con conexión a internet.
- Actualización automática: los usuarios no tienen que preocuparse por actualizar la aplicación, ya que se actualiza automáticamente en el servidor.
- Ahorro de espacio: no ocupa espacio en el dispositivo del usuario, lo que significa que no hay necesidad de descargar e instalar actualizaciones o archivos grandes.
- Multiplataforma: puede funcionar en diferentes sistemas operativos y dispositivos.
- Fácil de desarrollar: la mayoría de las herramientas de desarrollo web son gratuitas y no requieren una gran cantidad de recursos.

Por último, se enumeran las desventajas [14][17][18]:

- Dependencia de internet: para poder usar la aplicación, es necesario tener una conexión a internet estable.
- Limitaciones de funcionamiento: las aplicaciones web no pueden realizar ciertas funciones que son posibles en las aplicaciones nativas, como el acceso a algunos sensores y características del dispositivo.
- Rendimiento limitado: las aplicaciones web pueden ser más lentas que las nativas, especialmente cuando se trata de tareas intensivas en CPU o de uso intensivo de gráficos. Esto se debe en parte al hecho de que las aplicaciones web se ejecutan en un navegador, lo que puede limitar su capacidad de procesamiento.
- Falta de integración con el hardware del dispositivo: Las aplicaciones web tienen limitaciones en cuanto a la integración con el hardware del dispositivo, lo que significa que no pueden aprovechar todas las funciones del dispositivo, como el acceso a los sensores del móvil.
- Limitaciones de almacenamiento: Las aplicaciones web no pueden acceder al almacenamiento local del dispositivo. Esto significa que no pueden almacenar grandes cantidades de datos en caché y que no pueden funcionar sin conexión.
- Seguridad: Este tipo de aplicaciones son más vulnerables a ataques de seguridad, ya que dependen de una conexión a internet y están disponibles públicamente en la red. Esto significa que las aplicaciones web son más susceptibles a los ataques de hackers y al robo de datos.
- El servidor donde se aloja la aplicación web: si el servidor presenta fallos o problemas técnicos, la disponibilidad de la web app puede verse afectada. Si el servidor no se configura o administra correctamente puede ser vulnerable a ataques de seguridad que ya se han mencionado antes.

2.1.3 APLICACIONES HIBRIDAS

Las aplicaciones hibridas son aquellas que se desarrollan utilizando tecnologías web, como HTML, CSS y JavaScript, y se ejecutan dentro de un contenedor nativo que permite su acceso a las funcionalidades del dispositivo. Las características de las aplicaciones híbridos se enumeran a continuación:

- Se desarrollan con tecnologías web, como HTML, CSS y JavaScript.
- Se ejecutan dentro de un contenedor nativo que permite el acceso a las funcionalidades del dispositivo.
- Pueden ser multiplataforma, lo que significa que se pueden desarrollar para diferentes sistemas operativos móviles.
- Pueden ser actualizadas de manera rápida y sencilla, ya que las actualizaciones se realizan en el servidor y se envían al dispositivo del usuario sin necesidad de actualizar la aplicación en sí misma.
- Pueden tener una interfaz de usuario similar a la de una aplicación nativa, aunque puede ser menos intuitiva.

En resumen, las aplicaciones híbridas utilizan un contenedor nativo para cargar y ejecutar contenido web en su interior, lo que les permite tener un aspecto similar al de una aplicación nativa y funcionar en diferentes plataformas móviles sin tener que ser desarrolladas específicamente para cada una de ellas como ocurre con las aplicaciones nativas.

Las actualizaciones se desarrollan en el servidor porque el contenido que se ejecuta dentro del contenedor nativo es en realidad una aplicación web, y el servidor es donde se encuentra alojada esta aplicación.

Al actualizar la aplicación web alojada en el servidor, se puede mejorar la funcionalidad de la aplicación híbrida sin tener que distribuir una nueva versión de la aplicación a través de las tiendas de aplicaciones. Esto permite realizar actualizaciones con mayor frecuencia y rapidez sin necesidad de esperar a que las tiendas de aplicaciones aprueben y distribuyan una nueva versión de la aplicación. [14][15][17][20][21][22]

2.1.3.1 VENTAJAS Y DESVENTAJAS

En primer lugar, se enumeran las ventajas que presenta este tipo de aplicación [14][15][17][20][21][22]:

- Bajo costo de desarrollo: El desarrollo de aplicaciones híbridas puede ser más económico que el desarrollo de aplicaciones nativas, ya que no requiere la creación de diferentes versiones de la aplicación para cada plataforma.
- Acceso a múltiples plataformas: Las aplicaciones híbridas pueden ser diseñadas para funcionar en múltiples plataformas, como Android e iOS, lo que las hace más accesibles para un público más amplio.
- Facilidad de actualización: Las aplicaciones híbridas pueden ser actualizadas de manera rápida y sencilla, ya que las actualizaciones se llevan a cabo en el servidor y se envían al dispositivo.
- No es necesaria una conexión a internet para ejecutar la aplicación, el contenido de la aplicación web se descarga del servidor y se almacena en caché. Esto significa que, generalmente solo se necesita conexión a internet para descargar y actualizar el contenido de la aplicación web en el dispositivo del usuario.
- Se distribuyen a partir de la App Store y Google Play.

Para finalizar se exponen las desventajas que presentan [14][15][17][20][21][22]:

- Rendimiento limitado: Las aplicaciones híbridas pueden tener un rendimiento más lento que las aplicaciones nativas, especialmente cuando se trata de tareas que requieren un procesamiento intensivo. Además, generalmente suelen ocupar más espacio que las aplicaciones nativas. Esto puede dar lugar a una peor experiencia de usuario.
- Limitaciones en el acceso a funciones del dispositivo: Las aplicaciones híbridas pueden tener limitaciones en el acceso a ciertas funciones del dispositivo, lo que puede restringir la capacidad de la aplicación para ofrecer ciertas características.
- Interfaz de usuario no nativa: Aunque las aplicaciones hibridas pueden emular la apariencia de una aplicación nativa, la interfaz de usuario pude sentirse diferente y menos intuitiva que la de una aplicación nativa.

En conclusión, las aplicaciones hibridas ofrecen la ventaja de ser multiplataforma y tener un bajo costo de desarrollo, pero pueden tener limitaciones en el rendimiento y el acceso a las funciones del dispositivo, y la interfaz de usuario puede sentirse menos nativa.

2.1.4 SELECCIÓN DEL TIPO DE APLICACIÓN

Para llevar a cabo el desarrollo de la aplicación de rehabilitación se ha decidido elegir el desarrollo de una aplicación hibrida, a continuación, se muestran las razones que han llevado a tomar esta decisión:

- Costo y tiempo de desarrollo: desarrollar una aplicación híbrida puede ser más rápido y económico que desarrollar dos aplicaciones nativas para iOS y Android. Esto se debe a que las aplicaciones híbridas utilizan un único código base que se adapta a diferentes plataformas, lo que reduce el tiempo y el costo de desarrollo [22].
- Acceso a características del dispositivo: las aplicaciones hibridas tienen acceso a muchas de las características de los dispositivos móviles, como la cámara, los sensores, la geolocalización, etc [23].
- Experiencia de usuario: las aplicaciones híbridas pueden ofrecer una experiencia de usuario buena pero inferior a la de una aplicación nativa, pueden usar elementos de interfaz de usuario nativos. Esto permite que la aplicación híbrida se sienta más natural para los usuarios que una aplicación web [20].
- Funcionamiento offline: las aplicaciones híbridas pueden funcionar offline, ya que el contenido web se puede almacenar en caché en el dispositivo móvil. Esto significa que los usuarios pueden seguir utilizando la aplicación, aunque no tengan conexión a internet [21].
- Mantenimiento y actualizaciones: las aplicaciones híbridas se pueden actualizar de forma más sencilla que las aplicaciones nativas, ya que las actualizaciones se realizan en el servidor y los usuarios solo necesitan descargar la última versión de la aplicación. Esto puede reducir tiempo y los costos asociados con el mantenimiento y la actualización de la aplicación [17].

En resumen, las aplicaciones híbridas son una buena opción cuando se busca una experiencia de usuario similar a la de una aplicación nativa, pero se quiere reducir el tiempo y el costo de desarrollo, y se necesita un funcionamiento offline y una actualización sencilla. Sin embargo, si se requiere un alto rendimiento o acceso a características específicas de la plataforma, una aplicación nativa podría ser la mejor opción. Si el contenido principal de la aplicación es una página web, una aplicación web sería la mejor opción. En la *Fig.2* se muestra una tabla resumen en la que se muestran las propiedades que poseen o no, los diferentes tipos de aplicaciones que se han expuesto en esta sección [14][15][17][23].

Experiencia de usuario	EXCELENTE	EXCELENTE	BUENA	NORMAL
Velocidad y Rendimiento	MUY RÁPIDA	MUY RÁPIDA	BUENA	NORMAL
Seguridad	ALTA	ALTA	BUENA	BUENA
App Stores	SI	SI	NO	NO
Función Offline	SI	SI	NO	NO
Tiempo de Desarrollo	ALTO	MEDIO	BAJO	BAJO
Mantenimiento	ALTO	MEDIO	BAJO	BAJO
Coste de Desarrollo	ALTO	MEDIO	MEDIO	MEDIO

Fig. 2 Comparativa de los diferentes tipos de aplicaciones móviles [14].

2.2 PLATAFORMA Y ENTORNO DE DESARROLLO

En la sección anterior ya se ha elegido el tipo de tecnología con el que se va a desarrollar la aplicación, en esta hablaremos del framework o marco de trabajo y del lenguaje de programación elegidos para el desarrollo.

2.2.1 ENTORNO DE TRABAJO FLUTTER

En este primer apartado se va a profundizar más acerca de Flutter. Primero se realizará una descripción detallada, tras esto se pasará a mencionar las ventajas y desventajas, se hablará sobre el futuro que se prevé para este Kit de Desarrollo Software (SDK) y por último se mostrará como crear un proyecto en el editor Visual Studio Code.

2.2.1.1 HISTORIA DE FLUTTER

El proyecto Flutter fue presentado por primera vez en la conferencia Dart Developer Summit en octubre de 2017, aunque el trabajo en el proyecto se había iniciado varios años antes en 2015. El equipo de Flutter estaba buscando crear un framework que pudiera proporcionar una experiencia de desarrollo de aplicaciones móviles de alta calidad y reactiva, así como un alto rendimiento y una interfaz de usuario personalizable [24].

Flutter fue desarrollado por un equipo de ingenieros de Google, liderado por el ingeniero de software Eric Seidel. El equipo decidió utilizar el lenguaje de programación Dart, que fue desarrollado por

Google en 2011, para crear el framework. Dart fue elegido por su facilidad de uso, su capacidad para compilar a código nativo y su capacidad para manejar grandes proyectos [26].

La primera versión estable de Flutter fue lanzada en diciembre de 2018, y desde entonces ha habido numerosas actualizaciones y mejoras en el framework. En mayo de 2019, Flutter fue anunciado como el principal framework de desarrollo de aplicaciones móviles para Google, lo que indica el compromiso de la empresa con la plataforma [27].

Desde su lanzamiento, Flutter ha ganado una amplia aceptación entre los desarrolladores, gracias a su enfoque en la personalización, la reactividad y la rapidez, así como a su capacidad para crear aplicaciones nativas para múltiples plataformas a partir de una sola base de código. En la actualidad, Flutter es uno de los frameworks de desarrollo de aplicaciones móviles más populares del mundo, y ha sido utilizado para crear aplicaciones para empresas como Alibaba, eBay, Google, Tencent y más [25].

2.2.1.2 DESCRIPCIÓN

El framework de Flutter es una plataforma de desarrollo de aplicaciones móviles y web de alta calidad creada por Google. Utiliza el lenguaje de programación Dart y se basa en el concepto de widgets personalizables para construir la interfaz de usuario y proporcionar la funcionalidad de la aplicación [30].

Una de las principales características del framework de Flutter es su capacidad para crear aplicaciones nativas para múltiples plataformas, incluyendo iOS, Android, Windows, Mac, Linux y la web, a partir de una única base de código. Esto significa que los desarrolladores pueden crear aplicaciones que se ejecutan en múltiples plataformas con una sola aplicación [28].

El framework de Flutter también se centra en la reactividad y la rapidez, lo que significa que las aplicaciones creadas con Flutter tienen una respuesta rápida a las interacciones del usuario y animaciones fluidas. El framework utiliza un motor de renderizado personalizado que proporciona una alta calidad visual y una animación fluida, lo que hace que las aplicaciones creadas con Flutter tengan una sensación de calidad superior [17][29].

Otra característica importante del framework de Flutter es su enfoque en la personalización. Los desarrolladores pueden personalizar fácilmente la apariencia y la funcionalidad de las aplicaciones mediante la creación de widgets personalizados. Además, Flutter ofrece una amplia variedad de widgets predefinidos que se pueden personalizar según las necesidades de la aplicación [27].

El framework de Flutter también ofrece una amplia gama de herramientas de desarrollo, que incluyen un conjunto de herramientas de línea de comandos para crear, probar y depurar aplicaciones, así como un completo conjunto de herramientas de desarrollo integradas (IDE) para la edición de código y la gestión de proyectos [33].

Por último, el framework de Flutter tiene una gran comunidad de desarrolladores y una documentación que se actualiza continuamente, lo que hace que sea fácil para los nuevos desarrolladores aprender y utilizar el framework. Además, Flutter se integra bien con otras tecnologías y herramientas populares, como Firebase, lo que lo hace aún más atractivo para los desarrolladores [32].

2.2.1.3 VENTAJAS

Flutter ofrece una serie de ventajas para los desarrolladores de aplicaciones móviles, incluyendo [25][29][30][32][34][35]:

1. Desarrollo rápido de aplicaciones: Flutter es conocido por su rápido desarrollo de aplicaciones debido a su capacidad de "hot reload", que permite a los desarrolladores ver los cambios en tiempo real sin tener que reiniciar la aplicación [17].
2. Multiplataforma: Flutter te permite desarrollar aplicaciones para múltiples plataformas, como iOS, Android, web y escritorio, utilizando un solo código base. Esto supone poder ahorrar tiempo y recursos en el desarrollo de aplicaciones para diferentes plataformas.
3. Rendimiento: Flutter utiliza el lenguaje de programación Dart, que se compila a código nativo, lo que significa que las aplicaciones Flutter tienen un alto rendimiento y una respuesta rápida [34].
4. Diseño personalizado: Flutter tiene una amplia variedad de widgets personalizables y animaciones que te permiten crear una experiencia de usuario única y personalizada.
5. Comunidad activa: Flutter tiene una comunidad activa y en crecimiento [35]. Lo que significa que hay muchos recursos disponibles en línea, incluyendo documentación, tutoriales y paquetes de terceros.
6. Integración sencilla: Flutter se integra fácilmente con otras tecnologías y servicios, como Firebase, lo que te permite agregar funcionalidades adicionales a tus aplicaciones sin tener que escribir mucho código.

Flutter es por lo tanto una herramienta de desarrollo móvil muy potente y eficiente que permite a los desarrolladores crear aplicaciones de alta calidad para múltiples plataformas de manera rápida y eficiente. Con su amplia compatibilidad, excelente rendimiento, personalización y comunidad activa.

2.2.1.4 INCONVENIENTES

Aunque Flutter es un marco de desarrollo móvil muy popular y poderoso, también tiene algunos inconvenientes que deben ser considerados [25][29][30][32][34][35]:

1. Tamaño de la aplicación: Como Flutter incluye su propio motor de renderizado, el tamaño de la aplicación puede ser más grande que las aplicaciones nativas. Esto puede afectar la velocidad de descarga y la cantidad de espacio de almacenamiento que ocupa la aplicación en el dispositivo.
2. Curva de aprendizaje: Aunque Dart es un lenguaje de programación fácil de aprender, puede haber una curva de aprendizaje para los desarrolladores que no están familiarizados con él. Además, la creación de widgets personalizados en Flutter puede requerir un mayor nivel de conocimiento técnico y tiempo de desarrollo.
3. Soporte de terceros: Aunque Flutter tiene una gran comunidad de desarrolladores, algunos servicios y herramientas de terceros pueden no estar completamente integrados o ser compatibles con el marco de desarrollo.
4. Limitaciones de la plataforma: Aunque Flutter es un marco de desarrollo móvil muy flexible, algunas funciones avanzadas y específicas de la plataforma pueden no estar disponibles en

Flutter. Por ejemplo, el soporte para integraciones de hardware específicas puede ser limitado.

En resumen, Flutter es un marco de desarrollo móvil potente y versátil que ofrece muchas ventajas para los desarrolladores. Sin embargo, es importante tener en cuenta que también tiene algunos inconvenientes, como el tamaño de la aplicación, la curva de aprendizaje y la dependencia de Google.

2.2.1.5 VISION DE FUTURO

El mercado de aplicaciones móviles está experimentando un crecimiento exponencial año tras año. Cada vez son más las empresas que optan por desarrollar aplicaciones en lugar de páginas web, lo que representa una tendencia en alza en comparación con años anteriores.

Dentro de las diferentes tecnologías disponibles para desarrollar aplicaciones, las aplicaciones híbridas son las que más han evolucionado y crecido en popularidad. Actualmente, estas aplicaciones ofrecen funcionalidades similares a las de las aplicaciones nativas, lo que las hace aún más interesantes para los usuarios.

En este contexto, existen numerosos frameworks para desarrollar aplicaciones híbridas, siendo Flutter el que más está creciendo en popularidad. Según una encuesta realizada por JetBrains y recogida por Statista, cada vez son más los desarrolladores que eligen Flutter, pasando del 30% en 2019 al 46% en 2022, superando a su principal competidor, ReactNative [30][31]. En la *Fig. 3* se muestran estos resultados. Actualmente, se estima que existen alrededor de 500.000 aplicaciones desarrolladas con Flutter.

Es importante destacar que Google está trabajando en un nuevo sistema operativo llamado Fuchsia, cuya interfaz de usuario y aplicaciones estarán desarrolladas en Flutter. Esto demuestra el compromiso de la empresa en seguir potenciando y mejorando esta tecnología [30].

En definitiva, Flutter tiene un futuro muy prometedor debido a su rápida evolución y crecimiento en popularidad. Esto la convierte en una tecnología atractiva para los desarrolladores y empresas que buscan desarrollar aplicaciones de alta calidad y rendimiento.

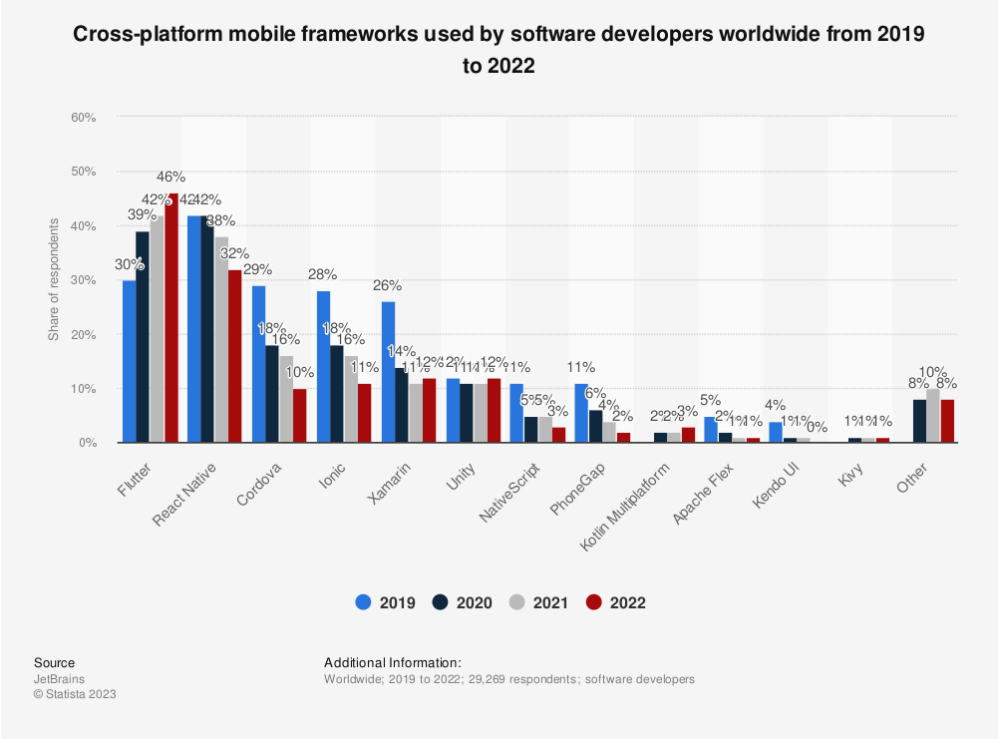

Fig. 3 Porcentaje de uso de los diferentes frameworks de desarrollo de aplicaciones de 2019 a 2022 [31].

2.2.2 DART

En esta sección, se describirá el lenguaje de programación utilizado en Flutter, incluyendo sus principales ventajas. Además, se realizará una comparación entre los editores de código Android Studio y Visual Studio Code. Se detallarán los pasos necesarios para instalar el editor elegido para este proyecto y, finalmente, se presentarán los pasos a seguir para crear un proyecto Flutter en Visual Studio Code.

2.2.2.1 DESCRIPCIÓN

Dart es un lenguaje de programación de alto nivel, desarrollado por Google en 2011, que se utiliza principalmente para crear aplicaciones web, móviles y de escritorio. Este lenguaje ofrece una alternativa a otros lenguajes populares como JavaScript y Java [30].

Una de las principales características de Dart es su capacidad para ser utilizado tanto en el lado del cliente como en el del servidor, lo que lo hace ideal para desarrollar aplicaciones full-stack. Además, Dart es tipado, lo que significa que cada variable o constante tiene un tipo específico. Sin embargo, el tipado en Dart es opcional, lo que permite a los desarrolladores simplificar la sintaxis del código mediante la inferencia de tipos.

La sintaxis de Dart es fácil de leer y escribir, con una estructura de control de flujo similar a la de otros lenguajes como Java o C#. La programación orientada a objetos es la base de Dart, en donde todo es un objeto, incluso las funciones. Dart permite crear objetos mediante clases y tiene herencia de clases, interfaces y mixins, lo que facilita la creación de código modular y reutilizable [37].

Dart también cuenta con una biblioteca estándar completa que incluye una gran cantidad de funcionalidades para realizar operaciones comunes, lo que significa que los desarrolladores pueden crear aplicaciones sin necesidad de utilizar bibliotecas de terceros. Además, Dart tiene una gran capacidad de programación asíncrona, lo que permite a los desarrolladores escribir código que no bloquee la ejecución del programa mientras se espera a que se realice una tarea, como una solicitud de red [37].

Finalmente, Dart puede ser compilado de dos maneras diferentes: JIT (Just In Time) y AOT (Ahead Of Time). El compilador JIT compila el código durante la ejecución del programa, lo que permite una mayor flexibilidad y un rápido tiempo de iteración en el proceso de desarrollo. El compilador AOT, por otro lado, compila el código antes de la ejecución del programa, lo que lo hace ideal para el despliegue de aplicaciones en producción y para mejorar el rendimiento en dispositivos móviles y web. En la *Fig.4* se muestra una imagen del esquema de estos dos tipos de compilación, donde se muestran los pasos que siguen ambos tipos [36].

Como conclusión, Dart es un lenguaje de programación moderno y eficiente que ofrece una amplia variedad de características útiles para los desarrolladores de aplicaciones.

Fig. 4 Pasos realizados en la compilación AOT y JIT [36]

2.2.2.2 EDITORES DE CÓDIGO

Android Studio y Visual Studio Code son dos de las opciones más populares para desarrollar en Flutter. Ambos son muy utilizados por la comunidad de Flutter, pero hay algunas diferencias entre ellos que pueden influir en la elección de uno u otro. A continuación, se presenta una comparación entre Android Studio y Visual Studio Code [17][38]:

1. Características y funcionalidades: Ambos editores de código tienen características y funcionalidades similares, como la integración con Flutter y el soporte para la depuración y la refactorización de código. Sin embargo, Android Studio tiene más herramientas específicas para el desarrollo de aplicaciones de Android, como el diseñador de diseño visual y la vista previa en tiempo real. Por otro lado, Visual Studio Code tiene una interfaz de usuario más personalizable y una gran cantidad de extensiones disponibles.

2. Rendimiento: Android Studio puede ser más pesado en términos de recursos de la computadora en comparación con Visual Studio Code, ya que es un IDE completo con muchas características adicionales. Sin embargo, esto puede depender del tamaño del proyecto y de la cantidad de extensiones instaladas en cada editor.
3. Comunidad y soporte: Ambos editores de código tienen una gran comunidad de usuarios y documentación disponible. Sin embargo, Android Studio es el IDE oficial de desarrollo de Android y Flutter, por lo que es probable que tenga una mayor cantidad de recursos y documentación disponibles específicamente para el desarrollo de aplicaciones de Android.
4. Integración con otras herramientas: Android Studio tiene una integración más estrecha con otras herramientas de desarrollo de Android, como el SDK de Android y Gradle, mientras que Visual Studio Code es más compatible con otras herramientas y tecnologías.

En general, la elección del editor de código para desarrollar en Flutter dependerá de factores como las preferencias personales, las necesidades del proyecto y la familiaridad con cada editor. Tanto Android Studio como Visual Studio Code tienen características y funcionalidades sólidas para el desarrollo de aplicaciones de Flutter.

En el caso específico de este proyecto de ayuda a la rehabilitación, se ha optado por utilizar Visual Studio Code debido a que no se trata de un proyecto de gran envergadura y porque se ha comprobado que funciona a mayor velocidad en el portátil utilizado para el desarrollo. Sin embargo, es importante tener en cuenta que la elección del editor de código dependerá de las necesidades y preferencias específicas de cada desarrollador y proyecto.

PASOS QUE SEGUIR PARA INSTALAR Y CONFIGURAR VISUAL STUDIO CODE COMO EDITOR DE FLUTTER:

Desde la página de Flutter nos muestran una guía de como instalar Visual Studio Code como editor de Flutter. A continuación, se muestran los pasos a seguir [39][40]:

1. Descargar e instalar Visual Studio Code: Acceder a la página oficial de Visual Studio Code (https://code.visualstudio.com/download) y descargar la versión que corresponda con el sistema operativo. Seguir los pasos de instalación para completar la instalación en el ordenador.
2. Descargar e instalar el SDK de Flutter desde su sitio web oficial (https://flutter.dev/docs/get-started/install).
3. Descargar e instalar Android Studio desde el sitio web oficial (https://developer.android.com/studio).
4. Iniciar Android Studio y seguir los pasos del ‘Asistente de configuración de Android Studio’, de esta forma se instala el SDK de Android más reciente, las herramientas de line de comandos de SDK de Android y las herramientas de compilación de SDK de Android, requeridas por Flutter para desarrollar en Android.
5. En la consola desde el directorio que contiene el SDK de Flutter, ejecutar el comando “C:\src\flutter>flutter doctor” para comprobar si se necesita alguna dependencia de la plataforma para completar la configuración. Si esta todo correcto se mostrara lo que aparece en la imagen mostrada a continuación, *Fig.5*.

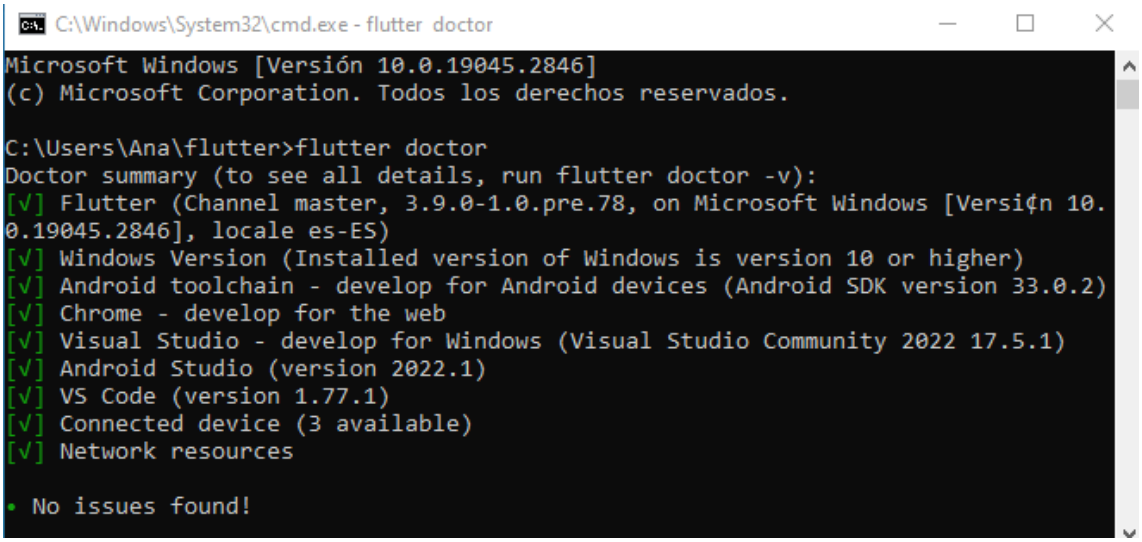

Fig. 5 Ejecución de flutter doctor en la consola

6. Iniciar Visual Studio Code y acceder a View>Command Palette, aquí selecciona 'Extensions: Install Extension'. También puede presionar 'Ctrl+Shift+X' (en Windows) o 'Cmd+Shift+x' (en Mac) para abrir la sección de Extensiones. En las imágenes que aparecen a continuación, *Fig.6* y *Fig.7*, se muestra cómo realizar este paso.

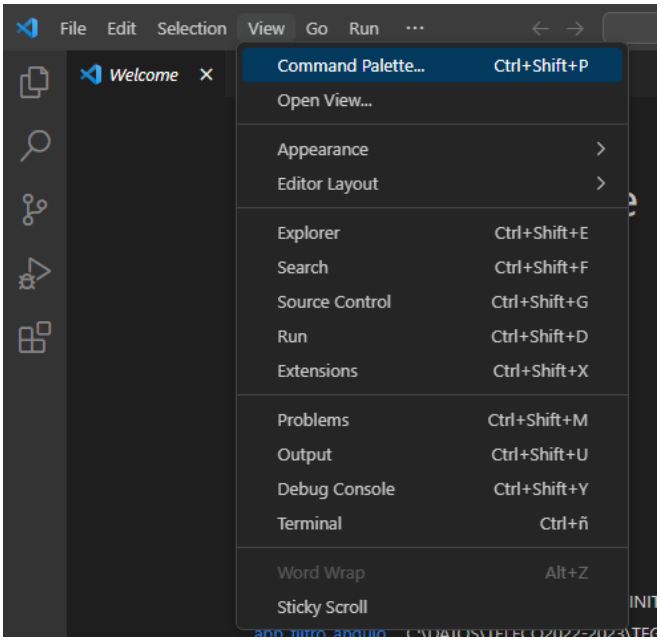

Fig. 6 View>Command Palette

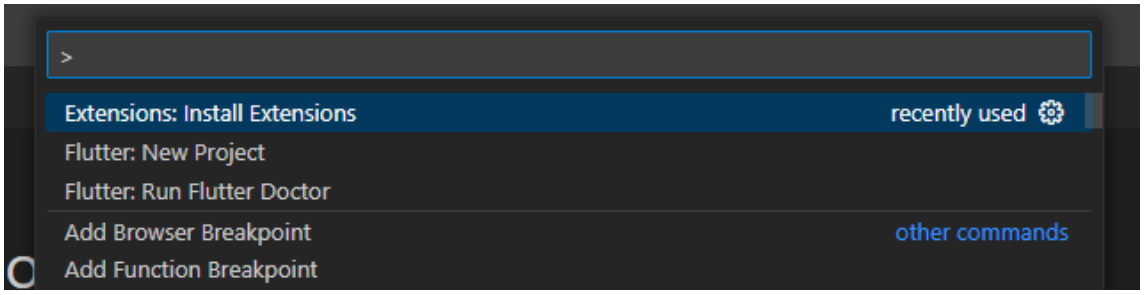

Fig. 7 Clicar en "Extensions: Install Extensions"

7. Dentro de la sección Extensiones buscamos Flutter y Dart e instalamos ambas, *Fig.8*.

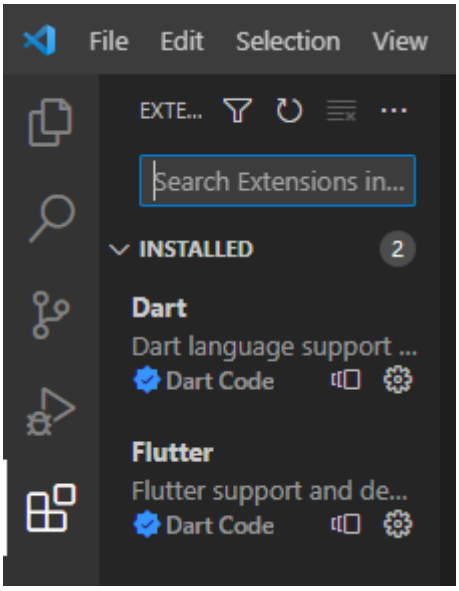

Fig. 8 Buscar las extensiones de Flutter y Dart, en este caso aparecen como instaladas porque ya lo están.

8. Una vez instaladas ambas hay que reiniciar Visual Studio Code para que se carguen correctamente. Y ya estaría todo listo para trabajar con el editor Visual Studio Code para proyectos Flutter.

2.2.3 CREACIÓN DE UN PROYECTO FLUTTER EN VISUAL STUDIO CODE

En esta sección se van a mostrar los pasos para crear un proyecto Flutter en Visual Studio Code:

1. Abrimos Visual Studio Code
2. Acceder a “View > Command Palette” o presionar ctrl+shift+P (en Windows)
3. En Command Palette seleccionamos Flutter: New Project, *Fig.9.*

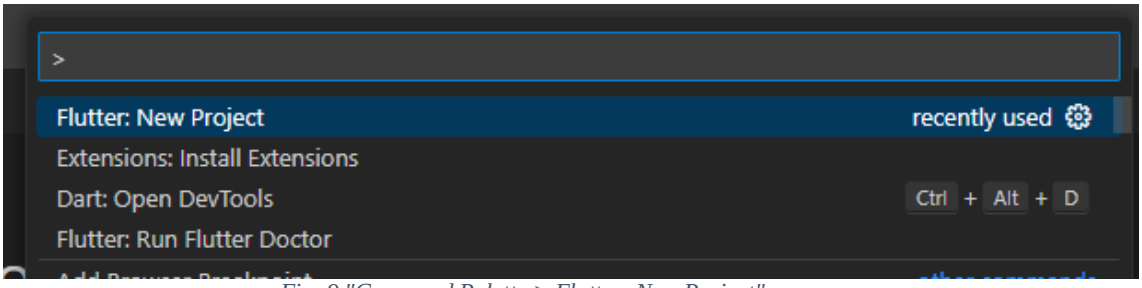

Fig. 9 "Command Palette > Flutter: New Project"

4. Seleccionamos Aplication, *Fig.10.*

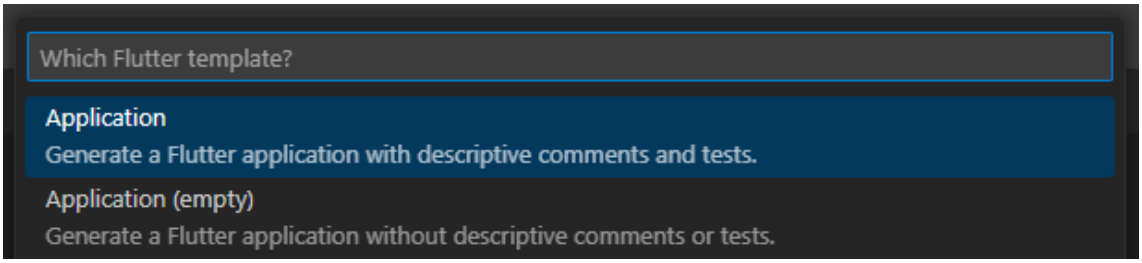

Fig. 10 Seleccionar la opción Application

5. Seleccionar el directorio donde se quiere guardar el proyecto Flutter.
6. Escribir un nombre para el proyecto.
7. Tras el paso 6 aparecerá una ventana emergente, clicar en “Yes, i trust the authors”, *Fig.11.*

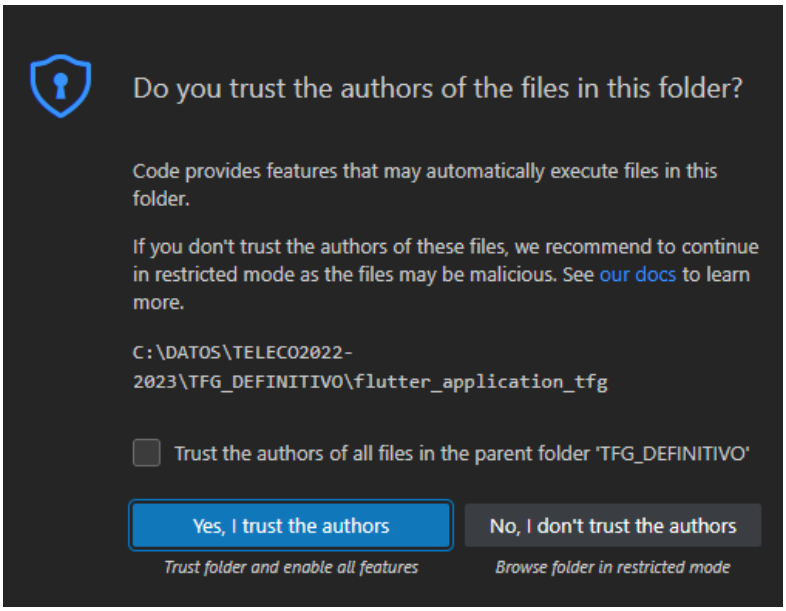

Fig. 11 Ventana emergente que aparece al crear una aplicación Flutter

8. Ya está creado el nuevo proyecto Flutter y listo para trabajar en él. Al generarse se crea una aplicación por defecto un “Hola Mundo”.

3. EL HARDWARE

En esta sección se va a exponer toda la información relacionada con la parte hardware del trabajo, es decir, los sensores del móvil. En primer lugar, se hablará brevemente de la goniometría, posteriormente se pasará a describir en qué consisten los sensores de acelerómetro, giroscopio y magnetómetro, haciendo hincapié en las características de los incluidos en los smartphones.

3.1 GONIOMETRÍA

La goniometría es una técnica de medición de ángulos que proviene del griego gonía (ángulo) y metrón (medida). En el ámbito de la salud, la goniometría tiene dos objetivos principales: evaluar la posición de la articulación en el espacio y medir el arco de movimiento de una articulación en cada uno de los tres planos del espacio (sagital, frontal y transversal). Esta técnica se utiliza comúnmente en fisioterapia, ortopedia y otras áreas de la medicina relacionadas con la movilidad y el funcionamiento articular [42].

El objetivo de la medida del rango de movilidad articular (ROM) es evaluar la capacidad de una articulación para realizar movimientos completos y sin dolor, y determinar si existen restricciones o limitaciones en el movimiento. Es muy útil para evaluar y tratar lesiones como desgarros de ligamentos, esguinces, tendinitis, artritis y otras dolencias articulares. Además, se utiliza para monitorear la progresión de la rehabilitación.

Para llevar a cabo el estudio goniométrico, se emplea un goniómetro, puede observarse en la *Fig. 12*, un instrumento que consta de un cuerpo y dos brazos con escala graduada, uno fijo y otro móvil, que se ajustan a la articulación a medir. El cuerpo del goniómetro es un transportador de ángulos de 180º o 360º, con intervalos de 1º, 5º o incluso 10º. El brazo fijo es una extensión del cuerpo, mientras que el brazo móvil va unido al cuerpo mediante un remache en el centro de este [42].

Fig. 12 Goniómetro universal (https://www.alssamedical.com/producto/goniometro-de-plastico-360/)

La goniometría se utiliza para evaluar el rango de movimiento de una articulación y determinar la presencia de limitaciones o restricciones en el movimiento. También se utiliza para monitorear la progresión de la rehabilitación de una lesión articular, a fin de asegurarse de que el paciente está recuperando adecuadamente su movilidad y funcionamiento articular. El fisioterapeuta puede utilizar la goniometría para comparar la movilidad de una articulación lesionada con la movilidad de una articulación sana y establecer objetivos realistas para la recuperación del paciente [41][42].

3.2 SENSORES DEL SMARTPHONE

Es indudable el significativo avance que ha experimentado la tecnología de los dispositivos móviles en los últimos años. Las aplicaciones móviles también han evolucionado enormemente gracias a los continuos progresos en tecnología. Actualmente, la mayoría de los dispositivos móviles incorporan múltiples sensores, clasificados en tres categorías: de movimiento, ambientales y de posición. Estos sensores han permitido el desarrollo de muchas aplicaciones en el terreno de la salud (mhealth), que utilizan estos sensores para la medición de datos y el control de la salud. Como se mencionó en la introducción, se ha registrado un aumento en los estudios sobre aplicaciones de este tipo en los últimos años. Además, el número de descargas de aplicaciones *mHealth* ha aumentado, debido a una mayor conciencia sobre el bienestar y la salud. Esto también ha generado un aumento en el uso de dispositivos portátiles como smartwatches o pulseras de ejercicio que miden el pulso cardiaco, el nivel de oxígeno en sangre, entre otros. Cabe destacar que los sensores de los smartphones utilizan la tecnología MEMS.

Los siguientes apartados se centrarán en ofrecer una descripción detallada de la tecnología MEMS utilizada en los sensores de los smartphones, así como de los propios sensores y su funcionamiento.

3.2.1 MEMS

MEMS, del inglés Micro Electro-Mechanical System, se trata de una tecnología de proceso utilizada para crear pequeños dispositivos o sistemas integrados que combinan componentes mecánicos y eléctricos. En su fabricación se emplean técnicas de procesamiento por lotes de circuitos integrados (CI), su tamaño puede variar desde unos pocos micrómetros hasta milímetros [43][44].

El término MEMS se emplea en EE.UU., en Europa se suele emplear el término MST (Microsystems Technology) [45].

Los MEMS consisten en microestructuras mecánicas, microsensores, microactuadores y microelectrónica, todo esto integrado en un mismo chip de silicio [46].

Los sensores detectan cambios en el entorno del sistema a través de la medición de fenómenos. La electrónica del chip procesa esta información, indicando a los actuadores que reaccionen produciendo algún tipo de cambio en el ambiente.

Los MEMS se pueden encontrar en sistemas que abarcan aplicaciones automotrices, médicas, electrónicas, de comunicaciones y de defensa.

El termino MEMS puede inducir a duda, pero se trata de una tecnología de fabricación; un paradigma para diseñar y crear dispositivos y sistemas mecánicos complejos, así como su electrónica integrada utilizando técnicas de fabricación por lotes.

La tecnología MEMS permite fabricar componentes y dispositivos con mayor rendimiento y confiabilidad, combinando las ventajas obvias de tamaño físico, volumen, peso y costo reducidos. [43][44][45][46]

3.2.2 GIROSCOPIO

Los giroscopios MEMS empleados en los smartphones, son giroscopios vibratorios de efecto Coriolis, un tipo de giroscopio que utiliza una estructura vibratoria en miniatura para detectar la rotación y la dirección del movimiento del dispositivo, *Fig.13*.

El efecto Coriolis se utiliza en los giroscopios MEMS vibratorios para medir la velocidad angular o la rotación del dispositivo. Estos giroscopios consisten en una estructura vibratoria que puede moverse en dos direcciones perpendiculares, como un péndulo. Cuando el dispositivo gira, la fuerza de Coriolis se produce como resultado de la desviación en la dirección de la vibración de la estructura vibratoria. Esta fuerza es perpendicular a la dirección de la vibración y es proporcional a la velocidad angular del giroscopio MEMS [47].

La fuerza del efecto Coriolis es ficticia y se expresa de la siguiente forma [47]:

$$\vec{F}_c = -2m(\vec{w} \times \vec{v})$$

Donde:

- $\vec{F}_c$: Fuerza del efecto Coriolis
- m; Masa del objeto
- w: Velocidad angular del sistema de referencia rotativo
- v: Velocidad del objeto en coordenadas cartesianas.

Es importante tener en cuenta que la fuerza de Coriolis solo aparece cuando se trabaja en un sistema de referencia rotativo y no es una fuerza real como la fuerza de la gravedad o la fuerza electromagnética.

La fuerza de Coriolis causa un desplazamiento en la estructura vibratoria en la dirección perpendicular a la dirección de vibración, lo que provoca una variación en la frecuencia de vibración. Esta variación de frecuencia se puede medir utilizando sensores integrados en el giroscopio MEMS y se utiliza para determinar la velocidad angular o la rotación del dispositivo.

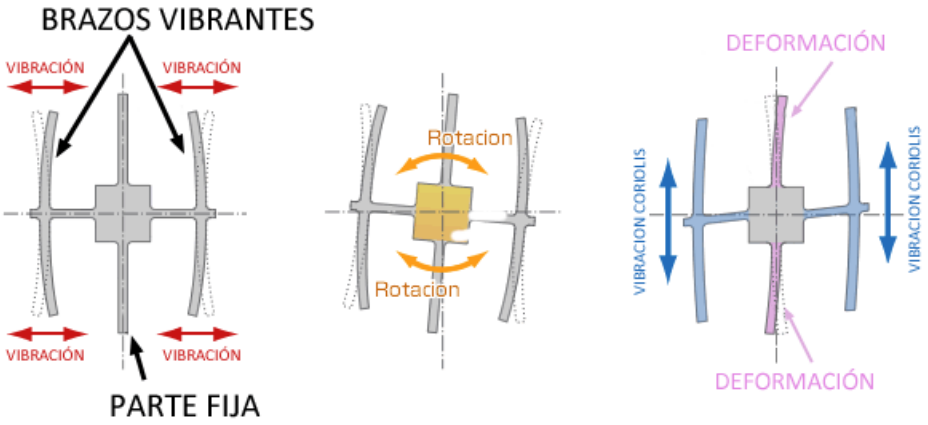

Fig. 13 Esquema de funcionamiento de un Giroscopio MEMS por vibración empleado en smartphones [47]

El giroscopio de un smartphone puede medir el movimiento angular en tres ejes de rotación: el eje X, el eje Y y el eje Z. El eje X se refiere al movimiento hacia adelante y hacia atrás del smartphone, mientras que el eje Y se refiere al movimiento lateral del dispositivo. El eje Z, por otro lado, se refiere al movimiento vertical del smartphone. La velocidad angular se suele expresar en en *°/s*, *rad/s* o revoluciones por segundo (*rps*).[49][47]

3.2.2.1 CONVENCIÓN DE ÁNGULOS

La convención Tait-Bryan de los ángulos de Euler es una convención comúnmente utilizada para describir la orientación de un objeto en el espacio tridimensional. Esta convención se basa en tres rotaciones elementales sucesivas, que se realizan en un orden específico, alrededor de tres ejes mutuamente perpendiculares. Los ángulos de Euler de rotación se definen como la rotación de un objeto alrededor de cada uno de estos ejes.

En la convención Tait-Bryan, se utilizan tres términos para describir las diferentes formas de rotación en un dispositivo móvil. En primer lugar, tenemos el "yaw" (ángulo de guiñada), que está relacionado con la inclinación del móvil. En segundo lugar, encontramos el "pitch" (ángulo de cabeceo), que mide la rotación alrededor del eje transversal del dispositivo. Por último, está el "roll" (alabeo en castellano), que indica la rotación alrededor del eje longitudinal del móvil.

Como se observa en la *Fig.14*, el ángulo de roll se refiere a la rotación alrededor del eje y, el ángulo de pitch se refiere a la rotación alrededor del eje x y el ángulo de yaw se refiere a la rotación alrededor del eje z.

La convención Tait-Bryan es utilizada en una amplia variedad de aplicaciones, desde la navegación y la robótica hasta la animación por computadora y los videojuegos. Es importante tener en cuenta que diferentes convenciones de ángulos de Euler pueden tener diferentes órdenes de rotación elemental y diferentes ejes de rotación, por lo que es importante comprender la convención específica que se está utilizando para evitar confusiones y errores.

La convención de ejes no depende del sistema operativo instalado en el dispositivo móvil, por lo que sirve para cualquier tipo de smartphone. [41][48]

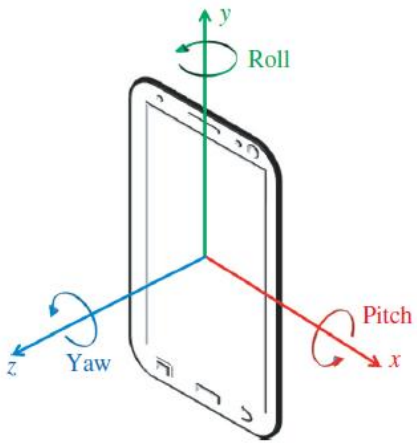

Fig. 14 Esquema de ángulos y ejes de Tait-Bryan [48]

3.2.2.2 ERRORES EN LAS MEDIDAS

Aunque los giroscopios vibratorios de efecto Coriolis son muy precisos, también pueden presentar errores en sus medidas debido a diversas causas, entre ellas:

- Errores de offset: pueden surgir desviaciones en la señal del giroscopio que no están relacionadas con la velocidad angular que se está midiendo. Estos errores pueden ser causados por la variación en el voltaje de alimentación, la temperatura, la vibración y otros factores ambientales.
- Errores de sensibilidad: pueden surgir desviaciones en la relación entre la velocidad angular y la señal de salida del giroscopio, lo que puede afectar la precisión de las medidas.
- Errores de no linealidad: la respuesta del giroscopio puede ser no lineal, lo que puede afectar la precisión de las medidas.
- Errores debidos a la aceleración: la aceleración puede generar fuerzas parasitarias que afectan la medida, especialmente en los giroscopios MEMS más sensibles.
- Errores debidos a la vibración: cualquier vibración en el giroscopio o en su entorno puede afectar la precisión de las medidas.

También hay que mencionar que los giroscopios vibratorios de efecto Coriolis pueden sufrir deriva, aunque en menor medida que otros tipos de giroscopios. La deriva es una desviación gradual de la medida de la velocidad angular en el tiempo, y puede ser causada por diversas fuentes, como la temperatura, la vibración y las variaciones en el voltaje de alimentación.

Sin embargo, la deriva en este tipo de giroscopios es generalmente mucho menor que en otros tipos, ya que la medida de la velocidad angular se realiza mediante la detección de una frecuencia de vibración específica, lo que permite una mayor estabilidad en la medida. Además, los giroscopios vibratorios de efecto Coriolis suelen incluir circuitos de retroalimentación que permiten corregir la deriva y mantener la precisión de las medidas en el tiempo.

Es importante destacar que estos errores pueden ser minimizados mediante una adecuada calibración y diseño del giroscopio, así como mediante el uso de técnicas de filtrado y algoritmos de fusión

sensorial para combinar la información de varios sensores y obtener una estimación más precisa de la orientación o velocidad angular del móvil.

3.2.3 ACELERÓMETRO

Los acelerómetros como su propio nombre indica, son dispositivos encargados de medir la aceleración o lo que es lo mismo la variación de velocidad por unidad de tiempo, generalmente medida en m/s^2, aunque también lo podemos encontrar medido en g, donde $1g = 9{,}8\ m/s^2$.

Hay diversos tipos según el tipo de tecnología empleada para medir la aceleración, mecánicos, capacitivos, piezoeléctricos, etc.

Las técnicas convencionales de medición de la aceleración se basan en la aplicación de la segunda ley de Newton $F = m \cdot a$, donde F es la fuerza, m la masa y a la aceleración. Los acelerómetros detectan la aceleración a través de una masa suspendida mediante un resorte elástico, de constante elástica k. Al aplicar una fuerza sobre el resorte la masa se desplaza una distancia x, mediante la Ley de Hooke, $F = k \cdot x$, obtenemos la fuerza aplicada. Llevando esto a la ecuación de Newton obtenemos que $k \cdot x = m \cdot a$, o lo que es lo mismo $a = k \cdot x/m$.

Los acelerómetros capacitivos MEMS son los empleados en los smartphones debido a su reducido tamaño y a la posibilidad de ir soldados a las placas de circuito de los móviles. Su funcionamiento está basado en la variación de la capacidad entre dos o más conductores entre los que hay un dieléctrico, en respuesta a cambios en la aceleración.

Al observar este tipo de acelerómetros, ver *Fig.15*, se puede apreciar una "H". Donde los delgados y largos brazos de esta están fijados al substrato y el resto de los elementos están libres para poder moverse. Está formado por unos filamentos finos que hacen de resorte, en cuyo centro se encuentra una masa, que será la placa móvil del condensador.

La aceleración ejerce una fuerza sobre la masa inercial generando un desplazamiento de las placas del condensador, que provoca un cambio en la capacidad de este. Este cambio se traduce en variaciones en la señal eléctrica que son directamente proporcionales a la fuerza aplicada de donde se obtiene la aceleración aplicando la segunda ley de Newton.

Con este tipo de dispositivos generalmente se obtiene la aceleración en los tres ejes cartesianos (X,Y,Z).

La fuerza de la gravedad actúa de forma continua, pero conocemos su valor de $9{,}8\ m/s^2$, por lo tanto, es fácil emplear el valor de cada eje del acelerómetro y obtener el ángulo de inclinación teniendo como resultado la posición del dispositivo en el espacio.

Una de las funciones principales de este tipo de sensor en los smartphones es obtener la posición del dispositivo para rotar la pantalla y facilitar así la visualización del contenido.

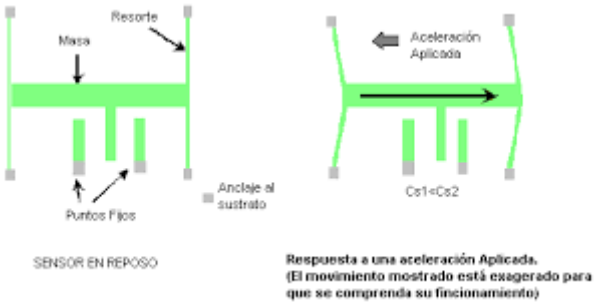

Fig. 15 Esquema acelerómetro MEMS empleado en smartphones [52]

A pesar de que el acelerómetro tiene la capacidad de medir cualquier ángulo, sufre del inconveniente de que sus mediciones son altamente ruidosas. El término "ruido" se refiere a fluctuaciones no deseadas en la señal de salida del sensor, las cuales pueden interferir en la medición precisa de la aceleración del dispositivo. Estas fluctuaciones pueden tener varias causas, como la interferencia electromagnética, las vibraciones mecánicas y el ruido térmico.

La interferencia electromagnética ocurre cuando la señal del acelerómetro se ve afectada por campos electromagnéticos producidos por otros dispositivos electrónicos cercanos al smartphone, como radios, televisores y otros teléfonos móviles. Por otro lado, las vibraciones mecánicas se producen cuando el dispositivo se mueve o vibra, lo que puede causar una fluctuación en la señal del acelerómetro. Además, el ruido térmico se produce debido a las fluctuaciones en la temperatura ambiente, las cuales pueden afectar el comportamiento del sensor.

Es importante tener en cuenta que el ruido en las medidas del acelerómetro puede tener un impacto significativo en la precisión y confiabilidad de las mediciones, lo que puede afectar a aplicaciones donde se requiere una alta precisión, como en la navegación o el seguimiento de la actividad física. Para reducir el impacto del ruido en las medidas, se pueden utilizar técnicas de filtrado de señales. [41][51][52]

3.2.4 MAGNETÓMETRO

Los magnetómetros de efecto Hall son sensores que utilizan el efecto Hall para medir la intensidad y la dirección de los campos magnéticos. Estos sensores se basan en una placa delgada de material semiconductor, que se coloca en un campo magnético. La placa está atravesada por una corriente eléctrica, que genera una diferencia de potencial eléctrico (voltaje) en la dirección perpendicular a la corriente y al campo magnético. Esta diferencia de potencial es proporcional a la intensidad del campo magnético.

El principio de funcionamiento de los magnetómetros de efecto Hall se basa en la ley de Faraday de la inducción electromagnética. Esta ley establece que un campo magnético variable en el tiempo induce una corriente eléctrica en un conductor cercano. En el caso de los magnetómetros de efecto Hall, el campo magnético es constante y la corriente eléctrica se aplica al material semiconductor. Sin embargo, el efecto es similar: el campo magnético produce una señal eléctrica proporcional a su intensidad.

Los magnetómetros de efecto Hall se utilizan en una amplia variedad de aplicaciones, incluyendo navegación, detección de metales, y monitoreo de campos magnéticos en dispositivos electrónicos. En los teléfonos móviles, se utilizan para detectar la orientación del dispositivo en relación con el campo magnético terrestre y para calibrar la brújula digital. [41]

3.2.5 FILTRO COMPLEMENTARIO

Es común encontrarse los sensores mencionados antes integrados en un único chip denominado IMU (Unidad de Medición Inercial). Estos dispositivos combinan sensores inerciales como el acelerómetro, giroscopio y magnetómetro. Realizan medidas de velocidad y orientación del dispositivo, así como también de la gravedad.

Como ya se ha comentado antes las medidas de los sensores acelerómetro y giroscopio presentan errores debidos a la deriva (drift) y al ruido. El acelerómetro es capaz de medir cualquier ángulo, pero sus medidas son muy ruidosas. En el caso del giroscopio como el que se emplea es un giroscopio por vibración, el ángulo lo obtienen integrando con respecto al tiempo la velocidad angular, lo que provoca que se vaya acumulando un error con los sucesivos cálculos.

La combinación de las medidas del acelerómetro y el giroscopio permite a la IMU obtener medidas de la orientación más precisas. Pero hay que tratar de evitar o eliminar estos problemas, para ello se hace uso de los filtros. Existen diversos filtros, uno de los más conocidos es el filtro de Kalman desarrollado en 1960 y considerado como uno de los mayores descubrimientos del siglo pasado.

El filtro de Kalman conlleva a la realización de cálculos complejos, que suponen una implementación y un tiempo de cálculo excesivo. Debido a esto es más habitual emplear el filtro complementario, que es una simplificación de Kalman.

El filtro complementario actúa como un filtro paso alto con el giroscopio para que únicamente permita pasar las señales de corta duración debido a que este dispositivo presenta problemas de deriva a medio y largo plazo. Sin embargo, con el acelerómetro actúa como filtro de paso bajo, ya que este dispositivo es muy rápido y sensible debido a esto sus datos son fiables sólo a largo plazo.

La expresión del filtro es la siguiente [51]:

$$\theta_k = \alpha \cdot \left(\theta_{k-1} + \theta_{gyro}\right) + (1 - \alpha) \cdot \theta_{accel}$$

Donde α y $(1 - \alpha)$ son las ganancias de los filtros paso alto y paso bajo respectivamente. El θ_{gyro} es el ángulo calculado por el giroscopio y θ_{accel} es el ángulo calculado con la salida del acelerómetro.

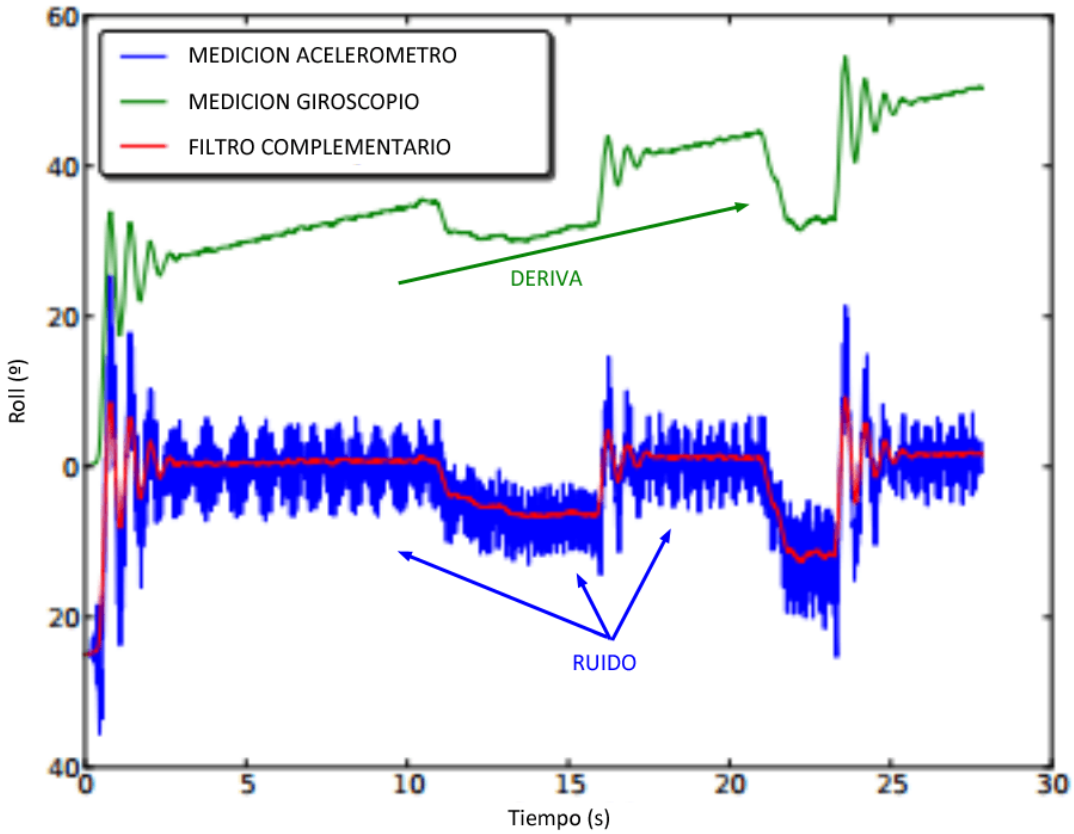

Fig. 16 Medición acelerómetro, giroscopio y filtro complementario

Con este filtro lo que se consigue es que la señal del giroscopio gobierna a corto plazo y la del acelerómetro lo hace a medio y largo plazo, esto puede verse en la *Fig.16*. Así se compensan las ventajas y defectos de cada sensor y se obtienen resultados más precisos.[50] [51]

4. DESARROLLO DE LA APLICACIÓN

Tras realizar una investigación, se comprobó que muchas de las aplicaciones de asistencia a la rehabilitación, que hacen uso de sensores como el acelerómetro y/o giroscopio empleaban la plataforma Arduino, también muchas empleaban MATLAB para el procesamiento de los datos de los sensores. Este tipo de aplicaciones utilizan una unidad de medición inercial (IMU) que contiene un acelerómetro y un giroscopio, la cual va conectada a una placa de Arduino. El paciente sostiene el dispositivo que contiene la IMU para registrar el movimiento, y todos los datos devueltos por los sensores son recopilados por el Arduino. Posteriormente, los datos son procesados en un ordenador donde se obtiene el ángulo de inclinación.

Inicialmente, se realizaron pruebas con la plataforma Arduino debido a la amplia documentación disponible en internet. Pero siguiendo con la investigación se comprobó que los smartphones actuales poseen una gran variedad de sensores, entre los que se incluye un acelerómetro de tres ejes e incluso un giroscopio de tres ejes en los de gamas superiores. Esto llevó a considerar desde el principio del proyecto la idea de desarrollar una aplicación móvil que aprovechara estos sensores, evitando así la necesidad de equipar al paciente con dispositivos adicionales.

La propuesta consiste en que la misma aplicación procese la información recogida por los sensores del smartphone y envíe al médico datos como el número de repeticiones realizadas, el tiempo de duración de cada repetición, el ángulo de movimiento alcanzado, la fecha y hora de inicio del ejercicio, tiempo total empleado en el ejercicio y media del ángulo alcanzado durante el ejercicio.

4.1 ChatGPT

Una parte interesante de este TFG ha sido la recomendación del tutor para comprobar la utilidad de las nuevas herramientas de IA en los desarrollos de ingeniería. Resulta muy práctico y formativo iniciarse en los recursos y facilidades que la IA proporciona para ayudar al profesional a llevar a cabo sus tareas técnicas, puesto que esta potente herramienta nos acompañará cada vez más en el futuro, aumentando nuestra capacidad de trabajo y mejorando el rendimiento.

Así pues, para llevar a cabo el desarrollo de la aplicación de ayuda a la rehabilitación, se ha contado en ocasiones con la ayuda de chatGPT. Esta IA que cada vez es más potente, ha servido de gran ayuda para obtener ideas y ejemplos de código que facilitaron la implementación. Además, se empleó para la corrección de errores que iban surgiendo a lo largo del desarrollo y para aclarar dudas en la comprensión del código y funcionalidades de Flutter.

4.1.1 CARACTERÍSTICAS

ChatGPT representa una de las inteligencias artificiales (IA) más potentes probadas hasta la fecha. Se trata de un sistema de chat basado en el modelo de lenguaje de IA ChatGPT-3.5, desarrollado por la empresa OpenAI y lanzado oficialmente el 30 de noviembre de 2022. El modelo ChatGPT-3.5 está compuesto por más de 175 millones de parámetros y ha sido entrenado con grandes cantidades de texto, lo que se conoce como modelos de lenguaje grandes (LLM), modelos de aprendizaje automático avanzados que pueden tanto comprender como generar lenguaje natural.

Actualmente ChatGPT se presenta en dos versiones, una gratuita basada en el modelo ChatGPT-3.5 y otra de pago ChatGPT Plus, que utiliza el modelo de lenguaje más avanzado ChatGPT-4 [54].

Todas las IA que conocemos han sido entrenadas mediante texto, realizando preguntas y proporcionando información. Con el tiempo, el sistema se va entrenando mediante correcciones para llevar a cabo su tarea de forma automática y lo más precisa posible.

La característica más destacada de chatGPT radica en su capacidad para generar respuestas muy completas y precisas. Asimismo, tiene la habilidad de expresarse de forma natural y proporcionar información muy exacta, lo que dificulta distinguir un texto creado por un humano de uno generado por esta IA. Aunque como cualquier IA comete errores por lo que no se puede considerar completamente infalible.

Además, esta IA es capaz de mantener el contexto de la conversación ya que reconoce todo lo que se ha hablado hasta el momento. Por lo tanto, si se realiza una pregunta relacionada con una respuesta anterior, identificará la relación y no será necesario proporcionarle de nuevo toda la explicación. [54][53]

Fig. 17 Logo de ChatGPT

4.1.2 USOS DE ChatGPT

Para comenzar a usar ChatGPT, simplemente hay que acceder a la página oficial, donde deberá crear una cuenta de forma gratuita para iniciar sesión por primera vez.

Tras el inicio de sesión podrá acceder al chat, como se muestra en la imagen. En la parte inferior encontrará una barra donde escribirá todas las cuestiones que quiera plantear a la IA, y en la parte de la izquierda aparecerá un historial con todas las cuestiones que haya realizado.

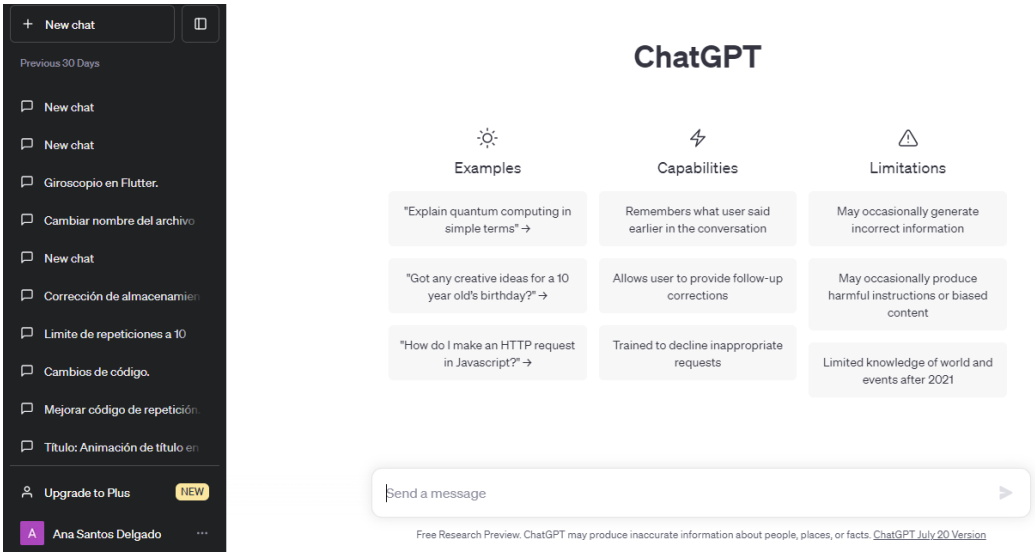

Fig. 18 Pantalla de inicio de ChatGPT

Hay que destacar también la versatilidad de ChatGPT, debido a que puede utilizarse en cualquier idioma. La IA responderá en el mismo idioma en el que se formuló la consulta.

Es importante tener en cuenta que todas las interacciones quedan registradas y podrán ser revisadas por los desarrolladores de OpenIA, con la finalidad de seguir entrenando a ChatGPT. Por lo que no es recomendable incluir información personal ni realizar peticiones que sean peligrosas o ilegales.

En el contexto de este proyecto chatGPT ha sido empleado para la generación de código. Se han presentado ideas para el desarrollo de la aplicación, y él ha generado fragmentos de código que podrían incorporarse directamente o realizando alguna modificación, en diversas partes del proyecto. Sin embargo, es importante destacar que el código generado no era perfecto, solía presentar fallos e incluso a veces perdía el contexto o generaba código obsoleto con las nuevas actualizaciones de Flutter.

A pesar de las limitaciones mencionadas en el párrafo anterior, ChatGPT era capaz de proporcionar comentarios tanto sobre el código que generaba como sobre el código que se le proporcionaba, en ocasiones también fue útil para corregir errores que surgieron durante el proceso de desarrollo. Su aporte resultó valioso en algunos aspectos y contribuyó a agilizar en cierta forma el trabajo a la hora de programar [54][53].

4.2 VERSIÓN DE PRUEBA: ARDUINO

Inicialmente, al realizar la búsqueda de artículos de investigación relacionados con el empleo de sensores en procesos de rehabilitación, se observó que muchos de ellos empleaban la plataforma Arduino, por lo que resultó fácil encontrar código documentado y explicado, que trabajase sobre los sensores de acelerómetro y giroscopio integrados en una IMU. Todo esto llevo a realizar una primera versión de prueba en Arduino.

Un inconveniente detectado en la versión de prueba de Arduino. Requería que los pacientes tuvieran que adquirir dispositivos adicionales aparte de sus móviles, utilizados como plataforma para la aplicación. Esto llevó a la versión final, en la que se prescindiría de la necesidad de dispositivos externos, aprovechando exclusivamente el potencial de los smartphones. En la actualidad, los smartphones están dotados de una diversidad de sensores, entre los que se incluyen el acelerómetro e incluso giroscopios en modelos de gama alta. La última versión se había considerado desde el principio como objetivo del proyecto, con la versión de Arduino se buscaba principalmente experimentar con un sensor comercial, analizar su comportamiento y ensayar algoritmos de procesado.

En esta sección se desarrollará una exposición detallada de Arduino.

Se presentará el proceso de cálculo de los ángulos de inclinación y rotación utilizando tanto el acelerómetro como el giroscopio, ya que estos cálculos constituyen la base fundamental de la aplicación. Todos estos ejemplos de Arduino donde se incluyen estos cálculos fueron encontrados en un tutorial en internet [55].

4.2.1 PRUEBAS CON ARDUINO UNO Y MPU6050

El MPU6050 constituye una Unidad de Medición Inercial (IMU por sus siglas en inglés). Combina un acelerómetro y un giroscopio, ambos con capacidad de detección en tres ejes. Las aplicaciones de este dispositivo se extienden en campos como navegación, goniometría y estabilización, entre otros.

En esta memoria en la sección dedicada al apartado de Hardware, se han abordado los principios generales del acelerómetro y giroscopio, por lo que directamente se pasará a estudiar con más detalle el MPU6050, así como la implementación de ejemplos prácticos del MPU6050 junto con la plataforma Arduino.

4.2.1.1 MÓDULO GIROSCOPIO Y ACELERÓMETRO MPU6050

Fig. 19 Módulo MPU6050 [55]

El módulo MPU6050, visible en la imagen superior, está equipado con un acelerómetro de tres ejes para medir la aceleración en las tres componentes X, Y y Z y un giroscopio de tres ejes para capturar

la velocidad angular en las tres componentes. El módulo indica claramente la orientación de los ejes, lo cual es esencial para evitar errores en los signos de las aceleraciones en cada componente.

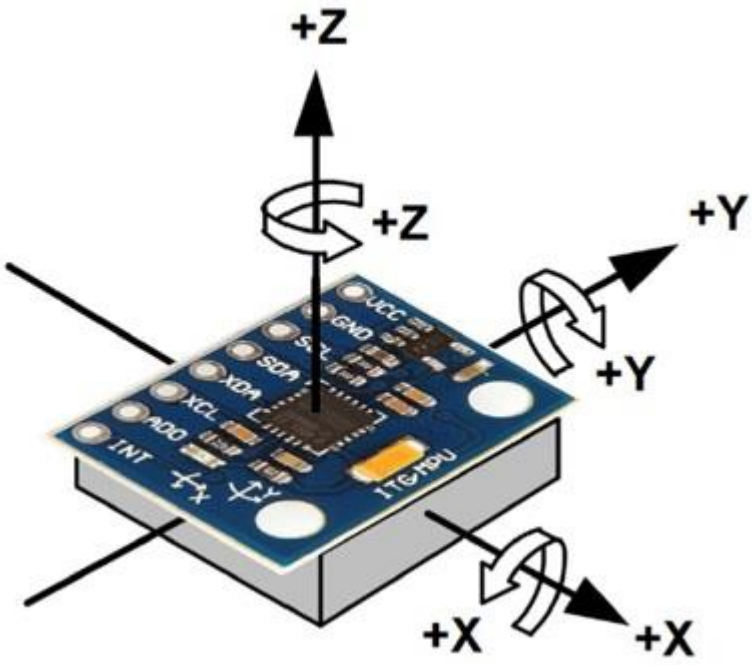

Fig. 20 Esquema de la dirección y signo de los ejes en el MPU6050 [55]

Este módulo utiliza el protocolo de comunicación I2C (Inter-Integrated Circuit en inglés), lo que hace posible que pueda trabajar con la mayoría de los microcontroladores. El protocolo I2C presenta una estructura maestro-esclavo, en esta configuración, el módulo opera como el esclavo mientras que la placa Arduino actúa como el maestro. Para establecer la comunicación, únicamente se emplean dos pines SCL, para la señal de reloj, y SDA, empleado para el intercambio de datos. Ambos pines están provistos de una resistencia pull-up en la placa, lo que facilita la conexión directa con el microcontrolador.

Existen dos direcciones I2C posibles:

Pin AD0	**Dirección I2C**
AD0 = HIGH (5V)	0x69
AD0 = LOW (GND)	0x68

Tabla 1 Direcciones I2C disponibles

El pin AD0 en el interior del módulo cuenta con una resistencia conectada a tierra (GND). Si este pin no se conecta, la dirección por defecto será 0x68. Además, el MPU6050 integra un regulador de voltaje en placa de 3,3V que puede ser alimentado con los 5V de la placa de Arduino.

Para implementar los códigos de este tutorial encontrado en internet [55], fue necesario descargar dos librerías, la librería MPU6050, desarrollada por Jeff Rowberg, y la librería I2Cdev, necesaria para la comunicación I2C del módulo. Ambas librerías deberán estar instaladas en el entorno de desarrollo de Arduino para poder desarrollar los códigos que hacen uso de los sensores del módulo.

La comunicación se lleva a cabo en modo I2C estándar. A continuación, se presentan tanto el esquema de conexiones entre el MPU6050 y la placa de Arduino UNO como en la *Tabla 2*, que detalla estas conexiones.[55]

MPU6050	Arduino UNO
VCC	5V
GND	GND
SCL	A5
SDA	A4

Tabla 2 Conexiones para el modo I2C estándar entre MPU6050 y Arduino UNO

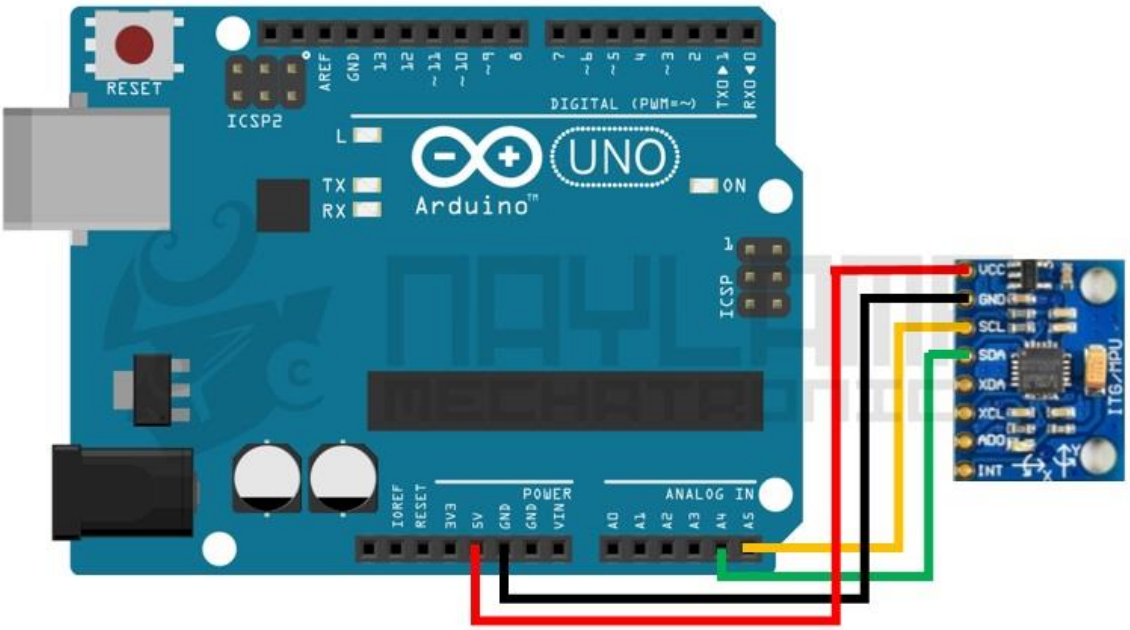

Fig. 21 Conexiones modo I2C estándar entre MPU6050 y Arduino UNO [55]

El montaje final se muestra en la imagen inferior: el módulo y la placa de Arduino se encuentran alojados en una caja de Raspberry Pi para facilitar el manejo durante los movimientos. Además, se ha dispuesto un cable de conexión entre el Arduino y el ordenador, con una longitud suficiente para permitir realizar movimientos de gran amplitud.

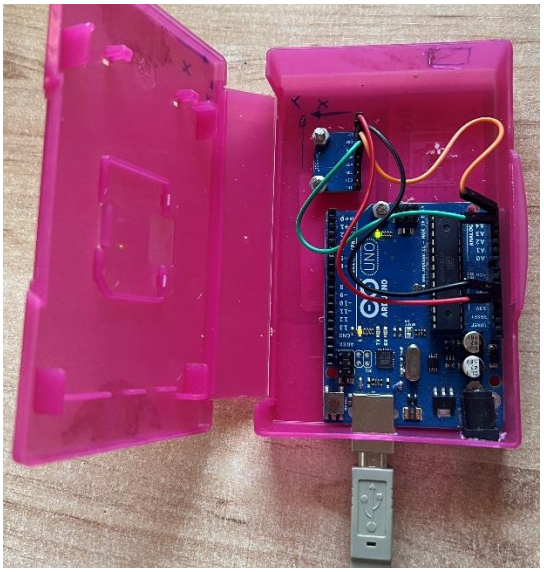

Fig. 22 Montaje Arduino UNO junto con IMU en una caja Raspberry PI, para recoger el movimiento

4.2.1.2 CÁLCULO DEL ANGULO DE INCLINACIÓN USANDO UNICAMENTE LOS VALORES DEL ACELERÓMETRO

Previo a la exposición del código en Arduino, se procede a desarrollar los cálculos para obtener el ángulo de inclinación del módulo a partir de los valores absolutos generados por el acelerómetro de tres ejes incorporado en el módulo.
Considerando que la gravedad es la única fuerza que actúa sobre el acelerómetro, los valores registrados en cada una de sus componentes corresponden a los de la gravedad en dicha componente. Dado que la gravedad es totalmente vertical, el ángulo de inclinación del plano del sensor se reflejará en los ángulos de la resultante.
En una configuración donde el sensor se inclina un ángulo θ y se considera el plano X-Z, el cálculo de dicho ángulo se lleva a cabo de la siguiente manera [55]:

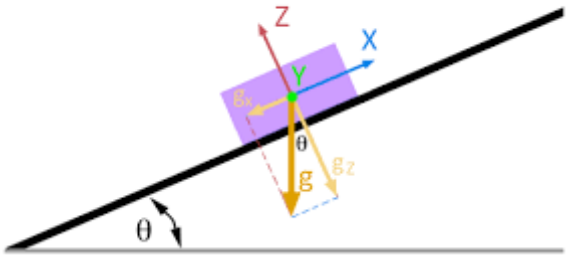

Fig. 23 Esquema para el cálculo del ángulo de inclinación en 2D[55]

Considerando el esquema anterior, la fórmula para el cálculo del ángulo de inclinación en un plano 2D se expresa como sigue:

$$\theta = \operatorname{atan}\left(\frac{g_x}{g_z}\right)$$

Para calcular los ángulos de inclinación en los ejes X e Y en un plano tridimensional, se emplean las siguientes fórmulas:

$$\theta_x = \tan^{-1}\left(\frac{a_x}{\sqrt[2]{{a_y}^2 + {a_z}^2}}\right)$$

$$\theta_y = \tan^{-1}\left(\frac{a_y}{\sqrt[2]{{a_x}^2 + {a_z}^2}}\right)$$

En estas fórmulas a_x, a_y y a_z representan las tres componentes del acelerómetro.

A continuación, se presenta el código comentado que se utiliza para calcular el ángulo de inclinación del módulo MPU6050. Este código se obtuvo de un tutorial de internet [55] y se ha comentado completamente para su correcta comprensión.

```
// Librerias I2C para controlar el mpu6050
// la libreria MPU6050.h necesita I2Cdev.h, I2Cdev.h necesita Wire.h
#include "I2Cdev.h"
#include "MPU6050.h"
#include "Wire.h"

// La dirección del MPU6050 puede ser 0x68 o 0x69, dependiendo
// del estado de AD0. Si no se especifica, 0x68 estará implicito
MPU6050 sensor;
// Variables que almacenaran los valores RAW (sin procesar) del acelerometro en //los ejes x,y,z
int ax, ay, az;
void setup() {
  Serial.begin(57600);    //Iniciando puerto serial
  Wire.begin();           //Iniciando I2C
  sensor.initialize();    //Iniciando el sensor

  if (sensor.testConnection()) Serial.println("Sensor iniciado correctamente");
  else Serial.println("Error al iniciar el sensor");
}
void loop() {
  // Leer las aceleraciones
  sensor.getAcceleration(&ax, &ay, &az);
  //Calcular los angulos de inclinacion:
  float accel_ang_x=atan(ax/sqrt(pow(ay,2) + pow(az,2)))*(180.0/3.14);
  float accel_ang_y=atan(ay/sqrt(pow(ax,2) + pow(az,2)))*(180.0/3.14);
  //Mostrar los angulos separadas por un [tab]
  Serial.print("Inclinacion en X: ");
  Serial.print(accel_ang_x);
  Serial.print("tInclinacion en Y:");
  Serial.println(accel_ang_y);
  delay(100);
}
```

Este fragmento de código se concentra en calcular los ángulos de inclinación del sensor en los ejes x e y. En la línea "*sensor.getAcceleration(&ax, &ay, &az);*", se capturan las aceleraciones a lo largo de los ejes x, y, z desde las direcciones del sensor, y se almacenan en las variables *ax*, *ay* y *az* respectivamente. En las dos líneas siguientes, "*float accel_ang_x=atan(ax/sqrt(pow(ay,2) + pow(az,2)))*(180.0/3.14);*" y "*float accel_ang_y=atan(ay/sqrt(pow(ax,2) + pow(az,2)))*(180.0/3.14);*", se realiza el cálculo del ángulo de inclinación en los ejes x e y, y estos valores se guardan en las variables "*accel_ang_x*" y "*accel_ang_y*" respectivamente.

La multiplicación de ambas expresiones por (180/3.14) tiene la función de convertir los radianes en grados, ya que la función "*atan()*" en C++ devuelve la arco tangente de un número en radianes. Las siguientes cuatro líneas se dedican a mostrar los ángulos por pantalla. Para concluir, la última línea "*delay(100);*", introduce un retraso de 10 milisegundos antes de proseguir con la siguiente iteración del bucle "*loop()*".

En la imagen que se presenta abajo, se aprecia el resultado obtenido al inclinar el módulo alrededor de 45º con respecto al eje Y:

Fig. 24 Resultado de inclinar el dispositivo 45º con respecto a Y

Tal y como se aprecia en la imagen superior, los ángulos obtenidos son próximos a 45º. Al emplear el acelerómetro para determinar los ángulos de inclinación con respecto a los ejes x e y, se ha constatado que los valores máximos alcanzan los 90º. Es relevante destacar que se ha observado una marcada influencia del ruido en el acelerómetro, es decir, cualquier vibración detectada por el sensor genera una rápida variación en los grados medidos.

4.2.1.3 CÁLCULO DEL ANGULO DE ROTACIÓN USANDO UNICAMENTE LOS VALORES DEL GIROSCOPIO

Como previamente se explicó en la sección dedicada al giroscopio, este sensor proporciona la velocidad angular, lo cual implica que para obtener el ángulo actual se debe realizar la integración de la velocidad angular, teniendo en cuenta también el ángulo inicial. Para ello se emplean las siguientes formulas:

$$\theta_x = \theta_{x_0} + \omega_x \cdot \Delta t$$
$$\theta_y = \theta_{y_0} + \omega_y \cdot \Delta t$$

En estas expresiones, θ_x , representa el ángulo que gira el eje x sobre sí mismo, de manera análoga a θ_y que se refiere al eje y. Δt representa el tiempo transcurrido entre dos mediciones consecutivas de la velocidad angular.
A continuación, se muestra como la velocidad angular se halla perpendicular al plano de rotación. Posteriormente, se presenta el código Arduino que calcula los ángulos de rotación en los planos x e y, basándose en las tres componentes del giroscopio.[55]

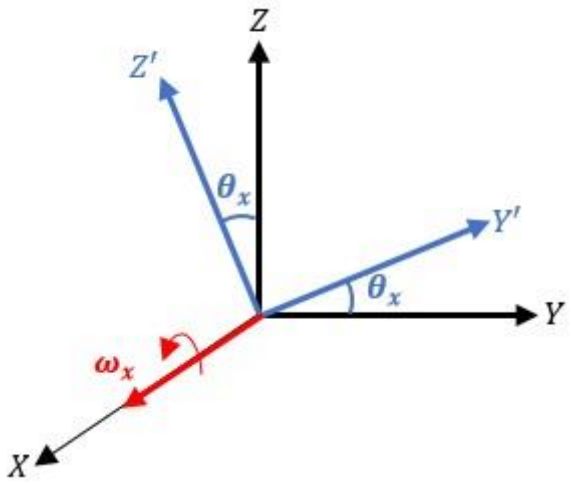

Fig. 25 Esquema planos de rotación y velocidad angular [55]

```
// Librerias I2C para controlar el mpu6050
// la libreria MPU6050.h necesita I2Cdev.h, I2Cdev.h necesita Wire.h
#include "I2Cdev.h"
#include "MPU6050.h"
#include "Wire.h"

// La dirección del MPU6050 puede ser 0x68 o 0x69, dependiendo
// del estado de AD0. Si no se especifica, 0x68 estará implicito
MPU6050 sensor;

// Valores RAW (sin procesar) del acelerometro y giroscopio en los ejes x,y,z
int gx, gy, gz;

long tiempo_prev, dt;
float girosc_ang_x, girosc_ang_y;
float girosc_ang_x_prev, girosc_ang_y_prev;
```

```
void setup() {
  Serial.begin(57600);    //Iniciando puerto serial
  Wire.begin();           //Iniciando I2C
  sensor.initialize();    //Iniciando el sensor

  if (sensor.testConnection()) Serial.println("Sensor iniciado correctamente");
  else Serial.println("Error al iniciar el sensor");
  tiempo_prev=millis();
}

void loop() {
  // Leer las velocidades angulares
  sensor.getRotation(&gx, &gy, &gz);

  //Calcular los angulos rotacion:
  dt = millis()-tiempo_prev;
  tiempo_prev=millis();
  girosc_ang_x = (gx/131)*dt/1000.0 + girosc_ang_x_prev;
  girosc_ang_y = (gy/131)*dt/1000.0 + girosc_ang_y_prev;
  girosc_ang_x_prev=girosc_ang_x;
  girosc_ang_y_prev=girosc_ang_y;

  //Mostrar los angulos separadas por un [tab]
  Serial.print("Rotacion en X:  ");
  Serial.print(girosc_ang_x);
  Serial.print("tRotacion en Y: ");
  Serial.println(girosc_ang_y);

  delay(100);
}
```

El código utiliza las mismas bibliotecas que ya se comentaron en el ejemplo anterior. Tras esto, se inicializa el sensor para que inicie la captura de los valores absolutos de cada componente. La variable "*tiempo_prev*", se inicializa con el valor actual del tiempo en milisegundos después de la correcta inicialización del sensor, y posteriormente, almacenará el tiempo de la medición anterior a la actual en las iteraciones subsiguientes. La variable "*dt*" almacena la diferencia entre el tiempo de la medida actual y el tiempo de la medición anterior (almacenado en "*tiempo_prev*").

En el bucle "*loop()*", se ejecuta la lógica principal del código, esto es, se obtienen los valores absolutos de cada componente del giroscopio y se almacena en las variables "*gx*", "*gy*" y "*gz*". El tiempo transcurrido entre las dos mediciones consecutivas se calcula restando el tiempo de la última medida al tiempo de la medida anterior. Para calcular los ángulos de rotación en los eje x e y, se emplean las siguientes líneas de código:
"*girosc_ang_x = (gx/131)*dt/1000.0 + girosc_ang_x_prev*;" y
"*girosc_ang_y = (gy/131)*dt/1000.0 + girosc_ang_y_prev;*". Las componentes "*gx*" y "*gy*" del giroscopio se dividen entre 131 para convertir las velocidades angulares en grados por segundo. La variable "*dt*", que almacena el tiempo entre la medida actual y la anterior, se divide entre 1000 para obtener el tiempo en milisegundos. A estos cálculos anteriores se le suma el ángulo obtenido previamente.

Las líneas siguientes tienen como objetivo mostrar por pantalla los ángulos de rotación en los ejes x e y.

Un problema que genera este código es el error de deriva o drift, previamente explicado en la sección de sensores. Este error surge de la integración de la velocidad angular y la suma del ángulo inicial, debido a mediciones temporales imprecisas y ruido en la lectura del sensor. Este error se acumula, aumentando en cada iteración.

La *Fig.26* ilustra el resultado obtenido al ejecutar el código del ejemplo. El dispositivo permanece en reposo sobre una superficie completamente plana. Sin embargo, se puede observar que a medida que avanzan las iteraciones, los ángulos de rotación en los ejes x e y aumentan progresivamente. Este fenómeno es atribuido al error de deriva. Este código no cumplió con las expectativas ni resultó ser útil para alcanzar el objetivo de obtener el ángulo de amplitud de movimiento para un ejercicio de rehabilitación.

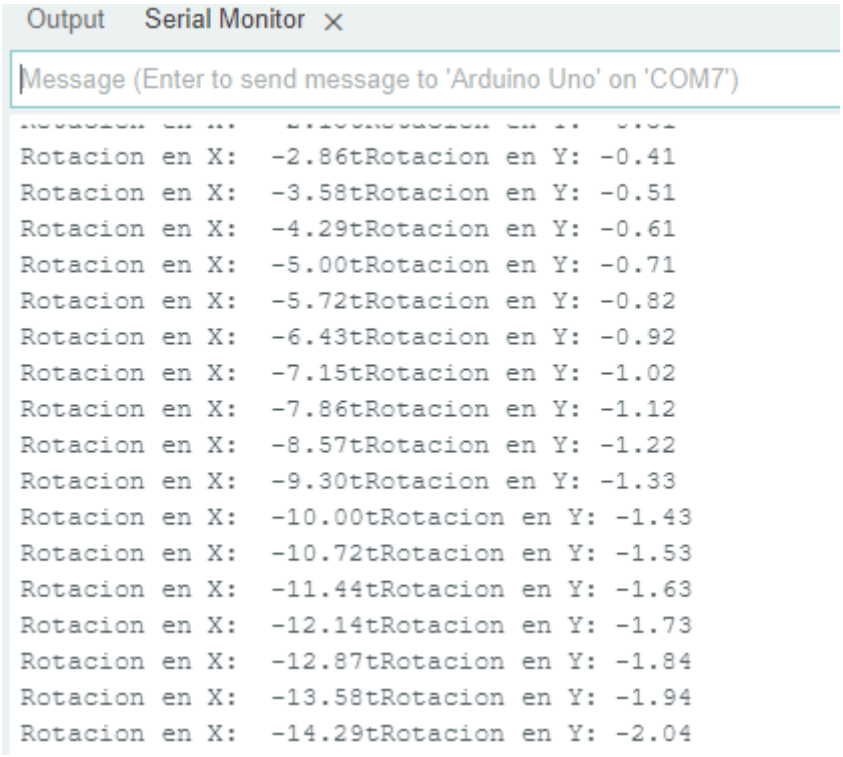

Fig. 26 Resultado de ejecutar el código estando el dispositivo en reposo

El drift suele eliminarse mediante el uso de filtros, siendo el filtro de Kalman el más eficiente, aunque su implementación es compleja debido a los requisitos de procesamiento que demanda. En su lugar, se emplea un filtro más simplificado, el filtro complementario, que fusiona las mediciones del acelerómetro y el giroscopio [55].

4.2.1.4 CÁLCULO DEL ANGULO FUSIONANDO ACELERÓMETRO Y GIROSCOPIO A TRAVÉS DEL FILTRO COMPLEMENTARIO

En este ejemplo, se llevará a cabo el cálculo del ángulo utilizando el filtro complementario, el cual fusiona los ángulos obtenidos a través del acelerómetro y el giroscopio. El filtro complementario destaca por su sencilla implementación y por ello es ampliamente empleado en la práctica.

El uso de un filtro complementario se debe a que cuando trabajamos únicamente con las medidas del acelerómetro, este resulta sensible a las aceleraciones provocadas por el movimiento del MPU o a fuerzas externas. No obstante, el ángulo no acumula errores en intervalos de tiempo prolongados. Sin embargo, cuando se trabaja únicamente con el giroscopio, este no es afectado por fuerzas externas, pero padece el inconveniente de la deriva, es decir, a lo largo del tiempo el error tiende a acumularse.

La ecuación empleada para el cálculo del ángulo mediante el filtro complementario es la siguiente [55]:

$$ángulo = 0.98 \cdot \left(ángulo + w_{giroscopio} \cdot dt\right) + 0.02 \cdot (ang_{acelerómetro})$$

A través de esta fórmula se aplica un filtro paso bajo al ángulo del acelerómetro, con el fin de atenuar las variaciones en la aceleración relacionadas con el movimiento del MPU. Por otro lado, el ángulo del giroscopio se pasa por un filtro paso alto. Los valores 0,98 y 0,02 son los más comúnmente empleados, aunque pueden modificarse, siempre deben sumar 1 [55].

A continuación, se presenta el código Arduino para implementar este ejemplo [55]:

```
// Librerias I2C para controlar el mpu6050
// la libreria MPU6050.h necesita I2Cdev.h, I2Cdev.h necesita Wire.h
#include "I2Cdev.h"
#include "MPU6050.h"
#include "Wire.h"

// La dirección del MPU6050 puede ser 0x68 o 0x69, dependiendo
// del estado de AD0. Si no se especifica, 0x68 estará implicito
MPU6050 sensor;

// Valores RAW (sin procesar) del acelerometro y giroscopio en los ejes x,y,z
int ax, ay, az;
int gx, gy, gz;
long tiempo_prev;
float dt;
float ang_x, ang_y;
float ang_x_prev, ang_y_prev;

void setup() {
  Serial.begin(57600);    //Iniciando puerto serial
  Wire.begin();           //Iniciando I2C
  sensor.initialize();    //Iniciando el sensor

  if (sensor.testConnection()) Serial.println("Sensor iniciado correctamente");
  else Serial.println("Error al iniciar el sensor");
}
void loop() {
```

```
  // Leer las aceleraciones y velocidades angulares
  sensor.getAcceleration(&ax, &ay, &az);
  sensor.getRotation(&gx, &gy, &gz);

  dt = (millis()-tiempo_prev)/1000.0;
  tiempo_prev=millis();

  //Calcular los ángulos con acelerometro
  float accel_ang_x=atan(ay/sqrt(pow(ax,2) + pow(az,2)))*(180.0/3.14);
  float accel_ang_y=atan(-ax/sqrt(pow(ay,2) + pow(az,2)))*(180.0/3.14);

  //Calcular angulo de rotación con giroscopio y filtro complemento
  ang_x = 0.98*(ang_x_prev+(gx/131)*dt) + 0.02*accel_ang_x;
  ang_y = 0.98*(ang_y_prev+(gy/131)*dt) + 0.02*accel_ang_y;
  ang_x_prev=ang_x;
  ang_y_prev=ang_y;

  //Mostrar los angulos separadas por un [tab]
  Serial.print("Rotacion en X:  ");
  Serial.print(ang_x);
  Serial.print("Rotacion en Y: ");
  Serial.println(ang_y);

  delay(10);
}
```

Tal como se abordó en los dos ejemplos anteriores, se inicia añadiendo las librerías esenciales para el control del MPU6050: I2Cdev.h, MPU6050.h, Wire.h. Acto seguido, se definen todas las variables necesarias en el código y se inician los sensores. Dentro del bucle “loop()”, se lleva a cabo el conjunto de cálculos para obtener los ángulos. Primero, se almacenan en las variables correspondientes los valores absolutos devueltos por los dos sensores. Como en el ejemplo anterior, se recurre a las variables “dt”, “tiempo_prev”, y a la función “millis()” para calcular el intervalo de tiempo entre dos lecturas consecutivas del giroscopio. Los procedimientos para calcular los ángulos en x e y del acelerómetro ya fueron explicados en el primer ejemplo.

Luego, se emplea la fórmula del filtro complementario para determinar el ángulo, fusionando los valores del acelerómetro y giroscopio:
“ang_x = 0.98*(ang_x_prev+(gx/131)*dt) + 0.02*accel_ang_x;” y
“ang_y = 0.98*(ang_y_prev+(gy/131)*dt) + 0.02*accel_ang_y;”

Los valores absolutos del giroscopio “gx” y “gy” se dividen entre 131, tal y como se especifica en el datasheet del MPU, tal como se explicó en el segundo ejemplo. Las variables “ang_x_prev “, y “ang_y_prev” se emplean para almacenar el ángulo actual, ya que en la siguiente iteración del bucle pasará a ser el ángulo previo.

Posteriormente las líneas siguientes se emplean para mostrar por pantalla los ángulos en el eje x y en el eje y:

“Serial.print("Rotacion en X: "); Serial.print(ang_x); Serial.print("Rotacion en Y: "); Serial.println(ang_y);”

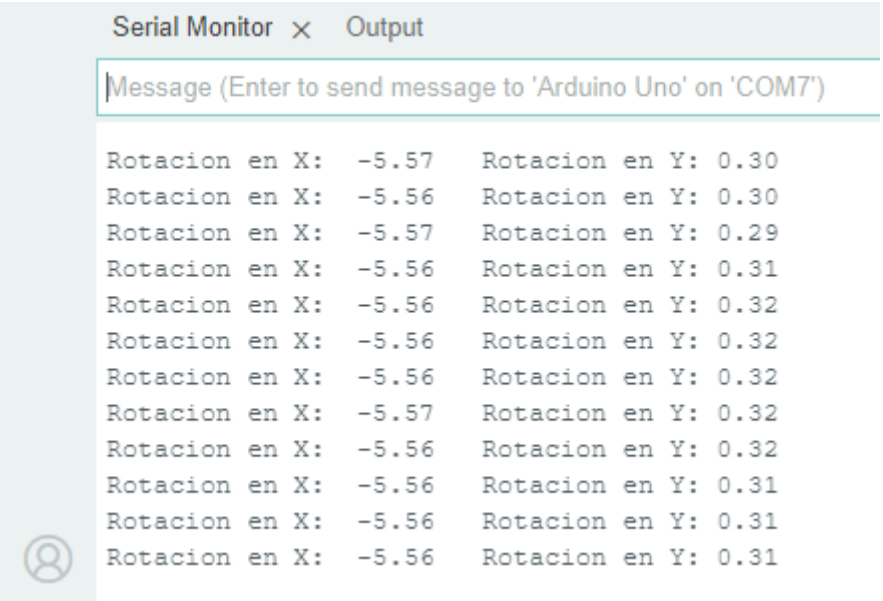

Fig. 27 Resultado de la ejecución con el dispositivo en reposo

La *Fig.27* muestra la ejecución del código que emplea el filtro complementario. En esta ejecución, el dispositivo permanece en reposo sobre una superficie plana. Se observa que aparece un pequeño offset de $-5{,}56º$ en el ángulo de rotación del eje x y de $0{,}31º$ en el ángulo de rotación del eje y. Además, se nota que, al mover el dispositivo a una posición determinada, se requieren varios segundos para que el ángulo alcance la magnitud correspondiente a dicha posición.
Este comportamiento no se ajusta de manera óptima a nuestro objetivo, que busca una respuesta ágil en la variación del ángulo durante el movimiento. Dado que el objetivo de la aplicación involucra medir tanto la amplitud del movimiento como el tiempo necesario para realizarlo, la espera de varios segundos hasta que se estabiliza el ángulo después de un movimiento afectaría a la precisión temporal de la medición real del ejercicio.

4.2.1.5 CONCLUSIONES TRAS EL ESTUDIO DE LOS EJEMPLOS

Tras analizar los tres códigos de Arduino que podrían resultarnos útiles para la medición del ángulo de amplitud del movimiento, se llegó a la conclusión de que el primero, el cual se basa exclusivamente en el acelerómetro, era el más adecuado para alcanzar el objetivo la aplicación. A pesar de que presentaba ruido en las mediciones, su respuesta era más rápida y el ángulo se modificaba con mayor velocidad.
Después de comprender la lógica subyacente en el cálculo de los ángulos mediante los sensores, surgió la idea de explorar las capacidades de los teléfonos inteligentes actuales y el desarrollo de aplicaciones móviles, particularmente en el entorno de Flutter. Se consideró la posibilidad de desarrollar una aplicación que hiciera uso de los sensores inerciales del dispositivo móvil para obtener la medición del ángulo del movimiento del ejercicio, utilizando únicamente el teléfono del paciente, evitando así la adquisición de dispositivos a mayores por parte del paciente.
Siguiendo este planteamiento, se avanzó hacia una segunda y última versión, que suponía el desarrollo de una aplicación móvil a través de Flutter. Se aplicó la misma lógica de cálculo del ángulo mediante sensores que se había empleado en Arduino. En este proceso, se llegó también a la conclusión de que la mejor opción, la más coherente con nuestro objetivo, radicaba en emplear únicamente el acelerómetro del dispositivo móvil.

4.3 APLICACIÓN FINAL

Como se mencionó anteriormente, la elección final fue desarrollar una aplicación móvil compatible tanto con Android como con iOS, que aprovechara directamente los sensores del teléfono móvil.

Se decidió utilizar el entorno de desarrollo Flutter, como se explicó en secciones anteriores, debido a su capacidad para crear aplicaciones visuales que funcionasen en ambos sistemas operativos. Otra decisión importante fue emplear únicamente el acelerómetro, a pesar del problema del ruido que afecta a este sensor. Se eligió esta opción porque permitía obtener una respuesta más rápida. Si bien se realizaron pruebas fusionando el acelerómetro y el giroscopio con un filtro complementario, esto no solo aumentaba el margen de error, sino que también requería más tiempo de procesamiento para calcular el ángulo correspondiente a la amplitud del movimiento. Este tiempo adicional tenía un impacto en la ejecución de los ejercicios, ya que uno de los aspectos que se miden es el tiempo empleado en cada repetición y en la realización completa del ejercicio.

El objetivo principal era crear una aplicación intuitiva y fácil de usar, ya que está dirigida a usuarios de todas las edades. Se consideró que, en el caso de las personas mayores de 65 años, que pueden no estar tan familiarizadas con el uso de dispositivos móviles como los usuarios más jóvenes, era esencial contar con un diseño sencillo, claro y fácil de entender.

La aplicación se compone de varias pantallas, y a continuación, se describirán sus funciones y su interfaz antes de entrar en los detalles técnicos del código.

PANTALLA DE BIENVENIDA:

La pantalla de bienvenida mostrada en la *Fig.28*, presenta una animación en la que el nombre de la aplicación, “SensorRehab”, se muestra y aumenta gradualmente de tamaño. La animación se inicia al abrirse la aplicación, y después de 5 segundos o al tocar la pantalla, mostrará automáticamente la pantalla de inicio.

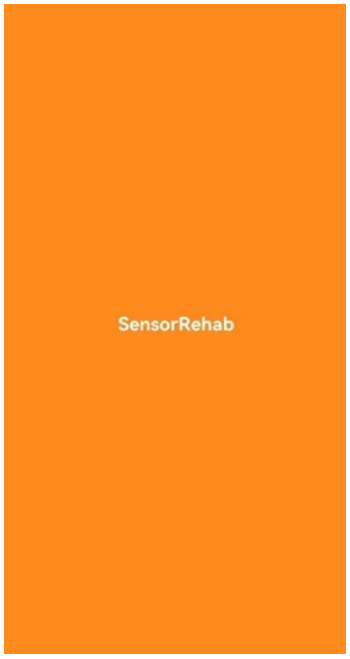

Fig. 28 Pantalla de bienvenida

PANTALLA DE INICIO:

En la parte superior de la pantalla de inicio, *Fig.29*, se encuentra una barra (appBar), que muestra el nombre de la aplicación. En el centro de la pantalla, se muestra un mensaje de bienvenida que dice "Bienvenido a sensorRehab" Y en la parte inferior, aparece una barra de navegación con dos iconos. El primero es un ícono de una casa, que aparece seleccionado en color naranja debido a que actualmente la aplicación está en la pantalla de inicio. El segundo ícono es una mancuerna, y al pulsarlo se accede al menú de ejercicios, ahí la mancuerna estará de color naranja.

Fig. 29 Captura de la pantalla inicio de la aplicación

PANTALLA MENÚ DE EJERCICIOS:

La pantalla menú de ejercicios, se trata de un menú deslizante que presenta tres ejercicios distintos. En la *Fig.30* de abajo, puedes observar dos capturas del menú de ejercicios. Cada ejercicio se encuentra dentro de un contenedor individual que incluye su nombre, una imagen representativa y un botón para iniciar la actividad. Al seleccionar el botón "Comenzar ejercicio," se dirigirá automáticamente a la página correspondiente del ejercicio seleccionado.

Fig. 30 Captura del menú ejercicios, se muestran los 3 ejercicios disponibles.

PANTALLA DEL EJERCICIO SELECCIONADO:

La pantalla de ejercicio es el lugar donde se efectúa la medición del ángulo de amplitud del movimiento durante la ejecución del ejercicio especificado. Al acceder a esta pantalla, la interfaz inicial se muestra en la *Fig.31*.

Fig. 31 Interfaz inicial al acceder el ejercicio de flexión de muñeca.

En la parte superior, se aprecia una barra de navegación o appBar que muestra el nombre del ejercicio, un ícono de flecha que permite regresar a la pantalla anterior (es decir, al menú de ejercicios), y un botón de información representado por un círculo blanco con una "i" en su interior. Al pulsar este botón, se despliega un cuadro de dialogo, como se muestra en la *Fig.32* que detalla todos los pasos necesarios para ejecutar el ejercicio correctamente. Dentro de este desplegable, también se incluye un botón para cerrar dicho menú.

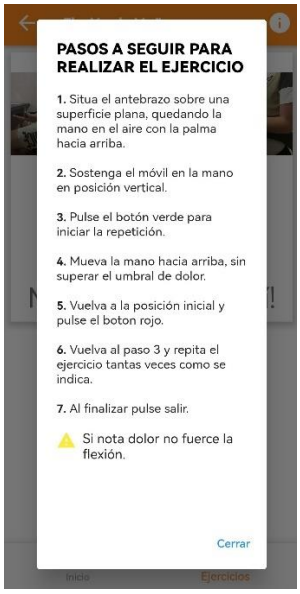

Fig. 32 Cuadro de dialogo con los pasos a seguir para realizar el ejercicio

Justo debajo de la appBar, encontramos un contenedor que alberga la imagen del ejercicio, el objetivo de repeticiones (en este caso, fijado en 10 repeticiones), el contador de repeticiones y el ángulo. Justo debajo de este contenedor, se ubica un botón circular con fondo verde y un icono de "play", se muestra en la *Fig.31*. situándose en la posición inicial del ejercicio se pulsa el botón verde y se comienza a realizar el movimiento, la interfaz se modifica a la mostrada en la *Fig.33*.

Fig. 33 Ejercicio iniciado.

La repetición finaliza al volver a la posición inicial pulsando el botón rojo. Al pulsar este botón cambia a verde, aparece el número de repetición realizada, el ángulo alcanzado y un mensaje para motivar al usuario, *Fig.34*. Todo este proceso se repite hasta que se alcanza el objetivo de las 10 repeticiones.

Fig. 34 Interfaz tras finalizar la repetición.

Una vez que se alcanza el objetivo, *Fig.35*, se muestra la información de la última repetición, la media del ángulo alcanzado y un mensaje de "¡Enhorabuena, has alcanzado tu objetivo!". Además, aparece un botón con el texto "*salir*", al pulsarlo se regresa al menú de ejercicios y automáticamente y de forma transparente al usuario se envía un correo al médico con un archivo adjunto que contiene la información del ejercicio donde se incluye la fecha y hora de inicio del ejercicio, el número de repeticiones, el ángulo alcanzado en cada repetición, el tiempo empleado y la media del ángulo alcanzado en el ejercicio.

Fig. 35 Interfaz de la pantalla ejercicio tras conseguir el objetivo.

La dinámica de la realización del ejercicio es la misma para los tres tipos diferentes de ejercicios.

4.4 CÓDIGO

Esta sección profundiza en los aspectos técnicos del código de la aplicación y su desarrollo en general y en el Anexo 1 se incluyen los códigos al completo.

La aplicación "SensorRehab" se ha construido utilizando el framework Flutter y se ha escrito en el lenguaje de programación Dart. El desarrollo del código se llevó a cabo en el entorno de desarrollo Visual Studio Code.

En cuanto al dispositivo móvil utilizado para el desarrollo de la aplicación y para registrar los ejercicios en las pruebas realizadas, se trata de un Huawei P20. Este dispositivo cuenta con un conjunto de sensores que incluyen el acelerómetro y el giroscopio, entre otros. Estos sensores son capaces de medir en tres ejes (x, y, z) y están alojados en un módulo inercial. El modelo de este módulo es el LSM6DSM, diseñado por la empresa iNEMO.

Para facilitar la explicación de los aspectos técnicos del código se irán explicando los diferentes archivos ".dart" , archivos que albergan el código de la aplicación.

Main.dart:

Es un archivo fundamental en la aplicación. Su función principal es iniciar la apliación y definir la estructura inicial de la misma.

La función *"*`main()`*"*, es la función principal que se ejecuta cuando la aplicación se inicia. Se encarga de llamar a *"*`runApp`*"*, que toma como argumento una instancia de*"* `MyApp`*"*.

"`MyApp`" es una clase que extiende "`StatelessWidget`", lo que indica que la aplicación no cambia de estado después de su construcción inicial.
El contructor "`MyApp`"recibe un parámetro opcional *"*`key`*"*. El "`const`" en "`MyApp`" indica que el constructor no puede modificar sus propiedades después de su creación.

El método *"*`build(BuildContext context)`" se llama automáticamente cuando se crea una instancia de "`MyApp`". Devuelve un objeto "`MaterialApp`", componente fundamental en Flutter para la creación de aplicaciones con Material Design. Dentro de este objeto se configuran dos propiedades:

- *"*`debugShowCheckedModeBanner: false`*"* Con esta propiedad en false se desactiva el banner de depuración que aparece en la esquina superior derecha de la aplicación cuando se ejecuta en modo depuración.
- *"*`home: SplashScreen()`*"*: Se especifica la pantalla inicial que se muestra cuando se abre la aplicación, en este caso es la pantalla "`SplashScreen()`*"*. La pantalla que en el apartado anterior se ha definido como pantalla de bienvenida (Fig.35).

Splash_screen.dart:

Este código Flutter representa la pantalla de inicio de la aplicación que incluye una animación de introducción antes de redirigir al usuario a la pantalla principal de la aplicación, que en este caso es la pantalla *"*`InicioPage()`*"*

Al principio se importan los paquetes necesarios, en este caso "*material.dart*" que proporciona widgets y herramientas para la construcción de la interfaz de usuario. Y "*inicio.dart*" que importa la clase *"*`InicioPage()`*"*

La clase *"*`SplashScreen()`*"*representa la pantalla de bienvenida de la aplicación.

La función *"*`State<SplashScreen> createState() => _SplashScreenState()`*"*,crea y devuelve una instancia de *"*`_SplashScreenState()`*"* que es una subclase de *"*`State<SplashScreen>`*"*

Esta subclase se encargará de manejar el estado y la lógica de la pantalla de inicio.

El método *"*`initState()`*"* se llama cuando se crea el estado del widget. Aquí se inicializa la animación y se configura el controlador de animación "`_animationController`" con una duración de 5 segundos.

La animación "`_animation`" se define utilizando un "`Tween`" que va desde 0.0 hasta 1.0. Significa que la animación cambiará gradualmente el tamaño del widget desde 0% hasta 100% en los 5 segundos definidos.

Con *"*`_animationController.forward();`*"* se inicia la animación hacia adelante.

Se agrega un oyente de estado *"*`_animationController`*",* para detectar cuando finaliza la animación. Cuando la animación finaliza se emplea *"Navigator.pushReplacement()"* para redirigir al usuario a la pantalla siguiente, que es *"*`InicioPage()`*"*

El método *"*`dispose()`*"* es llamado cuando el widget se elimina. De esta forma se asegura de liberar los recursos asociados al controlador de animación.

El método *"*`build(BuildContext context)`" se llama cuando el widget debe construir su representación visual. En este caso se construye un *"*`Scaffold`*"* con fondo naranja.

El widget principal es "`ScaleTransition`", lo aplica la animación al contenido que contiene, que es un widget de texto "*SensorRehab*". Este texto aumenta de tamaño gradualmente a medida que avanza la animación.

Se emplea *"*`GestureDetector`*"* que permite que el usuario toque la pantalla y avance manualmente a la siguiente pantalla si lo desea.

Inicio.dart:

Este código define la estructura de la página de inicio de una aplicación y crea una barra de navegación inferior que permite al usuario cambiar entre dos destinos principales: la página de inicio y la página de ejercicios. A continuación, se detalla el funcionamiento de este código.

La clase "`InicioPage()`" representa la página de inicio de la aplicación, tiene una propiedad "currentPage" para realizar un seguimiento de la página actual que se muestra en la barra de navegación. La lista "`pages`" contiene las páginas disponibles en la aplicación ("`HomePage()`" y "`ExercisePage()`").

La clase "`_InicioPageState`", se utiliza para gestionar el estado de la página de inicio. En el método "`build()`" se crea un "`Scaffold`" que contiene la barra de navegación y el contenido de la página actual. El contenido de la página actual se determina según el valor de *"*`currentPage`*"*.

La clase "`NavigationBar()`" es un widget independiente que representa la barra de navegación inferior. Tiene varias propiedades como "`final List<NavigationDestination> destinations`" (una lista de opciones de navegación), "`selectedIndex`" (el índice de la opción seleccionada) y "`onDestinationSelected`" (función que se llama cuando se selecciona una opción).

En el método "`build()`" de esta clase, se utiliza "`bottomNavigationBar`" para crear la barra de navegación inferior. Se configura con los elementos de "`destinations`" y se le pasa el índice seleccionado ("`selectedIndex`"). Cuando el usuario selecciona un elemento de la barra de navegación se llama a "`onDestinationSelected`".

La clase "`class NavigationDestination`" es una clase que define las propiedades de una opción de navegación, como un icono y una etiqueta.

Home.dart:

Este código junto con el anterior se encarga de construir la página de inicio.

La clase "`HomePage()`" representa una página cuyo contenido no cambia una vez que se construye, de ahí que sea "`StatelessWidget`".

El método "`build(BuildContext context)`" se llama cuando se construye la página de inicio y devuelve el widget que representa el contenido de esta página.

El contenido de la página se coloca dentro de un "`Scaffold`", que proporciona una estructura básica para la pantalla incluyendo la barra superior de navegación ("`appBar`") y el cuerpo de la página ("`body`").

"`AppBar()`" define una barra de aplicación que contiene el nombre de la aplicación y un fondo naranja ("`const Color(0xFFFF8B1F)`"). También se especifica el estilo del texto, incluyendo el tipo de letra, el tamaño de la fuente y el color de la letra.

El cuerpo de la aplicación se define como un "Center" que contiene un "`Column()`", que se encarga de centrar verticalmente los elementos en la pantalla.

Se crea un widget de texto que muestra "*Bienvenidos a SensorRehab*" incluyendo saltos de línea con "\n". El texto esta alineado al centro con "`TextAlign.center`" y se establece el tamaño de fuente a 30 y el estilo de la letra, negrita.

Con "`SizedBox`" se agrega un espacio vertical con una altura de 30.0 pixeles entre el texto y otros elementos de la columna.

menu_ejercicios.dart:

Este código representa la construcción de la página ejercicios. Esta página muestra una lista de ejercicios disponibles, cada uno con su nombre, una imagen ilustrativa y un botón para comenzar el ejercicio. A continuación, se explican los aspectos técnicos del código.

La clase "`ExercisePage`" representa la página de ejercicios. El contenido de la página se coloca dentro de un "`Scaffold`" que proporciona una estructura básica para la pantalla que incluye la appBar y el body.

La appBar contiene el texto "*EJERCICIOS*" sobre un fondo naranja.

El cuerpo de la página (body) se coloca dentro de un "`SingleChildScrollView()`", lo que permite desplazar verticalmente la pantalla si el contenido es demasiado grande y no cabe en la pantalla. El contenido principal es un "`Column()`" que contiene varias opciones de ejercicio, cada una de estas se crea llamando a "`_buildExercise`".

La función "`_buildExercise()`":

Recibe varios parámetros: "`name`" (nombre del ejercicio), *"*`image`*"* (ruta de la imagen del ejercicio correspondiente), *"*`type`*"*(tipo de ejercicio) y *"*`context`*"*.

Cada opción de ejercicio se representa en un contenedor que incluye un titulo con el nombre del ejercicio, una imagen del ejercicio y un botón que permite al usuario comenzar el ejercicio. A todos estos elementos se les aplica un estilo como el tamaño de la fuente, el estilo de fuente negrita, el tipo de letra y los colores consiguiendo así una apariencia atractiva y coherente de la página.

Se define una enumeración *"*`enum ExerciseType`*"* para representar las diferentes opciones de ejercicios. Cada opción esta asociada a un valor especifico, *"*`codo`*"*, "`munecaExtension`" y *"*`munecaFlexion`*"*.

Ejercicio_flexion_muneca.dart:

Únicamente se abordará la explicación del código asociado a la página de ejercicio de flexo-extensión de codo. No se proporcionará una explicación detallada para los archivos "*ejercicio_extension_muneca.dart"* y "*ejercicio_codo.dart*", ya que comparten la misma lógica que el archivo "*ejercicio_flexion_muneca.dart*".

La clase *"*`FlexionMunecaPage`*"* es un widget de estado, esto quiere decir que su contenido y estado pueden cambiar durante la vida útil de la página. Esta clase representa la página del ejercicio de flexión de la muñeca. El constructor *"*`const FlexionMunecaPage({super.key});`*"* se utiliza para crear una instancia de la clase.

El método "`initState`" se llama automáticamente cuando se crea la instancia del widget. En este caso se emplea para llamar a *"*`_resetFile()`*"*.

El método *"*`_resetFile()`*"*, verifica si existe el archivo y, si es así, borra su contenido. Si no existe crea uno en la ruta especificada. Este archivo será el que contenga la información del paciente sobre el ejercicio.

El método "`_start()`", llamado para comenzar a recibir información del acelerómetro. Pasa los valores absolutos *x*,*y* y *z* del acelerómetro a la función que calcula el ángulo y añade a una lista los ángulos calculados. También inicia el reloj.

El método *"*`_finRep()`*"*, se llama cuando se completa una repetición del ejercicio. Obtiene el ángulo máximo alcanzado durante el movimiento y pasa el numero de repetición, el ángulo máximo y el tiempo empleado en la repetición para escribirlo en el archivo.

El método *"*`_calcularAngulo()`*"*, calcula el ángulo de inclinación basado en los valores del acelerómetro. Se agrega un offset de 5 grados.

El método *"*`_writeDataToFile()`*"*, escribir los datos de una repetición en el archivo de registro. También registra la fecha y hora de inicio del ejercicio cuando se ejecuta por primera vez. Cuando se completan todas las repeticiones, calcula la media de los ángulos y el tiempo total empleado en el ejercicio.

El método *"*`sendEmailWithAttachment()`*"*, para el envío del correo electrónico al médico con el archivo adjunto. Utiliza una configuración de servidor SMTP.

El método "`dispose()`", para cancelar todas las suscripciones a eventos de sensores. Es llamado cuando se sale de la página.

El método "`_finEjercicio()`", llamado cuando se finaliza el ejercicio completo. Regresa a la página anterior (al menú de ejercicios) y llama a la función *"*`sendEmailWithAttachment()`*"* para enviar el correo.

El método "`build`", construye la interfaz de usuario de la página del ejercicio de flexión de muñeca.

Pubspec.yaml:

En este archivo se han incluido en el apartado *"dependencias:"*, las siguientes dependencias, necesarias para el correcto funcionamiento de la aplicación.

`sensors_plus: ^2.0.5`: para poder trabajar con el acelerómetro del móvil.
`intl: ^0.18.1`: para el formato de la fecha incluido en el archivo se necesitaba
`mailer: ^6.0.1`: biblioteca necesaria para la redacción y envíos de correo, permite incluir archivos adjuntos.
`path_provider: ^2.0.15`: para la búsqueda de rutas de uso común en el sistema de archivos

```
66  # To add assets to your application, add an assets section, like this:
67  assets:
68    - images/
```

Fig. 36 Captura de código donde se incluye la carpeta images

En la captura de código, *Fig.36,* se muestra el fragmento de código donde se incluye el directorio "*images*", carpeta donde se ubican las imágenes empleadas en la aplicación. De esta forma la aplicación flutter tendrá acceso a esas imágenes.

5. PROTOCOLO DE PRUEBAS Y RESULTADOS

5.1 PROTOCOLO PARA LAS PRUEBAS DE REHABILITACIÓN

Se presenta en este apartado un protocolo de pruebas del sistema de rehabilitación presentado en este TFG para pacientes con lesiones articulares y/o musculares que afectan al normal movimiento del brazo y la mano.

1.- Objetivo.

Evaluar la eficacia y la aceptabilidad de un sistema de rehabilitación basado en el registro de las repeticiones de un ejercicio determinado realizado por el paciente, así como la validez de las repeticiones en cuanto a cantidad y amplitud de los movimientos.

2.- Población de la muestra.

En estas pruebas iniciales del sistema aún no emplearemos pacientes, sino controles sanos. Se seleccionarán 7 personas sin lesiones articulares en el brazo ni muñeca y/o musculares y 1 persona en recuperación tras operación en la muñeca por síndrome del túnel carpiano. Los participantes deberán ser mayores de edad, no presentar otras patologías que interfieran con el ejercicio, y dar su consentimiento informado para participar en el estudio.

Las características de interés de los participantes se muestran en la tabla siguiente:

Identificador	Edad	Sexo	Facilidad Técnica	Angulo alcanzado en flexo-extensión codo	Angulo alcanzado en flexión de muñeca	Ángulo alcanzado en extensión de muñeca
Control 1	29	Masculino	si	90º	87º	77º
Control 2	19	Masculino	si	91º	86º	84º
Paciente 1	59	Masculino	si	89º	86º	74º
Control 3	24	Femenino	si	91º	88º	84º
Control 4	60	Femenino	si	92º	84º	77º
Control 5	24	Femenino	si	92º	88º	83º
Control 6	63	Masculino	no	91º	85º	79º
Control 7	25	Femenino	si	90º	85º	80º

Tabla 3 Características de interés de los participantes

El apartado de “Facilidad Técnica” se refiere a la familiaridad con el uso de dispositivos como los teléfonos móviles, ya que el sistema se implementa precisamente sobre un móvil y el usuario debe interaccionar con él, se indicará con un “si”, cuando el usuario este familiarizado con el uso del móvil y con un “no” en caso contrario. En cuanto a los apartados de los ángulos, representa la media del ángulo alcanzado en cada ejercicio.

En la *Tabla 3* se muestran los 7 controles, junto con un paciente resaltado en color rojo, que se trata del participante en proceso de rehabilitación.

Todos los sujetos serán reclutados voluntariamente y sus datos personales no serán expuestos. Por ello se elige un identificador de paciente en lugar de un nombre.

3.- Periodo de información y entrenamiento inicial.

Los sujetos del ensayo recibirán una explicación inicial sobre el objetivo del prototipo y sobre la naturaleza de las pruebas a realizar. Posteriormente se realizaría una prueba guiada por el técnico responsable de la prueba (en este caso la autora de este TFG, pero en su uso real sería el profesional rehabilitador quien se encargaría de esta labor). Esta prueba no se registraría y serviría únicamente para familiarizar al paciente con el uso del sistema.

Así pues, se explicará a los participantes en qué consiste el sistema de rehabilitación, cómo funciona y qué beneficios se esperan obtener con su uso. Se les mostrará el dispositivo y se les enseñará cómo sujetarlo correctamente con la mano. Se les indicará que el sistema registrará las repeticiones de cada ejercicio que realicen, junto con el ángulo alcanzado y el tiempo empleado en cada una de ellas. Se les informará que el sistema emitirá un mensaje cada vez que se realice una repetición, donde se le indicará el número de repetición que ha realizado, y el ángulo alcanzado. Además, si el usuario, está suficientemente familiarizado con el uso del móvil podrá ver el archivo enviado al médico, con toda la información, si accede en su correo al aparto de "enviados" en la parte de correos enviados. Se resolverán las posibles dudas o preguntas que tengan los participantes.

4.- Realización guiada de 3 tipos de pruebas empleando el sistema de rehabilitación.

Se realizarán tres tipos de pruebas con el sistema de rehabilitación: una prueba de flexión-extensión del codo, una prueba de flexión de la muñeca y otra prueba de extensión de la muñeca. Cada prueba tendrá un número de repeticiones válidas en este caso un total de 10 y se realizará bajo la supervisión de un fisioterapeuta (o un ingeniero como en este caso).

A. Prueba de flexión-extensión del codo: El participante se colocará sentado con el brazo apoyado sobre una mesa (sobre un elemento blando, si así lo desea), con el codo flexionado a 90 grados respecto al húmero y el antebrazo en posición neutral. El dispositivo se sostendrá de forma conveniente con la mano (ver *Fig.37*). El participante deberá realizar movimientos de flexión y extensión del codo, intentando alcanzar el rango requerido (colocar el antebrazo totalmente vertical sin forzar ni sentir dolor), como se muestra en la *Fig.38*. El sistema comenzará a registrar la repetición cuando se pulse el botón verde en la posición inicial del ejercicio. La repetición finalizará cuando el usuario pulse el botón rojo. Tras esto el paciente podrá ver en la pantalla el número de repetición realizada y el ángulo que ha alcanzado. Al finalizar el ejercicio se pulsará el botón "salir" y al médico automáticamente se le enviará un correo con un archivo adjunto que contiene toda la información. Cada paciente realizará esta prueba 3 veces con un descanso de 10 segundos entre cada prueba.

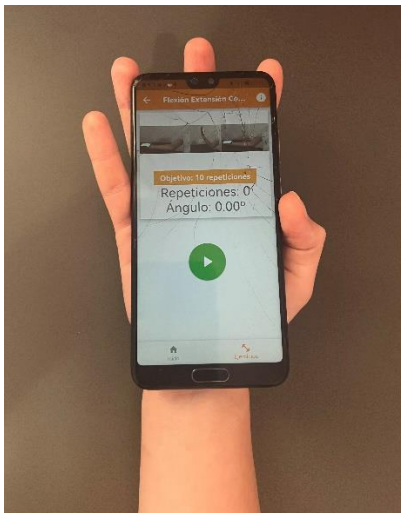

Fig. 37 Sujeción correcta del móvil.

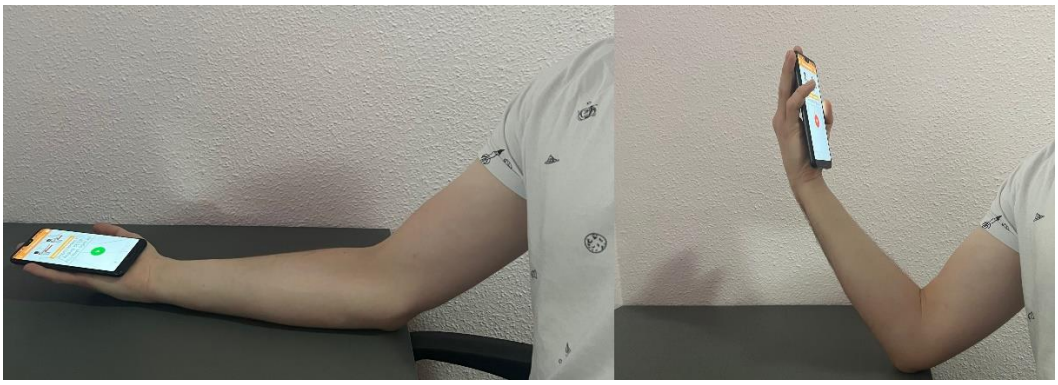

Fig. 38 Posición inicial del ejercicio de flexo-extensión codo.

A. Prueba de flexión de la muñeca: El participante se colocará sentado con el brazo apoyado sobre una mesa o un respaldo, con el codo flexionado a 90 grados y el antebrazo en posición neutral, con la mano en el aire. El dispositivo se sujetará con la mano de una forma conveniente (*Fig.37*). El participante deberá realizar movimientos de flexión de la muñeca, intentando alcanzar el máximo rango posible sin forzar ni sentir dolor (*Fig.39*). El sistema comenzará a registrar la repetición cuando se pulse el botón verde en la posición inicial del ejercicio. La repetición finalizará cuando el usuario pulse el botón rojo. Tras esto el participante podrá ver en la pantalla el número de repetición realizada y el ángulo que ha alcanzado. Al finalizar el ejercicio se pulsará el botón “salir” y al médico automáticamente se le enviará un correo con un archivo adjunto que contiene toda la información. Cada paciente realizará esta prueba 3 veces con un descanso de 10 segundos entre cada prueba.

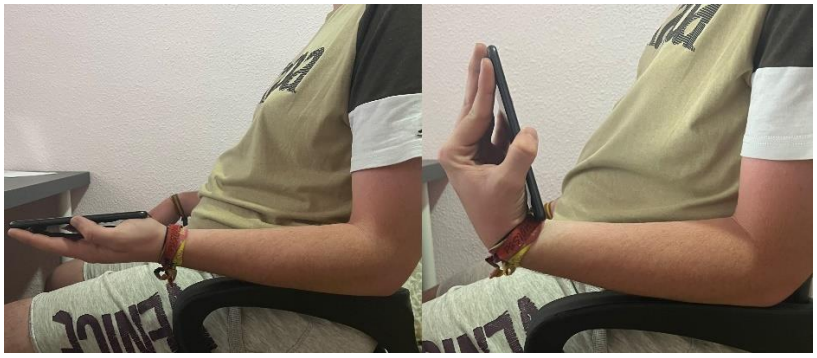

Fig. 39 Izda.: posición inicial. Dcha.: Rango máximo de flexión

B. Prueba de extensión de la muñeca: El participante se colocará sentado con el brazo apoyado sobre una mesa o un respaldo, con el codo flexionado a 90 grados y el antebrazo en posición neutral, con la mano en el aire. El dispositivo se sujetará con la mano de una forma conveniente (*Fig.37*). El participante deberá realizar movimientos de extensión de la muñeca, intentando alcanzar el máximo rango posible sin forzar ni sentir dolor (*Fig.39*). El sistema comenzará a registrar la repetición cuando se pulse el botón verde en la posición inicial del ejercicio. La repetición finalizará cuando el usuario pulse el botón rojo. Tras esto el participante podrá ver en la pantalla el número de repetición realizada y el ángulo que ha alcanzado. Al finalizar el ejercicio se pulsará el botón “salir” y al médico automáticamente se le enviará un correo con un archivo adjunto que contiene toda la información. Cada paciente realizará esta prueba 3 veces con un descanso de 10 segundos entre cada prueba.

Fig. 40 Izda.: posición inicial. Dcha.: Rango máximo de extensión.

5.2 RESULTADOS TRAS LA REALIZACIÓN DE LAS PRUEBAS

Las pruebas tenían como objetivo verificar la validez y veracidad de el prototipo de aplicación de ayuda a la rehabilitación propuesto en este proyecto.

Es importante indicar que en el estudio se incluyeron un total de 8 personas, de las cuales 7 no presentaban ningún tipo de lesión ni dolencia en codo ni muñeca, mientras que una de ellas estaba en proceso de rehabilitación tras una operación para corregir el síndrome del túnel carpiano.

Comparando los resultados de los 8 participantes incluidos en la *Tabla 3,* se puede ver que se han obtenido ángulos algo similares. Esto sugiere que la aplicación puede ser efectiva en la medición de ángulos o que los participantes comprendieron y aplicaron correctamente las instrucciones proporcionadas.

Además, es alentador que la mayoría de los participantes encontraron la aplicación intuitiva y fácil de usar después de recibir una guía de uso de esta. Esto indica que la interfaz de la aplicación podría ser amigable para los usuarios y que las instrucciones fueron efectivas para ayudar a los participantes a familiarizarse con la aplicación.

También algunos sugirieron que en los ejercicios en los que se trabaja la muñeca presentaban un cierto cuidado con el agarre del móvil, debido a la posibilidad de que se les deslizara de la mano. Este problema podría resolverse sujetándolo con una goma.

En los dos ejercicios donde se trabaja la extremidad de la muñeca, *Fig.41,* el ángulo máximo alcanzado está en torno a los 85º- 90º.

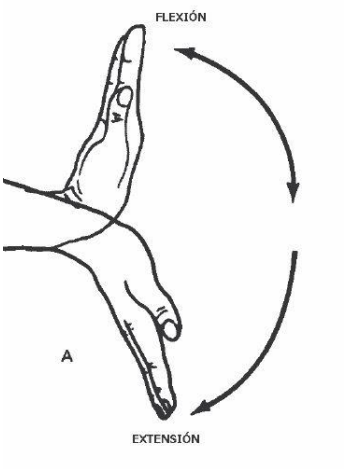

Fig. 41 Movimientos estudiados de la muñeca (https://www.tafadycursos.com/imagenes/cuerpohumano/Muneca-Flexion-Extension.jpg)

Como se puede apreciar en los datos de la *Tabla 3,* los ángulos obtenidos tienen sentido teniendo en cuenta el ángulo máximo y que se trata de personas sin lesiones ni dolencias, excepto una en periodo de rehabilitación, se trata del paciente que menor ángulo ha obtenido en el ejercicio de extensión de muñeca.

En cuanto al ejercicio de flexo-extensión del codo, una persona sana puede flexionar el codo a un ángulo superior a 90 grados. Este error puede atribuirse a dos posibles causas. En primer lugar, en ejercicios dinámicos con un amplio rango de movimiento, como el movimiento del codo, es posible que el acelerómetro se vea afectado por el propio movimiento, lo que podría contaminar las mediciones. En segundo lugar, la causa podría estar relacionada con el procesamiento en tiempo real y la ejecución de un exceso de operaciones cada vez que se calcula una muestra. El sistema podría funcionar de manera más efectiva si se dispusiera de tiempo para corregirlo y eliminar todas aquellas instrucciones innecesarias que se ejecutan durante cada muestreo.

Es importante destacar que la posición en la que se coge el móvil importa, hay que tratar de cogerlo lo más recto posible. Porque esto puede añadir unos grados de error a la medida. Este pequeño

inconveniente se solucionaría guiando correctamente al paciente en el proceso de aprendizaje de realización de la prueba.

Por último, cabe destacar que algunos de los usuarios que probaron la aplicación han destacado la comodidad de poder utilizar únicamente un teléfono móvil para monitorizarse y enviar la información directamente al médico.

6. CONCLUSIONES Y LÍNEAS FUTURAS

6.1 CONCLUSIONES

En este proyecto, se tenía como objetivo el desarrollo de una aplicación móvil de ayuda a la rehabilitación. Se optó por desarrollar una aplicación destinada a registrar información sobre tres ejercicios de rehabilitación específicos: flexo-extensión de codo, flexión de muñeca y extensión de muñeca. Inicialmente, se consideró la utilización de una placa Arduino en conjunto con una IMU externa colocada en el cuerpo del paciente, particularmente en el brazo y la mano. Este sistema podría encargarse de recopilar datos de los sensores y transmitirlos a una aplicación móvil para su posterior procesamiento y seguimiento. Fundamentalmente se buscaba experimentar con un sensor comercial que incorporase acelerómetro y giroscopio para analizar su comportamiento y ensayar algunos algoritmos de procesado. Sin embargo, de acuerdo con los objetivos del proyecto, se continuó el trabajo mediante el empleo de sistemas económicos, fiables y compactos que incorporasen tanto el sensor de forma integrada, como la capacidad de procesamiento y de comunicación; es decir sistemas basados en el uso de teléfonos móviles, eliminando así la necesidad de dispositivos adicionales. Este enfoque se adoptó, además, con el propósito de simplificar la experiencia del paciente, ya que la mayoría de las personas poseen un teléfono inteligente y están familiarizadas con su uso.

A lo largo de la ejecución del proyecto, se logró desarrollar una aplicación móvil utilizando el framework Flutter. Esta aplicación hace uso del acelerómetro del dispositivo para registrar el ángulo de amplitud del movimiento, también registra el tiempo requerido para completar cada ejercicio y el número de repeticiones. Toda esta información se transmite al terapeuta a través del correo electrónico, lo que permite llevar un seguimiento continuo del progreso del paciente desde su propio hogar.

Durante el proceso de desarrollo, surgieron desafíos relacionados con la precisión de los sensores al medir el ángulo de amplitud del movimiento. Finalmente, se optó por utilizar exclusivamente el acelerómetro, a pesar de las limitaciones asociadas al ruido en las mediciones, debido a que ofrecía resultados más confiables y una respuesta más rápida. Sin embargo, al probar una versión final de la aplicación en el laboratorio, se observaron problemas relacionados con la exactitud del ángulo medido. Estos inconvenientes se atribuyeron a la sensibilidad del sensor de aceleración a la dinámica del movimiento, a los posibles retardos entre las muestras del acelerómetro debido al procesado dentro del programa y a la forma en que se sostenía el teléfono móvil durante la realización de los ejercicios, lo que influyó en la precisión de las mediciones. Realmente no se buscaba una precisión exacta del ángulo sino obtener información de la dinámica del ejercicio y que además el médico con la información recogida compruebe la correcta ejecución y pueda asesorar mejor al paciente en sus ejercicios fuera del entorno clínico.

Un aspecto positivo destacable es que se logró desarrollar una aplicación que monitoriza el ángulo de amplitud del movimiento utilizando únicamente los recursos disponibles en el teléfono móvil, en este caso, el acelerómetro. Esto elimina la necesidad de contar con ordenadores u otros dispositivos adicionales.

No obstante, como áreas de mejora, se requiere un perfeccionamiento en el procesamiento de los datos del sensor para obtener mediciones más precisas y reducir el margen de error como se ha explicado en el apartado de resultados de las pruebas. Además, sería beneficioso incorporar más funciones de retroalimentación en la aplicación, ya que el prototipo actual se limita a mostrar imágenes y proporcionar una guía básica sobre cómo realizar los ejercicios. También podría incluirse una mayor variedad de ejercicios que trabajen otras extremidades.

6.2 LINEAS FUTURAS

En esta sección, se presentarán algunas sugerencias para una posible ampliación de la aplicación, con el objetivo de incorporar características más robustas y efectivas:

Mayor Variedad de Ejercicios: Se podría enriquecer la aplicación añadiendo una gama más amplia de ejercicios de rehabilitación. Además, se podrían introducir criterios más rigurosos, como la posibilidad de establecer un tiempo máximo para completar cada repetición. Esto ayudaría a personalizar aún más el programa de rehabilitación para cada paciente.

Clasificación de Ejercicios: Implementar sistemas de clasificación de ejercicios podría ser beneficioso. Esto permitiría detectar qué ejercicios se han completado correctamente, cuáles no se han podido finalizar y cuáles se han realizado de forma incorrecta. Esta información sería valiosa para evaluar el progreso del paciente de manera más precisa.

Base de Datos: La creación de una base de datos que almacene toda la información de los ejercicios realizados durante el proceso de rehabilitación. A través de técnicas de Big Data y Deep Learning, se podría analizar esta información para obtener una comprensión más profunda del progreso de los pacientes a lo largo del tiempo y adaptar el tratamiento en consecuencia.

Incrementar la retroalimentación: Para mejorar la interacción con la aplicación, se podría incorporar una retroalimentación más completa. Esto podría incluir señales auditivas, vibraciones u otros tipos de retroalimentación para indicar el inicio y finalización de cada repetición o para alertar al paciente sobre errores en la ejecución. Además, se podría permitir al paciente visualizar su progreso mediante gráficas o tablas que muestren su evolución durante todo el proceso de rehabilitación.

Portal para el médico: Permitiría a los profesionales de la salud modificar los ejercicios de forma personalizada para cada paciente, teniendo en cuenta su evolución y necesidades específicas. Esta función mejoraría la atención y la adaptación del tratamiento a lo largo del tiempo.

Mejoras en el Acelerómetro: Como ya se ha expuesto en el apartado resultados, al realizar movimientos de gran amplitud, como en el ejercicio del codo, las mediciones del acelerómetro se pueden ver afectadas por el propio movimiento. Otro problema reside en el procesamiento en tiempo real de los datos del acelerómetro, posiblemente debido a la sobrecarga de operaciones en el cálculo de cada muestra. El sistema podría potencialmente funcionar de manera más eficiente si se dispusiera de tiempo para realizar correcciones y eliminar las instrucciones superfluas durante el proceso de muestreo.

7. BIBLIOGRAFÍA

[1] Agnew, J. M. R., Hanratty, C. E., McVeigh, J. G., Nugent, C., & Kerr, D. P. (2022). An Investigation Into the Use of mHealth in Musculoskeletal Physiotherapy: Scoping Review. *JMIR Rehabilitation and Assistive Technologies*, *9*(1), e33609. https://doi.org/10.2196/33609

[2] Ntt Data. (2018, January 16). *La revolución del mHealth en salud: De las apps al dato de salud integrado*. EhCOS. https://www.ehcos.com/la-revolucion-del-mhealth-en-salud/#:~:text=El%20mHealth%20es%2C%20seg%C3%BAn%20definici%C3%B3n

[3] van Dijk-Huisman, H. C., Weemaes, A. T. R., Boymans, T. A. E. J., Lenssen, A. F., & de Bie, R. A. (2020). Smartphone App with an Accelerometer Enhances Patients' Physical Activity Following Elective Orthopedic Surgery: A Pilot Study. *Sensors*, 20(15), 4317. https://doi.org/10.3390/s20154317

[4] Ryan, S., Ní Chasaide, N., O' Hanrahan, S., Corcoran, D., Caulfield, B., & Argent, R. (2021). mHealth Apps for Musculoskeletal Rehabilitation: State of the Practice Review (Preprint). *JMIR Rehabilitation and Assistive Technologies*, *9*(3). https://doi.org/10.2196/34355

[5] Timmers, T., Janssen, L., van der Weegen, W., Das, D., Marijnissen, W.-J., Hannink, G., van der Zwaard, B. C., Plat, A., Thomassen, B., Swen, J.-W., Kool, R. B., & Lambers Heerspink, F. O. (2019). The Effect of an App for Day-to-Day Postoperative Care Education on Patients With Total Knee Replacement: Randomized Controlled Trial. *JMIR MHealth and UHealth*, *7*(10), e15323. https://doi.org/10.2196/15323

[6] Giggins, O. M., Sweeney, K. T., & Caulfield, B. (2014). Rehabilitation exercise assessment using inertial sensors: a cross-sectional analytical study. *Journal of NeuroEngineering and Rehabilitation*, *11*(1), 158. https://doi.org/10.1186/1743-0003-11-158

[7] Bell, K. M., Onyeukwu, C., McClincy, M. P., Allen, M., Bechard, L., Mukherjee, A., Hartman, R. A., Smith, C., Lynch, A. D., & Irrgang, J. J. (2019). Verification of a Portable Motion Tracking System for Remote Management of Physical Rehabilitation of the Knee. *Sensors (Basel, Switzerland)*, *19*(5). https://doi.org/10.3390/s19051021

[8] Hoogland, J., Wijnen, A., Munsterman, T., Gerritsma, C. L., Dijkstra, B., Zijlstra, W. P., Annegarn, J., Ibarra, F., Zijlstra, W., & Stevens, M. (2019). Feasibility and Patient Experience of a Home-Based Rehabilitation Program Driven by a Tablet App and Mobility Monitoring for Patients After a Total Hip Arthroplasty. *JMIR MHealth and UHealth*, *7*(1), e10342. https://doi.org/10.2196/10342

[9] Hempel, C., Sezier, A., & Terry, G. (2018). What helps or hinders clinicians in their decision-making processes when using or prescribing mHealth apps in practice? An exploratory study. *New Zealand Journal of Physiotherapy*, *46*(2), 73–78. https://doi.org/10.15619/nzjp/46.2.04

[10] O'Reilly, M. A., Slevin, P., Ward, T., & Caulfield, B. (2018). A Wearable Sensor-Based Exercise Biofeedback System: Mixed Methods Evaluation of Formulift. *JMIR MHealth and UHealth*, *6*(1), e33. https://doi.org/10.2196/mhealth.8115

[11] Bravo, M. (2023, January 13). *En 2023: Las apps móviles tendrán más espacios en nuestras vidas*. TecnoPymes Bolivia. https://tecnopymes.bo/2023/01/13/en-2023-las-apps-moviles-tendran-mas-espacios-en-nuestras-vidas/

[12] Majumder, S., & Deen, M. J. (2019). Smartphone Sensors for Health Monitoring and Diagnosis. *Sensors*, *19*(9), 2164. https://doi.org/10.3390/s19092164

[13] Actualizatec. (2021a, October). *App nativa: ventajas e inconvenientes I Glosario de App Marketing*. Actualízatec: Mobile & App Marketing. https://actualizatec.com/blog/app-nativa/

[14] GooApps. (2022, September 8). *¿App Nativa, App Híbrida o App Web?* GooApps®. https://gooapps.es/2022/09/08/diferencias-entre-app-nativa-app-web-y-app-hibrida/

[15] Zapater, S. (2022, May 28). *App híbrida o nativa: diferencias y ejemplos*. Blog de Hiberus Tecnología. https://www.hiberus.com/crecemos-contigo/app-hibrida-o-nativa/

[16] Fernández, C. (2021, September 22). *Aplicaciones nativas, todo lo que necesitas saber*. ABAMobile. https://abamobile.com/web/que-son-aplicaciones-nativas-y-ventajas/#:~:text=Qu%C3%A9%20son%20las%20aplicaciones%20m%C3%B3viles%20nativas&text=Se%20llaman%20aplicaciones%20nativas%20debido

[17] González Diez, J. (2019). Diseño y desarrollo de una aplicación móvil de juegos serios para niños y adolescentes con diversidad cognitiva. *Uvadoc.uva.es*, 22–37. https://uvadoc.uva.es/handle/10324/38791

[18] Carrillo, P. (2022, May 19). *¿Web App o App Nativa?* Doonamis. https://www.doonamis.es/web-app/

[19] Cavero, J. (2023, May 15). *¿Qué tecnología es la mejor para desarrollar mi aplicación web a medida?* MentorDay WikiTips. https://mentorday.es/wikitips/tecnologia-mejor-desarrollar-aplicacion-web-medida/#:~:text=Existen%20varias%20tecnolog%C3%ADas%20utilizadas%20para

[20] Actualizatec. (2021b, November). *¿Qué es una Aplicación Híbrida?* Actualízatec: Mobile & App Marketing. https://actualizatec.com/blog/apps-aplicaciones-hibridas/

[21] TechTarget. (2021, January). *¿Qué es Aplicación híbrida o app híbrida? - Definición en WhatIs.com*. ComputerWeekly.es. https://www.computerweekly.com/es/definicion/Aplicacion-hibrida-o-app-hibrida

[22] Deusto Formación. (2021, September 9). *Aplicaciones híbridas: qué son, usos y ejemplos | Deusto Formación*. Deusto. https://www.deustoformacion.com/blog/apps-moviles/todo-sobre-aplicaciones-hibridas

[23] Triguero, D. (2017, October 19). *Aplicaciones híbridas: Qué son y cuáles son sus ventajas y desventajas*. Profile Software Services. https://profile.es/blog/aplicaciones-moviles-hibridas-la-solucion-mas-eficiente-para-el-desarrollo-multiplataforma/#:~:text=Adem%C3%A1s%20nos%20facilita%20la%20conexi%C3%B3n

[24] Medina, F. (2022, May 11). *Mejores tecnologías para desarrollo de aplicaciones móviles 2022 - Armadillo Amarillo*. Armadillo Amarillo. https://www.armadilloamarillo.com/blog/mejores-tecnologias-para-desarrollo-de-aplicaciones-moviles-2022/

[25] C., R. (n.d.). *¿Por qué debería elegir Flutter sobre el desarrollo de aplicaciones nativas en 2022?* Cisin. https://www.cisin.com/coffee-break/es/technology/choose-flutter-over-native-app-development-in-2022.html

[26] Daily, S. E. (2018, July 9). *Flutter with Eric Seidel*. Software Engineering Daily. https://softwareengineeringdaily.com/2018/07/09/flutter-with-eric-seidel/

[27] Moreno Valle, J. (2022, May 20). *Flutter & Dart: primeros pasos para montar tu propia app*. Enmilocalfunciona. https://www.enmilocalfunciona.io/flutter-sdk-de-google-para-apps-multiplataforma/#:~:text=Flutter%20es%20un%20SDK%20gratuito

[28] Aures Tic. (2019, July 10). *¿Qué es Flutter? - Desarrollo de Aplicaciónes móviles | Aures Tic*. Aurestic. https://aurestic.es/que-es-flutter/

[29] Google Developers. (2018, November 19). *Cómo convencer a tus jefes o clientes para que usen Flutter*. Google Developers. https://developers-latam.googleblog.com/2018/11/como-convencer-tus-jefes-o-clientes.html

[30] Mezquita, A. (2021, December 10). *¿Por qué Flutter para desarrollar apps a partir de ahora?* 480. https://cuatroochenta.com/por-que-flutter/

[31] Sujay Vailshery, L. (2023, June 1). *Cross-platform mobile frameworks used by global developers 2020*. Statista. https://www.statista.com/statistics/869224/worldwide-software-developer-working-hours/

[32] I. Fuioaga. (2022, December 26). *Estado actual de Flutter*. Https://Www.abalit.org. https://www.abalit.org/blog/post/desarrollar-app-flutter-en-2023/es

[33] Amazon Web Services. (n.d.). *¿Qué es Flutter? - Explicación de la aplicación Flutter - AWS*. Amazon Web Services, Inc. https://aws.amazon.com/es/what-is/flutter/

[34] Barrionuevo, E. (2021). *CodiGo by Tecsup | Cursos de Programación | Bootcamp Aprende a Programar desde Cero*. Codigo.edu.pe. https://codigo.edu.pe/blog/por-que-elegir-flutter-para-desarrollar-aplicaciones-moviles

[35] Pasarela de Pagos. (n.d.). *Desarrollo Flutter*. Pasarelas de Pagos. https://www.pasarelasdepagos.com/desarrollo-flutter/

[36] EDteam. (2023). *¿Qué es Flutter? La tecnología que cambió el desarrollo multiplataforma | EDteam*. Ed.team. https://ed.team/blog/que-es-flutter-la-tecnologia-que-cambio-el-desarrollo-multiplataforma

[37] H, C. (2023, March 7). *Dart: el lenguaje de programación moderno para aplicaciones móviles y de escritorio.* 7ret Software. https://7ret.com/index.php/2023/03/07/dart-el-lenguaje-de-programacion-moderno-para-aplicaciones-moviles-y-de-escritorio/#:~:text=Dart%20tambi%C3%A9n%20cuenta%20con%20una

[38] Admin. (2023, February 11). *Flutter for Android Studio vs VS Code: Which is Better for App Development?* Instructive Tech. https://instructivetech.com/flutter-for-android-studio-vs-vs-code/#:~:text=Android%20Studio%20is%20a%20more

[39] Flutter. (n.d.-b). *Windows install*. Docs.flutter.dev. https://docs.flutter.dev/get-started/install/windows

[40] Flutter. (n.d.-a). *Set up an editor*. Docs.flutter.dev. https://docs.flutter.dev/get-started/editor?tab=vscode

[41] Miranda Sabando, S. (2019). Medición de los ángulos corporales mediante sensores de un smarthpone: Comparación de aplicaciones disponibles y estudio de su utilidad. *Uvadoc.uva.es*. https://uvadoc.uva.es/handle/10324/41668

[42] Montilla Quilarque, M. J. (2016, October 14). *Goniometría: La medición de los ángulos*. Entorno Empresarial. https://entornove.weebly.com/blog/goniometria-la-medicion-de-los-angulos

[43] Beeby, S. P. (2004). MEMS Mechanical Sensors20042MEMS Mechanical Sensors. Artech House, 2004. 334 pp., ISBN: 1-58053-536-4 £63.00 (hardcover). *Sensor Review*, *24*(3), 319–320. https://doi.org/10.1108/sr.2004.24.3.319.2

[44] Prime Faraday Technology. (2002). *An Introduction to MEMS (Micro-electromechanical Systems)*. Prime Faraday Technology Watch - School of Mechanical and Manufacturing Engineering Loughborough University.

[45] Wikipedia. (2006, November 17). *Sistemas microelectromecánicos*. Wikipedia.org; Wikimedia Foundation, Inc. https://es.wikipedia.org/wiki/Sistemas_microelectromec%C3%A1nicos

[46] Universidad de Sevilla. (n.d.). *Capítulo 2 Tecnología MEMS* . https://biblus.us.es/bibing/proyectos/abreproy/4966/fichero/e.+Tecnologia+MEMS.pdf

[47] Llamas, L. (2016a, September 4). *Cómo usar un giroscopio en nuestros proyectos de Arduino*. Luis Llamas. https://www.luisllamas.es/como-usar-un-giroscopio-arduino/

[48] de Jesus, V. L. B., Pérez, C. A. C., de Oliveira, A. L., & Sasaki, D. G. G. (2018). Understanding the gyroscope sensor: a quick guide to teaching rotation movements using a smartphone. *Physics Education*, *54*(1), 015003. https://doi.org/10.1088/1361-6552/aae3fc

[49] Ramírez Castro, J. L. (2019). Experimentación en Física con dispositivos móviles: O cómo usar los teléfonos y las tabletas inteligentes en el laboratorio escolar. In *Google Books*. Lorenzo Ramírez Castro. https://books.google.es/books?id=qH3oDwAAQBAJ&printsec=frontcover&hl=es&source=gbs_ge_summary_r&cad=0#v=onepage&q&f=false

[50] Tofé Blanco, P. (2015). *Diseño de controlador PID para plataforma giroestabilizada*. 1library.co. https://1library.co/article/filtro-complementario-c%C3%A1pitulo-teor%C3%ADa-acondicionamiento-sensores.nq7p1edq

[51] Llamas, L. (2016b, September 7). *Medir la inclinación con IMU, Arduino y filtro complementario*. Luis Llamas. https://www.luisllamas.es/medir-la-inclinacion-imu-arduino-filtro-complementario/

[52] Universidad de Sevilla . (n.d.). *Sensor medidor de Aceleración. ACELERÓMETRO*. Biblus.us.es. Retrieved 2023, from https://biblus.us.es/bibing/proyectos/abreproy/11638/descargar_fichero/Capitulo+4.pdf

[53] Enterprise DNA Experts. (2023, April 22). *What Is Chat GPT? - Everything You Need to Know*. Unlock the Power of Data. https://blog.enterprisedna.co/what-is-chat-gpt-everything-you-need-to-know/

[54] Fernández, Y. (2022, December 14). *ChatGPT: qué es, cómo usarlo y qué puedes hacer con este chat de inteligencia artificial GPT-3*. Xataka. https://www.xataka.com/basics/chatgpt-que-como-usarlo-que-puedes-hacer-este-chat-inteligencia-artificial

[55] Naylamp Mechatronics. (n.d.). *Tutorial MPU6050, Acelerómetro y Giroscopio*. Naylamp Mechatronics - Perú. Retrieved 2023, from https://naylampmechatronics.com/blog/45_tutorial-mpu6050-acelerometro-y-giroscopio.html

8. ANEXOS

ANEXO 1. CÓDIGOS DE LA APLICACIÓN

En este anexo se exponen los códigos al completo de cada archivo ".dart" incluido en la aplicación.

Main.dart:

```dart
import 'package:sensor_rehab/splash_screen.dart';
import 'package:flutter/material.dart';

//Función principal inicia la aplicación
void main() {
  //Inicia la aplicación MyApp
  runApp(const MyApp());
}

//Clase que define la aplicación
class MyApp extends StatelessWidget {
  const MyApp({super.key});

  @override
  Widget build(BuildContext context) {
    return const MaterialApp(
      //Desactiva el banner de depuración en la esquina superior derecha de la app
      debugShowCheckedModeBanner: false,

      //Pagina que se muestra nada más abrir la app
      home: SplashScreen(),
    );
  }
}
```

Splash_screen.dart:

```dart
import 'package:flutter/material.dart';
import 'package:sensor_rehab/inicio.dart';

class SplashScreen extends StatefulWidget {
  const SplashScreen({super.key});

  @override
  State<SplashScreen> createState() => _SplashScreenState();
}

class _SplashScreenState extends State<SplashScreen>
    with SingleTickerProviderStateMixin {
  //Controlador de animación para la transición
  late AnimationController _animationController;
```

```dart
  //Animación de escala que cambia gradualmene el tamaño del widget
  late Animation<double> _animation;
  //Widget que se muestra despues de la animación, es la pantalla de Inicio.
  Widget inicio = const InicioPage();

  @override
  void initState() {
    super.initState();

    //Inicializa el controlador de animación para la transición
    _animationController = AnimationController(
      vsync: this,
      duration: const Duration(seconds: 5), //Duración de la animación
    );

    //Define la animación usando Tween para la escala (de 0.0 a 1.0)
    //Es decir el texto irá aumentando de tamaño en los 5 segundos definidos antes
    _animation = Tween<double>(begin: 0.0, end: 1.0).animate(
     //Crea la animación
      CurvedAnimation(
        parent: _animationController, //utiliza _animationController como controlador de la
animacion
        curve: Curves.easeInOut, //Controla como acelera o desacelera la animación a medida
que progresa
      ),
    );

    //Inicia la animación hacia adelante
    _animationController.forward();

    //Para detectar cuando la animación se completa
    _animationController.addStatusListener((status) {
      if (status == AnimationStatus.completed) {
        // Navegar a la siguiente pantalla después de completar la animación
        Navigator.pushReplacement(
          context,
          MaterialPageRoute(builder: (BuildContext context) => inicio), //La siguiente
pagina es Inicio
        );
      }
    });
  }

  @override
  void dispose() {
    //Libera recursos al finalizar la pantalla
    _animationController.dispose();
    super.dispose();
```

```
  }

  @override
  Widget build(BuildContext context) {
    return GestureDetector(
      onTap: () {
        // Salta a la siguiente pantalla al tocar la pantalla de SplashScreen
        Navigator.pushReplacement(
          context,
          MaterialPageRoute(builder: (BuildContext context) => inicio),
        );
      },
      child: Scaffold(
        //Color del fondo de la pantalla de animación
        backgroundColor: const Color(0xFFFF8B1F),

        body: Center(
          child: ScaleTransition(
            scale: _animation, //Aplica la animación de escala al widget
            child: const Text(
              'SensorRehab', //Texto que va a ir aumentando de tamaño (es la animación)
              style: TextStyle(
                fontSize: 40.0, //Tamaño de la letra cuando finaliza la animación
                fontWeight: FontWeight.bold, //Texto en negrita
                color: Colors.white, //Color del texto
              ),
            ),
          ),
        ),
      ),
    );
  }
}
```

Inicio.dart:

```
import 'package:flutter/material.dart';
import 'package:sensor_rehab/menu_ejercicios.dart';
import 'package:sensor_rehab/home.dart';


//Clase que define la página de inicio
class InicioPage extends StatefulWidget {
  const InicioPage({Key? key}) : super(key: key);

  @override
  State<InicioPage> createState() => _InicioPageState();
}
```

```dart
class _InicioPageState extends State<InicioPage> {
  //Indice de la página actual en la barra de navegación inferior
  int currentPage = 0;
  List<Widget> pages = const [
    HomePage(), //Pagina de inicio
    ExercisePage(), //Pagina de ejercicios
  ];

  @override
  Widget build(BuildContext context) {
    return Scaffold(

      body: Navigator(
        onGenerateRoute: (settings) {
          return MaterialPageRoute(
            builder: (context) => pages[currentPage], //Muestra la página actual
          );
        },
      ),
      //crea una barra de navegación inferior
      bottomNavigationBar: NavigationBar(
        destinations: const [
          //Lista de las opciones posibles que hay en la barra inferior
          NavigationDestination(icon: Icon(Icons.home), label: 'Inicio'),
          NavigationDestination(
              icon: Icon(Icons.fitness_center_outlined), label: 'Ejercicios'),
        ],
        onDestinationSelected: (int index) {
          setState(() {
            currentPage = index; //Cmbia la página actual al hacer click en uno de los dos
iconos de la barra inferior
          });
        },
        selectedIndex: currentPage, //Índice de la página actual seleccionada en la barra
inferior
      ),
    );
  }
}

//Widget para la barra de navegación inferior
class NavigationBar extends StatelessWidget {
  final List<NavigationDestination> destinations; //Lista que contiene las páginas de
destino
  final int selectedIndex; //Índice de la pagina destino seleccionada
  final ValueChanged<int> onDestinationSelected; //Función llamada al seleccionar una
opcion
```

```
  const NavigationBar({
    Key? key,
    required this.destinations,
    required this.selectedIndex,
    required this.onDestinationSelected,
  }) : super(key: key);

  @override
  Widget build(BuildContext context) {
    return Container(
      decoration: const BoxDecoration(
        color: Colors.white,
        boxShadow: [
          BoxShadow(
            color: Colors.black12,
            offset: Offset(0, -1),
            blurRadius: 8.0,
          ),
        ],
      ),
      child: BottomNavigationBar(
        items: destinations.map((destination) {
          return BottomNavigationBarItem(
            icon: destination.icon, //Icono de la opción
            label: destination.label, //Etiqueta de la opción
          );
        }).toList(),
        currentIndex: selectedIndex, //Indice de la seleccion actual en la barra de
navegacion
        onTap: onDestinationSelected, //Funcion llamada al hacer clik en una de las
opciones posibles
        selectedItemColor: const Color(0xFFFF8B1F), //Color del icono seleccionado
        unselectedItemColor: Colors.grey, //Color del icono no seleccionado
      ),
    );
  }
}

//Clase que define las propiedades de una destinación de navegación
class NavigationDestination {
  final Icon icon; //Icono de la opcion
  final String label; //etiqueta de la opcion

  const NavigationDestination({
    required this.icon,
    required this.label,
  });
```

```
}
```

Home.dart:

```
import 'package:flutter/material.dart';

class HomePage extends StatelessWidget {
  const HomePage({Key? key}) : super(key: key); //Constructor de la clase

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      //Diseño de la appBar
       appBar: AppBar(
        backgroundColor: const Color(0xFFFF8B1F), //Fondo de la appBar
        title: const Text(
          'SensorRehab', //Titutlo de la appBar
          style: TextStyle(
            fontWeight: FontWeight.w900, //grosor de la letra
            fontFamily: 'Montserrat', //Tipo de letra
            fontSize: 24, //Tamaño de letra
            color: Colors.white, //Color de la letra
          ),
        ),
      ),
      //Diseño del cuerpo de la página
      body: const Center(
        child: Column(
          mainAxisAlignment: MainAxisAlignment.center, //Centra los elementos verticalmente
          children: [
            Text(
              'Bienvenido \na \nSensorRehab', //Texto con salto de linea
              textAlign: TextAlign.center, //Alinea el texto al centro
              style: TextStyle(fontSize: 30.0, fontWeight: FontWeight.bold,), //Estilo del
texto
            ),
            SizedBox(height: 30.0),

          ],
        ),
      ),
    );
  }
}
```

Menú_ejercicios.dart:

```
import 'package:flutter/material.dart';
```

```dart
import 'package:sensor_rehab/ejercicio_codo.dart';
import 'package:sensor_rehab/ejercicio_extension_muneca.dart';
import 'package:sensor_rehab/ejercicio_flexion_muneca.dart';

//Clase que define la página del menu ejercicios
class ExercisePage extends StatelessWidget {
  const ExercisePage({Key? key}) : super(key: key);

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      //Diseño de la appBar
      appBar: AppBar(
        backgroundColor: const Color(0xFFFF8B1F),
        title: const Text(
          'EJERCICIOS',
          style: TextStyle(
            fontWeight: FontWeight.w900,
            fontFamily: 'Montserrat',
            fontSize: 30,
            color: Colors.white,
          ),
        ),
      ),
      //permite desplazar verticalmente la pantalla
      body: SingleChildScrollView(
        child: Column(
          //Lista de hijos de la columna
          children: [

            //Llama a la funcion _buildExercise y la pasa los parametros necesarios
            _buildExercise('Flexo-Extensión de codo',
                'images/flexionExtensionCodo.jpg', ExerciseType.codo, context),
            _buildExercise('Extensión de muñeca', 'images/extensionMuneca.jpg',
                ExerciseType.munecaExtension, context),
            _buildExercise('Flexión de muñeca', 'images/flexionMuñeca.jpg',
                ExerciseType.munecaFlexion, context),

          ],
        ),
      ),
    );
  }

  //Funcion para construir cada opcion de ejercicio
  Widget _buildExercise(
      String name, String image, ExerciseType type, BuildContext context) {
    //Diseño del contenedor que representa una opción de ejercicio
```

```
    return Container(
      margin: const EdgeInsets.symmetric(horizontal: 10, vertical: 10), //Margenes
      decoration: BoxDecoration(
        color: Colors.white, //Color de fondo del contenedor
        //Diseño sombra del contenedor
        boxShadow: [
          BoxShadow(
            color: Colors.grey.withOpacity(0.5), //Color y opacidad de la sombra
            spreadRadius: 3, //Radio de difusión de la sombra
            blurRadius: 5, //Radio de desenfoque de la sombra
            offset: const Offset(0, 3), //Desplazamiento de la sombra con respecto al borde
          ),
        ],
      ),
      child: Column(
        mainAxisAlignment: MainAxisAlignment.center, //Centra el contenido de la columna
verticalmente
        children: [
          Padding(
            padding: const EdgeInsets.all(8.0),
            child: Text(
              name, //Nombre del ejercicio
              style: const TextStyle(
                fontSize: 20, //Tamaño de letra del titulo del ejercicio
                fontWeight: FontWeight.bold, //Letra negrita
                fontFamily: AutofillHints.photo, //Tipo de letra
              ),
            ),
          ),

         //Imágen del ejercicio
          Image.asset(
            image, //Ruta de la imagen del ejercicio
            height: 200, //Largo
            width: 600, //Ancho
          ),
          //Boton para acceder al ejercicio
          ElevatedButton(
            //Segun la opción de ejercicio que se elija se entrara en una pagina distinta
            onPressed: () {
              switch (type) {
                case ExerciseType.codo:
                  Navigator.push(
                    context,
                    MaterialPageRoute(
                        builder: (context) => const EjercicioCodoPage()),
                  );
                  break;
```

```
              case ExerciseType.munecaExtension:
                Navigator.push(
                  context,
                  MaterialPageRoute(
                      builder: (context) => const ExtensionMunecaPage()),
                );
                break;
              case ExerciseType.munecaFlexion:
                Navigator.push(
                  context,
                  MaterialPageRoute(
                      builder: (context) => const FlexionMunecaPage()),
                );
                break;

            }
          },
          //Estilo del boton
          style: ElevatedButton.styleFrom(
            foregroundColor: Colors.white, //Color del texto
            backgroundColor: const Color(0xFFFF8B1F), //Color del fondo del boton
            padding: const EdgeInsets.symmetric(
              vertical: 12,
              horizontal: 24,
            ),
            shape: RoundedRectangleBorder(
              borderRadius: BorderRadius.circular(15), //Bordes redondeados
            ),
          ),
          child: const Row(
            mainAxisSize: MainAxisSize.min,
            children: [
              Icon(Icons.play_arrow_sharp), //Icono de play
              SizedBox(width: 10),
              Text(
                'Comenzar ejercicio',
                style: TextStyle(
                  fontSize: 20, // tamaño de letra del texto del boton
                ),
              ),
            ],
          ),
        ),
        const SizedBox(height: 10), //Espaciado vertical
      ],
    ),
  );
}
```

```
}
//Define una enumeración para mostrar las diferentes opciones de ejercicios disponibles
enum ExerciseType { codo, munecaExtension, munecaFlexion}
```

ejercicio_codo.dart:

```
import 'package:flutter/material.dart';
import 'package:mailer/mailer.dart';
import 'package:sensors_plus/sensors_plus.dart';
import 'dart:math';
import 'dart:async';
import 'dart:io';
import 'package:path_provider/path_provider.dart';
import 'package:mailer/smtp_server.dart';
import 'package:intl/intl.dart' show DateFormat;

class EjercicioCodoPage extends StatefulWidget {
  const EjercicioCodoPage({super.key});
  @override
  State<EjercicioCodoPage> createState() => _EjercicioCodoPageState();
}

class _EjercicioCodoPageState extends State<EjercicioCodoPage> {
  //Lista para almacenar las suscripciones a los distintos streams de eventos de los
sensores
  final List<StreamSubscription<dynamic>> _streamSubscriptions =
      <StreamSubscription<dynamic>>[];

  List<double> listaAngulos = [];
  double sumMaxAng =0.0; //Almacena la suma de los angulos para calcular la media
  double totalT =0.0; //Variable que almacena el tiempo total empleado en realizar el
ejercicio
  double tRepe = 0.0; //almacenar el tiempo de una repetición
  int numRep = 0;
  double maxAngle = 0.0; //Recoge el ángulo máximo de la repetición actual
  double mediaAng=0.0;

  final Stopwatch _stopwatch = Stopwatch();
  late File archivo;
  bool _inicio = false;
  bool fechaHoraRegistrada = false; //controla que se registre la fecha y hora de inicio
del ejercicio

  @override
  void initState() {
    super.initState();
    //LLama al metodo _resetFile(), para resetear el archivo al inicio del ejercicio
    _resetFile();
```

```
  }

//Función para resetear el archivo cada vez que se vaya a realizar el ejercicio
  void _resetFile() async {
    try {
      //Obtiene el directorio de documentos de la aplicación
      Directory documentsDirectory = await getApplicationDocumentsDirectory();

      //Crea una referencia a la ruta especifica del archivo
      archivo = File('${documentsDirectory.path}/flexoExtensionCodo.txt');

      //Verifica la existencia del archivo
      if (await archivo.exists()) {
        //Si hay un archivo existente borra su contenido
        archivo.writeAsStringSync('');
      } else {
        //Crea un nuevo archivo en la ruta especificada antes
        archivo = await archivo.create(recursive: true);
      }
    } catch (e) {
      //Si ocurre un error lo muestra en la consola (Util durante el desarrollo de la app)
      // ignore: avoid_print
      print('Error al resetear el archivo: $e');
    }
  }

//Función para comenzar a recibir información del acelerómetro
  void _start() {
    //Listas que almacenan los valores absolutos devueltos por el acelerometro
    late List<double> accelerometerValues = [0, 0, 0];
    double angulo = 0.0;
    //se inicia la lectura del acelerómetro
    _streamSubscriptions.add(
      accelerometerEvents.listen((AccelerometerEvent event) {
        //Actualiza los valores del acelerómetro
        setState(() {
          accelerometerValues = [event.x, event.y, event.z];
          angulo = _calcularAngulo(accelerometerValues);
          listaAngulos.add(angulo);
          _stopwatch.start();
        });
      }),
    );
    setState(() {
      _inicio = true;
    });
  }
```

```
  void _finRep() {
    if (listaAngulos.isNotEmpty) {
      maxAngle = listaAngulos.reduce((max, value) => value > max ? value : max);
      sumMaxAng +=
          maxAngle; //Calculamos la suma de los ángulos alcanzados en cada repeticion
      listaAngulos.clear();
      setState(() {
        _inicio = false;
        numRep++;
        _stopwatch.stop(); //Para el reloj
        int ms =
            _stopwatch.elapsedMilliseconds; //Calcula el tiempo en milisegundos
        tRepe = ms / 1000; //Pasa a segundos el tiempo empleado en la repeticion
        totalT += tRepe;
      });
      _stopwatch.reset();
      _writeDataToFile(numRep, tRepe, maxAngle);
    }
  }

//Función que calcula el ángulo de inclinación a través de los datos del acelerómetro
  double _calcularAngulo(List<double> a) {
    //calculamos el ángulo de rotación en el eje Y en grados
    double accelAngY =
        (atan(a[1] / sqrt(pow(a[0], 2) + pow(a[2], 2))) * (180 / pi)) +
            5.0; //sumamos un ofset de 5º
    // ignore: avoid_print
    print("Angulo actual: ${accelAngY.toStringAsFixed(2)}º");

    return accelAngY;
  }

  //Función para escribir en el archivo la información del paciente
  void _writeDataToFile(int rep, double tiempo, double anguloRep) async {
    try {
      //Obtiene el directorio de documentos de la aplicación
      Directory documentsDirectory = await getApplicationDocumentsDirectory();
      archivo = File('${documentsDirectory.path}/flexoExtensionCodo.txt');

      //Crea el archivo si no existe
      await archivo.create(recursive: true);

      //Verifica si han sido registradas la fecha y hora
      if (!fechaHoraRegistrada) {
        // Obtener la fecha y hora actual
        DateTime now = DateTime.now();
        String fechaHora = DateFormat('yyyy-MM-dd HH:mm:ss').format(now);
```

```
        //Añade titulo al archivo
        const titulo = 'EJERCICIO FLEXO-EXTENSIÓN CODO\n';
        await archivo.writeAsString(titulo, mode: FileMode.append);

        //Añade la fecha y hora de inicio
        final inicioContent = 'Fecha y Hora de inicio: $fechaHora\n';
        await archivo.writeAsString(inicioContent, mode: FileMode.append);
        // Marcar que se ha registrado la fecha y hora para que no vuelva a acceder a este
if
        fechaHoraRegistrada = true;
      }

      //Agrega una linea con la información de cada repetición del ejercicio
      final content =
          'Repetición: $rep, Tiempo: $tiempo segundos, Ángulo:
${anguloRep.toStringAsPrecision(2)}º\n';
      await archivo.writeAsString(content, mode: FileMode.append);

      //Verificar que se han realizado todas las repeticiones
      if (rep == 10) {
        //Calcula la media del ángulo alcanzado en el ejercicio
        mediaAng = sumMaxAng / 10;

        //Añade una linea al archivo con la media de los ángulos
        final mediaContent =
            '\n\nMedia de ángulos: ${mediaAng.toStringAsPrecision(2)}º\n';
        await archivo.writeAsString(mediaContent, mode: FileMode.append);

        //Añade una linea al archivo con el tiempo total empleado en el ejercicio
        final tiempoTotalContent = '\nTiempo total: $totalT segundos\n';
        await archivo.writeAsString(tiempoTotalContent, mode: FileMode.append);
      }

      // Obtener la ubicación del archivo
      String filePath = archivo.path;
      // ignore: avoid_print
      print('Archivo creado en la ubicación: $filePath');

      // ignore: avoid_print
      print('Archivo creado y contenido escrito con exito.');
    } catch (e) {
      // ignore: avoid_print
      print('Error al crear y escribir en el archivo: $e');
    }
  }

  //Función para enviar correo con archivo adjunto
  void sendEmailWithAttachment() async {
```

```
    //Configuración del servidor SMTP
final smtpServer = hotmail('correoPaciente@hotmail.com', 'password');

    final message = Message()
      ..from =
          const Address(' correoPaciente@hotmail.com', 'Nombre Apellido1 Apellido2')
      ..recipients.add('correoMédico@gmail.com')
      //Asunto del correo
      ..subject = 'Evolución de la paciente Ana Santos Delgado'
      //Cuerpo del correo
      ..text = 'Adjunto de archivo con información del paciente'
      //Archivo adjunto
      ..attachments.add(FileAttachment(archivo));

    try {
      //Enviar el correo electrónico a través del servidor SMTP
      final sendReport = await send(message, smtpServer);

      // ignore: avoid_print
      print('Correo enviado: ${sendReport.messageSendingEnd}');
    } catch (e) {
      //Caso de error
      // ignore: avoid_print
      print('Error al enviar el correo: $e');
    }
  }

  //Liberar recursos al salir de la página
  @override
  void dispose() {
    super.dispose();
    //Cancela todas las suscripciones a los streams de eventos
    for (StreamSubscription<dynamic> subscription in _streamSubscriptions) {
      subscription.cancel();
    }
  }

  //boton finalizar para salir de la pagina del ejercicio
  void _finEjercicio() {
    //Regresa a la pagina anterior
    Navigator.pop(context);
    //Llama a la función para enviar el correo al doctor
    sendEmailWithAttachment();
  }

  //Se realiza todo el diseño de la interfaz de la página del ejercicio
  @override
  Widget build(BuildContext context) {
```

```
    return Scaffold(
      //Diseño de la appBar
      appBar: AppBar(
        backgroundColor: const Color(0xFFFF8B1F), //Color del fondo de la appbar
        title: const Text(
          'Flexión Extensión Codo', //Texto que aparece en la appBar
          //Estilo del texto
          style: TextStyle(
            fontWeight: FontWeight.w900, //Grosor de la fuente
            fontFamily: 'Montserrat', //Tipo de fuente
            fontSize: 20, //Tamaño de la letra
            color: Colors.white, //Color blanco de la letra
          ),
        ),
        leading: IconButton(
          icon: const Icon(Icons.arrow_back, color: Colors.white, size: 30),
          onPressed: () {
            Navigator.pop(context); //Botón para regresar a la pantalla anterior
          },
        ),
        actions: [
          IconButton(
            onPressed: () {
              //Botón de información para mostrar los pasos del ejercicio
              showDialog(
                context: context,
                builder: (BuildContext context) {
                  //Construir un cuadro de diálogo que muestra los pasos a seguir para
realizar el ejercicio
                  return AlertDialog(
                    title: const Text(
                      'PASOS A SEGUIR PARA REALIZAR EL EJERCICIO',
                      style: TextStyle(
                          fontWeight: FontWeight.w900,
                          fontFamily: 'Montserrat',
                          fontSize: 19,
                          color: Colors.black),
                    ),
                    content: Column(
                      crossAxisAlignment: CrossAxisAlignment.start,
                      children: [
                        //Muestra el primer paso del ejercicio
                        RichText(
                          text: TextSpan(
                            style: DefaultTextStyle.of(context)
                                .style, //Estilo de texto predeterminado
                            children: const [
                              TextSpan(
```

```
                        text: '1. ',
                        style: TextStyle(
                            fontWeight:
                                FontWeight.bold), //Grosor de la letra
                      ),
                      TextSpan(
                        text:
                            'Sobre una superficie plana situe el brazo
completamente estirado con la palma hacia arriba.(Para una mayor comodidad, puede colocar
una almohadilla debajo de la parte superior del brazo.)\n',
                        style: TextStyle(
                          fontSize: 14.5, //Tamaño de la letra
                        ),
                      ),
                    ],
                  ),
                ),
                RichText(
                  text: TextSpan(
                    style: DefaultTextStyle.of(context).style,
                    children: const [
                      TextSpan(
                        text: '2. ',
                        style: TextStyle(fontWeight: FontWeight.bold),
                      ),
                      TextSpan(
                        text:
                            'Situe el móvil en vertical sobre la palma\n',
                        style: TextStyle(
                          fontSize: 14.5,
                        ),
                      ),
                    ],
                  ),
                ),
                RichText(
                  text: TextSpan(
                    style: DefaultTextStyle.of(context).style,
                    children: const [
                      TextSpan(
                        text: '3. ',
                        style: TextStyle(fontWeight: FontWeight.bold),
                      ),
                      TextSpan(
                        text:
                            'Pulse el boton verde para iniciar la repetición\n',
                        style: TextStyle(
                          fontSize: 14.5,
```

```
                ),
              ),
            ],
          ),
        ),
        RichText(
          text: TextSpan(
            style: DefaultTextStyle.of(context).style,
            children: const [
              TextSpan(
                text: '4. ',
                style: TextStyle(fontWeight: FontWeight.bold),
              ),
              TextSpan(
                text:
                    'Flexione el brazo hasta donde pueda sin superar el
umbral de dolor.\n',
                style: TextStyle(
                  fontSize: 14.5,
                ),
              ),
            ],
          ),
        ),
        RichText(
          text: TextSpan(
            style: DefaultTextStyle.of(context).style,
            children: const [
              TextSpan(
                text: '5. ',
                style: TextStyle(fontWeight: FontWeight.bold),
              ),
              TextSpan(
                text:
                    'Vuelva a la posición inicial y pulse el botón
rojo.\n',
                style: TextStyle(
                  fontSize: 14.5,
                ),
              ),
            ],
          ),
        ),
        RichText(
          text: TextSpan(
            style: DefaultTextStyle.of(context).style,
            children: const [
              TextSpan(
```

```
                text: '6. ',
                style: TextStyle(fontWeight: FontWeight.bold),
              ),
              TextSpan(
                text:
                    'Vuelva al paso 3 y repita el ejercicio las veces que
se indica.',
                style: TextStyle(
                  fontSize: 14.5,
                ),
              ),
            ],
          ),
        ),
        RichText(
          text: TextSpan(
            style: DefaultTextStyle.of(context).style,
            children: const [
              TextSpan(
                text: '7. ',
                style: TextStyle(fontWeight: FontWeight.bold),
              ),
              TextSpan(
                text:
                    'Al finalizar pulse salir.\n',
                style: TextStyle(
                  fontSize: 15,
                ),
              ),
            ],
          ),
        ),
        const Row(
          crossAxisAlignment: CrossAxisAlignment
              .start, //Alineado a la izquierda de la fila
          children: [
            Icon(
              Icons.warning, //Icono de advertencia
              color: Colors.yellow,
            ),
            //espacio entre el icono y el texto
            SizedBox(width: 8),
            Expanded(
              //Mensaje de advertencia
              child: Text(
                  'Si nota dolor intenso no fuerce la flexión.'),
            ),
          ],
```

```
                    ),
                  ],
                ),
                actions: [
                  //Boton para cerrar el cuadro de diálogo con los pasos a seguir
                  TextButton(
                    onPressed: () {
                      Navigator.pop(
                          context); //Cierra el cuadro de diálogo al pulsar el boton
                    },
                    child: const Text('Cerrar'), //Texto del boton
                  ),
                ],
              );
            },
          );
        },
        icon: const Icon(Icons.info,
            color: Colors.white,
            size:
                30), //Icono de información, al pulsarle se abre el cuadro de dialogo
      ),
    ],
  ),
  //Diseño del resto de la pantalla sin la appBar
  body: Container(
    decoration: const BoxDecoration(
      color: Colors.white,
      boxShadow: [
        BoxShadow(
          color: Colors.black12,
          blurRadius: 10,
          offset: Offset(0, 5),
        )
      ],
    ),
    child: Column(
      children: [
        //Contenedor que muestra la imagen y el contenido relacionado al ejercicio
        Container(
          //Margen alrededor del contenedor
          margin: const EdgeInsets.symmetric(horizontal: 8, vertical: 8),
          decoration: BoxDecoration(
            color: Colors.white, //Color del contenedor
            //Sombra aplicada al contenedor
            boxShadow: [
              BoxShadow(
                color: Colors.grey
```

```
                        .withOpacity(0.5), //Color y opacidad de la sombra
                    spreadRadius: 3, //Radio de expansión de la sombra
                    blurRadius: 5, //Radio de desenfoque de la sombra
                    offset: const Offset(0,
                        3), //Desplazamiento de la sombra con respecto al borde
                  ),
                ],
              ),
              //Contenido del contenedor
              child: Column(
                //Widgets hijos del contenedor
                children: [
                  Image.asset(
                    'images/flexionExtensionCodo.jpeg', //Ruta de la imagen a mostrar
                    height: 150, //Largo de la imagen
                  ),
                  Container(
                    padding: const EdgeInsets.all(10),
                    alignment: Alignment.center, //Alineada al centro
                    child: Column(
                      children: [
                        //Cuadro con el objetivo de repeticiones a conseguir
                        Container(
                          padding: const EdgeInsets.symmetric(
                              horizontal: 16, vertical: 8),
                          decoration: const BoxDecoration(
                            color: Color.fromRGBO(255, 166, 2,
                                1), //Color del fondo del contenedor
                          ),
                          child: const Text(
                            'Objetivo: 10 repeticiones', //Texto que se muestra en el
contenedor
                            style: TextStyle(
                              fontSize: 20,
                              fontWeight: FontWeight.bold,
                              color: Colors.white,
                            ),
                          ),
                        ),

                        Text(
                          'Repeticiones: $numRep',
                          style: Theme.of(context).textTheme.headlineMedium,
                        ),
                        if (!_inicio)
                          Text(
                            'Ángulo: ${maxAngle.toStringAsFixed(2)}º',
                            style: Theme.of(context).textTheme.headlineMedium,
```

```
                ),
              if (!_inicio && numRep<10 && maxAngle>85)
              Text(
                  'Muy bien, sigue así!',
                  textAlign: TextAlign.center,
                  style: Theme.of(context).textTheme.headlineMedium,
                ),
              if (numRep == 10)
              Text(
                  'Media Alcanzada: ${mediaAng.toStringAsFixed(2)}º',
                  textAlign: TextAlign.center,
                  style: Theme.of(context).textTheme.headlineMedium,
                ),
                if (numRep == 10)
                Text(
                  'Enhorabuena, lograste el objetivo!',
                  textAlign: TextAlign.center,
                  style: Theme.of(context).textTheme.headlineMedium,
                )
            ],
          ),
        ),
      ],
    ),
  ),
),
const SizedBox(height: 30),
if (numRep < 10)
  if (!_inicio)
    Padding(
      padding: const EdgeInsets.all(5.0),
      child: Column(
        mainAxisAlignment: MainAxisAlignment.center,
        children: <Widget>[
          const SizedBox(height: 24),
          ElevatedButton(
              onPressed: _start,
              style: ElevatedButton.styleFrom(
                shape: const CircleBorder(),
                backgroundColor: Colors.green,
                padding: const EdgeInsets.all(24),
              ),
              child: const Icon(
                Icons.play_arrow,
                size: 48,
                color: Colors.white,
              )),
        ],
      ),
```

```
        ),
    if (_inicio)
      Padding(
        padding: const EdgeInsets.all(5.0),
        child: Column(
          mainAxisAlignment: MainAxisAlignment.center,
          children: <Widget>[
            const SizedBox(height: 24),
            ElevatedButton(
              onPressed: _finRep,
              style: ElevatedButton.styleFrom(
                shape: const CircleBorder(),
                backgroundColor: Colors.red,
                padding: const EdgeInsets.all(24),
              ),
              child: const Icon(
                Icons.stop,
                size: 48,
                color: Colors.white,
              ),
            ),
          ],
        ),
      ),
    if (numRep == 10)
      ElevatedButton(
        onPressed: _finEjercicio,
        style: ElevatedButton.styleFrom(
          backgroundColor: const Color(0xFFFF8B1F),
          shape: RoundedRectangleBorder(
            borderRadius: BorderRadius.circular(10.0),
          ),
        ),
        child: Container(
          width: 200.0, //Ancho del boton
          height: 50.0, //alto del botón
          alignment: Alignment.center,
          child: const Text(
            'Salir',
            style: TextStyle(
              color: Colors.white,
              fontSize: 18.0,
            ),
          ),
        ),
      ),
  ],
),
```

```
            ),
          );
        }
      }
```

Ejercicio_extension_muneca.dart:

```
import 'dart:async';
import 'package:flutter/material.dart';
import 'package:sensors_plus/sensors_plus.dart';
import 'dart:math';
import 'dart:io';
import 'package:path_provider/path_provider.dart';
import 'package:mailer/mailer.dart';
import 'package:mailer/smtp_server.dart';
import 'package:intl/intl.dart' show DateFormat;

class ExtensionMunecaPage extends StatefulWidget {
  const ExtensionMunecaPage({super.key});

  @override
  State<ExtensionMunecaPage> createState() => _ExtensionMunecaPageState();
}

class _ExtensionMunecaPageState extends State<ExtensionMunecaPage> {
  //Lista para almacenar las suscripciones a los distintos streams de eventos de los sensores
  final List<StreamSubscription<dynamic>> _streamSubscriptions =
      <StreamSubscription<dynamic>>[];

  List<double> listaAngulos = [];
  double sumMinAng = 0.0; //Variable que almacena la suma de todos los angulos para calcular la
media
  double totalT =0.0; //Variable que almacena el tiempo total empleado en realizar el ejercicio
  double tRepe = 0.0; //almacenar el tiempo de una repetición
  int numRep = 0;
  //El movimiento se hace en el otro sentido y los angulos son negativos
  double anguloMin = 0.0; //Recoge el ángulo mínimo de la repetición actual
  double mediaAng = 0.0;
  final Stopwatch _stopwatch = Stopwatch();
  late File archivo;
  bool _inicio = false;
  bool fechaHoraRegistrada = false; //controla que se registre la fecha y hora de inicio del
ejercicio

  @override
  void initState() {
    super.initState();
```

```
    //LLama al metodo _resetFile(), para resetear el archivo al inicio del ejercicio
    _resetFile();
  }

//Función para resetear el archivo cada vez que se vaya a realizar el ejercicio
  void _resetFile() async {
    try {
      //Obtiene el directorio de documentos de la aplicación
      Directory documentsDirectory = await getApplicationDocumentsDirectory();

      //Crea una referencia a la ruta especifica del archivo
      archivo = File('${documentsDirectory.path}/extensionMuñeca.txt');

      //Verifica la existencia del archivo
      if (await archivo.exists()) {
        //Si hay un archivo existente borra su contenido
        archivo.writeAsStringSync('');
      } else {
        //Crea un nuevo archivo en la ruta especificada antes
        archivo = await archivo.create(recursive: true);
      }
    } catch (e) {
      //Si ocurre un error lo muestra en la consola (Util durante el desarrollo de la app)
      // ignore: avoid_print
      print('Error al resetear el archivo: $e');
    }
  }

  //Función para comenzar a recibir información del acelerómetro
  void _start() {
    //Listas que almacenan los valores absolutos devueltos por el acelerometro
    late List<double> accelerometerValues = [0, 0, 0];
    double angulo = 0.0;
    //se inicia la lectura del acelerómetro
    _streamSubscriptions.add(
      accelerometerEvents.listen((AccelerometerEvent event) {
        //Actualiza los valores del acelerómetro
        setState(() {
          accelerometerValues = [event.x, event.y, event.z];
          angulo = _calcularAngulo(accelerometerValues);
          listaAngulos.add(angulo);
          _stopwatch.start();
        });
      }),
    );
    setState(() {
      _inicio = true;
    });
```

```
  }

  void _finRep() {
    if (listaAngulos.isNotEmpty) {
      anguloMin =
          listaAngulos.reduce((min, value) => value < min ? value : min);
      anguloMin = -anguloMin; //le quitamos el signo menos
      sumMinAng +=
          anguloMin; //Calculamos la suma de los ángulos alcanzados en cada repeticion
      listaAngulos.clear();
      setState(() {
        _inicio = false;
        numRep++;
        _stopwatch.stop(); //Para el reloj
        int ms =
            _stopwatch.elapsedMilliseconds; //Calcula el tiempo en milisegundos
        tRepe = ms / 1000; //Pasa a segundos el tiempo empleado en la repeticion
        totalT += tRepe;
      });
      _stopwatch.reset();
      _writeDataToFile(numRep, tRepe, anguloMin);
    }
  }

//Función para calcular el ángulo
  double _calcularAngulo(List<double> a) {
    //calculamos el ángulo de rotación en grados en el eje Y
    double accelAngY = atan(a[1] / sqrt(pow(a[0], 2) + pow(a[2], 2))) *
            (180 / pi) -
        5.0; //restamos un ofset de 5º porque aquí el ángulo que obtenemos es negativo
    return accelAngY;
  }

  void _writeDataToFile(int numRep, double tiempo, double anguloRep) async {
    try {
      Directory documentsDirectory = await getApplicationDocumentsDirectory();
      archivo = File('${documentsDirectory.path}/extensionMuñeca.txt');
      await archivo.create(recursive: true);
      if (!fechaHoraRegistrada) {
        // Obtener la fecha y hora actual
        DateTime now = DateTime.now();
        String fechaHora = DateFormat('yyyy-MM-dd HH:mm:ss').format(now);
        const titulo = 'EJERCICIO EXTENSIÓN MUÑECA\n';
        await archivo.writeAsString(titulo, mode: FileMode.append);
        final inicioContent = 'Fecha y Hora de inicio: $fechaHora\n';
        await archivo.writeAsString(inicioContent, mode: FileMode.append);

        fechaHoraRegistrada =
```

```
            true; // Marcar que se ha registrado la fecha y hora
      }
      final content =
          'Repetición: $numRep, Tiempo: $tiempo segundos, Ángulo:
${anguloRep.toStringAsPrecision(2)}º\n';
      await archivo.writeAsString(content, mode: FileMode.append);

      //Verificar que se han realizado todas las repeticiones
      if (numRep == 10) {
        //Calcula la media del ángulo alcanzado en el ejercicio
        mediaAng = sumMinAng / 10;

        //Añade una linea al archivo con la media de los ángulos
        final mediaContent =
            '\n\nMedia de ángulos: ${mediaAng.toStringAsPrecision(2)}º\n';
        await archivo.writeAsString(mediaContent, mode: FileMode.append);

        //Añade una linea al archivo con el tiempo total empleado en el ejercicio
        final tiempoTotalContent = '\nTiempo total: $totalT segundos\n';
        await archivo.writeAsString(tiempoTotalContent, mode: FileMode.append);
      }

      // Obtener la ubicación del archivo
      String filePath = archivo.path;
      // ignore: avoid_print
      print('Archivo creado en la ubicación: $filePath');

      // ignore: avoid_print
      print('Archivo creado y contenido escrito con exito.');
    } catch (e) {
      // ignore: avoid_print
      print('Error al crear y escribir en el archivo: $e');
    }
  }

  void sendEmailWithAttachment() async {
    //Configuración del servidor SMTP
    final smtpServer = hotmail('correoPaciente@hotmail.com', 'password');

    final message = Message()
      ..from =
          const Address('correoPaciente@hotmail.com', 'Nombre Apellido1 Apellido2')
      ..recipients.add('correoMédico@gmail.com')
      ..subject = 'Evolución de la paciente Ana Santos Delgado'
      ..text = 'Adjunto de archivo con información del paciente'
      ..attachments.add(FileAttachment(archivo));

    try {
```

```
    //Enviar el correo electrónico a través del servidor SMTP

    final sendReport = await send(message, smtpServer);
    // ignore: avoid_print
    print('Correo enviado: ${sendReport.messageSendingEnd}');
  } catch (e) {
    // ignore: avoid_print
    print('Error al enviar el correo: $e');
  }
}

@override
void dispose() {
  super.dispose();
  for (StreamSubscription<dynamic> subscription in _streamSubscriptions) {
    subscription.cancel();
  }
}

//boton finalizar para salir de la pagina del ejercicio
void _finEjercicio() {
  //Regresa a la pagina anterior
  Navigator.pop(context);
  //Llama a la función para enviar el correo al doctor
  sendEmailWithAttachment();
}

@override
Widget build(BuildContext context) {
  return Scaffold(
    appBar: AppBar(
      backgroundColor: const Color(0xFFFF8B1F),
      title: const Text(
        'Extensión de Muñeca',
        style: TextStyle(
          fontWeight: FontWeight.w900,
          fontFamily: 'Montserrat',
          fontSize: 16.5,
          color: Colors.white,
        ),
      ),
      leading: IconButton(
          icon: const Icon(Icons.arrow_back, color: Colors.white, size: 30),
          onPressed: () {
            Navigator.pop(context);
          }),
      actions: [
        IconButton(
```

```
            onPressed: () {
              showDialog(
                context: context,
                builder: (BuildContext context) {
                  return AlertDialog(
                    title: const Text(
                      'PASOS A SEGUIR PARA REALIZAR EL EJERCICIO',
                      style: TextStyle(
                          fontWeight: FontWeight.w900,
                          fontFamily: 'Montserrat',
                          fontSize: 19,
                          color: Colors.black),
                    ),
                    content: Column(
                      crossAxisAlignment: CrossAxisAlignment.start,
                      children: [
                        RichText(
                          text: TextSpan(
                            style: DefaultTextStyle.of(context).style,
                            children: const [
                              TextSpan(
                                text: '1. ',
                                style: TextStyle(fontWeight: FontWeight.bold),
                              ),
                              TextSpan(
                                text:
                                    'Situa el antebrazo sobre una superficie plana, quedando la
mano en el aire con la palma hacia arriba.\n',
                                style: TextStyle(
                                  fontSize: 15,
                                ),
                              ),
                            ],
                          ),
                        ),
                        RichText(
                          text: TextSpan(
                            style: DefaultTextStyle.of(context).style,
                            children: const [
                              TextSpan(
                                text: '2. ',
                                style: TextStyle(fontWeight: FontWeight.bold),
                              ),
                              TextSpan(
                                text:
                                    'Sostenga el móvil en la mano en posición vertical .\n',
                                style: TextStyle(
                                  fontSize: 15,
```

```
                ),
              ),
            ],
          ),
        ),
        RichText(
          text: TextSpan(
            style: DefaultTextStyle.of(context).style,
            children: const [
              TextSpan(
                text: '3. ',
                style: TextStyle(fontWeight: FontWeight.bold),
              ),
              TextSpan(
                text: 'Pulse el botón verde para iniciar la repetición.\n',
                style: TextStyle(
                  fontSize: 15,
                ),
              ),
            ],
          ),
        ),
        RichText(
          text: TextSpan(
            style: DefaultTextStyle.of(context).style,
            children: const [
              TextSpan(
                text: '4. ',
                style: TextStyle(fontWeight: FontWeight.bold),
              ),
              TextSpan(
                text:
                    'Mueva la mano hacia abajo, sin superar el umbral de
dolor.\n',
                style: TextStyle(
                  fontSize: 15,
                ),
              ),
            ],
          ),
        ),
        RichText(
          text: TextSpan(
            style: DefaultTextStyle.of(context).style,
            children: const [
              TextSpan(
                text: '5. ',
                style: TextStyle(fontWeight: FontWeight.bold),
```

```
                ),
                TextSpan(
                  text: 'Regrese a la posición de inicio y pulse el botón rojo.\n',
                  style: TextStyle(
                    fontSize: 15,
                  ),
                ),
              ],
            ),
          ),
          RichText(
            text: TextSpan(
              style: DefaultTextStyle.of(context).style,
              children: const [
                TextSpan(
                  text: '6. ',
                  style: TextStyle(fontWeight: FontWeight.bold),
                ),
                TextSpan(
                  text:
                      'Vuelva al paso 3. Repite el ejercicio las veces que se
indica.\n',
                  style: TextStyle(
                    fontSize: 15,
                  ),
                ),
              ],
            ),
          ),
          RichText(
            text: TextSpan(
              style: DefaultTextStyle.of(context).style,
              children: const [
                TextSpan(
                  text: '7. ',
                  style: TextStyle(fontWeight: FontWeight.bold),
                ),
                TextSpan(
                  text:
                      'Al finalizar pulse salir.\n',
                  style: TextStyle(
                    fontSize: 15,
                  ),
                ),
              ],
            ),
          ),
          const Row(
```

```
                    crossAxisAlignment: CrossAxisAlignment.start,
                    children: [
                      Icon(
                        Icons.warning,
                        color: Colors.yellow,
                      ),
                      SizedBox(
                          width: 8), //espacio entre el icono y el texto
                      Expanded(
                        child:
                            Text('Si nota dolor no fuerce la flexión.'),
                      ),
                    ],
                  ),
                ],
              ),
              actions: [
                TextButton(
                  onPressed: () {
                    Navigator.pop(context);
                  },
                  child: const Text('Cerrar'),
                ),
              ],
            );
          },
        );
      },
      icon: const Icon(Icons.info, color: Colors.white, size: 30),
    ),
  ],
),
body: Container(
  decoration: const BoxDecoration(
    color: Colors.white,
    boxShadow: [
      BoxShadow(
        color: Colors.black12,
        blurRadius: 10,
        offset: Offset(0, 5),
      )
    ],
  ),
  child: Column(
    children: [
      Container(
        margin: const EdgeInsets.symmetric(horizontal: 8, vertical: 8),
        decoration: BoxDecoration(
```

```
    color: Colors.white,
    boxShadow: [
      BoxShadow(
        color: Colors.grey.withOpacity(0.5),
        spreadRadius: 3,
        blurRadius: 5,
        offset: const Offset(0, 3),
      ),
    ],
  ),
  child: Column(
    children: [
      Image.asset(
        'images/extensionMuneca.jpeg',
        height: 150,
      ),
      Container(
        padding: const EdgeInsets.all(10),
        alignment: Alignment.center,
        child: Column(
          children: [
            Container(
              padding: const EdgeInsets.symmetric(
                  horizontal: 16, vertical: 8),
              decoration: const BoxDecoration(
                color: Color.fromRGBO(255, 166, 2, 1),
              ),
              child: const Text(
                'Objetivo: 10 repeticiones',
                style: TextStyle(
                  fontSize: 20,
                  fontWeight: FontWeight.bold,
                  color: Colors.white,
                ),
              ),
            ),
            Text(
              'Repeticiones: $numRep',
              style: Theme.of(context).textTheme.headlineMedium,
            ),
            if (!_inicio)
              Text(
                'Ángulo: ${anguloMin.toStringAsFixed(2)}º',
                style: Theme.of(context).textTheme.headlineMedium,
              ),
            if (!_inicio && numRep < 10 && anguloMin > 85)
              Text(
                'Muy bien, sigue así!',
```

```
                textAlign: TextAlign.center,
                style: Theme.of(context).textTheme.headlineMedium,
              ),
            if (numRep == 10)
              Text(
                'Media Alcanzada: ${mediaAng.toStringAsFixed(2)}º',
                textAlign: TextAlign.center,
                style: Theme.of(context).textTheme.headlineMedium,
              ),
            if (numRep == 10)
              Text(
                'Enhorabuena, lograste el objetivo!',
                textAlign: TextAlign.center,
                style: Theme.of(context).textTheme.headlineMedium,
              )
          ],
        ),
      ),
    ],
  ),
),
const SizedBox(height: 30),
if (numRep < 10)
  if (!_inicio)
    Padding(
      padding: const EdgeInsets.all(5.0),
      child: Column(
        mainAxisAlignment: MainAxisAlignment.center,
        children: <Widget>[
          const SizedBox(height: 24),
          ElevatedButton(
              onPressed: _start,
              style: ElevatedButton.styleFrom(
                shape: const CircleBorder(),
                backgroundColor: Colors.green,
                padding: const EdgeInsets.all(24),
              ),
              child: const Icon(
                Icons.play_arrow,
                size: 48,
                color: Colors.white,
              )),
        ],
      ),
    ),
if (_inicio)
  Padding(
    padding: const EdgeInsets.all(5.0),
```

```
                child: Column(
                  mainAxisAlignment: MainAxisAlignment.center,
                  children: <Widget>[
                    const SizedBox(height: 24),
                    ElevatedButton(
                      onPressed: _finRep,
                      style: ElevatedButton.styleFrom(
                        shape: const CircleBorder(),
                        backgroundColor: Colors.red,
                        padding: const EdgeInsets.all(24),
                      ),
                      child: const Icon(
                        Icons.stop,
                        size: 48,
                        color: Colors.white,
                      ),
                    ),
                  ],
                ),
              ),
            if (numRep == 10)
              ElevatedButton(
                onPressed: _finEjercicio,
                style: ElevatedButton.styleFrom(
                  backgroundColor: const Color(0xFFFF8B1F),
                  shape: RoundedRectangleBorder(
                    borderRadius: BorderRadius.circular(10.0),
                  ),
                ),
                child: Container(
                  width: 200.0, //Ancho del boton
                  height: 50.0, //alto del botón
                  alignment: Alignment.center,
                  child: const Text(
                    'Salir',
                    style: TextStyle(
                      color: Colors.white,
                      fontSize: 18.0,
                    ),
                  ),
                ),
              ),
            ),
          ],
        ),
      ),
    );
  }
}
```

Ejercicio_flexion_muneca.dart:

```
import 'dart:async';
import 'package:flutter/material.dart';
import 'package:sensors_plus/sensors_plus.dart';
import 'dart:math';
import 'dart:io';
import 'package:path_provider/path_provider.dart';
import 'package:mailer/mailer.dart';
import 'package:mailer/smtp_server.dart';
import 'package:intl/intl.dart' show DateFormat;

class FlexionMunecaPage extends StatefulWidget {
  const FlexionMunecaPage({super.key});

  @override
  State<FlexionMunecaPage> createState() => _FlexionMunecaPageState();
}

class _FlexionMunecaPageState extends State<FlexionMunecaPage> {
 //Lista para almacenar las suscripciones a los distintos streams de eventos de los
sensores
  final List<StreamSubscription<dynamic>> _streamSubscriptions =
      <StreamSubscription<dynamic>>[];
  List<double> listaAngulos = [];
  double sumMaxAng =0.0; //Variable que almacena la suma de todos los angulos para calcular
la media
  double totalT =0.0; //Variable que almacena el tiempo total empleado en realizar el
ejercicio
  double tRepe = 0.0; //almacenar el tiempo de una repetición
  int numRep = 0;
  double maxAngle = 0.0; //Recoge el ángulo máximo de la repetición actual
  double mediaAng=0.0;
  final Stopwatch _stopwatch = Stopwatch();
  late File archivo;
  bool _inicio = false;
  bool fechaHoraRegistrada =false; //controla que se registre la fecha y hora de inicio del
ejercicio

  @override
  void initState() {
    super.initState();

    //LLama al metodo _resetFile(), para resetear el archivo al inicio del ejercicio
    _resetFile();
  }
```

```dart
//Función para resetear el archivo cada vez que se vaya a realizar el ejercicio
  void _resetFile() async {
    try {
      //Obtiene el directorio de documentos de la aplicación
      Directory documentsDirectory = await getApplicationDocumentsDirectory();

      //Crea una referencia a la ruta especifica del archivo
      archivo = File('${documentsDirectory.path}/flexionMuñeca.txt');

      //Verifica la existencia del archivo
      if (await archivo.exists()) {
        //Si hay un archivo existente borra su contenido
        archivo.writeAsStringSync('');
      } else {
        //Crea un nuevo archivo en la ruta especificada antes
        archivo = await archivo.create(recursive: true);
      }
    } catch (e) {
      //Si ocurre un error lo muestra en la consola (Util durante el desarrollo de la app)
      // ignore: avoid_print
      print('Error al resetear el archivo: $e');
    }
  }

//Función para comenzar a recibir información del acelerómetro
  void _start() {
    //Listas que almacenan los valores absolutos devueltos por el acelerometro
    late List<double> accelerometerValues = [0, 0, 0];
    double angulo = 0.0;
    //se inicia la lectura del acelerómetro
    _streamSubscriptions.add(
      accelerometerEvents.listen((AccelerometerEvent event) {
        //Actualiza los valores del acelerómetro
        setState(() {
          accelerometerValues = [event.x, event.y, event.z];
          angulo = _calcularAngulo(accelerometerValues);
          listaAngulos.add(angulo);
          _stopwatch.start();
        });
      }),
    );
    setState(() {
      _inicio = true;
    });
  }

  void _finRep() {
```

```
    if (listaAngulos.isNotEmpty) {
      maxAngle = listaAngulos.reduce((max, value) => value > max ? value : max);
      sumMaxAng +=
          maxAngle; //Calculamos la suma de los ángulos alcanzados en cada repeticion
      listaAngulos.clear();
      setState(() {
        _inicio = false;
        numRep++;
        _stopwatch.stop(); //Para el reloj
        int ms =
            _stopwatch.elapsedMilliseconds; //Calcula el tiempo en milisegundos
        tRepe = ms / 1000; //Pasa a segundos el tiempo empleado en la repeticion
        totalT += tRepe;
      });
      _stopwatch.reset();
      _writeDataToFile(numRep, tRepe, maxAngle);
    }
  }

//Función que calcula el ángulo de inclinación a través de los datos del acelerómetro
  double _calcularAngulo(List<double> a) {
    //calculamos el ángulo de rotación en el eje Y en grados
    double accelAngY =
        (atan(a[1] / sqrt(pow(a[0], 2) + pow(a[2], 2))) * (180 / pi)) +
            5.0; //sumamos un ofset de 5º
    // ignore: avoid_print
    print("Angulo actual: ${accelAngY.toStringAsFixed(2)}º");

    return accelAngY;
  }


  void _writeDataToFile(int numRep, double tiempo, double anguloRep) async {
    try {
      Directory documentsDirectory = await getApplicationDocumentsDirectory();
      archivo = File('${documentsDirectory.path}/flexionMuñeca.txt');
      await archivo.create(recursive: true);
      if (!fechaHoraRegistrada) {
        // Obtener la fecha y hora actual
        DateTime now = DateTime.now();
        String fechaHora = DateFormat('yyyy-MM-dd HH:mm:ss').format(now);
        const titulo = 'EJERCICIO: FLEXIÓN DE MUÑECA\n';
        await archivo.writeAsString(titulo, mode: FileMode.append);
        final inicioContent = 'Fecha y Hora de inicio: $fechaHora\n';
        await archivo.writeAsString(inicioContent, mode: FileMode.append);

        fechaHoraRegistrada =
            true; // Marcar que se ha registrado la fecha y hora
      }
```

```
      final content =
          'Repetición: $numRep, Tiempo: $tiempo segundos, Ángulo:
${anguloRep.toStringAsPrecision(2)}º\n';
      await archivo.writeAsString(content, mode: FileMode.append);

       //Verificar que se han realizado todas las repeticiones
      if(numRep==10){
      //Calcula la media del ángulo alcanzado en el ejercicio
      mediaAng = sumMaxAng/10;

      //Añade una linea al archivo con la media de los ángulos
      final mediaContent = '\n\nMedia de ángulos: ${mediaAng.toStringAsPrecision(2)}º\n';
      await archivo.writeAsString(mediaContent, mode: FileMode.append);

      //Añade una linea al archivo con el tiempo total empleado en el ejercicio
      final tiempoTotalContent = '\nTiempo total: $totalT segundos\n';
      await archivo.writeAsString(tiempoTotalContent, mode: FileMode.append);
      }

      // Obtener la ubicación del archivo
      String filePath = archivo.path;
      // ignore: avoid_print
      print('Archivo creado en la ubicación: $filePath');

      // ignore: avoid_print
      print('Archivo creado y contenido escrito con exito.');
    } catch (e) {
      // ignore: avoid_print
      print('Error al crear y escribir en el archivo: $e');
    }
  }

  void sendEmailWithAttachment() async {
    //Configuración del servidor SMTP
    final smtpServer = hotmail('correoPaciente@hotmail.com', 'password');

    final message = Message()
      ..from =
          const Address('correoPaciente@hotmail.com', 'Nombre Apellido1 Apellido2')
      ..recipients.add('correoMédico@gmail.com')
      ..subject = 'Evolución de la paciente Ana Santos Delgado'
      ..text = 'Adjunto de archivo con información del paciente'
      ..attachments.add(FileAttachment(archivo));

    try {
      //Enviar el correo electrónico a través del servidor SMTP

      final sendReport = await send(message, smtpServer);
```

```
    // ignore: avoid_print
    print('Correo enviado: ${sendReport.messageSendingEnd}');
  } catch (e) {
    // ignore: avoid_print
    print('Error al enviar el correo: $e');
  }
}

@override
void dispose() {
  super.dispose();
  for (StreamSubscription<dynamic> subscription in _streamSubscriptions) {
    subscription.cancel();
  }
}

//boton finalizar para salir de la pagina del ejercicio
void _finEjercicio() {
  //Regresa a la pagina anterior
  Navigator.pop(context);
  //Llama a la función para enviar el correo al doctor
  sendEmailWithAttachment();
}

@override
Widget build(BuildContext context) {
  return Scaffold(
    appBar: AppBar(
       backgroundColor: const Color(0xFFFF8B1F),
      title: const Text(
        'Flexión de Muñeca',
        style: TextStyle(
          fontWeight: FontWeight.w900,
          fontFamily: 'Montserrat',
          fontSize: 16.5,
          color: Colors.white,
        ),
      ),
      leading: IconButton(
          icon: const Icon(Icons.arrow_back, color: Colors.white, size: 30),
          onPressed: () {
            Navigator.pop(context);
          }),
      actions: [
        IconButton(
          onPressed: () {
            showDialog(
              context: context,
```

```
                builder: (BuildContext context) {
                  return AlertDialog(
                    title: const Text(
                      'PASOS A SEGUIR PARA REALIZAR EL EJERCICIO',
                      style: TextStyle(
                          fontWeight: FontWeight.w900,
                          fontFamily: 'Montserrat',
                          fontSize: 19,
                          color: Colors.black),
                    ),
                    content: Column(
                      crossAxisAlignment: CrossAxisAlignment.start,
                      children: [
                        RichText(
                          text: TextSpan(
                            style: DefaultTextStyle.of(context).style,
                            children: const [
                              TextSpan(
                                text: '1. ',
                                style: TextStyle(fontWeight: FontWeight.bold),
                              ),
                              TextSpan(
                                text:
                                    'Situa el antebrazo sobre una superficie plana,
quedando la mano en el aire con la palma hacia arriba.\n',
                                style: TextStyle(
                                  fontSize: 15,
                                ),
                              ),
                            ],
                          ),
                        ),
                        RichText(
                          text: TextSpan(
                            style: DefaultTextStyle.of(context).style,
                            children: const [
                              TextSpan(
                                text: '2. ',
                                style: TextStyle(fontWeight: FontWeight.bold),
                              ),
                              TextSpan(
                                text: 'Sostenga el móvil en la mano en posición
vertical.\n',
                                style: TextStyle(
                                  fontSize: 15,
                                ),
                              ),
                            ],
```

```
                ),
              ),
              RichText(
                text: TextSpan(
                  style: DefaultTextStyle.of(context).style,
                  children: const [
                    TextSpan(
                      text: '3. ',
                      style: TextStyle(fontWeight: FontWeight.bold),
                    ),
                    TextSpan(
                      text: 'Pulse el botón verde para iniciar la repetición.\n',
                      style: TextStyle(
                        fontSize: 15,
                      ),
                    ),
                  ],
                ),
              ),
              RichText(
                text: TextSpan(
                  style: DefaultTextStyle.of(context).style,
                  children: const [
                    TextSpan(
                      text: '4. ',
                      style: TextStyle(fontWeight: FontWeight.bold),
                    ),
                    TextSpan(
                      text: 'Mueva la mano hacia arriba, sin superar el umbral de
dolor.\n',
                      style: TextStyle(
                        fontSize: 15,
                      ),
                    ),
                  ],
                ),
              ),
              RichText(
                text: TextSpan(
                  style: DefaultTextStyle.of(context).style,
                  children: const [
                    TextSpan(
                      text: '5. ',
                      style: TextStyle(fontWeight: FontWeight.bold),
                    ),
                    TextSpan(
                      text:
```

```
                                    'Vuelva a la posición inicial y pulse el boton
rojo.\n',
                                style: TextStyle(
                                  fontSize: 15,
                                ),
                              ),
                            ],
                          ),
                        ),
                        RichText(
                          text: TextSpan(
                            style: DefaultTextStyle.of(context).style,
                            children: const [
                              TextSpan(
                                text: '6. ',
                                style: TextStyle(fontWeight: FontWeight.bold),
                              ),
                              TextSpan(
                                text: 'Vuelva al paso 3 y repita el ejercicio tantas veces
como se indica.\n',
                                style: TextStyle(
                                  fontSize: 15,
                                ),
                              ),
                            ],
                          ),
                        ),
                        RichText(
                          text: TextSpan(
                            style: DefaultTextStyle.of(context).style,
                            children: const [
                              TextSpan(
                                text: '7. ',
                                style: TextStyle(fontWeight: FontWeight.bold),
                              ),
                              TextSpan(
                                text:
                                    'Al finalizar pulse salir.\n',
                                style: TextStyle(
                                  fontSize: 15,
                                ),
                              ),
                            ],
                          ),
                        ),
                        const Row(
                          crossAxisAlignment: CrossAxisAlignment.start,
                          children: [
```

```
                    Icon(
                      Icons.warning,
                      color: Colors.yellow,
                    ),
                    SizedBox(
                        width: 8), //espacio entre el icono y el texto
                    Expanded(
                      child:
                          Text('Si nota dolor no fuerce la flexión.'),
                    ),
                  ],
                ),
              ],
            ),
            actions: [
              TextButton(
                onPressed: () {
                  Navigator.pop(context);
                },
                child: const Text('Cerrar'),
              ),
            ],
          );
        },
      );
    },
    icon: const Icon(Icons.info, color: Colors.white, size: 30),
  ),
 ],
),
body: Container(
  decoration: const BoxDecoration(
    color: Colors.white,
    boxShadow: [
      BoxShadow(
        color: Colors.black12,
        blurRadius: 10,
        offset: Offset(0, 5),
      )
    ],
  ),
  child: Column(
    children: [
      Container(
        margin: const EdgeInsets.symmetric(horizontal: 8, vertical: 8),
        decoration: BoxDecoration(
          color: Colors.white,
          boxShadow: [
```

```
        BoxShadow(
          color: Colors.grey.withOpacity(0.5),
          spreadRadius: 3,
          blurRadius: 5,
          offset: const Offset(0, 3),
        ),
      ],
    ),
    child: Column(
      children: [
        Image.asset(
          'images/flexionMuneca.jpeg',
          height: 150,
        ),
        Container(
          padding: const EdgeInsets.all(10),
          alignment: Alignment.center,
          child: Column(
            children: [
                Container(
                  padding: const EdgeInsets.symmetric(
                      horizontal: 16, vertical: 8),
                  decoration: const BoxDecoration(
                    color:  Color.fromRGBO(255, 166, 2, 1),
                  ),
                  child: const Text(
                    'Objetivo: 10 repeticiones',
                    style: TextStyle(
                      fontSize: 20,
                      fontWeight: FontWeight.bold,
                      color: Colors.white,
                    ),
                  ),
                ),
              Text(
                 'Repeticiones: $numRep',
                 style: Theme.of(context).textTheme.headlineMedium,
               ),
               if (!_inicio)
                 Text(
                   'Ángulo: ${maxAngle.toStringAsFixed(2)}º',
                   style: Theme.of(context).textTheme.headlineMedium,
                 ),
               if (!_inicio && numRep<10 && maxAngle>85)
               Text(
                   'Muy bien, sigue así!',
                   textAlign: TextAlign.center,
                   style: Theme.of(context).textTheme.headlineMedium,
```

```
                ),
              if (numRep == 10)
              Text(
                  'Media Alcanzada: ${mediaAng.toStringAsFixed(2)}º',
                  textAlign: TextAlign.center,
                  style: Theme.of(context).textTheme.headlineMedium,
                ),
                if (numRep == 10)
                Text(
                  'Enhorabuena, lograste el objetivo!',
                  textAlign: TextAlign.center,
                  style: Theme.of(context).textTheme.headlineMedium,
                )
            ],
          ),
        ),
      ],
    ),
  ),
),
const SizedBox(height: 30),
if (numRep < 10)
  if (!_inicio)
    Padding(
      padding: const EdgeInsets.all(5.0),
      child: Column(
        mainAxisAlignment: MainAxisAlignment.center,
        children: <Widget>[
          const SizedBox(height: 24),
          ElevatedButton(
              onPressed: _start,
              style: ElevatedButton.styleFrom(
                shape: const CircleBorder(),
                backgroundColor: Colors.green,
                padding: const EdgeInsets.all(24),
              ),
              child: const Icon(
                Icons.play_arrow,
                size: 48,
                color: Colors.white,
              )),
        ],
      ),
    ),
if (_inicio)
  Padding(
    padding: const EdgeInsets.all(5.0),
    child: Column(
      mainAxisAlignment: MainAxisAlignment.center,
```

```
                children: <Widget>[
                  const SizedBox(height: 24),
                  ElevatedButton(
                    onPressed: _finRep,
                    style: ElevatedButton.styleFrom(
                      shape: const CircleBorder(),
                      backgroundColor: Colors.red,
                      padding: const EdgeInsets.all(24),
                    ),
                    child: const Icon(
                      Icons.stop,
                      size: 48,
                      color: Colors.white,
                    ),
                  ),
                ],
              ),
            ),
          if (numRep == 10)
            ElevatedButton(
              onPressed: _finEjercicio,
              style: ElevatedButton.styleFrom(
                backgroundColor: const Color(0xFFFF8B1F),
                shape: RoundedRectangleBorder(
                  borderRadius: BorderRadius.circular(10.0),
                ),
              ),
              child: Container(
                width: 200.0, //Ancho del boton
                height: 50.0, //alto del botón
                alignment: Alignment.center,
                child: const Text(
                  'Salir',
                  style: TextStyle(
                    color: Colors.white,
                    fontSize: 18.0,
                  ),
                ),
              ),
            ),
        ],
      ),
    ),
  );
 }
}
```